Ministério da Promoção de Investimentos e Fomento Empresarial
Unidade de Coordenação do Sistema Nacional de Qualificações

SISTEMA NACIONAL DE QUALIFICAÇÕES

Qualificação Profissional

DESENVOLVIMENTO DE APLICAÇÕES WEB FULL-STACK

TIC001.A2_5

Família Profissional Tecnologias de Informação e Comunicação

Ficha Técnica

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

Coordenador da UC-SNQ
Amílcar Alexandre Mendes

Família Profissional: Tecnologia de Informação e Comunicação

Coordenador(a) da Família Profissional

Agosto 2025

Apoio ao utilizador

Telefone: +238 333 70 55/76/61 | E-Mail: snqcv.uc@gmail.com | Website: www.snq.cv

Siglas e Acrónimos

TIC	Tecnologias de Informação e Comunicação
UC	Unidades de Competência
EC	Elementos de Competência
CD	Critérios de Desempenho
HTTPS	HyperText Transfer Protocol Secure
SQL	Structured Query Language
API	Application Programming Interface
XSS	Cross-Site Scripting
WCAG	Web Content Accessibility Guidelines
CSRF	Cross-Site Request Forgery
CSP	Content Security Policy
SRI	Subresource Integrity
WASM	WebAssembly
PWA	Progressive Web Apps
DAST	Dynamic Application Security Testing
SAST	Static Application Security Testing
RWD	Responsive Web Design

ÍNDICE

Unidades de Competência 3

PERFIL PROFISSIONAL 6

UNIDADES DE COMPETÊNCIA (UC) 7

UC 1: DESENVOLVER COMPONENTES DE SOFTWARE NO AMBIENTE DO CLIENTE PARA APLICAÇÃO WEB 7

UC2: DESENVOLVER COMPONENTES DE SOFTWARE ORIENTADAS À EXECUÇÃO NO BACKEND DE APLICAÇÕES WEB _0) 10

UC3: DESENVOLVER APLICAÇÕES WEB DE ALTO DESEMPENHO, USABILIDADE, E ESCALABILIDADE C000_0) 13

UC4: IMPLEMENTAR AS API E SERVIÇOS WEB PARA INTEGRAÇÃO ENTRE SISTEMAS, OTIMIZANDO O DESEMPENHO E A SEGURANÇA EM APLICAÇÕES WEB00_0) 16

UNIDADES FORMATIVAS TRANSVERSAIS 21

UNIDADE FORMATIVA : Inglês Técnico 21

MÓDULOS FORMATIVOS (MF) 24

MF1: Desenvolvimento Frontend e Otimização de Interfaces Web 24

Unidade Formativa 1: Otimização e Estruturação de Documentos Web (UF476) 24

Unidade Formativa 2: Programação Frontend e Interatividade Web(UF477) 28

MF2: Desenvolvimento de Aplicações Web no Ambiente do Servidor 31

Unidade Formativa 1: Arquiteturas e Tecnologias para Desenvolvimento Web no Servidor (UF478) 31

Unidade Formativa 2: Desenvolvimento e Integração de Aplicações Web no Servidor (UF479)34

MF3: Automação, Monitorização e Escalabilidade de Aplicações Web 37

Unidade Formativa 1: Gestão e Automação de Processos de Entrega Web (UF480) 37

Unidade Formativa 2: Monitorização e Escalabilidade de Aplicações Web (UF481) 40

MF4: Integração de APIs, Otimização e Segurança em Aplicações Web 42

Unidade Formativa 1: Integração e Gestão de APIs em Aplicações Web (UF482) 42

Unidade Formativa 2: Segurança e Qualidade em Aplicações Web (UF483) 45

MÓDULO FORMATIVO EM CONTEXTO REAL DE TRABALHO 48

PERFIL PROFISSIONAL

TIC001.A2_5

DESENVOLVIMENTO DE APLICAÇÕES WEB FULL-STACK

<table>
<tr><th colspan="5">PERFIL PROFISSIONAL</th></tr>
<tr><td>Código</td><td>TIC001.A2_5</td><td>Denominação</td><td colspan="2">DESENVOLVIMENTO DE APLICAÇÕES WEB FULL-STACK</td></tr>
<tr><td>Nível</td><td>5</td><td>Família Profissional</td><td colspan="2">Tecnologias de Informação e Comunicação (TIC)</td></tr>
<tr><td colspan="2">COMPETÊNCIA GERAL</td><td colspan="3">Desenvolver e implementar componentes de software para aplicações web, garantindo a sua verificação, documentação, alto desempenho, usabilidade, escalabilidade, segurança, integração das API e serviços, além de otimização e proteção contra vulnerabilidades para uma experiência eficiente e segura em diversos dispositivos e utilizadores.</td></tr>
<tr><td colspan="2" rowspan="5">UNIDADES DE COMPETÊNCIA</td><td>N.º</td><td>Denominação</td><td>Código</td></tr>
<tr><td>1</td><td>Desenvolver componentes de software no ambiente do cliente para aplicações web.</td><td>UC372_5</td></tr>
<tr><td>2</td><td>Desenvolver componentes de software orientadas à execução no backend de aplicações web.</td><td>UC373_5</td></tr>
<tr><td>3</td><td>Desenvolver aplicações web de alto desempenho, usabilidade e escalabilidade.</td><td>UC374_5</td></tr>
<tr><td>4</td><td>Implementar as API e serviços web para integração entre sistemas, otimizando o desempenho e a segurança em aplicações web.</td><td>UC375_5</td></tr>
<tr><td colspan="2" rowspan="3">AMBIENTE PROFISSIONAL</td><td colspan="3">Âmbito profissional:

O profissional exerce a sua atividade no departamento de informática vocacionada para o desenvolvimento de aplicações web, em entidades públicas ou privadas, independentemente da sua dimensão ou forma jurídica, atuando por conta própria ou de outrem. O seu trabalho pode estar subordinado, funcional e/ou hierarquicamente, a um superior e, ocasionalmente, pode ter funcionários sob a sua responsabilidade, seja de forma sazonal ou permanente. No desempenho da sua função aplica os princípios da acessibilidade universal, conforme a regulamentação vigente.</td></tr>
<tr><td colspan="3">Setores:

O profissional atua no setor dos serviços, no subsetor da análise, conceção e programação de aplicações informáticas com tecnologias web, bem como em qualquer outro setor que utilize sistemas informáticos para a gestão ou condução do seu processo de negócio através da Internet.</td></tr>
<tr><td colspan="3">Ocupações e postos de trabalho relacionados:

CNP 2010:

2512 Programador de software

2513 Programador web e multimédia.

Sugestões:

Desenvolvedor Full-Stack.</td></tr>
</table>

UNIDADES DE COMPETÊNCIA (UC)

UC 1: DESENVOLVER COMPONENTES DE SOFTWARE NO AMBIENTE DO CLIENTE PARA APLICAÇÃO WEB	
Código: UC372_5	Nível: 5

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

EC 1: Preparar o ambiente de desenvolvimento web, instalando, configurando e adaptando as ferramentas necessárias, para o desenvolvimento de páginas web.

CD 1.1. As ferramentas de desenvolvimento são analisadas e selecionadas conforme as especificações do projeto, considerando funcionalidades, características e tecnologias.

CD 1.2. A ferramenta de desenvolvimento escolhida é instalada considerando os requisitos mínimos e recomendados do sistema, garantindo compatibilidade com o sistema operativo e as versões suportadas.

CD 1.3. A ferramenta de desenvolvimento é configurada, através da modificação de menus e opções, seguindo a sua documentação técnica e as especificações do projeto.

CD 1.4. Após a instalação são realizados testes para validar a operacionalidade das funcionalidades essenciais, assegurando que o ambiente está corretamente configurado antes do início do desenvolvimento.

CD 1.5. A ferramenta de desenvolvimento é ajustada, monitorando o desempenho e realizando otimizações, para garantir a eficiência no desenvolvimento das páginas web.

CD 1.6. O código-fonte é armazenado e gerido num sistema de controle de versões (Git/GitHub), permitindo rastreamento de alterações, colaboração entre desenvolvedores e manutenção eficiente do histórico do projeto.

EC 2: Criar documentos web utilizando linguagens de marcação e tags para garantir a sua correta interpretação pelos diversos navegadores, em conformidade com as especificações de design da organização.

CD 1.1. Os documentos web são estruturados com elementos predefinidos da linguagem de marcação, seguindo as especificações de design e layout da organização.

CD 1.2. Os documentos web são codificados, manipulando elementos e atributos das linguagens de marcação padrão, com referência às melhores práticas de codificação, para garantir a funcionalidade e legibilidade do código.

CD 1.3. Os documentos web dinâmicos são implementados integrando linguagens de programação em linguagens de marcação, de acordo com os requisitos de comunicação entre cliente e servidor, para garantir a interatividade necessária.

CD 1.4. Os formulários web são desenvolvidos utilizando recursos da linguagem de marcação, com referência à estrutura de gestão de dados inseridos pelo utilizador, para garantir a integridade e segurança da informação.

CD 1.5. Os formulários web são validados pela integração de componentes de software para tratamento de eventos, com referência aos padrões de segurança e usabilidade, para garantir a funcionalidade correta em todas as interações do utilizador.

CD 1.6. Os metadados são definidos por meio da manipulação de tags de cabeçalho, com referência às necessidades de rastreabilidade de conteúdo pelos motores de busca, para garantir a otimização dos documentos web para SEO (Search Engine Optimization).

CD 1.7. Os documentos web são otimizados para carregamento rápido, com referência ao desempenho de rede e melhores práticas de SEO, para garantir que o tempo de resposta da página seja reduzido.

EC 3: Desenvolver componentes de software utilizando linguagens de script, em conformidade com as especificações de design de software da organização, para execução em navegadores web.

CD 2.1. Os componentes de software são desenvolvidos de forma modular e ordenada, com

referência aos padrões de design, para garantir a reutilização e a escalabilidade do código.

CD 2.2. Os componentes de software são codificados utilizando linguagens de script, com referência ao modelo de objetos do documento web, para garantir a usabilidade, acessibilidade e compatibilidade com os navegadores mais utilizados.

CD 2.3. Os eventos em documentos web são geridos utilizando recursos específicos da linguagem de marcação, com referência aos requisitos funcionais do projeto, para garantir a interatividade e o processamento adequado de cada evento.

CD 2.4. Os testes unitários são implementados definindo parâmetros de entrada e saídas esperadas para cada componente de software, com referência às melhores práticas de teste, para garantir a funcionalidade correta do código.

CD 2.5. O código-fonte é documentado detalhadamente, com referência à finalidade e funcionalidade de cada componente, para garantir a clareza do código e a fácil manutenção futura.

CD 2.6. O código dos componentes de software é otimizado para desempenho, com referência aos testes de velocidade e utilização de recursos, para garantir que os componentes operem com eficiência em diversos dispositivos e condições de rede.

EC 4: Desenvolver folhas de estilo utilizando ferramentas específicas de edição e pré-processamento, aplicando-as a documentos web para modificar a aparência dos seus elementos.

CD 3.1. As folhas de estilo são implementadas utilizando linguagens de marcação padrão, com referência às especificações de design da organização, para garantir a consistência visual nos documentos web.

CD 3.2. As folhas de estilo são codificadas através de ferramentas de pré-processamento, com referência às boas práticas de codificação e validação, para garantir a eficiência e manutenção das folhas de estilo.

CD 3.3. As folhas de estilo são associadas a documentos web, com referência aos layouts especificados pela organização, para garantir a aparência visual consistente e de alta qualidade.

CD 3.4. As folhas de estilo são validadas utilizando ferramentas de verificação de sintaxe, com referência aos padrões de compatibilidade e usabilidade, para garantir a sua correta aplicação em documentos web.

CD 3.5. As folhas de estilo são testadas em dispositivos móveis, com referência à responsividade do design, para garantir que os documentos web se ajustem corretamente a diferentes tamanhos de tela e resoluções.

CD 3.6. As folhas de estilo são configuradas para utilização em diferentes temas de cor e personalização de estilo, com referência às especificações de design da organização, para garantir uma experiência do utilizador adaptável e inclusiva.

EC 5: Implementar técnicas de otimização de desempenho em aplicações web, utilizando boas práticas de desenvolvimento, para garantir a eficácia e a velocidade da aplicação.

CD 5.1. O desempenho da aplicação web é monitorado utilizando ferramentas de análise de desempenho, com referência à carga de recursos e tempo de resposta, para identificar e resolver obstáculos de desempenho.

CD 5.2. A aplicação web é otimizada através da redução do tamanho dos arquivos, com referência à compressão de imagens, minificação de scripts e estilos, para melhorar a velocidade de carregamento.

CD 5.3. O código da aplicação web é otimizado utilizando técnicas de lazy loading e armazenamento em cache, com referência à utilização eficiente de recursos, para reduzir o tempo de carregamento e melhorar a experiência do utilizador.

CD 5.4. As consultas a base de dados são otimizadas com referência à utilização de índices, query caching e técnicas de redução de chamadas desnecessárias, para garantir a eficiência e o rápido acesso aos dados.

CD 5.5. A aplicação web é testada utilizando ferramentas de análise de tempo de resposta e

consumo de memória, com referência à melhoria contínua da experiência do utilizador, para garantir que o desempenho atenda aos padrões de qualidade estabelecidos.

Contexto Profissional

Meios de produção

Ferramentas de escritório. Equipamento informático e periféricos. Ferramentas multimédia. Sistemas operativos e parâmetros de configuração. Ferramentas de desenvolvimento rápido, layout, gráficos e animação. Máquinas virtuais. Navegadores. Protocolos de comunicação. Ambientes de desenvolvimento integrados. Ferramentas de depuração e teste. Componentes de software já desenvolvidos e/ou distribuídos por empresas de TI. Ambientes de desenvolvimento para aplicações locais com tecnologia Web. Estruturas de design. Servidores Web. Servidores de aplicações. Sistemas de segurança. Mecanismos de base de dados.

Produtos e resultados

Documentos web desenvolvidos com linguagens de marcação e tags, componentes de software desenvolvidos com linguagens de script, folhas de estilo para documentos web e componentes multimédia desenvolvidos com linguagens de programação e ferramentas.

Informação utilizada ou gerada

Documentação associada aos componentes desenvolvidos. Documentação do curso de formação. Design e especificações da aplicação. Padrões de design. Visão global do sistema a implementar, entregar e explorar. Manuais de utilizador e de operação para sistemas informáticos. Manuais de operação de software. Manuais das ferramentas de desenvolvimento utilizadas. Documentação para operar o ambiente do cliente. Suporte de assistência técnica. Regulamentações aplicáveis relativas à propriedade intelectual e aos direitos de autor. Manuais de usabilidade. Modelos de trabalho. Especificações de design gráfico corporativo. Regulamentação sobre a proteção ambiental, especialmente sobre a produção e gestão de resíduos e solos contaminados. Normas sobre prevenção de riscos profissionais.

UC2: DESENVOLVER COMPONENTES DE SOFTWARE ORIENTADAS À EXECUÇÃO NO BACKEND DE APLICAÇÕES WEB	
Código: UC373_5	Nível: 5

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

EC 1: Desenvolver componentes de software para execução em servidor web, utilizando ferramentas e linguagens específicas, em conformidade com as especificações de design de software estabelecidas pela organização.

CD 1.1. O servidor de aplicações web são instalados, modificando os ficheiros de configuração, cumprindo os protocolos de comunicação web padrão e permitindo a execução dos elementos de software no lado do servidor.

CD 1.2. Os componentes de software são estruturados, separando o código da camada de apresentação do código da lógica de negócio.

CD 1.3. Os componentes de software da lógica de negócio são desenvolvidos utilizando diferentes tecnologias de programação, codificando de forma modular e ordenada, seguindo os padrões de design estabelecidos pela organização e criando elementos reutilizáveis.

CD 1.4. Os testes unitários são codificados, definindo os parâmetros de entrada e a saída esperada para cada componente de software.

CD 1.5. O controlo de acesso aos componentes de software que implementam a lógica de negócio é desenvolvido de acordo com a política de segurança da organização, utilizando mecanismos de autenticação e autorização.

CD 1.6. O desempenho dos componentes de software no servidor é monitorado utilizando ferramentas específicas, com referência à capacidade de resposta e utilização de recursos, para garantir a escalabilidade e o desempenho da aplicação web.

EC 2: Desenvolver componentes de software para aceder a base de dados, utilizando tecnologias de desenvolvimento de software para integrar conteúdo na aplicação web.

CD 2.1. Os componentes de software para acesso e manipulação de dados são desenvolvidos para garantir a integridade da informação, seguindo os procedimentos, regras e princípios de verificação estabelecidos pela organização na fase de projeto,

CD 2.2. As ligações a base de dados são configuradas através da modificação dos atributos de configuração e implementação exigidos pelo sistema de gestão de base de dados, garantindo a segurança, o desempenho das aplicações web e o acesso simultâneo à informação.

CD 2.3. As transações com base de dados são implementadas utilizando padrões para definir, consultar e manipular dados, garantindo que a segurança e a integridade da informação sejam mantidas.

CD 2.4. Os componentes de software para manipulação de informação são desenvolvidos utilizando técnicas que permitem relacionar os dados com uma estrutura lógica de entidades, simplificando o código-fonte e facilitando a sua manutenção.

CD 2.5. Os testes unitários são codificados, definindo os parâmetros de entrada e a saída esperada de cada componente de software que acede a base de dados.

CD 2.6. O código-fonte é codificado, detalhando a finalidade e a funcionalidade de cada componente de acesso à base de dados, para permitir a utilização de ferramentas de geração automática de documentação.

EC 3: Desenvolver componentes de software de integração, utilizando ferramentas padrão para permitir a integração dos desenvolvimentos no ambiente do servidor com aplicações de terceiros.

CD 3.1. Os componentes de software de integração são desenvolvidos, utilizando os mecanismos, protocolos e ferramentas web padrão envolvidos na publicação e consumo de serviços, para permitir que sejam consumidos por aplicações de terceiros e facilitar o acesso a

funcionalidades externas.

CD 3.2. Os testes unitários e de integração são codificados, utilizando frameworks de teste automatizado, para definir os parâmetros de entrada e a saída esperada de cada componente.

CD 3.3. O código-fonte é documentado, detalhando a finalidade e a funcionalidade de cada componente de integração, para permitir a utilização de ferramentas de geração automática de documentação e garantir rastreabilidade dentro do pipeline de desenvolvimento.

CD 3.4. Os componentes de software de integração são continuamente testados e integrados ao repositório de código, através de ferramentas de integração contínua (CI/CD), para garantir consistência no desenvolvimento e integração eficiente com aplicações de terceiros.

CD 3.5. A compatibilidade dos componentes de software de integração com diferentes ambientes (on-premise, cloud e híbridos) é validada, utilizando estratégias de containerização e orquestração (Docker, Kubernetes), para assegurar a escalabilidade e portabilidade dos serviços.

EC 4: Monitorizar o desempenho de aplicações web no ambiente do servidor, utilizando ferramentas de análise e técnicas de otimização, em conformidade com os requisitos de escalabilidade e segurança da organização.

CD 4.1. O consumo de recursos dos componentes de software no servidor é analisado utilizando ferramentas de monitoramento de desempenho e observabilidade (ex: Prometheus, Grafana), para garantir eficiência e escalabilidade da aplicação.

CD 4.2. O monitoramento contínuo do desempenho da aplicação é realizado, por meio de ferramentas analíticas, para permitir a detecção e resolução proativa de obstáculos e falhas.

CD 4.3. Os logs do servidor são analisados, utilizando ferramentas de auditoria e monitoramento (ex: Elastic Stack, Splunk), para permitir a detecção proativa de falhas, ameaças de segurança e padrões de tráfego suspeitos.

CD 4.4. A segurança do ambiente do servidor é reforçada com práticas de endurecimento do sistema, aplicação contínua de patches de segurança, controlo de acessos e auditorias regulares, para mitigar vulnerabilidades e riscos cibernéticos.

CD 4.5. A implementação de técnicas de cache, compressão de dados e otimização de requisições ao servidor é realizada para reduzir latência e melhorar a experiência do utilizador.

CD 4.6. As estratégias de backup e recuperação de desastres são implementadas para garantir a integridade dos dados e a continuidade dos serviços, minimizando impactos de falhas no ambiente do servidor.

EC 5: Otimizar a execução de aplicações web no ambiente do servidor, utilizando técnicas de caching, balanceamento de carga e escalabilidade, para melhorar o desempenho e a eficiência dos sistemas.

CD 5.1. A implementação de técnicas de cache, compressão de dados e otimização de requisições ao servidor é realizada para reduzir latência e melhorar a experiência do utilizador.

CD 5.2. Os mecanismos de balanceamento de carga são configurados e ajustados para distribuir eficientemente o tráfego entre servidores e garantir alta disponibilidade e estabilidade do sistema.

CD 5.3. As estratégias de escalabilidade horizontal e vertical são aplicadas conforme a procura para assegurar que a aplicação web possa lidar com variações de carga sem degradação do desempenho.

CD 5.4. A utilização de conexões persistentes e otimização de consultas às bases de dados é implementada para minimizar tempos de resposta e reduzir o consumo de recursos do servidor.

CD 5.5. A configuração e otimização de servidores web e de aplicação são efetuadas para garantir tempos de resposta reduzidos e melhor aproveitamento dos recursos computacionais.

CD 5.6. Os obstáculos de desempenho são identificados e resolvidos através da otimização do código e da configuração do ambiente do servidor, considerando práticas DevOps, caching e técnicas de balanceamento de carga, para garantir estabilidade e eficiência.

Contexto Profissional

Meios de produção

Sistemas operativos e parâmetros de configuração. Equipamento informático e periféricos. Ferramentas de transferência de ficheiros (sincronização de conteúdos). Ferramentas de escritório. Ferramentas SOA. Ferramentas de desenvolvimento e depuração. Componentes de terceiros. Gestores de protocolo. Linhas de comunicação. Servidores Web. Servidores de aplicações. Sistemas de gestão de bases de dados. Máquinas virtuais. Navegadores atuais e novos conceitos tecnológicos. Sistemas de segurança. Ferramentas de controlo de alterações. Ferramentas de controlo de versão.

Produtos e resultados

Código executável dos componentes de software. Código-fonte dos componentes de software. Documentos estáticos e dinâmicos cujo conteúdo é código-fonte a interpretar. Documentação do desenvolvimento realizado. Componentes da camada do servidor.

Informação utilizada ou gerada

Manuais de utilizador e de operação para sistemas informáticos. Manuais de operação de software. Visão global do sistema a implementar, entregar e explorar. Padrões de design. Manuais das linguagens de programação utilizadas. Materiais do curso de formação. Regulamentação sobre proteção ambiental, especialmente sobre a produção e gestão de resíduos e solos contaminados. Design técnico definido. Suporte de assistência técnica. Modelos de trabalho. Documentos de desenvolvimento dos componentes produzidos. Normas sobre prevenção de riscos profissionais.

UC3: DESENVOLVER APLICAÇÕES WEB DE ALTO DESEMPENHO, USABILIDADE, E ESCALABILIDADE	
Código: UC374_5	Nível: 5

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

EC 1: Verificar a conformidade dos componentes de software com os requisitos funcionais e arquiteturais definidos no projeto, utilizando a documentação das visões 4+1, para garantir a rastreabilidade e estruturação da aplicação web.

CD 1.1. Os componentes de software são analisados através da verificação da visão de casos de uso e seus diagramas, com referência às especificações de requisitos funcionais, para garantir que o comportamento esperado seja validado.

CD 1.2. Os componentes de software são testados executando os casos de teste definidos, com referência ao design da interface de utilizador, para assegurar a conformidade com os requisitos funcionais.

CD 1.3. A arquitetura do software é validada através da análise de diagramas de pacotes e classes, com referência à estrutura lógica, dependências e execução de testes de carga e performance, para garantir que o design do software seja coerente, escalável e eficiente.

CD 1.4. O arranjo físico dos componentes de software nos servidores é verificado utilizando diagramas de implementação e análise de infraestrutura, com referência às dependências, requisitos de rede e capacidades de infraestrutura, para assegurar a coerência com a arquitetura definida e garantir que os recursos sejam adequadamente dimensionados.

CD 1.5. A documentação detalhada das visões 4+1 é mantida e atualizada regularmente, incluindo diagramas de casos de uso, pacotes, classes e implementação, com a realização de revisões constantes, para garantir que todas as perspetivas da arquitetura de software estejam alinhadas com os requisitos funcionais e não funcionais.

EC 2: Gerir e controlar alterações em aplicações web, utilizando ferramentas de controlo de versões centralizadas e descentralizadas, para garantir a rastreabilidade do código e a colaboração eficiente da equipa de desenvolvimento.

CD 2.1. As ramificações de desenvolvimento são criadas e geridas no repositório de código-fonte, com referência às diretrizes do projeto, para organizar o fluxo de trabalho colaborativo.

CD 2.2. As alterações são registadas no histórico do repositório, com referência às boas práticas de controlo de versão, para documentar a evolução do software.

CD 2.3. A integração de alterações é realizada através da fusão de branches, resolvendo conflitos e aplicando políticas de revisão, para manter a consistência do código-fonte.

CD 2.4. As revisões de código são conduzidas antes da integração, utilizando ferramentas de revisão colaborativa, para garantir a conformidade com os padrões estabelecidos.

CD 2.5. As revisões de código são conduzidas antes da integração, utilizando ferramentas de revisão colaborativa e automação de revisão de segurança, com análise de vulnerabilidades, para garantir a conformidade com os padrões estabelecidos e a segurança do código.

EC 3: Testar o código da aplicação web, utilizando ferramentas de teste, para formar uma rede de segurança e garantir a qualidade do desenvolvimento, a documentação e a produtividade.

CD 3.1. A qualidade do software é revista através da análise estática de código, com referência às métricas de software, para identificar e corrigir problemas de qualidade.

CD 3.2. Os métodos em classes de produção são testados utilizando técnicas de design de casos de teste, com referência às métricas de cobertura de teste, para assegurar a sua robustez.

CD 3.3. Os testes automatizados são implementados para validar o comportamento da aplicação, utilizando frameworks de teste, para reduzir erros e melhorar a manutenção do software.

CD 3.4. A ferramenta de testes unitários é utilizada, aplicando as técnicas de caixa negra e caixa branca com duplas, para desenvolver testes dinâmicos,

CD 3.5. A arquitetura do software é testada através da execução de testes dinâmicos e de performance, incluindo testes de carga e stress, utilizando ferramentas de teste de integração e de análise de desempenho, para verificar a conformidade com requisitos não funcionais e garantir a escalabilidade da aplicação.

EC 4: Publicar e manter a aplicação web em produção, utilizando processos de integração e entrega contínua (CI/CD), para garantir a sua estabilidade e evolução.

CD 4.1. Os artefatos de software são gerados e preparados para implantação, utilizando pipelines de automação, e padronização da entrega do software.

CD 4.2. O ambiente de produção é configurado ajustando servidores web e base de dados, com referência aos requisitos da aplicação, para garantir a execução adequada.

CD 4.3. A aplicação web é monitorizada em produção, utilizando ferramentas de logging e análise de desempenho, para garantir disponibilidade e identificar melhorias.

CD 4.4. A segurança da aplicação é reforçada implementando autenticação, criptografia e proteção contra ataques, com monitoramento contínuo de vulnerabilidades após a publicação, para mitigar riscos emergentes e garantir conformidade com padrões de segurança.

EC 5: Configurar e otimizar ambientes de produção para aplicações web, utilizando ferramentas e técnicas adequadas, para garantir desempenho, segurança e escalabilidade.

CD 5.1. O ambiente de produção é configurado instalando e ajustando servidores web e de aplicação, com referência às necessidades da aplicação, para garantir a correta execução e disponibilidade dos serviços.

CD 5.2. A segurança da aplicação é reforçada aplicando políticas de firewall, criptografia de dados e autenticação de acessos, com referência às boas práticas de segurança web, para proteger contra ameaças e ataques.

CD 5.3. A segurança dos componentes de software de acesso a base de dados é auditada utilizando ferramentas de análise de vulnerabilidades e monitoramento de riscos, com referência aos padrões de segurança da organização, para garantir a proteção dos dados e a integridade das transações.

CD 5.4. O desempenho da aplicação é otimizado utilizando mecanismos de cache, balanceamento de carga e compressão de dados, com referência às exigências de escalabilidade, para melhorar o tempo de resposta e suportar múltiplos acessos simultâneos.

CD 5.5. O monitoramento da aplicação web é implementado utilizando ferramentas de logging e métricas de desempenho, com referência ao comportamento da aplicação em produção, para identificar e corrigir falhas de forma proativa.

CD 5.6. A automação do processo de deployment é realizada configurando pipelines de integração e entrega contínua (CI/CD), com referência às ferramentas de gestão de versões e infraestrutura como código, para garantir atualizações seguras e ágeis.

Contexto Profissional

Meios de produção

Máquinas virtuais. Ferramentas de depuração. Ferramentas de teste. Equipamento informático e periféricos. Protocolos de comunicação. Sistemas operativos. Linguagens estruturadas. Linguagens orientadas a objetos. Linguagens de marcação. Navegadores Web. Servidores Web. Componentes de terceiros. Linguagens de estilo. Ferramentas de escritório. Ferramentas de desenvolvimento. Ambientes de desenvolvimento integrados. Ferramentas de gestão de projetos. Ferramentas de controlo de alterações. Ferramentas de integração contínua. Ferramentas de documentação. Sistemas de gestão de bases de dados.

Produtos e resultados

Controlo de alterações de aplicações web geridas. Componentes de software que foram testados e verificados para satisfazer os requisitos funcionais neles definidos. Código de aplicação web testado, com qualidade garantida e documentado. Aplicação web implantada.

Informação utilizada ou gerada

Especificações de design gráfico corporativo. Visão global do sistema a implementar, entregar e explorar. Manuais das ferramentas de desenvolvimento utilizadas. Manuais de utilizador e de operação para sistemas informáticos. Design e especificações da aplicação. Manuais de usabilidade. Documentação para operar o ambiente do cliente. Documentação para operar o ambiente do servidor. Manuais de operação de software. Regulamentação sobre proteção ambiental, especialmente sobre a produção e gestão de resíduos e solos contaminados. Suporte de assistência técnica. Modelos de trabalho. Normas sobre prevenção de riscos profissionais.

UC4: IMPLEMENTAR AS API E SERVIÇOS WEB PARA INTEGRAÇÃO ENTRE SISTEMAS, OTIMIZANDO O DESEMPENHO E A SEGURANÇA EM APLICAÇÕES WEB	
Código: UC375_5	Nível: 5

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

EC 1: Projetar e desenvolver API para integração de sistemas, garantindo eficiência, segurança e escalabilidade.

CD 1.1. As API são projetadas utilizando padrões como REST e GraphQL, com referência aos requisitos funcionais e de escalabilidade, para garantir a comunicação eficiente entre sistemas.

CD 1.2. Os endpoints da API são definidos e documentados utilizando ferramentas adequadas, com referência aos princípios de design de API, para garantir a clareza e padronização da interface.

CD 1.3. O consumo de API é implementado em aplicações web e móveis, com referência à compatibilidade e segurança, para garantir a integração confiável entre sistemas.

CD 1.4. Os métodos HTTP e os formatos de resposta são escolhidos adequadamente, com referência aos princípios RESTful, para garantir a comunicação eficiente e intuitiva da API.

CD 1.5. A API é documentada utilizando ferramentas como Swagger ou Postman, com referência às melhores práticas de documentação, para garantir a acessibilidade e compreensão dos desenvolvedores.

EC 2: Implementar autenticação e controlo de acesso em API e serviços web para garantir a segurança da comunicação entre sistemas.

CD 2.1. Os mecanismos de autenticação são implementados utilizando padrões como OAuth 2.0, JWT e OpenID Connect, com referência às práticas de segurança, para garantir a proteção contra acessos não autorizados.

CD 2.2. A autorização de utilizadores e aplicações é configurada com base em perfis e permissões, com referência ao princípio de privilégio mínimo, para garantir o acesso restrito a dados sensíveis.

CD 2.3. A transmissão de dados entre clientes e servidores é protegida utilizando HTTPS e criptografia, com referência às diretrizes de segurança da web, para evitar intercetação de informações.

CD 2.4. Os tokens de autenticação são gerenciados com políticas de expiração e renovação, com referência às boas práticas de segurança, para garantir a proteção contínua do acesso à API.

CD 2.5. Os testes de segurança são aplicados para identificar e corrigir vulnerabilidades, com referência a ataques comuns como injeção SQL e cross-site scripting (XSS), para fortalecer a proteção da API.

EC 3: Integrar API de terceiros e serviços web para ampliar a funcionalidade das aplicações.

CD 3.1. As API de terceiros são consumidas utilizando protocolos e bibliotecas adequadas, com referência à compatibilidade e requisitos da aplicação, para ampliar funcionalidades sem comprometer o desempenho.

CD 3.2. As credenciais de acesso a API externas são armazenadas de forma segura, com referência às práticas de segurança de informações sensíveis, para evitar vazamento de dados.

CD 3.3. Os dados recebidos de API externas são validados e normalizados, com referência às regras de negócio da aplicação, para garantir a integridade e consistência das informações.

CD 3.4. O desempenho da comunicação com API externas é monitorado e otimizado, com referência à latência e ao consumo de recursos, para garantir eficiência no tempo de resposta das aplicações.

CD 3.5. As políticas de rate limiting e caching são implementadas na integração com API externas,

com referência às restrições de utilização e otimização de performance, para garantir a estabilidade da aplicação.

EC 4: Implementar mecanismos de versionamento e manutenção de API para garantir a evolução contínua dos serviços.

CD 4.1. O versionamento de API é definido e gerido utilizando estratégias como URI versioning e headers, com referência às boas práticas de manutenção, para evitar incompatibilidades entre versões.

CD 4.2. As atualizações e novas funcionalidades são implementadas sem comprometer versões anteriores, com referência à retrocompatibilidade, para garantir a continuidade dos serviços.

CD 4.3. O monitoramento e logging são configurados para capturar métricas e falhas da API, com referência a ferramentas de observabilidade, para garantir a análise e correção de erros em tempo real.

CD 4.4. Testes automatizados são aplicados para validar a integridade da API, com referência a frameworks de teste, para garantir que novas versões não introduzam regressões.

CD 4.5. O desempenho da API é otimizado utilizando estratégias como compressão de respostas e balanceamento de carga, com referência às práticas de escalabilidade, para garantir alta disponibilidade.

EC 5: Projetar e desenvolver API e microsserviços utilizando arquiteturas baseadas em RESTful e GraphQL, implementando práticas modernas como containers (Docker) e orquestração (Kubernetes), para garantir escalabilidade, flexibilidade e fácil integração com sistemas legados e de terceiros.

CD 5.1. A utilização de arquiteturas RESTful e GraphQ no design de API é implementada para garantir uma comunicação eficiente, flexível e de fácil consumo por sistemas internos e de terceiros.

CD 5.2. A implementação de microsserviços é realizada com foco na modularidade para promover a escalabilidade independente de componentes e permitir a fácil manutenção e evolução dos serviços.

CD 5.3. A aplicação de containers (Docker) e a orquestração via Kubernetes são configuradas para garantir a implantação automatizada, escalabilidade dinâmica e gestão eficiente de recursos no ambiente de produção.

CD 5.4. Os mecanismos de controlo de versão de API são aplicados para permitir a evolução contínua das funcionalidades e garantir a compatibilidade com versões anteriores e a integração com sistemas legados.

CD 5.5. As API e os microsserviços são monitorados em tempo real, utilizando ferramentas de observabilidade e análise de performance, para identificar obstáculos, otimizar o desempenho e garantir uma comunicação rápida e eficiente entre os sistemas.

CD 5.6. A implementação de API e microsserviços é realizada com técnicas avançadas de otimização de desempenho, como a adoção de gateways de API, caching distribuído e circuit breakers, para garantir alta disponibilidade, redução de latência e resiliência em ambientes de tráfego intenso e em sistemas com alta procura de integração em tempo real.

EC 6: Implementar práticas avançadas de otimização e acessibilidade em aplicações web modernas, utilizando técnicas como lazy loading, análise automatizada de acessibilidade

CD 6.1. A implementação de técnicas de otimização como lazy loading e compressão de imagens é realizada para reduzir os tempos de carregamento da aplicação e melhorar a experiência do utilizador em dispositivos com diferentes capacidades de rede e hardware.

CD 6.2. A conformidade com as normas de acessibilidade (como WCAG) é realizada através de ferramentas automatizadas de análise de acessibilidade para assegurar que a aplicação seja utilizável por pessoas com diferentes deficiências e dispositivos assistivos.

CD 6.3. A realização de testes de carga e análise de performance contínua é realizada para monitorar o desempenho da aplicação, identificando e corrigindo obstáculos e pontos de falha e assegurando ao utilizador uma experiência fluida em condições de alto tráfego.

CD 6.4. A atualização constante das bibliotecas e frameworks da aplicação, bem como a aplicação de patches de segurança, é realizada para garantir que a aplicação esteja protegida contra vulnerabilidades conhecidas e seja compatível com as melhores práticas de segurança.

EC 7: Aplicar medidas de segurança para proteção da aplicação web contra vulnerabilidades e ataques.

CD 7.1. As entradas de dados são validadas e desinfetadas, com referência à prevenção de ataques como injeção SQL e Cross-Site Scripting (XSS), para garantir a integridade das informações

CD 7.2. Os mecanismos de autenticação e autorização são implementados utilizando padrões como OAuth, JWT e autenticação multifator, com referência às diretrizes de segurança, para proteger o acesso a dados sensíveis

CD 7.3. A criptografia de dados em trânsito e em repouso é aplicada utilizando protocolos como TLS e AES, com referência às boas práticas de segurança, para garantir a proteção contra vazamento de informações.

CD 7.4. Os cabeçalhos de segurança HTTP são configurados corretamente, com referência à proteção contra ataques como Clickjacking e Cross-Site Request Forgery (CSRF), para reforçar a segurança do navegador.

CD 7.5. As auditorias de segurança e testes de penetração são realizados regularmente, com referência às vulnerabilidades do Top 10 Open Web Application Security Project (OWASP), para identificar e corrigir possíveis falhas antes de exploração maliciosa.

CD 7.6. A aplicação de testes contínuos de segurança, como análise de vulnerabilidades automáticas (ferramentas SAST e DAST) e implementação de técnicas de mitigação de riscos, como Content Security Policy (CSP) e Subresource Integrity (SRI), é realizada para detetar e corrigir vulnerabilidades em tempo real, garantindo a integridade da aplicação e a proteção contra ataques modernos (ex.: XSS, CSRF).

CD 7.7. A aplicação de criptografia de ponta a ponta (end-to-end encryption) é implementada para proteger os dados do utilizador durante a comunicação e garantir a privacidade e a integridade das informações transmitidas.

Contexto Profissional

Meios de produção

Equipamento informático e periféricos. Ferramentas de escritório e documentação técnica. Sistemas operativos e ferramentas de configuração de servidores. Plataformas de desenvolvimento para API e serviços web. Ferramentas de teste e depuração de API (Postman, Swagger, etc.). Servidores web e servidores de aplicações. Bancos de dados relacionais e não relacionais. Ferramentas para gerenciamento de API (API Gateway, proxies de API, etc.). Protocolos de comunicação (HTTP, WebSockets, REST, GraphQL, etc.). Ambientes de desenvolvimento integrados (IDE). Ferramentas de autenticação e segurança (OAuth, JWT, certificados SSL/TLS). Sistemas de controlo de versão (Git, GitHub, GitLab, etc.). Plataformas de cloud computing para serviços web (AWS, Azure, Google Cloud).

Produtos e resultados

API documentadas e desenvolvidas seguindo padrões de interoperabilidade. Serviços web implementados e testados para integração entre sistemas. Interfaces de comunicação entre diferentes aplicações e plataformas. Documentação técnica das API e serviços implementados. Autenticação e controlo de acesso aplicados às API. Logs e monitoramento das API para análise de desempenho e segurança.

Informação utilizada ou gerada

Especificações técnicas para desenvolvimento de API e serviços web. Padrões e boas práticas para desenvolvimento de API RESTful e GraphQL. Documentação de sistemas a integrar. Manuais e guias de implementação de serviços web. Regulamentações e normas de segurança da informação (GDPR, ISO 27001, etc.). Logs de requisições e respostas para depuração e análise de erros. Configurações de servidores e ambientes de execução para API. Políticas de autenticação, autorização e proteção de dados.

PROGRAMA FORMATIVO ASSOCIADO AO PERFIL PROFISSIONAL

TIC001.A2_5

DESENVOLVIMENTO DE APLICAÇÕES WEB FULL-STACK

PROGRAMA FORMATIVO DA QUALIFICAÇÃO PROFISSIONAL			
Código: TIC001.A2_5	Denominação	DESENVOLVIMENTO DE APLICAÇÕES WEB FULL-STACK	
Nível: 5	Família Profissional	Tecnologias de Informação e Comunicação	
Duração indicativa	1230 horas		
Unidades de Competência	N.º	Denominação	Código
	1	Desenvolver componentes de software no ambiente do cliente para aplicações web.	UC372_5
	2	Desenvolver componentes de software orientadas à execução no backend de aplicações web.	UC373_5
	3	Desenvolver aplicações web de alto desempenho, usabilidade e escalabilidade.	UC374_5
	4	Implementar APIs e serviços web para integração entre sistemas, otimizando o desempenho e a segurança em aplicações web.	UC375_5

UNIDADES FORMATIVAS TRANVERSAIS				
Denominação			Duração	Código
Inglês Técnico			60 horas	UF468
MÓDULOS FORMATIVOS			UNIDADES FORMATIVAS	
N.º	Denominação	Código	Denominação	Código
1	Desenvolvimento Frontend e Otimização de Interfaces Web (210 horas)	MF372_5	Otimização e Estruturação de Documentos Web (110 horas)	UF476
			Programação Frontend e Interatividade Web (100 horas)	UF477
2	Desenvolvimento de Aplicações Web no Ambiente do Servidor (210 horas)	MF373_5	Arquiteturas e Tecnologias para Desenvolvimento Web no Servidor (120 horas)	UF478
			Desenvolvimento e Integração de Aplicações Web no Servidor (90 horas)	UF479
3	Automação, Monitorização e Escalabilidade de Aplicações Web (180 horas)	MF374_5	Gestão e Automação de Processos de Entrega Web (90 horas)	UF480
			Monitorização e Escalabilidade de Aplicações Web (90 horas)	UF481
4	Integração de APIs, Otimização e Segurança em Aplicações Web (210 horas)	MF375_5	Integração e Gestão de APIs em Aplicações Web (110 horas)	UF482
			Segurança e Qualidade em Aplicações Web (100 horas)	UF483
MT_TIC001.A2_5		Módulo Formativo em Contexto Real de Trabalho (360 horas)		

UNIDADES FORMATIVAS TRANSVERSAIS

UNIDADE FORMATIVA : Inglês Técnico

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

RA1: Compreender mensagens orais e escritas no contexto profissional.

- **CA 1.1.** Interpretar mensagens orais e escritas gerais e específicas com correção, utilizando técnicas de dedução pelo contexto e identificação da ideia principal.
- **CA 1.2.** Reconhecer a terminologia e expressões específicas do sector em discursos, documentos e mensagens diversas.
- **CA 1.3.** Localizar informação relevante em textos técnicos aplicando estratégias de leitura rápida e leitura orientada.
- **CA 1.4.** Responder a questões orais e escritas com base em documentos técnicos e mensagens profissionais, demonstrando compreensão da informação essencial.
- **CA 1.5.** Relacionar vocabulário técnico com situações práticas do contexto profissional, através de exercícios de correspondência, preenchimento de lacunas ou produção de frases contextualizadas.

RA2: Produzir mensagens orais e escritas adequadas a diferentes contextos profissionais.

- **CA 2.1.** Produzir mensagens orais e escritas com estrutura clara, coerência e terminologia específica, ajustadas ao registo e interlocutor.
- **CA 2.2.** Elaborar documentos profissionais aplicando normas de cortesia e formalidade adequadas ao contexto.
- **CA 2.3.** Transmitir instruções e informações profissionais utilizando recursos linguísticos habituais e palavras-chave relevantes.
- **CA 2.4.** Participar em situações profissionais (ex: reuniões, conferência e apresentações etc.), utilizando expressões funcionais adequadas e linguagem específica do setor.
- **CA 2.5.** Redigir e-mails, memorandos ou relatórios simples em inglês técnico, respeitando convenções de formatação e utilizando vocabulário apropriado ao objetivo e ao destinatário.

RA3: Identificar normas de protocolo e regras de comunicação formal e informal em contexto profissional.

- **CA 3.1.** Identificar regras de comunicação formal e informal em diferentes contextos e suportes de comunicação.
- **CA 3.2.** Descrever procedimentos de protocolo e etiqueta profissional em eventos, reuniões e comunicações empresariais.
- **CA 3.3.** Distinguir características de discursos formais e informais com exemplos adequados ao meio profissional.
- **CA 3.4.** Demonstrar a aplicação prática de normas de etiqueta em ambientes profissionais, como ao participar em reuniões, eventos e videoconferências, respeitando o protocolo de comunicação e comportamento esperado.

RA4: Aplicar estratégias de comunicação eficaz na gestão de reclamações e resolução de conflitos.

- **CA 4.1.** Aplicar técnicas de comunicação assertiva na gestão de situações de conflito e reclamação em contexto profissional.
- **CA 4.2.** Utilizar vocabulário adequado e tom de voz apropriado em situações de tensão, respeitando o protocolo empresarial.

CA 4.3. Redigir respostas a reclamações seguindo normas de cortesia e estruturas formais específicas.

CA 4.4. Analisar situações comunicativas profissionais, identificando comportamentos adequados e inadequados segundo normas de etiqueta e protocolo.

CA 4.5. Demonstrar a utilização de técnicas de escuta ativa e feedback construtivo para compreender e resolver reclamações de forma eficaz, mantendo a calma e o profissionalismo.

CA 4.6. Redigir e-mails ou cartas de resposta a reclamações, utilizando uma abordagem clara, cortês e focada na resolução do problema, de acordo com as normas de comunicação formal e protocolos empresariais.

RA5: Demonstrar uma atitude proativa e adequada na comunicação em ambiente profissional.

CA 5.1. Adotar normas de relação e protocolo respeitando a formalidade e a cortesia em interações presenciais, telefónicas e escritas.

CA 5.2. Solicitar esclarecimentos e confirmar mensagens de forma educada e assertiva, promovendo a compreensão mútua.

CA 5.3. Responder a reclamações e queixas com atitude profissional e respeitosa, mantendo a fluidez e o respeito na comunicação.

CA 5.4. Manter uma postura ativa e positiva durante as interações profissionais, demonstrando interesse e comprometimento nas respostas e na solução de problemas.

CA 5.5. Utilizar expressões e gestos apropriados para garantir uma comunicação respeitosa e profissional, seja em ambientes presenciais, por telefone ou por escrito.

Conteúdos

1. Compreensão Oral e Escrita em Contexto Profissional

1.1. Técnicas de compreensão de mensagens orais e escritas:

1.1.1. Dedução pelo contexto e identificação da ideia principal e secundária.

1.1.2. Localização rápida de informações específicas (scanning e skimming).

1.2. Terminologia e expressões técnicas do sector:

1.2.1. Vocabulário habitual em ambiente profissional.

1.2.2. Palavras-chave e expressões mais utilizadas em discursos, documentos e mensagens.

1.3. Tipos de mensagens e discursos profissionais:

1.3.1. Mensagens diretas, telefónicas, gravadas, radiofónicas e televisivas.

1.3.2. Instruções sobre operações e tarefas do posto de trabalho.

1.3.3. Solicitação de informações gerais e específicas.

2. Produção Oral e Escrita em Ambiente Profissional

2.1. Estruturação e clareza na produção de mensagens:

2.1.1. Estratégias para manter a fluidez na conversação (exemplos e perguntas de confirmação).

2.1.2. Estratégias de clarificação do discurso.

2.2. Elaboração de mensagens profissionais:

2.2.1. Redação de instruções de trabalho e comunicações internas.

2.2.2. Comunicação de mau funcionamento de equipamentos ou aplicações.

2.2.3. Produção de mensagens telefónicas, presenciais ou gravadas, com registo e terminologia apropriados.

2.3. Resposta a reclamações e resolução de problemas:

2.3.1. Utilização de expressões adequadas para gerir queixas.

2.3.2. Estruturas formais e corteses para comunicação de resposta.

3. Normas de Protocolo e Comunicação Formal e Informal

3.1. Normas de comunicação profissional: Normas de cortesia e formalidade em diferentes contextos e suportes (presencial, telefónico, escrito).

3.2. Protocolo em ambiente profissional:

3.2.1. Normas de saudação, recepção e despedida.

3.2.2. Procedimentos de etiqueta em reuniões, eventos e comunicações empresariais.

3.3. Características dos discursos formais e informais: Exemplos práticos aplicados ao contexto profissional.

4. Gestão de Reclamações e Resolução de Conflitos

4.1. Comunicação assertiva em situações de conflito: Técnicas de gestão de reclamações com foco na assertividade.

4.2. Adaptação do tom de voz e vocabulário: Aplicação de vocabulário adequado e utilização do tom apropriado em situações de tensão.

4.3. Redação de respostas a reclamações: Estrutura formal, normas de cortesia e resolução clara.

5. Atitudes e Comportamentos na Comunicação Profissional

5.1. Normas de relação interpessoal e protocolo empresarial: Comportamento adequado em interações presenciais, telefónicas e escritas.

5.2. Técnicas para promover a compreensão mútua: Solicitação de esclarecimentos e confirmação de mensagens de forma educada e assertiva.

5.3. Gestão respeitosa de reclamações: Resposta a queixas com atitude profissional, respeitosa e fluída.

MÓDULOS FORMATIVOS (MF)

MF1: Desenvolvimento Frontend e Otimização de Interfaces Web		
Código: MF372_5	Nível: 5	Duração Mínima: **210** Horas
Associado à UC372_5: Desenvolver componentes de software no ambiente do cliente para aplicações web.		

Este MF é subdividido nas seguintes Unidades Formativas: caso necessário

		Código
■ Unidade Formativa 1:	Otimização e Estruturação de Documentos Web	UF476
■ Unidade Formativa 2:	Programação Frontend e Interatividade Web	UF477

Unidade Formativa 1: Otimização e Estruturação de Documentos Web (UF476)

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

RA1: Aplicar técnicas de instalação de ferramentas de desenvolvimento web

CA 1.1. Analisar ferramentas de desenvolvimento (IDEs ou editores de código, servidores locais, etc.) para identificar as suas propriedades, características e tecnologias suportadas, conforme requisitos do projeto.

CA 1.2. Selecionar uma ferramenta de desenvolvimento com base nas funcionalidades e características, garantindo adequação ao projeto.

CA 1.3. Instalar e configurar a ferramenta de desenvolvimento para garantir compatibilidade com as especificações do website.

CA 1.4. Realizar testes pós-instalação para garantir que as ferramentas funcionam de acordo com as necessidades do projeto.

CA 1.5. Monitorar o desempenho das ferramentas de desenvolvimento e aplicar ajustes para melhorar a eficiência, conforme métricas estabelecidas.

CA 1.6. Garantir a compatibilidade entre a ferramenta escolhida e os sistemas operacionais e versões suportadas pelo projeto.

CA 1.7. Documentar o processo de instalação e configuração, incluindo preparação do ambiente Git, registando as etapas e mudanças feitas para futuras manutenções.

RA2: Desenvolver documentos web que possam ser interpretados por diferentes browsers, utilizando linguagens de marcação e tags e obedecendo às especificações de design.

CA 2.1. Estruturar um documento web utilizando elementos predefinidos da linguagem de marcação e seguindo especificações de design e de layout.

CA 2.2. Codificar um documento web utilizando elementos e atributos de linguagens de marcação padrão, aplicando folhas de estilo, integrando componentes de script de software.

CA 2.3. Integrar elementos multimédia, inserindo tipos de conteúdo e aplicando estilos, fazendo utilização de atributos específicos de uma linguagem de marcação.

CA 2.4. Definir metadados manipulando as tags de cabeçalho e armazenando informações de documentos web.

RA3: Desenvolver folhas de estilo, utilizando as facilidades de ferramentas específicas de edição e pré-processamento.

CA 3.1. Criar folhas de estilo utilizando linguagens de marcação padrão, classes e coleções de regras, seguindo as normas de estruturação de estilos para web.

CA 3.2. Codificar folhas de estilo de forma otimizada, utilizando as capacidades de uma ferramenta de pré-processamento, conforme as especificações de sintaxe e estrutura.

CA 3.3. Ligar folhas de estilo a um documento web, aplicando a marcação de estilo para texto, imagens e cores de fundo, em conformidade com as especificações de design visual.

CA 3.4. Utilizar ferramentas de validação de folhas de estilo, verificando a conformidade do código com as regras de sintaxe da linguagem e garantindo que possam ser aplicadas em diferentes documentos web.

RA4: Desenvolver e estruturar documentos web utilizando linguagens de marcação, garantindo a sua funcionalidade e conformidade com as especificações de design da organização.

CA 4.1. Estruturar documentos web utilizando elementos predefinidos da linguagem de marcação, garantindo que o código seja funcional e atenda às especificações de design da organização.

CA 4.2. Codificar documentos web manipulando tags de linguagens de marcação padrão, seguindo as melhores práticas de codificação para garantir a legibilidade e a performance do código.

CA 4.3. Implementar formulários web utilizando recursos da linguagem de marcação, assegurando a integridade e segurança dos dados inseridos pelos utilizadores.

CA 4.4. Validar os documentos web quanto à usabilidade e acessibilidade, garantindo que o conteúdo seja visualmente consistente e acessível em diferentes dispositivos e navegadores.

CA 4.5. Integrar componentes interativos aos documentos web utilizando linguagens de marcação e de script, garantindo a funcionalidade adequada e uma boa experiência do utilizador.

RA5: Demonstrar postura profissional e proatividade na validação e otimização de documentos e componentes web, focando na melhoria contínua da experiência do utilizador e na adaptação às novas tecnologias.

CA 5.1. Avaliar e melhorar continuamente os documentos web, com atitude proativa para otimizar o tempo de carregamento e a experiência do utilizador, garantindo o desempenho da página.

CA 5.2. Demonstrar cuidado na validação de componentes web, identificando áreas de melhoria para assegurar que as soluções implementadas atendam aos padrões de usabilidade e acessibilidade.

CA 5.3. Manter uma atitude responsável ao implementar práticas de segurança e proteção de dados nos formulários web, garantindo que os dados dos utilizadores estejam sempre protegidos.

CA 5.4. Demonstrar comprometimento com a adaptação às novas tecnologias, buscando constantemente inovações que contribuam para a melhoria da performance e usabilidade de aplicações web.

CA 5.5. Manter uma postura ética e profissional ao colaborar com a equipe de desenvolvimento, garantindo o alinhamento de metas e a entrega de soluções de alta qualidade.

Conteúdos

1. Linguagens de marcações

1.1. Características das linguagens de marcação.
- 1.1.1 Definição e propósito.
- 1.1.2 Comparação entre linguagens de marcação (HTML, XML, etc.).
- 1.1.3 Papel das linguagens de marcação na web.

1.2. 1.2 Elementos da linguagem de marcação.
- 1.2.1 Elementos e tags básicas.
- 1.2.2 Atributos de elementos.
- 1.2.3 Elementos de estruturação (head, body, footer, etc.)

1.3. 1.3 Estrutura de um documento web.
- 1.3.1 Estrutura básica de um documento HTML.
- 1.3.2 Divisão em cabeçalho, corpo e rodapé.
- 1.3.3 Navegação pelas propriedades de um documento web.

1.4. 1.4 Objetos predefinidos da linguagem de marcação.
1.4.1 Definição e utilização de objetos predefinidos.
1.4.2 Exemplos: imagens, vídeos, formulários.

2. Estrutura e navegação de documentos web

2.1. Ligações e Comunicação entre Páginas.
2.1.1 Como criar links (elemento <a>).
2.1.2 Navegação entre diferentes páginas web.
2.1.3 Ligação de documentos externos (CSS, JavaScript).

2.2. Listas de dados e elementos.
2.2.1 Listas ordenadas e não ordenadas.
2.2.2 Implementação de tabelas para estruturação de dados.
2.2.3 Elementos de formatação e agrupamento.

2.3. Formulários e tabelas.
2.3.1 Estruturação de formulários (campos de entrada, botões).
2.3.2 Validação de formulários (obrigatório, tipo de dados, etc.).
2.3.3 Criando tabelas responsivas e interativas.

3. Apresentação e Estilo

3.1. Estilo de Apresentação.
3.1.1 Tamanho do texto, cor e tipo de letra.
3.1.2 Outras características de formatação de texto.
3.1.3 Ulitização de CSS para formatação avançada (margens, padding, etc.).

3.2. 3.2 Posicionamento e paginação de páginas web.
3.2.1 Controle de layout com CSS.
3.2.2 Posicionamento de elementos (float, flexbox, grid).
3.2.3 Técnicas de paginação (divisão de conteúdo em várias páginas).

4. Elementos multimédia e efeitos dinâmicos

4.1. Elementos multimédia.
4.1.1 Utilização de imagens, vídeos e áudio em documentos web.
4.1.2 Integração de mapas interativos e animações (Canvas, SVG).
4.1.3 Implementação de efeitos dinâmicos (hover, transitions).

4.2. 4.2 Mapas Interativos.
4.2.1 Introdução a utilização de mapas interativos com HTML5.
4.2.2 Exemplos de integração com APIs de mapas (Google Maps, OpenStreetMap).

5. Design responsivo e metadados

5.1. Design responsivo
5.1.1 Como garantir que as páginas sejam visualizadas em diversos dispositivos.
5.1.2 Utilização de media queries para design adaptativo.
5.1.3 Ferramentas e boas práticas para design responsivo.

5.2. 5.2 Metadados e configurações de cookies.
5.2.1 Definição e aplicação de metadados (tags <meta>).
5.2.2 Implementação de cookies em páginas web.
5.2.3 Considerações de SEO e acessibilidade.

6. Depuração e documentação de código

6.1. Depuração de Código.
6.1.1 Identificação e correção de erros em HTML e CSS.
6.1.2 Ferramentas de depuração (console do navegador, extensões, etc.).

6.2. 6.2 Documentação de Código
6.2.1 Práticas de documentação de código para manter a legibilidade.
6.2.2 Implementação de comentários em HTML e CSS.
6.2.3 Boas práticas para manter o código organizado e escalável.

Outras Capacidades Associadas à Unidade Formativa

- Capacidade de estruturar e organizar conteúdo web.

- Capacidade de codificar e depurar eficazmente.
- Capacidade de trabalhar com multimédia e componentes interativos.
- Capacidade de adotar práticas de design responsivo.
- Capacidade de implementar segurança em formulários web.

Unidade Formativa 2: Programação Frontend e Interatividade Web(UF477)

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

RA1: Desenvolver componentes de software para serem executados num web browser, utilizando linguagens de script e atendendo a requisitos funcionais.

- **CA 1.1.** Codificar um componente de software utilizando uma linguagem de script, verificando a sua usabilidade, acessibilidade e execução em diferentes navegadores da Web.
- **CA 1.2.** Codificar a finalidade e a funcionalidade de um componente, utilizando tags que permitam a utilização de ferramentas de geração automática de documentação.
- **CA 1.3.** Implementar mecanismos de gestão de eventos, utilizando recursos específicos de uma linguagem de marcação para capturar eventos e manipular componentes de software.
- **CA 1.4.** Realizar testes de unidade de código, definindo os parâmetros de entrada e a saída esperada de um componente de software.
- **CA 1.5.** Implementar mecanismos de interação dinâmica em componentes de software, utilizando linguagens de script como JavaScript, para atualizar conteúdo da página de forma assíncrona sem a necessidade de recarregar a página inteira.
- **CA 1.6.** Optimizar o desempenho de componentes de software, utilizando boas práticas de programação, como minificação de código, redução de requisições HTTP e utilização de cache, para garantir a eficiência e melhorar a experiência do utilizador.

RA2: Desenvolver documentos web que possam ser interpretados por diferentes browsers, utilizando linguagens de marcação e tags e obedecendo às especificações de design.

- **CA 2.1.** Implementar documentos web dinâmicos, utilizando mecanismos de integração de linguagens de programação em linguagens de marcação e permitindo a comunicação entre o cliente e o servidor.
- **CA 2.2.** Implementar formulários web, utilizando os recursos de uma linguagem de marcação para a gestão de formulários.
- **CA 2.3.** Validar formulários web, utilizando componentes de software de tratamento de eventos.
- **CA 2.4.** Integrar APIs externas em documentos web, utilizando tecnologias como AJAX ou Fetch API para obter e exibir dados dinâmicos de fontes externas, mantendo a interação fluida entre o cliente e o servidor.
- **CA 2.5.** Desenvolver componentes de interface interativos, como menus suspensos, sliders e modais, utilizando HTML, CSS e JavaScript, garantindo que sejam compatíveis com diferentes dispositivos e navegadores.
- **CA 2.6.** Implementar validação em tempo real de formulários web, utilizando JavaScript ou outras ferramentas de script para validar os dados inseridos pelo utilizador, fornecendo feedback imediato sobre erros de entrada.

RA3: Aplicar as principais plataformas tecnológicas e linguagens de programação utilizadas no desenvolvimento frontend de interfaces web.

- **CA 3.1.** Identificar as principais plataformas tecnológicas e linguagens de programação para o desenvolvimento frontend, descrevendo as suas funcionalidades e características.
- **CA 3.2.** Explicar as vantagens e desvantagens das principais linguagens e plataformas utilizadas, considerando compatibilidade com navegadores e dispositivos.
- **CA 3.3.** mplementar a integração de conteúdo multimédia numa página web, utilizando os tipos de recursos apropriados (gráficos, áudio, vídeo, animações), e garantindo a compatibilidade entre diferentes navegadores.
- **CA 3.4.** Codificar componentes interativos, como formulários e menus dinâmicos, utilizando HTML, CSS e JavaScript, assegurando que funcionem corretamente em diferentes dispositivos e plataformas.

CA 3.5. Implementar um mecanismo de validação de formulários web, utilizando JavaScript, para garantir a integridade dos dados inseridos pelo utilizador, sem recorrer a recarregamentos da página.

CA 3.6. Aplicar boas práticas de acessibilidade, implementando recursos como ARIA e navegação por teclado, para garantir que os componentes interativos sejam acessíveis a utilizadores com deficiência.

RA4: Demonstrar responsabilidade na aplicação de boas práticas de acessibilidade e usabilidade no desenvolvimento de interfaces web.

CA 4.1. Aplicar práticas de acessibilidade e usabilidade, utilizando ferramentas de validação para garantir a acessibilidade do código.

CA 4.2. Realizar testes de usabilidade e acessibilidade, ajustando os componentes da página conforme os feedbacks obtidos.

CA 4.3. Realizar testes de desempenho da página web, identificando pontos de melhoria e implementando ajustes para otimizar o tempo de carregamento e a experiência do utilizador, usando ferramentas de teste e depuração.

CA 4.4. Adotar práticas de desenvolvimento sustentável no frontend, aplicando metodologias como a utilização de código limpo e otimização de recursos, para garantir que o site tenha uma boa performance e seja sustentável a longo prazo.

CA 4.5. Garantir a compatibilidade entre diferentes navegadores e dispositivos móveis, realizando testes em várias plataformas e corrigindo inconsistências para proporcionar uma experiência consistente em todos os contextos de aplicação.

CA 4.6. Utilizar boas práticas de codificação, como modularização e reutilização de código, para garantir que o frontend seja escalável e de fácil manutenção, seguindo padrões de mercado e convenções de código.

Conteúdos

1. Tipos de recursos multimédia

1.1. Gráficos multimédia
- 1.1.1. Utilização de imagens, gráficos e ícones em documentos web.
- 1.1.2. Otimização de carregamento e acessibilidade.

1.2. 1.2. Áudio.
- 1.2.1. Implementação de áudio interativo com HTML5.
- 1.2.2. Integração de arquivos de áudio e controle de reprodução.

1.3. 1.3. Vídeo.
- 1.3.1. Utilização de vídeos em formatos compatíveis com a web.
- 1.3.2. Controle de reprodução e adaptabilidade.

1.4. 1.4. Animações.
- 1.4.1. Implementação de animações interativas com CSS3 e JavaScript.
- 1.4.2. Melhoria da experiência do utilizador com animações.

1.5. 1.5. Elementos Interativos.
- 1.5.1. Criação de menus dinâmicos e formulários.
- 1.5.2. Sliders e modais.

2. Ferramentas para processamento de conteúdos multimédia

2.1. Ferramentas de edição.
- 2.1.1. Softwares para criação e manipulação de gráficos multimédia.

2.2. 2.2. Conversores de áudio e vídeo.
- 2.2.1. Ferramentas para conversão e otimização de media.

2.3. 2.3. Bibliotecas JavaScript
- 2.3.1. Integração de bibliotecas para gráficos 3d, áudio e animações.

3. Linguagens de script

3.1. 3.1. Metodologia de programação estruturada.
3.1.1. Abordagem de programação modular e organizada.

3.2. 3.2. Características das linguagens de script no ambiente cliente.
3.2.1. Manipulação do DOM com javascript.

3.3. 3.3. Ambientes para desenvolvimento de linguagens de script.
3.3.1. Utilização de editores e plataformas.

4. Fundamentos do JavaScript

4.1. Variáveis e tipos de dados.
4.1.1. Definição de variáveis e tipos comuns.

4.2. 4.2. Operadores
4.2.1. Operadores aritméticos, lógicos e de comparação.

4.3. 4.3. Estruturas de controle.
4.3.1. Estruturas de decisão e repetição.

4.4. 4.4. Funções
4.4.1. Definição e utilização de funções.

5. Gestão de Eventos

5.1. Eventos em javascript.
5.1.1. Captura e manipulação de eventos.

5.2. 5.2. Manipulação de elementos do DOM.
5.2.1. Modificação do DOM com métodos javascript.

Outras Capacidades Associadas à Unidade Formativa

- Capacidade de desenvolver e integrar funcionalidades dinâmicas em web.
- Capacidade de otimizar desempenho e acessibilidade no frontend.
- Capacidade de trabalhar com APIs e integração de dados dinâmicos.

MF2: Desenvolvimento de Aplicações Web no Ambiente do Servidor		
Código: MF373_5	Nível: **5**	Duração: **21**0 Horas
Associado à UC373_5: Desenvolver componentes de software orientadas à execução no backend de aplicações web.		

Este MF é subdividido nas seguintes Unidades Formativas: caso necessário

		Código
■ Unidade Formativa 1:	Arquiteturas e Tecnologias para Desenvolvimento Web no Servidor	UF478
■ Unidade Formativa 2:	Desenvolvimento e Integração de Aplicações Web no Servidor	UF479

Unidade Formativa 1: Arquiteturas e Tecnologias para Desenvolvimento Web no Servidor (UF478)

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

RA1: Analisar as arquiteturas e tecnologias de desenvolvimento web do lado do servidor, descrevendo as suas características, diferenças e necessidades de configuração.

CA 1.1. Identificar arquiteturas e tecnologias de desenvolvimento web do lado do servidor, analisando as suas características, vantagens e desvantagens, em diferentes contextos de aplicação.

CA 1.2. Definir mecanismos de execução de código no servidor, especificando os requisitos de configuração e as diferenças entre os principais modelos de execução.

CA 1.3. Identificar linguagens e ferramentas de programação web, comparando as suas características, compatibilidades e aplicações no desenvolvimento web.

CA 1.4. Explicar as vantagens da separação entre a camada de apresentação e a lógica de negócio, descrevendo padrões de design e tecnologias que permitem essa separação.

CA 1.5. Analisar os fatores de desempenho e segurança nas arquiteturas web do lado do servidor, considerando impactos na escalabilidade e manutenção das aplicações.

CA 1.6. Comparar modelos de hospedagem e implantação de aplicações web no servidor, avaliando diferentes opções, como servidores dedicados, cloud computing e containers.

CA 1.7. Identificar arquiteturas e tecnologias de desenvolvimento web no servidor, listando as suas características e avaliando sua aplicabilidade em diferentes contextos.

CA 1.8. Explicar os princípios de funcionamento de servidores web e servidores de aplicação, descrevendo os mecanismos de comunicação e processamento de requisições.

CA 1.9. Comparar diferentes modelos de execução de código no servidor, apresentando suas vantagens e limitações em termos de desempenho e escalabilidade.

RA2: Desenvolver e configurar componentes de software para execução em servidores web, utilizando ferramentas e linguagens específicas.

CA 2.1. Instalar e configurar um servidor web e um servidor de aplicações, ajustando parâmetros de escuta e diretórios de documentos.

CA 2.2. Modificar ficheiros de configuração do servidor, respeitando protocolos de comunicação web padrão e garantindo a execução dos elementos de software no lado do servidor.

CA 2.3. Implementar e testar a separação entre a camada de apresentação e a lógica de negócio, organizando os componentes de software de acordo com padrões estabelecidos.

CA 2.4. Desenvolver componentes de software no lado do servidor, utilizando tecnologias de programação adequadas e respeitando boas práticas de modularidade e reutilização de código.

CA 2.5. Configurar mecanismos de autenticação e autorização no servidor, implementando protocolos como OAuth, JWT ou sessões para controlo de acesso a recursos.

CA 2.6. Monitorizar e otimizar o desempenho dos componentes do software no servidor, utilizando ferramentas de análise e ajustes de configuração para melhorar a escalabilidade e a eficiência.

RA3: Garantir a segurança, a escalabilidade e a manutenção dos componentes de software no ambiente do servidor, adotando boas práticas de desenvolvimento e monitorização

CA 3.1. Monitorizar o desempenho dos componentes de software no servidor, utilizando ferramentas específicas e analisando a utilização de recursos.

CA 3.2. Configurar mecanismos de autenticação e autorização, garantindo a segurança dos componentes de software conforme as políticas da organização.

CA 3.3. Aplicar estratégias de otimização de desempenho, utilizando técnicas de cache, balanceamento de carga e escalabilidade para garantir a eficiência do sistema.

CA 3.4. Implementar práticas de logging e auditoria, utilizando ferramentas de monitorização para detetar e registar falhas de segurança e anomalias no sistema.

CA 3.5. Aplicar atualizações e patches de segurança regularmente, garantindo a mitigação de vulnerabilidades e a conformidade com as melhores práticas de segurança.

CA 3.6. Configurar e testar estratégias de backup e recuperação de desastres, assegurando a integridade dos dados e a continuidade dos serviços em caso de falha no servidor.

RA4: Implementar mecanismos de execução de código no servidor, utilizando linguagens e frameworks adequados para o desenvolvimento web no lado do servidor.

CA 4.1. Configurar um ambiente de desenvolvimento para execução de código no servidor, instalando e integrando linguagens como PHP, Node.js, ou Python.

CA 4.2. Criar e testar rotas e controladores para manipulação de requisições HTTP, garantindo a correta comunicação entre cliente e servidor.

CA 4.3. Implementar uma API no servidor, utilizando padrões REST ou GraphQL, respeitando boas práticas de segurança e escalabilidade.

CA 4.4. Validar e tratar dados recebidos no servidor, prevenindo vulnerabilidades como injeção de código e ataques CSRF/XSS.

CA 4.5. Integrar bases de dados ao servidor, utilizando ORM ou drivers específicos, garantindo a persistência e a integridade dos dados.

CA 4.6. Implementar e testar middleware para gestão de autenticação, registo de logs e manipulação de erros, assegurando a segurança e a estabilidade da aplicação.

Conteúdos

1. Arquiteturas de desenvolvimento web no servidor

1.1 Modelos de arquitetura
1.1.1 MVC (Model-View-Controller)
1.1.2 Microservices e sua aplicação em arquiteturas escaláveis
1.1.3 Serverless: vantagens e desafios

1.2 Tecnologias populares
1.2.1 PHP e seu ecossistema
1.2.2 Node.js e JavaScript
1.2.3 Python e frameworks para web
1.2.4 Ruby e desenvolvimento web moderno

1.3 Principais frameworks
1.3.1 Laravel (PHP) e desenvolvimento ágil
1.3.2 Express (Node.js) e APIs rápidas
1.3.3 Django e Flask (Python) para aplicações escaláveis

1.4 Modelos de execução e processamento no servidor
1.4.1 Multithreading e concorrência
1.4.2 Event-driven e processamento assíncrono
1.4.3 Forking e isolamento de processos

2. Desenvolvimento e configuração de servidores web

2.1 Tipos de servidores
2.1.1 Apache HTTP Server e configurações comuns
2.1.2 Nginx e otimização para alto desempenho
2.1.3 Tomcat e aplicações Java EE
2.1.4 Node.js como servidor backend

2.2 Configuração e otimização de servidores
2.2.1 Gerenciamento de portas e listeners
2.2.2 Configuração de diretórios de documentos
2.2.3 Práticas de segurança para servidores web

2.3 Gestão de protocolos para comunicação web
2.3.1 HTTP/HTTPS
2.3.2 WebSockets para comunicação em tempo real

2.3..3 GRPC e comunicação entre microserviços

3. Segurança no desenvolvimento web

3.1 Autenticação e autorização de utilizadores
3.1.1 OAuth
3.1.2 JWT
3.1.3 Sessões

3.2 Prevenção de ataques
3.2.1 Injeção de código
3.2.2 CSRF (Cross-Site Request Forgery)
3.2.3 XSS (Cross-Site Scripting)

3.2.4. Proteção contra ataques de força bruta e hijacking

3.3 Configuração de HTTPS, headers de segurança
3.3.1 CORS (Cross-Origin Resource Sharing)
3.3.2 Content Security Policy

4. Performance, escalabilidade e monitoramento

4.1 Caching de aplicações web
4.1.1 Memcached
4.1.2 Redis

4.1.3.Otimização de consultas em bases de dados

4.2 Balanceamento de carga
4.2.1 Algoritmos e estratégias

4.2.2. Ferramentas como HAProxy e Nginx Load Balancer

4.3 Testes de carga e monitoramento
4.3.1 Ferramentas como JMeter e grafana

5. Integração com bases de dados

5.1 Tecnologias de persistência
5.1.1 MySQL
5.1.2 PostgreSQL
5.1.3 MongoDB

5.2 Utilização de ORM (Object-Relational Mapping)
5.2.1 Sequelize
5.2.2 SQLAlchemy
5.2.3 Doctrine

5.3 Transações, backup e recuperação de dados

6. Desenvolvimento de APIs

6.1 API RESTful e GraphQL
6.1.1 Estrutura e design

6.2 Boas práticas de segurança e escalabilidade em APIs

6.2.1. Versionamento e compatibilidade de APIs

6.3 Documentação de APIs
6.3.1 Swagger
6.3.2 Postman

7. Testes e monitoramento de aplicações

7.1 Ferramentas de testes
7.1.1 Jest
7.1.2 Mocha
7.1.3 PHPUnit

7.2 Monitoramento de servidor
7.2.1 New Relic
7.2.2 Prometheus
7.2.3 Grafana

Outras Capacidades Associadas à Unidade Formativa

- Capacidade de analisar e implementar arquiteturas de desenvolvimento web no servidor.
- Capacidade de configurar e gerenciar servidores web e aplicações.
- Capacidade de garantir a segurança e autenticação em aplicações web.
- Capacidade de desenvolver apis seguras e escaláveis.

Unidade Formativa 2: Desenvolvimento e Integração de Aplicações Web no Servidor (UF479)

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

RA1: Implementar componentes de software para execução num servidor web.

CA 1.1. Identificar arquiteturas e tecnologias de desenvolvimento web do lado do servidor, analisando as suas características, vantagens e desvantagens em diferentes contextos de aplicação.

CA 1.2. Definir componentes de software da camada de apresentação de código, incorporando-os em linguagens de marcação, utilizando e manipulando componentes de software que implementam a lógica de negócio.

CA 1.3. Realizar testes de unidade de código, definindo parâmetros de entrada e a saída esperada de cada componente de software.

CA 1.4. Desenvolver o controlo de acesso aos componentes de software que implementam a lógica de negócio, garantindo que esteja em conformidade com uma política de segurança, utilizando mecanismos de sessão, autenticação e autorização.

CA 1.5. Codificar a finalidade e a funcionalidade de um componente, utilizando tags que permitam a utilização de ferramentas de geração automática de documentação.

CA 1.6. Instalar um servidor de web e um servidor de aplicações, descarregando pacotes de aplicações e implementando os seus módulos e estrutura de diretórios, respeitando os protocolos de comunicação web padrão.

CA 1.7. Configurar um servidor web, editando os parâmetros de escuta do serviço e a localização dos documentos web.

CA 1.8. Configurar as funcionalidades de um servidor de aplicações, editando os parâmetros de uma aplicação para definir a sua estrutura, bibliotecas partilhadas e as políticas de segurança para a sua execução.

RA2: Compreender os princípios e tecnologias de servidores web e de aplicações, incluindo sua instalação, configuração e estruturação, conforme as especificações de segurança e desempenho da organização.

CA 2.1. Identificar as arquiteturas e tecnologias de servidores web e de aplicações, analisando as suas características, vantagens e desvantagens, e aplicando-as em diferentes contextos de desenvolvimento web.

CA 2.2. Descrever os protocolos de comunicação web utilizados nos servidores de aplicação, como HTTP, HTTPS, WebSockets, e suas implicações na segurança e desempenho das aplicações.

CA 2.3. Compreender os princípios fundamentais de segurança e de desempenho aplicados no ambiente de servidor, como criptografia, autenticação e controle de acesso.

CA 2.4. Analisar os requisitos de escalabilidade e alta disponibilidade em servidores de aplicações, identificando soluções adequadas para garantir a continuidade do serviço em diferentes cenários.

RA3: Demonstrar atitude proativa na otimização e segurança das aplicações web.

CA 3.1. Monitorizar o desempenho das aplicações web no ambiente do servidor, utilizando ferramentas de monitorização como Prometheus ou Grafana, para garantir a escalabilidade e otimização de recursos.

CA 3.2. Implementar práticas de segurança, como endurecimento do sistema, aplicação de patches de segurança e controle de acesso, para proteger as aplicações web contra vulnerabilidades e ameaças externas.

CA 3.3. Aplicar técnicas de otimização, como cache, compressão de dados e balanceamento de carga, para melhorar o desempenho e a experiência do utilizador em aplicações web.

CA 3.4. Validar a compatibilidade dos componentes de software em diferentes ambientes (on-premise, cloud, híbridos) através de estratégias de containerização e orquestração (Docker, Kubernetes), assegurando a escalabilidade e portabilidade.

RA4: Desenvolver componentes de software para execução em servidor web

CA 4.1. Instalar e configurar servidores de web (ex: Apache, Nginx) e servidores de aplicação (ex: Tomcat, Node.js), editando os arquivos de configuração de forma a respeitar os protocolos de comunicação web padrão e garantir a execução eficiente dos componentes de software.

CA 4.2. Codificar componentes de software para a camada de apresentação, utilizando linguagens de marcação como HTML, CSS e JavaScript, e integrando-os com a lógica de negócios.

CA 4.3. Desenvolver a lógica de negócios em linguagens como PHP, Java, Python ou Node.js, estruturando o código de maneira modular e reutilizável, aplicando boas práticas de design como SOLID e DRY.

CA 4.4. Criar e configurar a segurança nos componentes de software da lógica de negócios, utilizando técnicas de autenticação e autorização, como OAuth2, JWT ou RADIUS, em conformidade com as políticas de segurança da organização.

CA 4.5. Realizar testes unitários e de integração para validar a funcionalidade dos componentes de software, utilizando frameworks como JUnit, Mocha ou PHPUnit, e definindo os parâmetros de entrada e a saída esperada.

Conteúdos

1. Tecnologias e arquiteturas de servidores web

1.1. Servidores web: Apache, Nginx, Tomcat, Node.js
1.2. Arquiteturas de servidores: LAMP, MEAN, etc.
1.3. Protocolos de comunicação: HTTP, HTTPS, WebSockets
1.4. Características e comparação entre servidores web

2. Instalação e configuração de servidores

2.1. Instalação de servidores web e de aplicação
2.2. Configuração de servidores de web (parâmetros de escuta, localização de documentos)
2.3. Estruturação de diretórios e módulos no servidor de aplicação

3. Desenvolvimento de componentes de software

3.1. Desenvolvimento da camada de apresentação (HTML, CSS, JavaScript)
3.2. Lógica de negócios: PHP, Java, Python, Node.js
3.3. Aplicação de boas práticas de design: SOLID, DRY, modularidade

4. Controle de acesso e segurança

4.1. Implementação de autenticação e autorização
4.2. Técnicas de segurança: criptografia, controle de acesso
4.3. Endurecimento do sistema e aplicação de patches de segurança

5. **Otimização e monitoramento de desempenho**

5.1. Técnicas de otimização: cache, compressão de dados, balanceamento de carga
5.2. Monitoramento de desempenho: Prometheus, Grafana
5.3. Estratégias de escalabilidade e alta disponibilidade

6. **Testes e validação de componentes**

6.1. Testes unitários e de integração com JUnit, Mocha, PHPUnit
6.2. Testes de compatibilidade e portabilidade em diferentes ambientes (on-premise, cloud, híbridos)

Outras Capacidades Associadas à Unidade Formativo

- Capacidade de implementação de tecnologias web.
- Capacidade de implementar técnicas de autenticação, autorização e endurecimento de sistemas.
- Capacidade de garantir a escalabilidade e alta disponibilidade de sistemas web.
- Capacidade de executar testes e validação de componentes de software em diferentes ambientes.

MF3: Automação, Monitorização e Escalabilidade de Aplicações Web		
Código: MF374_5	Nível: 5	Duração: 180 Horas
Associado à UC374_5: Desenvolver aplicações web de alto desempenho, usabilidade e escalabilidade.		

Este MF é subdividido nas seguintes Unidades Formativas: caso necessário

		Código
■ Unidade Formativa 1:	Gestão e Automação de Processos de Entrega Web	UF480
■ Unidade Formativa 2:	Monitorização e Escalabilidade de Aplicações Web	UF481

Unidade Formativa 1: Gestão e Automação de Processos de Entrega Web (UF480)

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

RA1: Aplicar processos de gestão de alterações em aplicações web, utilizando ferramentas de controlo de versões para manter a evolução do software.

- **CA 1.1.** Agendar uma alteração, atualizando a partir de uma ramificação ativa do repositório da aplicação, utilizando uma ferramenta de controlo de versões.
- **CA 1.2.** Confirmar uma alteração, atualizando uma ramificação ativa do repositório da aplicação, utilizando uma ferramenta de controlo de versões.
- **CA 1.3.** Mesclar alterações de diferentes branches, atualizando um branch ativo e resolvendo conflitos com ferramentas de controlo de versões.
- **CA 1.4.** Identificar uma alteração, seguindo o processo de desenvolvimento iterativo, utilizando uma ferramenta de controlo de versões.
- **CA 1.5.** Sincronizar um repositório local com um repositório remoto, utilizando comandos do terminal ou ferramentas de controlo de versões.
- **CA 1.6.** Utilizar uma ferramenta de controlo de versões para gestão de alterações, criando, removendo e atualizando ramos de repositórios.

RA2: Construir artefactos, utilizando ferramentas de gestão de projetos e controlo de versões, produzindo, validando e instalando-os para manter a evolução do software com entrega contínua.

- **CA 2.1.** Produzir artefactos, utilizando código do repositório e ferramentas de gestão de projetos, de acordo com os requisitos do projeto.
- **CA 2.2.** Validar artefactos, executando testes de integração contínua, utilizando uma ferramenta de controlo de versões e validando requisitos funcionais e não funcionais.
- **CA 2.3.** Instalar artefactos, utilizando uma ferramenta de gestão de projetos, seguindo o processo de implementação do projeto.
- **CA 2.4.** Executar a gestão de artefactos no repositório, agendando testes e configurando servidores de integração contínua, garantindo a entrega contínua.

RA3: Compreender os requisitos funcionais e arquiteturais do projeto, aplicando os conceitos das visões 4+1 para garantir a rastreabilidade e estruturação da aplicação web.

- **CA 3.1.** Analisar a documentação das visões 4+1, utilizando diagramas de casos de uso, pacotes e classes, para assegurar a estruturação coerente da aplicação web.
- **CA 3.2.** Validar os requisitos funcionais e arquiteturais do projeto, utilizando os diagramas das visões 4+1, para garantir que todos os componentes da aplicação estão alinhados com as necessidades do cliente e as especificações do projeto.

CA 3.3. Elaborar e atualizar os diagramas de implementação da arquitetura, com referência aos requisitos de rede e capacidades de infraestrutura, para garantir a coerência da aplicação com a arquitetura definida.

CA 3.4. Compreender os requisitos funcionais e arquiteturais, aplicando os conceitos das visões 4+1, para garantir a rastreabilidade dos componentes da aplicação web.

CA 3.5. Realizar a verificação da conformidade entre os requisitos funcionais e a arquitetura do sistema, utilizando as visões 4+1, para garantir que todas as partes do sistema sejam corretamente integradas e atendam aos requisitos estabelecidos.

RA4: Adotar uma postura proativa na manutenção e monitorização da aplicação em produção, visando garantir a sua estabilidade e identificar melhorias contínuas.

CA 4.1. Monitorizar a aplicação em produção, utilizando ferramentas de logging e métricas de desempenho, para garantir a disponibilidade e identificar áreas de melhoria contínua.

CA 4.2. Propor melhorias no desempenho da aplicação com base em dados de monitorização, realizando ajustes necessários para otimizar a execução da aplicação em produção.

CA 4.3. Identificar e reportar falhas de segurança ou problemas de desempenho, aplicando práticas de monitoramento contínuo, para mitigar riscos emergentes e melhorar a estabilidade da aplicação.

CA 4.4. Demonstrar atitude proativa na análise de relatórios de erros e na comunicação com a equipa de desenvolvimento, para garantir que as correções sejam implementadas de forma eficaz e eficiente.

CA 4.5. Colaborar com a equipa para implementar práticas de automação de monitorização e alertas, para garantir que a aplicação em produção seja monitorizada de forma eficiente e que as intervenções necessárias sejam realizadas de maneira rápida e eficaz.

Conteúdos

1. Gestão e automação de processos de entrega web

1.1. Introdução à integração contínua (CI) e entrega contínua (CD).
1.2. Benefícios da automação no processo de entrega web.
1.3. Ferramentas de integração e entrega contínua.

2. Instalação e configuração de ferramentas de CI/CD

2.1. Instalação e configuração do jenkins.
2.2. Configuração de pipelines de entrega contínua.
2.3. Instalação de Ferramentas de Contêineres (Docker).

3. Automatização do processo de entrega

3.1. Configuração de integração de testes automatizados.
3.2. Gerenciamento de dependências e versionamento.
3.3. Automação da Publicação e Deploy.

4. Monitoramento e manutenção da aplicação web em produção.

4.1. Monitorização de Aplicações em Produção.
4.2. Diagnóstico e Resolução de Problemas em Produção.
4.3. Melhoria Contínua no Processo de Monitoramento.

5. Práticas avançadas e melhores práticas em CI/CD

5.1. Estratégias de automação para ambientes complexos.
5.2. Best practices em automação de testes e deploy.
5.3. Integração com outras ferramentas de devops.
5.4. Implantação de aplicação web com jenkins e Docker.
5.5 Implementação de CI/CD em ambientes de microserviços.

Outras Capacidades Associadas à Unidade Formativa

- Capacidade de compreender e aplicar os requisitos funcionais e arquiteturais de aplicações web usando as visões 4+1
- Capacidade de aplicar processos de gestão de alterações em aplicações web com ferramentas de controlo de versões.
- Capacidade de produzir e validar artefactos de software utilizando ferramentas de gestão de projetos e controlo de versões.
- Capacidade de aplicar práticas de monitoramento e alertas em aplicações web para garantir a qualidade contínua.

Unidade Formativa 2: Monitorização e Escalabilidade de Aplicações Web (UF481)

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

RA1: Testes de programação do código de uma aplicação web, utilizando ferramentas de teste, construindo, exercitando e verificando o assunto em teste.

- **CA 1.1.** Analisar a qualidade do software utilizando ferramentas de análise estática.
- **CA 1.2.** Atualizar código seguindo princípios SOLID e utilizando métricas de software.
- **CA 1.3.** Realizar testes de código de métodos de classe de produção.
- **CA 1.4.** Implementar testes de código de produção.
- **CA 1.5.** Utilizar ferramentas de teste unitário para desenvolver testes dinâmicos.
- **CA 1.6.** Testar a arquitetura do software com ferramentas de teste de integração.
- **CA 1.7.** Desenvolver testes da arquitetura de um sistema, incluindo nós e ligações de rede.

RA2: Compreender os conceitos e técnicas fundamentais de monitorização e escalabilidade de aplicações web.

- **CA 2.1.** Explicar os conceitos de monitorização e escalabilidade em ambientes web, utilizando documentação técnica, para demonstrar o entendimento dos princípios e desafios do processo. Testar componentes de software conforme o design de interface de casos de utilização.
- **CA 2.2.** Identificar as principais métricas de desempenho para monitorização de aplicações web e compreender a importância de cada uma, utilizando ferramentas de monitoramento, para assegurar o acompanhamento eficaz da performance da aplicação.
- **CA 2.3.** Descrever as técnicas de escalabilidade horizontal e vertical aplicáveis a ambientes web, demonstrando a escolha adequada com base nas exigências de tráfego e performance da aplicação.
- **CA 2.4.** Explicar a importância da arquitetura de microserviços para escalabilidade de aplicações web, detalhando como ela facilita a distribuição de carga e a manutenção da performance.
- **CA 2.5.** Avaliar os principais desafios de monitorização em ambientes web distribuídos, discutindo como lidar com grandes volumes de dados e como a escalabilidade pode ser ajustada para diferentes tipos de aplicação**.**

RA3: Demonstrar postura proativa na manutenção da estabilidade e performance das aplicações web em produção.

- **CA 3.1.** Identificar falhas de segurança e problemas de desempenho nas aplicações web em produção, utilizando ferramentas de monitoramento contínuo e análise de dados, para sugerir melhorias proativas e corrigir falhas rapidamente.
- **CA 3.2.** Colaborar com a equipe de desenvolvimento e operações para implementar práticas de monitorização contínua e automação de alertas, para garantir que a aplicação seja monitorizada de forma eficiente e que as intervenções necessárias sejam realizadas rapidamente.
- **CA 3.3.** Implementar práticas de revisão contínua dos logs de erro e métricas de performance, utilizando ferramentas de análise e dashboards, para antecipar problemas e otimizar a aplicação.
- **CA 3.4.** Realizar ajustes de performance em tempo real durante períodos de pico de utilização, utilizando recursos de escalabilidade e balanceamento de carga, para garantir a estabilidade da aplicação web.
- **CA 3.5.** Propor e aplicar melhorias na infraestrutura de monitoramento e alarme, com base em análise de incidentes anteriores, para melhorar a resposta da equipe e a capacidade de prevenção de falhas.

RA4: Aplicar práticas e ferramentas para monitorar e otimizar a escalabilidade de aplicações web em produção.

CA 4.1. Implementar ferramentas de monitorização de desempenho e log num ambiente de produção, configurando o sistema para capturar métricas de desempenho e disponibilidade da aplicação, de acordo com as necessidades do projeto.

CA 4.2. Aplicar técnicas de escalabilidade em servidores web, utilizando mecanismos de balanceamento de carga, cache e compressão de dados, para otimizar o desempenho da aplicação, seguindo as especificações do projeto.

CA 4.3. Utilizar ferramentas de análise de tráfego e comportamento do utilizador para ajustar a capacidade de servidores em tempo real, visando garantir uma experiência do utilizador consistente e com alta disponibilidade.

CA 4.4. Configurar e otimizar as bases de dados para suportar grandes volumes de acessos simultâneos, aplicando técnicas de particionamento, replicação e indexação, conforme as necessidades de escalabilidade definidas no projeto.

Conteúdos

1. Fundamentos de monitorização e escalabilidade de aplicações web

1.1. Conceitos de monitorização e escalabilidade.
1.2. Tipos de monitorização.
1.3. Escalabilidade horizontal vs. Vertical.
1.4. Arquitetura de microserviços.
1.5. Impacto da infraestrutura no desempenho.

2. Ferramentas e técnicas de monitorização

2.1. Ferramentas de monitorização.
2.2. Logs e métricas de performance.
2.3. monitorização contínua.
2.4. Testes de performance.

3. Escalabilidade de aplicações web

3.1. Escalabilidade horizontal.
3.2. Escalabilidade vertical.
3.3. Caching e compressão de dados.
3.4. Otimização de banco de dados.

4. Práticas de testes de software

4.1. Testes de código.
4.2. Análise estática de código.
4.3. Ferramentas de testes unitários.

5. Práticas de Melhoria Contínua em Produção

5.1. Ajustes de performance em tempo real.
5.2. Segurança em produção.
5.3. Automação de Processos de Deploy (CI/CD).

Outras Capacidades Associadas à Unidade Formativa

- Capacidade de Compreender e Aplicar os Conceitos de Monitorização e Escalabilidade em Ambientes Web Utilizando Ferramentas e Técnicas Adequadas.
- Capacidade de Monitorizar e Avaliar a Performance de Aplicações Web em Produção, Utilizando Ferramentas de Monitoramento Contínuo e Análise de Dados.
- Capacidade de Configurar e Otimizar Bases de Dados para Suportar Grandes Volumes de Acessos Simultâneos em Ambientes Web de Alta Escalabilidade.
- Capacidade de Aplicar Técnicas de Segurança em Ambientes de Produção de Aplicações Web, Garantindo a Proteção de Dados e Integridade da Infraestrutura.

MF4: Integração de APIs, Otimização e Segurança em Aplicações Web		
Código: MF375_5	Nível: 5	Duração: 210 Horas
Associado à UC375_5: Implementar APIs e serviços web para integração entre sistemas, otimizando o desempenho e a segurança em aplicações web.		

Este MF é subdividido nas seguintes Unidades Formativas: caso necessário

		Código
■ Unidade Formativa 1:	Integração e Gestão de APIs em Aplicações Web	UF482
■ Unidade Formativa 2:	Segurança e Qualidade em Aplicações Web	UF483

Unidade Formativa 1: Integração e Gestão de APIs em Aplicações Web (UF482)

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

RA1: Compreender os princípios e padrões de design de APIs (REST, GraphQL) e sua aplicação na integração de sistemas.

CA 1.1. Explicar os conceitos fundamentais de APIs REST e GraphQL, destacando as suas diferenças e vantagens, com base em referências técnicas e exemplos práticos.

CA 1.2. Identificar os princípios de design de APIs, demonstrando boas práticas como uniformidade, versionamento e segurança, em conformidade com padrões reconhecidos.

CA 1.3. Analisar cenários de integração de sistemas, avaliando a aplicabilidade de REST e GraphQL em função de requisitos específicos de escalabilidade, desempenho e compatibilidade.

CA 1.4. Relacionar os princípios de design de APIs com casos reais de integração de sistemas, utilizando documentação técnica e estudos de caso do setor.

RA2: Desenvolver e documentar APIs para integração de sistemas, garantindo eficiência, segurança e escalabilidade.

CA 2.1. Implementar uma API funcional com endpoints definidos segundo padrões REST ou GraphQL, utilizando boas práticas de design e desenvolvimento.

CA 2.2. Configurar mecanismos de autenticação e autorização para controle de acesso à API, aplicando padrões de segurança como OAuth 2.0, JWT ou OpenID Connect.

CA 2.3. Testar a API desenvolvida verificando a comunicação entre sistemas, desempenho e segurança, utilizando ferramentas como Postman, Swagger ou testes automatizados.

CA 2.4. Documentar a API detalhando endpoints, métodos, parâmetros e respostas, seguindo diretrizes de documentação técnica e utilizando ferramentas adequadas como Swagger ou Redoc.

RA3: Implementar mecanismos de autenticação e controle de acesso em APIs para garantir a proteção da comunicação entre sistemas.

CA 3.1. Configurar um mecanismo de autenticação utilizando padrões como OAuth 2.0, JWT ou OpenID Connect, garantindo a proteção contra acessos não autorizados.

CA 3.2. Definir políticas de controle de acesso com base em cargos e permissões de utilizadores e aplicações, aplicando o princípio de privilégio mínimo.

CA 3.3. Implementar a transmissão segura de dados utilizando HTTPS e criptografia, assegurando a confidencialidade e integridade das informações trocadas.

CA 3.4. Testar a segurança da API verificando vulnerabilidades como injeção de tokens, ataques de replay e falhas na autorização, utilizando ferramentas de análise de segurança.

RA4: Adotar boas práticas na integração de APIs de terceiros, garantindo segurança, compatibilidade e conformidade com padrões de mercado.

CA 4.1. Configurar a integração com APIs de terceiros utilizando protocolos e bibliotecas adequadas, assegurando a compatibilidade com os requisitos da aplicação.

CA 4.2. Armazenar credenciais de acesso a APIs externas de forma segura, seguindo práticas recomendadas como variáveis de ambiente e cofres de segredos.

CA 4.3. Validar os dados recebidos de APIs externas antes de processá-los, garantindo a integridade e conformidade com as regras de negócio da aplicação.

CA 4.4. Monitorizar o desempenho da comunicação com APIs externas avaliando latência, tempo de resposta e consumo de recursos, utilizando ferramentas de observabilidade.

CA 4.5. Implementar políticas de rate limiting e caching para otimizar a performance e evitar sobrecarga, garantindo a estabilidade da aplicação.

Conteúdos

1. Fundamentos de APIs e padrões de design

1.1. Introdução às APIs

1.2. Arquitetura REST e princípios RESTful
1.3. GraphQL: Conceitos, vantagens e diferenças em relação ao REST
1.4. Princípios de design de APIs: uniformidade, versionamento e segurança
1.5. Documentação de APIs: OpenAPI, Swagger, Redoc
1.6. Exemplos práticos e estudos de caso

2. Desenvolvimento e Documentação de APIs

2.1. Ferramentas e frameworks para criação de APIs (Node.js, Flask, Django, Spring Boot)
2.2. Criação de endpoints seguindo padrões REST e GraphQL
2.3. Boas práticas no desenvolvimento de APIs: modularidade, desacoplamento e reusabilidade
2.4. Configuração e implementação de autenticação e autorização (OAuth 2.0, JWT, OpenID Connect)
2.5. Testes e depuração de APIs usando Postman, Swagger e testes automatizados

3. Segurança em APIs e Controle de Acesso

3.1. Princípios de segurança em APIs
3.2. Mecanismos de autenticação e autorização
3.3. Implementação de HTTPS e criptografia de dados
3.4. Proteção contra ataques comuns: injeção SQL, XSS, CSRF, ataques de replay
3.5. Implementação de medidas de segurança como CSP, SRI e políticas de expiração de tokens

4. Integração de APIs de Terceiros

4.1. Protocolo HTTP e consumo de APIs externas
4.2. Configuração e autenticação com APIs de terceiros
4.3. Gerenciamento seguro de credenciais de acesso (variáveis de ambiente, cofres de segredos)
4.4. Monitoramento de consumo e performance de APIs externas
4.5. Implementação de rate limiting e caching para otimização de desempenho

5. Versionamento e Manutenção de APIs

5.1. Estratégias de versionamento de APIs: URI versioning, headers
5.2. Garantia de retrocompatibilidade em atualizações
5.3. Monitoramento de APIs: ferramentas de logging e análise de métricas
5.4. Testes automatizados para validação contínua
5.5. Otimização de desempenho: compressão de respostas, balanceamento de carga

6. Arquiteturas Avançadas para APIs e Microsserviços

6.1. Arquiteturas baseadas em RESTful e GraphQL
6.2. Implementação de microsserviços e sua modularidade
6.3. Utilização de containers (Docker) e orquestração (Kubernetes)

6.4. Gateways de API e estratégias de caching distribuído
6.5. Monitoramento e observabilidade de microsserviços

7. Boas Práticas de Segurança e Otimização

7.1 Lazy loading e compressão para melhoria de desempenho
7.2. Conformidade com normas de acessibilidade (WCAG)
7.3. Aplicação de criptografia ponta a ponta
7.4. Proteção contra vulnerabilidades comuns com SAST e DAST
7.5. Testes de carga e análise de performance contínua

Outras Capacidades Associadas à Unidade Formativa

- Capacidade de compreender e aplicar os conceitos de integração e gestão de apis em aplicações web, utilizando padrões e boas práticas de desenvolvimento.
- Capacidade de projetar, implementar e documentar apis rest e graphql, assegurando escalabilidade, segurança e eficiência na comunicação entre sistemas.
- Capacidade de configurar e implementar mecanismos de autenticação e autorização em APIs, utilizando padrões como OAuth 2.0, JWT e OpenID Connect.;
- Capacidade de aplicar boas práticas de segurança na comunicação entre sistemas, garantindo a proteção de dados sensíveis e a conformidade com normas e padrões do setor.;
- Capacidade de implementar arquiteturas baseadas em microsserviços, utilizando containers, orquestração e gateways de api para escalabilidade e modularidade.

Unidade Formativa 2: Segurança e Qualidade em Aplicações Web (UF483)

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

RA1: Conhecer as principais vulnerabilidades de segurança em aplicações web e técnicas para mitigação de riscos.

- **CA 1.1.** Explicar as vulnerabilidades de segurança mais comuns em aplicações web, com base no OWASP Top 10, apresentando exemplos práticos e impactos.
- **CA 1.2.** Identificar falhas de segurança em aplicações web, analisando código-fonte, relatórios de testes de segurança ou simulações de ataques.
- **CA 1.3.** Comparar diferentes técnicas de mitigação de riscos em aplicações web, justificando a escolha da melhor abordagem para cada vulnerabilidade identificada.
- **CA 1.4.** Relacionar vulnerabilidades de segurança com casos reais de ataques cibernéticos, demonstrando as consequências e as medidas adotadas para mitigação.
- **CA 1.5.** Descrever boas práticas de segurança em aplicações web, incluindo autenticação segura, criptografia de dados e políticas de controle de acesso.

RA2: Implementar medidas de segurança para proteção de dados, incluindo criptografia, autenticação multifator e proteção contra ataques comuns (XSS, CSRF, SQL Injection).

- **CA 2.1.** Implementar criptografia de dados sensíveis em trânsito e em repouso, utilizando algoritmos seguros, como AES e RSA, em conformidade com os requisitos de segurança.
- **CA 2.2.** Configurar autenticação multifator (MFA) em aplicações web, integrando provedores de identidade, como Google Authenticator ou Authy, para reforçar a segurança de acesso.
- **CA 2.3.** Desenvolver mecanismos de proteção contra ataques XSS (Cross-Site Scripting), utilizando técnicas como a validação de entrada e a codificação de saídas, conforme as melhores práticas de segurança.
- **CA 2.4.** Implementar medidas de proteção contra ataques CSRF (Cross-Site Request Forgery), utilizando tokens CSRF e políticas de SameSite para cookies.
- **CA 2.5.** Mitigar riscos de SQL Injection em aplicações web, aplicando boas práticas como a utilização de consultas parametrizadas e ORM (Object-Relational Mapping).
- **CA 2.6.** Testar as medidas de segurança implementadas, avaliando sua eficácia na proteção contra ataques comuns e vulnerabilidades, utilizando ferramentas de análise de segurança, como OWASP ZAP ou Burp Suite.

RA3: Demonstrar compromisso com a qualidade e segurança no desenvolvimento de aplicações web, adotando metodologias de teste e monitoramento contínuo.

- **CA 3.1.** Adotar práticas de teste contínuo em todas as fases do desenvolvimento, utilizando ferramentas de integração contínua (CI) como Jenkins ou GitHub Actions para garantir a qualidade do código.
- **CA 3.2.** Implementar testes automatizados de unidade, integração e segurança, utilizando frameworks adequados (como JUnit, Selenium ou Jest), assegurando a cobertura de funcionalidades críticas.
- **CA 3.3.** Monitorizar a aplicação web em produção, utilizando ferramentas de monitoramento contínuo (como Prometheus, Grafana ou New Relic), para garantir a detecção proativa de problemas de performance e segurança.
- **CA 3.4.** Garantir a conformidade com as normas de segurança da aplicação, utilizando ferramentas de análise estática de código (como SonarQube) para identificar vulnerabilidades e falhas de segurança.
- **CA 3.5.** Aplicar metodologias ágeis e práticas de DevSecOps, assegurando que as medidas de segurança e qualidade sejam integradas ao processo de desenvolvimento de forma contínua.
- **CA 3.6.** Realizar revisões de código regulares, com foco em qualidade e segurança, utilizando pair programming ou peer reviews, garantindo a conformidade com as melhores práticas de

segurança e desempenho.

RA4: Aplicar práticas avançadas de otimização e monitorização **de desempenho em aplicações web, garantindo escalabilidade e eficiência.**

CA 4.1. Implementar técnicas de otimização de desempenho em aplicações web, como compressão de respostas, minificação de arquivos e otimização de imagens, garantindo redução no tempo de carregamento.

CA 4.2. Configurar e otimizar a utilização de cache (tanto em nível de servidor quanto de navegador), assegurando a melhoria na performance de aplicações web com alta demanda de tráfego.

CA 4.3. Monitorizar o desempenho da aplicação utilizando ferramentas como Google Lighthouse, New Relic ou Datadog, para avaliar métricas chave como tempo de resposta, latência e a utilização de recursos, garantindo melhorias contínuas.

CA 4.4. Aplicar técnicas de balanceamento de carga para distribuir eficientemente as requisições, assegurando a escalabilidade da aplicação em ambientes de alto tráfego.

CA 4.5. Realizar testes de carga e stress utilizando ferramentas como JMeter ou Gatling, para simular grandes volumes do utilizador e identificar gargalos de desempenho, com foco em garantir a estabilidade sob alta demanda.

CA 4.6. Implementar práticas de monitorização de desempenho em tempo real, utilizando ferramentas de observabilidade como Prometheus e Grafana, para detectar e resolver problemas de desempenho antes que afetem os utilizadores.

CA 4.7. Aplicar práticas de otimização de consultas a bancos de dados, incluindo a utilização de índices e otimização de consultas SQL, garantindo alta performance e baixa latência em grandes volumes de dados.

Conteúdos

1. Fundamentos de segurança e qualidade em aplicações web

1.1. Introdução à segurança em aplicações web.

1.1.1. Conceito de segurança em aplicações web.

1.1.2. A importância da segurança para proteger dados e garantir a confiança do utilizador.

1.1.3. Aplicações da segurança em contextos como autenticação, autorização e proteção contra ataques.

2. Principais vulnerabilidades em aplicações web: OWASP Top 10

2.1. Ameaças e vulnerabilidades mais comuns em aplicações web.

2.2. Impacto das vulnerabilidades nas aplicações web.

3. Princípios de arquitetura e segurança em web APIs

3.1. Definição de APIs e seu papel na segurança de aplicações web.

3.2. Como as APIs devem ser projetadas para serem seguras.

3.3. Principais riscos de segurança em APIs.

4. Padrões de design de APIs Seguras: versionamento, autenticação e criptografia

4.1. Design de APIs seguras com foco em autenticação e autorização.

4.2. Estratégias de versionamento de APIs.

4.3. Criptografia em APIs.

5. Boas Práticas para documentação de APIs: OpenAPI, Swagger, Redoc

5.1. Introdução à documentação de APIs e sua importância.

5.2. Ferramentas para documentação de APIs.

5.3. Como utilizar a documentação para melhorar a segurança e usabilidade da API.

6. Exemplos práticos e estudos de caso

6.1. Casos reais de falhas de segurança em aplicações web.

6.2. Boas práticas de implementação de segurança em projetos reais.

Outras Capacidades Associadas à Unidade Formativa

- Capacidade de implementar medidas de segurança para proteção de dados em aplicações web, utilizando criptografia, autenticação multifatorial e técnicas de defesa contra ataques comuns;
- Capacidade de monitorizar e avaliar a qualidade e o desempenho de aplicações web, utilizando ferramentas de monitoramento contínuo e testes automatizados;
- Capacidade de gerir a performance e escalabilidade de aplicações web, utilizando técnicas de cache, balanceamento de carga e testes de carga;
- Capacidade de adotar estratégias avançadas de segurança em apis, incluindo autenticação, autorização e proteção contra ameaças.

MÓDULO FORMATIVO EM CONTEXTO REAL DE TRABALHO	MT_TIC001.A2
Nível: 5	Duração indicativa: 360 Horas
Associado a todas as Unidades de Competência	

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

RA1: Desenvolver documentos web otimizados para diferentes navegadores, aplicando boas práticas de codificação e padrões de design.

- **CA 1.1.** Estruturar documentos web utilizando tags de marcação apropriadas e respeitando as normas de design.
- **CA 1.2.** Configurar e testar a funcionalidade de documentos web em múltiplos navegadores, garantindo sua compatibilidade.
- **CA 1.3.** Implementar código limpo utilizando HTML, CSS e JavaScript para garantir a eficiência e legibilidade do documento.
- **CA 1.4.** Integrar conteúdos multimédia (imagens, áudio, vídeo) de maneira eficiente, garantindo acessibilidade e usabilidade

RA2: Desenvolver componentes interativos e dinâmicos para interfaces web, utilizando linguagens de programação e atendendo aos requisitos de usabilidade.

- **CA 2.1.** Codificar componentes dinâmicos utilizando JavaScript, assegurando que o conteúdo da página seja atualizado sem necessidade de recarregar a página.
- **CA 2.2.** Implementar formulários interativos, utilizando validação e tratamentos de eventos de forma eficiente.
- **CA 2.3.** Integrar APIs externas para a comunicação dinâmica com o servidor, utilizando AJAX ou outras ferramentas.

RA3: Demonstrar responsabilidade na implementação de boas práticas de acessibilidade, usabilidade e performance.

- **CA 3.1.** Aplicar princípios de acessibilidade e usabilidade, utilizando recursos como ARIA para garantir uma boa experiência para todos os utilizadores.
- **CA 3.2.** Testar e validar componentes interativos, ajustando-os conforme necessário para garantir uma boa performance e compatibilidade em diferentes plataformas.
- **CA 3.3.** Manter uma postura ética, garantindo que os componentes desenvolvidos sigam boas práticas de segurança e protejam os dados dos utilizadores.

RA4: Integrar sistemas e tecnologias, garantindo a funcionalidade e a compatibilidade dos componentes de software no ambiente do servidor.

- **CA 4.1.** Implementar e testar APIs no servidor da empresa, utilizando padrões como REST ou GraphQL, e garantindo que os mecanismos de comunicação entre cliente e servidor sejam eficientes.
- **CA 4.2.** Configurar e otimizar a interação entre servidores web e servidores de aplicações, ajustando parâmetros de segurança, autenticação e autorização conforme as políticas da organização.
- **CA 4.3.** Validar a compatibilidade dos componentes de software com as plataformas e infraestruturas existentes na empresa, realizando testes de integração e compatibilidade.

RA5: Demonstrar atitudes proativas para garantir a segurança, a escalabilidade e a manutenção dos componentes de software no ambiente real da empresa.

- **CA 5.1.** Monitorizar o desempenho dos componentes de software no servidor da empresa, utilizando ferramentas específicas para garantir a escalabilidade e eficiência dos sistemas em produção.
- **CA 5.2.** Aplicar boas práticas de segurança no desenvolvimento e na configuração dos componentes de software, garantindo a proteção contra vulnerabilidades e ameaças.

CA 5.3. Implementar técnicas de otimização de desempenho e escalabilidade, como balanceamento de carga, caching e monitorização, garantindo a performance das aplicações em um ambiente real.

RA6: Aplicar o desenvolvimento e a configuração de componentes de software no servidor, conforme os padrões da empresa.

CA 6.1. Instalar e configurar servidores web e servidores de aplicação na empresa, ajustando parâmetros e configurando diretórios de documentos conforme as exigências da organização.

CA 6.2. Desenvolver componentes de software para execução no servidor da empresa, utilizando tecnologias e frameworks específicos para atender às necessidades da empresa.

CA 6.3. Integrar a camada de apresentação e lógica de negócio nos componentes de software, seguindo as melhores práticas estabelecidas pela organização.

RA7: Analisar o ambiente produtivo da empresa e a sua relação com as tecnologias de desenvolvimento web no servidor.

CA 7.1. Identificar e descrever as arquiteturas e tecnologias de desenvolvimento web do lado do servidor utilizadas na empresa, analisando suas características e aplicabilidade no contexto real.

CA 7.2. Comparar as soluções de hospedagem e implantação de aplicações web, avaliando as práticas adotadas pela empresa em termos de servidores dedicados, cloud computing e containers.

CA 7.3. Analisar os requisitos de configuração dos servidores web e servidores de aplicação da empresa, avaliando a adequação das escolhas feitas em relação às necessidades da organização.

RA8: Aplicar processos de gestão de alterações em aplicações web, utilizando ferramentas de controlo de versões para manter a evolução do software.

CA 8.1. Agendar uma alteração, atualizando a partir de uma ramificação ativa do repositório da aplicação, utilizando uma ferramenta de controlo de versões.

CA 8.2. Confirmar uma alteração, atualizando uma ramificação ativa do repositório da aplicação, utilizando uma ferramenta de controlo de versões.

CA 8.3. Mesclar alterações de diferentes branches, atualizando um branch ativo e resolvendo conflitos com ferramentas de controlo de versões.

CA 8.4. Identificar uma alteração, seguindo o processo de desenvolvimento iterativo, utilizando uma ferramenta de controlo de versões.

CA 8.5. Sincronizar um repositório local com um repositório remoto, utilizando comandos do terminal ou ferramentas de controlo de versões.

CA 8.6. Utilizar uma ferramenta de controlo de versões para gestão de alterações, criando, removendo e atualizando ramos de repositórios.

RA9: Construir artefactos, utilizando ferramentas de gestão de projetos e controlo de versões, produzindo, validando e instalando-os para manter a evolução do software com entrega contínua.

CA 9.1. Produzir artefactos, utilizando código do repositório e ferramentas de gestão de projetos, de acordo com os requisitos do projeto.

CA 9.2.Validar artefactos, executando testes de integração contínua, utilizando uma ferramenta de controlo de versões e validando requisitos funcionais e não funcionais.

CA 9.3. Instalar artefactos, utilizando uma ferramenta de gestão de projetos, seguindo o processo de implementação do projeto.

CA 9.4. Executar a gestão de artefactos no repositório, agendando testes e configurando servidores de integração contínua, garantindo a entrega contínua.

RA10: Compreender a aplicação dos princípios e práticas de integração e gestão de APIs em um ambiente de trabalho real.

CA10.1. Explicar os princípios de design de APIs, como REST e GraphQL, aplicados em situações reais de integração de sistemas, conforme as necessidades de uma empresa.

CA10.2. Identificar e documentar os padrões de integração de APIs utilizados na organização, garantindo conformidade com as melhores práticas de mercado.

CA10.3. Avaliar e comparar as soluções de integração adotadas pela empresa, considerando critérios como segurança, escalabilidade e desempenho.

CA10.4 Relatar casos de integração de APIs observados no ambiente de trabalho, destacando desafios e soluções implementadas pela equipa.

RA11: Desenvolver, testar e documentar APIs em ambiente de trabalho real, assegurando a sua integração eficaz com sistemas existentes.

CA11.1. Desenvolver uma API funcional para integração de sistemas internos, utilizando padrões como REST ou GraphQL, atendendo aos requisitos da empresa.

CA11.2. Testar a API em diferentes cenários de integração, avaliando a sua performance e segurança em conformidade com as exigências da organização.

CA11.3. Documentar a API com clareza, incluindo especificações de endpoints, métodos e parâmetros, para facilitar a utilização por outras equipas ou sistemas.

CA11.4. Implementar e verificar mecanismos de autenticação e autorização, utilizando práticas recomendadas de segurança, como OAuth 2.0, para controle de acesso à API.

RA12: Garantir a implementação de práticas de otimização e monitorização contínua das APIs no contexto real de trabalho, assegurando a escalabilidade e desempenho das aplicações.

CA12.1. Implementar técnicas de otimização de performance em APIs, como caching e compressão de respostas, para melhorar a eficiência e reduzir a latência.

CA12.2 Realizar testes de carga e stress para avaliar a performance das APIs sob alta demanda de tráfego, identificando e resolvendo gargalos.

CA12.3. Aplicar práticas de balanceamento de carga para distribuir requisições de forma eficiente, garantindo a escalabilidade das APIs em ambientes de produção.

CA12.4. Utilizar ferramentas de monitoramento e observabilidade para detectar e resolver problemas de desempenho em tempo real, como New Relic ou Datadog.


SNQ