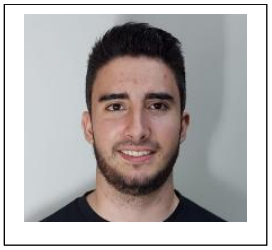


MÓDULO PROYECTO INTERMODULAR

Ciclo Superior Desarrollo de Aplicaciones Web

Departamento: Informática y Comunicaciones

IES “María Moliner”

Curso: 2025/2026

Grupo: S2I

Proyecto: MrFinance

Carlos Javier Pilar Sanz

Email: carlosj.pilsan.1@educa.jcyl.es

Tutor individual: María José

Tutor colectivo: Enrique Carballo Albarrán

Fecha de presentación: 4 Junio 2026

CC BY-NC-SA 4.0

Este trabajo se distribuye bajo licencia Creative Commons Atribución-NoComercial-CompartirIgual 4.0 Internacional.

" " " "

" " " "

"

" " " " " " " " " " " " " " "
" " " " " " " " " " "
" " " " " " " ""

" " " " " " " " " " " " "
" " " " " " " " " " "
" " " " " " " " " " " "

" " " " " " " " " " " " "
" " " " " " "

" " " " " " " " " " " "
" " " " " " " "

" "

"

Doy gracias por el apoyo en el desarrollo de este proyecto a:

" " " " "

" " " " " " **"María José"**

" " " " " " " " " " " "
" " " " "

Iván Santos Diez

Diego de la Esperanza

Ismael Monjas

Carlos Mesonero

Hugo San Juan Morales

Javier Bartolome

Ashkan Seminov Asimov

" " " " " " " " " " " " " " "

" " " " " " " "

" "

" " "

" " " " " " " " " " "
" " " " " " " " " " " " "
" " " " " " " " " " "
" " " " " " " " " " "
" " " " " " " " " " " "

" " " " " " " " " " "
" " " " " " " " " "
" " " " " " " " " " " "
" " " " " " " " " "
" " " " "

" " " " " " " " " " " " " " "
" " " " " " " " " " " " " " "
" " " " " " " " " " " " "
" " " " " " " " " "
" " " " " " " " " " " " "
" " " " "

" " " " " " " " " " " " " "
" " " " " " " " " " "
" " " " " " " " " " " "
" ' " "

" " " " " " " " " " "
" " " " " " " " " " " " " "
" " " " " " " " " " " " " "
"'" "

" " "

" " " "

"

" " 2

" " " 2

" 3

" " 4

" " 5

" " 9

9

" " 10

" " " " 10

" " " " 10

" " " 12

" " 12

" " " 13

" " " 15

" " 15

" " 16

" " " " " 16

" " 18

" " " 18

" " " 18

" 19

" " 19

" " 19

" " 29

" " " " 29

" " 29

" " " " 29

" 31

" " 31

" 31

" " 31

" " 31
" " 31
" " " " " 32
" 32
" " " " 32
" " 33
" " " " " 33
" 33
34
Como se a orientado finalmente la arquitectura en la versión de Docker. 36
37
" " 37
" " " " " " " 39
40
" " 40
Cuestiones de implementación y codificación reseñables 42
6.5 Gestión de y versionado 49
6.6 Pruebas realizadas 51
7. Manuales de usuario 54
7.1 Manual de usuario y configuración 54
7.2 Manual de instalación y configuración. 54
8. Posibles ampliaciones 56
9. Conclusiones y valoraciones 57
10. Bibliografía /Webgrafía 58
11. Anexos 59
"
" "

"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
" "

" " " "

"

" "

"

" " " " " " " " " " " " " "
"

" " " " " " " " " " " " " " "
" " " " " " " " " "
" " " " " " " " " " " " " "
" " " " " " " " "

" " " " " " " " " " " " "
" " " " " " " " " " " "

" " " " " " " " " " " " " "
" " " " " " " "

"

¿Cuál es el problema real?

" una visión clara de su situación económica" " gastos se dispersan"

- " **Falta de visibilidad** "
- " **Toma de decisiones activa** "
- " **Dispersión de información** "

Subproblema	Consecuencia actual	Solución que aporta MrFinance
Registro manual y disperso de gastos/ingresos		
Facturas en papel o perdidas		
Sin estadísticas automáticas		
Incapacidad de anticipar gastos futuros		
Acceso no seguro a datos sensibles		

Lo que SÍ resuelve:"

- " " " " "

Lo que NO aborda"*(Posibles Mejoras a futuro en producción):*"

- " " " " " " " "
- " " " " " " "
- " " "
- " " " " " "

Justificación técnica

" " " " " full-stack" " conocimientos de ciberseguridad" "

- **" " " " " " "base de datos + backend con autenticación"**
- **" " " " " "segura" "resistente a ataques "**
- **" " " " " " "frontend dinámico con gráficos"**
- **" " " " " " " "gestión de ficheros en servidor"**
- **" " " " " " diseño responsive + despliegue en cloud."**

  "

Resumen de la propuesta:

MrFinance" " " " " " " " " " " " " " " " " " " "

"

"

"

"

"

"

"

"

" " " " "

" " " " "

" " " " "

"

" " " " " " " " " " "
" " " " " " " " " " " " " "
" " " " " " " " " ""

"

- " **"capítulo 0"** " " " " " " "
" " " " " " " " " " "
" " " "
"
- " **"capítulo 1"** " " " " " " " " "
" " " " " " " " " " "
" " " " " " " " " "
" " " " " " " "

"

- " **"capítulo 2"** " " " " " " " " "
" " " " " "

"

- " **"capítulo 3"** " " " " " " " "
" " " " " " " " "
" " " " " " " " " " "

"

- " **"capítulo 4"** " " " " " " " " " " "
" " " " " " " " " " "
" " " " " " " " " " " " "
" " " " " " " " " " "
" " " "

"

- " **"capítulo 5"** " " " " " " " "
" " " " " " " " " " " "
" " " "

"

- " **"capítulo 6"** " " " " " " " " " "
" " " " " " " " "
" " " " " " " " " " " " " " "
" " " " " " " " " " "

" " " " " " " " " " " " "
" " " " " "

"

- " **"capítulo 7"** "

"

- " **"capítulo 8"** "

"

- " **"capítulo 9"** " " " " " " " " " " " " " "

"

- " " **"capítulo 10"** " " " " " " " **"capítulo 11"** "

" "

" " " " "

"

" " "

"

" " " " "

" " " "

" " " " " " " "Scrum "

" " " " " " "

" " " " " " " "Docker.

"

" " "git" "github.

"

"

" " " " " " " " " "nginx "

certbot "lets"encrypt "freemyip" "wireguard "

" "

"

" " " "

" " " " "

" " " " " "una aplicación web de gestión de ahorros personal " " "interfaz moderna y atractiva " " " "

" " " "

"

```yaml
Run All Services
services:

  # ────────────────────────────────────────────────
  # SERVICIO 1: Node.js (node:20-alpine)
  # ────────────────────────────────────────────────
  Run Service
  app:
    #build . apra usar el Dockerfile
    build: .
    container_name: nodejs_app
    restart: no
    ports:
      - "3000:3000"
    volumes:
      - ./app:/app
    environment:
      - NODE_ENV=development
      - DB_HOST=db
      - DB_PORT=3306
      - DB_USER=user
      - DB_PASSWORD=user
      - DB_NAME=appdb
    depends_on:
      db:
        condition: service_healthy

  # ────────────────────────────────────────────────
  # SERVICIO 2: Base de datos MySQL
  # ────────────────────────────────────────────────
  Run Service
  db:
    # Usa la imagen oficial de MySQL 8.0 directamente desde Docker Hub.
    image: mysql:8.0
    #nombre del contenedor
    container_name: mysql_db
    #restart no apra que no se ejecute todo el rato
    restart: no
    # Expone el puerto de MySQL a tu máquina local.
    # Útil para conectarte con clientes como workbench
    ports:
      - "3306:3306"

    # Variables de configuración de MySQL.
    environment:
      - MYSQL_ROOT_PASSWORD=root        # Contraseña del usuario root (admin total)
      - MYSQL_DATABASE=appdb            # Crea esta base de datos automáticamente
      - MYSQL_USER=user                 # Crea este usuario con acceso a appdb
      - MYSQL_PASSWORD=user             # Contraseña del usuario appuser

    volumes:
      # Volumen named: guarda los datos de MySQL en un volumen gestionado
      # por Docker. Los datos persisten aunque pares o borres el contenedor.
      - mysql_data:/var/lib/mysql
```

"

```yaml
      # Archivo SQL opcional de inicialización.
      # Si existe ./init.sql, MySQL lo ejecuta la PRIMERA vez que arranca
      - ./init.sql:/docker-entrypoint-initdb.d/init.sql

    # Healthcheck: Docker ejecuta este comando periódicamente para saber
    # si MySQL está listo para recibir conexiones.
    # El servicio "app" no arrancará hasta que esto devuelva éxito.
    healthcheck:
      test: ["CMD", "mysqladmin", "ping", "-h", "localhost", "-u", "user", "-puser"]
      interval: 1s  # Comprueba cada 1 segundos
      timeout: 5s    # Si no responde en 5s, cuenta como fallo
      retries: 5     # Tras 5 fallos seguidos, el servicio se marca como unhealthy

  # ──────────────────────────────────────────
  # SERVICIO 3: phpMyAdmin
  # ──────────────────────────────────────────
  ▷Run Service
  phpmyadmin:
    # Imagen oficial de phpMyAdmin.
    image: phpmyadmin:latest
    #nombre del contenedor
    container_name: phpmyadmin
    #restart no apra que no se ejecute todo el rato
    restart: no
    # Puerto de acceso: en tu navegador.
    ports:
      - "8080:80"
    #ajustes de php my admin
    environment:
      - PMA_HOST=db
      - PMA_PORT=3306.
      - PMA_USER=user
      - PMA_PASSWORD=user
      - UPLOAD_LIMIT=64M

    # phpMyAdmin necesita que MySQL esté funcionando antes de arrancar, lo comoprueba.
    depends_on:
      db:
        condition: service_healthy

# ──────────────────────────────────────────
# VOLÚMENES nombrados
# ──────────────────────────────────────────
volumes:
  mysql_data:
```

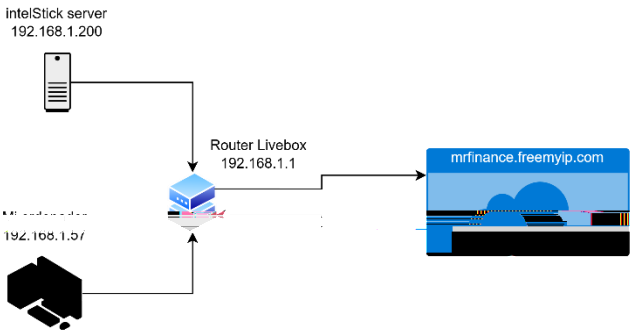
"

"

```dockerfile
FROM node:20-alpine

# Directorio de trabajo
WORKDIR /app

# Copia los archivos de dependencias primero (aprovecha la caché de Docker)
# Si no existen aún, el COPY falla silenciosamente gracias al entrypoint
COPY entrypoint.sh /entrypoint.sh
RUN chmod +x /entrypoint.sh

ENTRYPOINT ["/entrypoint.sh"]
```

"

```
#!/bin/sh
set -e

echo "Iniciando contenedor Node.js..."

# Si existe package.json, instala dependencias y arranca
if [ -f "/app/package.json" ]; then
  echo "package.json detectado. Instalando dependencias..."
  cd /app
  npm install

  # Arranca con el script "start" si existe, si no con node directamente
  if npm run | grep -q "^ *start$"; then
    echo "Ejecutando: npm start"
    exec npm start
  else
    # Intenta adivinar el entry point
    ENTRY=$(node -e "const p=require('./package.json'); console.log(p.main||'index.js')" 2>/dev/null || echo "index.js") ...
    [illegible]
    [illegible]
    fi
  else
    echo "No se encontró /app/package.json"
    echo "   Monta tu proyecto en ./app y reinicia el contenedor."
    # Mantiene el contenedor activo para poder depurar
    exec tail -f /dev/null
fi
```



```yaml
# ── Node.js + vite ──────────────────────────────────────────────
▷ Run Service
app:
  build:
    context: .
    dockerfile: Dockerfile
  container_name: node_app2
  volumes:
    - ./app:/app                    # código fuente en ./app del host
    - mrfinance_root_modules:/app/mrfinance/node_modules
    - mrfinance_server_modules:/app/mrfinance/server/node_modules
    - mrfinance_client_modules:/app/mrfinance/client/node_modules
  ports:
    - "587:587"
    - "3000:3000"
    - "5173:5173"
    - "8081:8081"
    # - "0.0.0.0:443:5173"
    # - "0.0.0.0:80:5173"
  environment:
    NODE_ENV: development
    DB_HOST: db
    DB_PORT: 3306
    DB_USER: user
    DB_PASSWORD: user
  depends_on:
    db:
      condition: service_healthy
  networks:
    - dev_net
  restart: no
```

```yaml
# ── Nginx ──────────────────────────────────────────────
▷ Run Service
nginx:
  image: jonasal/nginx-certbot:latest
  container_name: nginx_server
  ports:
    - "80:80"
    - "443:443"
  environment:
    - CERTBOT_EMAIL=noreply.mrfinance@gmail.com
  volumes:
    - ./app/mrfinance/client/dist:/usr/share/nginx/html:ro
    - ./nginx.conf:/etc/nginx/user_conf.d/default.conf:ro
    - certbot_letsencrypt:/etc/letsencrypt
  depends_on:
    - app
  networks:
    - dev_net
  restart: no
```

```yaml
# ── MySQL ──────────────────────────────────────────────────────────
▷ Run Service
db:
  image: mysql:latest
  container_name: mysql2
  environment:
    MYSQL_ROOT_PASSWORD: root
    MYSQL_DATABASE: MrFinanceV2
    MYSQL_USER: user
    MYSQL_PASSWORD: user
  volumes:
    - mysql_data:/var/lib/mysql
    - ./db/init:/docker-entrypoint-initdb.d
  ports:
    - "3306:3306"
  healthcheck:
    test: ["CMD", "mysqladmin", "ping", "-h", "localhost", "-u", "root", "-proot"]
    interval: 10s
    timeout: 5s
    retries: 5
  networks:
    - dev_net
  restart: no
```

"

```yaml
# ── phpMyAdmin ─────────────────────────────────────────────────────
▷ Run Service
phpmyadmin:
  image: phpmyadmin:latest
  container_name: phpmyAdmin2
  environment:
    PMA_HOST: db
    PMA_PORT: 3306
    PMA_USER: root
    PMA_PASSWORD: root
    UPLOAD_LIMIT: 100M
  ports:
    - "8080:80"
  depends_on:
    db:
      condition: service_healthy
  networks:
    - dev_net
  restart: no
```

"

" " " " " "

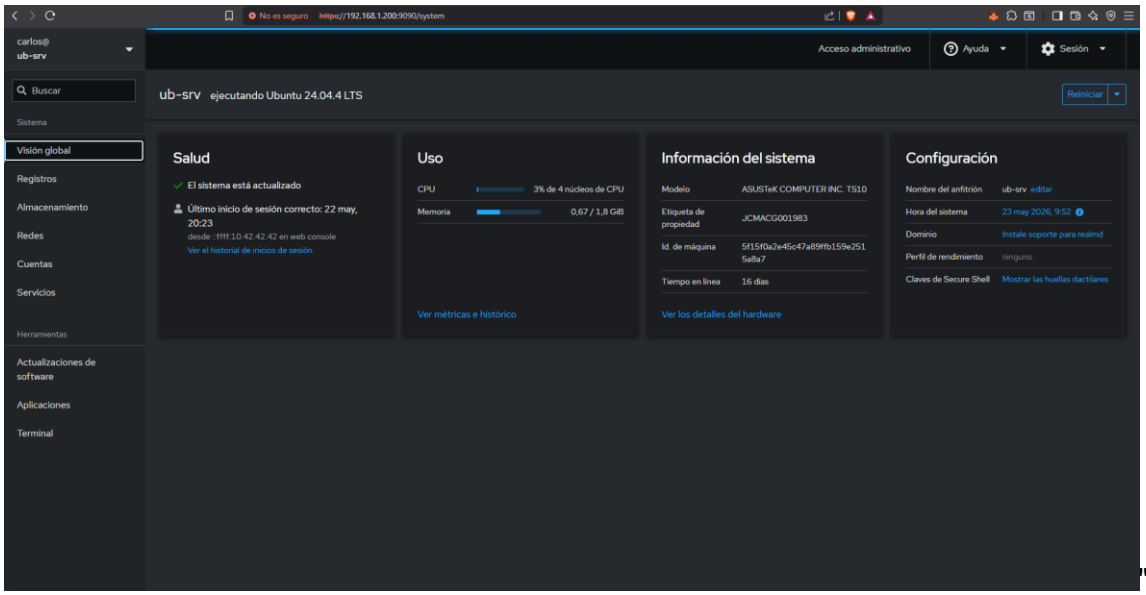
```yaml
# ── Volúmenes ──────────────────────────────────────────────────────
volumes:
  mysql_data:
  mrfinance_root_modules:
  mrfinance_server_modules:
  mrfinance_client_modules:
  certbot_letsencrypt:

# ── Red interna ────────────────────────────────────────────────────
networks:
  [illegible]:
    [illegible]: bridge
```

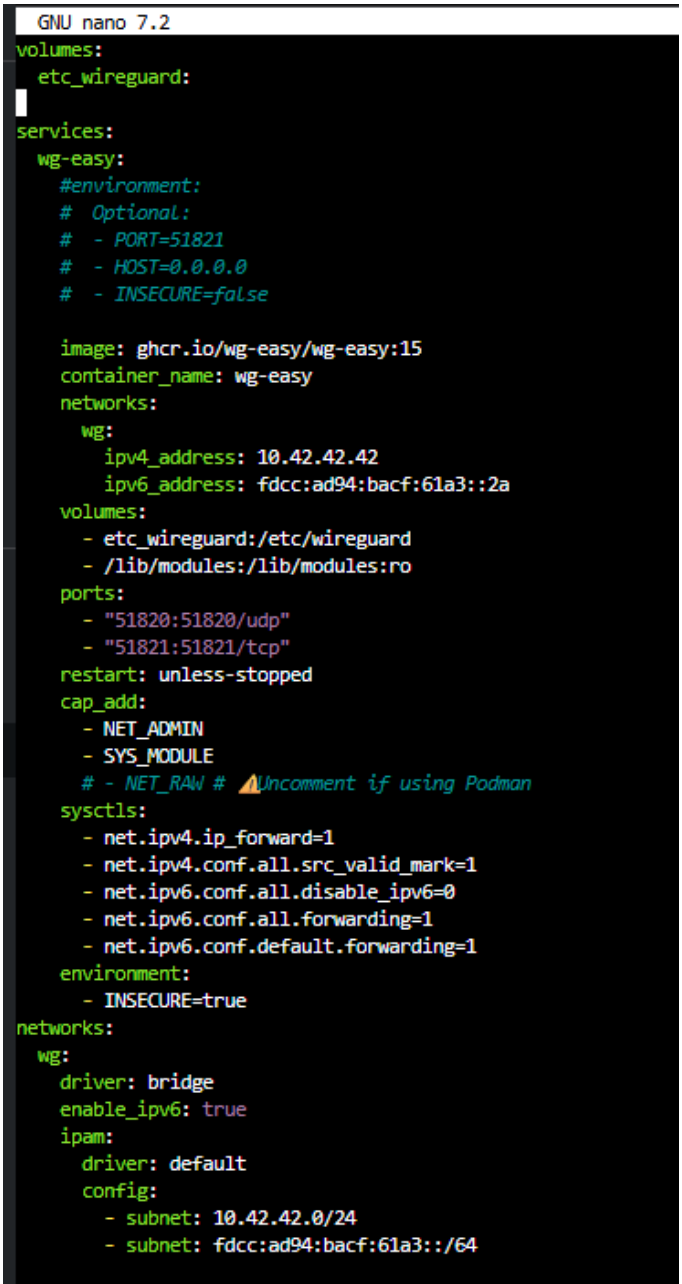
"

" " " "

```
#!/bin/sh
set -e

echo "======================================"
echo "  Iniciando contenedor MrFinance"
echo "======================================"

PROJECT_DIR="/app/mrfinance"

# Verificar que el proyecto existe
if [ ! -d "$PROJECT_DIR" ]; then
  echo ">>> ERROR: No se encontro el proyecto en $PROJECT_DIR"
  exit 1
fi

# ── Instalar dependencias del servidor ──────────────────────────────
echo ">>> Instalando dependencias del servidor..."
cd "$PROJECT_DIR/server"
npm install

# Dependencias que el codigo del servidor importa pero no estan en su package.json
echo ">>> Instalando dependencias adicionales del servidor (bcrypt, dotenv, nodemailer)..."
npm install bcrypt dotenv nodemailer
npm install multer path

# ── Instalar dependencias del cliente ───────────────────────────────
echo ">>> Instalando dependencias del cliente..."
cd "$PROJECT_DIR/client"
npm install

# ── Arrancar servidor y cliente ─────────────────────────────────────
echo "======================================"
echo "  MrFinance listo"
echo "  Client (Vite):  http://localhost:5173"
echo "  Server (API):   http://localhost:8081"
echo "======================================"

cd "$PROJECT_DIR"
exec concurrently -n SERVER,CLIENT -c blue,green \
  "cd $PROJECT_DIR/server && npx nodemon server.js" \
  "cd $PROJECT_DIR/client && npx vite --host --port 5173"

```

JonasAlfredsson





```
  GNU nano 7.2
volumes:
  etc_wireguard:

services:
  wg-easy:
    #environment:
    #  Optional:
    #  - PORT=51821
    #  - HOST=0.0.0.0
    #  - INSECURE=false

    image: ghcr.io/wg-easy/wg-easy:15
    container_name: wg-easy
    networks:
      wg:
        ipv4_address: 10.42.42.42
        ipv6_address: fdcc:ad94:bacf:61a3::2a
    volumes:
      - etc_wireguard:/etc/wireguard
      - /lib/modules:/lib/modules:ro
    ports:
      - "51820:51820/udp"
      - "51821:51821/tcp"
    restart: unless-stopped
    cap_add:
      - NET_ADMIN
      - SYS_MODULE
      # - NET_RAW # ⚠️Uncomment if using Podman
    sysctls:
      - net.ipv4.ip_forward=1
      - net.ipv4.conf.all.src_valid_mark=1
      - net.ipv6.conf.all.disable_ipv6=0
      - net.ipv6.conf.all.forwarding=1
      - net.ipv6.conf.default.forwarding=1
    environment:
      - INSECURE=true
networks:
  wg:
    driver: bridge
    enable_ipv6: true
    ipam:
      driver: default
      config:
        - subnet: 10.42.42.0/24
        - subnet: fdcc:ad94:bacf:61a3::/64
```

" " " "

" " " "

" " " " " "

Columna1	Columna2	Columna3	Columna4	Columna5	Columna6
" " " "

5.3 Esquema del flujo de trabajo y desarrollo

"

" " " "

" " "

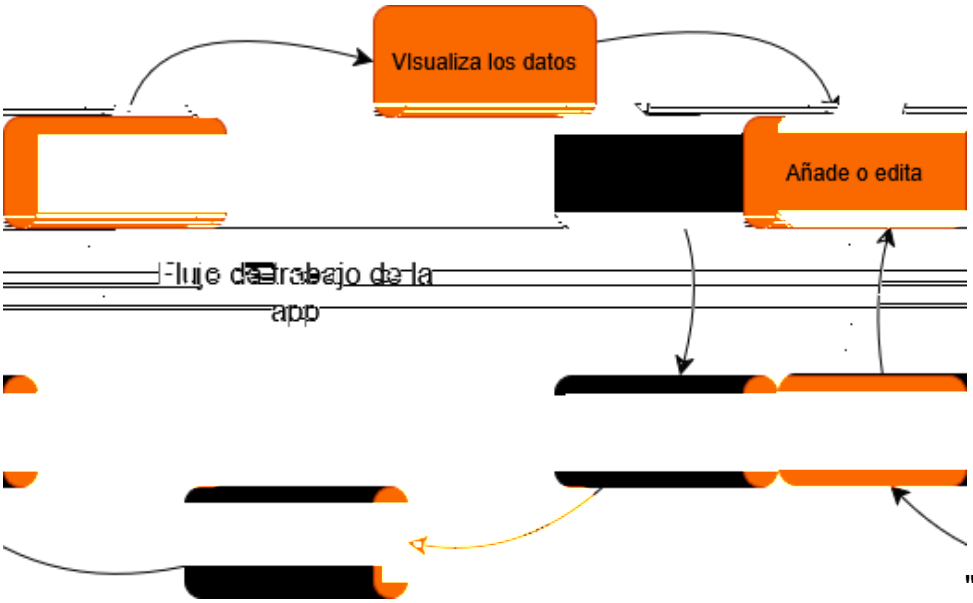
"

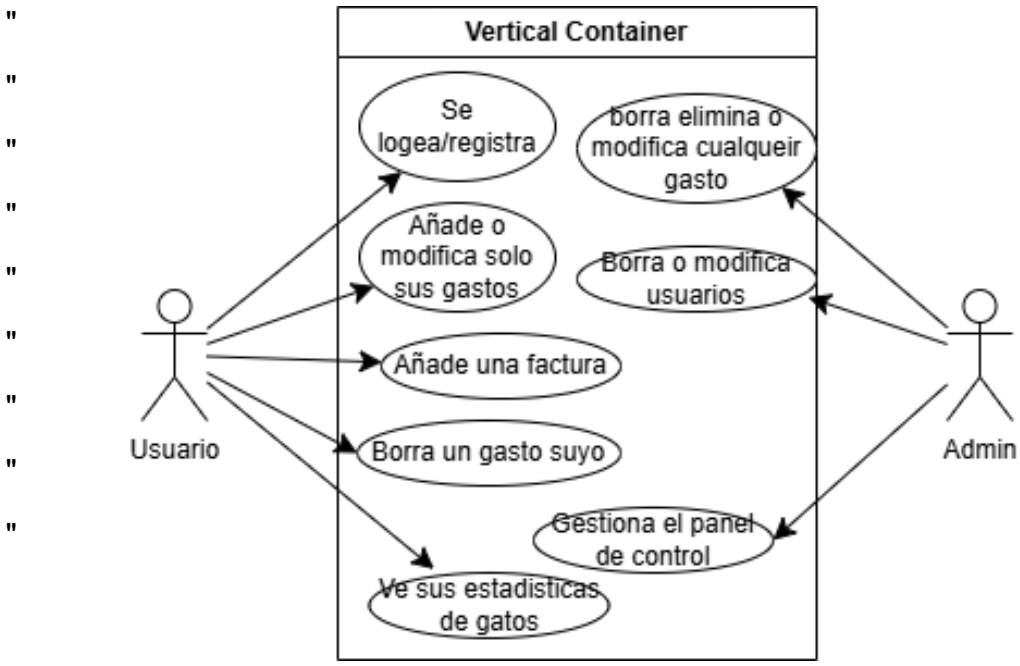
" " " "

"

" "

- " " " " " " " " "
- " " " "


Visualiza los datos
Añade o edita
Flujo de trabajo de la app


Vertical Container
Se logea/registra
borra elimina o modifica cualqueir gasto
Añade o modifica solo sus gastos
Borra o modifica usuarios
Añade una factura
Borra un gasto suyo
Gestiona el panel de control
Ve sus estadisticas de gatos
Usuario
Admin

"

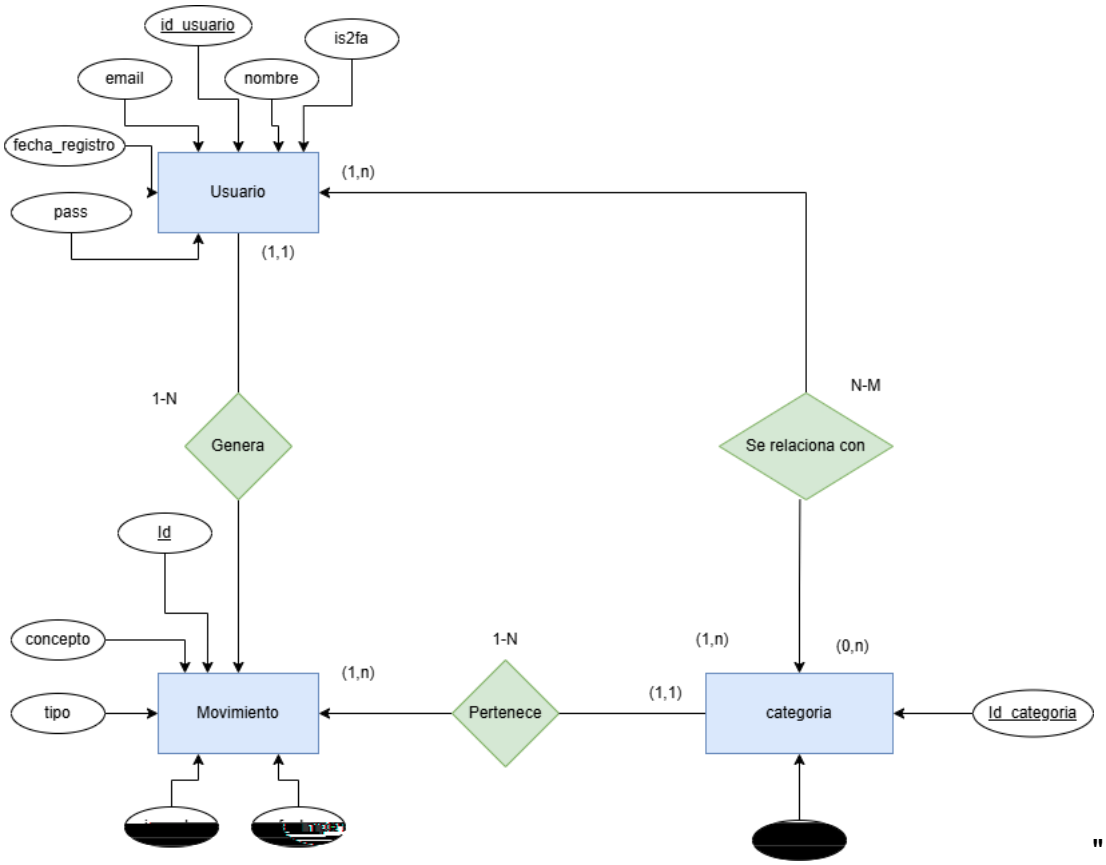
" " " "

"

" " " " " "

"

Modelo entidad relación:

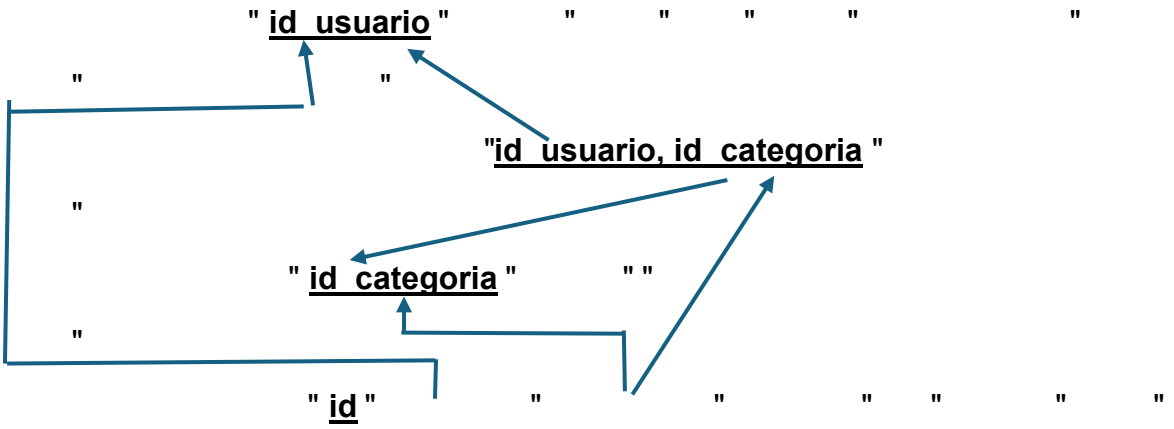


"

"

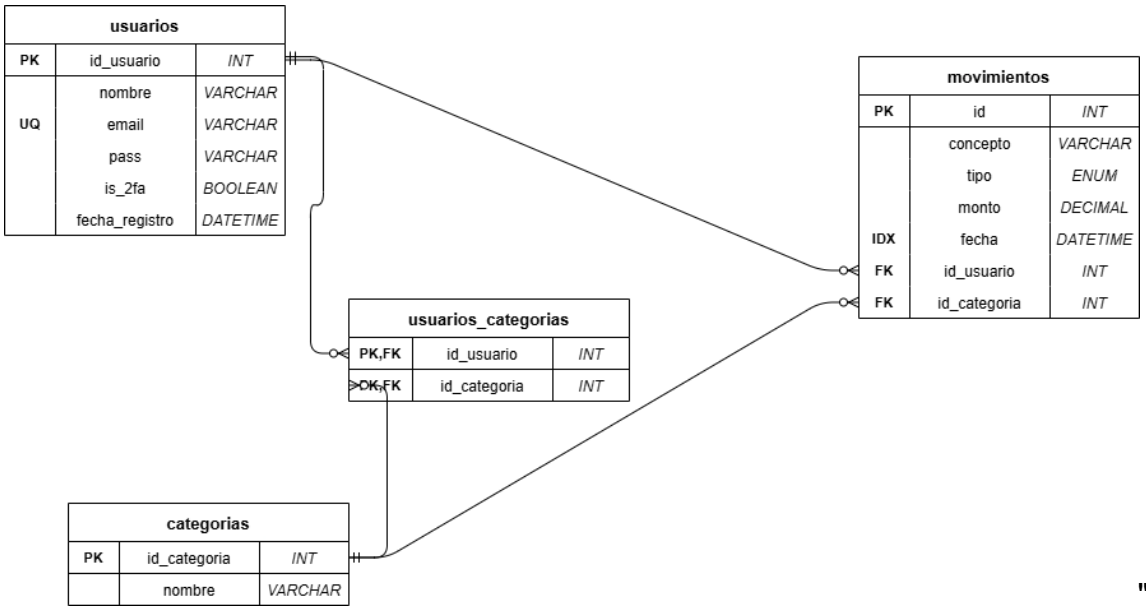
"

Grafo relacional:



"

Diccionario de datos de las tablas



"

"

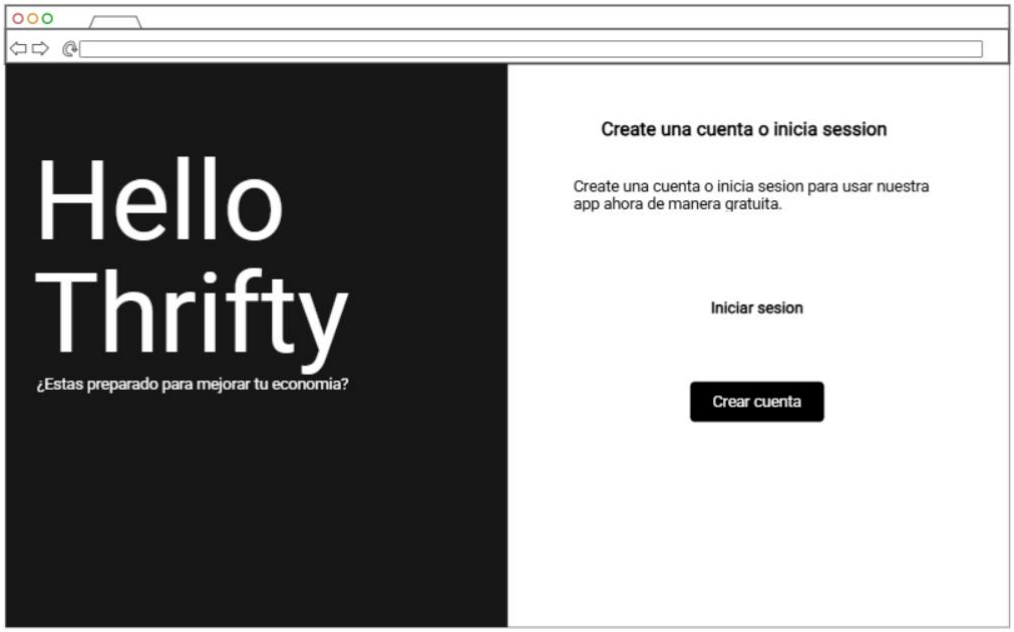
"

"

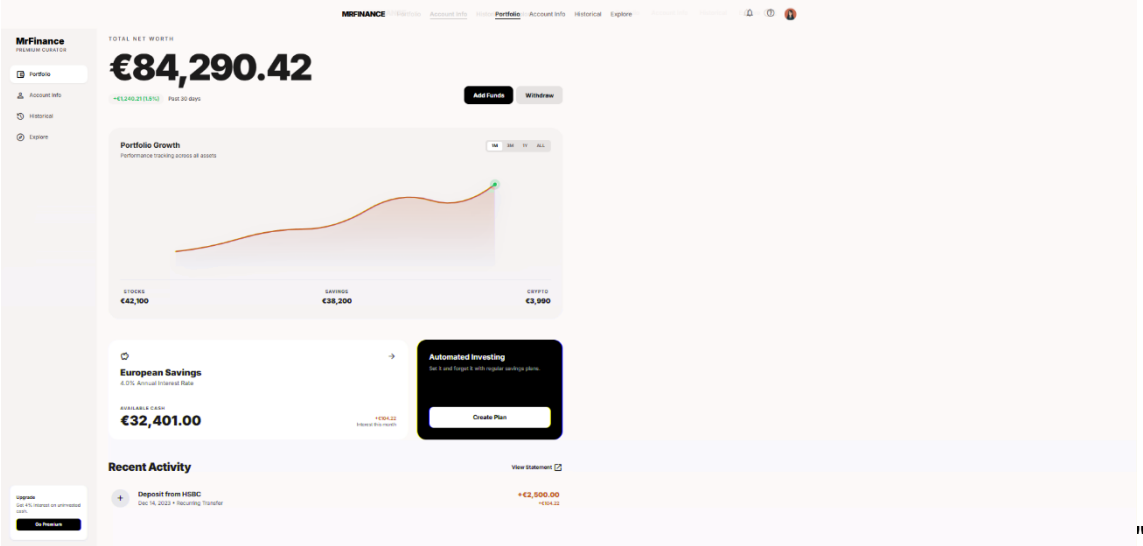
" " " " " " " " " " " " "

" " " " " " " " " " " " "

" " " " " "

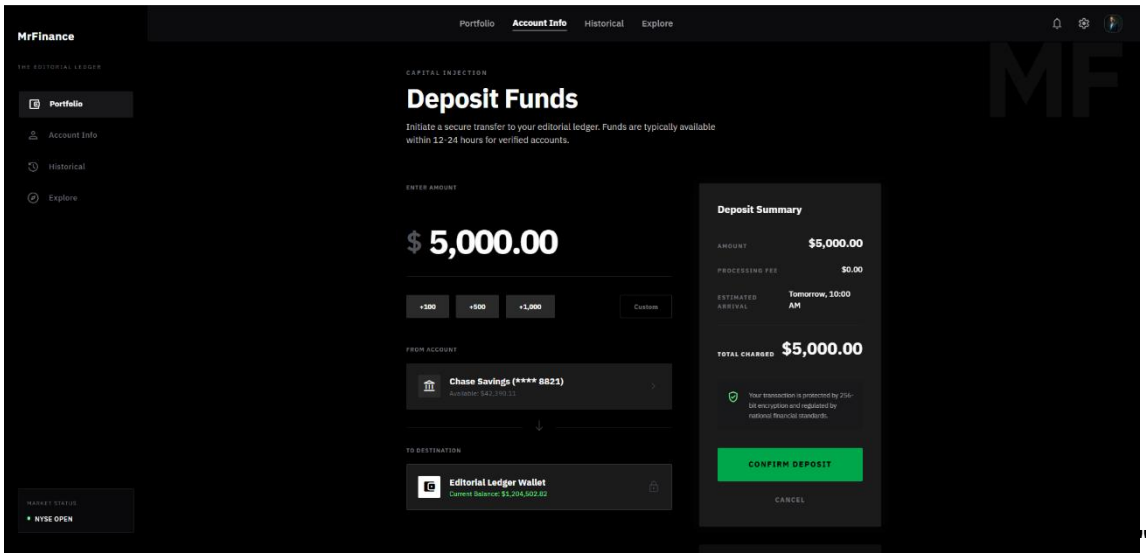

Hello
Thrifty
¿Estas preparado para mejorar tu economia?
Create una cuenta o inicia session
Create una cuenta o inicia sesion para usar nuestra app ahora de manera gratuita.
Iniciar sesion
Crear cuenta

"


MRFINANCE
Portfolio
Account Info
Historical
Explore
MrFinance
PREMIUM CURATOR
Portfolio
Account Info
Historical
Explore
TOTAL NET WORTH
€84,290.42
+€1,240.21 (1.5%)
Past 30 days
Add Funds
Withdraw
Portfolio Growth
Performance tracking across all assets
1M
3M
1Y
ALL
STOCKS
€42,100
SAVINGS
€38,200
CRYPTO
€3,990
European Savings
4.0% Annual Interest Rate
AVAILABLE CASH
€32,401.00
+€104.22
Interest this month
Automated Investing
Set it and forget it with regular savings plans.
Create Plan
Recent Activity
View Statement
Deposit from HSBC
Dec 14, 2023 • Recurring Transfer
+€2,500.00
+€104.22
Upgrade
Get 4% interest on uninvested cash.
Go Premium

"

"



"

"

Como se ha orientado finalmente la arquitectura en la versión de Docker.

"

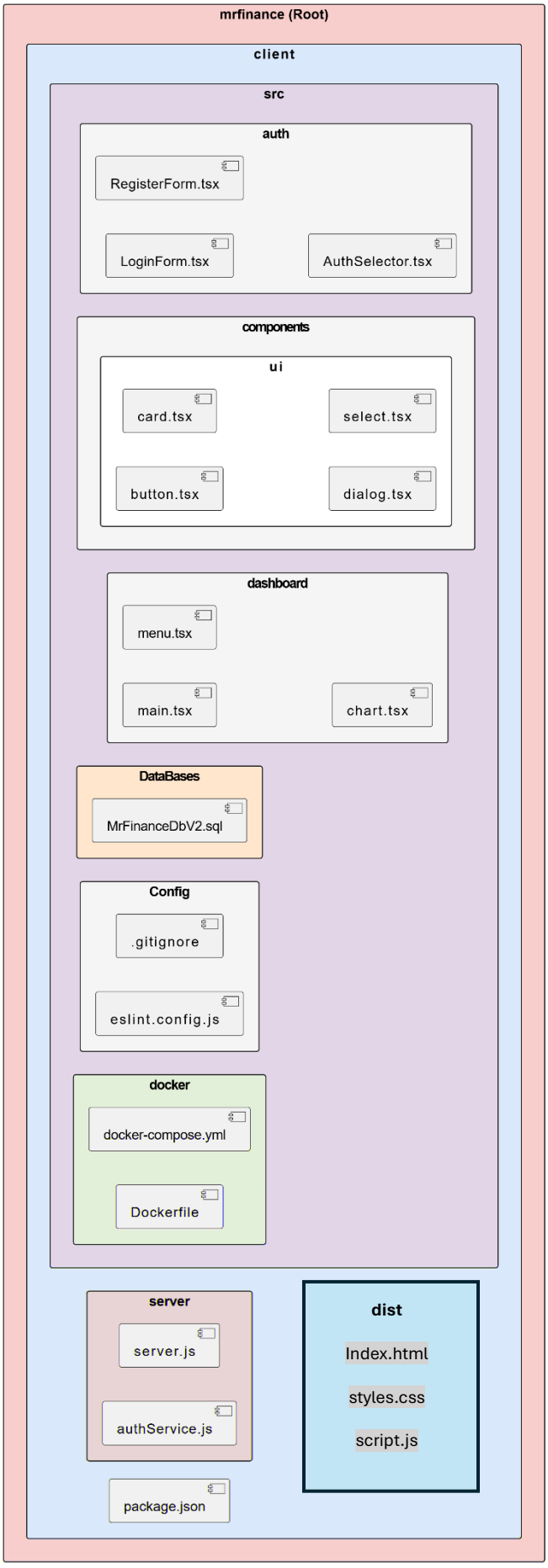
- **" docker compose up arranca 3 servicios"** "
- **" MySQL"** "
- **" entrypoint.sh"** "
- **" El login de MrFinance"** " "http://localhost:5173"

"

" "

" " "

"


mrfinance (Root)
client
src
auth
RegisterForm.tsx
LoginForm.tsx
AuthSelector.tsx
components
ui
card.tsx
select.tsx
button.tsx
dialog.tsx
dashboard
menu.tsx
main.tsx
chart.tsx
DataBases
MrFinanceDbV2.sql
Config
.gitignore
eslint.config.js
docker
docker-compose.yml
Dockerfile
server
server.js
authService.js
package.json
dist
Index.html
styles.css
script.js

Podemos ver en el esquema grafico como está estructurado el proyecto, esta es la estructura por defecto en un proyecto vite, algunas características principales de esta estructura son la separación clara del backend con el frontend y mi propia organización de ficheros a la hora de programar los componentes como la separación de la carpeta auth y dashboard, que separan el login de las vistas de cada usuario con sus respectivos datos etc.

Otro detalle es la carpeta components que engloba componentes react de shadcn para luego poderlos usar más adelante en el frontend."

```sql
-- ==========================================================
-- Base de datos: MrFinanceV2
-- Segunda version con triggers para el borrado automatico en vez de on delete cascade
-- ==========================================================

DROP DATABASE IF EXISTS MrFinanceV2;
CREATE DATABASE IF NOT EXISTS MrFinanceV2 DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;

USE MrFinanceV2;

-- ----------------------------------------------------------
-- Tabla: usuarios
-- ----------------------------------------------------------
CREATE TABLE usuarios (
  id              INT          NOT NULL AUTO_INCREMENT,
  nombre          VARCHAR(100) NOT NULL,
  email           VARCHAR(120) NOT NULL,
  pass            VARCHAR(120) NOT NULL,
  is_2fa          BOOLEAN      NOT NULL DEFAULT FALSE,
  fecha_registro  DATETIME     NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (id),
  -- unique key en los email para que no existan usuarios duplicados
  UNIQUE KEY uq_usuarios_email (email)
);

-- ----------------------------------------------------------
-- Tabla: categoria
-- ----------------------------------------------------------
CREATE TABLE categoria (
  id          INT          NOT NULL AUTO_INCREMENT,
  nombre      VARCHAR(100) NOT NULL,
  id_usuario INT NOT NULL,
  FOREIGN KEY (id_usuario) REFERENCES usuarios (id)
  PRIMARY KEY (id)
);


-- ----------------------------------------------------------
-- Tabla: movimientos
-- ----------------------------------------------------------
CREATE TABLE movimientos (
  id            INT                     NOT NULL AUTO_INCREMENT,
  concepto      VARCHAR(255)            NOT NULL,
  tipo          ENUM('ingreso','gasto') NOT NULL,
  monto         DECIMAL(10,2)           NOT NULL,
  fecha         DATETIME                NOT NULL DEFAULT CURRENT_TIMESTAMP,
  id_usuario    INT                     NOT NULL,
  id_categoria  INT                     NOT NULL,
  PRIMARY KEY (id),
  -- uso indices para que la bd cargue en cache esos campos y no tarden tanto las consultas
  INDEX idx_movimientos_usuario   (id_usuario),
  INDEX idx_movimientos_categoria (id_categoria),
  INDEX idx_movimientos_fecha     (fecha),
  -- foreing keys referentes a los usuarios y categorias
    FOREIGN KEY (id_usuario) REFERENCES usuarios (id)
    FOREIGN KEY (id_categoria) REFERENCES categoria (id)
);


-- ==========================================================
-- Triggers
-- ==========================================================

DELIMITER $$

-- ----------------------------------------------------------
-- Trigger: para antes de eliminar un usuario
-- Elimina primero sus movimientos para no crear datos sueltos.
-- Despues borra las categorias del usuario.
-- ----------------------------------------------------------
CREATE TRIGGER trg_before_delete_usuario
BEFORE DELETE ON usuarios
FOR EACH ROW
BEGIN
  DELETE FROM movimientos  WHERE id_usuario = OLD.id;
  DELETE FROM categoria WHERE id_usuario = OLD.id;
END$$

-- ----------------------------------------------------------
-- Trigger: para antes de eliminar una categoría
-- Impide el borrado si existen movimientos con esa categoria,
-- devolviendo un mensaje de error si se intenta hacer el delete.
-- ----------------------------------------------------------
CREATE TRIGGER trg_before_delete_categoria
BEFORE DELETE ON categoria
FOR EACH ROW
BEGIN
  DECLARE v_total INT;

  SELECT COUNT(*) INTO v_total
  FROM movimientos
  WHERE id_categoria = OLD.id;

  IF v_total > 0 THEN
```

"

```
    SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'No se puede eliminar la categoría: tiene movimientos asociados.';
  END IF;
END$$

DELIMITER ;


-- ============================================================
-- Datos de ejemplo
-- ============================================================

INSERT INTO usuarios (nombre, email, pass, is_2fa) VALUES
  ('Ana García', 'ana@example.com', '1231', FALSE),
  ('Luis Pérez',  'luis@example.com', '123', TRUE),
  ('pepe',  'pepe@example.com', 'pepe', FALSE);


INSERT INTO categoria (nombre, id_usuario) VALUES
  ('Alimentación',1),
  ('Transporte',2),
  ('Salario',3),
  ('Ocio',1),
  ('Ahorro',1);

INSERT INTO movimientos (concepto, tipo, monto, id_usuario, id_categoria) VALUES
  ('Nómina marzo',     'ingreso', 1800.00, 1, 3),
  ('Supermercado',     'gasto',     65.40, 1, 1),
  ('Metro mensual',    'gasto',     40.00, 1, 2),
  ('Ahorro mensual',   'ingreso',  200.00, 1, 5),
  ('Cena restaurante', 'gasto',     35.00, 2, 4),
  ('Freelance abril',  'ingreso',  500.00, 2, 3);
```

"

"

"

"

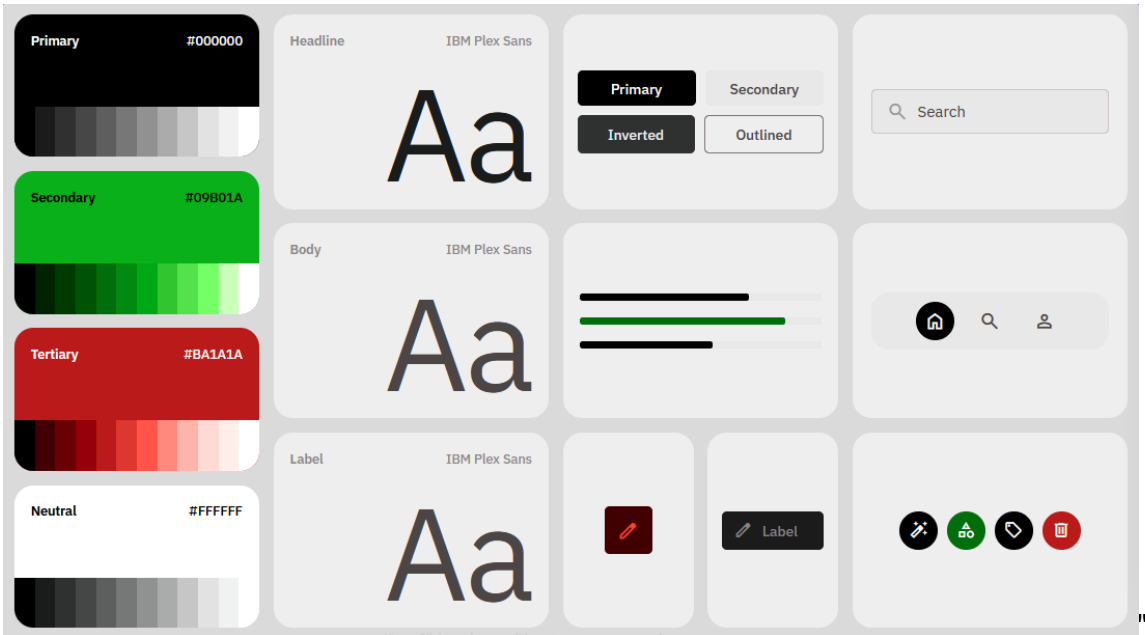
"

"

"

"

"



"

"

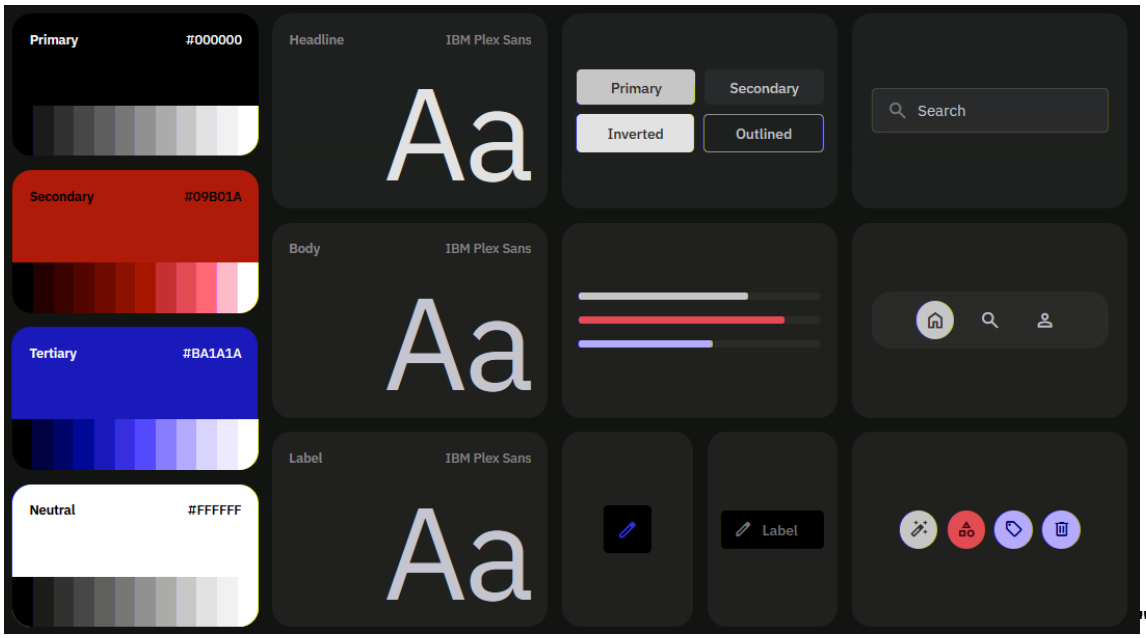
"

"

"

"

"


Primary
#000000
Secondary
#09B01A
Tertiary
#BA1A1A
Neutral
#FFFFFF
Headline
IBM Plex Sans
Aa
Body
IBM Plex Sans
Aa
Label
IBM Plex Sans
Aa
Primary
Secondary
Inverted
Outlined
Search
Label

Cuestiones de implementación y codificación reseñables

Código node.js (backend)

El backend es un servicio que se arranca en concurrency con el fronted, esto se define el package.json de la raíz, para que arranque a la vez el server y el client,

Se instala con:

npm init -y

npm install -D concurrently

Y después se añaden estas líneas al pakage.json

```
  "scripts": {
    "dev": "concurrently -n SERVER,CLIENT -c blue,green \"npm run dev --prefix server\" \"npm run dev --prefix client\"",
    "install:all": "npm install --prefix server && npm install --prefix client",
    "build": "npm run build --prefix client"
  },
  "devDependencies": {
    "concurrently": "^9.2.1"
  },
```

Luego al hacer npm run dev se ve así:

```
> concurrently -n SERVER,CLIENT -c blue,green "npm run dev --prefix server" "npm run dev --prefix client"

[SERVER]
[SERVER] > mrfinance-server@1.0.0 dev
[SERVER] > nodemon server.js
[SERVER]
[CLIENT]
[CLIENT] > mrfinance@0.0.0 dev
[CLIENT] > vite
[CLIENT]
[SERVER] [nodemon] 3.1.14
[SERVER] [nodemon] to restart at any time, enter `rs`
[SERVER] [nodemon] watching path(s): *.*
[SERVER] [nodemon] watching extensions: js,mjs,cjs,json
[SERVER] [nodemon] starting `node server.js`
[CLIENT]
[CLIENT]   VITE v8.0.9  ready in 2405 ms
[CLIENT]
[CLIENT]   ➜ Local:   http://localhost:5173/
[CLIENT]   ➜ Network: http://172.21.0.4:5173/
[SERVER] Servidor escuchando en puerto 8081 ✓
[SERVER] Conectado a MySQL ✓
```

Luego todo lo que es el código del backend está definido en la carpeta server en el archivo server.js, este archivo es ejecutado por los logs azules que pone server, se puede ver que arranca correctamente por las líneas de Servidor escuchando y conectado a mysql.

A la hora de programar este archivo me he encontrado con el reto de aprender js adaptado a un entorno backend y todo lo que eso conlleva. Estas son algunas capturas de funciones interesantes: Capturas de pantalla del archivo:

" " " " " "

```
You, 2 days ago | 2 authors (You and one other)
import express from 'express';
import mysql from 'mysql2';
import cors from 'cors';
import bcrypt from 'bcrypt';

//importo las funciones para el 2fa
import { enviarCodigoVerificacion, verificarCodigo } from './authService.js';

const app = express();

app.use(cors());
app.use(express.json());

const db = mysql.createConnection({
    host: "db",
    user: "user",
    password: "user",
    database: "MrFinanceV2",
    charset: "utf8mb4"
});
```

" " "

```
// Logica de login
app.post('/login', (req, res) => {
  const { email, password } = req.body;
    Ctrl+L for Agent
  const sql = "SELECT * FROM usuarios WHERE email = ?";
  db.query(sql, [email], async (err, data) => {
    if (err) {
      console.error("Error en query:", err.message);
      return res.status(500).json({ success: false, message: "Error del servidor" });
    }

    if (data.length === 0) {
      return res.status(401).json({ success: false, message: "Email o contraseña incorrectos" });
    }

    const user = data[0];
    const isBcryptHash = user.pass && user.pass.startsWith('$2');

    let passwordMatch = false;
    if (isBcryptHash) {
      passwordMatch = bcrypt.compareSync(password, user.pass);
    } else {
      passwordMatch = (password === user.pass);
      if (passwordMatch) {
        const salt = bcrypt.genSaltSync(10);
        const hashedPass = bcrypt.hashSync(password, salt);
        db.query("UPDATE usuarios SET pass = ? WHERE email = ?", [hashedPass, user.email]);
        console.log(`Contraseña migrada a bcrypt para: ${user.email}`);
      }
    }

    if (!passwordMatch) {
      return res.status(401).json({ success: false, message: "Email o contraseña incorrectos" });
    }

    // Comprobar si tiene 2FA activo en la BD
    if (user.is_2fa) {
      try {
        await enviarCodigoVerificacion(user.email);
        // No devolvemos los datos del usuario todavia, solo avisamos al frontend
        return res.status(200).json({ success: true, requires2fa: true, email: user.email });
      } catch (error) {
        console.error("Error enviando codigo 2FA:", error.message);
        return res.status(500).json({ success: false, message: "Error enviando el codigo de verificacion" });
      }
    }

    // Sin 2FA: login directo
    return res.status(200).json({ success: true, user });
  });
});
```

" " " "

```javascript
import transporter from "./mailer.js";

const codigosActivos = new Map();

async function enviarCodigoVerificacion(emailDestino) { //seactiva cuando se le pasa un mail
  const codigo = Math.floor(1000 + Math.random() * 9000).toString(); //genera un numero aleatorio de 4 digitos
  const expiracion = Date.now() + 10 * 60 * 1000; //establece la expiracion en 10 minutos

  codigosActivos.set(emailDestino, { codigo, expiracion }); //guarda el codigo y la expiracion

  const info = await transporter.sendMail({
    from: `"MR Finance" <${process.env.GMAIL_USER}>`, //correo del remitente
    to: emailDestino, //correo del destinatario
    subject: "Tu codigo de verificacion", //asunto del correo
    text: `Tu codigo es: ${codigo}\nExpira en 10 minutos.`, //texto del correo
    html: `
      <div style="font-family:sans-serif;max-width:400px;margin:auto;padding:24px;
                  border:1px solid #e0e0e0;border-radius:8px;">
        <h2 style="color:black;">Verificacion en 2 pasos de MR FINANCE</h2>
        <p>Usa el siguiente codigo para completar tu inicio de sesion:</p>
        <div style="font-size:36px;font-weight:bold;letter-spacing:8px;
                    text-align:center;color:black;padding:16px 0;">
          ${codigo}
        </div>
        <p style="color:gray;font-size:13px;">
          <strong color:black;>Este codigo expira en 10 minutos</strong>.<br>
          Si no solicitaste este codigo ignora este mensaje.
        </p>
      </div>
    `,
  });

  console.log("Codigo enviado. Message ID:", info.messageId);
  return true;
}

function verificarCodigo(emailDestino, codigoIngresado) { //se activa cuando se le pasa un mail y un codigo
  const entrada = codigosActivos.get(emailDestino);

  //comprueba que el codigo sea correcto
  if (!entrada) {
    return { valido: false, mensaje: "No hay codigo activo para este correo." };
  }

  if (Date.now() > entrada.expiracion) {
    codigosActivos.delete(emailDestino);
    return { valido: false, mensaje: "El codigo ha expirado." };
  }

  if (entrada.codigo !== codigoIngresado) {
    return { valido: false, mensaje: "Codigo incorrecto." };
  }

  codigosActivos.delete(emailDestino); //elimina el codigo
  return { valido: true, mensaje: "Verificacion exitosa." };      You, 6 days ago • 12/5 mas o menos tengo el
}

export { enviarCodigoVerificacion, verificarCodigo };
```

"

" " " " " " " " " " " " " " " " " " " "

```
import nodemailer from "nodemailer";
import dotenv from "dotenv";
dotenv.config();

const transporter = nodemailer.createTransport({
  host: "smtp.gmail.com",
  port: 587,
  secure: false, // STARTTLS, Gmail usa 587 con upgrade
  auth: {
    user: process.env.GMAIL_USER, //usuario
    pass: process.env.GMAIL_APP_PASS, //contraseña
  },
});

//comprueba que el servidor SMTP este listo y no de errores
transporter.verify((error) => {
  if (error) {
    console.error("Error de conexion SMTP:", error.message);
  } else {
    console.log("Servidor SMTP listo para enviar mensajes");
  }
});

export default transporter;
```

Endpoint que se encarga de verificar los códigos que llegan al server y función para registrarse

"

```
// Endpoint para verificar el codigo 2FA        You, 6 days ago • 12/5 mas o menos tengo el doble factor
app.post('/verify-2fa', (req, res) => {
  const { email, codigo } = req.body;

  //ni no se le pasa el mail o el codigo da error
  if (!email || !codigo) {
    return res.status(400).json({ success: false, message: "Faltan datos" });
  }

  const resultado = verificarCodigo(email, codigo);

  if (!resultado.valido) {
    return res.status(401).json({ success: false, message: resultado.mensaje });
  }

  // Codigo correcto: devuelve los datos del usuario
  const sql = "SELECT * FROM usuarios WHERE email = ?";
  db.query(sql, [email], (err, data) => {
    if (err) {
      return res.status(500).json({ success: false, message: "Error del servidor" });
    }
    return res.status(200).json({ success: true, user: data[0] });
  });
});

// Logica de registro
app.post('/register', (req, res) => {
    const { email, password, nombre } = req.body;

    // Comprueba que la contraseña sea de minimo 12 caracteres
    if (!password || password.length < 12) {
        return res.status(401).json({ success: false, message: "La contraseña es muy corta, tiene que ser de minimo 12 caracteres" });
    }

    // Primero comprueba si ya existe un usuario con ese email
    const sqlCheckUser = "SELECT * FROM usuarios WHERE email = ?";
    db.query(sqlCheckUser, [email], (err, data) => {
        if (err) {
            console.error("Error en query:", err.message);
            return res.status(500).json({ success: false, message: "Error del servidor" });
        }

        // Si ya existe, devuelve conflicto
        if (data.length > 0) {
            return res.status(409).json({ success: false, message: "El usuario ya existe" });
        }

        // Si no existe, encripta la contraseña y lo inserta
        const salt = bcrypt.genSaltSync(10);
        const hashedPass = bcrypt.hashSync(password, salt);

        const sqlInsertUser = "INSERT INTO usuarios (email, pass, nombre) VALUES (?, ?, ?)";
        db.query(sqlInsertUser, [email, hashedPass, nombre], (err, result) => {
            if (err) {
                console.error("Error en query:", err.message);
                return res.status(500).json({ success: false, message: "Error del servidor" });
            }
            const user = {
                id: result.insertId,
                email: email,
                pass: hashedPass,
                nombre: nombre
            };
            return res.status(200).json({ success: true, user });
        });
    });
});
```

" " " "

```
import axios from "axios";
import { useState } from "react";
import { AutenticationModal } from "./AuthenticationModal";

type Props = {
  onBack: () => void;
  onSuccess: (user: any) => void;
};

export default function LoginForm({ onBack, onSuccess }: Props) {
  //toda la logica del login
  //aqui guarda los fatos de email pass...
  const [email, setEmail] = useState("");
  [illegible]
  [illegible]

  // Estado para el modal 2FA
  const [modal2fa, setModal2fa] = useState(false);
  const [email2fa, setEmail2fa] = useState("");

  //funcion handleSubmit que se ejecuta al hacer submit en el form de log
  const handleSubmit = (evento: React.FormEvent) => {
    //evita que se recargue la pagina
    evento.preventDefault();
    setError("");
    Ctrl+L for Agent
    //peticion axios al backend
    axios
      .post("/api/login", { email, password })
      .then((res) => {
        if (res.data.success && res.data.requires2fa) {
          // El servidor ha enviado el código al correo → abrir modal 2FA
          setEmail2fa(res.data.email);
          setModal2fa(true);
        } else if (res.data.success) {
          // Login directo sin 2FA
          onSuccess(res.data.user);
        } else {
          setError(res.data.message);
        }
      }
      .catch((err) => {
        console.error(err);
        setError("No se pudo iniciar sesion porfavor vuelve a intentarlo
      });
```

"

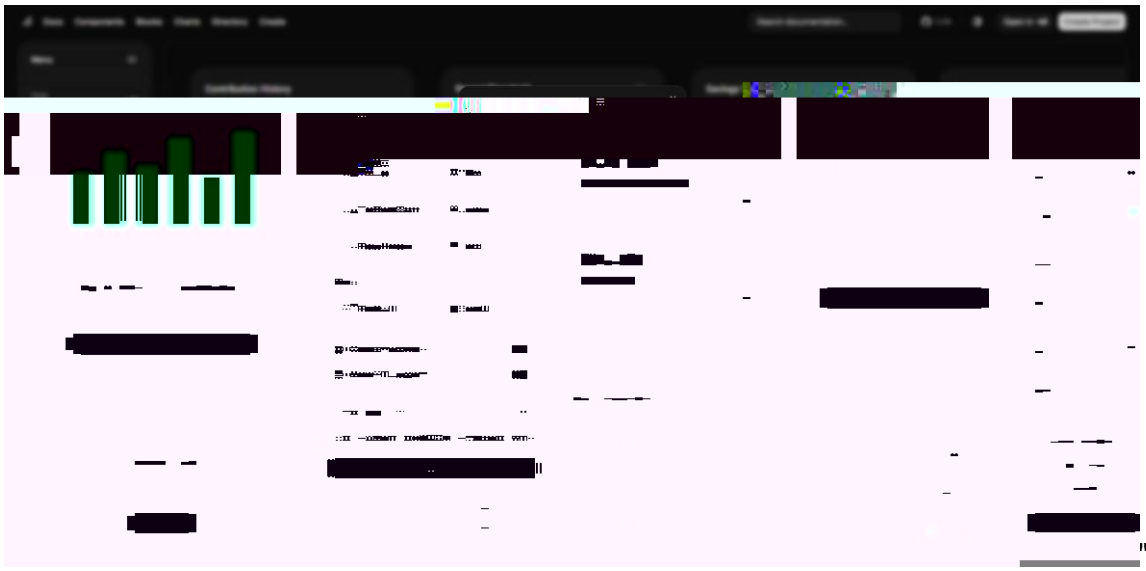
```
  // Cancelar 2FA → volver al formulario limpio
  function handleCancel2fa() {
    setModal2fa(false);
    setEmail2fa("");
    setPassword("");
    setError("");
  }

  return (
    <>
      <div className="loginForm">
        <form onSubmit={handleSubmit}>
          <h2>Iniciar sesión</h2>
          {error && <p style={{ color: "red" }}>{error}</p>}
          <input
            type="email"
            placeholder="Email"
            value={email}
            //aqui recoje el valor de el input y lo guarda con la funcion setEmail
            onChange={(e) => setEmail(e.target.value)}
          />
          <input
            type="password"
            placeholder="Contraseña"
            value={password}
            onChange={(e) => setPassword(e.target.value)}
          />
          <button type="submit">Entrar</button>
          <button type="button" onClick={onBack}>
            Volver
          </button>
        </form>
      </div>

      {/* Modal de verificación 2FA */}
      <AutenticationModal
        open={modal2fa}
        email={email2fa}
        onSuccess={(user) => {
          setModal2fa(false);
          onSuccess(user);
        }}
        onCancel={handleCancel2fa}
      />
    </>
  );
}
```

" "

Implementación de componentes shadcn:

" " " " " " " " " " " " "
" " " " " " " " " " " " " "
% " " "
" " " " " " " " " " " " " " " "
" " " "

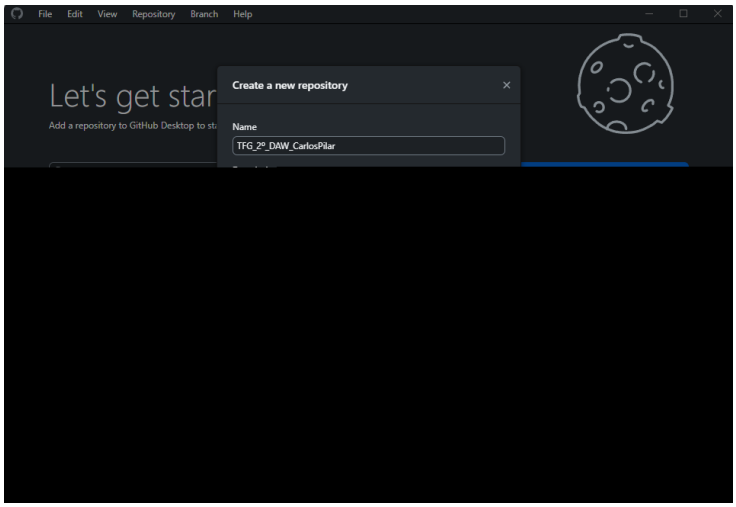
" " " " " "



"

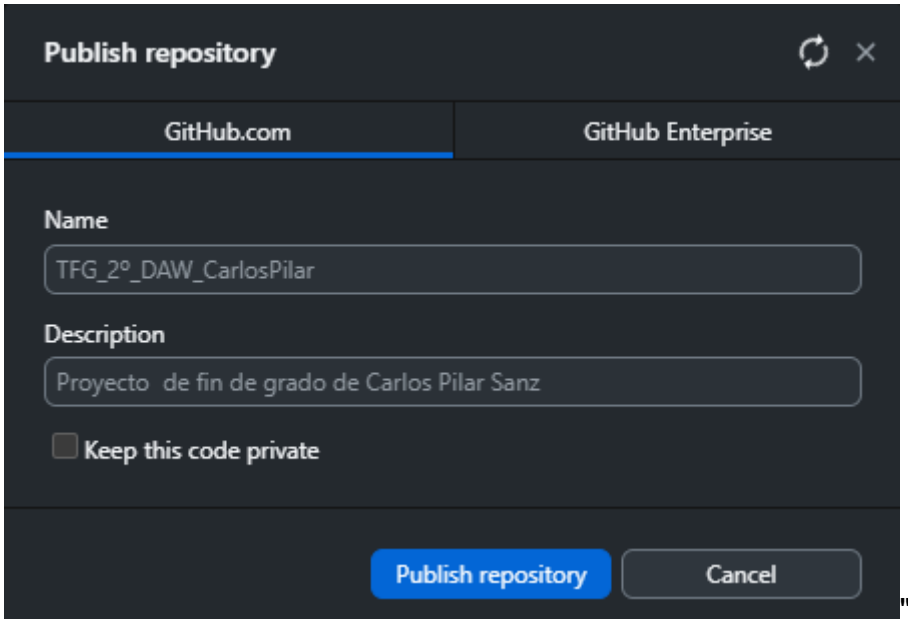
" "

```
PS C:\Users\carlo\Desktop\CodigoProyectop> git init
Initialized empty Git repository in C:/Users/carlo/Desktop/CodigoProyectop/.git/
PS C:\Users\carlo\Desktop\CodigoProyectop> git add *
PS C:\Users\carlo\Desktop\CodigoProyectop> git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   code.txt

PS C:\Users\carlo\Desktop\CodigoProyectop> git commit -m "Primer commit"
[master (root-commit) 9f75b45] Primer commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 code.txt
PS C:\Users\carlo\Desktop\CodigoProyectop> git status
On branch master
nothing to commit, working tree clean
PS C:\Users\carlo\Desktop\CodigoProyectop>
```



" " ""

"


Publish repository
GitHub.com
GitHub Enterprise
Name
TFG_2º_DAW_CarlosPilar
Description
Proyecto de fin de grado de Carlos Pilar Sanz
Keep this code private
Publish repository
Cancel

"

" "

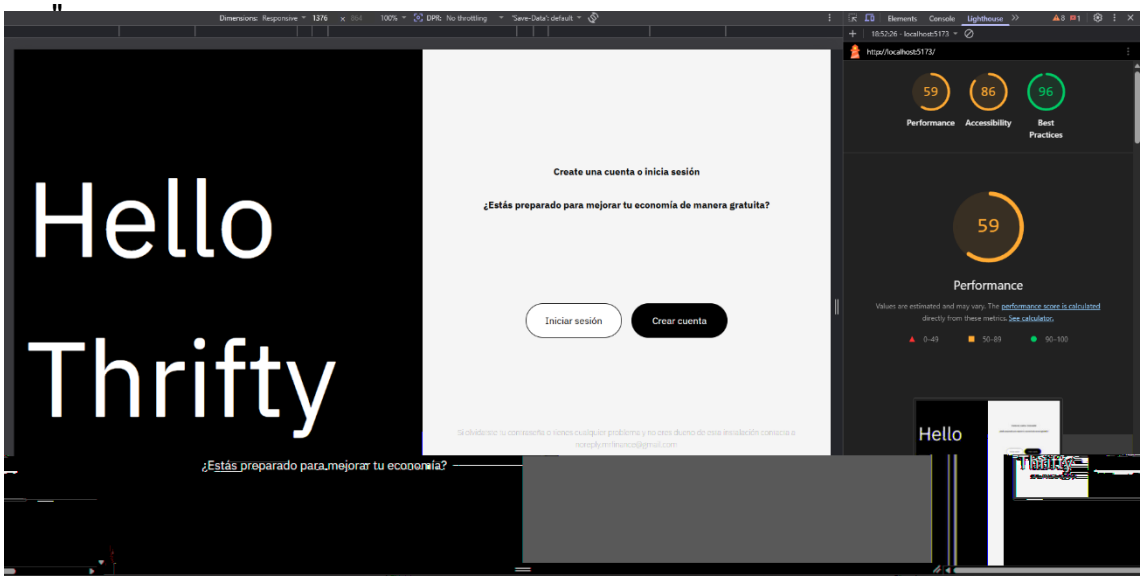
" " "

"

" " "

" " " " " " " " "

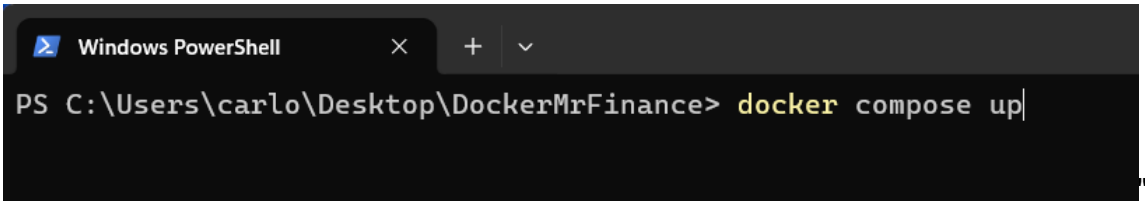
- " " " " " " "


Hello
Thrifty
¿Estás preparado para mejorar tu economía?
Create una cuenta o inicia sesión
¿Estás preparado para mejorar tu economía de manera gratuita?
Iniciar sesión
Crear cuenta
Elements
Consola
Lighthouse
http://localhost:5173/
59
Performance
86
Accessibility
96
Best Practices
59
Performance
Hello

" " " "

DockerMrFinance

```
server {
    listen 80;
    server_name mrfinance.freemyip.com;

    # Redirección automática a HTTPS
    location / {
        return 301 https://$host$request_uri;
    }
}

server {
    listen 443 ssl;
    server_name mrfinance.freemyip.com;

    ssl_certificate /etc/letsencrypt/live/TU-DOMINIO/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/TU-DOMINIO/privkey.pem;

    # Compilación estática de React (Frontend)
    location / {
        root /usr/share/nginx/html;
        index index.html;
        try_files $uri $uri/ /index.html;
    }

    # Redirección de peticiones de API al backend
    # El '/' al final de proxy_pass actúa como el rewrite de Vite, eliminando '/api/' de la petición
    location /api/ {
        proxy_pass http://app:8081/;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```



" " "

"

" " " " "

"

" " " "

" " " "

Conclusiones"técnicas"

" " " "

" " "

"

"

"

"

"

"

"

"

"

"

"

"

"

"

"

"

" "

"

- " Glosario de términos técnicos

Término	Definición
API"	''' " " "

Término	Definición
Node.js"	''' " " " " " " " " ' " " " " "
Express"	''' " " " " " " " " " " " " " " " " " "
MySQL"	''' " " " " " " " " " " " " " " " " " "
Trigger"	''' "
Shadcn/ui"	''' "
Tailwind CSS"	''' "
Concurrently"	''' "
Multer"	''' " " " " " " " " " " " " " " " " " "
Responsive"	''' "
OTP"	''' "
VPS"	''' "
phpMyAdmin"	''' " " " " " " " " " " " " " " " " " "
Cockpit"	''' "

Término	Definición
Pi-hole / DNS local"	''' " " "

- **Tabla resumen de pruebas realizadas y resultados**

ID	Resultado obtenido	¿Superada?
CP-01	Usuario creado correctamente con contraseña cifrada	Si
CP-02	Error 401 devuelto con mensaje de contraseña corta	Si
CP-03	Error 409 por email duplicado	Si
CP-04	Acceso directo al dashboard sin 2FA	Si
CP-05	Error 401 con credenciales incorrectas	Si
CP-06	Modal 2FA mostrado y correo enviado	Si
CP-07	Acceso concedido con código válido	Si
CP-08	Acceso denegado con código incorrecto o expirado	Si
CP-09	Categoría creada y mostrada correctamente	Si
CP-10	Categoría vacía o duplicada rechazada	Si
CP-11	Trigger bloquea el borrado de categoría con movimientos	Si
CP-12	Ingreso registrado y gráfico actualizado	Si
CP-13	Gasto registrado y balance actualizado	Si
CP-14	Importe negativo o cero bloqueado en frontend y backend	Si
CP-15	Movimiento editado correctamente	Si
CP-16	Movimiento eliminado de BD y frontend	Si
CP-17	Foto de perfil actualizada correctamente	Si
CP-18	Campo is_2fa actualizado en base de datos	Si
CP-19	Trigger elimina datos del usuario en cascada	Si

- **Licencias del proyecto**

El proyecto MrFinance utiliza tres niveles de licencia claramente diferenciados:

Código fuente — GNU AGPL v3 Garantiza la libertad de uso, estudio, modificación y distribución del código. Obliga a publicar cualquier mejora, incluso cuando el software se ofrece como servicio web, evitando que terceros privaticen el trabajo.

Documentación — CC BY-NC-SA 4.0 Permite compartir y adaptar la documentación siempre que se cite la autoría, no se haga con fines comerciales y se mantenga la misma licencia en las obras derivadas.

Marca y diseño — Todos los derechos reservados El nombre MrFinance, el logotipo, los diseños visuales y demás elementos de identidad de marca están reservados y no pueden ser utilizados sin autorización expresa del autor.

Los archivos correspondientes se encuentran en la raíz del repositorio bajo los nombres LICENSE, LICENSE-docs y README.md.