****

**INSTITUTO POLITÉCNICO DE LEIRIA**



MUSICAE

RELATÓRIO DE projeto em sistemas de informação

Relatório de Projeto em Sistemas de Informação para cumprimento dos requisitos necessários à realização da prova de apresentação de projeto do Curso Técnico Superior Profissional (TeSP) de **Programação de Sistemas de Informação** realizado sob a orientação de Sílvio Mendes.

Jason Mendes 2180601

pedro Alves 2180630

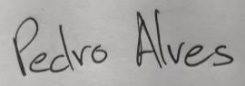
Pedro lopes 2180626

Programação em Sistemas de informação

07/02/2020

**DECLARAÇÃO**

Declaro que este Relatório se encontra em condições de ser apreciada (o) pelo júri a designar.

O estudante 1,

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Leiria, 7 de fevereiro de 2020

Relatório de Projeto em Sistemas de Informação para cumprimento dos requisitos necessários à realização da prova de apresentação de projeto do Curso Técnico Superior Profissional (TeSP) de **Programação de Sistemas de Informação** realizado sob a orientação de Sílvio Mendes.

**DECLARAÇÃO**

Declaro que este Relatório se encontra em condições de ser apreciada (o) pelo júri a designar.

O estudante 2,

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Leiria, .... de ............... de ...............

Relatório de Projeto em Sistemas de Informação para cumprimento dos requisitos necessários à realização da prova de apresentação de projeto do Curso Técnico Superior Profissional (TeSP) de **Programação de Sistemas de Informação** realizado sob a orientação de Sílvio Mendes.

**DECLARAÇÃO**

Declaro que este Relatório se encontra em condições de ser apreciada (o) pelo júri a designar.

O estudante 3,

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Leiria, .... de ............... de ...............

# Resumo

**MUSICAE**

**Jason Mendes**

**Pedro Alves**

**Pedro Lopes**

O nosso projeto consiste numa rede de procura de bandas e músicos. Por exemplo, se um músico for violinista e estiver à procura de uma banda, na aplicação / site escolhe os filtros de acordo com as suas preferências e há de lhe aparecer uma lista de bandas que estarão de acordo com as estas, neste caso uma banda que precisasse de um violinista + o resto dos filtros que o músico utilizou.

Temos um site totalmente interativo cujas funcionalidades são para registar, depois disso o utilizador faz login e personaliza o seu perfil consoante o tipo de instrumento que toca, anos de experiência, se quer levar mais a sério, se já esteve em alguma banda, etc… Depois disso poderá aceder a uma procura de bandas consoante o gosto dele e dos instrumentos que estão há procura, pode se candidatar para a banda tendo lá as informações da banda, a descrição, o estilo de música, experiencia, etc… e depois o dono da banda lá no site pode aceitar ou não.

No BackOffice do site conseguimos fazer a gestão do site, conseguindo ver qualquer tipo de informação introduzida pelos utilizadores. Os utilizadores normais não têm acesso ao mesmo.

No FrontOffice do site é o que um utilizador normal vê e é onde faz as procuras dos tais músicos / bandas.

Temos também uma aplicação android pois no mercado de aplicações não existe quase nenhuma sobre procura de bandas ou músicos então decidimos criar sobre esse tema. As funcionalidades são basicamente as referidas anteriormente, mas de uma forma mais interativa, com mais menus, algo mais simples e organizado pois o que um cliente quer numa aplicação é algo simples mas que esteja tudo muito organizado.

# Índices

O relatório deve conter o índice pr incipal, o índice de figuras e o índice de tabelas.

Índice

[Resumo v](#_Toc32000013)

[Índices vi](#_Toc32000014)

[Introdução 1](#_Toc32000015)

[Apresentação do trabalho. Deve incluir a organização e a estrutura do relatório. 1](#_Toc32000016)

[Metodologia 2](#_Toc32000017)

[Arquitetura do Sistema 3](#_Toc32000018)

[Gestão do Projeto 6](#_Toc32000019)

[Análise 8](#_Toc32000020)

[Desenho 14](#_Toc32000021)

[Implementação 20](#_Toc32000022)

[Testes 21](#_Toc32000023)

[Conclusão e trabalho futuro 26](#_Toc32000024)

[Webgrafia 27](#_Toc32000025)

[Anexos 28](#_Toc32000026)

Índice de Figuras

[Figura 1 – Gantt Chart 7](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999808)

[Figura 2 – Trello Lista 1 8](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999809)

[Figura 3 – Trello Lista 2 8](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999810)

[Figura 4 – Trello Lista 3 8](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999811)

[Figura 5 – Trello Lista 4 8](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999812)

[Figura 6 – Trello Lista 5 8](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999813)

[Figura 7 – Trello Lista 6 8](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999814)

[Figura 8 – Casos de Uso 15](#_Toc31999815)

[Figura 9 – Página de Login 16](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999816)

[Figura 10 – Página de Loading 16](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999817)

[Figura 11 – Registo Instrumentos 17](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999818)

[Figura 12 – Registo Géneros 17](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999819)

[Figura 13 – Registo Final 17](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999820)

[Figura 14 – Registo Detalhes Conta 17](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999821)

[Figura 15 – Feed Activity 18](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999822)

[Figura 16 – Login Activity 18](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999823)

[Figura 17 – Minhas Bandas 18](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999824)

[Figura 18 – Meu Perfil 18](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999825)

[Figura 19 - Candidatos 19](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999826)

[Figura 20 – Perfil Banda 19](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999827)

[Figura 21- Procurar Pessoas 19](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999828)

[Figura 22 – Criar Banda 19](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999829)

[Figura 23- Side Menu 20](file:///C:\wamp64\www\MusicaeAndroid\Relatórios\Modelo%20de%20Relatório%20Final%20_ProjetoSistemasInformação.docx#_Toc31999830)

# Introdução

Este relatório está dividido em diferentes pontos mostrando primeiro do que se trata o nosso projeto, o tema e como é composto, de seguida mostrando as partes mais visuais do que os utilizadores vão ver e depois sobre o que temos implementado em linguagem mais técnica e direta.

Tópicos deste Projeto:

* Metodologias:
* **Arquitetura do Sistema:** Falamos sobre termos técnicos que temos na nossa aplicação Android e no Website e o que é que o utilizador vê quando entra na app.
* **Gestão do Projeto:** Mostra de quem fez o quê no projeto em si e das datas de quando foram feitas.
* **Análise do Projeto:** Referimos os nossos Sistemas Funcionais, Não Funcionais e os Excluídos da ideia inicial.
* **Desenho:** Parte mais gráfica do nosso projeto. Contém imagens das Mockups, dos Casos de uso e Modelo de Dados.
* **Implementação:** aaaaaaaaa
* **Testes:** Tem os testes unitários, funcionais e de aceitação e o que é que fazem.
* **Conclusão**
* **Webgrafia:** Websites donde fomos buscar algumas ideias.

# Metodologia

Deve ser especificada a metodologia adotada no desenvolvimento do Projeto em Sistemas de Informação

# Arquitetura do Sistema

Na **Aplicação** **Android** contém:

* 14 Atividades – Consiste nas vistas que a aplicação tem.
* 3 Fragments – Para mostrar os dados que estão gravados na API.
* 7 Atividades WebService - Para obtenção e envio de Dados.
* 6 Atividades de introdução de dados pelo utilizador- Para inserção de dados.
* 2 Atividades WebService com os dados gravados anteriormente.
* 4 Adaptadores
* 4 Listeners – Para fazer Refresh e Update.
* Side Menu – Menu utilizado na nossa Aplicação.
* Registo de dados em ficheiros locais e base de dados SQL local e API REST

Funcionalidades Implementadas:

* Sistema de Login.
* Lembrar conta ao fazer login usando as ‘*SharedPreferences’*.
* Visualização do perfil do utilizador de acordo com as suas informações.
* Visualização das bandas.
* Visualização do Feed.
* Logout.
* Guardar os dados numa base de dados local.
* Guardar os dados através da REST API.

Funcionalidades Excluídas:

* Indústrias.
* Upload de Músicas.

No **Website** contém páginas de:

No **FrontOffice:**

* HomePage.
* Registo.
* Login.
* Perfil do Utilizador.
* Alteração de dados do Perfil.
* Criação de Banda.
* Bandas Atuais do Utilizador.

No **BackOffice**:

* Login
* Gestão de Bandas, Géneros, Habilidades, Músicos, Perfis, Utilizadores (Adicionar/Alterar/Remover).
* Acesso à API do Website.

Funcionalidades Implementadas:

* Framework Yii2.
* Sistema de Login (BackOffice e FrontOffice).
* Sistema de Autorização RBAC (BackOffice e FrontOffice).
* Gestão de Bandas, Géneros, Habilidades, Músicos, Perfis e Utilizadores (BackOffice).
* Acesso à API (BackOffice).
* Serviço de Autenticação para aceder à REST API (BackOffice).
* Sistema de Messaging utilizando o Mosquitto.
* Testes de Software (Unitários, Funcionais e de Aceitação).

Funcionalidades Excluídas:

* Indústrias.
* Upload de Músicas.

# Gestão do Projeto

Neste capítulo deve ser apresentado o plano do Projeto (Sugestão: utilizar a ferramenta de Gestão de projetos Trello):

* Gantt Chart;
* Identificação das principais actividades;
* Deliverables;
* Milestones;
* Equipa de Projeto e funções respetivas.

Utilizámos a ferramenta Trello pois achámos que seria mais fácil, eficaz e interativa para gerir e utilizarmos enquanto fazíamos o projeto.

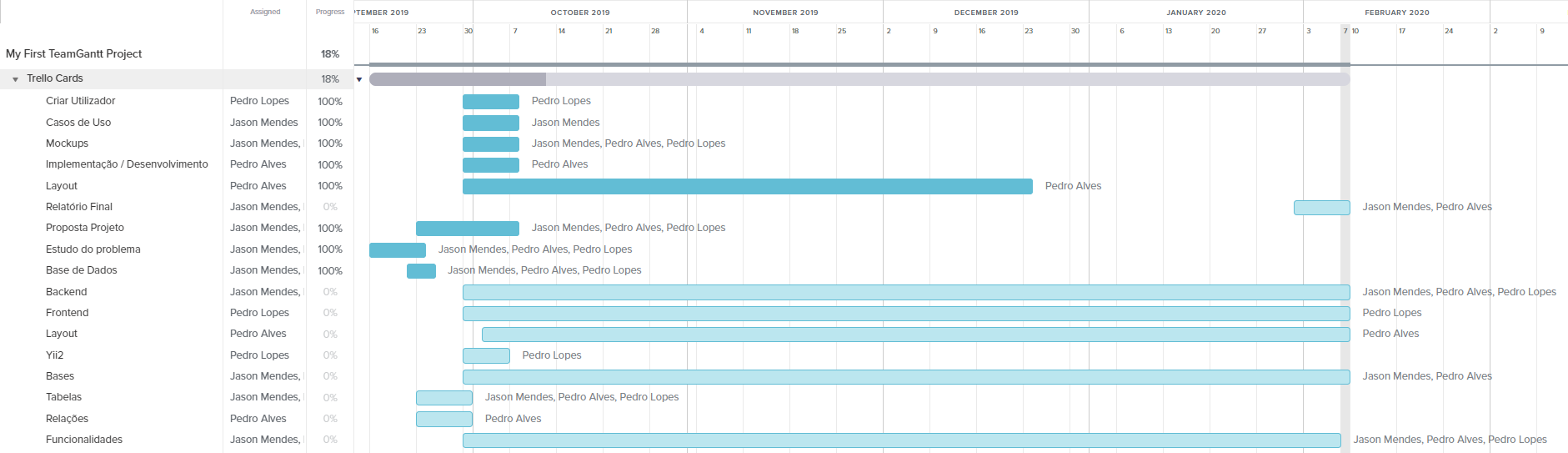


Figura 1 – Gantt Chart

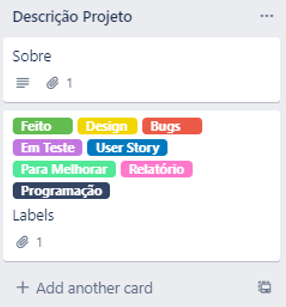


Figura 2 – Trello Lista 1

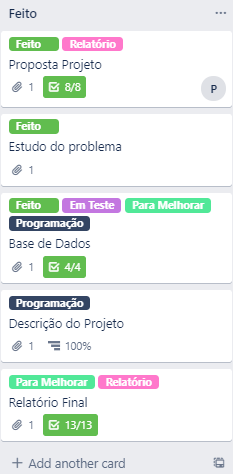


Figura 3 – Trello Lista 2

Figura 4 – Trello Lista 3



Figura 5 – Trello Lista 4

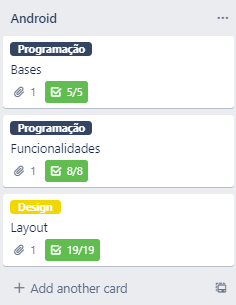


Figura 6 – Trello Lista 5

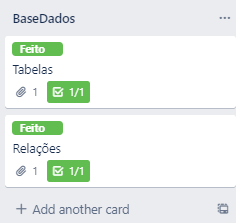


Figura 7 – Trello Lista 6

# Análise

O nosso projeto, como foi referido anteriormente, é uma rede social de procura entre músicos e bandas com o objetivo de se por exemplo um utilizador se registar como um músico que souber tocar guitarra e houver bandas à procura dum guitarrista esse utilizador pode entrar em contacto com eles via Website/Aplicação ou até mesmo por telemóvel, candidatando-se então para a mesma.

Requisitos Funcionais:

**Criação de conta:** Função que permite criar uma conta tanto na aplicação como no website.

* **Atores:** Pessoa
* **Pré-Condições:** Ter um e-mail válido.
* **Sequência de acontecimentos:** 1) O utilizador seleciona a opção de criar utilizador 2) Utilizador insere o nome, o e-mail e a password que deseja e submete.
* **Caminhos Alternativos:** N/A
* **Pós-Condições:** Utilizador já pode entrar na sua conta.

**Login:** Função que permite o utilizador entrar na aplicação com a sua conta.

* **Atores:** Utilizador
* **Pré-Condições:** Ter uma conta já criada.
* **Sequência de acontecimentos:** 1) Utilizador insere o seu username e a sua password.
* **Caminhos Alternativos:** N/A
* **Pós-Condições:** Utilizador já pode usufruir da aplicação.

**Logout:** Função que permite o utilizador sair da aplicação

* **Atores:** Utilizador
* **Pré-Condições:** Ter uma feita e estar logado.
* **Sequência de acontecimentos:** 1) Utilizador carrega no botão de logout.
* **Caminhos** **Alternativos:** N/A
* **Pós-Condições:** Utilizador tem de fazer login novamente para usufruir da aplicação

**Alteração de password:** Função que permite mudar a palavra-passe.

* **Atores**: Utilizador
* **Pré-Condições**: Ter uma conta já criada.
* **Sequência de acontecimentos:** 1) Utilizador seleciona opção para mudar password 2) Utilizador insere a password e de seguida insere a nova password que desejar.
* **Caminhos Alternativos:** N/A
* **Pós-Condições:** Utilizador já pode usufruir da aplicação.

**Editar Perfil:** Utilizador aqui poderá mudar o nome, idade, localização, género, email, descrição e mudar as fotos do perfil.

* **Atores:** Utilizador
* **Pré-Condições:** Ter uma feita e estar logado.
* **Sequência de acontecimentos:** 1) Utilizador vai ao seu perfil e carrega no botão “Editar Perfil”. 2) Utilizador procede a editar o seu perfil como enteder.
* **Caminhos** **Alternativos:** N/A
* **Pós-Condições:** Perfil ficará atualizado conforme as edições do utilizador.

**Criar Banda:** O utilizador aqui pode criar uma banda.

* **Atores:** Músico
* **Pré-Condições:** Ter conta feita e estar logado.
* **Sequência de acontecimentos:** 1) Utilizador vai ao menu.

2) Carrega no botão para criar banda.

3) Insere os dados da banda(estilo de música, descrição, dados pessoais, etc...) junto com fotografia e link da conta da banda no soundcloud se assim desejarem.

* **Caminhos** **Alternativos:** N/A
* **Pós-Condições:** Mostra a página da Banda criada.

**Adicionar Membros a uma Banda:** O utilizador aqui pode adicionar membros à banda criada.

* **Atores:** Músico ou Administrador
* **Pré-Condições:** Ter uma feita e estar logado.
* **Sequência de acontecimentos:** 1) Banda faz a pesquisa que pretende

2) Escolher dos músicos que lhe aparecer pela pesquisa feita.

3) Adicionar o músico que escolher.

* **Caminhos** **Alternativos:** N/A
* **Pós-Condições:** Mostra a página da Banda.

**Procurar Bandas:** Função que permite o utilizador procurar uma Banda ao seu gosto.

* **Atores:** Músico
* **Pré-Condições:** Ter uma conta feita e estar logado.
* **Sequência de acontecimentos:** 1) Utilizador carrega no botão para a procura de bandas. 2) Insere os filtros que quer dependendo do seu gosto musical ou do tipo de banda que deseja entrar.
* **Caminhos** **Alternativos:** N/A
* **Pós-Condições:** Mostra página de bandas consoante a procura feita pelo músico.

**Candidatar-se para a Banda:** Função que permite candidatar-se para entrar numa banda que deseja.

* **Atores:** Músico
* **Pré-Condições:** Fazer a procura de bandas consoante o seu gosto e abrir a página da mesma.
* **Sequência de acontecimentos:** 1) Carrega num botão para entrar na banda.
* **Caminhos** **Alternativos:** N/A
* **Pós-Condições:** Mostra que a candidatura foi enviada com sucesso e para aguardar pela resposta (Aceite/Rejeitado).

**Gestão da Candidaturas da Banda:** Função que permite aceitar / rejeitar músicos que querem entrar para a sua banda.

* **Atores:** Músico
* **Pré-Condições:** Ser o dono da banda ou estar na banda.
* **Sequência de acontecimentos:** 1) Abre a lista de candidaturas da banda podendo aceitar / rejeitar os músicos que estarão lá.
* **Caminhos** **Alternativos:** N/A
* **Pós-Condições:** Mostra página da banda.

**Gestão de Utilizadores:** Administrador poderá criar, alterar e banir utilizadores.

* **Atores:** Administrador
* **Pré-Condições:** Efetuar o login no Backend.
* **Sequência de acontecimentos:** 1) Administrador escolhe os utilizadores que desejar e efetua as alterações necessárias. (CRUD)
* **Caminhos** **Alternativos:** N/A
* **Pós-Condições:** N/A

**Gestão de Músicos:** Administrador poderá criar, alterar e banir músicos.

* **Atores:** Administrador
* **Pré-Condições:** Efetuar o login no Backend.
* **Sequência de acontecimentos:** 2) Administador escolhe o músico que desejar e efetua as alterações necessárias. (CRUD)
* **Caminhos** **Alternativos:** N/A
* **Pós-Condições:** N/A

**Gestão de Habilidades:** Administrador poderá criar, alterar e banir habilidades.

* **Atores:** Administrador
* **Pré-Condições:** Efetuar o login no Backend.
* **Sequência de acontecimentos:** 3) Administador escolhe a habilidade que desejar e efetua as alterações necessárias.
* **Caminhos** **Alternativos:** N/A
* **Pós-Condições:** N/A

**Gestão de Géneros:** Administrador poderá criar, alterar e banir géneros.

* **Atores:** Administrador
* **Pré-Condições:** Efetuar o login no Backend.
* **Sequência de acontecimentos:** 4) Administador escolhe o género que desejar e efetua as alterações necessárias
* **Caminhos** **Alternativos:** N/A
* **Pós-Condições:** N/A

**Gestão de Bandas:** Administrador poderá criar, alterar e banir músicos.

* **Atores:** Administrador
* **Pré-Condições:** Efetuar o login no Backend.
* **Sequência de acontecimentos:** 5) Administador escolhe a banda que desejar e efetua as alterações necessárias
* **Caminhos** **Alternativos:** N/A
* **Pós-Condições:** N/A

# Desenho

Especificar Protótipos, Casos de Uso, Modelo de Dados, Mockup’s, etc.

Casos de uso:

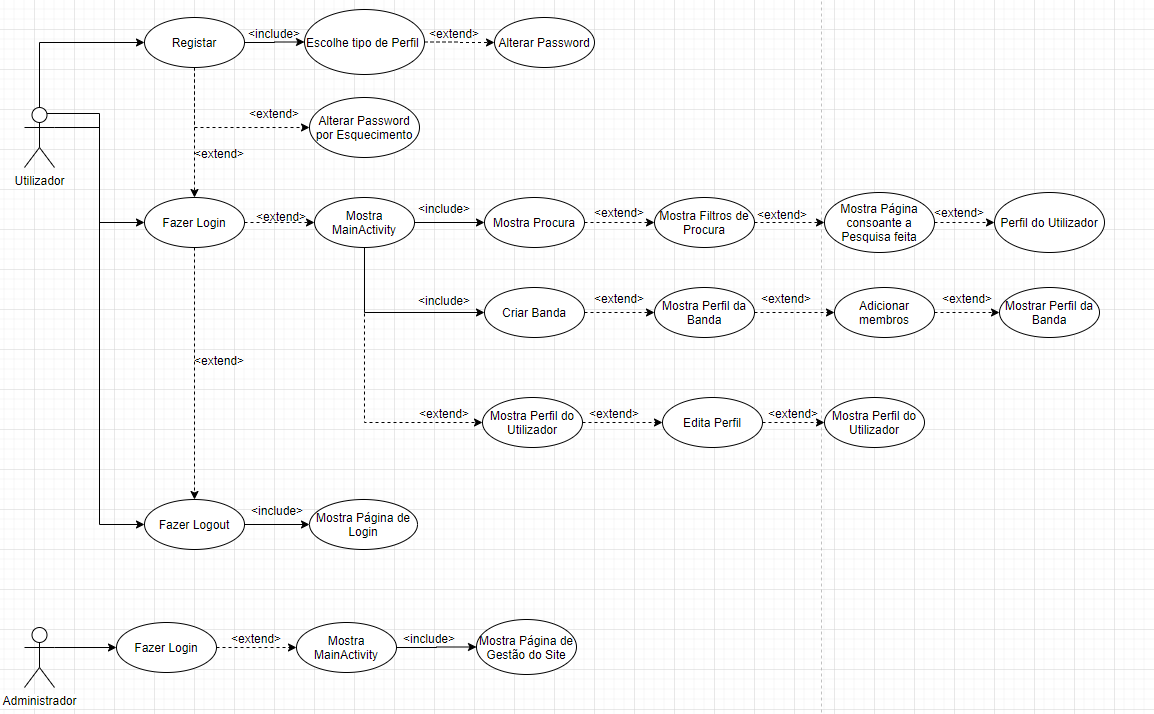


Figura 8 – Casos de Uso

Modelo de dados:

??????????????????????????????????????????????????????????????????????????????????????????????

Web Mockups:

---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Android Mockups:

SplashScreen WelcomeScreen



Figura 9 – Página de Login

Figura 10 – Página de Loading

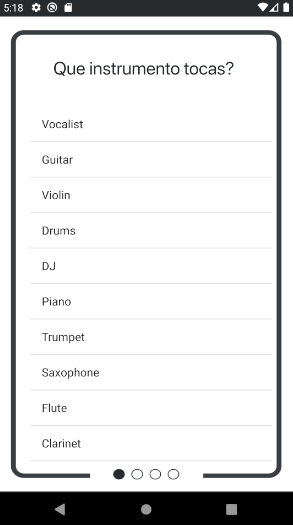


Figura 11 – Registo Instrumentos

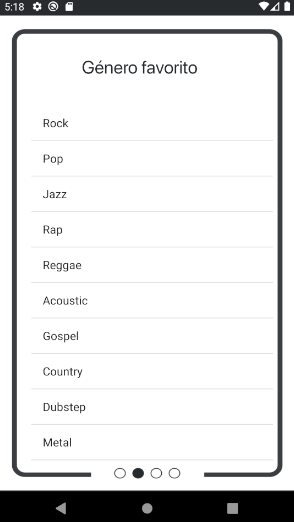


Figura 12 – Registo Géneros

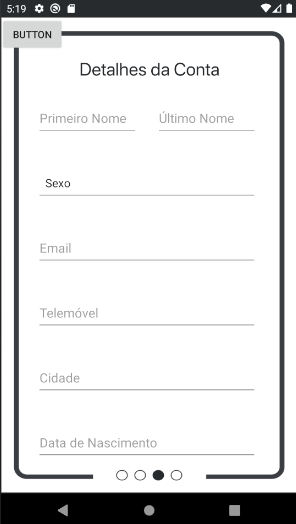
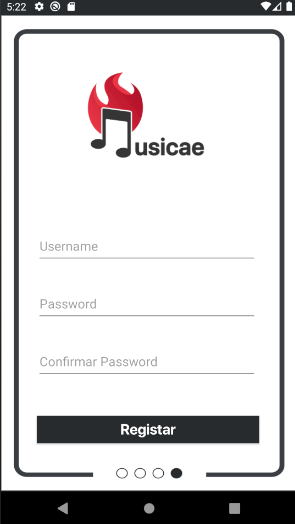


Figura 13 – Registo Final

Figura 14 – Registo Detalhes Conta

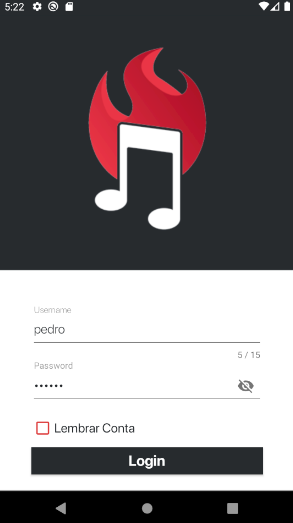
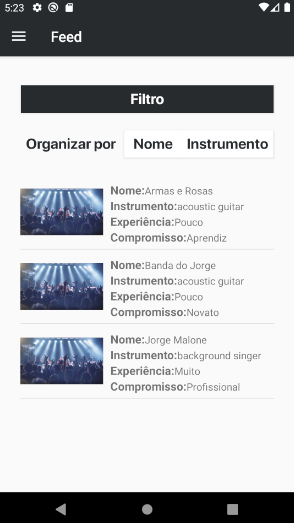


Figura 15 – Feed Activity

Figura 16 – Login Activity

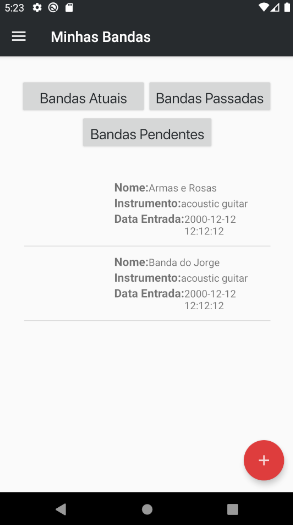


Figura 17 – Minhas Bandas



Figura 18 – Meu Perfil



Figura 19 - Candidatos

Figura 20 – Perfil Banda

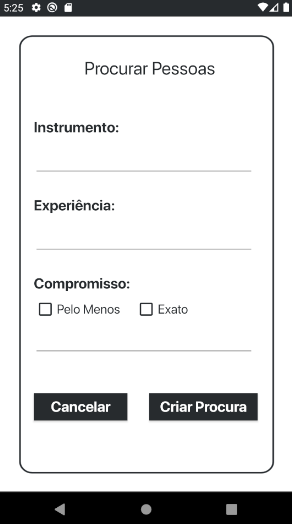


Figura 21- Procurar Pessoas

Figura 22 – Criar Banda

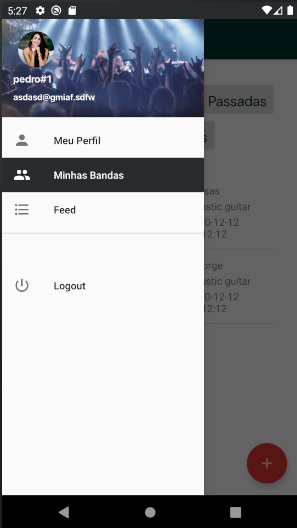


Figura 23- Side Menu

# Implementação

Explicar o processo de desenvolvimento do software.

# Testes

As ferramentas de testes integração/unitários e funcionais que nós utilizamos foi o ‘*Git Bash*’ para introduzir as linhas de comando das mesmas. Para os de Aceitação utilizamos o Selenium e o ChromeDriver para conseguirmos ligar ao Selenium.

O nosso projeto cumpre com todos os requisitos relacionados a testes.

Para a resolução dos testes foi criado uma base de dados clone, “musicaedb\_test”.

**Testes Unitários**

|  |  |  |
| --- | --- | --- |
| ProfileTest  (backend) | testValidation | Despoleta todas as regras de validação de todos os dados da tabela “profiles” |
| ProfileTest | testSavingUser | Cria um novo registo válido e guarda na BD, verificando depois se se encontra na mesma |
| ProfileTest | testUserNameCanBeChanged | Lê o registo anterior e aplica um update, verificando depois se se encontra na BD com o update aplicado |
| ProfileTest | testUserDeleted | Apaga o registo anterior, verificando depois se este não se encontra mais na BD |

|  |  |  |
| --- | --- | --- |
| MusicoTest  (backend) | testValidation | Despoleta todas as regras de validação de todos os dados da tabela “musicos” |
| MusicoTest | testSavingMusico | Cria um novo registo válido e guarda na BD, verificando depois se se encontra na mesma |
| MusicoTest | testMusicoCanBeChanged | Lê o registo anterior e aplica um update, verificando depois se se encontra na BD com o update aplicado |
| MusicoTest | testMusicoDeleted | Apaga o registo anterior, verificando depois se este não se encontra mais na BD |

|  |  |  |
| --- | --- | --- |
| HabilidadeTest  (backend) | testValidation | Despoleta todas as regras de validação de todos os dados da tabela “habilidades” |
| HabilidadeTest | testSavingHabilidade | Cria um novo registo válido e guarda na BD, verificando depois se se encontra na mesma |
| HabilidadeTest | testHabilidadeCanBeChanged | Lê o registo anterior e aplica um update, verificando depois se se encontra na BD com o update aplicado |
| HabilidadeTest | testHabilidadeDeleted | Apaga o registo anterior, verificando depois se este não se encontra mais na BD |

|  |  |  |
| --- | --- | --- |
| GeneroTest  (backend) | testValidation | Despoleta todas as regras de validação de todos os dados da tabela “generos” |
| GeneroTest | testSavingGenero | Cria um novo registo válido e guarda na BD, verificando depois se se encontra na mesma |
| GeneroTest | testGeneroCanBeChanged | Lê o registo anterior e aplica um update, verificando depois se se encontra na BD com o update aplicado |
| GeneroTest | testGeneroDeleted | Apaga o registo anterior, verificando depois se este não se encontra mais na BD |

|  |  |  |
| --- | --- | --- |
| BandaHistoricoTest  (backend) | testValidation | Despoleta todas as regras de validação de todos os dados da tabela “bandashistorico” |
| BandaHistoricoTest | testSavingBandasHistorico | Cria um novo registo válido e guarda na BD, verificando depois se se encontra na mesma |
| BandaHistoricoTest | testBandasHistoricoCanBeChanged | Lê o registo anterior e aplica um update, verificando depois se se encontra na BD com o update aplicado |
| BandaHistoricoTest | testBandasHistoricoDeleted | Apaga o registo anterior, verificando depois se este não se encontra mais na BD |

**Testes Funcionais**

|  |  |  |
| --- | --- | --- |
| LoginCest  (backend) | tryLogin | Iniciar a aplicação na Home Page. Navegar para a página de login. Preencher o formulário com dados de login válidos. Realizar a asserção pelo nome do utilizador |

|  |  |  |
| --- | --- | --- |
| CreateHabilidadeCest  (backend) | tryCreateHabilidade | (tryLogin)  Clicar em “habilidades”. Clicar “Criar habilidade”. É criada uma habilidade com dados válidos na BD. |

|  |  |  |
| --- | --- | --- |
| CreateGeneroCest  (backend) | tryCreateGenero | (tryLogin)  Clicar em “generos”. Clicar “Criar género”. É criado um género com dados válidos na BD. |

|  |  |  |
| --- | --- | --- |
| CreateBandaCest  (backend) | tryCreateBanda | (tryLogin)  Clicar em “bandas”. Clicar “Criar banda”. É criada uma banda com dados válidos na BD. |

|  |  |  |
| --- | --- | --- |
| CreateUserCest  (frontend) | tryCreateUser | Iniciar a aplicação na Home Page. Navegar para a página de Signup. Preencher o formulário com dados de signup válidos. (tryLogin) |

**Teste de Aceitação**

|  |  |  |
| --- | --- | --- |
| HomeCest  (frontend) | checkLogin | Iniciar a aplicação na Home Page. Navegar para a página de login. Preencher o formulário com dados de login válidos. Realizar a asserção pelo nome do utilizador |

# Conclusão e trabalho futuro

# Webgrafia

Fomos aos seguintes websites em busca de informação ou ideias:

<https://www.stackoverflow.com/>

<https://www.yiiframework.com/>

<https://forum.yiiframework.com/>

<https://codeception.com/>

# Anexos