Software Requirements Specification

for

StudyBuddy

Version 1.0 approved

Prepared by

JARED PEK

BARRY NG

GARY

ABHINAV

SHRUTIKHAA

Nanyang Technological University, Team 5

30/1/2024

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Jared, Barry, Gary, Abhinav, Shrutikhaa	30/1/2024	First Draft	1.0

1. Introduction

1.1 Purpose

People often find it difficult to find conducive places to study and work near them that provides them with free wifi and air conditioning. Public libraries are also usually overcrowded, and can be noisy and unconducive in the evenings when young children visit them. In addition, it is often difficult for students to study on their own, especially when they do not have any help with their work.

With StudyBuddy, this problem will be one of the past as our study areas are certified to provide a quiet and cool environment for you to study and work in. These areas are hosted by friendly members of your community, and will utilize an honor booking system to manage the number of vacancies.

Through our platform, you can identify study areas near you, listed in order of distance to you, and easily book a time slot at these platforms. In addition, you can also visit our store, where you can purchase stationery, books and study materials at an affordable price, which will then be delivered straight to you. Lastly, you can also seek help from our friendly AI bot and ask it any questions that you need regarding your work.

Overall, StudyBuddy enhances your work and study experience by providing you with the right environment, and the right tools to maximize your productivity.

The URL to our deployed application is https://studybuddyswe.vercel.app/, so feel free to access our platform!

1.2 Document Conventions

1. **Software Requirements Specification Format**
   1.1. IEEE 830-1998
   1.2. Priorities of higher-level requirements inherited by detailed level requirements.
2. **Font** → Times New Roman
3. **Main Header** → Size 18, Bold
4. **Sub-Section Header** → Size 14, Bold
5. **Content** → Size 12

Further conventions on any terms used can be found in Appendix A – Glossary/Data Dictionary.

1.3 Intended Audience and Reading Suggestions

This document is intended for all the stakeholders involved in this project – users, developers, testers, project managers and marketing staff.

It details all the features of our application, and a detailed explanation on the purpose behind each of them. The intended user flow will also be included, together with functional and non-functional requirements. We will also include mock-ups of the various pages of our platforms to allow for better visualization of the layout and design of our application.

1.4 Product Scope

Our product will be presented and available to the public in the form of a web application. The application will have a responsive and user-friendly and intuitive interface that is easy to navigate. Through our application, users will be able to maximize their productivity due to their improved access to conducive work and study areas, and increased convenience in acquiring any study materials that they require.

1.5 References

1. IEEE 830-1998 → https://www.cse.msu.edu/~cse870/IEEEXplore-SRS-template.pdf
2. HTTPS Protocol → https://web.dev/articles/why-https-matters/
3. RESTful APIs → https://aws.amazon.com/what-is/restful-api/
4. JSON-Web-Tokens → https://jwt.io/
5. Vite.js Framework → https://vitejs.dev/
6. Django REST Framework → https://www.django-rest-framework.org/
7. PostgreSQL Database → https://www.postgresql.org/docs/current/index.html
8. Vercel → https://vercel.com/docs/
9. Railway → https://railway.app/
10. GitHub Repository → https://github.com/softwarelab3/2006-SCSE-group5
11. Gemini AI → https://ai.google.dev/docs

2. Overall Description

2.1 Product Perspective

StudyBuddy is a new application and is not a part of any larger product family or product line.

2.2 Product Functions

1. Key features available to host users
   - 1.1. Management of study areas – create, edit and delete study areas.
2. Key features available to regular users
   - 2.1. Management of bookings – create, edit, delete bookings, and view past bookings.
   - 2.2. Management of cart items – add and remove products from the cart, check out the cart to confirm purchase, and view past purchases.
   - 2.3. AI Study Helper – ask questions related to homework to our AI bot
3. Key features available to everyone (host users, regular users and non-users)
   - 3.1. Search for study areas near you – obtain the user's live location and calculate distance via Google maps API.
   - 3.2. Study areas aggregated in order of closest distance to you.
   - 3.3. Browse all the available products in the store.

2.3 User Classes and Characteristics

1. Regular Students

<table>
<tr><th>Type</th><th>Frequency</th><th>Functions Used</th><th>Characteristics</th></tr>
<tr><td>Primary</td><td>Low</td><td>● Booking Management
○ View Bookings
○ Create Booking
○ Edit Booking
○ Delete Booking
● Ask Buddy AI</td><td>● Energetic
● Active
● Could be noisy</td></tr>
<tr><td>Secondary</td><td>High</td><td rowspan="4">● Booking Management
○ View Booking
○ Create Booking
○ Edit Booking
○ Delete Booking
● Cart Management
○ View Products
○ Add Products
○ Edit Product Quantity
○ Delete Product
○ Checkout Product
● View Orders
● Ask Buddy AI</td><td>● Energetic
● Active
● Could be noisy</td></tr>
<tr><td>Junior College</td><td>High</td><td rowspan="3">● More mature
● More focussed
● Likely to have heavier workload</td></tr>
<tr><td>Polytechnic</td><td>Medium</td></tr>
<tr><td>University</td><td>High</td></tr>
</table>

2. Management Staff

2.4 Operating Environment

1. Production Environment

1.1. Front-End
- 1.1.1. Language → Node.js 20.11
- 1.1.2. Framework → Vite.js 5.1.6
- 1.1.3. Deployment → Vercel (Hobby Plan), which uses Amazon Linux 2

1.2. Back-End
- 1.2.1. Language → Python 3.10.5
- 1.2.2. Framework → Django REST Framework 3.14
- 1.2.3. Database → SQLite 3.45
- 1.2.4. Deployment → Railway (Hobby Plan), which use Google Compute Engine
- 1.2.5. Security → Cloudflare Integration Free Plan

2. Development Environment

2.1. Front-End
- 2.1.1. Language → Node.js 20.11
- 2.1.2. Framework → Vite.js 5.1.6

2.2. Back-End
- 2.2.1. Language → Python 3.10.5
- 2.2.2. Framework → Django REST Framework 3.14
- 2.2.3. Database → SQLite 3.45

2.3. The application will run on localhost via port 4173 for the front-end, and port 8000 for the back-end, with either a Linux or Windows operating system.

2.5 Design and Implementation Constraints

1. The Vercel deployment platform has a hard 5 second limit on the response time for outgoing API calls. If a server takes over 5 seconds to respond, Vercel immediately initiates a timeout and throws an error. This could be resolved by upgrading the deployment plan to a paid one.

2.6 User Documentation

1. The source code will be provided in its entirety on the GitHub repository.
2. Comprehensive documentation on the various features and classes that we have implemented will be included together with this document.

2.7 Assumptions and Dependencies

1. All the data obtained from the Google Maps API will be accurate.
2. The user's device has a working GPS that accurately shows the user's current location.
3. The user has a sufficiently strong internet connection, since the features of our application like managing bookings and study areas require data to be sent to our server via the internet.

3. External Interface Requirements

3.1 User Interfaces

1. Refer to Appendix C on UI Mock-Ups to view the design and layout for all the pages that are available on our application.

3.2 Hardware Interfaces

1. StudyBuddy requires a mobile device or computer with any web browser installed, preferably Google chrome.
2. The device accessing the StudyBuddy platform must be connected to the internet in order to interact with our servers.
3. The device must contain a GPS unit in order for the platform to rank study areas nearest to the user. If no GPS is available, the study areas would only be sorted in alphabetical order.

3.3 Software Interfaces

1. StudyBuddy requires a mobile device or computer with any web browser installed, preferably at least Google chrome version 88.0.4324.69.
2. The architecture of our platform is as follows
    - 2.1. Layered Architecture
        - 2.1.1. Client Layer → Browser
        - 2.1.2. Presentation Layer → Front-End
            - 2.1.2.1. Vite.js → A lightweight React-based JavaScript framework for developing responsive web applications
            - 2.1.2.2. Vercel → For hosting the Vite.js front-end
        - 2.1.3. Business Layer → Back-End
            - 2.1.3.1. Django REST Framework → A Python framework for developing extensive REST APIs
            - 2.1.3.2. Railway → For hosting the django back-end
        - 2.1.4. Storage Layer → Database
            - 2.1.4.1. SQLite → A simple, lightweight and portable relational database
            - 2.1.4.2. Railway → For hosting the SQLite database

3.4 Communications Interfaces

1. Communication between the front and back-ends will be via the HTTPS protocol and TCP/IP network protocol.
2. Authentication will be done via JWT on both the front and back-ends, encrypted with the HS512 algorithm.

4. System Features

4.1 Register

4.1.1 Description

First time users can register for an account for StudyBuddy. With an account, users can book a slot at various study areas, purchase items from our store, and ask our AI bot any question he needs. A record of any bookings and purchases will be automatically stored in the system.

4.1.2 Response Sequences / Use Case

Use Case ID:	R1		
Use Case Name:	Register		
Created By:	Jared Pek	Last Updated By:	Jared Pek
Date Created:	30/1/2024	Date Last Updated:	30/1/2024

Actor:	New User
Description:	First-time users will be able to register an account at StudyBuddy, either as a Host, or as a Regular user.
Preconditions:	User should not have previously registered an account.
Postconditions:	User would have successfully created an account and logged in.
Priority:	High
Frequency of Use:	Medium
Flow of Events:	1. User clicks on “Register” on the navigation bar. 2. User will be redirected to the “register” page, where he is prompted to enter the username, email, password, password confirmation, first name, last name and mobile number. 3. User enters all the required information and clicks “Register”. 4. System verifies all the information and registers a new user, then authenticates the new user to the platform. 5. User is redirected to the home page.
Alternative Flows:	R1-A1 → If user enters an existing username 1. Error message “username already exists” will be displayed. 2. System returns to Step 2. R1-A2 → If user enters an existing email 1. Error message “account already exists” will be displayed. 2. System returns to Step 2.
Exceptions:	NIL
Includes:	NIL
Special Requirements:	NIL
Assumptions:	User has internet connection.
Notes and Issues:	NIL

4.1.3 Functional Requirements

1. The user must be able to register for an account on the system.
    - 1.1. The system must display 7 input fields for the required user to enter his information.
        - 1.1.1. 1 text field for the user's username.
        - 1.1.2. 1 email field for the user's email.
        - 1.1.3. 1 password field for the user's password.
        - 1.1.4. 1 password field for the user's password confirmation.
        - 1.1.5. 1 text field for the user's first name.
        - 1.1.6. 1 text field for the user's last name.
        - 1.1.7. 1 text field for the user's mobile number.
        - 1.1.8. The system must display a "Register" button
    - 1.2. The user must fill in the text, email and password fields before clicking on the "Register" button.
        - 1.2.1. The system must verify all the data provided.
    - 1.3. The system will first verify the provided information.
        - 1.3.1. The username is unique, and not already existing.
        - 1.3.2. The email is unique, in the correct email format, and not already existing.
        - 1.3.3. The password and password confirmation fields match, and are at least 8 characters long with both alphabetical and numerical characters.
        - 1.3.4. The mobile number contains the country code, denoted by the '+' symbol at the start, and contains at least 10 numerical digits.
    - 1.4. Upon verification, a new account will be created for the user.
    - 1.5. Upon creation, the user must be automatically authenticated into the system.

4.2 Login

4.2.1 Description

Users must login to the application in order to access features of the application, such as creating a booking or purchasing items from the store.

4.2.2 Response Sequences / Use Case

Use Case ID:	L1		
Use Case Name:	Login		
Created By:	Jared Pek	Last Updated By:	Jared Pek
Date Created:	30/1/2024	Date Last Updated:	30/1/2024

Actor:	User, Host
Description:	All users must login to view study areas, manage bookings, and purchase items from the store.
Preconditions:	User has previously registered an account.
Postconditions:	User is successfully authenticated to the system.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. User clicks on “Login” on the navigation bar. 2. User will be redirected to the “login” page, where he is prompted to enter the username and password. 3. User enters all the required information and clicks “Login”. 4. System verifies all the information and authenticates the user to the platform. 5. User is redirected to the home page.
Alternative Flows:	L1-A1 → If user credentials are incorrect 1. Error message “username or password is incorrect” will be displayed. 2. System returns to Step 2.
Exceptions:	NIL
Includes:	NIL
Special Requirements:	NIL
Assumptions:	User has internet connection.
Notes and Issues:	NIL

4.2.3 Functional Requirements

1. The user must be able to login to an account on the system.
    1.1. The system must display 2 input fields for the user to enter his credentials.
        1.1.1. 1 text field for the user’s username.
        1.1.2. 1 password field for the user’s password.
    1.2. The system must display a “Login” button.
    1.3. The user must fill in the text and password fields before clicking on the “Login” button.

1.3.1. The system must verify all the data provided.

1.4. The system will verify the provided information before authenticating the user to the platform.

1.4.1. The username corresponds with an existing user in the system.

1.4.2. The password belongs to the existing user that was identified in 1.4.1.

1.5. Upon verification, the user will be successfully authenticated to the system .

4.3 Create Study Area

4.3.1 Description

The host user can create a study area for regular users to book and study at.

4.3.2 Response Sequences / Use Case

Use Case ID:	SA1		
Use Case Name:	Create Study Area		
Created By:	Jared Pek	Last Updated By:	Jared Pek
Date Created:	30/1/2024	Date Last Updated:	30/1/2024

Actor:	Host, Google Maps API
Description:	Host users can create new study areas for regular users to use.
Preconditions:	1. Host user is already logged in to the system. 2. No study areas exist at the same location.
Postconditions:	Host user creates a new study area.
Priority:	High
Frequency of Use:	Medium
Flow of Events:	1. Host user clicks on the “Create” button. 2. User is redirected to a form that prompts the user for the address and postal code of the study area, unit number, description, capacity and opening hours etc of the study area. 3. User enters all the required data, then clicks on the “Create” button. 4. The system will verify that the area has not yet been created, then creates the new study area with the specified inputs. 5. The system also makes a request to the Google Maps API using the postal code provided, and saves the longitude and latitude of the location
Alternative Flows:	SA1-A1 → Study Area already exists 1. Error message “study area already exists” will be displayed. 2. System returns to Step 2.
Exceptions:	NIL
Includes:	NIL
Special Requirements:	NIL
Assumptions:	User has internet connection
Notes and Issues:	NIL

4.3.3 Functional Requirements

1. The host user must be able to create a new study area on the system.
    1.1. The system must display 9 input fields for the host user to enter the required data for the study area.
        1.1.1. 1 text field for the address.
        1.1.2. 1 text field for the postal code.

1.1.3. 1 text field for the name of the study area.
1.1.4. 1 number field for the level of the study area.
1.1.5. 1 number field for the unit number of the study area.
1.1.6. 1 number field for the capacity of the study area.
1.1.7. 1 time field for the time the study area opens.
1.1.8. 1 time field for the time the study area closes.
1.1.9. 1 text field for a description of the study area.

1.2. The system must display a “Create” button.

1.3. The user must fill in all the input fields before clicking the “Create” button.

1.3.1. The system must verify the data provided.

1.4. The system will verify the data provided by the user before creating the new study area.

1.4.1. All the fields are not empty.
1.4.2. The level, unit number and capacity must be integers larger than 0.

1.5. Upon verification, the new study area will be created.

1.6. System also makes a request to the Google Maps API to obtain the longitude and latitude of the location, and saves it to the database.

4.4 Delete Study Area

4.4.1 Description

The host user can close and delete an existing study area.

4.4.2 Response Sequences / Use Case

Use Case ID:	SA2		
Use Case Name:	Delete Study Area		
Created By:	Jared Pek	Last Updated By:	Jared Pek
Date Created:	30/1/2024	Date Last Updated:	30/1/2024

Actor:	Host
Description:	Host users can delete existing study areas.
Preconditions:	1. Host user is already logged in to the system. 2. Selected study area already exists.
Postconditions:	Host user deletes an existing study area.
Priority:	Medium
Frequency of Use:	Low
Flow of Events:	1. Host user clicks on the "Delete" icon for the study area that he wants to delete. 2. Host user is redirected to another page that prompts him to confirm the deletion by clicking the "Confirm" button. 3. Once the host user clicks on "Confirm", the system deletes the study area
Alternative Flows:	NIL
Exceptions:	NIL
Includes:	NIL
Special Requirements:	NIL
Assumptions:	User has internet connection
Notes and Issues:	NIL

4.4.3 Functional Requirements

1. The host must be able to delete an existing study area.
   - 1.1. For every study area displayed on the system, the system must include a "Delete" icon.
   - 1.2. Upon clicking on the "Delete" icon, the user will be redirected to another page that displays a "Confirm" button.
   - 1.3. Upon clicking on the "Confirm" button, the system will delete the selected study area from the system.

4.5 Edit Study Area

4.5.1 Description

The host user can edit the status, unit number, capacity, description and opening hours of the study area.

4.5.2 Response Sequences / Use Case

Use Case ID:	SA3		
Use Case Name:	Edit Study Area		
Created By:	Jared Pek	Last Updated By:	Jared Pek
Date Created:	30/1/2024	Date Last Updated:	30/1/2024

Actor:	Host
Description:	Host users can update existing study areas.
Preconditions:	1. Host user is already logged in to the system. 2. Selected study area already exists.
Postconditions:	Host user successfully updates an existing study area.
Priority:	Medium
Frequency of Use:	Low
Flow of Events:	1. Host user clicks on the “Edit” icon for the study area he wants to update. 2. Host user will be redirected to a form prompting him to enter the study area’s new status, unit number, capacity, description or opening hours. 3. Host user will enter the changes to the study area, then click on the “Confirm” button. 4. System updates the study area
Alternative Flows:	NIL
Exceptions:	NIL
Includes:	NIL
Special Requirements:	NIL
Assumptions:	User has internet connection
Notes and Issues:	NIL

4.5.3 Functional Requirements

1. The host user must be able to edit an existing study area on the system
    1.1. There must be a display for the “edit” icon for the study area he wants to update.
    1.2. The system must display the 6 input fields for the host to enter the edited data for the study area while containing the previously filled details:
        1.2.1. 1 integer field within range for the unit number
        1.2.2. 1 integer field within range for the capacity of the study area
        1.2.3. 1 date/time field for study area opening hours
        1.2.4. 1 date/time field for the study area closing hours

1.2.5. 1 text field for description
1.2.6. 1 Boolean field to update the status
1.3. The system must display a “Confirm” Button
1.4. The user must fill in all the input fields before clicking the “Confirm” Button.
1.4.1. The system must verify the data provided
1.5. The system will verify the data provided by the user before saving changes made to the existing study area.
1.5.1. All the fields are not empty.
1.5.2. Level, Unit number and capacity should contain integers larger than 0
1.6. Upon verification, the changes made to the study area will be saved

4.6 Create Booking

4.6.1 Description

The user can create a booking for a study area to study at.

4.6.2 Response Sequences / Use Case

Use Case ID:	B1		
Use Case Name:	Create Booking		
Created By:	Jared Pek	Last Updated By:	Jared Pek
Date Created:	30/1/2024	Date Last Updated:	30/1/2024

Actor:	User
Description:	1. Users can create a new booking at a study area. 2. User must not have booked any study area at the same time.
Preconditions:	User is already logged in to the system.
Postconditions:	User successfully makes a booking at a study area.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. User clicks on “Book” for the selected study area. 2. User is redirected to a form that prompts him to select the date and start time he will be there. 3. Once user selects the required information, he clicks on the “Confirm” button to create the booking.
Alternative Flows:	NIL
Exceptions:	NIL
Includes:	NIL
Special Requirements:	NIL
Assumptions:	User has internet connection
Notes and Issues:	NIL

4.6.3 Functional Requirements

1. User must be able to create a booking for an existing study area.
    - 1.1. The system must display 3 fields required for the booking.
        - 1.1.1. 1 date field for the date of the booking.
        - 1.1.2. 1 time field for the start time of the booking.
        - 1.1.3. 1 disabled time field for the end time of the booking.
            - 1.1.3.1. End time will be automatically calculated by the system 1 hour from the start time
    - 1.2. The system must display a “Confirm” button.
    - 1.3. The user must select an appropriate date and time for both fields before clicking on the “Confirm” button.
        - 1.3.1. The system must verify the data provided by the user..
    - 1.4. The system will verify the data provided by the user before creating a new booking for a study area.

1.4.1. All 3 fields are not empty and have not passed the current time.
1.4.2. The date provided must be within the opening days of the area.
1.4.3. The start and end times must be within the opening hours of the area.
1.5. Upon verification, a new booking for the study area will be created.

4.7 Booking

4.7.1 Description

The user can delete a booking that was previously made for a study area.

4.7.2 Response Sequences / Use Case

Use Case ID:	B2		
Use Case Name:	Delete Booking		
Created By:	Jared Pek	Last Updated By:	Jared Pek
Date Created:	30/1/2024	Date Last Updated:	30/1/2024

Actor:	User
Description:	Users can delete an existing booking for a study area.
Preconditions:	1. User is already logged in to the system. 2. User has already created a booking previously.
Postconditions:	User successfully deletes a booking.
Priority:	High
Frequency of Use:	Medium
Flow of Events:	1. User clicks on “Delete” for the selected booking. 2. Once the user clicks on the “Delete” button, the system will delete the booking.
Alternative Flows:	NIL
Exceptions:	NIL
Includes:	NIL
Special Requirements:	NIL
Assumptions:	User has internet connection
Notes and Issues:	NIL

4.7.3 Functional Requirements

1. The user must be able to delete an existing booking for a study area.
   - 1.1. For every study area booking displayed on the system, the system must include a “Delete” icon.
   - 1.2. Upon clicking on the “Delete” button, the system will delete the selected booking from the system.

4.8 Edit Booking

4.8.1 Description

The user can edit a booking that was previously made

4.8.2 Response Sequences / Use Case

Use Case ID:	B3		
Use Case Name:	Edit Booking		
Created By:	Jared Pek	Last Updated By:	Jared Pek
Date Created:	30/1/2024	Date Last Updated:	30/1/2024

Actor:	User
Description:	Users can edit an existing booking for a study area.
Preconditions:	1. User is already logged in to the system. 2. User has already created a booking previously.
Postconditions:	User successfully edits a booking.
Priority:	High
Frequency of Use:	Medium
Flow of Events:	1. User clicks on “Edit” for the selected booking. 2. User is redirected to a form that prompts the user for a new date and start time of booking. 3. User selects all the required details, and click on the “Confirm” button. 4. System updates the booking details.
Alternative Flows:	NIL
Exceptions:	NIL
Includes:	NIL
Special Requirements:	NIL
Assumptions:	User has internet connection
Notes and Issues:	NIL

4.8.3 Functional Requirements

1. The User must be able to edit a booking for a study area
    - 1.1. There will be a bookings tab that shows historical bookings of the user
    - 1.2. Each of the bookings can be expanded that will show more details about the booking such as the name of the study area, date, start time, and end time
    - 1.3. The system must display an “Edit” button in the same page
    - 1.4. Upon clicking the “Edit” button, the user will be redirected to an edit booking form
    - 1.5. The edit booking forming will have 3 fields
        - 1.5.1. 1 Date field for the booking day
        - 1.5.2. 1 Time field for the start time
        - 1.5.3. 1 disabled Time field for the end time
            - 1.5.3.1. End time is automatically calculated 1 hour from the start time

1.6. The system must display an “Update” button to save the changes made
1.7. Upon clicking the “Update” button, the system will verify all the changes made, then update the changes to the booking
    1.7.1. All field validation is the same as use case B1 on “Create Booking”

4.9 Add Product to Cart

4.9.1 Description

The user can add any product available in the store to his cart, in a quantity required by him.

4.9.2 Response Sequences / Use Case

Use Case ID:	C1		
Use Case Name:	Add Product to Cart		
Created By:	Jared Pek	Last Updated By:	Jared Pek
Date Created:	30/1/2024	Date Last Updated:	30/1/2024

Actor:	User, E-Commerce API
Description:	Users can add a product from the store to his cart.
Preconditions:	User is already logged in to the system.
Postconditions:	User successfully adds the product to the cart.
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1. User clicks on the “Store” icon on the navigation bar. 2. User is redirected to the store page, where he can view all the available products. 3. User clicks on the “Add to Cart” button. 4. The system will make a POST request to the E-Commerce API, which adds the selected product to the user’s cart.
Alternative Flows:	NIL
Exceptions:	NIL
Includes:	NIL
Special Requirements:	NIL
Assumptions:	User has internet connection
Notes and Issues:	NIL

4.9.3 Functional Requirements

1. User must be able to add items to his cart.
   - 1.1. Each item displayed will have their price, image and description with their own corresponding buy button
   - 1.2. After adding to cart, the cart icon must have an indicator that an item has been added to cart.

4.10 Remove Product from Cart

4.10.1 Description

The user can remove products that are in his cart.

4.10.2 Response Sequences / Use Case

Use Case ID:	C2		
Use Case Name:	Remove Product from Cart		
Created By:	Jared Pek	Last Updated By:	Jared Pek
Date Created:	30/1/2024	Date Last Updated:	30/1/2024

Actor:	User, E-Commerce API
Description:	Users can remove a product from his cart.
Preconditions:	1. User is already logged in to the system. 2. User already has a product in the cart.
Postconditions:	User successfully removes the product from his cart.
Priority:	Medium
Frequency of Use:	Low
Flow of Events:	1. User clicks on the “Cart” icon on the navigation bar. 2. User is redirected to the cart page, where he can view all the products that he has added to the cart. 3. User clicks on the “Delete” button to remove the product from his cart. 4. System makes a “DELETE” request to the E-Commerce API and deletes the product from the cart..
Alternative Flows:	NIL
Exceptions:	NIL
Includes:	NIL
Special Requirements:	NIL
Assumptions:	User has internet connection
Notes and Issues:	NIL

4.10.3 Functional Requirements

1. The user must be able to remove products from the cart
    1.1. The cart page will contain a ‘delete’ icon beside each product in the cart
    1.2. When the ‘delete’ icon is clicked, the item will be removed entirely from the user’s cart.
        1.2.1. PUT request made to the E-Commerce API with quantity of 0 for the specific product
    1.3. The ‘Order Summary’ field that tabulates all the prices of the products in the cart will be updated upon item’s removal
        1.3.1. The price will be reduced accordingly

4.11 Edit Product in Cart

4.11.1 Description

The user can edit the quantity of given products in the cart.

4.11.2 Response Sequences / Use Case

Use Case ID:	C3		
Use Case Name:	Edit Product in the Cart		
Created By:	Jared Pek	Last Updated By:	Jared Pek
Date Created:	30/1/2024	Date Last Updated:	30/1/2024

Actor:	User, E-Commerce API
Description:	Users can edit the quantity of a product in his cart.
Preconditions:	1. User is already logged in to the system. 2. User already has a product in the cart.
Postconditions:	User successfully edits the quantity of the product from his cart.
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1. User clicks on the “Cart” icon on the navigation bar. 2. User is redirected to the cart page, where he can view all the products that he has added to the cart. 3. User clicks on the “-” or “+” buttons to modify the quantity of the cart. 4. System makes a “PUT” request to the E-Commerce API with the new quantity of the product and updates the cart..
Alternative Flows:	NIL
Exceptions:	NIL
Includes:	NIL
Special Requirements:	NIL
Assumptions:	User has internet connection
Notes and Issues:	NIL

4.11.3 Functional Requirements

1. The user must be able to edit products in the cart
    - 1.1. The user must click on the “Cart” icon on the navigation bar.
        - 1.1.1. Clicking on the “Cart” icon should redirect the user to the cart page.
        - 1.1.2. The cart page should display a list of products added by the user, including information such as product name, quantity and price.
    - 1.2. Each product should have “+” and “-” buttons to allow the modification of the quantity of each product in the cart.
        - 1.2.1. The users should be able to increase or decrease the quantity of the product.
    - 1.3. Cart page dynamically updates with the new subtotal and grandtotal prices to reflect the changes in quantity without requiring a page refresh

4.12 Check Out

4.12.1 Description

The user can check out his cart and confirm a new order for the products in his cart.

4.12.2 Response Sequences / Use Case

Use Case ID:	C4		
Use Case Name:	Check Out		
Created By:	Jared Pek	Last Updated By:	Jared Pek
Date Created:	30/1/2024	Date Last Updated:	30/1/2024

Actor:	User, E-Commerce API
Description:	Users can check out his cart and confirm an order.
Preconditions:	1. User is already logged in to the system. 2. User already has at least 1 product in the cart.
Postconditions:	User successfully checks out his products and creates an order.
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1. User clicks on the “Cart” icon on the navigation bar. 2. User is redirected to the cart page, where he can view all the products that he has added to the cart. 3. User enters his address and postal code, then clicks on the “Check Out” button. 4. A new order for the products will be created. 5. All the products in his current cart will then be removed.
Alternative Flows:	NIL
Exceptions:	NIL
Includes:	NIL
Special Requirements:	NIL
Assumptions:	User has internet connection
Notes and Issues:	NIL

4.12.3 Functional Requirements

1. User must be able to check out the products in his cart.
   - 1.1. The system must display a “Check Out” button on the cart page.
   - 1.2. The cart must have at least 1 item before the user is able to click on the “Check Out” button.
   - 1.3. The system must display a form that contains 2 fields.
     - 1.3.1. 1 text field for the user’s delivery address, which is a string.
     - 1.3.2. 1 text field for the user’s delivery postal code, which is a string with 6 numeric characters that does not start with ‘0’
   - 1.4. Upon clicking on the “Check Out” button, the system will verify the data provided by the user before creating a new order for the selected products.
     - 1.4.1. The 2 fields are not empty.

1.4.2. The address is valid
1.4.3. The postal code are valid
1.5. Upon verification, a new order will be created for the selected products.
1.6. After an order is created, the system will remove all the products in the cart.

4.13 View Orders

4.12.1 Description

The user can view all orders that he has previously made at our store.

4.12.2 Response Sequences / Use Case

Use Case ID:	VO1		
Use Case Name:	View Order		
Created By:	Jared Pek	Last Updated By:	Jared Pek
Date Created:	30/1/2024	Date Last Updated:	30/1/2024

Actor:	User, E-Commerce API
Description:	Users can view all orders that he has placed.
Preconditions:	1. User is already logged in to the system.
Postconditions:	User successfully checks out his products and creates an order.
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	1. User clicks on the “Orders” icon on the navigation bar. 2. User is redirected to the orders page, where he can view all the orders that he has previously made but checking out his cart.
Alternative Flows:	NIL
Exceptions:	NIL
Includes:	Login
Special Requirements:	NIL
Assumptions:	User has internet connection
Notes and Issues:	NIL

4.12.3 Functional Requirements

2. User must be able to view all the orders that he has made in the past
    - 2.1. User clicks on the “orders” button on the navigation bar.
    - 2.2. User redirected to the orders page.
    - 2.3. System displays a list of all the orders that the user has previously made with the following details
        - 2.3.1. Order status, whether delivering preparing or completed
        - 2.3.2. Delivery address and postal code
        - 2.3.3. Date ordered
        - 2.3.4. Each product that was ordered
            - 2.3.4.1. Quantity ordered
            - 2.3.4.2. Unit price of product
            - 2.3.4.3. Subtotal price of the product
        - 2.3.5. Grant total price of the entire order

4.14 Ask Buddy AI

4.15.1 Description

Regular users can ask our AI bot questions that they need help with.

4.15.2 Response Sequences / Use Case

Use Case ID:	AB1		
Use Case Name:	Ask AI Buddy		
Created By:	Jared Pek	Last Updated By:	Jared Pek
Date Created:	30/1/2024	Date Last Updated:	30/1/2024

Actor:	User, Gemini API
Description:	Users can ask a question to our AI bot to seek help on his work.
Preconditions:	NIL
Postconditions:	User receives a response to his question.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. User navigates to the buddyai page by clicking on “BuddyAI” on the navigation bar. 2. User enters the question he wants to ask, then clicks on “Ask Buddy” button. 3. The system will make a POST request to the Gemini API, which then receives a response and displays the answer on the user’s screen.
Alternative Flows:	NIL
Exceptions:	NIL
Includes:	Login
Special Requirements:	NIL
Assumptions:	User has internet connection
Notes and Issues:	NIL

4.15.3 Functional Requirements

1. The user must be able to send a question to our AI bot and receive an answer to his question from the AI bot
    - 1.1. The system must display some fields for the user to interact with the bot
        - 1.1.1. The system must display 1 textarea field for the user to enter the question that he wants to ask
        - 1.1.2. The system must display 1 “Ask Buddy” button for the user to click on after he enters his question in the textarea field.
        - 1.1.3. Once the user enters his question, and clicks on “Ask Buddy”, the system will send a POST request with the question to the Gemini API, then receive a response from the API.

1.2. Once the response is received, the system must display a field for the user to view the response
    1.2.1. The system must display 1 textarea readonly field that takes the answer provided in the response as the value
1.3. User successfully asks a question and views a response from our bot

4.15 View Products

4.13.1 Description

All users can view all the available products in the e-commerce store.

4.13.2 Response Sequences / Use Case

Use Case ID:	P1		
Use Case Name:	View Products		
Created By:	Jared Pek	Last Updated By:	Jared Pek
Date Created:	30/1/2024	Date Last Updated:	30/1/2024

Actor:	User, E-Commerce API
Description:	Users can view all the available products in the store.
Preconditions:	NIL
Postconditions:	User views a list of all the available products on 1 page.
Priority:	High
Frequency of Use:	Medium
Flow of Events:	1. User clicks on “Store” on the navigation bar. 2. User will be redirected to the store page, where he can view all the products available for purchase.
Alternative Flows:	NIL
Exceptions:	NIL
Includes:	NIL
Special Requirements:	NIL
Assumptions:	User has internet connection
Notes and Issues:	NIL

4.13.3 Functional Requirements

1. The user must be able to view all available products on the store
    1.1. The system must display a “Store” button on the navigation bar
    1.2. Upon clicking the “Store” button, the user will be redirected to the Store page
    1.3. The system must then display all available products

4.16 View Study Areas

4.14.1 Description

All users can view the available study areas ranked in order of distance to them.

4.14.2 Response Sequences / Use Case

Use Case ID:	S1		
Use Case Name:	View Study Areas		
Created By:	Jared Pek	Last Updated By:	Jared Pek
Date Created:	30/1/2024	Date Last Updated:	30/1/2024

Actor:	User
Description:	Users can view all the available study areas around him.
Preconditions:	NIL
Postconditions:	User views a list of all the available study areas on 1 page.
Priority:	High
Frequency of Use:	High
Flow of Events:	1. User navigates to the home page. 2. System will obtain the user's live location. 3. System will aggregate the list of study areas and sort it in order of straight line distance to the user. 4. User can view the list of all the available study areas in the order that is closest to his location.
Alternative Flows:	NIL
Exceptions:	NIL
Includes:	NIL
Special Requirements:	NIL
Assumptions:	User has internet connection
Notes and Issues:	NIL

4.14.3 Functional Requirements

1. The user must be able to view all available study areas around him
   - 1.1. The app will obtain the users live location in "latitude" and "longitude"
   - 1.2. The system will use the "latitude" and "longitude" to calculate the distance between the user and the study area.
   - 1.3. The system will display a list of study areas available to him sorted in ascending order of distance between the user and the study area
   - 1.4. Each study area that is displayed in the list must display the name, location, image, available slots, timings and a button named "Book" that allows them to book a slot in the study area

5. Non-Functional Requirements

5.1 Usability Requirements

1. 90% of our users must be able to obtain the details of nearby study areas within 5 minutes of entering our platform. Study areas must be sorted in ascending order of distance to the user.
2. 75% of our users must be able to view all the available products in our store and add a product to his cart within 5 minutes of entering the platform.

5.2 Performance Requirements

1. The system must display the list of available study areas in order for distance to the user within 5 seconds of the user making a search.
2. The system must display the list of available products in the store within 5 seconds of the user entering the store.

5.3 Security Requirements

1. The system must hash and encrypt a user's password before storing it to the database.
2. All features except the searching for study area and viewing of the store and products must be accompanied with a JWT token.
    2.1. JWT token will be assigned to a user whenever he is successfully logged in.
    2.2. JWT token will be refreshed whenever the user navigates to another page on the site, or when he makes any requests to create, edit or delete objects.

5.4 Reliability Requirements

1. The web application must successfully load on the client's browser in less than 5 seconds.

5.5 Portability Requirements

1. Web Application should be fully functional on major web browsers, such as Google Chrome, Safari and Edge, while maintaining consistent performance and layout

5.6 Business Rules

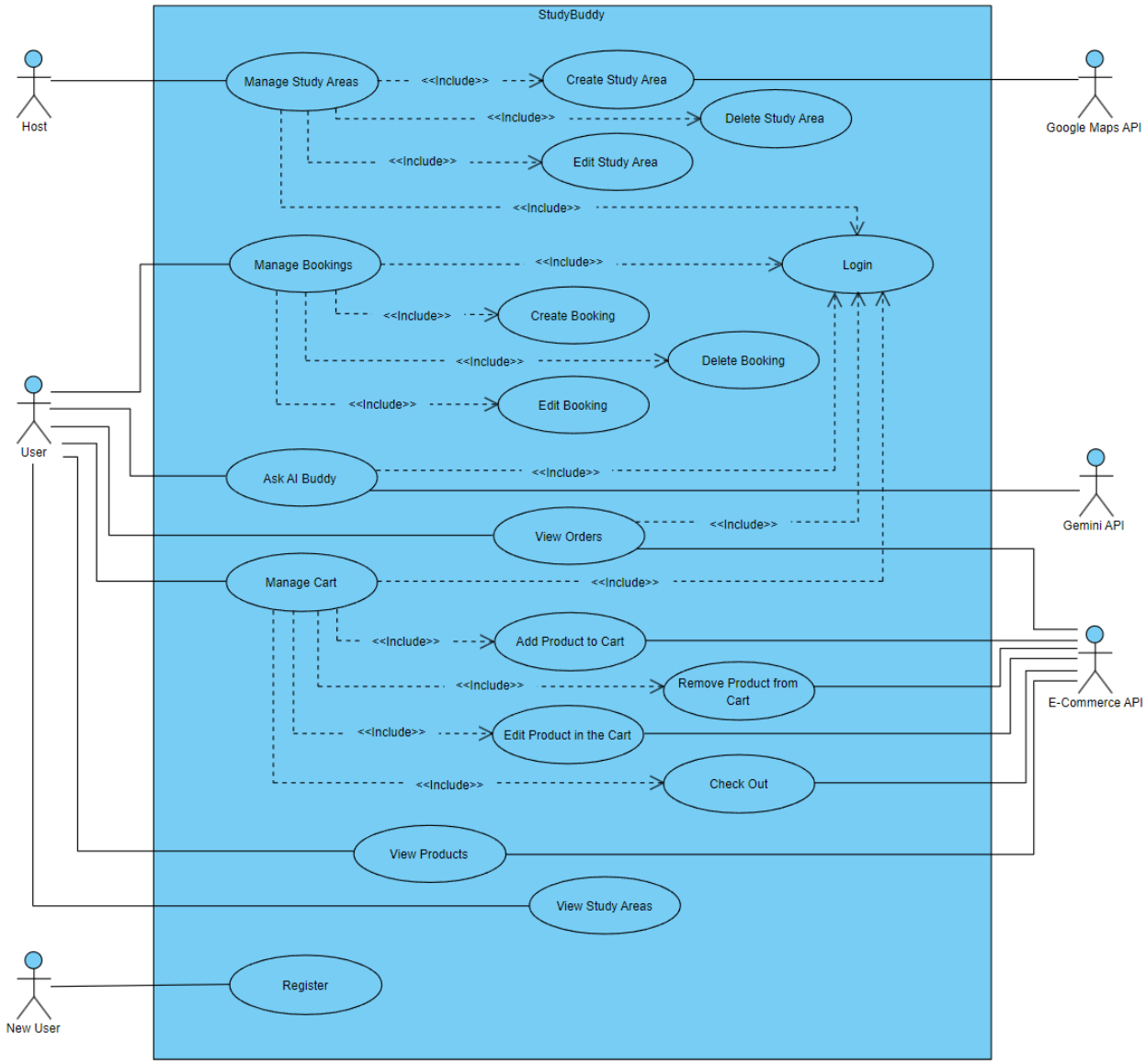
1. Host users must be authenticated for the management of study areas.
2. Regular users must be authenticated for the management of bookings, management of their carts, and asking our AI bot questions.
3. No authentication is required for viewing of study areas and store products.

Appendix A: Glossary / Data Dictionary

No.	Term	Definition
1.	Study Area	A physical place that can accommodate multiple study groups/individuals that are booked by Users and managed by the Host. Study Areas have a limit on the number of people it can accommodate at once.
2.	Store	The e-commerce store on our application that sells study materials and books.
3.	User	Lowest level profile available. Users can login and logout of the application. Users can view all available Study Areas and create, edit and delete Study Area bookings. Users can also ask our AI bot questions they need.
4.	Host	A User level profile with additional privileges and access to our admin page. A Host can create, edit, and delete Study Areas.

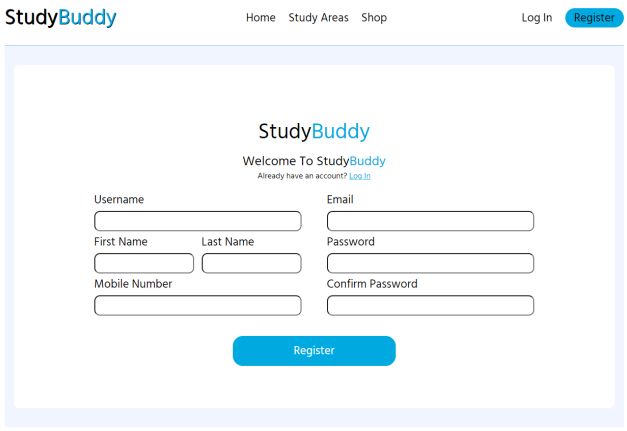
Appendix B: Analysis Models

1. Use Case Diagram

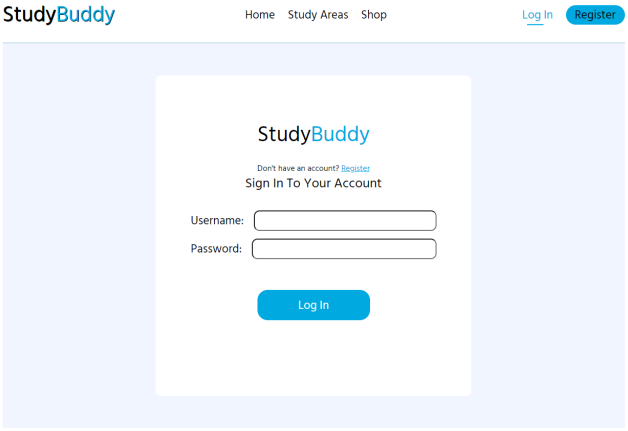


Appendix C: UI Mock-Ups

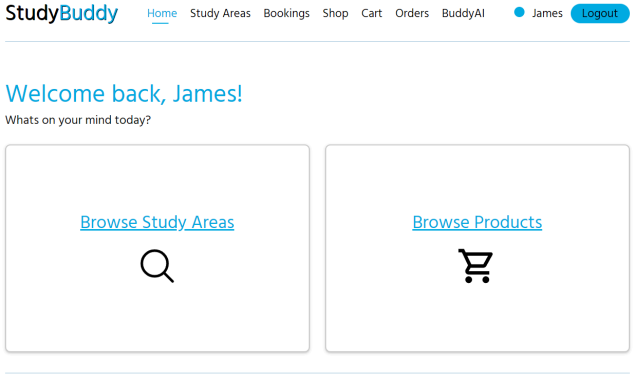
1. Register Page



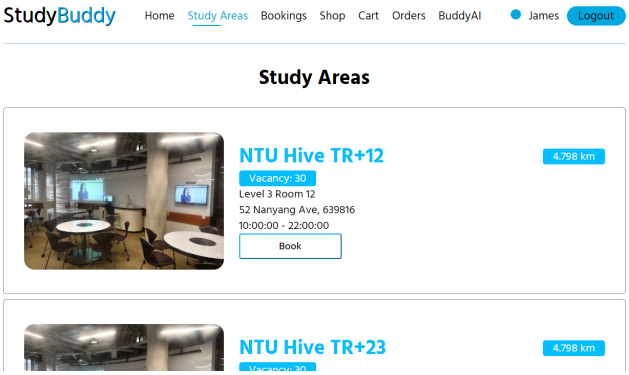
2. Login Page



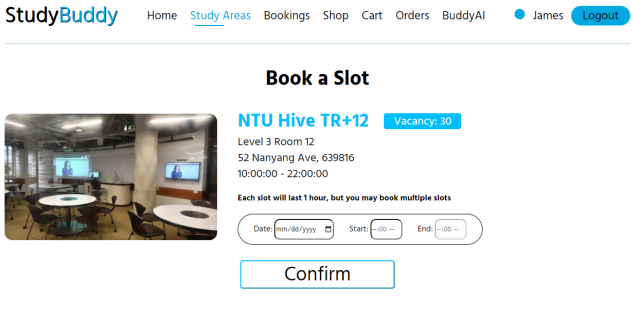
3. Home Page



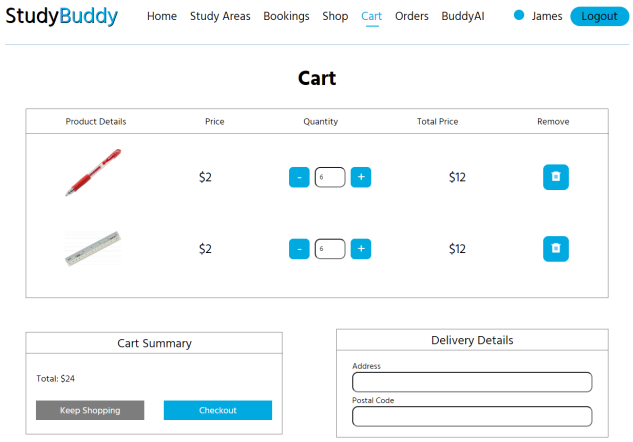
4. Available Study Areas Page



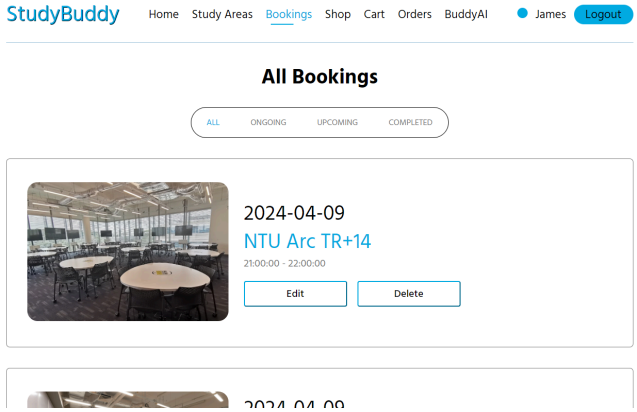
5. Create Booking Page



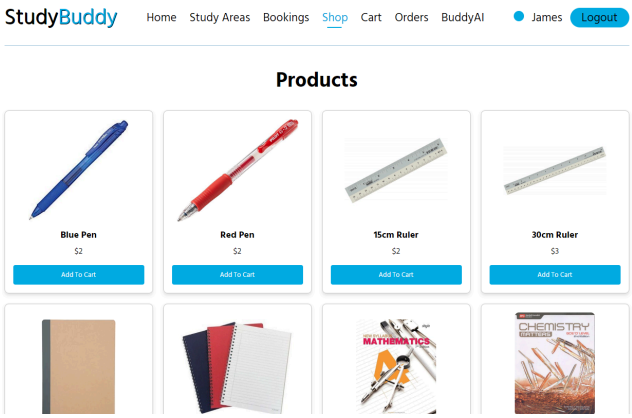
6. User Cart Page



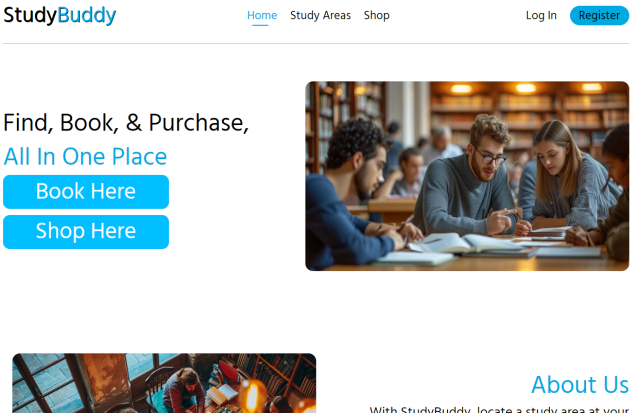
7. Current and Historical Bookings Page



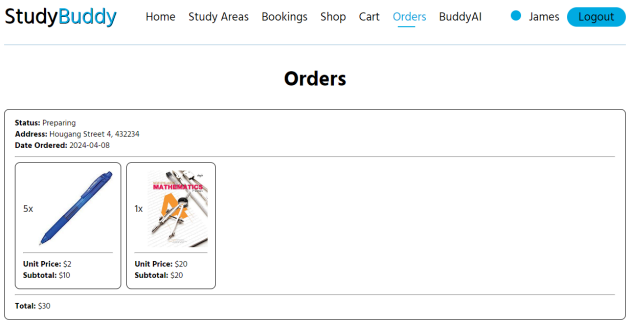
8. E-Commerce Store



9. Landing Page



10. Orders Page



11. Ask Buddy AI Page