
Software Requirements Specification

for

Borderpay.io

Version 1.0

Prepared by

Manish Meena	200560	CSE	manishm20@iitk.ac.in
Rathod Preet	200775	CSE	preetr20@iitk.ac.in

Course: CS731

Mentor TA: Sumit Lahiri

Date: 07-04-2024

1	INTRODUCTION	1
1.1	PRODUCT SCOPE.....	1
1.2	INTENDED AUDIENCE AND DOCUMENT OVERVIEW	1
1.3	DEFINITIONS, ACRONYMS AND ABBREVIATIONS	1
1.4	DOCUMENT CONVENTIONS	1
1.5	REFERENCES AND ACKNOWLEDGMENTS.....	1
2	OVERALL DESCRIPTION	2
2.1	PRODUCT OVERVIEW	2
2.2	PRODUCT FUNCTIONALITY	2
2.3	DESIGN AND IMPLEMENTATION CONSTRAINTS.....	2
2.4	ASSUMPTIONS AND DEPENDENCIES	2
3	SPECIFIC REQUIREMENTS	3
3.1	EXTERNAL INTERFACE REQUIREMENTS.....	3
3.1.1	<i>User Interfaces</i>	3
3.1.2	<i>Hardware Interfaces</i>	3
3.1.3	<i>Software Interfaces</i>	3
3.2	FUNCTIONAL REQUIREMENTS	4
4	OTHER NON-FUNCTIONAL REQUIREMENTS.....	5
4.1	PERFORMANCE REQUIREMENTS	5
4.2	SAFETY AND SECURITY REQUIREMENTS	5
4.3	SOFTWARE QUALITY ATTRIBUTES	5
5.	Log In/Sign Up Page.....	6
6.	Backend	

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.0	Manish Meena, Rathod Preet	The software supports creating accounts for Employers, Employees and Banks (Routing Banks, Central Banks and Forex Banks). It also supports creating contracts between Employees and Employers and general banking operations including domestic and cross-border payments in a decentralized manner using Hyperledger Fabric.	07/04/2024

1 Introduction

1.1 Product Scope

- **Borderpay.io** is a blockchain based App where users can create contracts and get paid or pay to facilitate Remote Working and payment settlement.
- The app supports all types of currency.
- It will charge transaction fee for cross-border payments and non-member bank transactions.
- It supports basic banking operations like deposit, withdraw, send and receive money.
- Users (both Employee and Employer) can revoke the contract any time after the first payment.
- All the contract details will remain private to only parties involved.
- It supports automated tax deduction, ensuring compliance and streamlining financial processes for both employees and employers.

1.2 Intended Audience and Document Overview

This document is intended for all employers, employees, freelancers and banks who want to upgrade to a decentralized solution for creating contracts and settling payments using Fiat money.

1.3 Definitions, Acronyms and Abbreviations

App - Application

1.4 Document Conventions

Bold: methods and objects.

Highlighted (yellow): important.

Highlighted (red): security related.

Highlighted (turquoise): final changes.

1.5 References and Acknowledgments

<we will keep on updating this, with every website/ documentation which we will go through for help regarding the project>

- <https://www.linkedin.com/pulse/cross-border-funds-transfer-cbft-using-hyperledger-fabric-shimpi/> : used to understand the problem statement more clearly.
- <https://hyperledger-fabric.readthedocs.io/en/release-2.5/> : referred while coding.

2 Overall Description

2.1 Product Overview

Introducing a ground-breaking decentralized app powered by Hyperledger Fabric, enabling seamless contract creation between employees and employers. With integrated basic banking operations such as deposit, withdraw, and peer-to-peer transactions on the blockchain, users enjoy the added flexibility of choosing their preferred currency. Experience secure, transparent, and efficient financial interactions like never before (as of now).

2.2 Product Functionality

- *Seamlessly manage contracts and financial operations for multiple employees and employers with ease.*
- *The app facilitates automated payments at intervals defined within the contract.*
- *Users have the flexibility to send or receive payments to and from non-member banks.*
- *Preliminary processes such as interviews and job openings are not currently supported prior to contract creation.*

2.3 Design and Implementation Constraints

Language Requirements: *The application will use:*

backend – Hyperledger Fabric in Golang

frontend - ReactJS

testing - JavaScript,

database - CouchDB,

Tools: *VSCode, Docker Desktop, Git*

2.4 Assumptions and Dependencies

Assumptions:

1. *Employers will have enough money to send to Employees.*
2. *The client/user will have a device with a web browser installed, to access the web-app.*
3. *Currency conversion rates will be hardcoded as of now.*
4. *Banks will charge very low transaction fees.*
5. *A uniform tax system across all countries is assumed.*
6. *Money deposits and withdraws are not monitored, adjustments made to the network database only upon mutual agreement between the bank and the client.*

Dependencies:

1. *Users will be provided with blockchain wallets for app interaction, necessitating wallet management and security awareness.*
2. *Users rely on digital identity for secure authentication and trust within the app.*
3. *Users need basic financial literacy and compliance understanding for contract creation, tax deductions, and financial transactions.*

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

First time users are required to sign-up and create a new account by providing some basic details and creating a password for the same. Old users can simply sign-in with their pre-existing account. They will be provided a user ID which would be visible to other clients along with other public details.

Employer needs to deposit some amount for emergency case.

Upon logging in, the user can initiate contract creation, approve for the contract, link account with their bank (if bank is already on the network).

Any party can revoke the contract after the first payment is completed.

An employee account can query all the details about its contract details like CTC, working hours about a particular employer or all its employers and its bank private details.

An employer account can query all the details about its contract details about a particular employee or all the employees working for it and its bank private details.

A bank can query all the details about its customer linked with the bank.

3.1.2 Hardware Interfaces

The app requires the following hardware:

- *PC / Mobile (to access the web-app through a browser)*
- *Internet connection*

3.1.3 Software Interfaces

The software interface of this app involves Hyperledger Fabric wrapped in APIs. React app utilizes these APIs to make calls, enabling changes in the blockchain state or querying blockchain data.

3.2 Functional Requirements

- 3.2.1 *User Registration: New users will be able to register through a valid email Id, they will then be needed to provide some confidential details, and a password for their account.*
- 3.2.2 *Login/ Sign In: Once a user has registered successfully, he/ she will be able to login to the app using their email Id and password.*
- 3.2.3 *Dashboard: After successful login the users will be redirected to the Dashboard, where they will be able to create contracts and manage existing contracts.*
- 3.2.4 *Contract Creation: Employer account can create a contract mentioning the CTC, duration of work, nature of work and frequency of the payment.*
- 3.2.5 *Contract Agreement: Employee account can agree the terms of the Employer and the currency will be decided based on the location of the Employee. Employee can reject the contract if any discrepancies are found. If Employee agrees, then deal is signed and approved by Employer and contract will be active and payments will be processed automatically as specified in the contract.*
- 3.2.6 *Advance payment: Employees can request advance payments, subject to employer approval. Upon approval, payments are processed, and the contract is locked in a specified duration based on the requested amount. Employer can choose how this advance payment can be done (In single transaction or multiple).*
- 3.2.7 *Dues: Due to any reasons, if the amount is not paid for 2 intervals consecutively, then the contract will be automatically revoked, and the Employer's deposit will be used to pay the Employee.*
- 3.2.8 *Banking Operations: Basic banking operations like withdraw, deposit, send and receive money will be supported with general rules like customers cannot withdraw more than the bank balance.*
- 3.2.9 *Tax: Tax will be automatically calculated and deducted during the transactions. For simplicity, we have chosen 10% of CTC to be deducted and deposited to Central Bank of the Employee's Country.*
- 3.2.10 *Transaction: Employer's bank will debit money from account and send to central bank of the Employer's country. For domestic transactions, the tax will be deducted at the central bank and remaining amount will be transferred to Employee's bank if it is part of the network, if not, the amount will be transferred to router bank, which will deduct a low transaction fee and remaining amount will be transferred to Employee's bank (outside the network). For International Transactions, central bank "A" (Employer's country) will transfer the amount to Forex Bank, which will deduct a low conversion charge and transfer the converted amount to central bank "B" (Employee's country), which will deduct the tax and remaining process will be same as domestic transactions.*

4 Other Non-functional Requirements

4.1 Performance Requirements

Response Time: The system should respond to user actions within a maximum of 3 seconds to ensure a seamless user experience.

Throughput: The system should handle a minimum of 100 transactions per second during peak usage hours to accommodate high traffic.

Scalability: The system should be able to scale horizontally to support an increasing number of users and transactions without compromising performance.

Resource Utilization: The system should efficiently utilize hardware resources to minimize server load and optimize performance.

Fault Tolerance: The system should be resilient to failures and capable of maintaining functionality in the event of hardware or network failures.

Concurrency: The system should support concurrent access by multiple users without data integrity issues or performance degradation.

4.2 Safety and Security Requirements

Data Encryption: All sensitive data, including user credentials and financial transactions, should be encrypted using industry-standard encryption algorithms to prevent unauthorized access.

Access Control: Role-based access control mechanisms should be implemented to restrict access to sensitive functionalities based on user roles (Employer, Employee, Bank).

Security Updates: Regular security updates and patches should be applied to the system to address vulnerabilities and mitigate security risks.

Secure Communication: Communication between system components should be encrypted using secure protocols to prevent eavesdropping and tampering.

Data Backup and Recovery: Regular backups of critical data should be performed to ensure data integrity and facilitate quick recovery in case of data loss or corruption.

4.3 Software Quality Attributes

Reliability: The system should be highly reliable, with minimal downtime and consistent performance under varying load conditions.

Maintainability: The codebase should be well-structured and modular to facilitate easy maintenance and future enhancements.

Usability: The user interface should be intuitive and user-friendly, with clear navigation and informative error messages to enhance user experience.

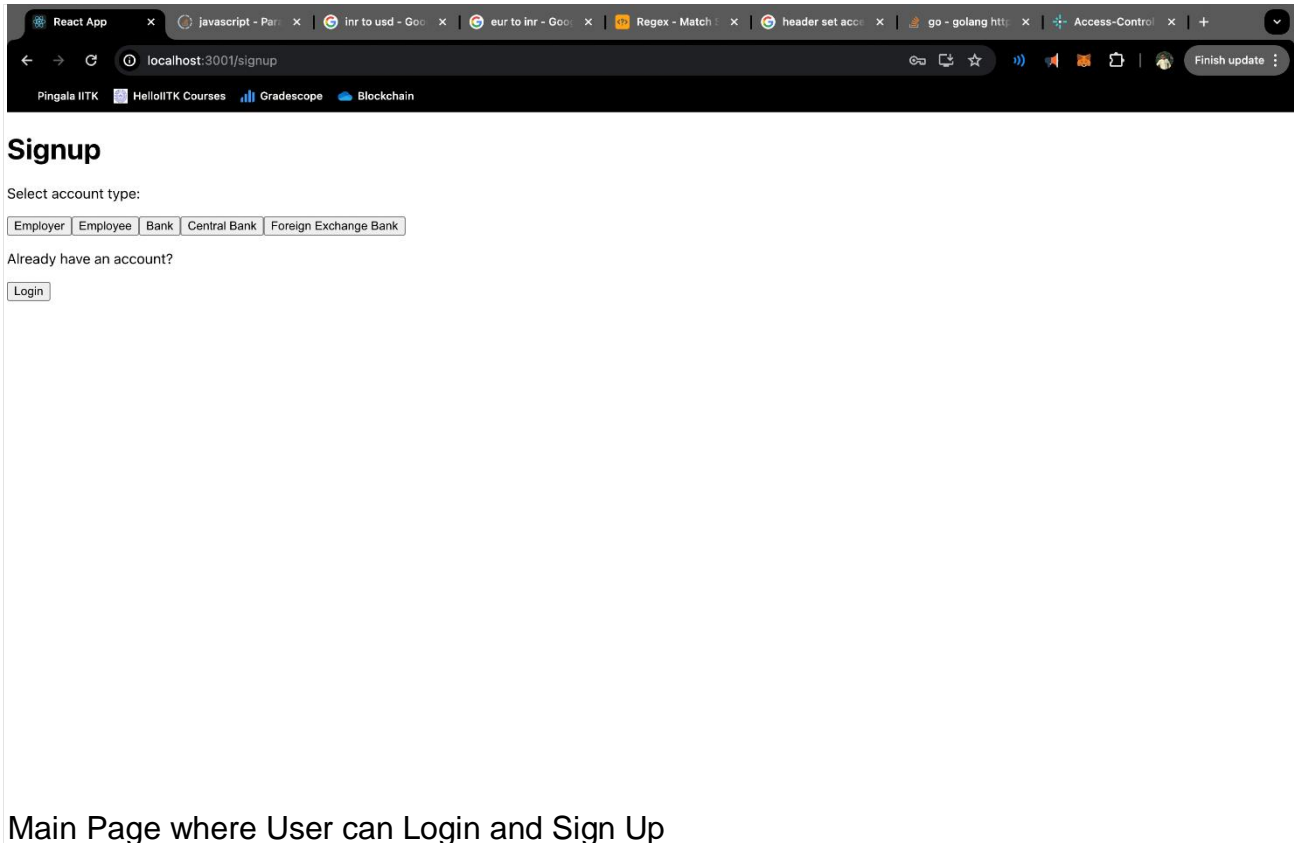
Portability: The system should be platform-independent and easily deployable across different operating systems and environments.

Interoperability: The system should seamlessly integrate with external systems and APIs to support interoperability with third-party applications and services.

Scalability: The system architecture should be designed to scale horizontally and vertically to accommodate future growth and changing requirements.

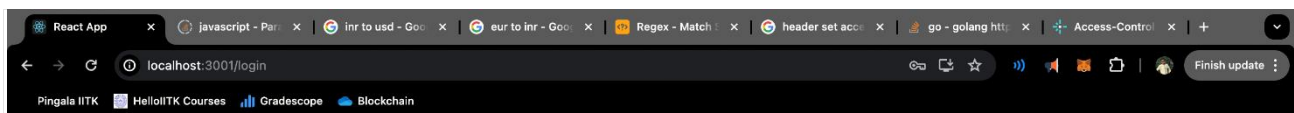
5.Log In / Sign Up Page

5.1 Main Page



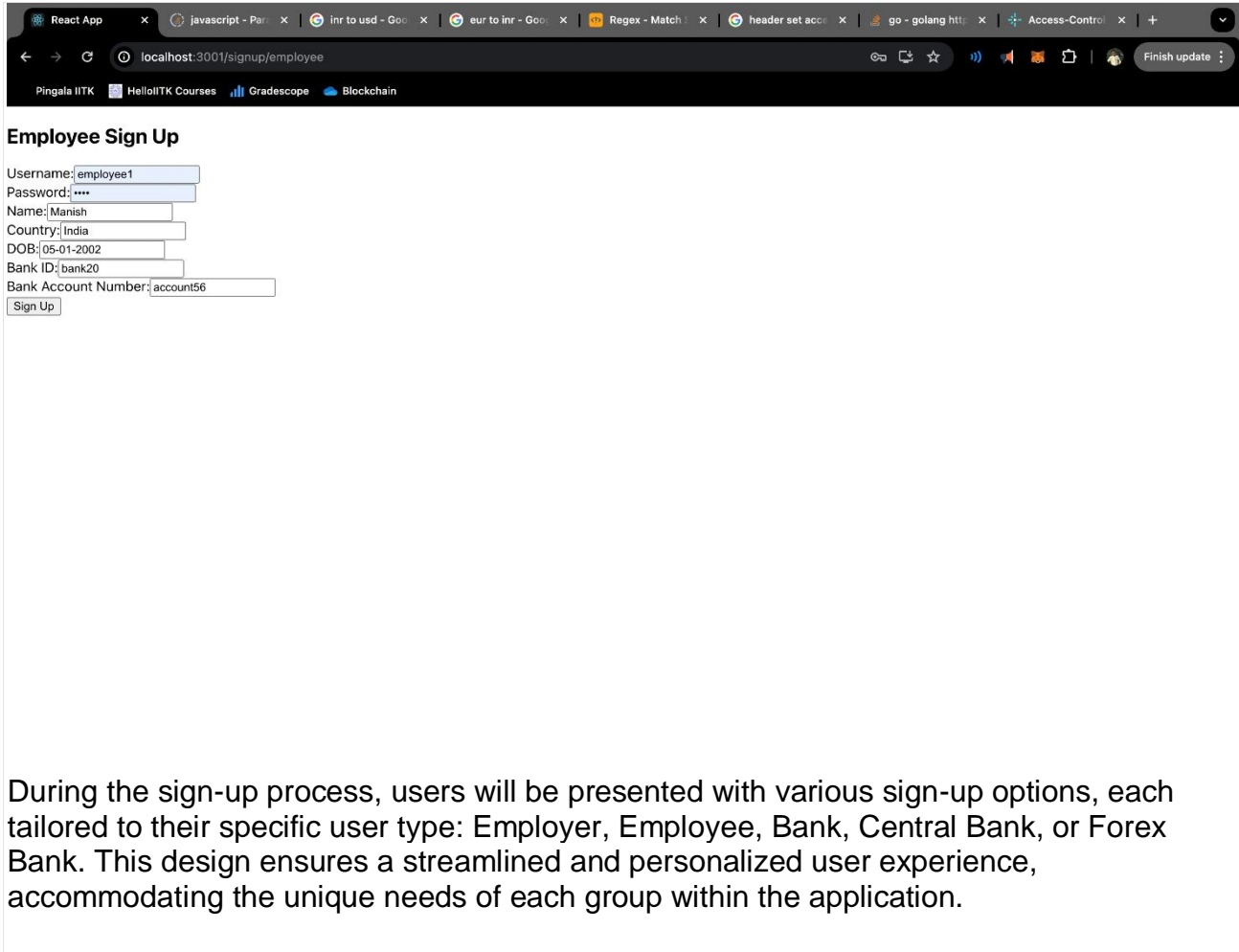
Main Page where User can Login and Sign Up

5.2 log In Page



Our application features a versatile login system that supports five distinct user types: Employer, Employee, Bank, Central Bank, and Forex Bank. All users will utilize the same login page for convenience.

5.3 Sign Up



The screenshot shows a web browser window with the address bar displaying `localhost:3001/signup/employee`. The browser's tab bar includes several open tabs: "React App", "javascript - Par...", "Inr to usd - Go...", "eur to inr - Go...", "Regex - Match...", "header set acc...", "go - golang htt...", "Access-Control...", and a plus sign for more tabs. The browser's address bar also shows a "Finish update" button. The main content area of the browser displays the "Employee Sign Up" form. The form fields are as follows:

- Username:
- Password:
- Name:
- Country:
- DOB:
- Bank ID:
- Bank Account Number:

Below the form fields is a "Sign Up" button.

During the sign-up process, users will be presented with various sign-up options, each tailored to their specific user type: Employer, Employee, Bank, Central Bank, or Forex Bank. This design ensures a streamlined and personalized user experience, accommodating the unique needs of each group within the application.

Employee Sign Up

Username:

Password:

Name:

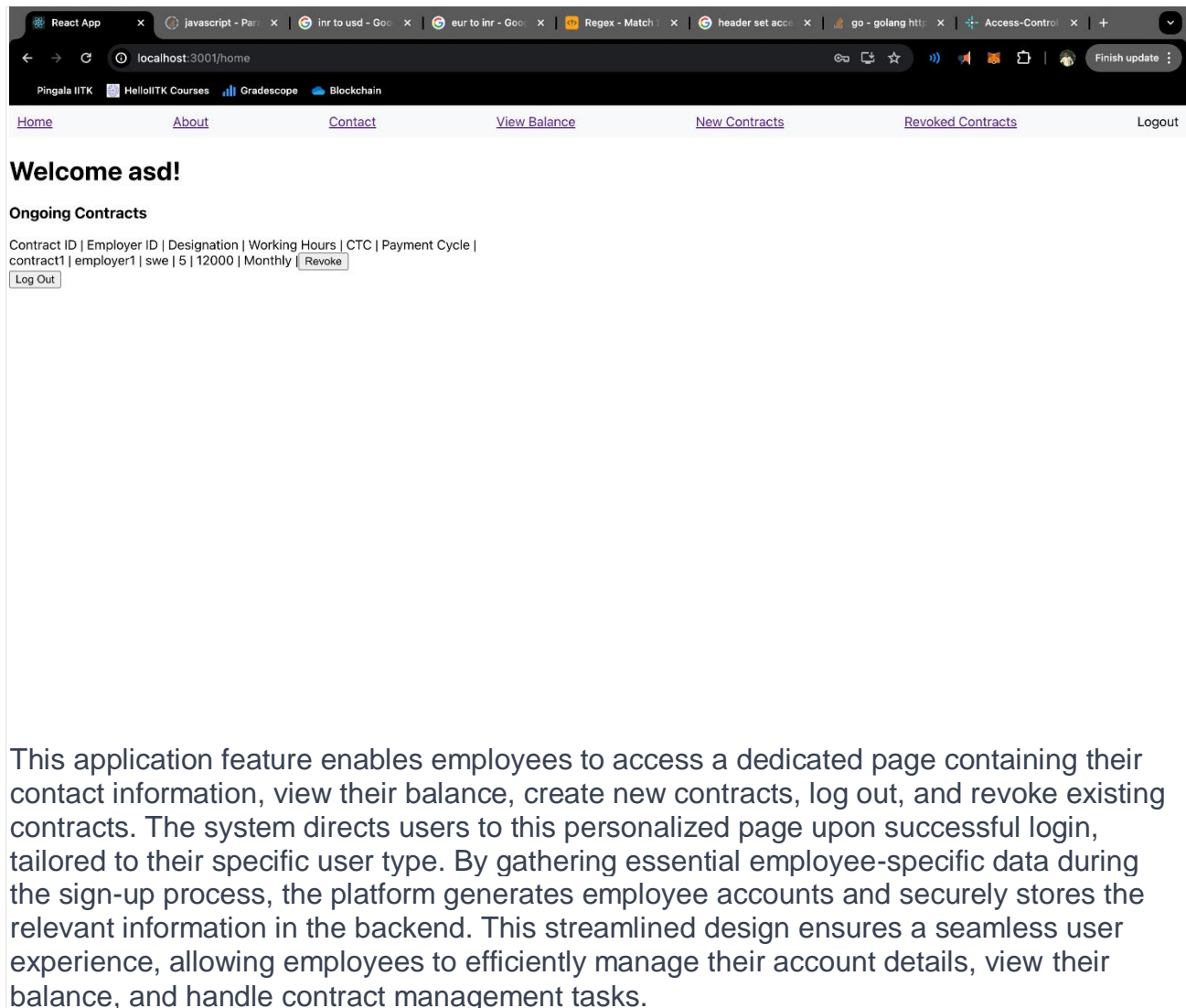
Country:

DOB:

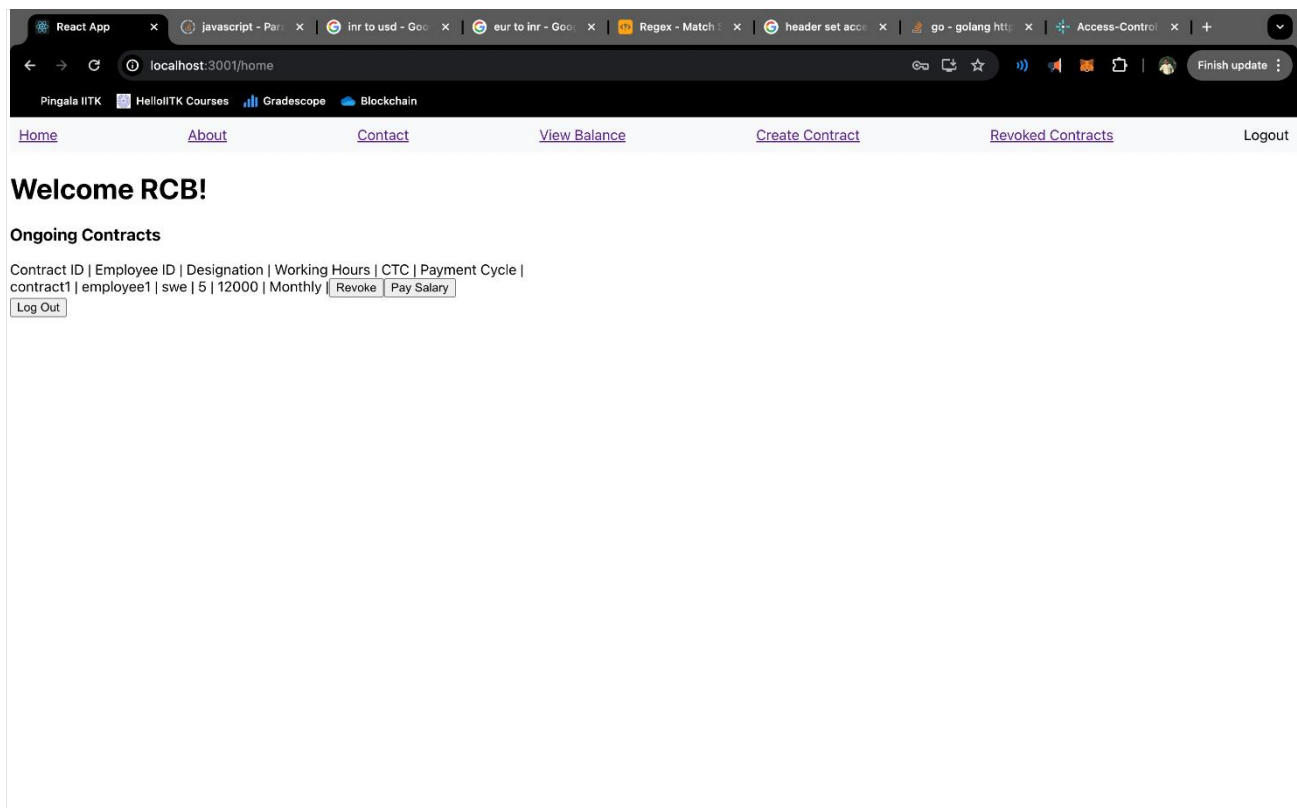
Bank ID:

Bank Account Number:

5.1 Employee Home Page



5.1 Employer Page

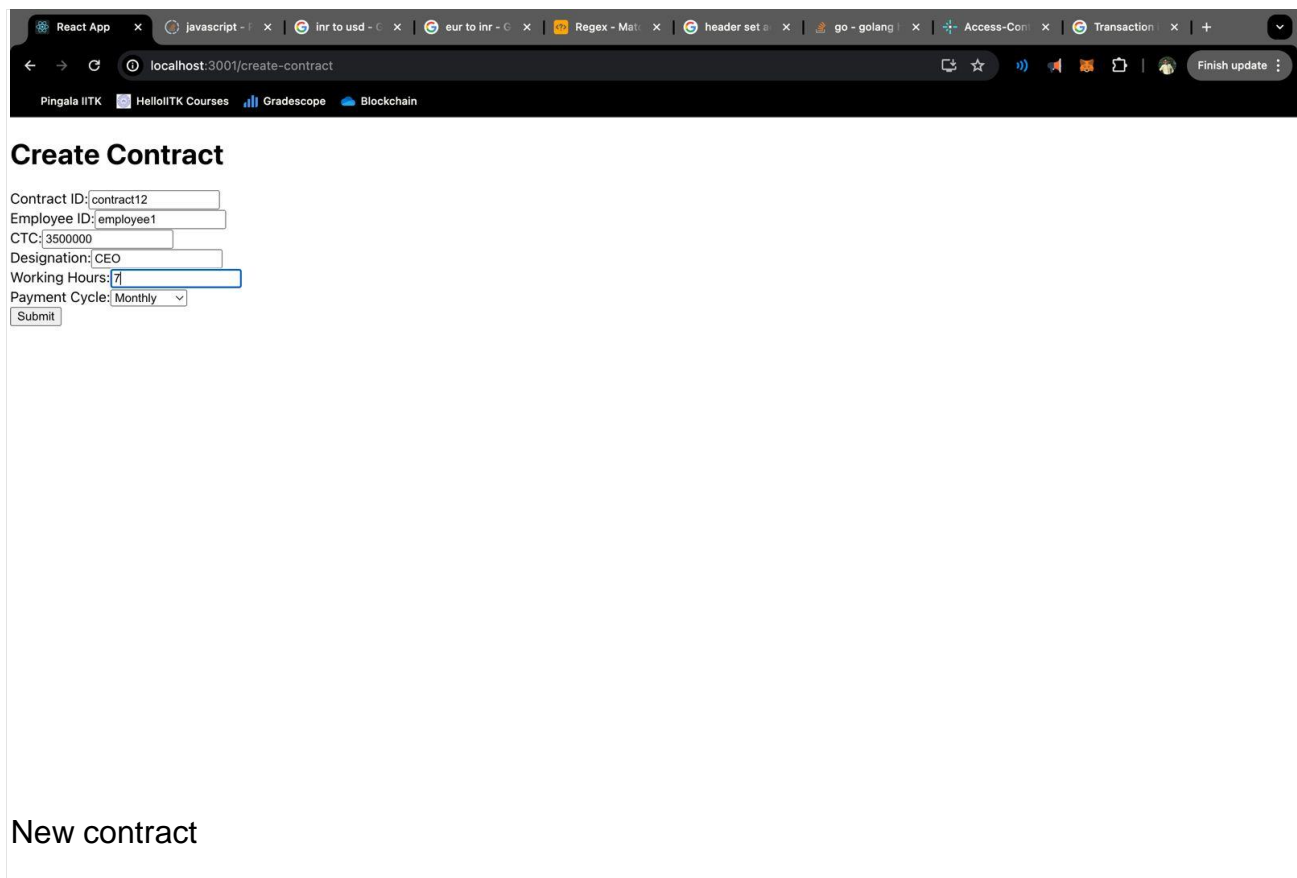


The Employer Home Page mirrors the Employee Home Page in functionality, with additional features designed specifically for employers. These enhanced features include the ability to pay employee salaries, revoke contracts, and accept or reject contract requests submitted by employees.

When an employee initiates a contract, the request is forwarded to the employer for review and consideration. Employers can then decide to accept or reject the contract proposal,

fostering a secure and efficient contract management process. By offering these tailored features, the platform facilitates seamless collaboration between employers and employees, ensuring a smooth user experience for both parties.

5.1 New Contract



The screenshot shows a web browser window with multiple tabs open. The active tab is titled 'localhost:3001/create-contract'. The browser's address bar shows the URL 'localhost:3001/create-contract'. Below the browser window, the 'Create Contract' form is displayed. The form contains the following fields and controls:

- Contract ID:
- Employee ID:
- CTC:
- Designation:
- Working Hours:
- Payment Cycle:
- Submit:

New contract

Appendix A - Group Log

- Group formation - 23/03/2024
- Idea discussion - 24/03/2024 to 27/03/2024
- Meet with TA -
- SRS document meet – 25 Jan (3 -5 pm on discord)
- 28 Jan (4-6 pm)
- 30 Jan (7 -9.30 pm)
- Discussion with TA – 30 jan
- SRS document discussion – 01 Feb (7 -8 pm)
- Frequent discussion on discord and whatsapp groups about the project.

Architecture Diagram

Transaction Use Case (International)