Project: Online TODO List

The TODO list is an application to track the day to day tasks of users. It is a simple UI which supports below functionalities:

- Registration of new users.
- Login for existing users.
- Validation and hence present the message when incorrect information is provided.
- Add Task and Remove Task.
- Supports the checking/unchecking of tasks.
- Users can view their existing tasks after login.
- It will show the last updated time for every task.
- The user's information gets stored in an in-memory H2 database (*UserInfo and TaskInfo tables*).

Below Technologies (Approaches) are Used

Spring Framework: It is a lightweight and powerful framework that supports web development in an easier and safer way. It provides an advantage over the application developed using Java EE standards where code becomes more complex when new functionalities are added and also degrade the performance due to bulkiness of application.

The content inside the configuration files is less as most of them are taken care of by the annotations in spring MVC. It supports modularity where code can be divided into various modules performing individual tasks. Servlets can also be used to write the application but we need to handle bindings, JSON body, Validation etc. manually, hence spring framework is preferred.

Spring Data JPA: In order to access the data from a relational database (here H2), we used Spring Data JPA which has a concept of JPA repositories (set of interfaces which defines query methods). On using JDBC we need to provide native queries to interact with databases while in Spring Data JPA no need to write queries. Hence Spring Data JPA is preferred to raw JDBC code.

H2 Database: It is an open-source lightweight in-memory database that is used to achieve minimal response time as the data doesn't persist on disk or SSD. As compared to other RDBMS, the query execution is fast and can be easily configured using spring framework. The Two tables (*UserInfo and TaskInfo)* were created to store the information about the user and user's task. Later these user details can be retrieved easily once login.

Spring MVC Test framework and JUnits: It used to write the unit test cases for controller methods (perform CRUD operations).

Lombok Java Library: It provides an environment to write less repetitive code using annotation processing (eg: no need to write getter and setters).

CSS and HTML: To structure the web page and provide document style (layout, colors and fonts).

AJAX JavaScript: It is used at the client-side to load data in the background and displays it on a webpage. The jQuery library has been used for HTML/DOM manipulation, CSS manipulation, event methods, and AJAX call.

Maven: Project management automation tool used to include the dependency, build and manage projects.

Design Patterns

1. ***Model View Controller Pattern***: The TODO application consists of presentation data, models and control information separately. It is important to note that the Model part contains no logic explaining how data is presented to the user. The controller is responsible to accept a request from user and send it to view for presentation.

2. ***Behavioral patterns - Template Design Pattern***: Defines steps required for any action (boilerplate code) and re-used the methods when needed (like query execution).

3. ***Front Controller Pattern***: Supports centralized request handling mechanism. A front controller can also use helpers to achieve the dispatching mechanism.

Unit Tests: JUnit and Mockito

The automated unit test cases have been written and executed for user registration, user login, task creation, task deletion and various authentication validations. We can find all the results in the logs during the maven project build. In case of any failure in these tests, the war file will not be created. Along with JUnits, Mockito has been used to mock the dummy implementation of the service. Mockito is used to execute and validate the code under test.

Enhancement (what more can be added)

1. To allow secure communication between the client and server we can use JSON web tokens. Once the user is logged in, each subsequent request will include the JWT, allowing the user to access routes, services, and resources that are permitted with that token.

2. HTTPS can be implemented to encrypt the user's sensitive data and hence ensure data security over the network during communication.

3. Load balancing can be implemented in to reduce the traffic on the backend server. It will distribute server load across multiple resources which will reduce the response time and increase throughput.

4. To minimize coupling and reduce complexity we can perform Modularity where we can split a monolithic codebase into multiple modules. So the functionality can be tested in isolation and it will increase efficiency.

5. Caching can be implemented to reduce the number of executions based on the information present in the cache, for example, configuration data can be cached, and also spring framework provides annotation based caching declaration and configuration.

6. In order to reduce the fetching time, the Elastic Search can be implemented which will act as secondary storage to index the data from the H2 database (primary storage).

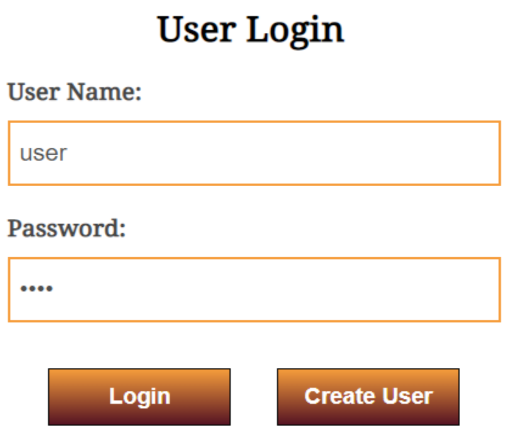
7. We can also give an option to edit the task on the UI followed by saving data in the database. Few more validations could be added like: string length limit etc.

Source code and War file: All the client-side, server-side and war files can be found on the Github. **https://github.com/guptahimDub/OnlineJavaTest**

Web Application Images

(http://localhost:8080/onlineTodoList/)

'User Login' for existing User and **'Create User'** for new users



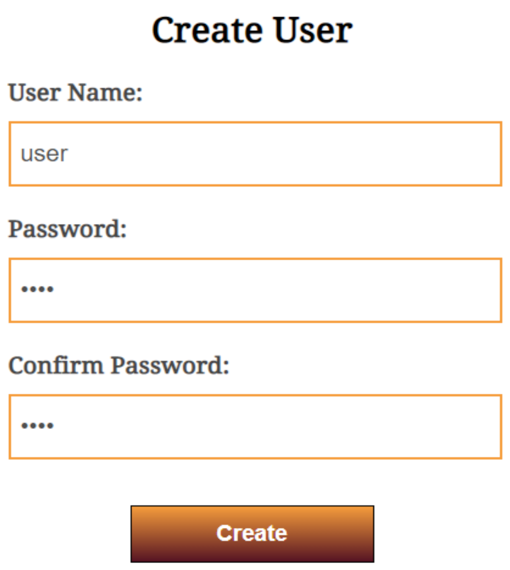
User Login

User Name:

user

Password:

••••



Create User

User Name:

user

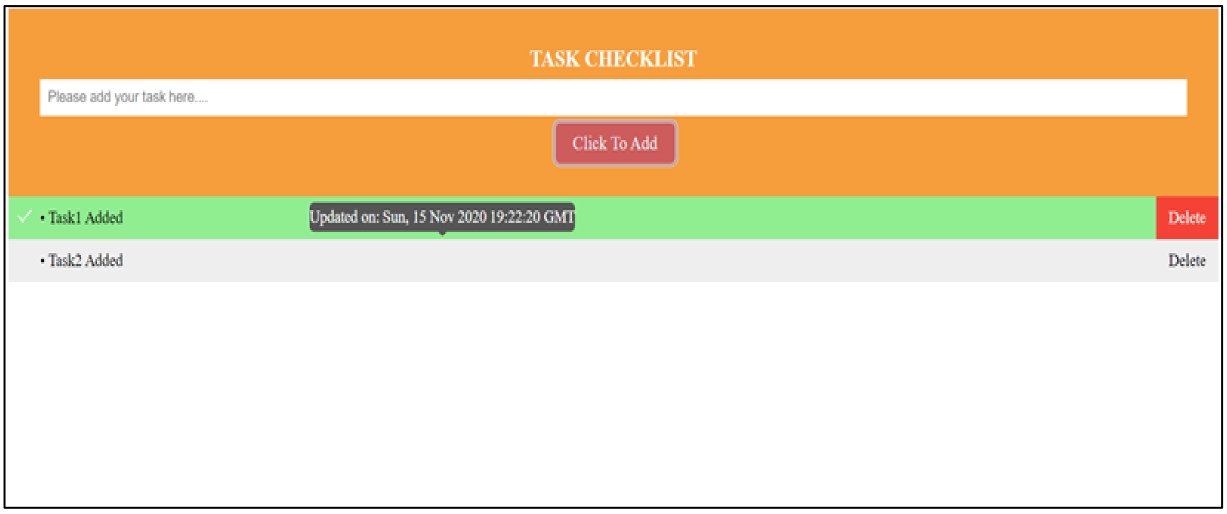
Password:

••••

Confirm Password:

••••

Final Application Display representing tasks and operations



TASK CHECKLIST

Please add your task here....

✓ • Task1 Added — Updated on: Sun, 15 Nov 2020 19:22:20 GMT — Delete

• Task2 Added — Delete