# TODO List Web Application

|  |  |
| --- | --- |
| Author | Vinay J Yadave |
| Document Create Date | 05-03-2021 |
| Technologies Used | Java,Spring Boot,Spring MVC,Junit,Maven,JSP,CSS,Tomcat Server |

# Approach and Design Pattern

To develop Todo list web application I have used the Spring Boot open-source framework. We can say spring is a framework of frameworks. Spring framework provides a comprehensive infrastructure support for developing Java applications.

# Why Spring Boot?

The problems with spring frameworks are it is a huge framework, requires multiple steps and also multiple build and deploy. Whereas spring boot is more of convention over configuration. Moreover, its production ready and standalone. Spring boot lets you bootstrap a spring application from the scratch. Spring boot makes it easy to create standalone, production-grade spring-based applications that you can “just run”.

# MVC Architecture

Spring MVC is one of the popular and best approaches for developing web-based applications.MVC stands for Model, View and Controller and separates all these layers. Basically, it provides loose coupling between the layers. In Spring MVC, the DispatcherServlet acts as a front controller. It is responsible to manage the flow of the spring   MVC application. I have used MVC design pattern to build the web application.

# Spring Data JPA

* To interact with the database I have used Spring data JPA (Java Persistence API).It provides JPA template class to integrate spring application with JPA.
* It gives abstract repositories that are implemented at run time by spring container and perform CRUD operations. As a developer we have to provide the abstract methods in the interface.
* The biggest advantage is Spring data JPA reduces the amount of boiler plate code required to write data access layer.

# Views and Database

In this Todo list project I have used H2 in memory database. The DB consists of two tables *Users* and *Tasks* table to hold the user and task data, respectively. H2 database makes it handy for the developers to test the application. For the views I have used Java Server Pages JSP, because performance is significantly better because JSP allows embedding Dynamic Elements in HTML Pages itself instead of having separate CGI files. Regular HTML, that is static HTML, does not contain dynamic [information](https://ecomputernotes.com/fundamental/information-technology/what-do-you-mean-by-data-and-information). So, it does not react to user input and is also not fit for accessing server-side resources. JSP contains both static and non-static content. As static part, it contains HTML.

**What Extra I Would Have Done**

# Spring Security

It is a framework that focuses on providing both authentication and authorization to Java applications. The real power of Spring Security is found in how easily it can be extended to meet custom requirements. Spring security is comprehensive, makes it easy for Servlets API integration and provides protection against attacks. Due to time constraints, I could not implement spring security. Spring provides authentication frameworks one of them is JWT based authentication. where a token will be generated at server side and validated for each user.

# Session Management

Spring Session provides an API and implementations for managing a user’s session information. while also making it trivial to support clustered sessions without being tied to an application container-specific solution Spring Session decouples the session management logic from the application, making it more tolerant.

Spring Session keeps information in the database, so it’s great to use in a clustered environment with multiple server nodes. Because of this, we don’t need to rely on the sticky session or session replication logic.

As session data is stored in the database, user session data is not lost if the application crashes. When the application started again, it picks up the user session from the database.

**Other Constraints to Be Considered**

Below are the few points that we should keep in mind for developing web applications.

# Security Constraints

A web application must be taken care of vulnerabilities and data breaches. Unfortunately, web applications are the most common ways of exploiting. Following cases must be considered for securing a web application:

Protect APIs using a Firewall.

By implementing secure authentications like two-factor authentication.

To the process of securely handling multiple requests to a web-based application from users.

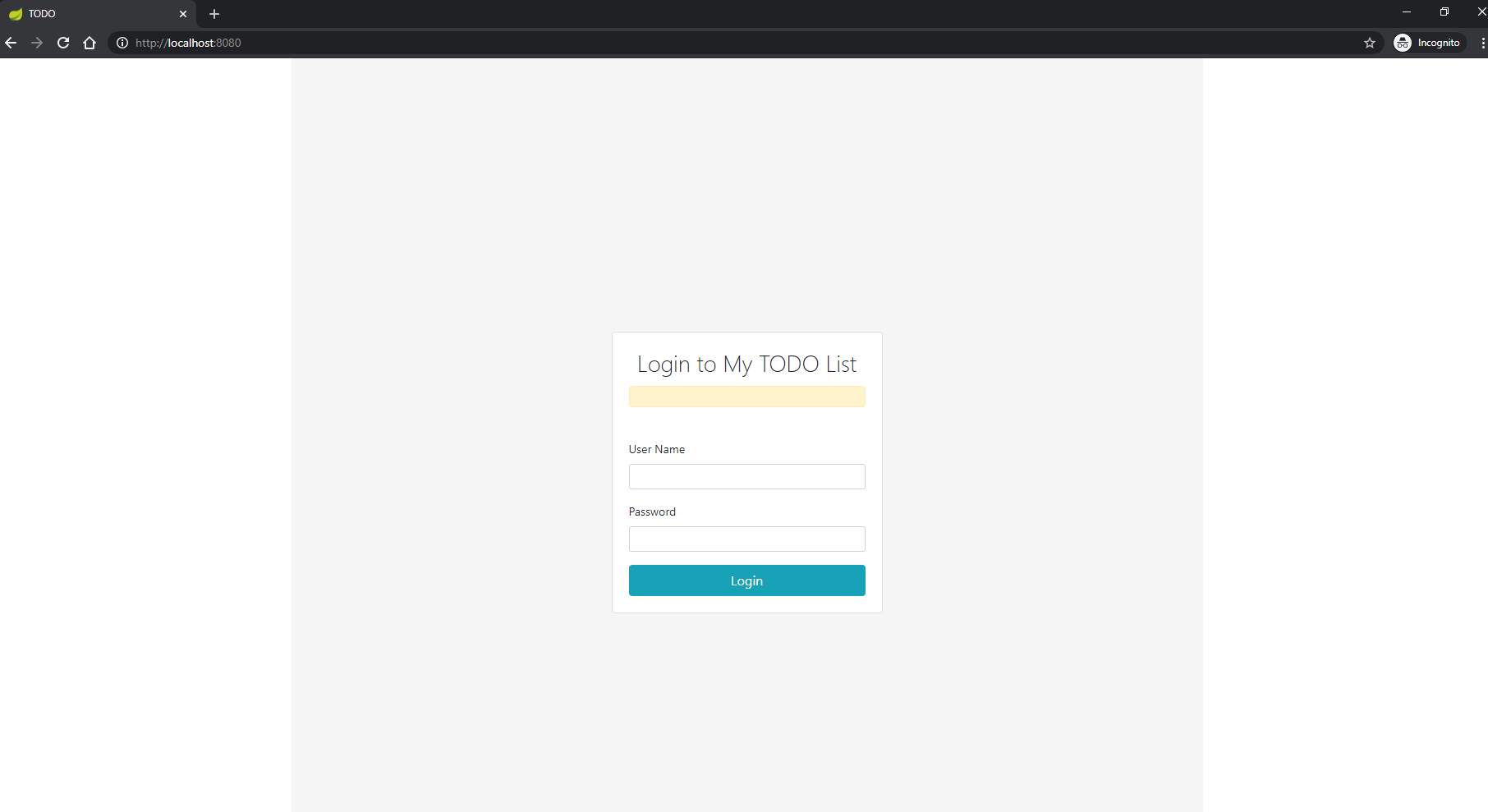
1. To avoid injection attacks in SQL using store procedure.
2. Form Validations for passwords.

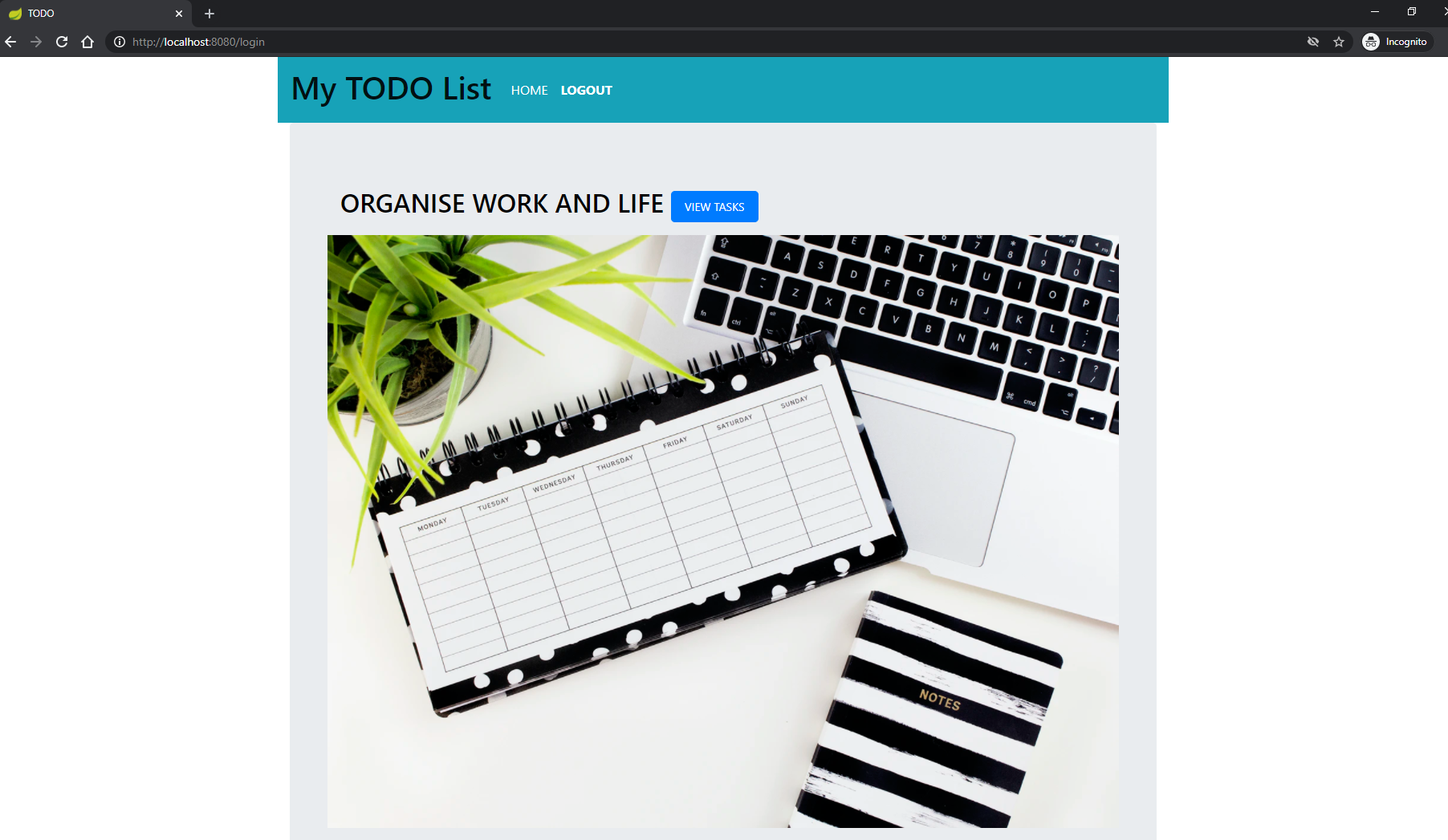
# Application Testing

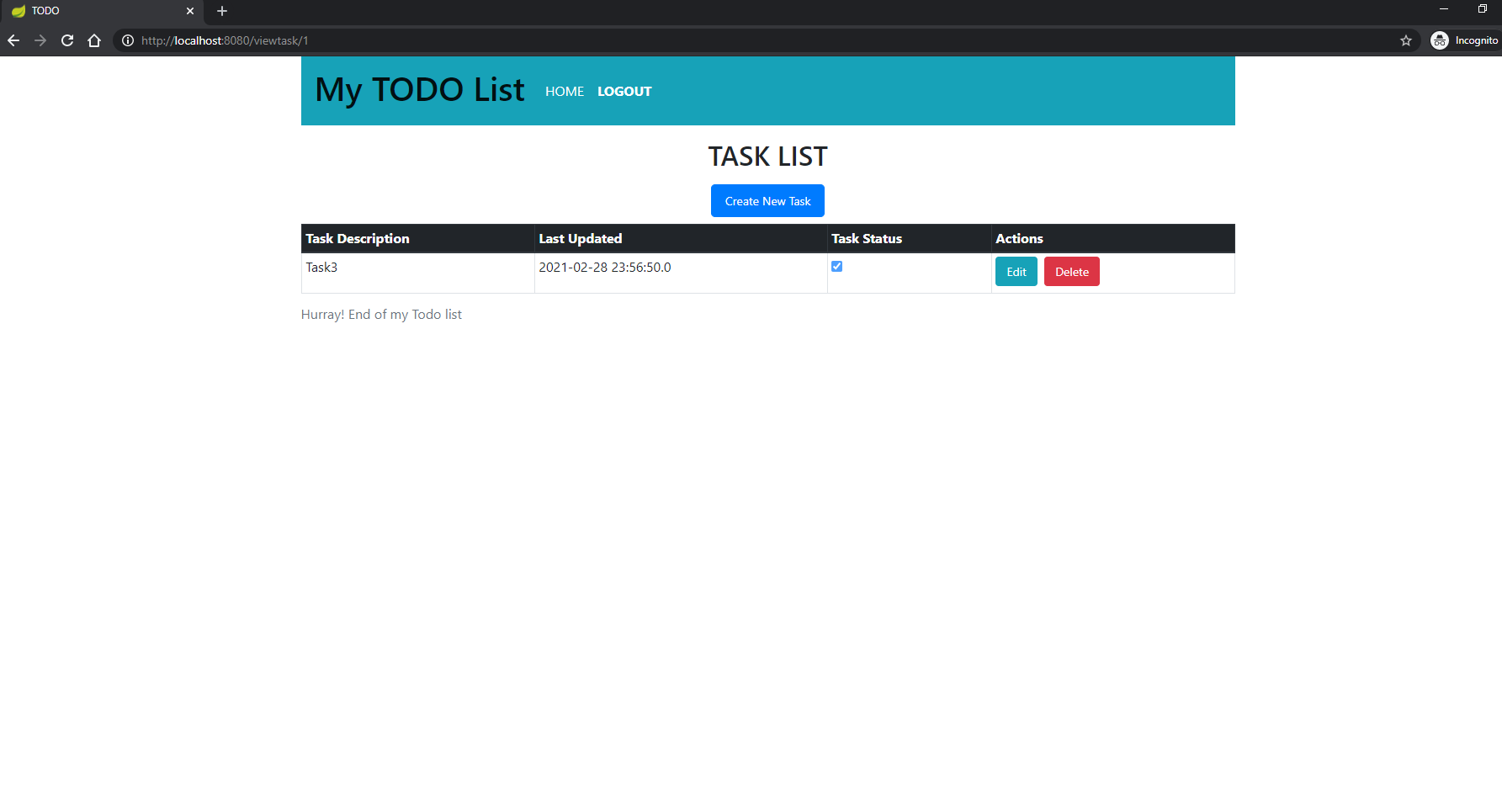
After Unit testing, we can go for SIT and UAT testing. Based on System requirements and design document, System Integration Testing (SIT) is conducted to support the interactions after the development of all the modules and integration so that the dependency on the availability of lower-level module functionality is verified.

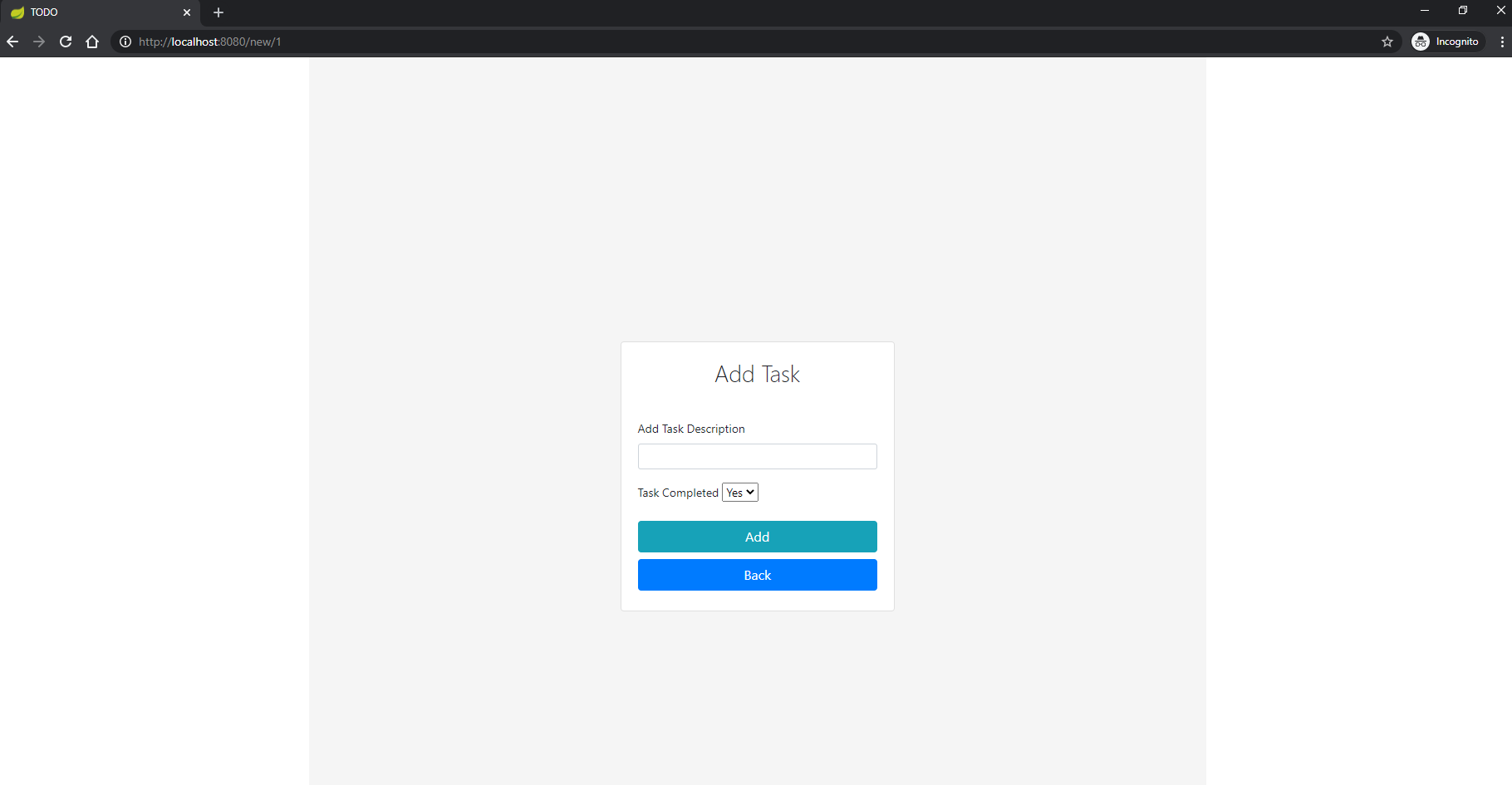
After SIT testing, User Acceptance Testing (UAT), also known as end-user testing must be used to test the software by the users to decide whether it can be accepted before production build once the functional, system and regression testing are completed. Finally, applications will be moved to production.

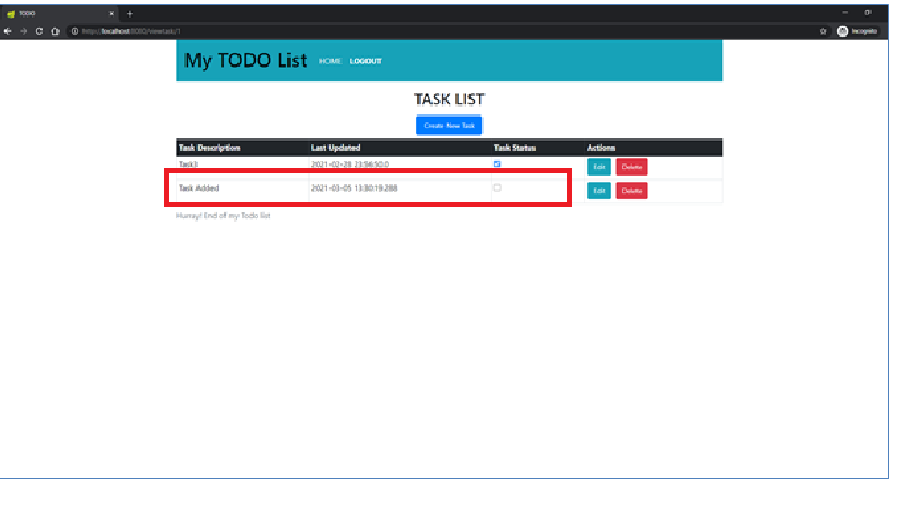
**Screen Shots of Todo web Application**

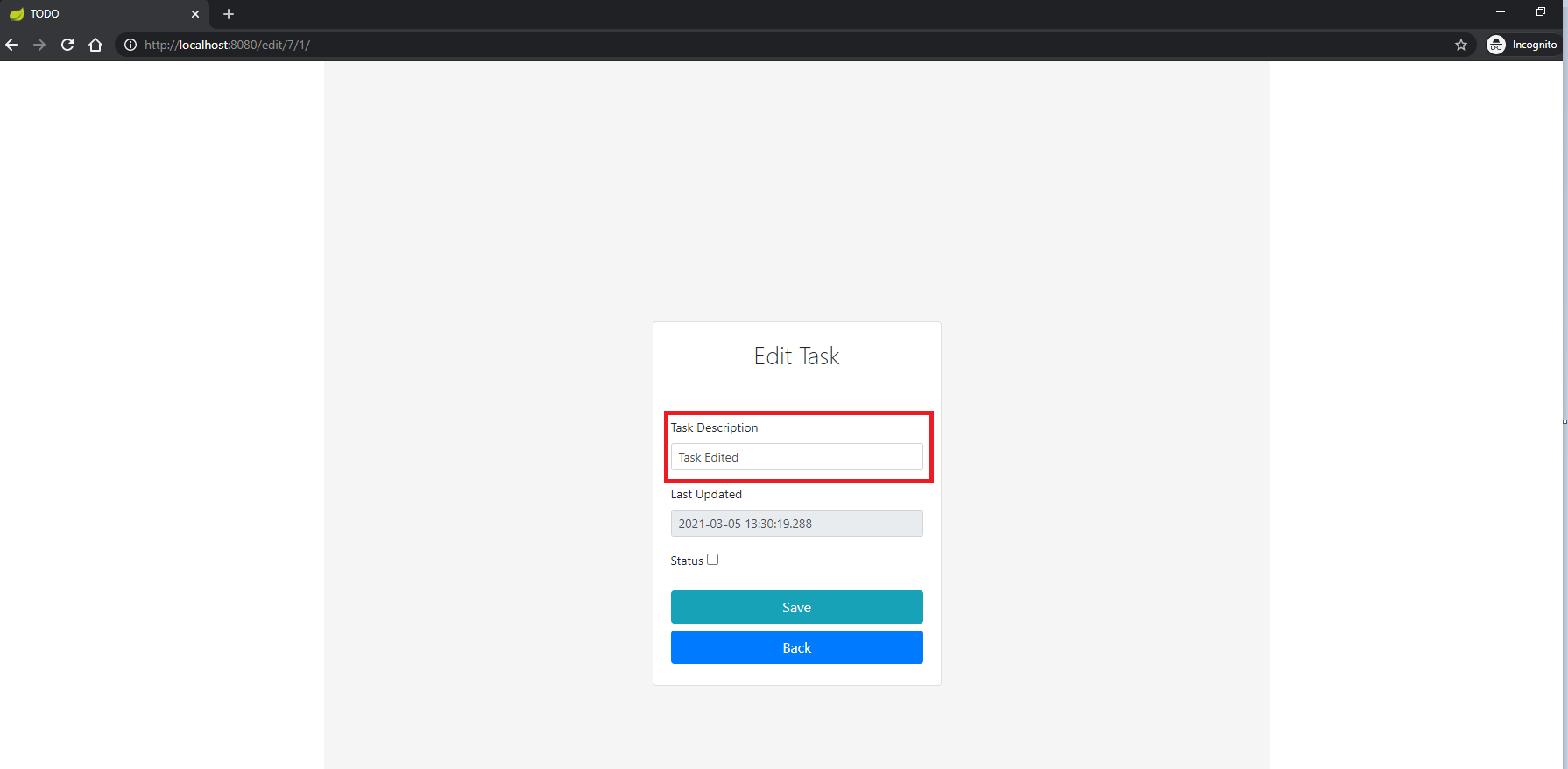
Login Page

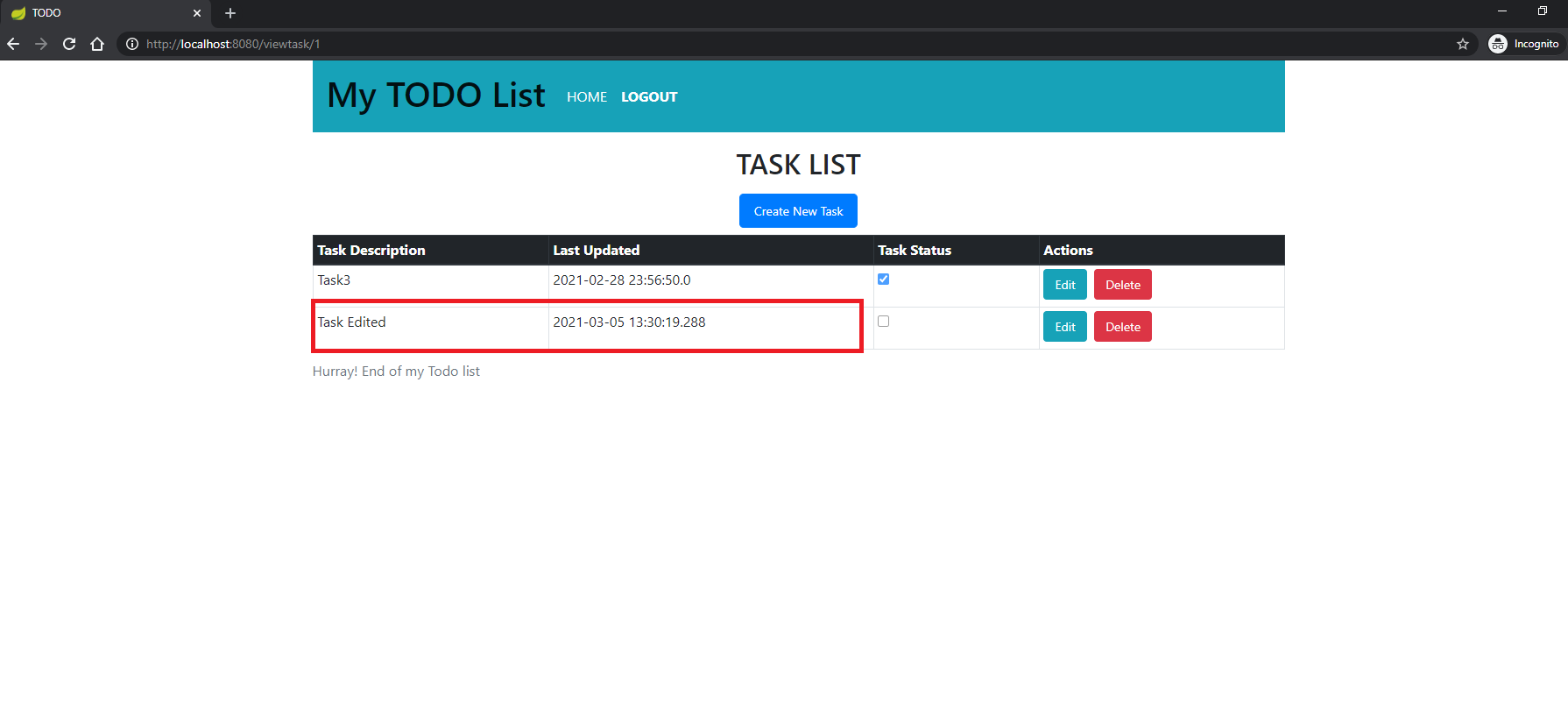
HomePage

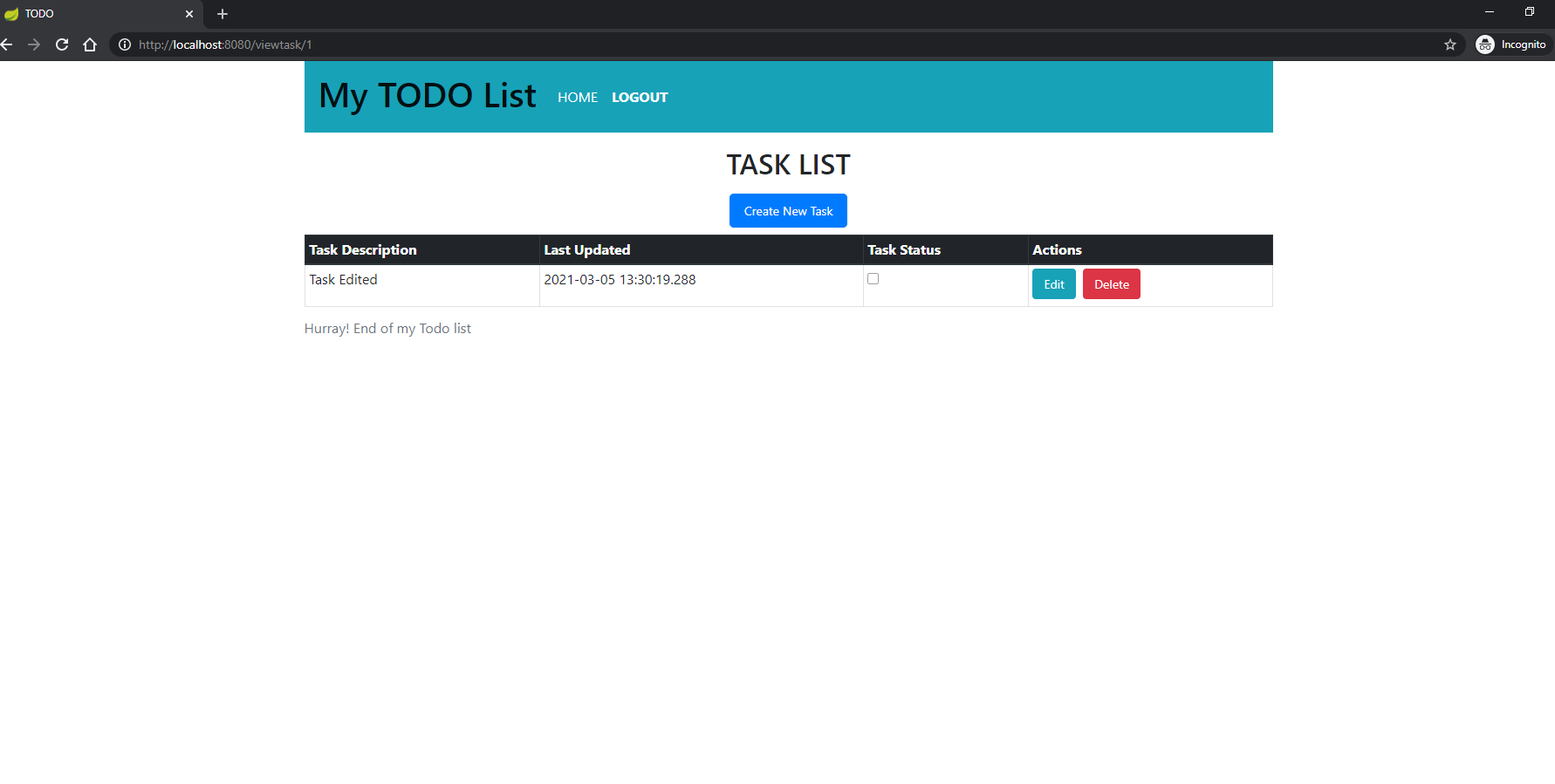
ViewTaskPage

 AddTaskPage

NewlyAddedTask Page

 EditTask Page

EditedTask

DeletedTask

**Conclusion:**

It was a good learning experience for me.There is a broad scope of improvements for the web application in the more extended run. To conclude, this project helped me to throughput my knowledge in front end technologies, java spring framework and data base.

**Steps to follow to run the TODO application in local machine.**

GIT HUB Link: <https://github.com/Vinay0809/DeloitteTodo>

Clone the spring boot TODO project using the above GitHub URL.

Import the above cloned project into IDE(Eclipse, STS, IntelliJ  etc)

Run maven build,” mvn clean install” (Maven should be already installed in your machine)

Run Application.java file, using run as java application.

In browser run the below URL

URL: <http://localhost:8080/>