**Escola Superior Pedagógica do Bengo**

**Departamento de Ensino Investigação e Extensão de Ciências Exactas**

**Curso de Licenciatura em Ensino da Informática**

# Título: **RELATÓRIO TÉCNICO DA APLICAÇÃO WEB "SPEX".**

Trabalho orientado pelo Eng.º Euclides Catumbela como requisito para obtenção da 3ª nota da unidade curricular de Desenvolvimento Web.

**Grupo n.º 05**

**3º Ano**

**Turma:** A-T

**Período:** Período

**Ano Académico:** 2024/2025

**Caxito, Abril de 2025**

**Escola Superior Pedagógica do Bengo**

**Departamento de Ensino Investigação e Extensão de Ciências Exactas**

**Curso de Licenciatura em Ensino da Informática**

Título: **RELATÓRIO TÉCNICO DA APLICAÇÃO WEB "SPEX".**

Trabalho orientado pelo Eng.º Euclides Catumbela como requisito para obtenção da 3ª nota da unidade curricular de Desenvolvimento Web.

**Caxito, Abril de 2025**

## DEDICATÓRIA

A todos os estudantes que têm enfrentado inúmeras dificuldades nas suas preparações para ingressarem no ensino superior.

## AGRADECIMENTOS

À Deus Todo-Poderoso, pela vida e pela saúde.

Às nossas famílias, por todo o apoio que nos têm dado em todas as etapas das nossas formações académicas.

Aos colegas e amigos que sempre estão dispostos a prover a ajuda necessária na resolução dos problemas académicos e pessoais que temos enfrentado.

## EPÍGRAFE

*"Aprender é a única coisa de que a mente nunca se cansa, nunca tem medo e nunca se arrepende."*

— Leonardo da Vinci

# ÍNDICE

## LISTA DE ABREVIATURAS

CRUD: Create, Read, Update, Delete

CSS: Cascading Style Sheet

DER: Diagrama Entidade-Relacionamento

DHCP: Dynamic Host Configuration Protocol

DNS: Domain Name System

HTML: HyperText Markup Language

HTTPS: HyperText Transfer Protocol Secure

IP: Internet Protocol

HTTP: HyperText Transfer Protocol

PHP: PHP HyperText Preprocessor

*PHP-FPM: PHP FastCGI Process Manager*

SGBD: Sistema de Gestão de Bancos de Dados

SPEX: Sistema de Preparação para Exames

SQL: Structured Query Language

SSL: Secure Sockets Layer

SVG: Scalable Vector Graphics

UML: Unified Modeling Language

## LISTA DE FIGURAS

## LISTA DE TABELAS

## RESUMO

Este relatório apresenta o desenvolvimento do Sistema de Preparação para Exames (SPEX), uma aplicação web educacional voltada à prática de testes simulados e à preparação de estudantes para exames académicos. O projeto foi desenvolvido utilizando uma arquitectura baseada em microsserviços, com a separação de funcionalidades entre os serviços de backend (PHP), frontend (CSS com Bulma e Javascript) e serviços de rede (DHCP, DNS). Foram utilizadas tecnologias modernas como MySQL, Git, GitHub, NGINX, entre outras, com ênfase na escalabilidade, segurança, desempenho e manutenibilidade. A metodologia Scrum foi adotada para a organização e acompanhamento do projeto, aliando práticas ágeis com ferramentas como Kanban. O relatório aborda desde a concepção da ideia e levantamento de requisitos até a modelagem e implementação da aplicação, ressaltando os desafios enfrentados e as soluções adotadas pela equipe.

## ABSTRACT

This report presents the development of the Exam Preparation System (SPEX), a web-based educational application designed to support students in practicing mock exams and preparing for academic assessments. The system architecture is built on microservices, separating functionalities among backend services (PHP), frontend (CSS with Bulma and Javascript), and network services (DHCP, DNS). Modern technologies were employed, including MySQL, Git, GitHub and NGINX, with a focus on scalability, security, performance, and maintainability. The Scrum methodology guided project organization and monitoring, complemented by Kanban tools. This report documents the entire development cycle, from idea conception and requirements engineering to system modeling and implementation, highlighting the challenges faced and the solutions implemented by the team.

# INTRODUÇÃO

## Descrição do Problema

O acesso ao ensino superior representa um grande desafio para muitos estudantes que concluem o ensino médio. A falta de condições financeiras para frequentar cursos preparatórios, a dificuldade de acesso à materiais didáticos atualizados e a ausência de plataformas que permitam a realização de testes simulados são alguns dos principais obstáculos enfrentados pelos candidatos. Como consequência, um número significativo de estudantes não consegue obter a pontuação necessária para ingressar na universidade, reduzindo suas oportunidades educacionais e profissionais.

Diante desse cenário, surge o **Sistema de Preparação para Exames (SPEX)**, uma aplicação web que visa proporcionar aos estudantes uma preparação realista para os exames de admissão. O SPEX visa oferecer materiais didáticos organizados, exames simulados interativos baseados em provas anteriores e um sistema de *ranking* que permite aos usuários avaliar seu desempenho e acompanhar sua evolução em relação a outros candidatos.

## Constituição do Grupo e Função de Cada Membro

O grupo está constituído pelos seguintes estudantes:

1. Gabriel Correia Pedro – Gerente do projecto e Programador *backend* (responsável por coordenar o progresso, desenvolver e implementar a lógica do servidor e a integração com a base de dados);
2. Paula Manuel Benjamin – Programadora *frontend* (responsável pelo design da interface e experiência do utilizador);
3. Gracieth Quicunga Paulo – Especialista em redes (responsável pela configuração do servidor web e da rede cliente-servidor);
4. Celestina Amélia Augusto Tomás – Documentadora (responsável por produzir o relatório técnico detalhado).

## Objectivos do Trabalho

OBJECTIVO GERAL: desenvolver uma aplicação web sob a arquitectura de rede cliente-servidor que possa ser acedido por dispositivos de diferentes tamanhos.

OBJECTIVOS ESPECÍFICOS:

1. Desenvolver modelos *UML* do sistema;
2. Desenvolver um *frontend* responsivo para todo o sistema;
3. Desenvolver uma base de dados para o sistema;
4. Desenvolver um *backend* robusto e confiável;
5. Configurar um servidor de aplicação para hospedar o sistema em produção;
6. Configurar um servidor de banco de dados para o sistema em produção;
7. Configurar servidores *DNS* e *DHCP*;
8. Integrar todos os componentes do sistema;
9. Disponibilizar o sistema para ser usado pelo público.

## Organização do Trabalho

O primeiro capítulo contém uma descrição do sistema e da sua arquitectura.

O segundo e o terceiro capítulo descrevem o processo de implementação do sistema. O segundo capítulo descreve o processo de configuração do servidor web. Já o terceiro capítulo contém detalhes sobre a configuração da rede cliente-servidor.

O quinto capítulo apresenta os resultados do trabalho.

O sexto capítulo apresenta uma reflexão crítica feita pelo grupo, sobre o trabalho realizado pelo grupo e aponta os principais problemas encontrados durante o processo.

O sétimo capítulo apresenta a conclusão do relatório e os trabalhos futuros, no âmbito do projecto *SPEX*.

Os diagramas *UML* e *DER* podem ser encontrados nos apêndices, no final do documento.

## DESCRIÇÃO DA ARQUITECTURA DO SISTEMA WEB

A arquitetura do sistema *SPEX* segue uma estrutura moderna e escalável, composta por várias camadas que interagem entre si para oferecer uma experiência robusta e eficiente para os usuários. A arquitetura foi projetada para suportar a realização de exames simulados e a geração de relatórios de desempenho, com uma base sólida para integração futura de novos recursos. Abaixo está uma descrição detalhada das camadas e componentes envolvidos.

O SPEX adota uma arquitetura cliente-servidor com separação de responsabilidades em dois computadores distintos:

a) Servidor de Aplicação (computador 1): hospeda o backend (PHP) e o frontend (HTML5 e CSS3, com Bulma).
b) Servidor de Banco de Dados (computador 2): MySQL, SGBD responsável pelo armazenamento de usuários, exames, resultados e relatórios.
c) Servidor de Serviços de Rede (computador 1): DNS (Bind9) e DHCP (ISC DHCP Server), para gerenciamento interno do domínio e endereçamento IP.

### Camada de Apresentação (Frontend)

A camada de apresentação é responsável pela interação com o usuário e é composta por uma interface web dinâmica e responsiva, desenvolvida com Bulma. O Bulma foi escolhido devido à sua capacidade de criar interfaces responsivas e de fácil manutenção.

### Camada de Lógica de Negócio (Backend)

O *backend* do sistema *SPEX* é responsável pelo processamento de dados, validações, cálculos de desempenho, e pela interação com o banco de dados. Ele é desenvolvido em *PHP* e gerenciado através do *PHP-FPM* (*FastCGI Process Manager*) para oferecer um desempenho eficiente e de baixo custo.

### Camada de Dados (Banco de Dados)

O banco de dados é a camada responsável pelo armazenamento e gerenciamento dos dados persistentes do sistema, incluindo informações sobre os estudantes, questões de

exames, respostas, e relatórios de desempenho. O MySQL foi escolhido como banco de dados relacional devido à sua robustez, escalabilidade e capacidade de realizar consultas complexas.

**Estrutura do Banco de Dados**

Tabela estudante: Contém informações dos estudantes, como nome, email e senha.

Tabela pergunta_cadastrada: Armazena as questões de exames, com campos como enunciado, resposta, curso, disciplina, tema.

Tabela exame_universidade: Guarda os exames reais das distintas IES, e que estão disponíveis no sistema.

Tabela histórico_estudante: Registra o desempenho do estudante em cada exame, incluindo a nota obtida.

A comunicação entre o backend e o banco de dados é realizada por meio de consultas SQL. Para garantir maior segurança e evitar vulnerabilidades como SQL Injection, o sistema faz uso de consultas parametrizada.

**Servidor Web (NGINX)**

O servidor web NGINX é utilizado para servir o frontend e direcionar as requisições para o PHP-FPM, que processa o backend. O NGINX é um servidor web eficiente e de alto desempenho, projetado para lidar com um grande número de conexões simultâneas.

**Configuração**

O NGINX é configurado para ouvir requisições na porta 80 (HTTP) e redirecionar as requisições PHP para o PHP-FPM. Arquivos estáticos (HTML, CSS, Javascript) são servidos diretamente pelo NGINX, enquanto as requisições dinâmicas são tratadas pelo PHP.

**Principais Funcionalidades da Aplicação**

1. Cadastro e login de estudantes;
2. Seleção de exames simulados por disciplina;
3. Resolução online com temporizador;
4. Correção automática e feedback;

5. Estatísticas de desempenho;
6. Relatórios por exame realizado e por usuário.

**Fluxo de Dados no Sistema**

AUTENTICAÇÃO: O estudante faz login na aplicação através do *frontend* (*Bulma*). O sistema verifica as credenciais no *backend* (PHP) e retorna uma resposta de sucesso ou falha.

SUBMISSÃO DE TESTES: O estudante visualiza as questões e fornece respostas, que são enviadas ao *backend*. O backend processa as respostas e regista no banco de dados.

GERAÇÃO DE RELATÓRIOS: O *backend* calcula o desempenho do estudante, acessa as informações do banco de dados e gera relatórios que são enviados de volta ao *frontend* para visualização.

ARMAZENAMENTO E RECUPERAÇÃO DE DADOS: O banco de dados *MySQL* armazena todos os dados relacionados aos estudantes, questões e resultados de exames. O *backend* acessa esses dados conforme necessário para gerar relatórios ou validar informações.

**Exemplo de um fluxo simples que pode ser realizado no sistema (passo à passo):**

1. O usuário acessa o site.
2. O sistema direcciona-o para a página inicial.
3. O usuário clica no link para a página de cadastro.
4. O usuário cadastra-se no *SPEX* com sucesso.
5. O sistema reencaminha-o de volta à página inicial.
6. O usuário faz login.
7. O sistema encaminha-o para a página do seu próprio *dashboard*.
8. O usuário clica no link para a página de Exames.
9. Na página de exames, seleciona “exame de universidades”.
10. Seleciona um exame da lista de exames apresentados.
11. O sistema processa o pedido e de seguida apresenta o exame solicitado.
12. O usuário responde às questões e envia o exame para validação pelo sistema.
13. O sistema corrige o exame automaticamente.

14. O resultado e as estatísticas são exibidos para o usuário.
15. Os dados ficam salvos no histórico do usuário para relatórios futuros.
16. O usuário é redirecionado para o *dashboard*.

**Ferramentas e Tecnologias Utilizadas**

i. PHP: linguagem utilizada para o desenvolvimento do backend.

ii. MySQL: sistema de gerenciamento de bases de dados no qual foi implementado o banco de dados da aplicação.

iii. Visual Studio Code (VS Code): editor de código no qual foram escritos os códigos PHP, HTML, CSS e Javascript. Também foi utilizado para manter o versionamento do projecto com Git e no GitHub.

iv. Git e GitHub: ferramentas de versionamento usadas no projecto.

v. Navegadores Web (Chrome, Edge, Firefox, Opera, TOR): utilizados para renderizar os códigos e testar as implementações.

vi. Bulma (v1.0.2): framework CSS moderno baseado em Flexbox6.

vii. Font Awesome Free (v6.7.1 - web): biblioteca de ícones em formato SVG.

**Escolhas de Implementação**

A implementação do projeto SPEX (Sistema de Preparação para Exames) foi fundamentada na escolha criteriosa de ferramentas e tecnologias que equilibram eficiência, acessibilidade, robustez e facilidade de manutenção. As decisões técnicas visaram, sobretudo, atender às necessidades do projeto, garantindo um ambiente de desenvolvimento ágil, colaborativo e com boa experiência para o usuário final. A seguir, detalham-se as principais justificativas para as tecnologias adotadas:

PHP: a linguagem PHP foi escolhida para o desenvolvimento do backend da aplicação por ser amplamente utilizada no desenvolvimento web, possuir vasta documentação, uma comunidade ativa e por permitir fácil integração com bancos de dados MySQL. Além disso, sua sintaxe acessível e recursos nativos para manipulação de

formulários e sessões facilitaram a implementação de funcionalidades como autenticação de usuários e manipulação de dados.

MySQL: o sistema de gerenciamento de banco de dados MySQL foi selecionado por sua confiabilidade, escalabilidade e compatibilidade com o PHP. Sendo uma solução de código aberto amplamente consolidada, o MySQL oferece bom desempenho em aplicações web, sendo ideal para armazenar as informações dos usuários, questões, resultados de simulações e outras entidades do sistema.

Visual Studio Code (VS Code): foi adotado como editor de código principal pela sua leveza, recursos de autocompletar, depuração integrada e suporte a múltiplas linguagens. Seus plugins permitiram uma integração eficiente com Git e GitHub, além de facilitar a escrita e organização dos códigos HTML, CSS, JavaScript e PHP.

Git e GitHub: a utilização do Git e do GitHub foi essencial para o versionamento do projeto, permitindo o controle das alterações de código, o trabalho colaborativo entre os membros da equipe e a rastreabilidade de bugs e melhorias. O uso dessas ferramentas também proporcionou organização e segurança no desenvolvimento, evitando a perda de dados e facilitando a gestão das versões.

Navegadores Web (Chrome, Edge, Firefox, Opera, TOR: a diversidade de navegadores utilizados teve como objetivo garantir a responsividade e compatibilidade do sistema em diferentes ambientes. Testar a aplicação nos principais navegadores do mercado permitiu verificar o comportamento visual e funcional do sistema, corrigindo possíveis inconsistências e assegurando uma experiência fluida ao usuário final.

Bulma (v1.0.2): o framework CSS Bulma foi escolhido por sua abordagem moderna baseada em Flexbox, o que facilitou a criação de layouts responsivos e limpos sem a necessidade de escrever muitos estilos personalizados. Bulma contribuiu para um desenvolvimento mais rápido e com menos complexidade, favorecendo também a padronização visual da aplicação.

Font Awesome Free (v6.7.1 - Web): a biblioteca Font Awesome foi incorporada para enriquecer a interface com ícones visuais modernos e intuitivos. A presença de ícones melhora a navegabilidade e a compreensão do usuário sobre as ações possíveis no sistema, contribuindo para uma usabilidade mais acessível e agradável.

### Estrutura de Diretórios do SPEX

A estrutura de diretórios para o sistema SPEX segue uma organização lógica para o desenvolvimento e a manutenção do código. A figura 1 apresenta a estrutura de directórios do SPEX.

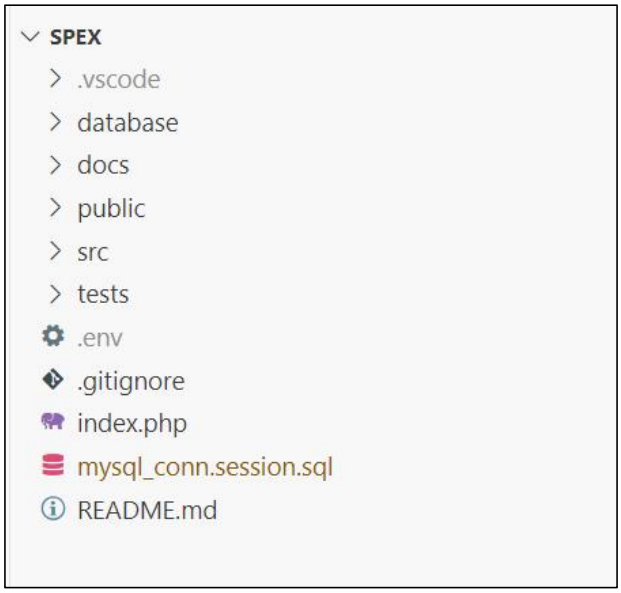


*Figura 1 - Estrutura de directórios do SPEX.*

### Boas Práticas de Desenvolvimento

Ao implementar o sistema SPEX, seguimos algumas boas práticas de desenvolvimento para garantir a escalabilidade, segurança e manutenção eficiente da aplicação.

Controle de Versão: utilizou-se o *Git* e o *GitHub* para versionar o código e gerenciar o histórico de alterações.

Desenvolvimento Modular: organizou-se o código de forma modular, separando responsabilidades de maneira clara entre controladores, modelos e visualizações.

Validação de Dados: implementou-se mecanismos de validações de entrada rigorosas para evitar problemas de segurança, como injeções *SQL*.

Armazenamento Seguro de Senhas: utilizou-se funções de *hash*, como *bcrypt*, para armazenar senhas de forma segura no banco de dados.

## CONFIGURAÇÃO DO SERVIDOR WEB

A configuração do servidor web é uma etapa crucial para garantir que a aplicação *SPEX* esteja disponível e acessível aos usuários. O servidor web escolhido para este projeto é o *NGINX*, uma solução altamente eficiente, conhecida por sua baixa utilização de recursos e por ser capaz de lidar com uma grande quantidade de conexões simultâneas. O *NGINX* será utilizado em conjunto com *PHP-FPM (FastCGI Process Manager)* para garantir a execução eficiente dos *scripts PHP* utilizados pela aplicação *SPEX*. Este capítulo descreve o processo de instalação e configuração do *NGINX* com *PHP-FPM*, destacando as etapas essenciais para que a aplicação funcione de maneira estável e segura, mesmo sem o uso de *SSL/TLS* e *HTTPS*.

A escolha do *NGINX* foi motivada pela sua eficiência no processamento de requisições *HTTP*, especialmente em cenários de alta carga. O *NGINX* é uma solução robusta, amplamente utilizada para servir aplicações web de alto desempenho devido à sua capacidade de manuseio de múltiplas conexões simultâneas de forma eficiente. A combinação do *NGINX* com o *PHP-FPM* garante que as páginas dinâmicas geradas em *PHP* sejam processadas rapidamente, sem sobrecarregar o servidor. Embora outras opções como o *Apache* ou *LiteSpeed* também possam ser usadas, o *NGINX* foi escolhido devido à sua leveza e à facilidade de configuração em ambientes de produção.

### Configuração do Servidor Web

Para instalar o *NGINX* em um sistema baseado em *Ubuntu*, os seguintes comandos podem ser utilizados:

*sudo apt update*

*sudo apt install nginx*

Após a instalação, o *NGINX* pode ser iniciado e configurado para iniciar automaticamente na inicialização do sistema:

*sudo systemctl start nginx*

*sudo systemctl enable nginx*

**Instalação do PHP e PHP-FPM**

O PHP-FPM é uma solução para o gerenciamento de processos PHP em servidores web. Para instalar o PHP e o PHP-FPM no servidor, execute os seguintes comandos:

*sudo apt install php-fpm php-mysql*

Após a instalação do PHP-FPM, a configuração padrão do PHP-FPM deve ser verificada para garantir que o serviço está configurado corretamente. A principal configuração que deve ser ajustada é o valor de user e group no arquivo de configuração do PHP-FPM (*/etc/php/8.x/fpm/pool.d/www.conf*), garantindo que o PHP-FPM seja executado sob o usuário correto (normalmente *www-data*):

*user = www-data*

*group = www-data*

**Configuração do NGINX para PHP**

Uma vez que o *NGINX* e o PHP-FPM estejam instalados, a configuração do *NGINX* para processar arquivos PHP deve ser realizada.

*1.* Editar o arquivo de configuração do servidor (*site default*): O arquivo de configuração do *NGINX* para o servidor web geralmente está localizado em */etc/nginx/sites-available/default.*

Este arquivo precisa ser modificado para incluir o bloco de configuração que define como o NGINX irá interagir com o PHP-FPM.

2. Configuração do Bloco Server: Dentro do arquivo de configuração do *NGINX*, altere ou adicione o seguinte conteúdo no bloco *server:*

```
server {
    listen 80;
    server_name localhost;
    root /var/www/html;
    index index.php index.html index.htm;

    location / {
        try_files $uri $uri/ =404;
    }

    location ~ \.php$ {
        include snippets/fastcgi-php.conf;
        fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;
    }

    location ~ /\.ht {
        deny all;
    }
}
```

*Figura 2 - Arquivo de configuração do servidor.*

**Explicação das configurações**

- listen 80: O servidor irá escutar na porta 80 (HTTP).
- *root /var/www/html*: Define o diretório onde os arquivos do SPEX serão armazenados. *location ~ \.php$*: Configura o NGINX para processar arquivos PHP utilizando o PHP-FPM.
- *fastcgi_pass unix:/var/run/php/php8.x-fpm.sock*: Define o socket que o NGINX usará para se comunicar com o PHP-FPM.

**Reiniciando o *NGINX***

Após modificar a configuração, foi necessário reiniciar o NGINX para aplicar as alterações. Para tal, executou-se, no terminal, o comando: *sudo systemctl restart nginx*

**Segurança e Acessos**

Embora o servidor web *SPEX* não utilize SSL/TLS nem HTTPS, era importante manter algumas práticas de segurança para garantir o acesso seguro ao servidor.

1. Firewall: Utilizou-se o UFW (Uncomplicated Firewall) para permitir apenas conexões HTTP (porta 80) ao servidor:

*sudo ufw allow 'Nginx HTTP'*

*sudo ufw enable*

Permissões de Arquivos: Garantir que os arquivos e diretórios do servidor web possuam permissões adequadas. O proprietário dos arquivos deve ser o usuário *www-data*, e os arquivos devem ter permissões restritas.

3. Logs de Acesso e Erro: O *NGINX*, por padrão, mantém logs de acesso e erro que são essenciais para o monitoramento do servidor. Os logs podem ser encontrados em:

Logs de acesso: */var/log/nginx/access.log*

Logs de erro: */var/log/nginx/error.log*

Monitorar esses logs regularmente ajuda a identificar problemas e possíveis ataques ao servidor.

**Teste e Validação**

Após configurar o servidor web, é essencial realizar testes para garantir que a aplicação *SPEX* esteja funcionando corretamente. Algumas etapas de teste incluem:

a) Acessar a aplicação pelo navegador utilizando o endereço IP ou nome de domínio do servidor.

b) Testar se as páginas PHP estão sendo processadas corretamente.

c) Verificar os logs de erro para garantir que não há problemas de configuração ou falhas de *script*.

# CONFIGURAÇÃO DA REDE CLIENTE-SERVIDOR

A configuração da rede cliente-servidor é uma parte fundamental para garantir a comunicação entre os diversos componentes do sistema *SPEX*. O modelo cliente-servidor estabelece a interação entre o servidor web, que hospeda a aplicação, e os clientes, que acessam a aplicação remotamente. Neste capítulo, serão abordadas as configurações necessárias para garantir que a comunicação entre o servidor e os dispositivos dos usuários (clientes) seja segura e eficiente, além de explicar os componentes envolvidos.

A infraestrutura de rede do Sistema de Preparação para Exames (*SPEX*) foi temporariamente configurada para funcionar em um ambiente de rede local, com o objetivo de demonstrar o sistema ao público durante a apresentação do projeto, sem necessidade de conexão à Internet ou custos com hospedagem externa. Essa escolha não compromete a escalabilidade futura da aplicação, que foi projetada para ser implantada também em servidores com acesso à Internet, mediante as devidas configurações de segurança e serviços de hospedagem.

## Arquitetura Cliente-Servidor

A arquitetura cliente-servidor envolve a comunicação entre o servidor, que executa a aplicação *SPEX*, e os clientes, que fazem solicitações *HTTP* para acessar os recursos da aplicação. Em um cenário típico:

Servidor: Hospeda a aplicação web, o banco de dados e outros serviços relacionados.

Clientes: Podem ser dispositivos como computadores, smartphones ou outros dispositivos que se conectam ao servidor via rede para acessar a aplicação.

A comunicação ocorre geralmente por meio de *HTTP* ou *HTTPS* (no caso de servidores com *SSL/TLS* configurados), utilizando o protocolo de rede *TCP/IP*.

## Topologia da Rede

A rede local foi configurada com dois servidores principais e múltiplos clientes conectados via cabo Ethernet ou *Wi-Fi*:

| DISPOSITIVO | IP ESTÁTICO | FUNÇÕES |
|---|---|---|
| Servidor A | 192.168.10.10 | Aplicação Web (NGINX + PHP-FPM), DNS, DHCP |
| Servidor B | 192.168.10.20 | Banco de Dados (MySQL) |
| Clientes | 192.168.10.100+ | IPs dinâmicos via DHCP, acesso via *spex.edu.ao* |

*Tabela 1 - Topologia da Rede Cliente-Servidor para o SPEX*

## Configuração da Rede do Servidor

### Configuração do IP Estático

A configuração de IP estático em uma máquina Ubuntu envolve a alteração dos arquivos de configuração de rede para garantir que o sistema sempre use o mesmo endereço IP.

### Passo 1: Identificar a interface de rede

Primeiro, identifique o nome da interface de rede que você deseja configurar. Para isso, abra o terminal e execute o seguinte comando: *ip a*

O comando acima lista todas as interfaces de rede disponíveis. Normalmente, a interface *Ethernet* é chamada de *enp0s3* (ou algo semelhante) e a interface de *Wi-Fi* pode ser chamada de *wlan0*.

### Passo 2: Editar o arquivo de configuração de rede

O *Ubuntu* usa o *Netplan* para configurar redes a partir da versão 18.04. Deve-se editar o arquivo de configuração do *Netplan* que geralmente está localizado em */etc/netplan/*. O arquivo padrão pode ser chamado algo como *01-netcfg.yaml*, mas o nome exato pode variar.

Abrindo o terminal e editando o arquivo de configuração com o editor de texto, nano:

*sudo nano /etc/netplan/01-netcfg.yaml*

```
network:
  version: 2
  renderer:
    ethernets:
      enp0s3:
      dhcp4: no  # Desabilita o DHCP (atribuição automática de IP)
      addresses:
        - 192.168.10.10/24  # Endereço IP estático e máscara de sub-rede
      gateway4: 192.168.10.1 # Gateway da rede local
      nameservers:
        addresses:
          - 192.168.10.10 # Servidor DNS local (geralmente o gateway é o DNS)
```

*Figura 3 - Configuração da rede Ethernet.*

**Configuração do Servidor A (Servidor Web, DNS e DHCP) com Wi-Fi**

O Servidor A, que hospeda os serviços de servidor web (NGINX + PHP-FPM), servidor DNS (Bind9) e servidor DHCP, foi configurado para suportar conexões tanto via cabo quanto via Wi-Fi.

**Configuração de Rede para Wi-Fi no Servidor A**

No Servidor A, foi configurado um IP estático para a interface de rede sem fio (Wi-Fi). O servidor foi configurado para conectar-se à rede sem fio, com o adaptador de rede wlan0, e receber o IP estático 192.168.10.10.

**Passos para configurar a rede Wi-Fi:**

Editar o arquivo de configuração de rede:

*sudo nano /etc/netplan/00-installer-config.yaml*

Adicionar as configurações para a rede Wi-Fi (ver figura _____).

Com isso, o Servidor A estará conectado à rede Wi-Fi com o IP estático configurado.

Após a configuração, o arquivo deve ser salvo e as configurações aplicadas com:

*sudo netplan apply*

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s3:
    dhcp4: no  # Desabilita o DHCP (atribuição automática de IP)
    addresses:
      - 192.168.10.10/24  # Endereço IP estático e máscara de sub-rede
    gateway4: 192.168.10.1 # Gateway da rede local
    nameservers:
      addresses:
        - 192.168.10.10 # Servidor DNS local (geralmente o gateway é o DNS)
  wifis:
    wlan0:
      dhcp4: no
      addresses:
        - 192.168.10.11/24
      gateway4: 192.168.10.1
      nameservers:
      addresses:
       - 192.168.10.10
      access-points:
        "SPEX-NET":
        password: "senha-secreta"  # apenas como exemplo

```

*Figura 4 - Configuração da rede Ethernet e Wi-Fi.*

### Conexão dos Clientes via Wi-Fi

Os clientes que necessitam acessar o sistema SPEX também foram configurados para se conectar à mesma rede Wi-Fi. Esses dispositivos podem acessar o sistema através do nome de domínio local spex.edu.ao, sem a necessidade de usar o endereço IP diretamente.

Os clientes que utilizam Wi-Fi obterão seus IPs automaticamente por meio do servidor DHCP, que foi configurado no Servidor A para distribuir endereços IPs dentro da faixa definida (exemplo: 192.168.10.100+).

Configuração semelhante foi aplicada no Servidor B, utilizando o IP 192.168.10.20.

### Serviços de Rede

DHCP (ISC DHCP Server): Responsável por fornecer IPs automáticos aos clientes. Exemplo de intervalo:

*range 192.168.10.100 192.168.10.200;*

DNS (Bind9): Instalado no Servidor A para permitir a futura resolução de nomes amigáveis na rede local. No Servidor A, criou-se o arquivo de zona para *spex.edu.ao*, em */etc/bind/db.spex.edu.ao.*

```
$TTL    604800
@       IN      SOA     spex.edu.ao. admin.spex.edu.ao. (
                              3         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL

@       IN      NS      spex.edu.ao.
@       IN      A       192.168.10.10
www     IN      A       192.168.10.10
```

**Atualização do Arquivo de Zona no Bind9**

Adicionou-se a zona no arquivo */etc/bind/named.conf.local:*

```
zone "spex.edu.ao" {

    type master;

    file "/etc/bind/db.spex.edu.ao";

};
```

Depois dessa etapa reiniciou-se o Bind9:

*sudo systemctl restart bind9*

**Testes**

Testou-se em uma computador cliente: *ping spex.edu.ao*

O resultado foi algo parecido com: *PING spex.edu.ao (192.168.10.10): 56 data bytes*

**Conexão com o Site pelo Nome de Domínio**

A partir desse ponto, os clientes conectados à rede local (com DNS apontando para o servidor A) poderão acessar o sistema via navegador com: *http://spex.edu.ao*

# RESULTADOS DO TRABALHO

O SPEX foi desenvolvido como uma ferramenta de simulação e preparação para exames, focada na educação de estudantes. O sistema foi projetado para permitir que os usuários (estudantes) simulem exames, revisem conteúdos relacionados aos temas abordados nos testes e obtenham feedback sobre seu desempenho.

A estrutura do sistema inclui funcionalidades como criação de exames, administração de questões, simulação de provas ou exames, visualização de resultados e análise de desempenho.

## Funcionalidades Implementadas

As funcionalidades implementadas no sistema foram:

Cadastro de Usuários: os estudantes podem criar uma conta e acessar o sistema com base nas suas credenciais.

Criação de Exames: os administradores podem criar questões, definindo os cursos, as disciplinas e os temas para os quais eles se aplicam.

Simulação de Exames: os estudantes podem realizar exames simulados com base nas questões criadas pelos administradores do sistema.

Resultados e Feedback: Após a realização dos exames, os estudantes recebem uma pontuação com base nas respostas correctas, junto com feedback detalhado sobre o seu desempenho.

Gestão de Conteúdos: os administradores podem disponibilizar aulas de diferentes disciplinas e temas no site para facilitar o estudo por parte dos estudantes; assim como as questões que serão utilizadas para gerar os exames.

Relatórios de Desempenho: relatórios detalhados sobre o desempenho dos estudantes nas provas simuladas, permitindo análise por disciplina, área de conhecimento e nível de dificuldade.

## Resultados dos Testes Realizados

Os testes realizados durante o desenvolvimento do SPEX ajudaram a validar a eficácia da aplicação e a identificar áreas de melhoria. Os principais testes incluem:

**Testes de Funcionalidade**

Cadastro de Usuários: verificou-se que o fluxo de registro e login estava funcionando correctamente, permitindo que os usuários se registrassem e acessassem as suas contas sem problemas.

Simulação de Provas: Os exames simulados estavam a gerar questões aleatórias de acordo com as preferências definidas pelos estudantes, e as respostas estavam a ser correctamente avaliadas.

Gerenciamento de Exames e Conteúdos: a criação, edição e exclusão de exames e questões estavam a funcionar conforme esperado, permitindo que os administradores mantivessem os conteúdos atualizados e relevantes.

**Testes de Performance**

Tempo de Resposta: durante os testes de carga, o sistema se comportou bem sob condições normais de uso, com tempos de resposta rápidos ao consultar questões e ao registrar respostas de usuários.

Capacidade de Escalabilidade: A aplicação foi projetada para escalar, entretanto, ainda não foram realizado testes com uma quantidade maior de usuários simultâneos. Mas espera-se que o sistema não revele problemas significativos de performance.

**Testes de Segurança**

Autenticação e Autorização: o sistema implementa autenticação segura com verificação de credenciais através de hash de senhas. A autorização de usuários foi configurada correctamente, garantindo que apenas usuários com permissões apropriadas pudessem acessar determinadas funcionalidades.

Proteção Contra Injeções: o código foi revisado para prevenir injeções de SQL e ataques XSS. Todas as entradas dos usuários são correctamente validadas e sanitizadas.

Testes de Vulnerabilidade: não foram realizados testes de vulnerabilidade suficientes para verificar a segurança da aplicação, de forma a permitir identificar e corrigir pontos sensíveis, como proteção de dados e transações seguras.

**Testes de Usabilidade**

Os testes de usabilidade realizados até a altura indicaram que o sistema é intuitivo para os estudantes, com navegação simples e funcionalidades claramente definidas. Entretanto, melhorias ainda podem ser feitas.

**Desempenho do Sistema**

O desempenho geral do sistema foi satisfatório, com tempos de resposta rápidos e sem gargalos significativos durante a execução de tarefas comuns. O uso do servidor NGINX com PHP-FPM ajudou a otimizar o desempenho da aplicação, garantindo uma boa resposta mesmo em momentos de maior carga

**Impacto no Contexto da Educação**

O sistema SPEX apresenta um grande potencial para transformar a preparação para exames, oferecendo aos estudantes uma plataforma interativa e eficaz para praticar e revisar conteúdos.

A capacidade de realizar simulações de exames em um ambiente controlado contribui para a redução da ansiedade dos estudantes e melhora a confiança no desempenho real durante os exames oficiais.

# REFLEXÃO CRÍTICA E PROBLEMAS ENCONTRADOS

Este capítulo tem como objetivo refletir sobre o processo de desenvolvimento e implementação do sistema SPEX, destacando tanto os sucessos quanto os desafios enfrentados durante o percurso. Serão abordados os principais problemas encontrados ao longo do projeto, as soluções adotadas para solucioná-los, e as lições aprendidas que podem ser aplicadas em projetos futuros. A reflexão crítica também incluirá uma análise das limitações do sistema e sugestões para aprimoramentos que visem aumentar a sua eficácia e escalabilidade.

O objetivo desta reflexão é proporcionar uma visão mais profunda sobre o processo de desenvolvimento do SPEX, não apenas destacando os resultados alcançados, mas também compreendendo as falhas e os desafios que surgiram. Com isso, será possível identificar áreas para melhoria contínua e aplicar essas lições em futuras iterações do sistema.

## Problemas Encontrados Durante o Desenvolvimento

Durante o desenvolvimento do SPEX, diversos problemas surgiram, desde questões técnicas até questões de natureza pessoal que impactaram na realização do projecto dentro dos prazos definidos. Os principais problemas enfrentados foram:

1) Dificuldades para implementar todas as funcionalidades no *backend* e todas as páginas do *frontend* que foram previstas, devido à escassez do tempo.
2) Dificuldade para instalar o sistema operacional *Linux Ubuntu*, o que motivou à mudança para o *Debian*.
3) Dificuldade para desenvolver o projecto de forma distribuída (com cada membro do grupo realizando as funções que lhe foram inicialmente atribuídas), devido a falta de experiência que foi manifestada.
4) Alguns membros viram-se muitas vezes impedidos de comparecer nos encontros programados por causa de assuntos de ordem pessoal; as situações de doença que

foram sendo manifestadas por alguns membros do grupo também contribuíram para um atraso na realização do trabalho.

# CONCLUSÃO E TRABALHOS FUTUROS

O desenvolvimento do Sistema de Preparação para Exames (SPEX) representa uma importante contribuição para o contexto educacional, ao oferecer uma plataforma moderna e interativa de apoio ao estudo. A adoção de tecnologias robustas e boas práticas de engenharia de software permitiu a criação de uma aplicação segura, modular e de fácil manutenção. O projeto também possibilitou à equipe a aplicação prática de conhecimentos teóricos adquiridos ao longo da formação, além de desenvolver habilidades de trabalho em equipe, gestão ágil e resolução de problemas técnicos. Apesar dos desafios encontrados, como a configuração dos serviços de rede, os objetivos do projecto foram alcançados com sucesso, culminando em um sistema funcional e com potencial de expansão. Espera-se que o SPEX evolua continuamente, incorporando novas funcionalidades e contribuindo cada vez mais para o sucesso académico dos estudantes.

Embora o SPEX tenha sido bem-sucedido na implementação de suas funcionalidades principais, existem áreas que podem ser melhoradas no futuro, incluindo:

Implementação de SSL/TLS e HTTPS: Embora o sistema esteja operando sem criptografia no momento, a implementação de SSL/TLS é altamente recomendada para garantir a segurança das transações e a privacidade dos usuários.

**Expansão de Funcionalidades**

A implementação de funcionalidades como feedback personalizado e recomendações de estudo baseadas no desempenho dos estudantes pode tornar a plataforma ainda mais útil.

## REFERÊNCIAS BIBLIOGRÁFICAS


Canonical Ltd. (n.d.). *Documentação Oficial do Ubuntu*. Recuperado de https://help.ubuntu.com/lts/pt_BR/

F5 Networks. (n.d.). *Documentação do NGINX*. Recuperado de https://docs.nginx.com/nginx/pt/

Fonticons, Inc. (n.d.). *Documentação do Font Awesome*. Recuperado de https://fontawesome.com/docs

García, J. (n.d.). *Documentação Oficial do Bulma*. Recuperado de https://bulma.io/documentation/

IBM. (n.d.). *O Modelo Cliente/Servidor*. Recuperado de https://www.ibm.com/docs/pt-br/cicsts/6.x?topic=interfaces-clientserver-model

Oracle Corporation. (n.d.). *Documentação do MySQL 8.4*. Recuperado de https://dev.mysql.com/doc/refman/8.4/pt/

The Debian Project. (n.d.). *Documentação do Debian*. Recuperado de https://www.debian.org/doc/index.pt.html

The PHP Group. (n.d.). *Gerenciador de Processos FastCGI (FPM)*. Recuperado de https://www.php.net/manual/pt_BR/install.fpm.php

The PHP Group. (n.d.). *Manual do PHP*. Recuperado de https://www.php.net/manual/pt_BR/index.php

W3C. (n.d.). *Especificações do CSS*. Recuperado de https://www.w3.org/Style/CSS/specs.pt.html

WHATWG. (n.d.). *Padrão HTML*. Recuperado de https://html.spec.whatwg.org/multipage/

# APÊNDICES

Diagramas UML Utilizados no Desenvolvimento do Sistema