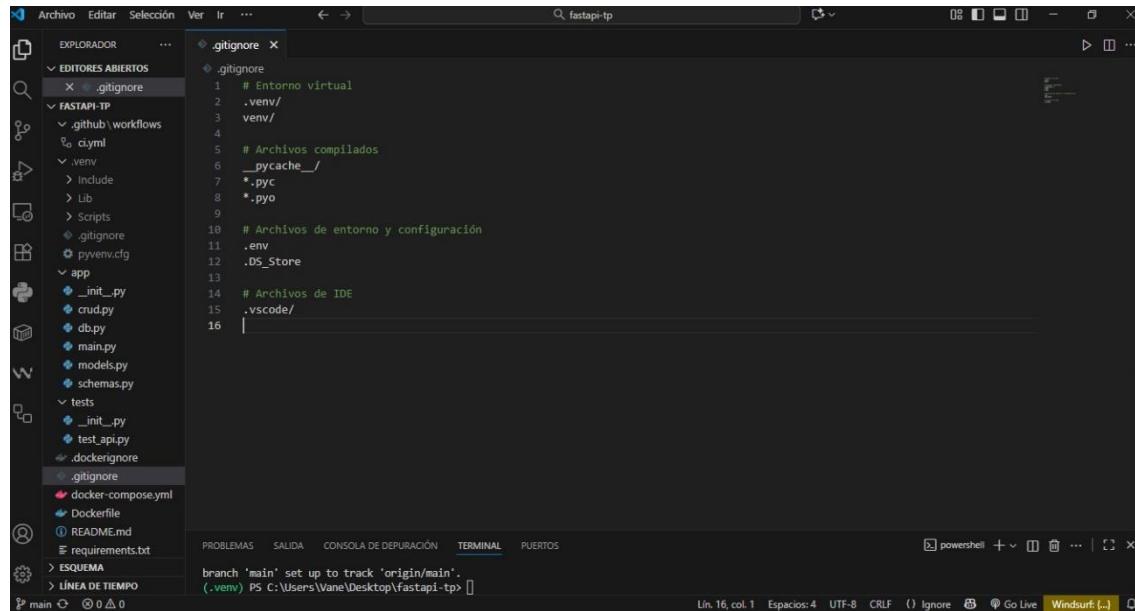


Trabajo Final Obligatorio

Evidencia de ejecución

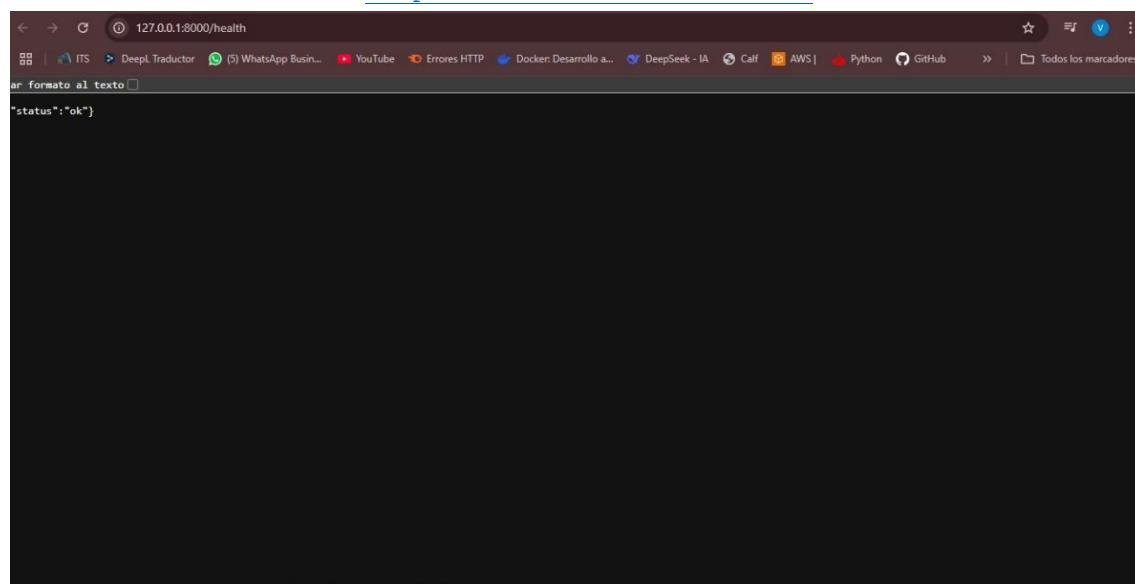
- Estructura inicial del proyecto



```
.gitignore
.
.
.
# Entorno virtual
.venv/
venv/
# Archivos compilados
__pycache__/
*.pyc
*.pyo
# Archivos de entorno y configuración
.env
.DS_Store
# Archivos de IDE
.vscode/

```

- La API corriendo en <http://localhost:8000/health>



```
127.0.0.1:8000/health
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Content-Length: 14
Date: Mon, 10 Jul 2023 14:45:43 GMT
Connection: keep-alive
Keep-Alive: timeout=5
{"status": "ok"}
```

Trabajo Final Obligatorio

Y en la terminal:

The screenshot shows the VS Code interface with the following details:

- Explorador:** Shows the project structure: .gitignore, main.py, .github, .venv, app, tests, Dockerfile, and requirements.txt.
- Editor:** Displays main.py with code for a FastAPI application.
- Terminal:** Shows the command line output of running the application with Uvicorn and performing a GET request to the /health endpoint.

```
from fastapi import FastAPI
app = FastAPI()
@app.get("/health")
def health_check():
    return {"status": "ok"}  
INFO: 127.0.0.1:8093 - "GET /health HTTP/1.1" 200 OK  
INFO: 127.0.0.1:8093 - "GET /favicon.ico HTTP/1.1" 404 Not Found
```

- Se crea database y el archivo products.db

The screenshot shows the VS Code interface with the following details:

- Explorador:** Shows the project structure including database.py, models.py, and schemas.py.
- Editor:** Displays main.py with code for creating tables in the database.
- Terminal:** Shows the command line output of installing psycopg2-binary and starting the application with Uvicorn.

```
from fastapi import FastAPI
from app.database import engine, Base
Base.metadata.create_all(bind=engine)
app = FastAPI()  
INFO: Will watch for changes in these directories: ['C:\\Users\\Vane\\Desktop\\Fastapi-tp']  
INFO: Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)  
INFO: Started reloader process [1868] using StatReload  
INFO: Started server process [6168]  
INFO: Waiting for application startup.  
INFO: Application startup complete.  
INFO: 127.0.0.1:8093 - "GET /health HTTP/1.1" 200 OK  
INFO: 127.0.0.1:8093 - "GET /favicon.ico HTTP/1.1" 404 Not Found
```

- Endpoints CRUD de Productos

The screenshot shows the VS Code interface with the following details:

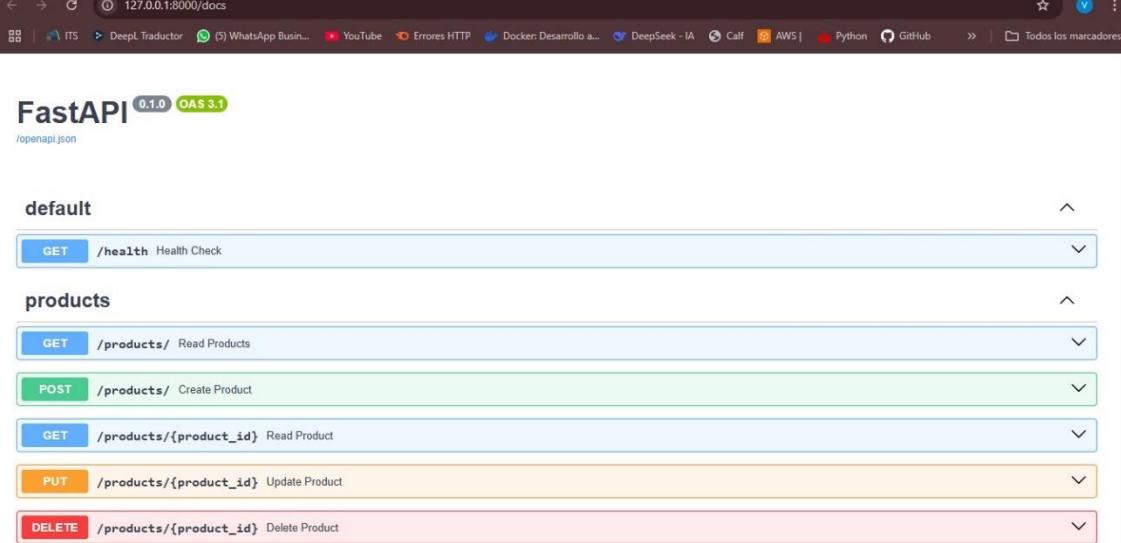
- Explorador:** Shows the project structure including routes, products.py, crud.py, database.py, db.py, main.py, and models.py.
- Editor:** Displays main.py with code for defining endpoints.
- Terminal:** Shows the command line output of starting the application with Uvicorn and performing GET requests to the /docs and /openapi.json endpoints.

```
INFO: Will watch for changes in these directories: ['C:\\Users\\Vane\\Desktop\\Fastapi-tp']  
INFO: Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)  
INFO: Started reloader process [2552] using StatReload  
INFO: Started server process [3088]  
INFO: Waiting for application startup.  
INFO: Application startup complete.  
INFO: 127.0.0.1:56859 - "GET /docs HTTP/1.1" 200 OK  
INFO: 127.0.0.1:56859 - "GET /openapi.json HTTP/1.1" 200 OK
```

Trabajo Final Obligatorio

- Registro de las rutas creadas CRUD en app/main.py

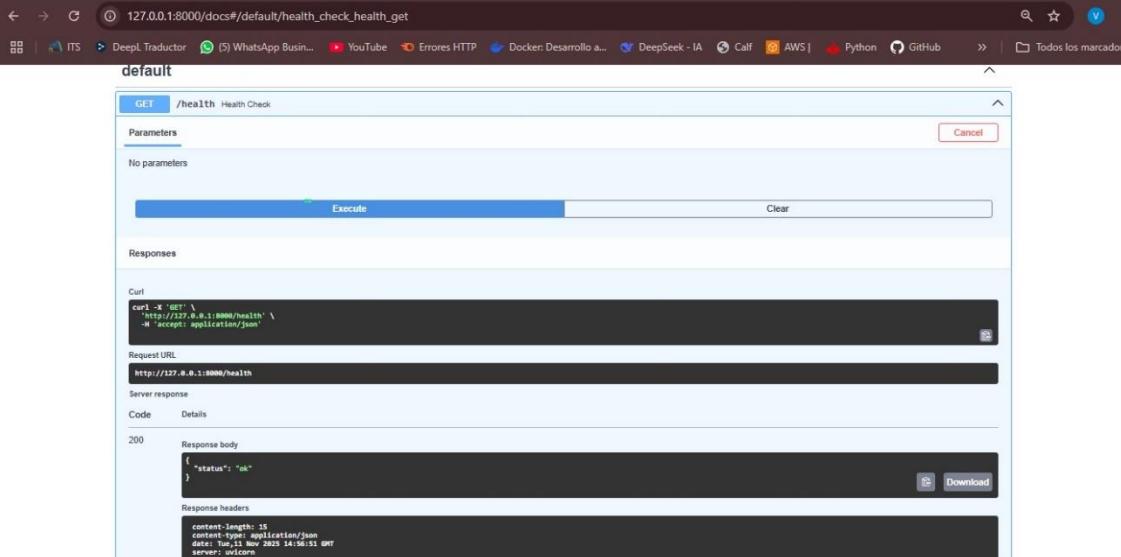
En <http://127.0.0.1:8000/docs>



The screenshot shows the FastAPI Swagger UI at <http://127.0.0.1:8000/docs>. The main title is "FastAPI 0.1.0 OAS 3.1". Below it, there's a link to "/openapi.json". The interface is organized into sections: "default" and "products". The "products" section contains five items: "GET /products/" (Read Products), "POST /products/" (Create Product), "GET /products/{product_id}" (Read Product), "PUT /products/{product_id}" (Update Product), and "DELETE /products/{product_id}" (Delete Product). Each item is represented by a colored button (blue for GET, green for POST, blue for GET, orange for PUT, red for DELETE) followed by the endpoint path.

Pruebas del CRUD de productos: la API funciona correctamente desde el navegador o Swagger UI. <http://127.0.0.1:8000/docs>

- GET /health — Health Check: devuelve un JSON simple que confirma que la API está viva

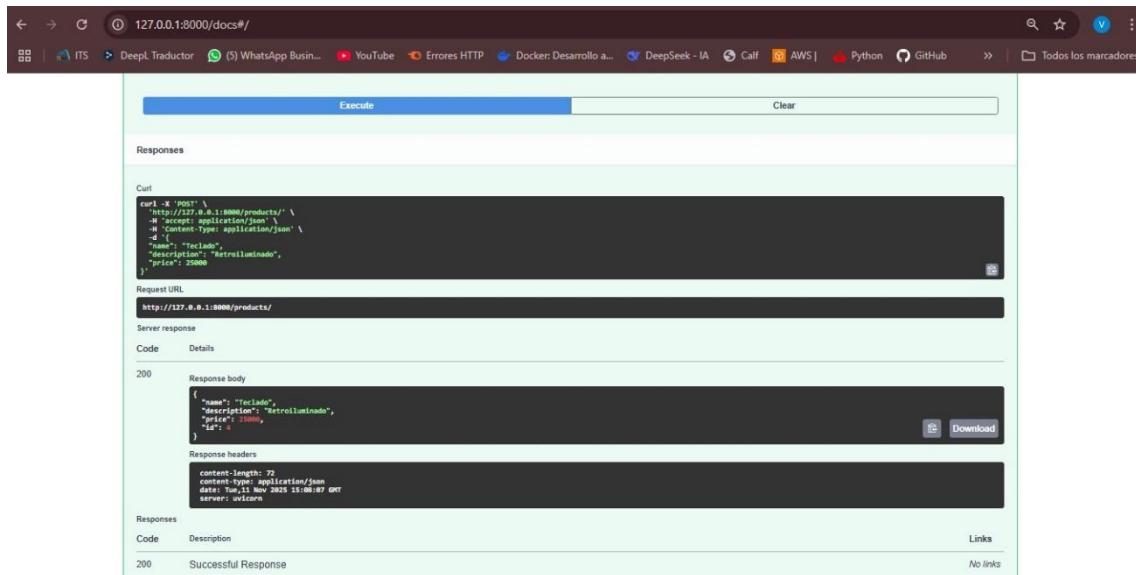


The screenshot shows the FastAPI Swagger UI at http://127.0.0.1:8000/docs#/default/health_check_health_get. The title is "default" and the endpoint is "GET /health Health Check". Under "Parameters", it says "No parameters". There are "Execute" and "Clear" buttons. Under "Responses", there's a "Curl" section with a command: "curl -X 'GET' \ 'http://127.0.0.1:8000/health' \ -H 'accept: application/json'", a "Request URL" field with "http://127.0.0.1:8000/health", a "Server response" section showing a 200 status code, and a "Code Details" section with a "Response body" containing {"status": "ok"} and "Response headers" including Content-Length: 15, Content-Type: application/json, Date: Tue, 11 Nov 2025 14:56:51 GMT, and Server: unicorn.

- POST /products/ — Crear producto: crea un nuevo registro en la base de datos.

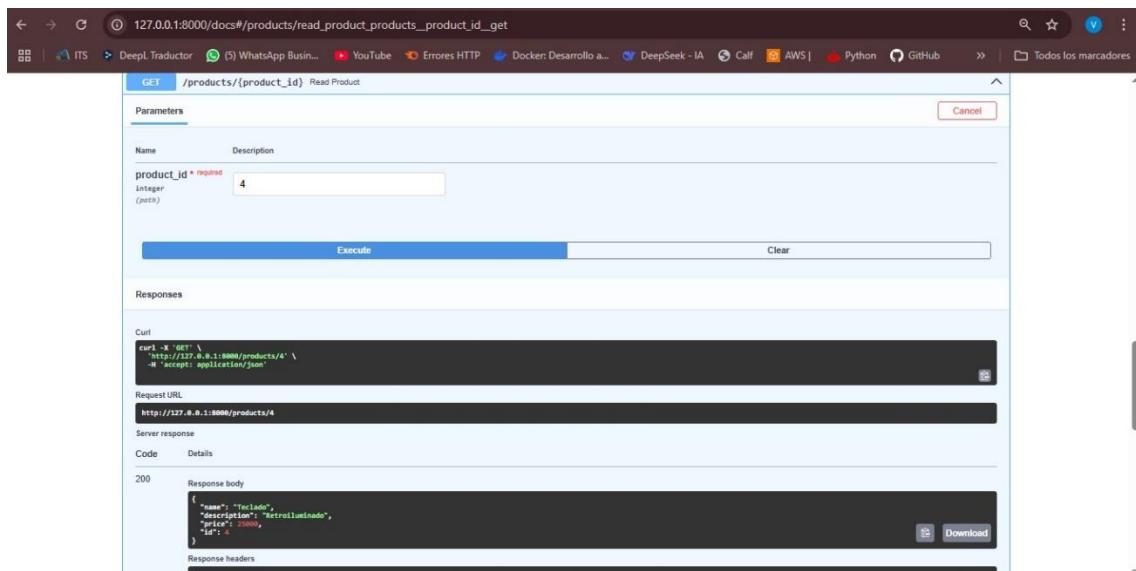
Trabajo Final Obligatorio

Resultado de post:



Curl
curl -X "POST" \
"http://127.0.0.1:8000/products/" \
-H "Content-Type: application/json" \
-d {
"name": "Teclado",
"description": "Retroiluminado",
"price": 25000
}
Request URL
<http://127.0.0.1:8000/products/>
Server response
Code Details
200 Response body
{
"name": "Teclado",
"description": "Retroiluminado",
"price": 25000,
"id": 4
}
Response headers
content-length: 72
content-type: application/json
date: Tue, 06 Nov 2025 13:06:07 GMT
server: uvicorn
Responses
Code Description Links
200 Successful Response
No links

- GET /products/{product_id} — Leer producto por id: devuelve el producto con el id dado. (en este caso el id 4, el producto creado antes)



GET /products/{product_id} Read Product
Parameters
Name Description
product_id * required Integer (path)
4
Execute Clear
Responses
Curl
curl -X "GET" \
"http://127.0.0.1:8000/products/4" \
-H "Accept: application/json"
Request URL
<http://127.0.0.1:8000/products/4>
Server response
Code Details
200 Response body
{
"name": "Teclado",
"description": "Retroiluminado",
"price": 25000,
"id": 4
}
Response headers

Trabajo Final Obligatorio

Ahora coloco un id de un producto inexistente para que devuelva error (id10, error 404)

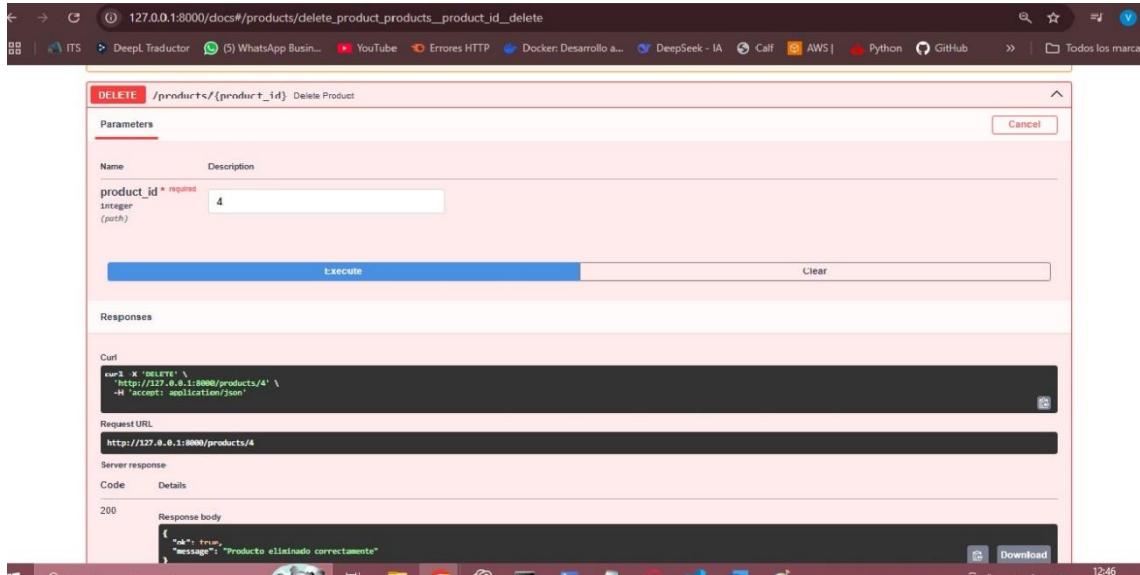
The screenshot shows a REST API documentation interface. A GET request is being made to `/products/{product_id}` with the parameter `product_id` set to `10`. The response details show a `404` error with the message `Error: Not Found` and a JSON response body: `{ "detail": "Producto no encontrado" }`.

- `PUT /products/{product_id}` — Actualizar producto: actualiza los campos del producto indicado por `product_id`.

The screenshot shows a REST API documentation interface. A PUT request is being made to `/products/{product_id}` with the parameter `product_id` set to `4`. The request body is a JSON object: `{"name": "Televisor y mouse", "description": "Revolucionadas", "price": 3000, "id": 4}`. The response details show a `200` OK status with a JSON response body matching the request body.

Trabajo Final Obligatorio

- **DELETE /products/{product_id}** — *Eliminar producto:* borra el producto con ese id



- Post desde terminal:

```
(.venv) PS C:\Users\Vane\Desktop\fastapi-tp> Invoke-RestMethod -Uri "http://127.0.0.1:8000/products/" -Method POST -ContentType "application/json" -Body '{
  "name": "Mouse",
  "price": 1200,
  "description": "Ergonomico"
}'
```

name	description	price	id
Mouse	Ergonomico	1200,0	5

- Get desde terminal: muestra los distintos productos creados como prueba desde el navegador o Swagger UI y los creados desde terminal

```
name          description          price id
---          -----
Teclado Retroiluminado Teclado retroiluminado RGB con switches rojos 2500,0 1
Teclado Retroiluminado Teclado retroiluminado RGB con switches rojos 2500,0 2
string        string              0,0   3
Mouse         Retroiluminado      10000,0 4
Mouse         Ergonomico         1200,0  5
```

```
(.venv) PS C:\Users\Vane\Desktop\fastapi-tp>
```

- PUT desde terminal:

```
(.venv) PS C:\Users\Vane\Desktop\fastapi-tp> Invoke-RestMethod -Uri "http://127.0.0.1:8000/products/1" -Method PUT -ContentType "application/json" -Body '{
  "name": "Mouse Ergonomico",
  "price": 1500,
  "description": "Version mejorada con sensor optico"
}'
```

name	description	price	id
Mouse Ergonomico	Version mejorada con sensor optico	1500,0	1

- DELETE desde terminal:

```
(.venv) PS C:\Users\Vane\Desktop\fastapi-tp> Invoke-RestMethod -Uri "http://127.0.0.1:8000/products/5" -Method DELETE
```

```
ok message
-----
True Producto eliminado correctamente
```

Trabajo Final Obligatorio

- Creo y pruebo tests

```
> pytest_cache
> .venv
> app
> tests
|> _pycache_
|> __init__.py
|> test_api.py
|> .dockerignore
|> .gitignore
=====
platform win32 -- Python 3.13.5, pytest-9.0.1, pluggy-1.6.0 -- C:\Users\VanE\Desktop\fastapi-tp\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: C:\Users\VanE\Desktop\fastapi-tp
plugins: anyio-4.11.0
collected 3 items

tests/test_api.py::test_health PASSED
tests/test_api.py::test_create_product PASSED
tests/test_api.py::test_get_products PASSED
```

- Creación del contenedor

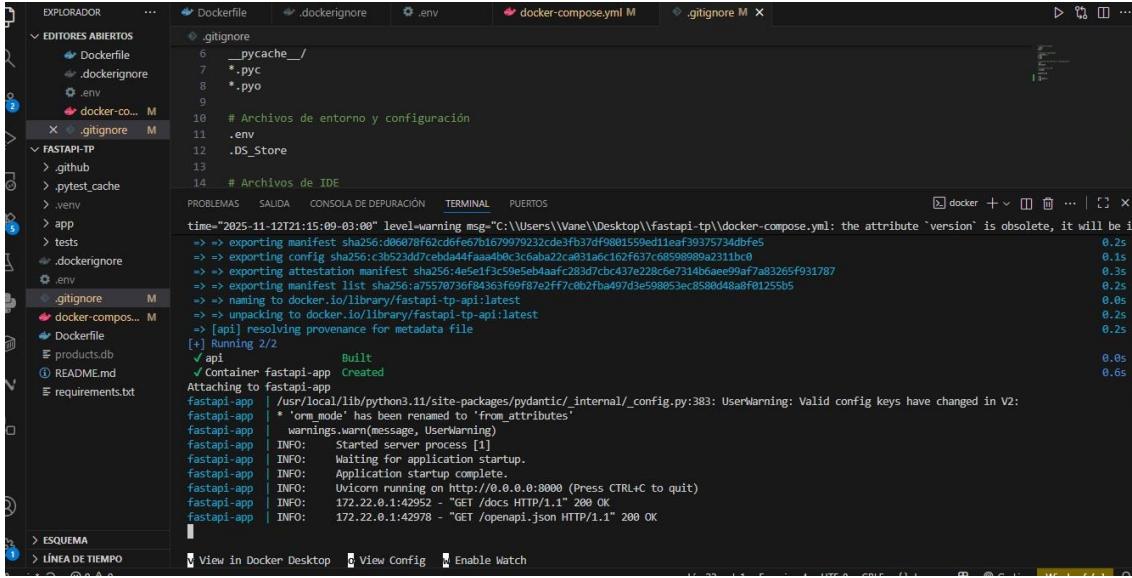
```
Archivo Editar Selección Ver Ir ...
Dockerfile .dockerignore x
.dockerngore
1 ovcache /
PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS
[+] Building 191.1s (13/13) FINISHED
--> >> transferring dockerfile: 821B
--> [internal] load metadata for docker.io/library/python:3.11-slim
--> [auth] library/python:pull token for registry-1.docker.io
--> [internal] load .dockerngore
--> >> transferring context: 188B
--> [internal] load build context: 18.12kB
--> [builder 1/5] FROM docker.io/library/python:3.11-slim@sha256:e4676722fb839e2e5cd8044a52
--> >> resolve docker.io/library/python:3.11-slim@sha256:e4676722fb839e2e5cd8044a52
--> >> sha256:658868b001a015sa1d3f5aa7f5a10ba02a7d55697301839dc047cd5d49670bc 248B /
--> >> sha256:f002d17b63f3f84a7f8a66f20cfa3aa4fc6d2a4069f05b6296b0abffcf2a8e1 14.36M
--> >> sha256:1ee9e106547f05aa3804dec2837c546439943d73d965a3f3f49f728db0be993 1.29M
--> >> sha256:d7ecded7702a5df6d6f79a71edc34b534d08f3051980e2c948fb72db3197fc 29.78M
--> >> extracting sha256:d7ecded7702a5df6d6f79a71edc34b534d08f3051980e2c948fb72db3197fc
--> > extracting sha256:1ee9e106547f05aa3804dec2837c546439943d73d965a3f3f49f728db0be993
--> > extracting sha256:f002d17b63f3f84a7f8a66f28cfca6iae4ffcd2a4069f985b6296b0abffcf2
--> > extracting sha256:658868b001a015sa1d3f5aa7f5a10ba02a7d55697301839dc047cd5d4967
--> [builder 2/5] WORKDIR /app
--> [builder 3/5] RUN apt-get update && apt-get install -y --no-install-recommends bu
--> [builder 4/5] COPY requirements.txt .
--> [stage-1 3/4] COPY .
--> [stage-1 4/4] COPY .
--> exporting to image
--> > exporting layers
--> > exporting manifest sha256:98cd50a1c190a2bf99ef4ff6f270a421d8aa341c5e2fcf22f
--> > exporting attestation manifest sha256:db6284239431ca29ee27571ce236f5512196c74a
--> > exporting manifest list sha256:80ff4451f41616ed4a94084781da062e3e0513a4ff0045
--> > naming to docker.io/library/fastapi-tp:latest
--> > unpacking to docker.io/library/fastapi-tp:latest
View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/v7n
(.venv) PS C:\Users\VanE\Desktop\fastapi-tp> docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
98685d956978 fastapi-tp:latest "uvicorn app.main:app..." 47 seconds ago Up 46 seconds 0.0.0.0:8000->8000/tcp fastapi-app
```

- Luego de resolver problemas de nombre (tenía otro contenedor con el mismo nombre) y de puerto (estaba siendo usado por Portainer) queda el contenedor corriendo

```
(.venv) PS C:\Users\VanE\Desktop\fastapi-tp> docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
98685d956978 fastapi-tp:latest "uvicorn app.main:app..." 47 seconds ago Up 46 seconds 0.0.0.0:8000->8000/tcp fastapi-app
```

Trabajo Final Obligatorio

- Dockercompose.yml



```
Dockerfile
.dockercfg
.dockerignore
.env
.docker-compose.yml
.gitignore

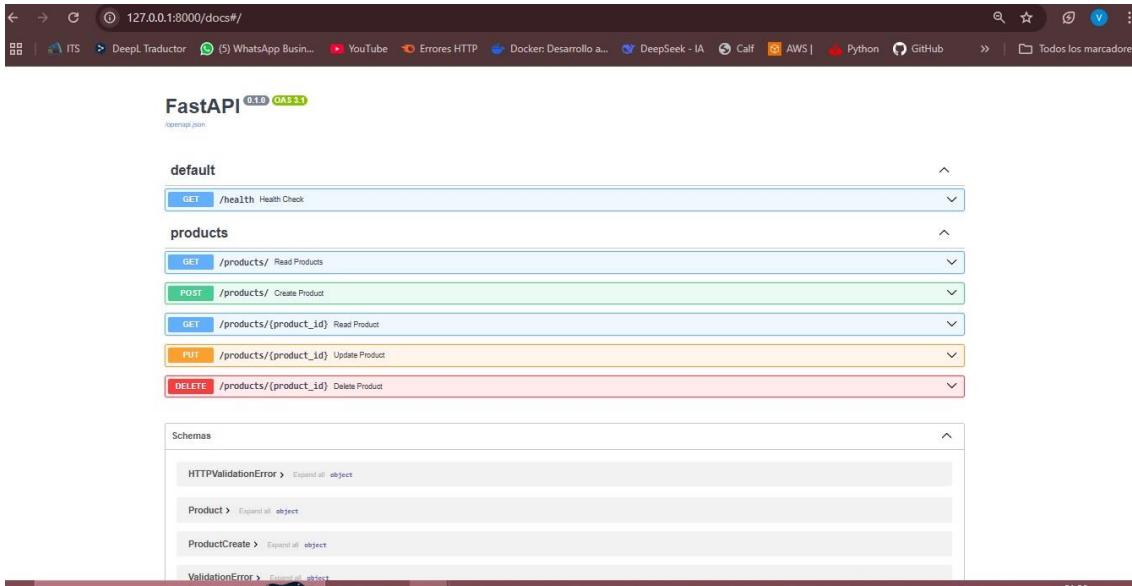
.gitignore
.pycache_/
*.pyc
*.pyo
*.py

# Archivos de entorno y configuración
.env
.DS_Store
# Archivos de IDE

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS
time="2025-11-12T21:15:09-03:00" level=warning msg="C:\Users\Vane\Desktop\fastapi-tp\\docker-compose.yml: the attribute `version` is obsolete, it will be ignored"
> => exporting manifest sha256:d0e607bf62cd6fe67b167997932cd3fb37df980159ed11eaef30375734dbfe5
=> => exporting config sha256:c30523dd7ebda4f4aa44b0c3c6abaa2ca031a6c162f637c60590969a2311bc0
=> => exporting attestation manifest sha256:4e5e1f3c595eb4aaafc283d7cbc437e228c6e7314b6aee09af7a83265f931787
=> => exporting manifest list sha256:a5570730fb4363f69f87e2ff7cd2fb9497dse5598659ca858d048a8f01255b5
=> => naming to docker.io/library/fastapi-tp-api:latest
=> => unpacking to docker.io/library/fastapi-tp-api:latest
=> [api] resolving provenance for metadata file
[+] Running 2/2
  ✓ api
  ✓ Container fastapi-app Created
Attaching to fastapi-app
fastapi-app  /usr/local/lib/python3.11/site-packages/pydantic/_internal/_config.py:383: UserWarning: Valid config keys have changed in V2:
fastapi-app    * 'orm_mode' has been renamed to 'from_attributes'
fastapi-app    | warnings.warn(message, UserWarning)
fastapi-app    INFO:     Started server process [1]
fastapi-app    INFO:     Waiting for application startup.
fastapi-app    INFO:     Application startup complete.
fastapi-app    INFO:     Uvicorn running on http://0.0.0.0:8000 (Press CTRL+C to quit)
fastapi-app    INFO:     172.22.0.1:42952 - "GET /docs HTTP/1.1" 200 OK
fastapi-app    INFO:     172.22.0.1:42978 - "GET /openapi.json HTTP/1.1" 200 OK
fastapi-app  | INFO:     172.22.0.1:42978 - "GET /openapi.json HTTP/1.1" 200 OK

View in Docker Desktop View Config Enable Watch
```

- <http://127.0.0.1:8000/docs> funcionando



FastAPI 0.1.0 (API 3.1)

openapi.json

default

GET /health Health Check

products

GET /products/ Read Products

POST /products/ Create Product

GET /products/{product_id} Read Product

PUT /products/{product_id} Update Product

DELETE /products/{product_id} Delete Product

Schemas

HTTPValidationError > Expand all object

Product > Expand all object

ProductCreate > Expand all object

ValidationError > Expand all object

Trabajo Final Obligatorio

- Agrego validaciones, manejo de errores claros y pruebas automáticas con pytest:
Validaciones en schemas.py, Creo un nuevo archivo de test:
tests/test_validations.py. Luego hice modificaciones en app/database.py y en
app/crud.py por warnings que se veían a la hora de ejecutar los tests:

The screenshot shows a code editor interface with several files open in the left sidebar under 'EDTORES ABIERTOS' (Editors Open). The files include .env, docker-compose.yml, .gitignore, schemas.py, database.py, crud.py, and __init__.py. In the main editor area, a file named crud.py is open, showing Python code for a CRUD API. Below the code editor is a terminal window titled 'PROBLEMAS' (Problems) which displays the output of a pytest run. The terminal shows the command 'PS C:\Users\Vane\Desktop\fastapi-tp> pytest -v' and the results of the test session, indicating 4 items passed in 2.75 seconds. The status bar at the bottom of the terminal shows the path 'C:\Users\Vane\Desktop\fastapi-tp', the command 'pytest -v', and the progress of the test execution.

Conclusión

El desarrollo de esta API sirvió para integrar conceptos fundamentales de FastAPI, manejo de base de datos con SQLAlchemy, validaciones con Pydantic y ejecución de pruebas automáticas con pytest.

Además, se trabajó con contenedores Docker para asegurar un entorno de ejecución consistente y portable, y se incorporó un flujo de Integración Continua con GitHub Actions.

El resultado final es una API funcional, testeada, contenedora y lista para desplegarse.

El repositorio incluye todo el código fuente, el archivo docker-compose.yml, el Dockerfile, los tests, y este documento con las capturas de los pasos más importantes del proceso.

Integración Continua:

Los tests automáticos se ejecutan en cada commit mediante GitHub Actions.

Link: <https://github.com/mevaktecnovane/fastapi-tp/actions>