

## **Transactions Dispute Portal Project**

### **Project Summary**

The **Transactions Dispute Portal** is a comprehensive solution designed to streamline the process of managing and resolving transaction disputes. It provides users with an intuitive interface to view account details, monitor disputable transactions, and initiate disputes efficiently. The portal ensures transparency and accountability by maintaining a complete dispute history, including statuses such as PENDING, APPROVED, or REJECTED.

The system is built using a microservices architecture with **Spring Boot** for backend services (Customer, Payment, and Dispute), and **Angular** for the user interface. **Redis Streams** facilitates real-time event communication between services, while **Docker** ensures seamless deployment and containerized execution of all components.

Key features include:

- Secure login (using JWT) and account overview
- Disputable transaction visibility with dispute initiation
- Real-time dispute status tracking
- Full-cycle dispute management from creation to resolution

### **How to run and use the application**

1. Download the applications from the below bitbucket links:

Customer service: <https://bitbucket.org/khayeh253/customer-service/src/master/>

Payment service: <https://bitbucket.org/khayeh253/payment-service/src/master/>

Dispute service: <https://bitbucket.org/khayeh253/disputedly-dispute-service/src/master/>

Disputedly UI: <https://bitbucket.org/khayeh253/disputedly-ui/src/master/>

Project infrastructure: <https://bitbucket.org/khayeh253/disputedly-infrastructure/src/main/>

## 2. Build the applications

Ensure you have docker installed on your machine.

Navigate to the “disputely-infrastructure” directory from the downloaded repositories as shown above; this project contains **docker-compose.yml** file and DB initialization scripts.

Open your command prompt from this path, then run the following docker command:

- **docker compose up --build**

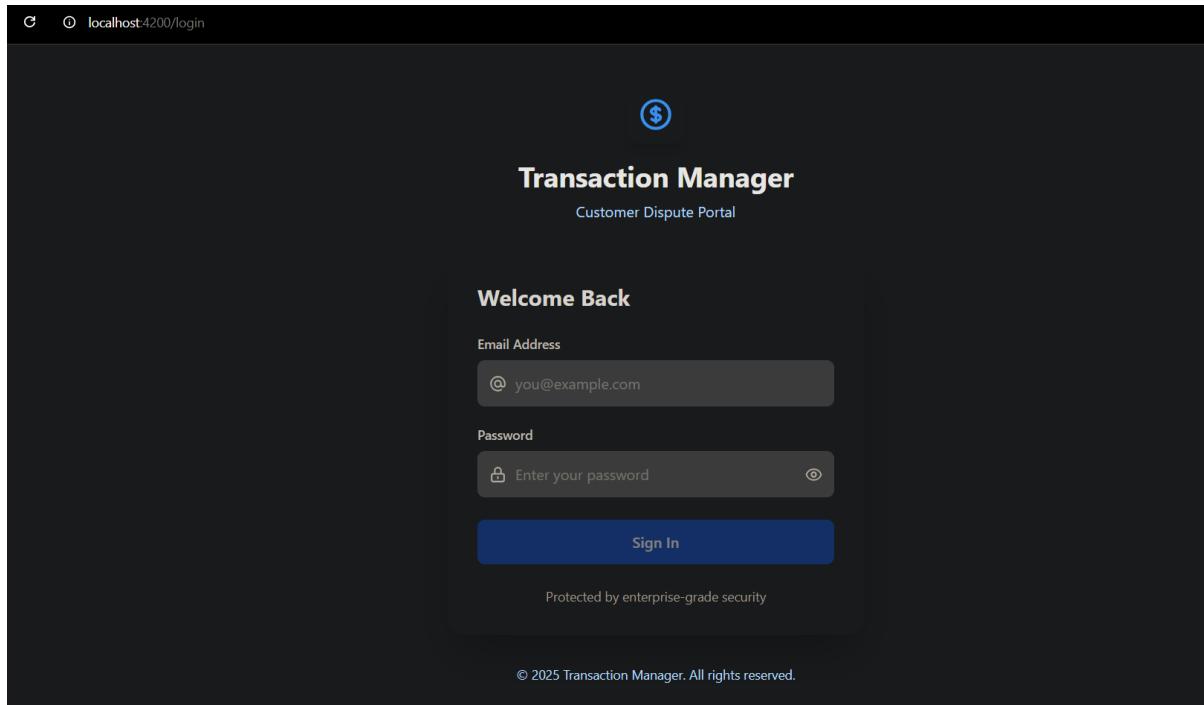
This will build and run all the applications for the “**Transactions Dispute Portal**” in the docker container.

## 3. Accessing and using the application

Open a web browser of your choice, then access the UI using this link:

<http://localhost:4200/>

This will redirect you to the login screen as show on the screenshot below:



You will need to login using the any of the following credentials (created during DB initialization):

Alice Johnson

- username: [alice@example.com](mailto:alice@example.com)
- password: password1

Bob Smith

- username: [bob@example.com](mailto:bob@example.com)
- password: password2

Carol Lee

- username: [carol@example.com](mailto:carol@example.com)
- password: password3

David Brown

- username: [david@example.com](mailto:david@example.com)
- password: password4

Eve Davis

- username: [eve@example.com](mailto:eve@example.com)
- password: password5

Once logged in you will see accounts for the logged in user, and account related information such as account number, balance, account type, etc; see example below:

The screenshot shows a dark-themed web application titled "Transaction Manager". At the top, it displays "Welcome, Alice Johnson" and a "Logout" link. Below the header, the title "Select an Account" is centered, with the sub-instruction "Choose an account to view transactions and manage disputes".

Two account cards are displayed side-by-side:

- SAVINGS** (Account ID: ACC100001)  
ACTIVE  
Available Balance: **R 1,000.00**  
Account ID: #1  
Created: 21 Nov 2025  
[View Transactions >](#)
- CHEQUE** (Account ID: ACC100002)  
ACTIVE  
Available Balance: **R 500.50**  
Account ID: #2  
Created: 21 Nov 2025  
[View Transactions >](#)

From the accounts shown, you can view account transactions by clicking “View Transactions” on an account. This will show only **disputable transactions** (transactions that have been disputed before are not visible) as well as **dispute history** “Disputed Transactions” tab. See example below:

The screenshot shows a web application titled "Transaction Manager" at the URL "localhost:4200/transactions". The user is logged in as "Alice Johnson" with a balance of "R 500.50". The main navigation bar includes "All Transactions" (selected), "Disputed Transactions" (with a red notification badge), and search/filter options for "Search transactions..." and "All Types". Below this, a specific transaction is displayed in a card format:

David Brown		DEBIT ORDER	PAID	R 2,000.50
Salary				
From	To			
Capitec	Absa			
ACC100002	ACC100005			
		Reference	TXN100002	Date 21 Nov 2025, 11:10
				<button>Dispute Transaction</button>

This is where the user can dispute a transaction, by clicking on the “Dispute Transaction” button displayed on each transaction.

After clicking the dispute transaction button, the user will be provided with the below screen where they can enter the dispute reason and submit the dispute; see example below:

## Dispute Transaction

X

### Transaction Details

Beneficiary:	David Brown
Amount:	R 2,000.50
Reference:	TXN100002
Date:	21 Nov 2025, 11:10

### Reason for Dispute \*

Did not authorize transaction

Cancel

Submit Dispute

This dispute will then be viewable on the dispute history tab (Disputed Transactions), with dispute details such as dispute status (PENDING, APPROVED or REJECTED) as well as disputed transaction details; see example below

Dispute #TXN100009  
Filed on 21 Nov 2025, 11:10

PENDING

Dispute Reason:  
Incorrect billing amount charged

Eve Davis EFT  
Incorrect billing

From Capitec To Capitec Reference TXN100009

R 600.00

Dispute #TXN100002  
Filed on 21 Nov 2025, 11:10

PENDING

Dispute Reason:  
Did not authorize transaction

David Brown DEBIT ORDER  
Salary

From Capitec To Absa Reference TXN100002

R 2,000.50

This creates a full cycle for creating and managing disputes.

After the user is done they can click on the Logout button on the top right-hand corner to logout.

## Key technologies used and their roles

*Application specific:*

**Spring boot:** for back-end micro services (customer service, dispute service and payment service)

**Angular:** for user interface

**Redis Streams:** for publishing (from dispute service) and consuming (by payment service) dispute creation events.

*Infrastructure:*

**Docker:** for deploying and running all the applications and their dependencies.