**STATEMENT OF PURPOSE**

**Microservice Requirements:**

* IntelliJ IDE
* H2 Database
* JDK 1.8
* Spring Boot
* Junit dependency
* Thymeleaf
* Spring Boot DevTools
* Gradle

**Microservice Development Plan:**

1. Build Gradle project from the spring initializr website, setting the language to Java and adding in all the required dependencies.
2. Generate project and extract all into the necessary folder.
3. Open file in IntelliJ and begin creating new packages and classes required for the microservice.
4. Create and apply the methods necessary to build the microservice.
5. Create HTML files under the templates folder to exhibit outputs to the webpage.

**What the microservice will do:**

This microservice acts as a personal planner which enables the user to input their daily tasks and take down notes as a reminder of the upcoming tasks for that current day.

**What business problem it addresses:**

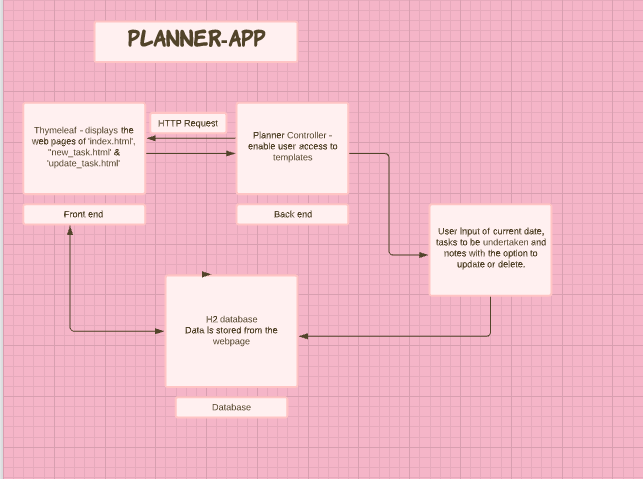
With ongoing duties needed to be fulfilled on a daily basis, this microservice will help assist individuals execute and keep track of the number of tasks and responsibilities they have listed.

**The target audience:**

This microservice is for individuals that that tend to have an excessive number of duties to carry out and would often need a reminder of how many tasks to complete.

**Microservice features:**

* Intake of user input such as the current date to enable user to list tasks set out for that day.
* Ability to store more than one set of tasks in microservice which allows user to input as many daily duties as possible.
* Enables user to update notes or tasks in case of errors made.
* Complete button – once tasks is completed, it will rid of the tasks/ notes listed.



**Implementation Plan:**

1. I generated my Gradle project from the spring initializr website.

2. I then created a new GitHub repository and named it ‘FinalProject’ and stored my Gradle project into that folder.

3. Once I opened up IntelliJ, I imported my ‘FinalProject’ folder into my IDE, and from there, I created my packages:

- controller

- model

-repository

-service

and then created my classes within those packages:

-Under controller package – PlannerController

-Under model package – Planner

-Under repository package – PlannerRepository

-Under service package – PlannerService, and PlannerServiceImpl

4. I started working on the model.Planner class and added in the necessary annotations, variables as well as generated the getters and setters’ method.

5. I then moved on to the PlannerService and PlannerServiceImpl and added in the necessary method and variables.

6. I then added “extends JpaRepository<Planner, Long> into the PlannerRepository interface.

7. Next, I added in methods into my controller class (using ‘GetMapping’ & ‘PostMapping’) which will allow the user to access and redirection to the webpages.

8. Once that was all sorted, I worked on the index.html file and worked on around building my table layout as well as the next and back button.

9. I then worked on changing the button appearances to add more shade to it, resulting in a more dimensional look.

10. After that, I added in my new\_task.html file allowing the user to add in a new task as well as adding additional buttons such as the ‘Save Tasks’ button.

11. Last file that I added into my templates was the update\_task.html. This allows the user to edit and update their planner when matters of errors have occurred or a change was needed to imply.

12. I then added in the H2 database into the application.properties as well as setting the server port to 8081.

13. I then created my controller test and ran a couple tests.

**Acknowledgments:**

[Stressed Busy GIF - Stressed Busy Work - Discover & Share GIFs (tenor.com)](https://tenor.com/view/stressed-busy-work-minnie-mouse-gif-8047187)

[Pagination and Sorting with Spring Boot, ThymeLeaf, Spring Data JPA, Hibernate, MySQL (javaguides.net)](https://www.javaguides.net/2020/06/pagination-and-sorting-with-spring-boot-thymeleaf-spring-data-jpa-hibernate-mysql.html)

[Spring Boot CRUD Web Application with Thymeleaf, Spring MVC, Spring Data JPA, Hibernate, MySQL - YouTube](https://www.youtube.com/watch?v=_5sAmaRJd2c)

[Background · Bootstrap v5.0 (getbootstrap.com)](https://getbootstrap.com/docs/5.0/utilities/background/)