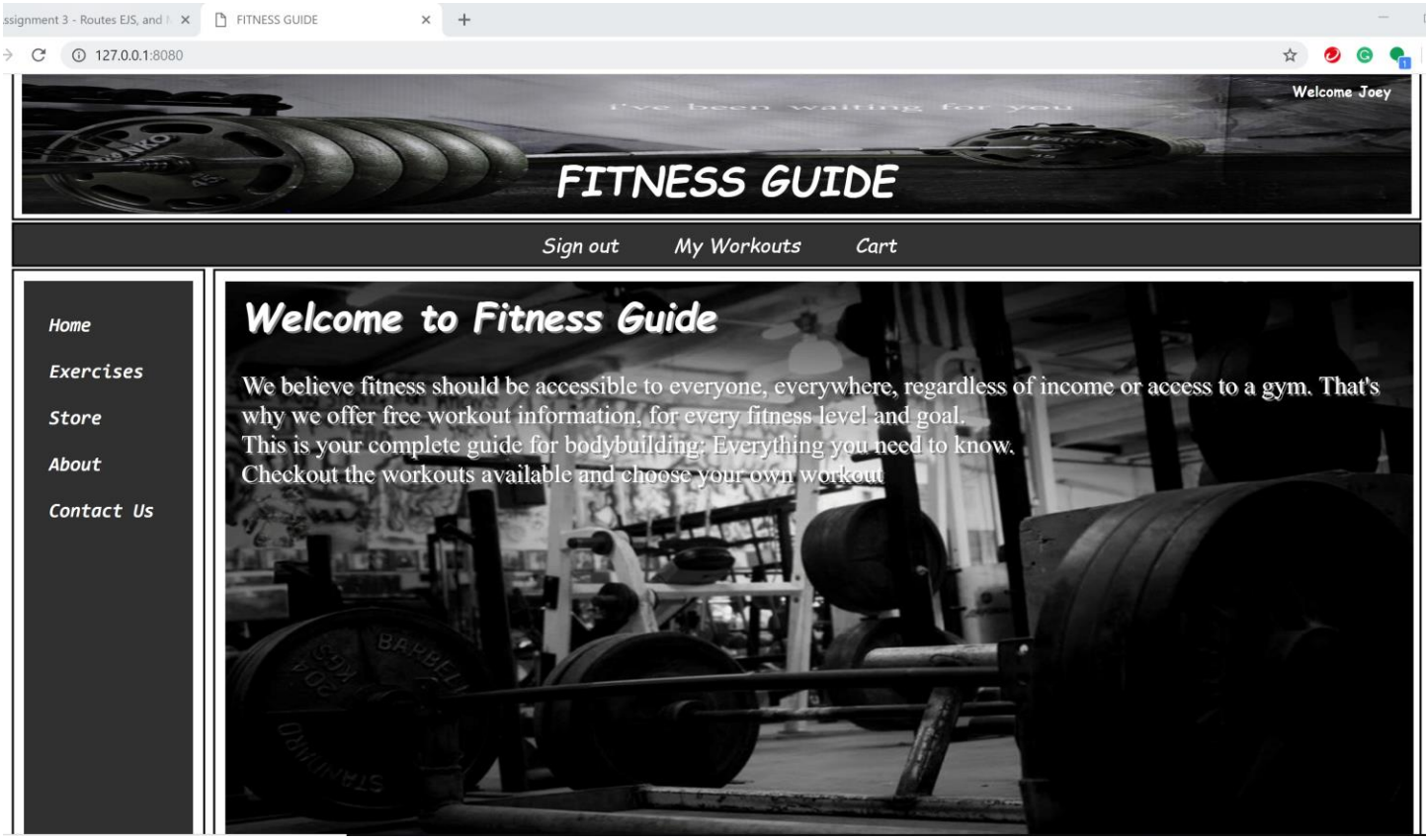
Assignment 5 - Routes, EJS, and MVC - Secure Application

Website screenshots:

Home Page of Website



- The website is a fitness guide that provides body workouts. Each body workout contains three exercises with their images.
- The website is using Model View Controller Framework.
- The website is using post requests to send data from one page to another page. When user signIn the data is stored in session.
- For security purpose I have created itemList in ejs folder which includes itemCodes of all the items currently visible in view. In controller I added validateSecurity() function which checks if the itemCode of item that it intended to modify is present in itemList sent from ejs file.
- Checked the input for get and post with express-validator.
- Login functionality for user

  User 1:
  Username - joey.smith@gmail.com
  Password – password1

  User 2:
  Username - chandler.bing@yahoo.com
  Password – password2

- I have added one item in user's bin by default for the first time user login.

- The user data is stored in MongoDB database named fitness
  Below are the collections in database
    1. Item
    2. User
    3. categories
    4. UserItems
- Files and Folder Information
    1. Assets → This directory includes images and stylesheet
    2. Controller →
       Catalog controller == This directory includes catalogue controller that handles all the routing of the website. The catalogue controllers also redirects the website to categories page if user enters wrong itemCode in the Url.
       Profile Controller == This controller is responsible for answering requests that pertain to user-specific functionality. These are the requests for accessing a service the application provides to support its users.
    3. Models → This directory includes two files
        - Item.js = It defines Item model and category model. The file includes getImageURL() and getCategoryURL() function that generates the url for specific item received from ItemDB database. The file contains item data schema for database
        - ItemList.js = This file include models that return list of items and categories.
        - User.js = This file includes userModel taking user details. The file contains user data schema for database
        - UserItem.js = This file includes userItemModel that takes item, rating and madeIt flag as its input
        - userProfile.js == This file includes userId and its item List. It also content functions to modify user item list. The file contains user profile schema for database
    4. Utility →
       itemDB == ItemDB contains mongoDB database connection to fetch and insert data for Items. It also includes getItem() and getItems() function to return specific item and list of items.
       UserDB == userDB contains mongoDB database connection to fetch and insert data for Users. UserDB consist list of two users and getUsers() function to return list of users. getUser() fetch specific user from database.
       UserItemDB == UserItemDB contains addItemRating() and addMadeIt() function to update item rating and flag in mongoDb database
    5. Views → It includes .ejs files that contain UI part of website. It includes subdirectory named navigation that contains 4 common navigation ejs files

- The website is designed as per the given instructions.
- The website is working as expected without any issues and stopping points.

Easy Parts

1. The login functionality was easier as similar to previous tasks.
2. Express validator library made easier to validate inputs

Challenging Parts

1. It was challenging to find various methods of express validator to validate input