

Міністерство освіти і науки України  
Національний технічний університет України „КПІ  
імені Ігоря Сікорського ”  
Факультет інформатики та обчислювальної техніки  
Кафедра інформатики і програмної інженерії

## **ЗВІТ**

лабораторної роботи №6  
з курсу «Основи WEB - технологій»

Тема: «Створення власного API на Python (FastAPI).»

Перевірив:

Викл. Альбрехт Й.О.

Виконала:

ст. Мельникова  
Катерина гр. ІС-12

Київ 2024

## Завдання1.

Створити web-додаток з використанням технології FastAPI, який буде працювати з інформацією відповідно вашому варіанту (Додаток 1. Таблиця 1)

### Хід роботи

Повний код проекту можна переглянути за посиланням:

<https://github.com/katerynamelnikova/web-technologies-labs>

Файл main.py:

```
from fastapi import FastAPI, HTTPException
from pymongo import MongoClient
from bson import ObjectId
from typing import List
from models.workermodels import Worker, WorkerInDB, WorkerUpdate

app = FastAPI()

client = MongoClient("mongodb://localhost:27017")
db = client["web-technologies-labs"]
collection = db["workers"]

def worker_serializer(worker) -> dict:
    return {
        "id": str(worker["_id"]),
        "name": worker["name"],
        "room": worker["room"],
        "department": worker["department"],
        "computer": worker["computer"],
    }

@app.get("/")
def read_root():
    return {"message": "This is a workers database API!"}

@app.get("/workers/", response_model=List[WorkerInDB])
def read_workers():
    workers = collection.find()
    return [worker_serializer(w) for w in workers]
```

```

@app.get("/workers/{worker_id}", response_model=WorkerInDB)
def read_worker(worker_id: str):
    worker = collection.find_one({"_id": ObjectId(worker_id)})
    if worker:
        return worker_serializer(worker)
    raise HTTPException(status_code=404, detail="Worker not found")


@app.post("/workers/", response_model=WorkerInDB)
def create_worker(worker: Worker):
    worker_id = collection.insert_one(worker.model_dump()).inserted_id
    created_worker = collection.find_one({"_id": worker_id})
    return worker_serializer(created_worker)


@app.patch("/workers/{worker_id}", response_model=WorkerInDB)
def update_worker(worker_id: str, worker: WorkerUpdate):
    update_data = {k: v for k, v in worker.model_dump().items() if v is not None}

    if not update_data:
        raise HTTPException(status_code=400, detail="No fields provided for update")

    result = collection.update_one(
        {"_id": ObjectId(worker_id)}, {"$set": update_data}
    )
    if result.modified_count == 1:
        updated_worker = collection.find_one({"_id": ObjectId(worker_id)})
        return worker_serializer(updated_worker)
    raise HTTPException(status_code=404, detail="Worker not found")


@app.delete("/workers/{worker_id}")
def delete_worker(worker_id: str):
    result = collection.delete_one({"_id": ObjectId(worker_id)})
    if result.deleted_count == 1:
        return {"message": "Worker deleted successfully"}
    raise HTTPException(status_code=404, detail="Worker not found")

```

## workermodels.py

```
from pydantic import BaseModel
from typing import Optional

class Worker(BaseModel):
    name: str
    room: str
    department: str
    computer: str

class WorkerInDB(Worker):
    id: str

class WorkerUpdate(BaseModel):
    name: Optional[str] = None
    room: Optional[str] = None
    department: Optional[str] = None
    computer: Optional[str] = None
```

## Отримані результати:

**FastAPI** 0.1.0 OAS 3.1  
/openapi.json

### default



GET	/	Read Root	▼
GET	/workers/	Read Workers	▼
POST	/workers/	Create Worker	▼
GET	/workers/{worker_id}	Read Worker	▼
PATCH	/workers/{worker_id}	Update Worker	▼
DELETE	/workers/{worker_id}	Delete Worker	▼

Curl

```
curl -X 'GET' \  
'http://localhost:8000/workers/' \  
-H 'accept: application/json'
```

Request URL

<http://localhost:8000/workers/>

Server response

Code	Details
------	---------

200

Response body

```
[  
  {  
    "name": "Мельник",  
    "room": "350",  
    "department": "IT",  
    "computer": "Macbook Pro",  
    "id": "66e3370050a82d9f91646a5a"  
  },  
  {  
    "name": "Іващенко",  
    "room": "100",  
    "department": "IT",  
    "computer": "Macbook Air",  
    "id": "66e33bd650a82d9f91646a63"  
  },  
  {  
    "name": "Мазур",  
    "room": "23",  
    "department": "HR",  
    "computer": "HP",  
    "id": "66e741765cc9685707ae8231"  
  }  
]
```



Download

Responses

Curl

```
curl -X 'POST' \  
'http://localhost:8000/workers/' \  
-H 'accept: application/json' \  
-H 'Content-Type: application/json' \  
-d '{  
  "name": "Іващук",  
  "room": "25b",  
  "department": "IT",  
  "computer": "Asus Vivobook Pro"  
}'
```

Request URL

<http://localhost:8000/workers/>

Server response

Code	Details
------	---------

200

Response body

```
{  
  "name": "Іващук",  
  "room": "25b",  
  "department": "IT",  
  "computer": "Asus Vivobook Pro",  
  "id": "66e824863517a13c6596af9d"  
}
```



Download

Curl

```
curl -X 'GET' \  
'http://localhost:8000/workers/66e824863517a13c6596af9d' \  
-H 'accept: application/json'
```

Request URL

<http://localhost:8000/workers/66e824863517a13c6596af9d>

Server response

Code	Details
------	---------

200

Response body

```
{  
  "name": "Іващук",  
  "room": "25b",  
  "department": "IT",  
  "computer": "Asus Vivobook Pro",  
  "id": "66e824863517a13c6596af9d"  
}
```



Download

Curl

```
curl -X 'PATCH' \
  'http://localhost:8000/workers/66e824863517a13c6596af9d' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "room": "25"
  }'
```

Request URL

http://localhost:8000/workers/66e824863517a13c6596af9d

Server response

Code	Details
200	<p>Response body</p> <pre>{   "name": "Іванчук",   "room": "25",   "department": "IT",   "computer": "Asus Vivobook Pro",   "id": "66e824863517a13c6596af9d" }</pre> <p> <a href="#">Download</a></p>


Curl

```
curl -X 'DELETE' \
  'http://localhost:8000/workers/66e741765cc9685707ae8231' \
  -H 'accept: application/json'
```

Request URL

http://localhost:8000/workers/66e741765cc9685707ae8231

Server response

Code	Details
200	<p>Response body</p> <pre>{   "message": "Worker deleted successfully" }</pre> <p> <a href="#">Download</a></p> <p>Response headers</p> <pre>content-length: 41 content-type: application/json date: Mon, 16 Sep 2024 12:31:03 GMT server: uvicorn</pre> <p>Responses</p>

## Висновок

Під час виконання даної лабораторної роботи, використовуючи технології FastApi, я створила API для роботи з даними працівників підприємства відповідно до варіанту та виконала дослідження створених endpoints за допомогою Postman та Swagger.