

UNIVERSIDADE FEDERAL DO PARANÁ

FILIPPE DE FREITAS MARTINS DA SILVA

VINICIUS DUARTE DOS SANTOS

VINICIUS EDUARDO MENDES SAVITRAZ

MANUAL DE INSTALAÇÃO

ARRUMAUFPR: SISTEMA DE ABERTURA E CONTROLE DE INCIDENTES DE
MANUTENÇÃO PREDIAL

CURITIBA

2023

Este manual tem como objetivo apresentar o passo a passo de instalação do software desenvolvido para o projeto. Envolvendo inicialmente a apresentação do sistema web e após do sistema mobile.

Os arquivos compactados estão disponíveis neste repositório, na pasta 'codigo-fonte'.

1. Sistema web

Os arquivos compactados estão disponíveis no repositório https://github.com/vinisavitraz/arruma_ufpr.git.

Para a instalação do sistema web, será necessária uma máquina com os seguintes requisitos:

- Acesso à internet para clonar o repositório;
- Docker instalado;
- Se já possuir o PostgreSQL instalado, deve parar o serviço para que não haja conflito de porta;

Para iniciar o processo de instalação, deve-se clonar o repositório "https://github.com/vinisavitraz/arruma_ufpr.git". Após baixado o repositório, abrir o prompt de comando e navegar até a raiz do projeto com o comando "cd 'caminho do projeto'".

Seguindo próximo passo, entrar na raiz do projeto via prompt, executar o comando "docker-compose up" (FIGURA 1).

Esse comando vai criar os containers necessários para a aplicação funcionar, como container do servidor e do banco de dados PostgreSQL, junto com a base dos dados `arruma_ufpr`.

FIGURA 1 – COMANDO "DOCKER-COMPOSE UP"

```

route +2ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:38:53 PM LOG [RoutesResolver] DashboardIncidentController {/dashboard/i
incident}): +0ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:38:53 PM LOG [RouterExplorer] Mapped {/dashboard/incident, GET} route +
1ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:38:53 PM LOG [RouterExplorer] Mapped {/dashboard/incident/create, GET}
route +1ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:38:53 PM LOG [RouterExplorer] Mapped {/dashboard/incident/types, GET} r
oute +1ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:38:53 PM LOG [RouterExplorer] Mapped {/dashboard/incident/types/create,
GET} route +1ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:38:53 PM LOG [RoutesResolver] LocationController {/location}: +0ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:38:53 PM LOG [RouterExplorer] Mapped {/location, GET} route +1ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:38:53 PM LOG [RouterExplorer] Mapped {/location/:id, GET} route +1ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:38:53 PM LOG [RouterExplorer] Mapped {/location, POST} route +1ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:38:53 PM LOG [RoutesResolver] ItemController {/object}: +0ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:38:53 PM LOG [RoutesResolver] IncidentController {/incident}: +1ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:38:53 PM LOG [NestApplication] Nest application successfully started +1
76ms
  
```

FONTE: Os autores (2023).

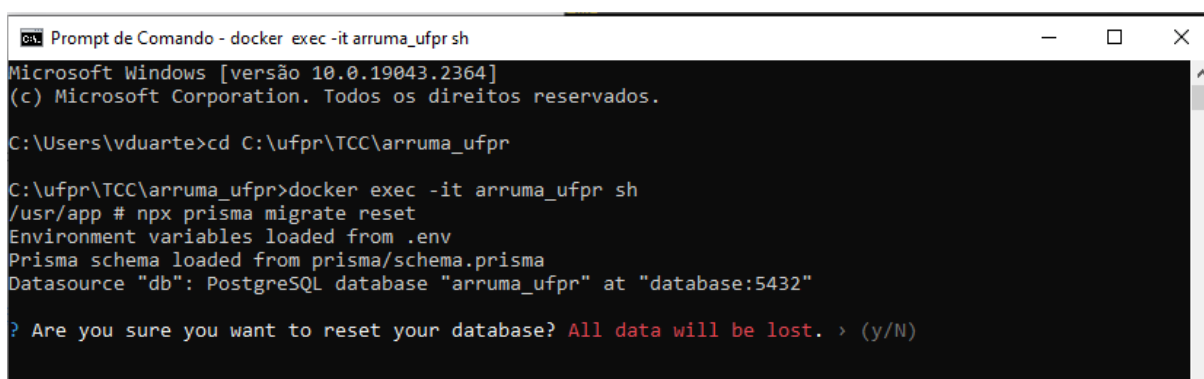
Assim que o processo de inicialização dos containers terminar, abrir um novo prompt de comando, acessar a pasta do projeto e executar o comando “docker exec -it arruma_ufpr sh”.

Esse comando vai acessar diretamente o prompt de comando do próprio container com o servidor.

Utilizando o prompt de comando do servidor, é necessário executar um novo comando “npx prisma migrate reset”.

Ao aparecer a pergunta “Are you sure you want to reset your database?” responder com a opção “y” (FIGURA 2).

FIGURA 2 – RESETAR BASE DE DADOS



```

Microsoft Windows [versão 10.0.19043.2364]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\vduarte>cd C:\ufpr\TCC\arruma_ufpr

C:\ufpr\TCC\arruma_ufpr>docker exec -it arruma_ufpr sh
/usr/app # npx prisma migrate reset
Environment variables loaded from .env
Prisma schema loaded from prisma/schema.prisma
Datasource "db": PostgreSQL database "arruma_ufpr" at "database:5432"

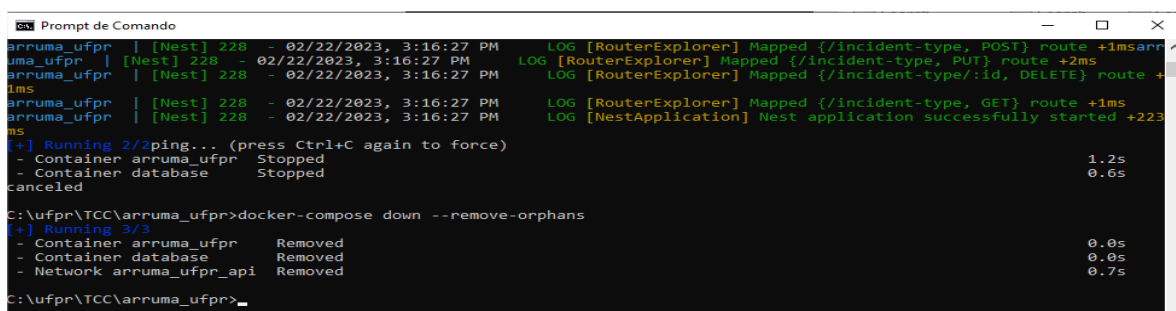
? Are you sure you want to reset your database? All data will be lost. > (y/N)
  
```

FONTE: Os autores (2023).

Esse comando vai criar as tabelas e dados iniciais na base de dados.

Agora, como último passo, é preciso reiniciar o docker utilizando o comando “docker-compose down --remove-orphans” (FIGURA 3) e na sequencia o “docker-compose up” (FIGURA 4) em um novo prompt de comando e o sistema vai estar configurado.

FIGURA 3 – PARAR O DOCKER



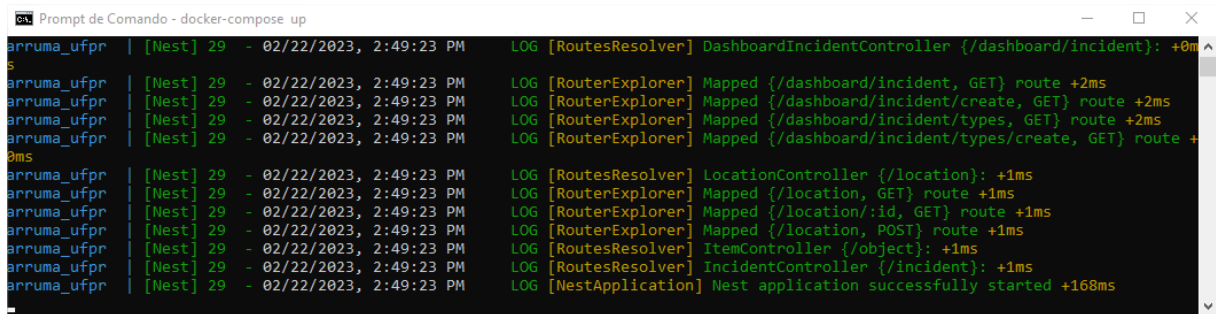
```

arruma_ufpr | [Nest] 228 - 02/22/2023, 3:16:27 PM LOG [RouterExplorer] Mapped {/incident-type, POST} route +1ms
arruma_ufpr | [Nest] 228 - 02/22/2023, 3:16:27 PM LOG [RouterExplorer] Mapped {/incident-type, PUT} route +2ms
arruma_ufpr | [Nest] 228 - 02/22/2023, 3:16:27 PM LOG [RouterExplorer] Mapped {/incident-type/:id, DELETE} route +1ms
arruma_ufpr | [Nest] 228 - 02/22/2023, 3:16:27 PM LOG [RouterExplorer] Mapped {/incident-type, GET} route +1ms
arruma_ufpr | [Nest] 228 - 02/22/2023, 3:16:27 PM LOG [NestApplication] Nest application successfully started +223ms
[+] Running 2/2ping... (press Ctrl+C again to force)
- Container arruma_ufpr Stopped 1.2s
- Container database Stopped 0.6s
canceled

C:\ufpr\TCC\arruma_ufpr>docker-compose down --remove-orphans
[+] Running 3/3
- Container arruma_ufpr Removed 0.0s
- Container database Removed 0.0s
- Network arruma_ufpr_api Removed 0.7s
C:\ufpr\TCC\arruma_ufpr>
  
```

FONTE: Os autores (2023).

FIGURA 4 – INICIAR O DOCKER



```
Prompt de Comando - docker-compose up
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:49:23 PM LOG [RoutesResolver] DashboardIncidentController {/dashboard/incident}: +0ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:49:23 PM LOG [RouterExplorer] Mapped {/dashboard/incident, GET} route +2ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:49:23 PM LOG [RouterExplorer] Mapped {/dashboard/incident/create, GET} route +2ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:49:23 PM LOG [RouterExplorer] Mapped {/dashboard/incident/types, GET} route +2ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:49:23 PM LOG [RouterExplorer] Mapped {/dashboard/incident/types/create, GET} route +
0ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:49:23 PM LOG [RoutesResolver] LocationController {/location}: +1ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:49:23 PM LOG [RouterExplorer] Mapped {/location, GET} route +1ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:49:23 PM LOG [RouterExplorer] Mapped {/location/:id, GET} route +1ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:49:23 PM LOG [RouterExplorer] Mapped {/location, POST} route +1ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:49:23 PM LOG [RoutesResolver] ItemController {/object}: +1ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:49:23 PM LOG [RoutesResolver] IncidentController {/incident}: +1ms
arruma_ufpr | [Nest] 29 - 02/22/2023, 2:49:23 PM LOG [NestApplication] Nest application successfully started +168ms
```

FONTE: Os autores (2023).

Agora o sistema vai estar disponível acessando a URL:
<http://localhost:3000/dashboard/login>

Para acessar o sistema, utilizar o usuário padrão de administrador E-mail:
admin@mail.com e Senha: 1234.

Com a aplicação rodando, a base de dados fica disponível usando as seguintes credenciais.

- Host: Localhost
- Port: 5432
- Database: arruma_ufpr
- User: postgres
- Senha: admin

2. Sistema mobile

Os arquivos compactados estão disponíveis no repositório
https://github.com/vinisavitraz/arruma_ufpr_app.git.

Para a instalação do sistema mobile, será necessária uma máquina com os seguintes requisitos:

- Acesso à internet para clonar o repositório;
- Android Studio instalado;
- Flutter instalado;
- Emulador ou dispositivo móvel;

- Para a versão iOS do app, o sistema operacional precisa do desenvolvedor precisa ser macOS;
- Para a versão Android do app, o sistema operacional do desenvolvedor pode ser Windows, Linux ou macOS;

Para iniciar o processo de instalação, é necessário clonar o repositório “https://github.com/vinisavitraz/arruma_ufpr_app.git”, após realizado o clone, abrir o projeto no Android Studio. Configurar o emulador ou conectar o dispositivo móvel via USB.

Antes de executar o sistema, deve ser alterado a variável “remoteHost” no arquivo “app_http_client.dart” com o IP da máquina que está executando o servidor web, portando, deve abrir o prompt de comando e executar o comando “ipconfig” no Windows ou “ifconfig” caso esteja usando MacOS/Linux, selecionar o IP e preencher a variável.

Abrir o prompt de comando e navegar até a raiz do projeto e executar o comando “flutter run” (FIGURA 5). Depois disso, o app será instalado no dispositivo móvel ou no emulador.

FIGURA 5 – COMANDO ‘FLUTTER RUN’

```
target file ios not found.
➔ arruma_ufpr_app git:(master) flutter run
Launching lib/main.dart on Vinicius in debug mode...
Automatically signing iOS for device deployment using specified development team in Xcode project: MWC4T3XW93
Running pod install... 1,887ms
Running Xcode build...
  └─Compiling, linking and signing... 6.6s
Xcode build done. 75.6s
Installing and launching... 22.8s
flutter: Active user info: admin@mine@mail.com
Syncing files to device Vinicius... 1,564ms

Flutter run key commands.
r Hot reload. 🔥🔥🔥
R Hot restart.
h List all available interactive commands.
d Detach (terminate "flutter run" but leave application running).
c Clear the screen
q Quit (terminate the application on the device).

👉 Running with sound null safety 👈

An Observatory debugger and profiler on Vinicius is available at: http://127.0.0.1:57847/ZBcenYXVPn8=/
The Flutter DevTools debugger and profiler on Vinicius is available at:
http://127.0.0.1:9100?uri=http://127.0.0.1:57847/ZBcenYXVPn8=/
```

FONTE: Os autores (2023).