HACKATHON 4º Período

Alunos: Rhuan Zucarelli; RA: 11210
Murilo V. Trindade; RA: 9055

Professor: Andre Dorigan

ENGENHARIA DE SOFTWARE

Na disciplina de Engenharia de Software temos como objetivo entender as necessidades de cada cliente para o projeto e transformar essas necessidades em requisitos de software que serão, posteriormente, entregue na forma de software funcionando.

Sabendo disso, para esse Hackathon, na disciplina de Engenharia de Software cada equipe deve criar um documento de texto, contendo

1. Os requisitos que serão desenvolvidos e entregues neste projeto. Os requisitos devem estar bem escritos e devem explicar o que cada um representa no sistema e como eles serão implementados pela equipe;

2. Um modelo de Diagrama de Casos de Uso que represente quais funcionalidades serão implementadas no sistema. Para os atores do diagrama as equipes podem trabalhar com os programadores do sistema (vocês) e quais serão as funcionalidades que cada programador será responsável por implementar.

1)

Mapa mental do projeto a ser desenvolvido:

FRAMEWORKS DE DESENVOLVIMENTO FRONT-END

- Fazer um projeto de listagem de _____, contendo informações dele ;
- Conter duas paginas:
  1 home,
  2 listagem da API;
- Conter um código " limpo".

API

DESENVOLVIMENTO PARA DISPOSITIVOS MÓVEIS - ANDROID

- Um APP que lista a API;
- Conter duas telas:
  1 Listar todos os ____ cadastrados
  2 Clicando no ____ selecionado exibir informações sobre ele.

TECNOLOGIAS EMERGENTES

- Documentar detalhado no README

O que acontece dentro do projeto:

O projeto consiste em um site desenvolvido em react que exibe uma API contendo dados de personagens do Rick and Morty, essa API também está ligado ao um aplicativo Android que assim como o site também exibe essa mesma lista de personagens com informações de cada um deles.

Na Parte de REACT, ela é feita da seguinte maneira, ele é composto pelo:

- **README.md:** Explica para o projeto o que ele tem que fazer para poder rodar, o que tem que fazer, o que tem que rodar, como vai baixar;
- **package.json:** É onde está localizado todas as configurações do react, como irá funcionar, os scripts, o browser;
- **package-lock.json:** É praticamente a mesma coisa que o package.json, só que com algumas extensões a mais como o babel, que é uma extensão de linguagem;
- **node_modules:** contém todas as dependências instaladas para o seu projeto;
- **.gitignore:** Ele vai ser um filtro para retirar coisas que não são nescessarias para subir no Git, como a node modules, que quando você da o comando para rodar ela já instala na sua maquina;
- **roots:** Serve para roterizar o que vai aparecer na sua tela;
- **App.jsx:** Ele importa toda a parte de rota e faz o percurso que elas vão seguir para jogar no index.jsx;
- **index.js:** é onde irá renderizar o App.jsx, quando a aplicação for construida ela seleciona o index e manda para o index.html, que fica dentro da pasta public;
- **pages:** Na pasta pages é onde estará localizado todas as paginas, como home, sobre;
- **footer:** Ano de desenvolvimento e quem desenvolveu;
- **home:** Onde é configurado a parte de puxar a API e como será tratado na tela;
- **sobre:** É a pagina que explica como foi criado o projeto;

- **scr:** Está a pasta componentes que é onde estão os componentes que mostram como estão configurado os cards, como tamanho descrição, como o NavBar;

- **Cards:** São basicamente o reflexo dos personagens;

- **NavBar:** Aparece toda a parte de roteirização e é a "onde" você clica para escolher qual pagina você quer ir;

- **ação:** É a pagina para passar as paginas, como são muitos personagens, não caberia tudo de uma vez;

- **public:** Renderiza na tela para o publico visualizar;

Já a parte de Android, existe uma pasta app, dentro dela a o a pasta java, nela tem 3 com o nome do **package**, a primeira pasta é a onde está praticamente todo o projeto, contendo:

- **Config**: Ele linka a API com o APP;

- **MainActivity:** Nele é listado os dados criando o adaptador que ir configurar como os dados são carregados, contexto que o objeto está, local onde estão os dados, item que servira de modelo para cada célula, quais campos dos dados serão carregados, objetos de tela onde dados vão ser carregados. Também será adiconado o adaptador criado na listView da tela carregando os dados do item selecionado na lista pelo index, criando o caminho para abrir a tela de detalhes, criando os parametros e adicionando os dados do item selecionado, adicionando os parametros no caminho de tela, abrindo a tela detalhes;

- **DetalhesActivity:** Que mostra a imagem e o nome dos personagens capturando o caminho de tela utilizado para abrir esta tela.

Ainda dentro do package, tem uma pasta chamada datasources, nela contem 2 arquivos:

- **BuscarDadosWeb:** Nele ele pega os dados da API e lista eles através do ArrayList<HashMap<String, String>> , capturando a primeira posição do vetor de strings, criando uma URL a partir do link recebido, criando uma conexão a partir da URL, Capturando o retorno que veio da conexão e salvando na memória e pegando o retorno dela para ser lido no buffer, enquanto existir dados para ler no reader ele salva op valor na variável linha e mudar o cursor para a próxima linha, faz isso ate o final do arquivo. Transforma o texto com os dados vindos da API em um objeto JSON, captura o vetor RESULTS que existe no objeto JSON, rodar toda a lista de objetos do RESULTS pega o item atual da lista e transforma em objeto JSON mapeando os campos do Array JSON para o Mapa de dados Nome, url e imagem;

- **DownloadImagem:** Ele liga com o imagem view e na public Download ele mapeia criando a URL a partir do link recebido, buscando os dados da URL e salvando em memória, criando um bitmap a partir dos dados salvos em memória depois ele retorna o bitmap baixado acima, no objeto de imagem da tela.

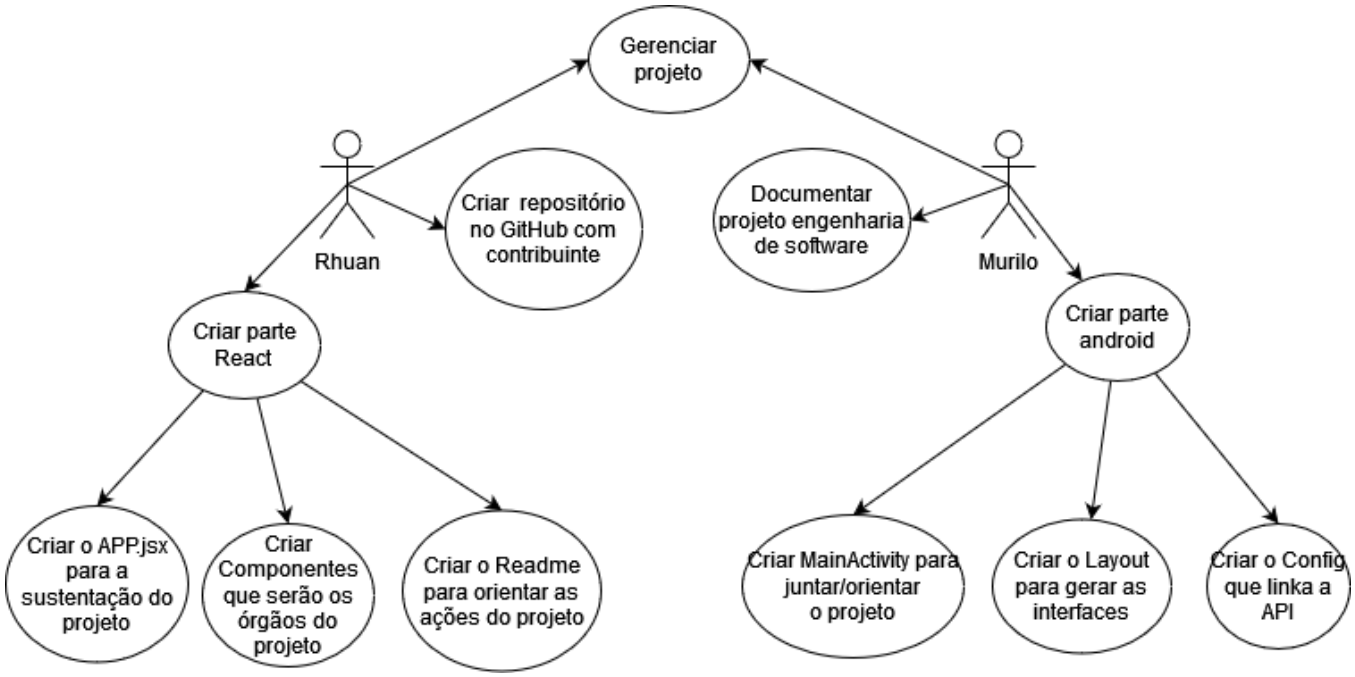
Fora da pasta java tem uma chamada res, dentro dela existe a pasta layout com os seguintes arquivos:

- **activity_detalhes.xml:** Com uma pagina responsável por mostrar o nome e a imagem do personagem;

- **activity_main.xml:** Que mostra a lista de personagem;

- **listview_modelo.xml:** Puxa os dados.

2)


Gerenciar projeto
Rhuan
Criar repositório no GitHub com contribuinte
Documentar projeto engenharia de software
Murilo
Criar parte React
Criar parte android
Criar o APP.jsx para a sustentação do projeto
Criar Componentes que serão os órgãos do projeto
Criar o Readme para orientar as ações do projeto
Criar MainActivity para juntar/orientar o projeto
Criar o Layout para gerar as interfaces
Criar o Config que linka a API