**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS**

**PUC Minas Virtual**

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

Trabalho de Conclusão de Curso

SisGePro – Sistema de Gerenciamento de Projetos

Matheus Santos de Oliveira

Belo Horizonte

Abril/2022.

# Trabalho de Conclusão de Curso

**Sumário**

Trabalho de Conclusão de Curso 3

1. Cronograma de trabalho 4

2. Diagrama de casos de uso 5

3. Requisitos não-funcionais 5

4. Protótipo navegável do sistema 5

5. Diagrama de classes de domínio 6

6. Modelo de componentes 7

6.1. Padrão arquitetural 7

6.2. Diagrama de componentes 7

6.3. Descrição dos componentes 7

7. Diagrama de implantação 8

8. Plano de Testes 9

9. Estimativa de pontos de função 10

10. Informações da implementação 11

11. Referências 12

## Cronograma de trabalho

|  |  |  |  |
| --- | --- | --- | --- |
| **Datas** | | **Atividade / Tarefa** | **Produto / Resultado** |
| **De** | **Até** |
| \_\_ / \_\_ / \_\_ | \_\_ / \_\_ / \_\_ | 1. Desdobramento de requisitos | 1. Lista dos requisitos funcionais 2. Lista dos requisitos não-funcionais |
| \_\_ / \_\_ / \_\_ | \_\_ / \_\_ / \_\_ | 1. Desenho do diagrama de casos de uso | 1. Diagrama de casos de uso |
| \_\_ / \_\_ / \_\_ | \_\_ / \_\_ / \_\_ | 1. Rascunho do protótipo navegável do sistema | 1. Protótipo navegável do sistema |
| \_\_ / \_\_ / \_\_ | \_\_ / \_\_ / \_\_ | 1. Desenho do diagrama de classes de domínio | 1. Diagrama de classes de domínio |
| \_\_ / \_\_ / \_\_ | \_\_ / \_\_ / \_\_ | 1. Definição do modelo de componentes | 1. Diagrama de componentes |
| \_\_ / \_\_ / \_\_ | \_\_ / \_\_ / \_\_ | 1. Desenho do diagrama de implantação | 1. Diagrama de implantação |
| \_\_ / \_\_ / \_\_ | \_\_ / \_\_ / \_\_ | 1. Elaboração do plano de testes | 1. Plano de testes |
| \_\_ / \_\_ / \_\_ | \_\_ / \_\_ / \_\_ | 1. Cálculo dos pontos de função | 1. Planilha de pontos de função |

## Diagrama de casos de uso

|  |
| --- |
|  |

## Requisitos não-funcionais

REQ-NF-SGP-001: O sistema deve ser capaz de atender requisições do usuário em até 150ms.

REQ-NF-SGP-002: O sistema deve atender acesso simultâneos de, ao menos, 50 usuários diferentes.

REQ-NF-SGP-003: O sistema deve apresentar disponibilidade de, ao menos, 99% das 05h às 21h, entre segunda-feira e sexta-feira.

REQ-NF-SGP-004: O sistema deve apresentar disponibilidade de, ao menos, 80%, entre sábado e domingo.

REQ-NF-SGP-005: O sistema deve garantir a confidencialidade das informações através de criptografia dos dados de usuários.

REQ-NF-SGP-006: O sistema deve possuir implementação compatível com navegadores presentes a partir da versão 7.0 do sistema Android.

REQ-NF-SGP-007: O sistema deve possuir implementação compatível com navegadores presentes a partir da versão 10.0 do sistema iOS.

REQ-NF-SGP-008: O sistema deve ser compatível com o navegador Google Chrome a partir de sua versão 55.0.2883.103.

REQ-NF-SGP-009: O sistema deve ser compatível com o navegador Mozilla Firefox a partir de sua versão 50.1.0.

REQ-NF-SGP-010: O sistema deve ser capaz de ser completamente utilizado por um usuário após de treinamento de três horas.

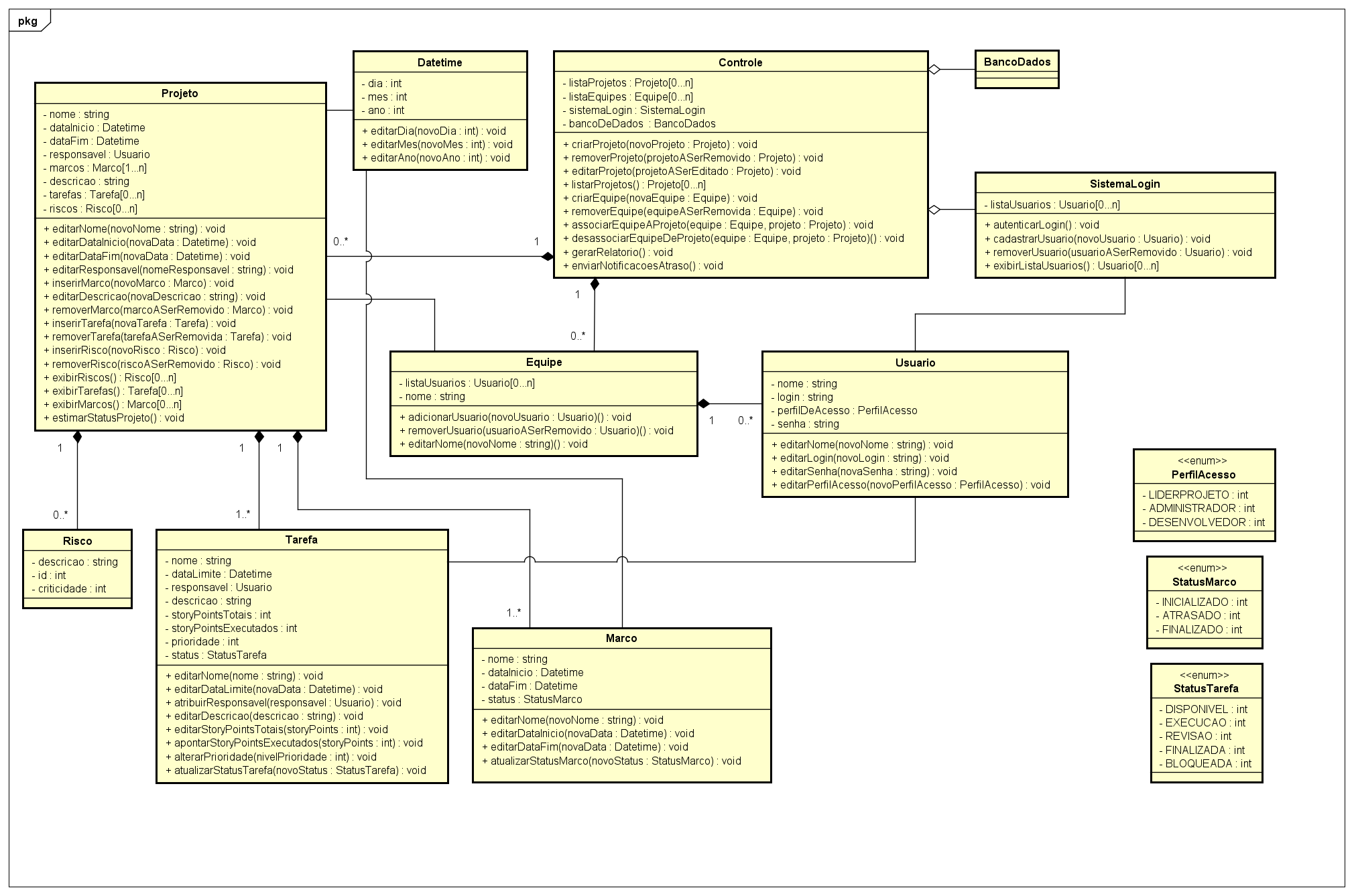
## Protótipo navegável do sistema

<Faça um protótipo navegável do sistema mostrando a tela inicial da aplicação e as de três casos de uso principais. A navegação entre as telas também precisa ser apresentada. Deve-se utilizar alguma ferramenta para a criação dos *wireframes* (como [Figma](http://figma.com) ou [Balsamiq](https://balsamiq.com/wireframes/), por exemplo).

Faça um vídeo de apresentação do protótipo navegável desenvolvido e disponibilize-o de forma que a banca de avaliação do TCC possa visualizá-lo. Esse vídeo deve ter duração de, no máximo, 3 minutos. Utilize, preferencialmente, o formato AVI.

Nesta seção, indique o *link* desse vídeo e do repositório (como o [GitHub](http://github.com), [Bitbucket](https://bitbucket.org/product/), etc) onde seu protótipo navegável está disponível.>

## Diagrama de classes de domínio



## Modelo de componentes

### Padrão arquitetural

O padrão arquitetural utilizado no desenvolvimento da aplicação será o MVVM (Model, View, View-Model) por apresentar os seguintes benefícios:

* Desacoplamento entre a camada lógica e a camada de apresentação;
* Testabilidade;
* Facilidade de manutenção.

Serão utilizadas as seguintes tecnologias:

* Angular (Front-end)
* Bootstrap (Front-end)
* Node.JS (Back-end)
* Postman (Testes)
* PostgreSQL (Banco de dados)
* NGINX (Servidor web)
* OAuth2 (Autenticação)

### Diagrama de componentes

Diagrama

Descrição gerada automaticamente

### Descrição dos componentes

|  |  |  |
| --- | --- | --- |
| **Número** | **Componente** | **Descrição** |
|  | Módulos de gerenciamento | São as páginas web desenvolvidas para visualização do sistema pelo usuário.  Correspondem ao *View*. |
|  | Framework Web | Framework responsável por acelerar o desenvolvimento dos módulos de gerenciamento e proporcionar sua comunicação com o servidor.  Corresponde ao *View-Model*.  Não necessita de aquisição, já que o Angular é uma plataforma de código aberto. |
|  | Servidor Javascript | Responsável pelas regras de negócio e pela manipulação dos dados da aplicação.  Corresponde ao *Model*.  Não necessita de aquisição, já que o NodeJS é um software de código aberto. |
|  | Servidor Web | Responsável por servir o sistema e prover respostas para requisições web.  Não necessita de aquisição, já que o NGINX é um software de código aberto. |
|  | Banco de dados | Responsável pela persistência dos dados da aplicação, através da manipulação das tabelas e registros do banco de dados.  Não necessita de aquisição, já que o PostgreSQL é um software de código aberto. |
|  | Módulo de autenticação | Responsável pela autenticação dos dados providos pelo usuário.  Não necessita de aquisição, já que o OAuth2 é um software de código aberto. |

## Diagrama de implantação

Diagrama

Descrição gerada automaticamente

## Testes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Número** | **Caso de uso** | **Objetivo do caso de teste** | **Entradas** | **Resultados esperados** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## Estimativa de pontos de função

<Indique o *link* do repositório (como o [GitHub](http://github.com), por exemplo) onde sua planilha de estimativa de pontos de função está disponível.>

## Informações da implementação

<Este é um item **não obrigatório**, mas desejável.

Informe aqui o *link* para seu repositório de código e quaisquer informações necessárias para seu acesso.>

## Referências

Para a elaboração deste trabalho foi utilizado os vídeos e notas de aula das disciplinas oferecidas no curso de Pós-graduação em Engenharia de Software da PUC-MG.