GamaTrack

Documento de Arquitetura

Versão 1.0

Histórico de Revisão

Data	Versão	Descrição	Autor(es)
02/11/2023	1.0	Versão inicial do documento	Eric Rabelo Nicollas Gabriel Isaque Colem Samuel Ribeiro Rodrigo Braz

Autores:

Matrícula	Nome	Descrição do papel assumido na equipe	% de contribuição ao trabalho
21/1030729	Eric Rabelo	Desenvolvedor Frontend	20%
22/1022300	Isaque Colem	Desenvolvedor Frontend	20%
21/1062802	Nicollas Gabriel	Desenvolvedor Backend	20%
21/1031486	Samuel Ribeiro	Desenvolvedor Backend	20%
19/0116498	Rodrigo Braz	Desenvolvedor Backend	20%

Sumário

1 Introdução

1.1 Propósito

Este documento descreve a arquitetura do sistema sendo desenvolvido pelo grupo na disciplina de MDS – Métodos de Desenvolvimento de Software – edição do segundo semestre de 2023, chamado "GamaTrack", com o objetivo de proporcionar uma visão completa do sistema para desenvolvedores, testadores e demais partes interessadas.

1.2 Escopo

O detalhamento do escopo pode ser encontrado no documento Declaração de Escopo do Produto no repositório da equipe no GitHub. Porém, em linhas gerais o escopo do produto compreende uma aplicação web na qual estudantes da FGA – UNB poderão pesquisar por professores da instituição, avaliá-los e consultar opiniões de outros estudantes sobre os mesmos.

2 Representação Arquitetural

2.1 Definições

O sistema seguirá uma arquitetura em três camadas.

2.2 Justificativa

A escolha do padrão arquitetural em três camadas foi justificada pela equipe devido à sua simplicidade e clara divisão de responsabilidades. Isso permite que subequipes possam trabalhar paralelamente em diferentes áreas do projeto sem conflitos, aumentando a eficiência do desenvolvimento. A camada de frontend construída com React.js lida com a interface do usuário, a camada de backend com Node.js abrange a lógica de negócios e as APIs, e o MongoDB serve como o banco de dados, cada um com funções bem definidas. Isso facilita a manutenção, escalabilidade e colaboração entre os membros da equipe, garantindo um desenvolvimento mais organizado e eficaz do projeto como um todo.

2.3 Detalhamento

O padrão arquitetural em três camadas promove uma estrutura sólida e organizada. Cada camada desempenha um papel específico no sistema, o que permite uma clara divisão de responsabilidades e facilita o desenvolvimento colaborativo.

Camada de Apresentação (Frontend com React.js): Nesta camada, temos a interface de usuário (UI) responsável por interagir com os usuários finais. Ela inclui componentes de interface que exibem formulários de avaliação, resultados de pesquisas, e outras informações relevantes. A camada de apresentação também lida com a lógica de exibição e a comunicação com o backend por meio de requisições HTTP para acessar os dados e serviços.

Camada de Lógica de Aplicação (Backend com Node.js): No backend, temos a lógica de negócios do sistema. Ela processa as solicitações do frontend, realiza validações, processa os dados, acessa o banco de dados MongoDB e executa ações como o cálculo de pontuações de professores, armazenamento de avaliações, etc. Essa camada é responsável por garantir a segurança e a consistência dos dados.

Camada de Dados (MongoDB): O MongoDB serve como o banco de dados subjacente para armazenar todas as informações relacionadas aos professores, avaliações e pesquisas. Ele é projetado para lidar com dados não estruturados, tornando-o adequado para um projeto que lida com avaliações e pesquisas, que podem ter diferentes tipos de informações. A camada de dados é acessada pela camada de lógica de aplicação para buscar e armazenar informações.

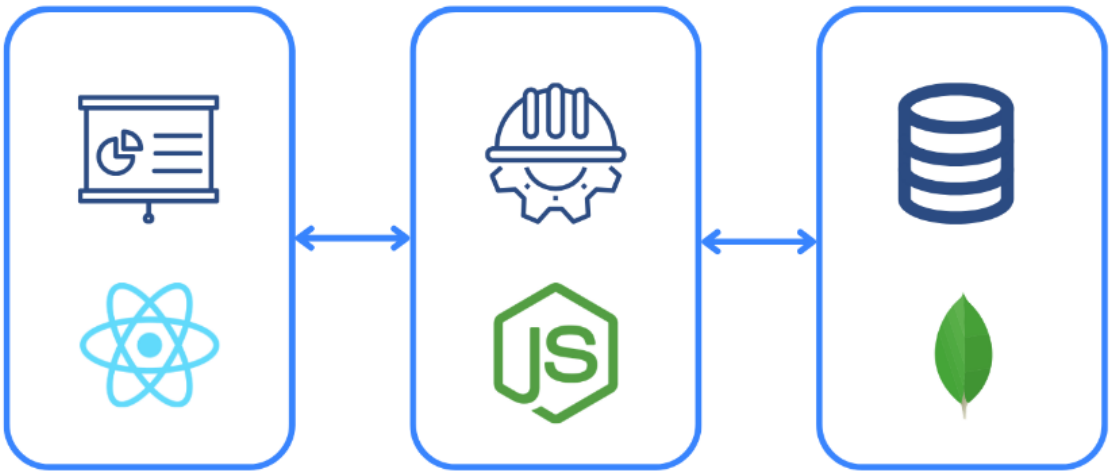

Imagem 1 - Representação das camadas

2.4 Metas e restrições arquiteturais

Segurança: O sistema deve enfatizar a segurança em todas as camadas da arquitetura, em conformidade com a Lei Geral de Proteção de Dados (LGPD) e a sensibilidade dos dados dos alunos. Isso inclui a implementação de robustos mecanismos de autenticação e autorização para proteger os dados sensíveis dos alunos, assegurando que somente usuários autorizados tenham acesso a essas informações.

Tempo de resposta: O sistema deve ser capaz de responder a 80% das requisições dos usuários em um tempo de no máximo 3 segundos. Além disso, é importante realizar testes de carga e monitoramento contínuo do desempenho do sistema para garantir que os tempos de resposta permaneçam dentro dos parâmetros especificados. Dessa forma, os usuários terão uma experiência rápida e responsiva ao utilizar o sistema.

Escalabilidade: O sistema deve ser capaz de acomodar, sem perda de desempenho, as solicitações de 50 usuários simultaneamente. Essa métrica foi estabelecida tendo em vista o uso potencial do software por alunos da FGA – UNB, uma comunidade com cerca de 5.000 alunos. Portanto, o objetivo é garantir que o sistema possa suportar um número significativo de usuários em um ambiente real de utilização, mantendo uma experiência de uso eficiente para todos.

Facilidade de manutenção do código: As camadas arquiteturais do sistema devem ser claramente separadas a fim de facilitar a manutenção do código. Isso implica em seguir boas práticas de organização e documentação do código-fonte.

Facilidade de uso do sistema pelos usuários finais: A interface do sistema deve ser amigável e intuitiva, priorizando a simplicidade e a objetividade. A finalidade é simplificar a

busca por feedback de professores, garantindo que os usuários possam navegar de forma descomplicada e encontrar as informações desejadas de maneira direta e eficaz.

2.5 Visão de Casos de uso (escopo do produto)

A visão de casos de uso do GamaTrack está alinhada com a Declaração de Escopo do Produto. Em resumo, nosso sistema permite que os usuários acessem uma plataforma web baseada em React.js para avaliar e pesquisar professores. Os casos de uso englobam registro e autenticação de usuários, pesquisa de professores, a avaliação dos mesmos e a consulta das avaliações de outros alunos. Além disso, o sistema oferecerá a funcionalidade de filtragem de professores e matérias de acordo com o curso, proporcionando uma experiência de busca mais específica e direcionada. A escolha da arquitetura em três camadas (frontend React.js, backend Node.js e MongoDB) se deve à necessidade de uma interface de usuário interativa e à gestão eficiente de dados, conforme delineado nos requisitos do projeto. A experiência técnica da equipe e a clara separação de responsabilidades em cada camada contribuíram para essa escolha arquitetural. O diagrama de casos de uso detalhado está disponível na Declaração de Escopo do Produto (Pollux, 2023, p 14-15).

2.6 Visão lógica

O sistema é subdividido nos seguintes módulos:

Módulo de Autenticação: Responsável por verificar a identidade do usuário. Composto por funcionalidades como login, logout, registro de usuário e recuperação de senha. Essencial para garantir que o cliente alvo - discentes da FGA - é o usuário, garantindo a confiabilidade das avaliações e definindo o que o Módulo de Interface de Usuário exibirá.

Módulo de Interface do Usuário: Este módulo é responsável por renderizar a interface do usuário e lidar com a interação do usuário, requisitando ao Módulo de Gerenciamento de Dados as atualizações necessárias, recebendo-as e exibindo conforme solicitado e adequado às permissões de usuário.

Módulo de Gerenciamento de Dados: Módulo responsável por todas as operações de CRUD dentro do banco de dados da aplicação, garantindo que não haja interação de dados direta entre o Módulo de Interface do Usuário e o banco de dados propriamente dito, servindo de intermédio entre os dois, assegurando assim a segurança e integridade dos dados do sistema.

A comunicação entre os módulos de autenticação, interface do usuário e gerenciamento de dados do GamaTrack ocorre da seguinte maneira:

Interface do Usuário para Autenticação: Quando um usuário tenta fazer login, criar, excluir ou editar cadastro, a interface do usuário coleta as credenciais do usuário e as envia para o módulo de autenticação.

Autenticação para Gerenciamento de Dados: O Módulo de Autenticação recebe as credenciais do usuário e as faz uma requisição para o Módulo de Gerenciamento de Dados para verificação.

Gerenciamento de Dados para Autenticação: O Módulo de Gerenciamento de Dados verifica as credenciais do usuário e retorna o resultado para o Módulo de Autenticação.

Autenticação para Interface do Usuário: O Módulo de Autenticação recebe o resultado da verificação e o envia para a interface do usuário. Se as credenciais forem válidas, o usuário será autenticado e terá acesso ao sistema.

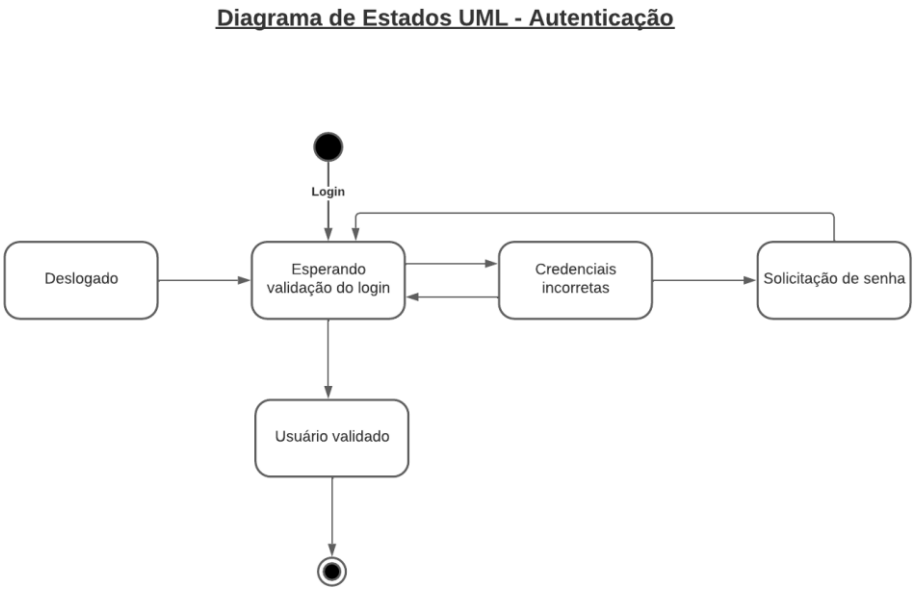

Imagem 2 – Diagrama de Estados

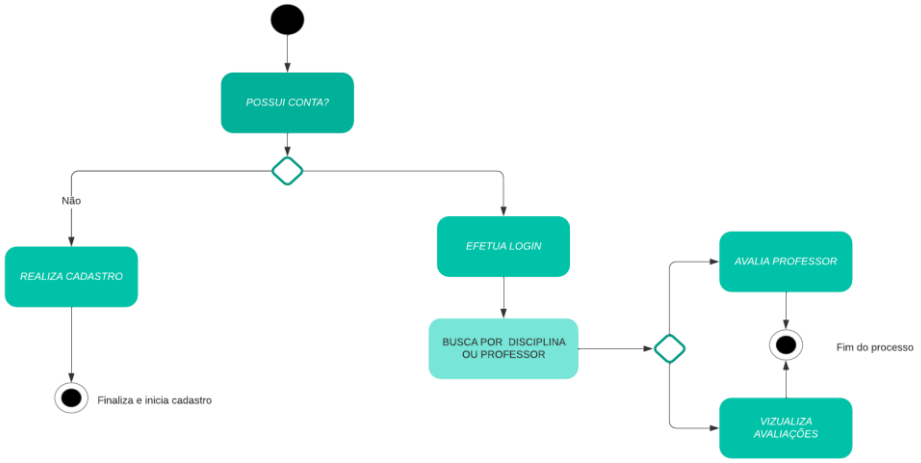

Imagem 3 – Diagrama de Atividades

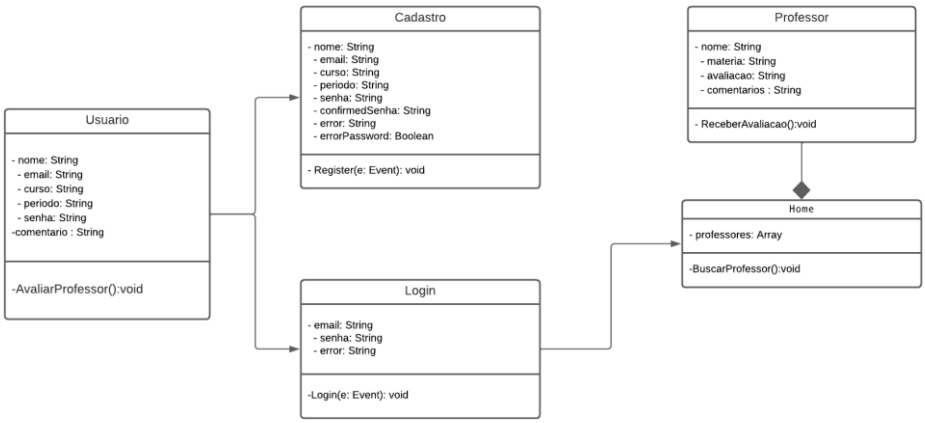

Imagem 4 – Diagrama de Classes

2.7 Visão de Implementação

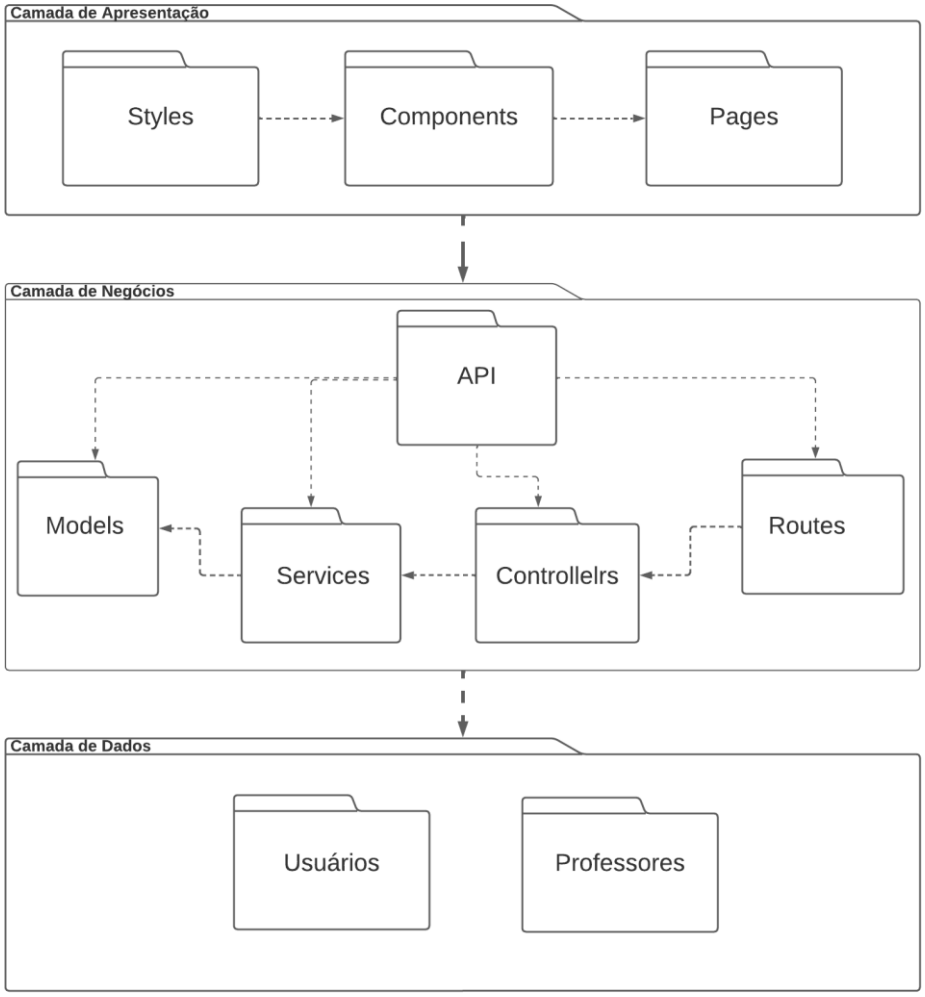

Imagem 5 – Diagrama de Pacotes

2.8 Visão de Implantação

A implantação do software Gama Track, é importante definir a infraestrutura de hardware necessária para garantir o desempenho e a escalabilidade do sistema. A infraestrutura deve ser capaz de lidar com número considerável de usuários e dados sem diminuir o desempenho.

As tecnologias escolhidas para a implantação do software devem ser capazes de suportar as necessidades do sistema. Com isso foi decidido desenvolver o frontend em React.js, uma biblioteca popular para criar interfaces de usuário. Já o backend vai ser desenvolvido em Node.js, uma plataforma popular para criar aplicativos da web escaláveis e de alto desempenho. Existem muitos recursos disponíveis on-line para aprender Node.js, incluindo tutoriais, documentação e fóruns de discussão.

O banco de dados escolhido para a implantação do software deve ser capaz de armazenar e gerenciar grandes quantidades de dados. O MongoDB é um banco de dados popular que pode ser uma boa opção para o projeto Gama Track. Ele é escalável, flexível e pode lidar com grandes volumes de dados. Além disso, o MongoDB é compatível com muitas linguagens de programação, incluindo Node.js.

2.9 Restrições adicionais

Ao se tratar em quesitos adicionais à aplicação, é importante levar em consideração os aspectos de qualidade sob requisitos não funcionais do software. Em usabilidade, a aplicação deve ser fácil de usar e intuitiva para os usuários finais. Além disso, a aplicação deve ser capaz de lidar com um grande volume de dados sem diminuir o desempenho. Isso ajudará a garantir que o sistema possa lidar com um grande número de usuários e dados sem diminuir o desempenho.

Sob quesito de suportabilidade, a aplicação deve ser executável em diferentes plataformas e sistemas operacionais. Isso ajudará a garantir que a aplicação possa ser usada por usuários com diferentes configurações de hardware e software.

Por fim, sob critérios de confiabilidade, o usuário deve ser capaz de identificar o estado de sua ação e reconhecer por meio de notificações na interface visual sobre possíveis erros ao buscar dados ou realizar tarefas dentro da aplicação. Isso ajuda a garantir que o usuário possa usar o sistema sem problemas e que possa identificar possíveis erros ou problemas com facilidade.

3 Bibliografia

GamaTrack. Declaração de Escopo do Produto. 2023. Disponível em: https://github.com/FGA0138-MDS-Ajax/2023-2-POLLUX/blob/docs/Entregas/Declara%C3%A7%C3%A3o%20de%20Escopo%20do%20Produto.pdf Acesso em: 02/11/2023.