**Easy Question - Arquitetura**

O aplicativo Easy Question tem como objetivo facilitar a vida do usuário que deseja tanto criar questões quanto gerar provas, proporcionando uma maior qualidade de avaliações institucionais, ao oferecer um sistema de auditoria, um banco de questões de fácil acesso e um design intuitivo e objetivo.

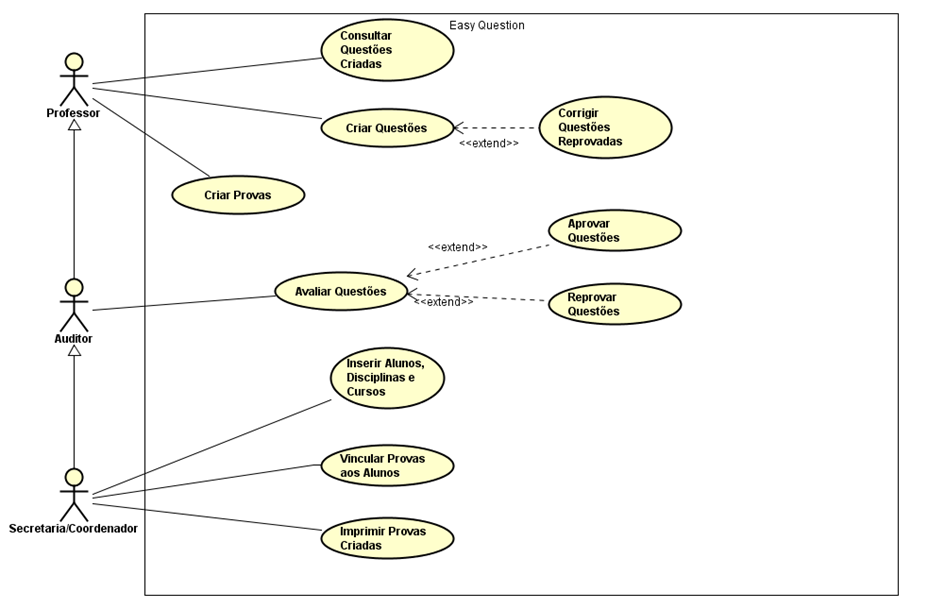
**Quadro 1** – Requisitos Funcionais

|  |  |  |
| --- | --- | --- |
| **ID** | **Requisito** | **Descrição** |
| RF1 | Cadastro de Usuário | Permitir o registro e a autenticação de diferentes tipos de usuário, como professores, coordenadores e administradores |
| RF2 | Criação de Questões | Permitir que os professores criem e armazenem questões para futuras avaliações. |
| RF3 | Revisão de Questões | Permitir que os professores revisem e editem questões existentes. |
| RF4 | Montagem de Provas | Permitir que os professores combinem questões de diferentes tópicos para criar provas personalizadas. |
| RF5 | Distribuição de Provas | Permitir a distribuição de provas para os alunos de maneira segura e organizada. |
| RF6 | Relatórios de Desempenho | Gerar relatórios detalhados de desempenho dos alunos com base nas provas realizadas. |

**Quadro 2** – Requisitos Não Funcionais

|  |  |  |
| --- | --- | --- |
| **ID** | **Requisito** | **Descrição** |
| RNF1 | Usabilidade Intuitiva | Garantir que a interface do usuário seja intuitiva e fácil de usar para todos os níveis de experiência em tecnologia. |
| RNF2 | Desempenho Rápido e  Confiável | Assegurar tempos de resposta rápidos e operação confiável, mesmo sob alta carga de usuários. |
| RNF3 | Segurança Robusta | Implementar medidas de segurança robustas para proteger dados sensíveis dos usuários e garantir a confidencialidade. |
| RNF4 | Escalabilidade | Garantir que o sistema possa escalar de forma eficiente com o crescimento da base de dados e aumento de usuários. |
| RNF5 | Manutenção Fácil | Assegurar que o código seja de fácil manutenção para permitir atualizações e melhorias contínuas sem interrupções. |
| RNF6 | Compatibilidade | Garantir que o sistema funcione em diversos navegadores e dispositivos sem problemas de compatibilidade. |

**Visão de casos de uso:**



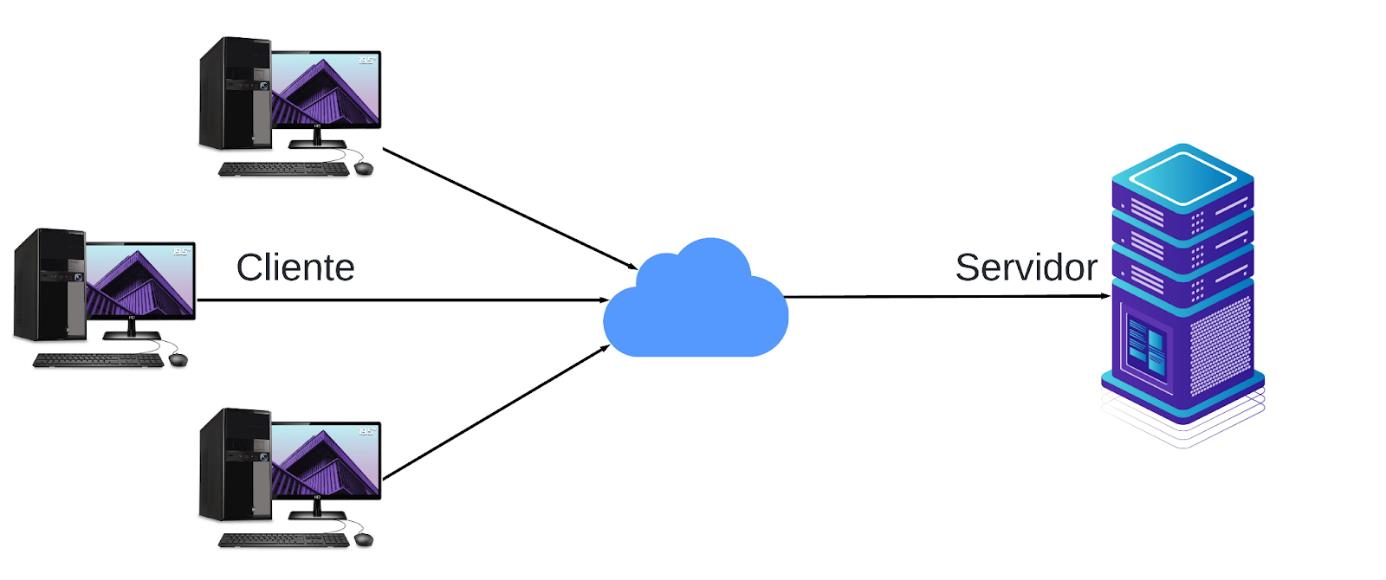
**Mecanismos arquiteturais:**

**Quadro 3** – Mecanismos arquiteturais

|  |  |  |
| --- | --- | --- |
| **MECANISMO DE ANÁLISE** | **MECANISMO DE DESIGN** | **MECANISMO DE IMPLEMENTAÇÃO** |
| Persistência | Banco de dados relacional | PostgreSQL |
| Recursos avançados de Web para consumo | Implementação de recursos para consumo. | API Rest |
| Camada de distribuição | Classe de comunicação com o banco, classe de persistência. | API Rest |
| Front-End | Interface de comunicação com o usuário do portal. | Framework React utilizando Typescript e Tailwind como *framework* de CSS |
| Back-End | Lógica e regra de negócio do aplicativo | API Rest com Node JS utilizando Express como *framework* de HTTP |
| Tratamento de exceções | Camada para tratar as exceções criando interações diferentes para usuários e técnicos. | *Middleware* para tratar erros e não estourar na cara do usuário |
| Build | Programação da IDE para validação do código fonte. | Visual Studio Code |

**Implantação:**

A equipe optou por utilizar a arquitetura Cliente-Servidor, por sua eficiência e simplicidade na troca de informações entre o usuário e o sistema. Nesse modelo, o cliente representa a interface que o usuário vê e com a qual interage, como telas de *login*, consulta de questões, provas, atualização e inserção de dados. Toda vez que o usuário realiza uma ação, como enviar dados ou fazer uma consulta, essa informação é enviada ao servidor que, por sua vez, funciona como uma central que processa essas solicitações e responde com os dados necessários. Essa separação entre o que o usuário vê (cliente) e o que acontece nos bastidores (servidor) torna o sistema mais organizado, facilitando o gerenciamento e a escalabilidade da aplicação.



**Conclusão:**

Com base nos estudos sobre a aplicação e reuniões com os principais interessados foram feitas várias análises para que fosse escolhida a arquitetura correta para o desenvolvimento fazendo com que fossem supridas as necessidades.