




ARQUITETURA
SERVELESS

SETTINGS API



SETTINGS (WHAT IS)

**SETTINGS REFERE-SE A UM CONJUNTO DE CONFIGURAÇÕES
INDIVIDUAIS PARA CADA USUÁRIO, COMO O NÚMERO DE LADOS
DE UM DADO**





SETTINGS

(PARA QUE SERVE)



**É RESPONSÁVEL POR GERENCIAR E FORNECER ACESSO AS
CONFIGURAÇÕES.**

FASTIFY

- **ALTA PERFORMANCE**
- **EXTENSÍVEL**
- **BASEADO EM ESQUEMA**
- **REGISTRO**
- **AMIGÁVEL AO DESENVOLVEDOR**
- **PRONTO PARA TYPESCRIPT**



FASTIFY É UMA ESTRUTURA WEB ALTAMENTE FOCADA EM FORNECER A MELHOR EXPERIÊNCIA AO DESENVOLVEDOR COM O MÍNIMO DE SOBRECARGA E UMA ARQUITETURA DE PLUGIN PODEROSA. É INSPIRADO EM HAPI E EXPRESS

PLUG-IN DE BANCO DE DADOS

```
import fp from 'fastify-plugin'

// the use of fastify-plugin is required to be able
// to export the decorators to the outer scope

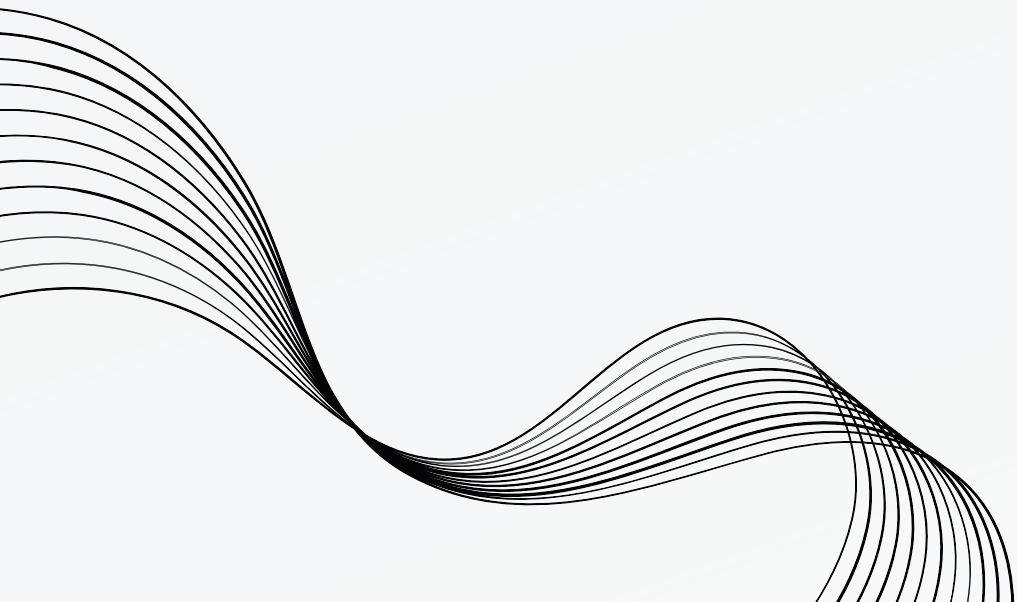
export default fp(async function (fastify, opts) {
  fastify.decorate('db', new MockDatabase());
});
```

```
class MockDatabase {
  constructor() {
    this.db = {};
  }

  async saveSettings(userId, settings) {
    await this.#delay();
    this.db[userId] = settings;
  }

  async getSettings(userId) {
    await this.#delay();
    return this.db[userId];
  }

  async #delay() {
    return new Promise(resolve => setTimeout(resolve, 10));
  }
}
```



CRIANDO AS ROTAS

ROTA PUT

```
fastify.put('/:userId', async function (request, reply) {  
  request.log.info(`Saving settings for user ${request.params.userId}`);  
  await fastify.db.saveSettings(request.params.userId, request.body);  
  reply.code(204);  
});
```

ROTA GET

```
fastify.get('/:userId', async function (request, reply) {  
  request.log.info(`Retrieving settings for user ${request.params.userId}`);  
  const settings = await fastify.db.getSettings(request.params.userId);  
  if (settings) {  
    return settings;  
  }  
  return { sides: 6 };  
});
```


ACESSANDO AS ROTAS

OS PASSOS INCLUEM O USO DE UM SERVIDOR INICIADO COM O COMANDO

```
npm start --workspace=settings-api
```

E A VISUALIZAÇÃO DE LOGS JSON. PARA TORNAR OS LOGS MAIS LEGÍVEIS, PODESSE USAR FERRAMENTA "PINO-PRETTY"

```
@sinedied → .../nodejs-microservices-template (main) $ npm start --workspace=settings-api | pino-pretty  
  
> settings-api@1.0.0 start  
> fastify start -l info app.js -a 0.0.0.0 -p 4001  
  
[09:55:46.100] INFO (31396): Server listening at http://0.0.0.0:4001
```

DOCKERFILE

CRIAÇÃO DE UM DOCKERFILE PARA CONTEINERIZAR UMA API DE CONFIGURAÇÕES

```
# syntax=docker/dockerfile:1
FROM node:18-alpine
ENV NODE_ENV=production

WORKDIR /app
COPY ./package*.json ./
COPY ./packages/settings-api ./packages/settings-api
RUN npm ci --omit=dev --workspace=settings-api --cache /tmp/empty-cache
EXPOSE 4001
CMD [ "npm", "start", "--workspace=settings-api" ]
```


RODANDO DOCKER

CONSTRUÇÃO DA IMAGEM DOCKER

```
cd packages/settings-api
```

```
docker build --tag settings-api --file ./Dockerfile ../..
```

EXECUÇÃO DA IMAGEM DOCKER

```
docker run --rm --publish 4001:4001 settings-api
```

ADIÇÃO DE SCRIPTS AO ARQUIVO PACKAGE.JSON

```
"scripts": {  
  "test": "tap \"test/**/*.test.js\"",  
  "start": "fastify start -l info app.js -a 0.0.0.0 -p 4001",  
  "dev": "fastify start -w -l info -P app.js -p 4001",  
  "docker:build": "docker build --tag settings-api --file ./Dockerfile ../..",  
  "docker:run": "docker run --rm --publish 4001:4001 settings-api"  
},
```

REFERÊNCIAS?

