CAN THO UNIVERSITY
COLLEGE OF INFORMATION AND COMMUNICATION TECHNOLOGY

GROUP PROJECT
COURSE: INTRODUCTION TO SOFTWARE ENGINEERING
COURSE CLASS: CT114HM02

PLANNER AND PRODUCTIVITY SOFTWARE “MY SECOND BRAIN”

Instructor:
Ph.D. Phan Phuong Lan

Group Members:
B2203594 Nguyen Hoang Vu
B2203578 Le Cao Anh Tai
B2203572 Thach Thanh Nhi
B2203589 Nguyen Thi Kim Tran

Can Tho, 04/2024

Table of Contents

Revision History

Name	**Date**	**Reason For Changes**	**Version**
Revising the use case and interface of schedule function	5/5/2024	The use case is unreasonable and the interface lacks some components.	1.1

References

[1] IEEE Computer Society, *IEEE Recommended Practice for Software Requirements Specifications*, IEEE Std 830-1998, 1998.

[2] Karl E. Wiegers, *Software Requirements Specification Template*, 1999.

[3] Bộ Thông tin và Truyền thông, *Hướng dẫn về các yêu cầu phi chức năng chung cho các hệ thống thông tin cung cấp dịch vụ công trực tuyến,* 2013.

[4] Joydip Kanjilal, *Comparing the MVC MVP and MVVM Design Patterns*, online, cited 2021.

[5] Các Nguyên Tắc Thiết Kế Phần Mềm, Viblo, online, cited 2022.

[6] Notion policy, online, cited February 2024.

[7] Database basis for building mobile app, Sonia Rebecca Menezes, online.

[8] Database design basics, Microsoft, online, cited March 2024.

[9] Simple Todo list, Dario Stefanutto, online, cited April 2024.

[10] Project CACC++ 4, Justin Newman, Rohit Sen, Matt Howard, Sam Belcher, Noah Engerer, online, cited April, 2024.

Acronyms and Abbreviations

#	**Acronyms and Abbreviations**	**Explanation**
1	M2B	My Second Brain (application’s name)
2	AES	Advanced Encryption Standard
3	TLS	Transport Layer Security
4	MVVM	Model - View - ViewModel

1. Plan

1.1 Group Organization

#	Student Code	Full Name	Role
1	B2203594	Nguyễn Hoàng Vũ	Team Leader
2	B2203578	Lê Cao Anh Tài	
3	B2203572	Thạch Thanh Nhi	
4	B2203589	Nguyễn Thị Kim Trân	Vice of Team Leader

1.2 Plan

Week	Task	Member (Name)	Deadline	Level of completion	Note
Software Requirement Specification					
6	Product perspective	Vũ	7/2/2024	100%	
	Schedule	Trân	7/2/2024	100%	Complete the construction of the main ideas for the function (overview, subfunctions, constraints, etc.)
	Mind map	Tài	7/2/2024	100%	
	To-do list	Nhi	7/2/2024	100%	
	Progress tracker	Vũ	7/2/2024	100%	
	Hardware platform	Vũ	7/2/2024	100%	
	Operating system and versions	Nhi, Trân	7/2/2024	100%	
	Other software applications and components coexist peacefully	Tài	7/2/2024	100%	
	Design and implementation constraints	all	7/2/2024	100%	
	External Interface Requirements	Tài	7/2/2024	100%	
	Nonfunctional Requirements	Vũ	7/2/2024	100%	
	Group meeting	all	12/2/2024	100%	Modify and complete “Design and Implementation Constraint”, go

					through each member’s part
11	Functional requirements of Schedule	Trân	06/03/2024	100%	
	Functional requirements of Mind map	Tài	06/03/2004	100%	
	Functional requirements of To-do list	Nhi	06/03/2004	100%	
	Functional requirements of Progress Tracker	Vũ	06/03/2004	100%	
	Group meeting	all	11/3/2024	100%	Modify Schedule & Mind map use cases, discuss the information that Progress Tracker function uses & modify Progress Tracker use case, go through each member’s part
Software Design					
12	Application Architecture	Trân	20/03/2024	100%	
	Data design	Vũ	20/03/224	100%	
	Detailed Design of Schedule function	Trân	20/03/2024	100%	
	Detailed Design of Mind map function	Tài	20/03/2024	100%	
	Detailed Design of To-do list function	Nhi	20/03/2024	100%	
	Detailed Design of Progress Tracker function	Vũ	20/03/2024	100%	
	Group meeting	all	25/03/2024	100%	Rediscuss the app’s main interface and main color, rediscuss each member’s interface

Unit Testing					
16	Schedule test case	Trân	15/4/2024	100%	
	Mind map test case	Tài	15/4/2024		
	To-do list test case	Nhi	15/4/2024		
	Progress Tracker test case	Vũ	15/4/2024	100%	
	Group meeting	all	16/4/2024	100%	Modify database tables, went through each function’s test case
Preparation (for presentation and report)					
17	Group meeting	all	22/4/2024	100%	Discuss general outline for PowerPoint, re-format word file
18	Group meeting	all	29/4/2024	100%	Re-design database, re-design interfaces
19	Group meeting	all	3/5/2024	100%	Review word file, work on PowerPoint presentation

1.3 Rules on the group

- Contact channel:
  - Email
  - Zalo group
  - Google drive: folder “Word” includes sample file word and report cover, folder “Usecase” stores each function’s use case model file and image, folder “Test” stores each function’s flowchart and test case tables in file word. The word report file is located in the main interface of the group drive.
- Group meeting:
  - Time: 6:30pm every Monday evening
  - Venue: Cafe outside LRC every Monday evening from 7pm to 9:30pm

2. Software Requirement Specification

2.1 Overall Description

2.1.1 Product Perspective

1. Evidence shows that people who regularly plan their work or study clearly, often have a higher success rate in achieving their goals than those who do not. The purpose is to create convenience and be a tool to accompany people's daily lives in managing and planning as well as sharing schedules with other stakeholders. Based on that, we developed My Second Brain, a management and planner app.
2. My Second Brain is a completely new and independent software, with many useful and improved functions than most other products on the market. It's a cross-platform mobile application, which can run on different operating systems, for instance, IOS and Android.

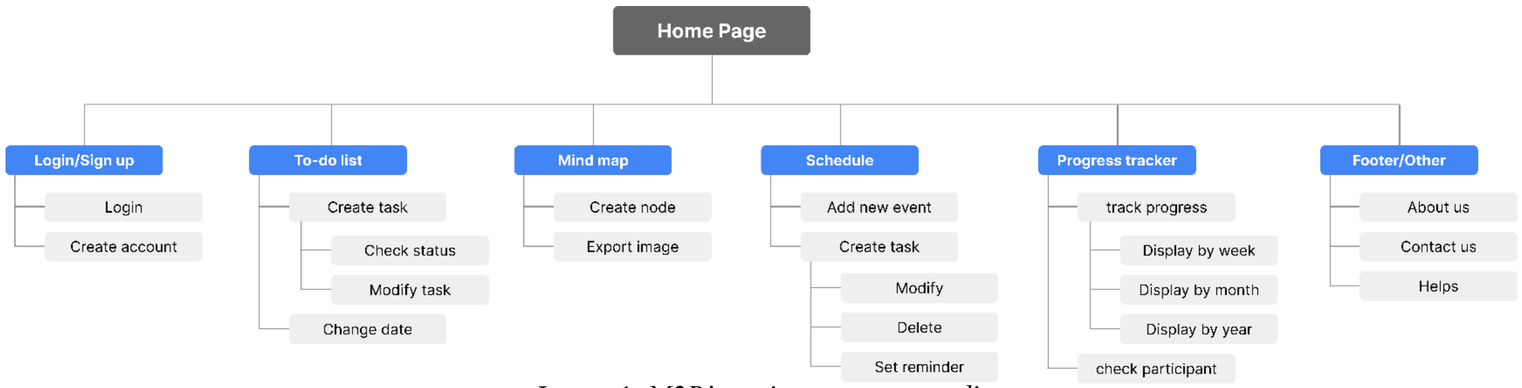


Image 1. M2B's major components diagram

2.1.2 Product Functions

1. Schedule: provides a helpful tool to help users schedule their activities by setting events at a certain time.
2. To-do list: list down the tasks on a day and manage your time to finish all of them.
3. Mind map: create a mind map to summarize all the content to help users grasp all the information.
4. Progress tracker: a managing plan implementation, thereby obtaining statistics on the level of plan completion to provide data for evaluating goals for each individual. There are some components in the progress tracker such as the tracking activity board, statistics about plan completion (data, diagram, etc.), and evaluating by week, month, and year.

2.1.3 User Classes and Characteristics

Our application is aimed at a class of users: people who use the app in their works or daily lives to manage their projects. Because using this application requires creating an account, which can be done using a Gmail account or by directly logging in with Facebook or Google credentials, the age of users must comply with the age policies of Facebook and Google.

2.1.4 Operating Environment

1. Operating systems for mobile devices: Samsung, Oppo, iPhone.
2. Operating systems and versions: Android 8.0 and up, IOS 14 or higher.
   APIs for integration: Google Calendar API.

2.1.5 Design and Implementation Constraints

1. Ensures the security of user data
   - Encryption at rest: Customer data is encrypted at rest using AES-256. Customer data is encrypted when on My Second Brain’s internal networks, at rest in Cloud storage, database tables, and backups.
   - Encryption in transit: Data sent in transit is encrypted using TLS 1.2 or greater.
   - The software requires devices to have at least 2 GB of RAM, and 200 MB of free memory to download.
2. Interfaces with other applications:
   The application has to integrate with the applications below to perform most of its functionalities:
   - Gmail: This integration enables users to create an M2B account and receive notifications via emails.
   - Facebook: This integration enables users to login directly with their account’s credentials.
   - Notification: the application should be able to integrate with the device’s notification system to send reminders directly to users’ phones.
3. Specific technologies, tools, and databases to be used
   - Encryption uses AES-256, TLS-12 or greater.
   - My Second Brain is hosted by Amazon Web Services and stores customer data using a combination of databases.
   - Anti-DDoS: My Second Brain leverages third-party applications for DDoS protection.
   - Data Center: My Second Brain is hosted on AWS, which handles data center physical security.
4. Language requirements
   - English or Vietnamese.
5. Programming languages
   - Python, Nodejs, React, Electron.
6. Communications protocols: HTTPS
7. Design conventions or programming standards:
   - For design conventions, the application should follow the MVVM model.
   - For programming standards:
   - Developers should follow the following naming conventions: use all lowercase letters and “_” to separate words for local variables and functions; use the prefix “G” for global variables.
   - Identified with a tab for each inner block entered.
   - Write comments for complex code blocks.
8. Safety and security considerations
   - User data are encrypted as presented in “Ensure the security of user data”.

9. Libraries and frameworks used must be compatible with HTTPS protocols.
10. Control functions: Data are backed up when the system goes wrong.

2.2 Specific Requirements

2.2.1 External Interface Requirements

1. User Interfaces

   The application interfaces will have the following common characteristics:
   - (1) The title will be displayed prominently at the top left corner of the interface. It will have a large font size and a clear, easily readable font style.
   - (2) Some action buttons will be placed horizontally with the title, typically on the right side. These buttons represent various functionality and actions within the application.
   - (3) The main action buttons, usually corresponding to key functions, will have a distinct dark blue color. They will stand out from other buttons, and their labels will be concise descriptions of the associated actions.
   - The dominant colors throughout the interface will be blue and white. These colors will be used consistently to maintain visual coherence and provide a cohesive look and feel. The application's interface ensures simplicity and ease of use.

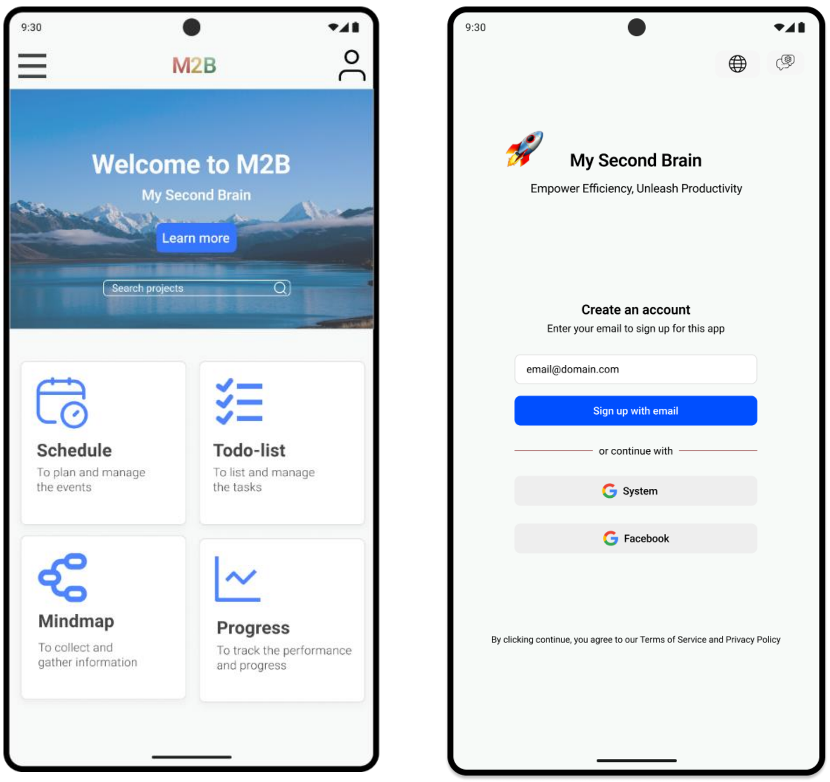

9:30
M2B
Welcome to M2B
My Second Brain
Learn more
Search projects
Schedule
To plan and manage the events
Todo-list
To list and manage the tasks
Mindmap
To collect and gather information
Progress
To track the performance and progress
9:30
My Second Brain
Empower Efficiency, Unleash Productivity
Create an account
Enter your email to sign up for this app
email@domain.com
Sign up with email
or continue with
System
Facebook
By clicking continue, you agree to our Terms of Service and Privacy Policy

Image 2. M2B's main and authentication interfaces

2. Hardware Interfaces
   - Supported device types: touchscreen devices (e.g., Smartphones, Tablets): The software interacts with the hardware through touch gestures, enabling users to navigate, manipulate, and interact with data. Users can perform actions such as scrolling, tapping, dragging, and pinch-to-zoom to control the software. The specific communication protocols to be used are not mentioned in the provided information.
3. Software Interfaces
   - Databases: MySQL
   - Operating Systems: React Native
   - Tools and libraries: Moment.js, React To-do-list, D3.js, mxGraph, draw.io,...
   - Incoming Data/Requests: The requests could be in the form of API calls or user interface interactions.
4. Communications Interfaces
   - Email Communication:
   - SMTP (Simple Mail Transfer Protocol) for sending emails and possibly
   - IMAP (Internet Message Access Protocol) or POP3 (Post Office Protocol) for receiving emails.
5. Web Browser Communication:
   - HTTP POST requests to add tasks to the to-do list, HTTP PUT requests to update a mind map, and HTTP DELETE requests to remove a diagram component.
6. Network Server Communication Protocols:
   - RESTful APIs (using HTTP or HTTPS) for CRUD (Create, Read, Update, Delete) operations with server resources.
   - FTP (File Transfer Protocol) or SFTP (SSH File Transfer Protocol) for transferring files to and from servers
   - Electronic Forms: HTTP POST requests with data encoded in formats such as JSON (JavaScript Object Notation). It's commonly used for exchanging structured data between the app and server for functionalities such as syncing schedules, to-do lists, mind maps, and diagrams.

2.2.2 Functional Requirement

Authentication

- o Sign up: create an M2B account for the user.
- o Sign in: user can login to the application with Google account or created M2B account.

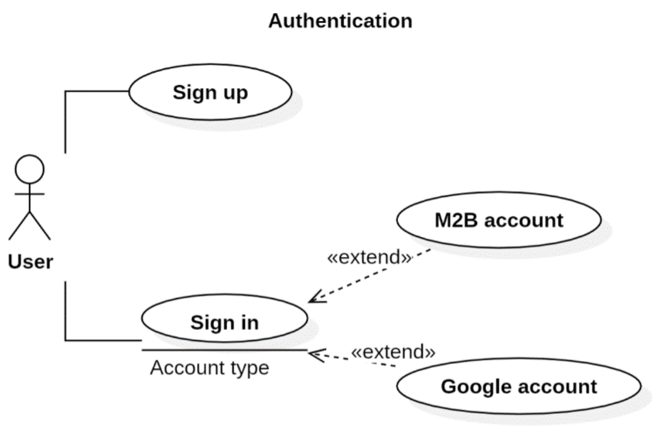


Image 3. Authentication use case

Use Case: Sign up	**ID:** UC001
Main actor: User	**Priority:** Essential
Brief description: *A valid user sign up an M2B account with their email.*	
Trigger: User	**Type:** external
Relationship: **+Association:** User – Sign up an M2B account **+Include:** None **+Extend:** None **+Generalization:** None	
Normal flow: 1. The user enters their email taps on “Sign up with email button”. 2. A new interface appears, allowing user to enter username and password. 3. System sends an authentication mail to the email entered. 4. The user verifies their email by clicking “confirm” button in the email received. 5. System creates the account and returns to Authentication interface.	
Alternative Flows: Sign in with Facebook	
Exceptional flows: 1. At step 1, the user enters an invalid email. The system will announce that the email is invalid and require the user to re-enter their email. 2. At step 2, the user leaves username or password blank, or enters a different password in “Re-enter password” box, system will announce an error and ask the user to re-enter. 3. At any time, the user may choose to cancel creating an account. At which point, the processing is discontinued, and the system returns to the Authentication interface.	

Use Case: Sign in	**ID:** UC002
Main actor: User	**Priority:** Essential
Brief description: *A valid user sign in to the system with their account.*	
Trigger: User	**Type:** external
Relationship: **+Association:** User – sign in to the system. **+Include:** None **+Extend:** None **+Generalization:** None	
Normal flow: 1. The user taps on "System" button at the Authentication interface. 2. A new interface appears, allowing user to enter username and password. 3. System checks username and password. 4. System leads user to main interface of the application after sign in successfully.	
Alternative Flows: Sign in with Facebook 1. The user taps on the "Sign in with Facebook" button on the authentication interface. 2. The system initiates the authentication process and prompts the user to log in using their Facebook credentials. 3. Upon successful authentication, the system redirects the user to the main interface of the application.	
Exceptional flows: At step 3, if the user's credential is invalid, the system will inform an error.	

2.2.2.1 Schedule

This subsection outlines 2 features that Schedule functionality should performs properly:

1. Event creation:

   Users can create a new event in the Schedule interface.

   The application should provide an intuitive interface for entering the event's details such as time, duration, destination and note.

2. Manage existing event:

   Modify: Users can modify their task by changing its details or add more information.

   Event deleting: Users can easily delete the events when they are no longer needed.

   Remider: the app sends reminders at a user-customized interval for upcoming activities.

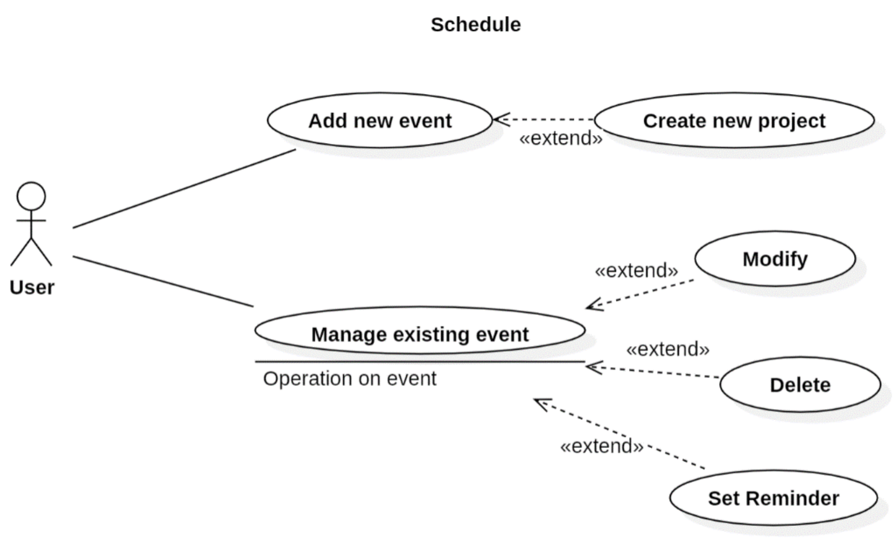


Image 4. Use case: Schedule

Use Case: Create New Project	**ID:** UC003
Main actor: User	**Priority:** Essential
Brief description: *A valid user logged on to the system.* She/he wants to create a new project.	
Trigger: User	**Type:** external
Relationship: **+Association:** User – Create New Project **+Include:** None **+Extend:** None **+Generalization:** None	
Normal flow: 1. The user taps "New project" button. 2. A new interface appears, allowing users to enter project's information. 3. The user enters information for the project, including project name, description, begin time and end time, and taps "Add" button. 4. System checks entered information. 5. System computes change and add the new project to the list of existing projects.	
Exceptional flows: 1. At any time, the user may choose to cancel creating the project. At which point, the processing is discontinued, no project is created, and the system returns to the previous interface which the user was working. 2. At step 4, if the user doesn't enter the project's title, select a begin time that is in the past, or chooses an ending time that is earlier than the beginning time, the system will display an error message and prompt the user to re-enter the required information.	

Use Case: Add New Event	**ID:** UC004
Main actor: User	**Priority:** Essential

Brief description: *A valid user logged on to the system.* She/he wants to create a new event.	
Trigger: User	**Type:** external
Relationship: **+Association:** User – Create New Event **+Include:** Add new project **+Extend:** None **+Generalization:** None	
Normal flow: 1. The user taps "Add new event" button in Schedule's main interface. 2. A list of existing projects drops down. 3. The user chooses a project to create a new event in it. 4. A new interface appears, allowing the user to enter the event's information. 5. The user enters information, including "Event's title", start time, end time, and note (optional) for the event and taps "Add" button. 6. System checks entered information. 7. System adds the event to chosen project. 8. System returns to Schedule function's main interface. The newly created event is displayed on top.	
Alternative Flows: (branch after step 2) 1. The new event is not in any existing project, the user creates a new project for that event. Include "Create New Project". 2. The newly created project appears on the top of the dropdown existing project list. 3. Return to step 3.	
Exceptional flows: 1. At any time, the user may choose to cancel creating the event. At which point, the processing is discontinued, no event is created, and the user will be returned to the main interface of the Schedule function. 2. At step 6, if the user doesn't enter the event's title, select a start time that is in the past, or chooses an ending time that is earlier than the beginning time, the system will display an error message and prompt the user to re-enter the required information.	

Use Case: Manage existing event	**ID:** UC005
Main actor: User	**Priority:** Essential
Brief description: *A valid user logged on to the system.* She/he wants to perform an operation on an existing event.	
Trigger: User	**Type:** external
Relationship: **+Association:** User – Create New Event **+Include:** None **+Extend:** None **+Generalization:** None	
Normal flow:	

1. The user taps on the three dots button on the right of the event he/she wants to manage.
2. A small window with three options appears, including “Modify”, “Delete” and “Set reminder”.
3. The user chooses “Modify” options.
4. A new interface appears, allowing user to enter event’s new information, including “Event’s title”, start time, end time, and note (optional).
5. The user enters information for the event and taps “Modify” button.
6. System check entered information.
7. System updates information for the event.
8. System returns to Schedule function’s main interface. The event is still displayed at its previous position, but with edited information.

Alternative Flows: (branch after step 2)
Delete event:

1. The user chooses “Delete” option.
2. System displays message asking for confirmation.
3. The user confirms his/her option.
4. System deletes the event.
5. System returns to Schedule main interface. The chosen event is deleted and no longer appears.

Alternative flow: (branch after step 2)

1. *The user chooses to cancel the deletion.*
2. *Systems returns to Schedule function’s main interface.*

Set reminder:

1. The user chooses “Set reminder” option.
2. A new interface appears, allowing the user to set time for reminder.
3. The user set reminding time and taps “Set” button.
4. System records and sends reminders according to the time set by the user.

Exceptional flows:

1. At any time, the user may choose to cancel creating the event. At which point, the processing is discontinued, no event is created, and the system returns to the Schedule function’s main interface.
2. At step 6, if the user doesn’t enter the event’s title, select a start time that is in the past, or chooses an ending time that is earlier than the beginning time, the system will display an error message and prompt the user to re-enter the required information.

2.2.2.2 To-do list

1. Task List:
   - A clean list view with checkboxes for completed tasks.
   - Priority indicators (high, medium, low).
   - Due dates and times for each task.
   - Option to filter tasks based on status or priority.
2. Task Details:
   - Clicking on a task opens a detailed view with additional notes.
   - Ability to set reminders or deadlines for tasks.
   - Options to edit, delete, or mark tasks as complete.

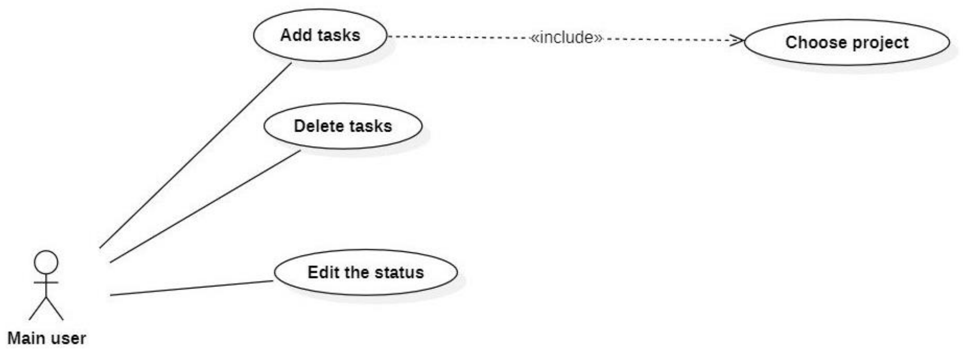


Image 5. To-do list use case

Use Case: Choose Project	**ID:** UC006
Main actor: User	**Priority:** Essential
Brief description: A valid user logged on to the system. He/she wants to choose an existing project.	
Trigger: User **Type:** Human actor	
Relationship: **+Association:** User – Choose project. **+Include:** None **+Extend:** None **+Generalization:** None	
Normal flow: 1. In To-do list main interface, the user taps on “Choose project” button. 2. A dropdown menu will apear, allowing user choose the project they need.	

Use Case: Add tasks	**ID:** UC007
Main actor: User	**Priority:** Essential
Brief description: *A valid user logged on to the system.* She/he wants to add the tasks.	
Trigger: User	**Type:** External

Relationship: **+Association:** User – Create New Task **+Include:** Choose project **+Extend:** None **+Generalization:** None
Normal flow: 1. Include "Choose project". 2. The user taps on "Add task" button. 3. A new blank space will appear, displaying the sequence number of the new task, allowing the user to enter the task name and type. 4. System checks entered information. 5. System creates the task and show interface.
Exceptional flows: 1. At any time, the user may choose to cancel creating the task. At which point, the processing is discontinued, no task is created, and the system returns to the To-do list function's main interface. 2. At step 5, if the user left task name empty, system will inform an error message.

Use Case: Edit status	**ID:** UC008
Main actor: User	**Priority:** Essential
Brief description: *A valid user logged on to the system.* She/he wants to edit the task' status.	
Trigger: User	**Type:** External
Relationship: **+Association:** User – Edit the task' status **+Include:** None **+Extend:** None **+Generalization:** None	
Normal flow: 1. The user taps on the task he/she wants to manage. 2. A small window with three options appears, including "Done", "Doing" and "Not started". 3. System updates information for the task 4. System returns to Todo-list function's main interface. The task is displayed with updated status.	
Exceptional flows: At any time, the user may choose to cancel edit the task status. At which point, the processing is discontinued. The task status is not updated.	

Use Case: Delete	**ID:** UC009
Main actor: User	**Priority:** Essential
Brief description: A valid user logged on to the system. He/she wants to delete the task.	
Trigger: User	**Type:** External

Relationship: **+Association:** User – Delete the task **+Include:** None **+Extend:** None **+Generalization:** None
Normal flow: 1. The user presses and holds the task he/she wants to delete. 2. System displays message asking for confirmation. 3. The user confirms deletion. 4. System deletes the chosen task. 5. System returns to To-do list function's main interface.
Alternative flows: (branch after step 2) 1. The user chooses "No" in confirmation message. 2. System returns to To-do list function's main interface, the task is not deleted.

2.2.2.3 Mind map

1. Mind Map Canvas:
   - Blank canvas for creating mind maps.
   - Intuitive drag-and-drop functionality for adding nodes.
   - Nodes with customizable colors, icons, and text.
   - Auto-arrange or manual positioning options.
2. Editing and Navigation:
   - Double-clicking a node opens an editor for text and additional details.
   - Zoom in/out and pan features for navigating larger mind maps.
   - Connecting nodes with lines to represent relationships.

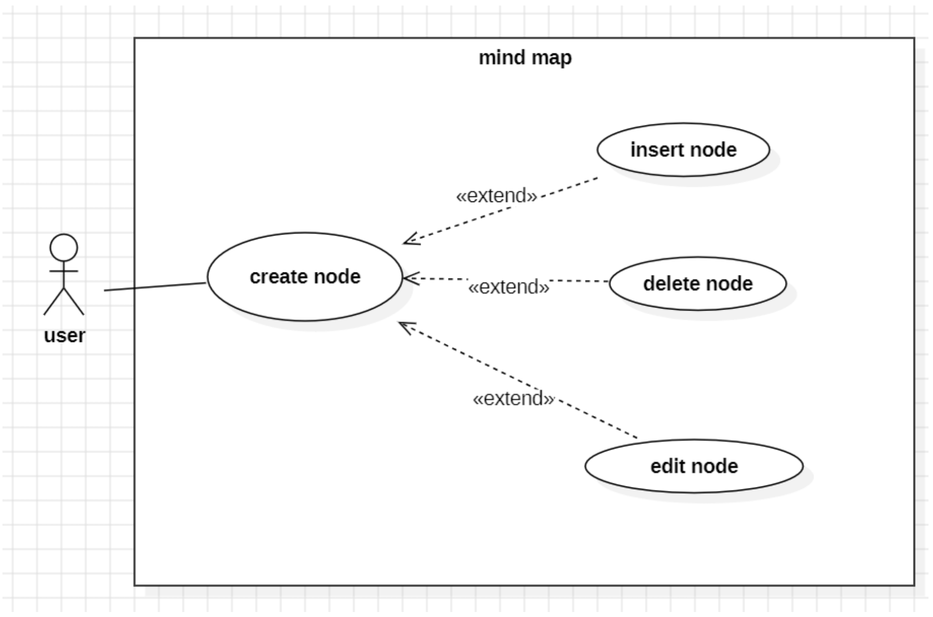


Image 6. Use case: Mind map

Use Case: Create New node	**ID:** UC010
Main actor: User	**Priority:** Essential
Brief description: *A valid user logged on to the system.* She/he wants to create a new node	
Trigger: User	**Type:** external
Relationship: **+Association:** User – Create New node **+Include:** None **+Extend:** None **+Generalization:** None	
Normal flow: 1. The user taps "Add” button. 2. A new interface appears, allowing users to enter central topic 's information (name of Mind map) 3. System checks entered information. 4. System computes change and add the new project to the list of existing projects. 5. The user taps “child node” or “sibling node” button in the bottom left corner of the screen. 6. If user click “child node”, a node at the same level as the root node appears. if user click “sibling node”, a node appears on another branch of the root node. 7. User can withdraw the node at the top of the navbar or custom them in menu button.	
Exceptional flows: If the user does not click whether the child or sibling button, the page will still be saved	

2.2.2.4 Progress Tracker

1. Showing project progress, evaluating the level of completion of the project by day, month, or year, and assessing it.
2. Displaying information about members who contribute to this project, such as email, name, department, etc.

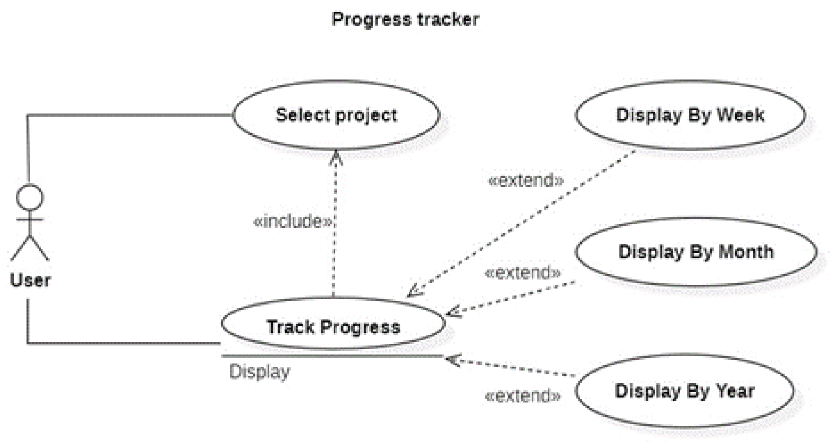


Image 7. Use case: Track Progress

Use Case: Select Project	**ID:** UC011
Main actor: User	**Priority:** Essential

Brief description: *A valid user logged on to the system. They* wants to supervise or track information of the project's progress.
Trigger: User **Type:** Human actor
Relationship: **+Association:** User – Track the project's progress and information. **+Include:** None **+Extend:** None **+Generalization:** None
Normal flow: **1.** The user press "select Project" button. **2.** A dropdown menu will apear, allowing user select the Project they need.

Use Case: Track Process	**ID:** UC012
Main actor: User	**Priority:** Essential
Brief description: *A valid user logged on to the system. They* wants to supervise or track information of the project's progress.	
Trigger: User **Type:** Human actor	
Relationship: **+Association:** User – Track the project's progress and information. **+Include: Select project.** **+Extend:** Display a dashboard by week, month or year **+Generalization:** None	
Normal flow: **1.** Include "Select project". **2.** Check if there is a project already selected. **3.** If all check is successful, display all the information that are relevant with that project. **4.** The user chooses how to display the dashboard site. The choices are by week, by month, and by year.	
Exceptional flows: None	

2.2.3 Nonfunctional Requirements

1. Performance
   - Time to search information: 5 (s)
   - Data sharing time: 15 (s)
   - Average processing time: 10 (s)
   - Average response time: 5 (s)
   - Number of concurrent users: 1000
   - Maximum database capacity of 1TB
   - Always ensure executable 20 shares per second.
2. Reliability
   - Except for maintenance or incidents, the system will be accessible 24/7.

- Maintenance is performed once a month, typically for approximately 2 hours each time.
- Recovery Time Objective: Estimated at 2 hours.
- Recovery Point Objective: Estimated at 24 hours.

3. Safety and Security
   - When using the system, the user will be asked to create an account, which contains information about name, address, phone number, as well as payment account, which is important information that the user can be the target of being beaten up by bad guys to commit illegal acts. To completely avoid such incidents from happening, we always put user safety and data security first.
4. Security:
   - My Second Brain is hosted on Amazon Web Services (AWS), one of the major cloud service providers.
   - The system will store information that users use for backup on the database, and when sharing it with others, the user will be asked to verify that information.
5. Privacy:
   - Data stored on the database will be kept confidential, we will not provide that information to any third party, except for using statistical data from your account. you aim to perform distinct system functions.
   - Data Access Level: Internal (i.e. we will only ever access your data for the purposes of troubleshooting problems or recovering content on your behalf.)
6. Integrity: prevent unauthorized access to accounts, verify user identity.
7. Adaptability:
   - Make sure the software can work with new updates of operating systems without errors.
   - The application size will always be scaled to correspond to the size of the mobile phone.
   - The system will always monitor and upgrade the database to meet the needs of data storage for the number of users.
8. Portability:
   - Supported on many mobile platforms such as IOS, Android.
   - Available in computer versions on Windows, MacOS, and Unix operating systems.
   - Allows data synchronization when used on different platforms.
9. Business Rules

ID	Rule Definition	Source
BR-11	Ensuring security and privacy of individual data.	Corporate security policy

BR-12	In case of system failure, the data must be backed up and restored.	Corporate security policy
BR-33	The backups and production data are encrypted and are monitored and alerted.	Corporate security policy
BR-34	Customer data is encrypted at rest using AES-256	Corporate security policy

3. Software Design

3.1 Application Architecture

3.1.1 Application Architecture Diagram

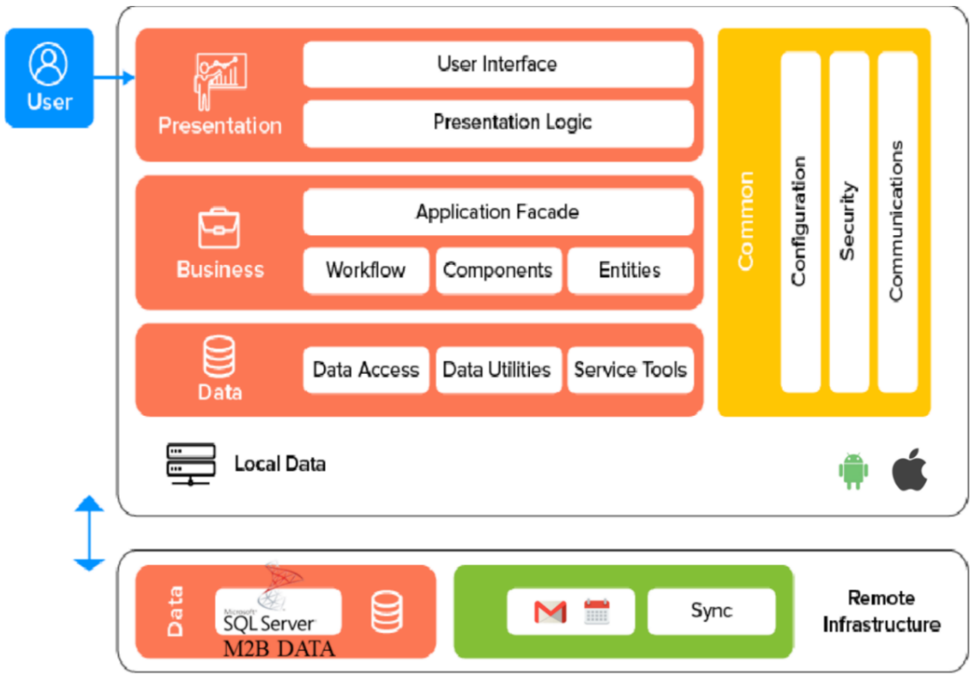


Image 8. M2B Architecture Diagram

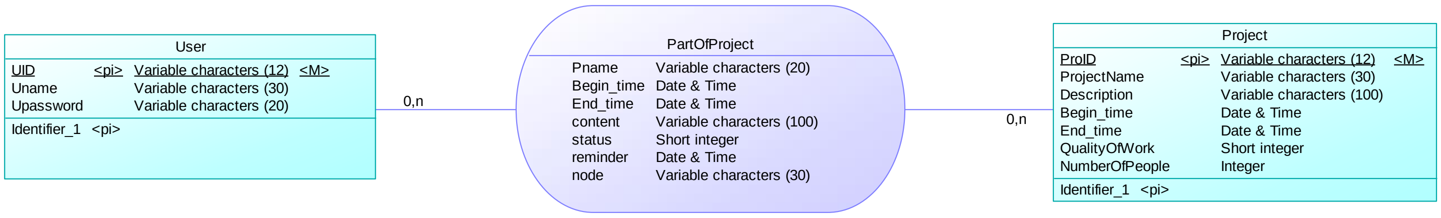
3.1.2 Description

1. M2B Database: Stores users' personal data such as documents, schedules, project information, account information.
2. Mail: Create user account. Send notification about project's progress, prompt user, connect user to assistant technology to support whenever their need.
3. Google Schedule: support for "schedule" function works properly.

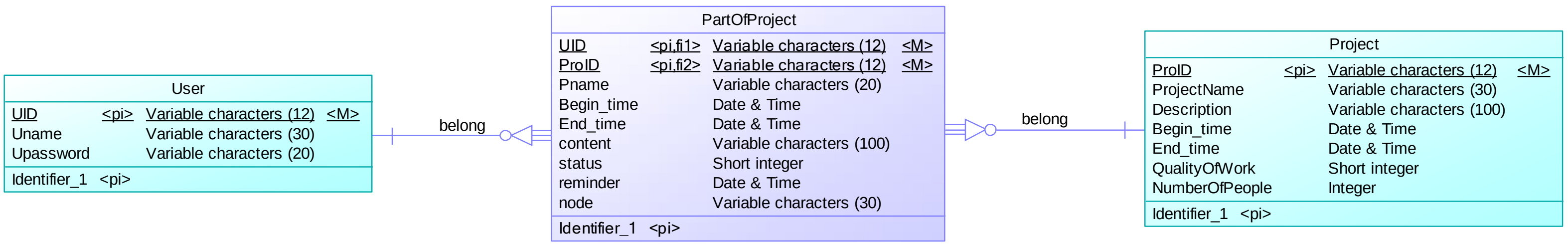
3.2 Data Design

3.2.1 Data Description

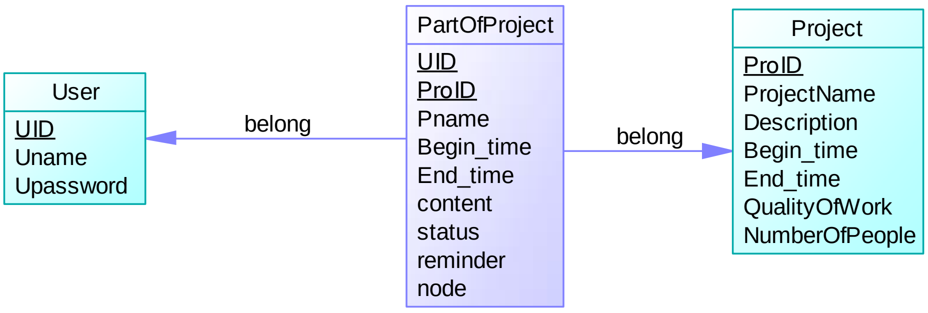
Conceptual Data Model


User
UID <pi> Variable characters (12) <M>
Uname Variable characters (30)
Upassword Variable characters (20)
Identifier_1 <pi>
0,n
PartOfProject
Pname Variable characters (20)
Begin_time Date & Time
End_time Date & Time
content Variable characters (100)
status Short integer
reminder Date & Time
node Variable characters (30)
0,n
Project
ProID <pi> Variable characters (12) <M>
ProjectName Variable characters (30)
Description Variable characters (100)
Begin_time Date & Time
End_time Date & Time
QualityOfWork Short integer
NumberOfPeople Integer
Identifier_1 <pi>

Logical Data Model


User
UID <pi> Variable characters (12) <M>
Uname Variable characters (30)
Upassword Variable characters (20)
Identifier_1 <pi>
belong
PartOfProject
UID <pi,fi1> Variable characters (12) <M>
ProID <pi,fi2> Variable characters (12) <M>
Pname Variable characters (20)
Begin_time Date & Time
End_time Date & Time
content Variable characters (100)
status Short integer
reminder Date & Time
node Variable characters (30)
Identifier_1 <pi>
belong
Project
ProID <pi> Variable characters (12) <M>
ProjectName Variable characters (30)
Description Variable characters (100)
Begin_time Date & Time
End_time Date & Time
QualityOfWork Short integer
NumberOfPeople Integer
Identifier_1 <pi>

Physical Data Model



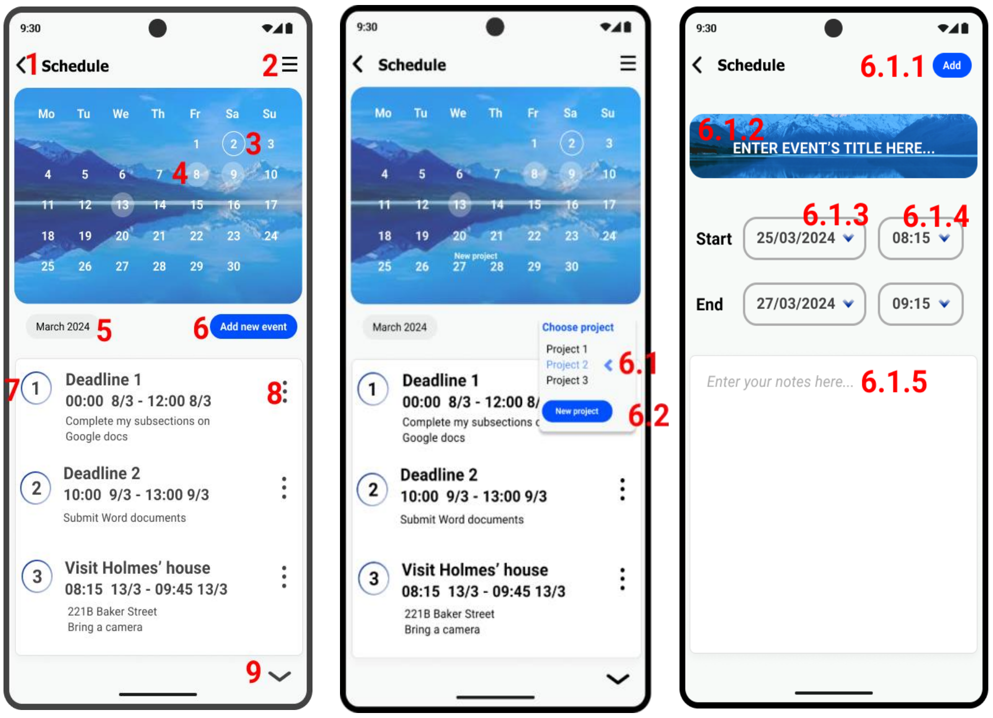
3.2.2 Data Dictionary

Element Name		Description
User		Represents a user account
UID	Varchar(12)	Unique identifier of the user account, limited to 12 characters, includes only lowercase letter and numbers.
Uname	Varchar(30)	Containing username, limited to 30 characters.
Upassword	Varchar(20)	Containing user password, limited to 20 characters.
Project		Represents a project created by the user.
ProID	Varchar (12)	Unique identifier of a project, limited to 12 characters, includes only lowercase letter and numbers.
ProjectName	Varchar(30)	Containing project name
Description	Varchar(100)	Defined by user to describe their project
Begin_time	Date&time	The beginning date and time of the project.
End_time	Date&time	The ending date and time of the project, the value of this attribute must be greater than the value of "Begin_time".
QualityOfWork	Short integer	Demonstrates the level of completion and responsiveness to project needs, with the range from 1 to 100.
NumberOfPeople	integer	The number of people participating in the project.
PartOfProject		Represents a part of a project, which is a schedule, a to-do list or a mind map
Pname	Varchar(20)	Name of the part, limited to 20 characters.
Begin_time	Date&time	The beginning date and time of the part.
End_time	Date&time	The ending date and time of the part, the value of this attribute must be greater than the value of "Begin_time".
Content	Varchar(100)	Gives more description
Status	Short integer	Shows the level of completion of the task (haven't started, in progress or completed, corresponding to the values -1,0,1, respectively).
Reminder	Date&time	The reminder time for an event.
Node	Varchar(30)	Represents a node in a mind map

3.3 Detailed Design

3.3.1 Schedule

1. Purpose: the Schedule function helps user to plan and manage their events effectively.
2. Interface:



Deadline 1
00:00 8/3 - 12:00 8/3
Complete my subsections on Google docs

Deadline 2
10:00 9/3 - 13:00 9/3
Submit Word documents

Visit Holmes' house
08:15 13/3 - 09:45 13/3
221B Baker Street
Bring a camera

Start 25/03/2024 08:15
End 27/03/2024 09:15

Image 9. Schedule interfaces

3. The components of interface:

No.	Controller type	Default value	Note
1	Button		Go back to main menu
2	Button		Open more options, such as find events or share schedule
3	Icon		Current day is white circled
4	Icon		Days with events are highlighted in opaque white
5	Combo box	Current month	Display content according to the currently displayed month
6	Button		Add new event

6.1	Select box (Appear after tapping on button 6)	Newest project	Choose a project from existing project list
6.1.1	Button		Submit mechanism for creating an event with the corresponding user-entered information.
6.1.2	Textbox		Enter event’s title
6.1.3	Date and Time picker	The current day	Choose a day
6.1.4	Date and Time picker	0 am	Choose time
6.1.5	Textbox		Enter more description or note for the event
6.2	Button		Create a new project
7	Card		The most recently created event is displayed at the top of the list of existing events.
8	Menu		Enables user to choose an operation for the event, including “modify”, “delete” and “set reminder”
9	Button		Swipes down if there are still available events left

4. Data to be used:

No.	**Table name / Data structure**	**Method**			
		Add	**Modify**	**Delete**	**Query**
1	User				x
2	PartOfProject	x			
3	Project				x

5. Process

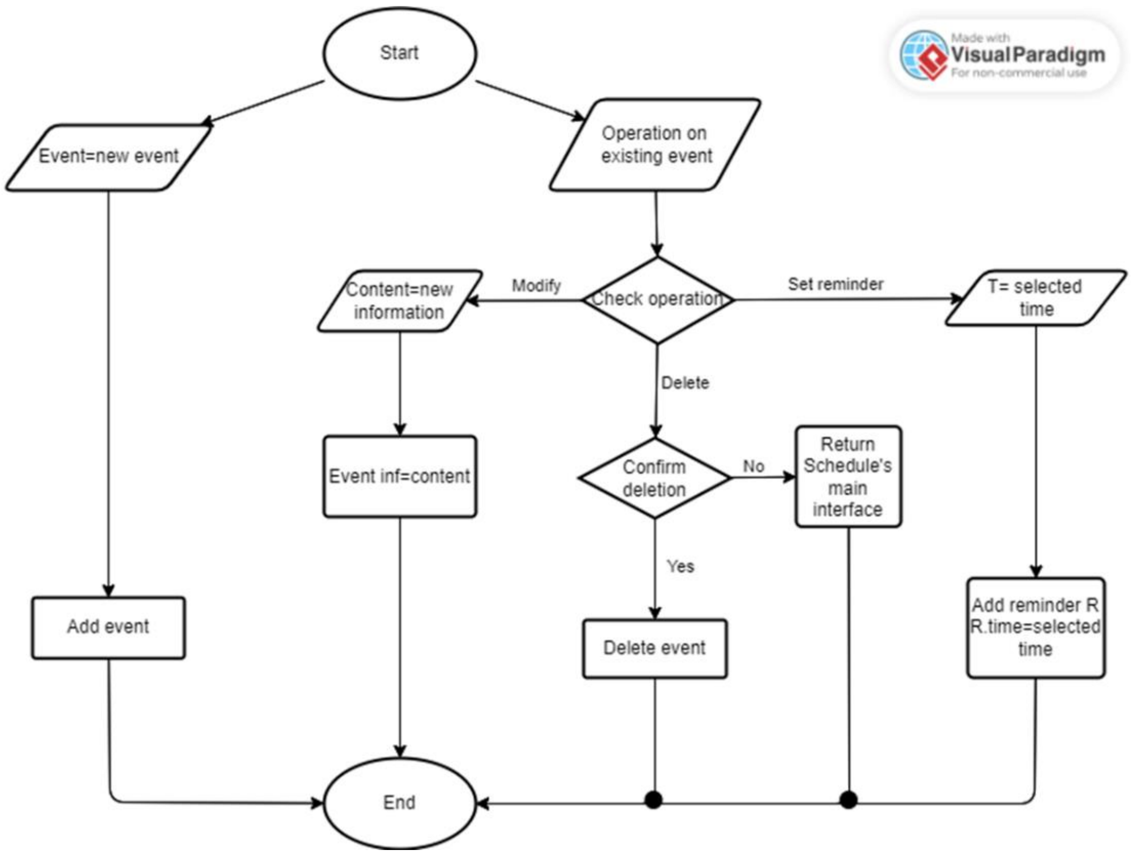


Image 10. Schedule flow chart

3.3.2 To-do list

1. Purpose: the To-do list function helps us to list and manage the tasks
2. Interface:

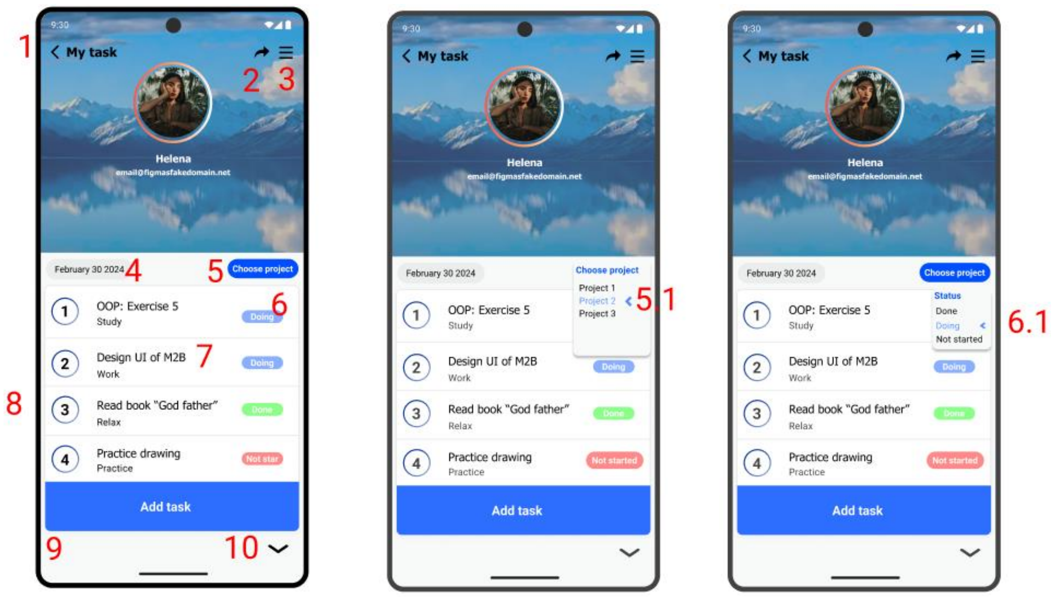


Image 11. To-do list interfaces

3. The components of interface:

No.	Controller type	Default value	Note
1	Button		Go back to main menu
2	Button		Share
3	Button		Open more options as set up to the to-do list
4	Textbox	Date format	Date display
5	Button		Project select list
5.1	Select box (Appear after tapping on button 5)	Newest project	Choose a project from existing project list
6	Button		Display the status tasks
6.1	Select box (Appear after tapping on button 6)		Choose a status of task, there're not started, doing and done.
7	Textbox		Display tasks' content
8	Icon		Display numbering of the tasks
9	Button		Add task
10	Slider		Scroll down if there are still more content

4. Data to be used:

No.	Table name / Data structure	Method			
		Add	Modify	Delete	Query
1	User				x
2	PartOfProject	x	x	x	
3	Project				x

5. Process:

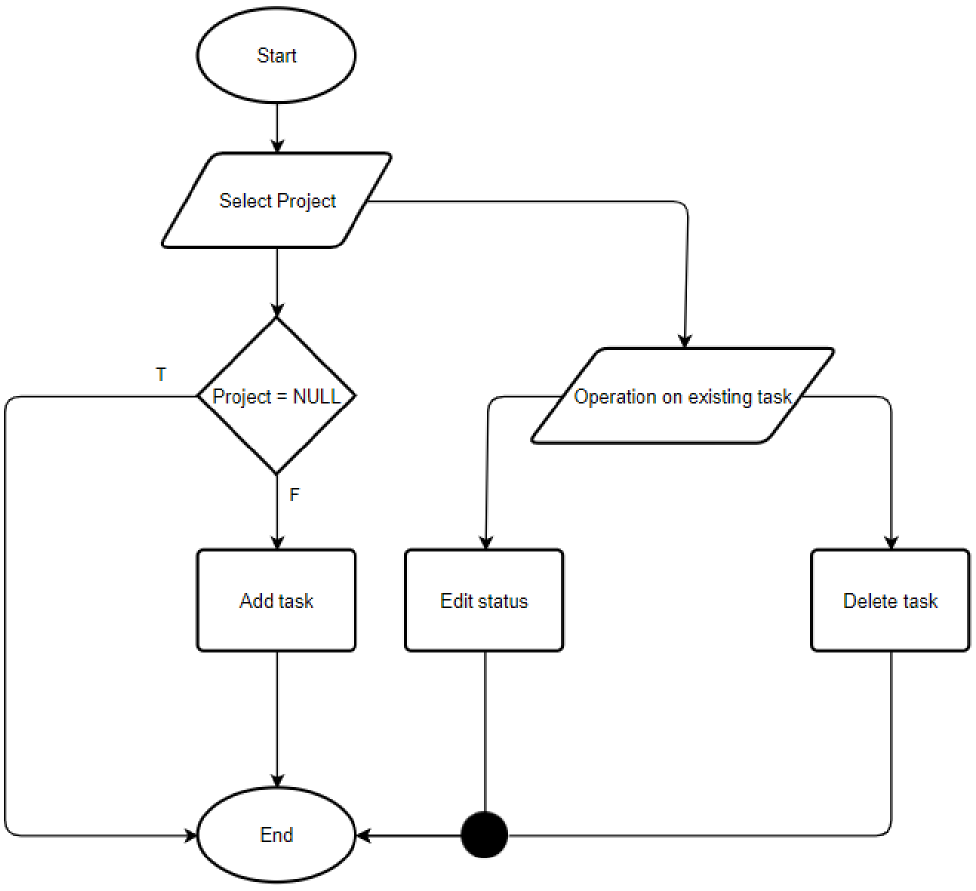


Image 12. To-do list flowchart

3.3.3 Mind map

1. Purpose: help user collects and gather the information together
2. Interface:

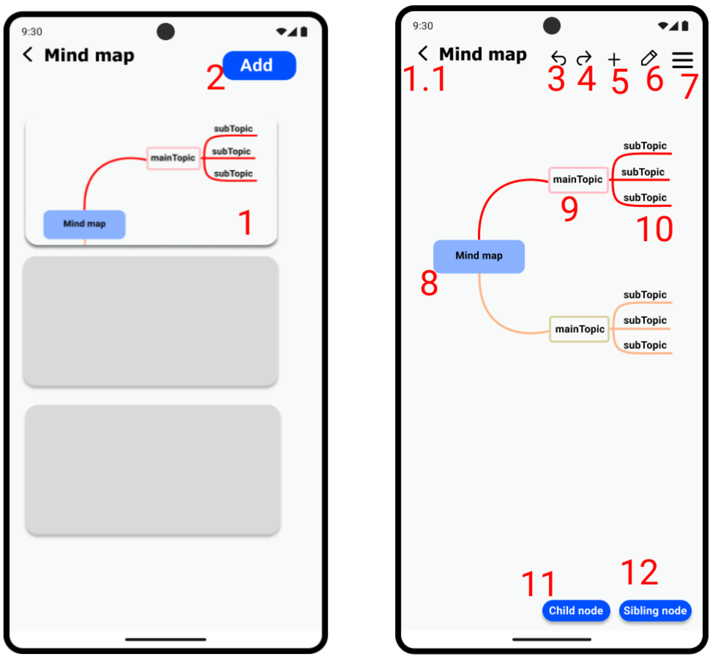


Image 13. Mind map interfaces

3. The components of interface:

No.	Controller type	Default value	Note
1	Button		Current created page
1.1	Button		Return the created page
3	Button		Undo create node
4	Button		Return the previous step when click Undo
5	Button		Insert links, images, etc.
6	Button		Custom node
7	Button		Menu button: export page, share, ...
8	Text		Display central node
9	Text		Display child node
10	Text		Display sibling node
11	Text		Create child node (main topic)
12	Button		Create sibling node (subtopic)
2	Button		Add a new page

4. Data to be used:

No.	Table name / Data structure	Method			
		Add	**Modify**	**Delete**	**Query**
1	User				x
2	PartOfProject	x	x	x	
3	Project				x

5. Process:

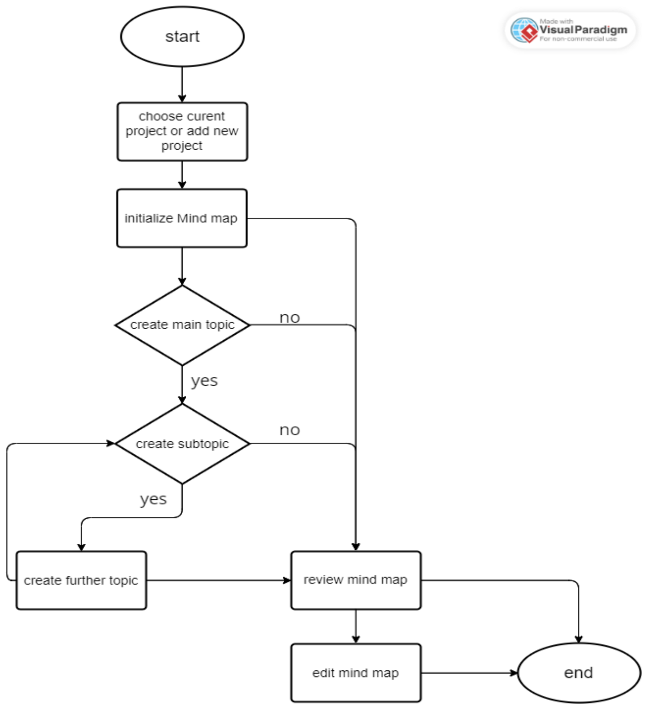


Image 14. Mind map flow chart

3.3.4 Progress tracker

1. Purpose: Allow user tracking their progress, statistic their project's performance, and the information of participants.
2. Interface:

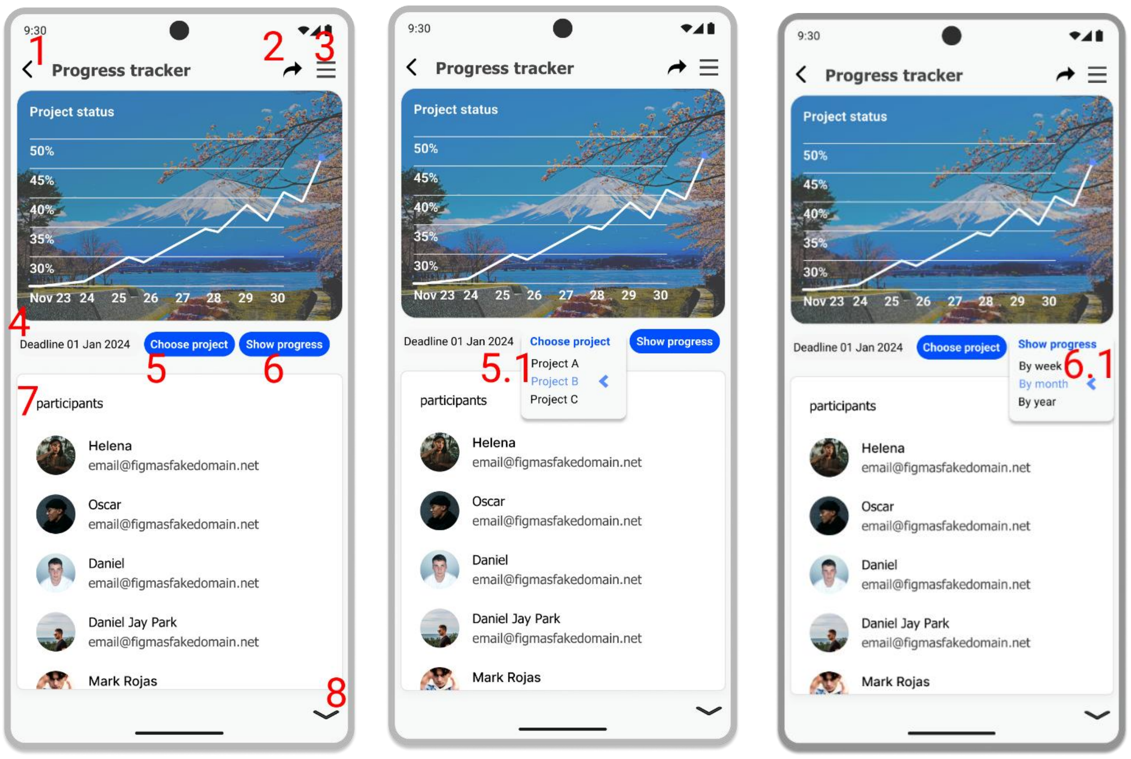


Image 15. Progress tracker interface

3. The components of interface

No.	Controller type	Default value	Note
1	Button		Go back
2	Button		Share the progress's infomation
3	Button		Open menu tab
4	Textbox	Date format	Display Date of project
5	Button		Choose the specific project.
5.1	Select box (Appear after tapping on button 5)	Newest Project existing in the system	Choose a project from existing project list
6	Button		Choose the specific display of chart
6.1	Select box (Appear after tapping on button 6)	By month	Choose display format by week, month, year.
7	Form		Display participants information
8	Button		Scroll to the end of this page

4. Data to be used

No.	Table name / Data structure	Method			
		Add	Modify	Delete	Query
1	User				x
2	PartOfProject				x
3	Project				x

5. Process:

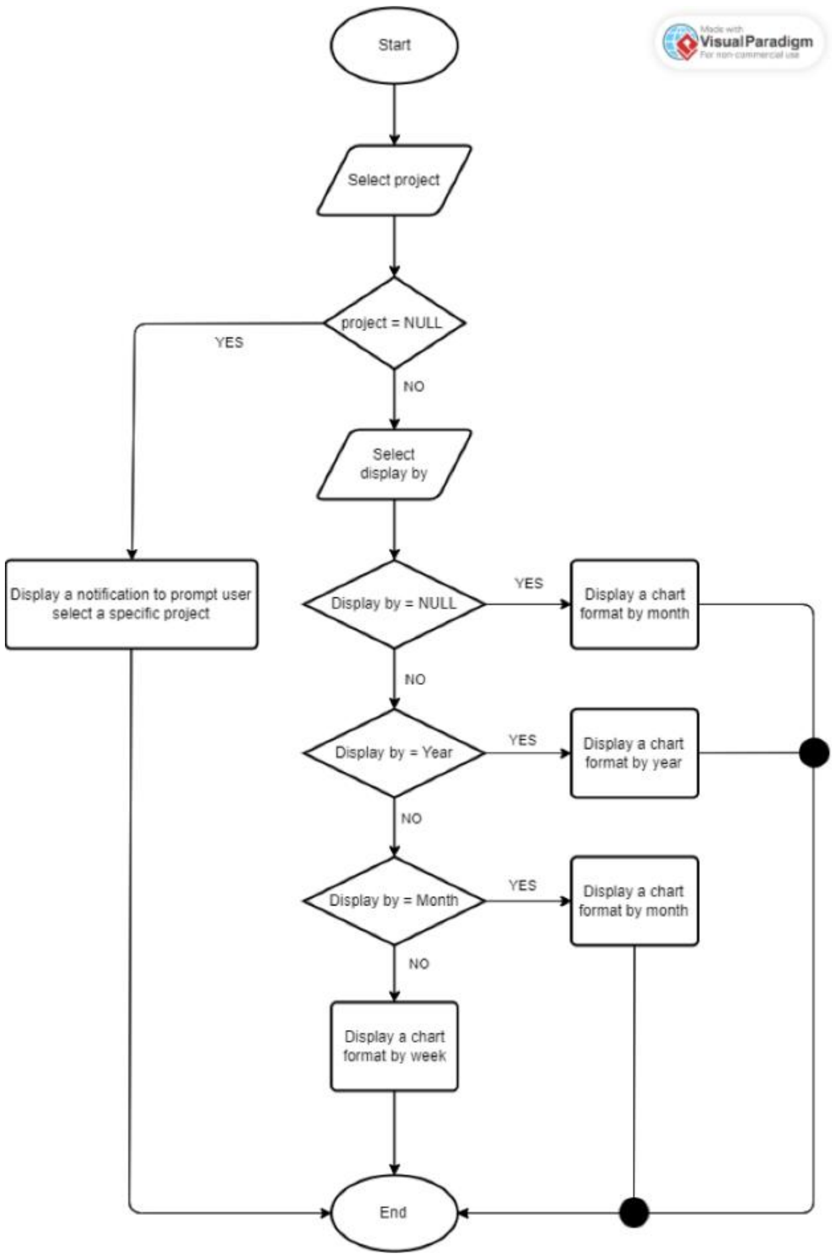

Start
Select project
project = NULL
YES
NO
Select display by
Display a notification to prompt user select a specific project
Display by = NULL
YES
Display a chart format by month
NO
Display by = Year
YES
Display a chart format by year
NO
Display by = Month
YES
Display a chart format by month
NO
Display a chart format by week
End

Image 16. Progress tracker flow chart

6. Constraints: show progress button must contain 3 String value by month, by week, by year.

3.4 References to Requirements

No.	**Design**	Functional Requirement
	Authentication	Authentication
1	Schedule	Function 1
2	To-do list	Function 2
3	Mind map	Function 3
4	Progress tracker	Function 4

4. Unit Test

4.1 Test 1 - Schedule

Test ID:	M2BT1
Test description:	Handles various user operations correctly and performs the corresponding actions as expected.
Created by:	Kim Tran
Date of creation:	04/09/2024
Date of review:	
Test priority:	
Pre-requisite:	
Post-requisite:	

4.1.1 Drawing the flow graph

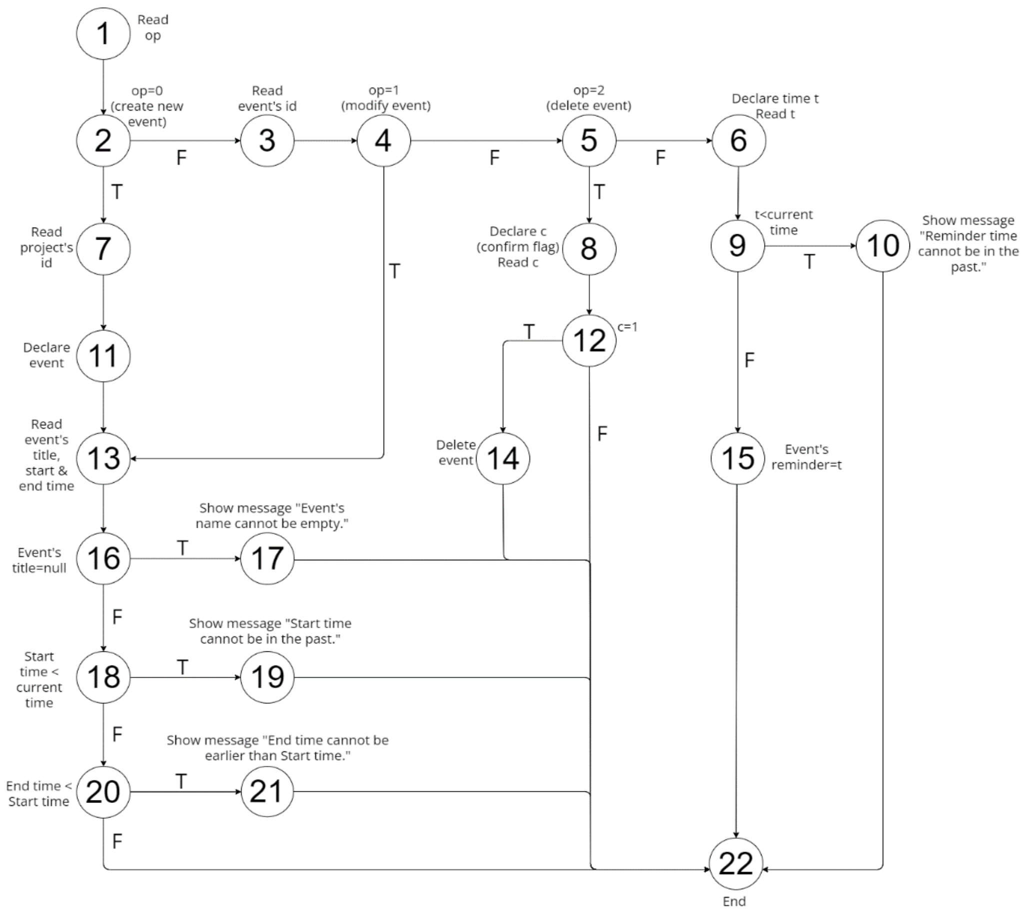

Read op
op=0 (create new event)
Read event's id
op=1 (modify event)
op=2 (delete event)
Declare time t Read t
Read project's id
Declare c (confirm flag) Read c
t<current time
Show message "Reminder time cannot be in the past."
c=1
Declare event
Read event's title, start & end time
Delete event
Event's reminder=t
Show message "Event's name cannot be empty."
Event's title=null
Show message "Start time cannot be in the past."
Start time < current time
Show message "End time cannot be earlier than Start time."
End time < Start time
End

Image 17. Schedule flowgraph

4.1.2 Determining a basis set of independent paths

V = P + 1 = 8 + 1 = 9
Nodes (P): (2), (4), (5), (9), (12), (16), (18), (20)

1: (1) -> (2) -> (7) -> (11) -> (13) -> (16) -> (17) -> (22)
2: (1) -> (2) -> (7) -> (11) -> (13) -> (16) -> (18) -> (19) -> (22)
3: (1) -> (2) -> (7) -> (11) -> (13) -> (16) -> (18) -> (20) -> (22)
4: (1) -> (2) -> (7) -> (11) -> (13) -> (16) -> (18) -> (20) -> (21) -> (22)
5: (1) -> (2) -> (3) -> (4) -> (13) -> (16) -> (18) -> (20) -> (22)
6: (1) -> (2) -> (3) -> (4) -> (5) -> (8) -> (12) -> (22)
7: (1) -> (2) -> (3) -> (4) -> (5) -> (8) -> (12) -> (14) -> (22)
8: (1) -> (2) -> (3) -> (4) -> (5) -> (6) -> (9) -> (10) -> (22)
9: (1) -> (2) -> (3) -> (4) -> (5) -> (6) -> (9) -> (15) -> (22)

1: op=0, event's title = null
2: op =0, event's tile !=null, start time < current time
3: op=0, event's title !=null, start time >= current time, end time < start time
4: op=0, event's title !=null, start time >= current time, end time >= start time
5: op=1, event's title != null, start time >= current time, end time >= start time
6: op=2, c!=1
7: op=2, c=1
8: op! = {0,1,2}, t < current time
9: op! = {0,1,2}, t >= current time

4.1.3 Preparing test cases

#Test Case		1	2	3	4
Input Data	op	0	0	0	0
	Project's id	001	001	001	001
	Event's title	null	Meeting A	Meeting A	Meeting A
	Event's start time		8:00 10/2/2022	8:00 10/10/2024	8:00 10/9/2024
	Event's end time			8:00 10/9/2024	8:00 10/10/2024
	Event's id				
	c				
	t				
Expected Result		Show message: "Event's title cannot be empty."	Show message: "Start time cannot be in the past."	Show message: "End time cannot be earlier than start time."	Event "Meeting A" is created and appears on the top of the list of existing events.
Actual Result					
Status (Pass/Fail)					

#Test Case		5	6	7	8	9
Input Data	op	1	2	2	3	3
	Project's id					
	Event's title	Meeting B				
	Event's start time	8:00 20/12/2024				
	Event's end time	9:00 20/12/2024				
	Event's id	003	002	002	002	002
	c		0	1		
	t				8:00 2/3/2023	8:00 30/6/2024
Expected Result		The information for Event 003 has been updated with the entered input.	No change happens. Return to Schedule's main interface.	Event 002 is deleted.	Show message: "Reminder time cannot be in the past."	A reminder is set for event 002 and will send notification at 8:00 30/6/2024
Actual Result						
Status (Pass/Fail)						

4.2 Test 2 – To do list

Test ID:	M2BT2
Test description:	Handles various user operations correctly and performs the corresponding actions as expected.
Created by:	Thanh Nhi
Date of creation:	04/09/2024
Date of review:	
Test priority:	
Pre-requisite:	
Post-requisite:	

4.2.1 Drawing the flow graph

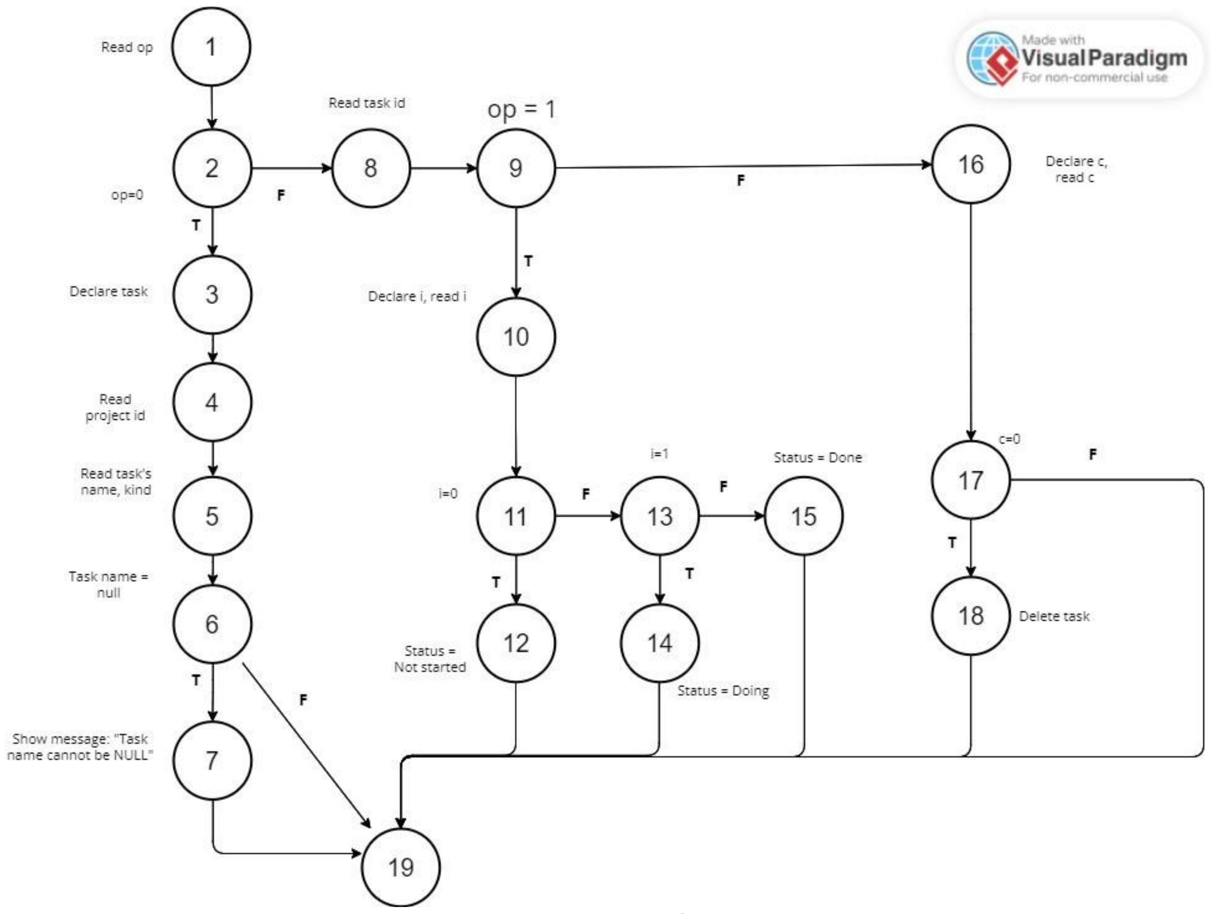


Image 18. To-do list flowgraph

4.2.2 Determining a basis set of independent paths

V = P + 1 = 6 + 1 = 7
Nodes (P): (2), (9), (11), (13), (17), (6)

1: (1) -> (2) -> (3) -> (4) -> (5) -> (6) -> (19)
2: (1) -> (2) -> (3) -> (4) -> (5) -> (6) -> (7) -> (19)
3: (1) -> (2) -> (8) -> (9) -> (10) -> (11) -> (12) -> (19)
4: (1) -> (2) -> (8) -> (9) -> (13) -> (14) -> (19)
5: (1) -> (2) -> (8) -> (9) -> (13) -> (15) -> (19)
6: (1) -> (2) -> (8) -> (9) -> (16) -> (17) -> (19)
7: (1) -> (2) -> (8) -> (9) -> (16) -> (17) -> (18) -> (19)

1: op=0, task name = null
2: op =0, task name !=null
3: op=1, i=0
4: op=1, i=1
5: op=1, i != {0,1}
6: op!={0,1}, c=0
7: op!={0,1}, c=1

4.2.3 Preparing test cases

#Test Case		1	2	3	4
Input Data	op	0	0	1	1
	Project's id	001	001		
	Task name	null	Read book		
	i			0	1
	c				
	Task's id			001	001
Expected Result		Show message: "Task name cannot be NULL."	Task read book is created	Status of the 001 is set as Not started	Status of the 001 is set as Doing
Actual Result					
Status (Pass/Fail)					

#Test Case		5	6	7
Input Data	op	1	2	2
	Project's id			
	Task name			
	i	2		
	c		0	1
	Task's id	001	001	001
Expected Result		Status of the 001 is set as Done	The task 001 is not Deleted	The task 001 is Deleted
Actual Result				
Status (Pass/Fail)				

4.3 Test 3 – Mind map

Test ID:	M2BT2
Test description	
Created by:	Anh Tài
Date of creation:	4/16/2024
Date of review:	
Test priority:	
Pre-requisite:	
Post-requisite:	

4.3.1 Drawing the flow graph

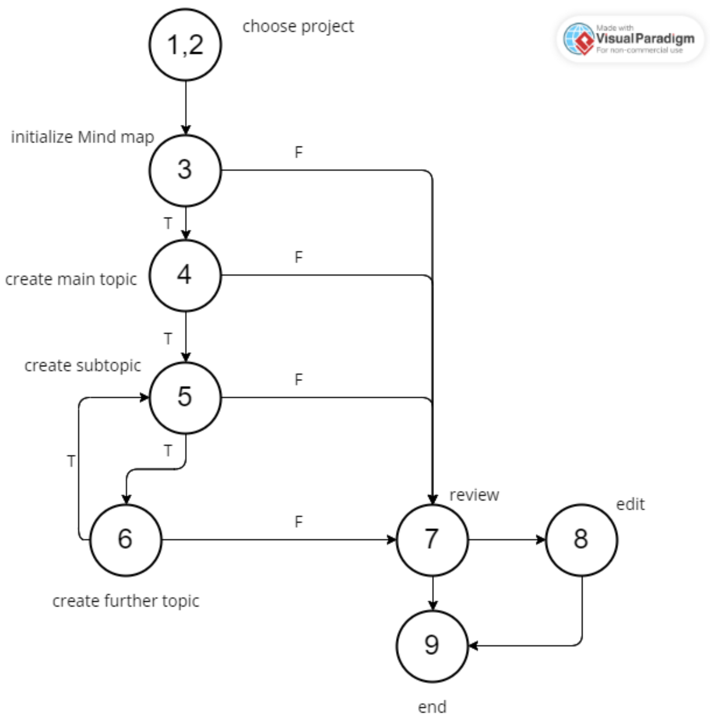


Image 19. Mind map flow graph

4.3.2 Determining a basis set of independent paths

V= P +1 = 4+1=5
Node (P) : (3), (5) ,(4), (7)
1: (1,2) ->(3) -> (7) -> (9)
2: (1,2)-> (3)-> (7)-> (8) -> (9)
3: (1,2) ->(3) -> (4)-> (7) -> (9)
4: (1,2) ->(3) -> (4)->(5)-> (7) -> (9)
5: (1,2) ->(3) -> (4)->(5)-> (6)-> (7) -> (9)

1: central topic = null
2: central topic != null, maintopic != null, subtopic != null
3: central topic != null, subtopic != null , maintopic = null
4: central topic != null, subtopic != null ,maintopic = null
5: central topic != null, maintopic != null , subtopic != null , further topic != null

# Test Case	main topic	central topic	subtopic	Expected Result	Actual Result	Status (Pass/ Fail)
1	Null	Null	Null	Can't create the other topic, there is only one central topic created with value =null		
2	“helloWorld”	“good”	“hoho”	Central topic = ”good”, main topic= ”helloWorld”, subtopic = ”hoho” User can edit and custom all topic		
3	Null	“city”	“love”	Central topic =”city” can’t create subtopic because main topic =null		
4	“shark”	“baby”	Null	Central topic =”baby” main topic =”shark” no exist subtopic		
5	“i”	“love”	“my school ”	Central topic =” love” main topic =” i” subtopic = “myschool” and can be create further topic from subtopic		

4.4 Test 4 - Progress Tracker

Test ID:	M2BT4
Test description	
Created by:	Hoàng Vũ
Date of creation:	4/3/2024
Date of review:	
Test priority:	
Pre-requisite:	
Post-requisite:	

4.4.2 Drawing the flow graph

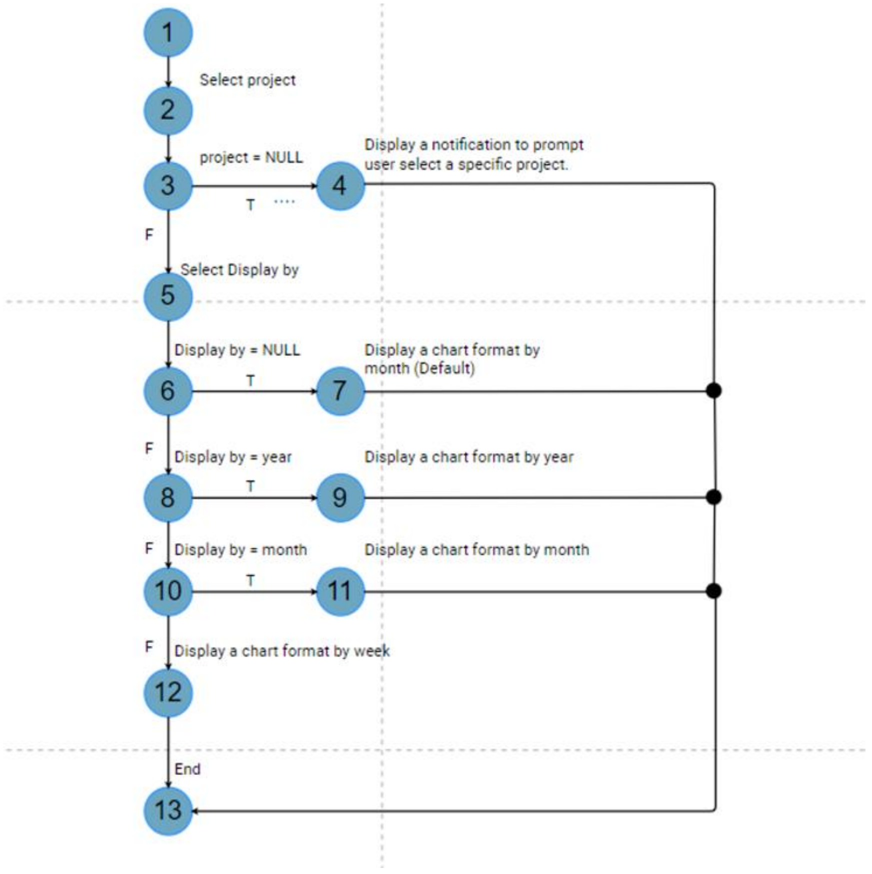

Select project
project = NULL
T
F
Display a notification to prompt user select a specific project.
Select Display by
Display by = NULL
Display a chart format by month (Default)
Display by = year
Display a chart format by year
Display by = month
Display a chart format by month
Display a chart format by week
End
1
2
3
4
5
6
7
8
9
10
11
12
13

Image 20. Progress Tracker flowgraph

4.4.3 Determining a basis set of independent paths

V = P + 1 = 4 + 1 = 5
Nodes (P): (3), (6), (8), (10)

1: (1) -> (2) -> (3) -> (4) - > 13
2: (1) -> (2) -> (3) -> (5) -> (6) -> (7) -> (13)
3: (1) -> (2) -> (3) -> (5) -> (6) -> (8) -> (9) -> (13)
4: (1) -> (2) -> (3) -> (5) -> (6) -> (8) -> (10) -> (11) -> (13)
5: (1) -> (2) -> (3) -> (5) -> (6) -> (8) -> (10) -> (12) -> (13)

1: (Project = null)
2: (Project != null, Display by = null)
3: (Project != null, Display by = null, Display by = year)
4: (Project != null, Display by = null, Display by != year, Display by = month)
5: (Project != null, Display by = null, Display by != year, Display by != month ->Display = week)

4.4.4 Preparing test cases

# Test Case	Input Data		Expected Result	Actual Result	Status (Pass/Fail)
	Project	**Display by**			
1	Null	Month	Display a notification to prompt user select a specific Project.		
2	Project A	Null	Display a chart format by month (default)		
3	Project B	Year	Display a chart format by year		
4	Project C	Month	Display a chart format by month		
5	Project D	Week	Display a chart format by week		