République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université M'hamed Bougara - Boumerdès



Faculté des Sciences
Département d'Informatique

Domaine : Mathématiques Informatique
Filière : Informatique
Spécialité : Systèmes d'informatique
N° de l'Arrêté d'habilitation de la spécialité : arrêté n °872 du 26/07/2016

Mémoire de fin d'études en vu de l'obtention du Diplôme de Licence Académique

Thème

Personal Task Management Web Application

Présenté par :
BOUDRAA Meriem
FEKAIRI Dounia
GUEDDOU Rahma
MOUKADEM Safaa

Soutenu le ...../..../..... Devant le jury composé de
AMIR Karim : Examinateur
OULD BABAALI Kawther : Encadreur

Acknowledgments

First of all, we would like to express our deepest gratitude to God Almighty for giving us the strength, patience, and perseverance to complete this work successfully despite all the challenges encountered.

We would like to sincerely thank our supervisor, **Ms. Ould Babaali Kawther**, for her valuable guidance, continuous support, availability, and encouragement throughout the realization of this project. Her advice and suggestions greatly contributed to the success of this work.

We are also grateful to our examiner, **Mr. Amir Karim**, for taking the time to evaluate our work and for her constructive feedback and remarks.

We would like to extend our thanks to all our professors who have contributed to our academic journey through their teaching, support, and dedication.

Finally, we would like to express our heartfelt gratitude to our parents and families for their unconditional love, patience, and constant encouragement throughout our studies.

Contents

General Introduction **1**
1- Introduction . . . 1
2- Problems . . . 1
3- Proposed Solution . . . 1

Chapter 1: Analysis **3**
1- Introduction . . . 3
2- Definition of a Task Management Application . . . 3
3- History of Task Management Applications . . . 3
4- Importance of Task Management Applications . . . 3
5- Objectives . . . 3
6- Functional Requirements . . . 4

Chapter 2: Conception **5**
1- Introduction . . . 5
2- Actors Identification . . . 5
3- Use Case Diagram . . . 5
4-Class Diagram . . . 11
5-Sequence Diagram . . . 11
6-Add Task Activity Diagram . . . 14

Chapter 3: Implementation **16**
1- Introduction . . . 16
2-Development Environment . . . 16
3-Application Architecture . . . 17
4-Database Implementation . . . 18
5-Main Functionalities . . . 19
6-User Interfaces . . . 20
7-Security . . . 26
8-Conclusion . . . 27

General Conclusion **28**

List of Figures

1 Authentication Use Case Diagram 6
2 Task Management Use Case Diagram 7
3 Notes Use Case Diagram 9
4 Shopping List Use Case Diagram 10
5 Class Diagram 11
6 Sequence Diagram – Add Task 12
7 Sequence Diagram – Add Note 13
8 Sequence Diagram – Add Shopping Item 14
9 Add Task – Activity Diagram 15
10 Three-tier architecture diagram: Frontend Spring Boot MySQL 17
11 *Login page screenshot* 21
12 *Registration page screenshot* 21
13 Task dashboard screenshot 22
14 *Add task form screenshot* 23
15 *Upcoming tasks page screenshot* 24
16 *Notes page screenshot* 25
17 *Shopping list page screenshot* 26

List of Tables

1 Authentication Use Cases Description 6
2 Task Management Use Cases Description 8
3 Notes Use Cases Description 9
4 Shopping List Use Cases Description 10
5 Development technologies 16
6 Database tables description 19

General Introduction

1- Introduction

Digital transformation has changed the way people work and organize their lives through technologies and web applications that improve access to information and enhance productivity. However, managing personal tasks remains difficult for many people due to numerous daily responsibilities. Traditional methods such as paper agendas or handwritten notes have limitations and do not always allow effective and structured activity tracking. In this context, web applications dedicated to task management have emerged as modern and effective solutions. They make it possible to centralize activities, add deadlines, classify tasks based on their priority, monitor their progress, and access their data anytime and anywhere through an Internet connection. It is within this framework that our final-year project is developed, consisting of the design and development of a web application for personal task management. The objective is to provide a simple, intuitive, and effective tool that allows users to create, modify, delete, and organize their daily tasks while benefiting from a clear and user-friendly interface that promotes better organization and improved productivity. In addition, the application integrates complementary features such as note management and shopping lists in order to provide users with a more complete platform to organize their personal activities.

2- Problems

Many users experience difficulties managing their daily tasks. As the number of tasks increases, it becomes difficult to identify urgent and important ones. Without a clear priority system, important tasks may be delayed. Organizing tasks can also become challenging, especially when users have many activities to manage at the same time. As the number of tasks increases, it becomes difficult to group them properly or quickly find a specific task. Without clear categories or simple organizational methods, planning becomes less efficient. In addition, some users face difficulties centralizing their personal notes and shopping lists in a single digital space, which may lead to information loss or poor organization of daily activities. Thus, this project aims to develop a personal task management web application that helps users define priorities and organize their tasks easily while providing a simple, intuitive, and user-friendly system.

3- Proposed Solution

In our project, we propose developing a personal task management web application that allows users to better organize their daily activities. This solution will be developed using modern web technologies and a database system to ensure efficient data storage and accessibility through the following features:

- Definition and management of task priorities in order to focus on the most important activities.
- Classification of tasks into different categories for clear and structured organization.
- Personal note management to allow users to store important information and access it easily.
- Creation of shopping lists to help users organize their daily needs in a practical and efficient way.
- Task tracking to identify tasks that are in progress or completed.
- A clear and intuitive interface to facilitate use and improve productivity.

Chapter 1: Analysis

1- Introduction

This chapter presents the objectives and functional requirements of the personal task management web application. The goal is to help users organize and track their daily tasks in a simple and efficient way.

2- Definition of a Task Management Application

A task management application is a software tool that allows users to organize, plan, and monitor their daily activities in a simple and efficient manner. It provides several functionalities such as task creation, priority definition, adding deadlines, activity categorization, and tracking the progress of tasks. These applications can be used on computers or smartphones and help users improve their personal organization and productivity

3- History of Task Management Applications

The first task management methods were based on traditional tools such as paper agendas, calendars, and handwritten lists. With the development of computing and the Internet, digital task management applications began to appear in order to facilitate the organization of personal and professional activities. With the evolution of technology and the widespread use of smartphones, these applications have significantly evolved thanks to new technologies and smartphones. Today, modern applications such as Todoist and Microsoft To Do allow users to manage their tasks remotely, synchronize their data, and access their information at any time.

4- Importance of Task Management Applications

In the modern era, task management applications have become essential due to the fast pace of life and the development of new technologies. The increase in daily responsibilities, the use of the Internet, and the widespread use of connected devices have made personal organization more complex. These applications therefore help save time, improve productivity, and better manage priorities. They also provide quick access to information from any connected device, which facilitates the monitoring of daily activities in an increasingly digital and connected world.

5- Objectives

The objective of this project is to develop a web application that enables users to easily manage their personal tasks. The application will provide the following functionalities:

- Creation and management of personal tasks;
- The ability to modify or delete tasks;
- Priority management to identify important tasks;
- Task organization through categories;
- A simple and user-friendly interface;
- Personal note management;
- Creation and organization of shopping lists.

This solution aims to improve the organization of daily activities and facilitate users' time management.

6- Functional Requirements

The functional requirements of the application can be summarized as follows:

- User authentication;
- Creation of new tasks;
- Modification of existing tasks;
- Deletion of tasks;
- Viewing the task list;
- Task priority management;
- Task categorization;
- Tracking task status (completed or pending);
- Personal note management;
- Creation and management of shopping lists;
- Searching for tasks or information.

Chapter 2: Conception

1- Introduction

This chapter presents the conception phase of the personal task management web application. The system design was modeled using UML diagrams in order to describe the functional behavior of the application and the interactions between the user and the system. Different diagrams such as use case diagrams and activity diagrams are presented to illustrate the application's functionalities and workflow.

2- Actors Identification

The system contains different actors that interact with the application functionalities.

- User: the main actor who manages tasks, notes, and shopping lists.
- Email System: external service used for email verification, reminders, and password recovery.
- File System: external component responsible for image storage and retrieval in notes management.

3- Use Case Diagram

Use case diagrams illustrate the interactions between the different actors and the functionalities provided by the application. They help identify the system services and define the responsibilities of each actor.

3.1 Authentication Use Case Diagram

The authentication module manages secure access to the application through user registration, login, email verification, and password recovery functionalities.
The following diagram illustrates the interactions between the user, the email system, and the authentication services provided by the application.

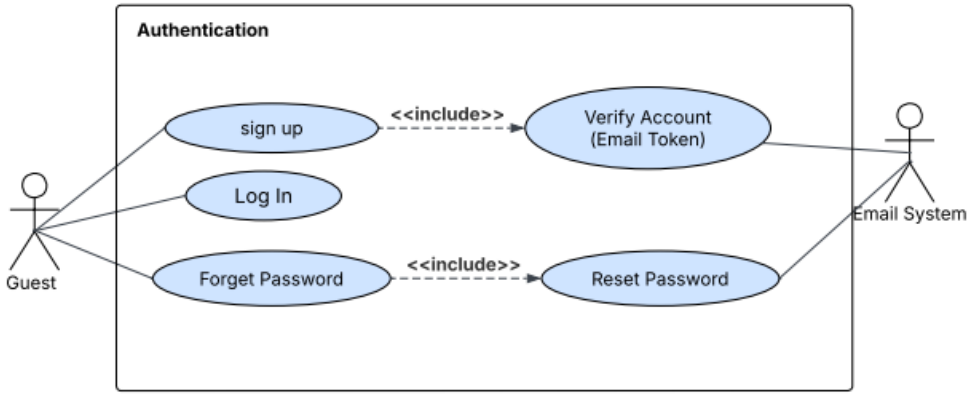


Figure 1: Authentication Use Case Diagram

A detailed description of the authentication use cases is presented in the following table.

1. Authentication	
Actors: User, Email System	**Description:** Secure access to the system through registration, login, and password recovery.
Use Cases	**Description**
Sign Up	Creates a new user account.
Log In	Authenticates the user and grants access to the system.
Forgot Password	Starts the password recovery process via email.
Verify Account	Confirms email ownership using a verification token.
Reset Password	Allows the user to set a new password securely.
Email System	Sends verification and reset password emails.

Table 1: Authentication Use Cases Description

3.2 Task Management Use Case Diagram

The task management module is one of the principal functionalities of the application. It allows users to create, manage, organize, and track their tasks efficiently.
The following diagram illustrates the interactions between the user and the different task management functionalities such as task creation, editing, searching, filtering, and completion tracking.

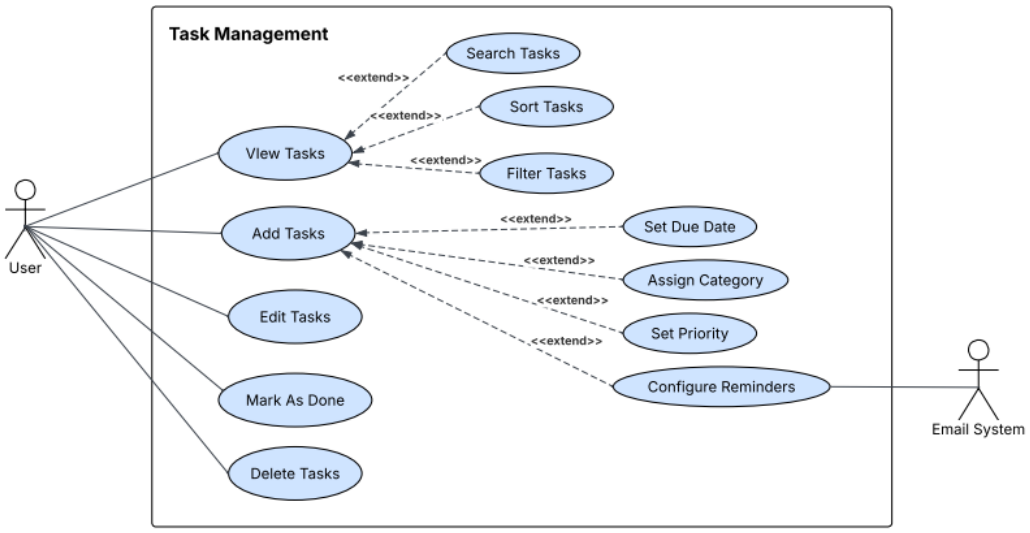


Figure 2: Task Management Use Case Diagram

A detailed textual description of the task management use cases is presented in the following table.

2. Task Management	
Actors: User, Email System	**Description:** Core module for managing personal tasks with CRUD, filtering, sorting, and scheduling features.
Use Cases	**Description**
View Task	Displays all tasks belonging to the authenticated user.
Add Task	Creates a new task with basic details (title, description, etc.).
Edit Task	Modifies the content or metadata of an existing task.
Delete Task	Permanently removes a task from the system.
Search Tasks	Filters the task list by keyword or criteria.
Mark as Done	Changes the status of a task to completed.
Sort Tasks	Reorders tasks by a selected attribute (date, priority, etc.).
Filter Tasks	Narrows the task list by category, status, or date range.
Set Due Date	Assigns an optional deadline to a task during creation or editing.
Assign Category	Links a task to a user-defined category.
Set Priority	Assigns a priority level (Low / Medium / High).
Configure Reminders	Optionally schedules email notifications for tasks.
Email System	External service responsible for sending reminder and notification emails to users.

Table 2: Task Management Use Cases Description

3.3 Notes Use Case Diagram

The following diagram illustrates the functionalities related to notes management. It describes the different operations that can be performed by the user such as creating, editing, deleting, searching, and pinning notes.

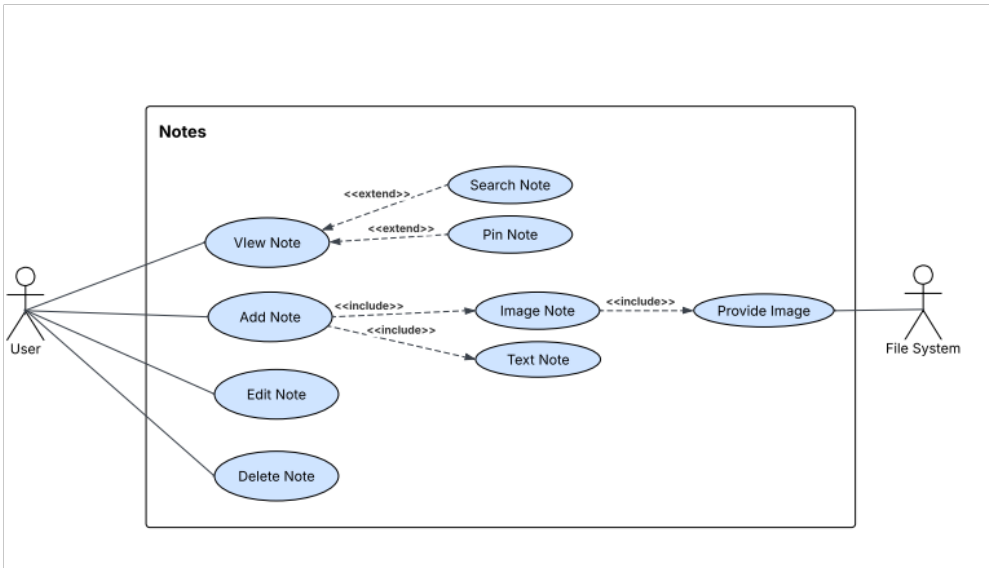


Figure 3: Notes Use Case Diagram

This module allows users to organize their personal notes efficiently and improve access to important information.

3. Notes	
Actors: User, File System	**Description:**Enables users to capture and organize free-form notes, supporting both plain text and image-based content with pinning and search capabilities.
Use Case	**Description**
View Note	Displays all notes created by the authenticated user.
Add Note	Creates a new note; requires selection of note type (Text or Image).
Edit Note	Updates the content or metadata of an existing note.
Delete Note	Permanently removes a note from the system.
Search Note	Filters notes by keyword or tag.
Pin Note	Marks a note as pinned so it appears at the top of the list.
Text Note	Included by Add Note captures plain text content.
Image Note	Included by Add Note captures image-based content.
Provide Image	Included by Image Note delegates image retrieval/upload to File System.
File System	External service that handles image storage and retrieval.

Table 3: Notes Use Cases Description

3.4 Shopping List Use Case Diagram

The shopping list module allows users to organize shopping categories and manage shopping items. The following diagram presents the interactions between the user and the shopping list functionalities.

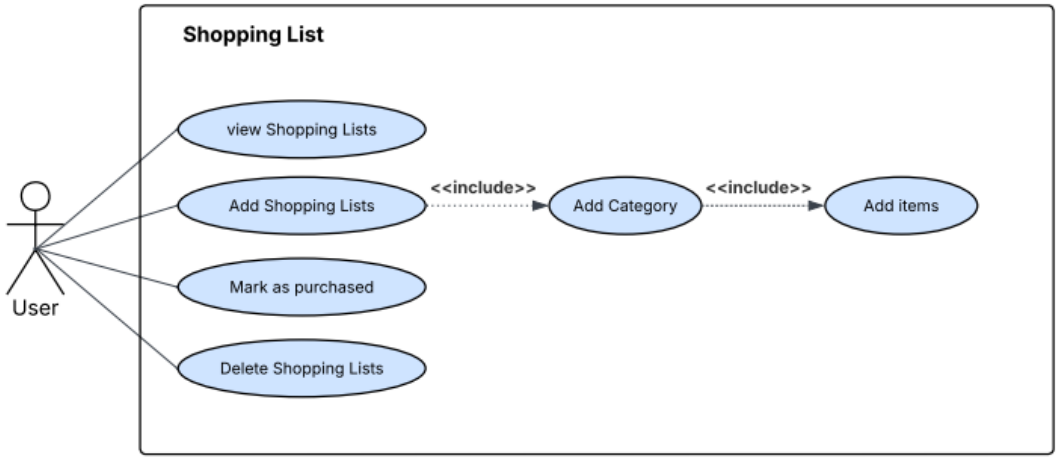


Figure 4: Shopping List Use Case Diagram

Through this module, users can create shopping lists, add items, assign categories, and mark purchased items.

4. Shopping List	
Actors: User	**Description:** Provides a shopping management module that allows users to organize multiple shopping lists and track purchased items.
Use Case	**Description**
View Shopping Lists	Displays all shopping lists belonging to the user.
Add Shopping Lists	Creates a new named shopping list.
Delete Shopping Lists	Permanently removes a shopping list and its items.
Add Items	Automatically included when creating or editing a list adds individual items.
Add Category	Automatically included with Add Items assigns a category to each item.
Mark as Purchased	Toggles the purchased status of an item within a list.

Table 4: Shopping List Use Cases Description

4-Class Diagram

The class diagram describes the overall structure of the application by defining its main entities and the relationships between them. The following diagram presents the static architecture of the system.

The diagram is organized around the central class `User`, which manages authentication and owns three main modules: notes, tasks, and shopping lists. The `Note` class is extended by two subclasses, `TextNote` and `ImageNote`, through an inheritance relationship. The `Task` class is associated with the `Reminder` class, as well as two enumerations: `Priority` and `Category`. Finally, the `ShoppingList` class is composed of one or more `ShoppingItem` objects.

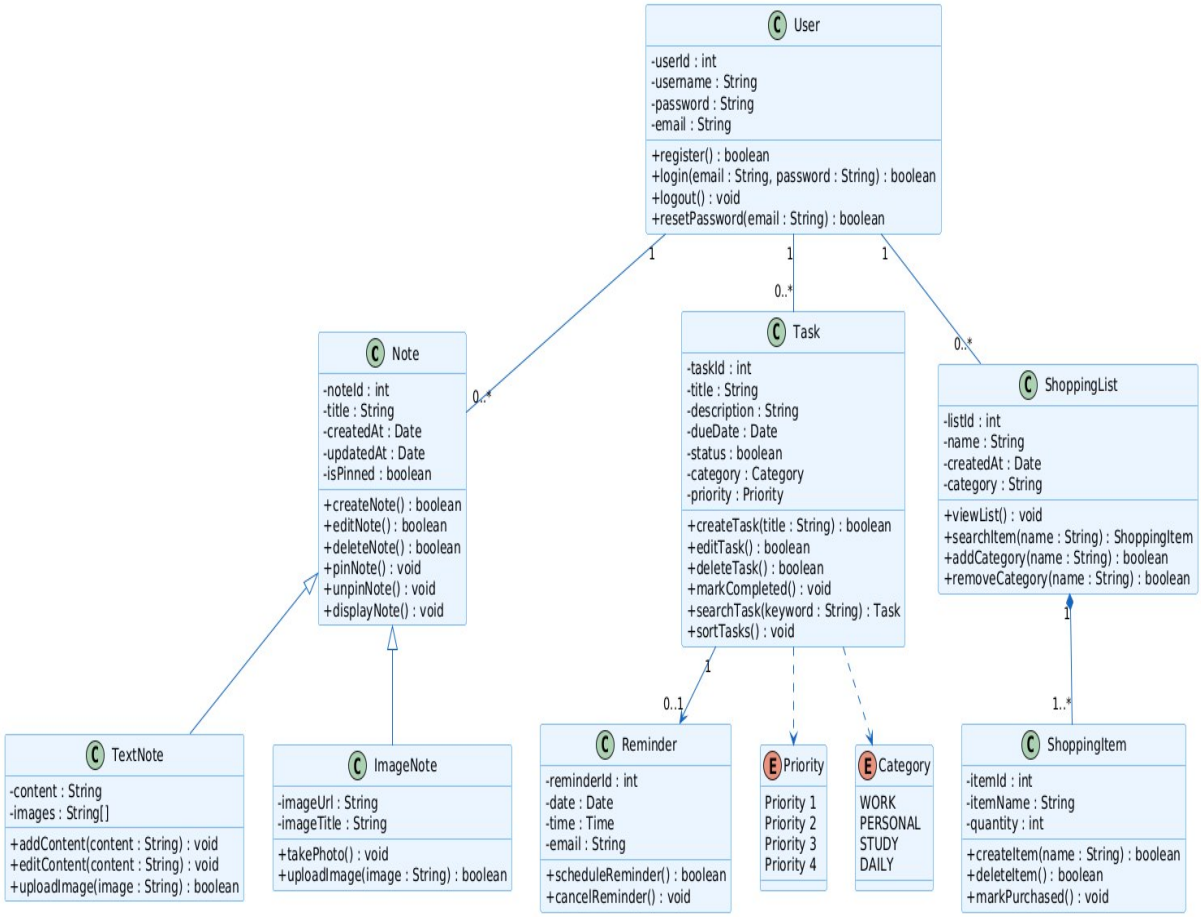


Figure 5: Class Diagram

5-Sequence Diagram

The system includes three main sequence diagrams: Add Task, Add Note, and Add Shopping Item, each describing how the user interacts with the application during creation processes.

5.1 Add Task Sequence Diagram

The task module allows users to create tasks with optional attributes such as category, priority, due date, and reminder. The following diagram presents the interactions between the user, the application, the `Task` class, and the `Reminder` class during the task creation process.

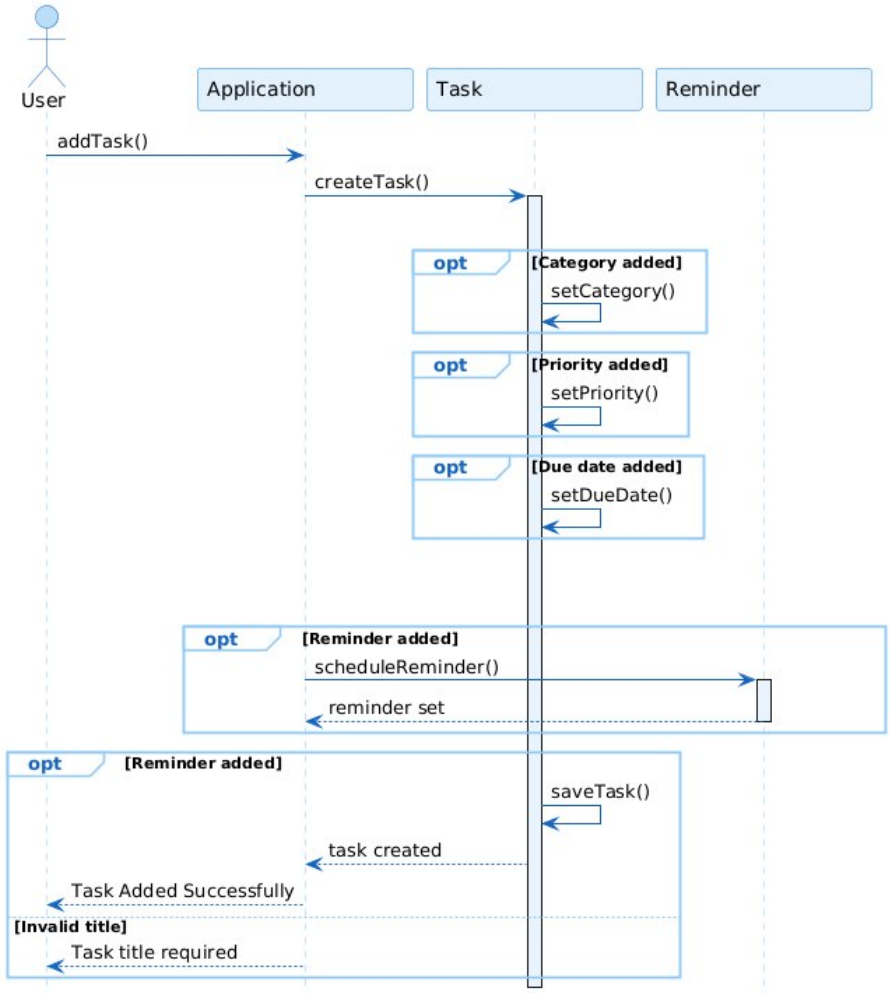


Figure 6: Sequence Diagram – Add Task

The user initiates the action by calling `addTask()`, which triggers `createTask()` on the `Task` object. Three optional fragments [**opt**] allow the user to enrich the task by setting a category, a priority level, and a due date. An additional optional fragment handles the scheduling of a reminder through the `Reminder` class. Once all optional attributes are set, `saveTask()` is called to persist the task. A final alternative fragment [**alt**] validates the task title: if valid, the user receives a success message; otherwise, an error message is displayed indicating that the title is required.

5.2 Add Note Sequence Diagram

The note module allows users to create and manage personal notes with optional text content. The following diagram presents the interactions between the user, the application,

and the `TextNote` class during the note creation process.

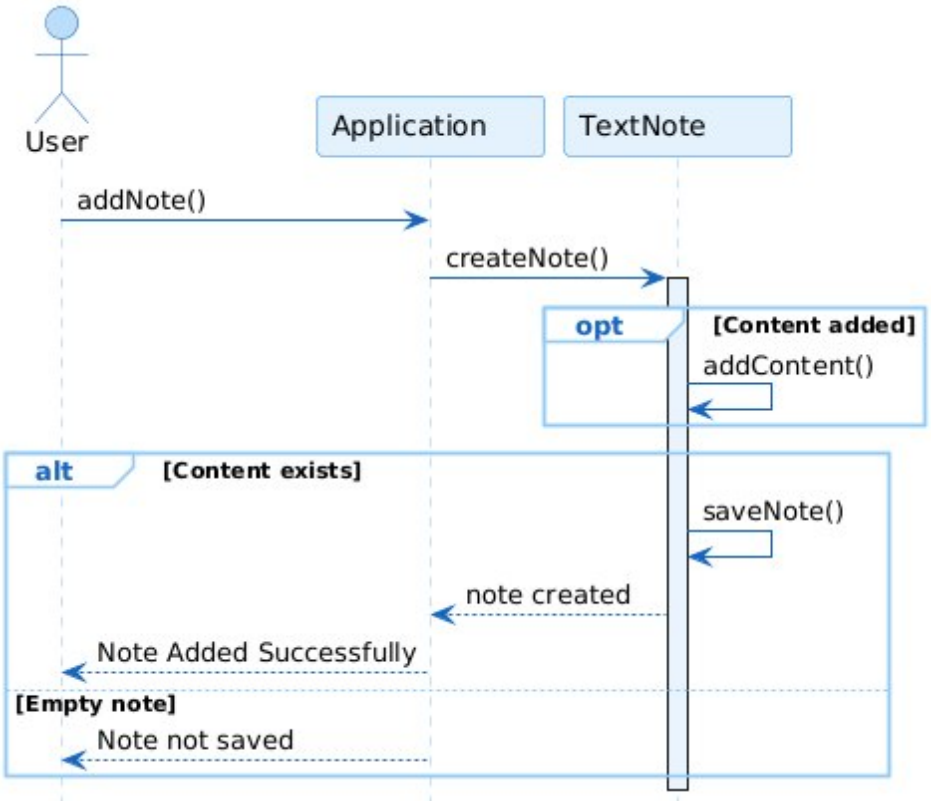


Figure 7: Sequence Diagram – Add Note

The user initiates the process by calling `addNote()`. The application then triggers `createNote()` on the `TextNote` object. If content is provided, the optional fragment [**opt**] activates the `addContent()` method. An alternative fragment [**alt**] then handles two cases: if content exists, `saveNote()` is called and the user receives a success confirmation; if the note is empty, it is not saved and the user is notified accordingly.

5.3 Add Shopping Item Sequence Diagram

The shopping list module allows users to organize shopping categories and manage shopping items. The following diagram presents the interactions between the user, the application, `ShoppingList`, and `ShoppingItem` during the item creation process.

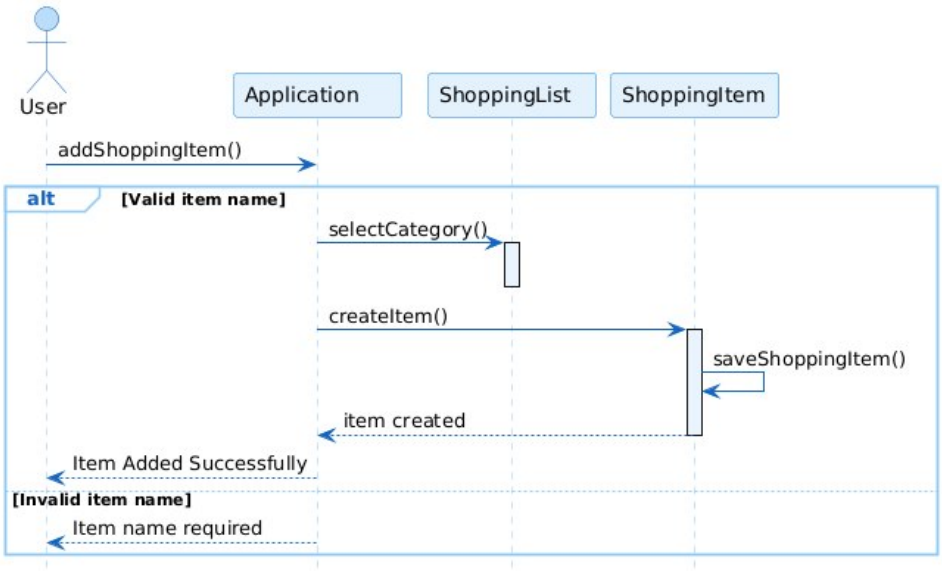


Figure 8: Sequence Diagram – Add Shopping Item

The user triggers the process by calling `addShoppingItem()`. An alternative fragment [**alt**] separates two scenarios based on the validity of the item name. If the name is valid, the application calls `selectCategory()` on `ShoppingList`, then `createItem()` on `ShoppingItem`, which internally executes `saveShoppingItem()` to persist the data. The user is then notified of the successful addition. If the item name is invalid or empty, the operation is rejected and the user receives an error message.

6-Add Task Activity Diagram

The following diagram illustrates the step-by-step flow of the task creation process, from opening the form to saving the task in the database.
The process begins when the user opens the task form and enters a task title. If the title is empty, an error message is displayed and the user is prompted to retry. Once a valid title is provided, the user may optionally enter a description, set a due date, assign a category, and define a priority level. The user is then asked whether they wish to configure a reminder; if so, a reminder date and time can be set. Finally, the user clicks *Save task*, upon which the system validates and saves the task to the database, displays a success message, and redirects the user to the task lis

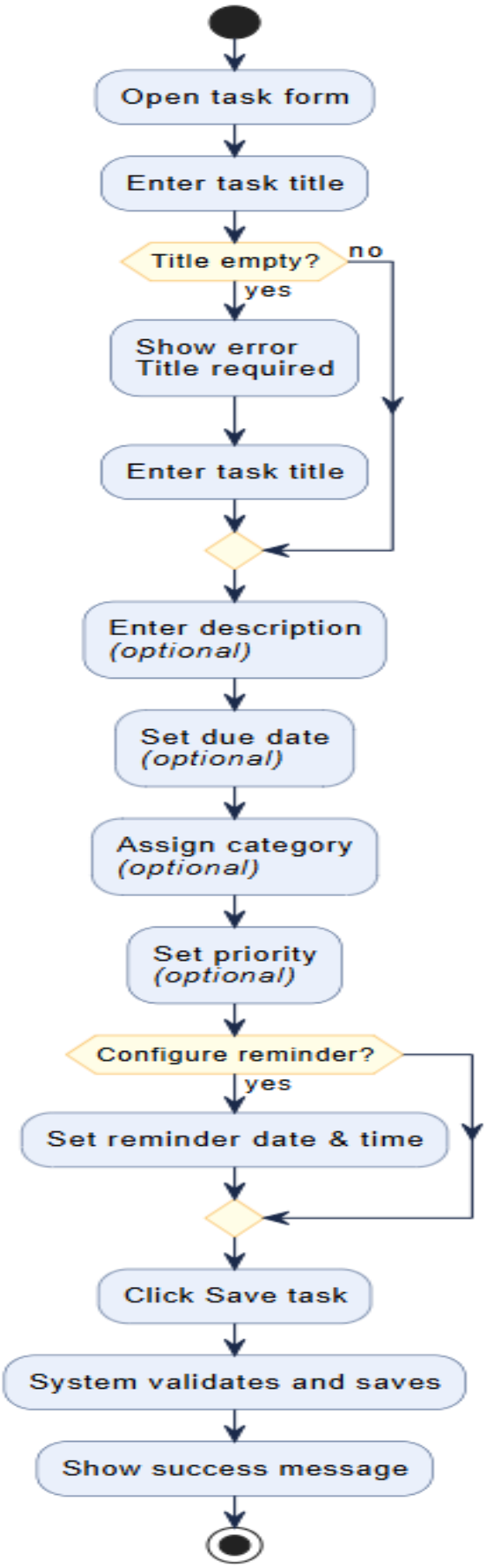


Figure 9: Add Task – Activity Diagram

Chapter 3: Implementation

1- Introduction

This chapter presents the implementation phase of the Personal Task Management Web Application. It covers the technical aspects of the project, including the development environment, the application architecture, the database design, the main functionalities, the user interfaces, and the implemented security mechanisms. The objective of this chapter is to provide a clear overview of how the system was developed and how the different components interact to deliver a functional and user-friendly web application.

2-Development Environment

The application was developed using modern technologies and tools suitable for web application development.The table below summarises the main technologies used throughout the project.

Component	**Technology**	**Role**
Programming Language	Java 17	Core backend language
Framework	Spring Boot 4.0.5	Application framework and embedded server
Template Engine	Thymeleaf	Server-side HTML rendering
Frontend	HTML/CSS/JavaScript	User interface and client-side logic
Database	MySQL 8.0	Relational data storage
ORM	Spring Data JPA + Hibernate	Database mapping and persistence
Build Tool	Apache Maven	Dependency management and build
Mail	Spring Mail (Gmail SMTP)	Email notifications and verification
IDE	IntelliJ IDEA / Eclipse	Development environments
Version Control	GitHub	Source code management and versioning

Table 5: Development technologies

2.1 Programming Language and Framework

Java 17 was used as the main backend programming language due to its stability and reliability in web application development. Spring Boot was used as the backend framework to simplify application configuration, manage business logic, and facilitate database access through Spring Data JPA.

2.2 Frontend Technologies

The frontend was developed using HTML5, CSS3, and JavaScript to create responsive and interactive user interfaces. Thymeleaf was used as the server-side template engine for rendering dynamic web pages.

2.3 Development Tools

IntelliJ IDEA and Eclipse were used as development environments for coding and testing the application. GitHub were used for version control and project management. The application was executed locally using the embedded Tomcat server provided by Spring Boot.

3-Application Architecture

The application follows a three-tier architecture composed of the presentation layer, the business logic layer, and the data layer. This architecture improves code organization, maintainability , scalability , and separation of concerns. In addition, the backend follows the MVC (Model-View-Controller) design pattern provided by Spring Boot.

3.1 Architecture Overview

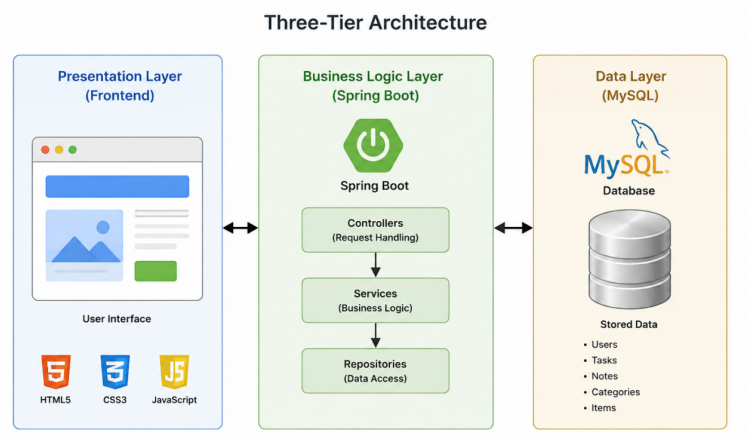


Figure 10: Three-tier architecture diagram: Frontend Spring Boot MySQL

3.2 Presentation Layer

The presentation layer was developed using HTML, CSS, JavaScript, and Thymeleaf. It provides the graphical user interface and allows users to interact with the application through forms and dynamic pages.

3.3 Business Logic Layer

The business logic layer was implemented using Spring Boot and follows the MVC (Model-View-Controller) architecture. It includes:

- Controllers: handle HTTP requests and responses.
- Services: contain the business logic of the application.
- Repositories: manage database operations using Spring Data JPA.

The main controllers are:

- AuthController for authentication and account management.
- TaskController for task management.
- NoteController for notes management.
- ShoppingController for shopping list management.

3.4 Data Layer

The data layer uses MySQL for relational data storage and Hibernate/JPA for database communication and persistence. Entity classes are mapped automatically to database tables using JPA annotations.

3.5 Communication Flow

When a user performs an action through the user interface, the frontend sends an HTTP request to the Spring Boot backend. The controller processes the request, communicates with the service and repository layers, interacts with the MySQL database, and finally returns the generated view or response to the user interface.

4-Database Implementation

The application uses a MySQL database named taskdb. The database structure is based on the main entities defined in the Java source code. These entities are mapped to database tables using Spring Data JPA and Hibernate.
The database includes the following main tables:

Table	Description
User	Stores registered users, authentication information, account status, and verification tokens.
Task	Stores user tasks, including title, description, priority, category, due date, status, order, and reminder information.
Note	Stores personal notes created by users, including title, content, creation date, status, and position.
Category	Stores shopping list categories created by users.
Item	Stores shopping list items and their purchase status.

Table 6: Database tables description

The relationships between the entities are defined as follows:

- One user can have many tasks.
- One user can have many notes.
- One user can have many shopping categories.
- One category can contain many items.
- Each item belongs to one category and one user.

These relationships are implemented using foreign keys and JPA annotations. The use of Hibernate allows the application to perform CRUD (Create, Read, Update, Delete) operations efficiently and simplify database interaction.

5-Main Functionalities

This section presents the principal functionalities implemented in the application to support personal task and activity management.

5.1 User Account Management

The application provides a complete user account management system including account registration, email verification, authentication, and password recovery. These functionalities ensure secure and personalized access to the platform.

5.2 Task Management

The task management module allows users to:

- create, edit, and delete tasks.
- manage task priorities and categories
- mark tasks as completed or pending.
- search and filter tasks.
- reorder tasks using drag-and-drop.
- receive email reminders.
- view overdue and upcoming tasks.
- view weekly task statistics.

5.3 Note Management

The note management module enables users to create and organize personal notes. Users can pin important notes, and quickly search for stored information.

5.4 Shopping List Management

The shopping list module allows users to create shopping categories and manage shopping items. Users can organize products and mark items as purchased to simplify shopping activity tracking.

5.5 Dashboard and Statistics

The application provides a dashboard that summarizes task information, including overdue tasks, completed tasks, and productivity statistics. This functionality helps users monitor their activities and improve task organization.

6-User Interfaces

The user interfaces were developed using Thymeleaf, HTML, CSS, and JavaScript. The application provides a simple, responsive, and interactive user interface that facilitates task and personal activity management.

6.1 Login Page

The login page allows users to access their accounts using their email address and password.

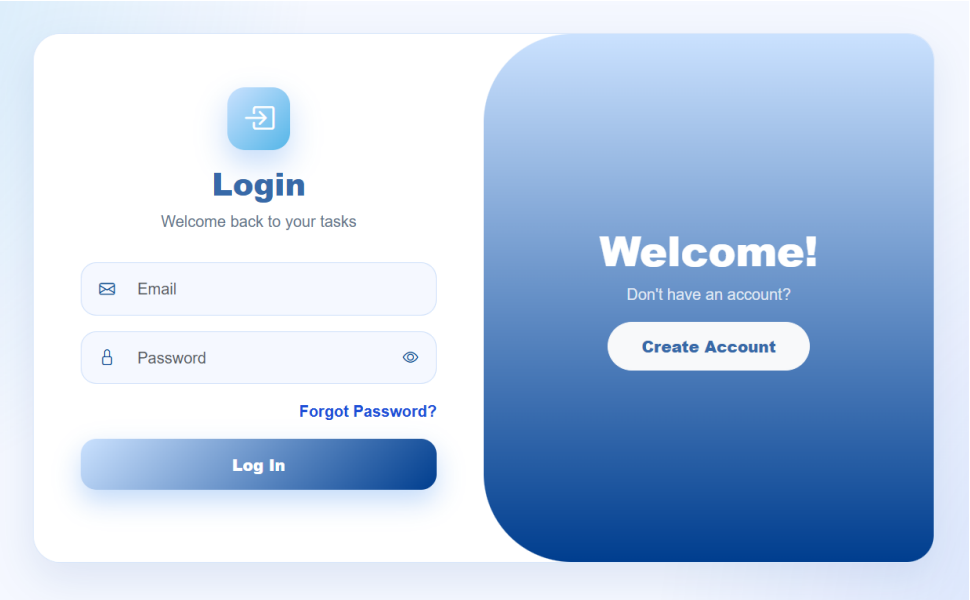


Figure 11: *Login page screenshot*

6.2 Registration Page

The registration page allows new users to create accounts by entering their personal information. After registration, a verification email is sent to activate the account.

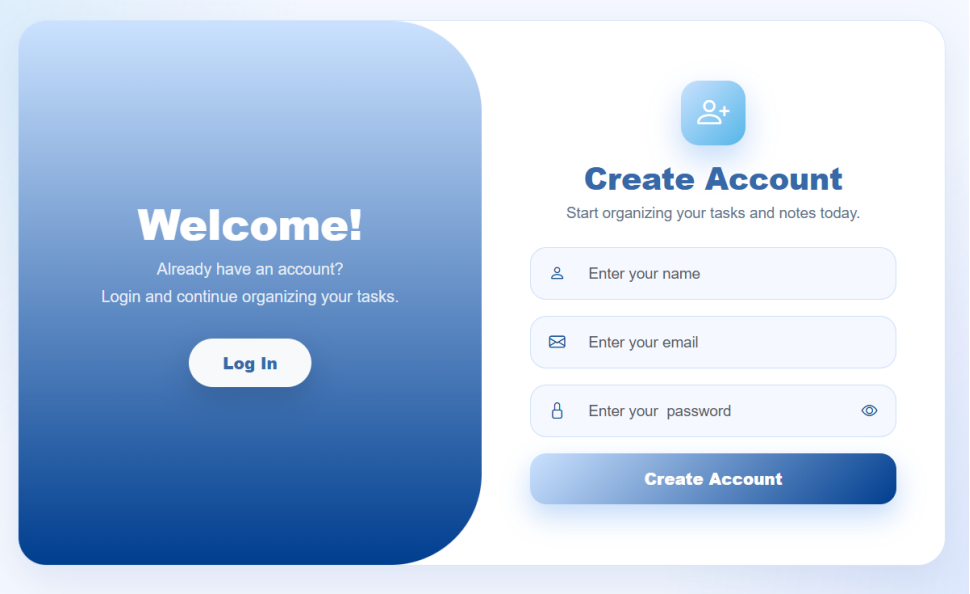


Figure 12: *Registration page screenshot*

6.3 Task Dashboard

The dashboard is the main interface of the application. It displays active, overdue, and completed tasks and provides functionalities such as task creation, search, filtering, drag-and-drop reordering, and productivity statistics.

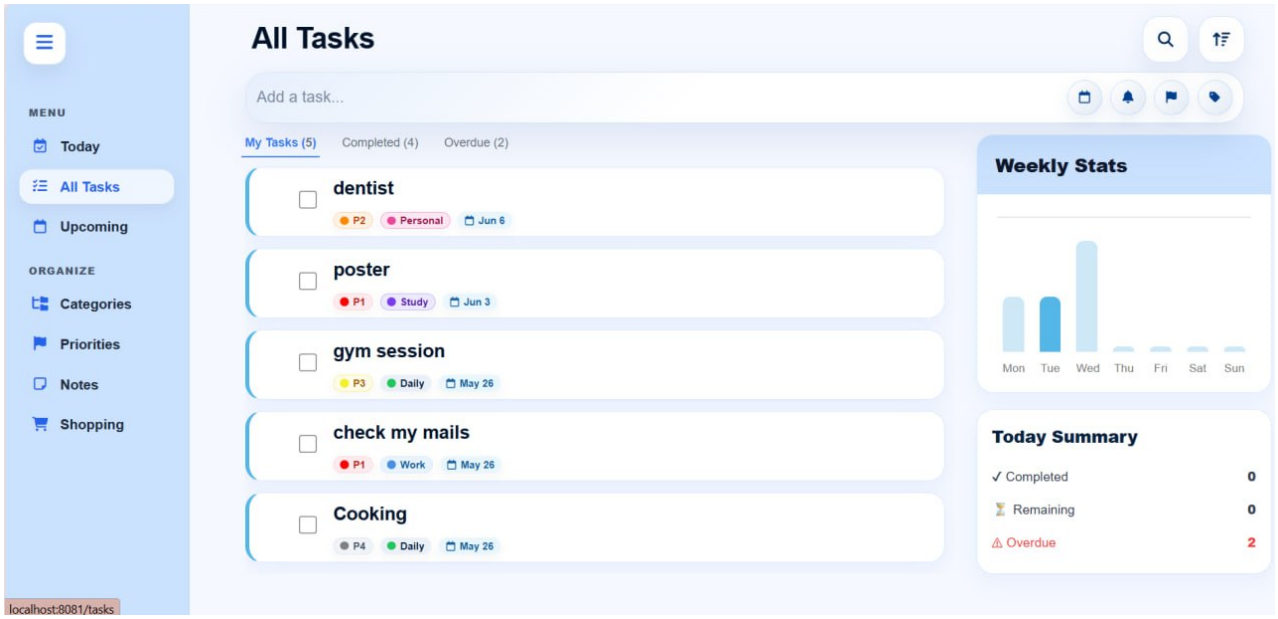


Figure 13: Task dashboard screenshot

6.4 Add Task Form

This interface allows users to create tasks by defining the title, description, priority, category, due date, and reminder settings.

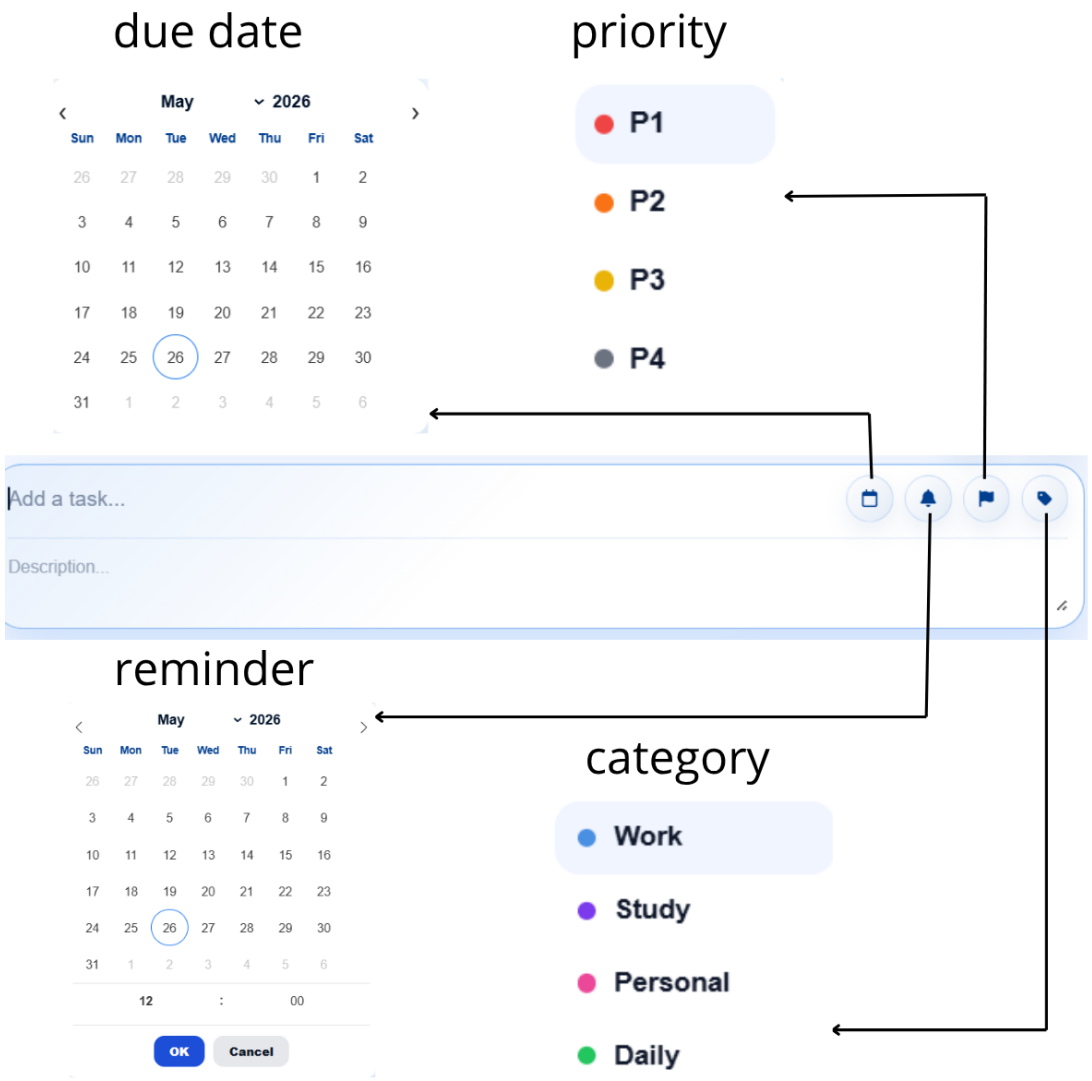


Figure 14: *Add task form screenshot*

6.5 Upcoming Tasks Page

The upcoming tasks page displays an interactive calendar that allows users to navigate between weeks and months and view scheduled tasks for each day.

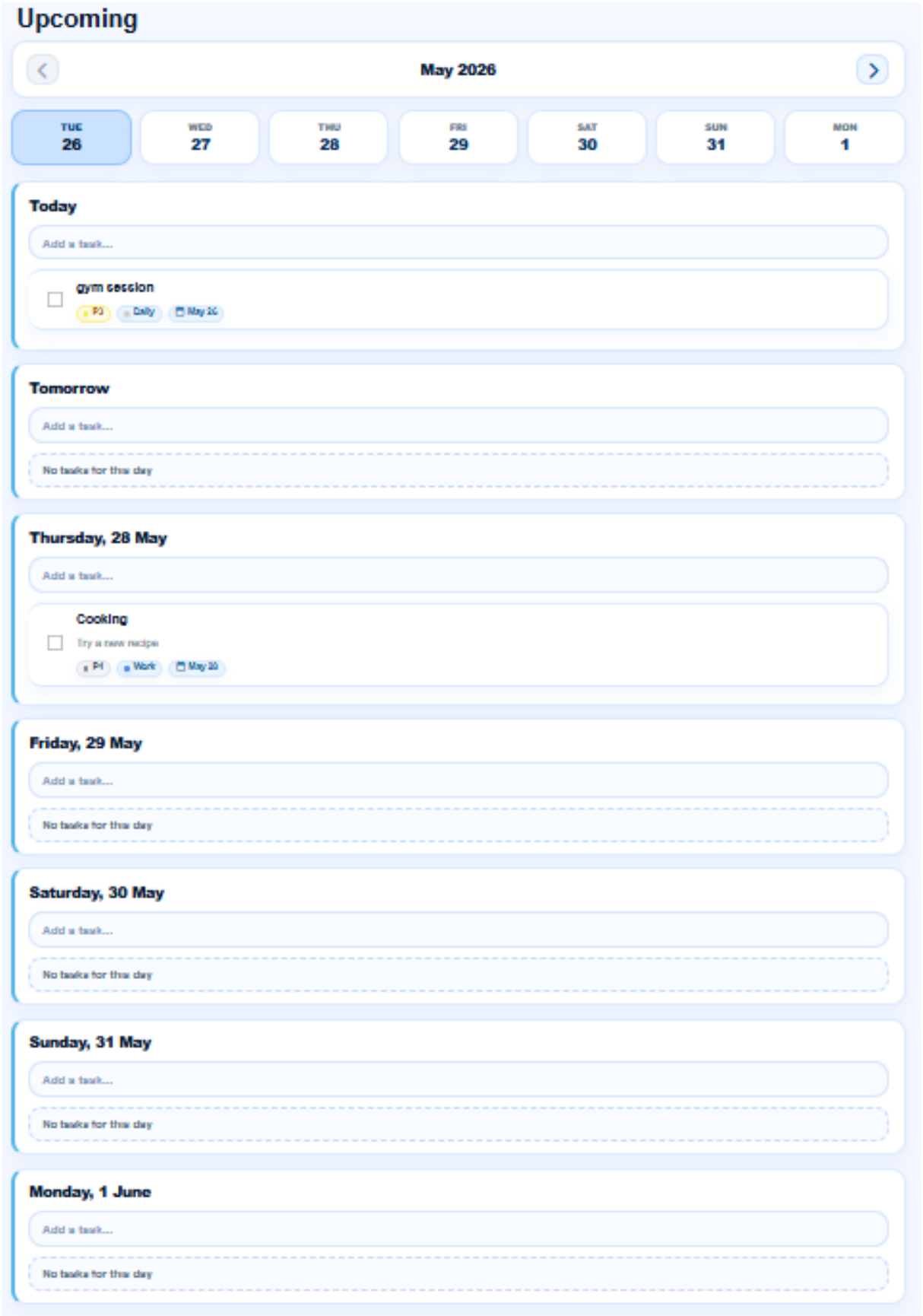


Figure 15: *Upcoming tasks page screenshot*

6.6 Notes Page

The notes page allows users to create, edit, delete, pin, and search personal notes.

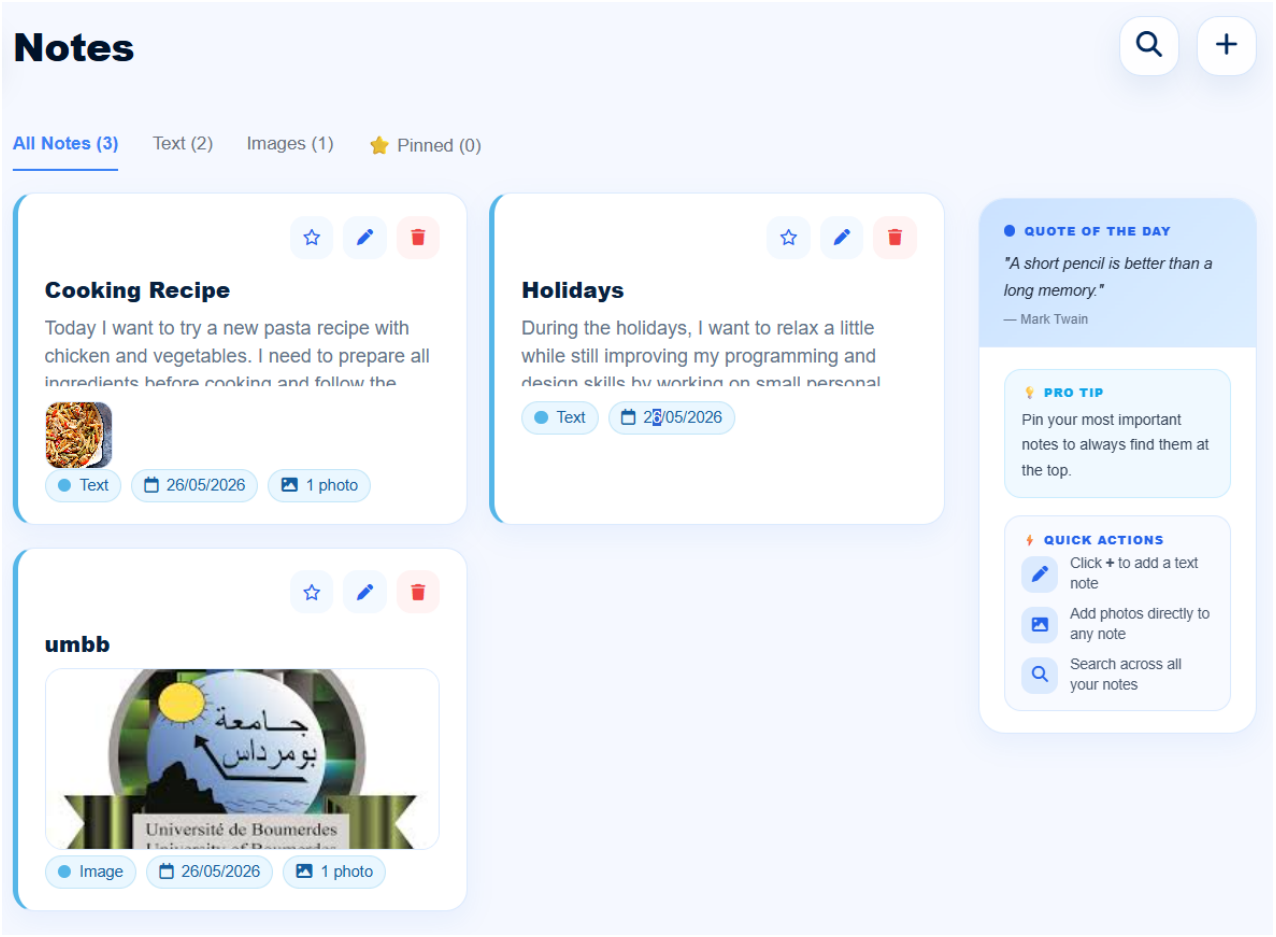


Figure 16: *Notes page screenshot*

6.7 Shopping List Page

The shopping list page allows users to manage shopping categories and items, add new products, and mark purchased items.

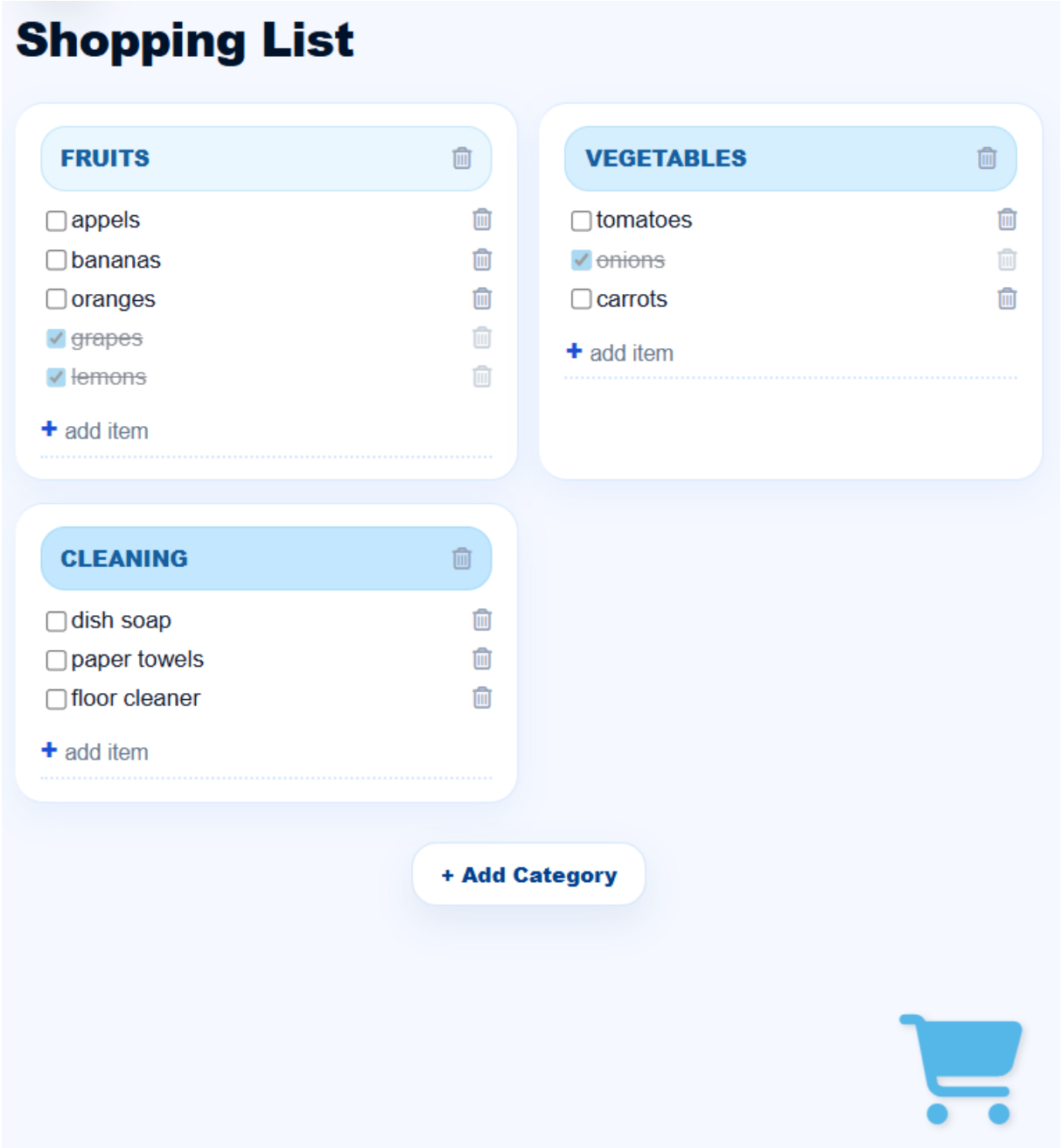


Figure 17: *Shopping list page screenshot*

7-Security

7.1 BCrypt Password Hashing

Your passwords are never saved as plain text. You use BCryptPasswordEncoder to hash the password before saving it to the database. Even if someone accessed the database, they cannot read the passwords.

7.2 Token-based Email Verification

When a user registers, a unique random token (UUID) is generated and sent by email as a link. The account only gets activated after the user clicks that link. This prevents fake or unverified accounts.

7.3 Token-based Password Reset

The same token system is used for password reset. A unique link is sent to the user's email. Once used, the token is deleted from the database so it cannot be reused.

7.4 Session-based Access Control

After login, the user is stored in the HTTP session. Every protected page checks if the session contains a logged user. If not, the user is redirected to the login page.

7.5 Duplicate Email Protection

During registration, the system checks if the email already exists in the database before creating a new account, preventing duplicate registrations.

8-Conclusion

This chapter presented the implementation of the Personal Task Management Web Application, including the technologies used, the application architecture, the database structure, the main functionalities, and the user interfaces. The developed system provides features such as task management, note management, shopping lists, reminders, and user authentication.
Overall, the application successfully meets the project objectives by providing a functional, user-friendly, and reasonably secure platform for personal task and productivity management.

General Conclusion

This project allowed us to design and develop a web application for personal task management. Our main goal was to help users organize their daily activities in a simple and efficient way. We started this project because many people struggle to stay organized when their tasks, notes, and shopping lists are scattered across different places. During the analysis phase, we clearly defined what the application needed to do. This gave us a strong foundation for the development phase. We used modern and reliable technologies, specifically Spring Boot 4.0.5 and MySQL, to build an application that is easy to use and works on any device with an internet connection. Today, the platform successfully allows users to create, update, delete, and track tasks, set priorities, organize activities into categories, manage personal notes, and receive email reminders all in one place. The application also includes security mechanisms such as BCrypt password hashing and token-based email verification to protect user accounts. While this first version is fully functional, there is still room for growth. In the future, the platform can be improved by adding features like push notifications, task sharing between users, a mobile application, and integration with external calendars. Overall, this project was a great learning experience. It allowed us to apply our academic knowledge to a real-world software project and gave us a deeper understanding of full-stack web development. We believe this application can serve as a solid foundation for a more complete and advanced productivity platform.

Bibliography

[1] Pirdaus, D. I., & Hidayana, R. A. (2024). *Analysis Testing Black Box and White Box on Application To-Do List Based Web*. International Journal of Mathematics, Statistics, and Computing, Vol. 2, No. 2, pp. 68–75. e-ISSN: 3025-0803. Available at: PDF Link

[2] Holop, R. (2022). *NCF TO-DO'ER: A To-Do List Web Application That Assists Students with Time Management Skills*. New College of Florida. Available at: Thesis Link

[3] ProofHub. (2024). *Top Note-Taking Apps for Productivity*. Available at: Website

[4] Todoist. (2024). *Todoist The To-Do List to Organize Work and Life*. Available at: Official Website

[5] Out of Milk. (2024). *Out of Milk Shopping List App*. Available at: Official Website

[6] Walls, C. (2016). *Spring Boot in Action*. Manning Publications.

[7] Walls, C. (2022). *Spring in Action* (6th Edition). Manning Publications.

[8] Sierra, K., & Bates, B. (2005). *Head First Java*. O'Reilly Media.

[9] Bauer, C., & King, G. (2015). *Java Persistence with Hibernate*. Manning Publications.

[10] Oracle Corporation. (2021). *Java SE 17 Documentation*. Available at: Documentation

[11] VMware. (2024). *Spring Boot Email Support (JavaMailSender)*. Available at: Docs

[12] Oracle Corporation. (2024). *MySQL 8.0 Reference Manual*. Available at: Manual

[13] Apache Software Foundation. (2024). *Apache Maven Getting Started Guide*. Available at: Guide

[14] Object Management Group (OMG). (2017). *Unified Modeling Language (UML) Specification, Version 2.5.1*. Available at: Specification

[15] GitHub, Inc. (2024). *GitHub Documentation Get Started with Version Control*. Available at: Docs