Manual de Instalação

Requerimentos

Para realizar a instalação é necessário ter os seguintes programas:

- Visual Studio Code ou o IntelliJ IDEA
- Node na versão 18 e com o npm 9
- MySQL 8.0.32
- Android Studio a versão mais nova

Para fazer a instalação do node seguir o tutorial : https://nodejs.org/en/learn/getting-started/how-to-install-nodejs ou https://www.freecodecamp.org/portuguese/news/como-instalar-o-node-js-e-o-npm-no-windows/

Para o MySQL: https://dev.mysql.com/doc/mysql-installation-excerpt/5.7/en/ ou https://www.alura.com.br/artigos/mysql-do-download-e-instalacao-ate-sua-primeira-tabela.

E para o Android Studio: https://developer.android.com/studio/install?hl=pt-br

Configuração do banco de dados

Acesse seu MySQL e crie o database ‘db_pepvagas’, isto pode ser feito através do comando:

```
CREATE DATABASE db_pepvagas;
```

Com o database criado você deve alterar o arquivo data_source.ts, que fica na pasta código-fonte/backend/database/, ele é um arquivo de configuração que vai conectar o backend com o database, você deve alterar os sequintes valores no arquivo.

```
...
username: "root",
password: "root",
...
```

Você pode criar um novo usuário no MySQL para ser usado pela plataforma ou usar algum já criado, mais garanta que ele tenha as permissões corretas para acessar essa database.

Executando o sistema como um desenvolvedor

Com o banco de dados configurado você de executar o backend da plataforma, para fazer isso execute o seguinte comando no terminar dentro da pasta código-fonte/backend:

```
npm i
```

Agora você pode executar o comando:

```
npm run dev
```

Caso tudo tenha funcionado deve aparecer a seguinte mensagem:

```
[.] Backend iniciado com sucesso!, porta: 4001
```

Agora para executar o frontend você deverá executar o mesmo comando para instalar as bibliotecas, só que na pasta projetoBase:

```
npm i
```

Com essa instalação feita agora você deve executar o seguinte comando:

```
ionic serve
```

Caso você tenha tido sucesso deve aparecer uma mensagem de sucesso no terminal e o seu navegador padrão deve abrir com o site.

Como gerar a versão mobile

Para realizar essa etapa é necessário ter o Android Studio instalado no seu computador, que está disponível em https://developer.android.com/studio?hl=pt-br, ele pode estar em qualquer versão, mais é recomendado usar a mais nova. Com ele baixado e instalado será necessário instalar um emulador para testar, você pode usar o que o Studio tem disponível nativamente ou usar outro como o Genymotion. Para mais informações de como usar o Android Studio é recomendado acessar a documentação deles, que pode ser acessada por https://developer.android.com/docs?hl=pt-br.

Para poder usar o código da plataforma no Android Studio é necessário gerar a versão do código que possa ser operado nele, isso pode ser feito através dos seguintes comandos:

```
ionic build
```

```
ionic cap add android
```

Lembre-se de apagar a pasta android antes de executar o comando acima

```
ionic cap sync
```

```
ionic cap build android
```

Ou para executar o código diretamente em um emulador ou celular aberto

```
ionic cap run
```

Se você executar o comando `cap build` ele irá abrir o android studio onde você poderá manipular e executar a plataforma. Agora adiocione a seguinte linha de código no andorid/app/src/main/AndroidManifest.xml na tag <application

Android:uisesCleartextTraffic=”true”

Para gerar o APK é necessário ter uma chave assinada, você pode criar uma para usar em testes, porem será necessário criar uma com a permissão do IFSP para que o aplicativo seja publicado.

Manual de Deploy

Building e deploy:

Entre no servidor de produção, antes de buildar o projeto, é necessário verificar se todos os pacotes estão atualizados. Vá até a raiz do projeto com o terminal, atualize a main branch com o ***“git pull”*** após ter certeza de que todos os pacotes estão atualizados, deve-se buildar cada lado da aplicação.

Frontend: dentro da pasta ***projetoBase*** verifique se todas as dependências estão atualizadas com o comando node ***“npm i”***, após atualizado, entre no arquivo services/axios.ts e descomente o domínio do site (***/*'http://vagas.pep2.ifsp.edu.br:8080/api' ||*/)*** para ***('http://vagas.pep2.ifsp.edu.br:8080/api' ||).*** *Volte para a raiz do projeto e rode o arquivo de configuração de deploy para o nginx* **‘deployFront.bash.save’** > Deseja excluir o deploy ja existente no nginx? (y/n) **coloque y,** automaticamente após esses passos, o frontend será atualizado na produção.

Backend: dentro da pasta ***backend*** verifique se todas as dependências estão atualizadas com o comando node ***“npm i”***, após atualizado, entre no arquivo database/data_source.ts e verifique se os dados de conexão com o mysql do Docker estão iguais a este:

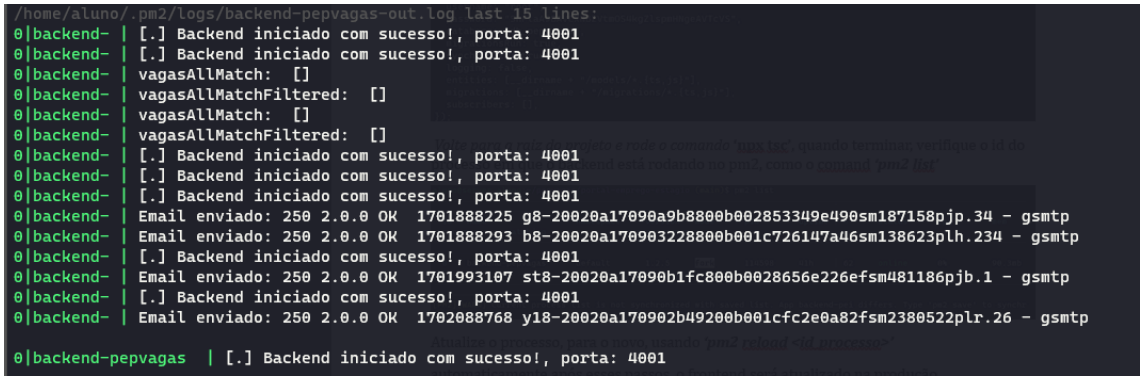
```
import "reflect-metadata";
import { DataSource } from "typeorm";
import dotenv from "dotenv";
dotenv.config({ path: __dirname + "/../.env" });


const port = process.env.DB_PORT as number | undefined;

export const AppDataSource = new DataSource({
  type: "mysql",
  host: "localhost",
  port: 3308,
  username: "root",
  password: "SnFlaA9BAVCXNBZVtmOS4kgZlspmHNgeAVTcVS",
  database: "db_pepvagas",
  migrationsRun: true,
  synchronize: true,
  logging: false,
  entities: [__dirname + "/models/*.{ts,js}"],
  migrations: [__dirname + "/migrations/*.{ts,js}"],
  subscribers: [],
});
```

Volte para a raiz do projeto e rode o comando **‘npx tsc’**, quando terminar, verifique o id do processo em que o backend está rodando no pm2, como o comand **‘pm2 list’**

Atualize o processo, para o novo, usando ***‘pm2 reload <id_processo>’***



Verifique se após o reload o backend foi iniciado com sucesso. Use o comand ***‘pm2 log <id_processo>’*** para verificar o log. Automaticamente após esses passos, o backend será atualizado na produção.

Observação

Caso alguma mudança no banco seja feita, ou caso o backend não esteja iniciando corretamente na produção, tente dropar o banco do Docker e criar de novo. Após isso, de reload no pm2 novamente.