System Guide

Group 3

Commerce Bank App

Tayler Singleton, Reed Heinrich, Connor Larson,
Samuel Feye, Landon Volkmann

Change History

Version: <1.0>
Modifier: <Tayler Singleton>
Date: 04/26/2020
Description of Change: <System Guide was Completed>

1 Introduction

The Commerce Bank application allows users to track their transactions using their online banking account. This document will provide instructions for accessing the application code and installing this web application.

2 Access React-Redux Application Code

Access to the internet and the semester-project-group-3 revision control repository used to store the code is required.

All the code for the React-Redux application is found in the SVN repository located at:
https://github.com/umkc-cs-451-2020-spring/semester-project-group-3

3 System Installation

3.1 Required Components

Access to a computer with Visual Studio Code, React, Redux, and Node.js is required in order to start and build the application.
3.2 Install Code
Launch Visual Studio Code

1. Access the code in the repository.
    a. Create a directory on the local file system
    b. Select Clone "Clone or download" on Github, copy the provided link
    c. In Visual Studio Code, select File -> Add Folder to Workspace -> Select newly-created directory.
    d. Open VS Code Terminal
    e. Clone a Git repository with Git: Clone command in the Command Palette(Ctrl+Shift+P)
    f. Paste the link into the Git: Clone prompt
    g. Open the folder just cloned
    h. Open Visual Studio Code Terminal
2. Node.js
    a. Go to https://nodejs.org/en/download/
    b. Download the LTS Verison(includes npm) using the installer
    c. Check the version of node and npm

        i. Using node -v and npm -v in the Visual Studio Code Terminal

3. In the project directory, you can run:
    a. npm install
        i. The command installs a package and any packages that it depends on
    b. npm start
        i. Runs the app in the development mode.
        ii. The page will reload if you make edits.
        iii. You will also see any lint errors in the console.
    c. npm test
        i. Launches the test runner in the interactive watch mode.
    d. npm run build
        i. Builds the app for production to the build folder.
        ii. It correctly bundles React in production mode and optimizes the build for the best
        iii. performance.
        iv. The build is minified and the filenames include the hashes.
        v. Your app is ready to be deployed!.
    e. npm run eject
        i. Note: this is a one-way operation. Once you eject, you can’t go back!
        ii. If you aren’t satisfied with the build tool and configuration choices, you can eject at any time. This command will remove the single build dependency from your project.
        iii. Instead, it will copy all the configuration files and the transitive dependencies (webpack, Babel, ESLint, etc) right into your project so you have full control over them. All of the commands except eject will still work, but they will point to the copied scripts so you can tweak them. At this point, you’re on your own.
        iv. You don’t have to ever use the eject. The curated feature set is suitable for small and middle deployments, and you shouldn’t feel obligated to use this feature. However, we understand that this tool wouldn’t be useful if you couldn’t customize it when you are ready for it.
4. React-Backend
    a. Run npm install to install the necessary packages for the project
    b. Run npm start to final connect the frontend and backend

4 System Maintenance

4.1 Node.js and NPM

4.1.1 Node.js

Node.js is the Javascript runtime built on the V8 Javascript engine. It allows developers to run the Javascript code on the server/computer. Gives developers the ability to use the same language on the frontend and backend. It is asynchronous and works on a single thread using non-blocking I/O calls. Node's event loop uses events and callbacks to support concurrency. Node modules can be core modules, modules installed via NPM, and custom modules which are files that have an export. Communication with the database for the application is being done with Node.js and the creation of the server the application is built on is done in the Node.js environment.

4.1.2 NPM

Stands for Node Package Manager. NPM commands are used to install third-party packages, which are then stored in the node_modules folder. All dependencies, dev dependencies, npm scripts, and information about the project are listed in the package.json file. There are two package.json files one is for the frontend and the other is for the backend. npm install is a necessary CLI command to run in the terminal as if the necessary dependencies have not been downloaded and stored in the local_module folder the application will not run properly.

4.2 Front-End

4.2.1 Front-End GUI/Forms

The user interface consists of three tabs: Dashboard, Transactions, and Notification Settings. Dashboard is a daily snapshot of the user's notifications and transactions. Transaction gives the summary of all the user's transactions, lets the user add transactions, and sorts transactions by date. Notification Settings gives the user the ability to Add/Delete/Edit notifications triggered by their transactions. These notifications are separated by category: transactions, balance, description, recurring alert. The user interface also includes a login screen, sign up screen, and forgot password screen.

In the pages folder, the component for rendering the tabs is App.js, and the component for rendering the login screen, sign up button/screen, and forgot password button/screen found in the LoginPage.js.

Once the user goes to the application, the application displays the login page. When the user inputs the correct email and password they are redirected to the main application page. The first tab to the user is introduced to is the Dashboard tab. Alongside the application they're the three tabs the user can navigate to Dashboard, Transactions, and Notification Settings.

4.2.2 React

Is a Javascript library used to develop user interfaces. The scr folder contents the application of React. The subfolders components and pages contain the UI components, which are components that do not contain a state, receive its data from props, are only concerned with the user interface, and functions are used to create these components. The files in the pages folder App.js and LoginPage,js are used to render the files in the component folder.

4.2.3 Redux

A predictable state container for Javascript apps. This library is used to maintain the state of the entire application in a single immutable object. The usage of this library is to take user interactions with the application to login in the user or tell the user the login information is incorrect, fetch transaction and notifications information from the database to display to the user, and correctly render change screen when the user interaction in the tabs or screens implemented in the LoginPage.js.

4.2.3.1 Actions

These are events sending data to the store. The data can come from user interactions, API calls, and form submissions. The store can only get information from the actions. Internal actions are considered simple Javascript objects that have the type property. which describes the action being done and the information to be sent to the store. Actions are created by the action creator and are just functions to return actions. To call the actions the dispatch method is needed.

4.2.3.2 Reducers

Pure functions used to take the current state of the application, an action, and final return the new state of the application. Reducers within our application are used to change to the state of the login page, prompt to the frontend if the fetching transactions and notification information was a success or a failure, and correctly change the state of which component is to be rendered to the client.

4.2.3.3 Store

It is an object used to hold the application state and has several helper methods to access the state of the application, dispatching of actions, and register listeners.

4.2.4 Material-UI

An open-source project that uses the implementation of Google's Material Design as React Components. This open-source project is being used to display forms and navigation between the different tabs.

4.3 Database and Backend

Within the bin folder, the www file holds the implementation of the HTTP server, the port number for the Express environment store, and listeners for the Port and HTTP server. This file is stored as a configuration in the package.json in the react-backend folder and using npm start command in the terminal will run the server to be connected to the frontend. Sql_scripts folder contain the subfolder stored_procedures one of the files is used the create the notification trigger and the other creates the transactions that will be displayed in the Dashboard and Transactions tabs.The other .sql files contained in sql_scripts are used to create the database, drop each table in the database and load the data that is to be displayed to the client. The config.js contains the configurations for the database and the app.js initializes all the routes created in the routes folder and does error handling.

4.3.1 Express JS Routers

Express is a Node.js web application framework used to provide features for a web or mobile application. The routes folder is used to take requests from the frontend and send back data from the MYSQL database The way each route takes requests is by setting up an express variable then creating a router variable assigned to express.Router(). This router variable now has HTTP methods. The .get() method of each router responses to each get request. The function inside the .get() method has three parameters: req: request object , res: response, and next: middleware object, which passes control to the next matching route. It then connects to the database by using the mysql variable, which has been assigned to the third-party MYSQL module. The mysql variable is used to query information from the database and the router variable then uses the .send() to send the queried information to the correct component to be rendered in the frontend.