

# Capítulo 3 : Resultado Final

---

## 3.1 Desenvolvimento

### 3.1.1 Express

Esta API foi formulada com recurso ao express, uma framework que atua com a funcionalidade de Web Server do Node.js e que visa adicionar novas características de forma simples, o que faz com que a organização de uma API seja facilitada relativamente aos middlewares e às rotas.

### 3.1.2 Docker

A API e o MySql são colocados em containers do Docker, para estes containers conseguirem comunicar entre si, foi criado um dockercompose que vai criar imagens e uma rede interna partilhada pelos dois container.

## 3.2 Instalação

*Para a instalação e compilação deste projeto é necessário:*

- Passo 1: Importar o ficheiro '**docker-compose.yaml**';
- Passo 2: Abrir a linha de comandos (CLI) e entrar no diretório onde se encontra o ficheiro importado anteriormente;;
- Passo 3: Executar o seguinte comando '**docker-compose up**';
- Passo 4: Abrir o Browser no URL '**http://localhost**'.

## 3.3 Instruções de Utilização

1. É possível o utilizador efetuar login ou visualização da API.
2. Se o utilizador não se autenticar, só pode efetuar os pedidos GET da API.
3. Se o utilizador autentica-se poderá efetuar todos os comandos.
4. Após efetuar o login, o utilizador pode ver as informações da conta do GitHub utilizada.

## 3.4 Detalhes de Implementação

*Objetivos cumpridos:*

- **database** - Criação da Schema e tabelas que serão usadas pelo container de MySQL;
- **setup** - Criação da Schema e tabelas que serão usadas pelo container de MySQL;
- **dockerfile** - Ficheiro dockerfile responsável pela configuração da imagem MySQL, definição de espaço de trabalho;
- **dockerfile** - Ficheiro responsável pela configuração da imagem MySQL;
- **docker-compose** - Ficheiro responsável pela ligação dos dois containers;
- **docker-compose** - Ficheiro responsável pela a obtenção dos dois containers, mais facilmente;