

JAN LEANDER MATUTES

BSIT - 3A

Module 3.1 Activity: Building a Full CRUD API with Pydantic

main.py

```
main.py  X
main.py > ...
1  from typing import List, Optional
2  from fastapi import FastAPI, HTTPException, Query, Path
3  from pydantic import BaseModel
4
5  app = FastAPI(title='Module3.1', version='1.0')
6
7  class StudentCreate(BaseModel):
8      name: str
9      course: str
10     gpa: Optional[float] = None
11
12    class StudentInDB(StudentCreate):
13        id: int
14
15    student_db: List[StudentInDB] = []
16    student_id_counter = 0
17
18    def _find_student_index(student_id: int) -> Optional[int]:
19        for idx, student in enumerate(student_db):
20            if student.id == student_id:
21                return idx
22        return None
23
24    @app.post("/students", response_model=StudentInDB, status_code=201)
25    def create_student(student: StudentCreate):
26        global student_id_counter
27        student_id_counter += 1
28        new_student = StudentInDB(id=student_id_counter, **student.dict())
29        student_db.append(new_student)
30        return new_student
31
32    @app.get("/students", response_model=List[StudentInDB])
33    def read_students(course: Optional[str] = Query(None, description="Filter by course")):
34        if course:
35            return [s for s in student_db if s.course.lower() == course.lower()]
36        return student_db
37
38    @app.get("/students/{student_id}", response_model=StudentInDB)
39    def read_student(student_id: int = Path(..., description="The ID of the student to retrieve")):
40        idx = _find_student_index(student_id)
41        if idx is None:
42            raise HTTPException(status_code=404, detail="Student not found")
43        return student_db[idx]
44
45    @app.put("/students/{student_id}", response_model=StudentInDB)
46    def update_student(student: StudentCreate, student_id: int = Path(..., description="The ID of the student to update")):
47        idx = _find_student_index(student_id)
48        if idx is None:
49            raise HTTPException(status_code=404, detail="Student not found")
50        updated = StudentInDB(id=student_id, **student.dict())
51        student_db[idx] = updated
52        return updated
53
54    @app.delete("/students/{student_id}")
55    def delete_student(student_id: int = Path(..., description="The ID of the student to delete")):
56        idx = _find_student_index(student_id)
57        if idx is None:
58            raise HTTPException(status_code=404, detail="Student not found")
59        student_db.pop(idx)
60        return {"message": "Student deleted successfully"}
```

Module3.1

1.0 OAS 3.1

/openapi.json

default

POST	/students	Create Student	^
GET	/students	Read Students	^
GET	/students/{student_id}	Read Student	^
PUT	/students/{student_id}	Update Student	^
DELETE	/students/{student_id}	Delete Student	^

1. POST: Create 2-3 different students (e.g., some in "BSIT" and some in "BSCS").

POST /students Create Student

Parameters

No parameters

Request body required

application/json

Edit Value | Schema

```
{  
    "name": "Arkin Gonzaga",  
    "course": "BSCS",  
    "gpa": 1.28  
}
```

Execute Clear

Responses

Curl

```
curl -X 'POST' \  
  'http://127.0.0.1:8000/students' \  
  -H 'accept: application/json' \  
  -H 'Content-Type: application/json' \  
  -d '{  
    "name": "Arkin Gonzaga",  
    "course": "BSCS",  
    "gpa": 1.28  
}'
```

Request URL

http://127.0.0.1:8000/students

Server response

Code Details

201 Response body

```
{  
    "name": "Arkin Gonzaga",  
    "course": "BSCS",  
    "gpa": 1.28,  
    "id": 3  
}
```

Download

Response headers

POST /students Create Student

Parameters

No parameters

Request body **required**

application/json

```
{ "name": "Charlene Angel Custodio", "course": "BSIT", "gpa": 1.00 }
```

Execute Clear

Responses

Curl

```
curl -X 'POST' \
  'http://127.0.0.1:8000/students' \
  -H 'Accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{ "name": "Charlene Angel Custodio", "course": "BSIT", "gpa": 1.00 }'
```

Request URL

http://127.0.0.1:8000/students

Server response

Code Details

201 Response body

```
{ "name": "Charlene Angel Custodio", "course": "BSIT", "gpa": 1.00, "id": 1 }
```

Download

Response headers

POST /students Create Student

Parameters

No parameters

Request body **required**

application/json

```
{ "name": "Jan Leander Matutes", "course": "BSIT", "gpa": 1.48 }
```

Execute Clear

Responses

Curl

```
curl -X 'POST' \
  'http://127.0.0.1:8000/students' \
  -H 'Accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{ "name": "Jan Leander Matutes", "course": "BSIT", "gpa": 1.48 }'
```

Request URL

http://127.0.0.1:8000/students

Server response

Code Details

201 Response body

```
{ "name": "Jan Leander Matutes", "course": "BSIT", "gpa": 1.48, "id": 1 }
```

Download

2. GET (All): Call the /students endpoint to see all students you created.

GET /students Read Students

Parameters

Name	Description
COURSE	Filter by course string (string null) (query)

Responses

Curl

```
curl -X 'GET' \
'http://127.0.0.1:8000/students' \
-H 'accept: application/json'
```

Request URL

```
http://127.0.0.1:8000/students
```

Server response

Code	Details
200	Response body

```
[  
  {  
    "name": "Jan Leander Matutes",  
    "course": "BSIT",  
    "gpa": 1.48,  
    "id": 1  
  },  
  {  
    "name": "Charlene Angel Custodio",  
    "course": "BSIT",  
    "gpa": 1.3,  
    "id": 2  
  },  
  {  
    "name": "Arkin Gonzaga",  
    "course": "BSCS",  
    "gpa": 1.2,  
    "id": 3  
  }]
```

Download

Response headers

```
content-length: 192  
content-type: application/json  
date: Sat, 25 Oct 2025 13:25:42 GMT  
server: unicorn
```

Responses

Code	Description
------	-------------

Links

3. GET (Query): Call the /students?course=BSIT endpoint to test the filtering.

GET /students Read Students

Parameters

Name	Description
COURSE	Filter by course string (string null) (query)

Responses

Curl

```
curl -X 'GET' \
'http://127.0.0.1:8000/students?course=BSIT' \
-H 'accept: application/json'
```

Request URL

```
http://127.0.0.1:8000/students?course=BSIT
```

Server response

Code	Details
200	Response body

```
[  
  {  
    "name": "Jan Leander Matutes",  
    "course": "BSIT",  
    "gpa": 1.48,  
    "id": 1  
  },  
  {  
    "name": "Charlene Angel Custodio",  
    "course": "BSIT",  
    "gpa": 1.3,  
    "id": 2  
  }]
```

Download

Response headers

```
content-length: 134  
content-type: application/json  
date: Sat, 25 Oct 2025 13:27:09 GMT  
server: unicorn
```

Responses

4. GET (One): Get one of the students by their ID (e.g., /students/1).

The screenshot shows a REST API tool interface for a GET request to the endpoint `/students/{student_id}`. The request parameters are set to `student_id: 2`. The responses section shows the curl command, request URL (`http://127.0.0.1:8000/students/2`), and the server response body and headers. The response body is a JSON object with name, course, gpa, and id fields. The response headers include content-length, content-type, date, and server.

Responses

Curl

```
curl -X 'GET' \
'http://127.0.0.1:8000/students/2' \
-H 'accept: application/json'
```

Request URL

`http://127.0.0.1:8000/students/2`

Server response

Code Details

200 Response body

```
{
  "name": "Charlene Angel Custodio",
  "course": "BSIT",
  "gpa": 3,
  "id": 2
}
```

Response headers

```
content-length: 67
content-type: application/json
date: Sat, 25 Oct 2025 13:27:42 GMT
server: unicorn
```

Responses

Code Description Links

200 Successful Response No links

5. PUT: Update one of the students (e.g., change the gpa or course for /students/1).

The screenshot shows a REST API tool interface for a PUT request to the endpoint `/students/{student_id}`. The request parameters are set to `student_id: 1`. The request body is a JSON object with name, course, gpa, and id fields. The responses section shows the curl command, request URL (`http://127.0.0.1:8000/students/1`), and the server response body and headers. The response body is a JSON object with name, course, gpa, and id fields.

PUT `/students/{student_id}` Update Student 75%

Parameters

Name Description

student_id * required The ID of the student to update

integer (path)

1

Request body required

Edit Value | Schema

```
{
  "name": "Jan Leander Matutes",
  "course": "BSCS",
  "gpa": 1.00
}
```

Responses

Curl

```
curl -X 'PUT' \
'http://127.0.0.1:8000/students/1' \
-H 'accept: application/json' \
-H 'Content-Type: application/json' \
-d '{
  "name": "Jan Leander Matutes",
  "course": "BSCS",
  "gpa": 1.00
}'
```

Request URL

`http://127.0.0.1:8000/students/1`

Server response

Code Details

200 Response body

```
{
  "name": "Jan Leander Matutes",
  "course": "BSCS",
  "gpa": 1,
  "id": 1
}
```

6. DELETE: Delete one of the students (e.g., /students/2).

The screenshot shows a REST API tool interface for a DELETE request to the endpoint `/students/{student_id}`. The student ID is set to 3. The tool includes sections for Parameters, Responses, and a detailed view of the response body and headers.

Parameters

Name	Description
<code>student_id</code> * required	The ID of the student to delete

Responses

Curl

```
curl -X 'DELETE' \
'http://127.0.0.1:8000/students/3' \
-H 'accept: application/json'
```

Request URL

```
http://127.0.0.1:8000/students/3
```

Server response

Code	Details
200	Response body { "message": "Student deleted successfully" } Response headers content-length: 42 content-type: application/json date: Sat, 25 Oct 2025 13:29:48 GMT server: uvicorn

Responses

Code	Description	Links
200	Successful Response	No links

Media type: application/json
Controls Accept header.
Example Value | Schema
"string"

7. GET (Invalid): Try to GET, PUT, or DELETE a student that doesn't exist (e.g., /students/99) to confirm you get a 404 "Not Found" error.

The screenshot shows a REST API tool interface for a PUT request to the endpoint `/students/{student_id}`. The student ID is set to 99. The tool includes sections for Parameters, Requests, and a detailed view of the response body and headers.

PUT /students/{student_id} Update Student

Parameters

Name	Description
<code>student_id</code> * required	The ID of the student to update

Request body application/json

Edit Value | Schema

```
{
  "name": "Jan Leander Matutes",
  "course": "BSCS",
  "gpa": 1.00
}
```

Responses

Curl

```
curl -X 'PUT' \
'http://127.0.0.1:8000/students/99' \
-H 'Content-Type: application/json' \
-d '{
  "name": "Jan Leander Matutes",
  "course": "BSCS",
  "gpa": 1.00
}'
```

Request URL

```
http://127.0.0.1:8000/students/99
```

Server response

Code	Details
404	Error: Not Found Response body { "detail": "Student not found" }

GET /students/{student_id} Read Student

Cancel

Parameters

Name	Description
student_id <small>* required</small>	The ID of the student to retrieve

integer
(path)

99

Execute Clear

Responses

Curl

```
curl -X 'GET' \
'http://127.0.0.1:8000/students/99' \
-H 'accept: application/json'
```

Request URL

http://127.0.0.1:8000/students/99

Server response

Code	Details	Links
404 <small>Undocumented</small>	Error: Not Found	

Response body

```
{ "detail": "Student not found" }
```

Download

Response headers

```
content-length: 38
content-type: application/json
date: Sat,25 Oct 2025 13:31:12 GMT
server: unicorn
```

Responses

Code	Description	Links
200	Successful Response	No links

Media type

application/json

Controls Accept header.

DELETE /students/{student_id} Delete Student

Cancel

Parameters

Name	Description
student_id <small>* required</small>	The ID of the student to delete

integer
(path)

99

Execute Clear

Responses

Curl

```
curl -X 'DELETE' \
'http://127.0.0.1:8000/students/99' \
-H 'accept: application/json'
```

Request URL

http://127.0.0.1:8000/students/99

Server response

Code	Details	Links
404 <small>Undocumented</small>	Error: Not Found	

Response body

```
{ "detail": "Student not found" }
```

Download

Response headers

```
content-length: 38
content-type: application/json
date: Sat,25 Oct 2025 13:32:04 GMT
server: unicorn
```

Responses

Code	Description	Links
200	Successful Response	No links

Media type

application/json

Controls Accept header.

Example Value | Schema