# TODO List Documentation

# Technology Used

Spring Boot Application with Spring MVC

Database = H2 Embedded in application

Tomcat = Embedded in Application

Default Port = 8080 (can be changed by command line or in configs)

Front End = Thymeleaf Template(HTML, BootStrap)

Backend   = Java

Java 8

Spring Data

Spring Security

Maven

Unit Test not implemented but configured in application

Spring User Authentication Form

# Application Architecture

## Front End

Web Browser -----🡪 Spring MVC Controller -----🡪 Thymeleaf Template -----🡪 Web Browser

## Back End

Controller--------🡪 Interface Service-----🡪Service------🡪Repository--------🡪Hibernate------🡪Database

# Functionalities Provided

1 User Can login, a login form is provided at start.

2 Ge a TODO List for the user who is logged in. ( Every user can view its own TODO List )

3 Add a new TODO task by clicking the add new button.

4 Update a task by clicking the update button and user can use check box if its finished.

5 Delete a Task by clicking the delete button in the list table.

6 Search is provided to search it by description only. In background we make sure search work by logged in user.

7 Logout button is provided as well.

8 When user update a task updated time is recorded and shown in the list.

9 Checkbox for task is finished is only clickable inside edit form and is read-only in the table.

10 Spring Security is configured with some dummy users and their dummy roles and some of them with access to certain pages.

11 Dynamic page loading and persistent data is provided.

Application can be tested with following user.

Login Password Role

1 test  = pwd123 ADMIN

2 test1 = test123 MANAGER

3 sa = admin MANAGER

4 test2 = test123 MANAGER

Database login detail

Login =  sa, Password  = admin

How to access database

localhost:8080/h2   h2 have no restrictions at this moment.

Localhost:8080 will take you to login form and after entering details you it will take you main list page.

# Some Suggestions and Details for Project Improvement

Spring boot applications are becoming the norms now a days as they provide a starter pack where we can easy configure and setup the spring environment rather than setting the up the spring frame work old ways where you have to configure lots of xml files before you can even print the “Hello World”

There are lots of package dependencies available for Gradle and Maven Spring Boot Applications.

I used Thymeleaf instead of JSP because first of all its new and I always try to learn new technologies and it was the opportunity to use it.

Thymeleaf is same as using JSP and you can access java code, java objects, spring beans and it include dynamic content delivery as well.

Thymeleaf templates are done in html and you can use bootstrap and jquery or JavaScript.

I was going to integrate jquery in this application but because of time constraint I used local event handlers to call the controller.

There are lots of room to improve in the application, first of all we can start with creating proper user login table and create secure password and save it securely.

Provide CSS or design in the forms as currently they are very simplistic.

We can have proper relationship with user login table and TODO list as one user can have many tasks and user can have access to its own tasks and we cannot show others.

That part is covered by creating a user in the table and then checking the logged in user and then query the data according to the user logged in.

There could be a data load at start according to the time a task is created and updated and we can do in descending order.

I have not implemented the Unit Test due to time constraint but Junit is configured in the application.

We need to create test for domain layer for the entities so that to check we have constructors, id etc.

We can also create unit test for service layer to test the methods which we are calling are working as they should be.

All the business logic can be tested with unit test.

For development I would like to use domain driven design which I am trying to implement which require more work but is more reliable.