
Final Report for LivingPlus

Prepared by

Muhammad Sibtain – 22i-0887

Abdullah Shakir – 22i-1138

Aqib Mehmood– 22p-9430

1 TABLE OF CONTENTS

2	Introduction	5
3	Functional Requirements	5
3.1	Hostel Registration.....	5
3.2	Room Management.....	6
3.3	Application Management.....	6
3.4	Tenant Profile Management.....	7
3.5	Complaint Management	7
3.6	Feedback Management	8
3.7	Parking Management	8
3.8	Payment Tracking	9
3.9	Invoice Generation.....	10
3.10	Notification Management	10
3.11	Tenant Account Management	10
3.12	Property Search	11
3.13	Property Viewing	12
3.14	Communication	12
3.15	Feedback and Reviews.....	13
3.16	Parking Requests (Tenant).....	13
4	Non-Functional Requirements	14
4.1	Performance Requirements.....	14
4.2	Safety Requirements	14
4.3	Security Requirements.....	14
4.4	Software Quality Attributes.....	14
4.5	Business Rules	14
5	User Stories	15
6	Product Backlog	32
7	Sprint 1 Backlog	36
8	Sprint 2 Backlog	37
9	Design Sprint 3	38
9.1	Sprint 3 items	38
9.2	Design Details.....	38
9.2.1	Send Notifications & Announcements.....	38
9.2.2	Generate and Send Invoices	38
9.2.3	Complaints and Feedback	38

9.2.4	Chat & Communication with Rental Owners	38
9.2.5	Track Rental Payments and Dues	38
10	Actual Implementation	39
10.1	Login Page.....	39
10.2	Signup Page.....	39
10.3	Dashboard	40
10.4	Owned Rentals	40
10.5	Register Hostel	41
10.6	Add Room	41
10.7	View Ratings.....	42
10.8	Generate and send invoices	42
10.9	Apply Rental	43
10.10	Chat and Communication	43
10.11	Give Feedback	44
10.12	Pay Bills.....	44
10.13	View Notifications	45
11	Product Burndown Chart	45
12	Sprint 3 Burndown Chart	46
13	Trello Board	46
13.1	Start	46
13.2	Sprint 2 Start	47
13.3	Sprint 2 END	47
13.4	Sprint 3 END	48
14	Black Box Testing	48
15	White Box Testing	49
15.1	Note	50
15.1.1	Covered	50
15.1.2	Not Covered	50
16	Work division	50
16.1	Muhammad Sibtain – 22i-0887	50
16.1.1	Contribution Area.....	50
16.1.2	Details.....	50
16.2	Abdullah Shakir – 22i-1138.....	50
16.2.1	Contribution Area.....	50
16.2.2	Details.....	50

16.3	Aqib Mehmood.....	51
16.3.1	Contribution Area.....	51
16.3.2	Details.....	51
17	Lessons Learnt.....	51

2 INTRODUCTION

The LivingPlus Rental and Hostel Management System is a full-stack web application developed to streamline the process of managing rental properties and hostel accommodations. Designed with the needs of landlords, hostel managers, and tenants in mind, the system aims to provide a seamless digital platform for listing, booking, payment management, and communication.

Built using the MERN stack — MongoDB, Express.js, React.js, and Node.js — the application follows the MVC architecture and leverages RESTful APIs to ensure a clean separation of concerns and scalable development practices. The frontend offers an intuitive and responsive interface for users, while the backend provides robust APIs to handle business logic and database operations.

LivingPlus addresses common challenges faced in traditional rental and hostel setups, such as scattered records, delayed payments, and lack of transparency in communication. The system incorporates features like user authentication, room availability tracking, payment logging, administrative dashboards, and a tenant-friendly booking process.

The project was developed using an agile methodology in a team of three students, progressing through multiple sprints. Each sprint focused on delivering key features iteratively, with continuous integration and testing to ensure quality and usability.

LivingPlus not only digitizes the property rental experience but also demonstrates the team's understanding of modern web development practices and software engineering principles.

3 FUNCTIONAL REQUIREMENTS

3.1 HOSTEL REGISTRATION

3.1.1 Description and Priority

This feature allows hostel owners to register their hostel profiles on the platform. This is a high-priority feature as it is essential for owners to list their properties.

Priority: High

3.1.2 Stimulus/Response Sequences

- **User Action:** Hostel owner navigates to the "Register Hostel" page.
- **System Response:** The system displays the hostel registration form.
- **User Action:** Hostel owner fills in the required details and submits the form.
- **System Response:** The system validates the input.
- **System Response:** If valid, the system saves the hostel profile and displays a confirmation message.
- **System Response:** If invalid, the system displays validation errors and prevents submission.

3.1.3 Functional Requirements

REQ-1: The system shall provide a user interface for hostel owners to register their hostel profiles.

REQ-2: The system shall provide fields for hostel details, including hostel name, address, capacity, and amenities.

REQ-3: The system shall validate all required fields before submission.

REQ-4: The system shall save the hostel profile upon successful submission.

REQ-5: The system shall display a confirmation message to the user after successful registration.
REQ-6: The system shall make the registered hostel visible in the owner's property list.
REQ-7: The system shall prevent submission if required fields are missing and display appropriate validation errors.

3.2 ROOM MANAGEMENT

3.2.1 Description and Priority

This feature enables hostel owners to view and update room availability and occupancy status. This is a high-priority feature, as it allows owners to efficiently manage their room inventory.

Priority: High

3.2.2 Stimulus/Response Sequences

- *User Action: Hostel owner navigates to the "Manage Rooms" page.*
- *System Response: The system displays a list of rooms with their occupancy status.*
- *User Action: Hostel owner updates the availability status of a room.*
- *System Response: The system updates the room status accordingly.*

3.2.3 Functional Requirements

REQ-8: The system shall provide a user interface for hostel owners to view a list of rooms.

REQ-9: The system shall display the current occupancy status of each room.

REQ-10: The system shall allow hostel owners to update the availability status of a room (e.g., from "Occupied" to "Available").

REQ-11: The system shall update the room status successfully upon request.

3.3 APPLICATION MANAGEMENT

3.3.1 Description and Priority

This feature allows hostel owners to review and approve tenant applications. This is a low-priority feature.

Priority: Low

3.3.2 Stimulus/Response Sequences

- *User Action: Hostel owner views pending tenant applications.*
- *System Response: The system displays the applications.*
- *User Action: Hostel owner approves an application.*
- *System Response: The system changes the application status to "Approved" and notifies the tenant.*

- User Action: Hostel owner rejects an application.
- System Response: The system changes the application status to "Rejected" and notifies the tenant.

3.3.3 Functional Requirements

REQ-12: The system shall provide a user interface for hostel owners to view pending tenant applications.

REQ-13: The system shall allow hostel owners to approve a tenant application.

REQ-14: The system shall change the application status to "Approved" when an application is approved.

REQ-15: The system shall notify the tenant of the application approval.

REQ-16: The system shall allow hostel owners to reject a tenant application.

REQ-17: The system shall change the application status to "Rejected" when an application is rejected.

REQ-18: The system shall notify the tenant of the application rejection

3.4 TENANT PROFILE MANAGEMENT

3.4.1 Description and Priority

This feature enables hostel owners to view tenant profiles and lease details. This feature is of moderate priority.

Priority: Moderate

3.4.2 Stimulus/Response Sequences

- User Action: Hostel owner views a tenant's profile.
- System Response: The system displays the tenant's personal information and lease details.

3.4.3 Functional Requirements

REQ-19: The system shall provide a user interface for hostel owners to view tenant profiles.

REQ-20: The system shall display tenant personal information and lease details.

3.5 COMPLAINT MANAGEMENT

3.5.1 Description and Priority

This feature allows hostel owners to receive and address tenant complaints. This is a high-priority feature.

Priority: High

3.5.2 Stimulus/Response Sequences

- User Action: Hostel owner views a tenant complaint.
- System Response: The system displays the complaint details.
- User Action: Hostel owner updates the complaint status or adds a resolution note.
- System Response: The system saves the changes.

3.5.3 Functional Requirements

REQ-21: The system shall provide a user interface for hostel owners to view tenant complaints.

REQ-22: The system shall allow hostel owners to update the status of a complaint.

REQ-23: The system shall allow hostel owners to add a resolution note to a complaint.

3.6 FEEDBACK MANAGEMENT

3.6.1 Description and Priority

This feature allows hostel owners to collect tenant feedback. This is a high-priority feature.

Priority: High

3.6.2 Stimulus/Response Sequences

- User Action: Hostel owner accesses the feedback section.
- System Response: The system displays all submitted feedback with ratings and comments.

3.6.3 Functional Requirements

REQ-24: The system shall provide a user interface for hostel owners to view tenant feedback.

REQ-25: The system shall display all submitted feedback, including ratings and comments.

3.7 PARKING MANAGEMENT

3.7.1 Description and Priority

This feature enables hostel owners to review and approve tenant parking requests. This feature is of moderate priority.

Priority: Moderate

3.7.2 Stimulus/Response Sequences

- User Action: Tenant submits a parking request with vehicle details.
- System Response: The system stores the request.
- User Action: Hostel owner reviews the parking request.

- System Response: The system displays the request details.
- User Action: Hostel owner approves or rejects the request.
- System Response: The system updates the request status.
- System Response: If approved, the system assigns a parking slot and notifies the tenant.

3.7.3 Functional Requirements

REQ-26: The system shall provide a user interface for tenants to submit parking requests.

REQ-27: The system shall allow tenants to enter vehicle details (e.g., make, model, license plate) when submitting a parking request.

REQ-28: The system shall provide a user interface for hostel owners to review parking requests.

REQ-29: The system shall display the request details, including tenant information and vehicle details.

REQ-30: The system shall allow hostel owners to approve or reject parking requests.

REQ-31: The system shall update the request status accordingly when a request is approved or rejected.

REQ-32: The system shall automatically assign an available parking slot based on predefined allocation rules.

REQ-33: The system shall notify the tenant with the assigned slot details and any relevant instructions upon approval.

3.8 PAYMENT TRACKING

3.8.1 Description and Priority

This feature allows hostel owners to track rental payments. This is a high-priority feature.

Priority: High

3.8.2 Stimulus/Response Sequences

- User Action: Hostel owner opens the payments section.
- System Response: The system displays a list of tenants with their payment status and due amounts.

3.8.3 Functional Requirements

REQ-34: The system shall provide a user interface for hostel owners to view rental transactions.

REQ-35: The system shall display a list of tenants with their payment status and due amounts.

3.9 INVOICE GENERATION

3.9.1 Description and Priority

This feature allows hostel owners to generate and send invoices. This is a high-priority feature.

Priority: High

3.9.2 Stimulus/Response Sequences

- User Action: Hostel owner generates an invoice.
- System Response: The system creates an invoice with accurate billing details and sends it to the tenant.

3.9.3 Functional Requirements

REQ-36: The system shall allow hostel owners to generate invoices.

REQ-37: The generated invoice shall contain accurate billing details.

REQ-38: The system shall send the generated invoice to the tenant's registered email.

3.10 NOTIFICATION MANAGEMENT

3.10.1 Description and Priority

This feature enables hostel owners to send notifications and announcements to tenants. This is a low-priority feature.

Priority: Low

3.10.2 Stimulus/Response Sequences

- User Action: Hostel owner drafts and sends a notification.
- System Response: The system sends the notification to all tenants

3.10.3 Functional Requirements

REQ-39: The system shall allow hostel owners to draft and send notifications.

REQ-40: The system shall ensure that all tenants receive the notification.

3.11 TENANT ACCOUNT MANAGEMENT

3.11.1 Description and Priority

This feature allows tenants to create accounts and manage their profiles. This feature is of moderate priority.

Priority: Moderate

3.11.2 Stimulus/Response Sequences

- User Action: Tenant enters valid information on the registration page and submits.
- System Response: The system creates a tenant account.
- User Action: Tenant updates profile details.
- System Response: The system updates the tenant's profile.
- User Action: Tenant requests a password reset.
- System Response: The system sends a reset link or OTP.

3.11.3 Functional Requirements

REQ-41: The system shall provide a user interface for tenants to create an account.

REQ-42: The system shall allow tenants to enter a valid email or phone number and complete the registration form.

REQ-43: The system shall successfully create a tenant account upon valid registration.

REQ-44: The system shall provide a user interface for tenants to update their profile details.

REQ-45: The system shall allow tenants to update their name, contact information, or preferences.

REQ-46: The system shall successfully update tenant profile details upon saving changes.

REQ-47: The system shall provide a user interface for tenants to reset their password.

REQ-48: The system shall allow tenants to request a password reset by providing their registered email or phone number.

REQ-49: The system shall send a reset link or OTP to allow tenants to set a new password.

3.12 PROPERTY SEARCH

3.12.1 Description and Priority

This feature allows tenants to search for rental properties using filters. This feature is of moderate priority.

Priority: Moderate

3.12.2 Stimulus/Response Sequences

- User Action: Tenant enters a keyword to search for properties.
- System Response: The system displays relevant properties.
- User Action: Tenant applies filters to refine the search.

- System Response: The system updates the search results based on the filters.
- User Action: Tenant sorts the search results.
- System Response: The system rearranges the results based on the sorting option.

3.12.3 Functional Requirements

REQ-50: The system shall provide a user interface for tenants to search for properties.

REQ-51: The system shall allow tenants to search for properties using keywords.

REQ-52: The system shall display relevant properties matching the keyword.

3.13 PROPERTY VIEWING

3.13.1 Description and Priority

This feature allows tenants to request property viewings or book units online. This feature is of moderate priority.

Priority: Moderate

3.13.2 Stimulus/Response Sequences

- User Action: Tenant requests a property tour.
- System Response: The system sends the request to the landlord.
- System Response: The system notifies the tenant of the confirmation or rejection of the tour request.
- System Response: The system sends reminders for scheduled viewings.
- User Action: Tenant cancels or reschedules a viewing.
- System Response: The system updates the booking and notifies the landlord

3.13.3 Functional Requirements

REQ-53: The system shall allow tenants to request a property tour (either in-person or virtual).

REQ-54: The system shall send the tour request to the landlord for approval.

REQ-55: The system shall notify the tenant of the confirmation or rejection of their tour request.

3.14 COMMUNICATION

3.14.1 Description and Priority

This feature enables communication between tenants and landlords. This feature is of moderate priority.

Priority: Moderate

3.14.2 Stimulus/Response Sequences

- User Action: Tenant initiates a chat with the landlord.
- System Response: The system allows sending and receiving messages.
- System Response: The system notifies tenants of new messages.

3.14.3 Functional Requirements

REQ-56: The system shall allow tenants to initiate a chat with the landlord.

REQ-57: The system shall allow tenants to send and receive messages within the platform.

REQ-58: The system shall notify tenants of new messages from landlords.

3.15 FEEDBACK AND REVIEWS

3.15.1 Description and Priority

This feature allows tenants to rate and review their rental experience. This is a high-priority feature.

Priority: High

3.15.2 Stimulus/Response Sequences

- User Action: Tenant provides a star rating and writes a review.
- System Response: The system stores the rating and review.
- System Response: The system displays reviews on the property listing.

3.15.3 Functional Requirements

REQ-59: The system shall allow tenants to provide a star rating (1-5) for a property.

REQ-60: The system shall allow tenants to write a review for a property.

REQ-61: The system shall display submitted reviews and ratings on the property listing.

3.16 PARKING REQUESTS (TENANT)

3.16.1 Description and Priority

This feature allows tenants to request a parking slot. This is a high-priority feature.

Priority: High

3.16.2 Stimulus/Response Sequences

- User Action: Tenant submits a parking request with vehicle details.
- System Response: The system records the request.

3.16.3 Functional Requirements

REQ-62: The system shall allow tenants to enter vehicle details (e.g., make, model, license plate) when submitting a parking request.

4 NON-FUNCTIONAL REQUIREMENTS

4.1 PERFORMANCE REQUIREMENTS

The system shall respond to **90% of user requests within 2 seconds** under normal load conditions.

4.2 SAFETY REQUIREMENTS

REQ-NF-7: The system shall ensure that user data (e.g., personal information, payment details) is stored securely to prevent unauthorized access.

4.3 SECURITY REQUIREMENTS

REQ-NF-8: The system shall use secure password storage techniques (e.g., hashing with salt) to protect user passwords.

REQ-NF-10: The system shall implement user roles and permissions to restrict access to sensitive data and functions (e.g., only hostel owners can manage their property listings).

4.4 SOFTWARE QUALITY ATTRIBUTES

REQ-NF-1: The system shall have a user-friendly interface that is easy to navigate and understand for both hostel owners and tenants.

REQ-NF-2: The system shall provide clear and helpful error messages to guide users in case of problems.

REQ-NF-3: The system shall be designed in a modular way to facilitate future maintenance and updates.

REQ-NF-4: The system should be able to function across different web browsers (Chrome, Firefox, Safari)

4.5 BUSINESS RULES

REQ-NF-5: The system shall enforce that only registered hostel owners can create and manage hostel listings.

REQ-NF-6: The system shall ensure that tenants can only submit applications for properties that are currently available.

REQ-NF-7: The system shall define the rules for allocating parking spots (e.g., first-come, first-served, priority for certain tenants).

5 USER STORIES

Story ID: 1	Story Title: Register a Hostel		Importance: High
User Story: As a Hostel Owner, I want to register my hostel profile, so that I can list my property on the platform.			Estimate: M
Acceptance Criteria: Scenario: Successful Registration <ul style="list-style-type: none"> • Given: Given I am logged into the system • When: I navigate to the "Register Hostel" page and fill in the required details • Then: the system should save the hostel profile and display a confirmation message • And: the hostel should be visible in my property list Scenario: Missing Required Information <ul style="list-style-type: none"> • Given: I am on the "Register Hostel" page • When: I submit the form without filling in required fields • Then: the system should show validation errors and prevent submission 		Type: <ul style="list-style-type: none"> <input type="checkbox"/> Search <input type="checkbox"/> Workflow <input checked="" type="checkbox"/> Manage Data <input type="checkbox"/> Payment <input type="checkbox"/> Report/View 	

Story ID:2	Story Title: View & Update Rooms		<div>Importance:</div> <div>High</div>
<div>User Story:</div> <p>As a Hostel Owner, I want to view and update available rooms and current occupancy status, so that I can manage room availability efficiently.</p>			<div>Estimate:</div> <div>L</div>
<div>Acceptance Criteria:</div> <div>Scenario: View Room List</div> <ul style="list-style-type: none"> • Given: Given I am on the "Manage Rooms" page • When: I view the list of rooms • Then: I should see the current occupancy status of each room <div>Scenario: Update Room Availability</div> <ul style="list-style-type: none"> • Given: a room is listed as "Occupied" • When: I set it to "Available" • Then: the room status should update successfully 		<div>Type:</div> <div> <input type="checkbox"/> Search <input type="checkbox"/> Workflow <input checked="" type="checkbox"/> Manage Data <input type="checkbox"/> Payment <input checked="" type="checkbox"/> Report/View </div>	

Story ID: 3	Story Title: Review Applicants		<div>Importance:</div> <div>Low</div>
User Story: As a Hostel Owner, I want to review and approve tenant applications, so that I can fill my hostel rooms with suitable tenants.			<div>Estimate:</div> <div>L</div>
Acceptance Criteria: Scenario: Approve Application <ul style="list-style-type: none"> Given: there are pending tenant applications When: I review an application and click "Approve" Then: the application status should change to "Approved" And: the tenant should receive a confirmation notification Scenario 2: Reject Application <ul style="list-style-type: none"> Given: there are pending tenant applications When: I review an application and click "Reject" Then: the application status should change to "Rejected" And: the tenant should receive a rejection notification 		<div>Type:</div> <div> <input type="checkbox"/> Search <input type="checkbox"/> Workflow <input checked="" type="checkbox"/> Manage Data <input type="checkbox"/> Payment <input type="checkbox"/> Report/View </div>	

Story ID: 4	Story Title: View Tenant Profiles		<div>Importance:</div> <div>Moderate</div>
<div>User Story:</div> <p>As a Hostel Owner, I want to view tenant profiles and lease details, so that I can manage tenant information efficiently.</p>			<div>Estimate:</div> <div>XL</div>
<div>Acceptance Criteria:</div> <ul style="list-style-type: none"> • Given: there are tenants in the hostel • When: I open a tenant's profile • Then: I should see their personal information and lease details 			<div>Type:</div> <div> <input type="checkbox"/> Search <input type="checkbox"/> Workflow <input type="checkbox"/> Manage Data <input type="checkbox"/> Payment <input checked="" type="checkbox"/> Report/View </div>

Story ID: 5	Story Title: Address Tenant	
User Story: As a Hostel Owner, I want to receive and address tenant complaints, so that I can improve their living experience.		Importance: High
		Estimate: XL
Acceptance Criteria: <ul style="list-style-type: none"> • Given: a tenant has submitted a complaint • When: I view the complaint • Then: I should be able to update its status and add a resolution note 		Type: <ul style="list-style-type: none"> <input type="checkbox"/> Search <input checked="" type="checkbox"/> Workflow <input type="checkbox"/> Manage Data <input type="checkbox"/> Payment <input type="checkbox"/> Report/View

Story ID: 6	Story Title: Collect Tenant Feedback	
User Story: As a Hostel Owner, I want to collect tenant feedback, so that I can enhance the quality of my hostel services.		Importance: High
		Estimate: XL
Acceptance Criteria: <ul style="list-style-type: none"> • Given: tenants have provided feedback • When: I access the feedback section • Then: I should see all submitted feedback with ratings and comments 		Type: <ul style="list-style-type: none"> <input type="checkbox"/> Search <input checked="" type="checkbox"/> Workflow <input type="checkbox"/> Manage Data <input type="checkbox"/> Payment <input type="checkbox"/> Report/View

Story ID: 7	Story Title: Accept Parking	<div>Importance:</div> <div>Moderate</div>
<div>User Story:</div> <p>As a Hostel Owner, I want to review and approve tenant parking requests so that I can allocate parking spaces efficiently and ensure proper management of available slots.</p>		<div>Estimate:</div> <div>M</div>
<div>Acceptance Criteria:</div> <p>Scenario 1: Landlord reviews a parking request</p> <ul style="list-style-type: none"> Given a tenant has submitted a parking request When the landlord navigates to the parking request management section Then the system should display the request details, including tenant information and vehicle details <p>Scenario 2: Landlord approves or rejects the request</p> <ul style="list-style-type: none"> Given the landlord is reviewing a parking request When they choose to approve or reject the request Then the system should update the request status accordingly <p>Scenario 3: System assigns and notifies the tenant of approval</p> <ul style="list-style-type: none"> Given the landlord has approved the parking request When a parking slot is allocated Then the system should notify the tenant with the assigned slot details and any relevant instructions 		<div>Type:</div> <div> <input type="checkbox"/> Search <input type="checkbox"/> Workflow <input type="checkbox"/> Manage Data <input type="checkbox"/> Payment <input checked="" type="checkbox"/> Report/View </div>

Story ID: 8	Story Title: Track Payments	<div>Importance: High</div> <div>Estimate: XL</div>
User Story: As a Hostel Owner, I want to track rental payments, so that I know which tenants have paid and who has pending dues.		
Acceptance Criteria: <ul style="list-style-type: none"> • Given: there are rental transactions • When: I open the payments section • Then: I should see a list of tenants with their payment status and due amounts 		Type: <ul style="list-style-type: none"> <input type="checkbox"/> Search <input type="checkbox"/> Workflow <input type="checkbox"/> Manage Data <input checked="" type="checkbox"/> Payment <input checked="" type="checkbox"/> Report/View

Story ID: 9	Story Title: Generate Invoices	<div>Importance: High</div> <div>Estimate: XL</div>
User Story: As a Hostel Owner, I want to generate and send invoices, so that tenants receive accurate billing information.		
Acceptance Criteria: <ul style="list-style-type: none"> • Given: a tenant's rent is due • When: I generate an invoice • Then: the invoice should contain accurate billing details and be sent to the tenant's registered email 		Type: <ul style="list-style-type: none"> <input type="checkbox"/> Search <input type="checkbox"/> Workflow <input type="checkbox"/> Manage Data <input checked="" type="checkbox"/> Payment <input type="checkbox"/> Report/View

Story ID: 10	Story Title: Send Notifications	<div>Importance:</div> <div>Low</div>
<div>User Story:</div> <div>As a Hostel Owner, I want to send notifications and announcements to tenants, so that I can keep them informed about important updates and events.</div>		<div>Estimate:</div> <div>L</div>
<div>Acceptance Criteria:</div> <ul style="list-style-type: none"> • Given: I have an important announcement • When: I draft and send a notification • Then: all tenants should receive the notification 		<div>Type:</div> <div> <input type="checkbox"/> Search <input checked="" type="checkbox"/> Workflow <input type="checkbox"/> Manage Data <input type="checkbox"/> Payment <input type="checkbox"/> Report/View </div>

Story ID: 11	Story Title: Create Tenant		Importance: Moderate
User Story: As a tenant, I want to create an account and manage my profile so that I can save my preferences and interact with the platform.			Estimate: S
Acceptance Criteria: Scenario 1: Tenant creates an account <ul style="list-style-type: none"> • Given: the tenant is on the registration page • When: they enter a valid email or phone number and complete the registration form • Then: their account should be successfully created Scenario 2: Tenant updates profile details <ul style="list-style-type: none"> • Given: the tenant is logged into their account • When: they update their name, contact info, or preferences and save changes • Then: the system should successfully update their profile Scenario 3: Tenant resets password <ul style="list-style-type: none"> • Given: the tenant has forgotten their password • When: they request a password reset and provide their registered email or phone number • Then: the system should send a reset link or OTP to allow them to set a new password 		Type: <ul style="list-style-type: none"> <input type="checkbox"/> Search <input type="checkbox"/> Workflow <input checked="" type="checkbox"/> Manage Data <input type="checkbox"/> Payment <input type="checkbox"/> Report/View 	

Story ID: 12	Story Title: Search properties		Importance: Moderate
User Story: As a tenant, I want to search for rental properties using filters like location, price, facilities and availability so that I can quickly find options that match my needs.			Estimate: M
Acceptance Criteria: Scenario 1: Tenant searches for properties using keywords <ul style="list-style-type: none"> • Given the tenant is on the property search page • When they enter a keyword related to a property (e.g., "hostel near university") • Then the system should display relevant properties matching the keyword Scenario 2: Tenant applies filters to refine search <ul style="list-style-type: none"> • Given the tenant is on the search results page • When they apply filters such as location, price range, property type, or facilities • Then the system should update the search results based on the selected filters Scenario 3: System returns relevant search results <ul style="list-style-type: none"> • Given the tenant has entered a keyword or applied filters • When they submit the search request • Then the system should return a list of properties that match the criteria Scenario 4: Tenant sorts search results <ul style="list-style-type: none"> • Given the tenant has a list of search results • When they select a sorting option (e.g., by price, rating, or proximity) • Then the system should rearrange the results accordingly 		Type: <input checked="" type="checkbox"/> Search <input type="checkbox"/> Workflow <input type="checkbox"/> Manage Data <input type="checkbox"/> Payment <input type="checkbox"/> Report/View	

Story ID: 13	Story Title: View Properties	
User Story: As a tenant, I want to request a property viewing or book a unit online so that I can inspect the property before committing to a rental.		Importance: Moderate
		Estimate: M
Acceptance Criteria: Scenario 1: Tenant requests an in-person or virtual property tour <ul style="list-style-type: none"> • Given the tenant is on the property listing page • When they request a property tour (either in-person or virtual) • Then the system should send the request to the landlord for approval Scenario 2: Tenant receives confirmation or rejection from the landlord <ul style="list-style-type: none"> • Given the tenant has requested a property tour • When the landlord approves or rejects the request • Then the system should notify the tenant of the confirmation or rejection • Scenario 3: System sends notifications for scheduled viewings <ul style="list-style-type: none"> • Given the tenant has an approved property viewing • When the scheduled time approaches • Then the system should send reminders to both the tenant and the landlord Scenario 4: Tenant cancels or reschedules a viewing <ul style="list-style-type: none"> • Given the tenant has a scheduled property tour • When they choose to cancel or reschedule the appointment • Then the system should update the booking and notify the landlord of the changes 		Type: <input type="checkbox"/> Search <input type="checkbox"/> Workflow <input type="checkbox"/> Manage Data <input type="checkbox"/> Payment <input checked="" type="checkbox"/> Report/View

Story ID: 14	Story Title: Communication		Importance: Moderate
As a tenant, I want to communicate with landlords through the platform’s messaging system so that I can clarify details and negotiate terms securely.			Estimate: L
Acceptance Criteria: Scenario 1: Tenant sends and receives messages within the platform <ul style="list-style-type: none">• Given the tenant is logged into their account and viewing a property listing• When they initiate a chat with the landlord• Then the system should allow them to send and receive messages Scenario 2: Tenant receives notifications for new messages <ul style="list-style-type: none">• Given the tenant has an active conversation with a landlord• When the landlord sends a message• Then the system should notify the tenant of the new message Scenario 3: Chat supports text, file attachments, and read receipts <ul style="list-style-type: none">• Given the tenant is in a chat conversation• When they send a text message or attach a file (e.g., ID proof, rental agreement)• Then the system should successfully deliver the message and display read receipts when the landlord views it Scenario 4: Messages are secure and encrypted <ul style="list-style-type: none">• Given the tenant is using the chat feature• When they send or receive a message• Then the system should ensure all messages are encrypted and securely stored		Type: <input type="checkbox"/> Search <input checked="" type="checkbox"/> Workflow <input type="checkbox"/> Manage Data <input type="checkbox"/> Payment <input type="checkbox"/> Report/View	

Story ID: 15	Story Title: Give Feedback	Importance: High
As a tenant, I want to rate and review my rental experience so that other tenants can make informed decisions based on real feedback.		Estimate: L
<p>Scenario 1: Tenant leaves a star rating and writes a review</p> <ul style="list-style-type: none"> • Given the tenant has completed their rental stay • When they navigate to the review section of the property • Then the system should allow them to provide a star rating (1-5) and write a review <p>Scenario 2: Reviews are visible on property listings</p> <ul style="list-style-type: none"> • Given the tenant has submitted a review • When other users view the property listing • Then the system should display the submitted review and rating <p>Scenario 3: Tenant can only review properties they have rented</p> <ul style="list-style-type: none"> • Given the tenant is logged into their account • When they attempt to submit a review for a property they have not rented • Then the system should prevent them from submitting the review and display an error message 		Type: <ul style="list-style-type: none"> <input type="checkbox"/> Search <input type="checkbox"/> Workflow <input type="checkbox"/> Manage Data <input type="checkbox"/> Payment <input checked="" type="checkbox"/> Report/View

<div> <div>Story ID: 16</div> <div>Story Title: Request Parking</div> </div> <div> <p>As a tenant, I want to request a parking slot through the platform so that I can secure a designated parking space for my vehicle.</p> </div>	<div>Importance:</div> <div>High</div> <div>Estimate:</div> <div>L</div>
<div> <p>Scenario 1: Tenant submits a parking request with vehicle details</p> <ul style="list-style-type: none"> Given the tenant needs a parking slot When they navigate to the parking request section Then the system should allow them to enter vehicle details (e.g., make, model, license plate) and submit a parking request <p>Scenario 2: System assigns a parking slot to the tenant</p> <ul style="list-style-type: none"> Given the tenant has submitted a parking request When the request is processed Then the system should automatically assign an available parking slot based on predefined allocation rules <p>Scenario 3: Tenant receives confirmation of parking slot allocation</p> <ul style="list-style-type: none"> Given the parking request has been approved When a parking slot is assigned Then the system should notify the tenant with parking details, including slot number and any relevant instructions </div>	<div>Type:</div> <div> <input type="checkbox"/> Search <input type="checkbox"/> Workflow <input type="checkbox"/> Manage Data <input type="checkbox"/> Payment <input checked="" type="checkbox"/> Report/View </div>

<div>Story ID: 17</div> <div>Story Title: Receive Notifications</div> <div>As a tenant, I want to receive notifications for new property listings, landlord responses, payment reminders, and maintenance updates so that I stay informed about important events.</div>	<div>Importance: High</div> <div>Estimate: XL</div>
<p>Scenario 1: Tenant receives real-time notifications for new property listings that match their preferences</p> <ul style="list-style-type: none"> • Given the tenant has set their property preferences • When a new listing that matches their criteria is added • Then the system should send a real-time notification to the tenant <p>Scenario 2: Tenant receives booking confirmations or cancellations</p> <ul style="list-style-type: none"> • Given the tenant has requested a property viewing or booking • When the landlord approves or cancels the request • Then the system should send a notification updating the tenant <p>Scenario 3: Tenant receives notifications for new messages from landlords</p> <ul style="list-style-type: none"> • Given the tenant has ongoing communication with a landlord • When the landlord sends a new message • Then the system should send a real-time notification to the tenant <p>Scenario 4: Tenant receives payment reminders</p> <ul style="list-style-type: none"> • Given the tenant has an upcoming or overdue payment • When the due date approaches or passes • Then the system should send a payment reminder notification <p>Scenario 5: Notifications can be received through in-app alerts</p> <ul style="list-style-type: none"> • Given the tenant is logged into the platform • When a new notification is triggered • Then the system should display an in-app alert <p>Scenario 6: Tenant can manage notification preferences</p> <ul style="list-style-type: none"> • Given the tenant wants to customize their notifications 	<div>Type:</div> <div> <input type="checkbox"/> Search <input type="checkbox"/> Workflow <input checked="" type="checkbox"/> Manage Data <input type="checkbox"/> Payment <input type="checkbox"/> Report/View </div>

<ul style="list-style-type: none">• When they access the notification settings• Then the system should allow them to enable or disable specific notifications	
--	--

6 PRODUCT BACKLOG

ID	User Story	Priority
1	As a Hostel Owner, I want to register my hostel profile, so that I can list my property on the platform.	High
2	As a Hostel Owner, I want to view and update available rooms and current occupancy status, so that I can manage room availability efficiently.	High
3	As a Hostel Owner, I want to review and approve tenant applications, so that I can fill my hostel rooms with suitable tenants.	Low
4	As a Hostel Owner, I want to view tenant profiles and lease details, so that I can manage tenant information efficiently.	Moderate
5	As a Hostel Owner, I want to receive and address tenant complaints, so that I can improve their living experience.	High
6	As a Hostel Owner, I want to collect tenant feedback, so that I can enhance the quality of my hostel services.	High
7	As a Hostel Owner, I want to review and approve tenant parking requests so that I can allocate parking spaces efficiently and ensure proper management of available slots.	Moderate
8	As a Hostel Owner, I want to track rental payments, so that I know which tenants have paid and who has pending dues.	High
9	As a Hostel Owner, I want to generate and send invoices, so that tenants receive accurate billing information.	High
10	As a Hostel Owner, I want to send notifications and announcements to tenants, so that I can keep them informed about important updates and events.	Low
11	As a tenant, I want to create an account and manage my profile so that I can save my preferences and interact with the platform.	Moderate
12	As a tenant, I want to search for rental properties using filters like location, price, facilities, and availability so that I can quickly find options that match my needs.	Moderate
13	As a tenant, I want to request a property viewing or book a unit online so that I can inspect the property before committing to a rental.	Moderate
14	As a tenant, I want to communicate with landlords through the platform's messaging system so that I can clarify details and negotiate terms securely.	Moderate
15	As a tenant, I want to rate and review my rental experience so that other tenants can make informed decisions based on real feedback.	High
16	As a tenant, I want to request a parking slot through the platform so that I can secure a designated parking space for my vehicle.	High
17	As a tenant, I want to receive notifications for new property listings, landlord responses, payment reminders, and maintenance updates so that I stay informed about important events.	High

ID: 1

User story: As a Hostel Owner, I want to register my hostel profile, so that I can list my property on the platform.

Priority: High

Estimate: M

ID: 2

User story: As a Hostel Owner, I want to view and update available rooms and current occupancy status, so that I can manage room availability efficiently.

Priority: High

Estimate: L

ID: 3

User story: As a Hostel Owner, I want to review and approve tenant applications, so that I can fill my hostel rooms with suitable tenants.

Priority: Low

Estimate: L

ID: 4

User story: As a Hostel Owner, I want to view tenant profiles and lease details, so that I can manage tenant information efficiently

Priority: Moderate

Estimate: XL

ID: 5

User story: As a Hostel Owner, I want to receive and address tenant complaints, so that I can improve their living experience.

Priority: High

Estimate: XL

ID: 6

User story: As a Hostel Owner, I want to collect tenant feedback, so that I can enhance the quality of my hostel services.

Priority: High

Estimate: XL

ID: 7

User story: As a Hostel Owner, I want to review and approve tenant parking requests so that I can allocate parking spaces efficiently and ensure proper management of available slots.

Priority: Moderate

Estimate: M

ID: 8

User story: As a Hostel Owner, I want to track rental payments, so that I know which tenants have paid and who has pending dues.

Priority: High

Estimate: XL

ID: 9

User story: As a Hostel Owner, I want to generate and send invoices, so that tenants receive accurate billing information.

Priority: High

Estimate: XL

ID: 10

User story: As a Hostel Owner, I want to send notifications and announcements to tenants, so that I can keep them informed about important updates and events.

Priority: Low

Estimate: L

ID: 11

User story: As a tenant, I want to create an account and manage my profile so that I can save my preferences and interact with the platform.

Priority: Moderate

Estimate: S

ID: 12

User story: As a tenant, I want to search for rental properties using filters like location, price, facilities and availability so that I can quickly find options that match my needs.

Priority: Moderate

Estimate: M

ID: 13

User story: As a tenant, I want to request a property viewing or book a unit online so that I can inspect the property before committing to a rental.

Priority: Moderate

Estimate: M

ID: 14

User story: As a tenant, I want to communicate with landlords through the platform's messaging system so that I can clarify details and negotiate terms securely.

Priority: Moderate

Estimate: L

ID: 15

User story: As a tenant, I want to rate and review my rental experience so that other tenants can make informed decisions based on real feedback.

Priority: High

Estimate: L

ID: 16

User story: As a tenant, I want to request a parking slot through the platform so that I can secure a designated parking space for my vehicle.

Priority: High

Estimate: L

ID: 17

User story: As a tenant, I want to receive notifications for new property listings, landlord responses, payment reminders, and maintenance updates so that I stay informed about important events.

Priority: High

Estimate: XL

7 SPRINT 1 BACKLOG

Sprint Goal:

To establish the core functionality for hostel owners to register their properties and manage basic applicant data.

User Stories Included in Sprint 1:

Story 1: Register a Hostel

As a Hostel Owner, I want to register my hostel profile, so that I can list my property on the platform.

Story 2: View & Update

As a Hostel Owner, I want to view and update available rooms and current occupancy status, so that I can manage room availability efficiently.

Story 11: Create Tenant

As a tenant, I want to create an account and manage my profile so that I can save my preferences and interact with the platform.

Story 12: Search properties

As a tenant, I want to search for rental properties using filters like location, price, facilities and availability so that I can quickly find options that match my needs.

Story 13: View Properties

As a tenant, I want to request a property viewing or book a unit online so that I can inspect the property before committing to a rental.

8 SPRINT 2 BACKLOG

Story 3: Review Applicants

As a Hostel Owner, I want to review and approve tenant applications, so that I can fill my hostel rooms with suitable tenants.

Story 16: Request Parking

As a tenant, I want to request a parking slot through the platform so that I can secure a designated parking space for my vehicle.

Story 7: Accept Parking

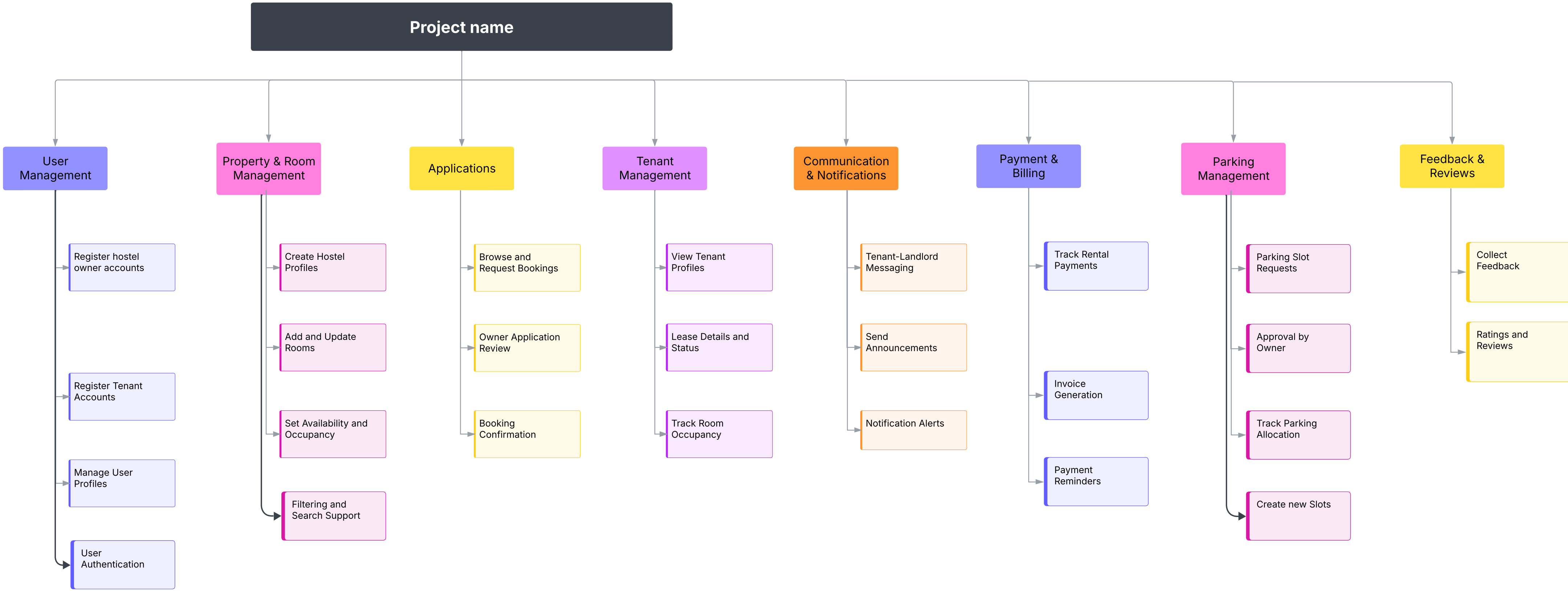
As a Hostel Owner, I want to review and approve tenant parking requests so that I can allocate parking spaces efficiently and ensure proper management of available slots.

Story 15: Give Feedback

As a tenant, I want to rate and review my rental experience so that other tenants can make informed decisions based on real feedback.

Story 15: Collect Tenant Feedback

As a Hostel Owner, I want to collect tenant feedback, so that I can enhance the quality of my hostel services.





Identifying Subsystems

LivingPlus (Top-Level Package): This is the main package, representing the entire application.

frontend: This package encapsulates all the client-side code, built with React.js.

- **src:** The source directory, a standard convention in React projects, containing the main application code.
- **components:** This sub-package holds the reusable UI components of the application (e.g., login form, sidebar, dashboard).
- **CSS:** Contains the stylesheets for styling the application.
- **tests:** Includes tests for the frontend components.
- **Uses (Dependency Arrow to components):** Indicates that the src package uses or depends on the backend, to access the database.
- **Imports (Dependency Arrow from components):** the components package in test imports main components so that they can be tested.

backend: This package contains the server-side code, built with Node.js and Express.js.

- **models:** Defines the data structures for the application, using Mongoose to interact with MongoDB. (e.g., schemas for rentals, Tenant, owner, Payment).
- **routes:** Handles the API endpoints that define how the server responds to requests (e.g., /rentals, /tenants).
- **tests:** Includes tests for the backend logic.
- **uses (Dependency Arrow to models and routes):** Indicates that the frontend package uses both the models (for data access) and routes (to define the API).
- **imports (Dependency Arrow from routes):** Suggests that the routes package uses models.

LivingPlus

frontend

src

components

css

tests

components

Imports

uses

backend

models

uses

routes

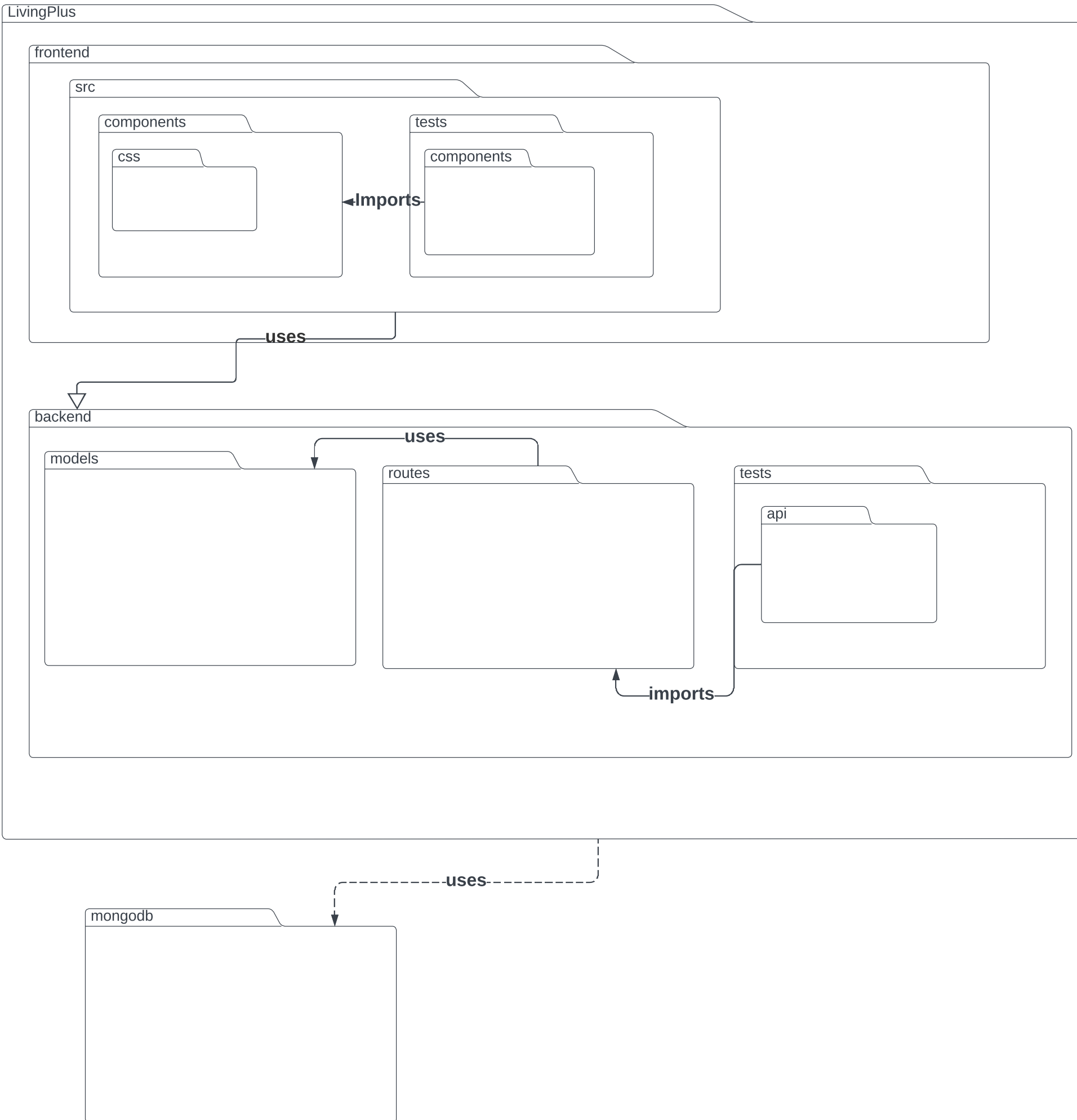
tests

api

imports

uses

mongodb

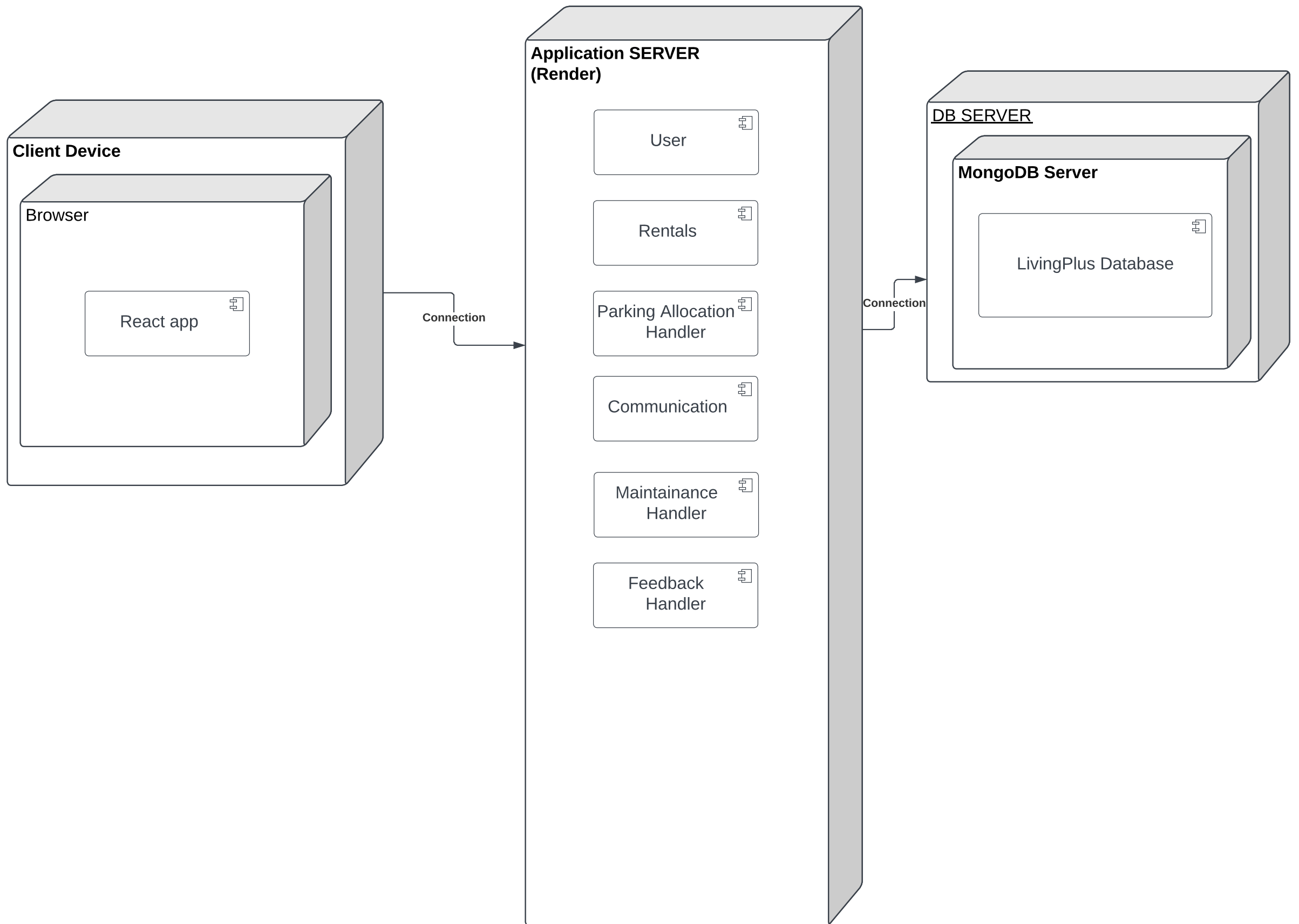


Architecture Styles

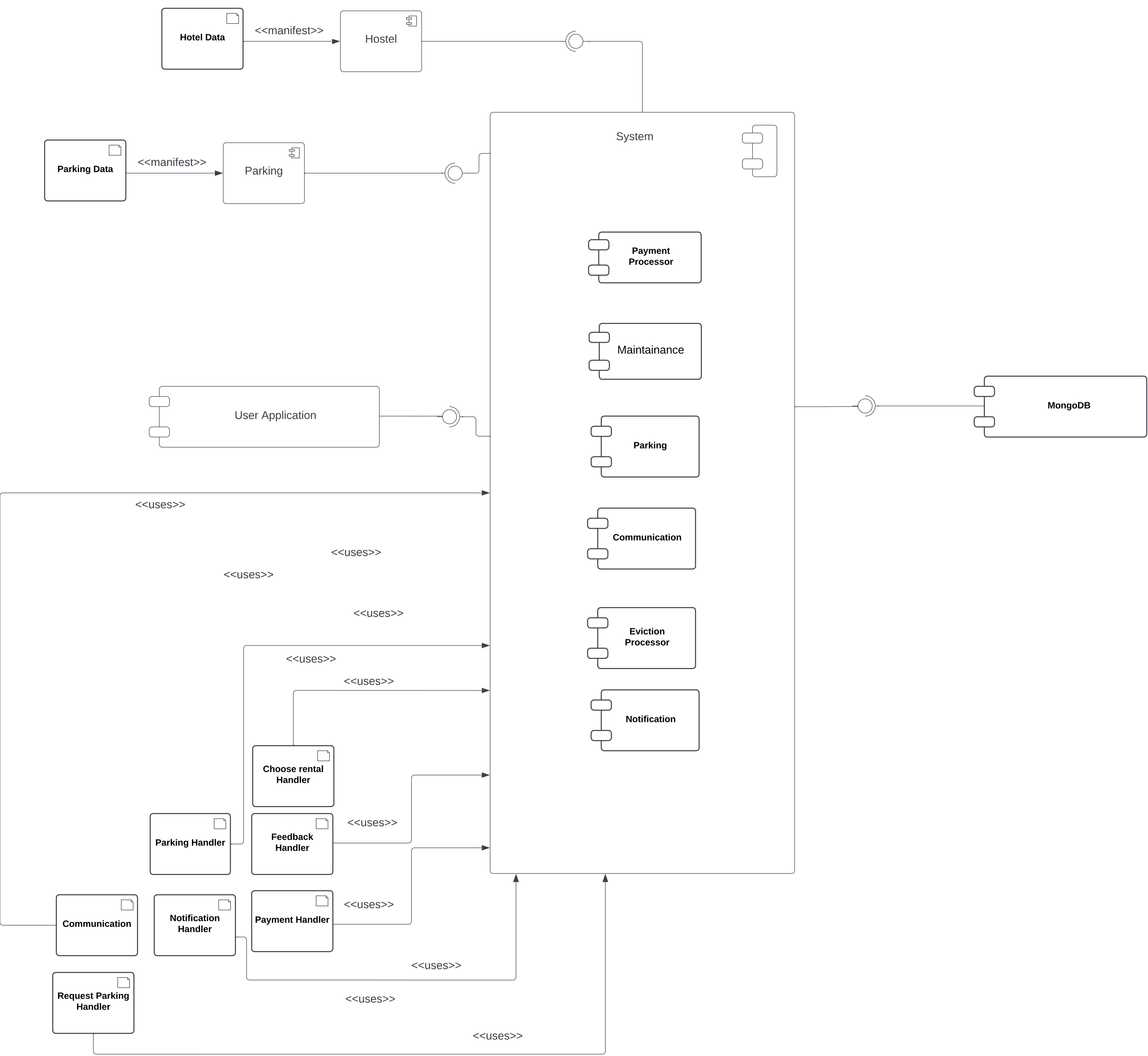
The LivingPlus web-based application is built using the MERN stack (MongoDB, Express.js, React.js, Node.js) and follows the **Model-View-Controller (MVC)** architectural style.

- **Model:** The data layer is handled using **MongoDB**, where all rental management-related data (e.g., properties, tenants, payments) are stored. The models define the structure of the database collections and are used to interact with the database.
- **View:** The frontend is developed using **React.js**, which acts as the View in the MVC pattern. React is responsible for presenting data to users and providing a dynamic, responsive user interface. It communicates with the backend via APIs.
- **Controller:** The controllers are implemented in **Express.js** within the Node.js backend. Controllers handle the business logic, processing incoming HTTP requests, interacting with the models, and sending appropriate responses back to the client.

By using MVC, LivingPlus separates concerns clearly, making the system more organized, easier to maintain, scalable, and testable. Each part (Model, View, Controller) is independently developed and maintained, improving collaboration and future extension of the application.

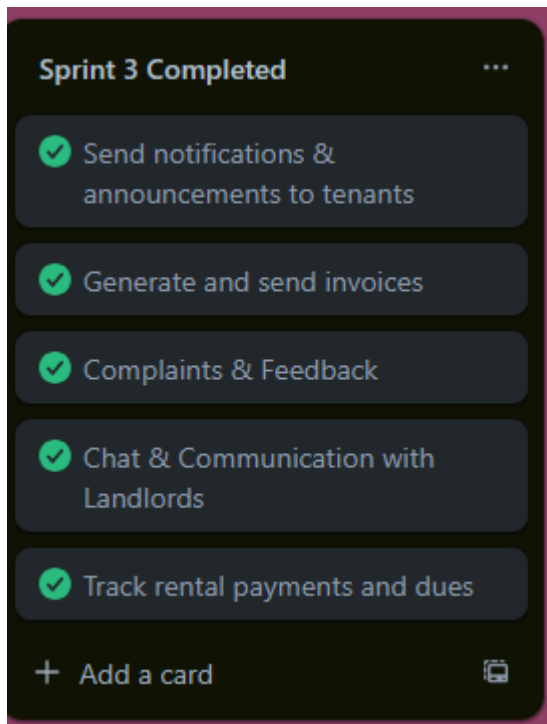


8. Component Diagram



9 DESIGN SPRINT 3

9.1 SPRINT 3 ITEMS



The goal of this sprint was to enhance communication and financial management functionalities within the application. The features designed include: tenant notifications, invoice generation, a system for complaints and feedback, landlord-tenant chat, and rental payment tracking.

9.2 DESIGN DETAILS

9.2.1 Send Notifications & Announcements

This feature allows landlords or property managers to send notifications and announcements to tenants.

9.2.2 Generate and Send Invoices

This feature allows landlords/managers to generate invoices for rent and other charges, and send them to tenants.

9.2.3 Complaints and Feedback

This feature allows tenants to submit complaints and feedback to rental owners.

9.2.4 Chat & Communication with Rental Owners

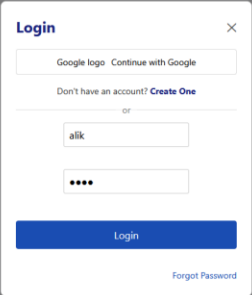
This feature allows direct communication between Tenants and rental owners.

9.2.5 Track Rental Payments and Dues

This feature allows rental owner & tenants to track unpaid dues and also pay them.

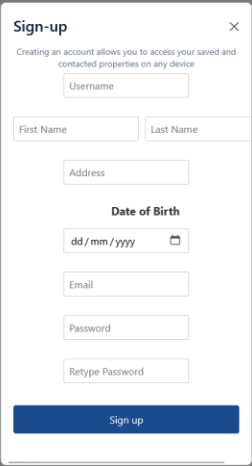
10 ACTUAL IMPLEMENTATION

10.1 LOGIN PAGE



A login form titled "Login" with a close button (X) in the top right corner. It features a "Google logo" and a "Continue with Google" button. Below this, it asks "Don't have an account? [Create One](#)". The form includes input fields for "Email" (containing "alik") and "Password" (masked with dots). A blue "Login" button is at the bottom, with a "Forgot Password" link below it.

10.2 SIGNUP PAGE



A sign-up form titled "Sign-up" with a close button (X) in the top right corner. It includes a sub-header: "Creating an account allows you to access your saved and contacted properties on any device". The form contains input fields for "Username", "First Name", "Last Name", "Address", "Date of Birth" (with a calendar icon), "Email", "Password", and "Retype Password". A blue "Sign up" button is at the bottom.

10.3 DASHBOARD

Living+ User Dashboard

Start Listing

A

Home

Dashboard

Messages

Profile

Settings

Logout

User Dashboard

User Information

Name: Ali Khan

Email: alik@gmail.com

Age: 25

Rented Rentals

You haven't rented any rentals yet.

Owned Rentals

Rental Name	Address	Available Rooms	Total Rooms
	123 Elm St.	4	4
	456 Maple Ave.	2	2

10.4 OWNED RENTALS

Living+ Owned Rentals

Start Listing

A

Home

Dashboard

Messages

Profile

Settings

Logout

Owned Rentals

Rental Name	Address	Available Rooms	Total Rooms
	123 Elm St.	4	4
	456 Maple Ave.	2	2

10.5 REGISTER HOSTEL

Living+Register Rental

🏠

🛡️

🔒

🏠

+

📅

📅

⚙️

📁

🔔

Start Listing

A

Hostel Details

Hostel Name

Address

Amenities

Capacity

Available Rooms

10.6 ADD ROOM

🛡️

🔒

🏠

+

📅

📅

⚙️

📁

✉️

🔗

Add Room

Select Rental

Select a rental

Room Type

Select room type

Description

Price

Room Picture (URL)

Add Room

10.7 VIEW RATINGS

Living+View Ratings

Sort (High to Low)

Alice Hostel

123 Elm St.

GymSwimming PoolGarden

Capacity: 4

Ratings☆☆☆☆0.0 / 5

Feedback

No feedback available

Dilawar Height

456 Maple Ave.

GymGarden

Capacity: 2

Ratings☆☆☆☆0.0 / 5

Feedback

No feedback available

10.8 GENERATE AND SEND INVOICES

Living+Payment

Start Listing

A

Payment

Select Tenant

Search Tenant

Amount

Amount

Status

Status

Sort

Tenant	Amount	Due Date	Status
Alex Jones	\$25000	09/05/2025	Pending
Alex Jones	\$100	30/04/2025	Pending
Alex Jones	\$100	30/04/2025	Pending
Alex Jones	\$100	30/04/2025	Pending
Alex Jones	\$100	30/04/2025	Pending
Alex Jones	\$100	30/04/2025	Pending
Alex Jones	\$100	30/04/2025	Pending

Select Tenant

Select Tenant

Due Date

dd / mm / yyyy

Amount

Enter amount

Generate Invoice

42

10.9 APPLY RENTAL

Living+ Apply Rental

Start Listing A

Home

Shield

Heart

Home

+

Calendar

Settings

Menu

Chat

Choose Rental

Choose Rental

Shanaz Hostel

Choose Room

67dd09289a81ff778eb1e

Apply

Available Properties

single Room

Rental: Shanaz Hostel

Status: 0

Price: \$5000/month

double Room

Rental: Shanaz Hostel

Status: 0

Price: \$3500/month

double Room

Rental: Shanaz Hostel

Status: 0

Price: \$12000/month

10.10 CHAT AND COMMUNICATION

Living+ Chats

Start Listing A

Home

Grid

Chat

Shield

Heart

Home

Chats

Property: N/A

Address: 25 Cedar Dr.

Last message: I am waiting for your response, mister!

Hello

I am waiting for your response, mister!

10.11 GIVE FEEDBACK

[illegible]

10.12 PAY BILLS

Living+Payment

Start Listing

Home

Shield

Key

Home

+

Calendar

Settings

Document

Chat

Payment

ti Sort

Due Date

Filter by date

Amount

Filter by amount

Status

Filter by status

\$25000.00

Due: 09/05/2025

Print Bill

Pay Bill

\$100.00

Due: 30/04/2025

Print Bill

Pay Bill

\$100.00

Due: 30/04/2025

Print Bill

Pay Bill

\$100.00

Due: 30/04/2025

Print Bill

Pay Bill

\$100.00

Due: 30/04/2025

Print Bill

Pay Bill

10.13 VIEW NOTIFICATIONS

Living+View Notifications

Start Listing

A

Notifications

Date

Filter by date

Description

Filter by description

Sort

2025-03-26T13:25:46.665043

Late rent payment warning.

2025-03-01T13:25:46.665043

Welcome to your new home!

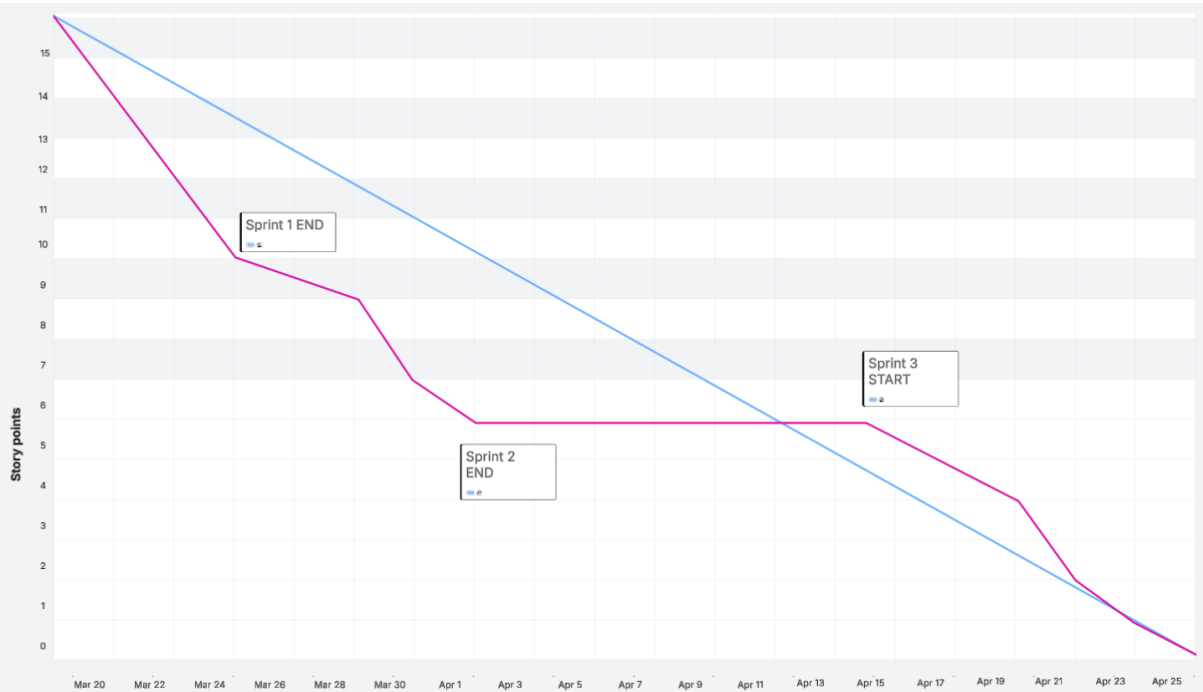
2025-02-19T13:25:46.665043

Upcoming rent payment reminder.

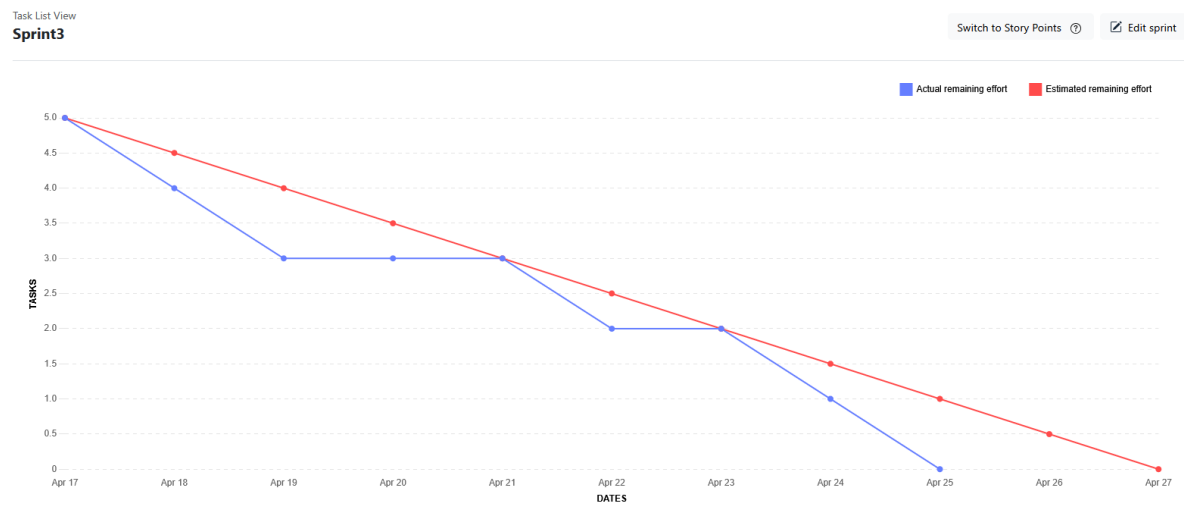
2025-02-15T13:25:46.665043

Lease renewal reminder.

11 PRODUCT BURNDOWN CHART



12 SPRINT 3 BURNDOWN CHART



13 TRELLO BOARD

13.1 START

The free trial of Premium has ended for Trello Workspace. [Explore plans](#) [Add payment method](#) X

LivingPLUS ☆ Workspace visible Board

Burndowns by Corrello Power-Ups Automation Filters Share ...

Product Backlog ...

- Approve tenant applications
- Assign rooms
- Track rental payments and dues
- Generate and send invoices
- Send notifications & announcements to tenants
- User Registration & Profile Management
- Property Search & Filters
- Booking & Viewing Requests
- Chat & Communication with Landlords
- Rental Agreement & Payment Handling
- Review & Rating System
- Maintenance Requests & Issue Reporting
- + Add a card

Sprint 1 Backlog ...

- + Add a card

Completed ...

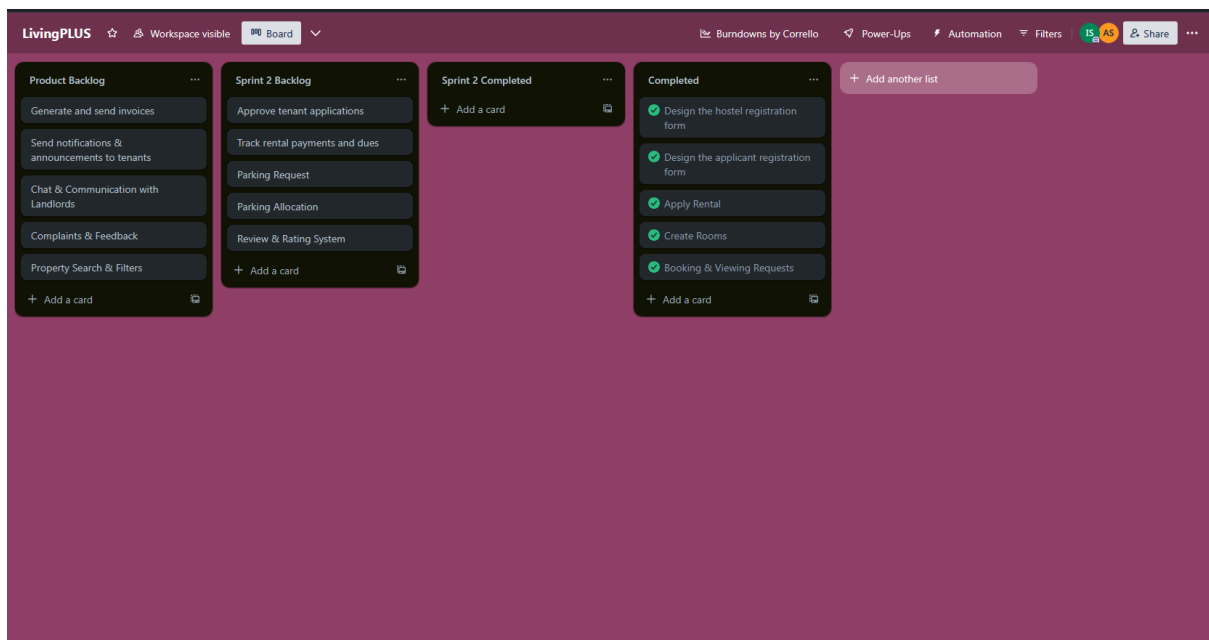
- ✓ Design the hostel registration form
- ✓ Design the applicant registration form
- ✓ Apply Rental
- ✓ Create Rooms
- ✓ Booking & Viewing Requests
- ✓ Property Search & Filters
- + Add a card

In Progress ...

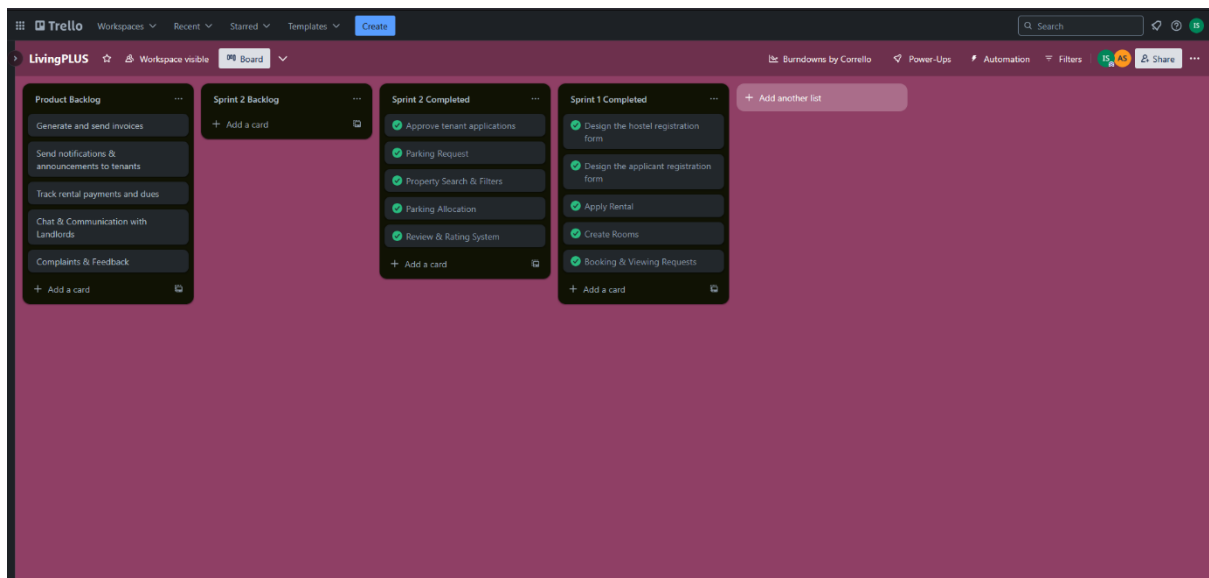
- + Add a card

+ Add another list

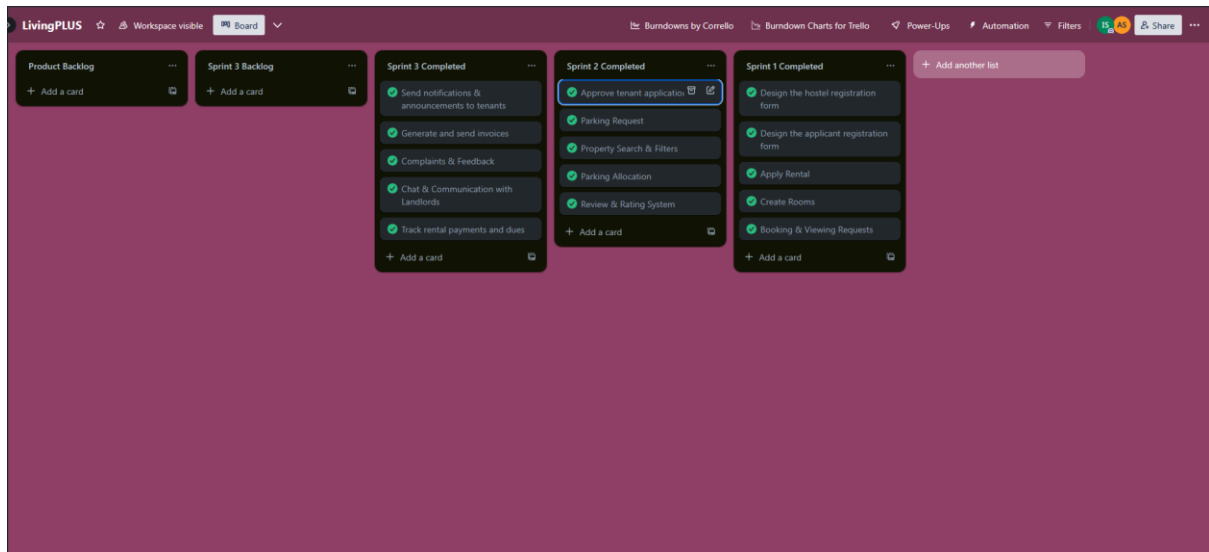
13.2 SPRINT 2 START



13.3 SPRINT 2 END



13.4 SPRINT 3 END



14 BLACK BOX TESTING

PASS tests/api/applicant.test.js (11.116 s)

Applicant API

- ✓ POST /applicant - should create a new applicant (240 ms)
- ✓ POST /applicant - should return 400 if fields are missing (23 ms)
- ✓ GET /applicant - should return list of applicants (9 ms)
- ✓ GET /applicant/:userName - should return specific applicant (8 ms)
- ✓ GET /applicant/:userName - should return 404 for invalid userName (9 ms)
- ✓ PUT /applicant/:id - should update applicant (16 ms)
- ✓ DELETE /applicant/:id - should delete applicant (14 ms)
- ✓ DELETE /applicant/:id - should return 404 if not found (7 ms)

Rental API

- ✓ POST /rentals - should create a new rental (14 ms)
- ✓ POST /rentals - should return 400 if fields are missing (4 ms)
- ✓ GET /rentals - should return list of rentals (5 ms)
- ✓ GET /rentals/:id - should return specific rental (7 ms)
- ✓ GET /rentals/:id - should return 404 for invalid ID (6 ms)
- ✓ PUT /rentals/:id - should update rental (12 ms)
- ✓ DELETE /rentals/:id - should delete rental (6 ms)
- ✓ DELETE /rentals/:id - should return 404 if not found (9 ms)

Room API

- ✓ POST /rooms - should create a new room (14 ms)
- ✓ GET /rooms - should return all rooms (5 ms)
- ✓ GET /rooms/:id - should return specific room (6 ms)
- ✓ PUT /rooms/:id - should update room (12 ms)
- ✓ DELETE /rooms/:id - should delete room (8 ms)
- ✓ DELETE /rooms/:id - should return 404 if not found (5 ms)
- ✓ GET /rooms/:id - should return 404 for invalid room ID (5 ms)
- ✓ POST /rooms - should return 400 for missing fields (7 ms)

Test Suites: 1 passed, 1 total

Tests: 24 passed, 24 total

Snapshots: 0 total

15 WHITE BOX TESTING

PASS

src/tests/components/SignUp.test.jsx (7.653 s)

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
All files	83.79	71.68	87.5	84.23	
Footer.jsx	100	100	100	100	
Header.jsx	100	50	100	100	10
HomePage.jsx	100	100	100	100	
Login.jsx	100	75	100	100	31
Sidebar.jsx	75	100	50	75	56,73
SignUp.jsx	53.84	33.33	80	52	38-67
dashboard.jsx	92.3	85.71	100	93.87	32-33,53
owner-sidebar.jsx	100	100	100	100	
paymentOwner.jsx	71.65	58.49	69.23	72	42-43,74-75,87-88,123,142,195-241,262-263,268-269,303,326-336,413-422
register-hostel.jsx	100	100	100	100	
renter-sidebar.jsx	100	100	100	100	
view-notifications.jsx	93.22	78.57	100	93.22	44-52
view-ratings.jsx	90.16	80.76	100	91.52	32-33,50,66,89

Test Suites: 5 failed, 6 passed, 11 total

Tests: 5 failed, 52 passed, 57 total

Snapshots: 0 total

Time: 13.279 s

Ran all test suites.

All files

83.79% Statements 338/398 71.68% Branches 338/384 87.5% Functions 71/80 84.23% Lines 338/398

Press n or j to go to the next uncovered block, b, p or k for the previous block.

Filter

File	Statements	Branches	Functions	Lines
Footer.jsx	100%	100%	100%	100%
Header.jsx	100%	50%	100%	100%
HomePage.jsx	100%	100%	100%	100%
Login.jsx	100%	75%	100%	100%
Sidebar.jsx	75%	100%	50%	75%
SignUp.jsx	53.84%	33.33%	80%	52%
dashboard.jsx	92.3%	85.71%	100%	93.87%
owner-sidebar.jsx	100%	100%	100%	100%
paymentOwner.jsx	71.65%	58.49%	69.23%	72%
register-hostel.jsx	100%	100%	100%	100%
renter-sidebar.jsx	100%	100%	100%	100%
view-notifications.jsx	93.22%	78.57%	100%	93.22%
view-ratings.jsx	90.16%	80.76%	100%	91.52%

Code coverage generated by Istanbul at 2025-04-26T12:12:53.981Z

All files HomePage.jsx

100% Statements 4/4 100% Branches 12/12 100% Functions 4/4 100% Lines 4/4

Press n or j to go to the next uncovered block, b, p or k for the previous block.

```
1 import React, { useState } from "react";
2 import "../css/HomePage.css";
3
4 const HomePage = () => {
5   // State for active tab
6   const [activeTab, setActiveTab] = useState("mission");
7
8   return (
9     <div className="home-page">
10       <div className="header">
11         <div className="logo">Logo</div>
12         <div className="nav-links">
13           <a href="#" className={activeTab}>Home</a>
14           <a href="#about">About us</a>
15           <a href="#manage">Manage property</a>
16           <a href="#dashboard">Dashboard</a>
17         </div>
18         <div className="auth-buttons">
19           <a href="/login" className="login-btn">Login</a>
20           <a href="/signup" className="signup-btn">Sign-up</a>
21         </div>
22       </div>
23       <div className="hero-section">
24         <div className="hero-content">
25           <h2>Your Perfect Rental Property</h2>
26           <p>Connecting users directly with property managers to discover their dream home</p>
27           <button className="get-started-btn">Get Started</button>
28         </div>
29       </div>
30       <div className="featured-properties">
31         <h2>Featured Listings</h2>
32       </div>
33     </div>
34   );
35 }
```


15.1 NOTE

15.1.1 Covered

The core functionality of the major React components is well covered. This includes testing the rendering of components, user interactions such as button clicks, form submissions, and API call handling (mocked). Components like the Dashboard, Property Management page, and Payment Owner page have good coverage, focusing on the main user flows and verifying correct data fetching and display.

15.1.2 Not Covered

Some areas, particularly complex form elements like the **Date Picker** and inputs involving third-party libraries like icons rendering etc, are not fully covered. Jest and React Testing Library have limitations in easily selecting and simulating interactions with custom third-party UI components.

Additionally, the **Signup page** has reduced coverage because of multiple nested try-catch blocks and asynchronous validation logic, which made it difficult for Jest to identify and assert all execution paths. Some error handling branches were also harder to trigger naturally through user events without over-mocking backend failures.

Therefore, parts of the signup and authentication flows have slightly lower test coverage.

16 WORK DIVISION

16.1 MUHAMMAD SIBTAIN – 22I-0887

16.1.1 Contribution Area

Backend Development, Frontend Development (React), Testing (Jest)

16.1.2 Details

Sibtain focused on implementing the Payment Tracking, Payment Generation, Invoice Generation, and Owner Dashboard functionalities, covering User Stories 8 and 9. He also developed the Applicant Approval system (User Story 3) and worked on the Notifications and Announcements features (User Stories 10 and 17), enabling hostel owners to communicate with tenants. In addition, Sibtain wrote the majority of white-box and black-box test cases using Jest and React Testing Library to ensure high test coverage. He also contributed to the UI/UX design for tenant-related views, optimizing user experience for payment workflows and other interactions.

16.2 ABDULLAH SHAKIR – 22I-1138

16.2.1 Contribution Area

Backend Development (Node.js, Express), Frontend Integration (React)

16.2.2 Details

Abdullah led the development of the Chat/Messaging System between tenants and hostel owners (User Story 14). He built the backend APIs using Express.js and MongoDB and integrated them with the frontend. Additionally, he developed key features such as the Owner Dashboard, Property Search (User Story 12), Hostel Registration (User Story 1),

Parking Request and Approval systems (User Stories 7 and 16), and Applicant Approval functionality (User Story 3), contributing to both backend logic and frontend integration.

16.3 AQIB MEHMOOD

16.3.1 Contribution Area

Authentication Module (Signup/Login), Frontend Development

16.3.2 Details

Aqib contributed primarily to the authentication and basic frontend development of the LivingPlus project. He developed the Signup and Login pages for both tenants and hostel owners, ensuring proper form validation, error handling, and integration with backend authentication APIs. Additionally, he handled user profile creation and updates, and managed session handling through localStorage, ensuring smooth login/logout functionality.

17 LESSONS LEARNT

Throughout the development of the LivingPlus project, the team gained invaluable experience in both technical and collaborative aspects. On the technical side, we learned the importance of modular architecture and clean code practices, especially when working with a MERN stack, as it ensured that the project remained scalable and maintainable. We also gained a deeper understanding of frontend-backend integration, particularly when managing complex data flows between React and Express. Another key lesson was the significance of unit testing and test coverage — especially using tools like Jest and React Testing Library — we also learned that writing test cases isn't as easy as it initially appeared. While we knew testing was important for ensuring code quality, we didn't fully appreciate the time and effort required to write comprehensive and meaningful tests. As we worked through the project, we realized that tests aren't just about covering basic functionality — they need to account for edge cases, interactions between different components, and asynchronous behaviors. We learned the importance of clear communication and task distribution, ensuring that each team member's strengths were leveraged effectively. Time management and consistent project planning through tools like Trello also played a crucial role in keeping the development process on track. Overall, the project taught us how to efficiently handle both individual and group responsibilities, adapt to challenges, and continuously improve our development practices.