Bufete Lionel Hutz y Asociados

Descripción General

El sistema del Bufete Lionel Hutz y Asociados se compone de tres capas principales: la interfaz de usuario (frontend), la lógica del negocio (backend) y la base de datos. Cada una de estas capas está contenida en un servicio Docker independiente, lo que facilita la administración, el despliegue y la escalabilidad del sistema.

Estructura del Proyecto

El proyecto está organizado en los siguientes directorios y archivos clave:

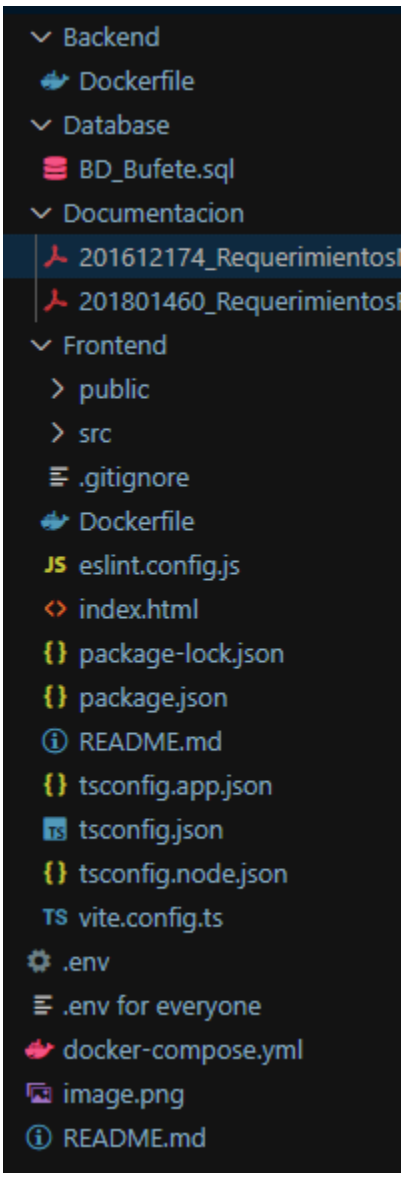
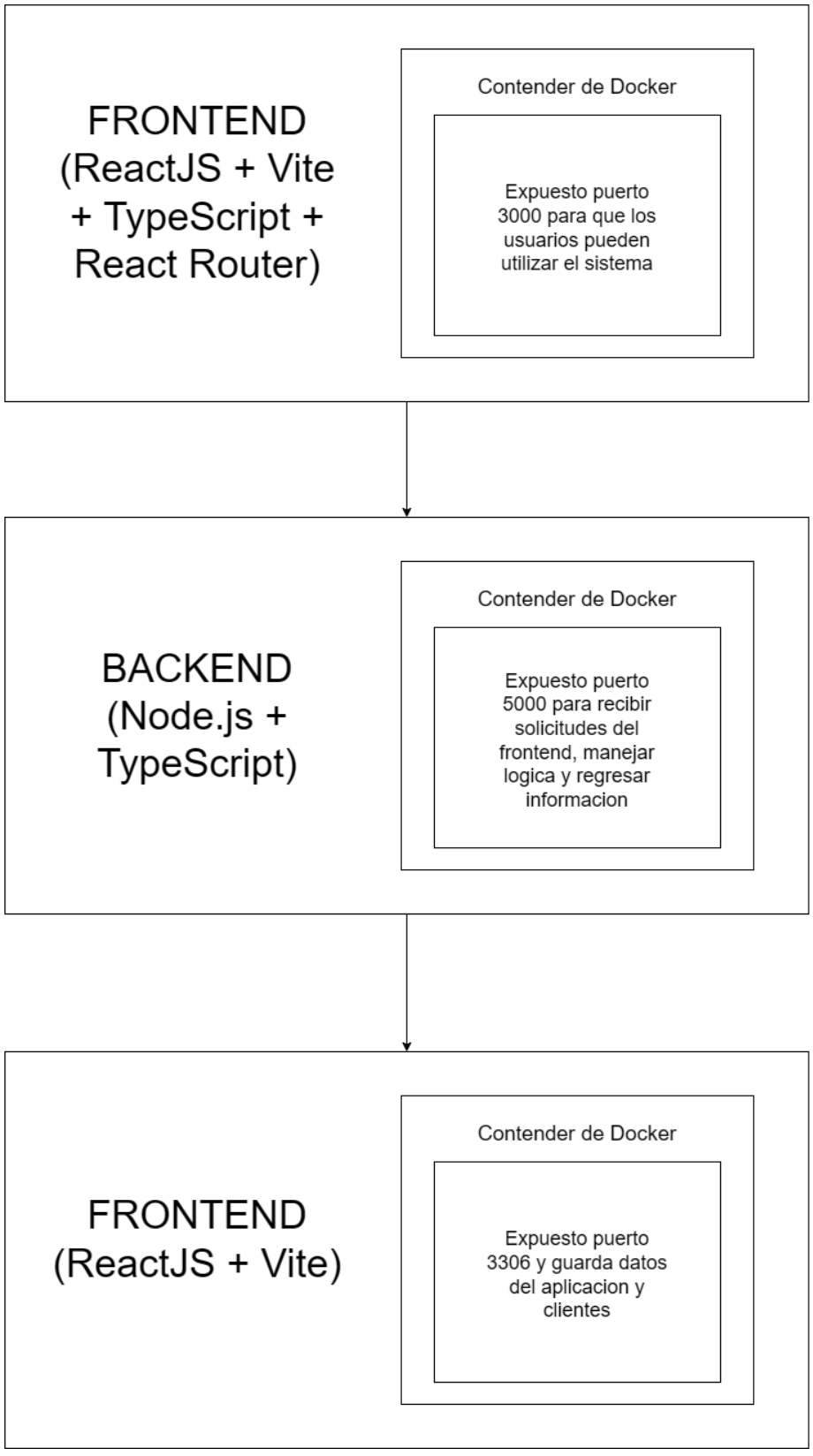


Diagrama de Arquitectura



Descripción de Componentes

1. **Frontend (Interfaz de Usuario)**
    - **Tecnologías**: React, Vite, TypeScript, React Router
    - **Descripción**: La capa de frontend es responsable de la interacción del usuario. Utiliza React para construir una interfaz de usuario dinámica, Vite como herramienta de construcción y servidor de desarrollo, y React Router para la gestión de rutas y navegación dentro de la aplicación.

Dockerfile:

```JavaScript
FROM node:18
WORKDIR /app
COPY package*.json ./
RUN npm install
COPY . .
RUN npm run build
EXPOSE 3000
RUN npm install -g serve
CMD ["serve", "-s", "dist"]
```

2. **Backend (Lógica del Negocio)**
    - **Tecnologías**: Node.js, TypeScript, Express
    - **Descripción**: La capa de backend maneja la lógica del negocio y sirve como intermediario entre el frontend y la base de datos. Utiliza Express para definir las rutas y manejar las solicitudes HTTP.

Dockerfile:

```JavaScript
FROM node:18
WORKDIR /app
COPY package*.json ./
RUN npm install
COPY . .
RUN npm run build
EXPOSE 5000
CMD ["npm", "run", "start"]
```

3. **Base de Datos (Persistencia)**
    - **Tecnologías**: MySQL
    - **Descripción**: La capa de base de datos se encarga de almacenar y gestionar los datos del sistema. Utiliza MySQL para proporcionar un almacenamiento relacional robusto y eficiente.

Configuración en docker-compose.yml:

```javascript
db:
  image: mysql:latest
  ports:
    - "3306:3306"
  environment:
    - MYSQL_ROOT_PASSWORD=${MYSQL_ROOT_PASSWORD}
    - MYSQL_DATABASE=${MYSQL_DATABASE}
  volumes:
    - db_data:/var/lib/mysql
    - ./Database/BD_Bufete.sql:/docker-entrypoint-initdb.d/init.sql
```

Integración y Despliegue

El `docker-compose.yml` orquesta los tres servicios:

```javascript
version: '3.9'

services:
  frontend:
    build:
      context: ./frontend
    ports:
      - "3000:3000"
    depends_on:
      - backend

  backend:
```

```
    build:
      context: ./backend
    ports:
      - "5000:5000"
    environment:
      - DB_HOST=db
      - DB_USER=root
      - DB_PASSWORD=${MYSQL_ROOT_PASSWORD}
      - DB_NAME=${MYSQL_DATABASE}
    depends_on:
      - db

  db:
    image: mysql:latest
    ports:
      - "3306:3306"
    environment:
      - MYSQL_ROOT_PASSWORD=${MYSQL_ROOT_PASSWORD}
      - MYSQL_DATABASE=${MYSQL_DATABASE}
    volumes:
      - db_data:/var/lib/mysql
      - ./Database/BD_Bufete.sql:/docker-entrypoint-initdb.d/init.sql

volumes:
  db_data:

networks:
  default:
    external:
      name: AyD2-Network
```

Pasos para el Despliegue

1. **Crear la Red Externa**

```
JavaScript
docker network create AyD2-Network
```

2. **Construir y Ejecutar los Servicios**

```
JavaScript
docker-compose up --build
```

3. **Acceder al Sistema**
    a. **Frontend**: Navegar a `http://localhost:3000`.
    b. **Backend**: La API está disponible en `http://localhost:5000`.