Laravel Student API & Docker

Faiza Tasnim

Course: Software Quality

Project: Student Management System

What is a REST API?

A **REST API** allows different computer systems to communicate using HTTP methods such as **GET**, **POST**, **PUT**, and **DELETE**.

Key Concepts

Client and Server

- Client sends request (browser, frontend app).
- Server processes request and returns JSON response.

Stateless Communication

- Each request contains all necessary info.
- Server does not store session info.

Resources

- Everything is a **resource** (user, student, etc.).

My Project: Student Management System

- Demonstrates CRUD operations for students.
- Uses Laravel REST API with **Sanctum Bearer Token** authentication.
- Tested with Postman, cURL.

API Security: Bearer Token

- Include token in request header:

```
Authorization: Bearer <your_token_here>
```

Laravel APIs Overview

- Admin and Student CRUD endpoints
- Token-protected routes
- Example endpoints and responses in the following slides

Example: Get All Students

Endpoint: GET `/api/home`

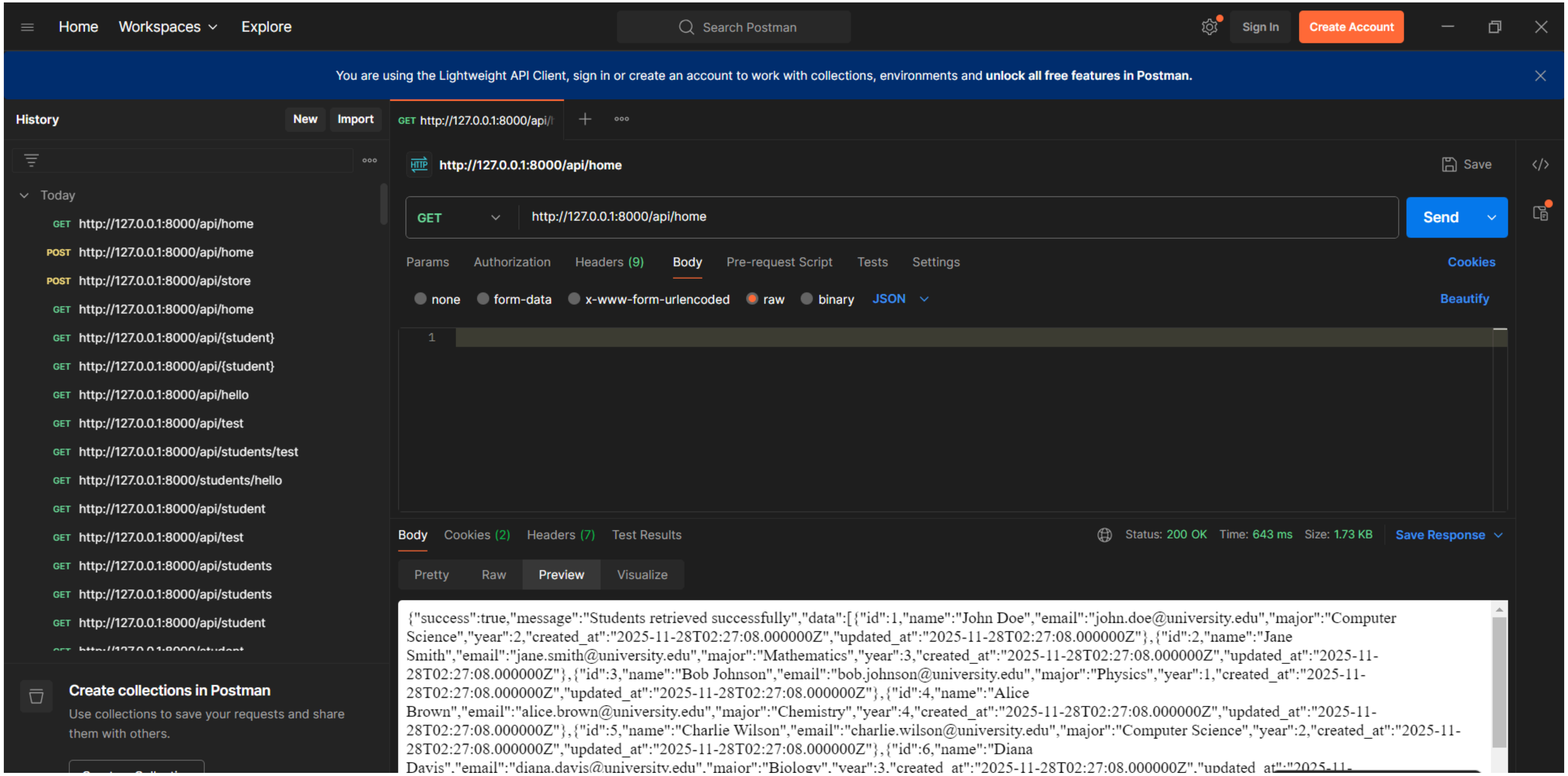
cURL:

```
curl -H "Authorization: Bearer <token>" http://localhost:8000/api/home
```

Response:

```
[
  {"id":1,"name":"John Doe","email":"john@example.com","major":"CS","year":3},
  {"id":2,"name":"Jane Smith","email":"jane@example.com","major":"IT","year":2}
]
```

Here is a example of working api:



You are using the Lightweight API Client, sign in or create an account to work with collections, environments and **unlock all free features in Postman.**

History

Today

```
GET  http://127.0.0.1:8000/api/home
POST http://127.0.0.1:8000/api/home
POST http://127.0.0.1:8000/api/store
GET  http://127.0.0.1:8000/api/home
GET  http://127.0.0.1:8000/api/{student}
GET  http://127.0.0.1:8000/api/{student}
GET  http://127.0.0.1:8000/api/hello
GET  http://127.0.0.1:8000/api/test
GET  http://127.0.0.1:8000/api/students/test
GET  http://127.0.0.1:8000/students/hello
GET  http://127.0.0.1:8000/api/student
GET  http://127.0.0.1:8000/api/test
GET  http://127.0.0.1:8000/api/students
GET  http://127.0.0.1:8000/api/students
GET  http://127.0.0.1:8000/api/student
```

Create collections in Postman

Use collections to save your requests and share them with others.

http://127.0.0.1:8000/api/home

GET http://127.0.0.1:8000/api/home

Status: 200 OK Time: 643 ms Size: 1.73 KB

Body — Preview

```
{"success":true,"message":"Students retrieved successfully","data":[{"id":1,"name":"John Doe","email":"john.doe@university.edu","major":"Computer
Science","year":2,"created_at":"2025-11-28T02:27:08.000000Z","updated_at":"2025-11-28T02:27:08.000000Z"},{"id":2,"name":"Jane
Smith","email":"jane.smith@university.edu","major":"Mathematics","year":3,"created_at":"2025-11-28T02:27:08.000000Z","updated_at":"2025-11-
28T02:27:08.000000Z"},{"id":3,"name":"Bob Johnson","email":"bob.johnson@university.edu","major":"Physics","year":1,"created_at":"2025-11-
28T02:27:08.000000Z","updated_at":"2025-11-28T02:27:08.000000Z"},{"id":4,"name":"Alice
Brown","email":"alice.brown@university.edu","major":"Chemistry","year":4,"created_at":"2025-11-28T02:27:08.000000Z","updated_at":"2025-11-
28T02:27:08.000000Z"},{"id":5,"name":"Charlie Wilson","email":"charlie.wilson@university.edu","major":"Computer Science","year":2,"created_at":"2025-11-
28T02:27:08.000000Z","updated_at":"2025-11-28T02:27:08.000000Z"},{"id":6,"name":"Diana
Davis","email":"diana.davis@university.edu","major":"Biology","year":3,"created_at":"2025-11-28T02:27:08.000000Z","updated_at":"2025-11-
```

Example: Create Student

Endpoint: POST `/api/store`

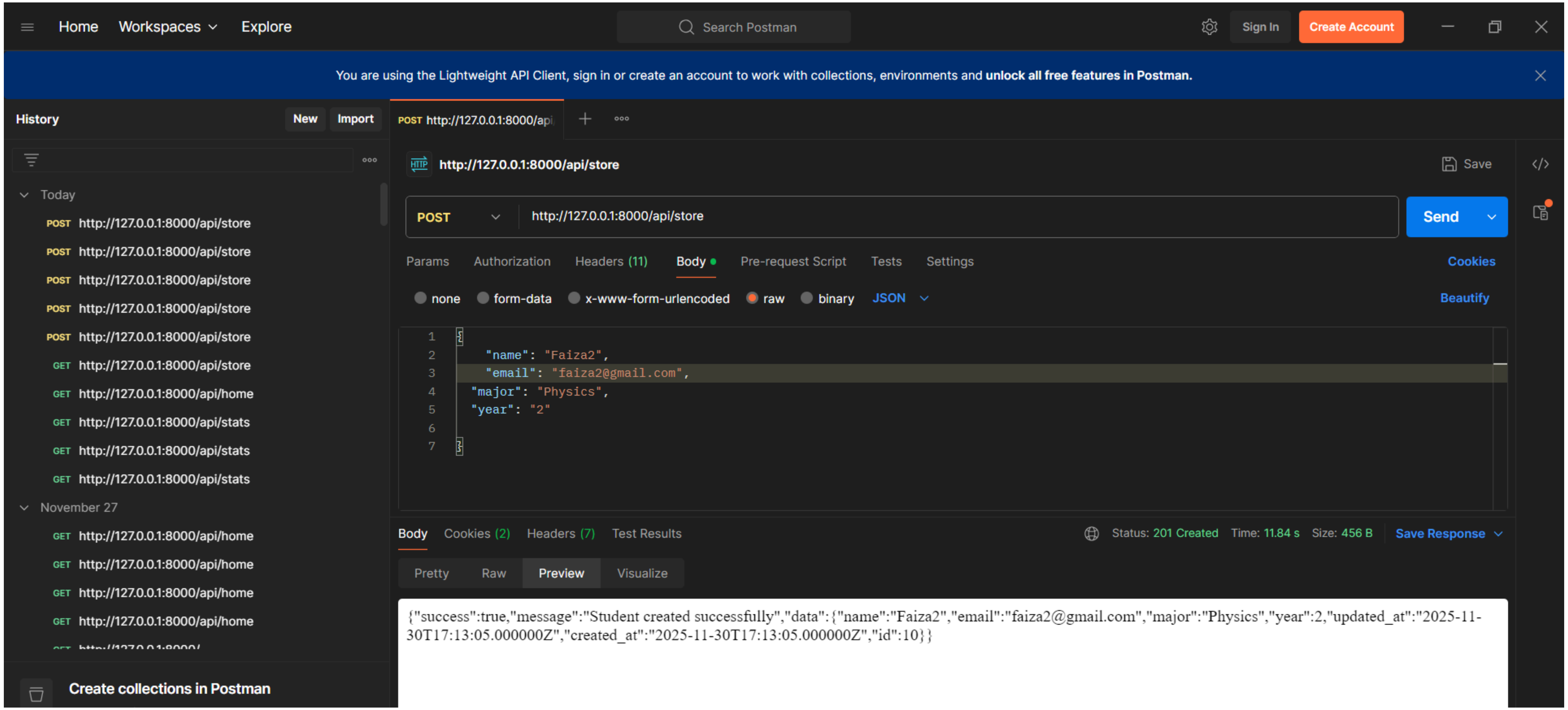
cURL:

```
curl -X POST -H "Authorization: Bearer <token>" -H "Content-Type: application/json" \
-d '{"name":"Alice","email":"alice@example.com","major":"CS","year":1}' \
http://localhost:8000/api/store
```

Response:

```
{
  "success":true,
  "message":"Student created successfully",
  "data":{"id":3,"name":"Alice","email":"alice@example.com","major":"CS","year":1}
}
```

Here is a example of working api with bearer token:



History:

- POST http://127.0.0.1:8000/api/store
- POST http://127.0.0.1:8000/api/store
- POST http://127.0.0.1:8000/api/store
- POST http://127.0.0.1:8000/api/store
- POST http://127.0.0.1:8000/api/store
- GET http://127.0.0.1:8000/api/store
- GET http://127.0.0.1:8000/api/home
- GET http://127.0.0.1:8000/api/stats
- GET http://127.0.0.1:8000/api/stats
- GET http://127.0.0.1:8000/api/stats
- GET http://127.0.0.1:8000/api/home
- GET http://127.0.0.1:8000/api/home
- GET http://127.0.0.1:8000/api/home
- GET http://127.0.0.1:8000/api/home

POST http://127.0.0.1:8000/api/store

Body: none / form-data / x-www-form-urlencoded / [✓ raw] / binary — JSON

```
{
    "name": "Faiza2",
    "email": "faiza2@gmail.com",
  "major": "Physics",
  "year": "2"

}
```

Body | Cookies (2) | Headers (7) | Test Results — Status: 201 Created Time: 11.84 s Size: 456 B

```
{"success":true,"message":"Student created successfully","data":{"name":"Faiza2","email":"faiza2@gmail.com","major":"Physics","year":2,"updated_at":"2025-11-30T17:13:05.000000Z","created_at":"2025-11-30T17:13:05.000000Z","id":10}}
```

Example: Update Student

Endpoint: PUT `/api/students/{id}`

cURL:

```
curl -X PUT -H "Authorization: Bearer <token>" -H "Content-Type: application/json" \
-d '{"major":"AI","year":2}' http://localhost:8000/api/students/3
```

Response:

```
{
  "success":true,
  "message":"Student updated successfully",
  "data":{"id":3,"name":"Alice","email":"alice@example.com","major":"AI","year":2}
}
```

Example: Delete Student

Endpoint: DELETE `/api/students/{id}`

cURL:

```
curl -X DELETE -H "Authorization: Bearer <token>" http://localhost:8000/api/students/3
```

Response:

```
{
  "success":true,
  "message":"Student deleted successfully"
}
```

Docker Containerization

- Laravel app container
- MySQL database container
- Docker Compose for orchestration
- Single command deployment using `run.sh`

Docker Compose Example

```yaml
services:
  app:
    build: .
    ports:
      - "8000:8000"
    depends_on:
      - db

  db:
    image: mysql:8.0
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: student_api
```

One-Command Deployment Script

run.sh:

```bash
#!/bin/bash

docker-compose up -d --build
php artisan migrate --seed
php artisan serve --host=0.0.0.0 --port=8000
```

API Testing Tools

- Postman for testing CRUD
- cURL for command line testing

Conclusion

- Laravel Student API implemented with Sanctum authentication
- Containerized with Docker