

Pontifícia Universidade Católica de Minas Gerais
Gerência de Configuração e Evolução de Software
Travel Friends

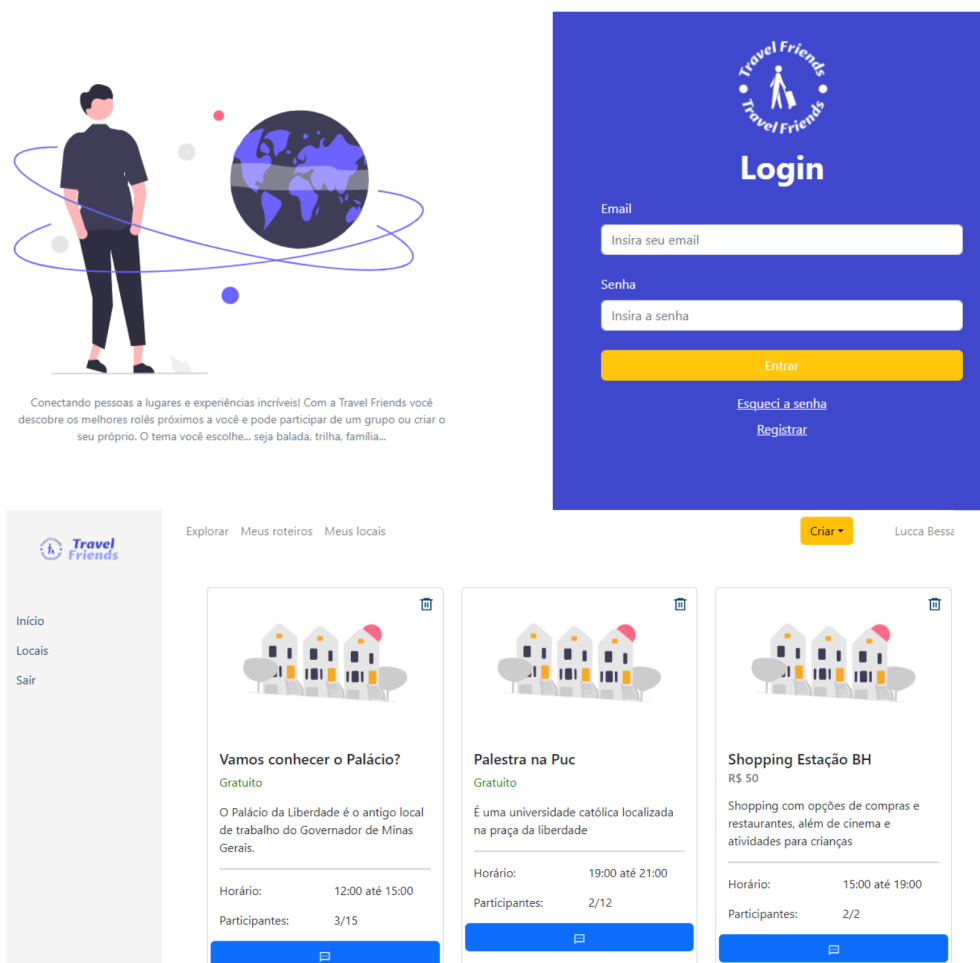
Belle Nerissa Aguiar Elizeu
Júlia Leite de Souza
Jully Ketely Alves da Silva
Lucca Vieira Paz de Bessa
Samuel Alves de Castro Baker

1 PROJETO

No atual cenário de pós-pandemia, pesquisas recentes apontaram que as pessoas estão planejando viagens em curto prazo. Entre os destinos que se destacam estão aqueles indicados por experiências de amigos e conhecidos, tanto nacionais quanto internacionais.

Nesse contexto de crescente mercado turístico, os viajantes estão em busca de praticidade e conforto, focando nisso o objetivo do projeto Travel Friends é de desenvolver uma aplicação móvel e Web capaz de fornecer aos seus usuários uma experiência única ao utilizar roteiros turísticos e realizar conexão com pessoas.

O objetivo geral do trabalho é apresentar uma aplicação que possa fomentar as atividades turísticas em diferentes localidades e conectar pessoas interessadas em trocar experiências. Para isso, os usuários vão poder criar encontros em pontos selecionados e disponibilizar para outras pessoas.



1.1 ESTRUTURAÇÃO DO PROJETO

```
travel-friends-web
src
├── @types
├── assets
│   ├── icons
│   └── images
├── components
│   ├── Button
│   ├── Card
│   ├── Divider
│   ├── FallbackWrapper
│   ├── InputTime
│   │   └── utils
│   ├── Navigation
│   └── Send
├── context
├── hooks
├── mock
├── pages
│   ├── Home
│   ├── Place
│   │   └── Form
│   ├── ResetPassword
│   ├── Roadmap
│   └── SignUp
├── PrivateRoute
├── services
│   ├── api
│   └── Requests
├── utils
├── index.css
├── index.tsx
└── routes.tsx
```

Figura 1 - Estrutura de documentos na aplicação *Front-End*



```
server
├── models
│   ├── RoadMap.js
│   └── User.js
├── routes
│   └── roadMapRoutes.js
├── .gitignore
├── Dockerfile
├── index.js
├── package-lock.json
├── package.json
└── README.md
```

Figura 2 - Estrutura de documentos na aplicação node API

2 REQUISITOS FUNCIONAIS

ID	Descrição	Prioridade
RF001	Após acessar a plataforma Web o usuário deve ser capaz de visualizar suas informações de perfil	Obrigatório
RF002	Após acessar a plataforma Web o usuário deve ser capaz de comentar os passeios a qual tem interesse	Obrigatório
RF003	Após acessar a plataforma Web o usuário e realizar cadastro, deve ler e confirmar que está em acordo com os termos de uso	Obrigatório
RF004	Após acessar a plataforma Web o usuário deve ser capaz de solicitar suporte	Desejável
RF005	Após acessar a plataforma Web o usuário deve ser capaz de visualizar imagens correspondentes ao tipo do local do passeio	Obrigatório

Tabela 1 - Requisitos funcionais levantados para aplicação

3 DISTRIBUIÇÃO DE TAREFAS

Sprint 1

Na *Sprint 1* foi realizado ajustes e melhorias no projeto que se encontra hospedado no Github, Passa-Palavra (<https://github.com/jullysilva/Passa-Palavra>). Como discutido em aula, as modificações foram adicionadas em suas respectivas *branches*, e consequentemente, mesclados a *branch* principal (*master*).

TAREFA	RESPONSÁVEL
Configuração do CI/CD	Jully Ketely
RF - Timer da partida	Samuel Baker
Validador de recursos	Belle Nerissa
Ajuste de layout #1	Júlia Souza
Ajuste de layout #2	Lucca Bessa

Tabela 2 - Funcionalidades implementadas na *Sprint 1* e respectivos responsáveis

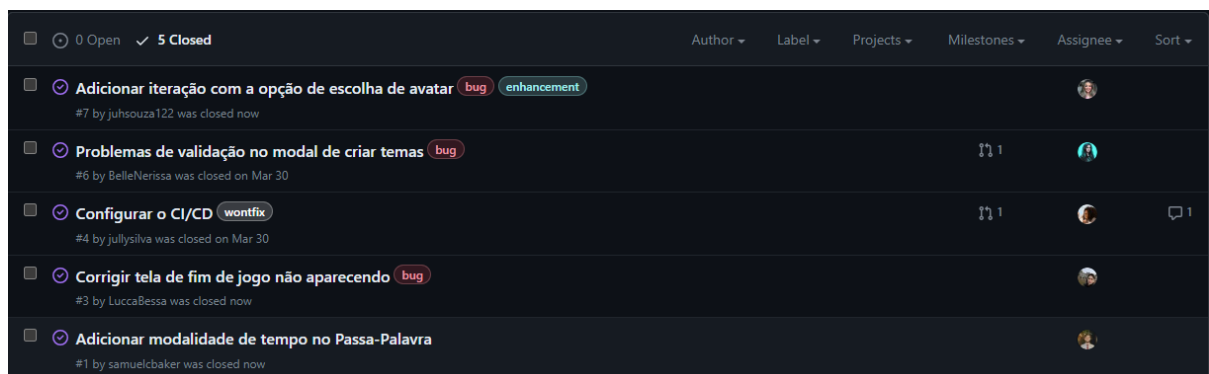


Figura 3 - Dashboard de tarefas do projeto para *Sprint 1*

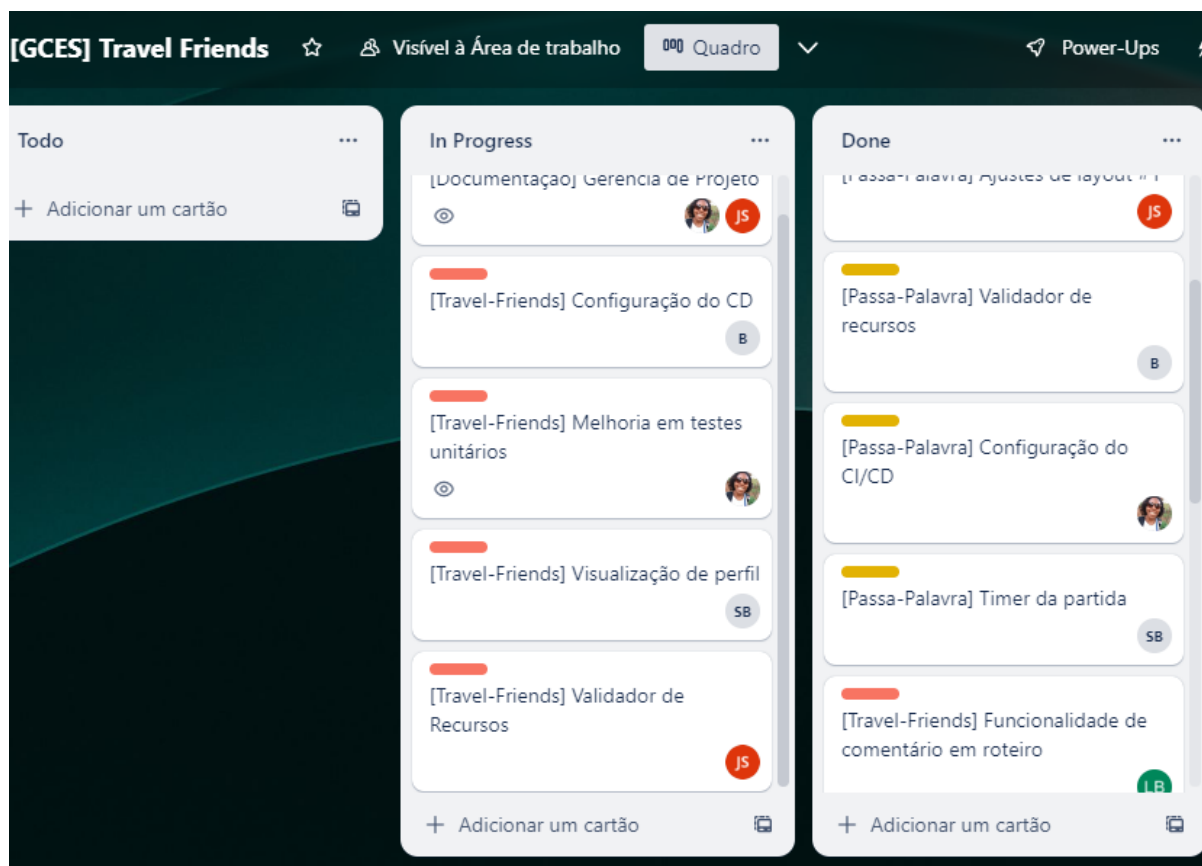


Figura 4 - Dashboard de tarefas do projeto para *Sprint 2:4*

Sprint 2

Na *Sprint 2* foram realizadas melhorias, publicação da API e configuração do CI, no projeto que se encontra hospedado no Github, Travel-Friends (<https://github.com/jullysilva/travel-friends>). Como discutido em aula, as modificações foram adicionadas em suas respectivas *branches*, e consequentemente, mesclados a *branch* principal (*master*).

TAREFA	RESPONSÁVEL
Configuração do CI	Samuel Baker
Melhoria de teste unitários	Samuel Baker
Publicação manual da API	Lucca Bessa

Tabela 3 - Funcionalidades implementadas na *Sprint 2* e respectivos responsáveis

Sprint 3

Na *Sprint 3* foi realizado a criação de imagens e containers no Travel-Friends (<https://github.com/jullysilva/travel-friends>). As modificações foram adicionadas em

suas respectivas *branches*, e consequentemente, mesclados a *branch* principal (*master*).

TAREFA	RESPONSÁVEL
Configuração do Docker (Containerização)	Jully Ketely

Tabela 4 - Funcionalidades implementadas na *Sprint 3* e respectivos responsáveis

Sprint 4

Na *Sprint 4* foram realizadas melhorias, publicação da API e configuração do CI, no projeto que se encontra hospedado no Github, Travel-Friends (<https://github.com/jullysilva/travel-friends>). Como discutido em aula, as modificações foram adicionadas em suas respectivas *branches*, e consequentemente, mesclados a *branch* principal (*master*).

TAREFA	RESPONSÁVEL
Configuração do CD	Samuel Baker
Melhoria de teste unitários + [RF005]: Visualização imagens dos locais	Belle Nerissa
[RF003]: Funcionalidade de comentário em roteiro	Lucca Bessa
Documentação e gerência do projeto	Jully Ketely
Resolução de Bugs	Lucca Bessa
Ajuste no validador de CPF	Júlia Souza
[RF003] Implementação de termos e condições de uso	Júlia Souza
[RF001] Visualização de informações do perfil	Samuel Baker

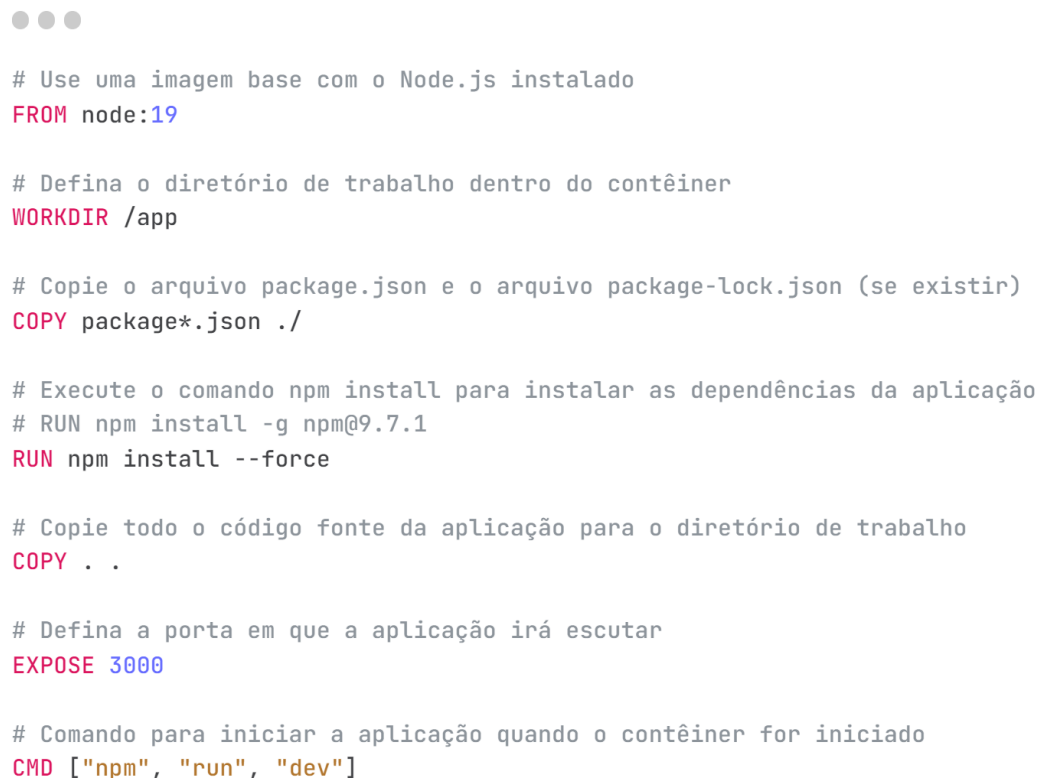
Tabela 4 - Funcionalidades implementadas na *Sprint 4* e respectivos responsáveis

4 PROCESSO DE DESENVOLVIMENTO

Na aplicação foi realizada a configuração das imagens Docker para aplicação *Front-end* em React TypeScript e *API* em NodeJS. Foi utilizado o Docker-Compose para orquestração de containers. Após isso, a criação do *Continuous Integration* (CI) para análise no *CircleCI* e publicação, utilizado o *Render*.

4.1 CONFIGURAÇÃO DO DOCKER

4.1.1 DOCKERFILE WEB

A screenshot of a code editor window showing a Dockerfile. The window has three gray circular window control buttons in the top-left corner. The code is a Dockerfile for a web application, with comments in gray and commands in red. The commands are: FROM node:19, WORKDIR /app, COPY package*.json ./, RUN npm install --force, COPY . ., EXPOSE 3000, and CMD ["npm", "run", "dev"].

```
# Use uma imagem base com o Node.js instalado
FROM node:19

# Defina o diretório de trabalho dentro do contêiner
WORKDIR /app

# Copie o arquivo package.json e o arquivo package-lock.json (se existir)
COPY package*.json ./

# Execute o comando npm install para instalar as dependências da aplicação
# RUN npm install -g npm@9.7.1
RUN npm install --force

# Copie todo o código fonte da aplicação para o diretório de trabalho
COPY . .

# Defina a porta em que a aplicação irá escutar
EXPOSE 3000

# Comando para iniciar a aplicação quando o contêiner for iniciado
CMD ["npm", "run", "dev"]
```

Figura 5 - Configuração do *Dockerfile* da aplicação web

4.1.2 DOCKERFILE API

```
# Use uma imagem base com o Node.js instalado
FROM node:19

# Defina o diretório de trabalho dentro do contêiner
WORKDIR /app

# Copie o arquivo package.json e o arquivo package-lock.json (se existir)
COPY package*.json ./

# Execute o comando npm install para instalar as dependências da aplicação
# RUN npm install -g npm@9.7.1
RUN npm install --force

# Copie todo o código fonte da aplicação para o diretório de trabalho
COPY . .

# Defina a porta em que a aplicação irá escutar
EXPOSE 8000

# Comando para iniciar a aplicação quando o contêiner for iniciado
CMD ["npm", "start"]
```

Figura 6 - Configuração do *Dockerfile* da aplicação node API

4.1.3 DOCKER-COMPOSE

```
version: '3'

services:
  frontend:
    image: frontend
    build: ./travel-friends-web
    ports:
      - 3000:3000
    depends_on:
      - node
    volumes:
      - ./travel-friends-web:/app/
      - /app/node_modules
  node:
    image: api
    build: ./server
    ports:
      - 8000:8000
    volumes:
      - ./server:/app/
      - /app/node_modules
```

Figura 7 - Configuração do *docker-compose* da aplicação

4.2 CI CircleCI



```
version: 2.1
orbs:
  node: circleci/node@5.1.0

jobs:
  test:
    working_directory: ~/travel-friends/travel-friends-web
    executor: node/default
    steps:
      - checkout:
          path: ~/travel-friends
      - node/install-packages:
          pkg-manager: yarn
      - run:
          command: yarn test
          name: Run tests

workflows:
  test_my_app:
    jobs:
      - test
```

Figura 8 - Configuração de CI

4.3 TESTES UNITÁRIOS

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
All files	59.3	75	50	59.3	
components/Button	100	100	100	100	
Button.tsx	100	100	100	100	
components/Divider	100	100	100	100	
Divider.styled.tsx	100	100	100	100	
mock	100	100	100	100	
index.js	100	100	100	100	
services/api	100	100	100	100	
index.js	100	100	100	100	
services/api/Requests	47.54	100	41.66	47.54	
login.ts	43.75	100	33.33	43.75	5-9,19,26-30
places.ts	53.57	100	60	53.57	9,18,25-30,35-40,48
roadmap.ts	41.17	100	25	41.17	8,15-20,24-26,30-32
utils	75	50	33.33	75	
generalUtils.ts	87.5	50	100	87.5	13
test-utils.tsx	50	100	0	50	6,16
Test Suites: 8 passed, 8 total					
Tests: 1 skipped, 17 passed, 18 total					
Snapshots: 0 total					
Time: 19.024 s					
Ran all test suites.					

Figura 9 - Coverage dos testes unitários na aplicação web

4.4 DEPLOY AUTOMATIZADO PELO RENDER

Ao final do desenvolvimento da aplicação realizamos o deploy do projeto pela plataforma render (<https://render.com/>) que é uma plataforma de hospedagem que permite realizar o deploy automatizado de projetos web por meio de uma pipeline própria. Todo commit realizado no repositório do git na branch especificada na configuração do render é automaticamente iniciada o processo de build e deploy para o ambiente hospedado do backend (<https://travel-friends-api.onrender.com>) e do frontend (<https://travel-friends-web.onrender.com/>).

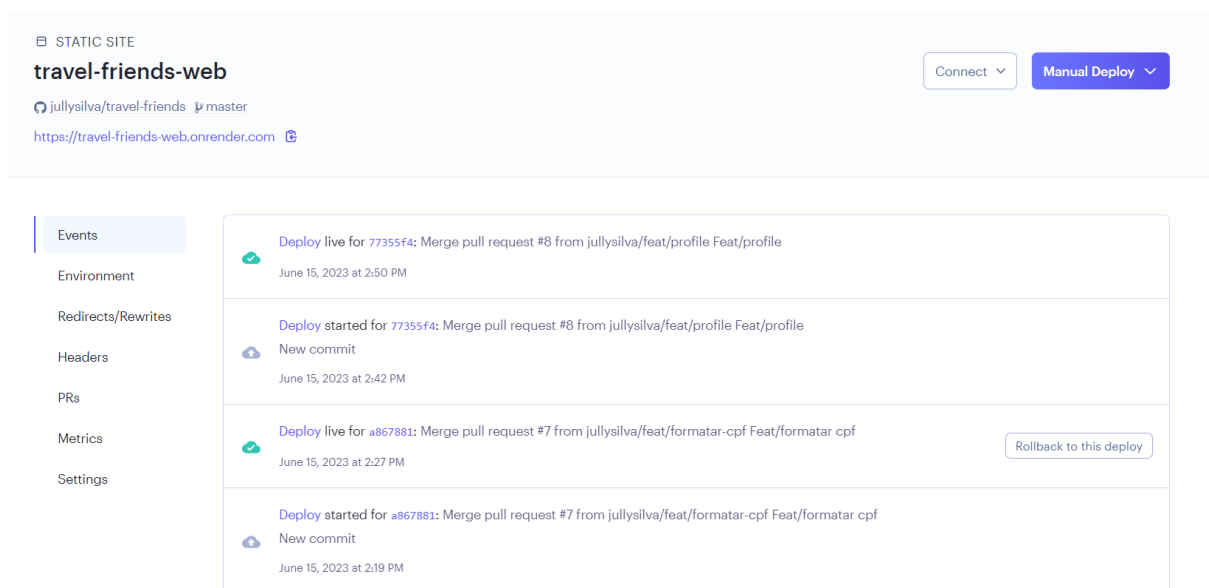


Figura 10 - Deploy da aplicação web no Render

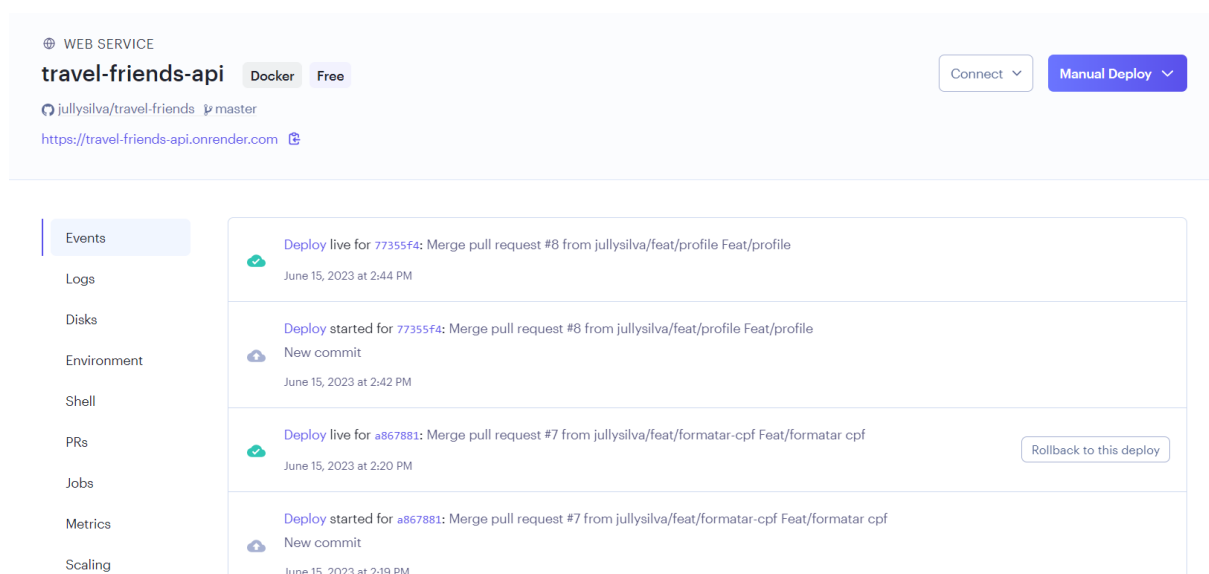


Figura 11 - Deploy da aplicação node API no Render

5 DESAFIOS E SOLUÇÕES

Desafios	Soluções
Executar o projeto no Docker	A configuração estava errada
Kubernetes	Não foi utilizado
Versão do NodeJS	Equalização da versão das máquinas
Aplicação com WebSocket inutilizado e gerando erros.	A funcionalidade foi desabilitada
Erros de CORS na aplicação backend	Adição da dependência “cors”

Tabela 5 - Desafios e soluções encontrados na implementação do projeto.