Full Stack Web Application for Consortium Blockchain

Final Report

Instructor

Dr. Kewei Sha

Mentor

Dr. Kwok-Bun Yue

Joses Sandeep Thamarai Selvan

Team Members

Srikar Vuppala

Tarun Jandhyala

Vivek Kukkapalli

Aravind Reddy Yakasi

Ravi Teja Nakkala

College of Science and Engineering

University of Houston -Clear Lake

2700 Bay Area Blvd, 77058

Table of Contents

Abstract

The goal of this project is to build a secure full stack web client application to interact with Blockchain Hyperledger fabric and to replicate the business process of organization for building complex systems using the MBSE process. This client application will help in managing real-world organization roles and responsibilities. Technologies used in this application are web application development using Angular, APIs (Application programming interfaces) development using Node.js, Database design and local database storage in Postgres and finally for Blockchain Hyperledger fabric using Fabcar. Organizations have distinct roles, and distinct roles have different access to the MBSE and implementation of MBSE management with creating, reading, updating, and deleting (CRUD) operations. Organization User credentials are stored in a local database (PostgreSQL). Required data for blockchain is stored in both database and wallet (File System)

1. Introduction

Blockchain is an innovative distributed global ledger technology that has caught the interest of both business and academia. It offers efficient data storage, immutability, traceability, tamper resistance, and transparency for data that can act as a general ledger for a variety of application areas. The objective of this project is to create a full-stack web client application that can interact with blockchain ledgers to leverage a blockchain network stripped down to only the components necessary to run an application.

We have taken Fabcar as a sample blockchain network to recreate business processes performing various CRUD operations used by organizations to develop sophisticated systems utilizing the MBSE process. Model-based system engineering (MBSE) is a formalized process that supports the development of complex systems' requirements, designs, analyses, verifications, and validations. Because complex systems are challenging to design, not all system components may be designed to the same high standard by one organization. Once the criteria have been defined, they should not be changed if a business wants to outsource some system components and if the system comprises secret information that must be secured throughout the firm. There are many kinds of issues in this area. With the use of blockchain technology, most issues are being resolved.

2. System Overview

2.1. Hardware Requirements

Operating System -	Ubuntu 20.04.1 and Windows 64 bit.
Processor -	2 GHz dual core
System Memory -	4 GB Ram
Hard Drive Space -	50 GB
Graphics -	VGA capable of 1024×728 resolution
USB or CD/DVD Drive -	At least one available for installer media
Internet Adapter -	Wired or wireless network

2.2. Software Requirements

2.2.1. Angular 13

Angular is an application design framework and development platform for creating single-page apps. Angular extends HTML attributes with Directives and binds data to HTML with expressions. It changes the static HTML to dynamic HTML. We used this technology to develop the user interface of the project as it is highly responsive, faster to build and easy to adapt.

2.2.2. Node JS

Node.js is an open-source, cross-platform, back-end JavaScript runtime environment and executes JavaScript code outside a web browser. It is used for server-side programming, and primarily deployed for non-blocking, event-driven servers. It is used for websites and back-end API services. We have used Node JS as middleware which interacts with frontend and blockchain.

2.2.3. PostgreSQL 13

PostgreSQL is a free and open-source relational database management system emphasizing extensibility and SQL compliance. PostgreSQL is used as the primary data store or data warehouse for many webs, mobile, geospatial, and analytics applications.

2.2.4. Blockchain (Hyperledger)

Hyperledger Fabric is a platform for distributed ledger solutions underpinned by a modular architecture delivering high degrees of confidentiality, resiliency, flexibility, and scalability. It is designed to support pluggable implementations of different components and accommodate the complexity and intricacies that exist across the economic ecosystem. We have used one of the Hyperledger fabrics called Fabcar in our project to perform operations from web client.

2.2.5. Visual Studio Code

Visual Studio Code is a source code editor that can be used with a variety of programming languages. Visual Studio Code combines the simplicity of a source code editor with powerful developer tooling, like IntelliSense code completion and debugging.

2.2.6. GitHub

One open-source version control system is Git. Git is a distributed version control system, which means that every developer's computer has access to the entire codebase and history, making branching and merging simple. GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere. We used GitHub to store the Application Files and manage the code distribution.

2.2.7. Postman

Postman is a standalone software testing API (Application Programming Interface) platform to build, test, design, modify, and document APIs. It is a simple Graphic User Interface for sending and viewing HTTP requests and responses. We used Postman, for testing purposes, one does not need to write any HTTP client network code. Instead, we build test suites called collections and let Postman interact with the API. In this tool, any functionality that any developer may need is embedded. This tool can make several types of HTTP requests like GET, POST, PUT, PATCH, and convert the API to code for languages like JavaScript and Python.

2.2.8. Shell Scripting

A shell is a command-line interpreter and typical operations performed by shell scripts include file manipulation, program execution, and printing text. The shell is a real programming language, complete with variables, control structures, and so forth. No matter how complicated a script gets, it is still just a list of commands executed sequentially.

3. System Architecture

An overview of the whole project architecture is provided below. It shows how each project's parts are related to one another. Whenever a user tries to access with a role called Admin user on the user interface which was designed using Angular and then it connects to the NodeJS at backend where we were designed the MBSE APIs that can interact with blockchain and in the comparable way backend node is connected to the Postgres DB for storing and retrieving the data. In that way, a secure full stack consortium web application was designed using the technologies mentioned in the diagram below.

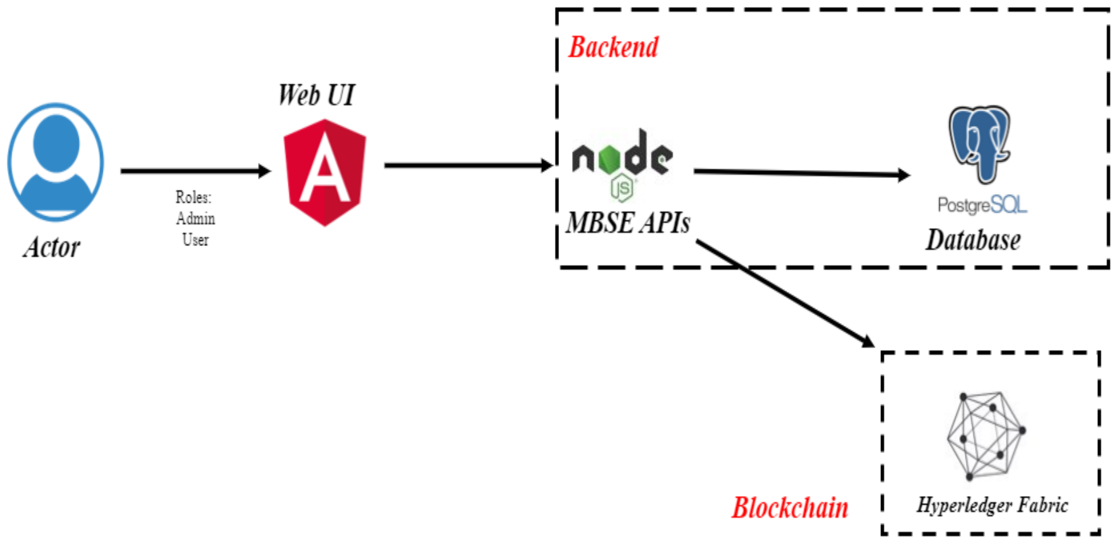


Figure 1: Application flow

4. Design Diagrams

4.1. Consortium Organization Roles

There are 2 roles for every consortium organization. Each consortium organization will have an admin with a specific authority to create a local user and map them to a fabric user. As an admin you can Create, Update, Delete and Map both Fabric User Identity and User where you can perform various creations.

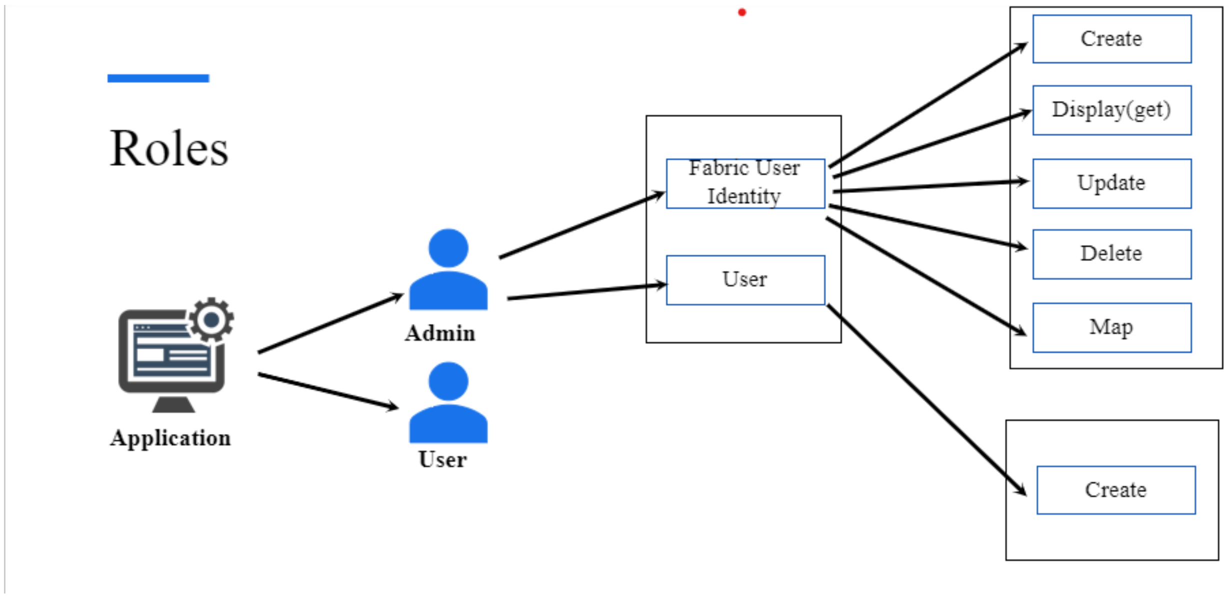


Figure 2: Organization roles

4.2. Relational Schema Designs

The relational schema diagram below shows the relationship between different entities that will communicate with each other in the application. It shows the mapping between the organization, employees and their distinct roles, and organizations with the projects. The relational schema below is built by referring to the UML diagram document made by Dr. Yue. As per this project's scope, we have not implemented the change request functionality but can be used by the future team to extend the application's functionalities.

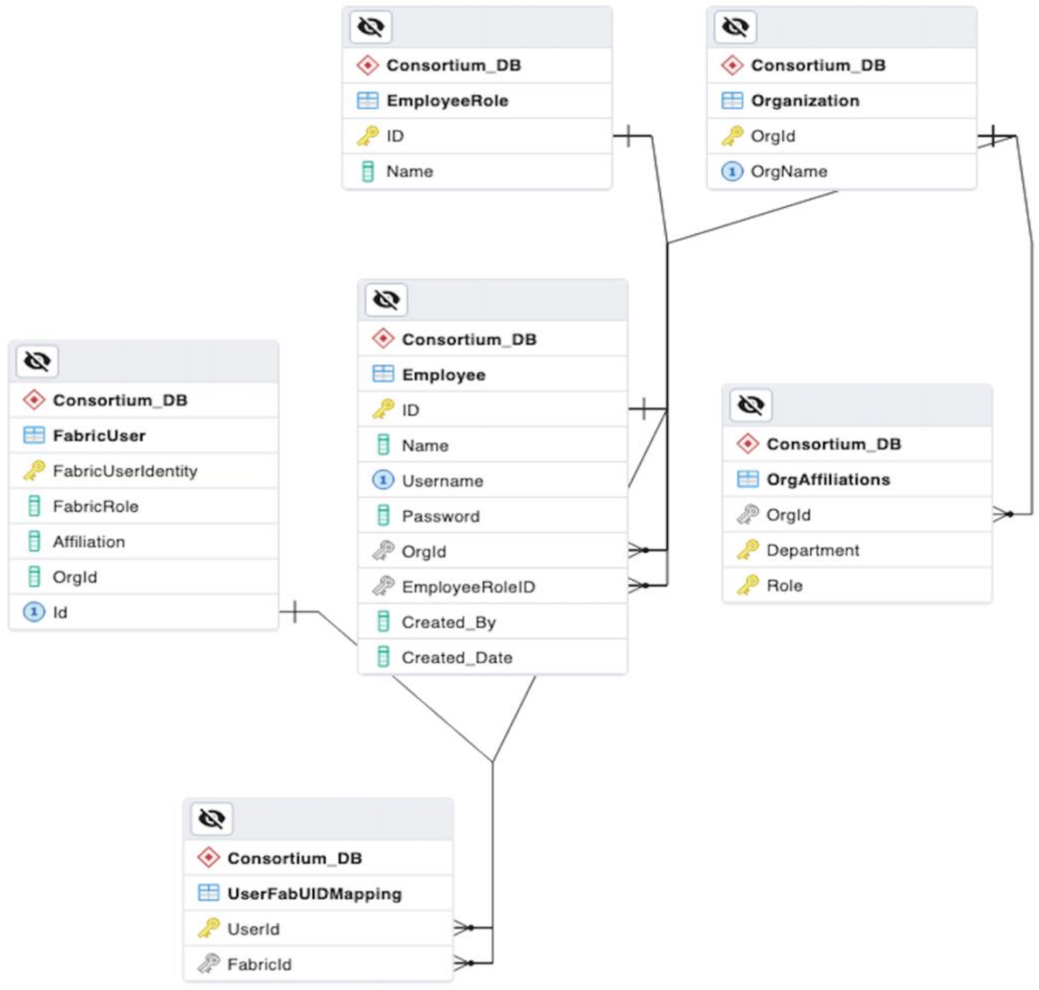


Figure 3: Relational schema

5. Use Case Details

5.1. Login User

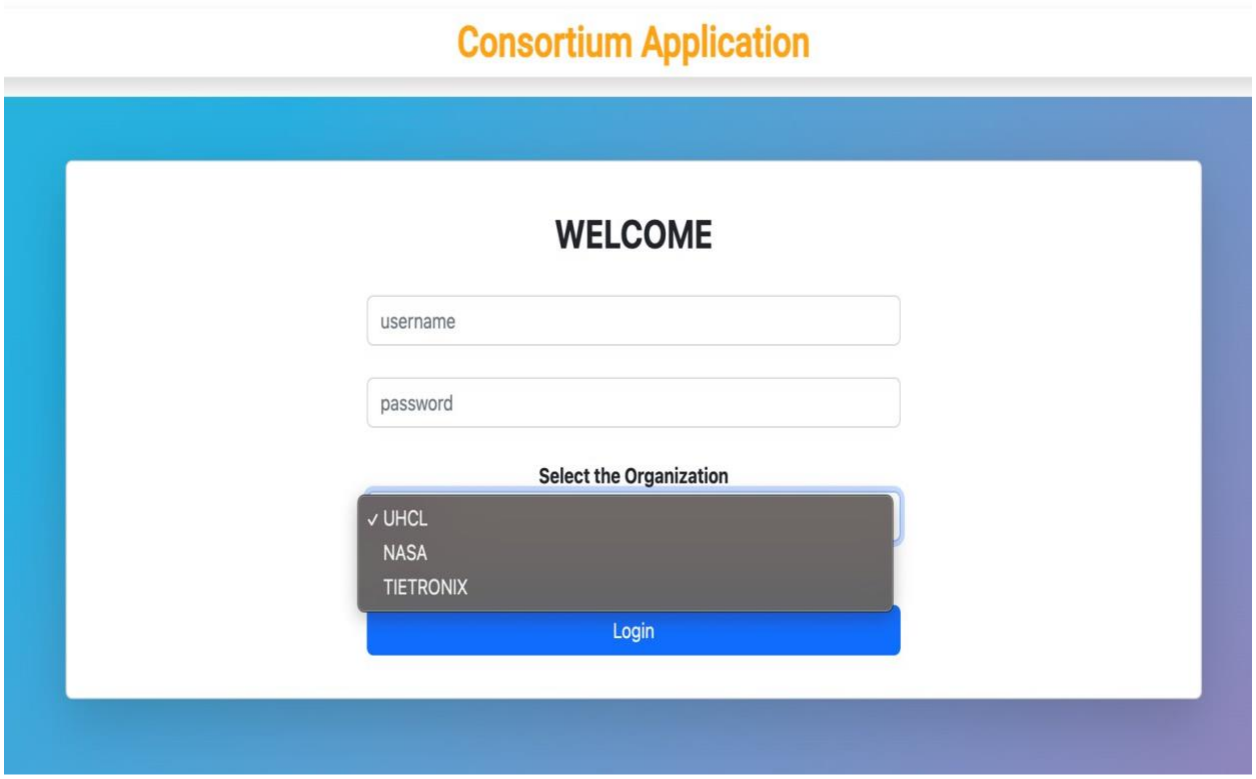


Figure 4: Application Login Page

- To Login to the application, User needs to provide his username, password, and the organization details
- Credentials will be sent to DB for validation
- On success, user will be redirected to the respective page based on their role (either admin or local user)
- On failure, Credentials Invalid error will be displayed, and user will stay on the same login page

5.2. Admin User

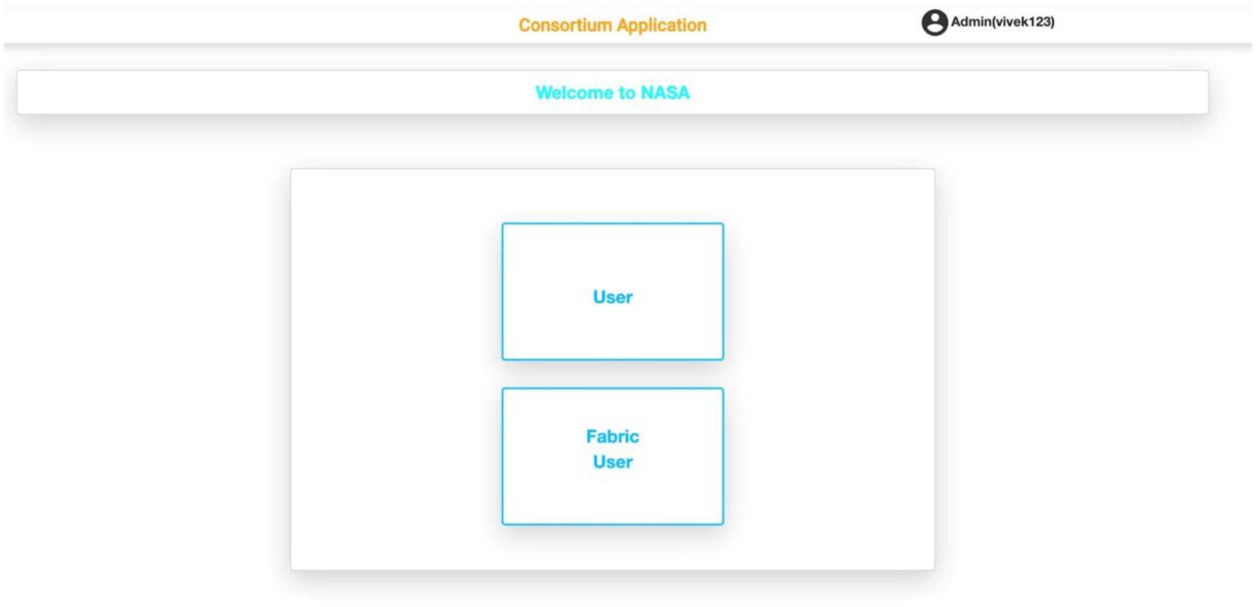
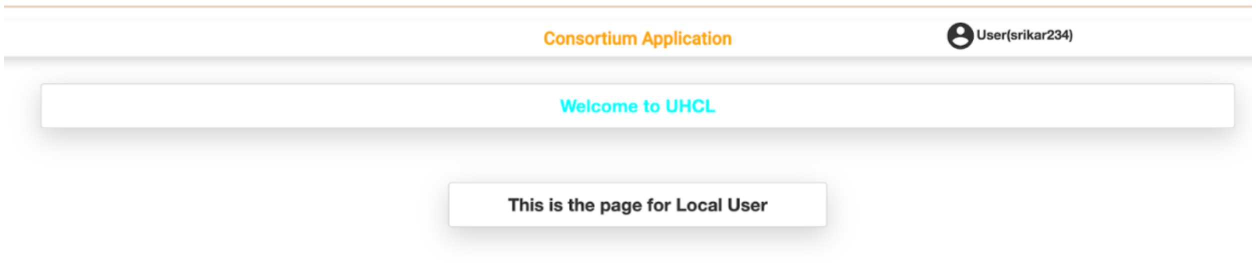


Figure 5: Admin Dashboard

- ➢ Admin user can perform operations related to Local user & Fabric User Identity
- ➢ Admin can only be able to work on operations to his/her respective organization

5.3. Local User

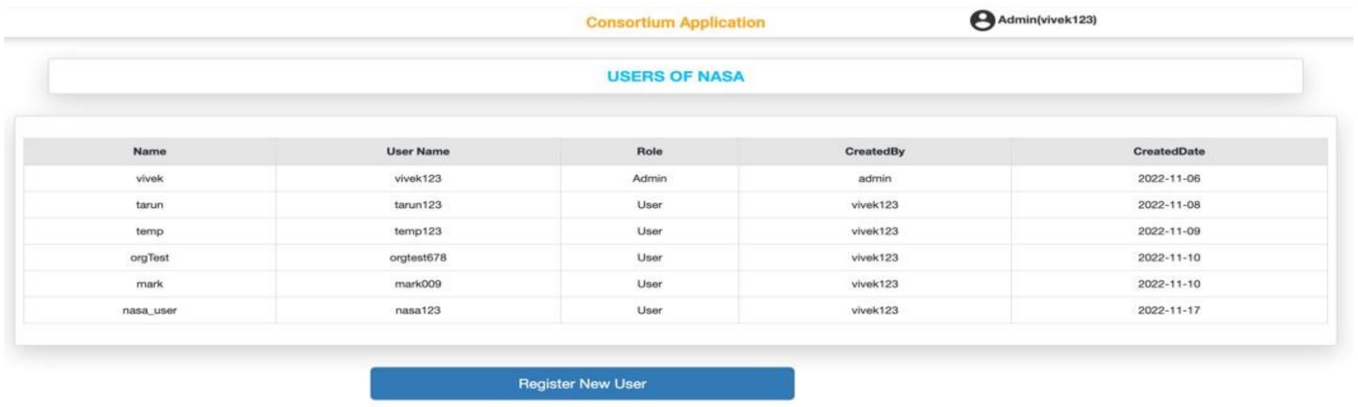


This is the page for Local User

Figure 6: Local User Dashboard

- ➢ Every Organization has separate roles/works for the local user
- ➢ Operations related to the local user can be added here

5.4. User List



Consortium Application

Admin(vivek123)

USERS OF NASA

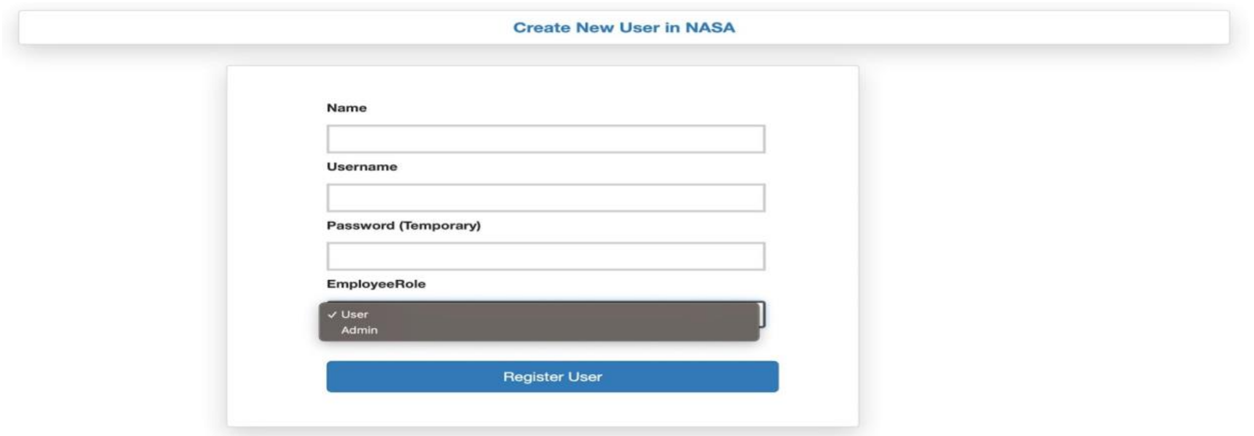
Name	User Name	Role	CreatedBy	CreatedDate
vivek	vivek123	Admin	admin	2022-11-06
tarun	tarun123	User	vivek123	2022-11-08
temp	temp123	User	vivek123	2022-11-09
orgTest	orgtest678	User	vivek123	2022-11-10
mark	mark009	User	vivek123	2022-11-10
nasa_user	nasa123	User	vivek123	2022-11-17

Register New User

Figure 7: List of Local Users

- ➢ List of all the users including both admins and local users will be displayed here
- ➢ Admin can create a new user or new admin using the "Register New user" option
- ➢ Operations like Update and Delete can be incorporated into this page

5.5. Create User



Create New User in NASA

Name

Username

Password (Temporary)

EmployeeRole

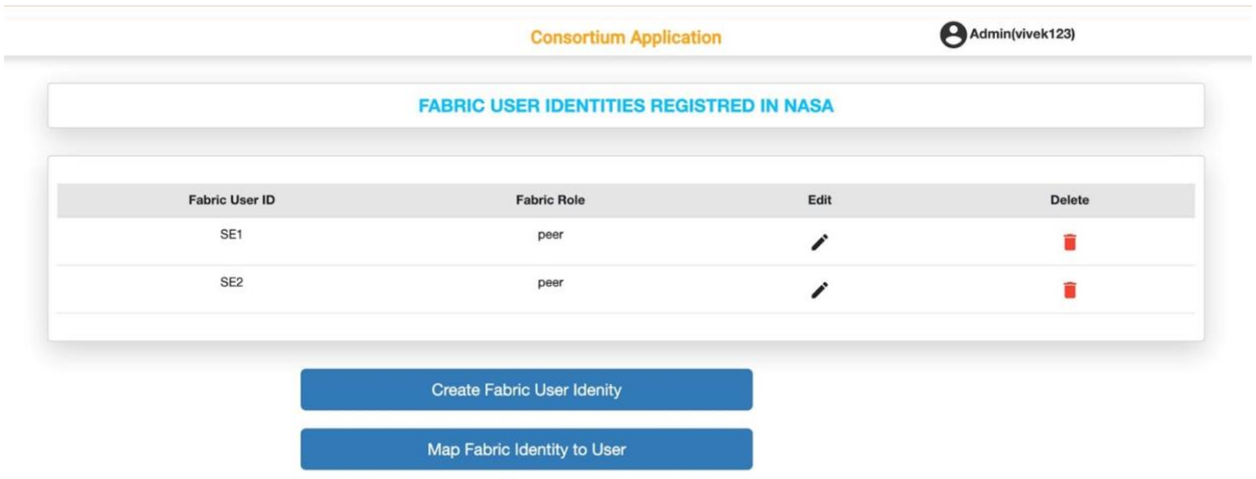
✓ User
Admin

Register User

Figure 8: Creating Local User/Admin

- Admin can create a new user providing the details like Name, Username, Password (temporary) and his role (User/Admin)
- The details will be sent to the DB
- If the username is already taken the user cannot be created
- On success, the details will be added in the DB and the page is redirected to User list
- On failure, failed to create user will be thrown and will remain on the same page

5.6. Fabric User Identity List



FABRIC USER IDENTITIES REGISTRED IN NASA

Fabric User ID	Fabric Role	Edit	Delete
SE1	peer	✎	🗑
SE2	peer	✎	🗑

Figure 9: List of Fabric User Identities

- List of all the Fabric User Identities associated with the respective organization will be displayed here
- When organization admin tries to access this page, an API will be called to get all the fabric identities
- On success, identity list will be displayed
- On failure, no list will be displayed
- Admin can access the functions like Create, Update, Delete & Mapping from here

5.7. Create Fabric User Identity

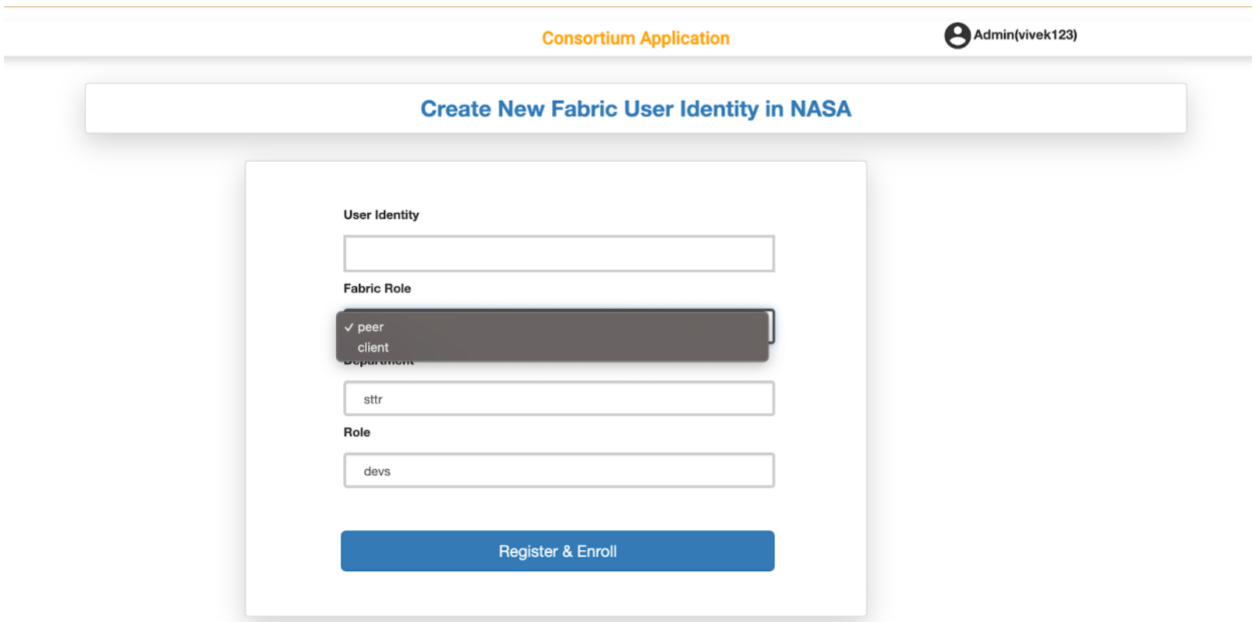
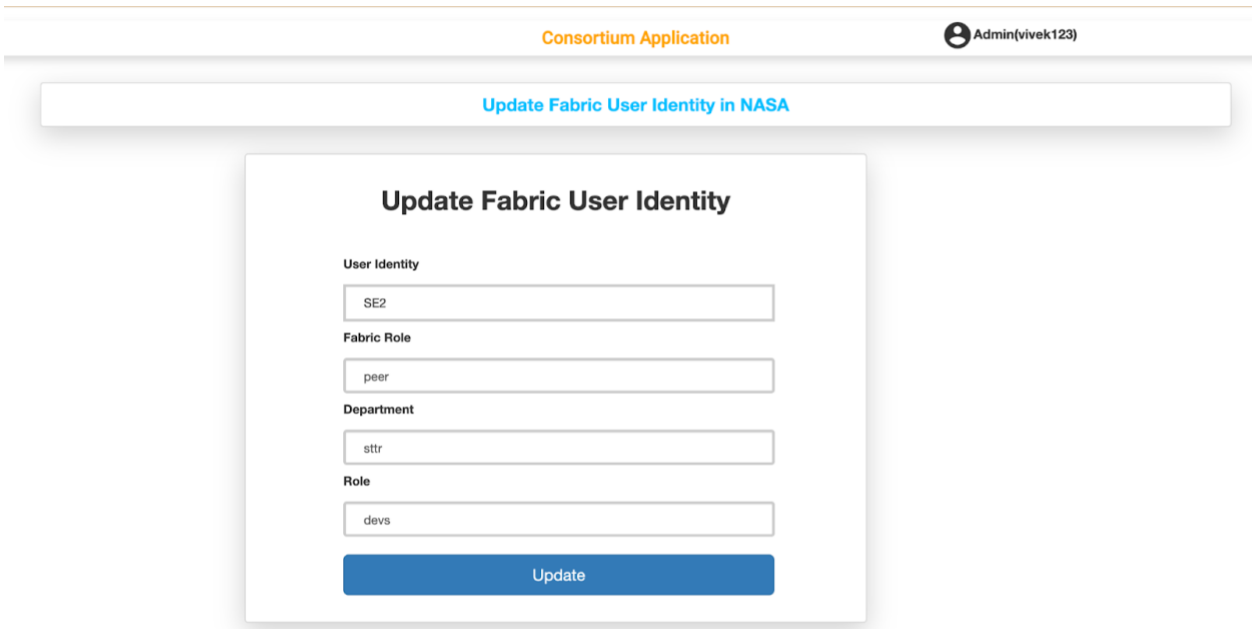


Figure 10: Creating a New Fabric User Identity

- Organization admin can create a new Fabric User Identity for the organization by providing the details like User Identity, Fabric Role (Peer/Client), Department (Will be getting from the DB), Once the department is selected Role will be filtered and admin will select one department
- The details will be sent it DB and to Blockchain
- If User Identity already exists with same affiliation, then Identity will not be created
- On success, Identity will be added to both DB and Wallet system and page will be redirected to fabric user identity list.
- On failure, Identity will not be created and will not be added in both wallet and DB

5.8. Update Fabric User Identity

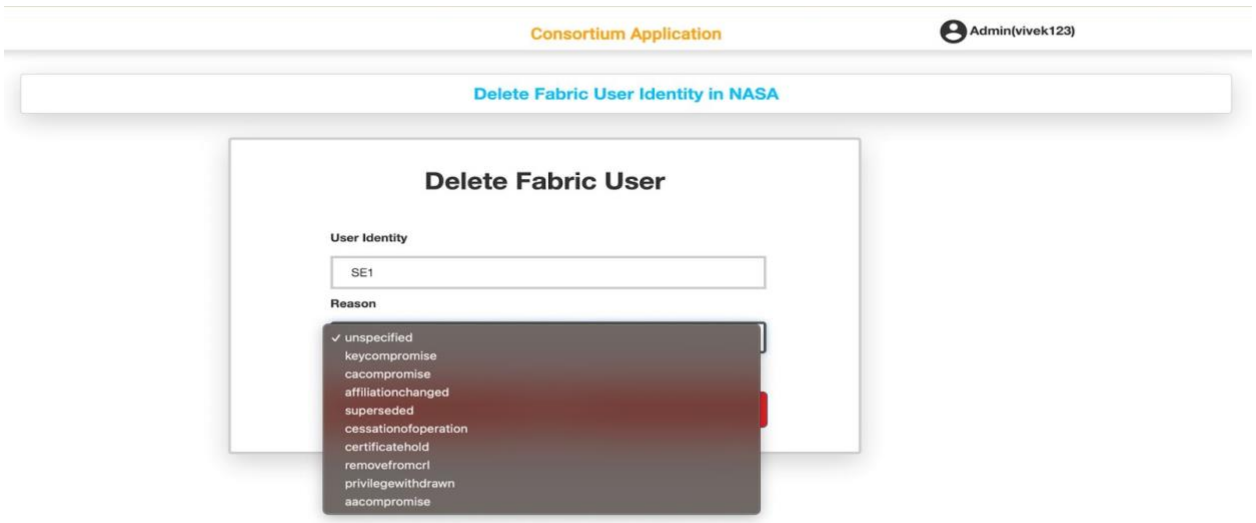

Consortium Application
Admin(vivek123)
Update Fabric User Identity in NASA
Update Fabric User Identity
Update

User Identity: SE2
Fabric Role: peer
Department: sttr
Role: devs

Figure 11: Updating Fabric User Identity

- When admin want to update, the data is collected and verified in the database that the user Identity is already exists with the updated affiliation
- On success, the details will be updated in both wallet system and DB and redirected to the Fabric User Identity List
- On failure, the details will not be updated and will stay on same page

5.9. Delete Fabric User Identity

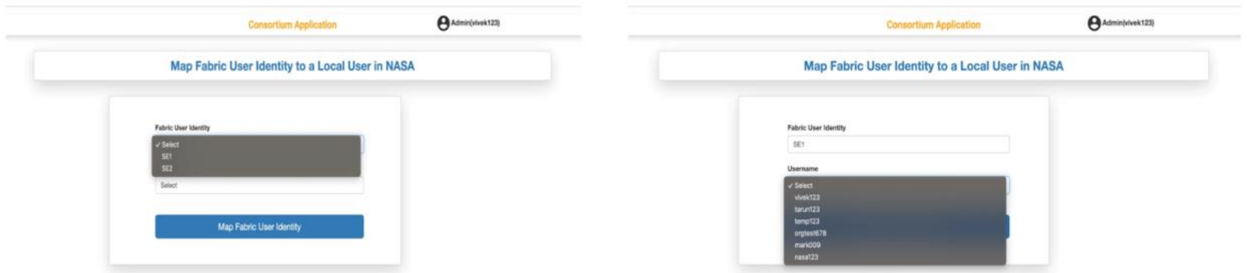


User Identity: SE1

Figure 12: Deleting Fabric User Identity

- ➢ When admin wants to delete an Identity, a reason for deletion needs to be specified
- ➢ On success, the identity will be deleted from the DB and wallet and redirected to the fabric user identity list
- ➢ On failure, error will be thrown and will stay on the same page

5.10. Map Fabric User Identity List to Local user



Fabric User Identity: SE1

Figure 13: Mapping Fabric User Identity to Local User

- When admin wants to map a fabric user identity to a local user, user identity and local username will be selected
- The details will be sent to the DB
- If mapping already exists to either fabric user identity or local user, the current mapping will not be performed
- On success, the UUID's of both fabric user identity and local user will be added to the mapping table and redirected to the fabric user identity list
- On failure, error will be thrown and will stay on the same page

6. Requirements Covered in this Project

- Project creation for local users, fabric users, crud operations, linking local users to fabric user
- 2 types of users (local users/admin) has separate page and functionality
- Organizational level user's implementation
- UI and database administration of employee roles in the Manager position.
- The Manager role's UI and database project mapping to the Organization
- Organization List view and Current Project List view in the database and user interface
- The UI and database views of the employees list

7. Limitations

- Only sample Block chain applications were tested using Fabcar but not on the full application testing part
- Consortium role is not defined in the back end i.e., smart contract network
- Proper access control is not implemented this time; two factor authentications are also not implemented this time
- Synchronization of data between Blockchain and local databases is not implemented in all stages
- Reading of logs is only obtained using command prompt as the child process is only invoked instead of using functions

8. Future Work

- Change Password for a user can be implemented
- Update and Delete operations for Local user
- Database interaction and perform validation with the blockchain can be checked
- If application can be implemented in a generic way can be implemented for other consortium applications too
- Adding multiple organizations to the blockchain network
- Add different routing for different organization in front end

9. Challenges in Project

- Identifying what data needs to be stored in a local database and blockchain
- Lack of computing resources to run and test the application. Only one system was available for the entire project to test the end-to-end application and every member had to take subsequent turns to test and demonstrate their work on only one system.
- Use of file systems in the blockchain makes data accessibility difficult
- Validating and keeping track of data changes in local databases and blockchains
- Lake of knowledge on previous applications
- Synchronization of Postgres DB and Wallet (File system)
- Refactoring of previous application code
- Calculating the effort needed to construct the UI and backend

10. Conclusion

By the end of this project, we were able to successfully create the full stack consortium application that interacts with the block chain, required for the users (both local and fabric users) with all the APIs for CRUD operations. The Endpoints created at blockchain were created to handle command line and API requests, and necessary data is stored in both Postgres DB and blockchain(wallet).

11. Individual Contributions

11.1. Srikar Contributions

- Worked as a Software Architect, Team Lead for the project managing the application from end to end
- Responsible for dividing tasks in a week and distributing across the team and observing the progress
- Built middleware application API design Flow which helped in interacting with the backend blockchain endpoint
- Created connections between frontend, middleware, backend applications thereby creating a smooth flow from end-to-end application
- Designed and architected the Database for the storage of user management information at Postgres and File system in middleware and backend, respectively
- Written Register User, User List logics and endpoints for saving and registering users into the file system wallet in blockchain
- Provided Knowledge Transfer (KT) for the team on Angular, Node and Postgres technologies
- Tested the application flow using Postman from both backend and middleware to check whether all endpoints are working as expected
- Refactored the frontend application to adjust the needs of user management which helped in generating a faster and responsive UI

11.2. Vivek Contributions

- Worked as backend developer for the consortium application
- Connected with stakeholders, comprehended the entire project, and gathered specifications
- Installed the blockchain network and rectified issues during the installation time
- Took KT sessions for team members on Blockchain and to understand the project

- Worked on converting UML Diagram into tables or documents and worked on to set up Postgres DB and Integrating with the backend API
- Analyzed the purpose of the data held in local databases and blockchains
- Created sample JSON data and designed sample JSON files from corresponding schema tables
- Presented the entire application and its workings on and demonstrated the screens created API's working model in the Client meeting
- Helped FE Team in Integrating Frontend Application with Supporting Backend Api's
- Assisted in testing the application flow using postman from both backend and middleware to check whether all endpoints are working as expected.
- Written create and display fabric user identity scripts in Fabcar
- Worked on parseArgs.js, childProjess.js, config.json scripts that helps to use the application using the command terminal along with the front end
- Designed and Developed capstone project website and hosted in university DCM servers
- Helped team in fixing issues and integrating everything with blockchain
- Helped team in preparing Final Documentation, Presentation

11.3. Aravind Contributions

- Worked as a backend developer for the consortium application
- As per software and hardware requirements created initial setup of the Blockchain Network (Fabcar) and the Web application and integrated everything
- Implemented Backend Api's for performing CRUD operation with Blockchain network
- Researching and understanding the, Blockchain Hyperledger fabric in the project by going through the documents and diagrams to help the team understanding it better
- Actively attended all the team meetings and mentor meetings to understand the requirements and delivered weekly objectives on time
- Taken ownership developing MBSE CRUD APIs to integrate it with Fabcar and testing, debugging the fabric based on the integration issues
- Integrated frontend route handlers of the web client with the node end points

- Written update and delete fabric user identity scripts in Fabcar
- Designed and implemented the postgres database tables schema as per the project requirements
- Assisted in testing the application flow using postman from both backend and middleware to check whether all endpoints are working as expected
- Designed and Developed capstone project website and hosted in university DCM servers
- Helped team in preparing Final Documentation, Presentation

11.4. Tarun Contributions

- Worked as a Front-end developer and Tester for the consortium application
- Gone through the Project Description document and requirement gathering analysis and created the website and project setup
- Worked on creating the components for edit and delete operations for the users
- Worked on API's modification created for the components like create user, password validation using Angular 13 and Node.js
- Implemented Changes in the UI as per the instructions mentioned especially in edit and delete user's pages
- Created the UI Screens for Create User, Admin login screen, Fabric User Identity List, Delete and Edit the Fabric User Identity
- Analyze and get involved in Relational Database Design and Architecture and make changes as per the model and requirement
- Documented weekly reports every week and took care of all the minutes for the meeting and updating on the DCM Server
- Developed the Front-End components and made reactive using Material Icons
- Responsible for Testing the entire application and informing the team members in case of issues and Worked on the Improvements in the existing code and functionality and User Interface
- Created Power point presentation slides and helped the team in creating the document

11.5. Ravi Teja Contributions

- Understanding the initial project requirements and working with the team members to gather the software and hardware requirements
- Created initial setup of the Blockchain Network (Fabcar) and the Web application and integrated everything
- Created the UI Screens for Update, Map and Create Fabric User Identity List
- Once done with the project requirements, went through the UML Diagram (Consortium, Organization, Employee, Project, and Role)
- Worked as one of the web developers for this project by implementing the features and functionality of a website using Angular as front end and Node js as backend
- Coordinated with each member in the team to build the consortium application by helping them with requirements gathering, design and implementation
- Participated in the Testing of the entire application and identified the issues and informed to the team members
- Worked on relational schema document, updated Postgres database based on latest changes
- Helped team in fixing issues and integrating everything with blockchain
- Helped the team in preparing the Final presentation and Document
- Developing and modifying the application's needs as they evolve

12. References

1. “Hyperledger Fabric SDK for Node.js Module: Fabric-Network.” Accessed December 6, 2022. https://hyperledger.github.io/fabric-sdk-node/release-2.2/module-fabric-network.html.
2. “Welcome to Hyperledger Fabric CA (Certificate Authority) — Hyperledger-Fabric-Cadocs Main Documentation.” Accessed December 6, 2022. https://hyperledger-fabric-ca.readthedocs.io/en/latest/.
3. https://github.com/IBM-Blockchain/microfab
4. https://angular.io/
5. https://www.npmjs.com/package/
6. https://material.angular.io/
7. https://app.dbdesigner.net/designer
8. https://hackr.io/tutorials/learn-angular
9. https://www.tektutorialshub.com/angular/angular-list-of-learning-resources/
10. https://nodejs.org/dist/latest-v18.x/docs/api/
11. https://learning.postman.com/docs/getting-started/introduction/
12. https://github.com/hyperledger/fabric-samples