Testing Document

Introduction of Testing Tool:

- During the system development process, API testing plays an important role in ensuring that all functions work as expected. The team used **Thunder Client**, an API testing tool integrated into Visual Studio Code, to send HTTP requests (GET, POST, DELETE, etc.) and verify the system responses.
- Thunder Client provides a simple and effective way to test APIs by displaying status codes and response data clearly, helping the team evaluate the correctness of the implemented features.

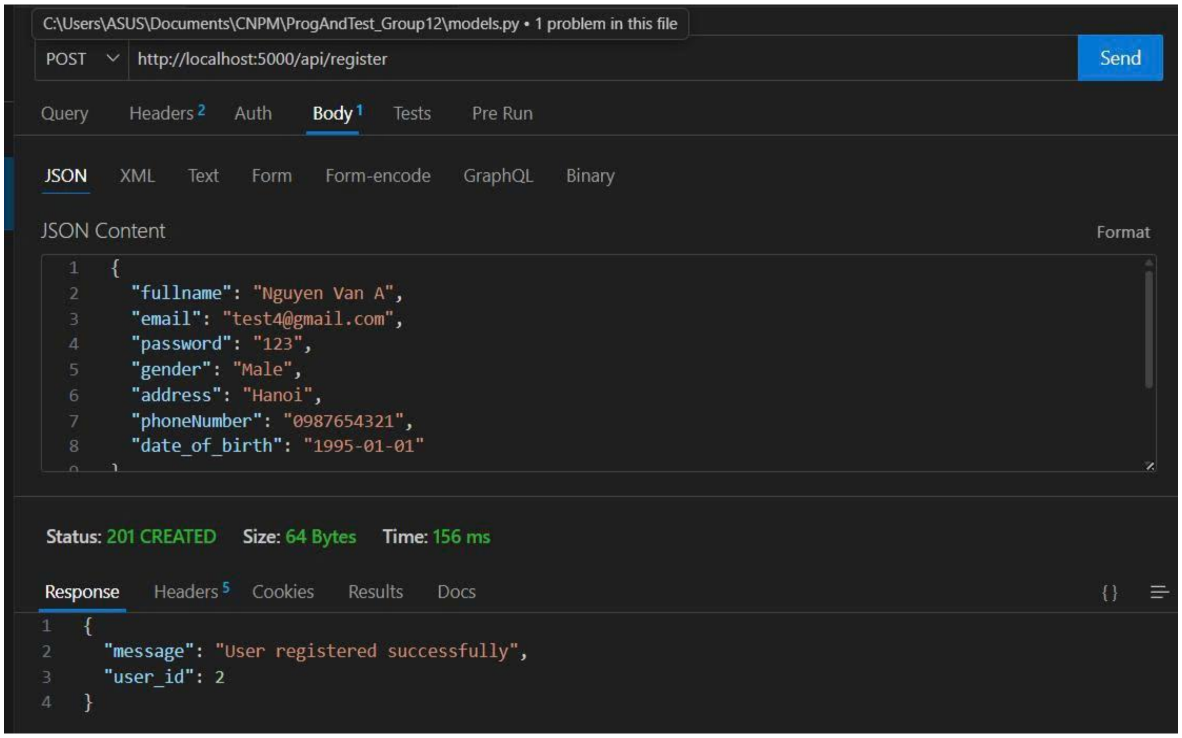
Test case:

Test Case ID	Function	Test Steps	Input Data	Expected Result
TC01	Register	1. Go to Register page 2. Enter valid information 3. Click Register	Full Name:Nguyen Van A Email: test@gmail.com Password: 123	User is registered successfully and a default AddressBook is created
TC02	Register	1. Go to Register page 2. Enter an email that already exists 3. Click Register	Full Name: Nguyen Van A Email: test@gmail.com Password: 123	Error message **"Email already exists"** is displayed
TC03	Login	1. Go to Login page 2. Enter valid username and password 3. Click Login	Username: student01 Password: 123456	User logs in successfully
TC04	Login	1. Go to Login page 2. Enter valid username and invalid password 3. Click Login	Username: student01 Password: wrongpass	Error message **"Invalid username or password"** is displayed
TC05	Add Contact	1. Login to the system 2. Go to Add Contact page 3. Enter valid contact information 4. Click Submit	Name: Nguyễn Văn A Email: vana@gmail.com Phone: 0912345678 Group IDs: [1,2]	New contact is added successfully and appears in the contact list
TC06	Edit Contact	1. Login to the system 2. Go to Contact Lis t3. Select a contact and click Edit 4. Update contact information	Name: An Phone: 0987654321 Email: an@gmail.com	Contact information is updated successfully

Test Case ID	Function	Test Steps	Input Data	Expected Result
		5. Click Save		
TC07	View Contact List	1. Login to the system 2. Navigate to Contact List page	AddressBook ID: 1	System returns a list of contacts successfully
TC08	View Contact List	1. Login to the system 2. Navigate to Contact List page	User account without any contacts	System displays an empty list or message **"No contacts found"**
TC09	Add Contact	1. Login to the system 2. Send POST request /api/books/2/contacts	Valid contact data	System returns error message: *AddressBook not found*
TC10	Add Contact to Group	1. Login to the system 2. Go to Add Contact page 3. Enter contact information 4. Select a group belonging to the same address book 5. Click Submit	Name: Nguyen Van A Email: vana@gmail.com Group: Friends	Contact is added successfully and assigned to the selected group
TC11	Add Contact to Group	1. Login to the system 2. Go to Add Contact page 3. Enter contact information 4. Select a group belonging to another address book 5. Click Submit	Name: David Group: Work (from another address book)	System displays an error message and contact is not added to the group

Test Case ID	Function	Test Steps	Input Data	Expected Result
TC12	Delete Contact	1. Login to the system 2. Send DELETE request /api/books/1/groups/1	Group ID: 01	Contact is deleted successfully
TC13		1. Login to the system 2. Send DELETE request /api/books/1/groups/1	Group ID: 1	

Evidence of using Thunder Client to test the code:



http://localhost:5000/api/register

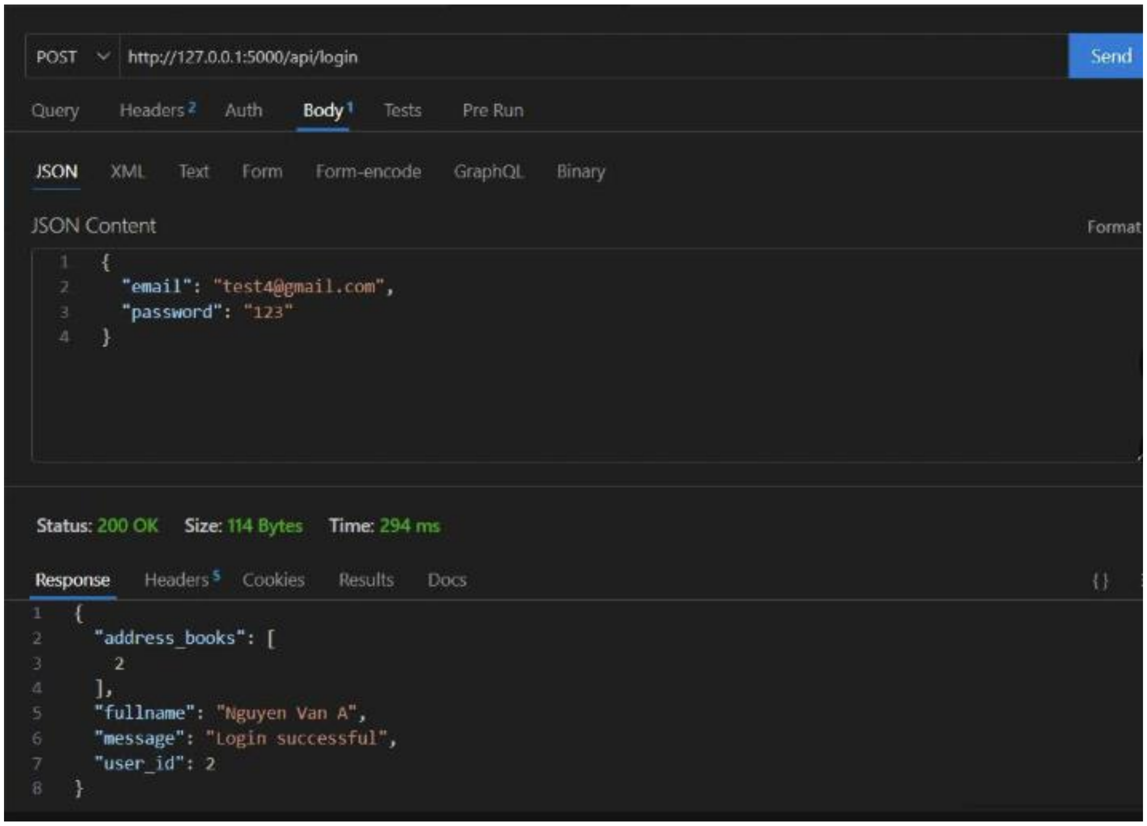
```
{
  "fullname": "Nguyen Van A",
  "email": "test4@gmail.com",
  "password": "123",
  "gender": "Male",
  "address": "Hanoi",
  "phoneNumber": "0987654321",
  "date_of_birth": "1995-01-01"
}
```

Status: 201 CREATED   Size: 64 Bytes   Time: 156 ms

```
{
  "message": "User registered successfully",
  "user_id": 2
}
```

- **The /api/register API works as expected**
- **Data is successfully sent and stored in the database**

- **The system automatically generates a user_id for the new user**



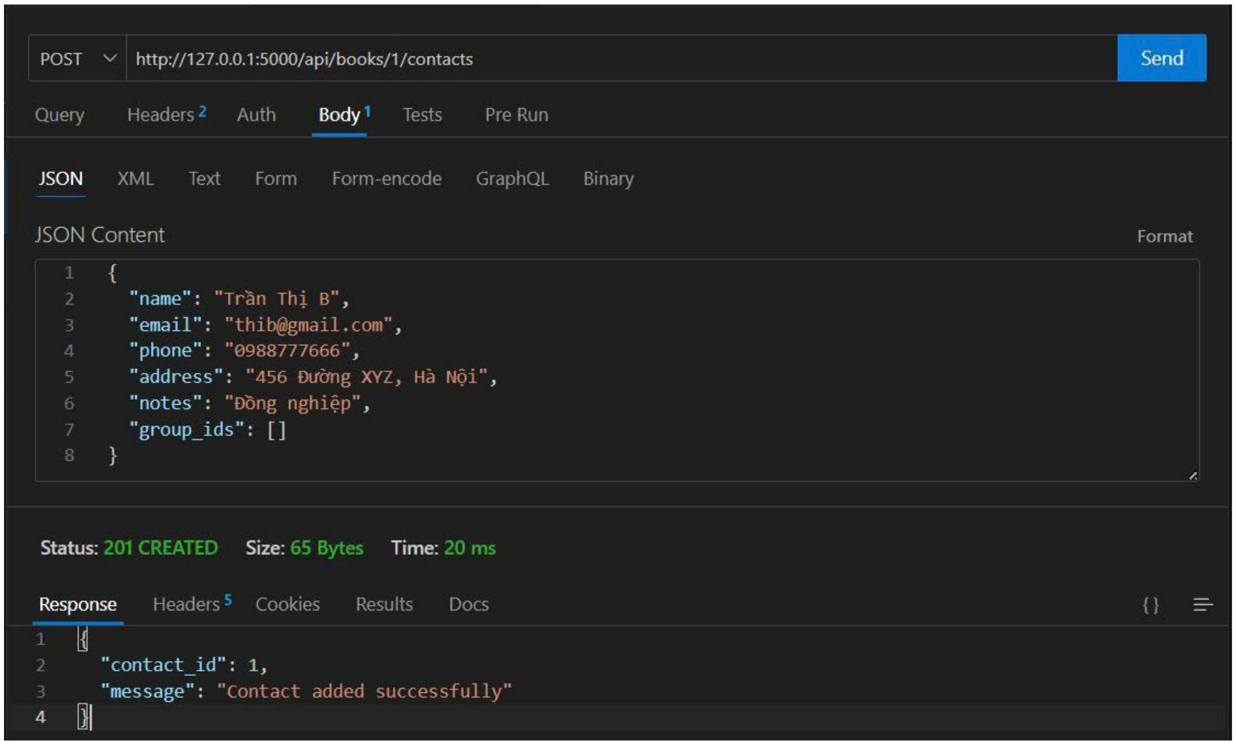
http://127.0.0.1:5000/api/login

```
{
  "email": "test4@gmail.com",
  "password": "123"
}
```

Status: 200 OK Size: 114 Bytes Time: 294 ms

```
{
  "address_books": [
    2
  ],
  "fullname": "Nguyen Van A",
  "message": "Login successful",
  "user_id": 2
}
```

- **Valid login information**
- **The system correctly verifies the email and password**
- **Returns user information and the associated AddressBook list**
- **The Login function works as expected\**



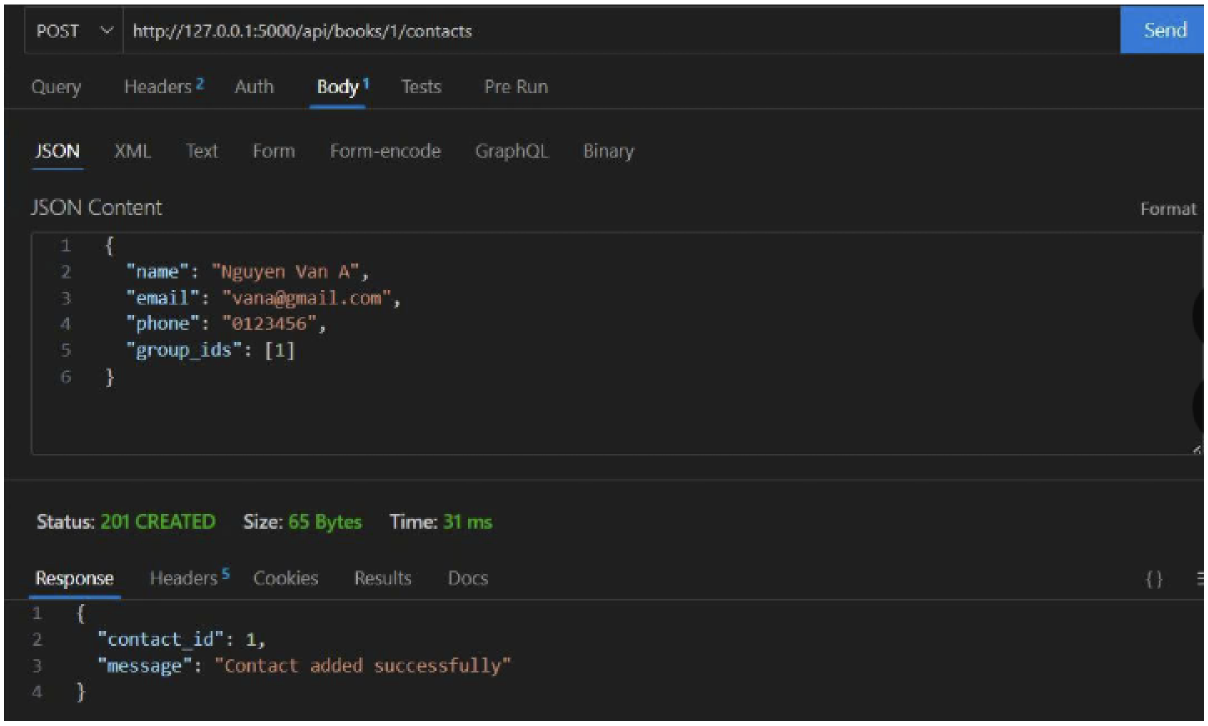
```
POST http://127.0.0.1:5000/api/books/1/contacts
```

```
{
  "name": "Trần Thị B",
  "email": "thib@gmail.com",
  "phone": "0988777666",
  "address": "456 Đường XYZ, Hà Nội",
  "notes": "Đồng nghiệp",
  "group_ids": []
}
```

Status: 201 CREATED Size: 65 Bytes Time: 20 ms

```
{
  "contact_id": 1,
  "message": "Contact added successfully"
}
```

- **API works correctly**
- **Contact is added successfully**
- **Data is stored in the database**
- **The system generates a contact_id automatically**



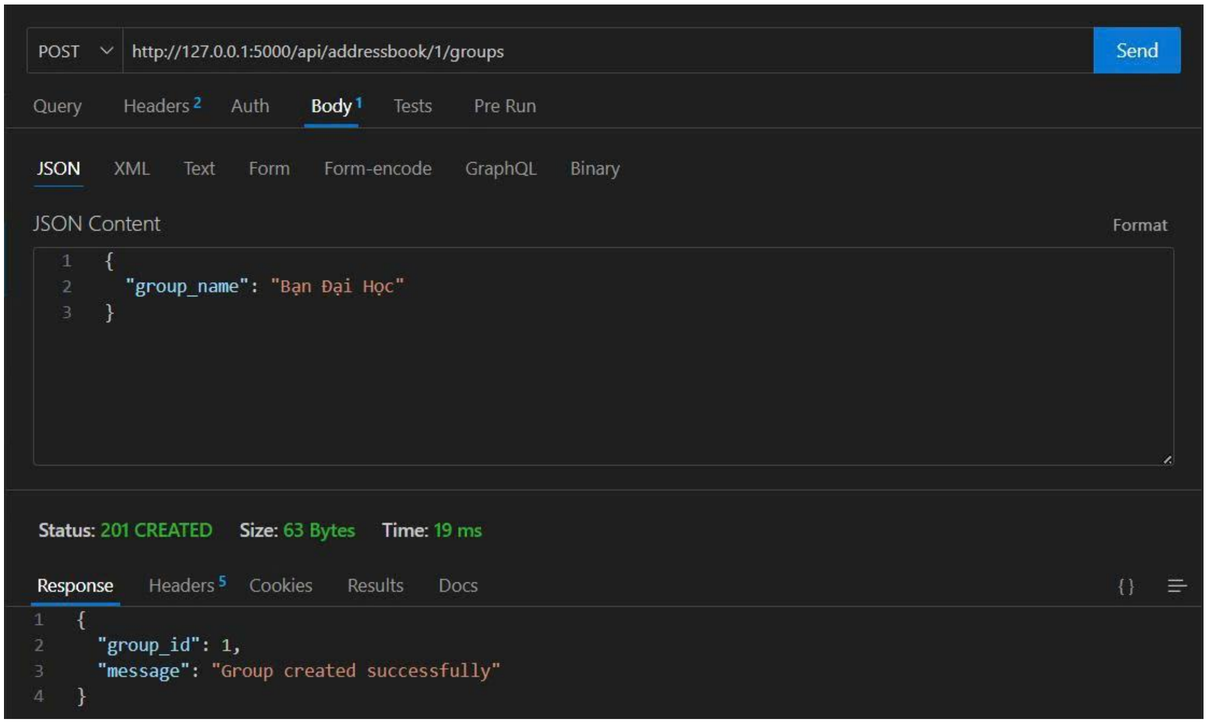
```
http://127.0.0.1:5000/api/books/1/contacts
```

```
{
  "name": "Nguyen Van A",
  "email": "vana@gmail.com",
  "phone": "0123456",
  "group_ids": [1]
}
```

Status: 201 CREATED   Size: 65 Bytes   Time: 31 ms

```
{
  "contact_id": 1,
  "message": "Contact added successfully"
}
```

- **Contact is added successfully to the specified group**
- **Contact information is processed correctly**
- **The system returns status 201 CREATED**
- **A contact_id is automatically generated**



```
http://127.0.0.1:5000/api/addressbook/1/groups
```

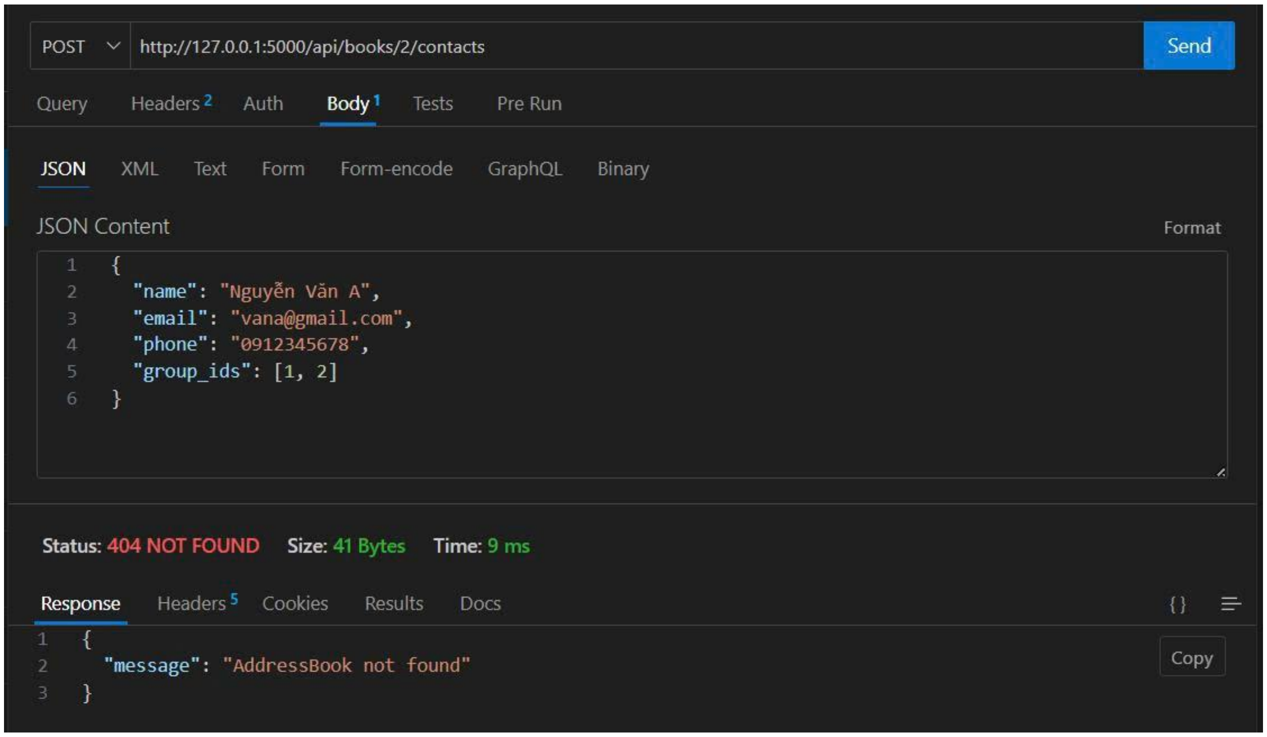
```
{
  "group_name": "Bạn Đại Học"
}
```

Status: 201 CREATED   Size: 63 Bytes   Time: 19 ms

```
{
  "group_id": 1,
  "message": "Group created successfully"
}
```

- **Group is created successfully**

- **Data is saved to the database**
- **The system returns status 201 CREATED**
- **A group_id is automatically generated**



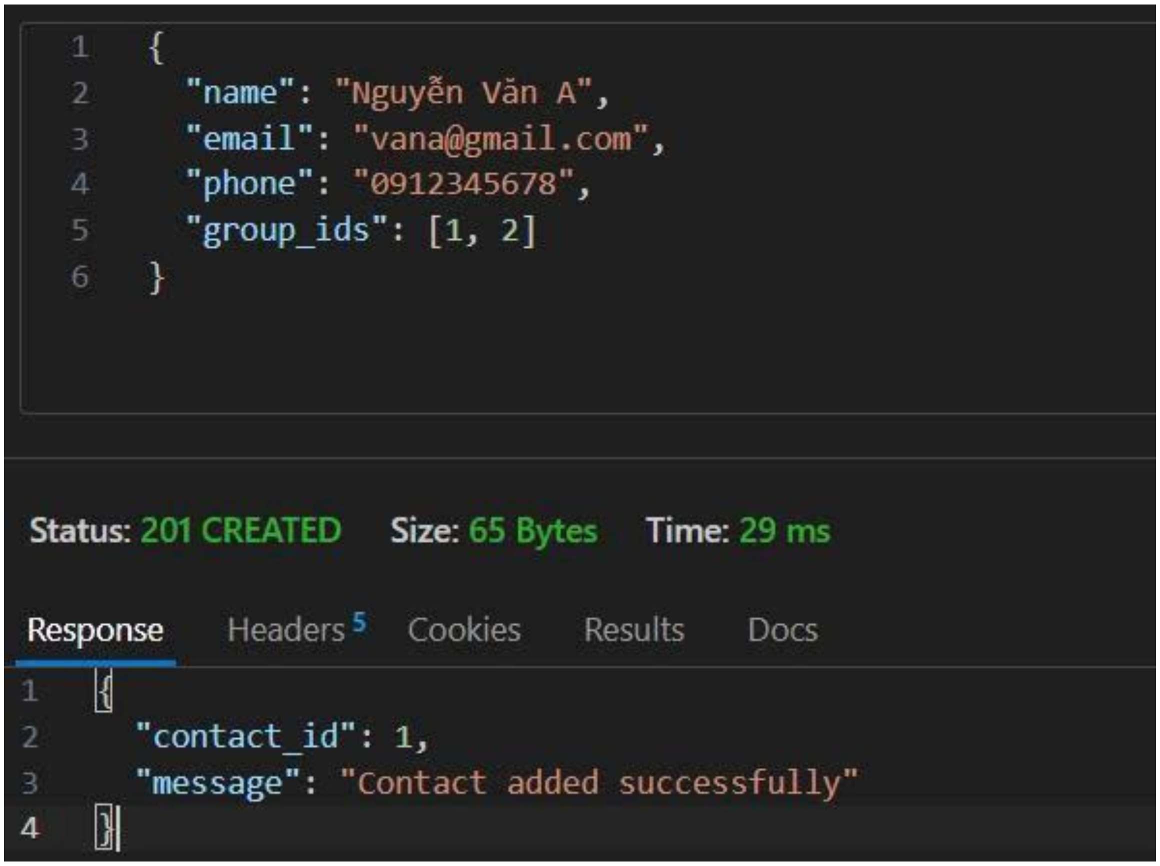
```
http://127.0.0.1:5000/api/books/2/contacts
```

```
{
  "name": "Nguyễn Văn A",
  "email": "vana@gmail.com",
  "phone": "0912345678",
  "group_ids": [1, 2]
}
```

Status: 404 NOT FOUND   Size: 41 Bytes   Time: 9 ms

```
{
  "message": "AddressBook not found"
}
```

- **The request failed because the AddressBook does not exist**
- **The system returns status 404 NOT FOUND**
- **Contact cannot be added to a non-existing AddressBook**

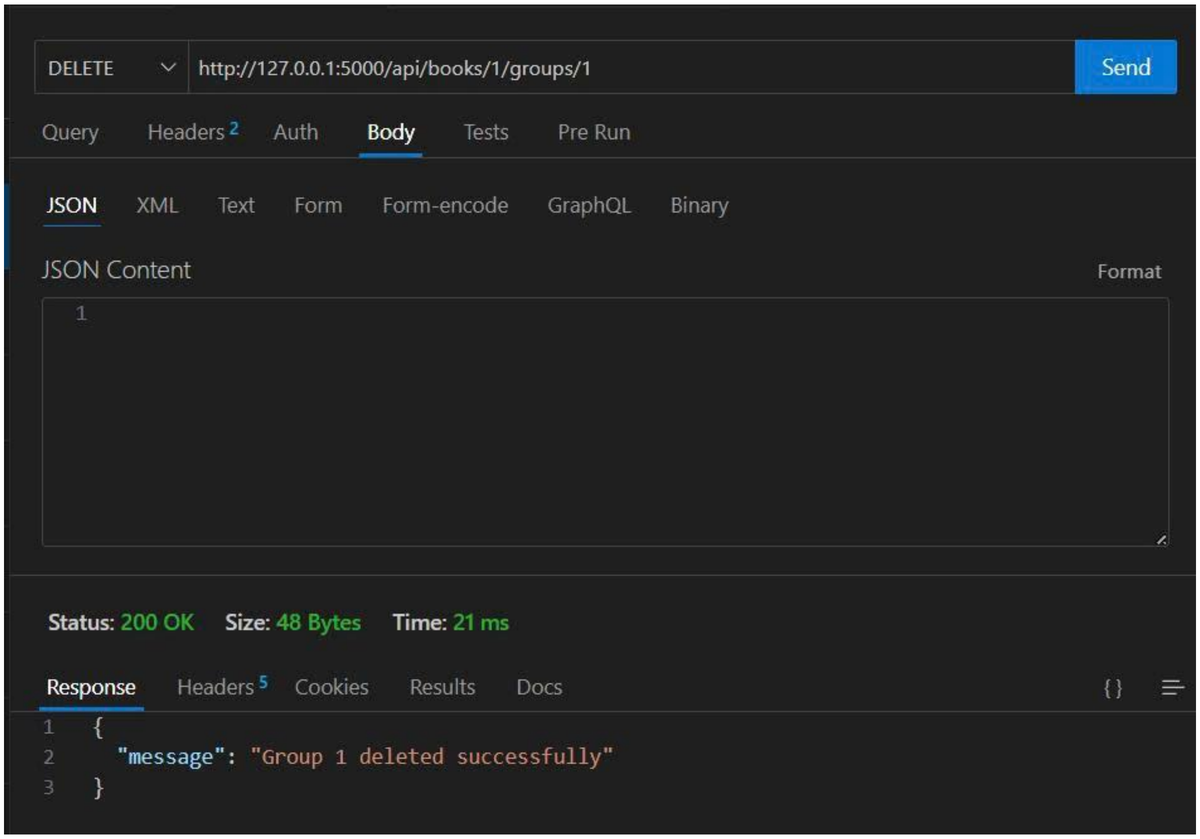


```
{
  "name": "Nguyễn Văn A",
  "email": "vana@gmail.com",
  "phone": "0912345678",
  "group_ids": [1, 2]
}
```

Status: 201 CREATED   Size: 65 Bytes   Time: 29 ms

```
{
  "contact_id": 1,
  "message": "Contact added successfully"
}
```

- **Contact information is valid**
- **The system successfully adds the contact to the AddressBook**
- **The API returns status 201 CREATED with a new contact_id**

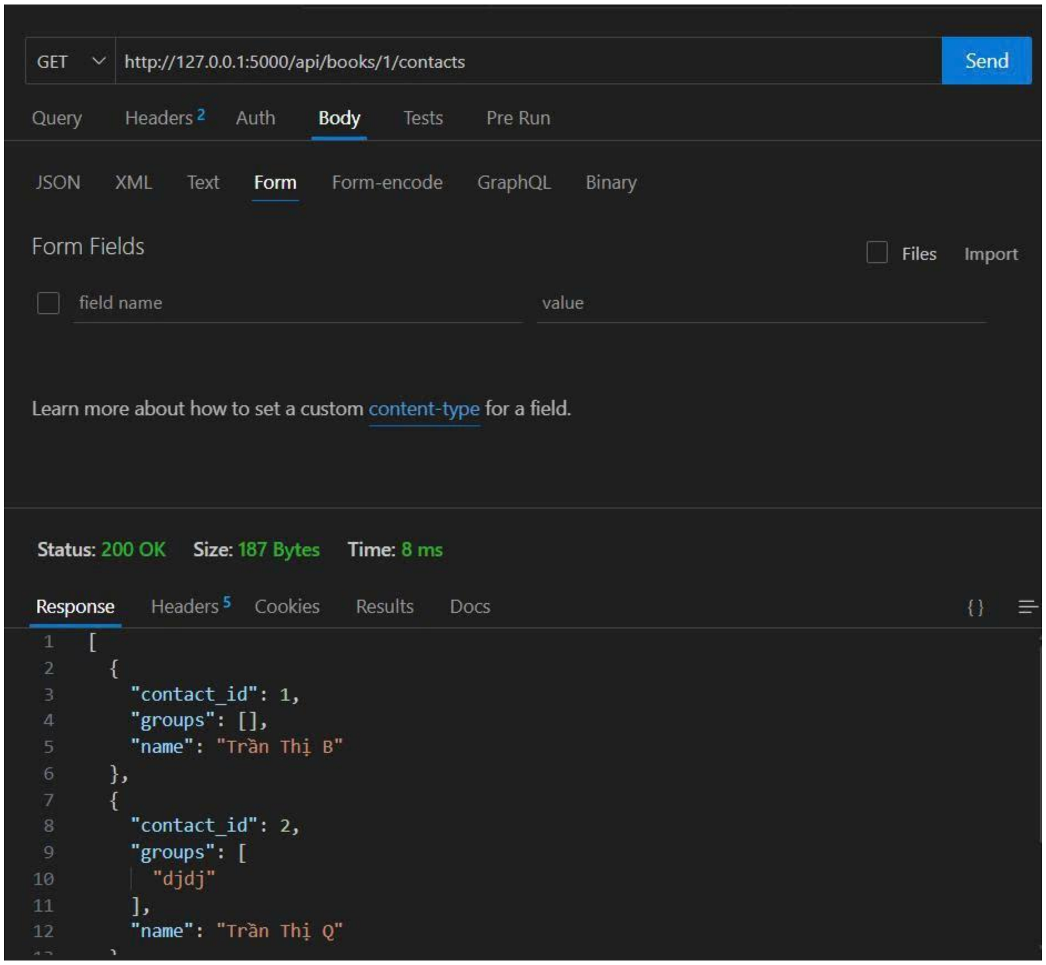


```
DELETE http://127.0.0.1:5000/api/books/1/groups/1
```

Status: 200 OK Size: 48 Bytes Time: 21 ms

```
{
  "message": "Group 1 deleted successfully"
}
```

- **The system successfully deletes the selected group**
- **The API returns status 200 OK**
- **Group data is removed from the database**



```
GET http://127.0.0.1:5000/api/books/1/contacts
```

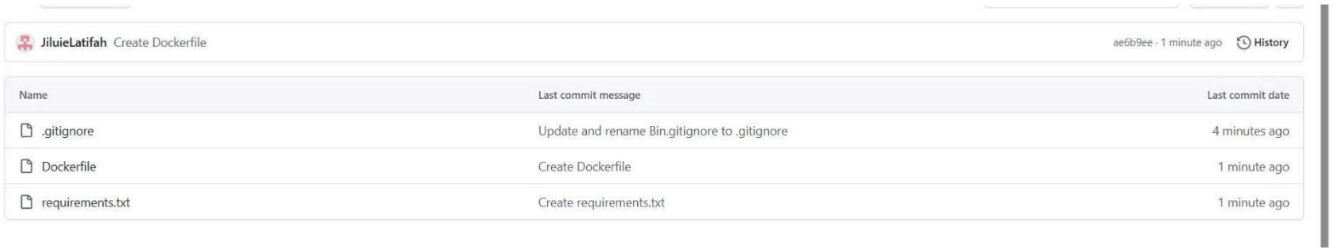
Learn more about how to set a custom content-type for a field.

Status: 200 OK Size: 187 Bytes Time: 8 ms

```
[
  {
    "contact_id": 1,
    "groups": [],
    "name": "Trần Thị B"
  },
  {
    "contact_id": 2,
    "groups": [
      "djdj"
    ],
    "name": "Trần Thị Q"
```

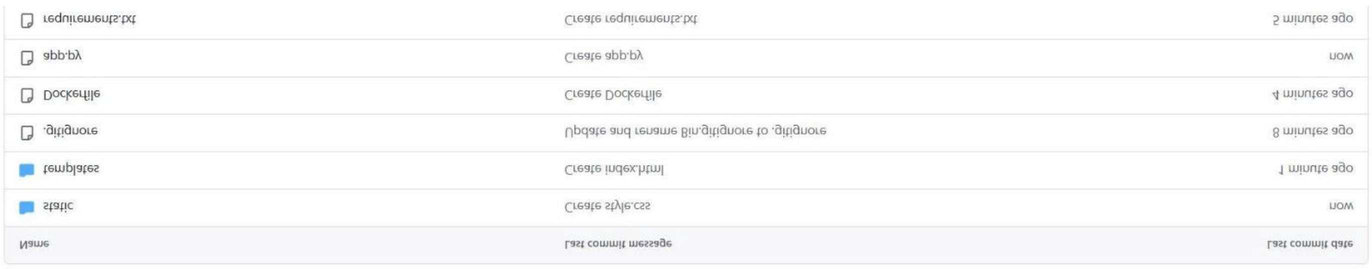
- **GET request to retrieve the contact list using Thunder Client, returning a successful 200 OK response with contact data.**

Evidence of using Git for teamwork

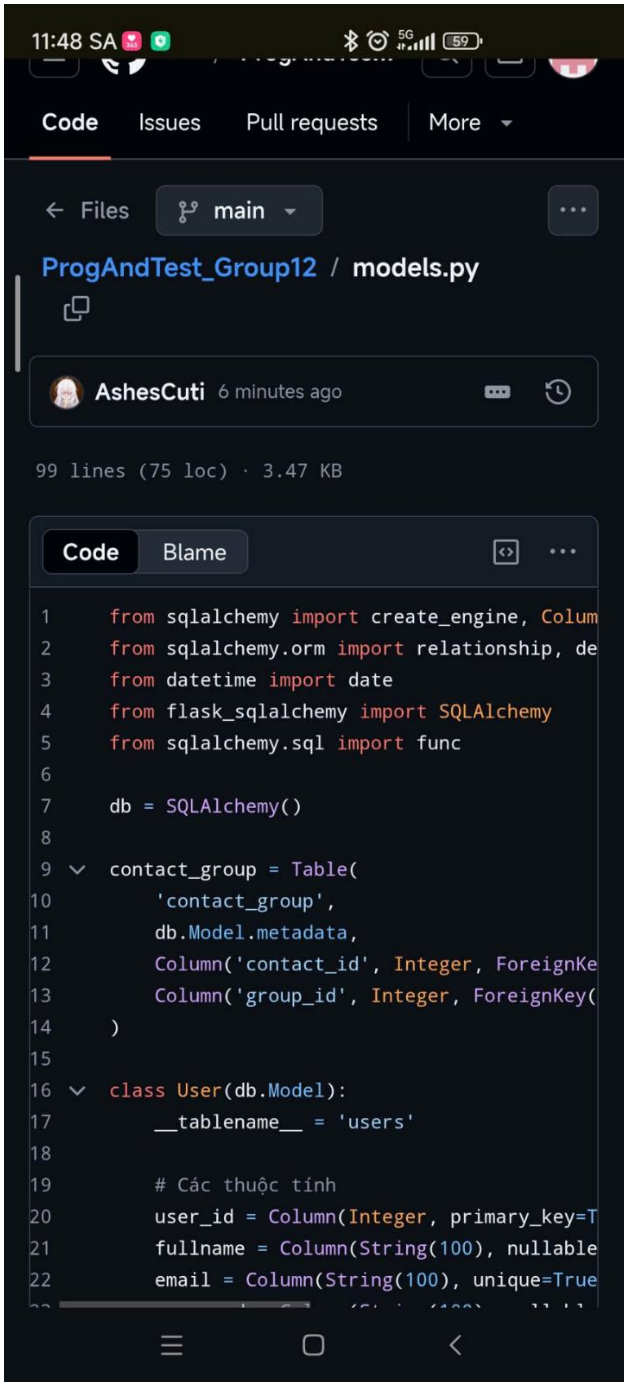


JilueLatifah Create Dockerfile ae6b9ee · 1 minute ago History

Name	Last commit message	Last commit date
.gitignore	Update and rename Bin.gitignore to .gitignore	4 minutes ago
Dockerfile	Create Dockerfile	1 minute ago
requirements.txt	Create requirements.txt	1 minute ago



Name	Last commit message	Last commit date
static	Create style.css	now
templates	Create index.html	1 minute ago
.gitignore	Update and rename bin.gitignore to .gitignore	8 minutes ago
Dockerfile	Create Dockerfile	4 minutes ago
app.py	Create app.py	now
requirements.txt	Create requirements.txt	5 minutes ago



ProgAndTest_Group12 / models.py

AshesCuti 6 minutes ago

99 lines (75 loc) · 3.47 KB

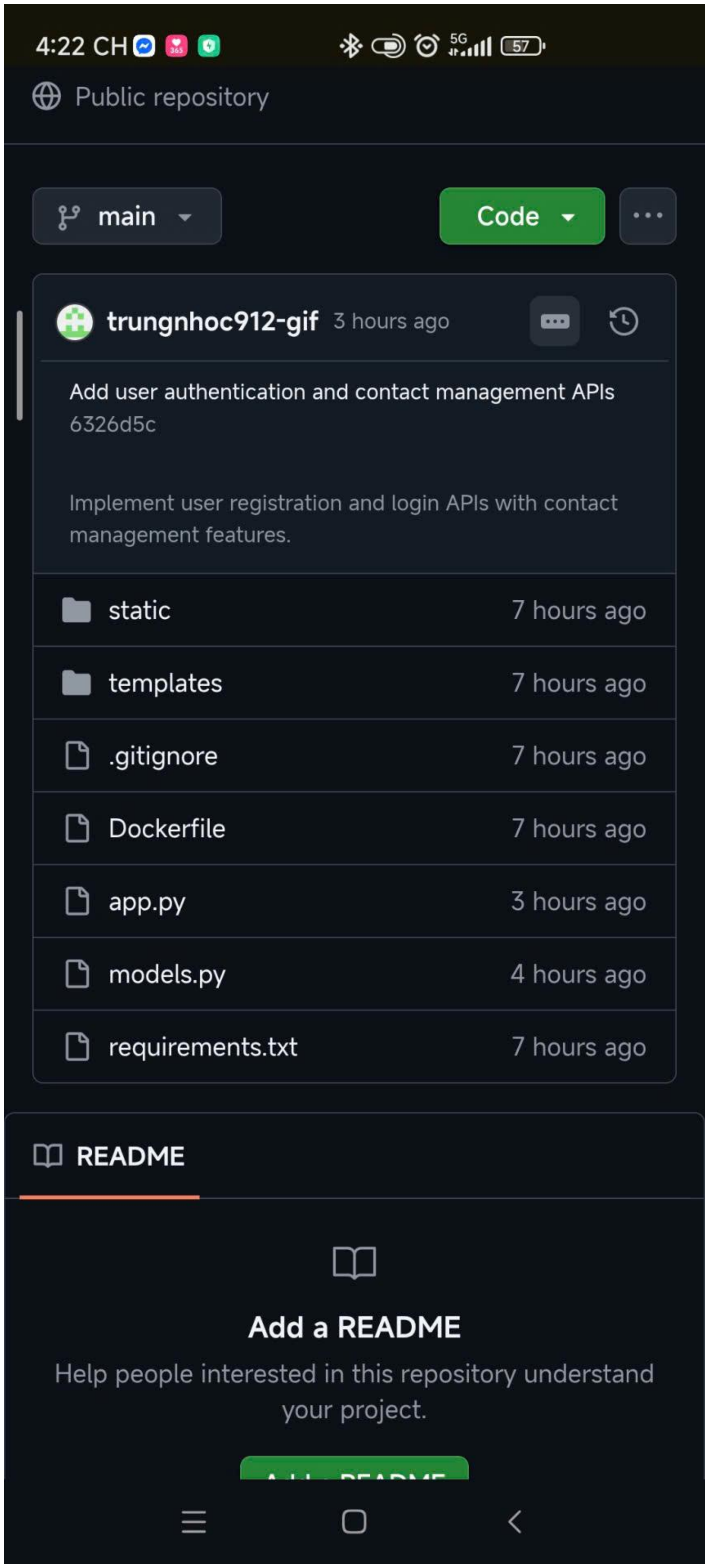
```python
from sqlalchemy import create_engine, Colum
from sqlalchemy.orm import relationship, de
from datetime import date
from flask_sqlalchemy import SQLAlchemy
from sqlalchemy.sql import func

db = SQLAlchemy()

contact_group = Table(
    'contact_group',
    db.Model.metadata,
    Column('contact_id', Integer, ForeignKe
    Column('group_id', Integer, ForeignKey(
)

class User(db.Model):
    __tablename__ = 'users'

    # Các thuộc tính
    user_id = Column(Integer, primary_key=T
    fullname = Column(String(100), nullable
    email = Column(String(100), unique=True
```


4:22 CH

Public repository

main

Code

trungnhoc912-gif 3 hours ago

Add user authentication and contact management APIs
6326d5c

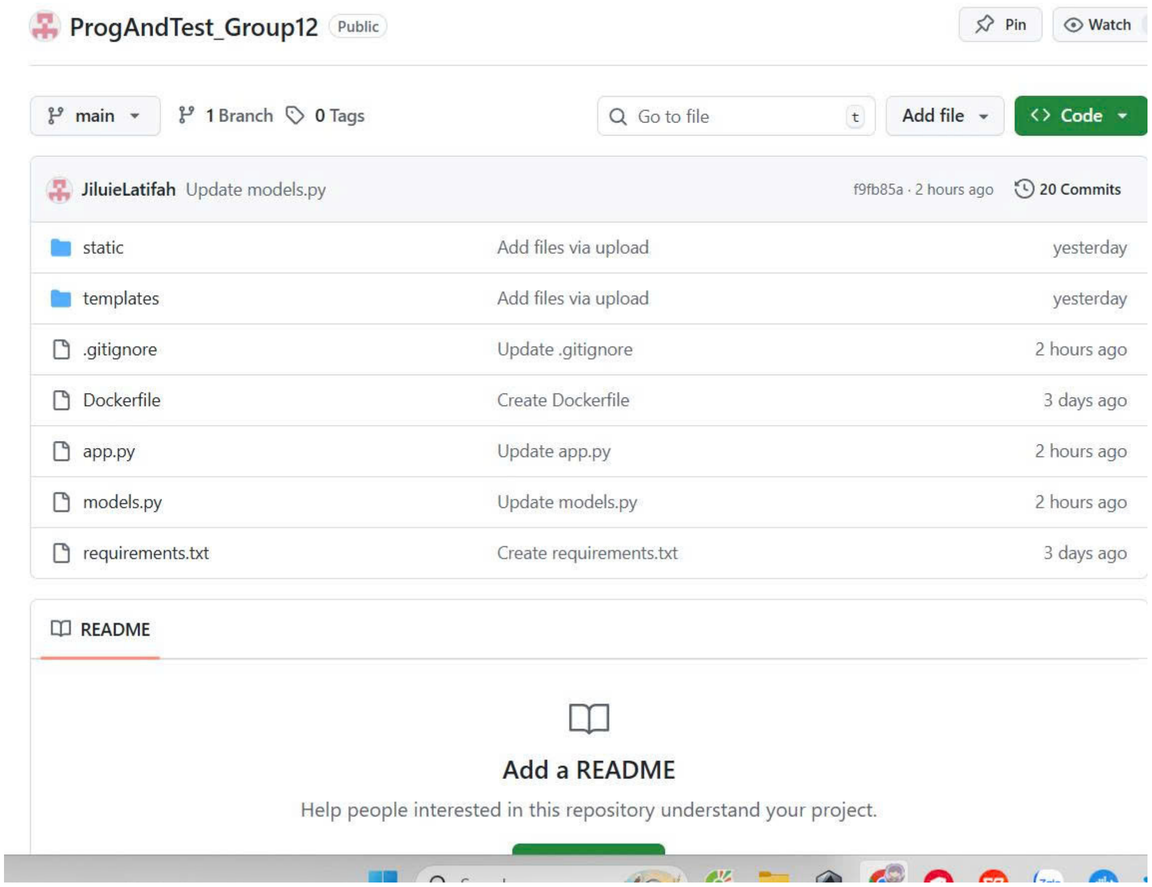
Implement user registration and login APIs with contact management features.

static	7 hours ago
templates	7 hours ago
.gitignore	7 hours ago
Dockerfile	7 hours ago
app.py	3 hours ago
models.py	4 hours ago
requirements.txt	7 hours ago

README

Add a README

Help people interested in this repository understand your project.



ProgAndTest_Group12 Public

Pin Watch

main 1 Branch 0 Tags Go to file t Add file Code

JiluieLatifah Update models.py f9fb85a · 2 hours ago 20 Commits

static	Add files via upload	yesterday
templates	Add files via upload	yesterday
.gitignore	Update .gitignore	2 hours ago
Dockerfile	Create Dockerfile	3 days ago
app.py	Update app.py	2 hours ago
models.py	Update models.py	2 hours ago
requirements.txt	Create requirements.txt	3 days ago

README

Add a README

Help people interested in this repository understand your project.