Use Cases Document

Hostel Room Allocation and Maintenance Management System

Course: Software Engineering

Date: December 10, 2025

---

1. Introduction

This document outlines the main use cases for our Hostel Room Allocation and Maintenance Management System. These use cases describe how different users interact with the system to accomplish their goals. We've identified the key workflows that students and staff need to perform, and documented how the system supports these activities.

---

2. System Users

2.1 Primary Users

Students: The main end users who need to request hostel rooms and report maintenance issues. They need a simple way to submit requests and track their status.

Staff/Wardens: The administrators who handle room allocations and manage maintenance work orders. They need efficient tools to process requests and update statuses.

2.2 System Role

The system itself handles automated processes like displaying status updates and managing the flow of information between students and staff.

---

3. Detailed Use Cases

UC-1: Student Requests a Room

Who: Student

When: Student is logged into the system and wants to request a hostel room

What Happens:

1. Student goes to the "Request Room" page from their dashboard
2. Student picks which hostel they prefer
3. Student chooses the type of room they want (Single, Double, Triple, or Quad)
4. Student selects which semester they need the room for
5. Student can add any special requirements or preferences (optional)
6. Student clicks submit
7. System shows a confirmation message
8. Student is taken to the status page where they can see their request is now "Pending"

If Something Goes Wrong:

- If the student forgets to fill in required fields, the system shows an error message
- Student fixes the missing information and submits again

Result: A new room request is created and shows as "Pending" status

---

UC-2: Student Checks Their Room Request Status

Who: Student

When: Student has already submitted a room request and wants to see what's happening

What Happens:

1. Student goes to the "My Status" page
2. System shows the current status of their request
3. Student can see all the important details:

- Which hostel they requested
- What type of room they asked for
- Which semester
- Current status (Pending, Allocated, or Rejected)

- When it was allocated (if it has been)
- Their room number (if allocated)

If No Request Exists:
- System shows a message saying "No active requests"

Result: Student knows exactly where their request stands

UC-3: Staff Allocates a Room to a Student

Who: Staff/Warden
When: Staff is logged in and there are pending room requests waiting to be processed

What Happens:
1. Staff goes to the "Allocate Room" page
2. System shows a list of all students waiting for room allocations
3. Staff looks through the requests and reviews the details:
- Student ID
- Which hostel they want
- Room type preference
- Semester
4. Staff clicks the "Allocate" button for a student
5. System shows a success message
6. The request status changes to "Allocated"
7. Student gets notified (this is simulated in our prototype)

If No Rooms Available:
- Staff can reject the request instead
- System updates status to "Rejected" and notifies the student

Result: Student gets their room allocated and the status is updated in the system

---

UC-4: Student Reports a Maintenance Issue

Who: Student
When: Student is logged in and needs to report something that needs fixing

What Happens:

1. Student goes to the "Maintenance" page

2. Student picks what type of problem it is:

- Plumbing (leaky taps, blocked drains, etc.)

- Electrical (faulty switches, power issues, etc.)

- HVAC (air conditioning, heating problems)

- Furniture (broken chairs, desks, etc.)

- Other (anything else)

3. Student writes a description of what's wrong

4. Student chooses how urgent it is (Low, Medium, High, or Urgent)

5. Student can specify which room number if needed (optional)

6. Student submits the request

7. System shows a confirmation message

8. A new maintenance ticket is created with "Open" status

If Information is Missing:

- System shows an error message

- Student fills in the missing details and submits again

Result: A maintenance request is created and staff can see it in their work orders

---

UC-5: Staff Views All Maintenance Requests

Who: Staff/Warden

When: Staff is logged in and wants to see what maintenance issues need attention

What Happens:

1. Staff goes to the "Work Orders" page

2. System shows a list of all maintenance requests

3. For each request, staff can see:

- Request ID (for tracking)

- Which student reported it

- What category the problem is

- Description of the issue

- How urgent it is

- Current status (Open, In Progress, Completed)
- When it was submitted

Result: Staff has a complete overview of all maintenance issues that need to be handled

---

UC-6: Staff Updates Maintenance Request Status

Who: Staff/Warden

When: Staff is looking at work orders and needs to update the progress

What Happens:

1. Staff selects a maintenance request from the list
2. Staff changes the status as work progresses:
- Open → In Progress (when they start working on it)
- In Progress → Completed (when it's fixed)
3. Staff can add notes about what was done (optional)
4. Staff saves the changes
5. System updates the work order status
6. Student gets notified about the status change (simulated in prototype)

Additional Options:

- Staff can assign the request to a specific technician
- Staff can add internal notes that students won't see

Result: The work order status is updated and everyone knows the current progress

---

UC-7: Student Views Their Dashboard

Who: Student

When: Student logs in and wants to see an overview of their account

What Happens:

1. Student goes to their dashboard (usually the first page after login)
2. System shows:
- A welcome message
- Quick action buttons (like "Request Room" or "View Status")

- Summary of recent activity

- Current room allocation status (if they have one)

Result: Student gets a quick overview of everything related to their hostel accommodation

---

UC-8: Staff Views Their Dashboard

Who: Staff/Warden

When: Staff logs in and needs to see what needs their attention

What Happens:

1. Staff goes to their dashboard

2. System shows:

- How many room allocation requests are pending

- How many maintenance requests are open

- Quick links to key functions

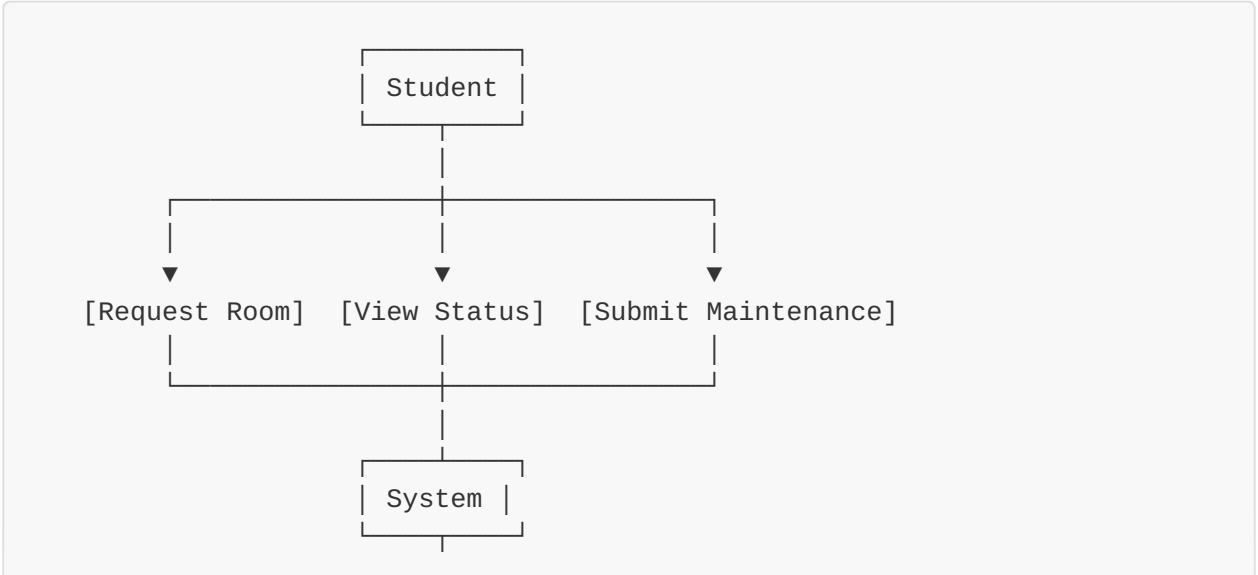
- Summary statistics

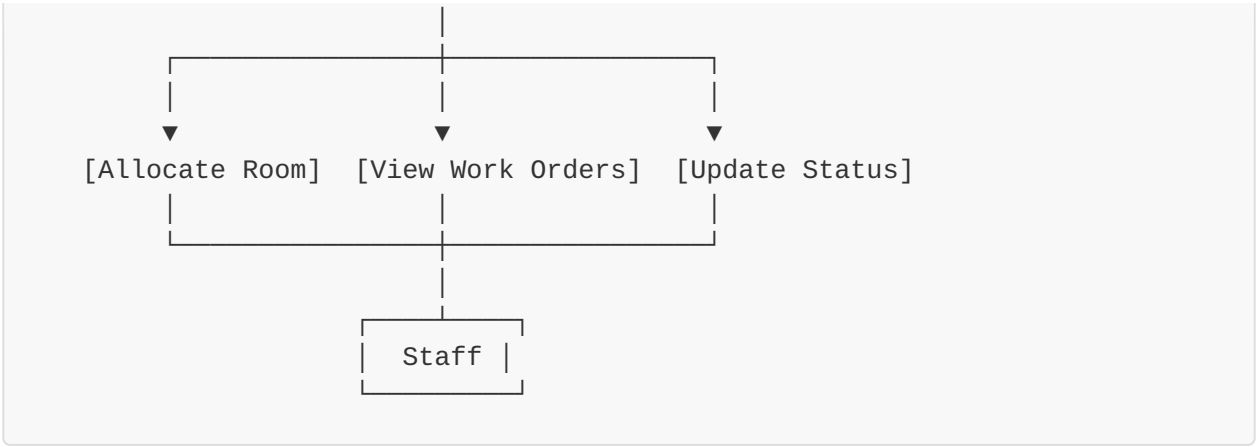
Result: Staff can quickly see what needs to be done and navigate to the right place

---

4. Use Case Diagram

The diagram below shows how students and staff interact with the system:





---

5. Important System Requirements

5.1 Usability

The system needs to be easy to use. Students and staff should be able to figure out how to do things without needing training. Error messages should be clear and tell users exactly what went wrong and how to fix it.

5.2 Performance

Pages should load quickly (under 2 seconds). When users submit forms, they should get immediate feedback. Status updates should appear right away so users always know what's happening.

5.3 Accessibility

The interface should be readable with good color contrast. Users should be able to navigate using just the keyboard. In the future, we plan to add screen reader support for visually impaired users.

---

6. Future Enhancements

While our current prototype focuses on the core functionality, there are several features we'd like to add in the future:

- **Room Swapping**: Allow students to request room swaps with other students
- **Waitlist Management**: Automatically manage waitlists when rooms are full
- **Automated Allocation**: System could automatically assign rooms based on rules and preferences
- **Payment Integration**: Handle accommodation fees and payments
- **Reporting Tools**: Generate reports on allocations and maintenance trends
- **Better Notifications**: Email and SMS notifications for status changes
- **Multi-semester Planning**: Plan room allocations for multiple semesters ahead

---

Document Version: 1.0
Last Updated: December 10, 2025
Course: Software Engineering
Date: December 10, 2025

Team Members:

- Fabilous Lashidi (Project Manager)
- Zvinaishe Marume (System Architect)
- Promise Siafwiyo (UI Designer)
- Tapiwa Chigome (UX Designer)
- Keith Mutabvuri (Researcher)
- Carlton Kampota (Frontend Developer)
- Tivonge Kambarani (Documentation Specialist)
- Tinotenda Gozi (System Analyst)
- Gufe Makomborero (Quality Assurance)

Author: Software Engineering Team