**ICIN Bank Online Internet Banking Web-app**

**Description**

Create a dynamic and responsive Java online banking web application to deposit, withdraw, and transfer the money between the accounts.

**Background of the problem statement:**

ICIN is one of the top banking firms that accepts deposits from the public for the purpose of lending loans to the public. It also invests an amount in securities.  
Recently, the business analysts noticed a drop in the number of customers of the bank. They found out that online banking systems of banks like AXIS and American Express are gaining more profits by eliminating middlemen from the equation. As a result, the team decided to hire a Full Stack developer who can develop an online banking web application with a rich and user-friendly interface.  
You are hired as one of the Full Stack Java developers and have been asked to develop the web application. The management team has provided you the requirements and their business model so that you can easily arrange different components of the application.

**Features of the application:**

1. Registration
2. Login
3. Account transactions
4. Transfers
5. Savings details
6. Profile settings
7. Requesting cheque books

**Recommended technologies:**

1. Database management: MySQL
2. Back-end logic: Java programming, Spring Boot framework
3. Front-end development: Angular 2, HTML/CSS,
4. Automation and testing technologies: Selenium and JUnit
5. DevOps and production technologies: Git, GitHub, Jenkins, Docker, and AWS.

**Project development guidelines:**

* The project will be delivered within four sprints with every sprint delivering a minimal viable product.
* It is mandatory to perform proper sprint planning with user stories to develop all the components of the project.
* The learner can use any technology from the above-mentioned technologies for different layers of the project.
* The web application should be responsive and should fetch or send data dynamically without hardcoded values.
* The learner must maintain the version of the application over GitHub and every new change should be sent to the repository.
* The learner must implement a CI/CD pipeline using Jenkins.
* The learner should also deploy and host the application on an AWS EC2 instance.
* The learner should also implement automation testing before the application enters the CI/CD pipeline.
* The learner should use Git branching to separately perform the basic automation testing of application.
* The learner should make a rich front-end of the application, which is user- friendly and easy for the user to navigate through the application.
* There will be two portals in the application, namely the admin and user portal.

**Admin Portal:**

It deals with all the back-end data generation and product information. The admin user should be able to:

* Authorize the roles and guidelines for the user
* Grant access to the user regarding money transfer, deposits, and withdrawal
* Block the user account in case of any threat
* Authorize the cheque book requests

**User Portal:**

It deals with the user activities. The user should be able to:

* Register or log in to the application to maintain a record of activities
* Deposit and withdraw money from the account
* View transactions and balance in the primary and savings account
* Transfer funds between different accounts and add recipients
* Request cheque books for different accounts

**Project Users Stories**

As a full stack developer, I want to develop banking online web application which has below mentioned functionalities,

* As a user, I want register as a customer of the bank and be able to login into internet banking application.
* As a user, I want to open savings, current accounts.
* As a user, I want to do accounting transactions such as deposit amount to account and withdraw from the account.
* As a user, I want to view the account balances.
* As a user, I want to transfer funds from one account to another account.
* As a user, I want to request for cheque books.
* As a user, I want to view the profile and logout successfully from the web-app.
* As an admin, I want to register and login successfully as an admin user.
* As an admin, I want to approve / reject customer, account, transactions, funds transfers and cheques requested by the customer.
* As an admin, I want to block user in case of any threats and logout successfully.
* As a developer, I want to build front-end banking web-application for customers to login and do banking operations.
* As a developer, I want to provide option for logout from customer online app.
* As a developer, I want to build front-end admin web-application for bank admin to login and perform admin operations.
* As a developer, I want to provide option for logout from the admin web-app.

**Sprint 1 (Week 1)**

* As a developer, I want to develop a front-end registration and login form for customer with angular.
* As a developer, I want to design and create all the database table required for the application with JPA.
* As a developer, I want to develop back-end spring boot logic which gets the data entered in the front-end form and updates the database.
* As a developer, I want to provide field validations for registration and login forms.
* As a developer, I want to create home page which gets displayed for the customer upon successful login.
* As a developer, I want to develop forms for opening account, perform account deposits and withdrawal, funds transfer and cheque requests.

**Sprint 2 (Week 2)**

* As a developer, I want to provide fields validations and auto populate customer Id, customer name based on logged in customer.
* As a developer, I want to auto-populate account related fields like currency, branch and account type based on the selected account no.
* As a developer, I want to develop account balance enquiry with account balances.
* As a developer, I want to develop an option to list out profile information for the logged in customer.
* As a developer, I want to develop logout functionality which logs of the current customer and takes back to login page.

**Sprint 3 (Week 3)**

* As a developer, I want to develop a front-end registration and login form for admin with angular.
* As a developer, I want to design and create all the database table required for the application with JPA.
* As a developer, I want to develop back-end spring boot logic which gets the data entered in the front-end form and updates the database.
* As a developer, I want to provide field validations for registration and login forms.
* As a developer, I want to create home page which gets displayed for the admin upon successful login.
* As a developer, I want to develop options for admin to approve / reject customer, account, transactions, funds transfers and cheques requested by the customer.
* As an admin, I want to develop an option for block user in case of any threats.
* As a developer, I want to develop logout functionality which logs of the admin screen and takes back to login page.

**Sprint 4 (Week 4)**

* As a developer, I want to develop scripts with testNG and selenium to automate whole app testing process in proper step by step sequence.
* The sequence for executing scripts is specified in the testNG.xml file.
* Push the project into git repository.
* Jenkins is used for CI/CD pipeline generation. Configure git repository and testNG testing for the project.
* Create EC2 instance and start docker the git repository of the project.
* Host the project into the docker hub.
* Now the web-app is available for public access.

**Sprint 5 (Week 5)**

* Testing the front-end application with different kinds of User input.
* Initializing git repository to track changes as development progresses.
* Pushing code to GitHub.
* Creating this specification document with application details, appearance, and user interactions.

## **Core concepts used in project**

* Database management: MySQL
* Back-end logic: Java programming, Spring Boot framework
* Front-end development: Angular 2, HTML/CSS,
* Automation and testing technologies: Selenium and JUnit
* DevOps and production technologies: Git, GitHub, Jenkins, Docker, and AWS

**Architecture diagram / flow chart:**

**User Operations,**

**Start**

**User registers account as customer and logs into the web application.**

**If user account is approved, user logs into his account successfully and performs operations.**

**Online app has options for customer to open account , perform account deposit & withdrawal , view balances , do funds transfer and request cheque books. View and edit customer profile.**

**User should be able to logoff successfully from the application.**

**End**

**Admin Operations,**

**Start**

**Admin registers account as admin and logs into the web application.**

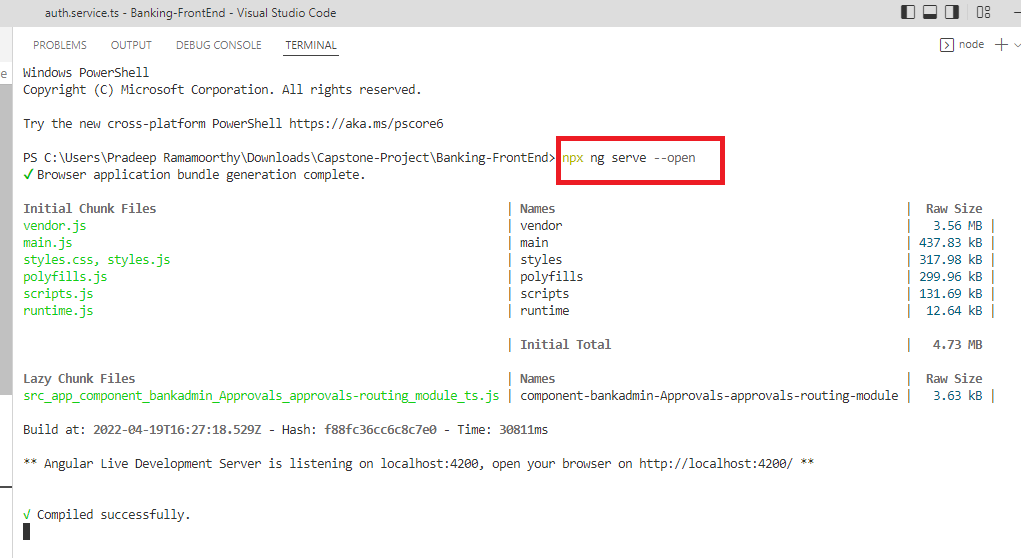
**Admin logs into his account successfully and can perform approval operations for customer transactions and block a user.**

**User should be able to logoff successfully from the application.**

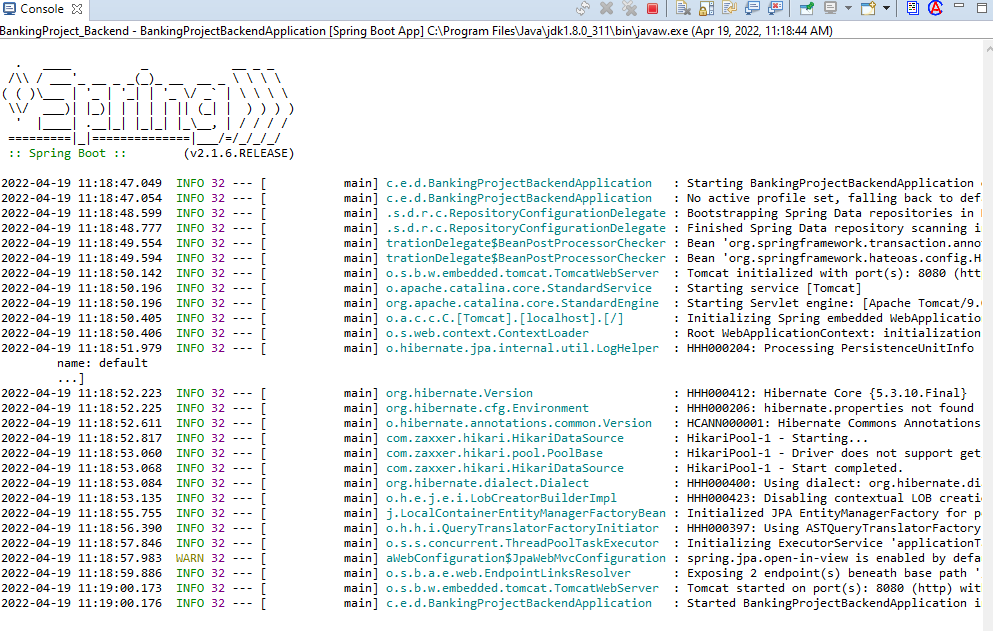
**End**

**Screenshots,**

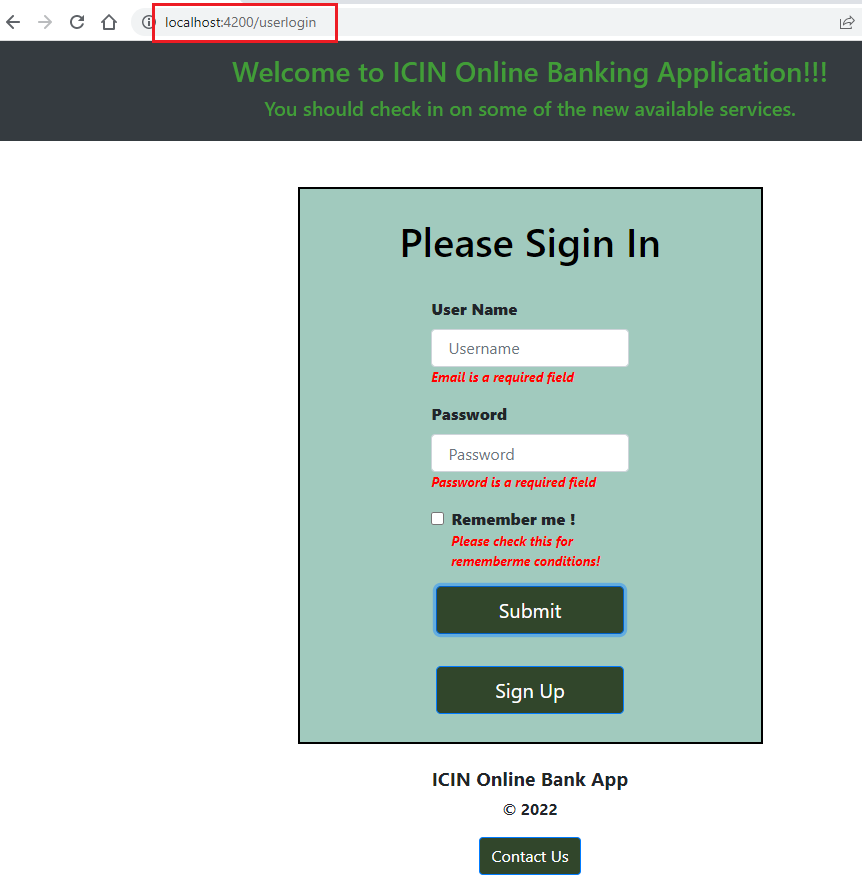
* Start the front-end angular app,



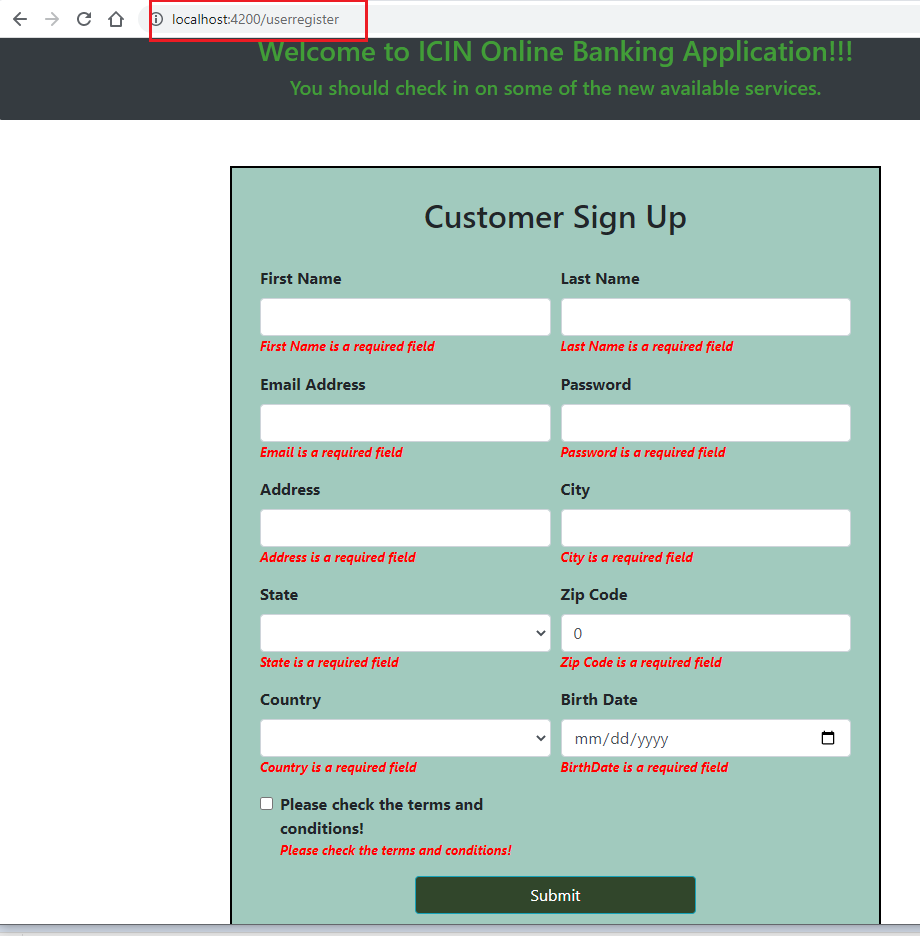
* Start the back-end sprint boot app,

****

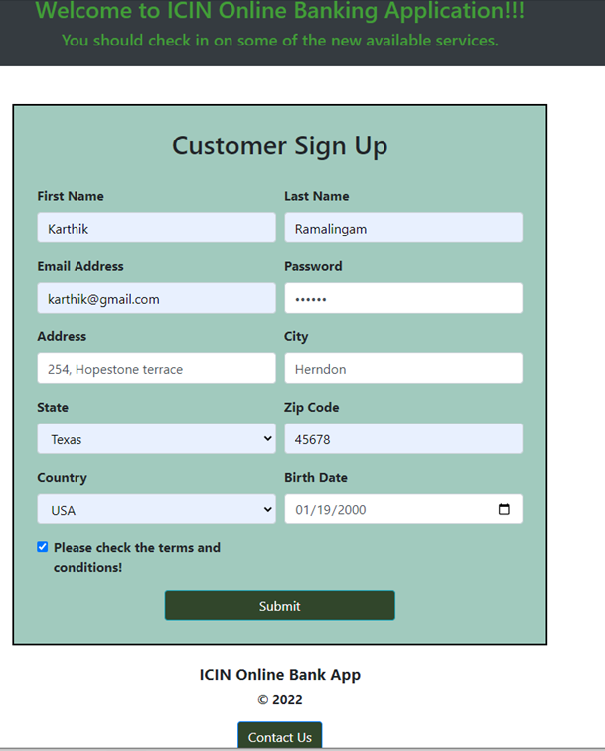
* Type the below highlighted URL in the browser to access the customer login form. Below screen shows basic validations.

****

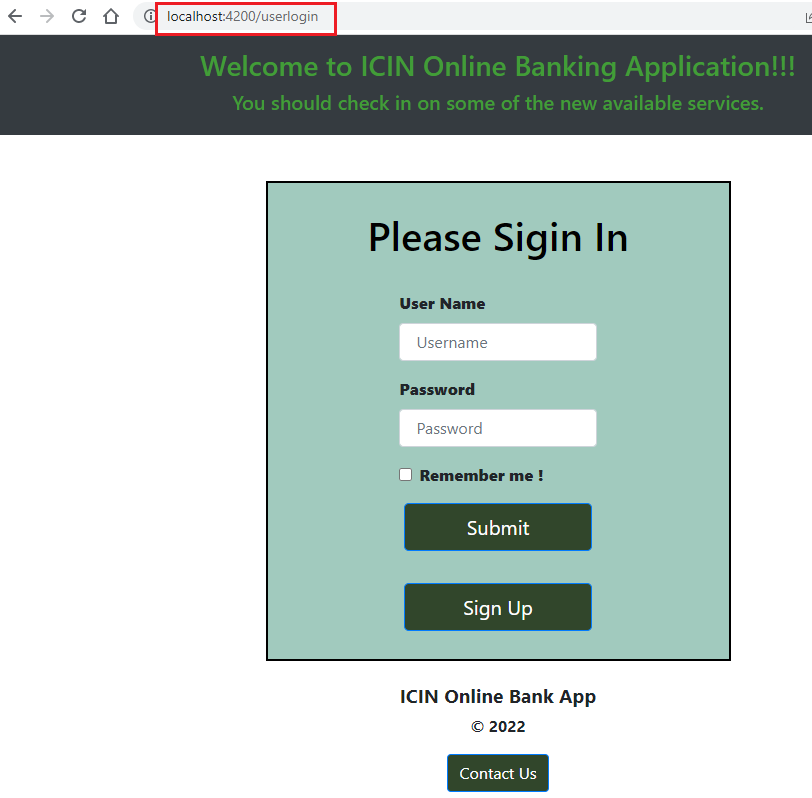
* Upon clicking sign-up button on above screen , takes to registration form. Below screen shows registration form validations.

****

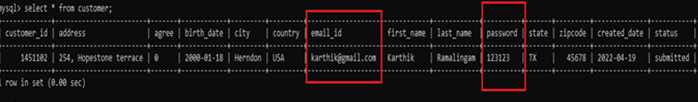
* Enter customer details in the form and click submit button.

****

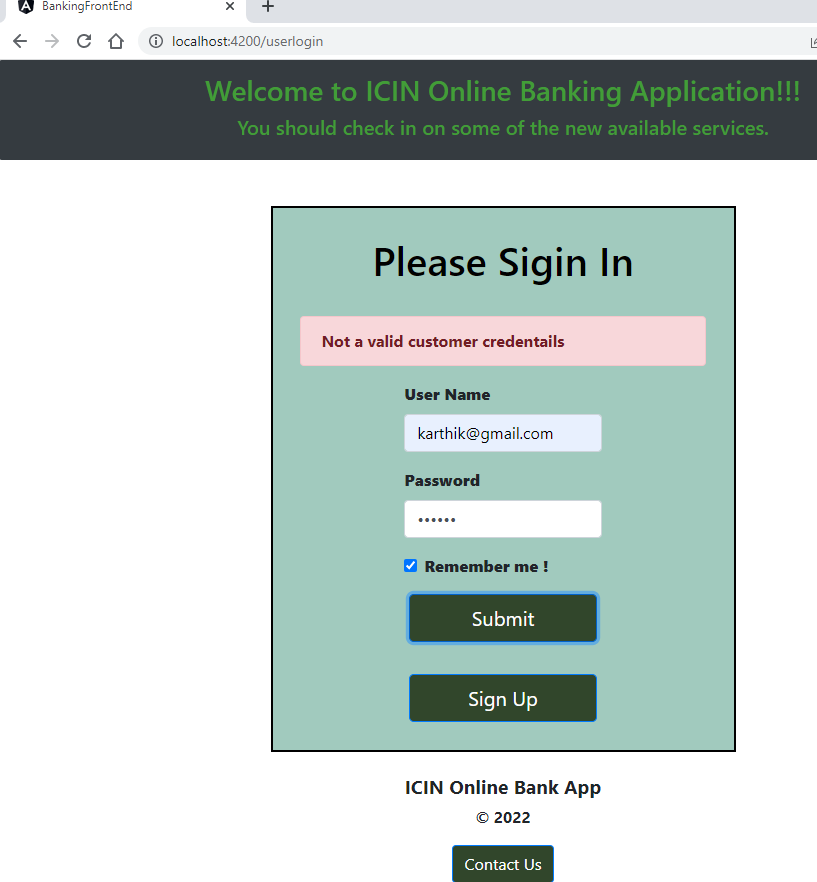
* Upon successful registration, takes back to login page.

****

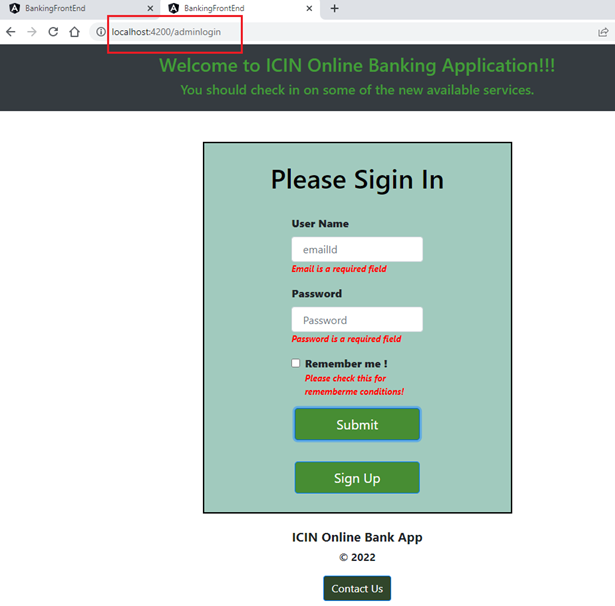
* Upon successful registration, customer table gets updated with details.

****

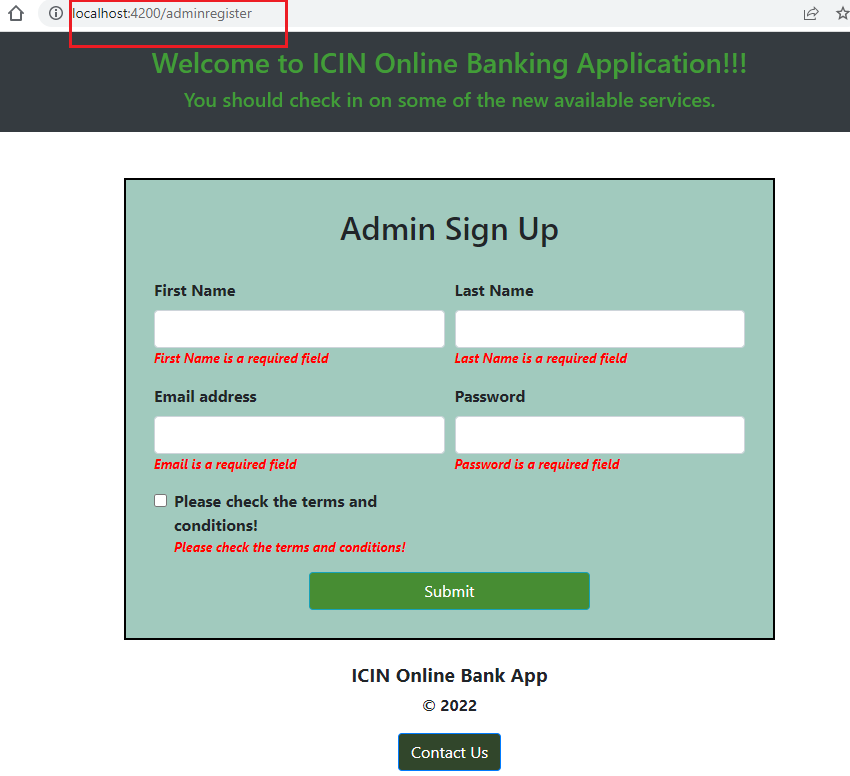
* Attempt to login with invalid customer credentials shows below validation error.

****

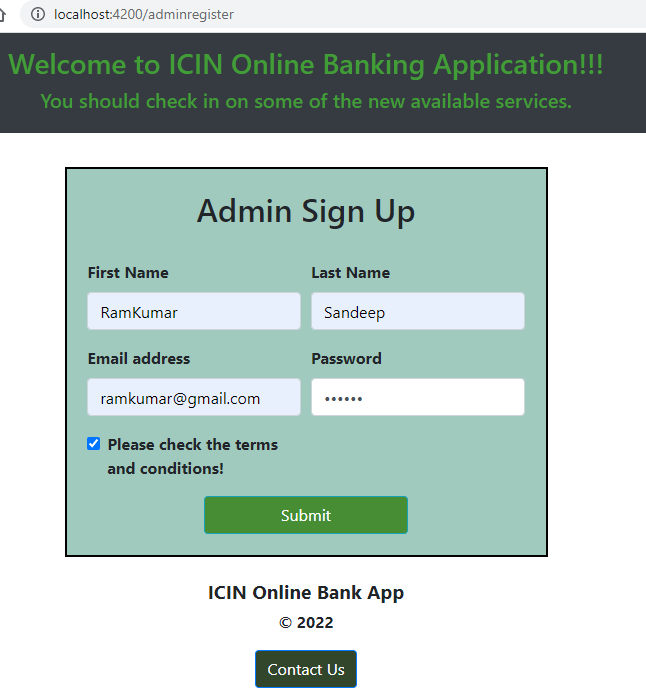
* Type the below highlighted URL in the browser to access the admin login form. Below screen shows basic admin login form validations.

****

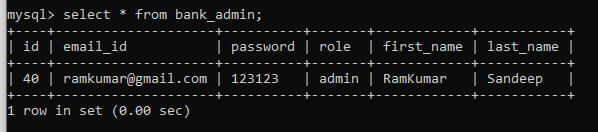
* Upon clicking Sign-up button takes into admin registration page. Below screen shows basic admin registration form validations.

****

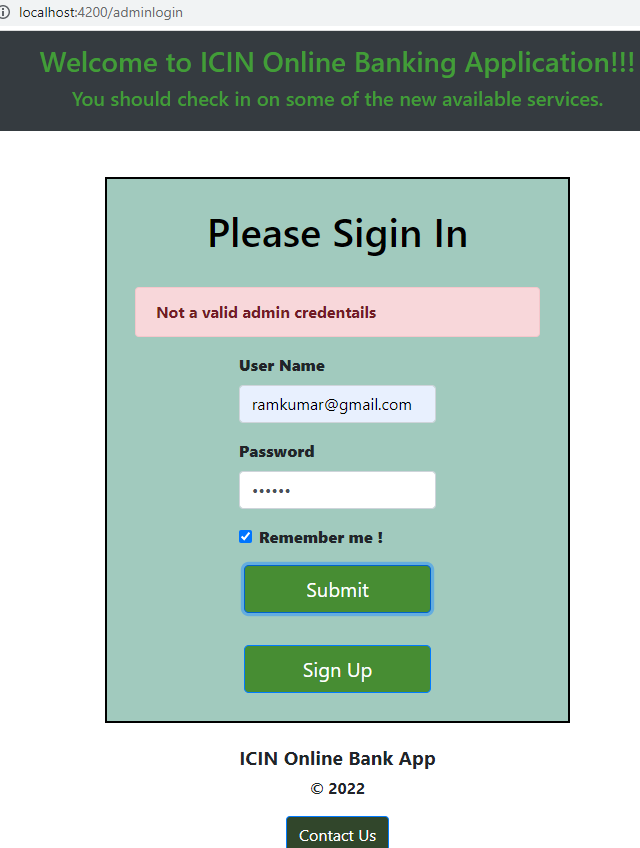
* Enter admin information in the form and click on the submit button.

****

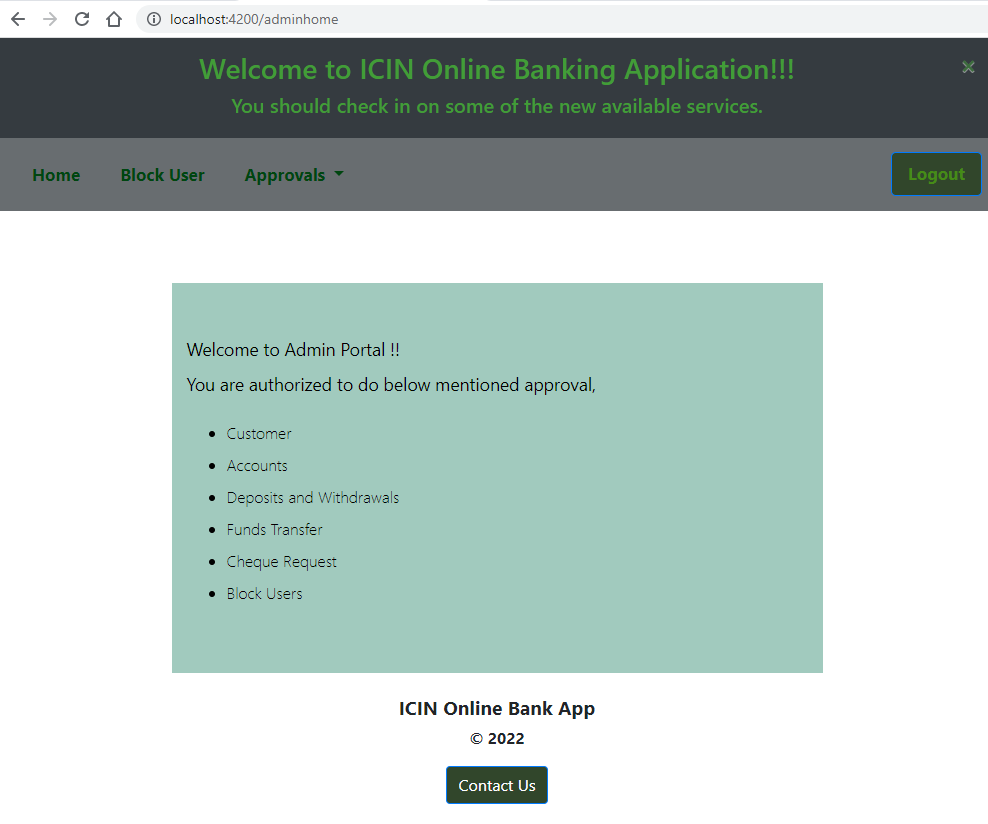
* Upon successful submit, database gets updated with form information.

****

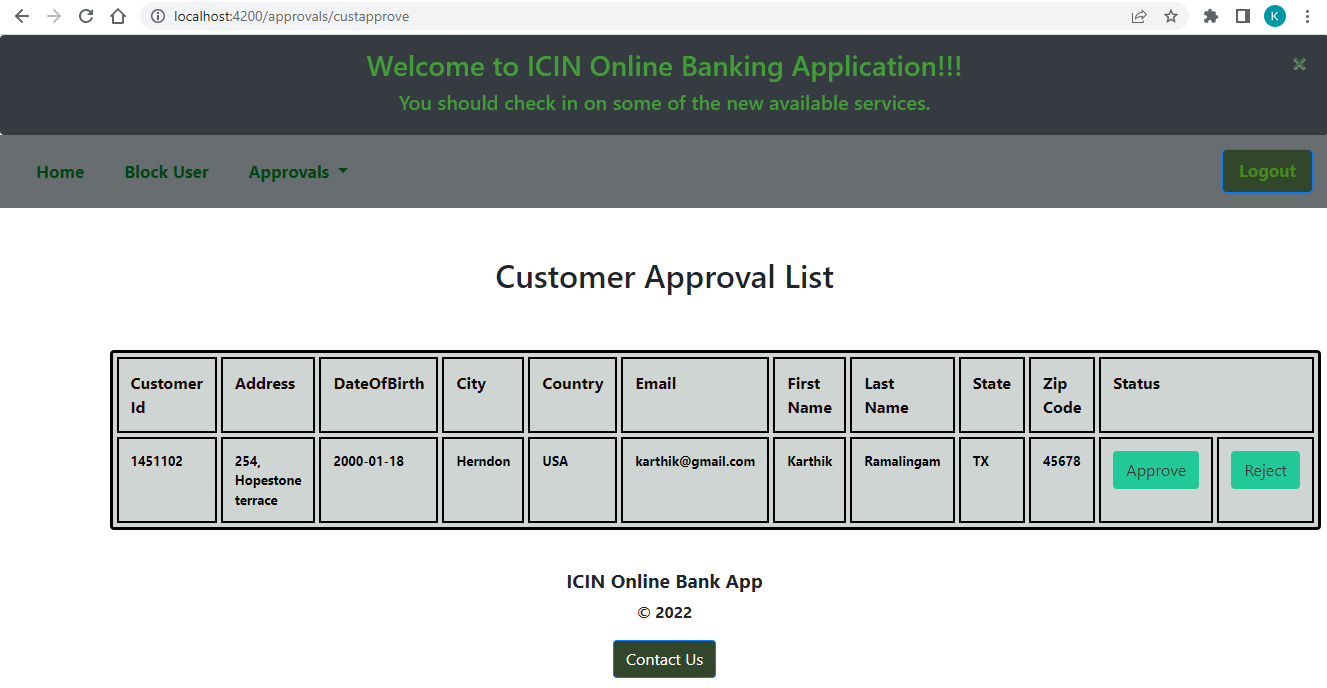
* Attempt to login with invalid admin credentials shows below error message.

****

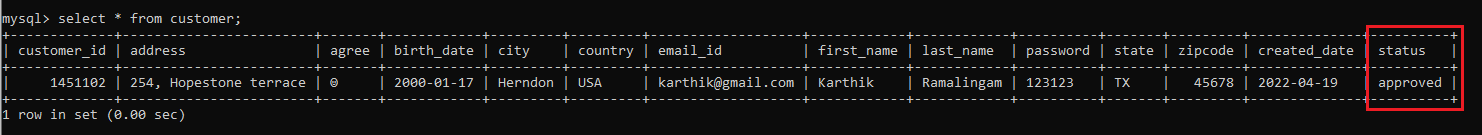
* Authorized admin users can log into the below shown home page.

****

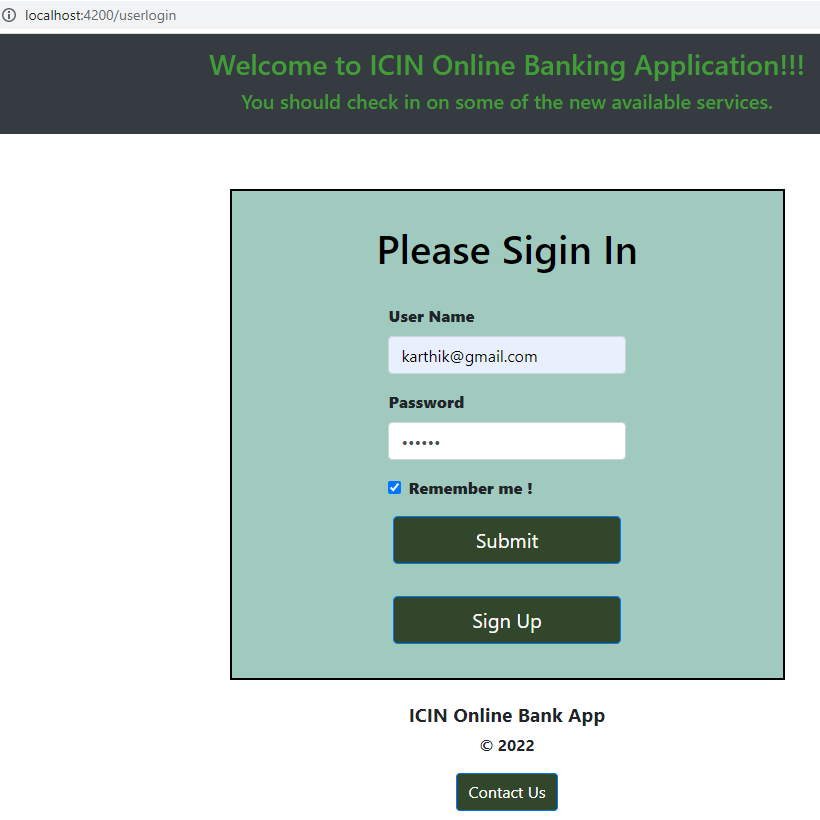
* Admin user’s customer approval/ reject screen,

****

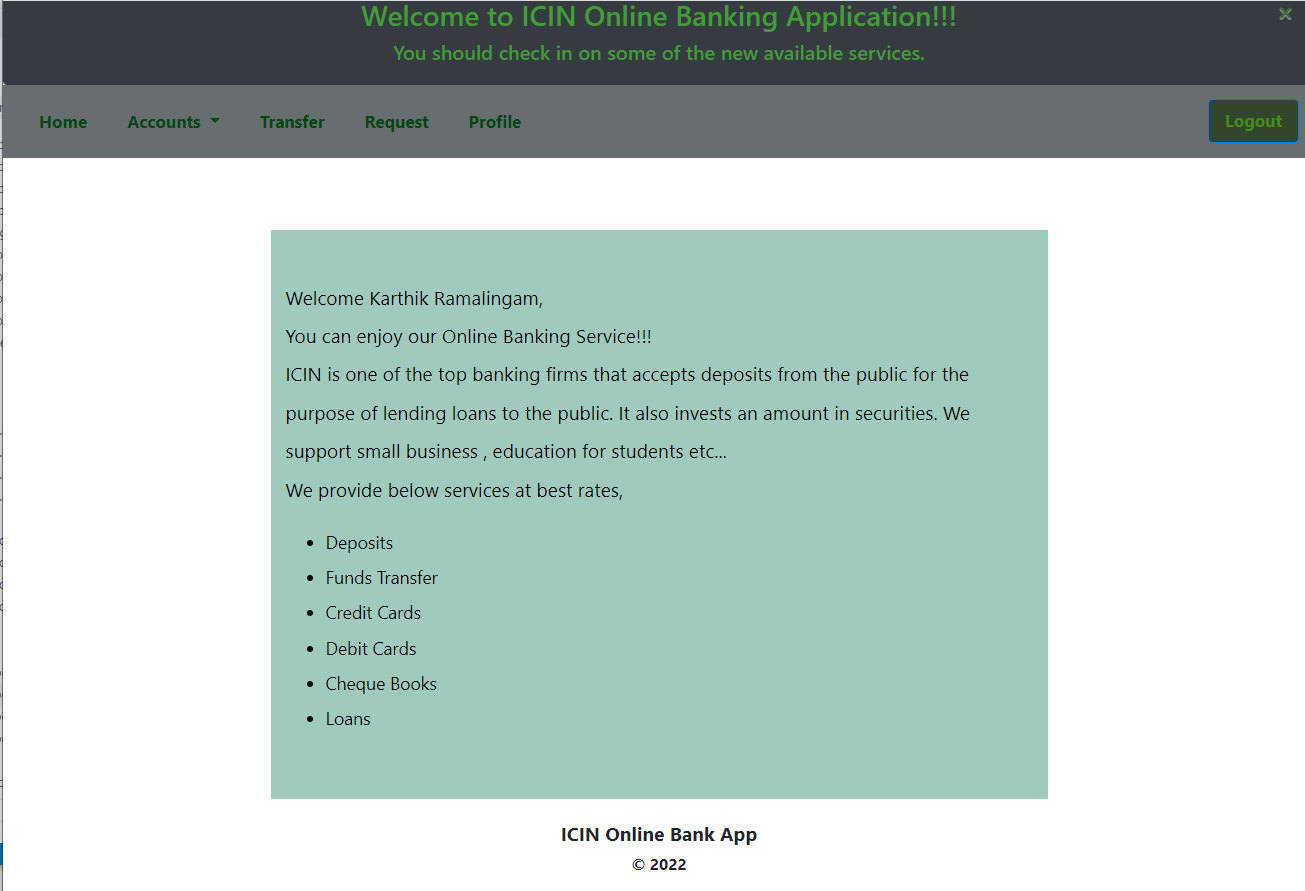
* Upon successful approval, database gets updated with ‘approved’ status.

****

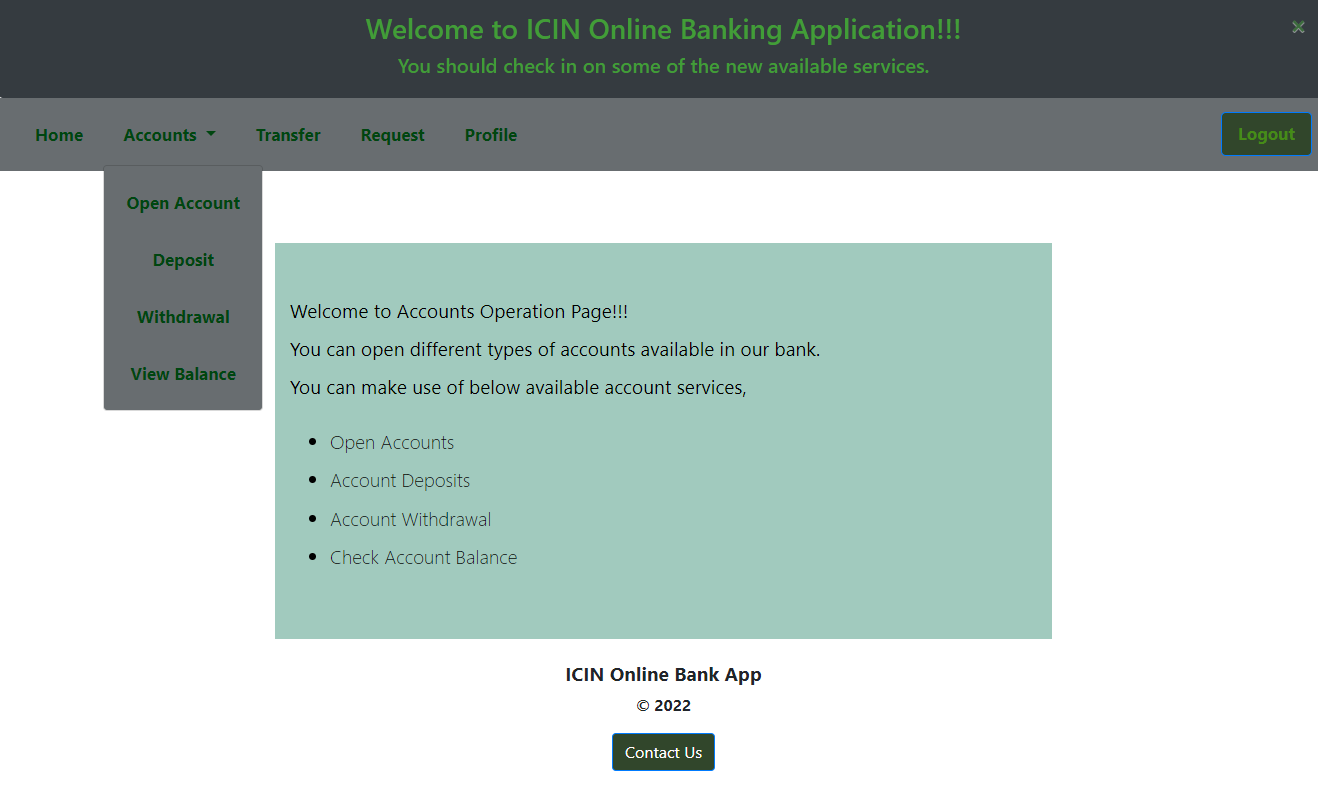
* After successful customer record approval, user can login successfully into the online banking app.

****

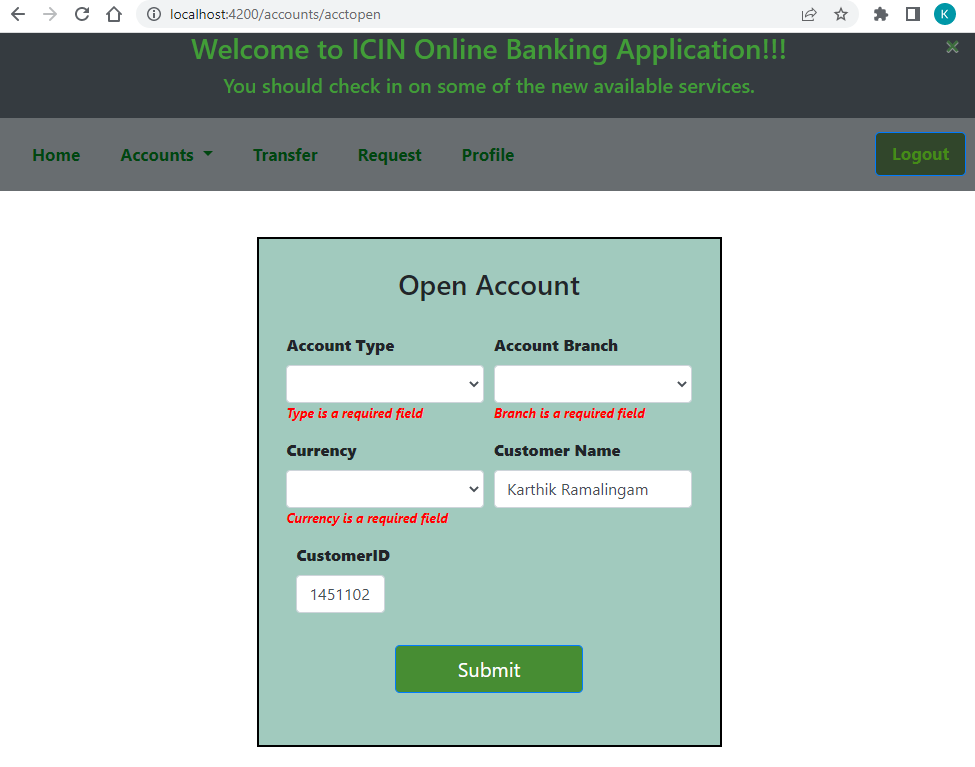
* Customer logs into the below shown home page.

****

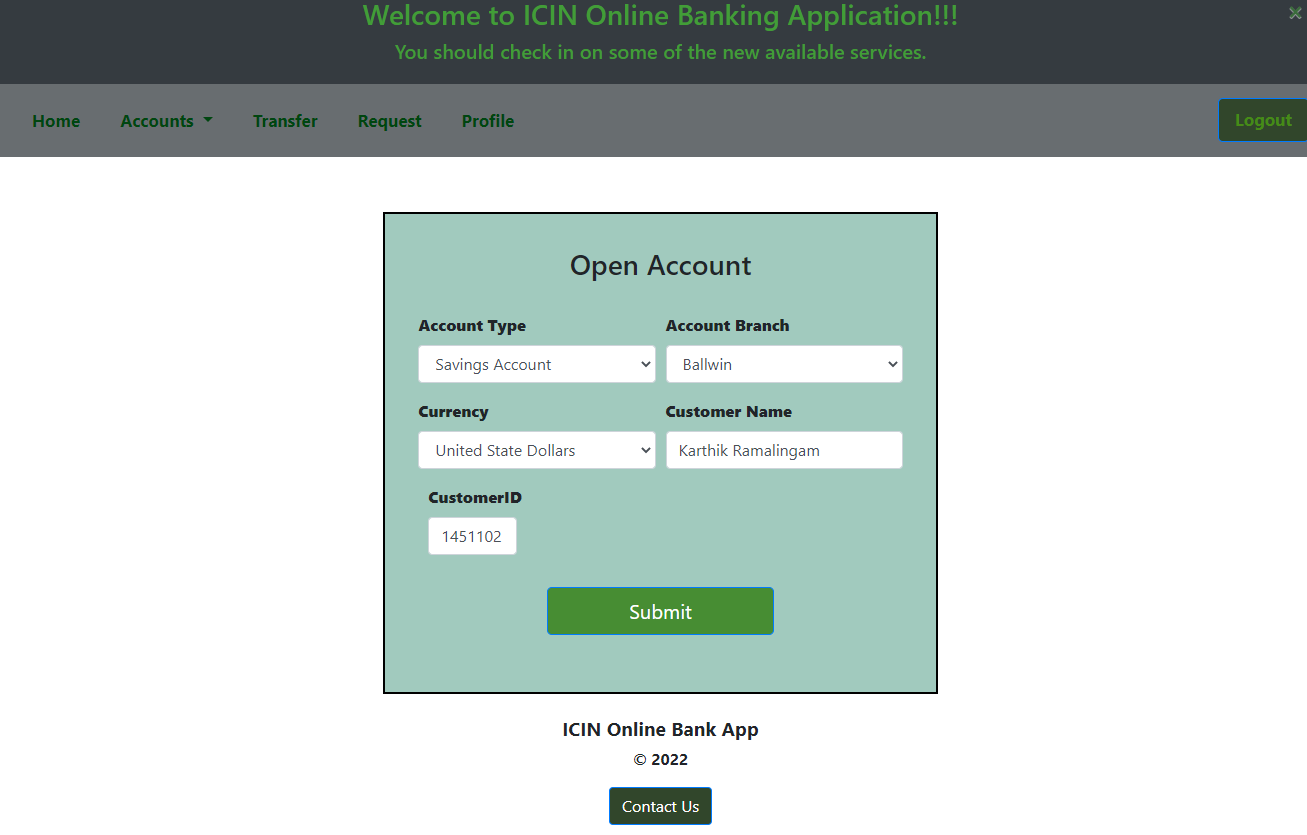
* Customer can perform accounts related transactions from the below shown options.

****

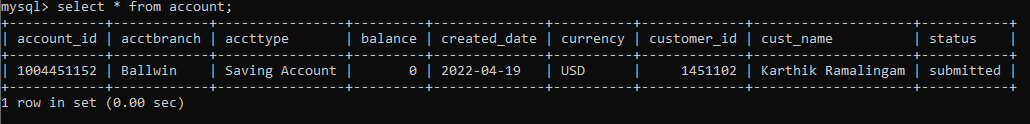
* ‘Open Account’ form basic validations,

****

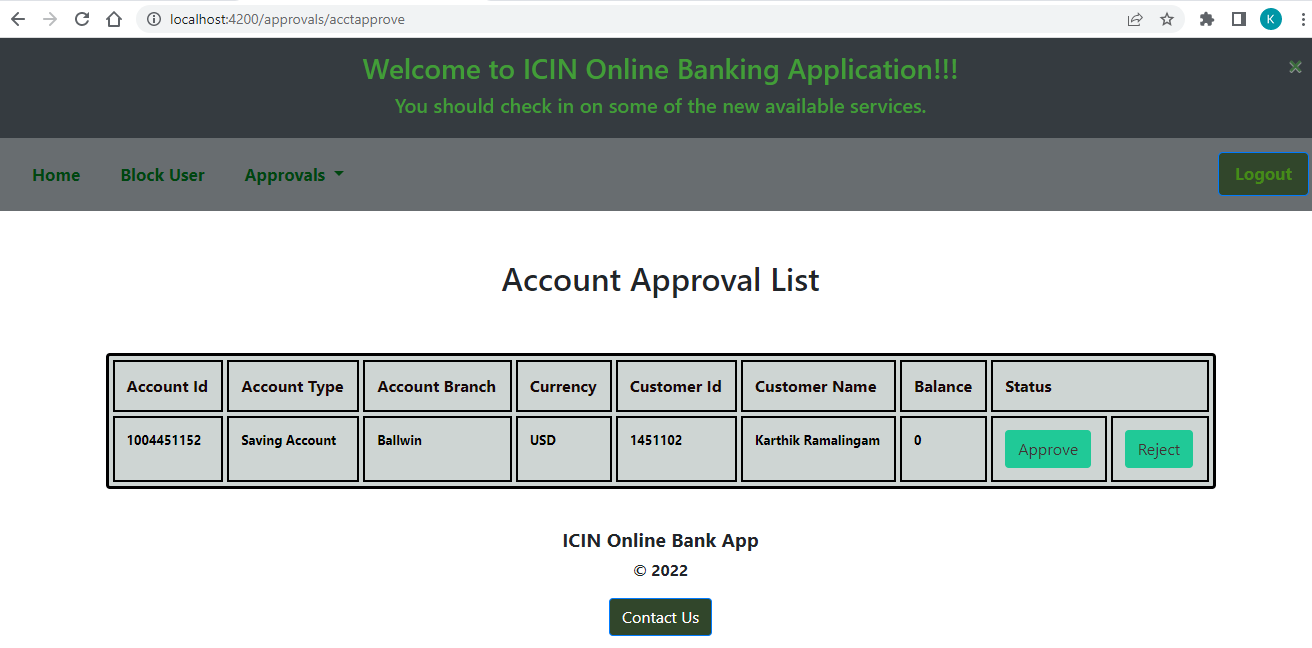
* Customer related fields gets auto populated based on logged in customer. Enter account related info and submit the form.

****

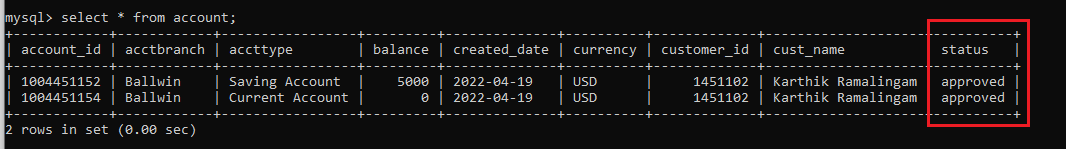
* Upon successful submit, database gets updated with the account details.

****

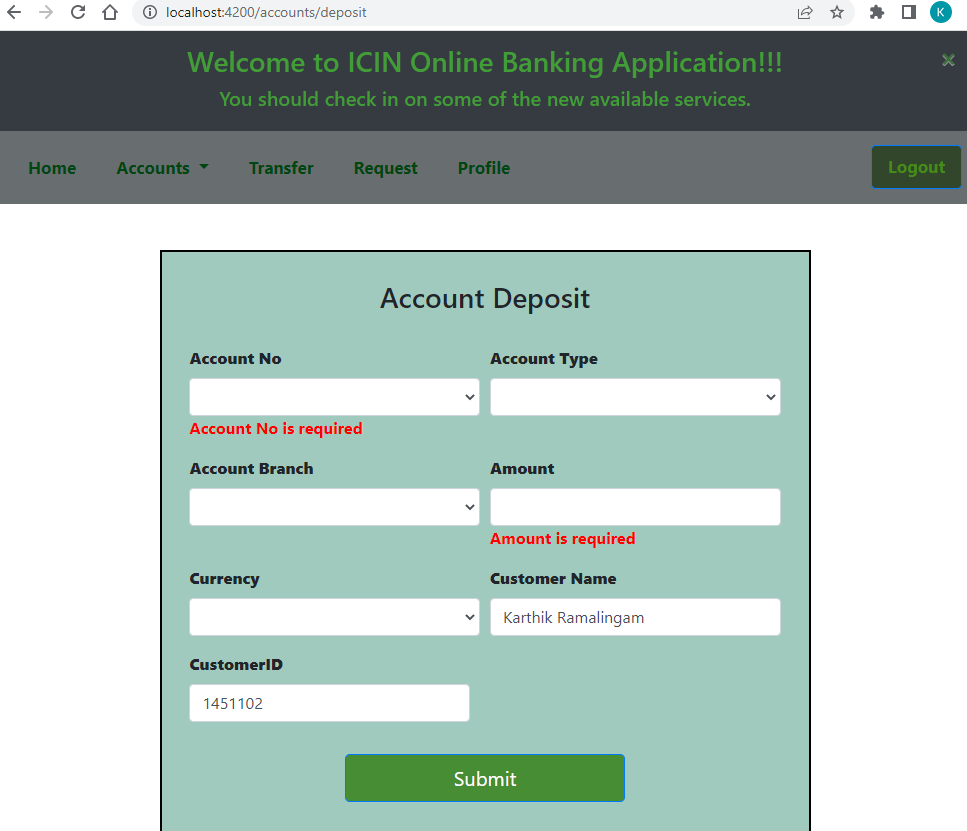
* Admin approves the account record,

****

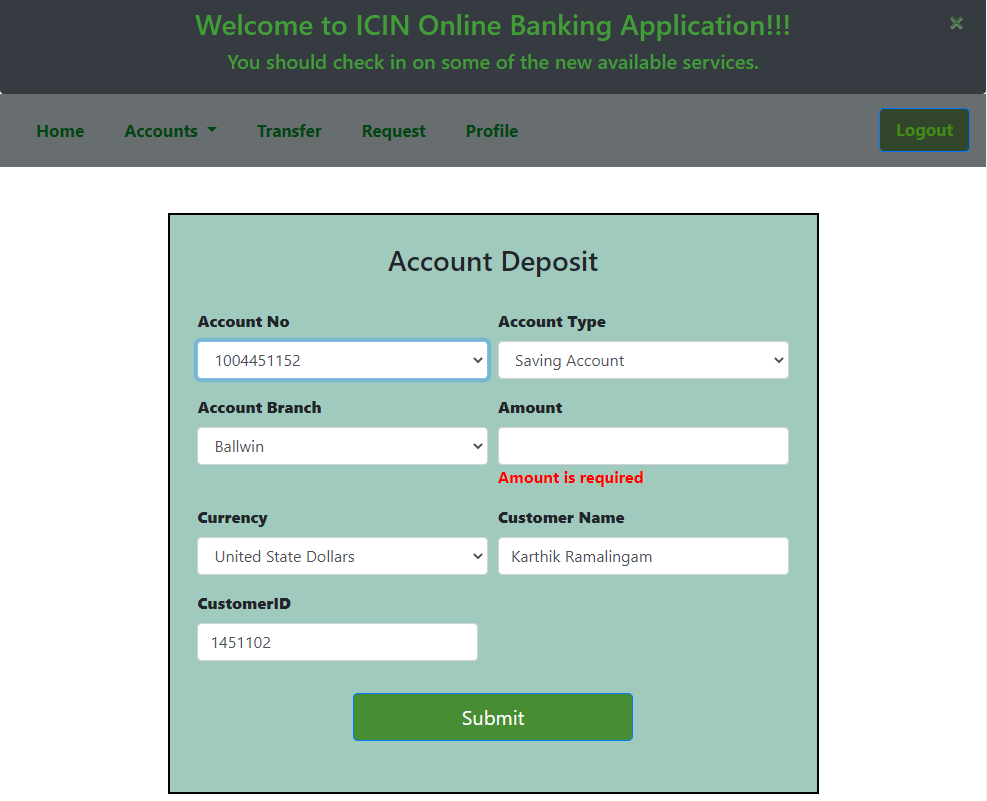
* Database gets updated with ‘approved’ status.

****

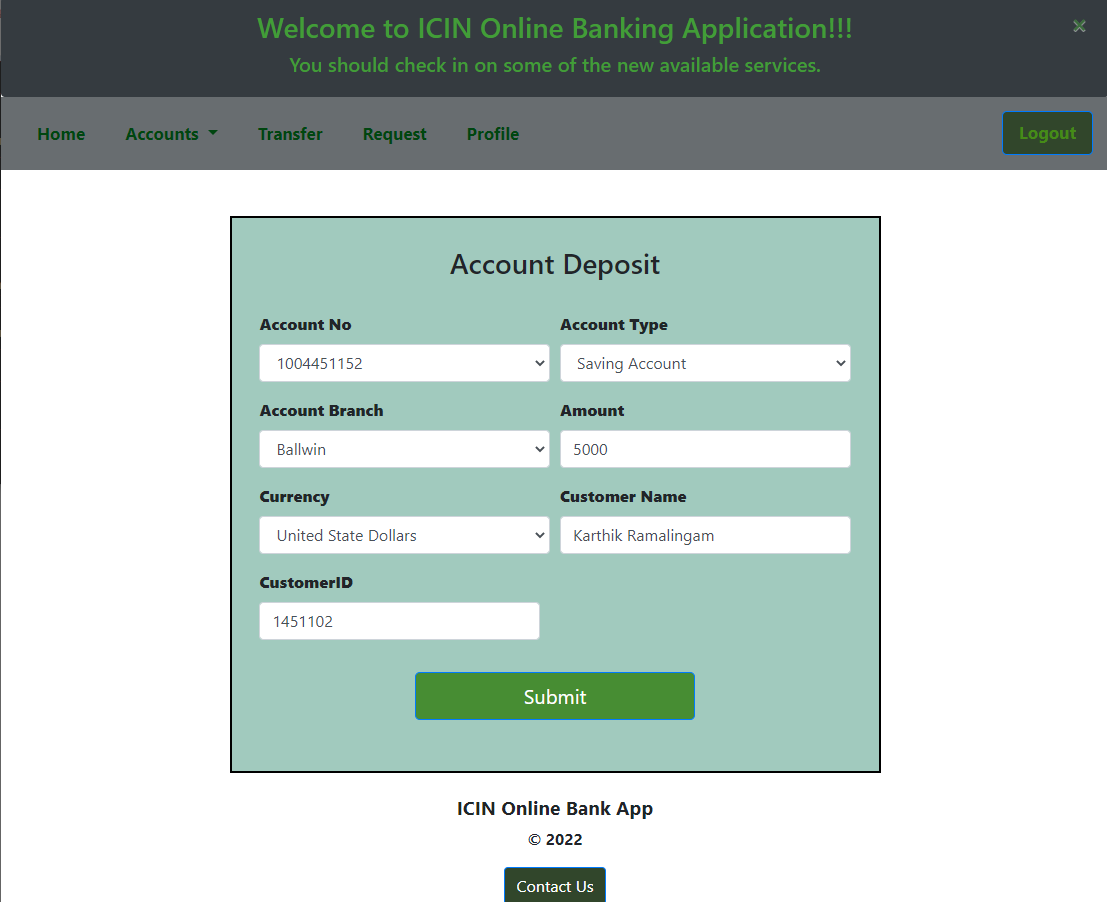
* ‘Account deposit’ form validations,

****

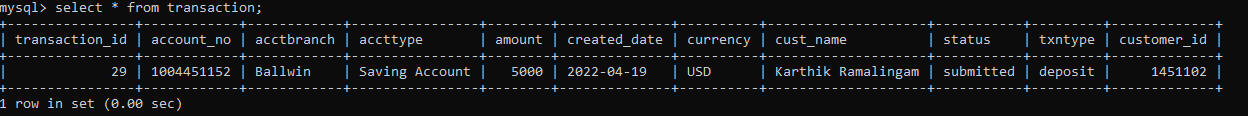
* Customer related fields gets auto populated based on logged in customer. Upon selecting account, account related fields get auto populated.

****

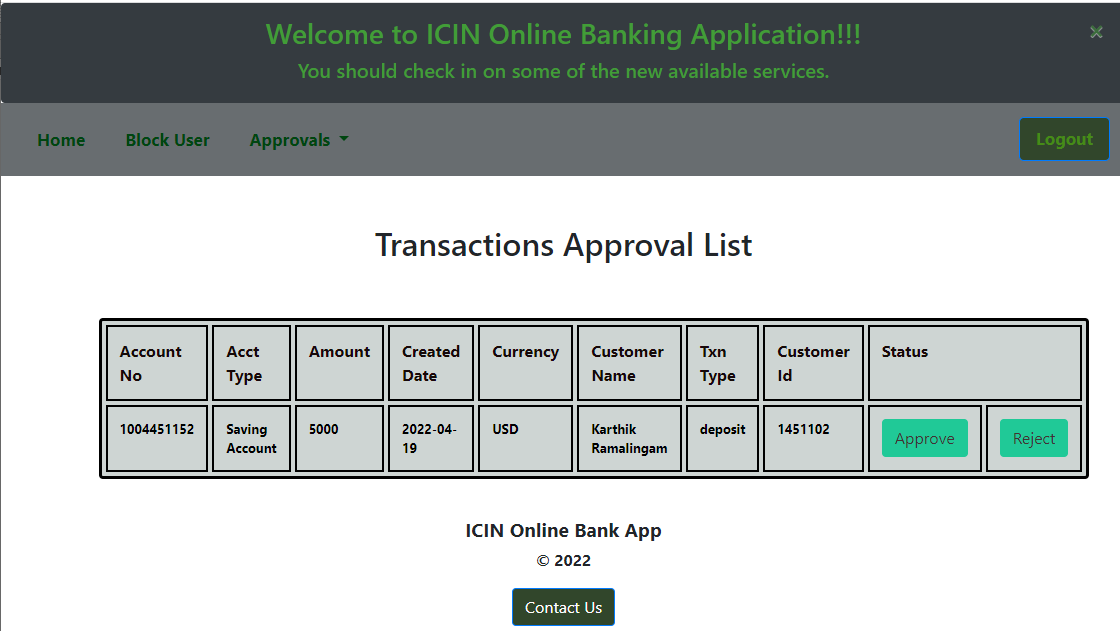
* Enter all deposit fields and click on submit button.

****

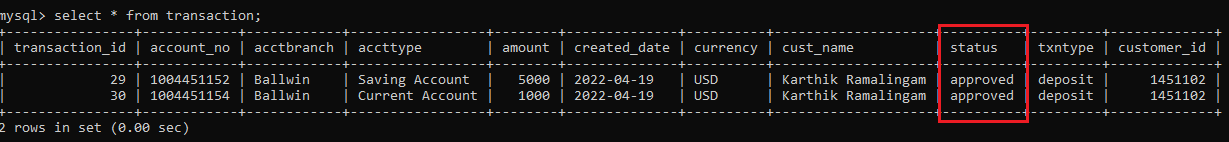
* Upon successful submit, database table gets updated with the deposit details.

****

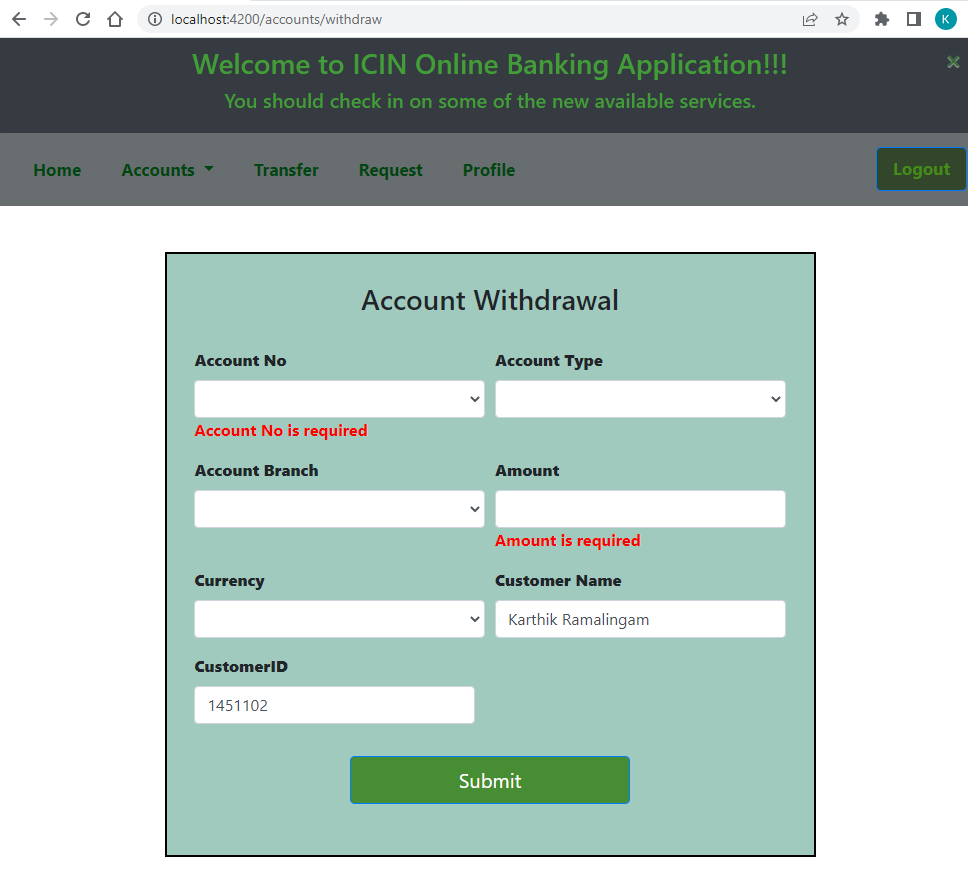
* Admin approves the transaction,

****

* Database table gets updated with ‘approved’ status,

****

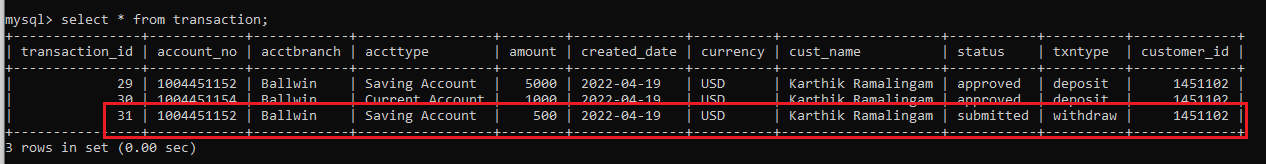
* ‘Account withdrawal’ form with validations,

****

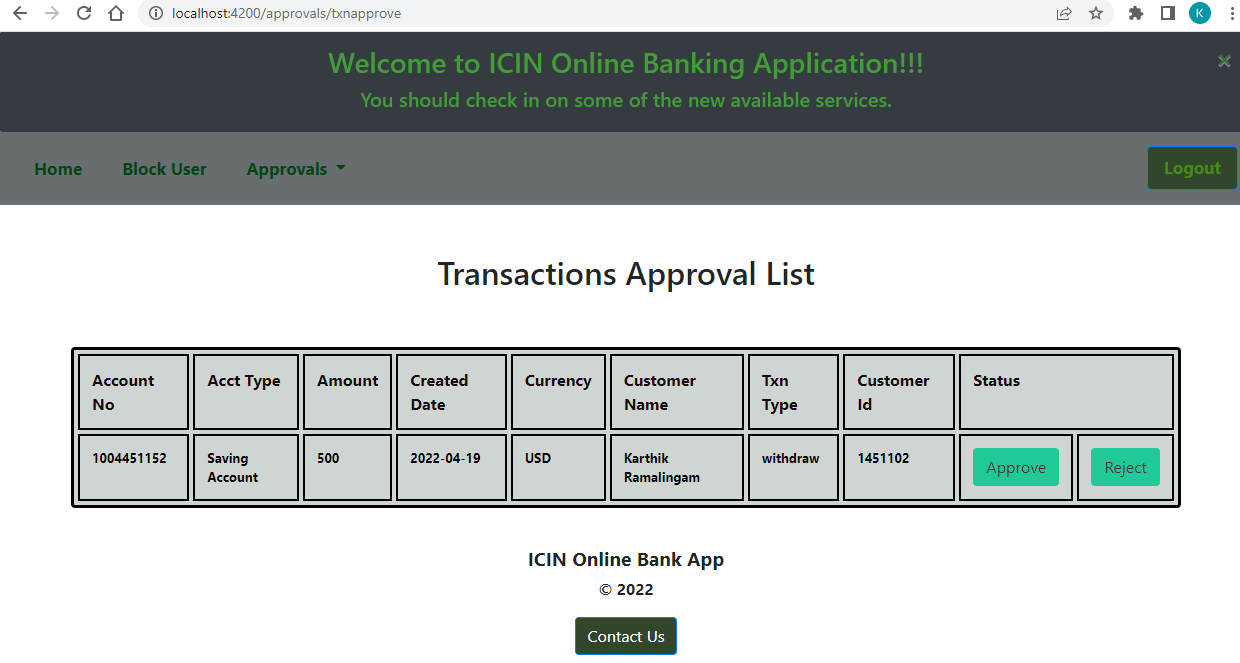
* Enter all account withdrawal fields and submit the form,

****

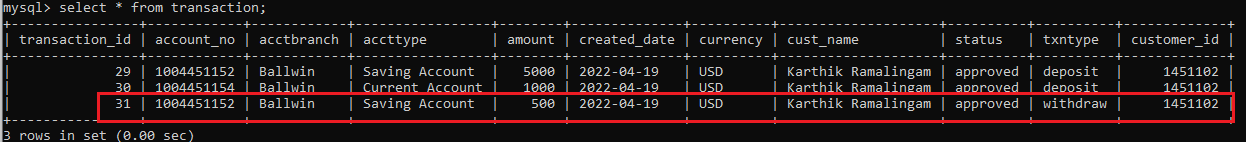
* Upon successful submission, database gets updated with the withdrawal details.

****

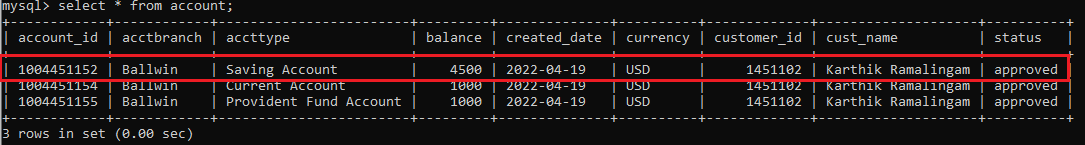
* Admin approves the withdrawal transaction,

****

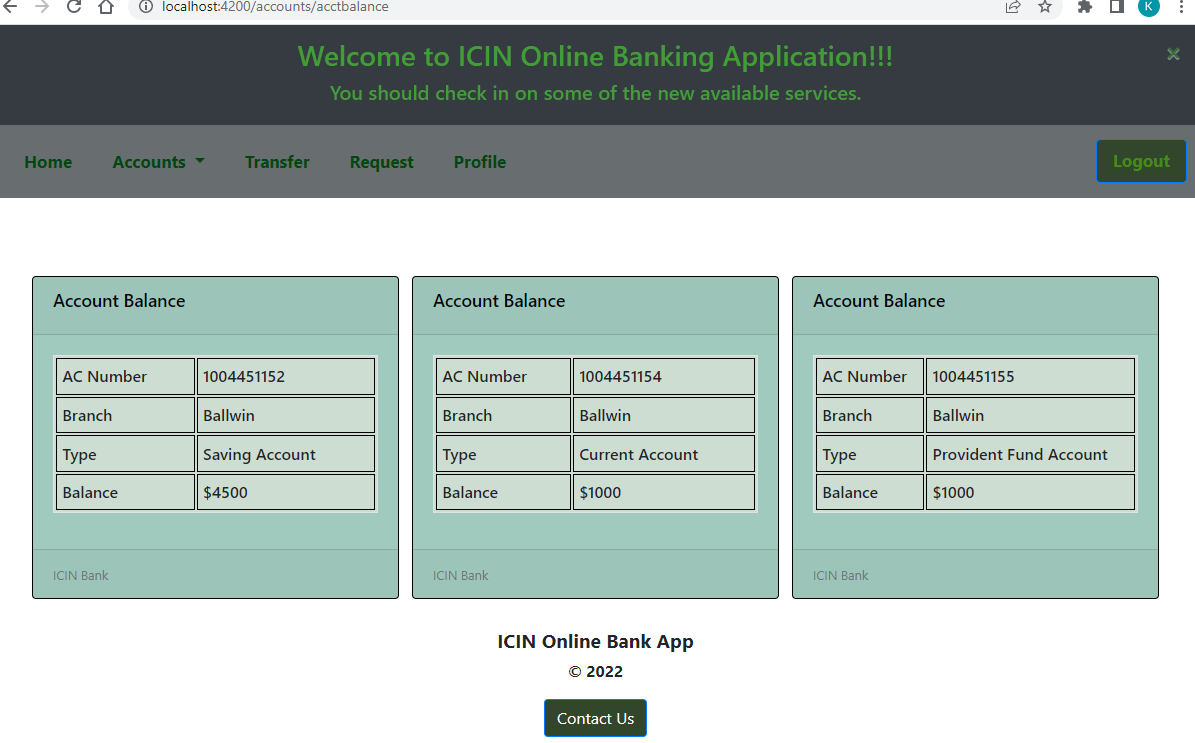
* Database gets updated with the status as ‘approved’.

****

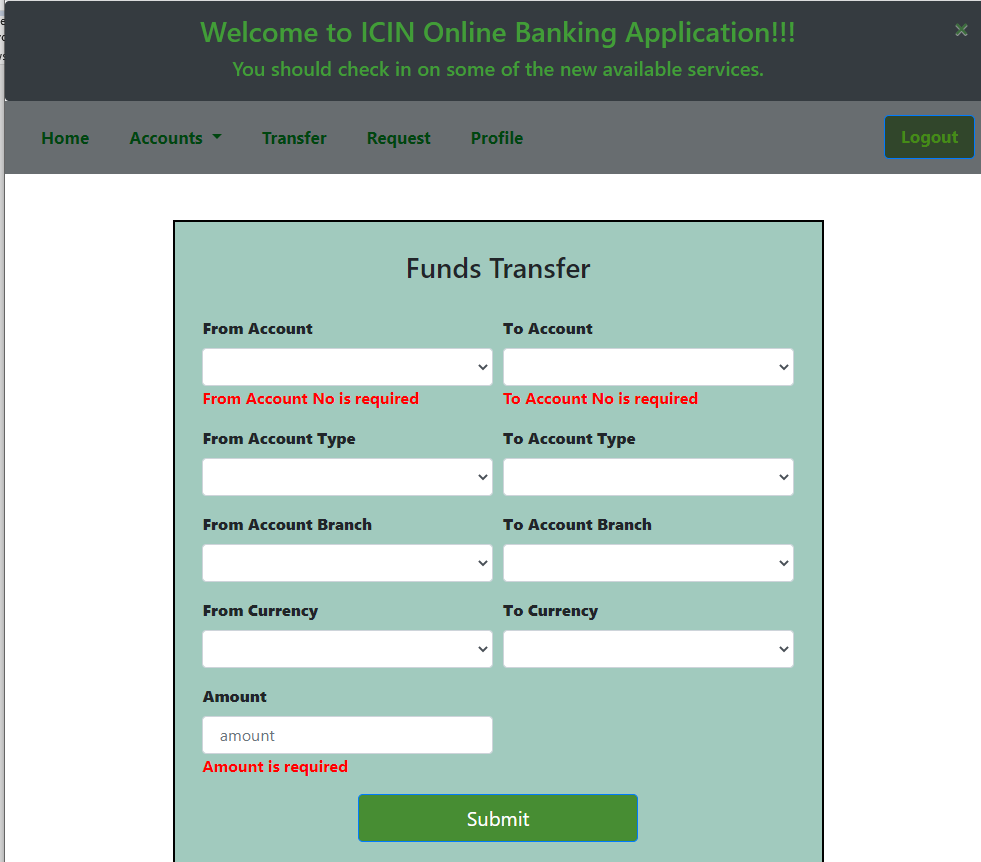
* Upon successful submission, database gets updated with the updated account balance based on the transaction.

****

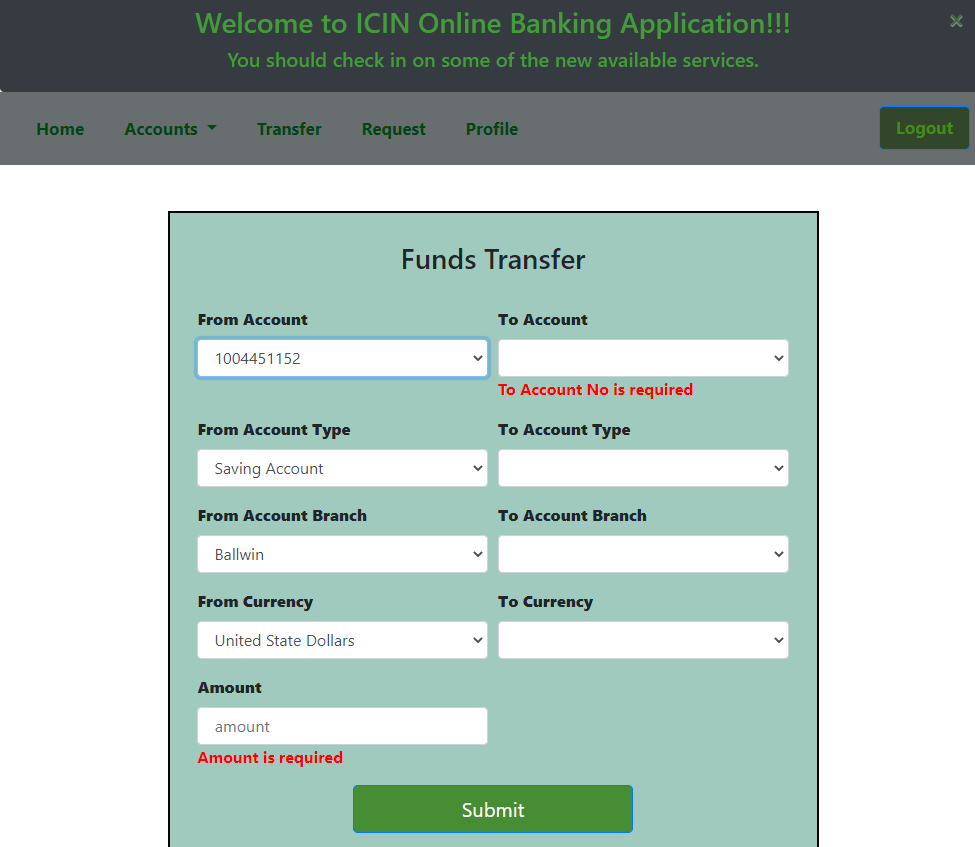
* Account balance enquiry,

****

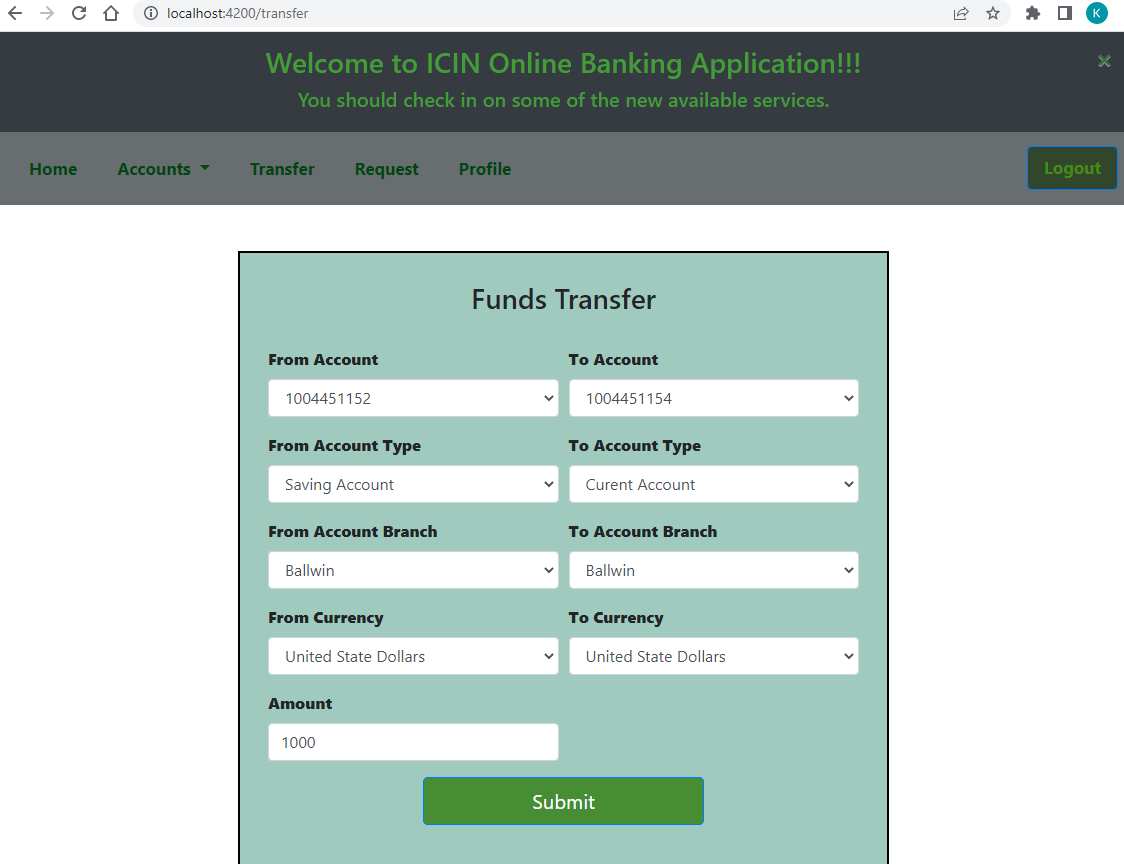
* ‘Funds transfer’ form with field validations,

****

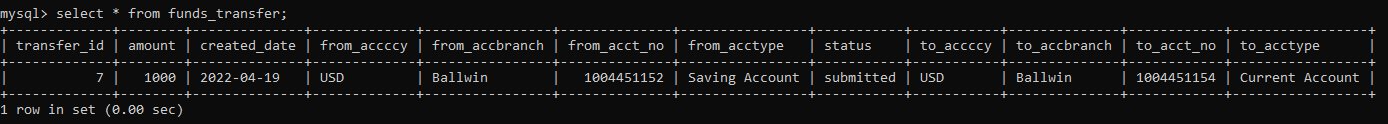
* Upon entering the account no, account related fields get auto populated.

****

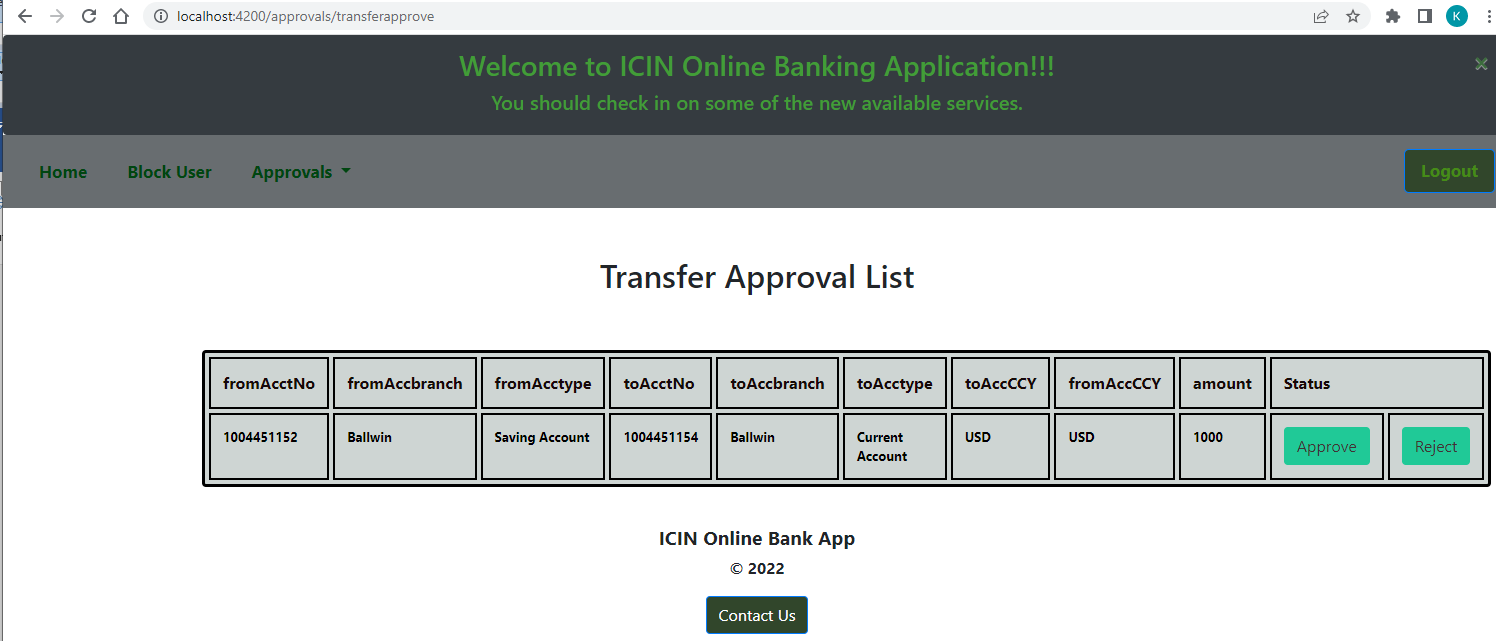
* Enter transfer details and submit the form.

****

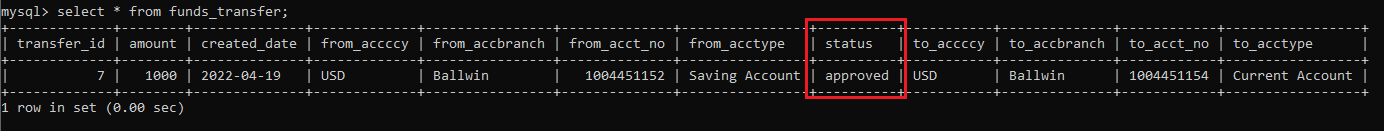
* Database gets updated with the transfer account details.

****

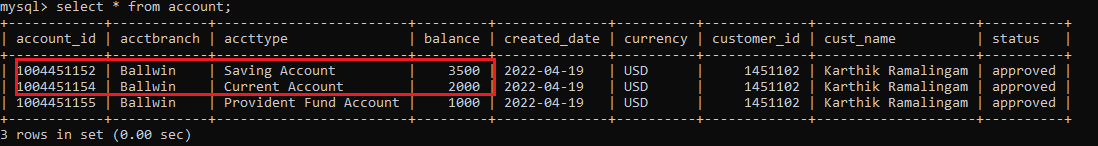
* Admin approves the transfer,

****

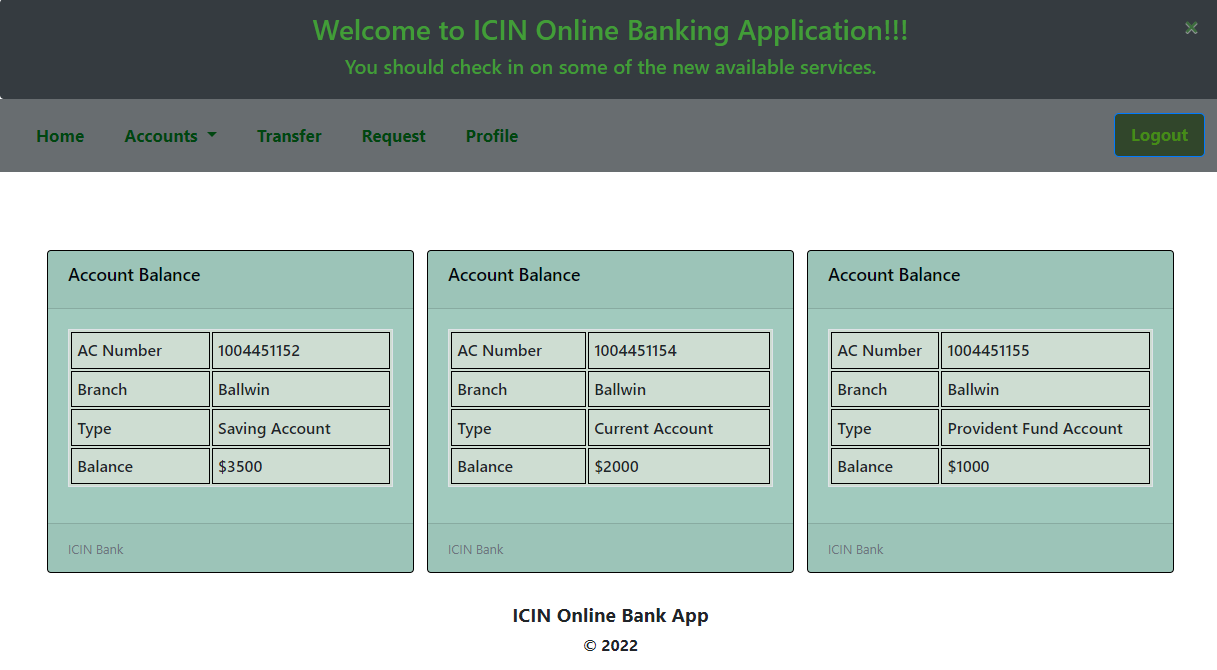
* Database gets updated with the ‘approved’ status.

****

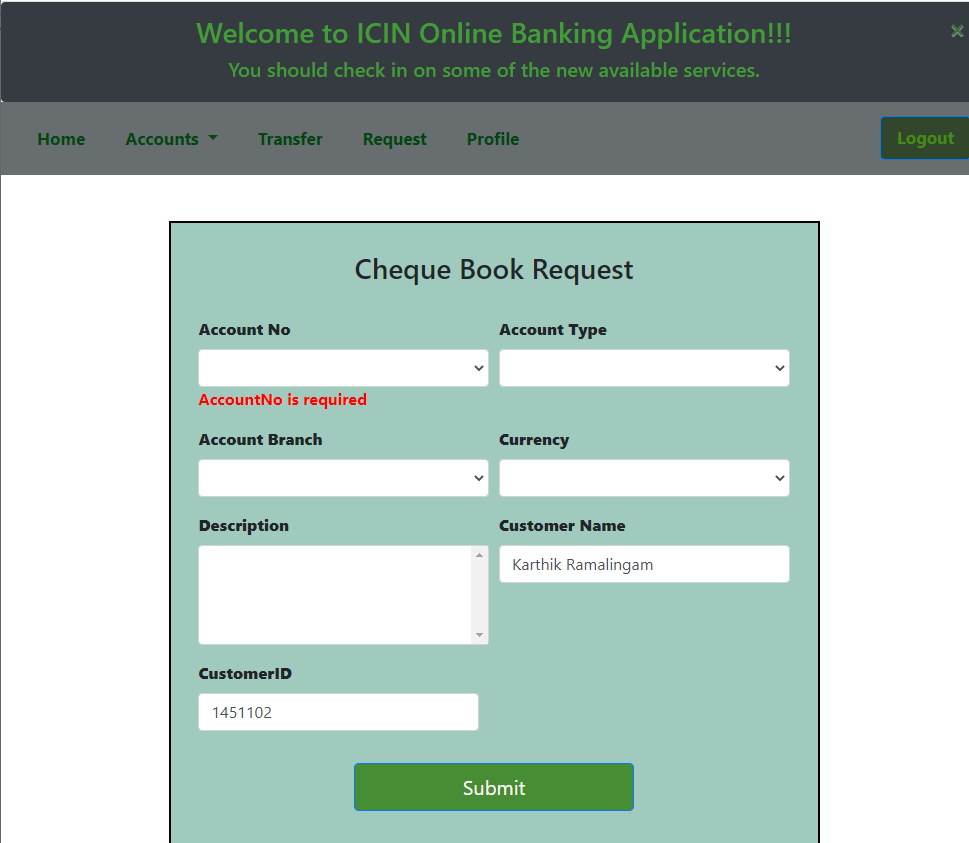
* Account balances are updated accordingly,

****

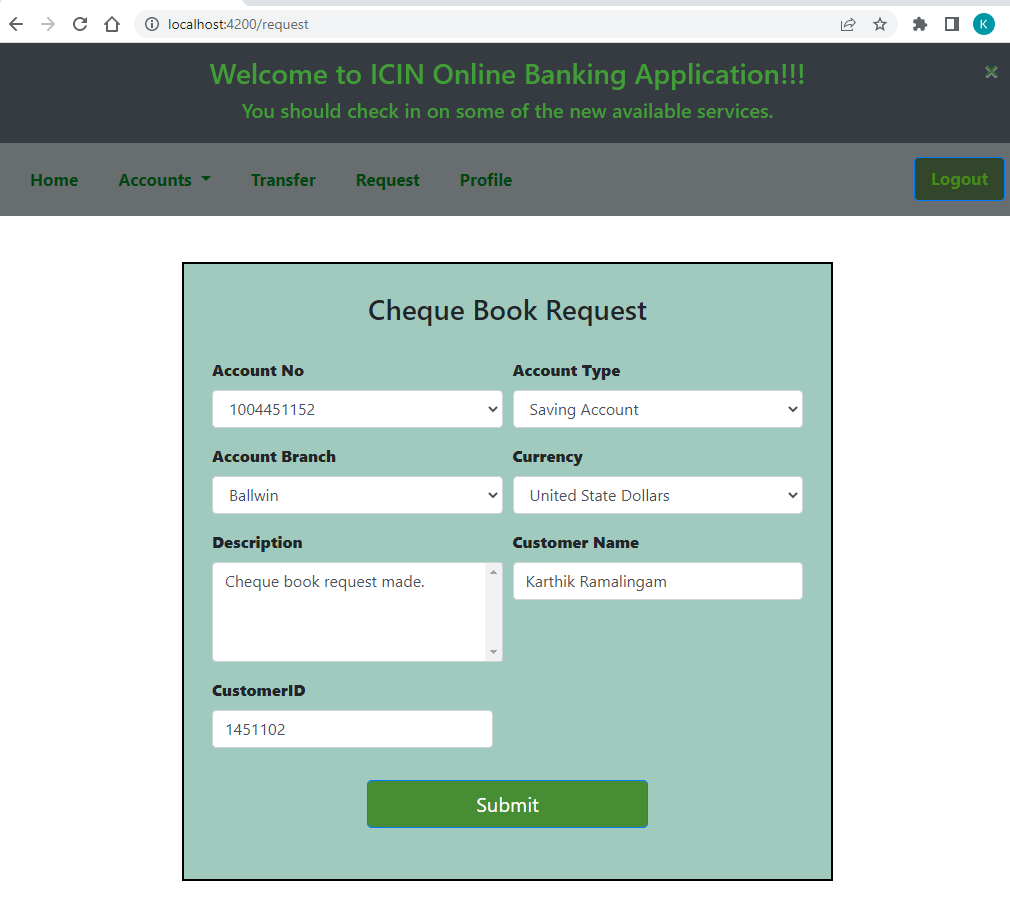
* View the account balance enquiry for the updated balance,

****

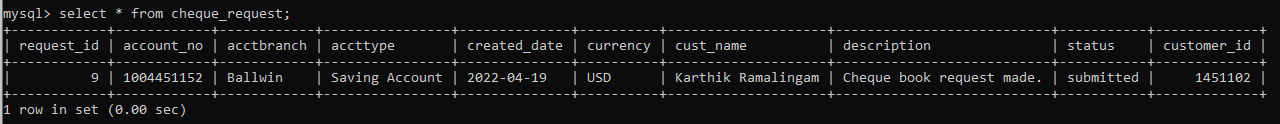
* ‘Cheque request’ form validations,

****

* Enter all the cheque request form fields and submit the form,

****

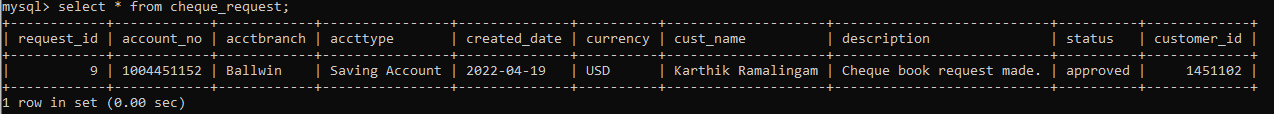
* Upon successful submission, database gets updated.

****

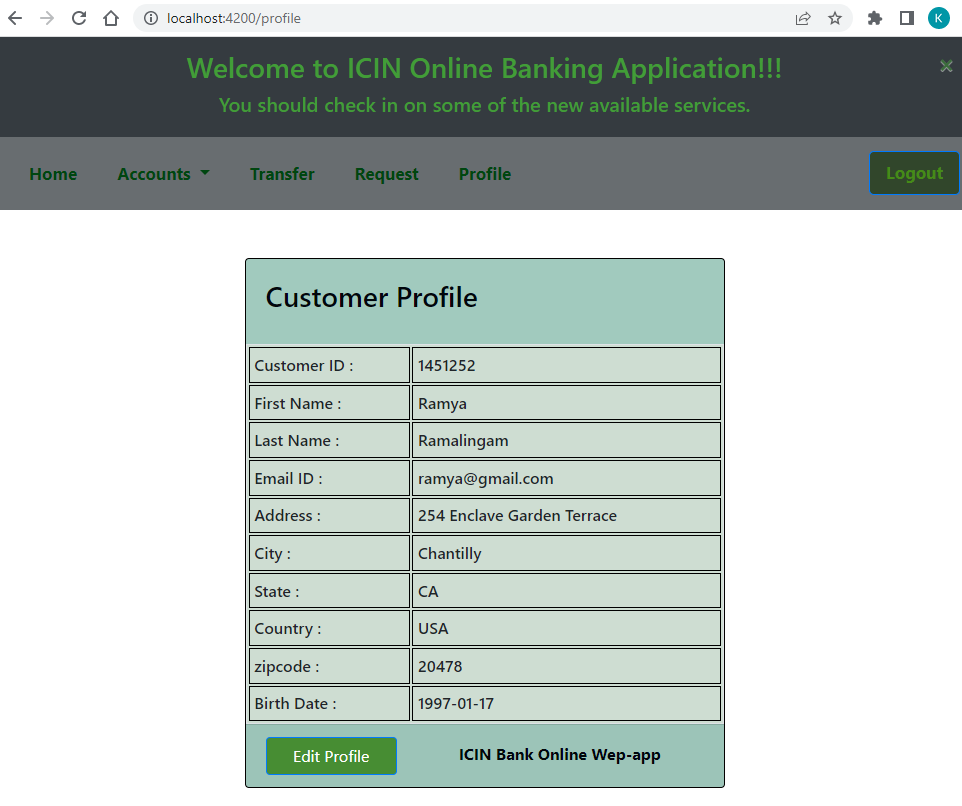
* Admin approves the cheque request,

****

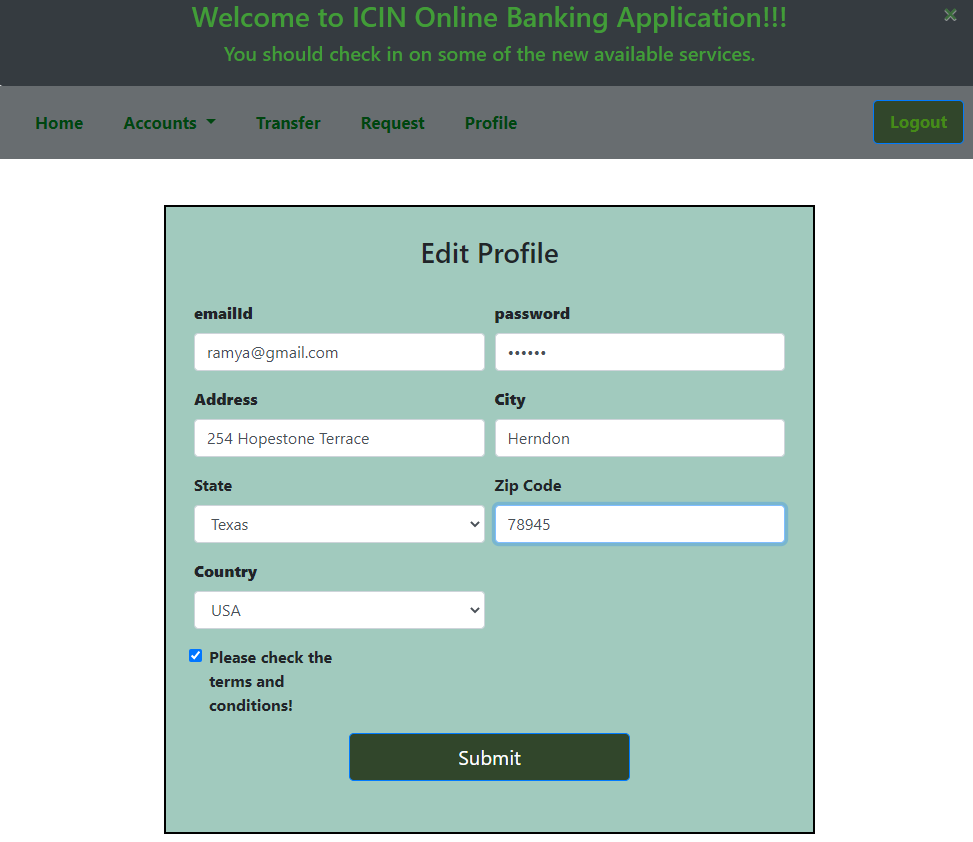
* Upon successful submission, cheque request table gets updated with ‘approved’ status.

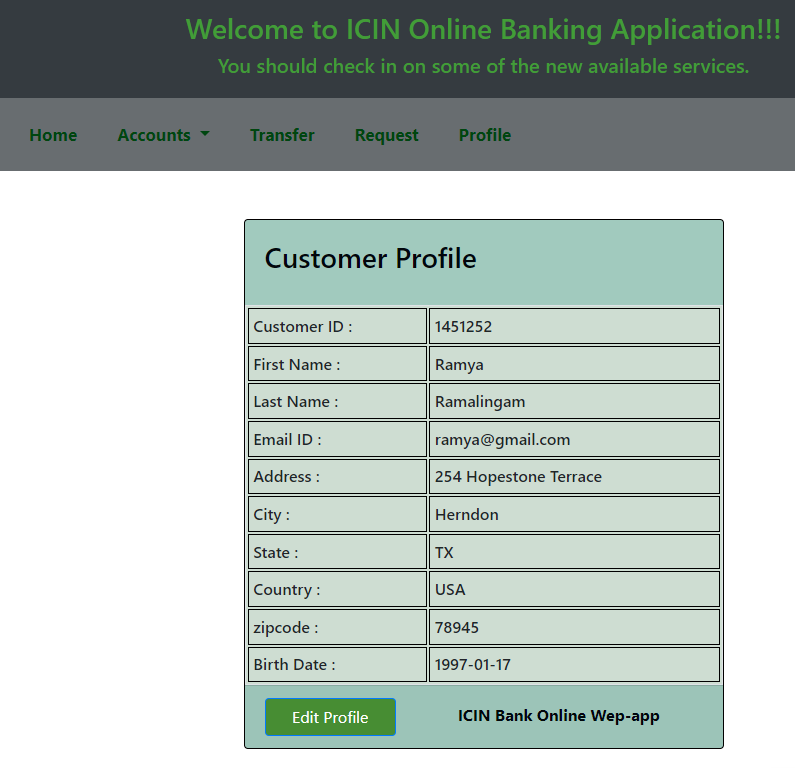
****

* Customer can view the profile as shown below,

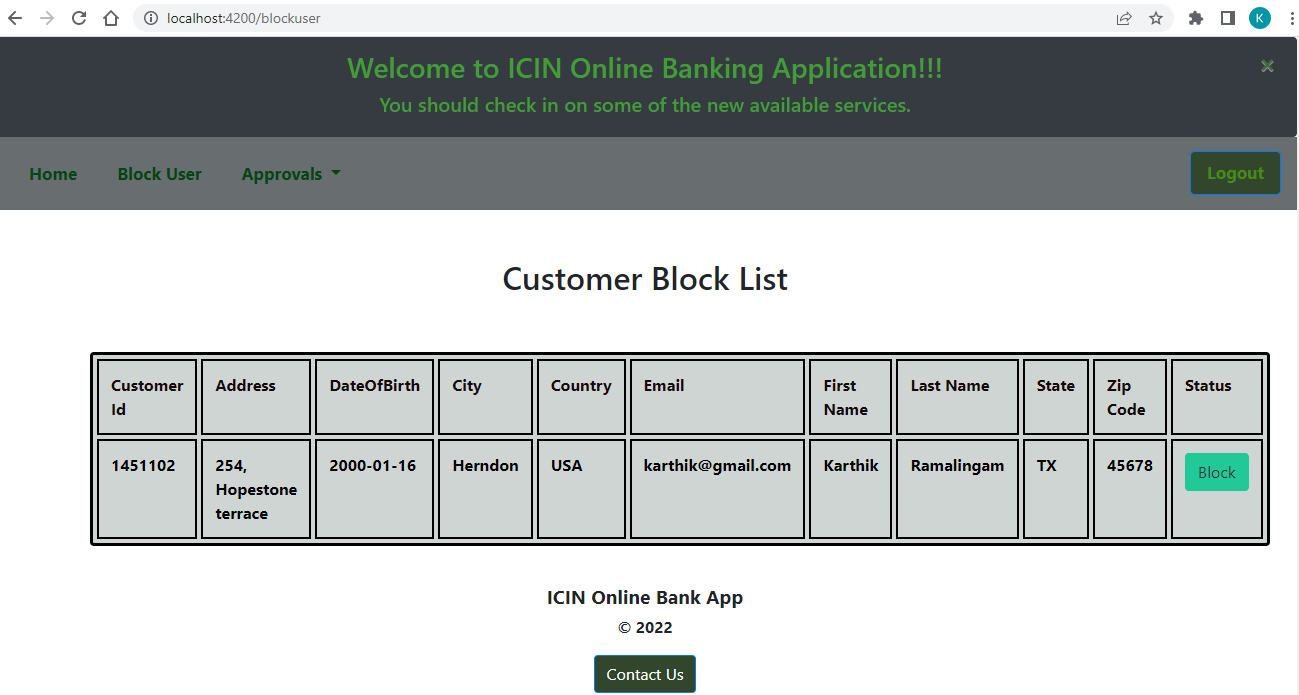
****

* Customer can edit the profile as shown below,

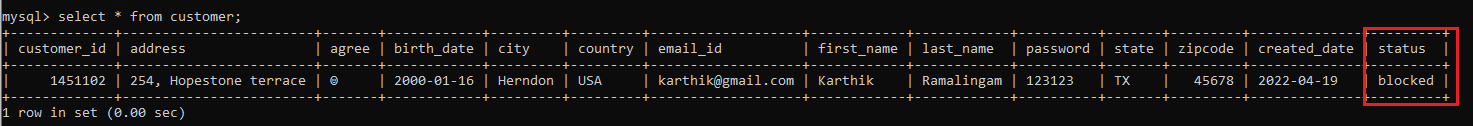
****

****

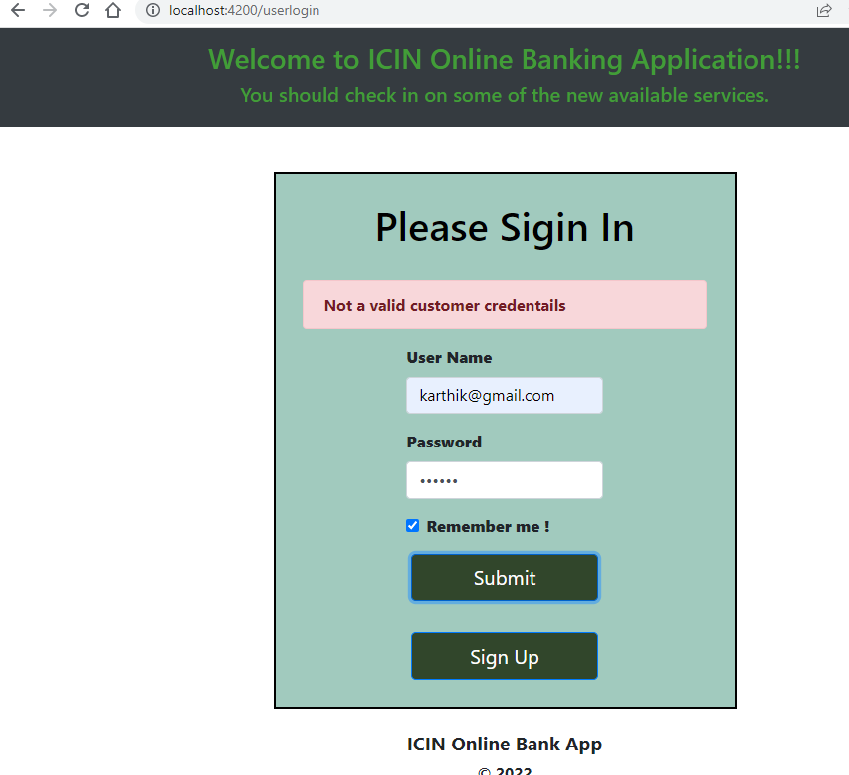
* Admin can block the users in-case of treats,

****

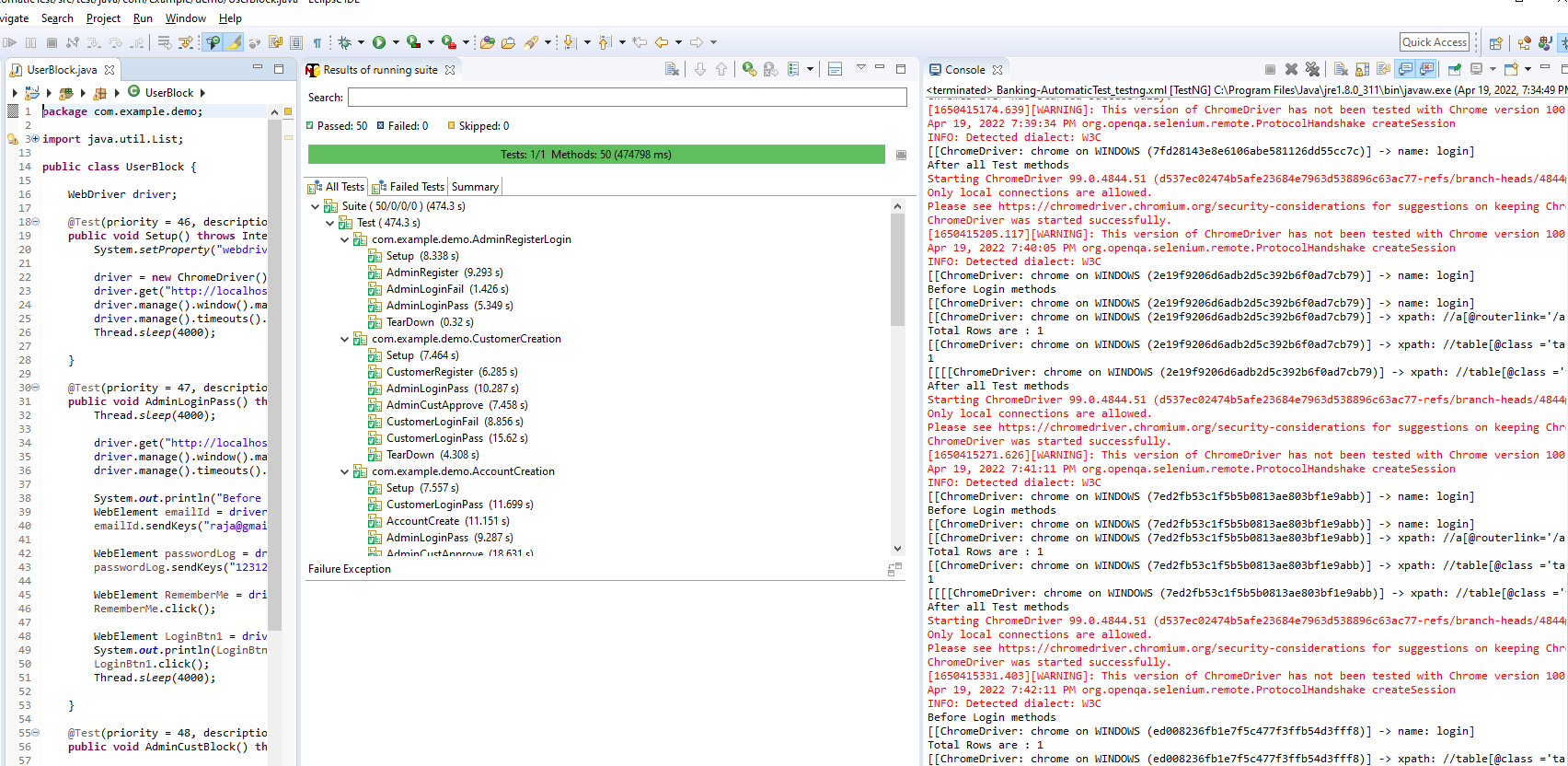
* Customer table gets updated with the ‘blocked’ status,

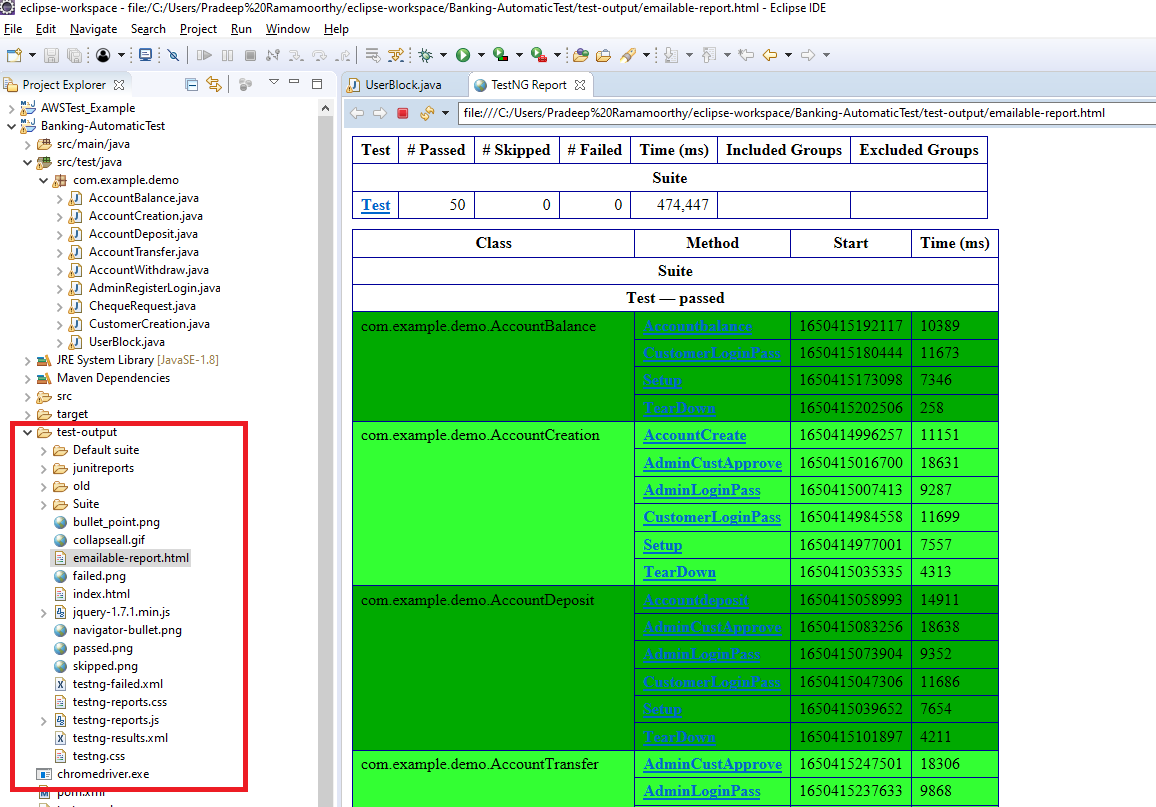
****

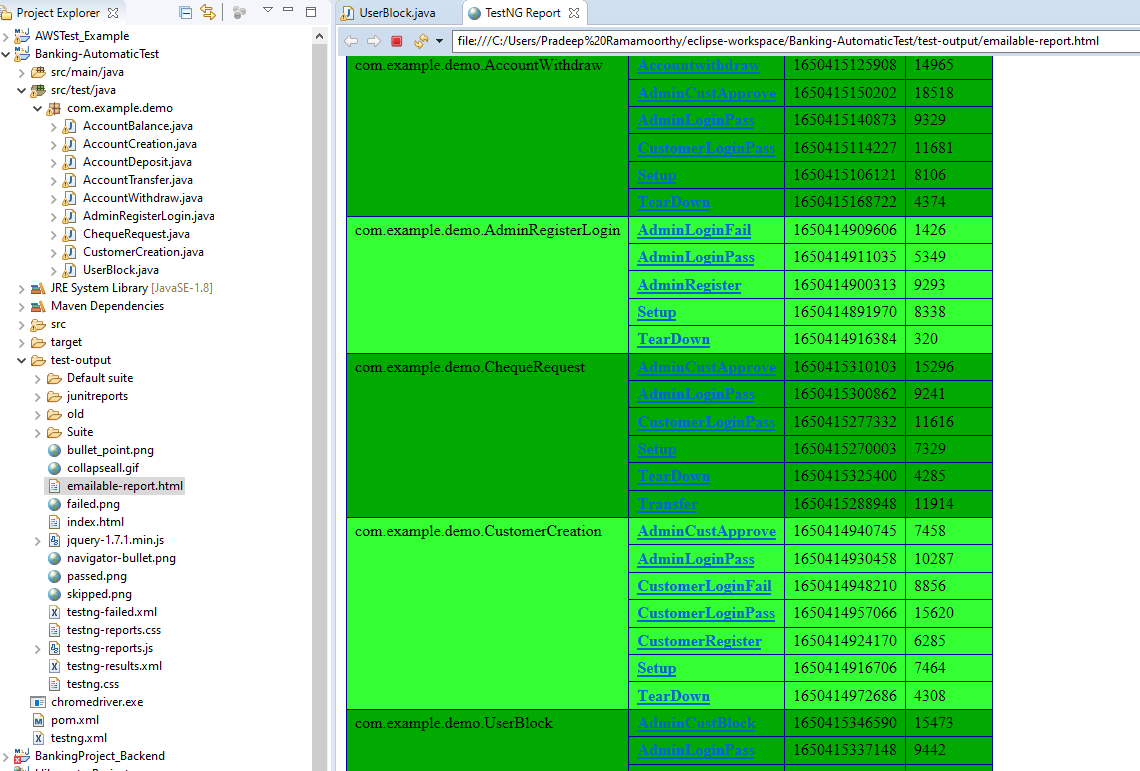
* Customer is not allowed to login after blocking,

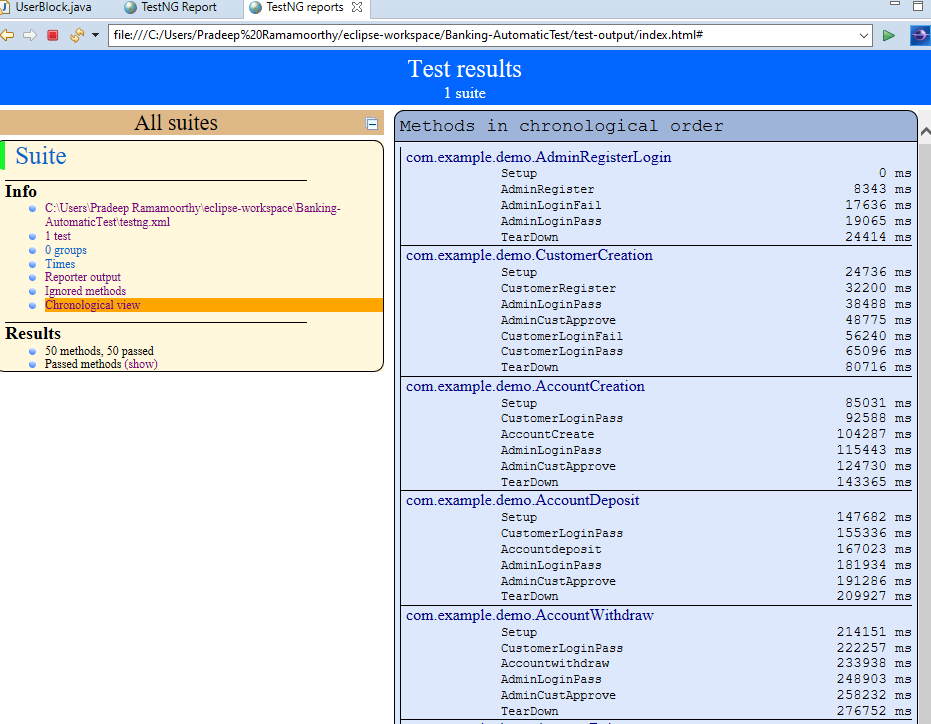
****

* TestNG and selenium automated tool testcase output,

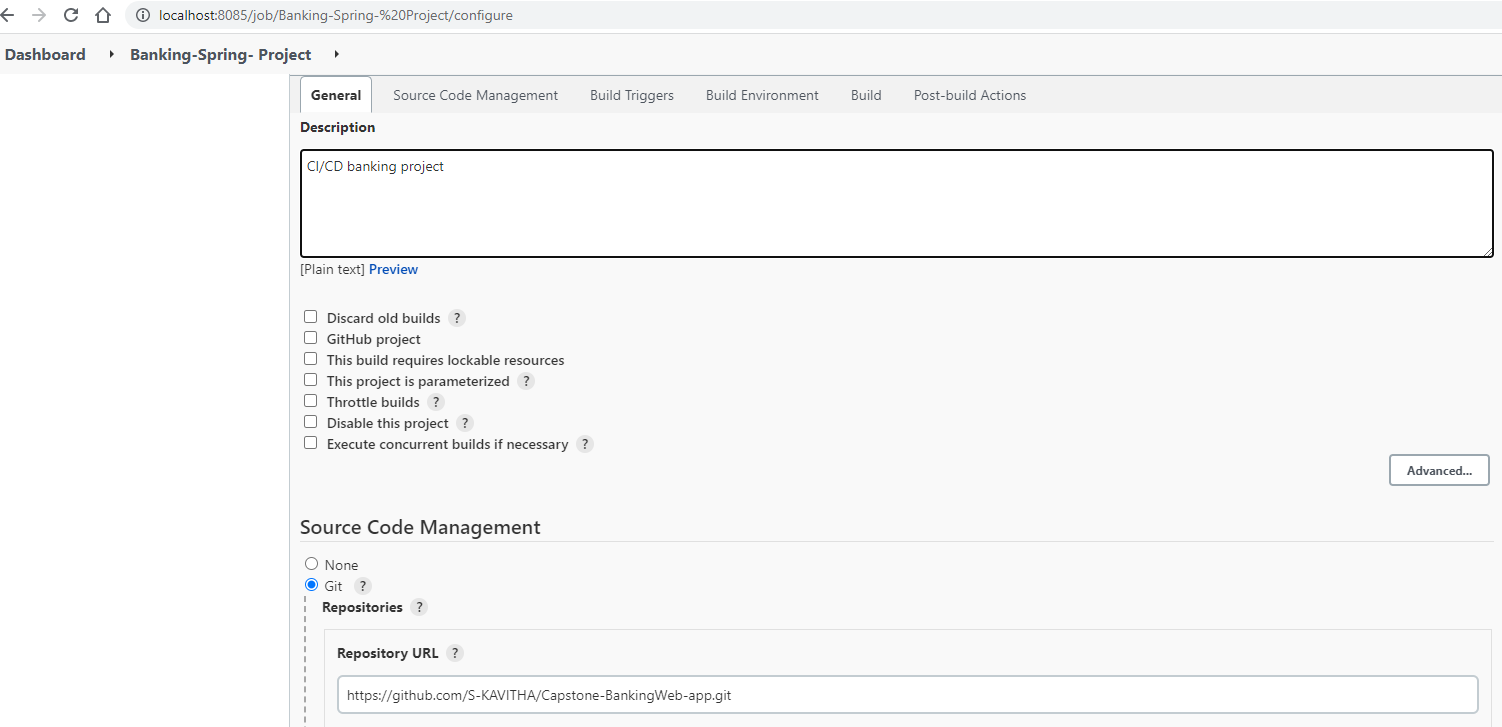
****

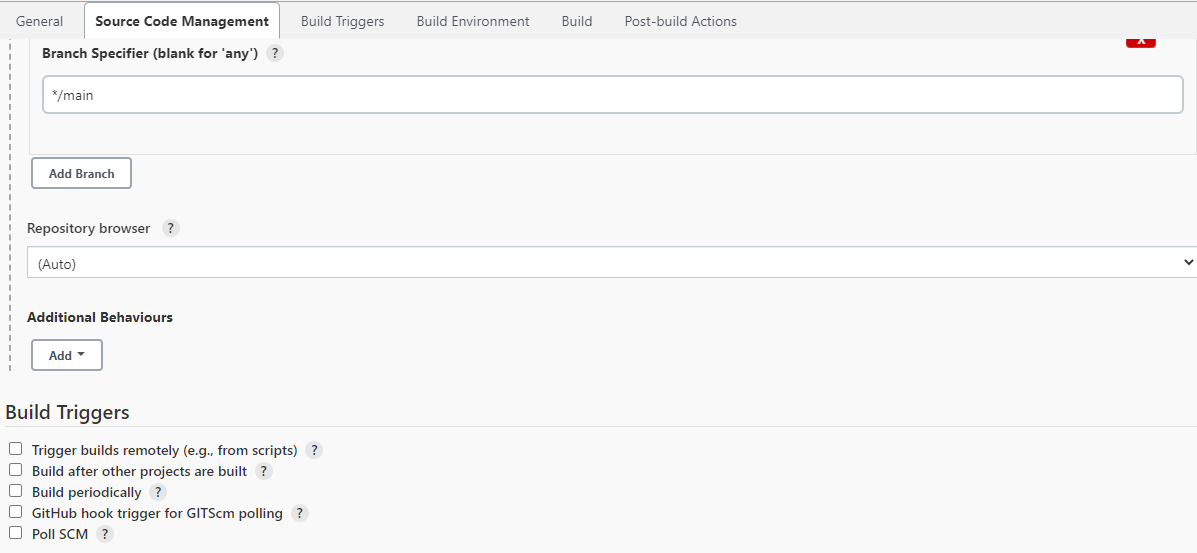
****

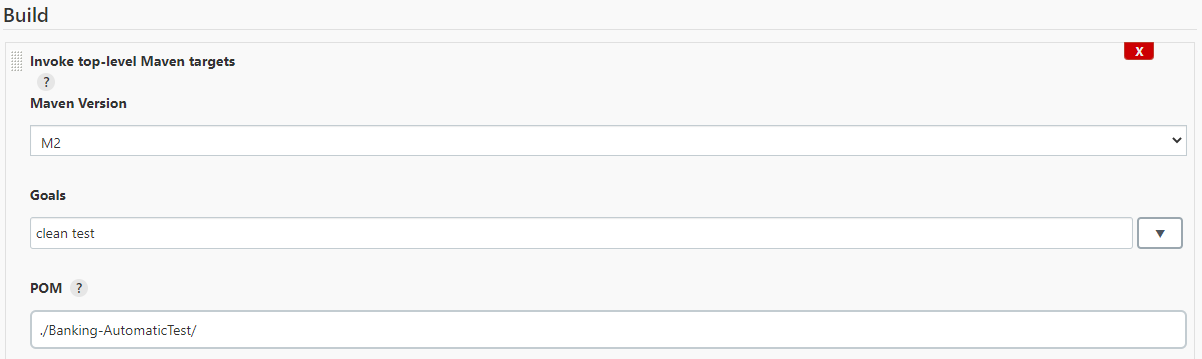
****

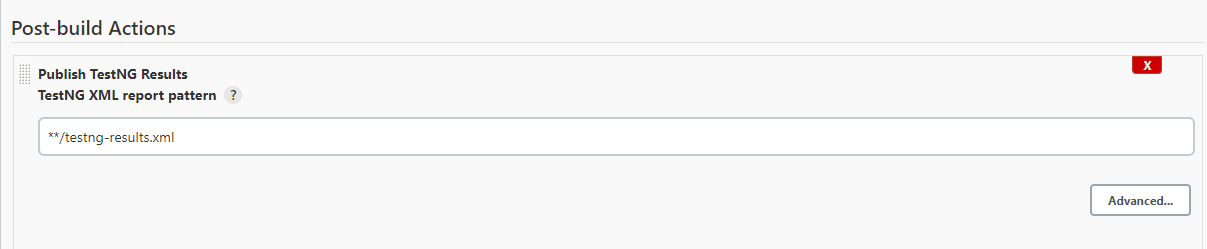
****

* Jenkins CI/CD configuration for the banking project,

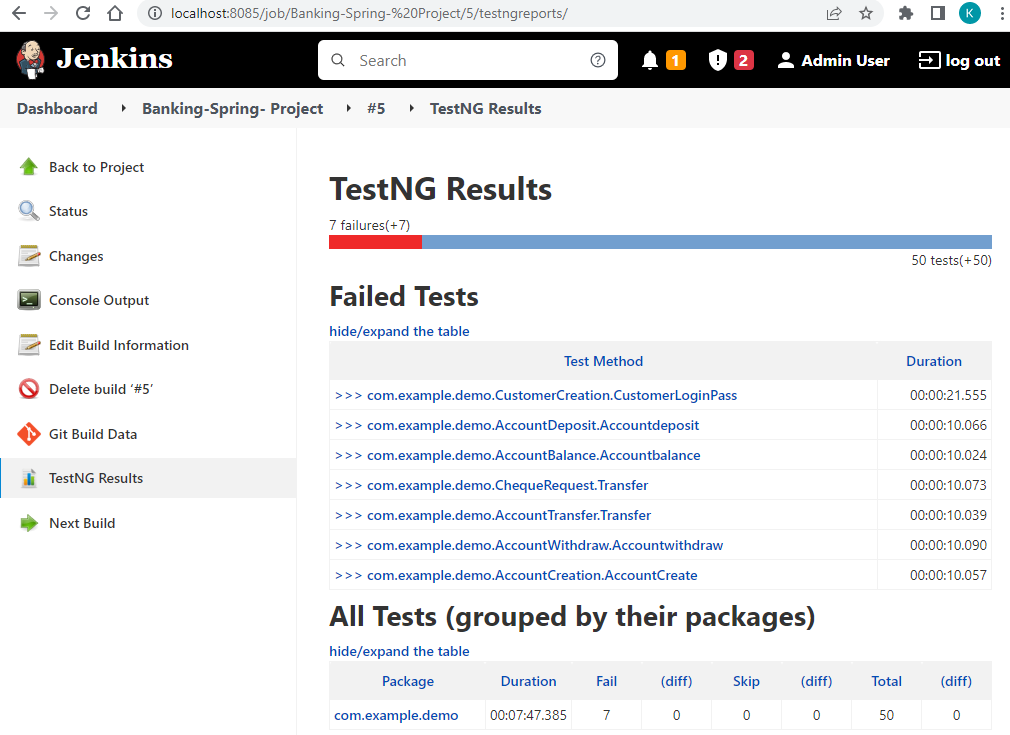
****

****

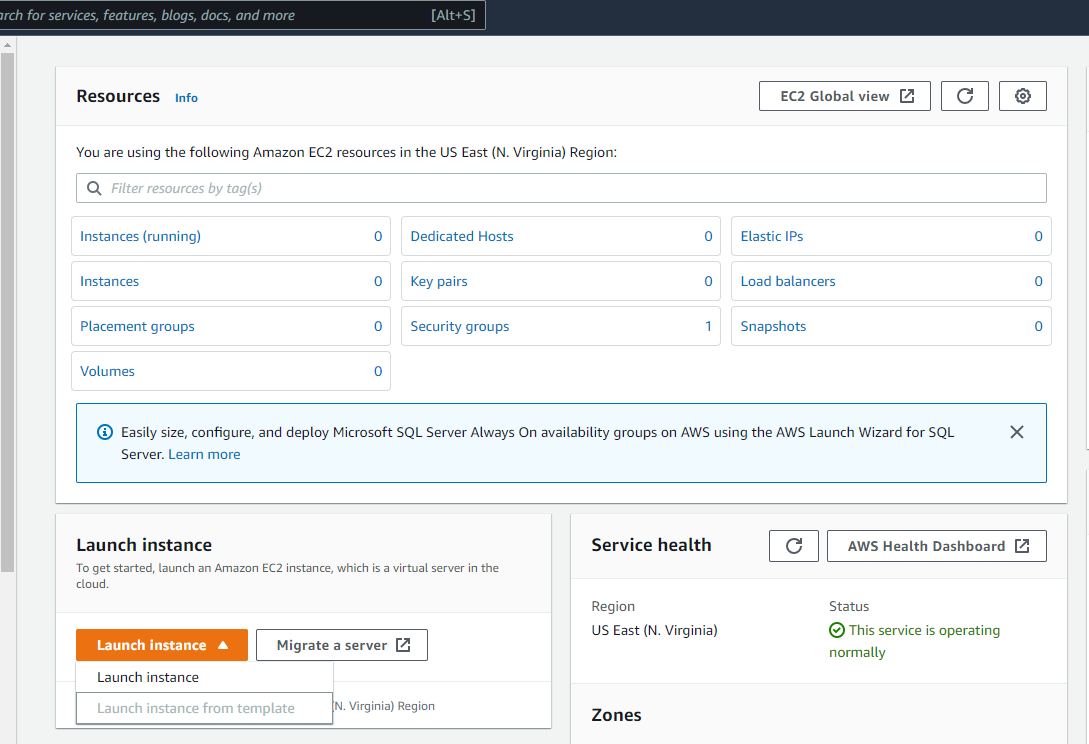
****

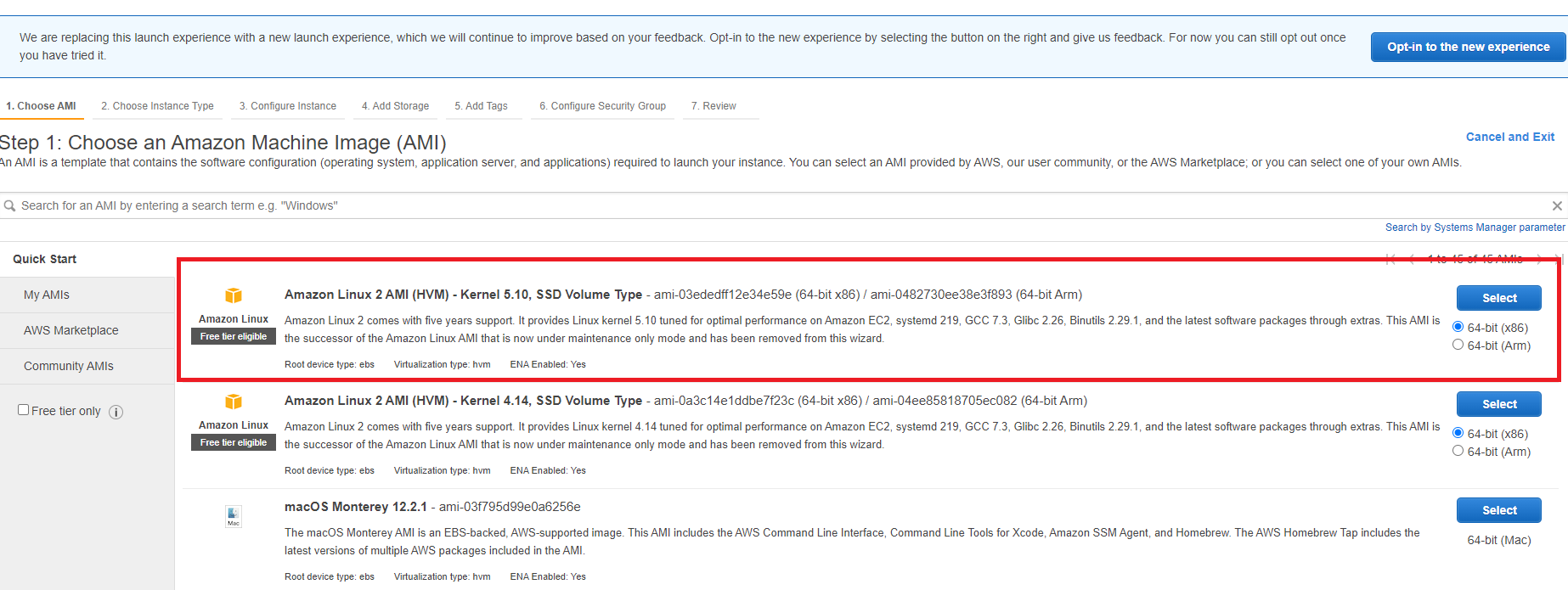
****

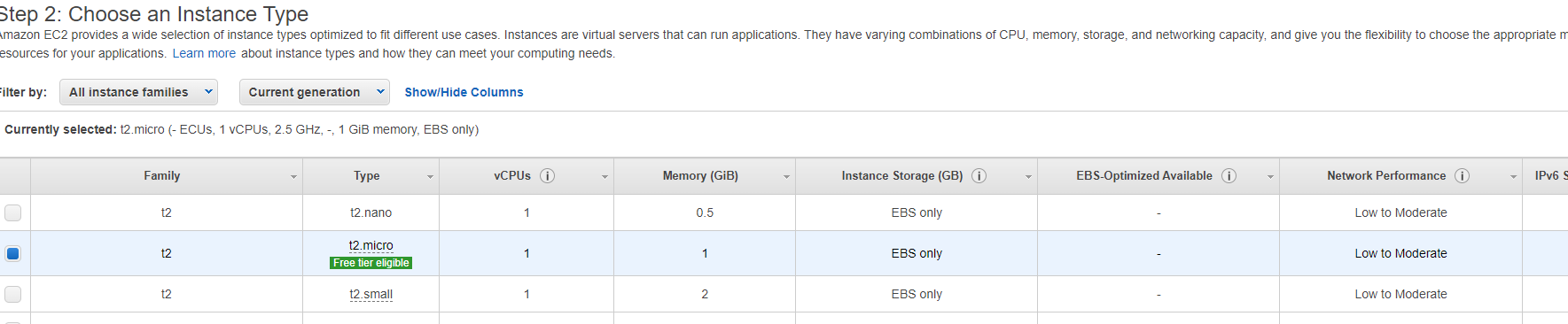
* Jenkins testNG sample output will list out total no of passed and failed testcases,

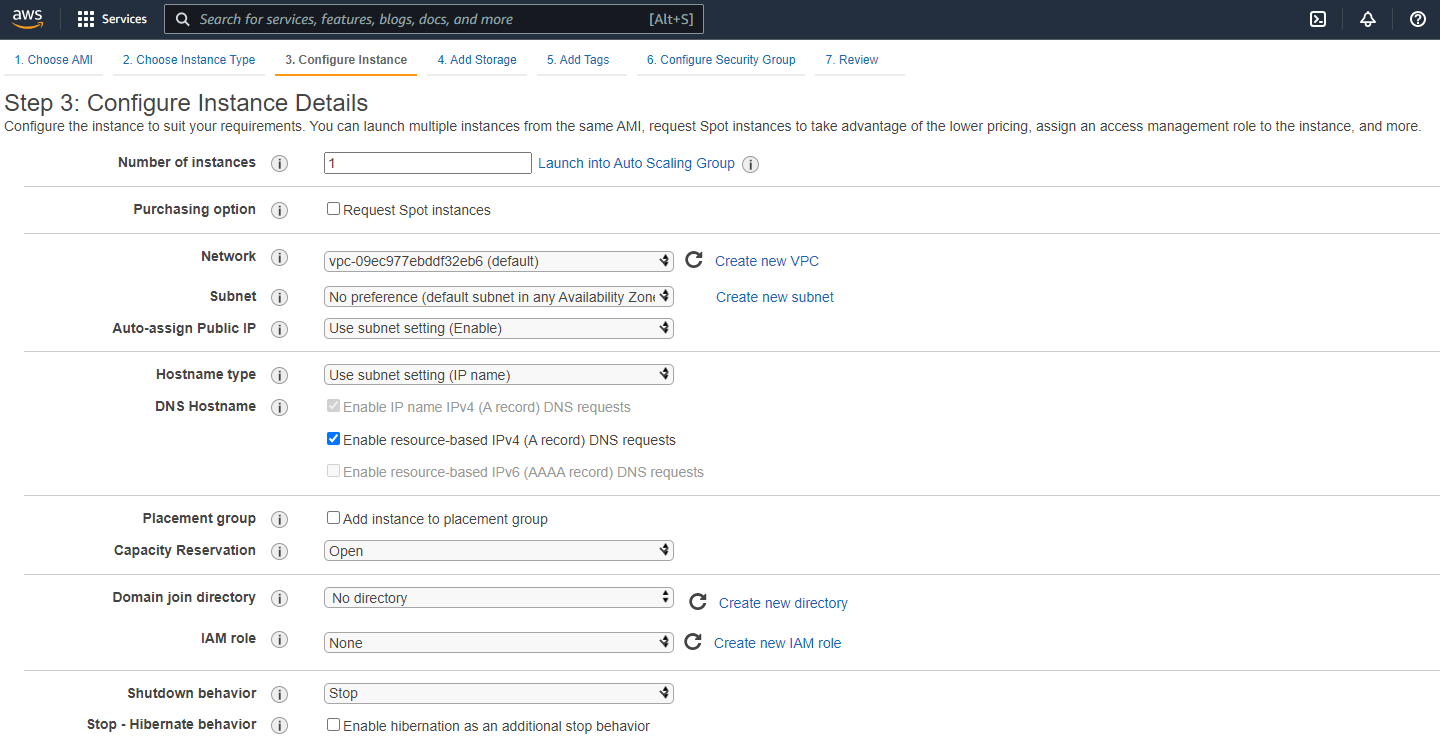
****

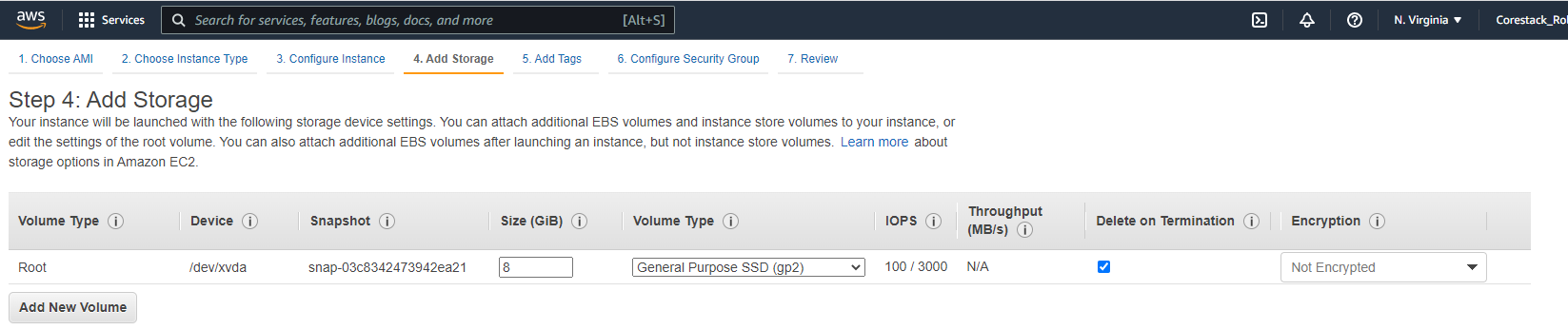
* Create AWS EC2 instance with launch instance option to host the web application,



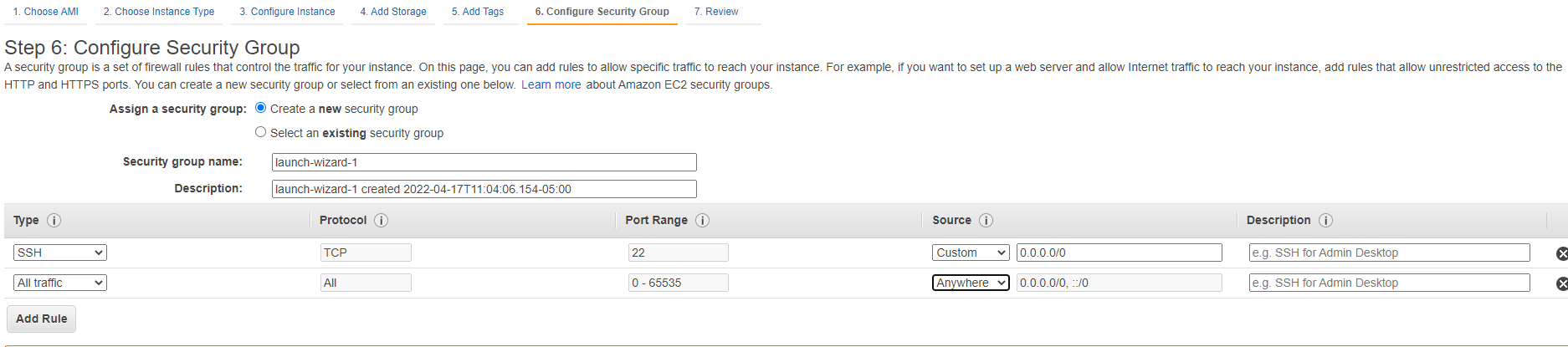




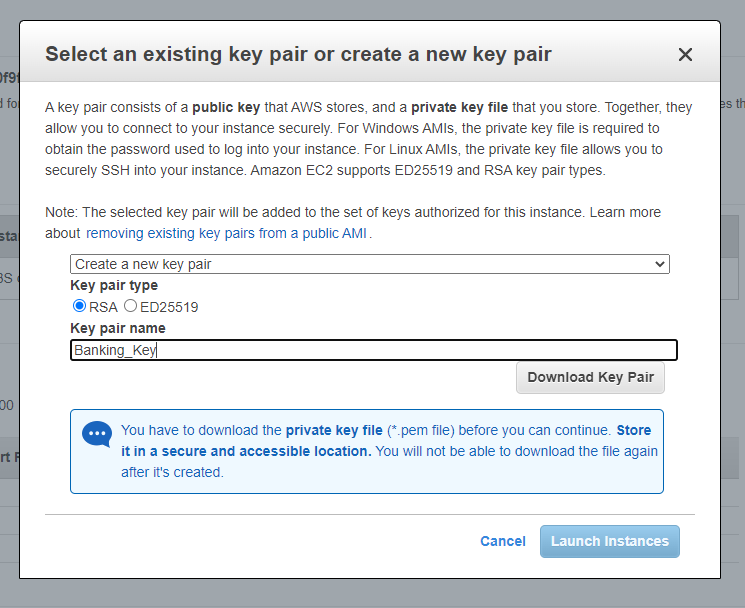




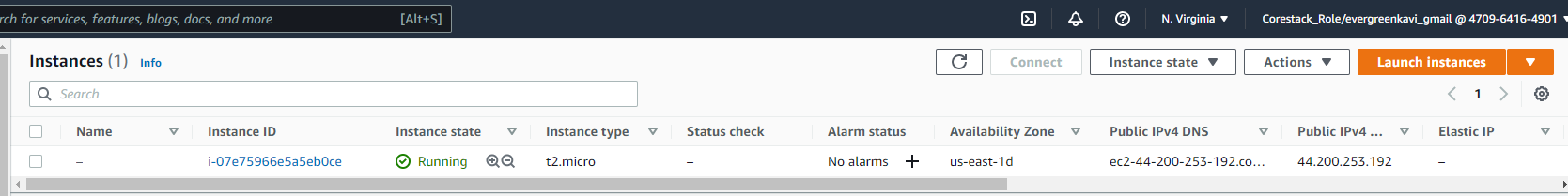
* Add Rule to allow all traffic access,



* Create new key and download the key,

****

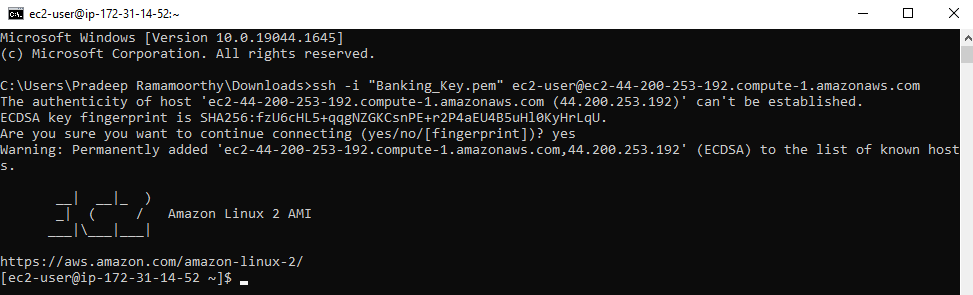
* AWS EC2 instance is running now,

****

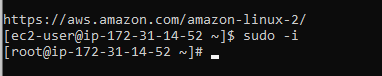
* Open the command prompt from the folder where key is located,

****

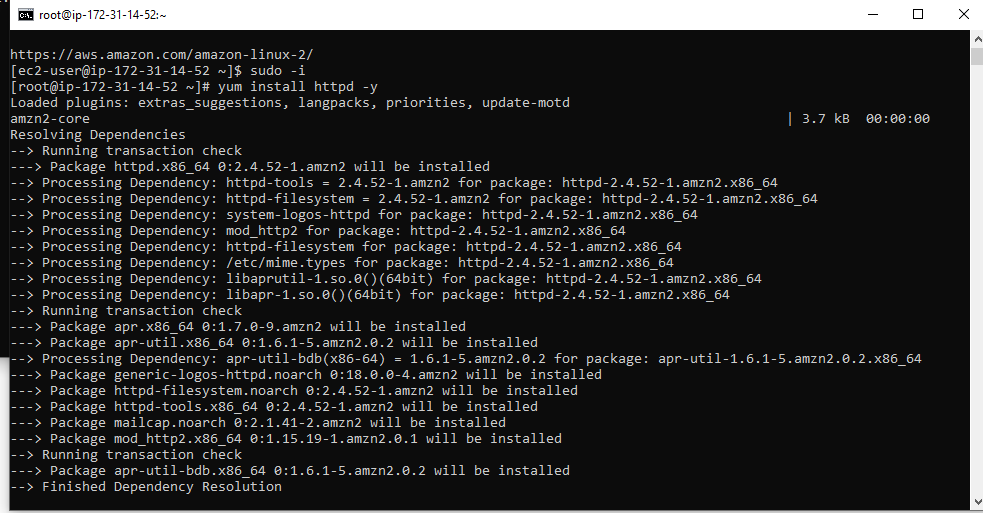
* Enter the SSH command with key as shown below to get into the instance,

****

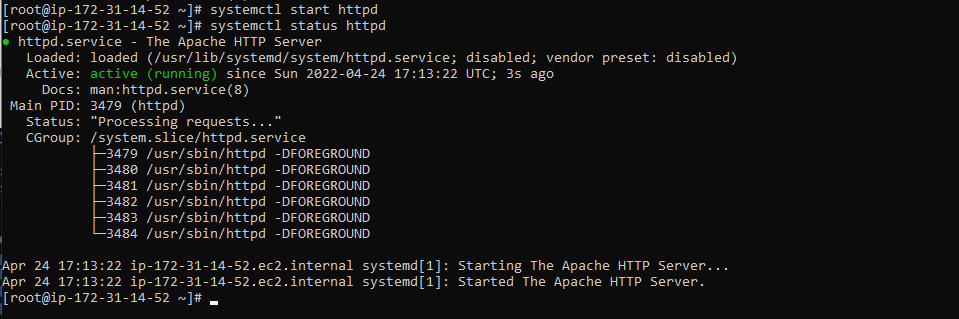
* Enter as root user,

****

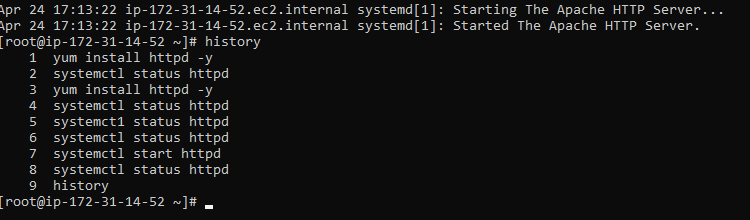
* Install httpd and check the status,

****

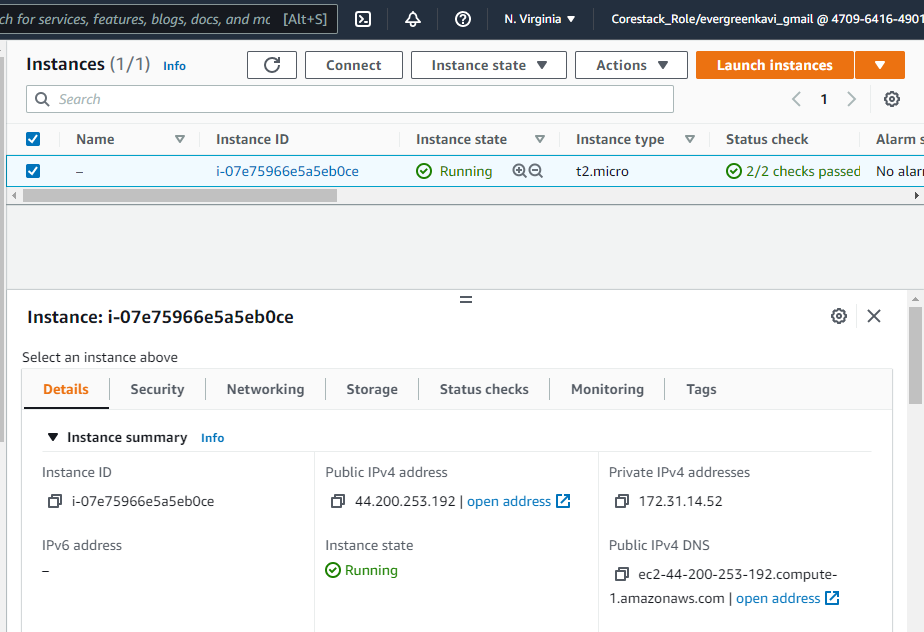
* Start the httpd and check the status to see if it is active and running,

****

* History of used commands,

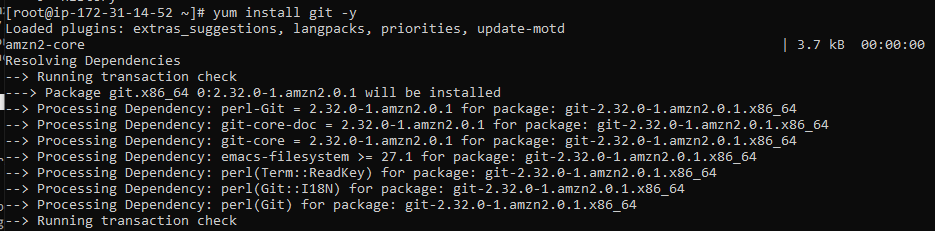
****

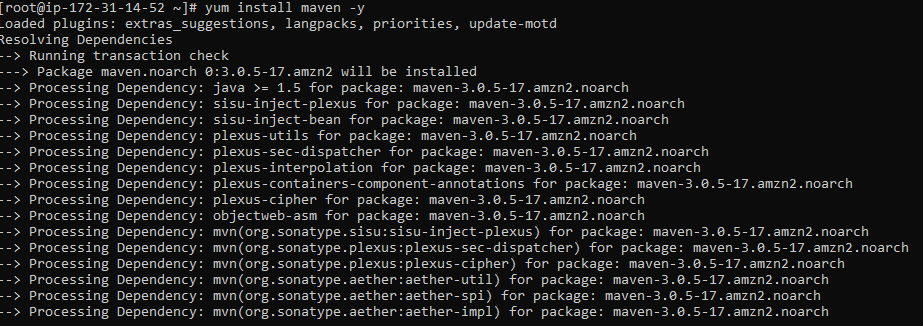
* Get the public IP address and check if gets access to test page,

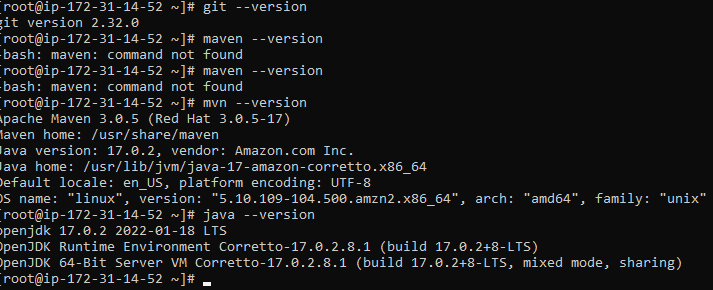
****

****

* Install git & maven and check the version,

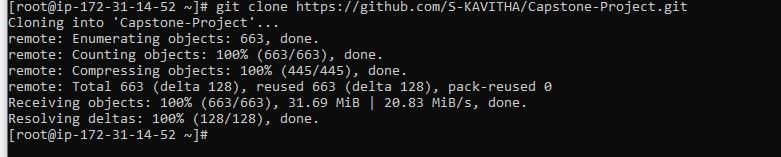
****



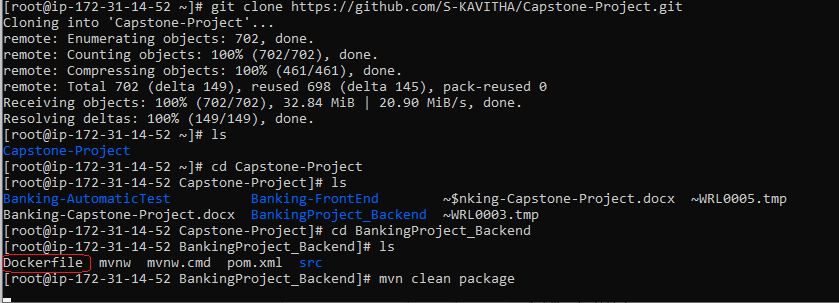


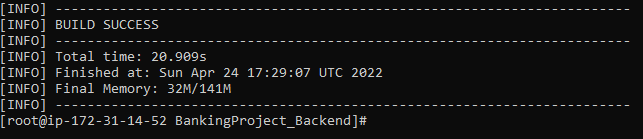
* Clone the git repository as shown below,

git clone https://github.com/S-KAVITHA/Capstone-BankingWeb-app.git

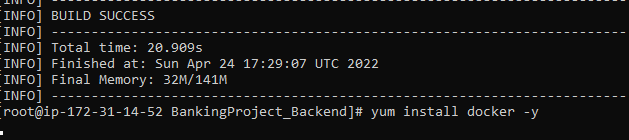


* Clean package the project,

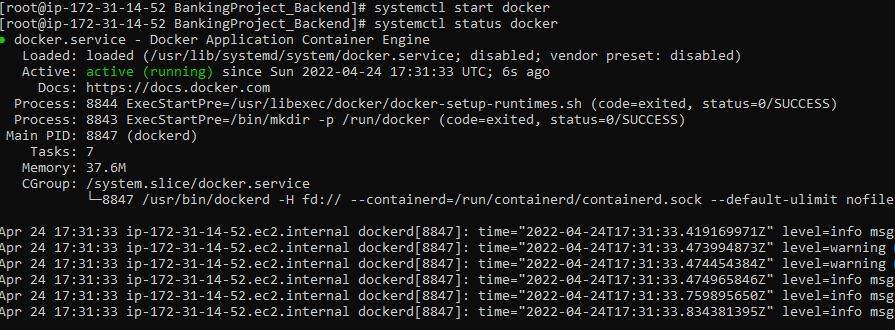




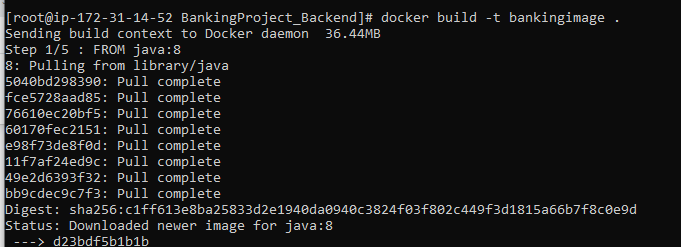
* Install docker and start docker,



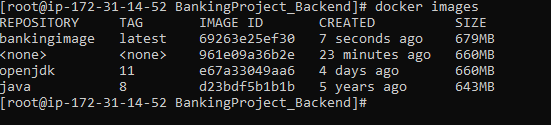
* Check the status of docker, if it is active and running,



* Build the docker image,



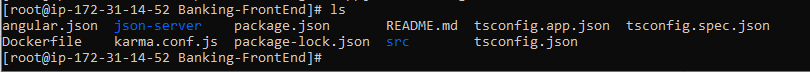
* Check the docker image,



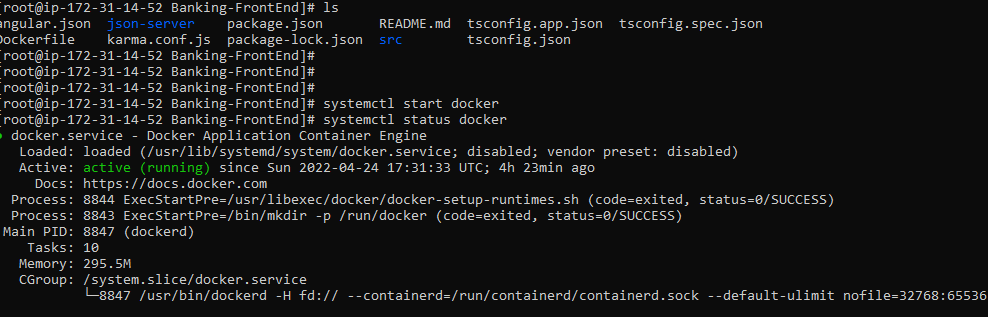
* Run the docker image,



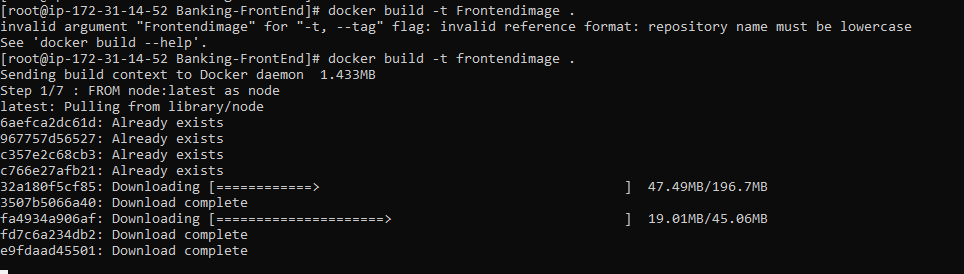
* Create and run the docker frontend docker image to access front end banking screen,

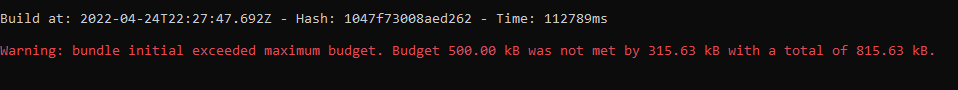


* Check the status of docker, if it is active and running,

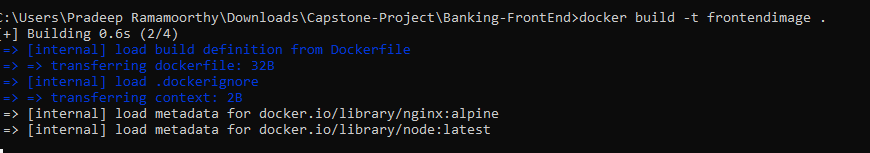


* Docker image build was unsuccessful. not able to create image in AWS EC2 instance due to size restriction,

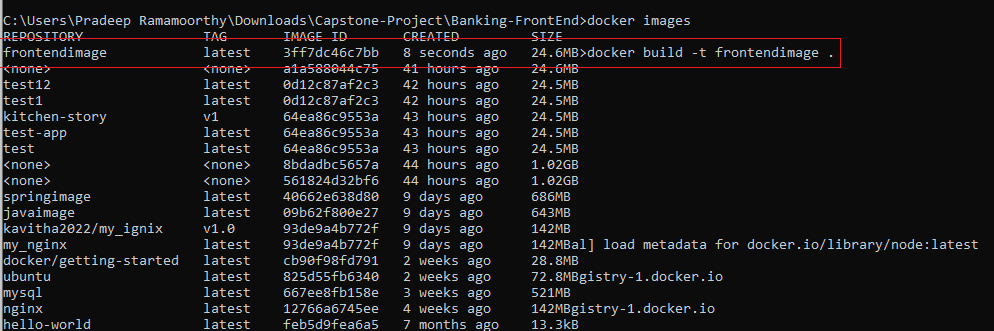




* Local host docker hosting for front end angular application. Build image using below docker command,



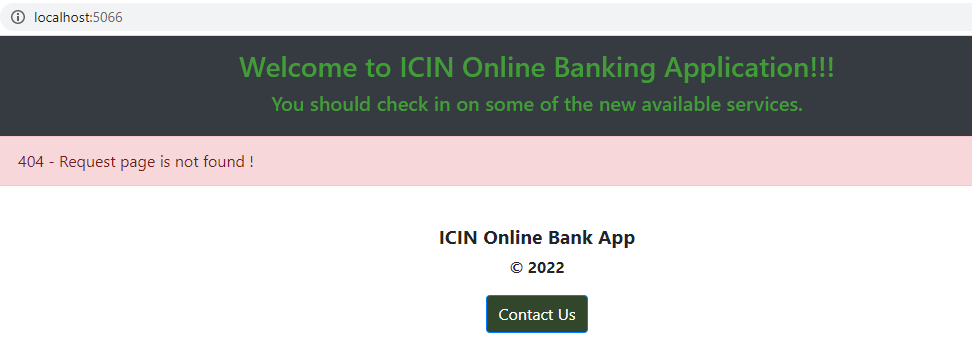
* Check the docker image,



* Run the docker image,



* Access the front-end website from docker hub 5066 port,



## **Pushing the code to GitHub repository**

* Open your command prompt and navigate to the folder where you have created your files.

**cd <folder path>**

* Initialize repository using the following command:

**git init**

* Add all the files to your git repository using the following command:

**git add .**

* Commit the changes using the following command:

**git commit -m <commit message>**

* Push the files to the folder you initially created using the following command:

**git push -u origin main**

**Source code for the project available under below repository,**

<https://github.com/S-KAVITHA/Capstone-BankingWeb-app.git>