INSTRUCCIONES PARA LA EJECUCIÓN DEL CÓDIGO Y DE LAS PRUEBAS

Autor: Carlos Lora Espinosa

Trabajo Fin de Grado: Curso de Adaptación al Grado de Informática – UNIR.

Índice

[ARCHIVOS PARA EJECUTAR LA APLICACIÓN 3](#_Toc202721294)

[INSTRUCCIONES PARA EJECUTAR LA APLICACIÓN 3](#_Toc202721295)

[Paso 1: Preparar el entorno en Visual Studio Code 3](#_Toc202721296)

[Paso 2: Ejecutar la aplicación: 3](#_Toc202721297)

[Paso 3: Probar la API 4](#_Toc202721298)

[Ejemplo de uso: 4](#_Toc202721299)

[ARCHIVOS PARA LAS PRUEBAS 4](#_Toc202721300)

## ARCHIVOS PARA EJECUTAR LA APLICACIÓN

* ***calculos.py*** - Lógica principal de presupuestación.
* ***api.py*** - API REST con FastAPI
* requirements.txt - Dependencias y configuración de Python (FastAPI, Uvicorn, Pydantic.
* models.py - Modelos de datos con Pydantic (usado por la API)
* service.py - Capa de servicios (usado por la API)

## INSTRUCCIONES PARA EJECUTAR LA APLICACIÓN

### Paso 1: Preparar el entorno en Visual Studio Code

1. Abrir el proyecto en Visual Studio Code: File → Open Folder → Seleccionar la carpeta del proyecto.
2. Crear un entorno virtual:

*```bash*

*# En la terminal de VS Code (Ctrl + `)*

*python -m venv venv*

*```*

1. Activar el entorno virtual:

*```bash*

*# En Windows:*

*venv\Scripts\activate*

*# En macOS/Linux:*

*source venv/bin/activate*

*```*

1. Instalar dependencias

*```bash*

*pip install -r requirements.txt*

*```*

### Paso 2: Ejecutar la aplicación:

1. Iniciar el servidor:

*```bash*

*python -m uvicorn api:app --reload --host 0.0.0.0 --port 8000*

*```*

1. Verificar que funciona:

   - Abrir navegador en: `http://localhost:8000/health`

   - Debería mostrar: `{"status": "ok"}`

### Paso 3: Probar la API

1. URL de la API: http://localhost:8000
2. Endpoint principal: POST <http://localhost:8000/presupuestovm>
3. Documentación automática:

 - Swagger UI: http://localhost:8000/docs

   - ReDoc: <http://localhost:8000/redoc>

### Ejemplo de uso:

1. Ir a http://localhost:8000/docs
2. Hacer clic en POST /presupuestovm
3. Hacer clic en "Try it out"
4. Pegar datos de ejemplo (ver `tests/datos\_validos.json`)
5. Hacer clic en "Execute"

## ARCHIVOS PARA LAS PRUEBAS

* **tests/test\_api.py** - Pruebas de endpoints de la API
* **tests/conftest.py** - Configuración de pytest
* **tests/datos\_validos.json** - Datos de prueba válidos
* **tests/datos\_invalidos.json** - Datos de prueba inválidos
* **test\_gestion\_calculos.py** - Pruebas de gestión de cálculos
* **test\_validacion\_calculos.py** - Pruebas de validación de datos

## INSTRUCCIONES PARA EJECUTAR LAS PRUEBAS

### Paso 1: Preparar el entorno (si no está preparado)

1. Activar el entorno virutal.

*```bash*

*# En Windows:*

*venv\Scripts\activate*

*# En macOS/Linux:*

*source venv/bin/activate*

*```*

2. Instalar dependencias (si no están instaladas)

*```bash*

*pip install -r requirements.txt*

*```*

### Paso 2: Ejecutar todas las pruebas

```bash

# Ejecutar todas las pruebas

pytest

# Ejecutar con más información

pytest -v

# Ejecutar con información detallada

pytest -vv

```

### Paso 3: Ejecutar pruebas específicas

```bash

# Solo pruebas de la API

pytest tests/

# Solo pruebas de cálculos

pytest "test calculos\_py/"

# Prueba específica

pytest tests/test\_api.py::test\_health\_check

# Pruebas con un patrón específico

pytest -k "validar"

```