UNIVERSIDADE FEDERAL DE UBERLÂNDIA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Engenharia de Software

File Jolt

Alexandre Justo Petusk Filipe
Arthur do Prado Labaki
Klesley Silva de Oliveira
Túlio Roquete Oliveira
Wellington Cesar Dias Lima

25-05, 2022

GBC064

1 Concepção do sistema

O conceito do sistema é uma aplicação web para *download* e *upload* de arquivos em geral, como executáveis de jogos, aplicativos e documentos, ainda com a possibilidade de versionamento de até 10 versões. Nessas postagens de arquivos é possível realizar comentários, além de poder avaliar a publicação com curtidas. O site também permite a busca desses arquivos pelos usuários, ou a busca por usuários, mostrando todas as suas respectivas publicações.

Esse conceito foi inspirado no site GameJolt (Yaprak e David, 2004), que é um serviço de hospedagem de videogames gratuitos, além de ser uma plataforma de comunidade social para conteúdo criado por jogadores. Nele desenvolvedores indies de jogos podem expor seus jogos para a sua comunidade de maneira mais fácil e gratuita, recebendo comentários sobre o seu jogo e curtidas. Ainda é possível criar comunidades de jogos específicos, como uma rede social voltada para determinado conteúdo.

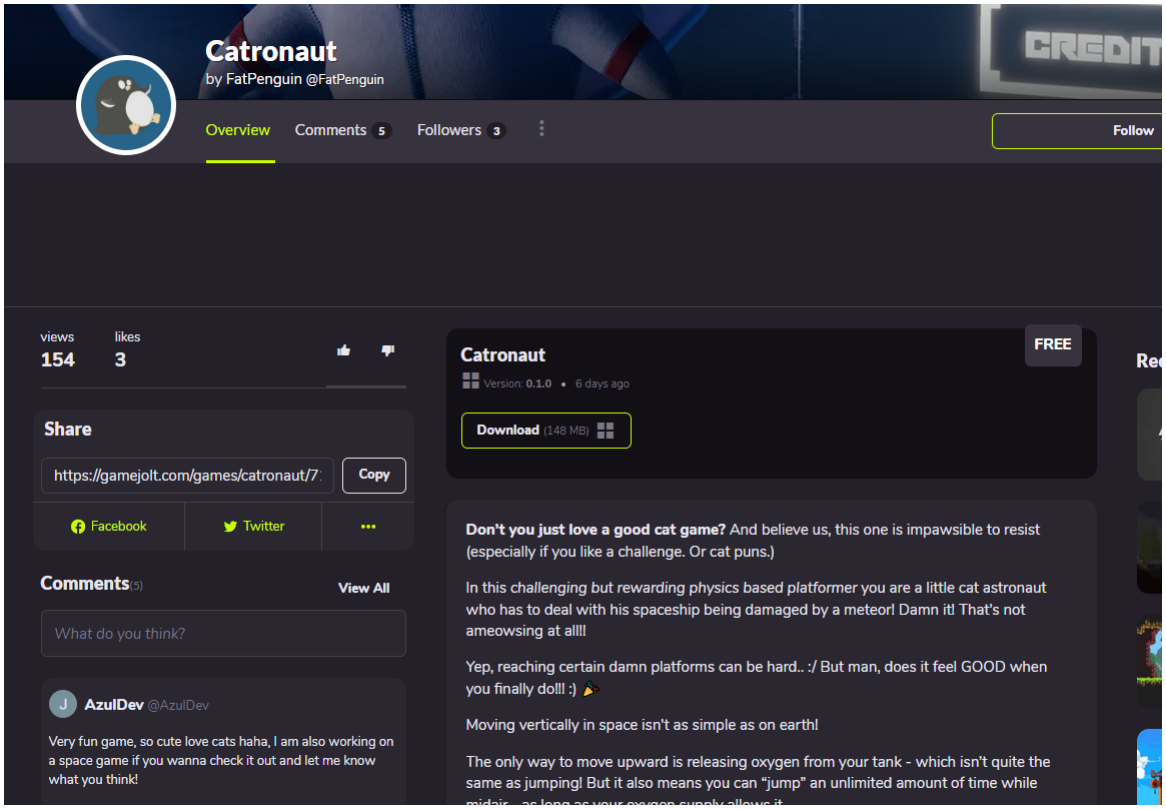

Figura 1: Exemplo de postagem de jogos

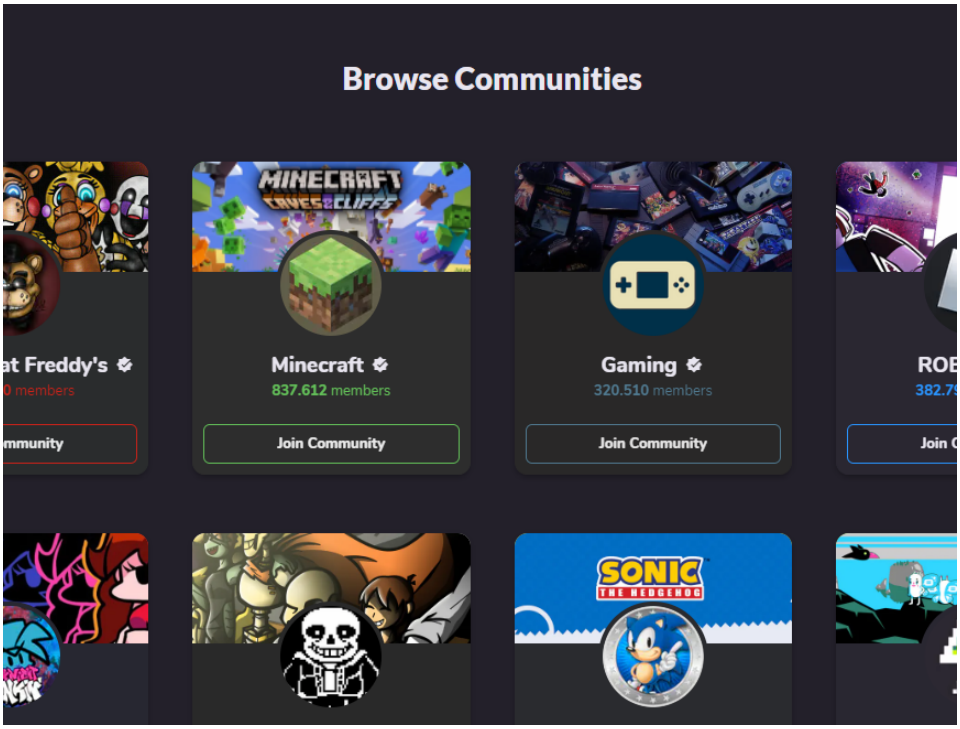

Figura 2: Exemplo de algumas comunidades disponíveis

2 Definição do ferramental

A ideia para este projeto é usar ferramentas simples mas que sejam eficientes e funcionais para uma aplicação web. Dado isso, para as operações do sistema *(Back-end)* será usado a linguagem de programação PHP com AJAX, junto com o sistema de gerenciamento de banco de dados PostgreSQL. Já na interface gráfica *(Front-end)*, pretende-se usar o HTML5 e CSS com o *framework* web Bootstrap e um pouco de JavaScript.

Já para o controle de gerenciamento de projeto, usaremos o aplicativo Trello utilizando a metodologia SCRUM. Para a criação dos diagramas do projeto será utilizado o Diagrams.net. E por fim, o repositório e versionamento do projeto será feito no GitHub, no qual o seu acesso está no link abaixo.

Link para o GitHub do projeto

3 Requisitos Preliminares do sistema

Os requisitos preliminares do projeto são:

- Realizar busca de arquivos específicos enviados ou usuários registrados;
- Permitir baixar os arquivos encontrados;
- Permitir a criação de contas para usuários com nome, e-mail e senha;
- Possibilitar o envio de arquivos, somente com conta;
- Poder atualizar esses arquivos enviados (versionamento);
- Comentar em arquivos enviados por si mesmo ou por outros usuários;
- Avaliar arquivos enviados por outros usuários;
- A aplicação deve ser web.

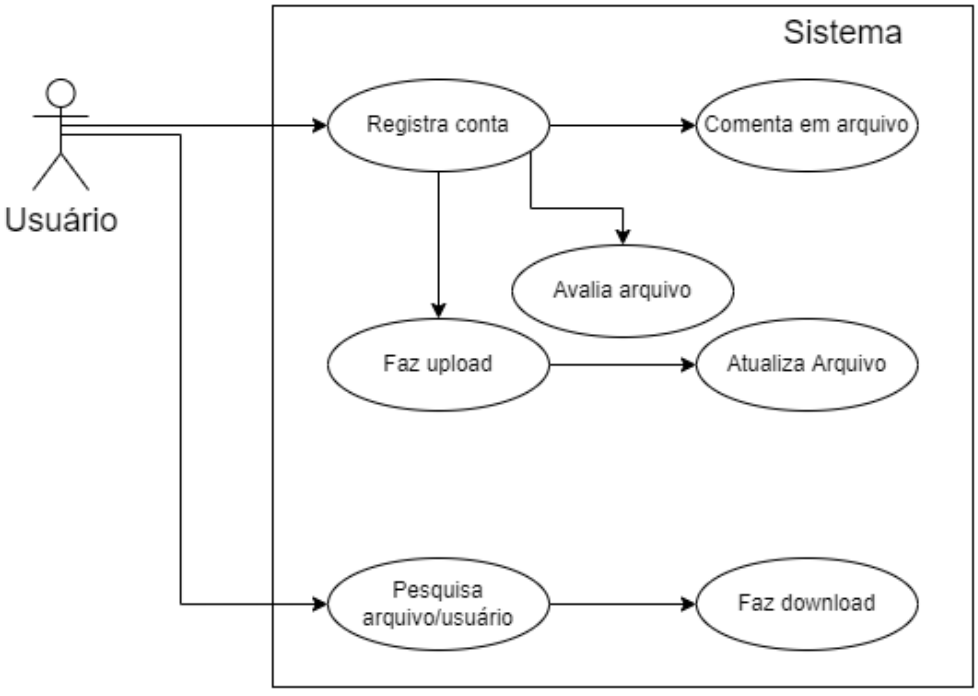

Figura 3: Casos de uso simplificado

4 Planejamento do trabalho

O planejamento do trabalho está disposto no Trello, em que cada parte da aplicação deverá ser encarregada por determinada pessoa ou conjunto de pessoas do grupo, distribuindo e facilitando o desenvolvimento do projeto. O link para visualização está abaixo.

Link para o Trello do projeto

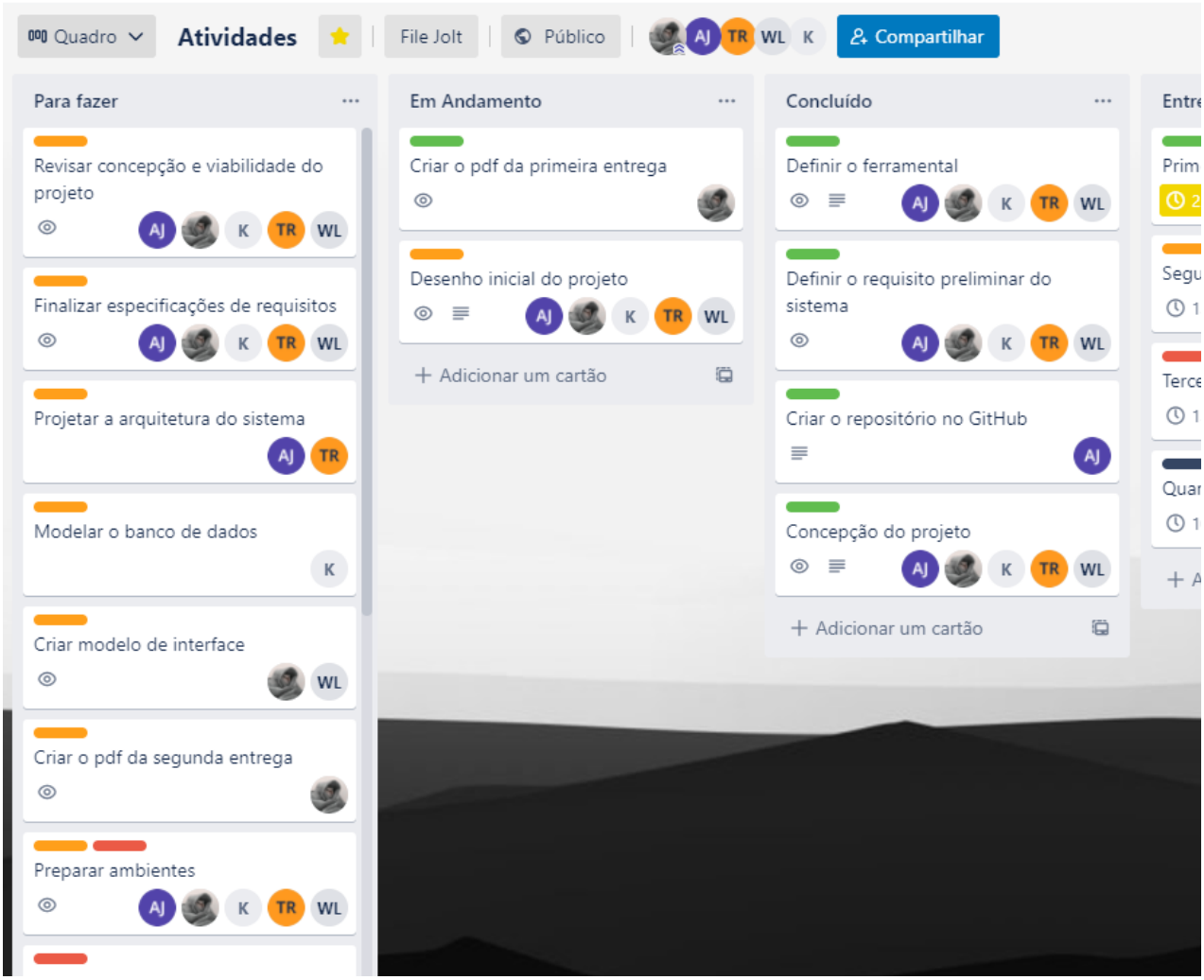

Figura 4: Uma parte do gerenciamento do projeto no Trello

Foi criado também uma primeira versão do diagrama de atividades, mostrando o fluxo de controle das possíveis atividades do projeto. Ele está inserido no repositório.

5 Revisão da Concepção

O conceito do nosso sistema a ser desenvolvido é uma inspiração no site GameJolt e um pouco do GitHub. O projeto será uma aplicação web para *download* e *upload* de arquivos em geral, como documentos, imagens, códigos e aplicações, possibilitando o versionamento do arquivo em até dez versões.
O File Jolt também permitirá que usuários consigam comentar em arquivos de outros, criando um ambiente de boa interação entre eles, além de permitir a avaliação do arquivo. Como o GameJolt, o projeto permitirá a busca por nomes de arquivos ou usuários, mostrando todas as suas respectivas publicações.
A nossa aplicação começara com um pequeno escopo, contemplando somente alunos de universidades, fazendo com que os files upados sejam majoritariamente relacionados a estudos. Com o passar do tempo, pode ser feita a ampliação desse escopo, que deverá ser discutida entre os criadores.

6 Estudo de Viabilidade

O estudo de viabilidade demonstra a qualidade do que é viável e tem fortes probabilidades de se realizar do modo planejado ou de se concretizar por reunir todas circunstâncias necessárias, visando trazer resultados teóricos antes de realmente iniciar o projeto. Um estudo de viabilidade, envolve vários aspectos teóricos, mas os três fundamentais são:

6.1 Estudo de viabilidade por demanda

O estudo de viabilidade por demanda mostra se o nosso projeto possui uma demanda que pode ser atendida. O foco do nosso projeto é atrair universitários, que possam compartilhar arquivos de estudo, mas futuramente poderemos ampliar esse escopo. Os concorrentes diretos da nossa aplicação são o Game Jolt, o GitHub e até o StackOverflow. Mesmo que eles sejam melhores em pontos separados, nenhum deles faz todas as operações que o File Jolt deve fazer. Como o GitHub que não permite comentário em repositórios, ou o Game Jolt não contempla outras comunidades.

6.2 Estudo de viabilidade econômica

O estudo de viabilidade econômica tem que ser realizada sempre com o intuito de fazer com que o objetivo seja rentável. Com isso, o projeto será realizado como um trabalho de disciplina, não necessitando de investimento inicial além do tempo. Também é considerado que a comunidade irá se tornar auto sustentável, não necessitando de grandes manutenções. Para impedir futuros problemas de espaço, existirá um tamanho máximo para cada usuário. Caso futuramente seja necessário, pode ser implementado uma seção de arquivos pagos, ou até uma mensalidade para uso do site, ou propagandas.

6.3 Estudo de viabilidade técnica

E por fim o estudo de viabilidade técnica consiste em saber se será possível realizar a aplicação. O projeto é relativamente simples, sendo uma aplicação web tradicional, além disso, utilizaremos linguagens habituais e fáceis de obter conhecimento, como o PHP e o HTML5 que são muito utilizadas e que tem uma enorme documentação disponível.

7 Especificação dos Requisitos Completa

Os requisitos do sistema define o que é solicitado ao sistema fazer e com quais limitações ele é requisitado a operar. Para demonstrar os requisitos do sistema, iremos separa-los em duas categorias:

7.1 Funcionais

Requisitos funcionais são todas as necessidades, características ou funcionalidades esperadas em um processo que podem ser atendidos pelo software. Sendo assim, os requisitos funcionais do projeto são:

- O sistema deve possibilitar o cadastro de novos usuários, sendo necessário informar nome de usuário, e-mail e uma senha. Existirá um passo para a verificação do e-mail, utilizando link de confirmação, que será enviado para o novo e-mail cadastrado.

- O sistema deve permitir a recuperação de senha para usuários já cadastrados, através do e-mail, enviando um e-mail disponibilizando a troca de senha.
- O usuário deve conseguir entrar em sua conta, utilizando usuário e senha que já foram cadastrados e validados.
- O usuário deve ser capaz de trocar o nome de usuário, e-mail ou senha caso já esteja logado na conta.
- Pessoas devem ser capaz de buscar nome de usuários ou arquivos existentes no banco de dados, mesmo sem estarem logados.
- Usuários devem conseguir baixar arquivos disponibilizados por outros usuários, através da página do respectivo arquivo. Também deve ser possível comentar na aba de comentários da página e avaliar o arquivo.
- Usuários devem ser capazes de upar arquivos, informando nome, descrição, imagens e o próprio arquivo.
- A aplicação deve permitir a modificação de características do arquivo upado, como mudar a descrição, nome ou imagens. Também dese ser possível a exclusão de arquivos já upados.
- Deve ser possível o versionamento de arquivos, mantendo até 10 versões no banco de dados.

7.2 Não Funcionais

Os requisitos não funcionais são todos aqueles relacionados à forma como o software tornará realidade os que está sendo planejado. Ou seja, enquanto os requisitos funcionais estão focados no que será feito, os não funcionais descrevem como serão feitos. Sendo assim, os requisitos não funcionais do projeto são:

- O sistema deve funcionar em aplicação web, acessível via browser e qualquer sistema operacional.
- A aplicação deve ter alta disponibilidade, funcionando em 100% do tempo.

- O sistema deve ser desenvolvido na linguagem PHP com AJAX e JavaScript, HTML5 e CSS com Bootstrap.
- A aplicação deve se comunicar com o banco de dados PostgreSQL.
- A aplicação permitirá até 1GB de espaço por usuário.
- A aplicação deve atender às normas legais, sendo proibido qualquer arquivo que as descumpram, como arquivos maliciosos, pirateados, obscenos ou ofensivos.
- Também será proibido comentário ou nome de usuário com conotação sexual ou ofensivo para outros.

8 Projeto Arquitetural Preliminar

8.1 Infraestrutura

Com o uso do PHP para criação do servidor, a arquitetura da infraestrutura se torna bastante simples. Já que o PHP consegue englobar a parte *frontend* e *backend*, ele se torna o único servidor necessário, além do banco de dados. Logo a infra do projeto é composta por estes dois componentes contidos em um contêiner *Docker* cada um. E possível ver um resumo desta arquitetura na figura cinco. Tudo isso estará presente em um servidor da nuvem pessoal, que disponibiliza a aplicação em eds.wellingtoncesar.com.br.

8.2 *Frontend*

A parte do sistema que pode ser vista pelo usuário esta sendo modelada por meio de um diagrama de paginas. Este representa as paginas web que poderão ser acessadas por meio do serviço e as formas como e possível navegar de uma para outra. Na imagem, o menu de navegação representa a barra superior presente em todas as paginas, e suas conexões são as paginas que estarão disponíveis a partir deste menu. A biblioteca e uma coleção de Files favoritos do usuário logado. A figura seis mostra o diagrama.

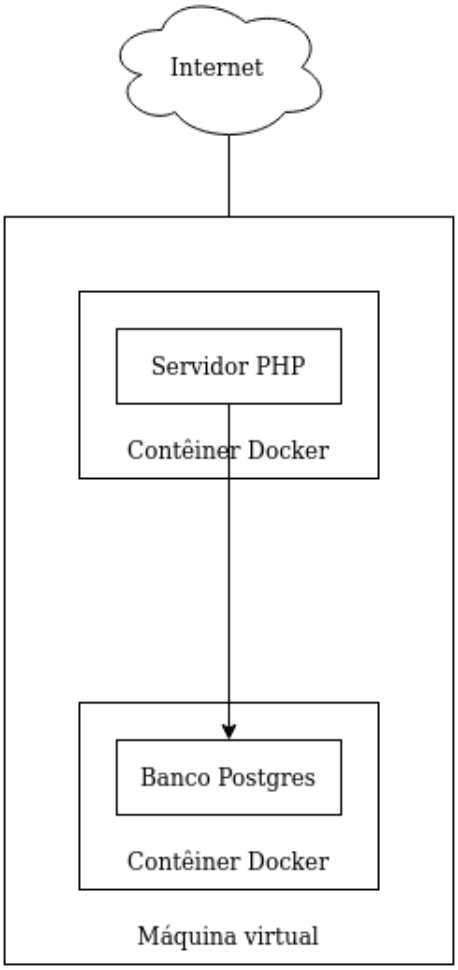

Figura 5: Resumo da arquitetura de infraestrutura

8.3 *Backend*

No mesmo servidor PHP, também haverá rotas de *backend*. Isto e, rotas que serão usadas pelo próprio programa para obter e carregar dados, ao invés de mostrar paginas HTML ao usuário. A especificação desta API foi feita no padrão *OpenAPI* (*Swagger*). O documento completo pode ser acessado **aqui** e visto com as ferramentas adequadas. A figura sete contem um resumo das rotas definidas.

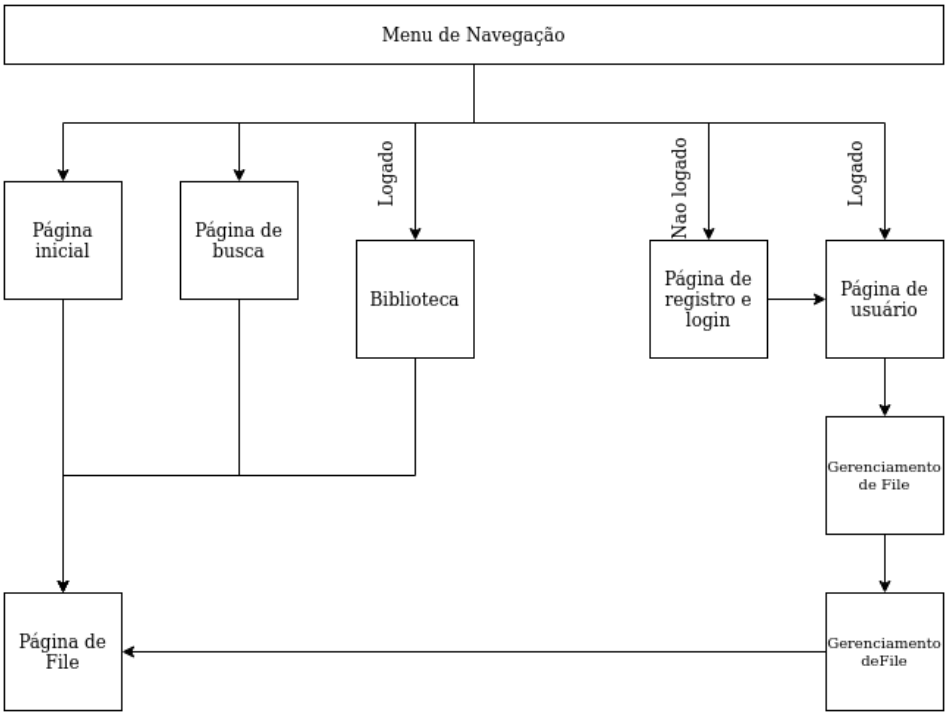

Figura 6: Resumo da arquitetura de frontend

9 Experiência de Gerenciamento

Para o gerenciamento do nosso projeto, utilizamos o *framework* para gerenciamento de projetos SCRUM, que é uma metodologia ágil que todos os membros tem algum conhecimento. Junto com o Trello, foi planejado o nosso projeto, mas com adaptações do SCRUM tradicional para nosso projeto.

Primeiro foi criado o *Product Backlog*, que se resume nos requisitos do projeto. Para a realização deles, foram separados nove *Sprints*, um a cada semana. O *Daily Scrum* foi modificado, foi criado um grupo no Whatsapp para conversas básicas do projeto em andamento. Também foi feito reuniões a cada inicio de cada *Sprint* e outra perto do termino dele, para organizar o que foi feito ou não (*Sprint Planning Meeting* e *Sprint Review Meeting*).

Em relação aos papeis do SCRUM, eles foram definidos da seguinte forma:

- *Product Owner:* Responsável por decidir o que será feito, que foi de certa maneira o professor Marcelo Maia;
- *Scrum Master:* Responsável por assegurar que a equipe respeite e siga os valores e as

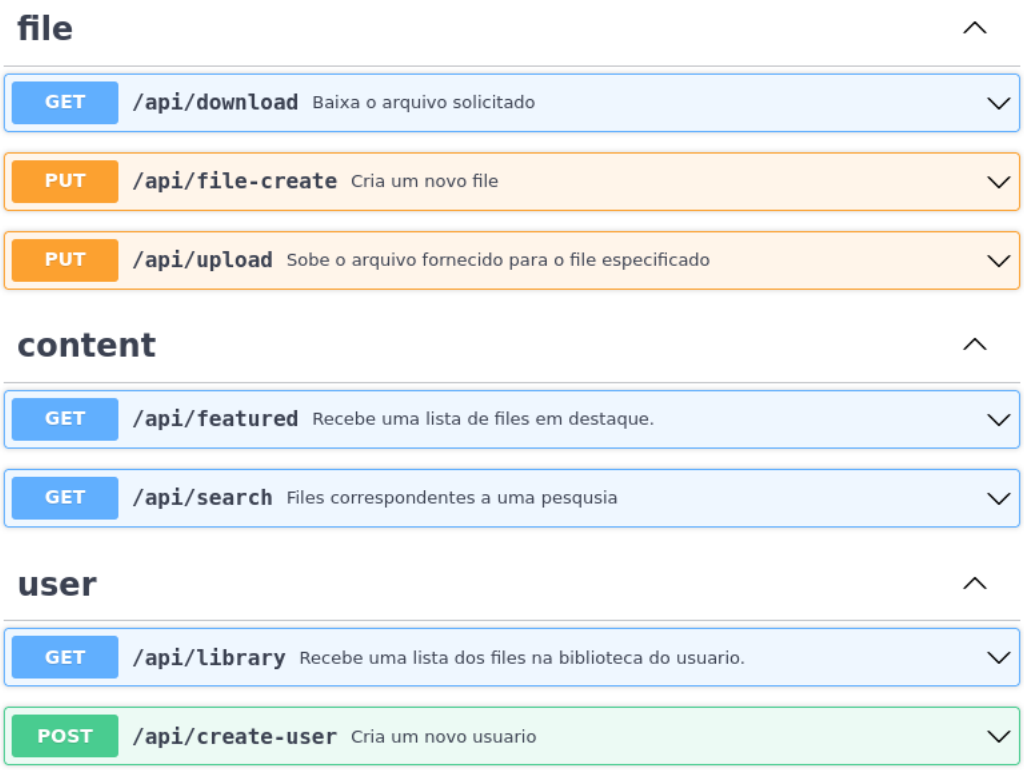

Figura 7: Resumo da arquitetura de infraestrutura

práticas do SCRUM, nesse caso foi dividido todo o tempo disponível em cinco, em que cada membro assumiu esse cargo por um período de tempo;

- *Time Scrum:* É a equipe de desenvolvimento, que é composta pelos cinco integrantes, até mesmo o *Scrum Master*.

10 Planejamento e Execução de Testes

Os testes planejados que seriam realizados são:

- Upar arquivo - subir um file qualquer no "Criar novo file";
- Baixar arquivo - baixar algum arquivo em sua respectiva página;
- Buscar por arquivos - utilizar a ferramenta de busca no *navbar* do site;
- Modificar arquivos existentes - modificar os files upados no "Alterar file";

- Criar conta - testar a criação de conta ;
- Recuperar conta - verificar se envia um email de recuperação;
- Logar - testar se existe a conexão com o banco de dados;
- Modificar a senha/email/usuário - modificar elementos no perfil do usuário logado;
- Comentar - verificar se o comentário salva no banco para cada file;
- Avaliar - somar em 1 no argumento de votos da página do file.

Porém não foi possível realiza-los, pois a aplicação não foi concluída.

11 Finalização do Projeto

Não foi possível concluir todos os requisitos do projeto por causa da falta de ~~tempo~~. Tudo o que foi desenvolvido durante a disciplina pode ser visto no Github já citado. O site desenvolvido está disponível no link.

12 Opiniões do Projeto

Podemos concluir que o projeto teve um resultado satisfatório, visto que foi possível entender diversas técnicas aprendidas em engenharia de software, além de aprender e utilizar diversas ferramentas que possibilitaram o desenvolvimento do projeto. A utilização de conceitos vistos serão uma boa base para a entrada no mercado de trabalho.

Um dos principais problemas enfrentados pelo grupo foi a falta de coordenação de tempo, no sentido de obter um período da semana que todos os participantes estariam disponíveis para realizar uma reunião sobre o andamento do projeto. Esse problema foi amenizado nas ultimas semanas, visto que foi dado um tempo em aula para isso. Um outro problema foi o uso das ferramentas escolhidas, visto que, existiam membros do grupo que não tinham conhecimento dela, demandando um tempo maior para o aprendizado dela, levando a um atraso nas iterações de desenvolvimento realizadas.

Sob esses problemas, uma solução para a coordenação do tempo seria filtrar os períodos em que todos os participantes estariam disponíveis, com isso, caso alguma reunião marcada tenha que ser adiada, existiriam outros períodos que poderiam ser substituídos, sem existir problemas de falta de alinhamento com horários. Já para o problema de conhecimento de ferramentas, uma maneira de amenizar esse tempo gasto no desenvolvimento, seria que, antes de começar a produção do projeto, todos os participantes gastariam alguns dias aprendendo todas as ferramentas que eles iriam utilizar (apenas a ferramenta que cada um iria utilizar), assim todos conseguiriam produzir ao mesmo tempo, sem ter que pausar o projeto inteiro por causa desse problema.

Mesmo com esses problemas, foi possível a realização de parte do projeto proposto, demonstrando que, mesmo com essas dificuldades, é possível realiza-lo. O gerenciamento do projeto foi muito importante, além de obter resultados positivos, em que, ele possibilitou o desenvolvimento da aplicação de uma maneira mais coordenada.