

Banking API and Web App

Project Brief

Task is to build internal API and Web Application for below features:

- Application must have custom login/logout pages with Multi factor/ 2 factor authentication using email-based OTP.
- Create a new bank account for a customer, with an initial deposit amount. A single customer may have multiple bank accounts.
- Transfer amounts between any two accounts, including those owned by different customers.
- Retrieve balances for a given account.
- Retrieve transfer history for a given account.
- Banking operations must be done through UI as well as from API.

Features Implemented

Login

- Every employee feature either with UI or API will only be available after successful authentication.
- After login, a session of 5 minutes is created.
- **Two - Factor Authentication feature is still in progress**

Logout

- After clicking on 'Logout' button, employee will be logged out and all the employee features will not be accessible.

Create Account

- For creating a new account for any customer, we need to make sure that specific customer exists in our database. If it's not, first creates the customer record and afterwards go for the account creation.

Account Opening Process

Create Customer Reocrd

For new customers, first create their records before going for opening account

[Click Here](#)

Create New Account

This is for creating a new account for an existing customer record

[Click Here](#)

- Once it's made sure that customer record exists, you can go for the account creation.

Balance Check

- For checking the balance of an account, just give the account number and you will get the current balance.

Money Transfer

- For transferring money, you need to provide withdrawal and deposit account number along with amount needed to be transfer.
- During this activity, below things will be checked:
 - 1 - Both the withdrawal and deposit account exist in database.
 - 2 - Withdrawal and deposit account number cannot be same.
 - 3 - Withdrawal account has balance equal to or more than the amount needed to be transfer.
 - 4 - Amount cannot be zero.

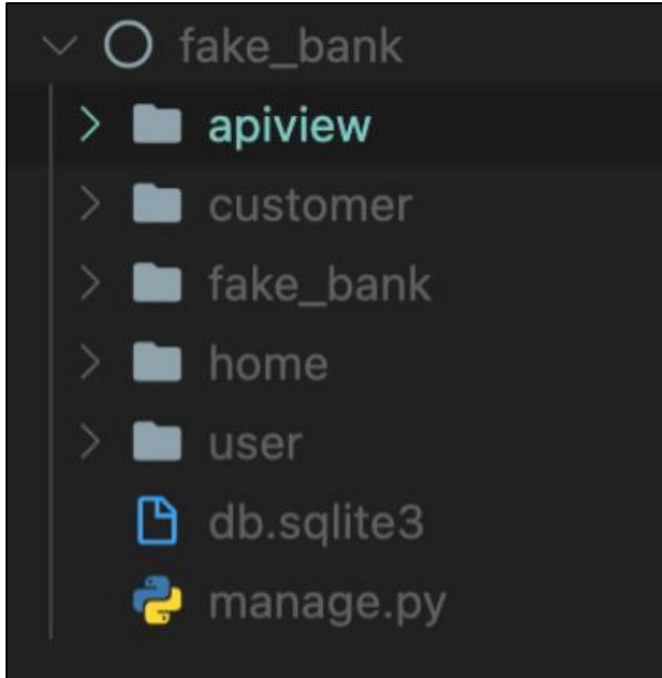
Transaction History

*** *This feature is still in progress* ***

Implementation

1 - Web Application

- Below are the apps created to meet the purpose:



- “home” app is handling all the homepage related requests.
- “user” app is handling all user related tasks, for example, login, logout etc.
- “customer” app is handling employee features which can eventually alter customer-related data.

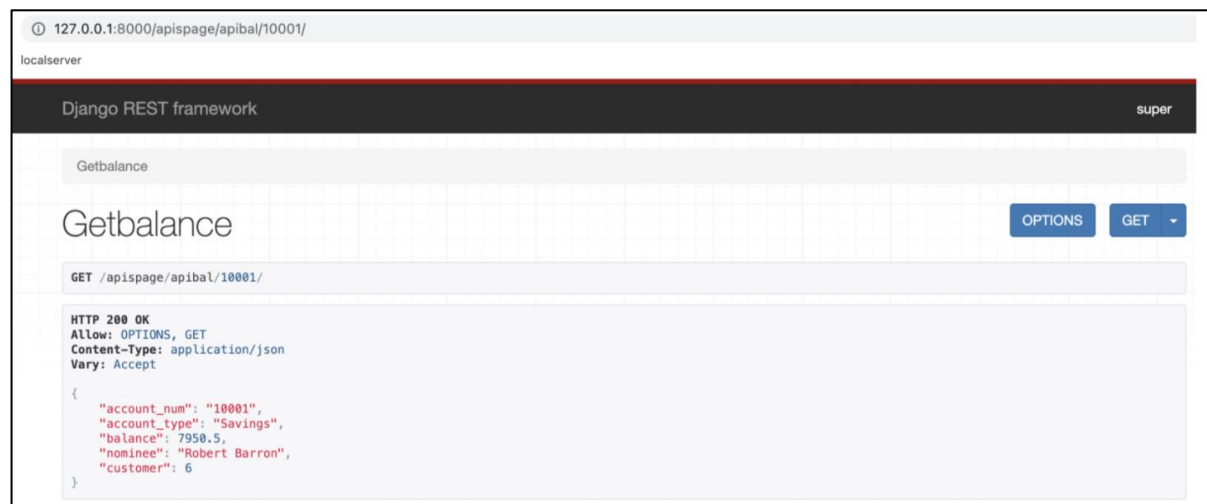
2 - API Implementation

For getting balance of a specific account

Request Type - GET

API URL - /apispag/apibal/<str:pk>/

where “pk” is account number

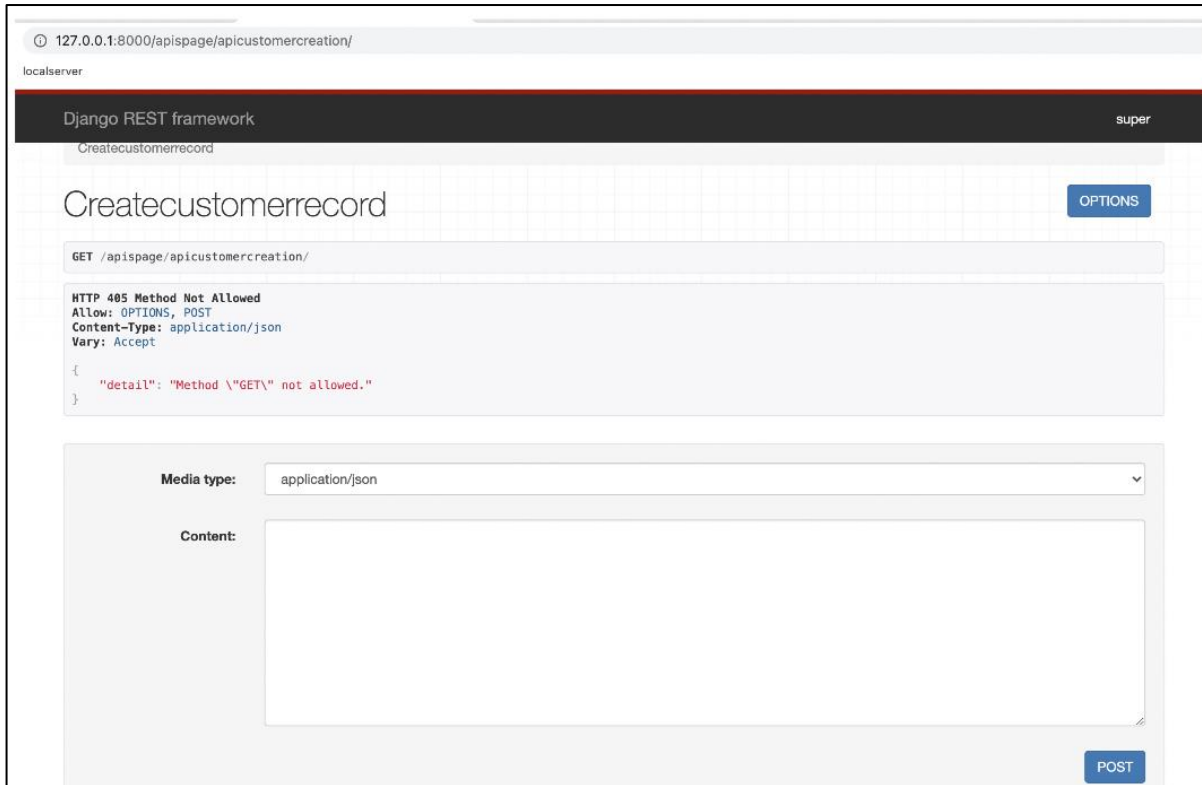


For creating a new customer record

Request Type - POST

API URL - /apispage/apicustomercreation/

```
{
  "first_name": "",
  "last_name": "",
  "email": "",
  "phone_no": "",
  "age": <age>
}
```



127.0.0.1:8000/apispage/apicustomercreation/

localhost

Django REST framework super

Createcustomerrecord

Createcustomerrecord OPTIONS

GET /apispage/apicustomercreation/

HTTP 405 Method Not Allowed
Allow: OPTIONS, POST
Content-Type: application/json
Vary: Accept

```
{
  "detail": "Method \"/apispage/apicustomercreation/"
```

Media type: application/json

Content:

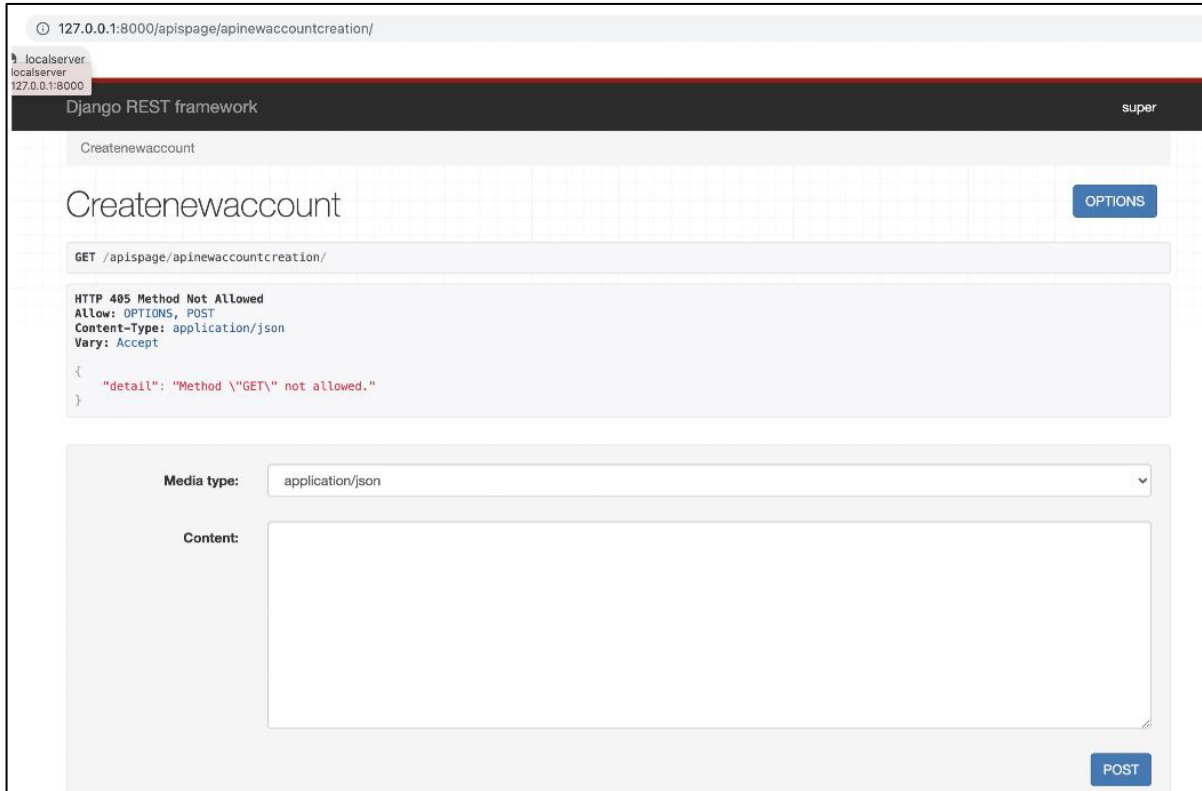
POST

For creating a new account for an existing customer record

Request Type - POST

API URL - apispage/apinewaccountcreation/

```
{
  "account_num": "",
  "account_type": "Savings/Current/Loan",
  "balance": 1000.00,
  "nominee": "",
  "customer": <cust_id>
}
```



127.0.0.1:8000/apispage/apinewaccountcreation/

localhost localhost 127.0.0.1:8000

Django REST framework super

Createnewaccount

Createnewaccount

OPTIONS

GET /apispage/apinewaccountcreation/

HTTP 405 Method Not Allowed
Allow: OPTIONS, POST
Content-Type: application/json
Vary: Accept

```
{
  "detail": "Method \"/apispage/apinewaccountcreation/"
```

Media type: application/json

Content:

POST

For transferring money between accounts

Request Type - POST

API URL - `apispage/apitransfermoney/`

```
{
  "with_acc_num": "",
  "tran_acc_num": "",
  "amount": 1000.00
}
```

The screenshot shows the Django REST framework API browser interface. The browser address bar displays `127.0.0.1:8000/apispage/apitransfermoney/`. The page title is "Transfermoney". The URL bar shows `GET /apispage/apitransfermoney/`. The response status is "HTTP 405 Method Not Allowed". The response headers are: `Allow: OPTIONS, POST`, `Content-Type: application/json`, and `Vary: Accept`. The response body is a JSON object: `{ "detail": "Method \"GET\" not allowed." }`. Below the response, there is a form for sending a request. The "Media type" dropdown is set to `application/json`. The "Content" field is empty. A "POST" button is visible at the bottom right of the form.

For listing out all customer details

Request Type - GET

API URL - apiviewofallcustomers/

The screenshot shows a web browser at the URL `127.0.0.1:8000/apispape/apiviewofallcustomers/`. The page is titled "Customers List" and displays the response for a GET request to `/apispape/apiviewofallcustomers/`. The response is an HTTP 200 OK with the following headers: `Allow: GET, HEAD, OPTIONS`, `Content-Type: application/json`, and `Vary: Accept`. The JSON response contains an array of four customer objects:

```
[
  {
    "id": 6,
    "first_name": "Arisha",
    "last_name": "Barron",
    "email": "arisha.barron@test.com",
    "phone_no": "8877665544",
    "age": 26
  },
  {
    "id": 7,
    "first_name": "Branden",
    "last_name": "Gibson",
    "email": "brandon.gibson@test.com",
    "phone_no": "9567832424",
    "age": 29
  },
  {
    "id": 8,
    "first_name": "Rhonda",
    "last_name": "Church",
    "email": "rhonda.church@gmail.com",
    "phone_no": "6675764532",
    "age": 23
  },
  {
    "id": 9,
    "first_name": "Georgina",
    "last_name": "Barron",
    "email": "georgina.barron@test.com",
    "phone_no": "9988776655",
    "age": 25
  }
]
```

For listing out all account numbers with other details

Request Type - GET

API URL - apiviewofallaccounts/

The screenshot shows a web browser at the URL `127.0.0.1:8000/apispape/apiviewofallaccounts/`. The page is titled "Customer Accounts List" and displays the response for a GET request to `/apispape/apiviewofallaccounts/`. The response is an HTTP 200 OK with the following headers: `Allow: GET, HEAD, OPTIONS`, `Content-Type: application/json`, and `Vary: Accept`. The JSON response contains an array of three account objects:

```
[
  {
    "account_num": "10001",
    "account_type": "Savings",
    "balance": 7950.5,
    "nominee": "Robert Barron",
    "customer": 6
  },
  {
    "account_num": "10002",
    "account_type": "Current",
    "balance": 8500.0,
    "nominee": "Robert Barron",
    "customer": 6
  },
  {
    "account_num": "10003",
    "account_type": "Savings",
    "balance": 46549.5,
    "nominee": "Laura Gibson",
    "customer": 7
  }
]
```

Implementation Tool

Language - Python v3.9.1

Web Framework - Django v3.1.5

API Framework - Django Rest Framework v3.12.2

Database - SQLite3

Dummy Super Account

username - super

password - welcome@123