Ministério das Finanças
e do Fomento Empresarial

Unidade de Coordenação do
Sistema Nacional de Qualificações

SISTEMA NACIONAL DE QUALIFICAÇÕES

Qualificação Profissional

Programação

TIC001.A1_5

Família Profissional Tecnologias de Informação e Comunicação

Ficha Técnica

Ministério das Finanças e do Fomento Empresarial
Unidade de Coordenação do Sistema Nacional de Qualificações (UC-SNQ)

Coordenador da UC-SNQ
Amílcar Alexandre Mendes

Família Profissional: Tecnologias de Informação e Comunicação - TIC
Coordenador do Comité Técnico Setorial (CTS)
Marcelo Francisco de Barros Correia

Especialistas tecnológicos e formativos do CTS
Anilton Brandão

Andricio Cabral

Emanuel Correia

Gaudêncio Fernandes

Técnicos da UC-SNQ
Marlene Moreno

República de Cabo Verde | Cidade da Praia | Achada Santo António
Novembro 2023

Apoio ao Utilizador
Telefone: +238 333 70 55/61 **| E-Mail:** snqcv.uc@gmail.com **| Website:** www.snq.cv **| Facebook:** ucsnq

REVOGADO

Índice

PERFIL PROFISSIONAL 3

UNIDADES DE COMPETÊNCIA (UC) 4

UC1: DESENVOLVER APLICAÇÕES *FRONT-END* (UC290_5) 4

UC2: DESENVOLVER APLICAÇÕES *BACK-END* (UC291_5) 8

UC3: DESENVOLVER APLICAÇÕES PARA *DESKTOP* (UC292_5) 11

UC4: DESENVOLVER APLICAÇÕES PARA DISPOSITIVOS MÓVEIS (UC293_5) 14

UC5: DESENVOLVER APLICAÇÕES PARA INTERNET DAS COISAS - *IoT* (UC294_5) 16

UC6: DESENVOLVER APLICAÇÕES PARA JOGOS (UC295_5) 19

PROGRAMA FORMATIVO DA QUALIFICAÇÃO PROFISSIONAL 22

UNIDADES FORMATIVAS TRANSVERSAIS 24

MÓDULOS FORMATIVOS (MF) 43

MF1: PROGRAMAÇÃO DE APLICAÇÕES *FRONT-END* (MF290_5) 43

MF2: PROGRAMAÇÃO DE APLICAÇÕES *BACK-END* (MF291_5) 52

MF3: PROGRAMAÇÃO DE APLICAÇÕES PARA DESKTOP (MF292_5) 61

MF4: PROGRAMAÇÃO DE APLICAÇÕES PARA DISPOSITIVOS MÓVEIS (MF293_5) 70

MF5: PROGRAMAÇÃO DE APLICAÇÕES IoT (MF293_5) 81

MF6: PROGRAMAÇÃO DE APLICAÇÕES DE JOGOS 96

MÓDULO FORMATIVO EM CONTEXTO REAL DE TRABALHO 114

REVOGADO

PERFIL PROFISSIONAL

TIC001.A1_5

PROGRAMAÇÃO

REVOGADO

<table>
<tr><th colspan="4">PERFIL PROFISSIONAL</th></tr>
<tr><td>Código: TIC001.A1_5</td><td colspan="3">Denominação: PROGRAMAÇÃO</td></tr>
<tr><td>Nível: 5</td><td colspan="3">Família Profissional: Tecnologias de Informação e Comunicação (TIC)</td></tr>
<tr><td>COMPETÊNCIA GERAL</td><td colspan="3">Desenvolver, instalar, testar e realizar a manutenção de aplicações informáticas nos diferentes contextos, independentemente do modelo e da plataforma de desenvolvimento utilizado, de acordo com os requisitos e especificações definidas, respeitando a segurança e a qualidade da aplicação.</td></tr>
<tr><td rowspan="7">UNIDADES DE COMPETÊNCIA</td><td>N.º</td><td>Denominação</td><td>Código</td></tr>
<tr><td>1</td><td>Desenvolver Aplicações Front-end</td><td>UC283_5</td></tr>
<tr><td>2</td><td>Desenvolver Aplicações Back-end</td><td>UC284_5</td></tr>
<tr><td>3</td><td>Desenvolver Aplicações para Desktop</td><td>UC285_5</td></tr>
<tr><td>4</td><td>Desenvolver Aplicações para Dispositivos Móveis</td><td>UC286_5</td></tr>
<tr><td>5</td><td>Desenvolver aplicações para Internet das Coisas (IoT)</td><td>UC287_5</td></tr>
<tr><td>6</td><td>Desenvolver Aplicações para Jogos</td><td>UC288_5</td></tr>
<tr><td rowspan="3">AMBIENTE PROFISSIONAL</td><td colspan="3">Âmbito profissional:
Desenvolve sua atividade profissional por conta própria ou por conta de outrem em pequenas, médias e grandes empresas dedicadas ao desenvolvimento de aplicações informáticas (software house) e de telecomunicações ou outras entidades, tais como agências de publicidade e propaganda, estúdios de animação, provedores de Internet, meios de comunicação digital, instituições de educação, lojas virtuais, empresas de desenvolvimento de jogos, automação residencial, indústria agrícola e de automação industrial.</td></tr>
<tr><td colspan="3">Sectores produtivos:
Esta qualificação enquadra-se no contexto da transformação digital, no setor de desenvolvimento de aplicações informáticas, nos setores produtivos, nos sistemas de informação como suporte do negócio e nos departamentos de tecnologias de informação ou por meio de outsourcing.</td></tr>
<tr><td colspan="3">Ocupações e postos de trabalho relacionados - CNP 2010:
• 2512 Programador de aplicações informáticas (Desktop).
• 2513 Programador web e multimédia (Front-end e Back-end).
• 2514 Programador de Aplicações (IoT, Mobile e Jogos).
• 2519 Outros analistas e programadores de aplicações informáticas e aplicações</td></tr>
</table>

REVOGADO

UNIDADES DE COMPETÊNCIA (UC)

UC1: DESENVOLVER APLICAÇÕES *FRONT-END*	
Código: UC283_5	**Nível: 5**

Elementos de Competência (EC) e Critérios de Desempenho (CD)

EC1: Realizar o levantamento e a análise dos requisitos do sistema a ser desenvolvido.

CD 1.1. O problema a ser solucionado é identificado delimitando as fronteiras de acordo com o micromundo.

CD 1.2. Os requisitos funcionais são levantados de acordo com as necessidades do cliente e utilizando as técnicas adequadas.

CD 1.3. Os requisitos funcionais levantados são analisados e hierarquizados de acordo com as prioridades do cliente.

CD 1.4. O modelo do sistema a implementar é definido de acordo com os requisitos levantados.

CD 1.5. A documentação técnica do modelo do sistema é produzida de acordo com o ciclo de vida do projeto (modelagem de sistema, UML).

EC2: Realizar o planeamento do sistema a desenvolver.

CD 2.1. A metodologia do trabalho é definida de acordo com os requisitos.

CD 2.2. O cronograma de execução do trabalho é elaborado de acordo com a metodologia e requisitos funcionais e não funcionais.

CD 2.3. Os equipamentos para execução (*hardware* e *software*) são inventariados de cordo com os requisitos.

CD 2.4. O orçamento é definido e apresentado ao cliente de acordo com os requisitos funcionais e não funcionais.

EC3: Realizar a instalação e a configuração do ambiente de desenvolvimento.

CD 3.1. Os equipamentos (*hardware* e *software*) são selecionados de acordo com o plano definido.

CD 3.2. Os equipamentos são instalados e configurados de acordo com plano de desenvolvimento.

CD 3.3. Os parâmetros do sistema são configurados de acordo com a tecnologia selecionada.

EC4: Desenvolver uma aplicação, utilizando uma linguagem de programação de iniciação.

CD 4.1. A documentação técnica é interpretada para implementação da aplicação.

CD 4.2. O código fonte da aplicação é escrito de acordo com a documentação técnica, utilizando uma linguagem de programação predefinida no plano.

CD 4.3. Os componentes e as bibliotecas desenvolvidos por terceiros ou fabricantes são reutilizados para incluir funcionalidades específicas no projeto, de acordo com as normas vigentes.

CD 4.4. O relatório de *debug* é analisado para possíveis intervenções corretivas.

CD 4.5. O teste da aplicação é realizado para a validação dos requisitos funcionais.

CD 4.6. A documentação técnica do componente é redigida conforme os modelos, normas e procedimentos estabelecidos no protocolo do projeto.

CD 4.7. O código desenvolvido e validado é armazenado no repositório do projeto como forma de controlar as versões da aplicação em desenvolvimento.

EC5: Implementar interface ergonómica (UX/UI).

CD 5.1. As funcionalidades da aplicação são identificadas no protótipo, em função do produto a ser desenvolvido.

CD 5.2. A área a ser trabalhada é delimitada de acordo com o dispositivo.

EVOGADO

CD 5.3. A janela do dispositivo é esboçada para facilitar a comunicação com o utilizador e permitir a comparação com a realidade.

CD 5.4. As funcionalidades da aplicação a ser desenvolvida são identificadas no protótipo, em função do produto a ser desenvolvido.

CD 5.5. O modelo de dados é identificado, para representar fielmente o negócio.

CD 5.6. O modelo conceptual é desenhado para a identificação das entidades do sistema.

CD 5.7. Os tipos de conteúdos e o local de posicionamento no protótipo são identificados de acordo com o objetivo da aplicação.

CD 5.8. O protótipo da aplicação é desenhado de acordo com os requisitos funcionais do sistema.

CD 5.9. O *template* é desenhado e ajustado ao *layout* do protótipo pré-estabelecido.

CD 5.10. O *template* desenhado é fatiado (*Slice*) e montado, utilizando as ferramentas de programação *front-end* (HTML, CSS, *Bootstrap*, JS, JQ).

CD 5.11. O protótipo desenhado é testado e validado junto ao utilizador final para garantir a qualidade do produto.

CD 5.12. O relatório de teste *Output* é analisado para possíveis reajustes.

CD 5.13. O protótipo testado e validado é numerado e catalogado para a identificação das ações associadas a cada janela.

EC6: Realizar integração de sistemas *web*.

CD 6.1. Os dados para consumo ou disponibilização são identificados de acordo com a integração a ser realizada.

CD 6.2. A troca de dados é realizada, utilizando as tecnologias de integração.

CD 6.3. A formatação dos dados é realizada, utilizando a linguagem de marcação (XML, Json) de acordo com modelo pré-definido.

EC7: Realizar *backup* e restaurar o sistema.

CD 7.1. Os ficheiros e os dados a serem guardados são identificados para efeitos de *backup*.

CD 7.2. O tipo de *backup* a ser realizado é definido de acordo com as normas pré-estabelecidas.

CD 7.3. A agenda e a forma como é realizada *backup* são definidas de forma manual ou automático, de acordo com a política de segurança pré-estabelecida.

CD 7.4. Os Mídias para *backup* ou serviços de *backup* em nuvem são selecionados para realizar a cópia de segurança do sistema.

CD 7.5. A cópia de segurança de ficheiros e dados é realizada de acordo com as normas e políticas de segurança pré-estabelecidas.

CD 7.6. A cópia de segurança realizada é testada, para garantir que os dados selecionados foram efetivamente copiados.

CD 7.7. Os Mídias são catalogados e armazenados de acordo com as normas de segurança pré-estabelecidas.

CD 7.8. Os dados do *backup* a serem restaurados são identificados e selecionados.

CD 7.9. Os dados são restaurados para o pleno funcionamento do sistema.

EC8: Realizar a manutenção do sistema.

CD 8.1. A documentação do código existente, a arquitetura, a estrutura dos dados e as interfaces são analisadas para definir o tipo de manutenção a realizar.

CD 8.2. Os constrangimentos que podem ocorrer durante o processo da realização de manutenção são equacionados ou previstos para evitar alguns problemas, tais como:

- Ausência ou deficiência na documentação;
- Dificuldade na identificação de manutenções realizadas anteriormente;
- Falta de controlo de versão;

- Baixa manutenibilidade do *software*.

CD 8.3. O tipo de manutenção a realizar é definida para a elaboração do plano de intervenção.

CD 8.4. As modificações necessárias são identificadas e avaliadas os seus impactos.

CD 8.5. O plano de manutenção é elaborado para garantir a gestão do sistema.

CD 8.6. As modificações são realizadas tendo em conta o plano pré-definido e as mudanças ocorridas no ambiente em que está inserido o sistema.

CD 8.7. As partes afetadas pelas modificações realizadas são documentadas de forma a evitar possíveis efeitos colaterais, tais como: Modificação da estrutura do código; Inclusão de códigos com erros; Desatualização da documentação, etc.

CD 8.8. O teste da aplicação é realizado para avaliar as funcionalidades das partes modificadas e/ou acrescentadas.

CD 8.9. Os parâmetros de otimização do *website* são ajustados, utilizando as ferramentas adequadas.

CD 8.10. O sistema otimizado é testado para verificar a performance desejada.

EC9: Realizar a hospedagem e publicação do sistema.

CD 9.1. O provedor de alojamento *web* é selecionado de acordo com critérios de hospedagem e publicação do *site*.

CD 9.2. As tecnologias para a hospedagem do *site* são selecionadas de acordo com o *site* desenvolvido.

CD 9.3. Os ficheiros da aplicação a serem hospedados são organizados de acordo com a estrutura do provedor.

CD 9.4. O *site* é hospedado e publicado utilizando as ferramentas de publicação adequadas.

EC10: Programar aplicação no ambiente *front-end,* utilizando ferramentas de desenvolvimento adequadas para obter a funcionalidade especificada no projeto.

CD 10.1. Os elementos necessários para codificar os componentes são identificados.

CD 10.2. A lógica da aplicação é implementada corretamente e o componente é codificado, utilizando ferramentas de desenvolvimento adequadas.

CD 10.3. O código do componente *software* é desenvolvido para produzir uma interface com as condições de usabilidade, acessibilidade e ergonomia descritas, em conformidade com os padrões de qualidade do produto.

CD 10.4. Os componentes são testados utilizando as baterias de teste previstas no projeto para verificar sua funcionalidade e corrigir os erros detetados.

CD 10.5. A documentação do componente *software* é produzida conforme o modelo, norma e procedimentos estabelecidos nas especificações do projeto.

CD 10.6. O código desenvolvido e validado é armazenado no repositório de *software* do projeto.

EC11: Reutilizar os componentes do *software* de interface de utilizador desenvolvidos por fabricantes para incluir funcionalidades específicas no projeto.

CD 11.1. O componente do *software* é reutilizado e configurado através de sua interface para adaptar o seu comportamento às funcionalidades requeridas e garantir a integridade do sistema.

CD 11.2. Os componentes são testados utilizando as baterias de teste previstas no projeto para verificar a sua funcionalidade.

CD 11.3. A documentação do produto *software* reutilizado é interpretada e utilizada, conforme os formatos, normas e procedimentos estabelecidos nas especificações do projeto e a documentação fornecida pelo fabricante.

CD 11.4. O código desenvolvido e validado é armazenado no repositório de *software* do projeto.

EC12: Reutilizar os componentes multimédia existentes para complementar a funcionalidade da interface de utilizador.

CD 12.1. Os componentes de áudio e vídeo especificados no desenho do projeto são incorporados à interface de utilizador e adaptados aos formatos *standard*.

REVOGADO

CD 12.2. Os elementos gráficos necessários são incorporados à interface de utilizador e adaptados aos formatos *standard*, utilizando ferramentas especificas, conforme as especificações recebidas.

CD 12.3. As animações multimédia requeridas são incorporadas à interface de utilizador, utilizando linguagem e ferramentas especificas.

CD 12.4. Os componentes multimédia desenvolvidos são ajustados às normas de acessibilidade, usabilidade e ergonomia.

CD 12.5. Os componentes são testados, utilizando as baterias de teste previstas no projeto para verificar a sua funcionalidade.

CD 12.6. O código desenvolvido e validado é armazenado no repositório do *software* do projeto.

EC13: Realizar conexão de acesso a bases de dados ou a outras estruturas de armazenamento de informação para integrar os conteúdos na lógica da aplicação.

CD 13.1. Os componentes são codificados, incluindo as funcionalidades de conexão com bases de dados ou outras estruturas de armazenamento de informação, de acordo especificações pré-estabelecidas.

CD 13.2. A conexão da aplicação com os Sistemas de Gestão de Bases de Dados (SGBD) é realizada para garantir o acesso aos dados da base de dados de acordo as especificações definidas.

CD 13.3. As aplicações de acesso a armazenamentos de dados são desenvolvidas, garantindo a segurança, consistência e integridade da informação.

CD 13.4. Os componentes desenvolvidos são testados para verificar as suas funcionalidades e a sua integração com outros componentes do projeto.

Contexto profissional

Meios de produção

Equipamentos informáticos e periféricos. Sistemas informáticos. Ferramentas informáticas de escritório. Ferramentas de desenvolvimento de aplicações informáticas. Ferramentas CASE, Ferramentas de desenvolvimento rápido. Linhas de comunicações. Sistemas de segurança. Ferramentas de controlo de versões (CVS). Sistema de Gestão de Base de Dados (SGB). Ferramentas de desenvolvimento *front-end* (CSS, HTML, entre outras), equipamentos de *Backup*. Ligação à internet. *Cloud*.

Produtos e resultados

Especificações técnicas interpretadas. Diagrama Entidade Relacionamento (DER) desenhado. Diagrama de Fluxo de Dados (DFD) desenhado. Modelos de dados desenhados. Diagramas e processos desenhados. Diagramas de UML Produzidos. Protótipo desenvolvido. *Template* desenhado, fatiado e montado. *Site* desenhado, desenvolvido, hospedado e publicado.

Informação utilizada ou gerada

Projeto aplicações informáticas em desenvolvimento. Manuais das ferramentas de desenvolvimento. Suportes técnicos de assistência. DER, DFD, Diagramas UML. Metodologia de desenvolvimento de aplicações informáticas. Informações sobre centro de processo de dados, *hardware* e aplicações informáticas.

REVOGADO

UC2: DESENVOLVER APLICAÇÕES *BACK-END*	
Código: UC284_5	**Nível: 5**

Elementos de Competência (EC) e Critérios de Desempenho (CD)

EC1: Realizar levantamento e análise dos requisitos do sistema a ser desenvolvido.

CD 1.1. O problema a ser solucionado é identificado e o escopo é limitado de acordo com a sua natureza e a necessidade do cliente.

CD 1.2. As necessidades do cliente são levantadas, utilizando técnicas adequadas.

CD 1.3. Os requisitos funcionais são especificados de acordo com as necessidades do cliente.

CD 1.4. Os requisitos funcionais especificados são analisados e hierarquizados de acordo com o grau de prioridade.

CD 1.5. A especificação dos requisitos é realizada para definir a funcionalidade do sistema a ser desenvolvido.

CD 1.6. Os requisitos funcionais são especificados em documentos de especificação técnica, respeitando as normas vigentes.

CD 1.7. Os requisitos funcionais especificados são revisados e validados em concertação com o cliente.

EC2: Realizar o planeamento de aplicativo *Back-end* a ser desenvolvido.

CD 2.1. A aplicação é esboçada para organizar as ideias e os conteúdos.

CD 2.2. A metodologia do trabalho é definida de acordo com os requisitos.

CD 2.3. O tipo da aplicação a ser desenvolvida é selecionado de acordo com o público alvo.

CD 2.4. O mercado é analisado para introduzir diferencial na aplicação a ser desenvolvida.

CD 2.5. O plano e o cronograma de execução do trabalho são elaborados de acordo com a metodologia e os requisitos funcionais e não funcionais.

CD 2.6. Os equipamentos necessários para execução dos trabalhos (*hardware* e *software*) são identificados a partir dos requisitos.

CD 2.7. O orçamento é definido e apresentado ao cliente de acordo com os requisitos funcionais e não funcionais.

EC3: Programar os módulos do *software,* no ambiente do servidor, de acordo com as especificações técnicas definidas no desenho do sistema.

CD 3.1. A linguagem de programação e o ambiente de desenvolvimento são definidos de acordo com as normas do projeto.

CD 3.2. O código fonte do programa é desenvolvido com recurso ao editor de texto - IDE assegurando a sua funcionalidade.

CD 3.3. O código fonte editado é compilado ou interpretado para depurar os erros, assegurando a plena funcionalidade.

CD 3.4. Os erros depurados são corrigidos e o código fonte é novamente compilado, para garantir a eliminação dos erros.

CD 3.5. O código objeto gerado é ligado (*Linker*) às bibliotecas necessárias ou a URL (*Uniform Resource Identifier* ou Identificador Uniforme de Recursos), para garantir a plena funcionalidade do sistema.

CD 3.6. O teste da aplicação desenvolvida é realizado, utilizando casos de testes preparados e as ferramentas necessárias.

CD 3.7. O ficheiro executável é gerado para a sua distribuição e instalação.

REVOGADO

EC4: Interpretar as estruturas de dados e o desenho da base de dados para realizar corretamente as tarefas de programação.

CD 4.1. As entidades do modelo de dados são identificadas no Diagrama Entidade Relacionamento (DER) para o desenho da base de dados.

CD 4.2. As restrições de integridade e semânticas da base de dados são identificadas no DER.

CD 4.3. As características do modelo físico, relevantes para a realização das tarefas de programação, são identificadas nos diagramas.

EC5: Manipular conteúdo de bases de dados de forma interativa.

CD 5.1. Os elementos relevantes para execução da operação são identificados na base de dados.

CD 5.2. A estrutura da base de dados e os seus componentes são consultados, utilizando as ferramentas de acesso.

CD 5.3. As operações de manipulação de dados são programadas conforme as especificações recebidas, utilizando ferramentas gráficas ou linguagem de manipulação de dados adequados.

CD 5.4. As operações de manipulação da base de dados são testadas para garantir a integridade dos dados.

EC6: Programar módulos de manipulação da base de dados.

CD 6.1. Os elementos da base de dados relevantes para o módulo a desenvolver são identificados.

CD 6.2. As operações sobre a base de dados são codificadas, utilizando a linguagem e os requisitos especificados na documentação do projeto.

CD 6.3. As transações são geridas para garantir a consistência dos dados na base de dados.

CD 6.4. As operações sobre a base de dados são otimizadas para aperfeiçoar o seu desempenho.

CD 6.5. As operações de manipulação da base de dados são testadas para garantir a integridade dos dados.

CD 6.6. Os módulos desenvolvidos são documentados conforme as normas do projeto.

EC7: Integrar sistemas distribuídos ou heterogéneos, utilizando as tecnologias de desenvolvimento para troca de dados e informação.

CD 7.1. As metodologias de integração são definidas para dar resposta à necessidade de integração de sistemas.

CD 7.2. As diferentes técnicas de integração de sistemas são utilizadas, tendo em conta o projeto.

CD 7.3. Os diferentes sistemas de informação são integrados, de acordo com o projeto, utilizando as técnicas de integração adequadas.

CD 7.4. A integração dos diferentes sistemas é realizada, utilizando tecnologias *standard*.

CD 7.5. As interfaces de acesso do sistema são utilizadas para garantir a troca de informação entre os sistemas dos diferentes serviços.

CD 7.6. As aplicações híbridas que utilizam repositórios de informação e bibliotecas de código heterogéneos, são desenvolvidas, de acordo com as necessidades e especificações do projeto.

CD 7.7. As aplicações desenvolvidas são testadas para garantir as suas funcionalidades.

Contexto profissional

Meios de produção

Equipamentos informáticos e periféricos. Ferramentas de documentação de código. Ferramentas de desenvolvimento e depuração. Linguagem de programação. Componentes de aplicações informáticas - API. Gestores de protocolos. Ferramentas CASE. Servidores Web. Servidores de aplicações. Servidores de Base de Dados. Sistemas de Gestão de Bases de Dados (SGDB). Ferramentas de transferência de arquivos. Máquinas virtuais. Sistemas de segurança. Ferramentas de controlo de versões (CVS). Cabos de interconexão dos dispositivos móveis.

REVOGADO

Produtos e resultados

Aplicações web desenhadas e programadas. Aplicações desktop desenhadas e programadas. Portais programados. Programas para redes sociais programados. Relatório e *dashboard* dos dados produzidos. Sistemas web desenvolvido, testado e implantado. API's desenvolvidos e reutilizados. Relatórios de Dados extraídos e manipulados. Bot e Chat programado.

Informação utilizada ou gerada

Diagramas do projeto. Diagrama Entidade Relacionamento. Diagrama de Fluxo de Dados. Estrutura global do projeto *software* em desenvolvimento. Manuais de utilizador dos sistemas informáticos. Manuais das linguagens de programação. Suportes de assistência técnica. Desenho técnico. Padrões de trabalho. Documentos de desenvolvimento dos componentes, Protótipo do sistema. Código (*source*) de componentes *software*. Código (*target*) de componentes *software*. Documentos estáticos e dinâmicos com código para interpretação. Documentação do desenvolvimento. Utilização de elementos multimídia Bibliotecas de *software*.

REVOGADO

UC3: DESENVOLVER APLICAÇÕES PARA *DESKTOP*	
Código: UC285_5	**Nível: 5**

Elementos de Competência (EC) e Critérios de Desempenho (CD)

EC1: Realizar o levantamento e a análise dos requisitos do sistema a ser desenvolvido.

CD 1.1. O problema a ser solucionado é identificado e o escopo é limitado de acordo com a sua natureza e a necessidade do cliente.

CD 1.2. As necessidades do cliente são levantadas, utilizando técnicas adequadas.

CD 1.3. Os requisitos funcionais são especificados de acordo com as necessidades do cliente levantados.

CD 1.4. Os requisitos funcionais especificados são analisados e hierarquizados de acordo com o grau de prioridade.

CD 1.5. A especificação dos requisitos é realizada como forma de especificar o *software* a ser desenvolvido.

CD 1.6. Os requisitos funcionais especificados são revisados e validados, para garantir que o entendimento dos problemas coincide com o que o cliente havia explicado.

EC2: Realizar planeamento de programação *Desktop*.

CD 2.1. A aplicação é desenhada para organizar ideias e conteúdo (textos, imagens e desenhos de apoio).

CD 2.2. A metodologia do trabalho é definida de acordo com os requisitos do sistema.

CD 2.3. A arquitetura do sistema é selecionada de acordo com o plano estabelecido.

CD 2.4. A linguagem de programação é selecionada de acordo com o ambiente de programação selecionado e garantindo a compatibilidade com o *hardware*.

CD 2.5. O Sistema de Gestão de Base de Dados (SGBD) é selecionado de acordo com o plano pré-estabelecido.

CD 2.6. O padrão da interface gráfica é definido tendo em vista o público-alvo.

CD 2.7. O cronograma de execução do trabalho é elaborado de acordo a metodologia e os requisitos funcionais e não funcionais.

CD 2.8. O plano de trabalho é elaborado de acordo com os requisitos levantados.

CD 2.9. Os equipamentos para execução (*hardware* e *software*) são inventariados de acordo com os requisitos do sistema a ser desenvolvido.

CD 2.10. O orçamento é definido e apresentado ao cliente de acordo com os requisitos Funcionais e Não funcionais.

EC3: Realizar a implementação do *software*/sistema.

CD 3.1. A documentação técnica é interpretada para implementação do sistema.

CD 3.2. O Sistema é implementado utilizando ferramentas de *Software*, bibliotecas de classes pré-existentes para agilizar a atividade assim como ferramentas *case* que dinamizam o processo de desenvolvimento, nas várias atividades onde inclui a geração de código fonte.

CD 3.3. A codificação da aplicação é desenvolvida de acordo com a documentação técnica utilizando uma linguagem de programação predefinida no plano.

CD 3.4. Os componentes desenvolvidos por terceiros ou fabricantes são reutilizados para incluir funcionalidades específicas no projeto, conforme as normas vigentes.

CD 3.5. O código fonte da aplicação desenvolvida é compilado para verificar a conformidade léxica, sintática e semântica (detetar anomalias).

CD 3.6. O relatório de *debug* é analisado para possível intervenção corretiva.

REVOGADO

CD 3.7. O teste do sistema é realizado para validação dos requisitos funcionais.

CD 3.8. Os módulos desenvolvidos e compilados são montados para garantir a funcionalidade integral do sistema.

CD 3.9. O código objeto do sistema desenvolvido é transformado em ficheiro executável para sua posterior instalação.

CD 3.10. A documentação técnica do componente é redigida conforme os formatos, as normas e os procedimentos estabelecidos no protocolo do projeto.

CD 3.11. O código desenvolvido e validado é armazenado no repositório do projeto como forma de controlar as versões da aplicação em desenvolvimento.

EC4: Realizar teste do sistema desenvolvido.

CD 4.1. O ambiente para realização do teste da aplicação é organizado conforme o plano de testes pré-estabelecidos.

CD 4.2. O tipo de teste é selecionado de acordo com o cenário projetado e o plano de teste pré-estabelecido.

CD 4.3. Os requisitos *hardware* e *software* do sistema são verificados antes da instalação do sistema desenvolvido.

CD 4.4. O ambiente de teste é definido de acordo com as normas pré-estabelecidas.

CD 4.5. Os programas e os dados de teste são organizados e armazenados em condições de segurança, conforme as normas de segurança da organização.

CD 4.6. O sistema desenvolvido é instalado e configurado com os parâmetros desejados para a realização dos testes programados.

CD 4.7. O sistema desenvolvido e instalado é versionado, utilizando *software* de versionamento adequado para gerir as diferentes versões testadas.

CD 4.8. Os procedimentos para realização de *backup* dos dados e teste de programas são estabelecidos.

CD 4.9. As ferramentas de desenvolvimento e teste são instaladas e configuradas conforme os parâmetros estabelecidos nas especificações do projeto de *software*.

CD 4.10. Os parâmetros do sistema relativos à ergonomia, segurança ou às condições de utilização do sistema são ajustados aos valores desejados dentro das normas e dos padrões da organização.

CD 4.11. Os dados para testes são preparados de acordo com o cenário de teste projetado.

CD 4.12. Os testes são realizados utilizando os dados preparados de acordo com os casos de teste definidos no plano de teste.

CD 4.13. Os testes de regressão são realizados para garantir o correto funcionamento do sistema como um todo e garantir que os novos defeitos introduzidos sejam encontrados e removidos.

CD 4.14. Os programas de gestão de projetos são utilizados na calendarização das atividades e tarefas.

CD 4.15. As ferramentas de testes são selecionadas e instaladas para produzir indicadores, gráficos e relatórios sobre processo de desenvolvimento e notificar sobre o ocorrido, como alarmes e anomalias.

CD 4.16. Os resultados de teste são comparados com os casos de teste projetados para analisar os possíveis defeitos do sistema.

CD 4.17. Os casos de teste e os resultados dos testes são documentados utilizando modelo pré-estabelecido respeitando as normas da organização.

EC5: Realizar a implementação do sistema desenvolvido e capacitar os utilizadores para a sua utilização.

CD 5.1. Os utilizadores são capacitados para utilizar o sistema desenvolvido em fase de testes.

CD 5.2. Os administradores do sistema são capacitados para administrar o sistema em fase de testes.

CD 5.3. Os canais de comunicação entre os administradores de sistemas e administradores de base de dados são criados e mantidos durante a implementação, em fase de testes, do sistema desenvolvimento.

REVOGADO

CD 5.4. A documentação e a apresentação do relatório de testes são produzidas com recurso a *software* específico, para sua posterior análise.

CD 5.5. O manual de utilizador é elaborado para ajudar o utilizador final.

CD 5.6. A documentação técnica e os manuais de instalação do sistema desenvolvido são elaborados para auxiliar a equipa de *deploy*.

CD 5.7. A comunicação ou partilha de informações entre as equipas de desenvolvimento é feita com recurso a *software* adequado, respeitando as normas e os padrões da organização, a fim de facilitar o trabalho de equipa.

Contexto profissional

Meios de produção

Equipamentos informáticos e periféricos. Ferramentas de documentação de código. Ferramentas de desenvolvimento e depuração. Linguagem de programação. Componentes aplicações informáticas API. Gestores de protocolos. Ferramentas CASE. Linhas de comunicações. Servidores Web. Servidores de aplicações. Servidores de base de dados. Sistemas de Gestão de Bases de Dados (SGBD). Ferramentas de transferência de arquivos. Máquinas virtuais. Sistemas de segurança. Ferramentas de controlo de versões (CVS). Ferramentas de partilha de informação e trabalho em equipa. Ferramentas para produção de relatórios e manuais de utilizador.

Produtos e resultados

Manual de utilizador produzido. Manual de instalação do sistema desenvolvido. Relatório de teste elaborado. Relatório de validação dos requisitos funcionais dos sistemas. Manual dos requisitos não funcionais do sistema elaborado. Plano de testes projetados. Cenário de testes definidos. Diagramas de Sistemas desenvolvidos.

Informação utilizada ou gerada

Sistemas desenvolvidos para diferentes plataformas. Modelos e diagramas do sistema implementado desenhado. Relatório de testes. Documentação técnica do projeto. Ficheiro de sistemas desenvolvidos. Documentos estáticos e dinâmicos que contém código fonte desenvolvido. Documentação do desenvolvimento realizado. Utilização de elementos multimídia existentes e bibliotecas de *software* existente.

REVOGADO

UC4: DESENVOLVER APLICAÇÕES PARA DISPOSITIVOS MÓVEIS	
Código: UC286_5	**Nível: 5**

Elementos de Competência (EC) e Critérios de Desempenho (CD)

EC1: Realizar o planeamento da programação de aplicativos móveis.

CD 1.1. A aplicação é esboçada para organizar ideias e conteúdos.

CD 1.2. A metodologia do trabalho é definida de acordo com os requisitos.

CD 1.3. O tipo de aplicação (nativo ou híbrido) a ser desenvolvida é selecionada de acordo com o projeto.

CD 1.4. O mercado é analisado para introduzir diferencial na aplicação a ser desenvolvida.

CD 1.5. O plano e o cronograma de execução do trabalho são elaborados de acordo com a metodologia e os requisitos funcionais e não funcionais.

CD 1.6. Os equipamentos necessários para execução dos trabalhos (*hardware* e *software*) são identificados a partir dos requisitos.

CD 1.7. O orçamento é definido e apresentado ao cliente de acordo com os requisitos funcionais e não funcionais.

EC2: Implementar interface ergonómico (UX/UI).

CD 2.1. O protótipo da aplicação é desenhado de acordo com os requisitos funcionais do projeto.

CD 2.2. O *template* é desenhado e ajustado ao *layout* de acordo com os dispositivos.

CD 2.3. O *template* é montado utilizando tecnologias *web* (HTML, CSS, *Bootstrap*, JS, JQ).

CD 2.4. O protótipo é testado para garantir a interação do utilizador com o a aplicação.

CD 2.5. O relatório de teste é analisado para possíveis reajustes.

EC3: Implementar a linguagem de programação multiplataforma para dispositivos móveis (Android/IOS).

CD 3.1. A documentação técnica é interpretada para implementação da aplicação.

CD 3.2. A codificação da aplicação é desenvolvida de acordo com a documentação técnica, utilizando uma linguagem de programação predefinida no projeto.

CD 3.3. O relatório de *debug* é analisado para possíveis intervenções corretivas.

CD 3.4. O teste da aplicação é realizado para validação dos requisitos funcionais.

Contexto profissional

Meios de produção

Equipamentos informáticos, dispositivos móveis e periféricos. *Software* de documentação de código fonte. *Software* de desenvolvimento e depuração. Linguagem de programação. Componentes de aplicações informáticas - API. Bibliotecas de *softwares*. Gestores de protocolos. Ferramentas CASE. Linhas de comunicações. Servidores *Web*. Servidores de aplicações. Servidores de Base de dados. Sistemas de Gestão de Bases de Dados (SGDB). Ferramentas de transferência de arquivos. Máquinas virtuais. Sistemas de segurança. Ferramentas de controlo de versões (CVS). Ligação à internet. GPS. Sensores. Sistema operativo Android, IoS, entre outros. Cabo de conexão dos dispositivos móveis.

REVOGADO

Produtos e resultados

Sistemas e aplicações para a internet desenvolvidos; *Sites*, lojas virtuais, portais, *bot, Chat* desenvolvidos.

Informação utilizada ou gerada

Sistemas e aplicações *mobile* para a internet, sites, lojas virtuais, portais; desenho de protótipos. Documentação de código fonte. Ficheiros do sistema desenvolvido. Documentos estáticos e dinâmicos com código fonte. Documentação do desenvolvimento. Elementos multimídia. Bibliotecas de *software*.

REVOGADO

UC5: DESENVOLVER APLICAÇÕES PARA INTERNET DAS COISAS - IoT	
Código: UC287_5	**Nível: 5**

Elementos de Competência (EC) e Critérios de Desempenho (CD)

EC1: Realizar a planificação do desenvolvimento de sistema para internet das Coisas - *IoT.*

CD 1.1. Os componentes físicos do projeto são identificados e selecionados.

CD 1.2. A arquitetura do sistema é selecionada de acordo as regras do negócio definidas no projeto.

CD 1.3. A arquitetura do projeto IoT é desenhado, tendo em conta os aspetos físicos e virtuais do projeto.

CD 1.4. O protótipo do sistema é desenhado, tendo em conta as especificações definidas no projeto.

EC2: Realizar a programação de microcontroladores e sensores.

CD 2.1. O esquema do circuito é interpretado e os respetivos componentes identificados.

CD 2.2. As *protoboards* (placas) de microcontroladores são selecionados de acordo com as atividades a desenvolver.

CD 2.3. Os componentes são montados e interligados na *protoboard*.

CD 2.4. A conexão do dispositivo na *protoboard* é realizada, utilizando as portas adequadas.

CD 2.5. O Ambiente de Desenvolvimento Integrado (IDE) é instalado e configurado para codificação da aplicação.

CD 2.6. A programação do microcontrolador é realizada para garantir o funcionamento adequado do dispositivo.

EC3: Realizar a recolha e análise dos dados para monitorização dos dispositivos.

CD 3.1. Os dados a serem recolhidos para monitorização do dispositivo são definidos de acordo com o projeto.

CD 3.2. Os padrões dos dados a recolher são definidos, para evitar erros de tratamentos e reduzir o tempo de processamento.

CD 3.3. As ferramentas para análise dos dados são selecionadas.

CD 3.4. A integração entre o dispositivo e o sistema é realizada, para a recolha dos dados.

CD 3.5. Os dados recolhidos do dispositivo são tratados para eliminar o desnecessário, de acordo com o objetivo.

CD 3.6. Os dados tratados são processados e analisados, utilizando técnicas e modelos de Inteligência do Negócio (BI) e de *Big Data*.

CD 3.7. A cópia de segurança dos dados recolhidos é efetuada de acordo com a política de segurança predefinida.

CD 3.8. As ferramentas para monitorização dos dispositivos são identificadas e selecionadas de acordo com a plataforma.

EC4: Realizar a conetividade entre dispositivos de *IoT* e a Internet.

CD 4.1. O dispositivo de IoT é conectado à rede IP com base em padrões.

CD 4.2. A comunicação e a interoperabilidade entre o dispositivo de IoT e a rede IP são asseguradas, utilizando protocolos de comunicação e API.

CD 4.3. A proteção da informação entre o dispositivo de IoT e a rede IP é assegurada, utilizando as técnicas de proteção de dados.

CD 4.4. Os dispositivos e os sensores de IoT são conectados, utilizando as tecnologias de rede, de acordo com o programa de monitorização.

CD 4.5. A conetividade é realizada com a finalidade de transmitir os dados.

REVOGADO

CD 4.6. Os dados produzidos são enviados para servidores (virtual ou físico) para facilitar o processo de monitorização.

CD 4.7. O roteador ou *Gateway* é configurado utilizando os protocolos de comunicação padrão para interconectar dispositivos à Internet.

EC5: Aplicar a aprendizagem da máquina (*Machine Learning*) na programação dos dispositivos.

CD 5.1. Os dados gerados pelos dispositivos de IoT são analisados, utilizando as técnicas de Inteligência Artificial (IA), para construção de modelos de inteligência artificial.

CD 5.2. O modelo de IA é construído, utilizando técnicas de IA, a partir dos dados gerados pelo dispositivo de IoT.

CD 5.3. A máquina é treinada para testar a aprendizagem.

CD 5.4. Os dados produzidos pelos dispositivos IoT são analisados em tempo real, utilizando técnicas de IA para monitorização do dispositivo.

CD 5.5. Os dados analisados são transmitidos em tempo real utilizando as tecnologias.

CD 5.6. Os dados analisados são disponibilizados através de dispositivos móveis, plataforma web, entre outros, para monitorizar os dispositivos de IoT.

EC6: Aplicar técnicas de segurança para proteger os dispositivos de IoT.

CD 6.1. O código de acesso forte ou biometria é definido, utilizando as técnicas adequadas para proteção do dispositivo.

CD 6.2. A camada de dados do dispositivo é separada da camada de regras de negócios, utilizando ferramentas adequadas.

CD 6.3. A configuração automática das atualizações de antivírus é feita na aplicação, para proteger os dispositivos de ataques cibernéticos.

CD 6.4. Os dados transferidos entre os dispositivos conectados são criptografados para proteger as informações.

CD 6.5. Os *softwares* de atualização dos dispositivos são instalados para garantir a segurança dos equipamentos.

CD 6.6. Os recursos disponíveis não utilizados, nos dispositivos, são desativados para reduzir as possíveis oportunidades de ataque.

Contexto profissional

Meios de produção

Equipamentos informáticos. Dispositivos móveis e periféricos. *Software* de documentação de código. Ferramentas de desenvolvimento e depuração. *Software* de prototipagem. Linguagem de programação. Aplicações informáticas. API. Gestores de protocolos. Ferramentas *CASE*. Linhas de comunicações. Servidores *Web*. Servidores de aplicações. Servidores de Base de dados. Sistemas de Gestão de Bases de Dados (SGDB). Ferramentas de transferência de arquivos. Máquinas virtuais. Sistemas de segurança. Ferramentas de controlo de versões (CVS). Ligação à internet. GPS. Sensores. Sistema operativo que *Protoboard*. Kit de Arduino. Cabo USB. Microcontroladores. Componentes eletrónicos. Módulo *Bluetooth*, Módulo NodeMCU. Potenciómetro linear. Multímetro Digital.

Produtos e resultados

Dispositivos ligados à internet - dispositivos IoT. Dados analisados e tratados. Modelos de IA produzidos. Dispositivos programados. Monitorização dos dispositivos de IoT realizados. Sensores manipulados e calibrados.

Informação utilizada ou gerada

Diagramas do projeto. Diagrama Entidade Relacionamento. Diagrama de Fluxo de Dados. Estrutura global do projeto *software* em desenvolvimento. Manuais de utilizador dos sistemas informáticos. Manuais das linguagens

REVOGADO

de programação. Suportes de assistência técnica. Desenho técnico. Documentos de desenvolvimento dos componentes. Protótipo do sistema. Documentos estáticos e dinâmicos com código fonte desenvolvido. Documentação do desenvolvimento. Elementos multimídia. Bibliotecas de *software*.

REVOGADO

UC6: DESENVOLVER APLICAÇÕES PARA JOGOS	
Código: UC288_5	**Nível: 5**

Elementos de Competência (EC) e Critérios de Desempenho (CD)

EC1: Realizar planificação para programação de jogos.

CD 1.1. O roteiro para programação de jogos é interpretado, de acordo com o cenário.

CD 1.2. A mecânica do jogo é definida, de acordo com o roteiro para definir as regras do jogo.

CD 1.3. A movimentação das personagens de jogos é definida, de acordo com as ações definidas no cenário.

CD 1.4. A narrativa do jogo (personagens dos jogos, Cenário, tempo do jogo) é elaborada e interpretada, de cordo com o cenário.

CD 1.5. O conteúdo, a ambientação, o gênero e o tipo de jogo são esboçados de acordo com o roteiro.

CD 1.6. A tecnologia do jogo é definida, em conformidade com o roteiro e o projeto.

CD 1.7. A linguagem e/ou motores (*engine*) de programação de jogo é definida, tendo em vista os objetivos definidos no projeto.

CD 1.8. O cenário do jogo é modelado, utilizando as técnicas e ferramentas adequadas.

EC2: Realizar prototipagem do jogo.

CD 2.1. O *software* de prototipagem de jogo é selecionado.

CD 2.2. A tipologia de prototipagem é definida.

CD 2.3. Os elementos da cena para prototipagem são definidos.

CD 2.4. O protótipo é desenhado de acordo com o roteiro definido.

CD 2.5. A mecânica e a narrativa do jogo são testadas por meio de interação.

CD 2.6. O protótipo produzido é renderizado para correção de defeitos de produção.

EC3: Codificar jogos para diferentes plataformas (PC, *Mobile*, Consolas).

CD 3.1. O *framework* e ou motor de jogo é selecionado.

CD 3.2. A velocidade do movimento dos elementos é definida de acordo com o cenário.

CD 3.3. Os elementos do cenário são animados em todas as direções.

CD 3.4. Os ícones de animação são desenhados para direcionar os movimentos.

CD 3.5. Os eventos associados aos ícones são definidos.

CD 3.6. A movimentação dos elementos é feita de acordo com a ordem dada aos ícones.

CD 3.7. Os ícones do jogo são alterados quando pressionados, de acordo com o cenário.

CD 3.8. O jogo é testado para o ajustar aos padrões definidos.

EC4: Publicar aplicativos de jogos nos diferentes canais/plataformas digitais.

CD 4.1. As plataformas e os canais de publicação de aplicativo de jogos são selecionados tendo em conta os requisitos de publicação.

CD 4.2. A Inscrição na plataforma é realizada de acordo com os critérios pré-estabelecidos para criação da conta.

CD 4.3. Os ficheiros do aplicativo desenvolvido são organizados para publicação na plataforma.

CD 4.4. Os ficheiros do aplicativo são hospedados nos servidores de produção, utilizando ferramentas adequadas.

CD 4.5. Teste de aplicativo no ambiente de produção é realizado para garantir o bom funcionamento e a qualidade do produto.

REVOGADO

CD 4.6. A publicação de aplicativo de jogos é realizada nas diferentes plataformas.

Contexto profissional

Meios de produção

Equipamentos informáticos e periféricos. Ferramentas de documentação de código. Ferramentas de desenvolvimento e depuração. Linguagem de programação. Aplicações informáticas. API. Gestores de protocolos. Ferramentas CASE. Linhas de comunicações. Servidores web. Servidores de aplicações. Servidores de Base de dados, Sistemas de Gestão de Bases de dados - SGBD. Ferramentas de transferência de arquivos. Máquinas virtuais. Sistemas de segurança. Ferramentas de controlo de versões (CVS). Cabos de interconexão dos dispositivos móveis. Plataforma. Consola. Canais de jogos: Linguagem de modelação 2D/3D. *Frameworks* para desenvolver jogos, *frameworks* para prototipagem de jogos.

Produtos e resultados

Jogos desenvolvidos e disponíveis para diversas plataformas; API desenvolvidos para jogos digitais. Código (*source*) de componentes *software* desenvolvido. Código (*target*) de componentes *software* desenvolvido. Documentos estáticos e dinâmicos com código desenvolvidos. Documentação dos jogos produzidos. Relatório e *dashboard* dos dados produzidos. Sistemas web desenvolvido, testado e implantado.

Informação utilizada ou gerada

Diagramas do projeto. Diagrama Entidade Relacionamento, Diagrama de Fluxo de Dados. Estrutura global do projeto *software* em desenvolvimento. Manuais de utilizador dos sistemas de jogos. Manuais dos motores de jogos. Manuais das linguagens de programação. Prototipagem de jogos. Lista dos canais para publicação de jogos. Desenho técnico de jogos. Documentos de desenvolvimento dos componentes. Protótipo do sistema, Linguagem de Modelação 2D/3D; Elementos gráficos. Documentos estáticos e dinâmicos com código fonte. Documentação de desenvolvimento de jogos. Elementos multimídia, bibliotecas de *software*.

REVOGADO

PROGRAMA FORMATIVO ASSOCIADO AO PERFIL PROFISSIONAL

TIC001.A1_5

PROGRAMAÇÃO

REVOGADO

PROGRAMA FORMATIVO DA QUALIFICAÇÃO PROFISSIONAL			
Código	TIC001.A1_5	**Denominação**	PROGRAMAÇÃO
Nível	5	**Família profissional**	Tecnologias de Informação e Comunicação (TIC)
Duração Indicativa		2.240 horas	

	N.º	Denominação	Código
Unidades de Competência	**1**	Desenvolver aplicações front-end	**UC283_5**
	2	Desenvolver aplicações back-end	**UC284_4**
	3	Desenvolver aplicações desktop	**UC285_5**
	4	Desenvolver aplicações para dispositivos móveis	**UC286_5**
	5	Desenvolver aplicações para Internet das Coisas (IoT)	**UC287_5**
	6	Desenvolver aplicações para Jogos	**UC288_5**

MÓDULO FORMATIVO (MF)			UNIDADES FORMATIVAS (associadas aos MF)	
N.º	**Denominação**	**Código**	**Denominação**	**Código**
1	Unidades Formativas (transversais) **360 horas**		Desenho do Algoritmo (35 horas)	**UF374**
			Metodologia de Desenvolvimento de *Software* (40 horas)	**UF375**
			Desenho de Interfaces (40 horas)	**UF376**
			Framework para programação *web* (30 horas)	**UF377**
			Programação estruturada básica (50 horas)	**UF378**
			Programação orientada a objetos (50 horas)	**UF379**
			Programação web (80 horas)	**UF380**
			Segurança de Sistemas de Informação (35 horas)	**UF381**
1	Programação de aplicações front-end **(170 horas)**	**MF283_5**	Linguagem de Marcação - HTML Avançada (30 horas)	**UF382**
			Linguagem de Formatação CSS Avançada (30 horas)	**UF383**
			Linguagem de Programação - JavaScript Avançada (60 horas)	**UF384**
			Integração de Sistemas (50 horas)	**UF385**
2	Programação de aplicações *back-end* **(200 horas)**	**MF284_5**	Programação web do lado do servidor (50 horas)	**UF386**
			Estrutura da base de dados relacional (30 horas)	**UF387**
			Implementação de Base de Dados (30 horas)	**UF388**

REVOGADO

			Programação em SQL (40 horas)	**UF389**
			Programação Orientada a Objetos avançada (50 horas)	**UF390**
3	Programação de aplicações para desktop **(170 horas)**	**MF285_5**	Programação estruturada avançada (50 horas)	**UF391**
			Programação Orientada a Objetos avançada (50 horas)	**UF390**
			Estrutura da base de dados relacional (30 horas)	**UF387**
			Programação em SQL (40 horas)	**UF389**
4	Programação de aplicações para dispositivos móveis **(200 horas)**	**MF286_5**	Arquitetura de Sistemas Operativos para Dispositivos Móveis (30 horas)	**UF392**
			Programação Mobile Android (60 horas)	**UF393**
			Programação Mobile IoS (60 horas)	**UF394**
			Programação Mobile Multiplataforma (50 horas)	**UF395**
5	Programação de aplicações para IoT **(400 horas)**	**MF287_5**	Fundamentos de eletrónica (50 horas)	**UF396**
			Fundamentos de Rede de Comunicação (40 horas)	**UF397**
			Sistemas embarcados e microcontroladores (80 horas)	**UF398**
			Inteligência artificial para IoT (60 horas)	**UF399**
			Análise de dados e Monitorização dos dispositivos IoT (50 horas)	**UF400**
			Programação Mobile Multiplataforma para IoT (120 horas)	**UF401**
6	Programação de Aplicações para Jogos **(500 horas)**	**MF288_5**	Guionismo de jogos digitais (80 horas)	**UF402**
			Desenho de Jogos (40 horas)	**UF403**
			Modelação e Animação 2D e 3D (60 horas)	**UF404**
			Áudio para Jogos (40 horas)	**UF405**
			Inteligência artificial aplicada a jogos (80 horas)	**UF406**
			Motores de Jogos (80 horas)	**UF407**
			Programação de Jogos (120 horas)	**UF408**
MT_TIC001.A1		**Módulo Formativo em Contexto Real de Trabalho (360 horas)**		

REVOGADO

UNIDADES FORMATIVAS TRANSVERSAIS

UNIDADE FORMATIVA 1: UF374 Desenho do Algoritmo (35 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA1. Conhecer as técnicas de programação estruturada.

- **CA 1.1.** Identificar os tipos de dados básicos utilizados na definição de um algoritmo.
- **CA 1.2.** Identificar as variáveis e as constantes de estrutura de dados.
- **CA 1.3.** Conhecer expressões e operadores utilizados no processamento de dados.
- **CA 1.4.** Indicar as variáveis de entrada, processamento e saída de dados.
- **CA 1.5.** Diferenciar as estruturas de controlo de fluxo de execução de instruções.
- **CA 1.6.** Distinguir as estruturas de dados homogéneos.
- **CA 1.7.** Diferenciar a função do procedimento.

RA2. Representar algoritmo em forma de fluxograma e Pseudocódigo.

- **CA 1.1.** Definir e escrever algoritmo.
- **CA 1.2.** Identificar as diferentes formas de representação de algoritmos.
- **CA 1.3.** Definir o fluxograma.
- **CA 1.4.** Identificar e caraterizar as simbologias que compõem um fluxograma.
- **CA 1.5.** Diferenciar um fluxograma de um pseudocódigo.
- **CA 1.6.** Numa situação prática de representação de algoritmos em fluxograma:
  - Selecionar as simbologias de entrada e saída de dados;
  - Selecionar as simbologias de processamento;
  - Desenhar o fluxograma do algoritmo.
- **CA 1.7.** Numa situação prática de representação de algoritmos em pseudocódigo:
  - Definir as palavras chaves de entrada, saída e processamento de dados;
  - Declarar e inicializar as variáveis e constantes da estrutura de dados;
  - Definir a estrutura condicional adequada ao fluxo de dados;
  - Definir o ciclo repetitivo adequado ao fluxo de dados;
  - Escrever o pseudocódigo.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA2 relativamente aos CA 2.6 e CA 2.7.

Conteúdos

1. Introdução à Lógica Matemática.
   - 1.1. Proposições e conectivos.
   - 1.2. Operações lógicas sobre proposições.
   - 1.3. Construção de tabelas-verdade.
2. Conceito e formas de representação de algoritmos.
   - 2.1. Descrição narrativa.
   - 2.2. Fluxograma convencional.
   - 2.3. Pseudocódigo.
3. Técnicas Básicas de Programação.
   - 3.1. Tipos de dados.

REVOGADO

3.1.1. Tipos Inteiros.

3.1.2. Tipos Reais.

3.1.3. Tipos Caracteres.

3.1.4. Tipos Lógicos.

3.2. Variáveis e Constantes.

3.2.1. Armazenamento de dados na memória.

3.2.2. Conceito e utilidade de cariáveis.

3.2.3. Conceito e utilidade de Constantes.

3.2.4. Definição de variáveis e constantes em algoritmos.

3.3. Expressões e Operadores.

3.3.1. Operadores de atribuição.

3.3.2. Operadores aritméticos.

3.3.3. Operadores relacionais.

3.3.4. Operadores lógicos.

3.3.5. Operadores literais.

3.3.6. Expressões aritméticas.

3.3.7. Expressões lógicas.

3.3.8. Expressões literais.

3.3.9. Avaliação das expressões.

3.4. Instruções primitivas.

3.4.1. Comando de Atribuição.

3.4.2. Comandos de entrada de dados.

3.4.3. Comandos de saída de dados.

3.4.4. Entrada, processamento e saída de dados.

3.5. Estruturas de controlo de fluxo de execução.

3.5.1. Estrutura sequencial.

3.5.2. Estruturas de decisão: Simples, Composta e Múltipla.

3.5.3. Estruturas de repetição: Laços condicionais, Laços contados.

3.5.4. Estruturas de controlo encadeadas ou aninhadas.

3.6. Estruturas de dados homogéneas

3.6.1. Matrizes de uma dimensão ou vetores: Atribuição, Leitura e Escrita e Método da Bolha de Classificação.

3.6.2. Matrizes com Mais de Uma Dimensão: Atribuição, Leitura, Escrita e operações básicas.

3.7. SubAlgoritmos.

3.7.1. Conceito.

3.7.2. Função.

3.7.3. Procedimento.

3.7.4. Variáveis Globais e Locais.

3.7.5. Parâmetros.

3.7.6. Mecanismos de Passagem de Parâmetros.

3.7.7. Refinamentos Sucessivos.

REVOGADO

UNIDADE FORMATIVA 2: UF375 Metodologia de Desenvolvimento de *Software* (40 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Realizar o levantamento, a análise e especificação de requisitos do sistema.

- **CA 1.1.** Identificar e delimitar o micromundo (área de atuação).
- **CA 1.2.** Identificar e analisar o problema.
- **CA 1.3.** Indicar as técnicas utilizadas no levantamento de requisitos.
- **CA 1.4.** Numa situação prática:
  - Realizar o levantamento dos requisitos, aplicando as técnicas adequadas;
  - Analisar e definir os requisitos do sistema.

RA 2: Desenhar o diagrama de Caso de Uso do sistema.

- **CA 2.1.** Definir caso de uso.
- **CA 2.2.** Descrever um caso de uso.
- **CA 2.3.** Identificar as funcionalidades do sistema.
- **CA 2.4.** Distinguir os diferentes autores do sistema.
- **CA 2.5.** Reconhecer a relação entre casos de uso.
- **CA 2.6.** Indicar a relação entre autores.
- **CA 2.7.** Numa situação prática, desenhar um diagrama de caso de uso, utilizando as ferramentas de diagramas UML.

RA 3: Desenhar o diagrama de Classe e objeto do sistema.

- **CA 3.1.** Indicar as classes de um sistema.
- **CA 3.2.** Caraterizar os componentes de uma classe.
- **CA 3.3.** Relacionar as classes de um sistema.
- **CA 3.4.** Numa situação prática de desenho de um diagrama de classe:
  - Identificar as classes do sistema a utilizar;
  - Definir os atributos e métodos das classes;
  - Definir o relacionamento entre as classes;
  - Instanciar objeto.

Resultados de Aprendizagem cujas aquisições devem ser complementadas em ambiente real de trabalho

- RA1 relativamente ao CA 1.4.
- RA2 relativamente ao CA 2.7.
- RA3 relativamente ao CA 3.4.

Conteúdos

1. Introdução à análise de *software*.
   - 1.1. Evolução das metodologias de modelação.
   - 1.2. Modelação Visual.
   - 1.3. Definição de UML.
   - 1.4. Método iterativo e incremental.
2. Diagramas.
   - 2.1. Diagramas de Casos de Uso.
     - 2.1.1. Atores e Casos de Uso.

2.1.2.Comunicação entre Atores e Casos de Uso.

2.1.3.Cenário Principal e Cenários Secundários.

2.1.4.Relações de *include*, *extend* e generalização.

2.2. Diagramas de Classes.

2.2.1.Objectos e Classes.

2.2.2.Associações e Multiplicidade.

2.2.3.Atributos e Operações.

2.2.4.Classes Associativas.

2.2.5.Generalização e Herança.

2.2.6.Agregação e Composição.

2.3. Diagramas de Atividades.

2.3.1.Atividades.

2.3.2.Transição entre Atividades.

2.3.3.Condicionais.

2.3.4.Processamento Paralelo.

2.4. Diagramas de Interação.

2.4.1 Diagramas de Sequência.

2.4.2 Diagramas de Colaboração.

2.4.3 Diagramas de Estado.

2.5 Diagrama de Componentes.

2.6 Diagrama de Implantação.

UNIDADE FORMATIVA 3: UF376 Desenho de Interfaces (40 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Identificar as especificações e funcionalidades do *software* a ser programado.

CA 1.1. Explicar a relação do protótipo com a *interface* final e com os outros componentes da aplicação informática.

CA 1.2. Distinguir as funcionalidades presentes no protótipo.

CA 1.3. Numa situação prática de especificação das funcionalidades do programa:

- Utilizar o modelo ou diagramas de caso de uso para organizar as funcionalidades;
- Testar a funcionalidade do desenho de interface em relação às necessidades requeridas.

RA 2: Organizar o desenho da *interface* de utilizador em função dos requisitos funcionais definidas no projeto.

CA 2.1. Identificar as ferramentas utilizadas no desenho de *interface*.

CA 2.2. Identificar os elementos que compõem uma *interface*.

CA 2.3. Explicar as funcionalidades das diferentes ferramentas utilizadas no desenho da *interface*.

CA 2.4. Numa situação prática de desenho de uma *interface*:

- Selecionar a ferramenta de desenho de *interface*;
- Identificar os componentes de *interface* da aplicação;
- Produzir casos de teste para a integração de *interface* com outros componentes da aplicação;
- Adaptar os componentes desenvolvidos a modificações das especificações recebidas;
- Desenhar *interface* de acordo com os requisitos do projeto;
- Testar a interação entre o utilizador e a *interface*;
- Preparar os componentes produzidos para a sua distribuição;
- Produzir os documentos de instalação, configuração ou administração necessários, de acordo com as normas e critérios de usabilidade.

RA 3: Reutilizar e adaptar *templates* e outros componentes multimédia no desenvolvimento da *interface* de utilizador.

CA 3.1. Explicar a legislação sobre propriedade intelectual e direitos de autor relativa aos elementos multimédia.

CA 3.2. Explicar as características e as funcionalidades das ferramentas de tratamento de componentes multimédia.

CA 3.3. Identificar a tecnologia e as ferramentas utilizadas no desenho de *templates*.

CA 3.4. Numa situação prática de reutilização de *templates* para desenvolvimento de uma *interface*:

- Projetar as transformações necessárias a fim de obter componentes multimédia, conforme os requisitos do projeto *software*;
- Verificar a adequação do *template* adaptado ao padrão definido no projeto.

RA 4: Desenhar a arquitetura de um sistema de informação.

CA 4.1. Identificar e caraterizar os diferentes espaços *web*.

CA 4.2. Diferenciar protótipo de alta, média e baixa fidelidades.

CA 4.3. Distinguir Tipologia de prototipagem.

CA 4.4. Diferenciar elementos de um *site*.

CA 4.5. Identificar os métodos e as técnicas de avaliação de *interface*.

REVOGADO

CA 4.6. Numa situação prática de desenho da arquitetura de um sistema de informação:

- Desenhar o protótipo do sistema;
- Adaptar o *template* às funcionalidades do *site web*;
- Testar o protótipo para garantir a sua funcionalidade.

Resultados de Aprendizagem cujas aquisições devem ser complementadas em ambiente real de trabalho

- RA1 relativamente ao CA 1.5.
- RA2 relativamente ao CA 2.2.
- RA3 relativamente ao CA 3.4.
- RA4 relativamente ao CA 4.5.

Conteúdos

1. Conceitos de Interface Homem-Computador (IHC).
   1.1. Conceito de Interface Homem-Computador (IHC).
   1.2. Evolução e Propriedades dos Sistemas Interativos.
   1.3. Dispositivos de Interação.
   1.4. Estilos de Interação.
   1.5. Aplicações.
2. Fatores humanos em IHC.
   2.1. Teoria, princípios e regras básicas.
   2.2. Fatores ergonómicos.
   2.3. Métodos de Implementação.
3. Conceção de interfaces.
   3.1. Aspetos da Psicologia Cognitiva e Semiótica.
   3.2. Ergonomia de *Software*.
4. Padrões para interface.
   4.1. Ciclo de Vida.
   4.2. Metodologias.
   4.3. Técnicas e ferramentas de conceção.
   4.4. Implementação de sistemas interativos e *interfaces*.
   4.5. Interfaces inteligentes.
5. Projeto de interface.
   5.1. Projeto de diálogo Homem-Máquina.
   5.2. Modelagem Estrutural.
   5.3. Modelagem da Arquitetura.
6. Avaliação de *interfaces*.
   6.1. Usabilidade e Ergonomia.
   6.2. Definição e métodos para avaliação.
   6.3. Métodos e técnicas para avaliar *interface* com utilizador.
   6.4. Princípios de avaliação em IHC, segundo as Heurísticas de Nielsen e o *framework* DECIDE.
7. Desenvolvimento de UI's para dispositivos móveis
   7.1. Introdução sobre design para pequenas *interfaces*.

7.2. Princípios de *design* de interação para dispositivos móveis.

7.3. Elementos e cuidados comuns na construção de pequenas *interfaces*.

7.4. Exemplos de *interfaces* para dispositivo controlo (*Mobile websites*, aplicações, *touch UI*, *low-end devices*).

7.5. UI *Guidelines*, *Design Patterns* (Nokia, Android, iPhone, Windows Phone).

7.6. Desenvolvendo *widgets* para *mobile*.

UNIDADE FORMATIVA 4: UF377 Framework para programação Web (30 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Conhecer os diferentes modelos de arquiteturas de *frameworks*.

CA 1.1. Definir *Framework*.

CA 1.2. Descrever as vantagens e desvantagens na utilização de *framework*.

CA 1.3. Diferenciar os tipos de *framework*.

CA 1.4. Explicar as funcionalidades de *framework*.

CA 1.5. Enumerar os modelos de *framework* existentes.

CA 1.6. Explicar a diferença entre os *Model-View-Controller* (*MVC*) e *Model-View-Template* (*MVT*).

RA 2: Instalar e configurar *frameworks*.

CA 2.1. Identificar os requisitos de *hardware* e *software* para instalação de um *framework*.

CA 2.2. Identificar o servidor *web* e servidor de base de dados compatível com o *framework*.

CA 2.3. Descrever a estrutura de diretório do *framework*.

CA 2.4. Identificar os parâmetros necessários para configuração do servidor.

CA 2.5. Numa situação prática de instalação e configuração de um *framework* para desenvolver aplicações:

- Selecionar o *framework*;
- Selecionar o Servidor *Web*;
- Selecionar o Servidor de Base de dados;
- Selecionar os *plugins* exigidos pelos *framework* para gerar a compatibilidade;
- Instalar *plugins*, servidor *web*, servidor de base de dados, *framework*;
- Configurar o servidor *web*, servidor da base de dados.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA2 relativamente ao CA 2.5.

Conteúdos

1. Introdução ao *framework*.
2. Configuração do ambiente de trabalho.
3. Estrutura hierárquica do *framework*.
4. Comandos.
5. Definição de rotas.
6. Padrão MVC e MVT.
7. *Model*.
8. *Controller*.
9. *Views*.
10. *Templates*.
11. Demonstração do *framework*.

VOGADO

UNIDADE FORMATIVA 5: UF378 Programação Estruturada Básica (50 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Conhecer as funcionalidades a serem implementadas no *software,* de acordo com os requisitos.

CA 1.1. Indicar as funções que são implementadas em cada fase de desenvolvimento de um programa.

CA 1.2. Diferenciar as linguagens de programação utilizadas no desenvolvimento de programa.

CA 1.3. Caraterizar os Ambientes de Desenvolvimentos utilizados para programação.

CA 1.4. Relacionar os componentes a serem programados com as bibliotecas internas.

RA 2: Criar um programa utilizando uma linguagem de programação estruturada.

CA 2.1. Definir o conceito de uma linguagem estruturada.

CA 2.2. Identificar as diferentes classes de linguagem de programação.

CA 2.3. Identificar as partes constituintes de um programa.

CA 2.4. Descrever a estrutura de uma linguagem de programação estruturada.

CA 2.5. Identificar os diferentes tipos de dados utilizados numa linguagem de programação estruturada.

CA 2.6. Explicar as regras de definição e declaração de uma variável.

CA 2.7. Identificar as sintaxes utilizadas numa linguagem de programação estruturada.

CA 2.8. Explicar as características e funcionalidades das ferramentas de desenvolvimento do programa.

CA 2.9. Explicar a sintaxe da linguagem de programação estruturada.

CA 2.10. Numa situação prática de desenvolvimento de um programa, utilizando uma linguagem de programação estruturada:

- Selecionar a linguagem de programação para implementação do algoritmo;
- Definir as variáveis de entrada, saída e processamento de dados;
- Declarar, atribuir e inicializar as variáveis da estrutura de dados;
- Declarar e definir as constantes;
- Definir as estruturas de acordo com o algoritmo;
- Escrever o código fonte;
- Compilar o código fonte;
- Realizar *debug* do código fonte;
- Testar o programa.

RA 3: Organizar e documentar o código fonte.

CA 3.1. Explicar os procedimentos mais comuns de organização do código fonte.

CA 3.2. Descrever as funcionalidades das ferramentas de controlo de versão (*Git*, CVS, *Mercurial*, TFS, *Bazaar*, entre outros).

CA 3.3. Identificar os documentos mais habituais associados a desenvolvimento do *software*.

CA 3.4. Numa situação prática de organização e documentação de código fonte:

- Selecionar as ferramentas de versionamento;
- Organizar o código;
- Comentar o código fonte;
- Versionar o código fonte;

REVOGADO

- Documentar o código fonte.

Resultados de Aprendizagem cujas aquisições devem ser complementadas em ambiente real de trabalho

- RA1 relativamente ao CA 1.5.
- RA2 relativamente ao CA 2.10.
- RA3 relativamente ao CA 3.4.

Conteúdos

1. Conceitos básicos de linguagens de programação estruturada.
   1.1. Computadores, programas, dados e instruções.
   1.2. Estrutura de um programa.
   1.3. Operações básicas de entrada e de saída.
2. Sintaxe da linguagem.
   2.1. Constantes e variáveis.
   2.2. Operadores.
   2.3. Tipos de dados.
3. Entrada e saída de dados.
   3.1. Instruções básicas.
4. Estruturas de controlo.
   4.1. Seleção.
   4.2. Sequência.
   4.3. Repetição.
   4.4. Recursividade.
5. Estruturas de dados.
   5.1. Vetores.
   5.2. Matrizes.
   5.3. Registos.
   5.4. Conjunto e arquivos.
6. Funções, procedimentos e passagem de parâmetros.

UNIDADE FORMATIVA 6: UF379 Programação Orientada a Objetos (50 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Caraterizar uma linguagem Orientada a Objetos.

- **CA 1.1.** Caraterizar os diferentes tipos de linguagem de programação.
- **CA 1.2.** Descrever uma linguagem orientada a objetos.
- **CA 1.3.** Definir uma classe.
- **CA 1.4.** Diferenciar uma classe de um objeto.
- **CA 1.5.** Identificar os atributos de uma classe.
- **CA 1.6.** Identificar o papel dos métodos de uma classe.
- **CA 1.7.** Distinguir tipos de visibilidade dos atributos e dos métodos de uma classe.
- **CA 1.8.** Distinguir as classes de uma linguagem orientada aos objetos.
- **CA 1.9.** Conhecer a sintaxe de uma linguagem orientada a objetos.
- **CA 1.10.** Classificar os objetos de uma classe.

RA 2: Conhecer os princípios da programação orientada a objetos utilizados na construção de um programa.

- **CA 2.1.** Reconhecer os elementos de construção de classes e objetos.
- **CA 2.2.** Aplicar técnicas de programação orientada a objetos no desenvolvimento de componentes de *software*.
- **CA 2.3.** Explicar a necessidade de utilizar as estruturas de dados complexas.
- **CA 2.4.** Identificar as bibliotecas de classes e funções existentes numa linguagem de programação orientada a objetos.
- **CA 2.5.** Distinguir os métodos de entrada/saída de dados mais adequados ao ambiente de programação.

RA 3: Editar e compilar o código fonte.

- **CA 3.1.** Descrever a estrutura de um programa orientado a objetos.
- **CA 3.2.** Descrever os diferentes tipos de dados existentes.
- **CA 3.3.** Explicar as regras de definição de uma variável.
- **CA 3.4.** Explicar as regras de definição de uma classe.
- **CA 3.5.** Diferenciar tipos de variáveis.
- **CA 3.6.** Reconhecer as funções de entrada e saída de dados.
- **CA 3.7.** Descrever as funções das estruturas de controlo.
- **CA 3.8.** Descrever a funcionalidade dos diferentes tipos de operadores.
- **CA 3.9.** Numa situação prática da edição de código orientada a objeto:
  - Definir as classes;
  - Declarar os atributos e métodos das classes;
  - Definir a visibilidade dos atributos e métodos das classes;
  - Declarar e inicializar as variáveis;
  - Editar código fonte:
  - Compilar o código fonte:
  - Realizar *debug* do código fonte;

- Testar o programa.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA3 relativamente ao CA 3.9.

Conteúdos

1. Programação orientada a objetos.
   1.1. Introdução á orientação a objetos:
      1.1.1. Missão, objetivos e aspetos de qualidade de *software*;
      1.1.2. Objeto, classe;
      1.1.3. Estado, comportamento e referencias;
      1.1.4. Identidade, abstração e encapsulamento;
      1.1.5. Métodos, atributos e variáveis estáticas;
      1.1.6. Criação e remoção de objetos, construtores, referências, *aliasing*, mecanismos de passagem de parâmetros, *overloading*;
      1.1.7. Herança e polimorfismo;
      1.1.8. Herança e *dynamic binding*;
      1.1.9. Classe abstrata e herança;
      1.1.10. *Interface*.
2. Tratamento de exceções.
3. Tipos estruturados de dados:
   3.1. *Strings*;
   3.2. Vetores;
   3.3. Matrizes;
   3.4. Arquivos textos.
4. Modularidade:
   4.1. Métodos estáticos (funções);
   4.2. Passagem de parâmetros (por valor e referencia);
   4.3. Bibliotecas de vínculos estáticos.
5. Pacotes e espaços de nomes.
6. Sintaxe da linguagem.
   6.1. Constantes e variáveis.
   6.2. Operadores.
   6.3. Tipos de dados.
   6.4. Entrada e saída de dados.
   6.5. Conversão de tipos.
7. Instruções e estruturas de controlo.
8. Coleções de objetos.
   8.1. Listas, conjuntos e Mapas.
   8.2. Tipos genéricos.
9. Interface gráfica com utilizador (GUI).

UNIDADE FORMATIVA 7: UF380 Programação Web (80 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Caraterizar a arquitetura de um sistema de informação.

- **CA 1.1.** Identificar os diferentes tipos de servidores num sistema.
- **CA 1.2.** Descrever as características do servidor (*Hardware* e *software*).
- **CA 1.3.** Identificar os protocolos de segurança utilizados no servidor.
- **CA 1.4.** Descrever os serviços de comunicação utilizados na transmissão de dados (largura da banda).

RA 2: Realizar programação para *web*.

- **CA 2.1.** Conhecer as linguagens de programação para desenvolvimento *web*.
- **CA 2.2.** Descrever a estrutura de um *web site*.
- **CA 2.3.** Explicar a função do protocolo "http" na programação *web*.
- **CA 2.4.** Diferenciar um *web site* estático de um dinâmico.
- **CA 2.5.** Identificar os métodos e as técnicas utilizados no desenvolvimento *web*.
- **CA 2.6.** Identificar as novas tecnologias utilizadas em linguagens de programação *web*.
- **CA 2.7.** Numa situação prática de programação de uma página *web*:
  - Selecionar as ferramentas de desenvolvimento *web*;
  - Codificar *site* estático e/ou dinâmico;
  - Testar o site utilizando *browser* para visualização.

RA 3: Realizar a hospedagem e a publicação do sistema.

- **CA 3.1.** Caraterizar os diferentes tipos servidores.
- **CA 3.2.** Descrever os diferentes tipos de *softwares* para servidores.
- **CA 3.3.** Identificar a versão e arquitetura dos *softwares* para servidores.
- **CA 3.4.** Definir um domínio.
- **CA 3.5.** Descrever o papel do protocolo TCP/IP na comunicação entre os servidores.
- **CA 3.6.** Descrever o papel de *Proxy server*.
- **CA 3.7.** Identificar os *softwares* para carregar ficheiros.
- **CA 3.8.** Identificar o sistema de segurança implementado.
- **CA 3.9.** Avaliar os parâmetros do servidor.
- **CA 3.10.** Numa situação prática de hospedagem e publicação do sistema:
  - Selecionar o plano de hospedagem;
  - Verificar a compatibilidade dos *softwares* instalados com o do sistema desenvolvido;
  - Solicitar o domínio num provedor;
  - Configurar o domínio do sistema;
  - Realizar *upload* dos ficheiros do sistema com recurso a *software* específico;
  - Testar as funcionalidades da aplicação;
  - Publicar o *site*.

Resultados de Aprendizagem cujas aquisições devem ser complementadas em ambiente real de trabalho

- RA2 relativamente ao CA 2.7.
- RA3 relativamente ao CA 3.10.

Conteúdos

1. Introdução à *Web*.
2. HTML.
   - 2.1. Fundamentos de HTML.
     - 2.1.1. Conceitos Básicos.
     - 2.1.2. A estrutura de um ficheiro HTML x.x.
     - 2.1.3. Características gerais da linguagem HTML.
     - 2.1.4. Comandos básicos de formatação de texto.
     - 2.1.5. Listas e enumerações; Interligação de documento.
     - 2.1.6. Inclusão de imagens.
     - 2.1.7. Definição de cores.
     - 2.1.8. Imagens mapeadas.
     - 2.1.9. Multimídia e animações.
     - 2.1.10. Formatação utilizando tabela.
     - 2.1.11. Páginas com *Frames*.
     - 2.1.12. Criação de Formulários.
     - 2.1.13. *Web site* e aplicação *web*.
     - 2.1.14. Editores e IDE.
3. CSS.
   - 3.1. Fundamentos de CSS.
   - 3.2. Introdução ao CSS (*Cascading Style Sheets*).
     - 3.2.1. Sintaxe Básica do CSS.
     - 3.2.2. Vincular o CSS a uma página HTML.
     - 3.2.3. Identificadores – id e *class*.
     - 3.2.4. Seletores:
       - 3.2.4.1. Utilizando hierarquia dos elementos;
       - 3.2.4.2. Utilizando atributos HTML;
       - 3.2.4.3. Trabalhando com pseudo-classe;
       - 3.2.4.4. Utilizando pseudo-elementos.
     - 3.2.5. Propriedade *background*.
     - 3.2.6. Propriedade texto.
     - 3.2.7. Propriedade de Cor.
     - 3.2.8. Propriedade de Fontes.
     - 3.2.9. Dimensões – Largura e Altura.
     - 3.2.10. Espaçamento – *Padding*.
     - 3.2.11. Espaçamento Externo – *Margin*.
     - 3.2.12. Propriedade bordas:

3.2.12.1. Bordas em Tabelas;
3.2.12.2. Bordas Arredondadas;
3.2.13. Propriedade DIV.
3.2.14. *Box Model*.
3.2.15. Posicionamento:
3.2.15.1. *Float e Clear*.
3.3. Animação.
3.3.1. *Transform*.
3.3.2. *Transition*.
3.3.3. *Animation* e *Keyframes*.
3.4. *Display*.
3.5. Trabalhando com *Media Queries*.

4. JavaScript.
4.1. Introdução ao JavaScript.
4.2. Ferramentas do JavaScript.
4.3. Variáveis e Constantes.
4.4. JavaScript em Navegadores:
4.4.1. *Document Object Model* (DOM) e eventos;
4.4.2. Elementos da linguagem JavaScript.
4.5. Conceitos *Web*.
4.5.1. Introdução HTTP.
4.5.2. Métodos GET e POST.
4.5.3. Formulários e componentes básicos.
4.5.4. Criação de páginas dinâmicas.
4.6. Operadores.
4.6.1. Operadores Matemáticos ou Aritméticos.
4.6.2. Operadores de Atribuição.
4.6.3. Operadores de Comparação.
4.6.4. Operadores Lógicos.
4.6.5. Operadores Bit a Bit.
4.6.6. Operadores Especiais.
4.7. Estrutura de controlo:
4.7.1. Condicional;
4.7.2. Repetitiva;
4.7.3. Recursiva;
4.7.4. Sequencial.
4.8. Funções;
4.9. Eventos.
4.9.1. Eventos HTML.
4.9.2. Tratamento de eventos.
4.10. Objetos JavaScript.

4.10.1. Objetos *Boolean* e *Number* do Javascript
4.10.2. Objeto *String* do Javascript
4.10.3. Objeto *Date* do Javascript
4.10.4. Objeto *Math* do Javascript
4.10.5. Objeto *RegExp* do Javascript;
4.10.6. Hierarquia dos objetos do browser;
4.11. *Array*.
4.11.1. Propriedades e Métodos
4.12. Controlo de estado.
4.12.1. *Cookies*.
4.12.2. Controlo de Sessão.
4.12.3. Estado da página.
4.13. Controlo de Acesso.
4.13.1. Autenticação e autorização.
4.14. Conexão à base de dados.
4.14.1. Componentes de acesso a base de dados.

5. PHP
5.1. Introdução.
5.1.1. Características da linguagem PHP.
5.1.2. História do PHP.
5.1.3. O que é possível fazer com PHP.
5.1.4. Lado cliente e lado servidor.
5.2. Iniciando com PHP.
5.2.1. Instalando o executável do PHP.
5.2.2. Primeiro exemplo na linguagem.
5.2.3. Aplicações *console* no PHP.
5.2.4. Sintaxe básica e construtores.
5.2.5. Utilizando PHP e HTML.
5.2.6. *Short Tags*.
5.2.7. Dinâmica dos arquivos PHP.
5.3. Variáveis e constantes.
5.3.1. Regras de nomeação de variáveis.
5.3.2. Regras para nomeação de constantes.
5.3.3. Comentários no PHP.
5.3.4. Tipos de dados escalares.
5.3.5. Declarando *Arrays*.
5.3.6. *Arrays* Associativos.
5.3.7. *Arrays* multidimensionais.
5.3.8. Inserção de novos itens em *arrays*.
5.4. Operadores.
5.4.1. Operadores Aritméticos.

5.4.2. Operadores de comparação.
5.4.3. Operadores de igualdade e tipo.
5.4.4. Operadores lógicos.
5.4.5. Operador de concatenação.
5.4.6. Tipos de declaração de *String*.

5.5. Condicionais.
5.5.1. *Condicional if*.
5.5.2. Trabalhando com *elseif* e *else*.
5.5.3. *Condicional switch case*.

5.6. Estruturas de repetição.
5.6.1. Laço *while*.
5.6.2. Laço *for*.
5.6.3. Laço *foreach*.
5.6.4. Ler *arrays* com *foreach*.
5.6.5. *Continue* e *Break*.

5.7. Funções.
5.7.1. Funções internas do PHP.

5.8. Erros e exceções.
5.8.1. Tipos de erros.
5.8.2. Tratando exceções.

6. Hospedagem.
6.1. Registo de um *site*.
6.2. Hospedagem de um *site*.
6.3. Planos de hospedagem.
6.4. Domínios e subdomínios.
6.5. Disponibilidade de domínios.
6.6. Registo do domínio.
6.7. Registo do domínio na *Hostnet*.
6.8. Configuração de DNS.
6.9. Ferramentas para *upload* de ficheiro.
6.9.1. FTP e a estrutura de pastas.
6.10. *Links*.
6.11. Gestão da conta.
6.12. Gestão do *Site*.
6.13. Gestão de Aplicações.
6.14. Gestão *Emails*.

UNIDADE FORMATIVA 8: UF381 Segurança de Sistemas de Informação (35 Horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Conhecer a estrutura de um plano de segurança de informação.

- **CA 1.1.** Identificar os elementos do sistema afetos a um plano de segurança de informação.
- **CA 1.2.** Identificar os níveis de segurança de um plano de segurança de Informação.
- **CA 1.3.** Distinguir os sistemas que podem aparecer no plano de segurança de informação.
- **CA 1.4.** Descrever as medidas de segurança a aplicar em cada sistema.
- **CA 1.5.** Associar as permissões de acesso aos níveis de segurança descritos no plano de implementação.

RA 2: Reconhecer os mecanismos de acesso aos servidores lógicos do ponto de vista das especificações de segurança de informação.

- **CA 2.1.** Caraterizar o contexto da utilização dos servidores identificando as possíveis ameaças e consequentes riscos de segurança.
- **CA 2.2.** Projetar as medidas de segurança para proteção de dados.
- **CA 2.3.** Descrever as funções e características dos mecanismos de controlo de acesso lógico.

RA 3: Realizar Testes dos serviços de segurança num servidor.

- **CA 3.1.** Identificar os requisitos de segurança aplicáveis aos servidores.
- **CA 3.2.** Descrever as características de segurança e as funcionalidades de um servidor.
- **CA 3.3.** Identificar as funções dos serviços executados no servidor e os possíveis riscos de segurança.
- **CA 3.4.** Num cenário prático, de realização de teste dos serviços de segurança num servidor:
  - Estabelecer os parâmetros de configuração dos serviços;
  - Testar a compatibilidade de segurança entre o servidor e a aplicação.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA3 relativamente ao CA 3.4.

Conteúdos

1. Segurança de sistema informático.
   - 1.1. Introdução à segurança da informação.
   - 1.2. Conceitos e princípios de segurança da informação.
   - 1.3. A segurança e o ciclo de vida da informação.
   - 1.4. Classificação e controlo dos ativos de informação.
   - 1.5. Aspetos Humanos da Segurança da Informação;
2. Tipos de ataques.
   - 2.1. Atacantes externos e internos.
   - 2.2. Mecanismos de proteção habituais.
   - 2.3. Ameaças, tipos e níveis.
3. Gestão de riscos.
   - 3.1. Processamento da gestão de riscos.
   - 3.2. Alternativas mais frequentemente utilizadas.

REVOGADO

3.3. Metodologias de identificação e análise de riscos.

3.4. Redução do risco. Controlos e medidas de salvaguarda.

4. Plano de implantação.

4.1. Determinação do nível de segurança existente frente ao nível necessário para os requisitos de segurança dos processos de negócio.

4.2. Seleção de medidas de salvaguarda para alcançar os níveis de segurança requeridos.

4.3. Elaboração do plano de implementação das medidas selecionadas.

5. Segurança lógica do sistema.

5.1. Mecanismo do sistema operativo para controlo de acessos: Políticas de utilizadores e palavras-passe.

5.2. Sistemas de arquivos.

5.3. Sistemas de permissões: Diretivas no diretório de utilizadores.

5.4. Listas de controlo de acesso aos arquivos (*Access Control List - ACL*).

6. A Segurança no desenvolvimento de sistemas.

6.1. Arquitetura de sistemas seguros.

6.2. Segurança em Redes de Computadores.

6.3. Controlo de Acesso;

6.4. Registos (*logs*) de atividades do sistema.

7. Chaves.

7.1. Criptografia Simétrica e Assimétrica.

7.2. Envelope Digital.

7.3. Resumos de Mensagem.

7.4. Assinatura Digital.

7.5. Infraestrutura de Chaves Públicas.

8. Técnicas de autenticação para controlo de acesso.

8.1. Processo de Autenticação.

8.2. Meios de autenticação de utilizadores.

8.3. Sistemas de autenticação fracos.

8.4. Sistemas de autenticação fortes.

8.5. Sistemas biométricos de autenticação.

9. Segurança a nível do servidor.

9.1. Protocolos, serviços e portas utilizados no sistema informático.

9.2. Ferramentas de análise de portas e serviços abertos.

9.3. Ferramentas de análise de tráfico.

9.4. Fechar serviços, ferramentas e portas prescindíveis.

9.5. Configuração da utilização de protocolos seguros.

9.6. Atualização de segurança nos sistemas informáticos.

REVOGADO

MÓDULOS FORMATIVOS (MF)

MF1: PROGRAMAÇÃO DE APLICAÇÕES FRONT-END (MF283_5)		
Código: MF283_5	Nível: 5	Duração: 170 horas
Associado à UC283_5: Desenvolver Aplicações Front-end.		

Este MF está subdividido nas seguintes Unidades Formativas:

		Código
■ **UNIDADE FORMATIVA 1:**	Linguagem de Marcação - HTML Avançada	UF382
■ **UNIDADE FORMATIVA 2:**	Linguagem de Formatação - CSS Avançada	UF383
■ **UNIDADE FORMATIVA 3:**	Linguagem de Programação - JavaScript Avançada	UF384
■ **UNIDADE FORMATIVA 4:**	Integração de Sistemas	UF385

UNIDADE FORMATIVA 1: UF382 Linguagem de Marcação - HTML Avançada (30 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA1. Caraterizar as linguagens de marcação.

CA 1.1. Identificar os tipos de linguagem de marcação.

CA 1.2. Identificar as áreas de utilização das linguagens de marcação.

CA 1.3. Classificar as linguagens de marcação.

CA 1.4. Explicar as vantagens da utilização das linguagens de marcação (XML) no armazenamento da informação.

CA 1.5. Diferenciar a linguagem HTML da linguagem *eXtended Markup Language* (XML).

RA2. Codificar uma página web utilizando a linguagem de marcação HTML.

CA 2.1. Identificar as diferentes versões da linguagem HTML.

CA 2.2. Distinguir os *browsers*.

CA 2.3. Explicar a estrutura de um ficheiro HTML.

CA 2.4. Explicar as funcionalidades dos diferentes TAG da linguagem HTML.

CA 2.5. Conhecer a sintaxe da linguagem HTML.

CA 2.6. Numa situação prática de codificação de uma página web, utilizando a linguagem de marcação HTML:

- Identificar a estrutura do documento HML;
- Editar o código, respeitando a sintaxe;
- Testar a página *web* codificada, utilizando *browser* compatível.

RA3. Realizar a hospedagem e a publicação do sistema.

CA 3.1. Caraterizar os diferentes tipos servidores.

CA 3.2. Descrever os diferentes tipos de *softwares* para servidores.

CA 3.3. Identificar a versão e arquitetura dos *softwares* para servidores.

CA 3.4. Definir um domínio.

CA 3.5. Descrever o papel do protocolo TCP/IP na comunicação entre os servidores.

REVOGADO

CA 3.6. Descrever o papel de *Proxy server*.

CA 3.7. Identificar os *softwares* para carregar ficheiros.

CA 3.8. Identificar o sistema de segurança implementado.

CA 3.9. Avaliar os parâmetros do servidor.

CA 3.10. Numa situação prática de hospedagem e publicação do sistema:

- Rececionar, registar, carimbar e encaminhar o documento para despacho;
- Selecionar o plano de hospedagem;
- Verificar a compatibilidade dos *softwares* instalados com o do sistema desenvolvido;
- Solicitar o domínio num provedor;
- Configurar o domínio do sistema;
- Realizar upload dos ficheiros do sistema com recurso a *software* específico;
- Testar as funcionalidades da aplicação;
- Publicar o *site*.

Resultados de Aprendizagem cujas aquisições devem ser complementadas em ambiente real de trabalho

- RA2 relativamente ao CA 2.6.
- RA3 relativamente ao CA 3.10.

Conteúdos

1. A estrutura de um ficheiro HTML Avançada
   1.1. *Web site* e aplicação *web*.
   1.2. Editores e IDE.
   1.3. *Tag* da estrutura do documento HTML
2. HTML e suas APIs.
   2.1. O que são APIs.
   2.2. JavaScript.
3. Controlo de Áudio e Vídeo.
   3.1. Controlo de Áudio e Vídeo.
   3.2. Criando o elemento.
   3.3. Propriedades.
   3.4. Métodos.
   3.5. Eventos.
4. Controlo da Mídia.
   4.1. Drag & Drop.
      4.1.1. Arrastando Elementos.
      4.1.2. Soltando Elementos.
5. Armazenando Dados.
   5.1. Armazenando Dados.
   5.2. *Cookies*.
   5.3. *WebStorage*.
   5.4. *IndexedDB*.

REVOGADO

6. Leitura de Arquivos.
    6.1. Leitura de Arquivos.
    6.2. Carregar Arquivos.
    6.3. Ler Arquivos.
    6.4. Salvar Arquivos.
7. Canvas.
    7.1. Canvas.
    7.2. Nosso primeiro Canvas.
    7.3. Desenhar no Canvas.
    7.4. Animações.
    7.5. Salvar Canvas como Imagem.
8. SVG.
    8.1. SVG.
    8.2. Imagens *Bitmap* e Imagens Vetoriais.
    8.3. Vantagens e Desvantagens.
    8.4. Imagem em SVG.
    8.5. Estilizar SVG com CSS.
9. Pré-processadores HTML.
    9.1. Sintaxe mais simples.
    9.2. Evitando Repetições.
    9.3. Definindo Condições.

REVOGADO

UNIDADE FORMATIVA 2: UF383 Linguagem de Formatação - CSS Avançada (30 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA1. Formatar uma página web utilizando a linguagem de formatação CSS.

CA 1.1. Diferenciar a linguagem de formatação da linguagem de marcação.

CA 1.2. Descrever a estrutura da linguagem de formatação CSS.

CA 1.3. Conhecer a sintaxe da linguagem de formatação CSS.

CA 1.4. Indicar os diferentes tipos de seletores utilizados na linguagem de formatação CSS e suas funções.

CA 1.5. Numa situação prática de formatação de uma página web:

- Definir o estilo de formatação;
- Adicionar estilos CSS (*Inline*, interno e externo);
- Aplicar os efeitos aos diferentes tipos de seletores.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA1 relativamente ao CA 1.5.

Conteúdos

1. *Display*.
   1.1. Propriedade *display*.
   1.2. Propriedade visibility e *display*.
   1.3. Propriedade *list*.
2. Transformações 2D e 3D.
   2.1. Transformações 2D.
   2.2. Transformações 3D.
3. Filtros:
   3.1. *Blur*(px);
   3.2. *Brightness*(%);
   3.3. *Contrast*(%);
   3.4. *Drop-shadows*(posição_horizontal posição_vertical dispersão cor);
   3.5. *Grayscale*(%);
   3.6. *Hue-rotate*(ângulo);
   3.7. *Invert*(%);
   3.8. *Opacity*(%);
   3.9. *Saturate*(%);
   3.10. *Sepia*(%);
   3.11. *CSSgram*.
4. Transições.
   4.1. A importância das transições.
   4.2. Nossa primeira transição.
   4.3. Transições com JavaScript.
   4.4. Curva da Velocidade da Transição.
   4.5. Propriedades de Transição.

REVOGADO

5. Animações.
   5.1. Criando uma Animação.
   5.2. Animações com várias mudanças.
   5.3. Propriedades de Animação.
   5.4. Interagindo com JavaScript.

6. *Media Queries*.
   6.1. *Media Types.*
   6.2. *Media Queries*.

7. Criando *Designs* Responsivos.
   7.1. O que é *Design* Responsivo.
   7.2. *Viewport*.
   7.3. Unidades de Medida Relativas.

8. *Layouts* com *Flex Box*.
   8.1. *Layouts* com *Flex Box*.
   8.2. Propriedades do Contêiner.
   8.3. Propriedades dos Itens.

9. Pré-processadores CSS.
   9.1. O que são Pré-processadores.
   9.2. Funcionalidades

REVOGADO

UNIDADE FORMATIVA 3: UF384 Linguagem de Programação - JavaScript Avançada (60 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA1. Construir uma página *web* dinâmica.

- **CA 1.1.** Diferenciar a linguagem de programação interpretada das outras linguagens de programação para *web*.
- **CA 1.2.** Explicar o funcionamento do servidor *web*.
- **CA 1.3.** Explicar o papel do protocolo "http" na comunicação *web*.
- **CA 1.4.** Descrever a estrutura da linguagem de programação JavaScript.
- **CA 1.5.** Conhecer a sintaxe da linguagem de programação JavaScript.
- **CA 1.6.** Explicar o papel de GET e POST na transmissão dos dados.
- **CA 1.7.** Explicar a função de HTML, CSS, PHP e JavaScript numa página *web*.
- **CA 1.8.** Explicar como JavaScript é inserido numa página *web*.
- **CA 1.9.** Numa situação prática de construção de uma página *web* dinâmica:
  - Definir a interação a aplicar à página *web*;
  - Seleccionar o tipo de animação a aplicar aos objectos;
  - Escrever o código HTML da página *web*;
  - Escrever o código JavaScript para animação, *inline* ou num arquivo separado;
  - Testar o código;
  - Testar a página *web* criada.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA1 relativamente ao CA 1.9.

Conteúdos

1. HTML DOM.
   1.1. Alterar estilos CSS.
   1.2. Eventos.
   1.3. *Nodes* (Elementos).
2. *Window Object.*
   2.1. *Window.Screen* e *Window.Location*.
   2.2. *Window.History* e *Window.Navigator*.
3. *Alert* em Javascript.
   3.1. *Javascript setInterval* e *setTimeout*. Direito à informação.
4. *Javascript Cookies.*
5. *JavaScript Object Notation* (JSON).
6. *Javascript OO*: Pilares da Programação Orientada a Objetos.
   6.1. *Javascript OO*: Conceitos da Programação Orientada a Objetos.
7. AJAX.
8. *JQuery.*
9. Armazenamento de Dados.
   9.1. Armazenar dados no navegador.
   9.2. *LocalStorage.*
   9.3. *SessionStorage*:

RE VOGADO

9.3.1.Banco de Dados;

9.3.2.Inserir Dados;

9.3.3.Pesquisar Dados;

9.3.4.Atualizar Dados;

9.3.5.Remover Dados.

REVOGADO

UNIDADE FORMATIVA 4: UF385 Integração de Sistemas (50 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA1. Caraterizar a tecnologia utilizada no compartilhamento de dados na Web.

CA 1.1. Caraterizar os tipos de integração de sistemas mais utilizados.

CA 1.2. Distinguir os conceitos de integração, interoperabilidade e compatibilidade.

CA 1.3. Identificar os requisitos técnicos utilizados na integração de sistemas.

CA 1.4. Caraterizar as tecnologias utilizadas no desenvolvimento *Web*.

CA 1.5. Indicar a função dos protocolos utilizados na troca de mensagem.

CA 1.6. Diferenciar os protocolos "https" e "http".

CA 1.7. Explicar o papel dos protocolos na integração de sistemas.

CA 1.8. Diferenciar o protocolo *SOAP* do *REST* na integração de sistemas.

CA 1.9. Explicar os processos de comunicação utilizada pela API.

CA 1.10. Caraterizar a arquitetura dos diferentes sistemas.

RA2. Desenvolver uma API (*Aplication Programation Interfaces*) para a integração de sistemas.

CA 2.1. Explicar o papel de *web services* na integração de sistemas.

CA 2.2. Deferenciar *web services* com o protocolo SOAP do *web services* com protocolo REST.

CA 2.3. Descrever a arquitetura de comunicação entre *web services* e a aplicação.

CA 2.4. Explicar a função de *Web Services Description Lunguage* (WSDL).

CA 2.5. Diferenciar as seguintes tecnologias de *Web Services*: JSON, XML, WSDL, SOAP e UDDI.

CA 2.6. Descrever a ontologia do domínio de web semântica.

CA 2.7. Descrever o papel de URI (*Uniform Ressource Locator*) na web semântica.

CA 2.8. Diferenciar URL de URI.

CA 2.9. Descrever o papel de metadados na *web* semântica.

CA 2.10. Numa situação prática de desenvolvimento de uma API, utilizando uma linguagem de integração:

- Identificar a arquitetura dos sistemas existentes;
- Definir a arquitetura da API a desenvolver;
- Definir a estrutura de dados a ser consumido;
- Editar e compilar o programa;
- Realizar *debug* do programa.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA2 relativamente ao CA 2.10.

Conteúdos

1. Introdução às Tecnologias.
   1.1. Linguagem de Compartilhamento de Dados.
2. *Web Services.*
   2.1. Certificados Digitais.
   2.2. SOAP.
      2.2.1. Conceito.

REVOGADO

2.2.2. Serviços.

2.2.3. Ciclo de vida dos serviços.

2.2.4. Migração dos serviços.

2.2.5. Integração entre empresas.

3. Linguagem de marcação.

3.1. Tipos de linguagem.

3.2. Linguagem XML.

3.2.1. Apresentação da linguagem.
3.2.2. Sintaxe.
3.2.3. Definição de tipos de documentos.
3.2.4. Especificações associadas.
3.2.5. Processadores de XML.
3.2.6. XML-RPC.

3.3. *Web Services* REST.

3.3.1. Linguagem JSON.
3.3.2. Construção da camada de *Network*.
3.3.3. *Request* Nativo.
3.3.4. Padrão *RESTful*.
3.3.5. Arquitetura REST.

4. *Web* Semântica.

4.1. Introdução a *Web* semântica.

4.2. *Web Ontology Language* (OWL).

4.3. RDF (*Resource Description Framework*) e *RDF Schema*.

Requisitos básicos do contexto formativo do Módulo

Espaços:

Os espaços onde deve decorrer o contexto formativo devem preencher os requisitos básicos estipulados pelo Sistema de Acreditação das Entidades Formadoras. (cf. Decreto-Lei nº6/2013, de 11 de fevereiro; Boletim Oficial nº9; I Série).

Plano de contingência de prevenção e controlo COVID 19.

Professor / Formador:

- O professor ou formador deve possuir um Certificado de Aptidão Profissional que prove que o mesmo é detentor de formação pedagógica na abordagem por competências, segundo a lei.
- O professor ou formador deve ter a formação técnica relacionada com os conteúdos da Unidade/Módulo Formativo.
- O professor ou formador deve ter experiência profissional mínima de 3 anos nas competências incluídas na Unidade de Competência associada à Unidade/Módulo Formativo.

Requisitos de acesso ao módulo formativo

As condições de acesso ao módulo formativo constam no BO nº 20, I Série, Portaria Conjunta nº 09/2020 de 20 de fevereiro - Estrutura e Organização do Catálogo Nacional de Qualificações e Portaria Conjunta nº 10/2020 de 20 de fevereiro - Quadro Nacional de Qualificações.

REVOGADO

MF2: PROGRAMAÇÃO DE APLICAÇÕES BACK-END		
Código: MF284_5	Nível: **5**	Duração: 200 Horas
Associado à UC284_5: Desenvolver aplicações Back-end.		

Este MF está subdividido nas seguintes Unidades Formativas:

		Código
■ **UNIDADE FORMATIVA 1:**	Programação web do lado do servidor	**UF386**
■ **UNIDADE FORMATIVA 2:**	Estrutura da base de dados relacional	**UF387**
■ **UNIDADE FORMATIVA 3:**	Implementação de Base de Dados	**UF388**
■ **UNIDADE FORMATIVA 4:**	Programação em SQL	**UF389**
■ **UNIDADE FORMATIVA 5:**	Programação Orientada a Objetos avançada	**UF390**

UNIDADE FORMATIVA 1: UF386 Programação web do lado do servidor (50 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA1. Caraterizar diferentes tipos de servidores.

CA 1.1. Descrever a funcionalidade de um servidor;

CA 1.2. Distinguir os tipos de servidores existentes (*Proxy, FTP, DNS, DHCP, Email*, entre outros);

CA 1.3. Listar os principais protocolos de comunicação utilizados na internet;

CA 1.4. Descrever os mecanismos de comunicação entre cliente e o servidor;

CA 1.5. Explicar a infraestrutura tecnológica da internet.

RA2. Programar do lado do servidor (*server side*).

CA 2.1. Identificar as diferentes linguagens de programação web utilizadas na programação do lado do servidor.

CA 2.2. Conhecer as caraterísticas das linguagens de programação *web* utilizadas na programação do lado do servidor.

CA 2.3. Diferenciar as linguagens de programação *web* do lado do servidor das do cliente.

CA 2.4. Diferenciar os tipos de Sistema de Gestão de Base de Dados (SGBD).

CA 2.5. Numa situação prática de programação do lado do servidor:

- Selecionar a linguagem de programação *web* a utilizar;
- Editar o código fonte;
- Compilar o código fonte;
- Depurar erros e voltar a compilar.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA2 relativamente ao CA 2.5.

Conteúdos

1. Apresentação da linguagem
   1.1. Tipos de linguagem de programação para *Back- end.*
   1.2. Arquitetura e funcionamento.

REVOGADO

2. Sintaxe básica.
    2.1. Tipos de dados suportados.
    2.2. Tipos básicos.
    2.3. Cadeias de carateres.
    2.4. Arrays.
    2.5. Constantes.
    2.6. Variáveis.
    2.7. Âmbito.
3. Operadores.
4. Estruturas de controlo.
5. Funções.
    5.1. Parâmetros.
    5.2. Passagem de parâmetros.
    5.3. Retorno de valores.

REVOGADO

UNIDADE FORMATIVA 2: UF387 Estrutura da base de dados relacional (30 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA1. Realizar o levantamento dos requisitos para desenho do Modelo Entidade Relacionamento (MER).

CA 1.1. Identificar as diferentes técnicas utilizadas no levantamento de requisitos.

CA 1.2. Diferenciar as técnicas utilizadas no levantamento de requisitos.

CA 1.3. Numa situação prática:

- Identificar e delimitar o micromundo (área de atuação);
- Identificar e analisar o problema;
- Realizar o levantamento dos requisitos aplicando técnicas adequadas;
- Especificar os requisitos para o desenho do Diagrama Entidade Relacionamento (DER).

RA2. Desenhar o Modelo Entidade Relacionamento (MER).

CA 2.1. Identificar os diferentes modelos de estrutura de dados existentes.

CA 2.2. Descrever a funcionalidade de modelo de 3 camadas.

CA 2.3. Definir entidades, atributos e relação.

CA 2.4. Diferenciar tipos de entidades e atributos.

CA 2.5. Explicar a função de cardinalidade e a participação da entidade no relacionamento.

CA 2.6. Identificar as simbologias utilizadas no desenho do diagrama.

CA 2.7. Explicar a função da normalização da base de dados.

CA 2.8. Diferenciar as diferentes formas de normalização (FN) de uma base de dados.

CA 2.9. Numa situação prática do desenho do MER:

- Determinar as entidades;
- Identificar os relacionamentos;
- Adicionar os atributos;
- Unir as entidades por meio das relações;
- Adicionar a cardinalidade da relação.

Resultados de Aprendizagem cujas aquisições devem ser complementadas em ambiente real de trabalho

- RA1 relativamente ao CA 1.2.
- RA2 relativamente ao CA 2.9.

Conteúdos

1. Conceitos
   - 1.1. Bases de dados.
   - 1.2. Dados e informação.
   - 1.3. Modelo de dados.
2. Arquitetura de uma Base de Dados
   - 2.1. Ficheiros e Bases de Dados.
   - 2.2. Tipos de Base de Dados:

REVOGADO

2.3. Bases de dados relacionais, hierárquico, orientado a objetos.

2.4. Arquitectura de um sistema gestor de base de dados.

3. Modelo relacional.

3.1. Estrutura de dados relacional.

3.2. Regras de integridade do modelo.

3.3. Gestão de dados do modelo relacional.

3.4. Tabelas, registos, campos e chaves.

3.5. Modelos Entidade Relacionamento.

3.6. Modelo de três camadas.

3.6.1. Modelo Conceitual, Lógico e Físico.

4. Normalização.

5. Representação na forma não normalizada.

REVOGADO

UNIDADE FORMATIVA 3: UF388 Implementação de Base de Dados (30 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Interpretar a estrutura de uma base de dados e desenhar o modelo lógico de acordo com o Diagrama Entidade Relacionamento - DER.

CA 1.1. Descrever as vantagens e desvantagens da utilização de base de dados.

CA 1.2. Identificar os componentes do Diagrama Entidade Relacionamento.

CA 1.3. Descrever a função de cada componente do diagrama na implementação do modelo físico.

CA 1.4. Identificar as restrições no diagrama.

CA 1.5. Definir a forma de representação do domínio de um atributo.

CA 1.6. Diferenciar os modelos de bases de dados existentes no mercado.

CA 1.7. Explicar as relações entre diferentes entidades de acordo com o Diagrama de Entidade Relacionamento (DER).

CA 1.8. Numa situação prática de desenho do modelo lógico a partir do Diagrama Entidade Relacionamento de um projeto de base de dados:

- Identificar e interpretar as simbologias utilizadas no DER;
- Identificar as entidades a partir das simbologias;
- Identificar o tipo de relação entre as entidades;
- Definir as regras de integridade;
- Desenhar um dicionário de dados;
- Desenhar o modelo lógico.

RA 2: Criar uma base de dados de acordo com DER.

CA 2.1. Identificar os objetos de uma base de dados.

CA 2.2. Descrever a função dos componentes dos objetos de uma base de dados.

CA 2.3. Diferenciar as linguagens Definição (DDL), Manipulação (DML) e Controlo (DCL).

CA 2.4. Descrever os comandos da Linguagem de Definição dos Dados (DDL).

CA 2.5. Numa situação prática da implementação de uma base de dados:

- Criar uma base de dados utilizando o comando DDL;
- Criar tabelas utilizando comando DDL.

RA 3: Manipular os dados, utilizando linguagem DML.

CA 3.1. Distinguir os tipos de linguagens de SGBD existentes.

CA 3.2. Explicar as funções dos comandos utilizados para manipulação dos dados.

CA 3.3. Explicar a sintaxe e a semântica das instruções de linguagem de manipulação de dados.

CA 3.4. Numa situação prática da manipulação dos dados, utilizando a linguagem DML:

- Identificar as restrições de integridade e consistência da base de dados;
- Escrever a sintaxe da consulta para extrair informação contida na base de dados, com diferentes níveis de complexidade, utilizando a linguagem DML;
- Escrever sintaxe para inserção de dados na base de dados, utilizando a linguagem DML;
- Escrever a sintaxes para atualização e eliminação de dados utilizando linguagem DML;
- Codificar scripts para automatizar as tarefas;
- Redigir os documentos descritivos dos procedimentos realizados de acordo com as normas pré-estabelecidas.

REVOGADO

Resultados de Aprendizagem cujas aquisições devem ser complementadas em ambiente real de trabalho

- RA1 relativamente ao CA 1.8.
- RA2 relativamente ao CA 2.5.
- RA3 relativamente ao CA 3.4.

Conteúdos

1. SGBD's.
   1.1. Domínios de utilização.
   1.2. Arquitetura.
2. Sintaxe da linguagem SQL.
   2.1. Sintaxe da linguagem de definição de dados DDL.
      2.1.1. Criação de bases de dados.
      2.1.2. Criação de tabelas.
      2.1.3. Criação de *Views*.
      2.1.4. Criação de *index*.
   2.2. Sintaxe da Linguagem de Manipulação de dados.
      2.2.1. Manutenção de bases de dados.
      2.2.2. Manipulação dos dados.
      2.2.3. Implementação de segurança e integridade.
   2.3. Permissões de acesso.
3. Gestão de objetos da base de dados.
4. Importação e exportação de dados

REVC

UNIDADE FORMATIVA 4: UF389 Programação em SQL (40 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Caraterizar os componentes da SQL (*Strutured Query Lunguage*).

CA 1.1. Caraterizar as Linguagens de Definição (DDL), Manipulação (DML) e Controlo (DCL) de dados.

CA 1.2. Descrever a função dos comandos pertencentes a cada uma das linguagens.

RA 2: Manipular os dados, utilizando SQL.

CA 2.1. Descrever as sintaxes dos comandos SQL.

CA 2.2. Explicar o papel das classes de funções utilizadas pela SQL.

CA 2.3. Numa situação prática de manipulação de dados, utilizando a SQL:

- Criar e eliminar Base de Dados;
- Criar, atualizar e eliminar tabelas;
- Inserir e atualizar os dados numa tabela;
- Selecionar os dados numa ou mais tabelas.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA2 relativamente ao CA 2.3.

Conteúdos

1. Elementos de programação.
   - 1.1. Conceitos.
   - 1.2. Variáveis.
   - 1.3. Comando de fluxo de execução.
   - 1.4. Transação.
   - 1.5. Tratamento de Erros.
   - 1.6. Tabelas Temporárias.
2. Sintaxe da SQL.
   - 2.1. Sintaxe da Linguagem de Definição de Dados DDL:
     - 2.1.1. Criação de bases de dados;
     - 2.1.2. Criação de tabelas;
     - 2.1.3. Criação de *Views*;
     - 2.1.4. Criação de índex.
   - 2.2. Sintaxe da linguagem de Manipulação de dados:
     - 2.2.1. Manutenção de bases de dados;
     - 2.2.2. Manipulação de dados.
3. Implementação de segurança e integridade.
   - 3.1. Permissões de acesso.
4. *Procedure* e funções analíticas.
   - 4.1. Utilização de transações, descrevendo o mecanismo de bloqueios.
   - 4.2. Nível de isolamento de uma transação.
5. Estruturas de controlo.

EVOGA

UNIDADE FORMATIVA 5: UF390 Programação orientada a objetos avançada (50 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Conhecer as bibliotecas gráficas para o desenho da *interface.*

CA 1.1. Identificar as bibliotecas gráficas existentes no compilador.

CA 1.2. Descrever a forma de utilização das bibliotecas internas e externas do compilador.

CA 1.3. Identificar os elementos gráficos de uma interface e as hierarquias dos componentes gráficos.

RA 2: Desenvolver um programa com *interface* gráfica para conexão à base de dados.

CA 2.1. Descrever os tipos abstratos de dados.

CA 2.2. Explicar a forma de conexão com base de dados relacionais.

CA 2.3. Explicar a forma de utilização de listas, pilhas e matrizes.

CA 2.4. Numa situação prática de desenvolvimento de um programa com *interface* gráfica para conexão à base de dados:

- Importar a biblioteca para *interface* gráfica;
- Importar a biblioteca de acesso a base de dados;
- Definir os tipos de dados abstratos;
- Definir a forma de conexão com a base de dados;
- Utilizar *drivers* para conectar à base de dados;
- Testar a conetividade com a base de dados;
- Compilar o programa;
- Realizar *debug* do programa;
- Realizar teste do programa.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA2 relativamente ao CA 2.8.

Conteúdos

1. Programação Orientada a Objectos Avançado.
   - 1.1. *Simple Graphics Library.*
   - 1.2. *Exceptions.*
   - 1.3. *Containers Generics.*
   - 1.4. *Nested Classes.*
   - 1.5. *Collections Framework*.
   - 1.6. Java I/O.
   - 1.7. Programação Concorrencial.
2. Conceitos Avançados.
   - 2.1. Java e Bases de Dados Relacionais (JDBC).
   - 2.2. Java e programação com componentes.
3. API de coleções.
   - 3.1. Coleções.
     - 3.1.1. A estrutura da API de coleções.
     - 3.1.2. Listas, filas e pilhas.

3.1.3. Conjuntos.

4. *Interface* gráfica de utilizador.
   4.1. O *framework* de *interface* de utilizador.
   4.2. Hierarquia de componentes.
   4.3. Eventos e ações.
   4.4. Janelas e painéis.
   4.5. Menus.
5. Programação em pequenos *devices* (*palm*, *celular*, etc.).
6. Programação em rede (*socket* e protocolos de comunicação).
7. Invocação remota de método.
8. Serialização e persistência de objetos.
   8.1. Serialização de objetos.
   8.2. Arquivos e fluxos**.**

Requisitos básicos do contexto formativo do Módulo

Espaços:

Os espaços onde deve decorrer o contexto formativo devem preencher os requisitos básicos estipulados pelo Sistema de Acreditação das Entidades Formadoras. (cf. Decreto-Lei nº6/2013, de 11 de fevereiro; Boletim Oficial nº9; I Série). Plano de contingência de prevenção e controlo COVID 19.

O Professor / Formador deve:

- Possuir um Certificado de Aptidão Profissional que prove que o mesmo é detentor de formação pedagógica na abordagem por competências, segundo a lei.
- Ter a formação técnica relacionada com os conteúdos do Módulo Formativo.
- Ter experiência profissional mínima de 3 anos, comprovada nas competências incluídas na Unidade de Competência associada ao Módulo Formativo.

Requisitos de acesso ao módulo formativo

As condições de acesso ao módulo formativo constam no BO nº 20, I Série, Portaria Conjunta nº 09/2020 de 20 de fevereiro - Estrutura e Organização do Catálogo Nacional de Qualificações e Portaria Conjunta nº 10/2020 de 20 de fevereiro - Quadro Nacional de Qualificações.

REVOGADO

MF3: PROGRAMAÇÃO DE APLICAÇÕES PARA DESKTOP		
Código: MF285_5	Nível: 5	Duração: 170 Horas
Associado à UC285_5: Desenvolver aplicações para Desktop		

Este MF está subdividido nas seguintes Unidades Formativas:

		Código
■ **UNIDADE FORMATIVA 1:**	Programação estruturada avançada	**UF391**
■ **UNIDADE FORMATIVA 2:**	Programação orientada a objetos avançada	**UF390**
■ **UNIDADE FORMATIVA 3:**	Estrutura da base de dados relacional	**UF387**
■ **UNIDADE FORMATIVA 4:**	Programação em SQL	**UF389**

UNIDADE FORMATIVA 1: UF391 Programação estruturada avançada (50 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA1. Caraterizar o desenvolvimento modular.

CA 1.1. Diferenciar programa e subprograma de uma função e procedimentos.

CA 1.2. Explicar as formas de passagens de parâmetros.

CA 1.3. Explicar como utilizar subprogramas.

CA 1.4. Descrever a forma como utilizar as bibliotecas internas e externas.

RA2. Desenvolver um programa dividindo em subprogramas.

CA 2.1. Descrever os tipos abstratos de dados.

CA 2.2. Explicar a de reutilização de subprogramas ou bibliotecas.

CA 2.3. Explicar como e quando podemos utilizar lista, pilha e matrizes.

CA 2.4. Numa situação prática de desenvolvimento de um programa:

- Importar a biblioteca para interface gráfica;
- Importar a biblioteca de acesso a base de dados;
- Definir os tipos de dados abstratos;
- Definir a forma de conexão;
- Utilizar drivers para conectar a base de dados;
- Testar a conetividade;
- Compilar o programa;
- Realizar *debug* do programa;
- Realizar teste do programa.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA2 relativamente ao CA 2.4

Conteúdos

1. Funções.
   1.1. Definição e invocação.
   1.2. Passagem de Parâmetros.

REVOGADO

1.3. Procedimentos.

2. Organização de um programa em subprogramas.
3. Ponteiros.
4. Sequências de caracteres.
5. Estruturas de dados.
   5.1. Matrizes.
   5.2. Ficheiros.
   5.3. União.
6. A API de coleções.
   6.1. Coleções.
      6.1.1. A estrutura da API de coleções.
      6.1.2. Listas, filas e pilhas.
      6.1.3. Conjuntos.
7. Bibliotecas.
8. GCI.
9. Interface Gráfica.
10. Banco de dados.
11. DLL.
12. Sockets.

REVOGADO

UNIDADE FORMATIVA 2: UF390 Programação orientada a objetos avançada (50 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA1. Conhecer as bibliotecas gráficas para o desenho da *interface*.

CA 1.1. Identificar as bibliotecas gráficas existentes no compilador.

CA 1.2. Descrever a forma de utilização das bibliotecas internas e externas do compilador.

CA 1.3. Identificar os elementos gráficos de uma interface e as hierarquias dos componentes gráficos.

RA2. Desenvolver um programa integrando com interface gráfica e acesso a base de dados.

CA 2.1. Descrever os tipos abstratos de dados.

CA 2.2. Explicar a forma de conexão com base de dados relacionais.

CA 2.3. Explicar a forma de utilização de listas, pilhas e matrizes.

CA 2.4. Numa situação prática de desenvolvimento de um programa com interface gráfica:

- Importar a biblioteca para interface gráfica;
- Importar a biblioteca de acesso a base de dados;
- Definir os tipos de dados abstratos;
- Definir a forma de conexão com a base de dados;
- Utilizar drivers para conectar à base de dados;
- Testar a conetividade com a base de dados;
- Compilar o programa;
- Realizar *debug* do programa;
- Realizar teste do programa.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA2 relativamente ao CA 2.8.

Conteúdos

1. Programação Orientada a Objectos Avançado.
   - 1.1. Simple Graphics Library.
   - 1.2. Exceptions.
   - 1.3. Containers Generics.
   - 1.4. Nested Classes.
   - 1.5. Collections Framework.
   - 1.6. Java I/O.
   - 1.7. Programação Concorrencial.
2. Conceitos Avançados.
   - 2.1. Java e Bases de Dados Relacionais (JDBC).
   - 2.2. Java e programação com componentes.
3. API de coleções.
   - 3.1. Coleções.
     - 3.1.1. A estrutura da API de coleções.

REVOGADO

3.1.2. Listas, filas e pilhas.

3.1.3. Conjuntos.

4. Interface gráfica de utilizador.
   4.1. O framework de interface de utilizador.
   4.2. Hierarquia de componentes.
   4.3. Eventos e ações.
   4.4. Janelas e painéis.
   4.5. Menus.

5. Serialização e persistência de objetos.
   5.1. Serialização de objetos.
   5.2. Arquivos e fluxos.

REVOGADO

UNIDADE FORMATIVA 3: UF387 Estrutura da base de dados relacional (30 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA1. Realizar o levantamento dos requisitos para desenho do Modelo Entidade Relacionamento (MER).

CA 1.1. Identificar as diferentes técnicas utilizadas no levantamento de requisitos.

CA 1.2. Diferenciar as técnicas utilizadas no levantamento de requisitos.

CA 1.3. Numa situação prática:

- Identificar e delimitar o micromundo (área de atuação);
- Identificar e analisar o problema;
- Realizar o levantamento dos requisitos aplicando técnicas adequadas;
- Especificar os requisitos para o desenho do Diagrama Entidade Relacionamento (DER).

RA2. Desenhar o Modelo Entidade Relacionamento (MER).

CA 2.1. Identificar os diferentes modelos de estrutura de dados existentes.

CA 2.2. Descrever a funcionalidade de modelo de 3 camadas.

CA 2.3. Definir entidades, atributos e relação.

CA 2.4. Diferenciar tipos de entidades e atributos.

CA 2.5. Explicar a função de cardinalidade e a participação da entidade no relacionamento.

CA 2.6. Identificar as simbologias utilizadas no desenho do diagrama.

CA 2.7. Explicar a função da normalização da base de dados.

CA 2.8. Diferenciar as diferentes formas de normalização (FN) de uma base de dados.

CA 2.9. Numa situação prática do desenho do MER:

- Determinar as entidades;
- Identificar os relacionamentos;
- Adicionar os atributos;
- Unir as entidades por meio das relações;
- Adicionar a cardinalidade da relação.

Resultados de Aprendizagem cujas aquisições devem ser complementadas em ambiente real de trabalho

- RA1 relativamente ao CA 1.2.
- RA2 relativamente ao CA 2.9.

Conteúdos

1. Conceitos.
   1.1. Bases de dados.
   1.2. Dados e informação.
   1.3. Modelo de dados.
2. Arquitectura de uma base de dados.
   2.1. Ficheiros e bases de dados.
   2.2. Tipos de base de dados.
   2.3. Bases de dados relacionais, hierárquico, orientado a objetos.
   2.4. Arquitectura de um Sistema de Gestão de Base de Dados.
3. Modelo relacional.

3.1. Estrutura de dados relacional.
3.2. Regras de integridade do modelo.
3.3. Gestão de dados do modelo relacional.
3.4. Tabelas, registos, campos e chaves.
3.5. Modelos Entidade Relacionamento.
3.6. Modelo de três camadas.
3.6.1. Modelo Conceitual, Lógico e Físico.

4. Normalização.
5. Representação na forma não normalizada.

REVOGADO

UNIDADE FORMATIVA 4: UF389 Programação em SQL (40 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA1. Caraterizar os componentes da *Strutured Query Lunguage* (SQL).

CA 1.1. Caraterizar as linguagens de Definição (DDL), Manipulação (DML) e Controlo (DCL) de dados.

CA 1.2. Descrever a função dos comandos pertencentes a cada uma das linguagens.

RA2. Manipular os dados utilizando a SQL.

CA 2.1. Descrever as sintaxes dos comandos SQL.

CA 2.2. Explicar o papel das classes de funções utilizadas pela linguagem SQL.

CA 2.3. Numa situação prática de manipulação de dados utilizando a linguagem SQL:

- Utilizar comandos para criar e eliminar Base de Dados;
- Utilizar comandos para criar, atualizar e eliminar tabelas;
- Utilizar comandos para inserir, atualizar os dados numa tabela;
- Utilizar comandos para selecionar os dados numa ou mais tabelas.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA2 relativamente ao CA 2.3.

Conteúdos

1. Elementos de programação.
   - 1.1. Conceitos.
   - 1.2. Variáveis.
   - 1.3. Comando de fluxo de execução.
   - 1.4. Transação.
   - 1.5. Tratamento de Erros.
   - 1.6. Tabelas Temporárias.
2. Sintaxe da linguagem.
   - 2.1. Sintaxe da linguagem de definição de dados DDL:
     - 2.1.1. Criação de bases de dados;
     - 2.1.2. Criação de tabelas;
     - 2.1.3. Criação de Views;
     - 2.1.4. Criação de *index*.
   - 2.2. Sintaxe da linguagem de Manipulação de dados:
     - 2.2.1. Manutenção de bases de dados;
     - 2.2.2. Manipulação de dados.
3. Operadores:
   - 3.1. Operadores relacionais;
   - 3.2. Operadores aritméticos;
   - 3.3. Operadores de concatenação.
4. Implementação de segurança e integridade:
   - 4.1. Permissões de acesso.
5. Funções e Procedimentos.
   - 5.1. Definição de Variáveis e Tipos de Dados.

REVOGADO

5.1.1. Usando variáveis em PL/SQL ou *Store Procedure*.
5.1.2. Como Reconhecer Unidades Léxicas PL/SQL ou *Store Procedure*.
5.1.3. Como Reconhecer Tipos de Dados.
5.1.4. Utilização de Tipos de Dados Escalares.
5.1.5. Como Escrever Instruções Executáveis PL/SQL ou *Store Procedure.*
5.1.6. Blocos Aninhados e Escopo Variável.

5.2. Funções.
5.2.1. Funções de conversão.
5.2.2. Funções matemáticas.
5.2.3. Funções de data.
5.2.4. Funções agregadas.

5.3. PL/SQL ou Store Procedures, Views e Triggers.
5.3.1. PL/SQL ou Stored procedure.
5.3.2. Triggers (gatilhos).

5.4. Utilização de Código SQL em Instruções PL/SQL ou *Store Procedure*.
5.4.1. Revisão de SQL DML.
5.4.2. Recuperar Dados em Códigos PL/SQL.
5.4.3. Manipulação de Dados em PL/SQL.
5.4.4. Utilização de Instruções de Controlo de Transações.

5.5. Estruturas de Programa para Controlar o Fluxo de Execução.
5.5.1. Controlo Condicional:
5.5.1.1. Instruções IF;
5.5.1.2. Instruções CASE.
5.5.2. Controle Interativo:
5.5.2.1. Loops Básicos;
5.5.2.2. Loops WHILE e FOR;
5.5.2.3. Loops Aninhados.

5.6. Utilização de Tipos de Dados Compostos.
5.6.1. Registos Definidos pelo utilizador.
5.6.2. Indexação de Tabelas de Registos.

5.7. Utilização de Cursores e Parâmetros.
5.7.1. Introdução a Cursores Explícitos.
5.7.2. Utilização de Atributos de Cursores Explícitos.
5.7.3. Loops FOR de Cursor.
5.7.4. Cursores com Parâmetros.
5.7.5. Utilização de Cursores para UPDATE.
5.7.6. Utilização de Vários Cursores.

5.8. Tratamento de Exceções.
5.8.1. Como Tratar Exceções.
5.8.2. Como Interceptar Exceções do Oracle Server.
5.8.3. Como Interceptar Exceções Definidas pelo Usuário.
5.8.4. Como Reconhecer o Escopo de Exceções.

5.9. Utilizar e Gerir Procedimentos.
5.9.1. Criar de Procedimentos.
5.9.2. Utilizar Parâmetros em Procedimentos.
5.9.3. Especificar Parâmetros

5.10. Utilizar e Gerir Funções.
5.10.1. Criar Funções.
5.10.2. Utiliza Funções em Instruções SQL.

REVOGADO

5.10.3. Revisão do Dicionário de Dados.
5.10.4. Gerir Funções e Procedimentos Armazenados.
5.10.5. Revisão de Privilégios de Objeto.

Requisitos básicos do contexto formativo do Módulo

Espaços:

Os espaços onde deve decorrer o contexto formativo devem preencher os requisitos básicos estipulados pelo Sistema de Acreditação das Entidades Formadoras. (cf. Decreto-Lei nº6/2013, de 11 de fevereiro; Boletim Oficial nº4; I Série).

Plano de contingência de prevenção e controlo COVID 19.

O Professor / Formador deve:

- Possuir um Certificado de Aptidão Profissional que prove que o mesmo é detentor de formação pedagógica na abordagem por competências, segundo a lei.
- Ter a formação técnica relacionada com os conteúdos do Módulo Formativo.
- Ter experiência profissional comprovada nas competências incluídas na Unidade de Competência associada ao Módulo Formativo.

Requisitos de acesso ao módulo formativo

As condições de acesso ao módulo formativo constam no BO nº 20, I Série, Portaria Conjunta nº 09/2020 de 20 de fevereiro - Estrutura e Organização do Catálogo Nacional de Qualificações. Portaria Conjunta nº 10/2020 de 20 de fevereiro - Quadro Nacional de Qualificações.

REVOGADO

MF4: PROGRAMAÇÃO DE APLICAÇÕES PARA DISPOSITIVOS MÓVEIS		
Código: MF286_5	Nível: 5	Duração: 200 Horas
Associado à UC286_5: Desenvolver aplicações para dispositivos móveis.		

Este MF está subdividido nas seguintes Unidades Formativas:

		Código
■ **UNIDADE FORMATIVA 1:**	Arquitetura de Sistemas Operacionais para Dispositivos Móveis	**UF392**
■ **UNIDADE FORMATIVA 2:**	Programação Mobile Android	**UF393**
■ **UNIDADE FORMATIVA 3:**	Programação Mobile IoS	**UF394**
■ **UNIDADE FORMATIVA 4:**	Programação Mobile Multiplataforma	**UF395**

UNIDADE FORMATIVA 1: UF392 Arquitetura de Sistemas Operacionais para Dispositivos Móveis (30 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA1. Conhecer a arquitetura de funcionamento do Sistema Operativo para dispositivos móveis.

- **CA 1.1.** Identificar os diferentes sistemas operativos (S.O) para dispositivos móveis.
- **CA 1.2.** Diferenciar S.O para dispositivos móveis dos outros S.O.
- **CA 1.3.** Caraterizar a arquitetura do S.O de *Android, Apple* IoS e *Windows Phone.*
- **CA 1.4.** Caraterizar as camadas de *midlleware* para dispositivos móveis.
- **CA 1.5.** Identificar os diferentes sensores existentes nos dispositivos móveis.
- **CA 1.6.** Diferenciar o desenvolvimento nativo do desenvolvimento multiplataforma.
- **CA 1.7.** Diferenciar os tipos de processadores.

RA2. Realizar a proteção de dados nos dispositivos móveis utilizando os recursos do sistema operativo.

- **CA 2.1.** Descrever o sistema de segurança a nível de acesso a dispositivos móveis.
- **CA 2.2.** Indicar o sistema de proteção de dados utilizados nos dispositivos móveis.
- **CA 2.3.** Explicar o nível de segurança no tráfego de dados, entre os dispositivos móveis.
- **CA 2.4.** Explicar o nível de segurança de dados existentes nos dispositivos móveis.
- **CA 2.5.** Explicar as técnicas e procedimentos de segurança utilizadas no armazenamento de dados.
- **CA 2.6.** Descrever a forma de proteção do S.O contra vírus e *malware*.
- **CA 2.7.** Explicar as técnicas de recuperação de dados utilizados nos dispositivos móveis.
- **CA 2.8.** Numa situação prática de proteção de dados num dispositivo controlo:
  - Selecionar o nível de segurança de acesso;
  - Aplicar um código de segurança;
  - Encriptar os dados nos dispositivos;
  - Selecionar uma software de proteção contra vírus e *malware*;
  - Recuperar os dados.

REVOGADO

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA2 relativamente ao CA 2.8.

Conteúdos

1. Aplicações Móveis
   1.1. Introdução a história da mobilidade.
   1.2. Principais sistemas operacionais dos dispositivos móveis.
2. Arquitetura dos sistemas operacionais para dispositivos móveis.
   2.1. Arquitetura do Sistema *Android*.
   2.2. Arquitetura do Sistema *Apple* IOS.
   2.3. Arquitetura do *Windows Phone.*
   2.4. Plataformas de Desenvolvimento
      2.4.1. Desafios no desenvolvimento de aplicativos.
      2.4.2. Importância da usabilidade
      2.4.3. Funcionalidades dos ambientes de desenvolvimento
3. Segurança em Dispositivos Móveis
   3.1. Segurança.
   3.2. Segurança de acesso ao dispositivo.
   3.3. Segurança na transmissão de dados entre dispositivo.
   3.4. Segurança dos dados armazenados no dispositivo
4. Proteção contra vírus e *malwares*
   4.1. Diferenciação entre vírus e *malwares*.
   4.2. Proteção contra perda de dados
5. Desenvolvimento de aplicativos seguros
   5.1. Técnicas e procedimentos de segurança.
   5.2. Armazenamento seguro/criptografia.
   5.3. Funcionalidades de *backup*/restauração

REVOGADO

UNIDADE FORMATIVA 2: UF393 Programação Mobile Android (60 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA1. Conhecer a arquitetura da plataforma *mobile - Android*.

CA 1.1. Descrever a arquitetura da plataforma Android.

CA 1.2. Explicar a função de *Activity* no desenvolvimento *mobile-Android*.

CA 1.3. Descrever as caraterísticas visuais da plataforma Android.

CA 1.4. Identificar os APIs utilizados na interação com os recursos *mobile*.

CA 1.5. Indicar os diferentes formatos de media suportados pela plataforma Android.

CA 1.6. Identificar os diferentes serviços utilizados na troca de mensagem *mobile*.

CA 1.7. Indicar os diferentes recursos existentes nos dispositivos móveis, utilizados na comunicação mobile.

RA2. Explicar o ciclo de vida de uma aplicação *Android.*

CA 2.1. Definir os conceitos sobre ciclo de vida de *Activities*.

CA 2.2. Explicar a interação entre o utilizador e os dispositivos móveis através de *views* e *layouts*.

CA 2.3. Identificar os diferentes elementos de um *layout*.

RA3. Desenvolver uma aplicação *mobile-android*.

CA 3.1. Definir uma aplicação *mobile*.

CA 3.2. Identificar os diferentes tipos de aplicações *mobile*.

CA 3.3. Diferenciar as aplicações mobile das restantes aplicações.

CA 3.4. Descrever a funcionalidade dos aplicativos *mobile*.

CA 3.5. Identificar os critérios utilizados na escolha do tipo de aplicação a desenvolver.

CA 3.6. Numa Situação prática de desenvolvimento de uma aplicação *mobile:*

- Escolher o tipo de aplicação a desenvolver;
- Selecionar o ambiente de desenvolvimento;
- Definir as funcionalidades da aplicação;
- Selecionar os recursos do dispositivo a ser utilizado;
- Escrever o código fonte;
- Compilar o código fonte;
- Realizar *debug* do código fonte;
- Realizar teste da aplicação.

REVOGADO

Resultados de Aprendizagem cujas aquisições devem ser complementadas em ambiente real de trabalho

- RA2 relativamente ao CA 2.8.
- RA3 relativamente ao CA 3.6.

Conteúdos

1. Introdução ao Android
   1.1. Introdução
      1.1.1. Máquina Virtual Dalvik.
      1.1.2. Instalação e Configuração de Ambiente.
      1.1.3. Instalação do Java (JDK).

1.1.4. Instalação do *Android Studio.*

2. Fundamentos de Programação em *Android.*
   2.1. Criação da Primeira Aplicação em Android no *Android Studio*.
3. Instalação SDK's.
   3.1. Criar um Emulador para *Android*.
   3.2. AVD *Manager*.
   3.3. Genymotion.
4. Conceitos básicos de *Android.*
   4.1. Activity, View e o Método setContentView (view).
   4.2. A Classe R.
   4.3. Criação da Interface Visual em XML.
   4.4. Criação da Interface Visual com a API Java.
   4.5. Acesso aos Elementos da Tela.
   4.6. Ciclos de Vida.
5. Modelo, Visão e Controle.
   5.1. O que é modelo, visão e controle.
   5.2. Atividades, o início de tudo.
   5.3. XML e arquivos de manifesto.
   5.4. Como implementar a camada de modelo.
   5.5. Objetos imutáveis.
   5.6. Quais eventos podem ser tratados.
6. Visões
   6.1. Como definir layouts gráficos.
   6.2. Como separar visão de modelo.
   6.3. Como criar layouts em arquivos XML.
   6.4. Como adicionar eventos à aplicação.
   6.5. Objetos Gráficos.
   6.6. Depuração simples: imprimindo eventos.
   6.7. Lidando com botões e *spinners.*
   6.8. Carregando múltiplas atividades em uma aplicação.
   6.9. O padrão *adapter.*
   6.10. Mostrando vários itens com listas.
   6.11. Os diversos *layouts.*
7. Sudoku
   7.1. Refinando *layouts.*
   7.2. Lidar com telas em modo paisagem.
   7.3. Invocar atividades dentro de atividades.
   7.4. Usar temas para melhorar *layouts.*
   7.5. Depurar via a consola de saída.

REVOGADO

7.6. Carregar uma atividade.

8. Gráficos 2D.
   8.1. Representar cores.
   8.2. Desenhar formas simples.
   8.3. Lindar com eventos de baixo nível.
   8.4. Como criar animações simples.
   8.5. Como adicionar um *key-pad* à aplicação.
   8.6. Representando iterações entre atividades.
9. Comunicação
   9.1. Recursos Multimídia.
      9.1.1. Como executar aplicações que contêm músicas.
      9.1.2. Como abrir vídeos em *Android*.
      9.1.3. Quais formatos de músicas são aceitos.
      9.1.4. Quais formatos de vídeo.
      9.1.5. Como otimizar a utilização de recursos.
      9.1.6. O padrão *proxy.*
   9.2. Comunicação entre atividades.
      9.2.1. O que são intenções.
      9.2.2. Como invocar uma atividade a partir de outra.
      9.2.3. Como invocar uma atividade específica.
      9.2.4. Como criar *parsers* em Java.
      9.2.5. Como ler páginas HTML.
      9.2.6. O que são expressões regulares
   9.3. Comunicação via *sockets*
      9.3.1. O que são *sockets*.
      9.3.2. O que são data *object accesses.*
      9.3.3. Como criar servidores de serviços distribuídos.
      9.3.4. Aplicações *multi-threading* simples.
      9.3.5. O padrão de projetos *command.*
      9.3.6. O que é o princípio da inversão de dependências.
   9.4. Manipulando Dados.
      9.4.1. Persistência.
      9.4.2. Como armazenar as opções de preferência de uma aplicação.
      9.4.3. Como tocar músicas em uma atividade *Android*.
      9.4.4. Como armazenar dados usando *Bundles.*
      9.4.5. Como usar o sistema de arquivos do aparelho controlo.
      9.4.6. Como manipular arquivos de texto em Java.
      9.4.7. Como armazenar objetos diretamente em arquivos
   9.5. Bases de Dados
      9.5.1. Introdução à SQL.
      9.5.2. A base de dados SQLite.

REVOGADO

9.5.3. Como criar e manipular uma base de dados em Android.

9.5.4. Como apresentar dados na tela do aparelho celular.

9.5.5. Como criar um log simples de eventos.

9.5.6. Quais são as estruturas de dados mais comuns em Java.

9.6. Provedores de Conteúdo.

9.6.1. O que são provedores de conteúdo.

9.6.2. Como ler dados de outras aplicações.

9.6.3. Como identificar recursos de forma única.

9.6.4. Como deixar dados disponíveis para outras aplicações.

9.6.5. Como estabelecer níveis de proteção para dados.

9.6.6. Como alterar dados de outras aplicações.

9.7. O Aparelho Controlo de Comunicação.

9.7.1. Localização e Sensores.

9.7.2. O que são serviços baseados em localização.

9.7.3. Como usar a API de mapas do Google.

9.7.4. Como sobrepor diferentes visões a um mapa.

9.7.5. Como integrar o mapa aos demais componentes gráficos.

9.7.6. Quais visões podem ser acopladas à mapas.

9.7.7. Como usar serviços de localização sem mapas.

9.8. Integração com Recursos do Aparelho.

9.8.1. Notificação.

9.8.2. GPS.

9.8.3. Câmara.

9.8.4. Broadcast Receiver.

9.8.5. HTTP *Connect.*

9.8.6. Alarm Manager.

9.8.7. Handler.

9.8.8. Mapas.

9.8.9. SMS.

9.8.10. Áudio

9.9. Mapas

9.9.1. Como obter um certificado para usar a API de mapas do google.

9.9.2. Como importar as bibliotecas necessárias à manipulação de mapas.

9.9.3. Como exibir um mapa no aparelho controlo.

9.9.4. Como ler as coordenadas geográficas de um ponto no mapa.

9.9.5. Como obter um endereço a partir de um ponto no mapa.

9.9.6. O padrão *decorator.*

9.10. Limitações do Aparelho

9.10.1. Interfaces gráficas fragmentadas.

9.10.2. O que são *callbacks.*

9.10.3. Como um ecrã se comunica com a outra.

REVOGADO

9.10.4. Exibir múltiplos ecrãs.

9.10.5. Enviar mensagens para um fragmento.

9.11. Bateria, Chamadas e Eventos de Difusão.

9.11.1. Como monitorar a bateria.

9.11.2. Como lidar com chamadas telefónicas.

9.11.3. O que são eventos de difusão.

9.11.4. O que são intenções assíncronas.

9.11.5. O que são intenções pendentes.

9.12. Navegando entre Janelas.

9.12.1. Como projetar estratégias efetivas e práticas de navegação entre janelas.

9.12.2. O que é a barra de ações.

9.12.3. Como implementar navegação horizontal entre fragmentos.

9.12.4. Como voltar à atividade mãe a partir da atividade filha.

9.12.5. Que tipo de opções podemos inserir na barra de ações.

9.13. Princípios da Programação Orientada a Objetos.

9.13.1. Software de Qualidade.

9.13.2. Fatores de Qualidade de Software.

9.13.3. Fatores de qualidade externos e internos.

9.13.4. Anti-Fatores de Qualidade.

9.13.5. A rigidez dificultando mudanças.

9.13.6. A imobilidade dificultando o reúso.

REVOGADO

UNIDADE FORMATIVA 3: UF394 Programação Mobile IoS (60 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA1. Conhecer a arquitetura da plataforma mobile – IoS.

CA 1.1. Descrever a arquitetura da plataforma IoS.

CA 1.2. Identificar as caraterísticas visuais da plataforma IoS.

CA 1.3. Identificar os APIs utilizados na interação com os recursos mobile.

CA 1.4. Identificar os diferentes formatos de media suportados pela plataforma IoS.

CA 1.5. Identificar os diferentes serviços utilizados na troca de mensagem *mobile*.

CA 1.6. Indicar os diferentes recursos existentes nos dispositivos móveis, utilizados na comunicação *mobile*.

RA2. Explicar o ciclo de vida de uma aplicação IoS.

CA 2.1. Definir os conceitos sobre ciclo de vida de *Activities*.

CA 2.2. Explicar a interação entre o utilizador e os dispositivos móveis através de views e layouts.

CA 2.3. Identificar os diferentes elementos de um *layout.*

RA3. Desenvolver uma aplicação *mobile*-IoS.

CA 3.1. Definir uma aplicação mobile.

CA 3.2. Identificar os diferentes tipos de aplicações mobile.

CA 3.3. Diferenciar as aplicações mobile das restantes aplicações.

CA 3.4. Descrever a funcionalidade dos aplicativos mobile.

CA 3.5. Identificar os critérios utilizados na escolha do tipo de aplicação a desenvolver.

CA 3.6. Numa situação prática de desenvolvimento de uma aplicação mobile:

- Escolher o tipo de aplicação a desenvolver;
- Selecionar o ambiente de desenvolvimento;
- Definir as funcionalidades da aplicação;
- Selecionar os recursos do dispositivo a ser utilizado;
- Escrever o código fonte;
- Compilar o código fonte;
- Realizar *debug* do código fonte;
- Realizar teste da aplicação.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA3 relativamente ao CA 3.6

Conteúdos

1. Introdução

1.1. Dispositivos iOS.
1.2. O sistema operativo.
1.3. Como desenvolver para iOS.
1.4. Padrão *Model-View-Controller*.
1.5. Compilador Xcode e *Terminal.*
1.6. Criar e estruturar um projeto;
1.7. *Interface Builder* e *Storyboards.*

2. Sintaxe do Swift

2.1. Introdução ao Swift.

REVOGADO

2.2. Xcode *Playgrounds.*
2.3. Variáveis e constantes.
2.4. Tipos de dados e operadores.
2.5. Decisões.
2.6. Ciclos.
2.7. Coleções.
2.7.1. Arrays.
2.7.2. Dicionários.
2.8. Funções.
2.9. Opcionais.
2.10. Blocos.

3. Programação orientada a objetos em Swift.
3.1. Orientação a objetos.
3.2. Classes.
3.3. Propriedades.
3.4. Métodos.
3.5. Estruturas.
3.6. Enumerações.
3.7. Protocolos e extensões.

4. Introdução à programação em Swift para IoS.

5. Criação de aplicações com interface gráfica.
5.1. Ciclo de vida de uma aplicação.
5.2. Views, View hierarchy e view controllers.
5.3. O storyboard.
5.3.1. Outlets e actions.
5.3.2. Layout adaptativo – auto-layout.
5.4. Navegação.
5.4.1. Navigation Controller.
5.4.2. Table View.
5.4.3. Swipe Table View Cell.
5.4.4. Collection View.
5.4.5. Scroll View.
5.4.6. JSON Parsing.
5.5. Tabelas.
5.6. Coleções.
5.7. Persistência de dados – *User defaults.*
5.8. Notificações.
5.9. Gestão de dependências – Cocoa Pods.

REVOGADO

UNIDADE FORMATIVA 4: UF395 Programação Mobile Multiplataforma (50 horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Planear o desenvolvimento *mobile* multiplataforma.

CA 1.1. Definir o conceito de uma aplicação multiplataforma.

CA 1.2. Diferenciar os três tipos de desenvolvimentos *mobile*: nativo, hibrido e multiplataforma.

CA 1.3. Diferenciar os tipos de frameworks para desenvolvimento mobile multiplataforma.

CA 1.4. Identificar framework que permite desenvolvimento de aplicações mobile hibrido.

CA 1.5. Identificar as vantagens e desvantagens da utilização de *frameworks.*

CA 1.6. Identificar os tipos de Sistemas de Gestão de Base de Dados (SGDB) suportados pelos *frameworks.*

CA 1.7. Explicar o processo de comunicação entre uma aplicação e API externa.

CA 1.8. Explicar a compatibilidade da persistência de dados do *frameworks.*

CA 2.1. Numa situação prática de planeamento de um sistema *mobile* multiplataforma:

- Selecionar a tecnologia;
- Selecionar *framework* a utilizar;
- Selecionar a linguagem de programação para desenvolvimento multiplataforma.

RA 2: Desenvolver aplicação *mobile* multiplataforma.

CA 2.2. Identificar as tecnologias utilizadas no desenvolvimento multiplataforma.

CA 2.3. Identificar os conteúdos multimédia utilizadas no desenvolvimento multiplataforma.

CA 2.4. Identificar a resolução de diferentes dispositivos móveis utilizados para ajustar os ecrãs dos dispositivos;

CA 2.5. Numa situação prática de desenvolvimento mobile multiplataforma:

- Selecionar a tecnologia;
- Selecionar framework;
- Selecionar a linguagem de programação para o desenvolvimento mobile multiplataforma;
- Desenhar a interface;
- Editar o código;
- Realizar debug da aplicação;
- Realizar teste do aplicativo.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA2 relativamente ao CA 2.5.

Conteúdos

1. **Introdução.**
   - 1.1. Plataformas de Desenvolvimento mobile.
   - 1.2. Windows - Preparando o ambiente de desenvolvimento.
   - 1.3. OSX - Preparando o ambiente de desenvolvimento.
2. **Desenvolvimento Web Mobile (HTML5/ CSS3/ Javascript).**
   - 2.1. Recursos do HTML5, CSS3 e Javascript (com JSON) úteis para Criação de Aplicações Móveis.
   - 2.2. Conhecimentos básicos de JavaScript.

2.3. Páginas mobile com Design Responsivo.

3. **Desenvolvimento Web Mobile com Framework Multiplataforma.**
4. **Ligação à base de dados.**

4.1. Base dados local.

5. **Publicação de APP.**

5.1. Publicando seu App na Apple Store.

5.2. Publicando seu App na Google Play.

Requisitos básicos do contexto formativo do Módulo

Espaços:

Os espaços onde deve decorrer o contexto formativo devem preencher os requisitos básicos estipulados pelo Sistema de Acreditação das Entidades Formadoras. (cf. Decreto-Lei nº6/2013, de 11 de fevereiro; Boletim Oficial nº4; I Série).

Professor / Formador:

- O Professor ou formador deve possuir um Certificado de Aptidão Profissional que prove que o mesmo é detentor de formação pedagógica na abordagem por competências, segundo a lei.
- O professor ou formador deve ter a formação técnica relacionada com os conteúdos da Unidade/Módulo Formativo.
- O professor ou formador deve ter experiência profissional de um mínimo de 3 anos nas competências incluídas na Unidade de Competência associada à Unidade/Módulo Formativo.

Requisitos de acesso ao módulo formativo

As condições de acesso ao módulo formativo constam BO nº 20, I Série, Portaria Conjunta nº 09/2020 de 20 de fevereiro – Estrutura e Organização do Catálogo Nacional de Qualificações e Portaria Conjunta nº 10/2020 de 20 de fevereiro – Quadro Nacional de Qualificações.

REVOGADO

MF5: PROGRAMAÇÃO DE APLICAÇÕES PARA IoT		
Código: MF287_5	Nível: 5	Duração: 400 Horas
Associado à UC287_5: Desenvolver aplicações para Internet das Coisas - IoT		

Este MF está subdividido nas seguintes Unidades Formativas:

		Código
■ **UNIDADE FORMATIVA 1:**	Fundamentos de eletrónica	**UF396**
■ **UNIDADE FORMATIVA 2:**	Fundamentos de Rede de Comunicação	**UF397**
■ **UNIDADE FORMATIVA 3:**	Sistemas embarcados e microcontroladores	**UF398**
■ **UNIDADE FORMATIVA 4:**	Inteligência artificial para IoT	**UF399**
■ **UNIDADE FORMATIVA 5:**	Análise de dados e Monitorização dos dispositivos IoT	**UF400**
■ **UNIDADE FORMATIVA 6:**	Programação mobile multiplataforma para IoT	**UF401**

UNIDADE FORMATIVA 1: UF396 Fundamentos de eletrónica (50 Horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Conhecer os conceitos e os componentes básicos da eletrónica.

- **CA 1.1.** Explicar o papel da eletrónica na Internet das Coisas.
- **CA 1.2.** Diferenciar os diferentes sistemas de numeração.
- **CA 1.3.** Distinguir a eletrónica analógica da digital.
- **CA 1.4.** Diferenciar os tipos de corrente elétrica.
- **CA 1.5.** Explicar o sentido da corrente.
- **CA 1.6.** Explicar os sistemas de numeração.
- **CA 1.7.** Identificar as diferentes portas lógicas e suas funções.
- **CA 1.8.** Explicar a função dos principais componentes eletrónicos.
- **CA 1.9.** Explicar as caraterísticas dos diferentes componentes.

RA 2: Realizar medição dos componentes eletrónicos.

- **CA 2.1.** Explicar a teoria de lei de Ohm.
- **CA 2.2.** Distinguir as unidades de grandezas básicas utilizadas na medição elétrica/eletrónica.
- **CA 2.3.** Diferenciar os aparelhos utilizados na medição da grandeza elétricas/eletrónica.
- **CA 2.4.** Indicar a importância da calibração dos equipamentos de medição.
- **CA 2.5.** Numa situação prática de medição de grandezas elétricas/eletrónicos:
  - Selecionar os componentes a serem medidos;
  - Selecionar o aparelho de medição;
  - Calibrar o equipamento de medição em função do parâmetros de medição;
  - Realizar a medição.

VALIDADO

RA 3: Interpretar um esquema do circuito eletrónico.

CA 3.1. Conhecer as simbologias dos componentes eletrónicos.

CA 3.2. Realizar leitura de esquema para identificação dos componentes.

CA 3.3. Identificar os componentes eletrónicos básicos.

CA 3.4. Explicar as caraterísticas dos diferentes componentes.

CA 3.5. Explicar as ligações da representação esquemática.

RA 4: Montar um circuito eletrónico numa *protoboard*.

CA 4.1. Explicar a função de uma *protoboard*.

CA 4.2. Distinguir os diferentes modelos de *protoboard*.

CA 4.3. Conhecer as técnicas de montagem de *protoboard*.

CA 4.4. Explicar a estrutura de uma *protoboard*.

CA 4.5. Descrever o papel de barramento numa *protoboard*.

CA 4.6. Explicar a função de jumpers na montagem de uma *protoboard*.

CA 4.7. Numa situação prática de montagem de um circuito numa *protoboard*:

- Interpretar o esquema de um circuito eletrónico;
- Identificar as áreas disponíveis numa *protoboard*;
- Selecionar os componentes para montagem de *protoboard* de acordo com esquema;
- Montar os componentes de acordo com o esquema fornecido;
- Conectar os componentes utilizando os fios de jumpers;
- Testar o circuito.

Resultados de Aprendizagem cujas aquisições devem ser complementadas em ambiente real de trabalho

- RA2 relativamente ao CA 2.5.
- RA4 relativamente ao CA 4.8.

Conteúdos

1. **Sistemas de Numeração.**
   - 1.1. Numeração decimal.
   - 1.2. Sistema binário de numeração.
   - 1.3. Sistema octal de numeração.
   - 1.4. Sistema hexadecimal de numeração.
2. **Portas e Operações Lógicas.**
   - 2.1. Função e Porta lógica - E "AND".
   - 2.2. Função e Porta Lógica - OU "OR".
   - 2.3. Função e Porta Lógica - NÃO "NOT".
   - 2.4. Função e Porta Lógica - NÃO E, NE ou "NAND" "NO AND".
   - 2.5. Função e Porta Lógica - NÃO OU, "NOU" ou "NOR" "NO OR".
   - 2.6. Função e Porta Lógica - "Exclusive-OR" "XOR".
3. **Codificadores e Decodificadores.**
   - 3.1. Codificador Decimal-Binário.
   - 3.2. Decodificador Binário-Decimal.

3.3. Conhecendo um Mostrador Digital - "LED" - "Display" de 7 Segmentos.

4. Componentes eletrónicos básicos.

5. Protoboard:

5.1. Definição;

5.2. Como funciona uma *protoboard*;

5.3. Modelos de *protoboard*;

5.4. Fios de Jumper.

REVOGADO

UNIDADE FORMATIVA 2: UF397 Fundamentos de rede de comunicação (40 Horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Caraterizar uma rede de comunicação.

- **CA 1.1.** Definir redes locais.
- **CA 1.2.** Distinguir LAN dos outros tipos de redes.
- **CA 1.3.** Identificar e caraterizar os dispositivos básicos de uma rede local.
- **CA 1.4.** Identificar e caraterizar as diferentes topologia da rede.
- **CA 1.5.** Listar as camadas do modelo OSI.
- **CA 1.6.** Explicar a função das diferentes Camada.

RA 2: Caraterizar a arquitetura dos protocolos de comunicação.

- **CA 2.1.** Enumerar os diferentes protocolos de comunicação existente;
- **CA 2.2.** Descrever o papel de protocolos na transmissão de dados;
- **CA 2.3.** Explicar a diferença entre MQTT e o protocolo TCP/IP;
- **CA 2.4.** Explicar o processo de transmissão de dados;
- **CA 2.5.** Explicar a técnica de transmissão de dados sem fio.

RA 3: Caraterizar uma rede de sensores sem fio (RSSF).

- **CA 3.1.** Explicar o conceito de sensor.
- **CA 3.2.** Descrever o papel de sensores na comunicação sem fio.
- **CA 3.3.** Explicar o papel de rede de sensores sem fio na IoT.
- **CA 3.4.** Enumerar as principais vantagens de uma RSSF na comunicação.
- **CA 3.5.** Explicar a função de endereço Media Access Control (MAC) e roteamento na RSSF.
- **CA 3.6.** Numa situação prática da configuração de uma rede de sensores sem fio:
  - Preparar o ambiente Contiki OS;
  - Acessar os periféricos do kit de desenvolvimento;
  - Estabelecer a comunicação entre os nós, utilizando protocolo UDP para rede de Sensores;
  - Configurar parâmetros de RSSF para o Contiki OS;
  - Testar o roteamento para rede de sensores.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA3 relativamente ao CA 3.6.

Conteúdos

1. **Padrões de comunicação para IoT.**
   - 1.1. Protocolos de camada física e enlace na IoT.
   - 1.2. Conetar os dispositivos de internet das coisas à Internet e roteamento em IoT.
   - 1.3. Protocolos de transporte na IoT: TCP e UDP.
   - 1.4. Protocolos de aplicação na IoT.
2. **Plataformas de comunicação para interoperabilidade.**
3. **Fundamentos de Redes de Sensores, aplicações.**
   - 3.1. Características de uma rede de sensores sem fio.

REVOGADO

3.2. Internet das coisas.

3.3. MAC e Roteamento para rede de sensores sem fio.

4. Tecnologias de comunicação sem fio.

4.1. 802.15.4 (ZigBee e 6LoWPAN).

4.2. 802.11 (Wi-Fi).

4.3. 802.15.1 (Bluetooth).

5. Redes de Sensores sem Fio:

5.1. Aplicações.

5.2. Plataformas.

UNIDADE FORMATIVA 3: UF398 Sistemas embarcados e microcontroladores (80 Horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Caraterizar um sistema embarcado.

- **CA 1.1.** Definir um sistema embarcado.
- **CA 1.2.** Indicar a função de um sistema embarcado.
- **CA 1.3.** Explicar o sistema embarcado do Arduíno.
- **CA 1.4.** Definir Android embarcado.
- **CA 1.5.** Identificar as áreas de aplicação de sistemas embarcados.
- **CA 1.6.** Identificar as interfaces dos sistemas embarcados.
- **CA 1.7.** Indicar a forma de comunicação dos sistemas embarcados com o mundo externo.
- **CA 1.8.** Identificar as partes constituinte de um sistema embarcado.
- **CA 1.9.** Diferenciar os tipos de processamentos de sistemas embarcados.
- **CA 1.10.** Indicar a função de um software embarcado.

RA 2: Caraterizar os componentes de um sistema embarcado: sensores, atuadores e dispositivos de E/S.

- **CA 2.1.** Explicar o conceito de sensor.
- **CA 2.2.** Identificar os diferentes tipos de sensores utilizados nos sistemas embarcados.
- **CA 2.3.** Indicar a função dos sensores num sistema embarcado.
- **CA 2.4.** Explicar o conceito de atuadores.
- **CA 2.5.** Indicar a função dos atuadores num sistema embarcado.

RA 3: Instalar e configurar o ambiente de desenvolvimento (IDE).

- **CA 3.1.** Caraterizar o sistema operativo utilizado para desenvolvimento de sistema.
- **CA 3.2.** Distinguir os tipos de microcontroladores.
- **CA 3.3.** Diferenciar firmware dos outros tipos de softwares.
- **CA 3.4.** Descrever a ambiente de desenvolvimento.
- **CA 3.5.** Identificar as diferentes linguagens de programação existente.
- **CA 3.6.** Identificar os diferentes tipos de linguagem de programação utilizadas no ambiente de desenvolvimento.
- **CA 3.7.** Indicar as bibliotecas de desenvolvimentos de sistemas embarcados.
- **CA 3.8.** Numa situação prática de instalação e configuração de um IDE:
  - Selecionar e instalar o IDE de acordo com o projeto;
  - Configurar os parâmetros do IDE;
  - Selecionar a linguagem de programação.

RA 4: Programar Microcontroladores/microprocessadores.

- **CA 4.1.** Diferenciar microcontrolador de microprocessador.
- **CA 4.2.** Explicar a função dos microcontroladores.
- **CA 4.3.** Conhecer os principais componentes da arquitetura de microcontroladores.
- **CA 4.4.** Descrever as funções dos diferentes comandos utilizados na programação.
- **CA 4.5.** Caraterizar as diferentes linguagens de programação utilizadas nos sistemas embarcados.

CA 4.6. Explicar os diferentes tipos de operadores e suas funcionalidades.

CA 4.7. Explicar a papel da função e subfunção num programa.

CA 4.8. Distinguir as diferentes estruturas de controlo.

CA 4.9. Descrever a função das bibliotecas.

CA 4.10. Numa situação prática de programação de microcontrolador:

- Declarar e inicializar variáveis;
- Programar sistemas embarcados baseados em microcontroladores;
- Utilizar estruturas de controlo para manipular LEDs;
- Utilizar funções das bibliotecas para manipular os recursos do microcontroladores;
- Controlar sensores, atuadores e dispositivos de E/S a partir de microcontroladores;
- Compilar o código fonte;
- Realizar Debug;
- Realizar teste da aplicação.

Resultados de Aprendizagem cujas aquisições devem ser complementadas em ambiente real de trabalho

- RA3 relativamente ao CA 3.8.
- RA4 relativamente ao CA 4.10.

Conteúdos

1. **Histórico e evolução dos sistemas embarcados.**
2. **Introdução a Sistemas Embarcados.**
   - 2.1. Problemas Fundamentais em Sistemas Embarcados.
   - 2.2. Aplicações.
   - 2.3. Mercado.
3. **Hardware para Sistemas Embarcados.**
   - 3.1. Unidades de Processamento: ASIC/ASSP, Microprocessadores, Lógica Reconfigurável.
   - 3.2. Memórias.
   - 3.3. Microcontroladores.
     - 3.3.1. Arquitetura de microcontroladores.
     - 3.3.2. Portas de E/S.
     - 3.3.3. Periféricos.
   - 3.4. Interfaces de comunicação.
   - 3.5. Unidades de entrada e saida.
     - 3.5.1. Sensores.
     - 3.5.2. Atuadores.
       - 3.5.2.1. Servomotor.
       - 3.5.2.2. Motor de passos.
       - 3.5.2.3. Motor de corrente contínua.
   - 3.6. Dispositivos de entrada e saída.
     - 3.6.1. LEDs e botões.
     - 3.6.2. Display de 7 segmentos.
     - 3.6.3. Display LCD.

3.6.4.Teclado.

4. Programação de microcontroladores.

4.1. Software para Sistemas Embarcados.

4.2. Linguagens de programação para sistemas embarcados (Assembly, C/C++, JAVA, outros).

4.3. Sistemas de tempo real.

4.4. Introdução a Sistemas Operacionais Embarcados (RTOS).

4.5. Firmware.

4.6. Ambientes de desenvolvimento.

4.7. Controlo de periféricos.

4.8. Controlo de dispositivos de E/S.

4.9. Interação com sensores e atuadores.

4.10. Modulação da largura de pulso (PWM)

UNIDADE FORMATIVA 4: UF399 Inteligência Artificial para IoT (60 Horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Conhecer o conceito de Aprendizagem Máquina (Machine Learning-ML).

CA 1.1. Definir o conceito aprendizagem máquina.

CA 1.2. Descrever os principais tipos de aprendizagem máquina.

CA 1.3. Diferenciar os conceitos de aprendizagem máquina, generalização e overfiting.

CA 1.4. Diferenciar sistemas especialistas de sistemas inteligentes.

CA 1.5. Diferenciar aprendizagem máquina de aprendizagem profunda (Deep learning).

CA 1.6. Indicar as principais tarefas que podem ser realizadas com Aprendizagem Máquina.

CA 1.7. Descrever as técnicas de aprendizagem máquina supervisionada e não supervisionada.

CA 1.8. Explicar a diferença entre Inteligência Artificial (IA) e Aprendizagem Máquina.

RA 2: Realizar treinamento de uma máquina.

CA 2.1. Distinguir os modelos de aprendizagem máquina.

CA 2.2. Indicar o papel dos modelos de aprendizagem na aprendizagem máquina.

CA 2.3. Classificar os algoritmos de aprendizagem máquina.

CA 2.4. Caraterizar os principais algoritmos de aprendizagem máquina.

CA 2.5. Identificar as plataformas para desenvolvimento de aprendizagem máquina.

CA 2.6. Descrever o papel de modelos matemáticos no treinamento de máquinas.

CA 2.7. Diferenciar modelo de algoritmo.

CA 2.8. Descrever as vantagens e desvantagens das diferentes plataformas de desenvolvimento.

CA 2.9. Numa situação prática de treinamento de uma máquina:

- Coletar os dados necessários;
- Preparar os dados;
- Selecionar o modelo de aprendizagem;
- Identificar o algoritmo em função do modelo selecionado;
- Treinar a máquina:
- Avaliar o desempenho da máquina;
- Ajustar os parâmetros, visando melhorar a qualidade e a eficiência do modelo;
- Realizar a predição da máquina.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA3 relativamente ao CA 3.9.

Conteúdos

1. **Conceitos de Aprendizagem Máquina, conjunto de treinamento, generalização, *overfiting* e validação**.
2. **Técnicas de aprendizagem máquina.**
   2.1. Aprendizagem não supervisionada.
   2.2. Aprendizagem supervisionada e não supervisionada.
   2.3. Aprendizagem por reforço.

REVOGADO

3. **Principais tarefas:**
   3.1. Associação;
   3.2. Agrupamento;
   3.3. Classificação;
   3.4. Regressão;
   3.5. Previsão de séries temporais.
4. **Algoritmos de treinamento.**
   4.1. Avaliação de desempenho.
5. **Modelos.**
   5.1. Reconhecimento de Padrões.
   5.2. Teoria de probabilidade.
   5.3. Teoria da decisão.
   5.4. Modelos lineares de funções básicas.
   5.5. Modelos generativos probabilísticos.
   5.6. Modelos discriminativos probabilísticos.

REVOGADO

UNIDADE FORMATIVA 5: UF400 Análise de dados e monitorização dos dispositivos IoT (50 Horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Caraterizar o processo de mineração de dados.

CA 1.1. Identificar as principais técnicas de mineração de dados.

CA 1.2. Explicar as etapas de mineração de dados.

CA 1.3. Diferenciar os métodos de mineração de dados.

RA 2: Analisar os dados coletados dos dispositivos IoT.

CA 2.1. Conhecer as metodologias de análise de dados.

CA 2.2. Identificar os parâmetros utilizados para análise ou medição dos dados.

CA 2.3. Definir a forma de medição dos parâmetros.

CA 2.4. Numa situação prática de análise de dados**:**

- Identificar e selecionar as fontes de dados;
- Selecionar os parâmetros dos dados a serem analisados;
- Coletar os dados para análise;
- Selecionar e aplicar o software adequado para análise de dados;
- Interpretar o modelo de dados produzido;
- Interpretar os resultados.

RA 3: Monitorizar os dispositivos com base nos dados analisados.

CA 3.1. Diferenciar os tipos de informação produzidos pelos dispositivos:

- Telemetria;
- Metadados;
- Estado;
- Comandos;
- Respostas.

CA 3.2. Identificar as causas de monitorização de um dispositivo.

CA 3.3. Distinguir a monitorização local da remota.

CA 3.4. Diferenciar o monitoramento do mundo nativo de nuvem de microserviços dos restantes.

CA 3.5. Distinguir as diferente medições.

CA 3.6. Caraterizar boas métricas.

CA 3.7. Numa situação prática de monitorização de um dispositivo com base nos dados analisados:

- Identificar o objeto sujeito a monitorização (dispositivos, ambiente, entre outros);
- Selecionar os recursos a serem monitorizados;
- Selecionar as ferramentas (software) para o processo de monitorização;
- Configurar os recursos a serem monitorizados;
- Realizar teste de monitorização.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA3 relativamente ao CA 3.7.

REVOGADO

Conteúdos

1. **Introdução à mineração de dados.**
   1.1. Perspetivas sobre processos de extração de conhecimento.
   1.2. Caracterização do processo de mineração de dados.
2. **Etapas operacionais do processo.**
   2.1. Pré-processamento.
   2.2. Mineração de dados.
   2.3. Pós-processamento.
3. **Tarefas primárias.**
   3.1. Classificação.
   3.2. Regressão.
   3.3. Agrupamento.
   3.4. Composição de tarefas.
4. **Métodos de mineração de dados.**
   4.1. Métodos tradicionais.
   4.2. Métodos bio-inspirados.
   4.3. Mineração de Grafos.
5. **Big Data.**

REVOGADO

UNIDADE FORMATIVA 6: UF 401 Programação mobile multiplataforma para IoT (120 Horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Caraterizar a linguagem de programação *mobile* para IoT.

CA 1.1. Conhecer os conceitos de IoT.

CA 1.2. Diferenciar as linguagens de programação para IoT.

CA 1.3. Indicar as plataformas de programação de IoT existentes.

RA 2: Diagnosticar *hardware* e realizar conexão dos dispositivos periféricos.

CA 2.1. Identificar os *hardwares* e os dispositivos utilizados na programação de IoT.

CA 2.2. Identifcar os diferentes sensores e outros recursos dos dispositivos.

CA 2.3. Descrever o funcionamento dos diferentes dispositivos.

CA 2.4. Identificar as portas de conexão dos dispositivos periféricos utilizadas na programação de IoT.

CA 2.5. Identificar os diferentes tipos de cabos utilizados na conexão dos dispositivos.

CA 2.6. Numa situação prática de conexão dos equipamentos:

- Selecionar os periféricos para conexão;
- Selecionar os cabos;
- Selecionar o hardware;
- Conectar os dispositivos periféricos ao processador.

RA 3: Programar IoT utilizando linguagens de programação mobile.

CA 3.1. Indicar as funcões das diferentes bibliotecas utilizadas na programação IoT.

CA 3.2. Numa situação prática de programação de IoT para dispositivos mobile:

- Selecionar a linguagem de programação a ser utilizado;
- Selecionar os componentes ou outras bibliotecas disponibilizadas por terceiros;
- Programar aplicativo mobile para IoT;
- Realizar debug;
- Realizar teste.

Resultados de Aprendizagem cujas aquisições devem ser complementadas em ambiente real de trabalho

- RA 2 relativamente ao CA 2.6.
- RA 3 relativamente ao CA 3.2.

Conteúdos

1. Introdução a Dispositivos Móveis.

1.1. O que são dispositivos móveis.
1.2. Restrições de recursos em dispositivos móveis.
1.3. Tipos de dispositivos móveis.
1.4. Características dos dispositivos móveis.
1.5. Sistemas operacionais para dispositivos móveis.
1.6. Comunicação sem fio em dispositivos móveis.

2. Plataforma de Desenvolvimento.

2.1. Plataformas de desenvolvimento baseadas em camadas.

REVOGADO

2.2. Linguagens de programação aplicadas.
2.3. Características dos ambientes de desenvolvimento.
2.4. *Frameworks* disponíveis.

3. Introdução a Programação para Dispositivos Móveis.
3.1. Componentes visuais.
3.1.1.Saída de dados.
3.1.2.Entrada de dados.
3.1.3. Multímidia.
3.1.4. Webview.
3.1.5.componentes compostos de apresentação.
3.1.6.*Layouts*.
3.1.7.Apresentação de listas personalizadas.
3.2. Comunicação via rede.

4. **Estruturando documentos com (X)HTML**.
4.1. Diferenças entre HTML e XHTML.
4.2. Estrutura ou corpo básicos
4.3. Estruturas semântica da página web.

5. Framework HTML xx.
5.1. Validação de códigos.
5.2. Variáveis e tipos de dados.
5.3. Referentes à linguagem de programação.
5.4. Comandos de decisão, repetição e operadores.
5.5. Funções e métodos.
5.6. Classes e componentes adicionais referente a linguagem de programação.
5.7. Controlos e interface gráfica para o utilizador.

6. Técnicas CSS para experiências visuais ricas:
6.1. Transformação;
6.2. Transição;
6.3. Rotação;
6.4. Movimentação (drag and drop) de elementos.

7. JavaScript.

8. Framework Jquery .

9. Suporte a execução de scripts Javascript em outra thread (web workers).

10. APIs e frameworks para animação em jogos web.

11. Conceito Orientação a objetos para IoT.
11.1. Conceitos de orientação a objetos:
11.1.1. Classes e objetos;
11.1.2. Métodos;
11.1.3. Encapsulamento;
11.1.4. Herança;
11.1.5. Sobrecarga e sobrescrita;
11.1.6. Polimorfismo;
11.1.7. Tratamento de exceções.

REVOGADO

12. Ambientes gráficos:

12.1. AWT;

12.2. Swing.

13. Introdução ao Android

13.1. Máquina Virtual Dalvik.

13.2. Instalação e Configuração de Ambiente.

13.3. Criando um Emulador parndroid.

13.4. Activity, View e o Método setContentView (view).

13.5. A Classe R.

13.6. ListActivity – Exibição de uma Lista de Itens na Tela.

13.7. ArrayAdapter – Preenchimento da Lista.

13.8. SimpleCursorAdapter – Exibição dos Contatos de Agenda em uma Lista.

13.9. Criação de um ListAdapter Customizado com Imagens.

13.10. MapaActivity.

13.11. Intent .

13.12. Interface .

13.13. Persistência de Dados.

13.14. HTTP – WEB – JSON/GSON.

14. Integração com Recursos do Aparelho.

15. **Persistência de dados.**

15.1. Conceito de persistência e banco de dados.

15.2. Persistindo dados em arquivos.

16. Serialização de dados.

16.1. Operações básicas manipulação de dados.

16.2. Banco de dados embutido em aplicações web.

Requisitos básicos do contexto formativo do Módulo

Espaços:

Os espaços onde deve decorrer o contexto formativo devem preencher os requisitos básicos estipulados pelo Sistema de Acreditação das Entidades Formadoras. (cf. Decreto-Lei nº6/2013, de 11 de fevereiro; Boletim Oficial nº4; I Série).

Professor / Formador:

- O Professor ou formador deve possuir um Certificado de Aptidão Profissional que prove que o mesmo é detentor de formação pedagógica na abordagem por competências, segundo a lei.
- O professor ou formador deve ter a formação técnica relacionada com os conteúdos da Unidade/Módulo Formativo.
- O professor ou formador deve ter experiência profissional de um mínimo de 3 anos nas competências incluídas na Unidade de Competência associada à Unidade/Módulo Formativo.

Requisitos de acesso ao módulo formativo

As condições de acesso ao módulo formativo constam BO nº 20, I Série, Portaria Conjunta nº 09/2020 de 20 de fevereiro – Estrutura e Organização do Catálogo Nacional de Qualificações e Portaria Conjunta nº 10/2020 de 20 de fevereiro – Quadro Nacional de Qualificações.

REVOGADO

MF6: PROGRAMAÇÃO DE APLICAÇÕES PARA JOGOS		
Código: **MF288_5**	Nível: **5**	Duração: **500 Horas**
Associado à UC288_5: Desenvolver aplicações para Jogos		

Este MF está subdividido nas seguintes Unidades Formativas:

		Código
■ **UNIDADE FORMATIVA 1:**	Guionismo de jogos digitais.	**UF402**
■ **UNIDADE FORMATIVA 2:**	Desenho de Jogos	**UF403**
■ **UNIDADE FORMATIVA 3:**	Modelação e animação 2D e 3D	**UF404**
■ **UNIDADE FORMATIVA 4:**	Áudio para Jogos	**UF405**
■ **UNIDADE FORMATIVA 5:**	Inteligência artificial aplicada a jogos	**UF406**
■ **UNIDADE FORMATIVA 6:**	Motores de Jogos	**UF407**
■ **UNIDADE FORMATIVA 7:**	Programação para Jogos	**UF408**

UNIDADE FORMATIVA 1: UF402 Guionismo de jogos digitais (80 Horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Interpretar um Guião

CA 1.1. Descrever a estrutura e os elementos básicos de um guião.

CA 1.2. Descrever os tipos de diálogos e as narrativas de um guião.

CA 1.3. Diferenciar a narrativa áudiovisual tradicional da narrativa áudiovisual nos games.

CA 1.4. Descrever a estratégias narrativas usadas nos jogos - The Last of Us, Horizon Zero Dawn e GTA V.

CA 1.5. Explicar como histórias podem potencializar jogos que não precisam de histórias: (Mortal Kombat, Call of Duty, FIFA, NBA 2K).

CA 1.6. Diferenciar o guião literário do guião técnico.

CA 1.7. Numa situação prática de interpretação de um guião:

- Identificar as personagens;
- Identificar e descrever as funções das personagens;
- Identificar o argumento;
- Identificar as cenas, as sequências e os subenredos;
- Enquadrar as cenas no tempo e no espaço.

RA 2: Definir o plano de realização do jogo tendo em conta o Guião

CA 2.1. Identificar os diferentes tipos de planos de imagem.

CA 2.2. Diferenciar ideia de *storyline*.

CA 2.3. Descrever *storytelling* nos *eSports* - Criação de narrativas para competições.

CA 2.4. Identificar os tipos de movimentos da câmara.

CA 2.5. Caraterizar a ambiente captação de imagem (condições do espaço, visita técnica).

CA 2.6. Distinguir as técnicas de filmagem.

REVOGADO

CA 2.7. Conhecer as técnicas de posicionamentos e ângulos da câmara.

CA 2.8. Numa situação prática de definição do plano de realização:

- Interpretar o guião;
- Definir o objetivo da produção;
- Realizar a decupagem do guião;
- Elaborar um storyboard a partir do guião;
- Definir plano de enquadramento da câmara.

Resultados de Aprendizagem cujas aquisições devem ser complementadas em ambiente real de trabalho

- RA1 relativamente ao CA 1.11.
- RA 2 relativamente ao CA 2.8.

Conteúdos

1. **Noções básicas de Pré-Produção:**
   - 1.1. Storyline;
   - 1.2. Sinopse;
   - 1.3. Pitching;
   - 1.4. Argumento;
   - 1.5. Repérage;
   - 1.6. Storyboarding;
   - 1.7. Animatic.
2. **Técnicas de comunicação escrita e visual.**
   - 2.1. Produtos áudiovisuais.
3. **Guião.**
   - 3.1. Tipos de Guião:
     - 3.1.1. Guião literário;
     - 3.1.2. Guião técnico;
     - 3.1.3. Guião para jogos.
4. **Narrativa.**
   - 4.1. Técnicas narrativas.
   - 4.2. Análise de narrativas.
   - 4.3. Narrativas tradicionais x interativas.
   - 4.4. Tipos de narrativa e suas características.
   - 4.5. Narrativa embutida x narrativa emergente.
   - 4.6. Narrativas em jogos.
     - 4.6.1. Apreciação:
       - 4.6.1.1. Adventures em texto;
       - 4.6.1.2. Livros-texto (“escolha sua aventura”);
       - 4.6.1.3. Aventuras gráficas aponta-e-clica;
       - 4.6.1.4. RPGs computacionais;
       - 4.6.1.5. “Interactive novels”.
   - 4.7. Processo de criação de narrativas.

REVOGADO

4.8. Introdução ao Design de Narrativas.

4.9. Narrativas interativas lineares.

4.10. Narrativas interativas ramificadas.

4.11. Narrativas interativas de múltiplos finais.

4.12. Estrutura do discurso narrativa.

4.13. Elementos da narrativa.

4.14. Fundamentos e forma da escrita do argumento.

4.14.1. O argumento a escrita e as imagens.

4.14.2. Noções básicas de Argumento e Guião.

4.15. Tipos de personagens.

4.15.1. Análise de personagens.

4.15.2. Processo de criação de personagens.

4.15.3. Criação de quests: associando personagens, mecânica e narrativa.

5. Storyboard.

5.1. Organização dos elementos no ecrã.

5.2. Informação escrita.

5.3. Continuidade.

5.4. Composição do espaço.

5.5. Balanceamento e teoria do flow.

6. Análise do *gameplay*.

REVOGADO

UNIDADE FORMATIVA 2: UF403 Desenho de jogos (40 Horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Conhecer os componentes e as técnicas de desenvolvimento de Jogos.

- **CA 1.1.** Identificar os componentes de um jogo digital.
- **CA 1.2.** Classificar os jogos digitais.
- **CA 1.3.** Descrever as fases de desenvolvimento de um jogo.
- **CA 1.4.** Explicar as técnicas utilizadas para desenho de jogos.

RA 2: Desenhar diagramas UML para o desenvolvimento de um jogo.

- **CA 2.1.** Indicar os padrões de desenho de jogos.
- **CA 2.2.** Conhecer as etapas de engenharia de software.
- **CA 2.3.** Descrever o processo de desenvolvimento de jogos;
- **CA 2.4.** Identificar as metodologias de desenvolvimento de software utilizadas no desenvolvimento de jogos.
- **CA 2.5.** Identificar os requisitos funcionais e não funcionais para desenvolvimento de jogos.
- **CA 2.6.** Numa situação prática de desenho de diagrams UML para desenvolvimento de jogos:
  - Definir os requisitos funcionais e não funcionais do jogo;
  - Desenhar os diagramas UML do projeto de jogo.

RA 3: Caraterizar e descrever a mecânica dos jogos.

- **CA 3.1.** Definir a mecânica de jogos.
- **CA 3.2.** Identificar os elementos da mecânica de jogos.
- **CA 3.3.** Explicar a função da mecânica no desenvolvimento de jogos.
- **CA 3.4.** Descrever os diferentes níveis de mecânica de jogos.
- **CA 3.5.** Descrever o ciclo temporal dos níveis.
- **CA 3.6.** Conhecer os sistemas de coordenadas aplicadas no desenvolvimento de jogos.
- **CA 3.7.** Identificar as grandezas físicas utilizadas no desenvolvimento de jogos.
- **CA 3.8.** Definir colisão.
- **CA 3.9.** Explicar os algoritmos de deteção de colisão.
- **CA 3.10.** Numa situação prática de mecânica de jogos:
  - Interpretar roteiro;
  - Identificar as personagens da narrativa;
  - Identificar a mecânica associada a personagem;
  - Definir sistemas de coordenadas do objeto;
  - Aplicar a mecânica no objeto de acordo o guião.

RA 4: Desenhar um jogo digital.

- **CA 4.1.** Numa situação prática de desenho de um jogo digital:
  - Interpretar roteiro;
  - Identificar as personagens da narrativa;
  - Identificar a mecânica associada a personagem;

REVOGADO

- Desenhar o protótipo do jogo utilizando os diagramas UML desenhado, aplicado mecânica dos jogos;
- Realizar teste de qualidade.

Resultados de Aprendizagem cujas aquisições devem ser complementadas em ambiente real de trabalho

- RA2 relativamente ao CA 2.6.
- RA3 relativamente ao CA 3.10.
- RA4 relativamente ao CA 4.1.

Conteúdos

1. **Fundamentos de jogos.**
2. **Elementos de um jogo.**
3. **Análise sistemática de requisitos de jogos digitais.**
   - 3.1. Introdução a requisitos de jogos digitais.
   - 3.2. Fluxo de trabalho.
   - 3.3. Tipos de requisitos: Funcionais e Não Funcionais.
   - 3.4. Conceito de casos de uso e atores.
   - 3.5. Diagrama da UML para análise de requisitos.
   - 3.6. Detalhamento do caso de uso.
4. **Definição e elaboração das mecânicas dos jogos.**
   - 4.1. Grandezas de movimento:
     - 4.1.1. Trajetória;
     - 4.1.2.Deslocamento;
     - 4.1.3.Velocidade;
     - 4.1.4.Aceleração;
     - 4.1.5.Atrito.
   - 4.2. Sistemas de colisão.
     - 4.2.1.Algoritmos de deteção de colisão
   - 4.3. Movimento de projéteis:
     - 4.3.1.Leis de Newton;
     - 4.3.2.Força;
     - 4.3.3.Gravidade.
5. **Sistemas baseados nas regras definidas no projeto do jogo.**
6. **Elaboração de fases.**
7. **Balanceamento de jogos.**
8. **Prototipagem e testes.**
   - 8.1. Ferramenta de Prototipagem de jogos.
9. **Documentos de game design.**

REVOGADO

UNIDADE FORMATIVA 3: UF404 Modelação e Animação 2D e 3D (60 Horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Caraterizar um espaço 2D e 3 D.

CA 1.1. Definir o conceito de modelação 3D.

CA 1.2. Identificar os operadores básicos utilizados na modelação 3 D.

CA 1.3. Descrever um espaço 3D.

RA 2: Realizar modelação 2D e 3D.

CA 2.1. Indicar os métodos de modelação 3D.

CA 2.2. Indicar as principais técnicas de modelação 3D.

CA 2.3. Indicar os métodos de texturização de objetos.

CA 2.4. Numa situação prática de modelação 3 D:

- Selecionar Software a utilizar;
- Definir o sistema de coordenada no espaço 3 D;
- Selecionar ou conceber o objeto a ser modelado;
- Aplicar textura;
- Realizar Rigging.

RA 3: Realizar animação 2D e 3D.

CA 3.1. Identificar as técnicas de animação 3D.

CA 3.2. Identificar os principais softwares utilizados na animação 3D.

CA 3.3. Diferenciar animação 3D das outras técnicas de animação.

CA 3.4. Numa situação prática de realização de uma animação 3D:

- Selecionar o software para animação;
- Selecionar o objeto a ser animado;
- Animar o objeto aplicando os métodos de animação 3D.

Resultados de Aprendizagem cujas aquisições devem ser complementadas em ambiente real de trabalho

- RA2 relativamente ao CA 2.4.
- RA3 relativamente ao CA 3.4.

Conteúdos

1. Introdução à modelagem 2D e 3D.

1.1. Técnicas de Modelagem 2D e 3D.

1.2. Operadores Básicos.

1.2.1. Espaço 2D e 3D – Coordenadas XYZ e UV.

1.2.2. Camadas.

1.2.3. Conceitos básicos de computação gráfica e espaço 3D.

1.2.3.1. Vértice, Aresta, Linhas, Polígonos.

1.2.3.2. Face, Malha, Câmera, Modelos de iluminação.

1.2.4. Hierarquias, nodes, atributos e conexões.

1.3. Introdução aos tipos de objetos.

1.3.1. Nurbs - curvas e superfícies.

1.3.2. Polígonos.

1.3.3. Subdivision Surface.

2. Software de modelagem 3D aplicado à criação de jogos digitais.

2.1. Malha e objetos.

2.2. Modos.

2.3. Rotação.

2.4. Movimentação.

2.5. Escalonamento.

2.6. Câmaras.

2.7. Seleção e agrupamento de objetos.

2.8. Extrusão.

2.9. Bevel.

2.10. Subdivisão.

2.11. Materiais e textura.

2.12. Modelagem poligonal.

2.13. Modelagem de curvas.

2.14. Sculping Modeling (formas orgânicas).

3. Colorização em modelos 2D e 3D:

3.1. Materias, texturas, shaders, mapeamento e texturização UV;

3.2. Mapeamento de normais, bump mapping, Baking, Pintura 3D.

4. História e princípios da animação gráfica.

4.1. Tipos de animação gráfica.

5. Elementos básicos de uma animação gráfica:

5.1. Palco, cena, camadas, linha do tempo(timeline);

5.2. Conceito de quadros por segundo (frames),Simbolos,Biblioteca;

5.3. Movimento.

5.4. Interpolação e Mascara.

5.5. Pontos-chave de animação e mudança de poses.

6. Criação de animações 2D e 3D.

6.1. Envolvendo personagens, textos, imagens e sons.

6.2. Movimentação de personagens e objetos.

6.3. Roteiro e Storyboards para animações.

6.4. Animatic.

6.5. Planos e enquadramento de câmara.

6.6. Ferramentas gráficas para a criação de animações.

6.7. Conceito de Sprites:

6.7.1.Posicionamento,

6.7.2.Espelhamento, visibilidade, animação corrente, atualização.

7. Animação em camadas.

7.1. Parallax

REVOGADO

7.2. Scrolling aplicado à animação gráfica.

7.3. Pós-produção de animações.

8. Animação de modelos 2D e 3D:

8.1. Conceitos de animação, Esqueleto, keyframe, interpolação.

8.2. Animação corporal e facial.

8.3. Dinâmica de corpos macios.

8.4. Animação de tecidos.

REVOGADO

UNIDADE FORMATIVA 4: UF405 Áudio para jogos (40 Horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Captar, manipular e gerar elementos de áudio.

CA 1.1. Explicar a importancia de áudio num jogo.

CA 1.2. Elencar os parámetros do som.

CA 1.3. Descrever papel de amplitude, fase, comprimento de onde e frequência no som.

CA 1.4. Defenir as medidas de grandeza para áudio.

CA 1.5. Elencar as técnicas de sonarização de jogos digitais.

CA 1.6. Distinguir as ferramentas (Software) utilizado na captura e manipulação de som.

CA 1.7. Diferenciar um designer de som para jogos de outros produtores de som.

CA 1.8. Numa situação prática de captura e manipulação de som para jogos digitais:

- Criar a trilha sonora para o jogo;
- Criar os efeitos sonoros para o jogo;
- Introduzir o áudio criado no jogo.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA1 relativamente ao CA 1.8.

Conteúdos

1. Fundamentos do som e seus parâmetros.

1.1. Amplitude, freqüência, fase e comprimento de onda.

1.2. Introdução a decibéis, dBm, dBu, dBw.

2. Processo analógico e digital de sinais.

3. Técnicas de sonorização de jogos digitais.

3.1. História da composição de som e de música em jogos digitais.

3.2. O som e a trilha sonora utilizada como ferramenta narrativa em jogos digitais.

3.3. Teoria e técnicas básicas de composição de trilha sonora musical.

3.4. Análise sonora e musical da sonorização de jogos digitais.

3.5. Construção e desenvolvimento temático e formal na composição musical.

3.6. Buffers de som em jogos digitais.

3.7. Criação de trilhas, temas e efeitos especiais para jogos digitais.

3.8. Teorias e técnicas para a sonorização de jogos digitais.

3.9. Análise sonora e musical.

4. Softwares para gravação de áudio digital.

4.1. Gravação de áudio digital multipista.

4.2. Tratamento do áudio prégravado: normalização, fades, volume, redução de ruídos, equalização, compressão da dinâmica, reverberação, delay.

4.3. Efeitos sonoros.

4.4. Arquivos de áudio.

4.5. Mixagem.

4.6. Masterização.

4.7. Locução e dublagem.

REVOGADO

UNIDADE FORMATIVA 5: UF406 Inteligência artificial aplicada a jogos (80 horas)

RA 1: Conhecer os conceitos de Inteligência Artifical.

CA 1.1. Definir o conceito de Inteligência Artificial.

CA 1.2. Conhecer os principais campos da inteligência artificial e suas aplicações.

CA 1.3. Indicar as técnicas e os algoritmos de Inteligência Artificial (IA) aplicadas em jogos.

CA 1.4. Diferenciar sistemas especialistas de sistemas inteligentes.

CA 1.5. Conhecer as principais tarefas realizadas na Aprendizagem Máquina.

CA 1.6. Diferenciar as técnicas supervisionadas e não supervisionadas utilizadas na resolução de problemas de jogos.

CA 1.7. Diferenciar IA e ML (*Maschine Learning*).

CA 1.8. Indicar a função de uma máquina de estado nos jogos digitais.

CA 1.9. Explicar a função dos agentes inteligentes nos jogos digitais.

CA 1.10. Indicar a função da lógica de Fuzzy no desenvolvimento de jogos.

CA 1.11. Explicar como a inteligência artificial pode ser usada na soluções de problemas.

RA 2: Realizar treinamento da máquina

CA 2.1. Explicar o processo da moldagem de personagem/objeto a partir da Inteligência artificial.

CA 2.2. Identificar as plataformas de desenvolvimento de aprendizagem máquina.

CA 2.3. Estabelecer a diferença entre modelo e algoritmo.

CA 2.4. Indicar a função dos algoritmos na manipulação das ações de jogos.

CA 2.5. Explicar a importância de Inteligência Artificial (IA) na navegação tridimensional da camara num jogo.

CA 2.6. Indicar as vantagens e as desvantagens das diferentes plataformas de desenvolvimento de jogos.

CA 2.7. Numa situação prática de treinamento de uma máquina:

- Coletar e preparar os dados necessários para o treinamento de uma máquina;
- Selecionar o algoritmo em função dos agentes;
- Aplicar o algoritmo para treinar a máquina;
- Realizar teste da máquina;
- Ajustar os parâmetros, caso necessário, para melhorar a qualidade e a eficiência da máquina;
- Realizar a predição da máquina.

Resultado de Aprendizagem cuja aquisição deve ser complementada em ambiente real de trabalho

- RA 2 relativamente ao CA 2.7.

Conteúdos

1. Introdução a Inteligência Artificial (IA).

1.1. História e evolução da inteligência artificial.

1.2. Conceitos básicos de inteligência artificial.

1.3. Paradigmas de inteligência artificial.

1.4. Sistemas inteligentes.

1.5. Áreas de aplicação da inteligência artificial.

1.6. IA tradicional X IA em Jogos.

2. Sistemas Baseados em Regras.

REVOGADO

2.1. Estrutura de um sistema baseado em regras.

2.2. Base de conhecimento.

2.3. Motor de inferencia.

2.4. Representação do conhecimento.

2.5. Aquisição do conhecimento.

2.6. Sistemas baseados em regras aplicados a jogos.

3. Árvores de decisão.

3.1. Introdução a árvores de decisão.

3.2. Representação de uma árvore de decisão.

3.3. Árvore de decisão aplicada a jogos.

4. Máquinas de Estados Finitos.

4.1. Estados e regras de transição.

4.2. Funcionamento de uma máquina de estados finita.

4.3. Máquina de estados finita com inteligência artificial em jogos.

5. Sistemas Multiagentes.

5.1. Tipos de Agentes.

5.2. Comportamentos dos Agentes.

5.3. Agentes inteligentes em Jogos.

6. Algoritmos de busca.

6.1. Busca cega.

6.2. Busca heurística.

6.3. Busca condicionada em jogos.

7. Aplicações de Inteligência artificial em jogos digitais.

REVOGADO

UNIDADE FORMATIVA 6: UF407 Motores de jogos (80 Horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Caraterizar os motores de Jogos.

CA 1.1. Definir um motor de jogo.

CA 1.2. Distinguir os motores de jogos licenciados dos motores gratuitos.

CA 1.3. Enumerar as vantagens e as desvantagens dos motores de jogos.

CA 1.4. Distinguir os diferentes tipos de motores de jogos.

RA 2: Realizar a conexão dos dispositivos periféricos utilizados nos jogos.

CA 2.1. Identificar os hardwares e os dispositivos utilizados na programação de jogos.

CA 2.2. Indicar a função das placas gráficas e de som na programação de jogos.

CA 2.3. Identificar as portas de conexão dos dispositivos periféricos utilizadas na programação de jogos.

CA 2.4. Identificar os diferentes tipos de cabos utilizados na conexão dos dispositivos.

CA 2.5. Numa situação prática de conexão dos dispositivos para um jogo:

- Selecionar os periféricos para conexão;
- Selecionar os cabos;
- Selecionar o hardware;
- Conectar os dispositivos periféricos ao processador.

RA 3: Programar jogos utilizando motores de jogos.

CA 3.1. Indicar a funcão de motores de jogos.

CA 3.2. Explicar como os diversos componentes de um motor de jogo 3D podem ser usados no desenvolvimento de diversos tipos de jogos.

CA 3.3. Indicar as funcões das diferentes bibliotecas utilizadas pelos motores de jogos.

CA 3.4. Diferenciar os motores de jogos gráficos de motores físicos.

CA 3.5. Explicar como manipular e controlar modelos 3D animados dentro do motor.

CA 3.6. Numa situação prática de programação de jogos:

- Interpretar o guião;
- Desenhar o storyboard;
- Definir a mecânica de jogos;
- Definir a animação;
- Selecionar o motor de jogos a ser utilizado;
- Identificar os recursos dos motores de jogos.
- Verificar a compatibilidade dos motores de jogos com as plataformas.
- Selecionar os componentes ou outras bibliotecas disponibilizadas por terceiros;
- Aplicar componentes de inteligência artificial e física do motor 3D num jogo;
- Implementar algoritmos em linguagem script do motor, para programação da jogabilidade;
- Realizar teste do jogo.

REVOGADO

Resultados de Aprendizagem cujas aquisições devem ser complementadas em ambiente real de trabalho

- RA 2 relativamente ao CA 2.5.
- RA 3 relativamente ao CA 3.5.

Conteúdos

1. Introdução a motor de jogo.

1.1. Definição de motor de jogo.

1.2. Componentes de um motor de jogo.

1.3. Tipos, gêneros e plataformas de motores de jogos.

1.4. Motores de jogos presentes na indústria.

1.5. Histórico dos motores de jogos presentes na indústria.

1.6. Descrição dos principais motores de jogos.

2. Elementos básicos de motor de jogos 3D.

2.1. Introdução aos Motores de Jogos 3D

2.2. Estrutura do Motor de Jogo

2.3. Ferramentas do Motor de Jogo

2.4. Componentes visuais.

3. Programação para o Motor do Jogo

3.1. Hierarquia de objetos na engine.

3.1.1.Eventos.

3.1.2.Viewport.

3.1.3.Cenas de Jogo.

3.2. Introdução à câmera, iluminação e textura.

3.3. Tipos de dispositivos de interface humana.

3.4. Entrada e saída de dados.

3.5. Movimentação, colisão e outras interações básicas.

3.6. Escala, movimentação e rotação.

3.7. Importação de modelos 3D e animações.

3.8. Aplicação de materiais aos modelos.

3.9. Movimentação de personagens.

3.10. Construção de scripts para programação de comportamentos em objetos do jogo.

4. Efeitos especiais.

5. Aplicação de sons e efeitos sonoros ao jogo.

REVOGADO

UNIDADE FORMATIVA 6: UF408 Programação de Jogos (120 Horas)

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA 1: Caraterizar as linguagens de programação para jogos.

CA 1.1. Conhecer os conceitos de jogos digitais.

CA 1.2. Diferenciar as linguagens de programação para jogos.

CA 1.3. Diferenciar as linguagens de programação de jogos em função de canais de distribuição.

RA 2: Diagnosticar hardware e realizar conexão dos dispositivos periféricos

CA 2.1. Identificar os hardwares e os dispositivos utilizados na programação de jogos.

CA 2.2. Indicar a função das placas gráficas e de som na programação de jogos.

CA 2.3. Identificar as portas de conexão dos dispositivos periféricos utilizadas na programação de jogos.

CA 2.4. Identificar os diferentes tipos de cabos utilizados na conexão dos dispositivos.

CA 2.5. Numa situação prática de conexão dos equipamentos:

- Selecionar os periféricos para conexão;
- Selecionar os cabos;
- Selecionar o hardware;
- Conectar os dispositivos periféricos ao processador.

RA 3: Programar jogos utilizando linguagens de programação de jogos.

CA 3.1. Indicar as funções das diferentes bibliotecas utilizadas pelos motores de jogos.

CA 3.2. Numa situação prática de programação de jogos:

- Interpretar o guião;
- Desenhar o storyboard;
- Definir a mecânica de jogos;
- Definir a animação;
- Selecionar a linguagem de jogos a ser utilizado;
- Selecionar os componentes ou outras bibliotecas disponibilizadas por terceiros;
- Programar Jogos;
- Realizar teste do jogo;
- Publicar jogos utilizando diferentes canais.

RA 4: Programar jogos para as principais plataformas de desenvolvimento de jogos para dispositivos móveis.

CA 4.1. Conhecer as limitações e características do desenvolvimento de jogos para dispositivos móveis.

CA 4.2. Conhecer as restrições e vantagens de cada plataforma de desenvolvimento.

CA 4.3. Explicar arte para jogos para dispositivos móveis compatíveis com as restrições de cada plataforma.

CA 4.4. Conhecer os mecanismos de distribuição e comercialização de jogos para dispositivos móveis.

CA 4.5. Numa situação prática programar jogos para dispositivos móveis:

- Selecionar a plataforma;
- Interpretar o guião;
- Codificar jogos;

REVOGADO

- Depurar jogos;
- Testar jogos.

Resultados de Aprendizagem cujas aquisições devem ser complementadas em ambiente real de trabalho

- RA 2 relativamente ao CA 2.5.
- RA 3 relativamente ao CA 3.5.
- RA 4 relativamente ao CA 4.5.

Conteúdos

1. **Introdução a jogos digitais para web.**
   1.1. Noções de Web Design para jogos digitais.
   1.2. Funcionamento de jogos digitais para web.
   1.3. Tipos de Jogos Web: Jogos casuais; single player, multiplayer e sociais.
   1.4. Definição de formatos (web, mobile ou Desktop);
2. **Estruturando documentos com (X)HTML**.
   2.1. Diferenças entre HTML e XHTML.
   2.2. 1. Programação de jogos para web.
   2.3. Estrutura ou corpo básicos
   2.4. Estruturas semântica da página web;
3. **Framework HTML5 2D/3D.**
   3.1. Validação de códigos.
   3.2. Variáveis e tipos de dados.
   3.3. Referentes à linguagem de programação;
   3.4. Comandos de decisão, repetição e operadores;
   3.5. Funções e métodos;
   3.6. Classes e componentes adicionais referente a linguagem de programação.
   3.7. Controlos e interface gráfica para o utilizador.
4. **Ambiente de animação.**
   4.1. Camada de Desenho.
   4.2. Canvas e seus atributos.
      4.2.1. Canvas game.
   4.3. Criando uma animação com imagens mapeadas.
   4.4. SVG e seus atributos;
5. **Manipulação de elementos multimídia em jogos.**
   5.1. Suporte a áudio com Web Audio.
   5.2. Suporte a vídeo com Web Video.
   5.3. Suporte à câmera, microfone e outros recursos.
6. **Programação de gráficos e animações**
   6.1. Trabalhando com animações em web.
   6.2. Interface gráfica em jogos.
   6.3. Gráficos baseados em vetor para web.
   6.4. Gráficos 2D e 3D através do elemento Canvas.

REVOGADO

6.5. APIs e frameworks para animação em jogos web.

7. **Som:**
   7.1. Áudio e seus atributos.
   7.2. Vídeo e seus atributos.
   7.3. Publicação de arquivo.
8. **Técnicas CSS para experiências visuais ricas:**
   8.1. Transformação;
   8.2. Transição;
   8.3. Rotação;
   8.4. Movimentação (drag and drop) de elementos.
9. **Animações utilizando o CSS3.**
10. **Game Loop.**
11. **JavaScript.**
    11.1. Detectando teclas pressionadas.
    11.2. Limitando a movimentação do jogador e criando uma animação pelo JavaScript.
    11.3. Animando a div inimigo.
    11.4. Criando a movimentação das demais divs do jogo.
    11.5. Criando a função disparo.
12. **Framework Jquery Collision.**
13. **Pontuação.**
14. **Criar uma barra de energia.**
15. **Aumentar a dificuldade do jogo.**
16. **Sons.**
17. **Game over.**
18. **Reiniciando o jogo.**
19. **Framework Jquery Hammer.**
20. **Adaptando o jogo.**
21. **Suporte a execução de scripts Javascript em outra thread (web workers).**
22. **APIs e frameworks para animação em jogos web.**
23. **Conceito Orientaçao a objetos para jogos.**
    23.1. Conceitos de orientação a objetos:
       23.1.1. Classes e objetos;
       23.1.2. Métodos;
       23.1.3. Encapsulamento;
       23.1.4. Herança;
       23.1.5. Sobrecarga e sobrescrita;
       23.1.6. Polimorfismo..
       23.1.7. Tratamento de exceções
24. **Ambientes gráficos:**
    24.1. AWT;
    24.2. Swing

RE ADO

25. Elementos de Jogos 2D/3D.

25.1. Game Design.

25.2. Sprites e TileMaps;

25.3. Animação e desenho 2D;

25.4. Captura Eventos Teclado;

26. Conexão a banco de dados.

27. Matrizes e variáveis complexas.

28. Mecanismos de Física:

28.1. Aproximação, movimento retangular, colisão e gravidade;

28.2. Framework Box2D;

29. Tratamento de eventos:

29.1. Windows;

29.2. Apple;

29.3. Linux.

30. Jogos Conectados

30.1. Multijogador;

30.2. WebSockets;

30.3. Framework Socket IO;

31. Introdução ao Android

31.1. Máquina Virtual Dalvik

31.2. Instalação e Configuração de Ambiente

31.3. Criando um Emulador para Android

31.4. Activity, View e o Método setContentView (view)

31.5. A Classe R

31.6. ListActivity – Exibição de uma Lista de Itens na Tela

31.6.1. ArrayAdapter – Preenchimento da Lista

31.6.2. SimpleCursorAdapter – Exibição dos Contatos de Agenda em uma Lista

31.6.3. Criação de um ListAdapter Customizado com Imagens

31.6.4. MapaActivity

31.7. Intent.

31.8. Interface Gráfica – Gerenciadores de Layout.

31.9. Persistência de Dados.

31.10. HTTP – WEB – JSON/GSON.

31.11. Integração com Recursos do Aparelho.

32. Programação de Jogos 2D com códigos em Android.

32.1. Utilização do Unity para criação de jogos 2D e 3D.

33. Persistência de dados.

33.1. Conceito de persistência e banco de dados.

33.2. Persistindo dados em arquivos.

33.3. Serialização de dados.

33.4. Operações básicas manipulação de dados.

REVOGADO

33.5. Base de dados embutida em aplicações web.

Requisitos básicos do contexto formativo do Módulo

Espaços:

Os espaços onde deve decorrer o contexto formativo devem preencher os requisitos básicos estipulados pelo Sistema de Acreditação das Entidades Formadoras. (cf. Decreto-Lei nº6/2013, de 11 de fevereiro; Boletim Oficial nº4; I Série).

Professor / Formador:

- O Professor ou formador deve possuir um Certificado de Aptidão Profissional que prove que o mesmo é detentor de formação pedagógica na abordagem por competências, segundo a lei.
- O professor ou formador deve ter a formação técnica relacionada com os conteúdos da Unidade/Módulo Formativo.
- O professor ou formador deve ter experiência profissional de um mínimo de 3 anos nas competências incluídas na Unidade de Competência associada à Unidade/Módulo Formativo.

Requisitos de acesso ao módulo formativo

As condições de acesso ao módulo formativo constam BO nº 20, I Série, Portaria Conjunta nº 09/2020 de 20 de fevereiro – Estrutura e Organização do Catálogo Nacional de Qualificações e Portaria Conjunta nº 10/2020 de 20 de fevereiro – Quadro Nacional de Qualificações

REVOGADO

MÓDULO FORMATIVO EM CONTEXTO REAL DE TRABALHO	MT_TIC001.A1
Nível: 5	Duração indicativa: 360 Horas
Associado a todas as Unidades de Competência	

Resultados de Aprendizagem (RA) e Critérios de Avaliação (CA)

RA1. Representar algoritmo em forma de fluxograma e Pseudocódigo.

- **CA 1.1.** Selecionar as simbologias de entrada e saída de dados.
- **CA 1.2.** Selecionar as simbologias de processamento.
- **CA 1.3.** Desenhar o fluxograma do algoritmo.
- **CA 1.4.** Definir as palavras chaves de entrada, saída e processamento de dados.
- **CA 1.5.** Declarar e inicializar as variáveis e constantes da estrutura de dados.
- **CA 1.6.** Definir a estrutura condicional adequada ao fluxo de dados.
- **CA 1.7.** Definir o ciclo repetitivo adequado ao fluxo de dados.
- **CA 1.8.** Escrever o pseudocódigo.

RA2. Realizar o levantamento, a análise e especificação de requisitos do sistema.

- **CA 2.1.** Identificar e delimitar o micromundo (área de atuação).
- **CA 2.2.** Realizar o levantamento dos requisitos, aplicando as técnicas adequadas.
- **CA 2.3.** Analisar e definir os requisitos do sistema.

RA3. Desenhar o diagrama de Caso de Uso do sistema.

- **CA 3.1.** Numa situação prática, desenhar um diagrama de caso de uso, utilizando as ferramentas de diagramas UML.

RA4. Desenhar o diagrama de Classe e objeto do sistema.

- **CA 4.1.** Identificar as classes do sistema a utilizar.
- **CA 4.2.** Definir os atributos e métodos das classes.
- **CA 4.3.** Definir o relacionamento entre as classes.
- **CA 4.4.** Instanciar objeto.

RA5. Identificar as especificações e funcionalidades do *software* a ser programado.

- **CA 5.1.** Utilizar o modelo ou diagramas de caso de uso para organizar as funcionalidades.
- **CA 5.2.** Testar a funcionalidade do desenho de interface em relação às necessidades requeridas.

RA6. Organizar o desenho da *interface* de utilizador em função dos requisitos funcionais definidas no projeto.

- **CA 6.1.** Selecionar a ferramenta de desenho de interface.
- **CA 6.2.** Identificar os componentes de interface da aplicação.
- **CA 6.3.** Produzir casos de teste para a integração de interface com outros componentes da aplicação.
- **CA 6.4.** Adaptar os componentes desenvolvidos a modificações das especificações recebidas.
- **CA 6.5.** Desenhar interface de acordo com os requisitos do projeto.
- **CA 6.6.** Testar a interação entre o utilizador e a interface.
- **CA 6.7.** Preparar os componentes produzidos para a sua distribuição.

PROGRESSADO

CA 6.8. Produzir os documentos de instalação, configuração ou administração necessários, de acordo com as normas e critérios de usabilidade.

RA7. Reutilizar e adaptar *templates* e outros componentes multimédia no desenvolvimento da *interface* de utilizador.

CA 7.1. Projetar as transformações necessárias a fim de obter componentes multimédia, conforme os requisitos do projeto software.

CA 7.2. Verificar a adequação do *template* adaptado ao padrão definido no projeto.

RA8. Desenhar a arquitetura de um sistema de informação.

CA 8.1. Desenhar o protótipo do sistema.

CA 8.2. Adaptar o *template* às funcionalidades do site web.

CA 8.3. Testar o protótipo para garantir a sua funcionalidade.

RA9. Instalar e configurar *frameworks.*

CA 9.1. Selecionar o *framework*.

CA 9.2. Selecionar o Servidor *Web*.

CA 9.3. Selecionar o Servidor de Base de dados.

CA 9.4. Selecionar os plugins exigidos pelos *framework* para gerar a compatibilidade.

CA 9.5. Instalar *plugins*, servidor *web,* servidor de base de dados, *framework*.

CA 9.6. Configurar o servidor *web*, servidor da base de dados.

RA10.Criar um programa utilizando uma linguagem de programação estruturada.

CA 10.1. Selecionar a linguagem de programação para implementação do algoritmo.

CA 10.2. Definir as variáveis de entrada, saída e processamento de dados.

CA 10.3. Declarar, atribuir e inicializar as variáveis da estrutura de dados.

CA 10.4. Declarar e definir as constantes.

CA 10.5. Definir as estruturas de acordo com o algoritmo.

CA 10.6. Escrever o código fonte.

CA 10.7. Compilar o código fonte.

CA 10.8. Realizar *debug* do código fonte.

CA 10.9. Testar o programa.

RA11.Organizar e documentar o código fonte.

CA 11.1. Selecionar as ferramentas de versionamento;

CA 11.2. Organizar o código.

CA 11.3. Comentar o código fonte.

CA 11.4. Versionar o código fonte.

CA 11.5. Documentar o código fonte.

RA12.Editar e compilar o código fonte.

CA 12.1. Definir as classes.

CA 12.2. Declarar os atributos e métodos das classes.

CA 12.3. Definir a visibilidade dos atributos e métodos das classes.

CA 12.4. Declarar e inicializar as variáveis.

CA 12.5. Editar código fonte.

REVOGADO

CA 12.6. Compilar o código fonte.

CA 12.7. Realizar *debug* do código fonte.

CA 12.8. Testar o programa.

RA13.Realizar programação para *web.*

CA 13.1. Selecionar as ferramentas de desenvolvimento web.

CA 13.2. Codificar site estático e/ou dinâmico.

CA 13.3. Testar o site utilizando *browser* para visualização.

RA14.Realizar Testes dos serviços de segurança num servidor.

CA 14.1. Estabelecer os parâmetros de configuração dos serviços.

CA 14.2. Testar a compatibilidade de segurança entre o servidor e a aplicação.

RA15.Codificar uma página web utilizando a linguagem de marcação HTML.

CA 15.1. Identificar a estrutura do documento HML.

CA 15.2. Editar o código, respeitando a sintaxe.

CA 15.3. Testar a página *web* codificada, utilizando *browser* compatível.

RA16.Realizar a hospedagem e a publicação do sistema.

CA 16.1. Rececionar, registar, carimbar e encaminhar o documento para despacho.

CA 16.2. Selecionar o plano de hospedagem.

CA 16.3. Verificar a compatibilidade dos softwares instalados com o do sistema desenvolvido.

CA 16.4. Solicitar o domínio num provedor.

CA 16.5. Configurar o domínio do sistema.

CA 16.6. Realizar upload dos ficheiros do sistema com recurso a software específico.

CA 16.7. Testar as funcionalidades da aplicação.

CA 16.8. Publicar o site.

RA17.Formatar uma página web utilizando a linguagem de formatação CSS.

CA 17.1. Definir o estilo de formatação.

CA 17.2. Adicionar estilos CSS (Inline, interno e externo).

CA 17.3. Aplicar os efeitos aos diferentes tipos de seletores.

RA18.Construir uma página *web* dinâmica.

CA 18.1. Definir a interação a aplicar à página web.

CA 18.2. Selecionar o tipo de animação a aplicar aos objetos.

CA 18.3. Escrever o código HTML da página web.

CA 18.4. Escrever o código JavaScript para animação, *inline* ou num arquivo separado.

CA 18.5. Testar o código.

CA 18.6. Testar a página *web* criada.

RA19.Desenvolver uma API (*Aplication Programation Interfaces*) para a integração de sistemas.

CA 19.1. Identificar a arquitetura dos sistemas existentes.

CA 19.2. Definir a arquitetura da API a desenvolver.

CA 19.3. Definir a estrutura de dados a ser consumido.

CA 19.4. Editar e compilar o programa.

REVOGADO

CA 19.5. Realizar *debug* do programa.

RA20. Programar do lado do servidor (*server side*).

CA 20.1. Selecionar a linguagem de programação *web* a utilizar.

CA 20.2. Editar o código fonte.

CA 20.3. Compilar o código fonte.

CA 20.4. Depurar erros e voltar a compilar.

RA21. Realizar o levantamento dos requisitos para desenho do Modelo Entidade Relacionamento (MER).

CA 21.1. Identificar e delimitar o micromundo (área de atuação).

CA 21.2. Identificar e analisar o problema.

CA 21.3. Realizar o levantamento dos requisitos aplicando técnicas adequadas.

CA 21.4. Especificar os requisitos para o desenho do Diagrama Entidade Relacionamento (DER).

RA22. Desenhar o Modelo Entidade Relacionamento (MER).

CA 22.1. Determinar as entidades.

CA 22.2. Identificar os relacionamentos.

CA 22.3. Adicionar os atributos.

CA 22.4. Unir as entidades por meio das relações.

CA 22.5. Adicionar a cardinalidade da relação.

RA23. Interpretar a estrutura de uma base de dados e desenhar o modelo lógico de acordo com o Diagrama Entidade Relacionamento – DER.

CA 23.1. Identificar e interpretar as simbologias utilizadas no DER.

CA 23.2. Identificar as entidades a partir das simbologias.

CA 23.3. Identificar o tipo de relação entre as entidades.

CA 23.4. Definir as regras de integridade.

CA 23.5. Desenhar um dicionário de dados.

CA 23.6. Desenhar o modelo lógico.

RA24. Criar uma base de dados de acordo com DER.

CA 24.1. Criar uma base de dados utilizando o comando DDL.

CA 24.2. Criar tabelas utilizando comando DDL.

RA25. Manipular os dados, utilizando linguagem DML.

CA 25.1. Identificar as restrições de integridade e consistência da base de dados.

CA 25.2. Escrever a sintaxe da consulta para extrair informação contida na base de dados, com diferentes níveis de complexidade, utilizando a linguagem DML.

CA 25.3. Escrever sintaxe para inserção de dados na base de dados, utilizando a linguagem DML.

CA 25.4. Escrever a sintaxes para atualização e eliminação de dados utilizando linguagem DML.

CA 25.5. Codificar scripts para automatizar as tarefas.

CA 25.6. Redigir os documentos descritivos dos procedimentos realizados de acordo com as normas pré-estabelecidas.

RA26. Manipular os dados, utilizando SQL.

CA 26.1. Criar e eliminar Base de Dados.

REVOGADO

CA 26.2. Criar, atualizar e eliminar tabelas.

CA 26.3. Inserir e atualizar os dados numa tabela.

CA 26.4. Selecionar os dados numa ou mais tabelas.

RA27. Desenvolver um programa com *interface* gráfica para conexão à base de dados.

CA 27.1. Importar a biblioteca para interface gráfica.

CA 27.2. Importar a biblioteca de acesso a base de dados.

CA 27.3. Definir os tipos de dados abstratos.

CA 27.4. Definir a forma de conexão com a base de dados.

CA 27.5. Utilizar drivers para conectar à base de dados.

CA 27.6. Testar a conetividade com a base de dados.

CA 27.7. Compilar o programa.

CA 27.8. Realizar *debug* do programa.

CA 27.9. Realizar teste do programa.

RA28. Desenvolver um programa dividindo em subprogramas.

CA 28.1. Importar a biblioteca para interface gráfica.

CA 28.2. Importar a biblioteca de acesso a base de dados.

CA 28.3. Definir os tipos de dados abstratos.

CA 28.4. Definir a forma de conexão.

CA 28.5. Utilizar drivers para conectar a base de dados.

CA 28.6. Testar a conetividade.

CA 28.7. Compilar o programa.

CA 28.8. Realizar *debug* do programa.

CA 28.9. Realizar teste do programa.

RA29. Desenvolver um programa integrando com interface gráfica e acesso a base de dados.

CA 29.1. Importar a biblioteca para interface gráfica.

CA 29.2. Importar a biblioteca de acesso a base de dados.

CA 29.3. Definir os tipos de dados abstratos.

CA 29.4. Definir a forma de conexão com a base de dados.

CA 29.5. Utilizar drivers para conectar à base de dados.

CA 29.6. Testar a conetividade com a base de dados.

CA 29.7. Compilar o programa.

CA 29.8. Realizar *debug* do programa.

CA 29.9. Realizar teste do programa.

RA30. Realizar o levantamento dos requisitos para desenho do Modelo Entidade Relacionamento (MER).

CA 30.1. Identificar e delimitar o micromundo (área de atuação).

CA 30.2. Identificar e analisar o problema.

CA 30.3. Realizar o levantamento dos requisitos aplicando técnicas adequadas.

CA 30.4. Especificar os requisitos para o desenho do Diagrama Entidade Relacionamento (DER).

REVOGADO

RA31.Desenhar o Modelo Entidade Relacionamento (MER).

CA 31.1. Determinar as entidades.

CA 31.2. Identificar os relacionamentos.

CA 31.3. Adicionar os atributos.

CA 31.4. Unir as entidades por meio das relações.

CA 31.5. Adicionar a cardinalidade da relação.

RA32.Manipular os dados utilizando a SQL.

CA 32.1. Utilizar comandos para criar e eliminar Base de Dados.

CA 32.2. Utilizar comandos para criar, atualizar e eliminar tabelas.

CA 32.3. Utilizar comandos para inserir, atualizar os dados numa tabela.

CA 32.4. Utilizar comandos para selecionar os dados numa ou mais tabelas.

RA33.Realizar a proteção de dados nos dispositivos móveis utilizando os recursos do sistema operativo.

CA 33.1. Selecionar o nível de segurança de acesso.

CA 33.2. Aplicar um código de segurança.

CA 33.3. Encriptar os dados nos dispositivos.

CA 33.4. Selecionar uma software de proteção contra vírus e *malware*.

CA 33.5. Recuperar os dados.

RA34.Desenvolver uma aplicação *mobile-android.*

CA 34.1. Escolher o tipo de aplicação a desenvolver.

CA 34.2. Selecionar o ambiente de desenvolvimento.

CA 34.3. Definir as funcionalidades da aplicação.

CA 34.4. Selecionar os recursos do dispositivo a ser utilizado.

CA 34.5. Escrever o código fonte.

CA 34.6. Compilar o código fonte.

CA 34.7. Realizar *debug* do código fonte.

CA 34.8. Realizar teste da aplicação.

RA35.Desenvolver uma aplicação *mobile*-IoS.

CA 35.1. Escolher o tipo de aplicação a desenvolver.

CA 35.2. Selecionar o ambiente de desenvolvimento.

CA 35.3. Definir as funcionalidades da aplicação.

CA 35.4. Selecionar os recursos do dispositivo a ser utilizado.

CA 35.5. Escrever o código fonte;

- Compilar o código fonte.
- Realizar *debug* do código fonte.

CA 35.6. Realizar teste da aplicação.

RA36.Planear o desenvolvimento *mobile* multiplataforma.

CA 36.1. Selecionar a tecnología.

CA 36.2. Selecionar framework a utilizar.

CA 36.3. Selecionar a linguagem de programação para desenvolvimento multiplataforma.

REVOGADO

RA37.Desenvolver aplicação *mobile* multiplataforma.

- **CA 37.1.** Selecionar a tecnología.
- **CA 37.2.** Selecionar framework.
- **CA 37.3.** Selecionar a linguagem de programação para o desenvolvimento *mobile* multiplataforma.
- **CA 37.4.** Desenhar a interface.
- **CA 37.5.** Editar o código.
- **CA 37.6.** Relaizar *debug* da aplicação.
- **CA 37.7.** Realizar teste do aplicativo.

RA38.Realizar medição dos componentes eletrónicos.

- **CA 38.1.** Selecionar os componentes a serem medidos.
- **CA 38.2.** Selecionar o aparelho de medição.
- **CA 38.3.** Calibrar o equipamento de medição em função dos parâmetros de medição.
- **CA 38.4.** Realizar a medição.

RA39.Interpretar um esquema do circuito eletrónico.

- **CA 39.1.** Conhecer as simbologias dos componentes eletrónicos.
- **CA 39.2.** Realizar leitura de esquema para identificação dos componentes.
- **CA 39.3.** Identificar os componentes eletrónicos básicos.
- **CA 39.4.** Explicar as caraterísticas dos diferentes componentes.
- **CA 39.5.** Explicar as ligações da representação esquemática.

RA40.Montar um circuito eletrónico numa *protoboard.*

- **CA 40.1.** Interpretar o esquema de um circuito eletrónico.
- **CA 40.2.** Identificar as áreas disponíveis numa protoboard.
- **CA 40.3.** Selecionar os componentes para montagem de protoboard de acordo com esquema.
- **CA 40.4.** Montar os componentes de acordo com o esquema fornecido.
- **CA 40.5.** Conetar os componentes utilizando os fios de jumpers.
- **CA 40.6.** Testar o circuito.

RA41.Caraterizar uma rede de sensores sem fio (RSSF).

- **CA 41.1.** Preparar o ambiente Contiki OS.
- **CA 41.2.** Acessar os periféricos do kit de desenvolvimento.
- **CA 41.3.** Estabelecer a comunicação entre os nós, utilizando protocolo UDP para rede de Sensores.
- **CA 41.4.** Configurar parâmetros de RSSF para o Contiki OS.
- **CA 41.5.** Testar o roteamento para rede de sensores.

RA42.Instalar e configurar o ambiente de desenvolvimento (IDE).

- **CA 42.1.** Selecionar e instalar o IDE de acordo com o projeto.
- **CA 42.2.** Configurar os parámetros do IDE.
- **CA 42.3.** Selecionar a linguagem de programação.

RA43.Programar Microcontroladores/microprocessadores.

- **CA 43.1.** Declarar e inicializar variáveis.
- **CA 43.2.** Programar sistemas embarcados baseados em microcontroladores.

CA 43.3. Utilizar estruturas de controlo para manipular LEDs.

CA 43.4. Utilizar funções das bibliotecas para manipular os recursos dos microcontroladores.

CA 43.5. Controlar sensores, atuadores e dispositivos de E/S a partir de microcontroladores.

CA 43.6. Compilar o código fonte.

CA 43.7. Realizar *Debug*.

CA 43.8. Realizar teste da aplicação.

RA44.Realizar treinamento de uma máquina.

CA 44.1. Coletar os dados necessários.

CA 44.2. Preparar os dados.

CA 44.3. Selecionar o modelo de aprendizagem.

CA 44.4. Identificar o algoritmo em função do modelo selecionado.

CA 44.5. Treinar a máquina.

CA 44.6. Avaliar o desempenho da máquina.

CA 44.7. Ajustar os parâmetros, visando melhorar a qualidade e a eficiência do modelo.

CA 44.8. Realizar a predição da máquina.

RA45.Analisar os dados coletados dos dispositivos *IoT*.

CA 45.1. Identificar e selecionar as fontes de dados.

CA 45.2. Selecionar os parâmetros dos dados a serem analisados.

CA 45.3. Coletar os dados para análise.

CA 45.4. Selecionar e aplicar o software adequado para análise de dados.

CA 45.5. Interpretar o modelo de dados produzido.

CA 45.6. Interpretar os resultados.

RA46.Monitorizar os dispositivos com base nos dados analisados.

CA 46.1. Identificar o objeto sujeito a monitorização (dispositivos, ambiente, entre outros).

CA 46.2. Selecionar os recursos a serem monitorizados.

CA 46.3. Selecionar as ferramentas (software) para o processo de monitorização.

CA 46.4. Configurar os recursos a serem monitorizados.

CA 46.5. Realizar teste de monitorização.

RA47.Diagnosticar *hardware* e realizar conexão dos dispositivos periféricos.

CA 47.1. Selecionar os periféricos para conexão.

CA 47.2. Selecionar os cabos.

CA 47.3. Selecionar o hardware.

CA 47.4. Conectar os dispositivos periféricos ao processador.

RA48.Programar *IoT* utilizando linguagens de programação *mobile.*

CA 48.1. Selecionar a linguagem de programação a ser utilizado.

CA 48.2. Selecionar os componentes ou outras bibliotecas disponibilizadas por terceiros.

CA 48.3. Programar aplicativo mobile para IoT.

CA 48.4. Realizar *debug.*

CA 48.5. Realizar teste.

REVOGADO

RA49.Interpretar um Guião.

- **CA 49.1.** Identificar as personagens.
- **CA 49.2.** Identificar e descrever as funções das personagens.
- **CA 49.3.** Identificar o argumento.
- **CA 49.4.** Identificar as cenas, as sequências e os subenredos.
- **CA 49.5.** Enquadrar as cenas no tempo e no espaço.

RA50.Definir o plano de realização do jogo tendo em conta o Guião.

- **CA 50.1.** Interpretar o guião.
- **CA 50.2.** Definir o objetivo da produção.
- **CA 50.3.** Realizar a decupagem do guião.
- **CA 50.4.** Elaborar um storyboard a partir do guião.
- **CA 50.5.** Definir plano de enquadramento da câmara.

RA51.Desenhar diagramas UML para o desenvolvimento de um jogo.

- **CA 51.1.** Definir os requisitos funcionais e não funcionais do jogo.
- **CA 51.2.** Desenhar os diagrams UML do projeto de jogo.

RA52.Caraterizar e descrever a mecânica dos jogos.

- **CA 52.1.** Interpretar roteiro.
- **CA 52.2.** Identificar as personagens da narrativa.
- **CA 52.3.** Identificar a mecânica associada a personagem.
- **CA 52.4.** Definir sistemas de coordenadas do objeto.
- **CA 52.5.** Aplicar a mecânica no objeto de acordo o guião.

RA53.Desenhar um jogo digital.

- **CA 53.1.** Interpretar roteiro.
- **CA 53.2.** Identificar as personagens da narrativa.
- **CA 53.3.** Identificar a mecânica associada a personagem.
- **CA 53.4.** Desenhar o protótipo do jogo utilizando os diagramas UML desenhado, aplicado mecânica dos jogos.
- **CA 53.5.** Realizar teste de qualidade.

RA54.Caraterizar um espaço 2D e 3 D.

- **CA 54.1.** Definir o conceito de modelação 3D.
- **CA 54.2.** Identificar os operadores básicos utilizados na modelação 3 D.
- **CA 54.3.** Descrever um espaço 3D.

RA55.Realizar modelação 2D e 3D.

- **CA 55.1.** Selecionar Software a utilizar.
- **CA 55.2.** Definir o sistema de coordenada no espaço 3 D.
- **CA 55.3.** Selecionar ou conceber o objeto a ser modelado.
- **CA 55.4.** Aplicar textura.
- **CA 55.5.** Realizar Rigging.

RA56.Realizar animação 2D e 3D.

REVOGADO

CA 56.1. Selecionar o software para animação.

CA 56.2. Selecionar o objeto a ser animado.

CA 56.3. Animar o objeto aplicando os métodos de animação 3D.

RA57.Captar, manipular e gerar elementos de áudio.

CA 57.1. Criar a trilha sonora para o jogo.

CA 57.2. Criar os efeitos sonoros para o jogo.

CA 57.3. Introduzir o áudio criado no jogo.

RA58.Realizar treinamento da máquina.

CA 58.1. Coletar e preparar os dados necessários para o treinamento de uma máquina.

CA 58.2. Selecionar o algoritmo em função dos agentes.

CA 58.3. Aplicar o algoritmo para treinar a máquina.

CA 58.4. Realizar teste da máquina.

CA 58.5. Ajustar os parâmetros, caso necessário, para melhorar a qualidade e a eficiência da máquina.

CA 58.6. Realizar a predição da máquina.

RA59.Caraterizar os motores de Jogos.

CA 59.1. Definir um motor de jogo.

CA 59.2. Distinguir os motores de jogos licenciados dos motores gratuitos.

CA 59.3. Enumerar as vantagens e as desvantagens dos motores de jogos.

CA 59.4. Distinguir os diferentes tipos de motores de jogos.

RA60.Realizar a conexão dos dispositivos periféricos utilizados nos jogos.

CA 60.1. Selecionar os periféricos para conexão.

CA 60.2. Selecionar os cabos.

CA 60.3. Selecionar o hardware.

CA 60.4. Conectar os dispositivos periféricos ao processador.

RA61.Programar jogos utilizando motores de jogos.

CA 61.1. Indicar a função de motores de jogos.

CA 61.2. Explicar como os diversos componentes de um motor de jogo 3D podem ser usados no desenvolvimento de diversos tipos de jogos.

CA 61.3. Indicar as funções das diferentes bibliotecas utilizadas pelos motores de jogos.

CA 61.4. Diferenciar os motores de jogos gráficos de motores físicos.

CA 61.5. Explicar como manipular e controlar modelos 3D animados dentro do motor.

CA 61.6. Interpretar o guião.

CA 61.7. Desenhar o storyboard.

CA 61.8. Definir a mecânica de jogos.

CA 61.9. Definir a animação.

CA 61.10. Selecionar o motor de jogos a ser utilizado.

CA 61.11. Identificar os recursos dos motores de jogos.

CA 61.12. Verificar a compatibilidade dos motores de jogos com as plataformas.

CA 61.13. Selecionar os componentes ou outras bibliotecas disponibilizadas por terceiros.

REVOGADO

CA 61.14. Aplicar componentes de inteligência artificial e física do motor 3D num jogo.

CA 61.15. Implementar algoritmos em linguagem script do motor, para programação da jogabilidade.

CA 61.16. Realizar teste do jogo.

RA62. Diagnosticar hardware e realizar conexão dos dispositivos periféricos.

CA 62.1. Selecionar os periféricos para conexão.

CA 62.2. Selecionar os cabos.

CA 62.3. Selecionar o hardware.

CA 62.4. Conectar os dispositivos periféricos ao processador.

RA63. Programar jogos utilizando linguagens de programação de jogos.

CA 63.1. Interpretar o guião.

CA 63.2. Desenhar o storyboard.

CA 63.3. Definir a mecânica de jogos.

CA 63.4. Definir a animação.

CA 63.5. Selecionar a linguagem de jogos a ser utilizado.

CA 63.6. Selecionar os componentes ou outras bibliotecas disponibilizadas por terceiros.

CA 63.7. Programar Jogos.

CA 63.8. Realizar teste do jogo.

CA 63.9. Publicar jogos utilizando diferentes canais.

RA64. Programar jogos para as principais plataformas de desenvolvimento de jogos para dispositivos móveis.

CA 64.1. Selecionar a plataforma.

CA 64.2. Interpretar o guião.

CA 64.3. Codificar jogos.

CA 64.4. Depurar jogos.

CA 64.5. Testar jogos.

REVOGADO