Cátedra: Implantación de Sistemas

DOCUMENTO TÉCNICO – LABORATORIO

Proyecto: TODO APP – Sistema de Gestión de Tareas

Docente: MARLIN YAJAIRA MOZ MARIN

Autores:

KERIM SELIM MELHADO HANDAL

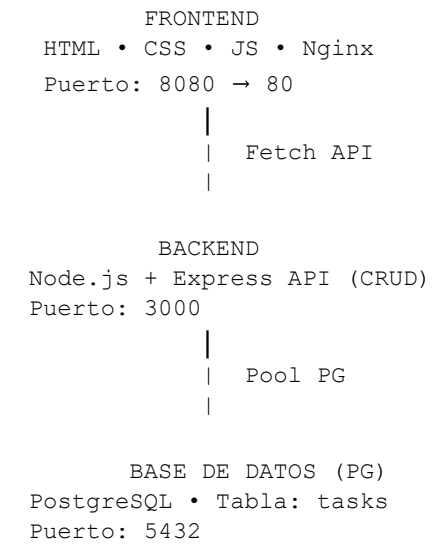
ALFONSO ANTONIO CORTEZ AGUILAR

Grupo: 02

Fecha: 15 de octubre de 2025

1. ARQUITECTURA DEL SISTEMA

El sistema TODO APP se compone de tres contenedores principales: el frontend, el backend y la base de datos. Cada uno cumple un rol específico dentro de la arquitectura cliente-servidor. El frontend provee la interfaz visual construida en HTML, CSS y JavaScript; el backend procesa las solicitudes mediante Express y Node.js; y la base de datos, PostgreSQL, almacena de forma persistente las tareas creadas por el usuario.



El flujo de datos se inicia cuando el usuario interactúa con la interfaz. Las peticiones HTTP son enviadas al backend, que procesa la solicitud, realiza operaciones sobre la base de datos y devuelve una respuesta JSON al cliente. Esta comunicación está contenida dentro de la red interna de Docker Compose, garantizando aislamiento y seguridad. (Ver Figura 1: Diagrama general de arquitectura).

2. DOCKERFILES

Se desarrollaron dos Dockerfiles: uno para el backend (Node.js + Express) y otro para el frontend (Nginx). A continuación, se muestra el contenido de cada archivo y las decisiones técnicas implementadas.

Backend:

```
FROM node:18-alpine
WORKDIR /app
COPY package*.json ./
RUN npm ci --omit=dev
COPY src ./src
EXPOSE 3000
CMD ["node", "src/server.js"]
```

Explicación: se utiliza una imagen base liviana (“node:18-alpine”) para optimizar el tamaño. Se define un directorio de trabajo y se copian primero los archivos de dependencias para aprovechar la caché de Docker. El comando “npm ci --omit=dev” instala las dependencias de producción, excluyendo las de desarrollo. Finalmente, se expone el puerto 3000 y se define el comando de ejecución principal del servidor.

Frontend:

```
FROM nginx:alpine
COPY nginx.conf /etc/nginx/conf.d/default.conf
COPY . /usr/share/nginx/html
EXPOSE 80
```

Explicación: se parte de la imagen “nginx:alpine”, optimizada para servir archivos estáticos. Se copia la configuración personalizada hacia la carpeta de Nginx y se transfieren los archivos de la interfaz al directorio raíz del servidor. El puerto 80 se expone para permitir el acceso HTTP. (Ver Figura 2: Construcción exitosa de las imágenes Docker).

3. DOCKER COMPOSE

El archivo “docker-compose.yml” define la configuración de los tres servicios del sistema: db, backend y frontend. Cada servicio especifica su imagen, variables de entorno, puertos y dependencias. Se utiliza un volumen persistente llamado “pgdata” para conservar los datos de la base de datos. Además, se define un healthcheck para verificar que PostgreSQL esté listo antes de iniciar el backend.

```
version: "3.8"
services:
db:
    image: postgres:15-alpine
environment:
POSTGRES_USER: appuser
      POSTGRES_PASSWORD: secret       POSTGRES_DB: appdb
ports: ["5432:5432"]     volumes:       -
pgdata:/var/lib/postgresql/data     healthcheck:
test: ["CMD-SHELL", "pg_isready -U appuser -d appdb"]
interval: 5s       retries: 10

  backend:
    build: ./backend
ports: ["3000:3000"]
environment:       PORT:
3000
      DB_HOST: db
      DB_USER: appuser
      DB_PASSWORD: secret
DB_NAME: appdb
depends_on:       db:
```

```
        condition: service_healthy

  frontend:
    build: ./frontend
ports: ["8080:80"]
depends_on:       -
backend

volumes:
pgdata:
```

Justificación: el servicio “db” contiene el healthcheck que asegura su disponibilidad antes de que el backend se inicie. El volumen “pgdata” mantiene los datos de forma persistente. Los contenedores comparten una red interna generada por Compose. (Ver Figura 3: Ejecución de docker-compose up -d mostrando los tres servicios activos).

4. ESTRATEGIA GIT

Se implementó un modelo de branching basado en Git Flow simplificado. La rama principal “main” almacena el código estable, mientras que “feature/backend” y “feature/frontend” se emplearon para el desarrollo modular de cada componente. Cada commit sigue la convención de Conventional Commits, lo que facilita la lectura y el control del historial.

```
main
  feature/backend
       feat: implementa API CRUD
       fix: corrige conexión DB
       docs: agrega README
  feature/frontend
        feat: crea interfaz HTML/CSS
        style: mejora diseño visual
        fix: ajuste en rutas JS
```

Durante la integración de las ramas se presentaron pequeños conflictos en el archivo README.md, los cuales fueron resueltos manualmente utilizando la herramienta de merge de Git.

5. VOLÚMENES Y PERSISTENCIA

Para garantizar la persistencia de los datos se empleó un volumen llamado “pgdata” asociado al servicio de base de datos. Este volumen almacena permanentemente los registros de las tareas, evitando la pérdida de información tras detener o reiniciar los contenedores.

Comandos utilizados para la inspección de volúmenes:

```
docker volume ls
docker volume inspect todo-app_pgdata
```

Durante las pruebas se comprobó la persistencia creando nuevas tareas, deteniendo los contenedores con “docker-compose down” y volviendo a levantarlos. Las tareas se conservaron correctamente.

6. PRUEBAS Y VALIDACIÓN

Las pruebas se realizaron ejecutando el comando “docker-compose ps”, que confirmó el estado “Up” de los tres servicios. Posteriormente, se accedió a la aplicación mediante el navegador en la dirección http://localhost:8080, verificando que las operaciones CRUD funcionaran correctamente.

También se analizaron los logs de los servicios para validar la correcta conexión con la base de datos y el inicio exitoso del servidor. A continuación, se muestran los resultados más relevantes:

```
docker-compose ps NAME                COMMAND                  STATUS
PORTS todo-db-1           "docker-entrypoint.s…"   Up 5 seconds    5432/tcp todo-
backend-1      "docker-entrypoint.s…"   Up 4 seconds    0.0.0.0:3000->3000/tcp
todo-frontend-1     "/docker-entrypoint.…"   Up 4 seconds    0.0.0.0:8080->80/tcp
```

Creación del Backend

```
aguilar@DESKTOP-769DMD4:~/todo-app/backend$ ls -la
find . -type f
total 24
drwxr-xr-x 3 aguilar aguilar 4096 Oct 14 22:33 .
drwxr-xr-x 6 aguilar aguilar 4096 Oct 14 22:33 ..
-rw-r--r-- 1 aguilar aguilar   58 Oct 14 18:21 .dockerignore
-rw-r--r-- 1 aguilar aguilar  324 Oct 14 18:22 Dockerfile
-rw-r--r-- 1 aguilar aguilar  414 Oct 14 22:32 package.json
drwxr-xr-x 2 aguilar aguilar 4096 Oct 14 22:36 src
./package.json
./.dockerignore
./src/index.js
./Dockerfile
aguilar@DESKTOP-769DMD4:~/todo-app/backend$ cd ~/todo-app
aguilar@DESKTOP-769DMD4:~/todo-app$ git status
On branch feature/backend
Your branch is up to date with 'origin/feature/backend'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        backend/

nothing added to commit but untracked files present (use "git add" to track)
aguilar@DESKTOP-769DMD4:~/todo-app$ git add .
aguilar@DESKTOP-769DMD4:~/todo-app$ git status
On branch feature/backend
Your branch is up to date with 'origin/feature/backend'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   backend/Dockerfile
        new file:   backend/package.json
        new file:   backend/src/index.js

aguilar@DESKTOP-769DMD4:~/todo-app$ git commit -m "feat: implementación completa del backend con puntos finales de API y configuración de la base de datos"
[feature/backend 7036be0] feat: implementación completa del backend con puntos finales de API y configuración de la base de datos
 3 files changed, 160 insertions(+)
 create mode 100644 backend/Dockerfile
 create mode 100644 backend/package.json
 create mode 100644 backend/src/index.js
```

```
aguilar@DESKTOP-769DMD4:~/todo-app$ git commit -m "feat: implementación completa del backend con puntos finales de API y configuración de la base de datos"
[feature/backend 7036be0] feat: implementación completa del backend con puntos finales de API y configuración de la base de datos
 3 files changed, 160 insertions(+)
 create mode 100644 backend/Dockerfile
 create mode 100644 backend/package.json
 create mode 100644 backend/src/index.js
aguilar@DESKTOP-769DMD4:~/todo-app$ git push origin feature/backend
Username for 'https://github.com': Aguilar0023
Password for 'https://Aguilar0023@github.com':
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 4 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (7/7), 2.11 KiB | 432.00 KiB/s, done.
Total 7 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Aguilar0023/todo-app.git
   4e4b499..7036be0  feature/backend -> feature/backend
aguilar@DESKTOP-769DMD4:~/todo-app$ nano docker-compose.yml
aguilar@DESKTOP-769DMD4:~/todo-app$ ls -la docker-compose.yml
-rw-r--r-- 1 aguilar aguilar 894 Oct 14 22:51 docker-compose.yml
```

Los logs mostraron la creación exitosa de la tabla “tasks” y la inicialización de Express en el puerto 3000. En conjunto, los resultados confirmaron la correcta implementación, despliegue y persistencia del sistema.

```
aguilar@DESKTOP-769DMD4:~/todo-app$ find frontend/ -type f
frontend/index.html
frontend/nginx.conf
frontend/styles.css
frontend/app.js
frontend/Dockerfile
aguilar@DESKTOP-769DMD4:~/todo-app$ git log --oneline --graph
* 1e54cb0 (HEAD -> feature/frontend, origin/feature/frontend) feat(frontend): base UI con HTML/CSS/JS y nginx.conf + Dockerfile
* 4e4b499 chore: initial project structure with gitignore
aguilar@DESKTOP-769DMD4:~/todo-app$ git show --name-only HEAD
commit 1e54cb0501a65f26cecab3047883643adda6a136 (HEAD -> feature/frontend, origin/feature/frontend)
Author: KerimHandal2 <kerim.handal2@gmail.com>
Date:   Wed Oct 15 10:58:24 2025 -0600

    feat(frontend): base UI con HTML/CSS/JS y nginx.conf + Dockerfile

frontend/Dockerfile
frontend/app.js
frontend/index.html
frontend/nginx.conf
frontend/styles.css
aguilar@DESKTOP-769DMD4:~/todo-app$ ls -la frontend/
total 28
drwxr-xr-x 2 aguilar aguilar 4096 Oct 15 19:11 .
drwxr-xr-x 6 aguilar aguilar 4096 Oct 15 19:09 ..
-rw-r--r-- 1 aguilar aguilar  104 Oct 15 19:11 Dockerfile
-rw-r--r-- 1 aguilar aguilar 1304 Oct 15 19:11 app.js
-rw-r--r-- 1 aguilar aguilar  487 Oct 15 19:11 index.html
-rw-r--r-- 1 aguilar aguilar  158 Oct 15 19:11 nginx.conf
-rw-r--r-- 1 aguilar aguilar  747 Oct 15 19:11 styles.css
```