ToDo Application

Author: Sagar Parmar

Email: sagarparmar4@gmail.com

GitHub Link: https://github.com/sagarparmar4/todo-app

Current functionalities

1. User can sign in using unique login and password securely.
   a. Login page is yet to be integrated with backend.
   b. Currently Basic Authentication is implemented with default username as 'user' and password as 'pass@123'.
2. On successful login, user can view their task list along with its created and last modified date, description and status.
3. User also has the ability to logout of the application.
4. User can add or remove task. User can also check or uncheck any task on their list.
5. All changes are saved to view it during next sign in by the same user.

Project Outputs

1. **Why certain approaches were used, why others were not selected?**
   a. A combination of AOP and OOP driven approach was used to develop the project. Purpose of this was mainly to break down everything dynamically so that is gets easier to modify code in future.
   b. It also helps to implement new features with less refactoring required.
   c. Helper methods and classes were created to maintain consistency.
   d. An agile model was used to design and develop the project with milestones being: Entity designing, Basic business logic, UI designing, Integration. At end of each milestone, implemented components were tested to ensure application stability.
   e. Following were the project stages:
      i. Deciding project architecture (Project Structure, Entities and their properties, etc.).
      ii. Developing and testing backend.
      iii. Experimenting and finalizing UI design.
      iv. Developing UI and integrating it with backend.
      v. Final testing, error resolution and generating final build.
2. **Any design patterns used?**
   a. Frontend was designed to be a sign-page application to improve UX.
   b. Web MVC (Model-View-Controller) design pattern was used.
   c. Using Web MVC, its easier to logically break down application into different components.

d. It also helps to scale up and maintain things easily without adding much complexity.

3. **Anything extra could be done if more time was allocated?**

a. Change security configuration use more secure method instead of Basic Authentication (for example, use Web Tokens, or use RSA encrypt HTTP body).

b. Add functionality to register new users and also validate then using email.

c. Improve frontend responsiveness and cross browser support. Add responsive design support for mobile devices

d. Provide option to add deadlines for their task and also to notify users via email when deadline is near or missed

e. Add option to filter tasks based on status or assign then to user defined categories

f. Fine tune JSON responses using JSON views and also improve security (for e.g., hide username and password from JSON responses)

4. **Anything else to know about?**

a. Application was tested using Google Chrome v100+ and according to Google Lighthouse, the application responds to user actions within 10 to 20 milliseconds

b. At some places application contains partially implemented features. TODO statements are written code to highlight partially implemented components

c. Toastr was implemented to show non-blocking notifications on certain user actions

d. Concepts or Packages used

    i. Backend: Spring Boot 2.7, Spring Security, Spring JPA, Spring Web, Junit, H2 database

    ii. Frontend: Angular 8, Routing, RxJS, HTTP Client, NgxToastr

    iii. Others: REST APIs, MVC driven approach, AOP architecture

Project Snapshots: Login Page & Dashboard