Trabalho de Programação orientada a objetos

Dalmo Scalon Inácio - 12111BSI274

Guilherme Castilho - 12021BSI225

José Luzia da Silva Neto - 12021BSI264

Lucas Gabriel Dutra de Souza - 12121BSI226

Victor Hugo Morais da Silva - 11921BSI231

O trabalho consiste no desenvolvimento de um jogo de perguntas e respostas.

O jogo funciona de forma singlePlayer, ou seja, o jogador joga sozinho, mas pode comparar sua performance com outros jogadores através de um ranking.

Antes de iniciar o jogo o jogador deve informar seu nome de usuário, que será usado para guardar seu progresso e para posicionamento no ranking.

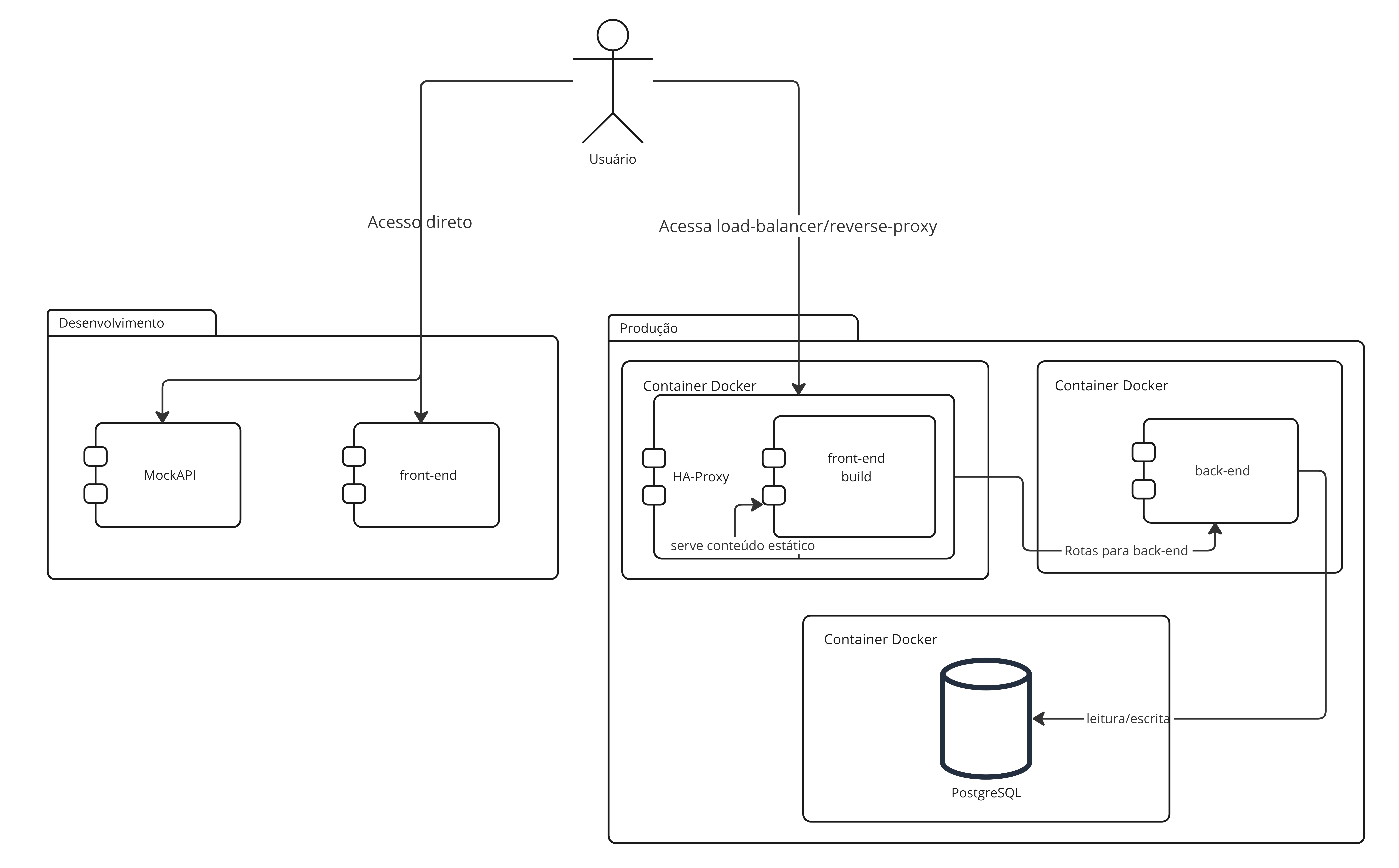
Ao iniciar o jogo o jogador escolhe um entre várias opções de questionários ou quizes disponíveis de variados assuntos, cada questionário possui várias perguntas, cada pergunta possui uma pontuação de acordo com seu nível de dificuldade e para cada pergunta existem várias respostas/opções disponíveis para escolher sendo apenas uma a opção correta. Durante o decorrer do questionário o jogador tem a opção de pular uma pergunta, mas ao fazer isso terá um desconto na sua pontuação equivalente à metade da pontuação da questão.

Os questionários, perguntas e respostas disponíveis são cadastradas previamente por um usuário administrador em uma página do sistema criada para este fim.

A aplicação será composta por:

* Banco de dados PostgreSQL com deploy via container docker
* FrontEnd desenvolvido em HTML, CSS e JS utilizando bibliotecas React Js, Gatsby Js, Tailwind CSS e React Query
* BackEnd desenvolvido em Java com biblioteca SpringBoot com deploy via container docker
* Reverse Proxy / Balanceador de Carga / Servidor de conteúdo estático HaProxy com deploy via container Docker.

Os componentes da aplicação se ligarão de acordo com o diagrama abaixo:



A modelagem da aplicação é composta pelos pacotes: Domain, controller e repository, sendo o pacote repository composto pelo subpacotes interfaces e postgres.

O pacote Domain reúne as classes que definem a regra de negócio do jogo Quis, tais como Jogador, Questionário, Pergunta, Resposta, etc. É composto pelas classes:

* UsernameVO
* UsuarioIdVO
* SenhaVO
* Usuario
* Jogador
* Administrador
* Jogo
* Quis
* Questao
* Resposta
* Ranking
* RankingLinha

O pacote Controller reúne as classes de controle REST que compõe uma API REST, responsável por receber requisições HTTP e acionar as classes de acesso de dados de acordo com as definições das classes de domínio. É composto pelas classes:

* UsuarioController
* AdministradorController
* JogadorController
* QuizController
* QuestaoController
* RespostaController
* RankingController

O pacote Repository.Interfaces reúne as interfaces de acesso de dados, essas interfaces descrevem a forma como deve ocorrer a comunicação entre as entidades de domínio e o repositório de dados. É composto pelas classes:

* UsuarioRepository
* AdministradorRepository
* JogadorRepository
* QuizRepository
* QuestaoRepository
* RespostaRepository
* Rankingepository

O pacote Repository.Postgres reúne as classes que implementam as interfaces de acesso de dados, sendo responsável por fazer a conexão com banco de dados Postgres e fazer operações de leitura e escrita. É composto pelas classes:

* PostgresUsuarioRepository
* PostgresAdministradorRepository
* PostgresJogadorRepository
* PostgresQuizRepository
* PostgresQuestaoRepository
* PostgresRespostaRepository
* PostgresRankingRepository

Os métodos e atributos de cada classe seguem o modelo dos diagramas abaixo:

