**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS**

**PUC Minas Virtual**

**Pós-graduação *Lato Sensu* em Engenharia de *Software***

Trabalho de Conclusão de Curso

SafirTec - Sistema de Gestão de Projetos

Claudia Cristina Farias Rodrigues Pinto

Belo Horizonte

Abril de 2022

# Trabalho de Conclusão de Curso

**Sumário**

[Trabalho de Conclusão de Curso 2](#_Toc99988624)

[1. Cronograma de trabalho 3](#_Toc99988625)

[2. Diagrama de casos de uso 4](#_Toc99988626)

[3. Requisitos não-funcionais 4](#_Toc99988627)

[4. Protótipo navegável do sistema 5](#_Toc99988628)

[5. Diagrama de classes de domínio 6](#_Toc99988629)

[6. Modelo de componentes 6](#_Toc99988630)

[6.1. Padrão arquitetural 6](#_Toc99988631)

[6.2. Diagrama de componentes 8](#_Toc99988632)

[6.3. Descrição dos componentes 9](#_Toc99988633)

[7. Diagrama de implantação 9](#_Toc99988634)

[8. Plano de Testes 10](#_Toc99988635)

[9. Estimativa de pontos de função 12](#_Toc99988636)

[10. Informações da implementação 13](#_Toc99988637)

[11. Referências 13](#_Toc99988638)

## Cronograma de trabalho

|  |  |  |  |
| --- | --- | --- | --- |
| **Datas** | | **Atividade / Tarefa** | **Produto / Resultado** |
| **De** | **Até** |
| 17/01/22 | 31/01/22 | 1. Análise do documento de requisitos | Elementos base para próximas etapas do projeto |
| 01/02/22 | 07/02/22 | 1. Preparação do Diagrama de casos de uso | Diagrama de casos de uso |
| 09/02/22 | 15/02/22 | 1. Preparação do Diagrama de classe de domínio | Diagrama de classes de domínio |
| 16/02/22 | 17/02/22 | 1. Análise de requisitos não funcionais | Descrição de requisitos não funcionais |
| 01/03/22 | 18/03/22 | 1. Definição do padrão arquitetural | Descrição do Padrão arquitetural |
| 15/03/22 | 20/03/22 | 1. Preparação do Diagrama de componentes | Diagrama de componentes |
| 20/03/22 | 20/03/22 | 1. Análise de componentes | Descrição dos componentes |
| 21/03/22 | 21/03/22 | 1. Preparação do Diagrama de Implantação | Diagrama de Implantação |
| 23/03/22 | 23/03/22 | 1. Criação dos casos de testes para o plano de testes do sistema | Plano de testes |
| 25/03/22 | 05/04/22 | 1. Análise dos pontos de função e preenchimento da planilha | Estimativa com pontos de função |
| \_\_ / \_\_ / \_\_ | \_\_ / \_\_ / \_\_ | 11. |  |
| \_\_ / \_\_ / \_\_ | \_\_ / \_\_ / \_\_ | 12. |  |
| \_\_ / \_\_ / \_\_ | \_\_ / \_\_ / \_\_ | 13. |  |
| \_\_ / \_\_ / \_\_ | \_\_ / \_\_ / \_\_ | 14. |  |
| \_\_ / \_\_ / \_\_ | \_\_ / \_\_ / \_\_ | 15. |  |
| \_\_ / \_\_ / \_\_ | \_\_ / \_\_ / \_\_ | 16. |  |
| \_\_ / \_\_ / \_\_ | \_\_ / \_\_ / \_\_ | 17. |  |
| \_\_ / \_\_ / \_\_ | \_\_ / \_\_ / \_\_ | 18. |  |

## Diagrama de casos de uso

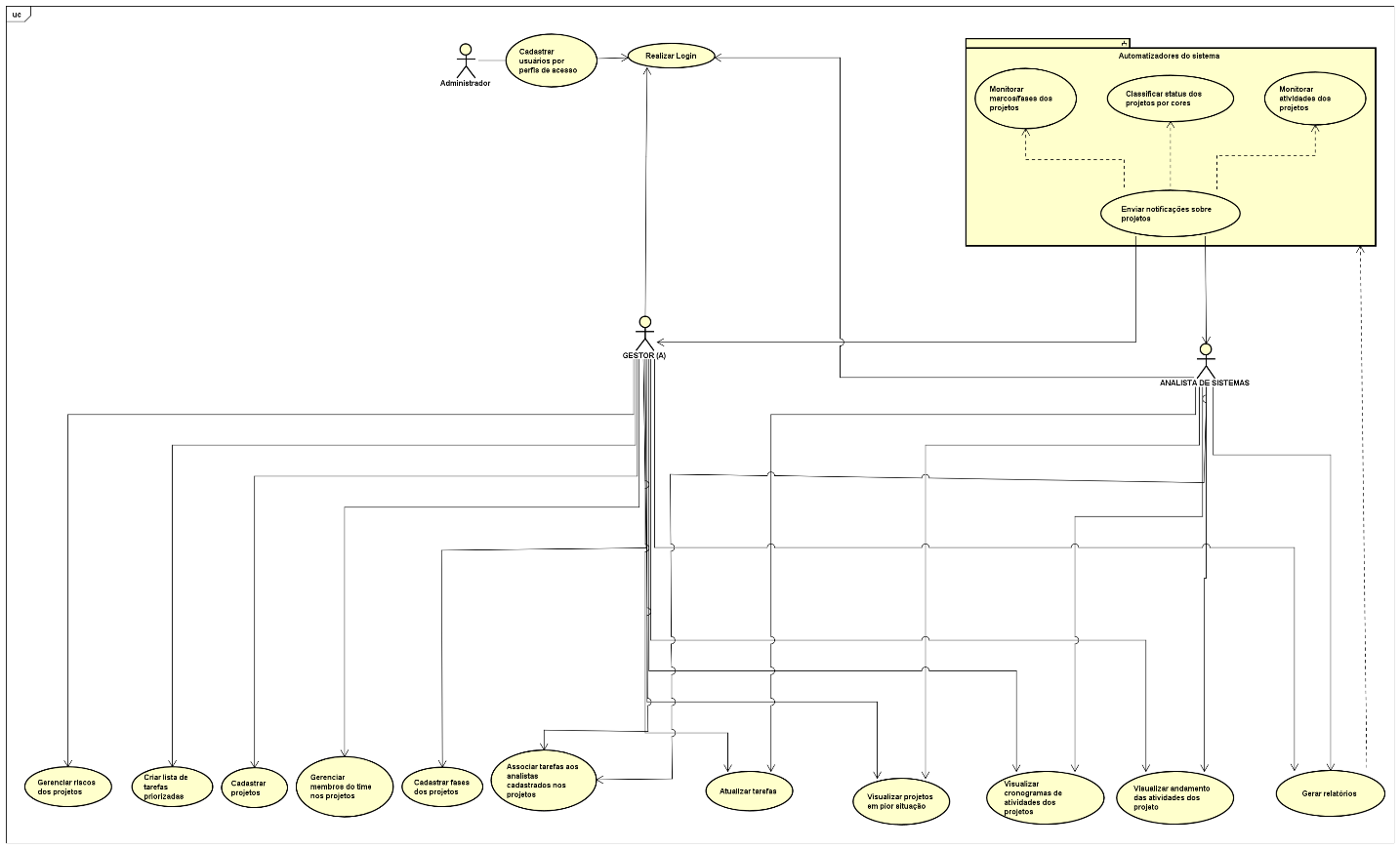


Figura 1: Diagrama de Casos de Uso

## Requisitos não-funcionais

De acordo com Roger Pressman, “[...] um requisito não-funcional pode ser descrito como um atributo de qualidade, de desempenho, de segurança ou como uma restrição geral de um sistema.” (PRESSMAN, 2021, 9. ed., p.109). Pensando em questões de confiabilidade, segurança, usabilidade e suas restrições, e, conforme requisitos descritos no documento “Sistemas de Gerenciamento de Projetos”, foram identificados os seguintes requisitos não-funcionais em alto nível:

* **Acessibilidade e Usabilidade:**

**Racional:** Em geral, para qualquer software, a usabilidade e a interface gráfica devem ser amigáveis, intuitivas, de fácil utilização e adaptação para diferentes usuários. Para Marcos Mendes “[...] A acessibilidade é o atributo de qualidade que possibilita que um sistema seja universalizado e acessado pelo máximo número de pessoas e dispositivos” (MENDES, 2020, v.1.9.3, pg. 121). E, de acordo com o documento de requisitos disponibilizado, alguns requisitos solicitam a facilidade de utilização de suas funcionalidades além de detalhes específicos para a visualização de sua interface gráfica.

* **Segurança:**

**Racional:** Sistemas com divisões hierárquicas de cadastros e de visualizações, por exemplo, necessitam possuir perfis e grupos de acesso diferenciados. E, de acordo com os requisitos, o sistema deve ter este controle de segurança baseado em perfis de acesso.

* **Multiplataforma:**

**Racional:** Na atualidade, o acesso às aplicações e sistemas é muito maior via dispositivos móveis do que via Desktop. O estudo [“2020 State of Digital Report”](https://www.similarweb.com/corp/reports/2020-digital-trends-lp/) monitorou a rede global de computadores e dispositivos móveis entre janeiro de 2017 e dezembro 2019. A análise apontou que, em 2020, as visualizações através de dispositivos móveis superaram as 116 bilhões de visualizações, contra 107 bilhões registradas em computadores.[[1]](#footnote-2) No documento de requisitos a solicitação é explicita para que o sistema possa ser acessado através da *Web* e dispositivos móveis (*smartphones e tablets*).

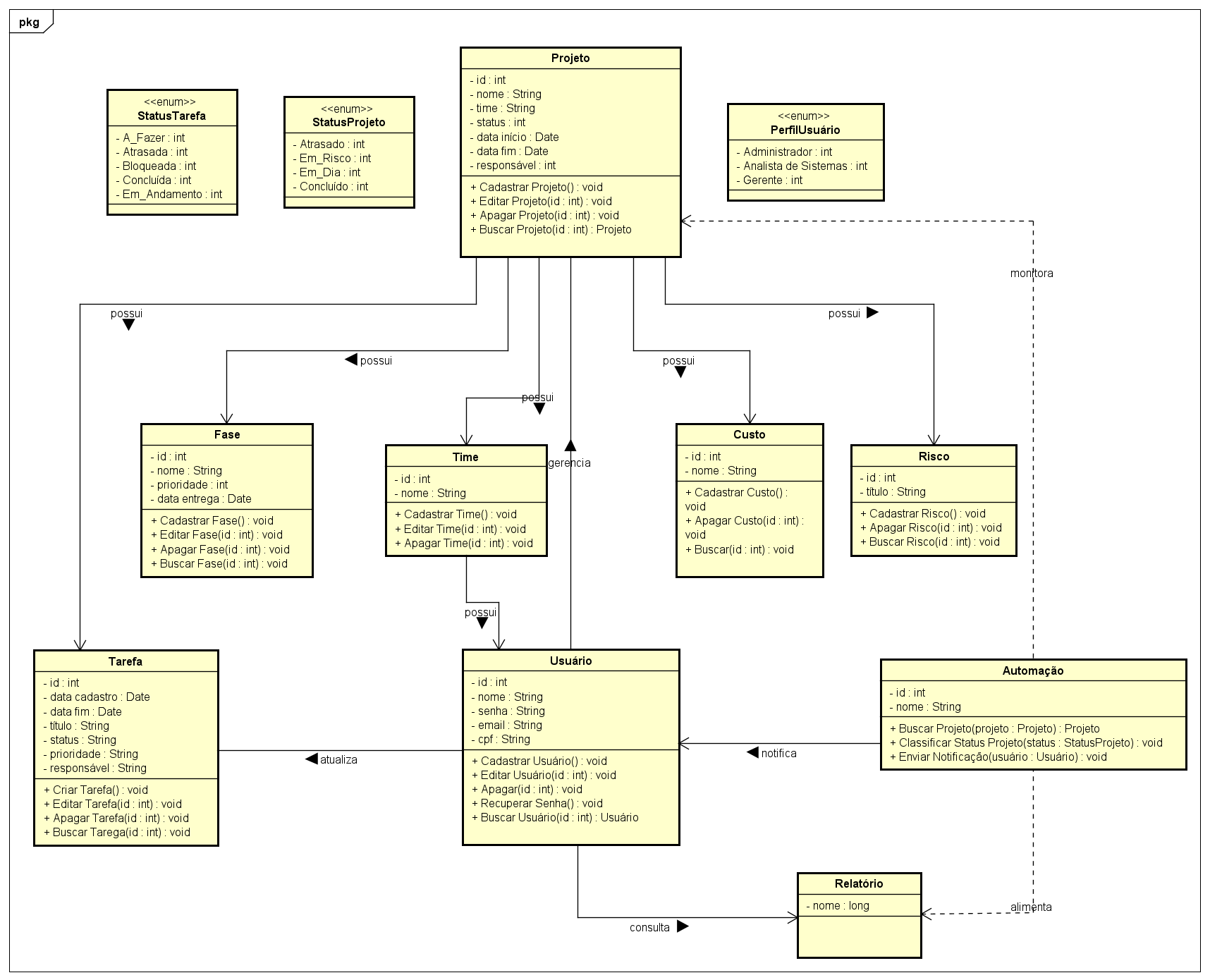
## Protótipo navegável do sistema

* Repositório GitHub com o protótipo navegável:

[***https://github.com/caufarris/pucmgengsoftware***](https://github.com/caufarris/pucmgengsoftware)

* Link para acesso ao vídeo de demonstração do protótipo navegável:

## Diagrama de classes de domínio

**

*Figura 2 – Diagrama de Classes de Domínio*

## Modelo de componentes

### Padrão arquitetural

Após identificar os requisitos funcionais e não funcionais, foram definidos alguns padrões arquiteturais e estruturais para o sistema de forma híbrida. A maioria dos frameworks atuais e populares do mercado se baseiam no padrão de arquitetura MVVM *(ModelView-View-Model)*. Esta arquitetura funciona de forma mais dinâmica, trazendo velocidade ao sistema, pois a camada *ViewModel,* além de fazer a mediação entre a visão e o modelo, também funciona como uma réplica dos dados do modelo, presentes na visão, fazendo com que as chamadas ocorram localmente, como uma espécie de cache, por exemplo (MENDES, Marcos; Vídeo aula 3.1.2, Padrão Arquitetural MVVM, Arquitetura de Aplicações Web, 2020). Desta forma, o padrão MVVM foi escolhido para a camada de visão do usuário (*Front End*).

Em linhas gerais, o fluxo de funcionamento do padrão MVVM é capaz de tornar as chamadas e respostas do sistema mais rápida que o padrão MVC.

Para o *Back End,* o sistema também utilizará o padrão arquitetural de microsserviços. Como microsserviços são pequenos serviços autônomos, cada um implementa uma pequena parte da função do negócio e pode ser implantado e/ou removido dos ambientes de forma independente. Outra vantagem é a evolução tecnológica, pois há a possibilidade de utilizar distintas tecnologias para cada novo microsserviço a ser implementada, além de acelerar o desenvolvimento do sistema. Relacionado à persistência de dados, “com a abordagem de microsserviços, cada serviço pode gerenciar o seu próprio banco de dados, diferentes instâncias da mesma tecnologia de banco de dados, ou sistemas diferentes de banco de dados” (MENDES, Marcos, Arquitetura de Sistemas Web-Princípios, Práticas e Tecnologias, 2020). As funcionalidades serão expostas através de *Web* API, já que esta pode ser acessada por quaisquer dispositivos móveis e portais *Web*.

A infraestrutura utilizada será a *IaaS* (*Infrastructure as a Service*), esta infraestrutura permite a criação e gerenciamento de aplicativos em nuvem. Este modelo de computação em nuvem permite que as infraestruturas de processamento, armazenamento, banco de dados, memória e servidores sejam disponibilizadas via internet. Uma das vantagens em possuir os serviços em nuvem, é poder acessar a aplicação de qualquer lugar do mundo, possuir uma segurança no armazenamento de informações, além de ter a vantagem do baixo custo de manutenção.

Tanto o WebApp, quanto o WebAPI serão construídos e implantados pelos componentes de integração contínua, contendo ferramenta para testes afim de garantir a qualidade de código, prevenir de prováveis bugs e até mesmo prevenir falhas de segurança.

Estas escolhas têm o intuito de facilitar a substituição de tecnologias caso desejada pelo cliente, seja por motivação tecnológica ou devido ao crescimento do sistema. Além de ter custos mais baixos, manutenção facilitada e ter uma ampla rede de suporte da comunidade de tecnologia.

No item 6.3 (Descrição de componentes) deste documento, estas escolhas serão descritas de forma mais detalhada.

As tecnologias e frameworks adotados são:

|  |  |
| --- | --- |
| **Back-End** | NodeJS, Express, TypeScript e NestJS |
| **Front-End** | Angular, HTML, CSS, Boostrap e Ionic |
| **Mobile** | Apache Cordova |
| **devOps** | Git, Jenkins, SonarQube e Docker |
| **Banco de Dados** | MySQL |
| **IaaS** | AWS e Terraform |
| **Libs** | Apache Log4J, KeyCloack (Gestão de acessos) |

*Tabela 1: Tecnologias e Frameworks*

### Diagrama de componentes

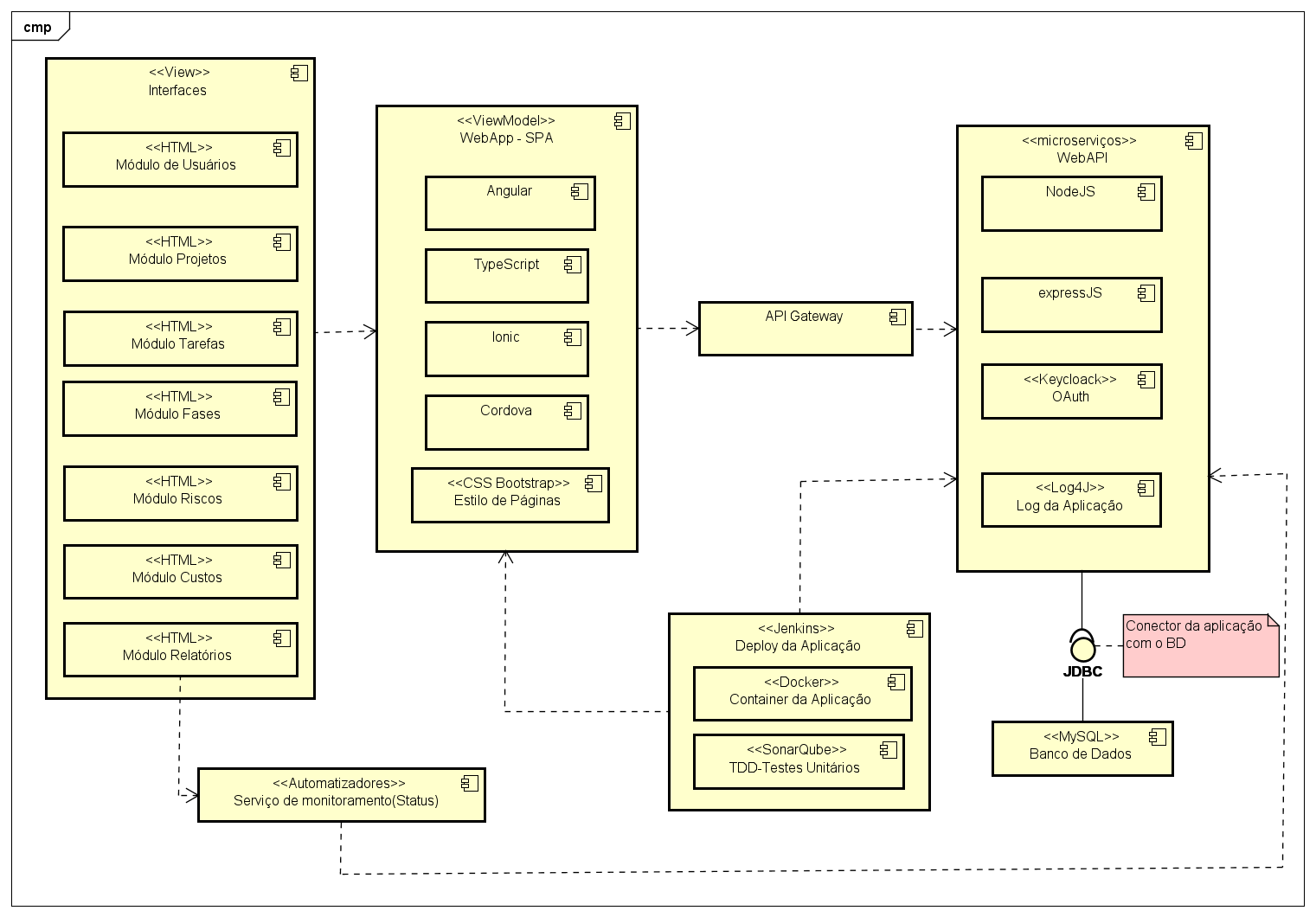


Figura 3 – Diagrama de Componentes

### Descrição dos componentes

|  |  |  |
| --- | --- | --- |
| **Número** | **Componente** | **Descrição** |
|  | WebApp - SPA | Componente contendo as linguagens e frameworks para desenvolvimento do *Front End*, que serão renderizados nos navegadores em diferentes tipos de dispositivos. |
|  | Serviço de Monitoramento | Componente do sistema de monitoramento de status do projeto para visualização e consulta por parte do cliente no módulo Relatórios. |
|  | API *Gateway* | Componente de gerenciamento de API´s que é utilizado para otimizar a comunicação entre cliente e os serviços disponíveis do *Back End.* |
|  | WebAPI | Componente contendo as linguagens, bibliotecas e frameworks de desenvolvimento do *Back End* em microsserviços que serão invocados por outros módulos do sistema, utilizando, também, serviços de autenticação (gestão de acessos) e logs do sistema. |
|  | Jenkins e Docker | Componentes utilizados para construir, gerenciar as dependências e implantar os módulos da aplicação nos respectivos containers. Além do uso do SonarQube, para inspeção contínua da qualidade do código. |
|  | MySQL | Gerenciador de dados, utilizado para persistência de dados da aplicação. |

## Diagrama de implantação

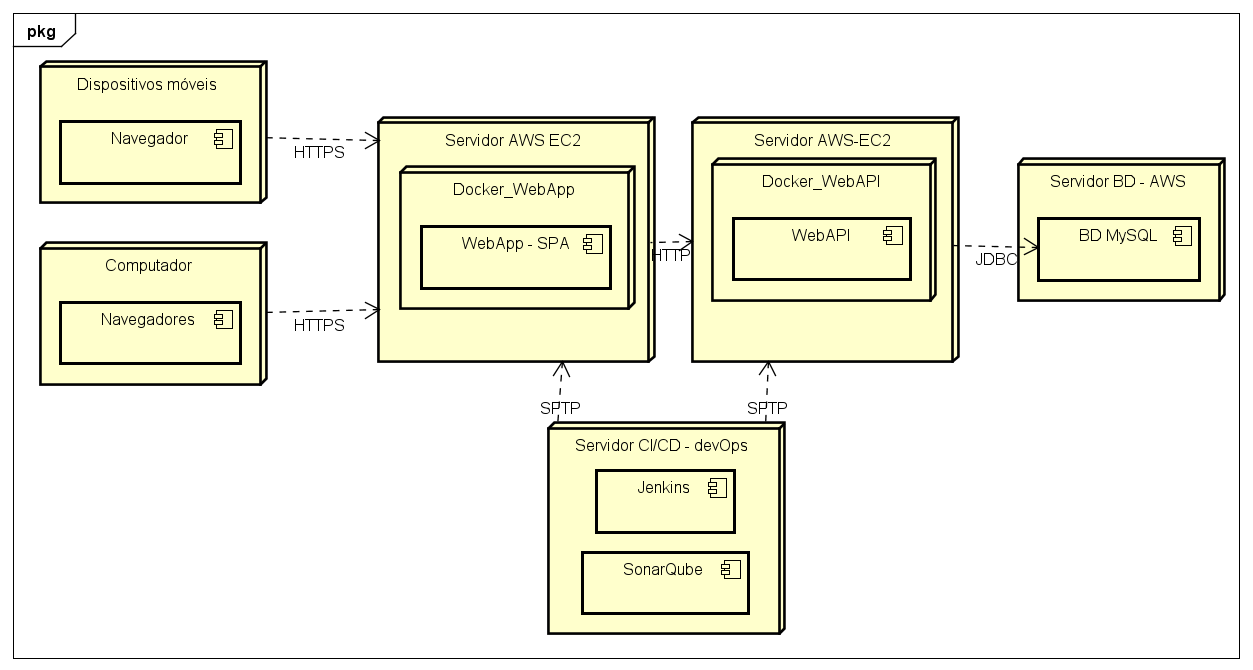
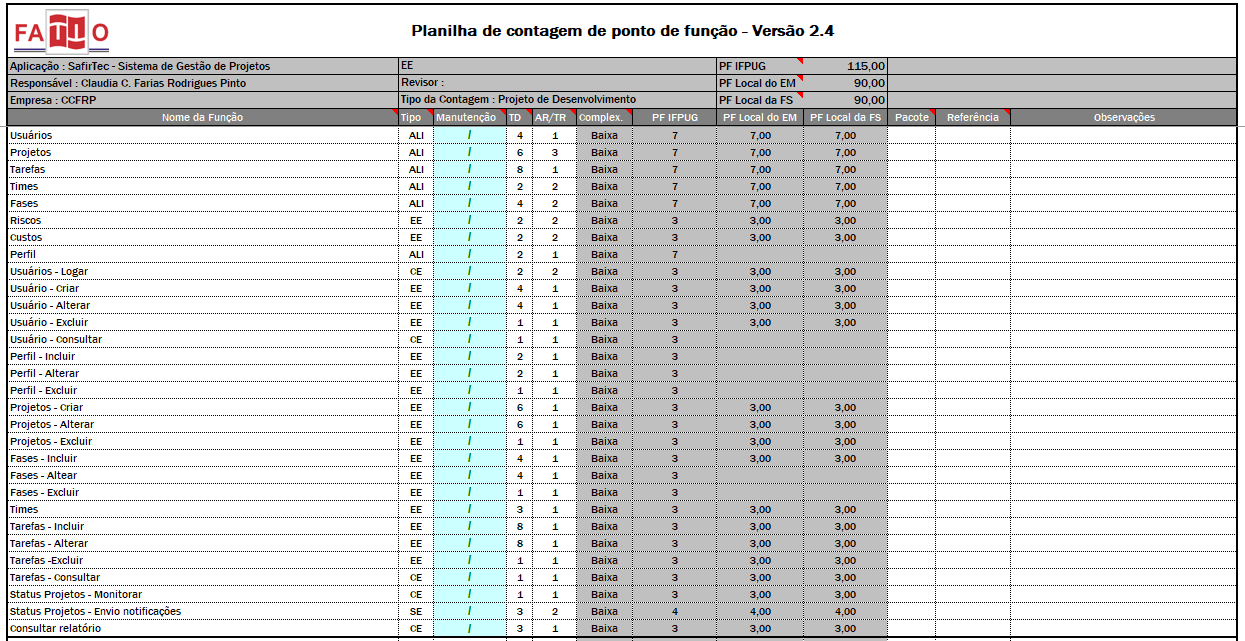


Figura 4 – Diagrama de Implantação

## Plano de Testes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID CT** | **Caso de uso** | **Objetivo do caso de teste** | **Entradas** | **Resultados esperados** |
|  | Realizar Login no sistema de acordo com o perfil de acesso concedido | Acessar o sistema com o perfil Gerente e ter acesso aos módulos do sistema concedidos ao perfil | Dado que o usuário tenha acesso ao sistema com o perfil de Gerente de Projetos  Quando o usuário realizar o login com dados válidos | Então o usuário deverá ter acesso aos módulos:  -Projetos  -Times  -Tarefas  -Fases  -Custos  -Riscos  -Relatórios |
|  | Realizar Login no sistema de acordo com o perfil de acesso concedido | Acessar o sistema com o perfil Colaborador e ter acesso aos módulos do sistema concedidos ao perfil | Dado que o usuário tenha acesso ao sistema com o perfil de Colaborador  Quando o usuário realizar o login com dados válidos | Então o usuário deverá ter acesso aos módulos:  -Projetos em que está cadastrado  -Tarefas em que está associado  -Relatórios dos projetos em que está cadastrado |
|  | Login no sistema de acordo com o perfil de acesso concedido | Negar acesso ao sistema se dados de login forem inválidos | Dado que o usuário tenha acesso ao sistema  Quando o usuário inserir dados inválidos para realizar login | Então o sistema deverá exibir mensagem de alerta informando que os dados são inválidos |
|  | Cadastrar projetos | Cadastrar projetos preenchendo os campos com dados válidos | Dado que o usuário tenha acesso ao sistema com o perfil de Gerente de Projetos  Quando o usuário acessar a página de projetos  E inserir dados válidos nos campos: Nome, Responsável, Time, Descrição, Data início e Data Fim | Então o sistema deverá cadastrar o projeto com todos os dados inseridos pelo usuário e persistir as informações no banco de dados do sistema e exibir o projeto na listagem de projetos cadastrados |
|  | Excluir projetos | Excluir projeto cadastrado | Dado que existam projetos cadastrados no sistema  Quando o usuário selecionar um projeto e a opção de excluir o projeto | Então o sistema deverá exibir uma mensagem de confirmação de exclusão do projeto  E após confirmação do usuário, excluir o projeto da listagem de projetos cadastrados e inativar o projeto no banco de dados, mantendo o histórico do projeto no sistema |
|  | Cadastrar tarefas | Cadastrar tarefas do projeto preenchendo os campos com dados válidos | Dado que o usuário tenha acesso cadastro de tarefas  Quando o usuário acessar a página de tarefas  E inserir dados válidos nos campos: Título, Descrição, Responsável, Prioridade, Estimativa, Data de Limite de entrega e Projeto Associado | Então o sistema deverá cadastrar a tarefa de acordo com todos os dados inseridos pelo usuário, persistir as informações no banco de dados do sistema e exibir a tarefa na listagem de tarefas cadastradas |
|  | Excluir tarefas | Excluir tarefas do sistema | Dado que existam tarefas cadastradas no sistema  Quando o usuário selecionar uma tarefa e opção de excluir tarefa | Então o sistema deverá exibir uma mensagem de confirmação de exclusão da tarefa  E após confirmação do usuário, excluir a tarefa da listagem de tarefas cadastradas |
|  | Relatórios | Classificar relatórios por status com cores indicativas de alerta | Dado que o usuário tenha acesso à página de relatórios  E existam projetos cadastrados no sistema  E com diferentes status de prazo  Quando o usuário acessar a página de relatórios | Então o sistema deverá exibir uma listagem com os projetos cadastrados  E exibir os status com cores específicas de acordo com o status do projeto:  VERMELHO: Atrasado  AMARELO: Com Risco  VERDE: No Prazo |

## Estimativa de pontos de função

*Figura 5 – Análise de Pontos de Função*

Link do repositório com o documento da Análise de Pontos de Função: [*https://github.com/caufarris/tcc/tree/main/documentos*](https://github.com/caufarris/tcc/tree/main/documentos)

## 10. Informações da implementação

Foi criado o código fonte para exibição do protótipo navegável do sistema. Elaborado em HTML e CSS.

[*https://github.com/caufarris/tcc*](https://github.com/caufarris/tcc)

## 11. Referências

MENDES, Marcos. **Arquitetura de Sistemas Web: Princípios, Práticas e Tecnologias**. Belo Horizonte: Copyright © por Marco Aurélio de Souza Mendes sob uma Licença Creative Commons Atribuição-CompartilhabIgual 4.0 Internacional 2020, v. 1.9.3, pg. 35-134;

MENDES, Marcos. Vídeo aulas. **Arquitetura de Sistemas Web, Fundamentos de Arquitetura de Software.** ***In*:** PONTÍFICIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS. 2020. Disponível em: <https://pucminas.instructure.com/>;

PRESSMAN, Roger S. MAXIM, Bruce R. **Engenharia de Software: uma abordagem profissional.** Porto Alegre: AMGH Editora Ltda, 2021;

SOMMERVILLE, Ian. **Engenharia de Software**. São Paulo: Pearson Education do Brasil Ltda, 2018.

1. https://www.similarweb.com/corp/reports/2020-digital-trends-lp/ [↑](#footnote-ref-2)