ANEXOS

Anexo A — `ejemplo_database_yml` (contenido de `config/database.yml` de ejemplo)

```
# PostgreSQL - development, test, production

development:
  adapter: postgresql
  encoding: unicode
  database: zona44_development
  pool: <%= ENV.fetch("RAILS_MAX_THREADS") { 5 } %>
  username: <%= ENV.fetch('DB_USERNAME') { 'zona44' } %>
  password: <%= ENV.fetch('DB_PASSWORD') { 'password' } %>
  host: <%= ENV.fetch('DB_HOST') { 'localhost' } %>
  port: <%= ENV.fetch('DB_PORT') { 5432 } %>

test:
  adapter: postgresql
  encoding: unicode
  database: zona44_test
  pool: 5
  username: <%= ENV.fetch('DB_USERNAME') { 'zona44' } %>
  password: <%= ENV.fetch('DB_PASSWORD') { 'password' } %>
  host: <%= ENV.fetch('DB_HOST') { 'localhost' } %>
  port: <%= ENV.fetch('DB_PORT') { 5432 } %>

production:
  adapter: postgresql
  encoding: unicode
  database: <%= ENV['DB_NAME'] %>
  pool: <%= ENV.fetch("RAILS_MAX_THREADS") { 5 } %>
  username: <%= ENV['DB_USERNAME'] %>
  password: <%= ENV['DB_PASSWORD'] %>
  host: <%= ENV['DB_HOST'] %>
  port: <%= ENV['DB_PORT'] %>
  # sslmode: require
```

--

Anexo B — `docker-compose.yml` (ejemplo para desarrollo con Postgres)

```
version: '3.8'
services:
  db:
    image: postgres:14
    restart: always
    environment:
      POSTGRES_USER: zona44
      POSTGRES_PASSWORD: secret
      POSTGRES_DB: zona44_development
    volumes:
      - db_data:/var/lib/postgresql/data
    ports:
      - "5432:5432"

  web:
    build: .
    command: bash -lc "bundle install && bundle exec rails server -b 0.0.0.0"
    ports:
      - "3000:3000"
    environment:
      DB_HOST: db
      DB_USERNAME: zona44
      DB_PASSWORD: secret
      DB_NAME: zona44_development
      RAILS_ENV: development
    depends_on:
      - db
    volumes:
      - .:/app

volumes:
  db_data:
```

--

Anexo C — `backup_restore_scripts.sh` (script de backup/restore para Postgres)

```
#!/bin/bash
# Script simple de backup/restore para PostgreSQL
# USO:
#   ./backup_restore_scripts.sh backup <db_name> <out_file>
#   ./backup_restore_scripts.sh restore <db_name> <dump_file>

PGHOST=${PGHOST:-"localhost"}
PGPORT=${PGPORT:-5432}
PGUSER=${PGUSER:-"zona44"}

case "$1" in
  backup)
```

```
    DB_NAME="$2"
    OUT_FILE="$3"
    if [ -z "$DB_NAME" ] || [ -z "$OUT_FILE" ]; then
      echo "Uso: $0 backup <db_name> <out_file>"
      exit 1
    fi
    echo "Realizando backup de $DB_NAME a $OUT_FILE"
    PGPASSWORD=${PGPASSWORD:-"$PGPASSWORD"} pg_dump -U "$PGUSER" -h
"$PGHOST" -p "$PGPORT" -F c -b -v -f "$OUT_FILE" "$DB_NAME"
    ;;
  restore)
    DB_NAME="$2"
    DUMP_FILE="$3"
    if [ -z "$DB_NAME" ] || [ -z "$DUMP_FILE" ]; then
      echo "Uso: $0 restore <db_name> <dump_file>"
      exit 1
    fi
    echo "Restaurando $DUMP_FILE a $DB_NAME"
    PGPASSWORD=${PGPASSWORD:-"$PGPASSWORD"} pg_restore -U "$PGUSER" -h
"$PGHOST" -p "$PGPORT" -d "$DB_NAME" -v "$DUMP_FILE"
    ;;
  *)
    echo "Comandos soportados: backup, restore"
    exit 1
    ;;
esac

# Permisos: chmod +x backup_restore_scripts.sh
# Recomendación: ejecutar desde usuario con permisos o usando sudo -u postgres si es
necesario.
```

--

Anexo D — `.env.example` (ejemplo de variables de entorno)

```
# Ejemplo .env (no commitear este archivo con credenciales reales)
# Copiar a .env y rellenar valores en cada entorno
DB_HOST=127.0.0.1
DB_PORT=5432
DB_USERNAME=zona44
DB_PASSWORD=secret_password
DB_NAME=zona44_production
RAILS_ENV=production
RAILS_MASTER_KEY=your_rails_master_key_here
```