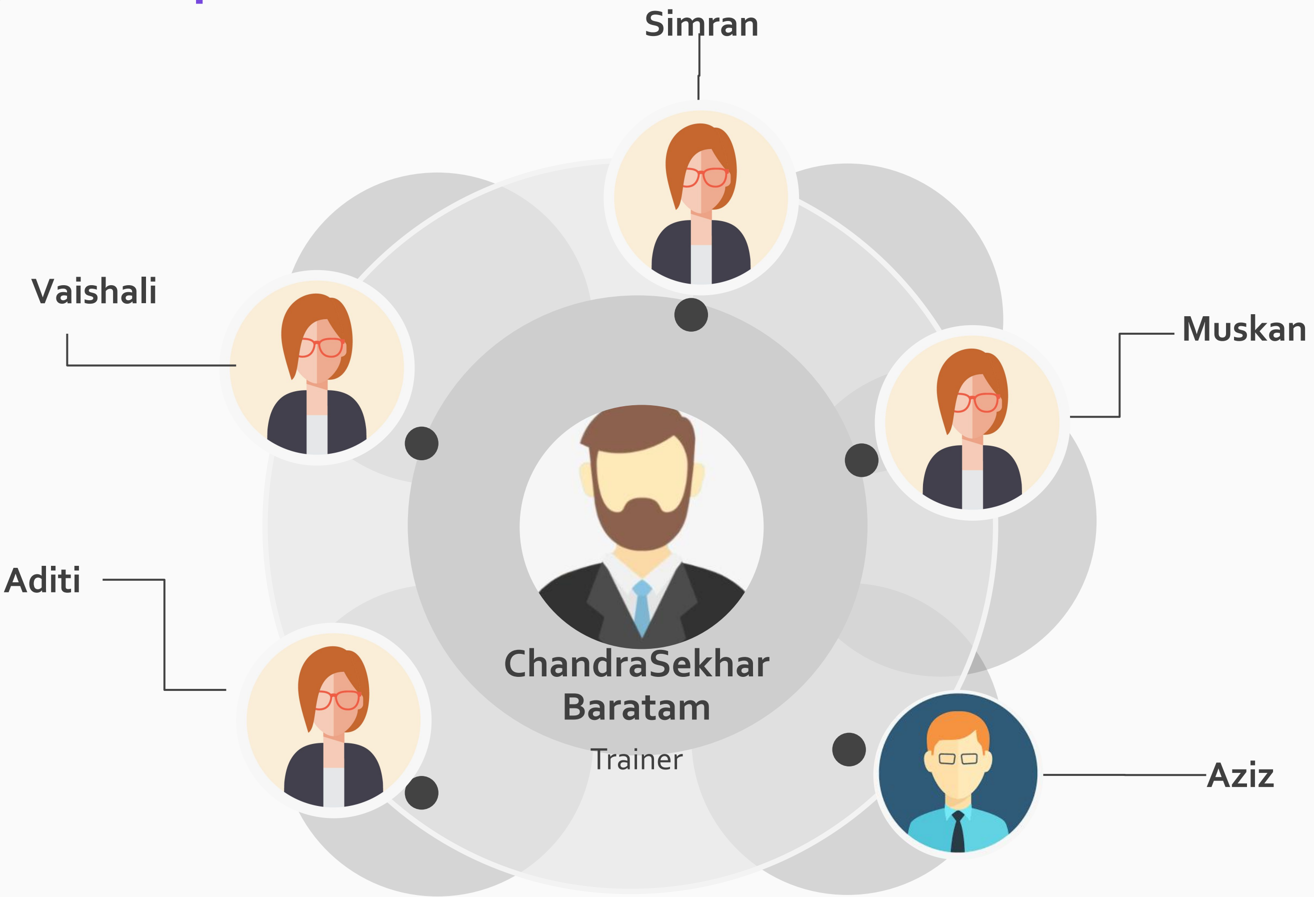
Bank Transfer App : EzPay

Making your payments easier..

Team Members - Squad 2

Career Shaper™



User Story Overview:

1. **Project Goal:** To create a secure and efficient full-stack application that simplifies digital payments, focusing on transactions like UPI and bank transfers.
2. **Key Features:** The application provides a user-friendly interface for making quick payments, while ensuring transactional data is secure and reliable.
3. **Target Audience:** Users seeking a seamless and trustworthy platform for their daily financial transactions.

Expected Components:

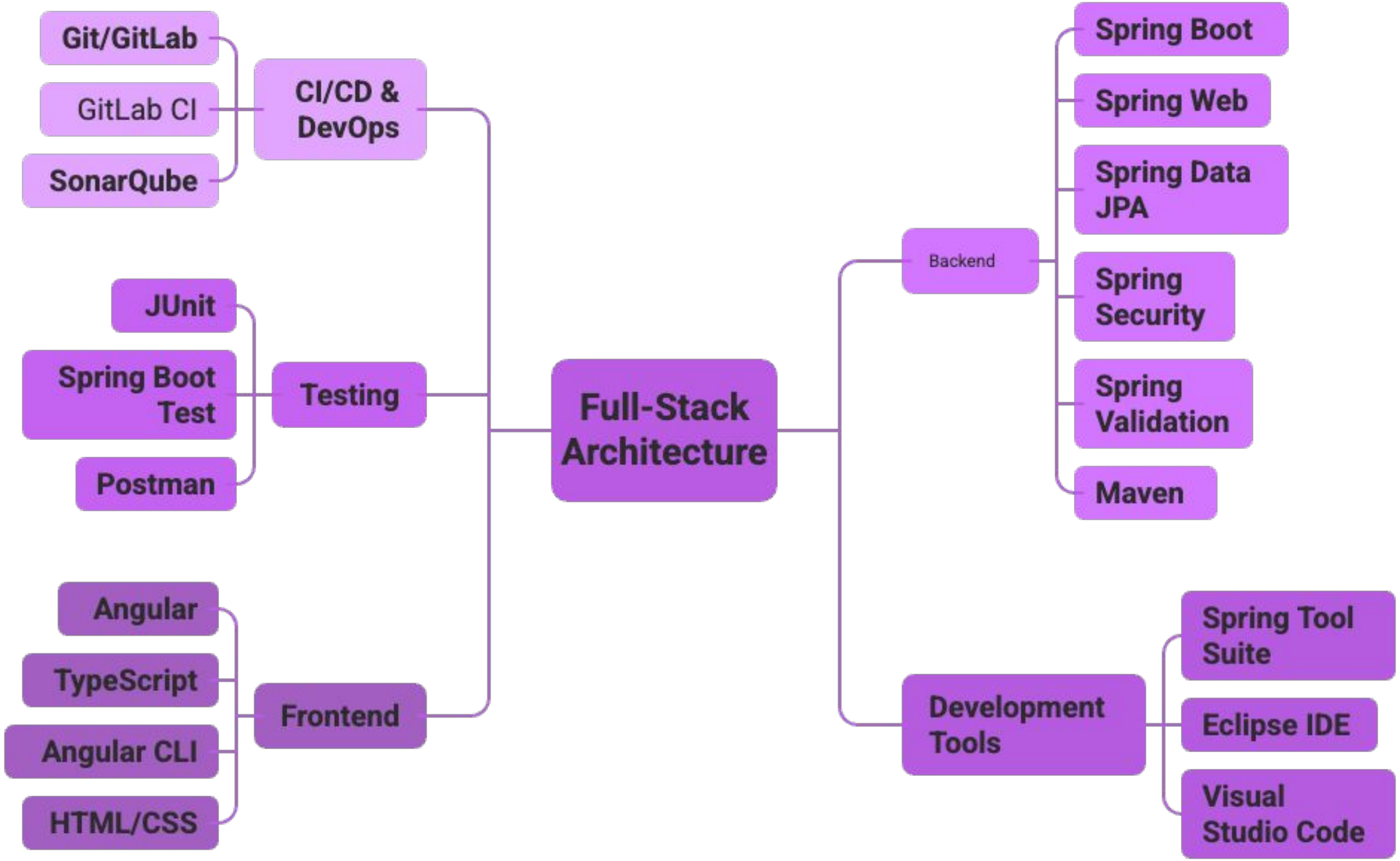
2.1 Bank Transfers:

- A user wants to initiate a bank transfer from their account to a recipient's account.
- **Expected Course of Action:**
  a. The user selects "Bank Transfer," enters the recipient's details (accountNo, IFSC), and confirms the amount.
  b. The system initiates the transfer and displays a success message upon confirmation.
- The system is designed to handle potential failures, such as invalid bank details or a failed transfer, and will notify the user with a specific reason.
- **Why This is Important:** Bank transfers are a fundamental aspect of digital finance, and our system provides a reliable, secure way for users to manage these transactions.

Expected Components:

2.2 UPI Payments::

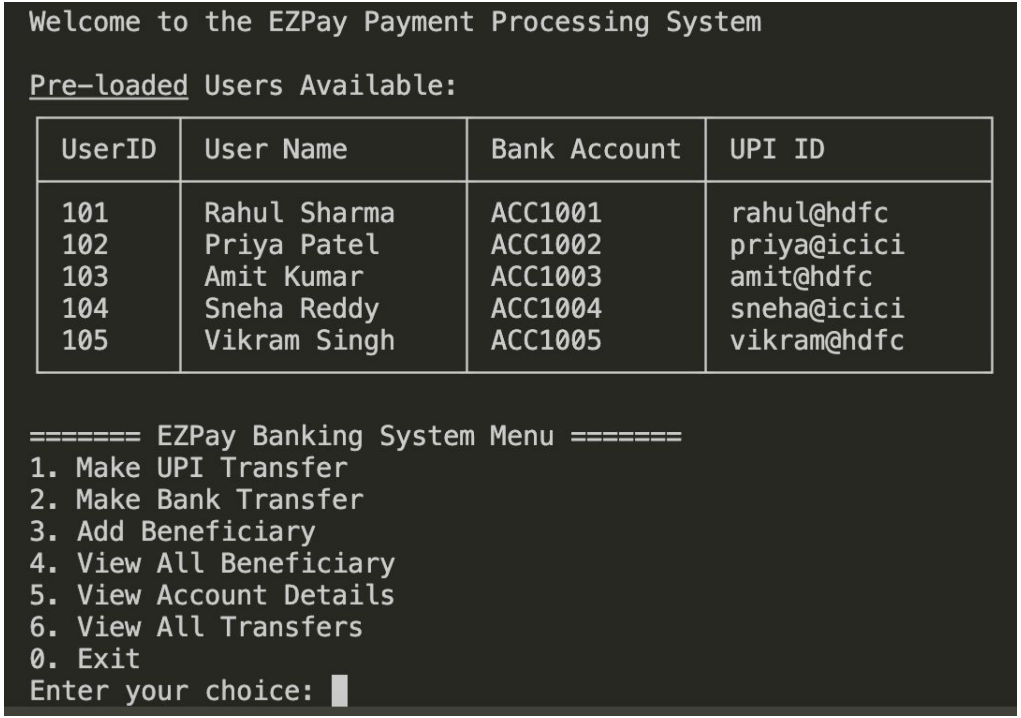
- A user wants to make a payment using their registered UPI ID through the application.
- **Expected Course of Action:**
  - User selects "Make UPI Payment," enters the recipient's UPI ID and the payment amount.
  - System processes the transaction through the UPI service and provides real-time status updates to the user.
- system has built-in error handling for situations like an invalid UPI ID or insufficient funds, ensuring the user is always informed of the transaction status.
- **Why This is Important:** UPI has become a dominant form of digital payment. Integrating this feature ensures our application is relevant and provides a fast, hassle-free payment method.


Full-Stack Architecture
CI/CD & DevOps
Git/GitLab
GitLab CI
SonarQube
Testing
JUnit
Spring Boot Test
Postman
Frontend
Angular
TypeScript
Angular CLI
HTML/CSS
Backend
Spring Boot
Spring Web
Spring Data JPA
Spring Security
Spring Validation
Maven
Development Tools
Spring Tool Suite
Eclipse IDE
Visual Studio Code

Sprint 1 - Core Java Demo

Preloaded Users and Menu

- System starts with preloaded users.
- Users can perform :
  a. UPI transfers
  b. bank transfers
  c. view balances
  d. manage beneficiaries.

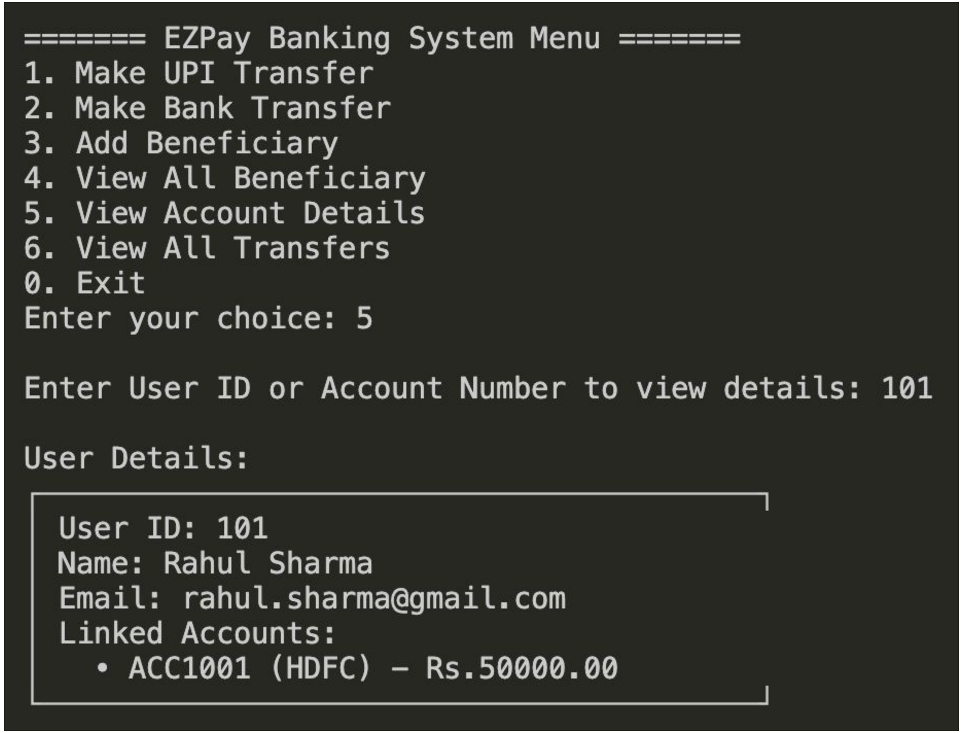


```
Welcome to the EZPay Payment Processing System

Pre-loaded Users Available:
┌────────┬──────────────────┬───────────────┬──────────────────┐
│ UserID │ User Name        │ Bank Account  │ UPI ID           │
├────────┼──────────────────┼───────────────┼──────────────────┤
│ 101    │ Rahul Sharma     │ ACC1001       │ rahul@hdfc       │
│ 102    │ Priya Patel      │ ACC1002       │ priya@icici      │
│ 103    │ Amit Kumar       │ ACC1003       │ amit@hdfc        │
│ 104    │ Sneha Reddy      │ ACC1004       │ sneha@icici      │
│ 105    │ Vikram Singh     │ ACC1005       │ vikram@hdfc      │
└────────┴──────────────────┴───────────────┴──────────────────┘

======= EZPay Banking System Menu =======
1. Make UPI Transfer
2. Make Bank Transfer
3. Add Beneficiary
4. View All Beneficiary
5. View Account Details
6. View All Transfers
0. Exit
Enter your choice:
```

Sprint 1

Account Details

- Users can fetch account details using **User ID or Account Number**.
- Displays:
  - User info (ID, Name, Email).
  - Linked accounts with bank & balance.

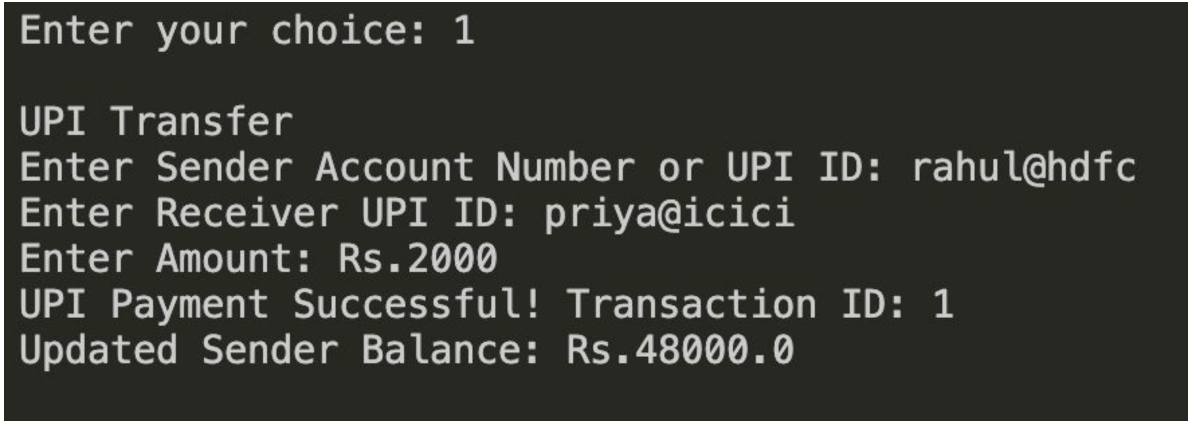


```
======= EZPay Banking System Menu =======
1. Make UPI Transfer
2. Make Bank Transfer
3. Add Beneficiary
4. View All Beneficiary
5. View Account Details
6. View All Transfers
0. Exit
Enter your choice: 5

Enter User ID or Account Number to view details: 101

User Details:
┌─────────────────────────────────────────────┐
 User ID: 101
 Name: Rahul Sharma
 Email: rahul.sharma@gmail.com
 Linked Accounts:
   • ACC1001 (HDFC) - Rs.50000.00
└─────────────────────────────────────────────┘
```

Sprint 1

UPI Transfer

- Sender enters **UPI ID / Account No.** and receiver's UPI ID.
- Enter transfer amount.
- System validates and updates balance.
- Generates unique **Transaction ID**.

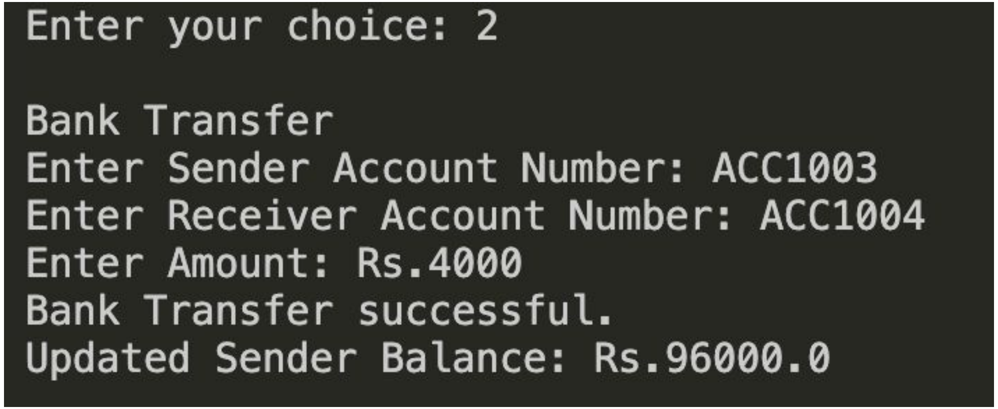


```
Enter your choice: 1

UPI Transfer
Enter Sender Account Number or UPI ID: rahul@hdfc
Enter Receiver UPI ID: priya@icici
Enter Amount: Rs.2000
UPI Payment Successful! Transaction ID: 1
Updated Sender Balance: Rs.48000.0
```

Sprint 1

Bank Transfer

- Transfers can be made directly between account numbers.
- Amount debited from sender & credited to receiver.
- Balance auto-updated.

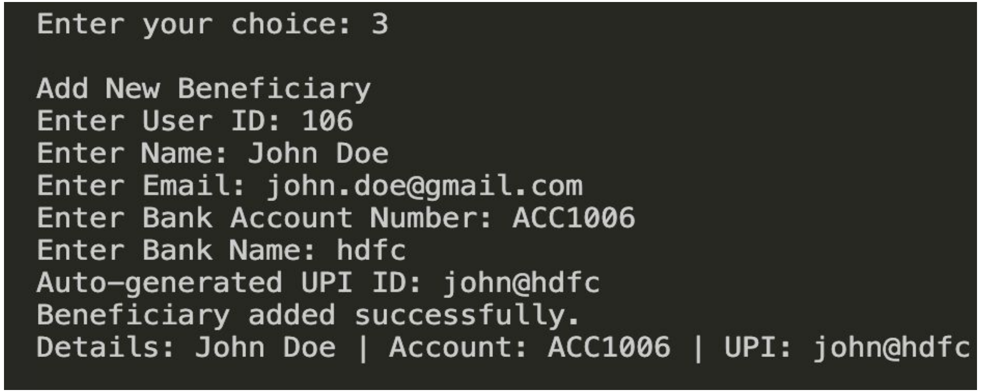


```
Enter your choice: 2

Bank Transfer
Enter Sender Account Number: ACC1003
Enter Receiver Account Number: ACC1004
Enter Amount: Rs.4000
Bank Transfer successful.
Updated Sender Balance: Rs.96000.0
```

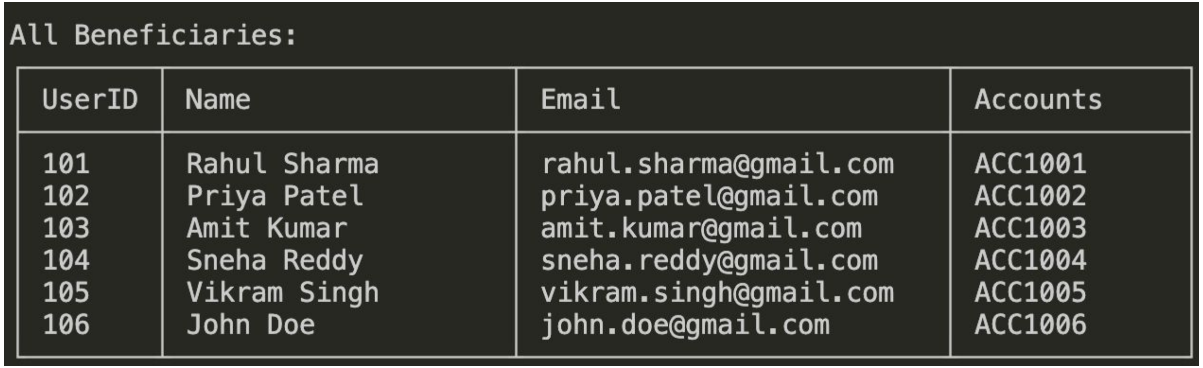
Sprint 1

Add Beneficiary

- Users can add new beneficiaries with details:
  - User ID, Name, Email, Bank Account, Bank Name.
- System auto-generates UPI ID.
- Beneficiaries stored for future transfers.



```
Enter your choice: 3

Add New Beneficiary
Enter User ID: 106
Enter Name: John Doe
Enter Email: john.doe@gmail.com
Enter Bank Account Number: ACC1006
Enter Bank Name: hdfc
Auto-generated UPI ID: john@hdfc
Beneficiary added successfully.
Details: John Doe | Account: ACC1006 | UPI: john@hdfc
```

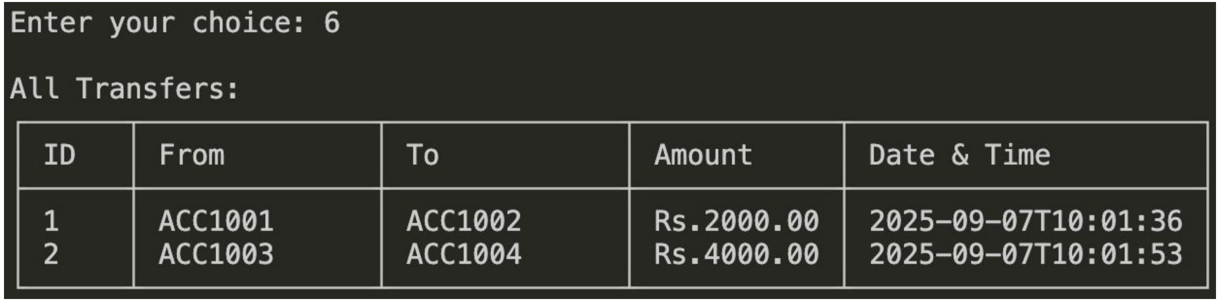


```
All Beneficiaries:
+--------+-------------------+------------------------+----------+
| UserID | Name              | Email                  | Accounts |
+--------+-------------------+------------------------+----------+
| 101    | Rahul Sharma      | rahul.sharma@gmail.com | ACC1001  |
| 102    | Priya Patel       | priya.patel@gmail.com  | ACC1002  |
| 103    | Amit Kumar        | amit.kumar@gmail.com   | ACC1003  |
| 104    | Sneha Reddy       | sneha.reddy@gmail.com  | ACC1004  |
| 105    | Vikram Singh      | vikram.singh@gmail.com | ACC1005  |
| 106    | John Doe          | john.doe@gmail.com     | ACC1006  |
+--------+-------------------+------------------------+----------+
```

Sprint 1

Transfer History

- All transactions logged with:
  - Transaction ID, Sender, Receiver, Amount, Date & Time.

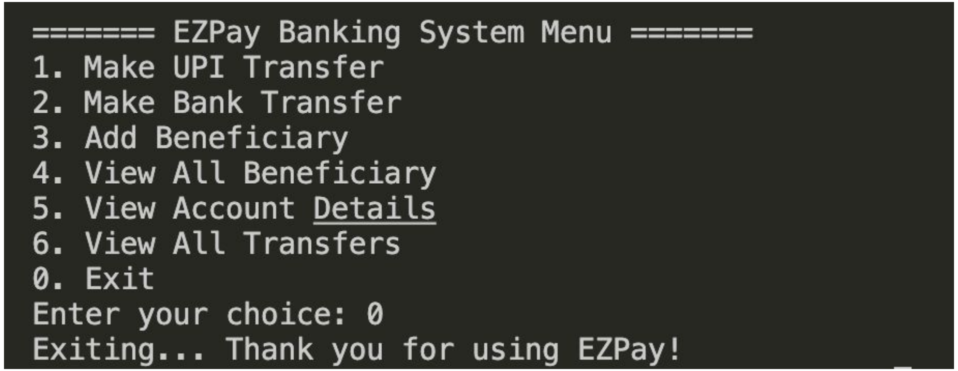


```
Enter your choice: 6

All Transfers:
┌────┬──────────┬──────────┬────────────┬─────────────────────┐
│ ID │ From     │ To       │ Amount     │ Date & Time         │
├────┼──────────┼──────────┼────────────┼─────────────────────┤
│ 1  │ ACC1001  │ ACC1002  │ Rs.2000.00 │ 2025-09-07T10:01:36 │
│ 2  │ ACC1003  │ ACC1004  │ Rs.4000.00 │ 2025-09-07T10:01:53 │
└────┴──────────┴──────────┴────────────┴─────────────────────┘
```

- Ensures full transparency & traceability.

Exit

- Clean exit with **“Thank you for using EZPay”** message.



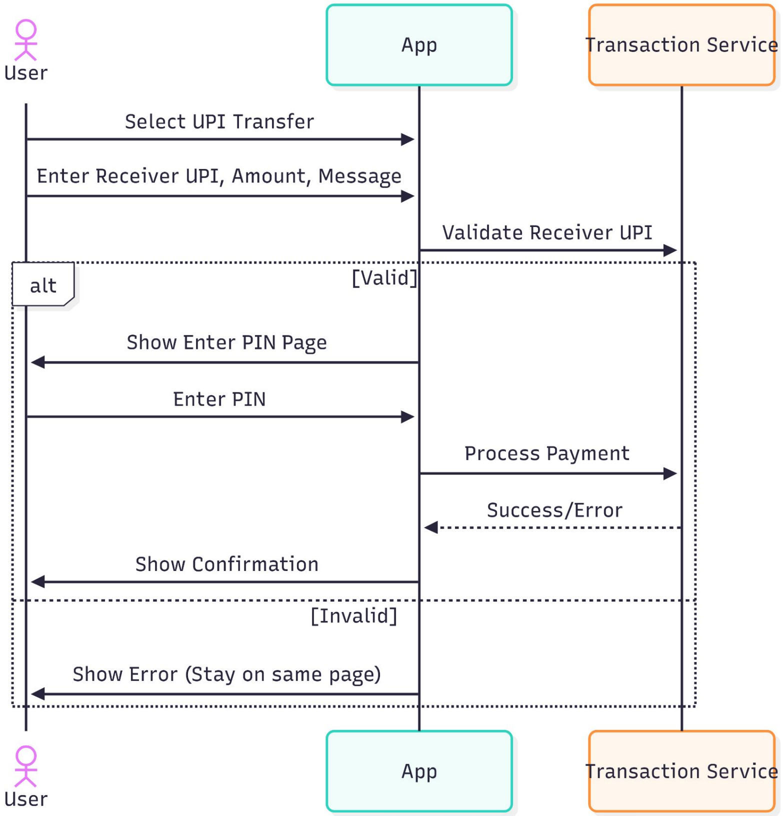
```
======= EZPay Banking System Menu =======
1. Make UPI Transfer
2. Make Bank Transfer
3. Add Beneficiary
4. View All Beneficiary
5. View Account Details
6. View All Transfers
0. Exit
Enter your choice: 0
Exiting... Thank you for using EZPay!
```

Sprint 2 : Development of frontend in Angular and business logic using spring boot JPA for UPI Payments and Bank Transfer

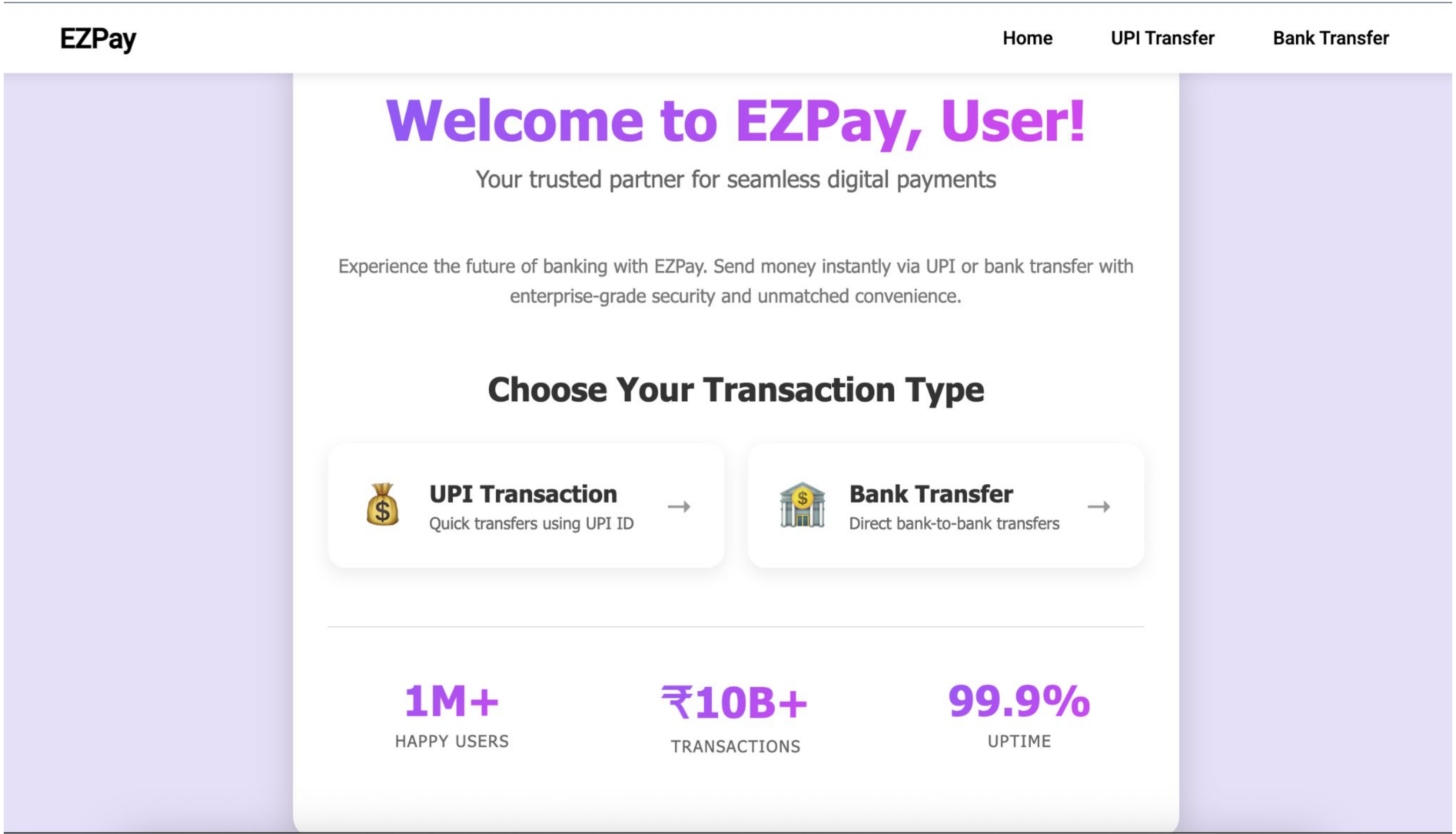
Vaishali Aggarwal - Developer for UPI Payments
Muskan - Developer for Bank Transfer
Aditi - Tester for UPI Payments
Simran - Tester for Bank Transfers
Aziz - Code Reviewer

UPI Transaction Sequence Diagram

Career Shaper™



Landing Page



Welcome to EZPay, User!

Your trusted partner for seamless digital payments

Experience the future of banking with EZPay. Send money instantly via UPI or bank transfer with enterprise-grade security and unmatched convenience.

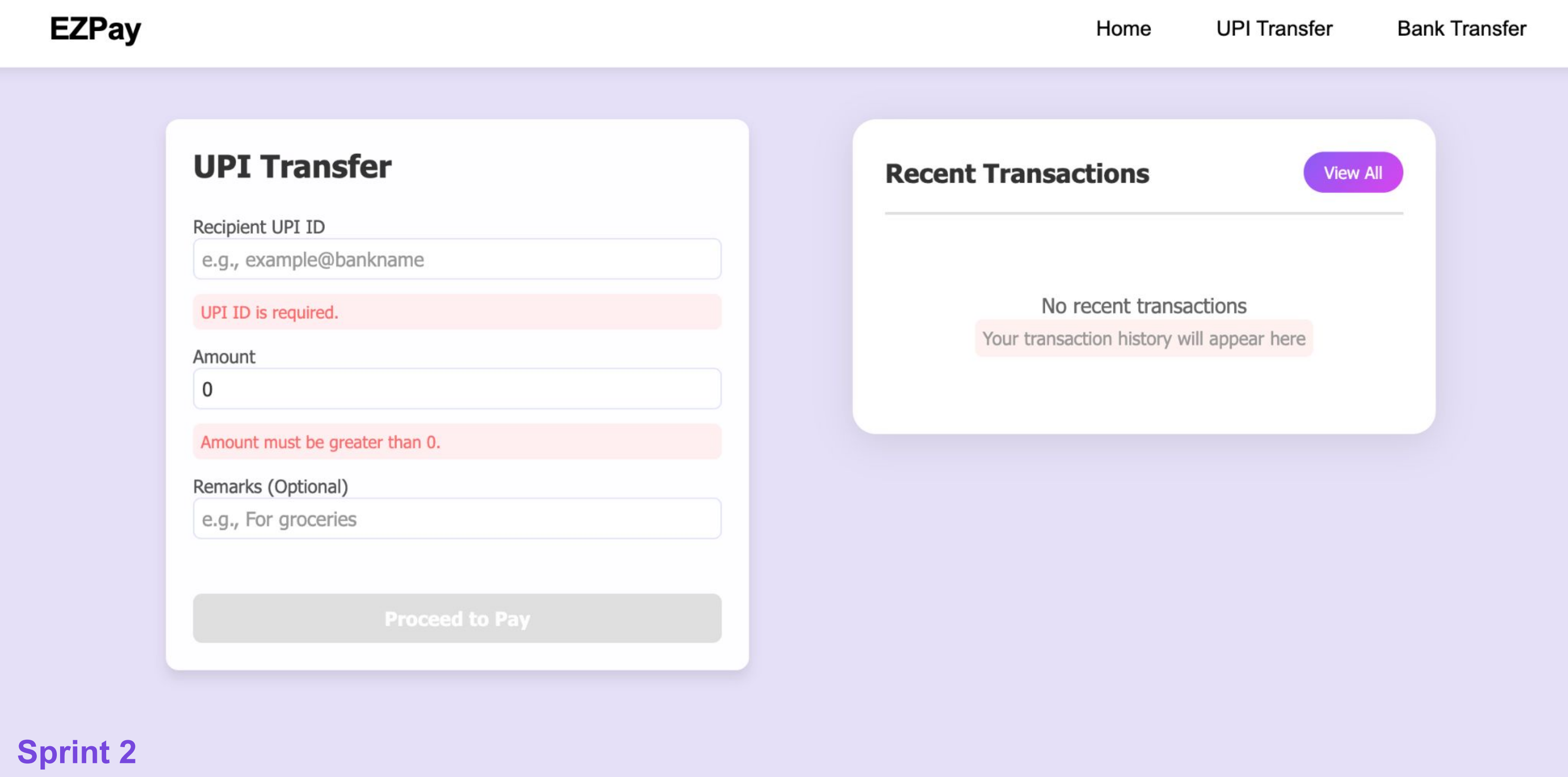
Choose Your Transaction Type

💰 **UPI Transaction** Quick transfers using UPI ID →

🏦 **Bank Transfer** Direct bank-to-bank transfers →

1M+	₹10B+	99.9%
HAPPY USERS	TRANSACTIONS	UPTIME

UPI Transfer page showing validations



UPI Transfer

Recipient UPI ID

e.g., example@bankname

UPI ID is required.

Amount

0

Amount must be greater than 0.

Remarks (Optional)

e.g., For groceries

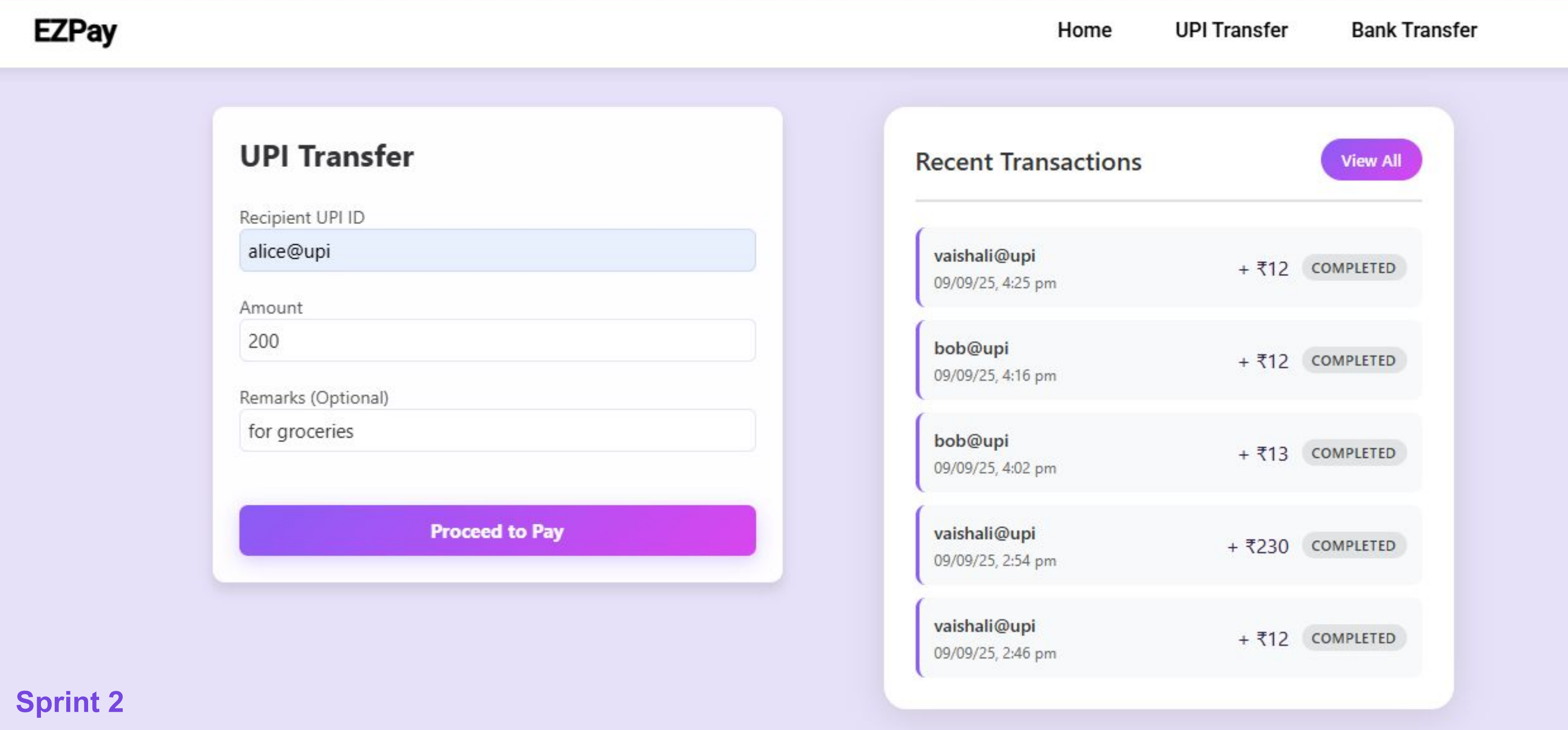
Recent Transactions

No recent transactions

Your transaction history will appear here

Sprint 2

UPI Transfer page with data



UPI Transfer

Recipient UPI ID: alice@upi

Amount: 200

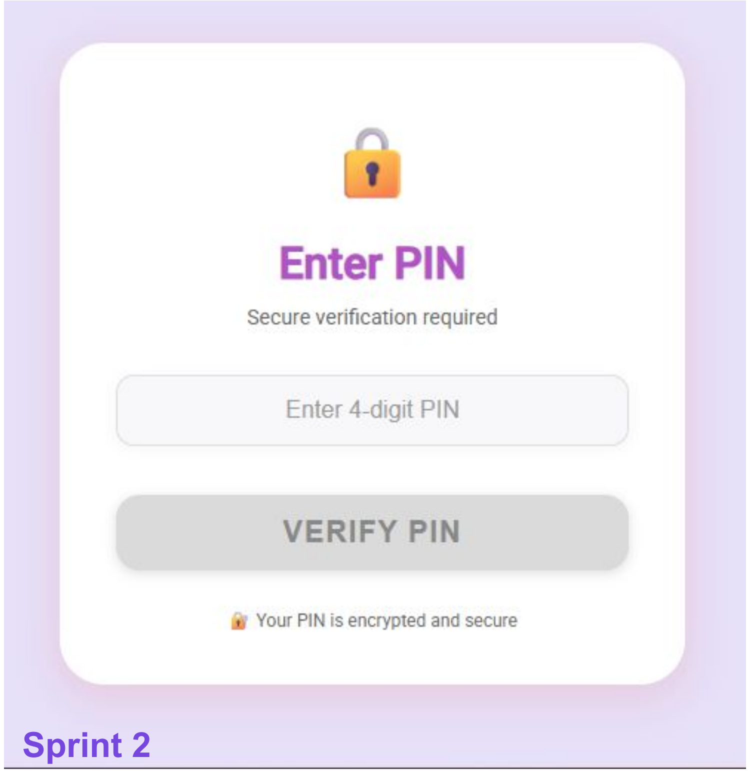
Remarks (Optional): for groceries

Recent Transactions

vaishali@upi 09/09/25, 4:25 pm	+ ₹12	COMPLETED
bob@upi 09/09/25, 4:16 pm	+ ₹12	COMPLETED
bob@upi 09/09/25, 4:02 pm	+ ₹13	COMPLETED
vaishali@upi 09/09/25, 2:54 pm	+ ₹230	COMPLETED
vaishali@upi 09/09/25, 2:46 pm	+ ₹12	COMPLETED

Sprint 2

PIN Entry Page



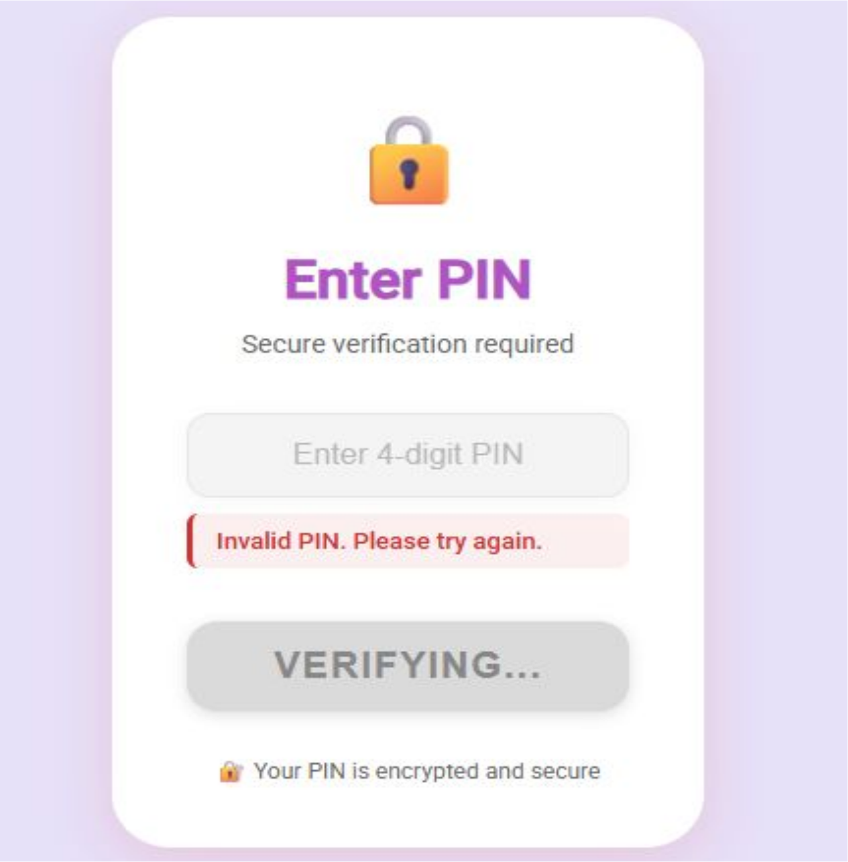
🔒

Enter PIN

Secure verification required

🔐 Your PIN is encrypted and secure

Sprint 2



🔒

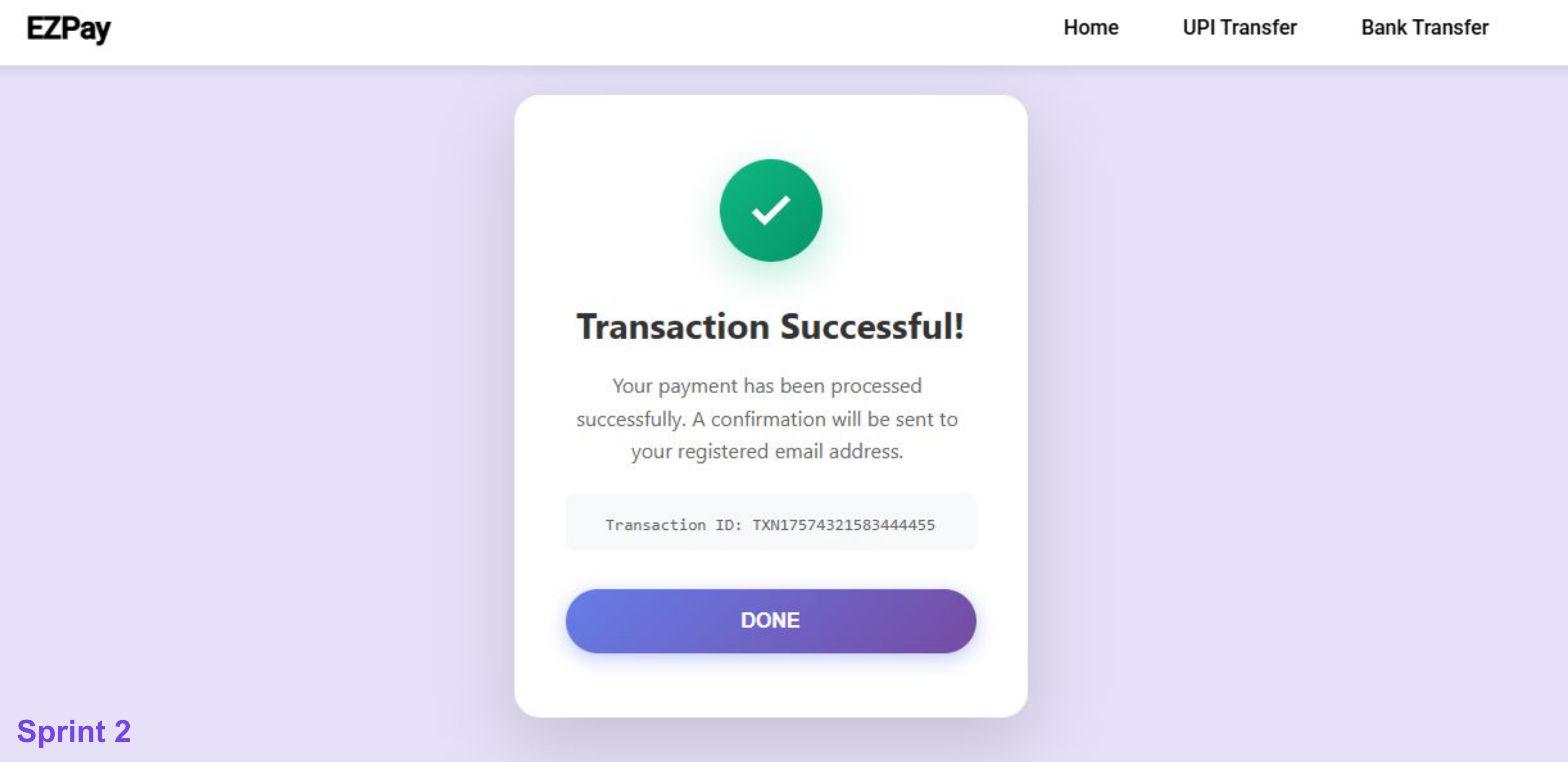
Enter PIN

Secure verification required

Invalid PIN. Please try again.

🔐 Your PIN is encrypted and secure

Transaction Successful Page after verifying the PIN



Transaction Successful!

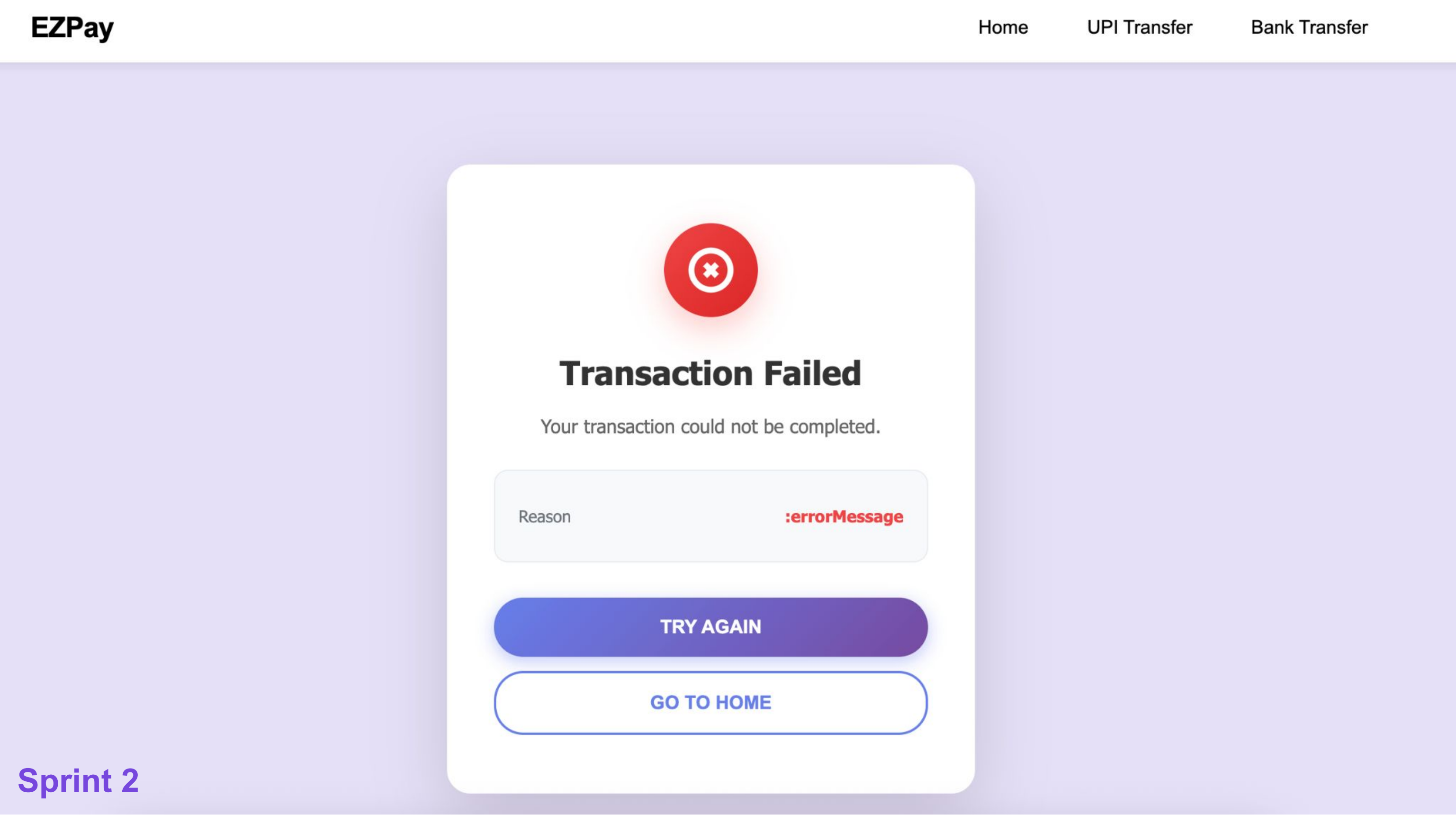
Your payment has been processed successfully. A confirmation will be sent to your registered email address.

Transaction ID: TXN17574321583444455

Sprint 2

Transaction Failed Page

Career Shaper™



EZPay

Transaction Failed

Your transaction could not be completed.

Reason :errorMessage

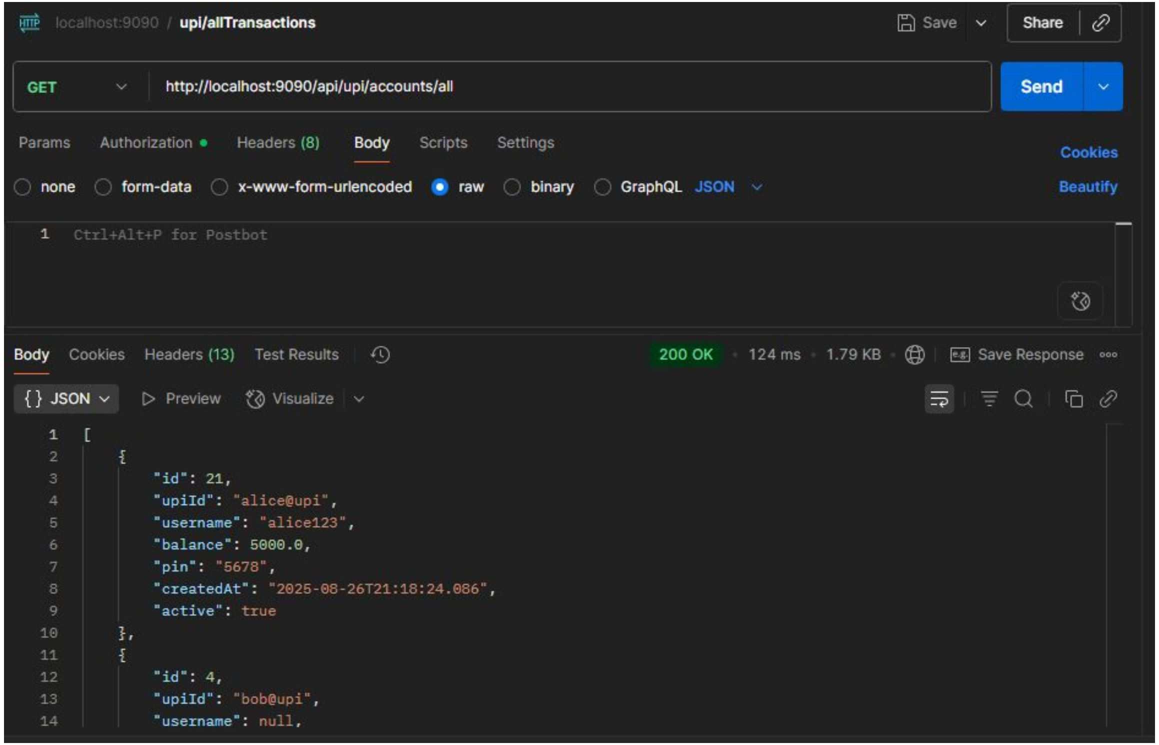
Sprint 2

Postman Screenshots

UPI Accounts

All the UPI accounts fetched using get method to the following URL:

http://localhost:9090/api/upi/accounts/all



GET http://localhost:9090/api/upi/accounts/all

○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```
Ctrl+Alt+P for Postbot
```

200 OK · 124 ms · 1.79 KB

{} JSON

```
[
    {
        "id": 21,
        "upiId": "alice@upi",
        "username": "alice123",
        "balance": 5000.0,
        "pin": "5678",
        "createdAt": "2025-08-26T21:18:24.086",
        "active": true
    },
    {
        "id": 4,
        "upiId": "bob@upi",
        "username": null,
```

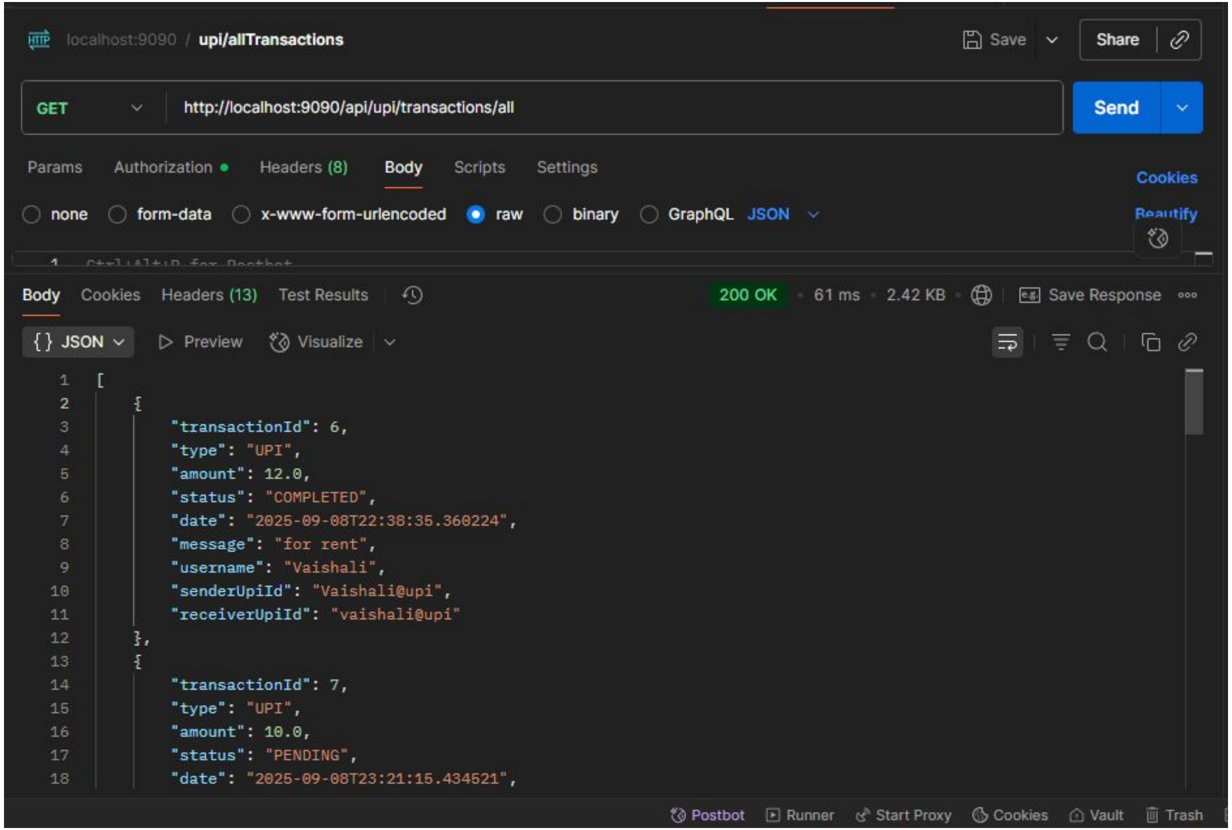
Sprint 2

Creating UPI Account

The request body and the post method to the following URL to create an UPI Account.

http://localhost:9090/api/upi/accounts/create

Note: An UPI Account is automatically created when user registers.



GET http://localhost:9090/api/upi/transactions/all

○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```
Ctrl+Alt+P for Postbot
```

200 OK · 61 ms · 2.42 KB

{} JSON

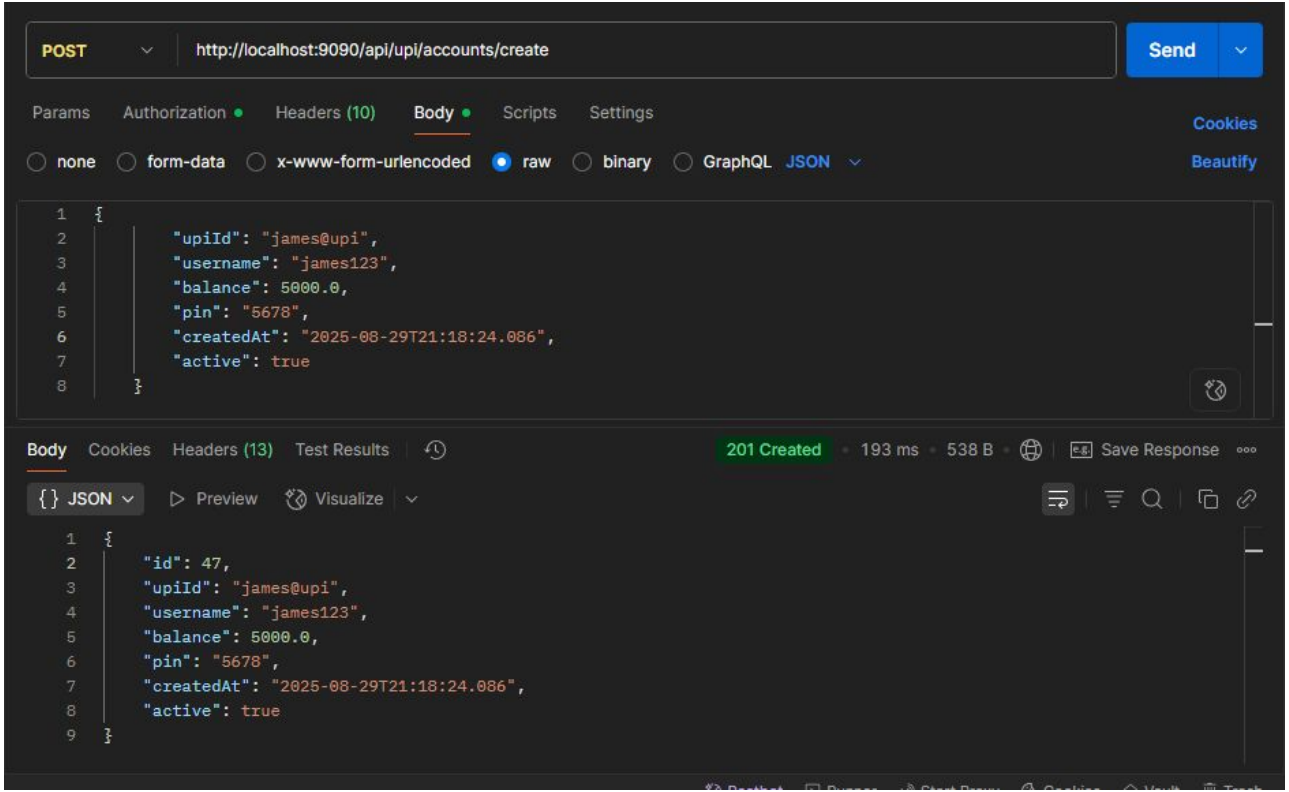
```
[
    {
        "transactionId": 6,
        "type": "UPI",
        "amount": 12.0,
        "status": "COMPLETED",
        "date": "2025-09-08T22:38:35.360224",
        "message": "for rent",
        "username": "Vaishali",
        "senderUpiId": "Vaishali@upi",
        "receiverUpiId": "vaishali@upi"
    },
    {
        "transactionId": 7,
        "type": "UPI",
        "amount": 10.0,
        "status": "PENDING",
        "date": "2025-09-08T23:21:15.434521",
```

Sprint 2

All Transactions

The screenshot shows all the upi transactions fetched using get method to the following URL:

http://localhost:9090/api/upi/transactions/all



POST http://localhost:9090/api/upi/accounts/create

Body: ● raw JSON

```
{
        "upiId": "james@upi",
        "username": "james123",
        "balance": 5000.0,
        "pin": "5678",
        "createdAt": "2025-08-29T21:18:24.086",
        "active": true
    }
```

201 Created · 193 ms · 538 B

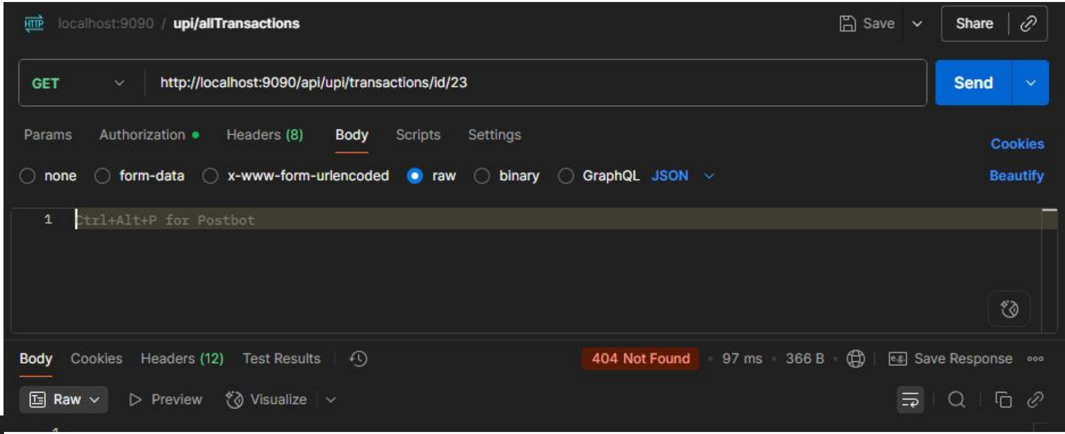
```
{
    "id": 47,
    "upiId": "james@upi",
    "username": "james123",
    "balance": 5000.0,
    "pin": "5678",
    "createdAt": "2025-08-29T21:18:24.086",
    "active": true
}
```

Sprint 2

Finding transaction by ID

The screenshot shows the upi transaction fetched using id by get method to the following URL:

http://localhost:9090/api/upi/transactions/id/{id}



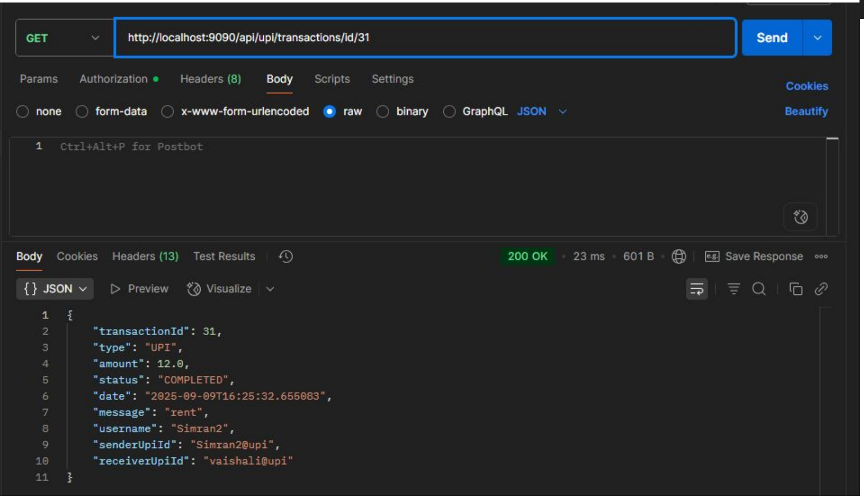
localhost:9090 / upi/allTransactions

GET http://localhost:9090/api/upi/transactions/id/23

Body: ○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```
Ctrl+Alt+P for Postbot
```

404 Not Found · 97 ms · 366 B



GET http://localhost:9090/api/upi/transactions/id/31

Body: ○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```
Ctrl+Alt+P for Postbot
```

200 OK · 23 ms · 601 B

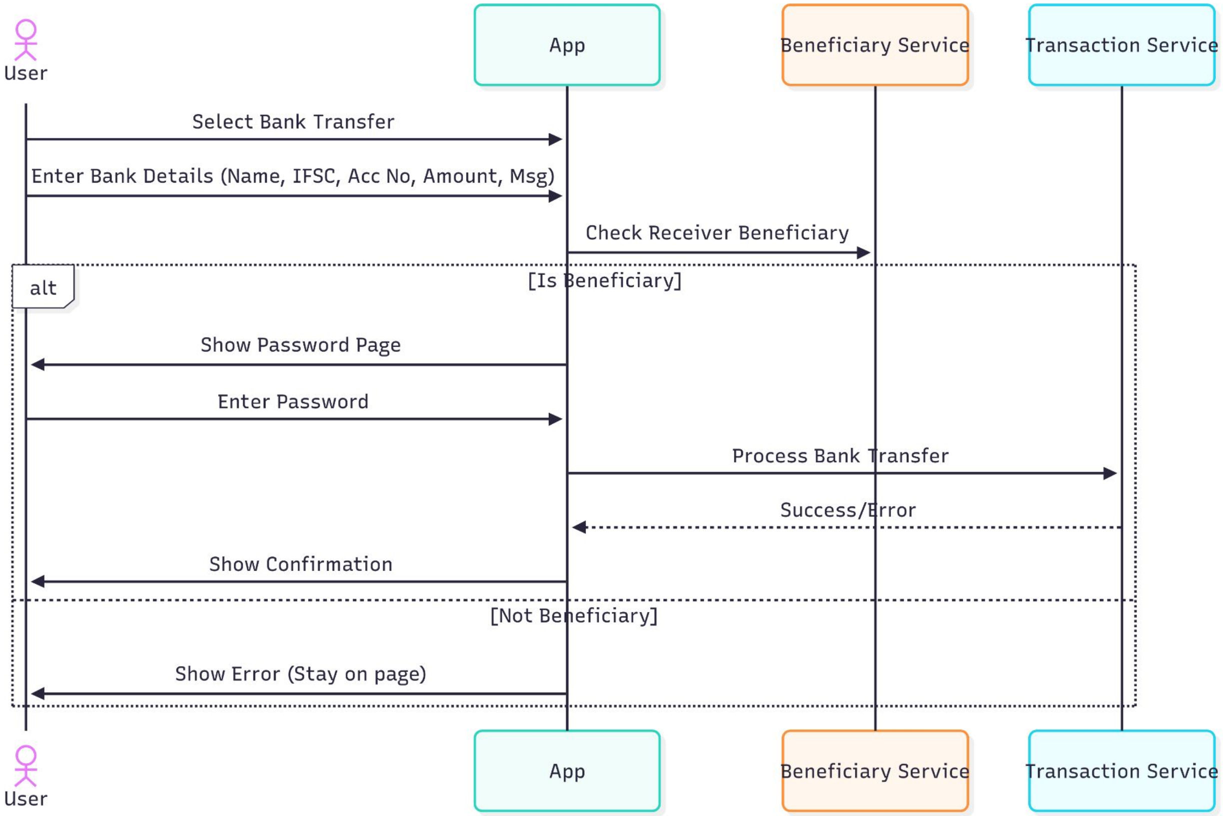
```
{
    "transactionId": 31,
    "type": "UPI",
    "amount": 12.0,
    "status": "COMPLETED",
    "date": "2025-09-09T16:25:32.655083",
    "message": "rent",
    "username": "Simran2",
    "senderUpiId": "Simran2@upi",
    "receiverUpiId": "vaishali@upi"
}
```

If a transaction is not found, response is returned with status code 404.

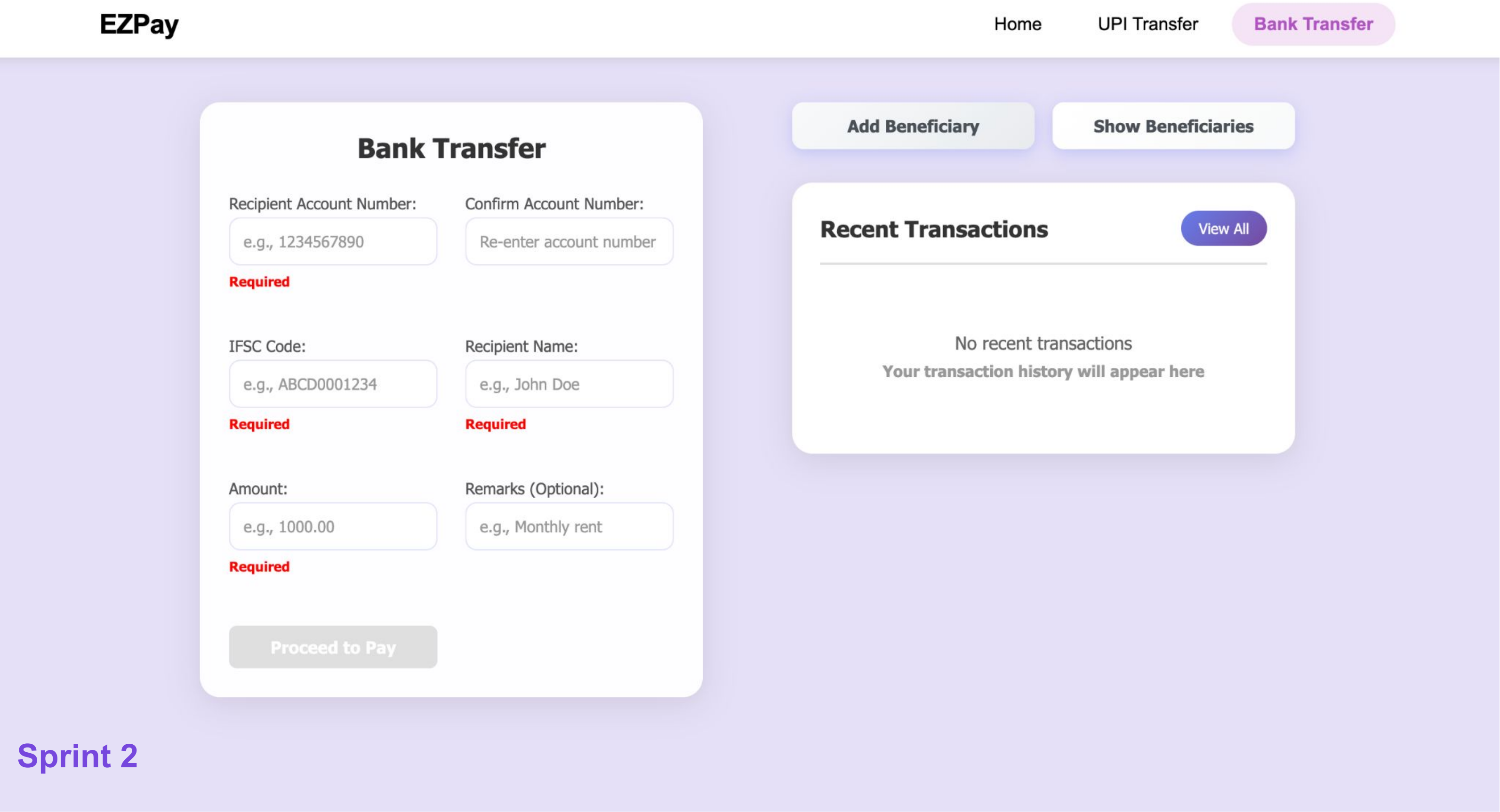
Conversely, if the transaction is found, response is returned with status code 200.

Sprint 2

Bank Transfer Sequence Diagram



Bank Transfer Page with Validations



Bank Transfer

Recipient Account Number: e.g., 1234567890
Required

Confirm Account Number: Re-enter account number

IFSC Code: e.g., ABCD0001234
Required

Recipient Name: e.g., John Doe
Required

Amount: e.g., 1000.00
Required

Remarks (Optional): e.g., Monthly rent

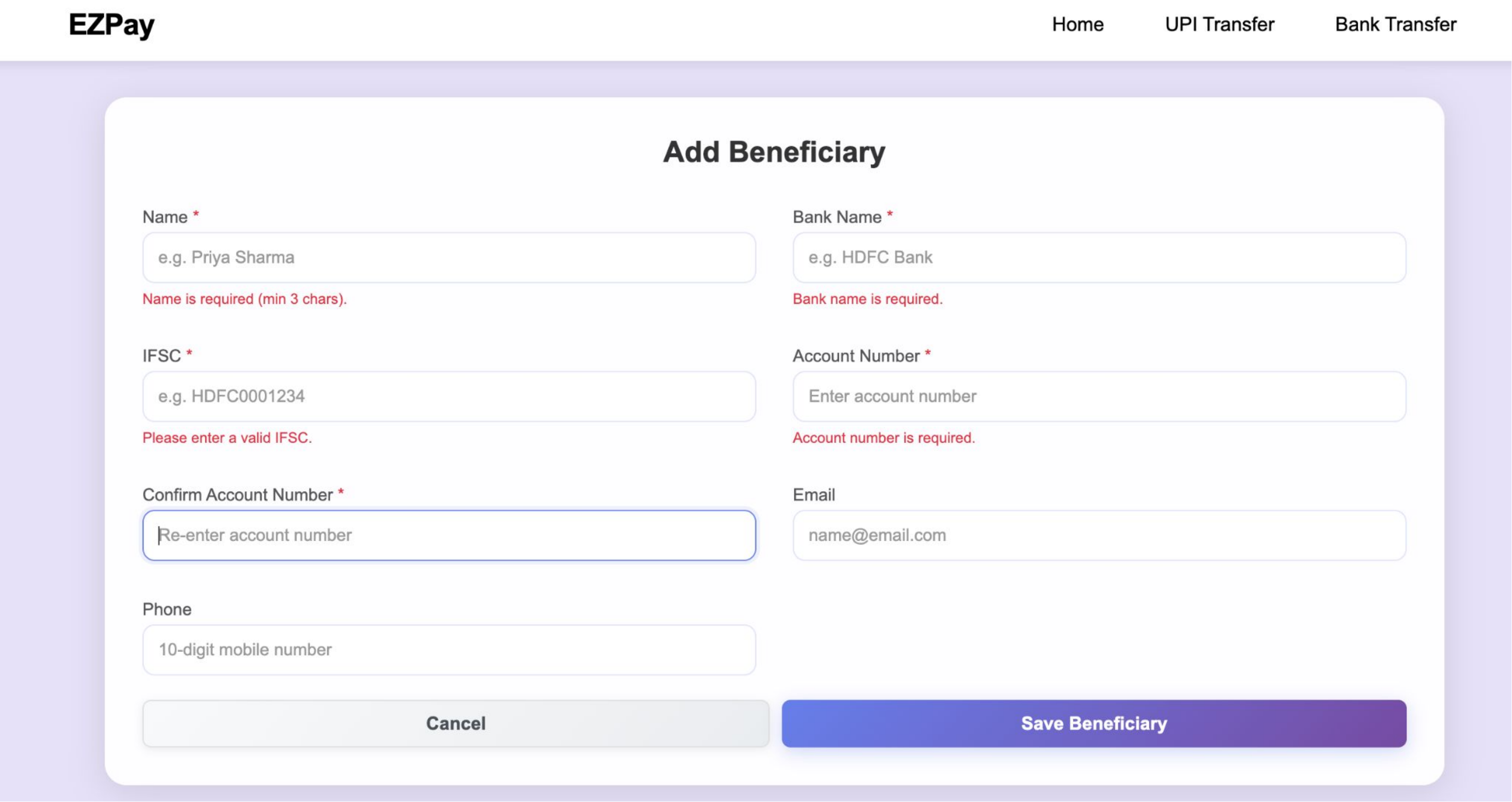
Recent Transactions

No recent transactions

Your transaction history will appear here

Sprint 2

Add Beneficiary Page with Validations



Add Beneficiary

Name *
e.g. Priya Sharma
Name is required (min 3 chars).

Bank Name *
e.g. HDFC Bank
Bank name is required.

IFSC *
e.g. HDFC0001234
Please enter a valid IFSC.

Account Number *
Enter account number
Account number is required.

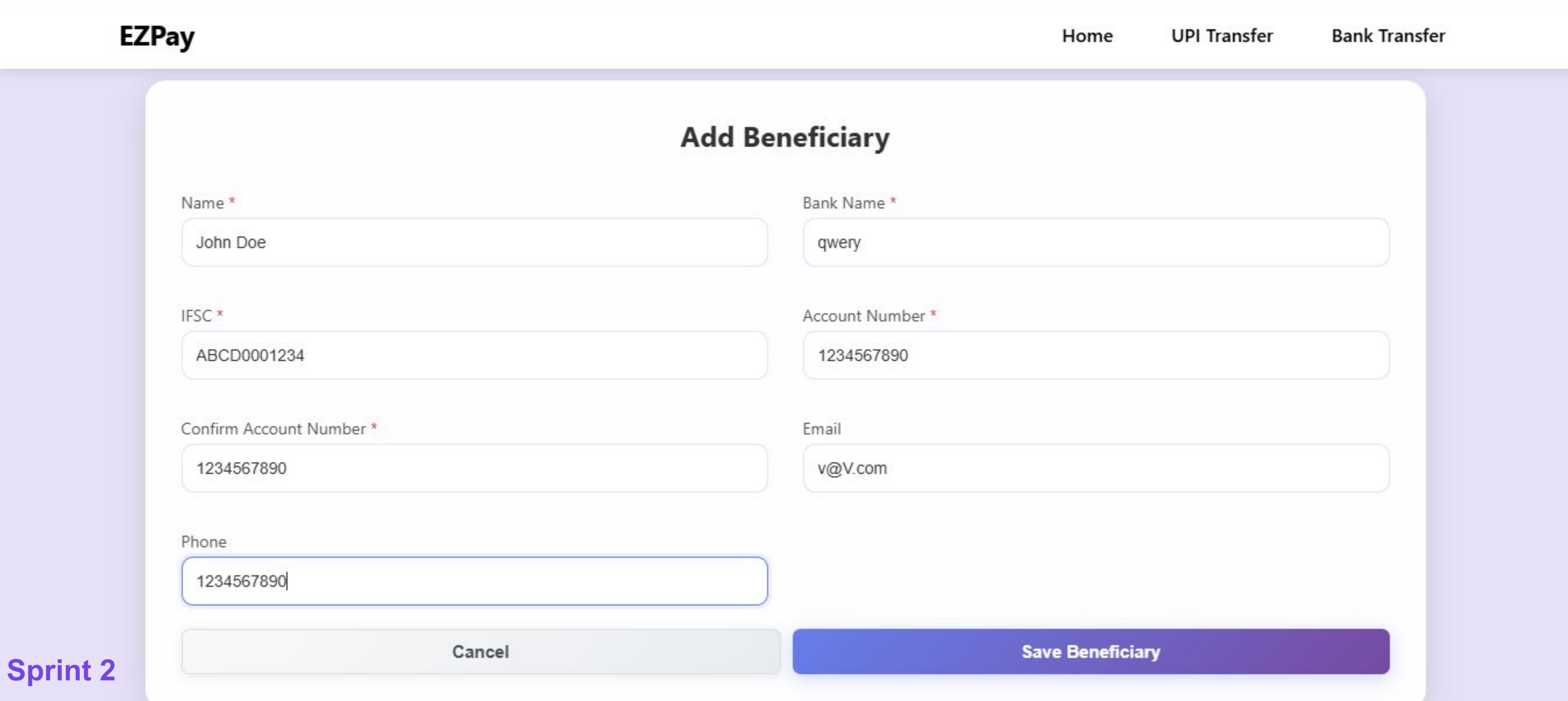
Confirm Account Number *
Re-enter account number

Email
name@email.com

Phone
10-digit mobile number

Sprint 2

Add Beneficiary Page with data



Add Beneficiary

Name * John Doe

Bank Name * qwery

IFSC * ABCD0001234

Account Number * 1234567890

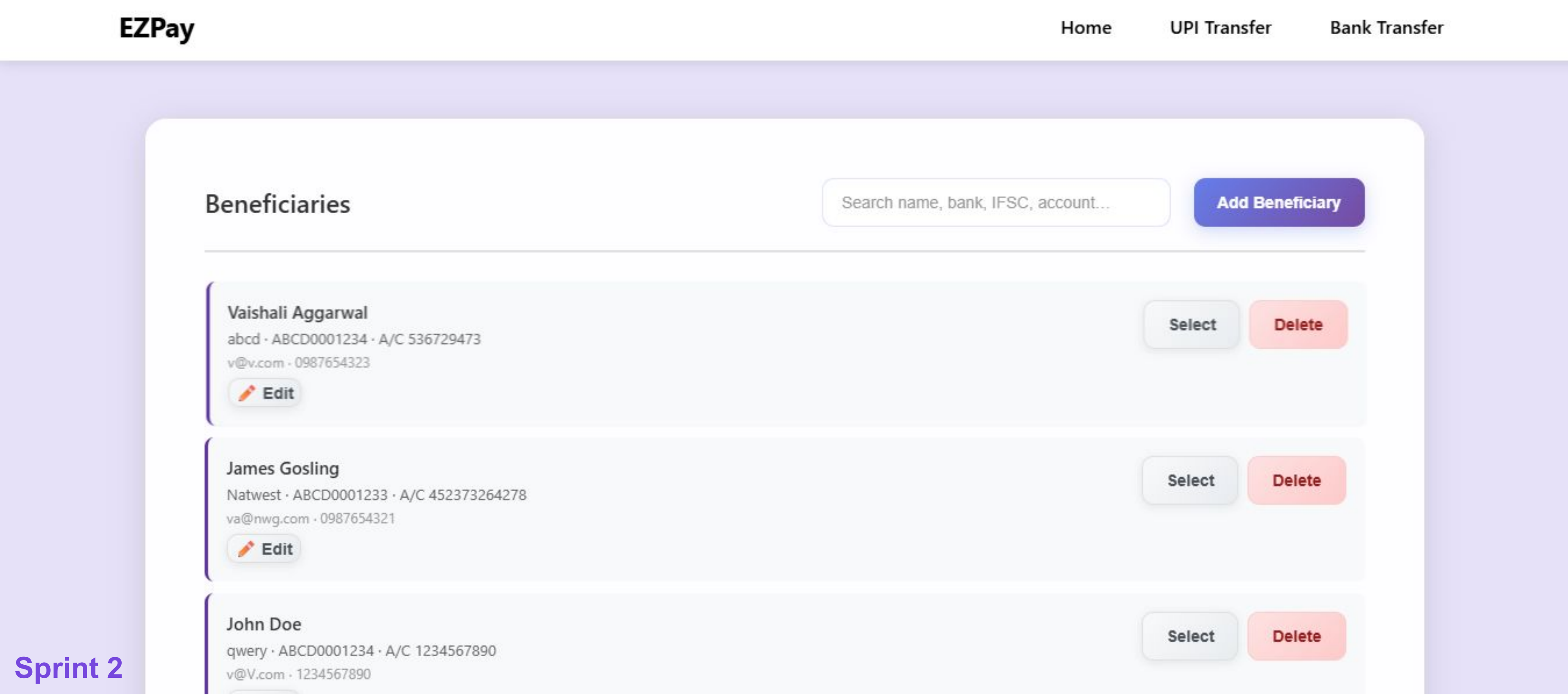
Confirm Account Number * 1234567890

Email v@V.com

Phone 1234567890

Sprint 2

Show Beneficiary Page



Beneficiaries

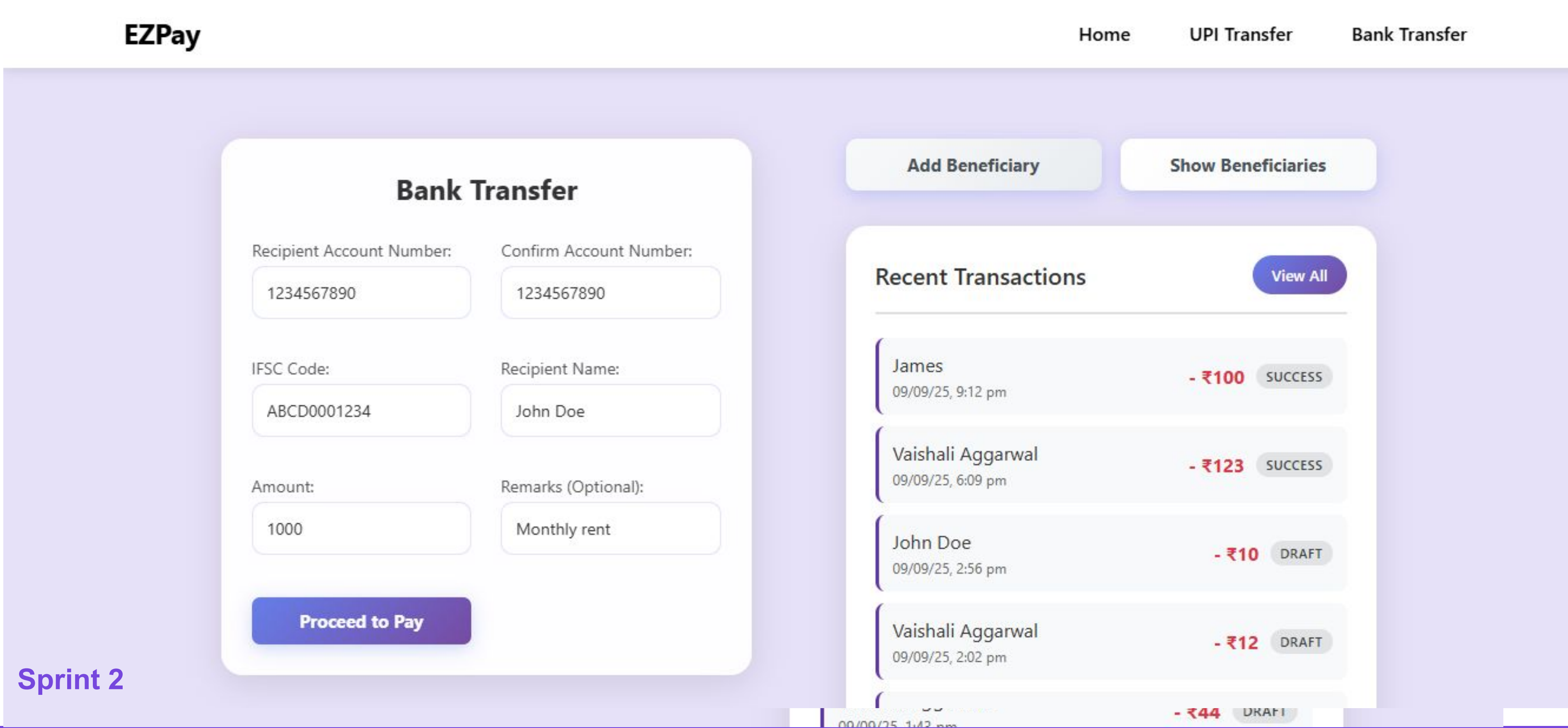
Vaishali Aggarwal
abcd · ABCD0001234 · A/C 536729473
v@v.com · 0987654323

James Gosling
Natwest · ABCD0001233 · A/C 452373264278
va@nwg.com · 0987654321

John Doe
qwery · ABCD0001234 · A/C 1234567890
v@V.com · 1234567890

Sprint 2

Pre Filled Data on Bank Transfer Page



Bank Transfer

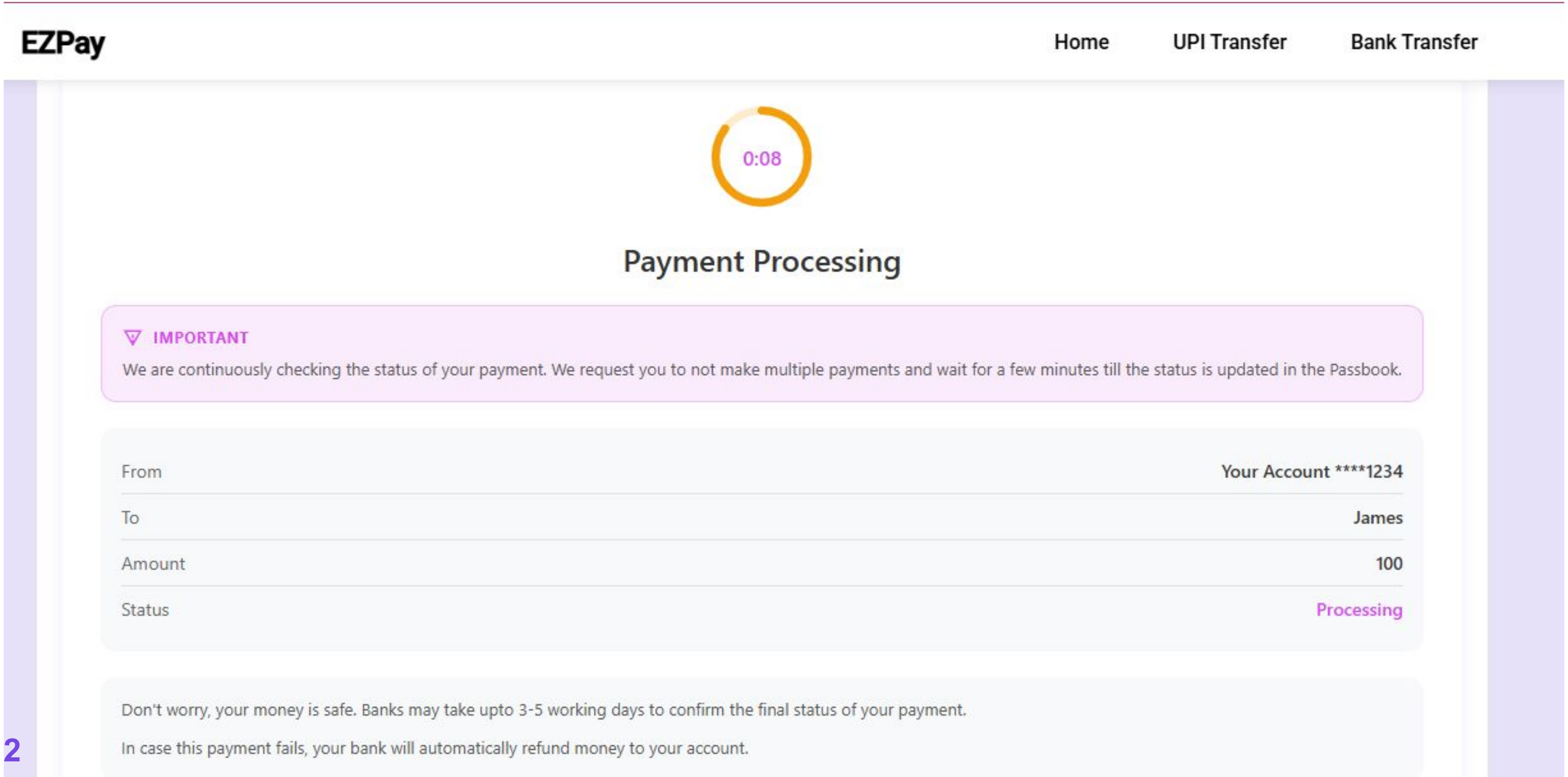
Recipient Account Number: 1234567890	Confirm Account Number: 1234567890
IFSC Code: ABCD0001234	Recipient Name: John Doe
Amount: 1000	Remarks (Optional): Monthly rent

Recent Transactions

James 09/09/25, 9:12 pm	- ₹100	SUCCESS
Vaishali Aggarwal 09/09/25, 6:09 pm	- ₹123	SUCCESS
John Doe 09/09/25, 2:56 pm	- ₹10	DRAFT
Vaishali Aggarwal 09/09/25, 2:02 pm	- ₹12	DRAFT
09/09/25, 1:43 pm	- ₹44	DRAFT

Sprint 2

Payment Processing Page



Payment Processing

IMPORTANT

We are continuously checking the status of your payment. We request you to not make multiple payments and wait for a few minutes till the status is updated in the Passbook.

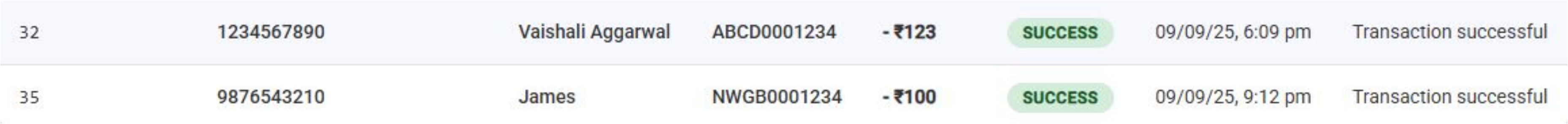
From	Your Account ****1234
To	James
Amount	100
Status	Processing

Don't worry, your money is safe. Banks may take upto 3-5 working days to confirm the final status of your payment.

In case this payment fails, your bank will automatically refund money to your account.

Sprint 2

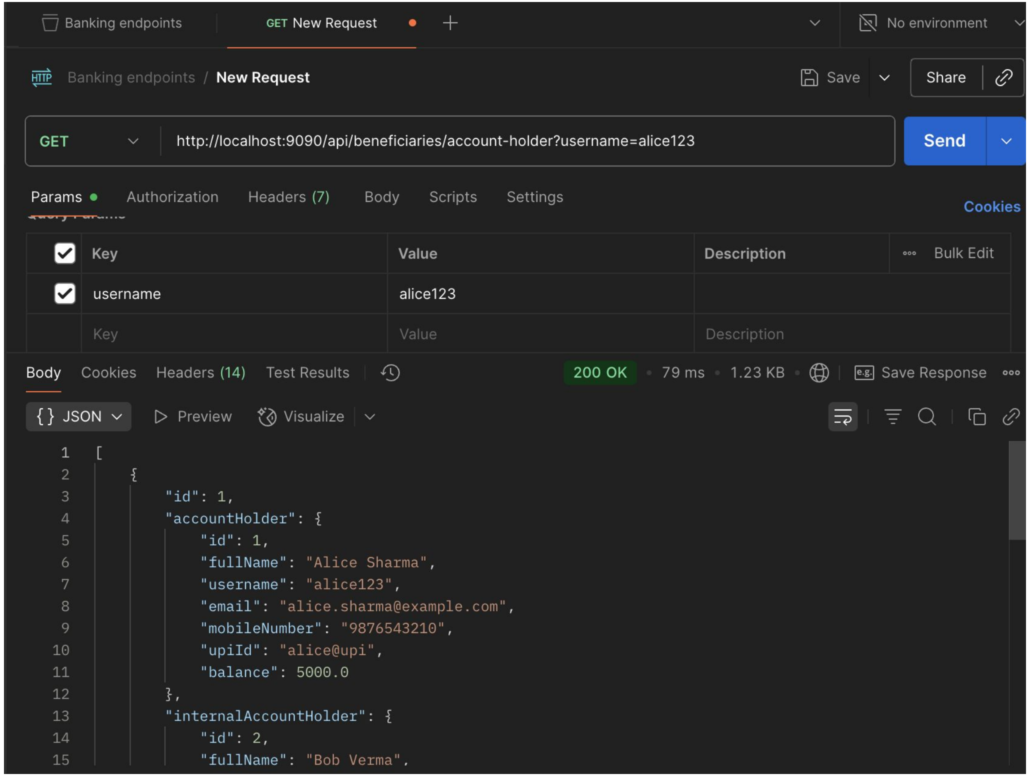
Records of Successful Transaction



32	1234567890	Vaishali Aggarwal	ABCD0001234	- ₹123	SUCCESS	09/09/25, 6:09 pm	Transaction successful
35	9876543210	James	NWGB0001234	- ₹100	SUCCESS	09/09/25, 9:12 pm	Transaction successful

Sprint 2

Postman Screenshots - basic CRUD endpoints for Banking



GET http://localhost:9090/api/beneficiaries/account-holder?username=alice123

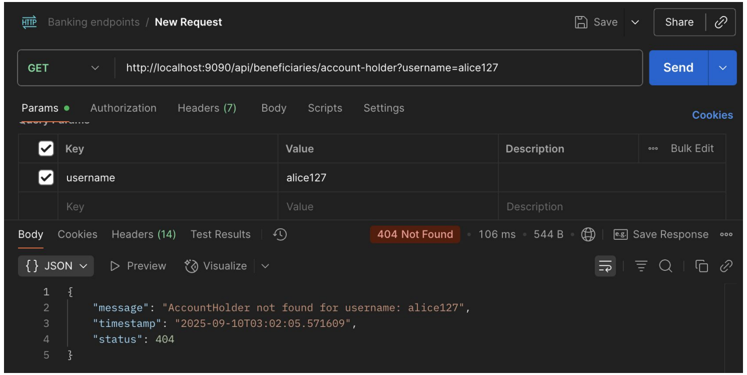
☑	Key	Value	Description
☑	username	alice123	
	Key	Value	Description

200 OK • 79 ms • 1.23 KB

```
[
    {
        "id": 1,
        "accountHolder": {
            "id": 1,
            "fullName": "Alice Sharma",
            "username": "alice123",
            "email": "alice.sharma@example.com",
            "mobileNumber": "9876543210",
            "upiId": "alice@upi",
            "balance": 5000.0
        },
        "internalAccountHolder": {
            "id": 2,
            "fullName": "Bob Verma",
```

Beneficiaries corresponding to a specific username (alice123 here) accessed using the url http://localhost:9090/api/beneficiaries/account-holder?username=alice123

Exceptions handled gracefully (stubbings returned)



GET http://localhost:9090/api/beneficiaries/account-holder?username=alice127

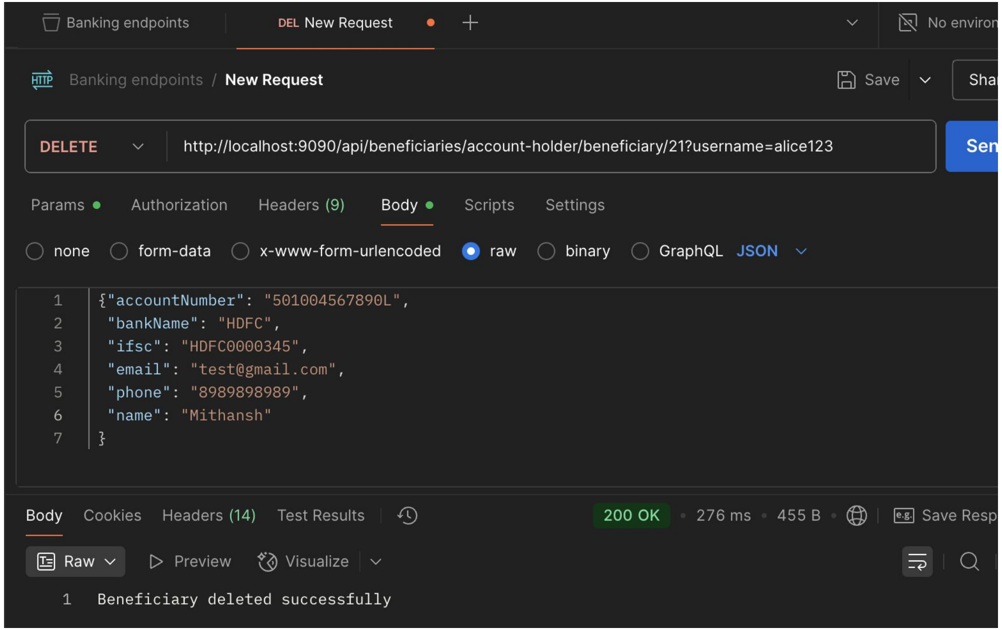
☑	Key	Value	Description
☑	username	alice127	
	Key	Value	Description

404 Not Found • 106 ms • 544 B

```
{
    "message": "AccountHolder not found for username: alice127",
    "timestamp": "2025-09-10T03:02:05.571609",
    "status": 404
}
```

Sprint 2

Postman Screenshots - basic CRUD endpoints for Banking



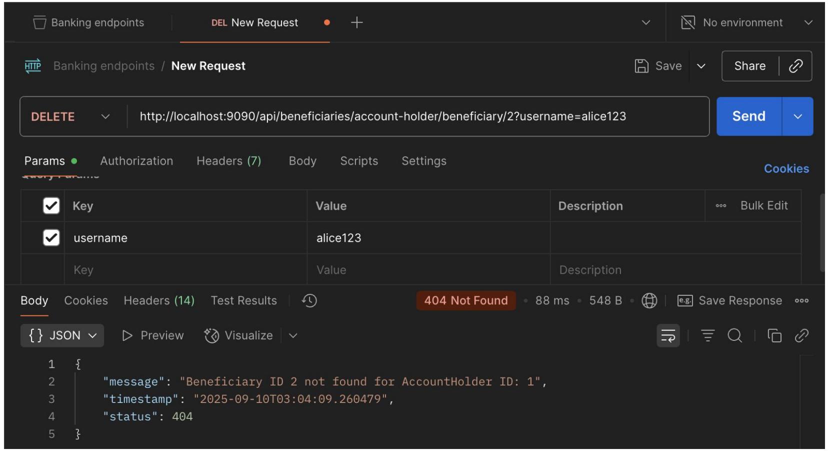
DELETE http://localhost:9090/api/beneficiaries/account-holder/beneficiary/21?username=alice123

```
{"accountNumber": "501004567890L",
 "bankName": "HDFC",
 "ifsc": "HDFC0000345",
 "email": "test@gmail.com",
 "phone": "8989898989",
 "name": "Mithansh"
}
```

200 OK • 276 ms • 455 B

```
Beneficiary deleted successfully
```

Deletion of beneficiary for a specific username by the beneficiaryId http://localhost:9090/api/beneficiaries/account-holder/beneficiary/21?username=alice123



DELETE http://localhost:9090/api/beneficiaries/account-holder/beneficiary/2?username=alice123

☑	Key	Value	Description
☑	username	alice123	
	Key	Value	Description

404 Not Found • 88 ms • 548 B

```
{
    "message": "Beneficiary ID 2 not found for AccountHolder ID: 1",
    "timestamp": "2025-09-10T03:04:09.260479",
    "status": 404
}
```

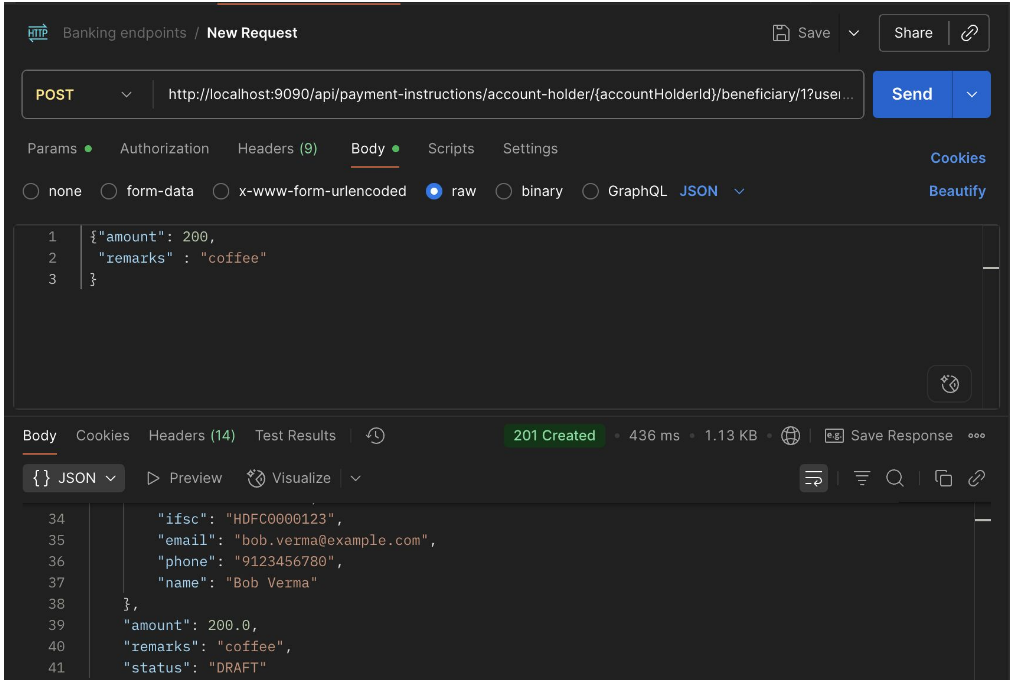
Exceptions handled gracefully

Sprint 2

Payment Instruction Creation and Transaction Execution

Career Shaper™

Amount and remarks added, and a payment instruction draft is created



POST http://localhost:9090/api/payment-instructions/account-holder/{accountHolderId}/beneficiary/1?use…

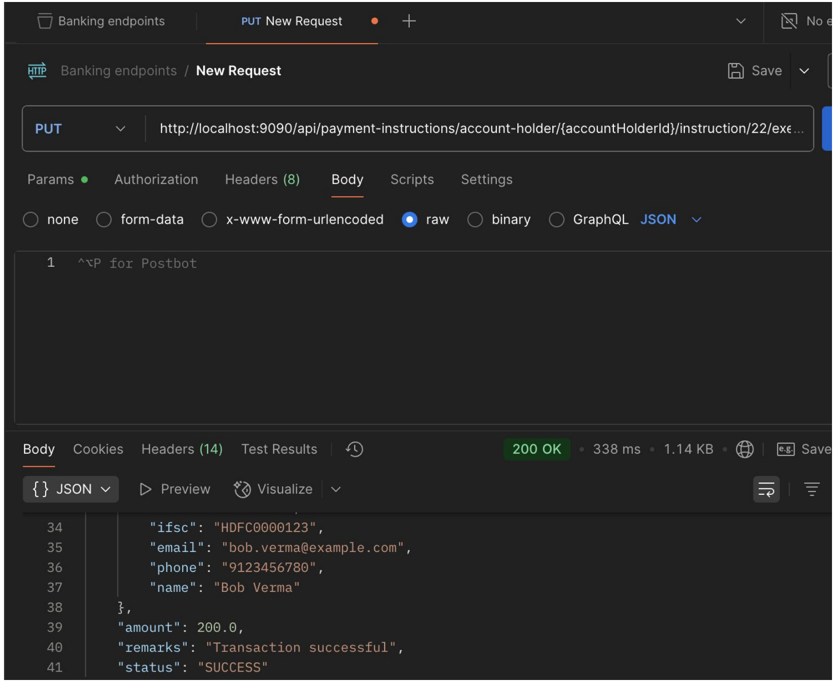
○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```
{"amount": 200,
 "remarks" : "coffee"
}
```

Body — 201 Created • 436 ms • 1.13 KB — { } JSON

```
        "ifsc": "HDFC0000123",
        "email": "bob.verma@example.com",
        "phone": "9123456780",
        "name": "Bob Verma"
    },
    "amount": 200.0,
    "remarks": "coffee",
    "status": "DRAFT"
```

A transfer is made and the status is updated



PUT http://localhost:9090/api/payment-instructions/account-holder/{accountHolderId}/instruction/22/exe…

○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

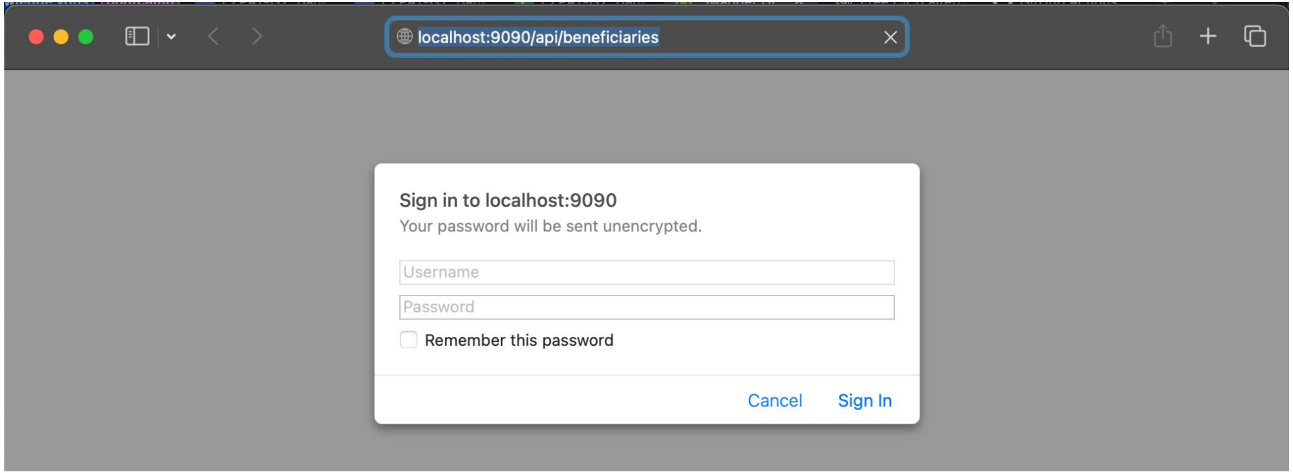
Body — 200 OK • 338 ms • 1.14 KB — { } JSON

```
        "ifsc": "HDFC0000123",
        "email": "bob.verma@example.com",
        "phone": "9123456780",
        "name": "Bob Verma"
    },
    "amount": 200.0,
    "remarks": "Transaction successful",
    "status": "SUCCESS"
```

Sprint 2

Spring Security incorporated for endpoints that could leak confidential data

Authentication is prompted to view all the account holders and their corresponding beneficiaries
Implemented using HTTP Basic Authentication for now



Sign in to localhost:9090

Your password will be sent unencrypted.

Username

Password

□ Remember this password

1. **Get all account holders**
   `GET /api/account-holders`

2. **Get account holder by ID**
   `GET /api/account-holders/{id}`

3. Delete account holder
`DELETE /api/account-holders/{id}`

4. Get all payment instructions
`GET /api/payment-instructions`

Sprint 3

THANK YOU