**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS**

**PUC Minas Virtual**

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

Projeto Integrado

Relatório Técnico

SGR – Sistema de Gestão de Repúblicas

Rafael Simões Coelho de Souza

Belo Horizonte

Junho de 2022.

# Projeto Integrado

**Sumário**

Projeto Integrado 3

1. Cronograma de Trabalho 7

2. Introdução 8

3. Definição Conceitual da Solução 10

3.1 Diagrama de Casos de Uso 11

3.2 Requisitos Funcionais 13

3.3 Requisitos Não-funcionais 14

4. Protótipo Navegável do Sistema 15

5. Diagrama de Classes de Domínio 15

6. Arquitetura da Solução 15

6.1 Padrão Arquitetural 15

6.2 C4 model - Diagrama de Contexto 15

7. Frameworks de Trabalho 18

8. Estrutura Base do Front End 18

9. Modelo Relacional ou Projeto de Banco de Dados NoSQL 22

10. Plano de Testes 23

11. Relatório de Execução de Testes de Software 25

12. Apropriação de Horas no Projeto 27

13. Código da Aplicação 28

14. Avaliação Retrospectiva 29

14.1 Objetivos Estimados 29

14.2 Objetivos Alcançados 30

14.3 Lições aprendidas 31

## Cronograma de Trabalho

|  |  |  |  |
| --- | --- | --- | --- |
| **Datas** | | **Atividade / Tarefa** | **Produto / Resultado** |
| **De** | **Até** |
| 26/11/2021 | 01/12/2021 | 1.Organização inicial das idéias e cenários de aplicação do projeto SGR e seus direcionamentos futuros. | Criação do texto de introdução tratando o cenário macro da aplicação em diversos aspectos. |
| 02/12/2021 | 04/12/2021 | 2. Expansão no detalhamento do texto de introdução partindo dos fatores de aspecto macro para micro, propiciando maior entendimento da aplicação. | Melhor contextualização do problema e separação dos três respectivos cenários que serão implantados futuramente no projeto. |
| 05/12/2021 | 08/12/2021 | 3.Criação do Diagrama de Casos de Uso da aplicação. | Relacionamento entre atores e ações. Enumeração de classes a serem utilizadas posteriormente na confecção do Diagrama de classes de Domínio e determinação de requisitos funcionais |
| 09/12/2021 | 10/12/2021 | 4.Especificação de requisitos funcionais da aplicação. | Enumeração de 24 requisitos funcionais categorizados para a aplicação. |
| 11/12/2021 | 12/12/2021 | 5.Especificação dos requisitos não funcionais da aplicação. | Enumeração de 8 requisitos não funcionais para a aplicação. |
| 13/12/2021 | 16/12/2021 | 6. Criação do Diagrama de Classes de Domínio. | Diagrama com o relacionamento entre as respectivas classes e métodos gerado a partir da Enumeração das classes encontradas no Diagrama de Casos de Uso. Será utilizado posteriormente para a criação das Entidades da aplicação. |
| 17/12/2021 | 22/12/2021 | 7. Criação do Protótipo interativo no Figma abordando os três cenários escolhidos para desenvolvimento dentro do Diagrama de Casos de Uso. | Wireframes, Protótipo no Figma, Vídeo do Protótipo navegável. |
| 23/12/2021 | 23/12/2021 | 8. Criação do repositório da aplicação no GitHub. | Submissão dos arquivos de documentação para a pasta docs do repositório, sendo estes o Relatório Técnico, os diagramas de Casos de Uso e Classes de Domínio, os wireframes e vídeo do Protótipo navegável gerado no Figma. |
| 14/02/2022 | 15/02/2022 | 9. Entrega da 1ª etapa do Projeto Integrado. | Submissão dos itens desenvolvidos na primeira etapa do Projeto Integrado para avaliação do orientador. |
| 22/02/2022 | 01/03/2022 | 10. Correção e ajustes após retorno do orientador na primeira etapa. | Período alocado para eventuais correções apontadas pelo orientador do projeto após a entrega da primeira etapa. |
| 02/03/2022 | 02/03/2022 | 11. Definição do padrão arquitetural da aplicação. | Especificação sobre o padrão escolhido para o projeto e suas tecnologias. |
| 03/03/2022 | 05/03/2022 | 12. Criação do Diagrama C4 Model. | Texto explicativo e Diagrama de contexto da aplicação. |
| 06/03/2022 | 07/03/2022 | 13. Frameworks de trabalho | Especificação dos frameworks utilizados no desenvolvimento da aplicação. |
| 08/03/2022 | 09/03/2022 | 14.Estrutura base do Front End. | Imagens do Layout mestre e menu de opções do sistema. |
| 10/03/2022 | 15/03/2022 | 15. Desenvolvimento de uma funcionalidade. | Funcionalidade implementada e vídeo demonstrativo da mesma. |
| 16/03/2022 | 18/03/2022 | 16. Submissão do Projeto no Heroku. | Configuração do projeto no Heroku e ajustes na ferramenta. |
| 19/03/2022 | 19/03/2022 | 17. Submissão do Relatório Técnico e demais itens desenvolvidos na 2ª etapa do Projeto na pasta docs no GitHub. | Documentação atualizada no repositório do projeto. |
| 14/04/2022 | 15/04/2022 | 18. Entrega da 2ª etapa do Projeto Integrado. | Submissão dos itens da 2ª etapa do Projeto integrado para avaliação do orientador. |
| 22/04/2022 | 25/04/2022 | 19. Correção e ajustes após retorno do orientador na segunda etapa. | Período alocado para eventuais correções apontadas pelo orientador do projeto após a entrega da segunda etapa. |
| 26/04/2022 | 27/04/2022 | 20. Modelo relacional ou Projeto de banco de dados NoSql. | Modelo de banco de dados do projeto e respectiva imagem para este documento. |
| 28/04/2022 | 01/05/2022 | 21. Construção inicial do Projeto com base na documentação levantada nos passos anteriores. | Criação do banco de dados com base no modelo das entidades.  Criação de Entidades, Controladores e Views em seu estado inicial no projeto. |
| 02/05/2022 | 22/05/2022 | 22. Desenvolvimento da aplicação e respectivos casos de uso selecionados. | Período destinado a implantação do sistema de acordo com a documentação estabelecida anteriormente. |
| 23/05/2022 | 25/05/2022 | 23. Desenvolvimento do plano de testes. | Criação de 6 casos de teste para serem aplicados na aplicação e respectivos resultados esperados. |
| 25/05/2022 | 28/05/2022 | 24. Relatório de execução dos testes. | Relatório gerado a partir da execução dos testes propostos anteriormente e seus respectivos resultados coletados. |
| 29/05/2022 | 30/05/2022 | 25. Informações relacionadas a implementação. | Abordagem sobre aquilo que foi implantado no desenvolvimento da aplicação com base na documentação, testes dentre outros. |
| 31/05/2022 | 01/06/2022 | 26. Avaliação retrospectiva. | Analise sobre as etapas e processos percorridos até a conclusão do projeto. |
| 01/06/2022 | 02/06/2022 | 27. Objetivos estimados,Objetivos alcançados e Lições aprendidas. | Delimitação sobre aquilo que foi planejado e executado no desenvolvimento da aplicação. |
| 03/06/2022 | 04/06/2022 | 28. Desenvolvimento do vídeo final de apresentação do projeto integrado. | Vídeo final com apresentação da implementação do Projeto Integrado. |
| 05/06/2022 | 06/06/2022 | 29. Atualização final do GitHub e Heroku com os arquivos e documentação relacionados ao Projeto Integrado. | Organização final de tudo aquilo que foi desenvolvido,testado, documentado e implementado. |
| 14/06/2022 | 15/06/2022 | 30. Entrega da 3ª Etapa do Projeto Integrado. | Submissão final do Projeto integrado para avaliação do orientador. |

**Observação: acrescente ou retire linhas, caso seja necessário.**

## Introdução

A Universidade Federal de Ouro Preto possui 59 repúblicas federais que abrigam 800 universitários (Dados referentes a 2022, disponíveis [neste link).](http://www.prace.ufop.br/assistencia-estudantil/moradia-estudantil/modalidade-de-gestao-compartilhada) Estas repúblicas são geridas pelos próprios universitários em um sistema denominado autogestão, sendo algumas destas moradias centenárias.

No município existem inúmeras repúblicas particulares que adotam processo semelhante de admissão de moradores, mas este trabalho focará nas repúblicas federais, tratando as demais de forma empírica em um número que gira em torno de 400 repúblicas.

A organização de uma república ouro pretana atende a um princípio de hierarquia e promoção do universitário por etapas desde seu ingresso na casa como calouro (bixo), passando pela sua escolha como morador e finalmente chegando a sua formatura como ex- aluno.

Em relação ao processo de seleção do calouro, ele poderá ser escolhido ou vetado após um período de batalha. Caso seja escolhido passará a integrar o grupo de moradores em seu mais baixo nível de hierarquia, caso contrário deverá procurar outra república para morar.

Este modelo de gestão proporcionou o desenvolvimento de laços entre gerações, visto que os universitários que se formam seguem tendo vinculo de amizade e acompanhando as decisões dos integrantes que ingressam posteriormente, fortalecendo a república ao longo do tempo.

Para manter uma república preservada é interessante que o custo de permanência estudantil seja baixo, logo a taxa de ocupação deve ser próxima de sua totalidade. De maneira complementar o número de ex alunos formados e calouros (bixos) batalhando são um indicativo de que a república é bem sucedida no ambiente universitário.

Baseado no que foi exposto, o trabalho se pauta no cadastro e edição dos dados de uma república e preservação deste ciclo de hierarquia e registro do mesmo por meio das promoções entre membros ao longo das gerações, perpetuando a organização destas entidades tão importantes e representativas em Ouro Preto.

Por objetivos específicos deste sistema, temos:

1. **Módulo para gestão de uma República.**

Tal módulo tem por finalidade permitir que a história, estrutura, localização e número de vagas limite são elementos fundamentais para a gestão da moradia. As informações aqui obtidas poderão ser repassadas posteriormente (caso necessário) ao frontend da aplicação (caso a república possua um site,cenário não tratado neste projeto). O quantitativo de vagas será utilizado na geração da taxa de ocupação da república gerada via relatório que auxiliará os moradores nesta política de divulgação de vagas de forma que a totalidade do preenchimento esteja sempre próxima do ideal. A saída deste relatório contemplará além da taxa de ocupação, a relação dos membros por curso que residem na república ao longo do tempo.

1. **Módulo para gestão de Membros**

Um membro é uma Entidade que poderá comportar os perfis de Bixo, Morador e Ex Aluno em um período. Um membro possui filiação, telefones de contato, email, status de atividade e falecimento, além do apelido que o tornará único na república. Por meio de um membro podemos navegar para as classes que são referenciadas por ele.

O Membro está em cada uma das fases associado a um perfil. Quando criado poderá ser associado ao perfil Bixo, posteriormente as vinculações ocorrerão em Morador e Ex Aluno por meio do método de promoção,respeitando a seguinte ordem; de Bixo para Morador a partir do processo de escolha e de morador para Ex Aluno a partir do processo de promoção para Ex Aluno. Por meio da conferencia de perfil de um Membro podemos determinar a real situação de todos aqueles que passaram pela república.

A associação de membros e suas respectivas classes de apoio ajudarão na composição do número de membros ativos na república no momento, visando auxiliar na geração do relatório da taxa de ocupação.

1. **Promoção de ciclo de passagem ( Calouro-Bixo para Morador).**

“Uma república sem bixos está destinada ao fracasso, vide que os bixos de hoje serão os moradores de amanhã, e os moradores atuais serão os ex alunos que aumentarão a rede de apoio à manutenção da república no futuro”.

Logo, o Bixo, associado ao Membro, deverá ter seus dados do processo seletivo informados. Datas de início e término e se o mesmo foi escolhido ou vetado na batalha de vaga.

Caso seja escolhido, ou promovido a Morador, deverá ser criado um registro novo de Morador, a atualização do perfil atual de Morador associado ao Membro, e inserção deste bixo na mais baixa hierarquia de moradores da república.

O decano é o morador mais velho de uma geração e ocupa o topo da hierarquia, sendo a ele delegado a gestão da república em seu maior nível.

## Definição Conceitual da Solução

## Diagrama de Casos de Uso

|  |
| --- |
| UseCase Diagram0.png |

## Requisitos Funcionais

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Descrição Resumida** | **Dificuldade (B/M/A)\*** | **Prioridade**  **(B/M/A)\*** |
| RF01 | O administrador deve ser inserido automaticamente na criação do sistema. | B | A |
| RF02 | O administrador deve cadastrar/editar/atualizar/excluir Cursos que serão vinculados posteriormente aos membros.  Curso com algum Membro vinculado não poderá ser excluído. | M | A |
| RF03 | O administrador deverá listar todos os cursos cadastrados. | M | M |
| RF04 | O administrador deverá listar os membros de acordo com um curso específico. | M | B |
| RF05 | O administrador deve cadastrar/editar/atualizar/excluir Cidades que serão vinculadas posteriormente aos membros.  Cidade com algum Membro vinculado não poderá ser excluída. | M | A |
| RF06 | O administrador deve cadastrar/editar/atualizar/excluir Membros que serão vinculados posteriormente aos perfis de Bixo, Morador e Ex Aluno.  Membros que estejam vinculados em algum destes perfis não poderão ser excluídos.  O apelido de um Membro é um campo único e que não pode se repetir. | M | A |
| RF07 | O administrador deverá listar os membros cadastrados no sistema, apresentando o nível ocupado pelos mesmos no momento (Bixo,Morador, Ex Aluno) e possibilidade de vinculação em um destes perfis de acordo com qual nível o membro esteja associado. | A | A |
| RF08 | O administrador poderá informar o falecimento de um Membro | B | B |
| RF09 | O administrador poderá ativar/desativar um membro específico, sendo que nestes casos:   1. Caso o Membro possua um perfil de Bixo deverá ser inativado e o processo seletivo será marcado como falso se estiver em andamento. 2. Caso o Membro possua um perfil de Morador, a situação de moradia deverá ser marcada como falsa. | A | A |
| RF10 | O administrador deverá cadastrar/editar/atualizar/excluir Bixos que serão associados a um membro específico. | A | A |
| RF11 | O administrador poderá efetuar o veto de um Bixo em processo de batalha, sendo que neste caso, o processo de batalha será finalizado e o membro correspondente desativado.  Uma chamada ao método de geração do relatório da taxa de ocupação da república será feita de forma que o cenário esteja atualizado. | M | A |
| RF12 | O administrador poderá promover o bixo em processo de batalha a morador, sendo que neste caso o processo de batalha será concluído, o bixo será escolhido, sendo gerado um novo perfil de morador associado para o membro em questão, inserido no menor nível de hierarquia.  Uma chamada ao método de geração do relatório da taxa de ocupação da república será feita de forma que o cenário esteja atualizado. | M | A |
| RF013 | O administrador deverá listar todos os moradores cadastrados, residentes e vinculados a membros ativos. | M | M |
| RF14 | O administrador deverá gerenciar a hierarquia de um Morador específico, podendo promovê-lo ou rebaixá-lo. | B | M |
| RF15 | O administrador poderá promover um Morador a Decano. Poderá existir apenas um decano no grupo de moradores ativos e residentes. | M | M |
| RF16 | O administrador poderá promover um perfil de Morador a Ex Aluno. Neste caso será criado um novo registro de Ex Aluno associado ao Membro em questão, com a data de formatura configurada inicialmente como a data atual da promoção, sendo que a mesma poderá ser editada posteriormente.  O perfil de Morador residente na república deverá ser desativado.  Uma chamada ao método de geração do relatório da taxa de ocupação da república será feita de forma que o cenário esteja atualizado. | M | A |
| RF17 | O administrador deverá listar todos os ex alunos cadastrados e ativos ordenados pela data de formatura de forma ascendente. | B | A |
| RF18 | O administrador deverá editar/ atualizar a data de formatura de um Ex Aluno específico. | B | A |
| RF19 | O administrador poderá cadastrar/editar/atualizar/ excluir uma República.  A República deverá possuir ao menos uma vaga cadastrada.  A República não poderá ser excluída caso existam membros cadastrados. | B | A |
| RF20 | O administrador poderá anunciar vagas pela República. | B | M |
| RF21 | O administrador poderá cadastrar/editar/atualizar/ excluir Fotos associadas à República. | M | B |
| RF22 | O administrador poderá cadastrar/editar/atualizar/ excluir Vídeos associados à República. | M | B |
| RF23 | O administrador deverá listar todos os relatórios com a taxa de ocupação ordenada pela data de criação de forma decrescente. | M | A |
| RF24 | O administrador deverá gerar o relatório contendo a lista de Ex Alunos homenageados na Festa do Doze de Outubro no ano corrente. Os Ex Alunos selecionados deverão ter data de formatura proporcionais a 5 anos , ordenados de forma decrescente , os que possuírem maior tempo de formado aparecerão primeiro nesta listagem. | B | A |

\*B=Baixa, M=Média, A=Alta.

**Observação: acrescente quantas linhas forem necessárias.**

## Requisitos Não-funcionais

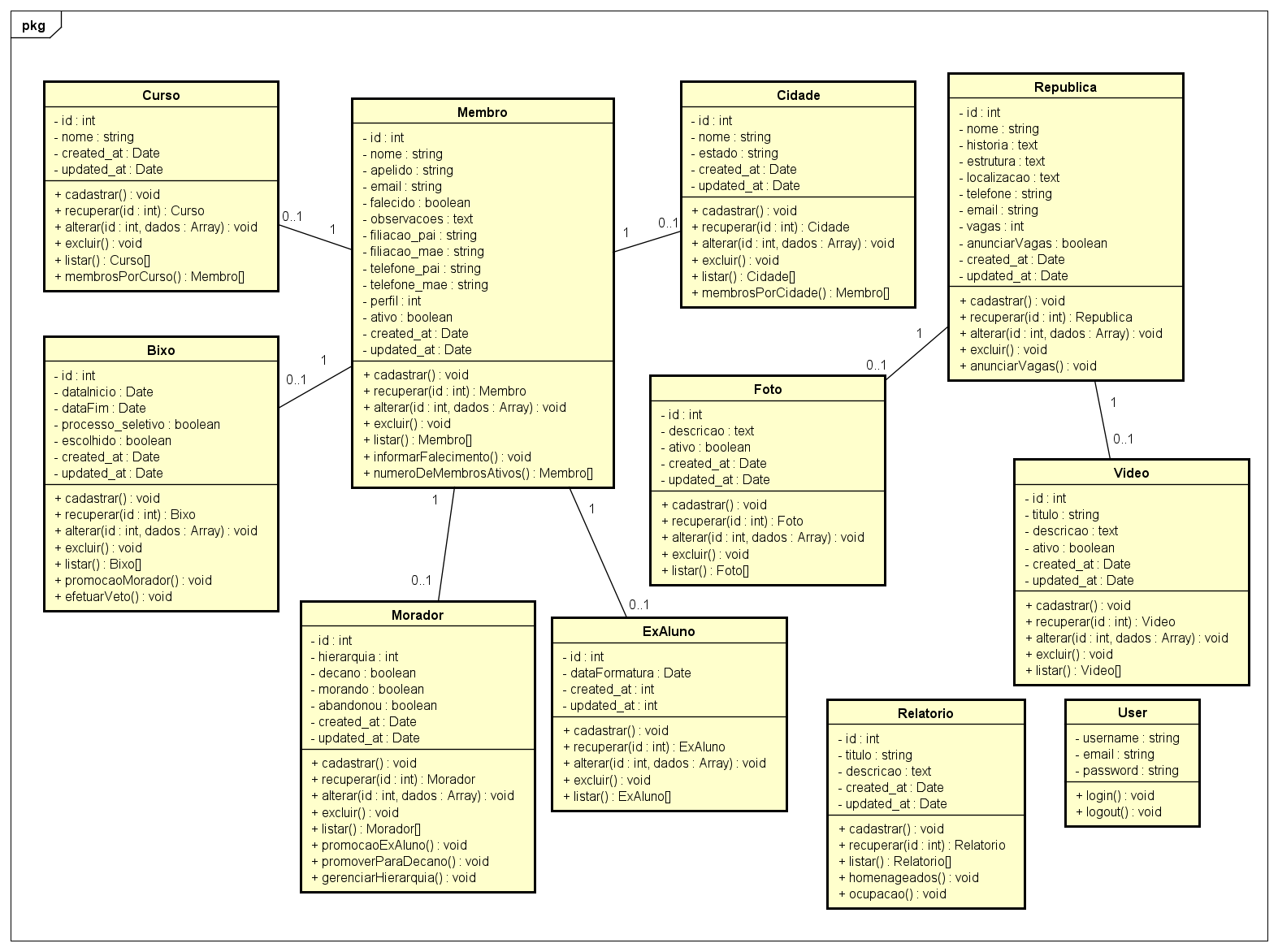
|  |  |  |
| --- | --- | --- |
| **ID** | **Descrição** | **Prioridade**  **B/M/A** |
| RNF01 | O sistema deve ter alta disponibilidade, estando ativo 24 horas por dia x 7 dias por semana. | A |
| RNF02 | A Linguagem de programação deve ser o PHP. | A |
| RNF03 | O Framework utilizado deve ser o Laravel em versão superior ou igual a 5.8. | A |
| RNF04 | O banco de dados utilizado na aplicação deve ser o MySQL. | A |
| RNF05 | O layout do sistema deve ser responsivo. | A |
| RNF06 | A biblioteca Bootstrap deve ser utilizada para tratar a responsividade. | A |
| RNF07 | O sistema deverá executar nos navegadores Edge, Google Chrome e Mozilla Firefox | M |
| RNF08 | A paginação de todos os módulos do sistema deverá ser feita a cada 5 registros. | B |

**Observação: acrescente quantas linhas forem necessárias.**

## Protótipo Navegável do Sistema

|  |  |
| --- | --- |
| Projeto SGR (GitHub) | [Link do Projeto SGR](https://github.com/rscsouza/sgr) |
| Protótipo interativo (Vídeo) | [Link para vídeo do protótipo interativo em .mp4](https://github.com/rscsouza/sgr/raw/master/docs/prototipo_sgr_v1.mp4) |
| Protótipo interativo (Figma) | [Link para apresentação do protótipo interativo no Figma](https://www.figma.com/proto/mS8Ayxau6kdmbjjz1Yi8NP/Untitled?node-id=43%3A394&scaling=contain&page-id=0%3A1&starting-point-node-id=43%3A394) |
| Protótipo interativo (Wireframes) | [Link para wireframes do protótipo interativo](https://github.com/rscsouza/sgr/raw/master/docs/sgr_wireframes.pdf) |

## Diagrama de Classes de Domínio



## Arquitetura da Solução

## Padrão Arquitetural

O padrão escolhido para o projeto foi o MVC.

O Framework utilizado para a implementação do projeto é o Laravel, sendo a linguagem de programação escolhida para o Backend o PHP.

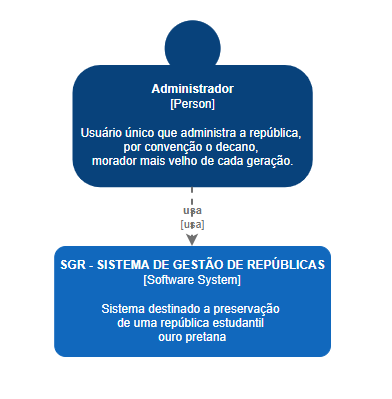
O projeto utiliza a estrutura do Laravel para o frontend denominada “Template Blade” para exibição das Views provenientes dos Controllers.

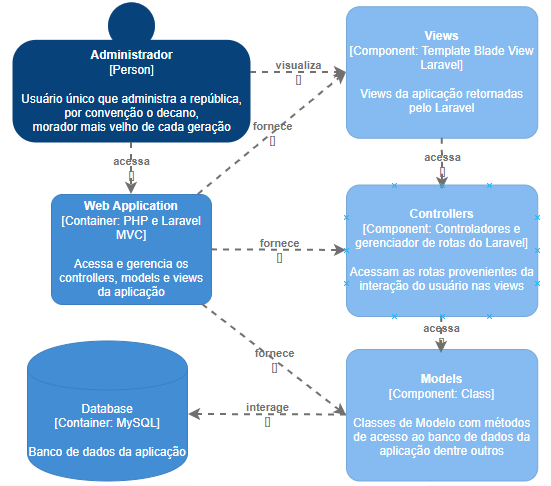
Ainda na parte de Frontend do projeto estão sendo utilizadas a biblioteca JQuery, Javascript nativo para interações do lado do cliente e a biblioteca Bootstrap Twitter em sua versão 5 para tratar a responsividade entre diferentes dispositivos.

## C4 model - Diagrama de Contexto

O primeiro diagrama apresenta a interação entre o usuário único do sistema (administrador), que por convenção é o decano , morador mais velho de cada geração da república.

O segundo diagrama mostra o administrador agindo no ambiente do framework , que retorna um conjunto de views ,controllers e models. Ao interagir com uma destas views, um controller e acionado junto ao gerenciador de rotas e este resgata as informações provenientes deste caminho, que se por ventura envolver acesso ao banco de dados, passa pelas classes de Model, retornando assim o resultado da operação após conclusão do processo para uma view específica no sistema, mantendo a consistência do modelo MVC.





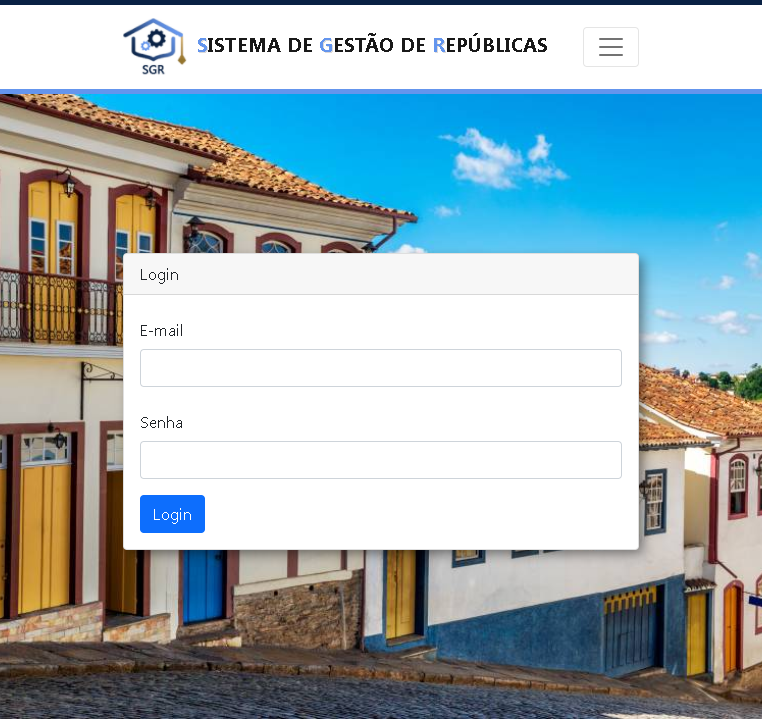
## Frameworks de Trabalho

O framework utilizado no frontend , backend e persistência é o Laravel que é voltado para a linguagem PHP. Na parte de interações com o usuário do lado do cliente temos a utilização da biblioteca JQuery e Javascript nativo.

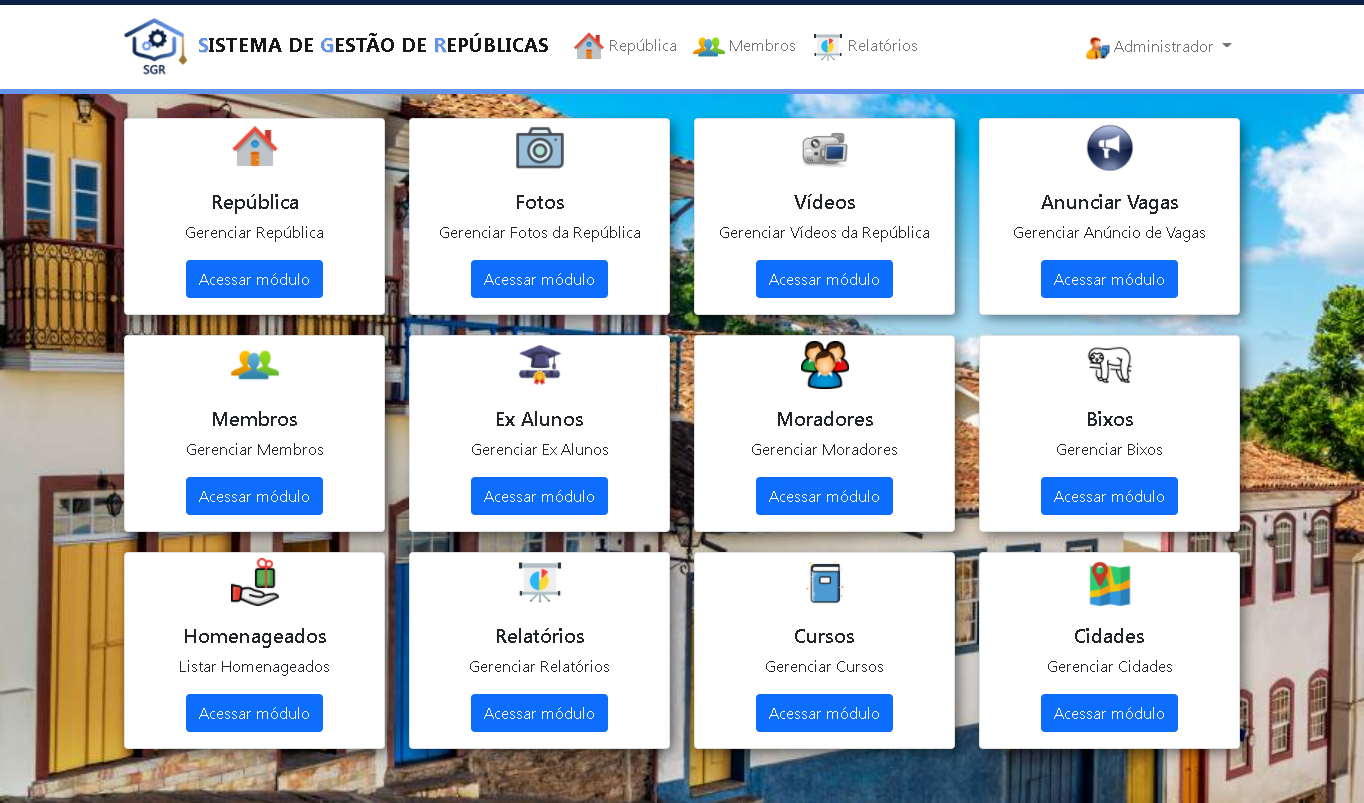
## Estrutura Base do Front End



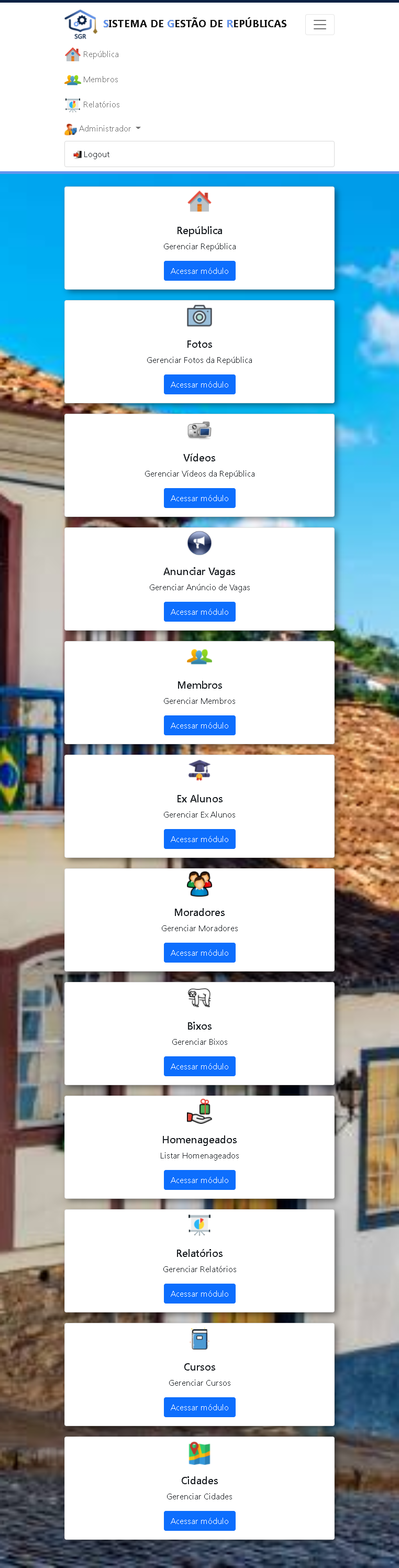
Tela de login (desktop)



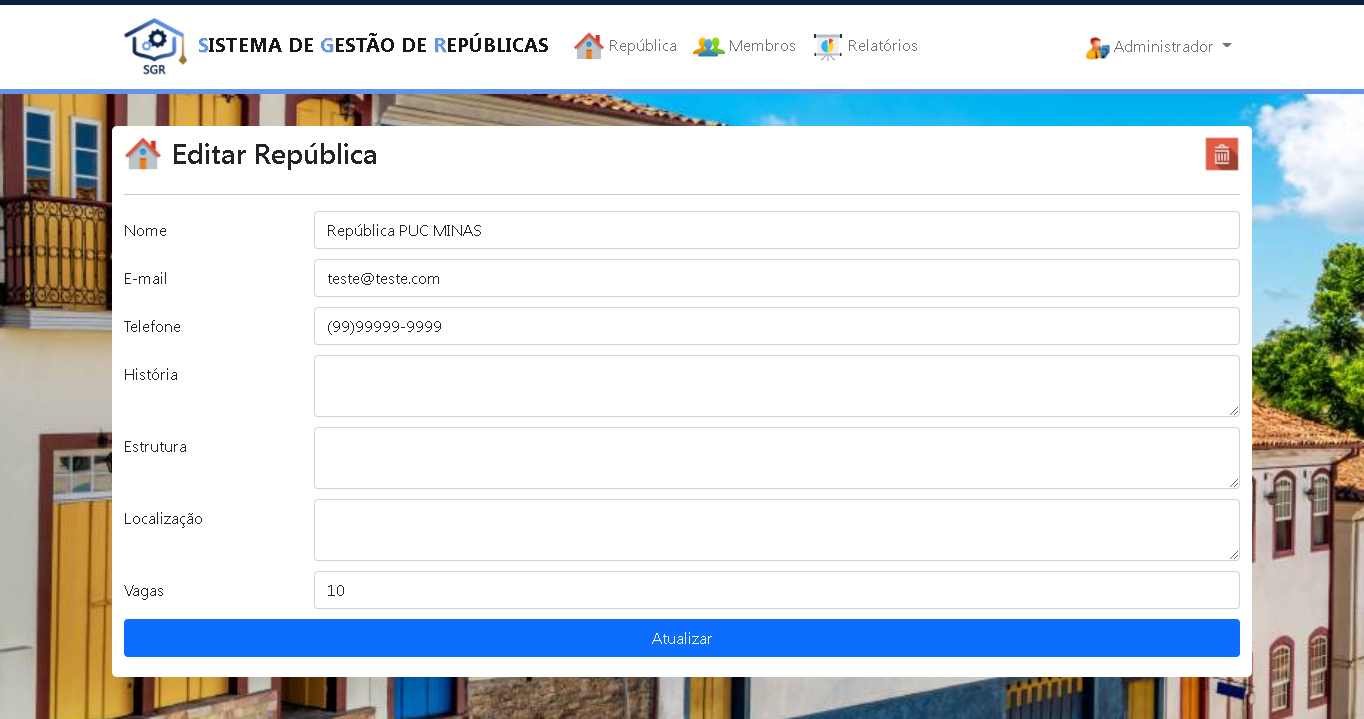
Tela de login (mobile)



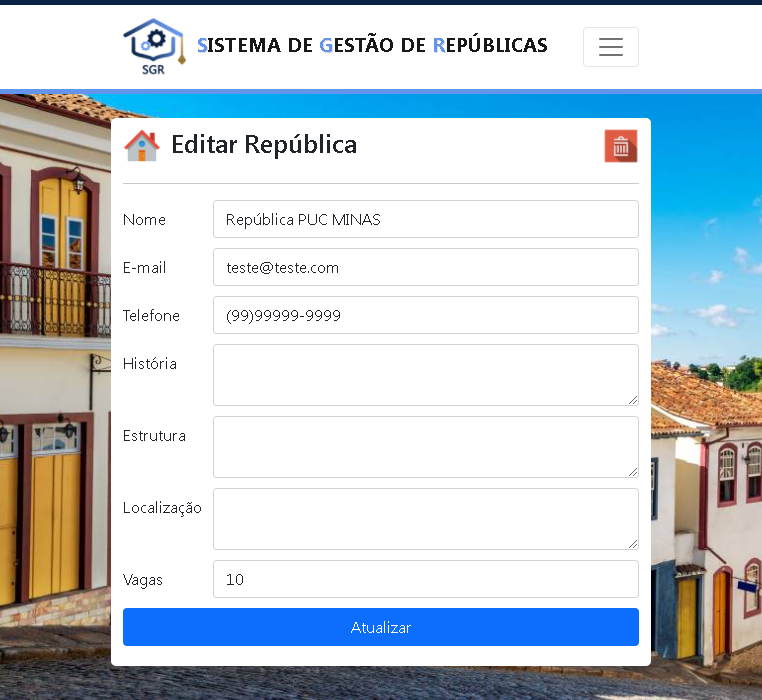
Tela de layout base (desktop) e menu do sistema ao topo



Tela de layout base (mobile) e menu do sistema ao topo

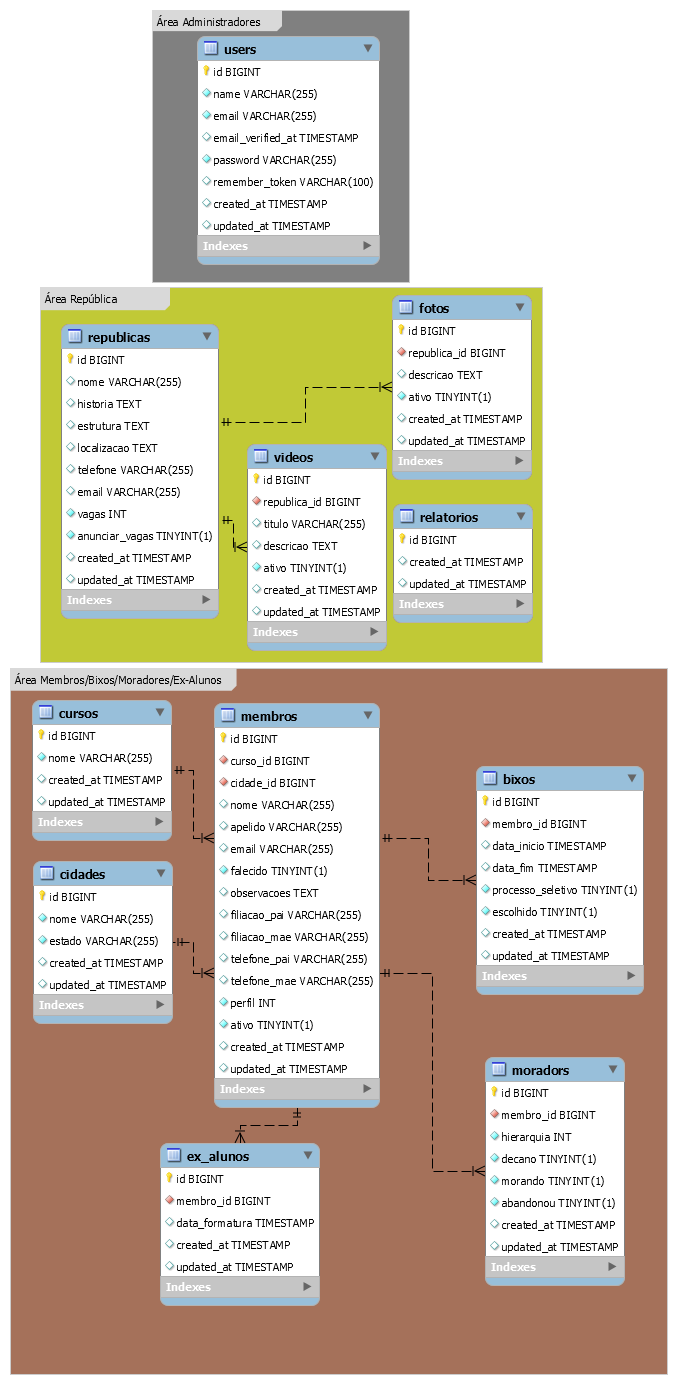


Tela da funcionalidade desenvolvida para segunda etapa (desktop)



Tela da funcionalidade desenvolvida para segunda etapa (mobile)

## Modelo Relacional ou Projeto de Banco de Dados NoSQL



## Plano de Testes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Número** | **Caso de uso** | **Objetivo do caso de teste** | **Entradas** | **Resultados esperados** |
|  | Administrar República – Edição da república | Verificar a edição dos dados informados no formulário e se os mesmos são enviados e salvos de forma consistente.  A república deve ter ao menos uma vaga cadastrada.  No menu principal, ir em República > Gerenciar República.  De forma alternativa , a área em questão pode ser acessada clicando no botão República na página inicial. | **Nome:** República PUC MINAS teste.  **Email:** [teste@teste.com](mailto:teste@teste.com)  **Telefone:**123456  **História:** teste de história  **Estrutura:** teste de estrutura  **Localização:** teste de localização  **Vagas:** 5 | Redirecionamento para a página inicial do sistema com a mensagem de sucesso exibida:  **República editada com sucesso!** |
|  | Administrar República – Exclusão da república com Membros | Verificar se a república poderá ser excluída caso apresente membros cadastrados no sistema.  No menu principal, ir em República > Gerenciar República> clicar no botão excluir no canto direito do formulário.  De forma alternativa , a área em questão pode ser acessada clicando no botão República na página inicial. | A entrada será a própria interação.  No caso, o botão de exclusão deverá ser clicado. | Redirecionamento para a página inicial do sistema com a mensagem de erro exibida:  **República não pode ser excluída pois existem membros cadastrados!** |
|  | Administrar Membros – Cadastrar Membro com apelido já registrado | Verificar a consistência cadastral dos dados cadastrados. Caso exista um membro com o mesmo apelido cadastrado previamente, o cadastro deverá ser negado.  No menu principal ir em Membros> Gerenciar Membros> clicar no botão adicionar no canto direito do formulário de listagem.  De forma alternativa, a área em questão pode ser acessada clicando no botão Membros na página inicial. | **Nome:** Pedro  **Apelido:** Quinzinho  **Email:** [pedro@p.com](mailto:pedro@pedro.com)  **Curso:** Arquitetura  **Cidade:** Abaeté/MG  **Pai:** Geraldo  **Telefone:** 123456  **Mãe:** Joana  **Telefone:** 123456  **Observações:** teste de apelido único | Redirecionamento para a tela do formulário em questão com todos os campos preenchidos após tentativa de submissão com a mensagem de erro exibida:  **Já existe um membro com este apelido cadastrado!** |
|  | Administrar Membros – Cadastrar Membro sem apelido registrado | Verificar a consistência cadastral dos dados cadastrados. Será informado um apelido não utilizado por nenhum membro  No menu principal ir em Membros> Gerenciar Membros> clicar no botão adicionar no canto direito do formulário de listagem.  De forma alternativa, a área em questão pode ser acessada clicando no botão Membros na página inicial. | **Nome:** Pedro  **Apelido:** Peralta  **Email:** [pedro@p.com](mailto:pedro@pedro.com)  **Curso:** Arquitetura  **Cidade:** Abaeté/MG  **Pai:** Geraldo  **Telefone:** 123456  **Mãe:** Joana  **Telefone:** 123456  **Observações:** teste de cadastro com apelido não utilizado | Redirecionamento para a página de Membros com a mensagem de sucesso exibida:  **Membro cadastrado com sucesso! Pedro/Peralta** |
|  | Promover para Morador – Promover bixo para morador | Verificar a promoção de um bixo em processo seletivo para morador junto ao sistema.  No menu principal, ir em República > Gerenciar Bixos> selecionar o bixo José Mário/ Mineirão e no select clicar em **Promover a Morador**  De forma alternativa, a área em questão pode ser acessada clicando no botão Bixos na página inicial. | A entrada será a própria interação.  No caso, a opção do select do bixo em questão, **Promover a morador** deverá ser selecionada. | Redirecionamento para a página de Moradores com a mensagem de sucesso exibida:  **O bixo foi promovido a morador com sucesso: José Mario/Mineirão**  O novo morador constará na listagem na posição de hierarquia 2 e não será o decano.  Ao acessar o módulo de Bixos posteriormente , os dados relativos a sua escolha e finalização do processo seletivo serão exibidos como:  **Morador já consolidado** |
|  | Promover para Morador – Validação de bixo já promovido | Verificar se um bixo já promovido a morador poderá ser promovido novamente via inserção de url da respectiva rota de tratamento no sistema. Isso não é permitido. | Inserção da URL relativa a promoção do morador Paulo Mateus/ Paulinho no navegador e tentar promovê-lo novamente  [**https://app-sgr.herokuapp.com/promocao-bixo-a-morador/2**](https://app-sgr.herokuapp.com/promocao-bixo-a-morador/2) | Redirecionamento para uma página **404 – Not Found** visto que a operação em questão não é permitida. |

## Relatório de Execução de Testes de Software

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Número** | **Caso de teste** | **Saída esperada** | **Resultados encontrados** | **Aprovado?** |
|  | Administrar República – Edição da república | Redirecionamento para a página inicial do sistema com a mensagem de sucesso exibida:  **República editada com sucesso!** | Redirecionamento para a página inicial do sistema com a mensagem de sucesso exibida:  **República editada com sucesso!** | SIM |
|  | Administrar República – Exclusão da república com Membros | Redirecionamento para a página inicial do sistema com a mensagem de erro exibida:  **República não pode ser excluída pois existem membros cadastrados!** | Redirecionamento para a página inicial do sistema com a mensagem de erro exibida:  **República não pode ser excluída pois existem membros cadastrados!** | SIM |
|  | Administrar Membros – Cadastrar Membro com apelido já registrado | Redirecionamento para a tela do formulário em questão com todos os campos preenchidos após tentativa de submissão com a mensagem de erro exibida:  **Já existe um membro com este apelido cadastrado!** | Redirecionamento para a tela do formulário em questão com todos os campos preenchidos após tentativa de submissão com a mensagem de erro exibida:  **Já existe um membro com este apelido cadastrado!** | SIM |
|  | Administrar Membros – Cadastrar Membro sem apelido registrado | Redirecionamento para a página de Membros com a mensagem de sucesso exibida:  **Membro cadastrado com sucesso! Pedro/Peralta** | Redirecionamento para a página de Membros com a mensagem de sucesso exibida:  **Membro cadastrado com sucesso! Pedro/Peralta** | SIM |
|  | Promover para Morador – Promover bixo para morador | Redirecionamento para a página de Moradores com a mensagem de sucesso exibida:  **O bixo foi promovido a morador com sucesso: José Mario/Mineirão**  O novo morador constará na listagem na posição de hierarquia 2 e não será o decano.  Ao acessar o módulo de Bixos posteriormente , os dados relativos a sua escolha e finalização do processo seletivo serão exibidos como:  **Morador já consolidado** | Redirecionamento para a página de Moradores com a mensagem de sucesso exibida:  **O bixo foi promovido a morador com sucesso: José Mario/Mineirão**  O novo morador constará na listagem na posição de hierarquia 2 e não será o decano.  Ao acessar o módulo de Bixos posteriormente , os dados relativos a sua escolha e finalização do processo seletivo serão exibidos como:  **Morador já consolidado** | SIM |
|  | Promover para Morador – Validação de bixo já promovido | Redirecionamento para uma página **404 – Not Found** visto que a operação em questão não é permitida. | Redirecionamento para uma página **404 – Not Found** visto que a operação em questão não é permitida. | SIM |

## Apropriação de Horas no Projeto

|  |  |  |
| --- | --- | --- |
| **Histórico de apropriação de horas** | | |
| **Data do registro** | **Atividade** | **Quantidade de horas** |
| 26/11/2021 -01/12/2021 | Organização inicial das idéias e cenários de aplicação do projeto SGR e seus direcionamentos futuros. | 30 |
| 02/12/2021 - 04/12/2021 | Expansão no detalhamento do texto de introdução partindo dos fatores de aspecto macro para micro, propiciando maior entendimento da aplicação. | 16 |
| 05/12/2021 -08/12/2021 | Criação do Diagrama de Casos de Uso da aplicação. | 21 |
| 09/12/2021 - 10/12/2021 | Especificação de requisitos funcionais da aplicação. | 8 |
| 11/12/2021 - 12/12/2021 | Especificação dos requisitos não funcionais da aplicação. | 8 |
| 13/12/2021 - 16/12/2021 | Criação do Diagrama de Classes de Domínio. | 24 |
| 17/12/2021 - 22/12/2021 | Criação do Protótipo interativo no Figma abordando os três cenários escolhidos para desenvolvimento dentro do Diagrama de Casos de Uso | 40 |
| 23/12/2021 - 23/12/2021 | Criação do repositório da aplicação no GitHub | 2 |
| 07/02/2022 | Entrega dos itens relativos a primeira etapa do projeto integrado | 1 |
| 02/03/2022 | Definição do padrão arquitetural da aplicação. | 2 |
| 03/05/2022 -05/03/2022 | Criação do Diagrama C4 Model. | 3 |
| 06/03/2022 - 07/03/2022 | Frameworks de trabalho | 1 |
| 08/03/2022 - 09/03/2022 | Estrutura base do Front End. | 5 |
| 10/03/2022 -15/03/2022 | Desenvolvimento de uma funcionalidade. | 24 |
| 15/03/2022 -16/03/2022 | Correção e ajustes após retorno do orientador na primeira etapa. | 12 |
| 16/03/2022 - 18/03/2022 | Submissão do Projeto no Heroku. | 2 |
| 30/03/2022 - 31/03/2022 | Submissão do Relatório Técnico e demais itens desenvolvidos na 2ª etapa do Projeto na pasta docs no GitHub. | 3 |
| 31/03/2022 | Entrega da 2ª etapa do Projeto Integrado. | 1 |
| 22/04/2022 -25/04/2022 | Correção e ajustes após retorno do orientador na segunda etapa. | 0 |
| 26/04/2022 -27/04/2022 | Modelo relacional ou Projeto de banco de dados NoSql. | 2 |
| 28/04/2022 – 01/05/2022 | Construção inicial do Projeto com base na documentação levantada nos passos anteriores. | 18 |
| 02/05/2022 -22/05/2022 | Desenvolvimento da aplicação e respectivos casos de uso selecionados | 80 |
| 23/05/2022 – 25/05/2022 | Desenvolvimento do plano de testes. | 4 |
| 25/05/2022 – 28/05/2022 | Relatório de execução dos testes. | 3 |
| 29/05/2022 – 30/05/2022 | Informações relacionadas a implementação. | 2 |
| 31/05/2022 – 01/06/2022 | Avaliação retrospectiva. | 2 |
| 01/06/2022 – 02/06/2022 | Objetivos estimados,Objetivos alcançados e Lições aprendidas. | 3 |
| 03/06/2022 – 04/06/2022 | Desenvolvimento do vídeo final de apresentação do projeto integrado. | 5 |
| 05/06/2022 – 06/06/2022 | Atualização final do GitHub e Heroku com os arquivos e documentação relacionados ao Projeto Integrado. | 2 |
| 14/06/2022 – 15/06/2022 | Entrega da 3ª Etapa do Projeto Integrado. | 1 |
| **HORAS TOTAIS** |  | **325** |

## Código da Aplicação

Repositório no GitHub: [Link do Projeto SGR](https://github.com/rscsouza/sgr)

Endereço da aplicação no Heroku: [Link do Projeto SGR no Heroku](http://app-sgr.herokuapp.com/login)

Usuário: [admin@sgr.com](mailto:admin@sgr.com)

Senha: senha

Vídeo com o protótipo desenvolvido na primeira etapa: [Link do vídeo](https://github.com/rscsouza/sgr/raw/master/docs/prototipo_sgr_v1.mp4)

Vídeo com a primeira funcionalidade desenvolvida para a segunda etapa (desktop): [Link do vídeo](https://github.com/rscsouza/sgr/raw/master/docs/crud_republica_desktop.mp4)

Vídeo com a primeira funcionalidade desenvolvida para a segunda etapa (móbile): [Link do vídeo](https://github.com/rscsouza/sgr/raw/master/docs/crud_republica_mobile.mp4)

Vídeo final apresentando todas as funcionalidades para a terceira etapa : [Link do vídeo](https://github.com/rscsouza/sgr/raw/master/docs/video_final_desktop.mp4)

## Avaliação Retrospectiva

O cenário iniciou-se pelo entendimento do problema que deveria ser solucionado. No caso, a abstração dos processos que levam a manutenção de uma república estudantil ouro pretana.

Inicialmente foi gerado um cronograma tendo como finalidade a execução daquilo que estava planejado para o projeto dentro de seus respectivos prazos.

Após analisar o problema em si, foi gerada uma definição conceitual do mesmo que posteriormente veio a ser desmembrada na criação dos casos de uso da aplicação, bem como os diagramas de classes de domínio. Os requisitos funcionais e não funcionais da aplicação surgiram oriundos dos casos de uso, sendo necessária a escolha de três casos de uso para o desenvolvimento dentro do universo estabelecido para a aplicação.

Tendo estes cenários definidos, foi desenvolvido um protótipo primitivo e wireframes para demonstrar o funcionamento previsto para o sistema, sendo estes encaminhados ao orientador, no caso, simulando o cliente que ficaria a par daquilo que seria entregue posteriormente.

O cronograma com as atividades previstas também foi elaborado e repassado ao orientador. Com base nisso e em sua aprovação, foram definidas a arquitetura da solução, no caso o MVC,o diagrama C4 Model da aplicação, frameworks de trabalho e estrutura base do frontend. As horas do projeto eram apropriadas de forma incremental em cada entrega por etapas.

O modelo de banco de dados foi definido posteriormente de acordo com os levantamentos anteriores provenientes do Diagrama das Classes de Domínio, sendo na seqüência implementada uma primeira funcionalidade, no caso um CRUD de República que permitia a execução das operações selecionadas. Sendo este o primeiro artefato entregue ao cliente, os demais seriam entregues de forma incremental na terceira etapa.

Com a validação do orientador a partir dos requisitos da segunda etapa, foi gerada a imagem do Modelo Relacional de SGBD, no caso utilizando o MySQL e o Workbench.

A aplicação foi desenvolvida tendo como base todos os levantamentos documentais anteriores, considerando os casos de uso selecionados aqueles que possuem links para execução de suas atividades no sistema, aqueles que estão como inativos constam como desabilitados ou em “Em desenvolvimento”.

Foram definidos planos de testes para serem executados em cima dos Casos de Uso selecionados para o projeto e após a execução dos testes nos mesmos, o preenchimento do formulário com seus respectivos resultados.

Após validação dos testes e de acordo com a proposta do projeto, o sistema em questão foi submetido a plataforma Heroku para simulação e teve seu código fonte disponibilizado no Github para fins de avaliação.

Sobre o GitHub e Heroku deste projeto, eles sempre se encontraram disponíveis e atualizados em cada etapa de entrega com os respectivos artefatos solicitados para avaliação.

## Objetivos Estimados

O objetivo desta aplicação é a simulação da manutenção da estrutura gerencial de uma república ouro pretana. Por ser um processo histórico e de inúmeras peculiaridades foram selecionados três casos de uso correspondentes aos itens comuns da organização que são:

- A República: cadastro/ edição / exclusão da entidade República, com dados pertinentes a organização em cenários diversos.

- Membros: cadastro / edição / exclusão da Entidade Membro. Membros são vinculados a Bixos, Moradores e Ex Alunos e esta associação dentre as etapas deveria ser respeitada de forma a simular o processo.

- Promoção Calouro-Bixo a Morador: o processo seletivo em uma república ouro pretana é fundamental, visto que o bixo ou calouro que ingressa será posteriormente promovido a morador, tornando-se necessário a promoção do mesmo por meio de critérios hierárquicos a morador.

## Objetivos Alcançados

Todos os objetivos elencados como estimados acima foram alcançados conforme descritos em sua totalidade.

## Lições aprendidas

|  |  |  |
| --- | --- | --- |
|  | **Retrospectiva (Lições Aprendidas)** | |
|  | **Descrição da Lição** | **Classificação** |
| 1 | Utilização do Framework Laravel em sua versão 8.0 no lugar da 5.8 em que estava acostumado a atuar. | Positiva |
| 2 | Ampliação do conhecimento em relação ao Heroku , sendo que nunca havia utilizado o mesmo para simular meu código em um ambiente online. | Positiva |
| 3 | Utilização do db4free.net para geração de banco de dados online no MySQL , também não havia utilizado esta abordagem | Positiva |
| 4 | Correção da Entidade Membro em relação ao tipo de perfil, inicialmente havia proposto três variáveis booleanas para determinar em qual perfil o Membro se encontrava( Nenhum Bixo, Morador ou Ex aluno). Por sugestão do orientador, simplifiquei para uma única variável inteira , o que facilitou a validação em relação a cada etapa via código. | Positiva |
| 5 | Organização hierárquica bem definida entre classes de forma a simular o comportamento esperado para execução das atividades republicanas. | Positiva |
| 6 | Limitações do modelo MVC em relação a usabilidade em conjunto ao Laravel, determinadas partes do sistema poderiam apresentar comportamento de SPA – Single Page Application , sendo utilizada outra abordagem , talvez o MVVM em Angular. | Negativa |
| 7 | Utilização do Booststrap para tratar a responsividade entre dispositivos. | Positiva |
| 8 | Obter conhecimento mais amplo sobre documentação e melhores práticas de programação. | Positiva |
| 9 | Troca de orientadores no decorrer do projeto. **Sobre a orientação não tenho o que reclamar**, somente alguns pequenos percalços iniciais em relação ao alinhamento de expectativas sobre a execução do projeto que tiveram que ser repassadas posteriormente pelo antigo orientador, isso gerou uma certa insegurança inicial justamente pela falha na comunicação institucional. Creio que tenha impactado na nota de minha primeira etapa que foi relativamente baixa. | Negativa |
| 10 | Descrição dos termos técnicos do projeto integrado na página do Canvas gera ambigüidade de interpretação em alguns casos ,visto que o material parece não estar unificado, sendo esta uma dúvida e queixa de outros alunos. | Negativa |
| 11 | Possibilidade de criar um cronograma e segui-lo em busca da perfeição ao longo das etapas | Positiva |
| 12 | Conhecimento da ferramenta Astah para confecção de diagramas | Positiva |