GNDEC Hostels Dashboard

MINOR PROJECT REPORT

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

BACHELOR OF TECHNOLOGY
(Computer Science and Engineering)

Submitted By:

Priyanka (2203534)
Priyanka Lochan (2203535)
Harjot Kaur (2302718)

Submitted To:

Er. Manjot Kaur Gill

Department of Computer Science and Engineering
Guru Nanak Dev Engineering College
Ludhiana ,141006

ABSTRACT

The GNDEC Hostels Dashboard is a sophisticated web-based system designed to computerize and streamline the hostel management system of Guru Nanak Dev Engineering College (GNDEC). As the number of students living in hostels has grown and manual room allotment and complaint resolution have become more complicated, this system is designed to offer an effective, transparent, and easy-to-use digital solution for both administrators and students.

The most notable feature of the dashboard is its intelligent room allocation system. When multiple students request the same room, the system compares their CGPAs and assigns the room to the student with the higher CGPA. Once allotted, the room is marked as "occupied", preventing further selection and ensuring a fair, merit-based distribution.

With a single click, admins can see detailed information of an allotted room's occupant, including name, URN, branch, parent's name, contact number, email, and address, allowing for quick verification and communication.

Another key feature is the complaint management module. Students can submit maintenance or service complaints online, which are stored in the backend database. Admins can view, sort, and respond to these complaints via a dedicated dashboard, ensuring timely resolution and maintaining hostel quality standards.

ACKNOWLEDGEMENT

We are highly grateful to the Dr. Sehijpal Singh, Principal, Guru Nanak Dev Engineering College (GNDEC), Ludhiana, for providing this opportunity to carry out the minor project work at Guru Nank Dev Engineering College (GNDEC).

The constant guidance and encouragement received from Dr. Kiran Jyoti H.O.D. CSE Department, GNDEC Ludhiana has been of great help in carrying out the project work and is acknowledged with reverential thanks.

We would like to express a deep sense of gratitude and thanks profusely to Er. Manjot Kaur Gill , without her wise counsel and able guidance, it would have been impossible to complete the project in this manner.

We express gratitude to other faculty members of computer science and engineering department of GNDEC for their intellectual support throughout the course of this work.

Finally, we are indebted to all whosoever have contributed in this report work.

Priyanka
Priyanka Lochan
Harjot Kaur

List of Figures

Table of Contents

Contents **page no**

Chapter 1 Introduction

1.1 Introduction to Project

Hostel management is a significant function of any educational institution, particularly in resident campuses like Guru Nanak Dev Engineering College (GNDEC), where many students stay in college hostels. Historically, the handling of hostel functions such as room allotment, record maintenance of the students, and complaint management was done manually in a time-consuming manner and generally resulted in inefficiencies, inaccuracies, and delays.

In order to overcome these challenges and ensure a more efficient, transparent, and accessible system, we have created the GNDEC Hostels Dashboard — an online application that seeks to automate and simplify the tasks related to hostels for both the students and the administrators. The system implements a merit-based room allotment system to ensure that students with better academic performance are given preference while choosing rooms. The system also offers real-time room status updates and instant access to the student data for hostel authorities.

Another essential element of the project is the complaint management module, which allows students to file service or maintenance-related complaints via an online interface. These complaints are recorded, maintained, and addressed by administrators via a centralized dashboard, optimizing response times and the overall hostel experience.

The system is constructed with HTML, CSS, Bootstrap, and JavaScript for the frontend, making it a responsive and interactive user interface. The backend is Express.js, with MySQL as the database to hold student records, room information, and complaint logs.

1.2 Project Category

The GNDEC Hostels Dashboard is classified under Application Software Development under the sub-group Web-Based Information Systems. The project also belongs to the application areas of hostel management.

This category is concerned with developing and deploying software solutions that facilitate administrative, operational, and decision-making activities in organizations. The project seeks to automate major hostel-related activities like room allocation, student information management, and complaint handling, substituting conventional manual systems with a centralized digital platform.

The project uses a client-server design, supporting real-time interaction between users (administrators and students) and the system. It uses up-to-date web development frameworks to support responsive user interaction, as well as secure and effective data storage and retrieval systems through a MySQL database.

The main topics covered under this category are:

a) Web Technologies (HTML, CSS, JavaScript, Bootstrap) for developing a user-friendly and responsive interface.

b) Server-Side Development using Express.js for application logic and API routing.

c) Database Management with MySQL for storing and retrieving student records, room allocation information, and complaints.

d) Automation and System Integration, especially in room assignment according to CGPA and real-time complaint management.

1.3 Objectives

a) To automate the tasks of room allocation and complaint handling.

b) To notify students about their approval/non-approval of complain

c) To display information regarding various events.

1.4 Problem Formulation

Hostel management in academic institutions like Guru Nanak Dev Engineering College (GNDEC) involves several routine yet critical tasks such as room allocation, complaint handling, student record management, and communication of important information. Traditionally, these processes are handled manually, which often results in inefficiencies, data errors, delays in response, lack of transparency, and difficulty in managing and accessing student information.

Manual room allocation can lead to conflicts when multiple students apply for the same room, especially when there is no clear, merit-based decision mechanism. Furthermore, the absence of a digital complaint management system causes delays in resolving maintenance or service-related issues, directly impacting the quality of hostel life. Additionally, students often miss out on important hostel-related announcements and event notifications due to the lack of a centralized communication channel.

To address these issues, there is a strong need for an integrated, automated solution that can streamline hostel operations, ensure fair and efficient room allocation, improve complaint resolution speed, and provide timely updates to students.

1.5 Recognition of Need

Currently, hostel management operations in GNDEC are performed manually using a lot of physical paperwork and registers. Separate student information is kept in various departments, which leads to duplication and inconsistency. Room allotments are manually performed according to the students' reported preferences and any special facility requirements. The rooms are classified into three categories: Single, Double, and Corner. In addition, all student-related information of hostel facilities, events, and regulations is preserved in printed booklets, while student and hostel staff records are kept in handwriting registers.

Such a manual system poses some critical drawbacks. In the first place, it is extremely prone to errors and lacks consistency because repetitive paperwork and scope for human errors while entering the data or recording. Secondly, records stored only on paper are very susceptible to physical destruction, theft, or loss because of natural disasters like fire or water accidents. Thirdly, the present system is not efficient and time-saving, frequently resulting in delay and inconvenience for students and administrative personnel alike. The manual process also makes it hard to update, delete, or retrieve particular records fast and accurately. As the population of hostel dwellers keeps growing, keeping precise and easily accessible records for each student has become more challenging, further complicating hostel management procedures.

In light of these challenges, it is clear that the current system is no longer viable or efficient. There is an obvious need for a computerized Hostel Management System that computerizes data handling, centralizes data, and enhances the overall efficiency and reliability of hostel administration. A system like this would ensure ready access to correct information, make room allocation and retrieving information simpler, and consequently benefit the students as well as the hostel staff by minimizing effort and maximizing service quality.

1.6 Existing System

In the current hostel management system at Guru Nanak Dev Engineering College (GNDEC), most administrative tasks are carried out manually or using basic digital tools such as spreadsheets and paper-based forms. Room allocation is typically done through in-person applications or offline submissions, which can lead to confusion, delays, and conflicts, especially when multiple students request the same room. The absence of an automated, merit-based allocation system makes it difficult to ensure fairness and transparency.

Similarly, the complaint handling process is often inefficient. Students usually submit complaints either verbally or by writing them in complaint registers maintained by the hostel authorities. This outdated process makes it hard to track the status of complaints, leads to delays in action, and lacks a proper feedback mechanism to inform students about resolution progress.

Additionally, important announcements and event notifications are generally communicated through physical notice boards or word of mouth, which is unreliable and often results in students missing out on key information.

Overall, the existing system is time-consuming, prone to errors, lacks real-time communication, and fails to meet the expectations of today's tech-savvy student population. These limitations highlight the urgent need for an integrated, automated, and user-friendly solution like the GNDEC Hostels Dashboard.

1.7 Proposed System

The GNDEC Hostels Dashboard is a web-based solution designed to address the limitations of the existing manual hostel management process. The proposed system aims to automate key functions such as room allocation, complaint handling, and event notifications, providing a centralized platform that is accessible to both students and hostel administrators.

In the proposed system, room allocation is handled intelligently through a merit-based algorithm that compares the CGPAs of students when multiple requests are made for the same room. The room is automatically assigned to the student with the higher CGPA, and once allocated, it is marked as "occupied," preventing duplicate allotments. This ensures a fair, efficient, and transparent allocation process.

The system also includes a complaint management module, where students can lodge maintenance or service-related complaints online. These complaints are stored in a secure backend database and presented to administrators through an organized dashboard interface. Admins can view, filter, and respond to complaints, and the system will notify students about the approval or non-approval status of their complaint submissions.

Another important feature is the event and announcement section, where hostel authorities can post updates about upcoming events, maintenance work, or other hostel-related notices. This ensures all students receive real-time information without relying on physical notice boards.

1.8 Unique features of the proposed system

The GNDEC Hostels Dashboard introduces several unique features that differentiate it from the existing manual system and enhance its functionality. These features are designed to improve the efficiency, transparency, and user experience of hostel management at Guru Nanak Dev Engineering College (GNDEC):

a) **Merit-Based Room Allocation System**

   A standout feature of the system is the automated merit-based room allocation. When multiple students request the same room, the system automatically compares their CGPAs and allocates the room to the student with the higher CGPA. This ensures a fair, transparent, and merit-based distribution of rooms, eliminating biases and conflicts in room allotment.

b) **Complaint Management System with Notifications**

   The complaint management module is another key feature. Students can easily lodge maintenance or service-related complaints directly through the platform. These complaints are tracked in the system, and students are notified about the approval or non-approval of their complaints. This ensures faster resolution and keeps students informed about the status of their requests.

c) **Event and Announcement Notification System**

   Unlike the traditional method of posting notices on physical boards, the system includes an automated notification system for hostel-related announcements and events. This feature ensures that all students receive real-time updates about important events, maintenance schedules, or other crucial information, keeping them informed at all times.

d) **Centralized Digital Platform for All Operations**

   The entire hostel management process, including room allocation, student information, complaint handling, and event notifications, is integrated into a single digital platform. This reduces the need for multiple manual systems, provides a centralized record, and improves the efficiency of hostel administration.

Chapter 2 Requirement Analysis and System Specification

2.1 Feasibility Study

Feasibility is the measure of how beneficial / practical an information system will be to an organization. A feasibility study looks at the viability of an idea with an emphasis on identifying potential problems and attempts to answer one main question: Will the idea work and should you proceed with it. Feasibility studies are often conducted to assess the potential risks, benefits, and challenges associated with implementing the project. Ultimately, a project is considered feasible if it can be implemented successfully while meeting its objectives within the available resources and constraints. The GNDEC Hostels Dashboard project is designed to computerize and automate the hostel facility management of Guru Nanak Dev Engineering College. The system is meant to offer features like room allocation, complaint registration and tracking, and notice posting, with accessibility and ease of use for students, staff, and wardens. This report covers the project feasibility, in-depth software requirements, and the SDLC model of development implemented.

The feasibility study will assist in deciding whether the proposed system is feasible from different perspectives — technical, economic, and operational.

2.1.1 Technical Feasibility

The GNDEC Hostels Dashboard is technologically feasible based on the tools and technologies used. It will be built on open-source and standard technologies like HTML, CSS, JavaScript, and React.js for the frontend. The backend will be handled by Node.js with Express.js or Python with Django, and data will be stored in databases such as MySQL or MongoDB. These technologies are popular, well-supported, and compatible with the current infrastructure of GNDEC. Above all, the necessary software tools like IDEs (e.g., VS Code), version control tools (e.g., GitHub), and testing libraries are free and simple to use, and this makes technical implementation of the project realistic and feasible.

2.2.2 Economical Feasibility

Economically, the project is low-cost and very feasible. Because the whole development is open-source software, there are no license costs. Student programmers can conduct the development as a course project, minimizing or even doing away with the employment of external developers. Hosting can either be done on institutional servers or using low-cost cloud platforms. Furthermore, the system will minimize paperwork, manual monitoring, and administrative work in the long term, thereby warranting the minute initial investment through savings in the future.

2.2.3 Operational Feasibility

Operationally, the project will enhance the current workflow of hostel management by a great deal. The dashboard will be simple and easy to use, so students can easily interact with the system for functions like checking the status of the room or making complaints. Similarly, wardens and hostel staff can view student information, change complaint statuses, and post notices without cumbersome steps. With adequate user training and documentation, the system's adoption should be seamless. In general, the system will promote efficiency, transparency, and ease of use for all parties.

2.2 Software Requirement Specification (SRS)

The software requirement specification establishes the total behavior and limitations of the system to be developed.

2.2.1 Data Requirements

The system will handle various types of data. For students, this would involve information like name, roll number, department, year, hostel block, room number, and contact details. For staff personnel such as wardens, the system will retain names, designation, and contact details. The complaint module will record each complaint by a unique ID, issue type, student ID, date of submission, and status of resolution. The room inventory will also have information on room numbers, occupancy status, and available facilities.

2.2.2 Functional Requirements

The dashboard will provide a number of important functionalities. User authentication is basic, where students, wardens, and administrators get role-based access. Students can see their profiles, check room allocation, and report complaints. Admins can allocate rooms manually or automatically. Wardens can view and update complaints and post announcements. All users will get customized dashboards displaying relevant data. The system will also have a notification system to inform users of updates, like complaint resolutions or new notices.

2.2.3 Performance Requirements

Performance is also very important to user satisfaction. The dashboard should react to user requests within 2 to 3 seconds in normal loads. It should be able to serve at least 100 users at the same time without any noticeable performance degradation. Additionally, the system will include automatic daily backups to guarantee that all critical information is backed up in case of system failure or corrupted data.

2.2.4 Dependability Requirements

The system should be trustworthy, with an aim uptime of not less than 99.5%. The application will then be accessible nearly all the time, with maintenance breaks occurring occasionally. It will also ensure data accuracy by checking for valid entries and avoiding duplication. For the sake of recovery in the event of crashes or errors, system logging and error handling that is easy to use will be employed.

2.2.5 Maintainability Requirements

The system will be designed with modular architecture so that other developers in the future can maintain or extend it easily. Each part of the system — frontend, backend, and database — will be kept in a clean, maintainable organization. There will be extensive documentation, including code comments and an API guide, so others can learn and work on the project without any confusion. Version control through Git will enable collaboration and tracking of changes.

2.2.6 Look and Feel Requirements

The user interface should be simple, intuitive, and visually in line with GNDEC branding. A contemporary design strategy with tools such TailwindCSS will provide visual uniformity. Icons, color schemes, and typography shall be in line with GNDEC's image. The interface will be responsive, i.e., will function properly on mobile phones, tablets, and desktops to improve the accessibility of the system for everyone.

2.3 SDLC Model: Iterative Waterfall Model

The Iterative Waterfall Model is selected as the development methodology of this project. This model provides development in discrete, sequential stages — i.e., requirement analysis, system design, implementation, testing, deployment, and maintenance — but at the same time, it permits feedback and changes at the close of each iteration. Thus, it is even more flexible compared to the basic waterfall model and appropriate for projects of academic and institutional nature wherein some requirements do change during the course of development.

During the Requirement Analysis, all the goals of a system and user requirements are collected. The System Design translates these needs into an organized structure and data model. During the Implementation, the developers code the actual applications for frontend and backend. The Testing verifies that the system is bug-free and follows the original specifications. After testing is finished, the Deployment stage entails deploying the system in production and opening it up to actual users. Lastly, the Maintenance stage considers ongoing updates, bug fixes, and enhancements. Information obtained while in or after any of these stages might mean going back partially into previous stages, that is why the model is said to be iterative.

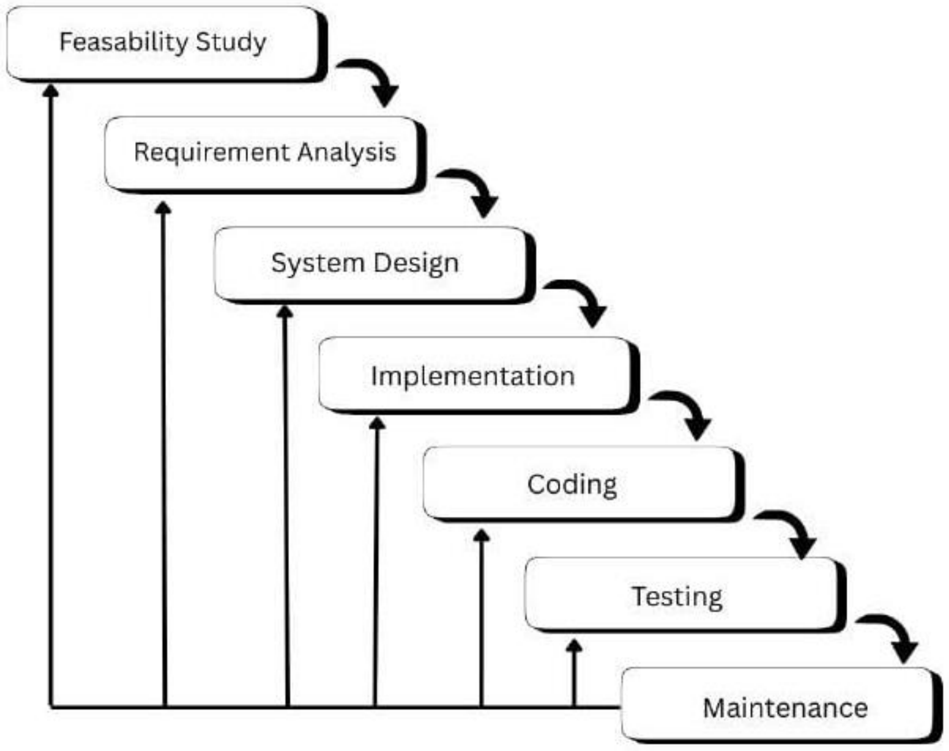


Figure 2.1: Iterative Waterfall Model

2.3.1 Key Features

The Iterative Waterfall Model is an enhanced form of the traditional waterfall model in which development progresses sequentially but with some backward movement between phases. As opposed to the strict structure of the traditional waterfall, this model adds iteration loops, which facilitate refinements and enhancements prior to final deployment.

Its key features are:

a) **Sequential Development:** Every phase logically follows the next one.

b) **Iteration Support:** There is the possibility to return to earlier stages according to feedback or problems found during later stages.

c) **Clear Milestones:** Each stage needs to be done and evaluated before proceeding.

d) **Documentation-Rich:** All phases are well documented, facilitating communication and maintenance.

2.3.2 Phases of the Iterative Waterfall Model

a) **Feasibility Study :** This step verifies if the project is possible and worth executing. It considers technical elements (can we build it?), financial considerations (is it cost-efficient?), and functional requirements (will it function in the real world?). If the project is viable, it proceeds.

b) **Requirement Analysis :** Here, all the necessary features and user expectations are collected. This consists of what the system is to do (functional requirements) and how well it should operate (non-functional requirements). All this information is put down in the Software Requirement Specification (SRS).

c) **System Design :** During this stage, the overall architecture of the system is designed. It involves decisions regarding database structure, user interfaces, system architecture, and technology stack. This design serves as a blueprint to developers while coding.

d) **Implementation :** The developers begin constructing the system based on the design. The system is typically segmented into modules, and each component is coded, tested, and merged. This phase converts design into executable software.

e) **Coding :** Coding is the process of actually writing the source code of the program. The developers make use of chosen programming languages and frameworks to construct the modules. Writing simple unit tests for every component is also part of this step.

f) **Testing :** Once coded, the system is tested to identify and correct bugs. Testing confirms the system performs as intended, is secure, and fulfills all user needs. If issues are discovered, the project can loop back to earlier stages for corrections.

g) **Maintenance :** After the system goes live, it is placed in maintenance. This involves repairing any new problems, implementing new features, and modifying the system as necessary. Maintenance keeps the system operational and beneficial in the long term.

2.3.3 Advantages of the Iterative Waterfall Model

a) **Increased Flexibility:** It can be embedded at mid-stage, enhancing the end product.

b) **Reduced Risk:** Issues get detected early in life because regular reviews and iterative development take place.

c) **Improved Quality Control:** End-to-end checking and validation make software more reliable.

d) **Clean Process:** Suitable for academic or organizational projects with standard milestones and stages.

e) **Appropriate for Medium-Level Complexity Projects:** Such as the GNDEC Hostels Dashboard project, where minimal requirements are identified but may evolve a little.

Chapter 3 System Design

3.1 Design Approach

GNDEC Hostels Dashboard is a centralized web-based application to ease hostel management for Guru Nanak Dev Engineering College. The portal enables communication between hostel wardens and students by offering online access to the critical services like details of rooms, student information, complaint tracking, and notices. Only pre-registered students are allowed to login using credentials offered by the college. Among its main features is a dynamic Notice Board, viewable to every logged-in member, on which administrators can enter key announcements, events, or updates about life in the hostels. It strives to maintain hostel management in an open, efficient, and paperless format.

3.2 Detail Design

The Detailed Design of the GNDEC Hostels Dashboard offers a detailed description of the system architecture, functional modules, and data management plan. This chapter describes the objectives of the project and how they are technically tackled, illustrates data movement by using Data Flow Diagrams (DFDs), and specifies the data relationships by an Entity-Relationship (E-R) model. By dividing the system into logical pieces, this architecture promotes clarity, maintainability, and compliance with the project's functional and non-functional requirements. The aim is to create a scalable, dependable, and user-friendly platform for effective hostel management.

3.2.1 Objectives and Solutions

a) **Objective 1:** To automate the tasks of room allocation and complaint handling.

   Admins are able to allot rooms digitally using the dashboard, where room availability and occupancy are managed automatically. Students are able to raise complaints via an online form, which are saved in the database and allocated with statuses such as "Pending", "In

Progress", or "Resolved". Admins are able to change complaint statuses, which are then updated on the student interface.

b) **Objective 2:** To notify students about their approval/non-approval of complaint.

   Every complaint filed by a student is monitored, and every update provided by the admin—approval, rejection, or closing—is reflected on the dashboard of the student in real time. This reduces the amount of manual communication and enhances response transparency.

c) **Objective 3:** To display information regarding various events.

   An interactive notice board module is also included through which admins are able to publish notices for events, maintenance, deadlines, or general announcements. These are pulled from the backend and are shown dynamically to all students to enable timely dissemination of information.

3.2.2 System Architecture

The dashboard takes a three-tier web structure that promotes modularity and maintainability. Front-end development utilizes React.js or ordinary HTML/CSS and JavaScript to design responsive user interfaces. The backend, developed in Node.js and Express.js, is responsible for business logic, API routes, authentication, and data manipulation. The database layer persists data with a relational database such as MySQL.

- **Presentation Layer :** The frontend interface has a simple and user-friendly experience. When opening the site, users are brought to a login page, and only current students or admins have the option to input valid credentials. After logging in, students see their dashboard, consisting of personal information, room assignment, complaint status, and a Notice Board with recent announcements. Admin users see an enhanced dashboard with switches to control the allocation of rooms, modify student records, answer complaints, and publish notices.

- **Application Layer :** The backend system processes all the core functionalities. Built in Node.js using Express.js, it contains API routes for login authentication, complaint posting, retrieval of student details, and notice management. There is no register route, and all students are required to be pre-added by an admin. Admin-specific routes provide endpoints for posting and updating notices, allocating rooms, and handling student information. Student-specific routes enable them to view personal info, file complaints, and view announcements on the Notice Board.
- **Data Layer :** The database is designed to mirror the central hostel functions. The users table holds usernames, password hashes. The students table adds user data with hostel name, room number, branch, and academic year. The rooms table monitors hostel rooms, where room numbers, occupancy status, and capacity are recorded. Complaints are logged in the complaints table with columns such as title, description, status, submission date, and associated student ID. The notices table has columns such as title, message, date, and posted_by, which drives the Notice Board facility. This schema is capable of secure logins, extensive record-keeping, and a dynamic announcements system.

3.3 Data Flow Diagram (DFDs)

i. Level 0 DFD

The Level 0 Data Flow Diagram depicts a top-level overview of the GNDEC Hostels Dashboard System. It highlights the two main external actors: Student and Admin, who use the system. Students make requests like login, information about rooms, complaint submission, or viewing a notice. Admins handle backend operations like complaint management, room assignment, and publishing notices. The GNDEC Hostel Dashboard System is the CPU that receives requests and sends back relevant responses or data. This diagram is used to picture how data moves between users and the system without going into internal details.

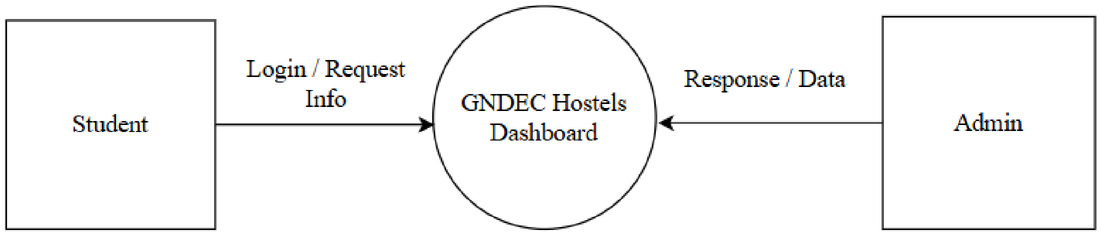


Figure 3.1 : Level 1 DFD

a) Students : Students are the primary users of the dashboard. They interact with the system for various purposes such as:

- Logging in to their hostel account
- Viewing hostel room details
- Submitting and tracking complaints
- Viewing notices and announcements

b) GNDEC Hostels Dashboard : This is the central processing unit of the application. It receives requests from both students and admins, processes them, and communicates with the backend services and database.

c) Admins : Admins are responsible for managing the internal operations of the hostels. They use the dashboard system to:

- Approve/reject complaints
- Allocate or modify room assignments

- Post new notices
- Monitor hostel activity and data

ii. Level 1 DFD

This Level 1 DFD divides the system into sensible modules. External users (Admins and Students) interact with internal processes (such as notice viewing, complaint management, and authentication). These processes exchange data with corresponding data stores to retrieve or save information. The design ensures smooth data flow from source to processing and storage, keeping the modular and secure hostel management system structure intact.

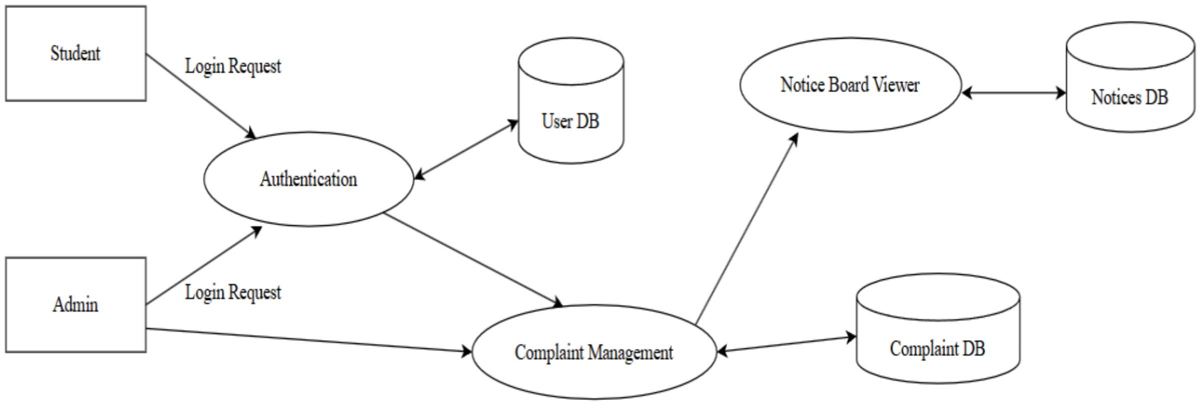


Figure 3.2 : Level 1 DFD

a) Student

- The Student is an external entity that interacts with the system by logging in and submitting complaints.
- They request information related to room allocation, complaints, and notices.
- Their actions drive the core operations of the system, including logging in to access relevant data.

b) Admin

- The Admin is also an external entity who logs into the system to manage hostel operations.
- They are responsible for reviewing complaints, posting notices, and managing room allocations.
- Admins act as moderators, ensuring the smooth functioning of hostel management tasks.

c) Authentication

- The Authentication process ensures that only authorized students and admins can access the system.
- It handles the login process, verifying credentials against the User DB to ensure security.
- It is the primary entry point to the system for all users, ensuring proper access control.

d) User DB

- The User DB stores the details of all registered users (students and admins), including login credentials and personal information.
- This data store is crucial for the authentication process, enabling secure login and user validation.
- It serves as a centralized repository of user data that can be accessed whenever authentication is required.

e) Complaint Management

- The Complaint Management process handles all user complaints related to hostel facilities.
- Students can submit complaints, and admins can review, approve, reject, or resolve them.
- This process acts as the backbone of the grievance redressal system in the hostel.

f) Complaint Database

- The Complaint Database stores all the complaints raised by students, including their status, resolution, and any updates from admins.
- It is used to track the history of complaints and their resolutions, allowing admins to monitor progress and manage complaints efficiently.
- This data store provides transparency and accountability for the complaint management process.

g) Notice Board Viewer

- The Notice Board Viewer process allows students to view the notices posted by admins on the hostel notice board.
- It retrieves and displays important announcements, event information, and alerts to the students.
- This process ensures that students are informed about the activities, updates, and events in the hostel.

h) Notices Database

- The Notices Database stores all the notices posted by the admin, including event details, maintenance schedules, and general announcements.
- It provides a persistent storage of all notices for easy access and display to students.
- This database is updated as new notices are posted by the admin.

3.5 User Interface Design

Login Interface :

The login page of the GNDEC Hostels Dashboard is the secure entry point to the system. It has a full-screen background image of Guru Nanak Dev Engineering College, Ludhiana, which immediately establishes a familiar and welcoming tone for the users. The middle of the page is dominated by a semi-transparent dark login panel with the caption "Hostel Dashboard: Login Portal". The login panel contains input fields for "Password" and "Username", as well as a blue "Login" button. The look is simple but professional to allow users to locate the login form easily. The page is only open to authorized users, assisting in maintaining the security of the system by excluding unauthorized entries.

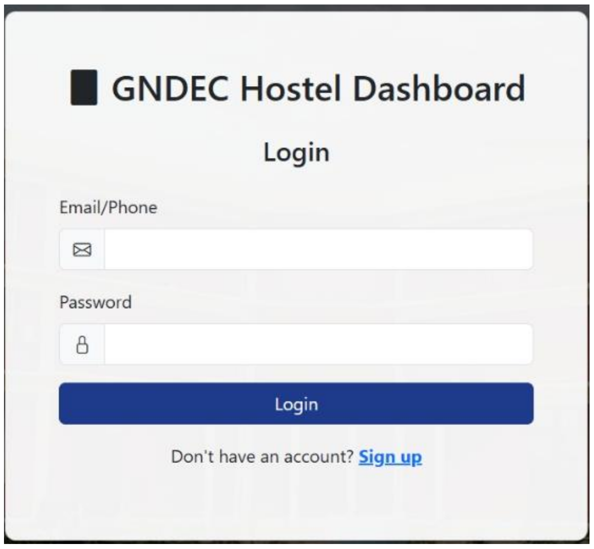


Figure 3.3 : Login Interface

Room Allocation Interface :

GNDEC Hostels Dashboard's room allocation page aims to make assigning hostel rooms to students simpler and digital. This page is smartly sectioned into two parts. On the left is an exhaustive form called "Allocate Room by CGPA" which gathers student information like Name, CGPA, University Roll Number, Branch, Course, Choice of Room Number, and contact details like parents' names and phone number. This form facilitates room assignment on the basis of academic performance for fairness and transparency. At the right end of the page is a search functionality where administrative users can input a room number and retrieve related room information immediately. The design is user-friendly and easy to use, enabling hostel staff to perform their functions more effectively while providing students with a seamless room allocation experience.

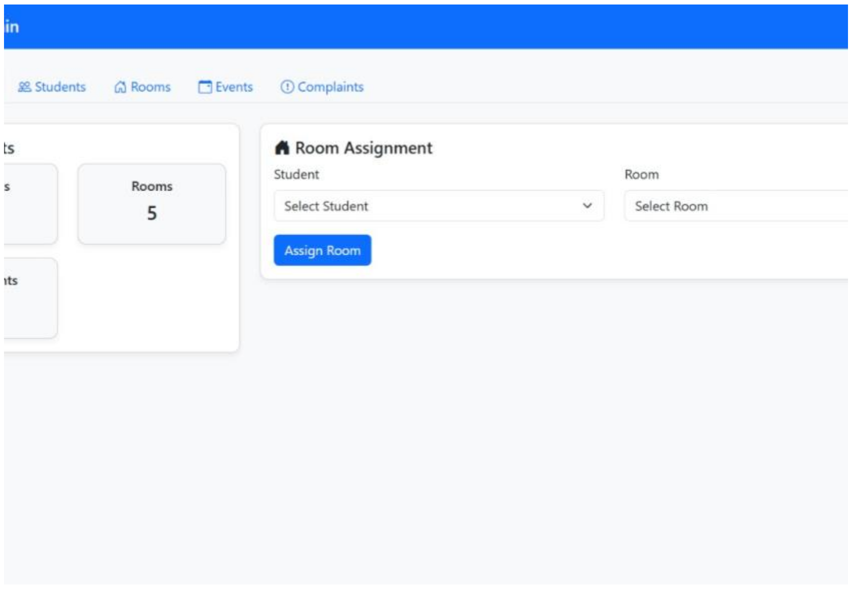
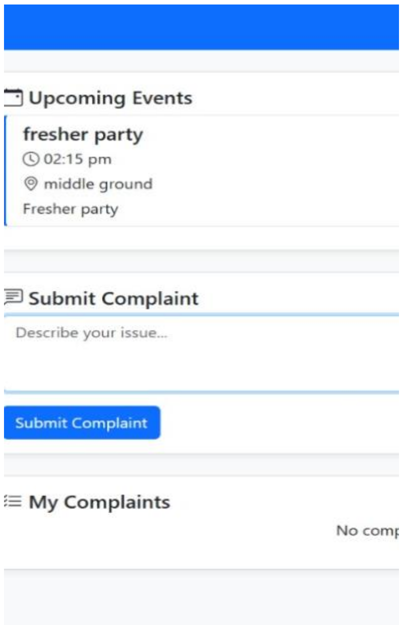


Figure 3.4 : Room Allocation Interface

Complaint Management Page :

The page for the student complaint form is a key feature that allows the residents to forward their complaints in-person directly from the dashboard. The setting of the interface has a peaceful image of a college campus in the background, giving the atmosphere a welcoming and student-centric feel. The center of the page has the form that requests vital information such as Student Name, URN

(University Roll Number), contact details, Program, Year, Branch, and the complaint. There are dropdowns to choose between the nature of the complaint, i.e., electricity or neatness, and the individual against whom the complaint should be issued, e.g., the warden or the caretaker. This systematic composition makes it easier to classify grievances and ensures their forwarding to correct personnel. Accountability, transparency, and prompt rectification of complaints by students who reside in hostels are maintained.


Upcoming Events

fresher party

02:15 pm

middle ground

Fresher party

Submit Complaint

Describe your issue...

My Complaints

No comp

Figure 3.5 : Complaint Form Interface

3.6 Methodology

Developing the GNDEC Hostels Dashboard system requires a structured approach to ensure smooth functionality, efficient data management, and alignment with user needs. Here is a step-by-step methodology for developing the GNDEC Hostels Dashboard:

3.6.1 Requirements Gathering

a) Stakeholder Meetings: Interacted with hostel wardens, students, and admin staff to identify key needs like complaint tracking, notice management, and room info access.

b) Feature Identification: Outlined core functionalities — student login, complaint submission, notice viewing, admin response, and backend management.

3.6.2 System Design

a) Architecture Planning: Designed a three-tier architecture with frontend (dashboard interface), backend (logic & API handling), and database (MySQL).

b) Database Schema: Created entity-relationship diagrams to model users, complaints, notices, and rooms.

c) UI Mockups: Designed simple, intuitive wireframes for student and admin views for usability and clarity.

3.6.3 Development

a) Frontend: Developed using HTML, CSS, JavaScript for responsive and interactive interface.

b) Backend: Implemented using PHP for server-side scripting and logic handling.

c) Database Integration: MySQL used to manage persistent data for users, complaints, and notices.

3.6.4 Testing

a) Functional Testing: Verified each feature (login, complaint submission, notice display) for correctness.

b) User Testing: Conducted sessions with real users (students/admins) for feedback on usability and performance.

c) Bug Fixing: Resolved issues related to form validation, session management, and database operations.

3.6.5 Deployment

a) Hosting: Deployed the dashboard on a local or institutional web server.

b) Access Controls: Configured user roles and secure login mechanisms.

c) Final Review: Validated the full system in real-world-like conditions before go-live.

Chapter 4 Implementation and Testing

The Implementation and Testing phase is an important phase in the development cycle of the GNDEC Hostels Dashboard project. This phase consists of bringing the system design to life in the form of a functional solution through actual coding, integration, and testing of the software components. Implementation phase emphasized creating a secure, high-performance, and easy-to-use web application supporting the requirements of students, wardens, and administrators for operating hostels of Guru Nanak Dev Engineering College (GNDEC). This meant creating both client-side (frontend) and server-side (backend) functionality and integrating a formal database.

4.1 Introduction to Languages, IDE's, Tools and Technologies Used for Project Work

The development of the GNDEC Hostels Dashboard involved the use of various programming languages, frameworks, IDEs, and tools to build a fully functional, responsive, and database-driven web application. The technologies were chosen to ensure performance, scalability, and maintainability of the system.

4.1.1 Languages

- **HTML5** (HyperText Markup Language version 5) is the core language used to structure and organize content on the web. In the GNDEC Hostels Dashboard project, HTML5 was used to define the layout and skeleton of all web pages, including forms, buttons, tables, headers, footers, and other structural elements. It allowed us to create semantic and accessible pages that are easily interpreted by web browsers. Features such as the `<section>`, `<article>`, `<nav>`, and `<footer>` tags enhanced the readability and maintainability of the code. Additionally, HTML5's built-in form validation attributes and support for multimedia elements contributed to a more efficient and modern user interface.

- **CSS3** (Cascading Style Sheets level 3) was used to design and style the visual elements of the web application. It helped define the look and feel of the dashboard, including colors, fonts, spacing, layout, and responsiveness. CSS3 features like Flexbox and Grid were particularly useful in creating a responsive layout that adapts to different screen sizes, such as desktops, tablets, and mobile devices. Transitions, animations, and hover effects were also used to enhance the user experience and make the interface more engaging. By separating content (HTML) from presentation (CSS), the project maintained a clean structure and allowed easy updates to the design without altering the content structure.
- **JavaScript** played a crucial role in adding interactivity and dynamic behavior to the GNDEC Hostels Dashboard. It was used to handle form validations, manage dynamic content updates, and implement logic on the client side, such as showing/hiding elements based on user interactions. JavaScript enabled real-time feedback on forms, improving usability by informing users of errors before submission. Additionally, it was used in coordination with React.js to create a modular and reactive frontend, where components could update automatically based on changes in data or user actions. JavaScript's versatility and event-driven architecture made it an essential part of building a responsive and interactive web application.

4.1.2 IDEs

- **Visual Studio Code (VS Code):** Visual Studio Code was the main Integrated Development Environment (IDE) used for the project. Its rich set of features, such as intelligent code completion (IntelliSense), debugging, integrated Git, terminal, and a broad selection of extensions for JavaScript, React, and other technologies of interest, made it a productive and powerful development environment.

4.1.3 Frameworks/Libraries

- **React.js**

  React.js is a powerful JavaScript library developed by Meta for building user interfaces, especially single-page applications. It is component-based, meaning you can break down your UI into reusable pieces, making development more modular and maintainable. React's virtual DOM optimizes rendering performance, and its ecosystem includes tools like React Router (for navigation between pages) and Redux (for centralized state management), which are useful in building complex dashboard interfaces.

- **Tailwind CSS**

  Tailwind CSS is a utility-first CSS framework that lets you style components by applying pre-defined classes directly in your HTML or JSX. Instead of writing custom CSS, you use Tailwind's class names (like `bg-blue-500`, `text-center`, `p-4`) to rapidly design responsive and consistent UIs. It's particularly effective in dashboard development because it promotes clean, scalable design and pairs seamlessly with React for building polished interfaces quickly.

- **Express.js (Node.js)**

  Express.js is a minimal and flexible web application framework built on Node.js. It allows you to create RESTful APIs efficiently using JavaScript—the same language as your frontend, which keeps your full stack consistent. Express handles routing, middleware, and HTTP requests, making it ideal for building the server-side logic that powers your dashboard (e.g., fetching hostel data, managing complaints, handling logins, etc.).

- **MySQL**

  MySQL is a widely-used open-source relational database management system. It is ideal for structured data like student profiles, hostel room allocations, attendance logs, and complaint records. MySQL supports SQL (Structured Query Language), which allows for complex queries, joins, and data integrity. It integrates well with Express.js using libraries like `mysql2` or ORMs such as Sequelize, helping you manage data transactions easily and securely.

4.2 Testing Techniques in context of project work

Testing is a crucial process in software development that involves evaluating a system or application to identify defects, errors, or discrepancies between expected and actual behavior. Testing ensures that the software meets quality standards, functions correctly, and satisfies user requirements. For testing, black-box testing is done. Black- box testing means examining the functionality of the system without peering into its internal structures or workings. This method of test can be applied virtually to every level of software testing: unit testing, integration system and acceptance. The black box is a powerful technique to check the system under test from the user's perspective. The testing team does not cover the inside details such as code, server logic, and development method. So, the testing involves the following steps:

a) Unit testing: The individual UI of each screen was separately tested.

b) Integration Testing: Two or more screens were then combined and examined to find out whether they are working functionally correct when combined with each other.

c) System Testing: Different modules together combined and checked to find out if there is any bug in its working. After this, the whole software was tested as a whole.

d) Acceptance Testing: Software was then given to the third party for testing.

4.3 Output Screens

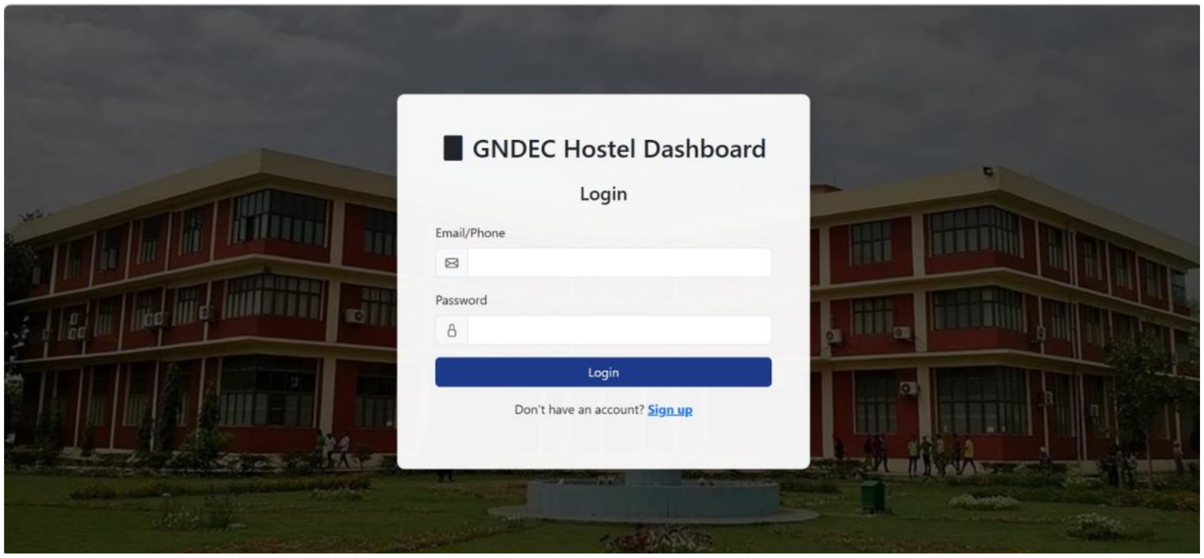


Figure 4.1: Output Screen 1

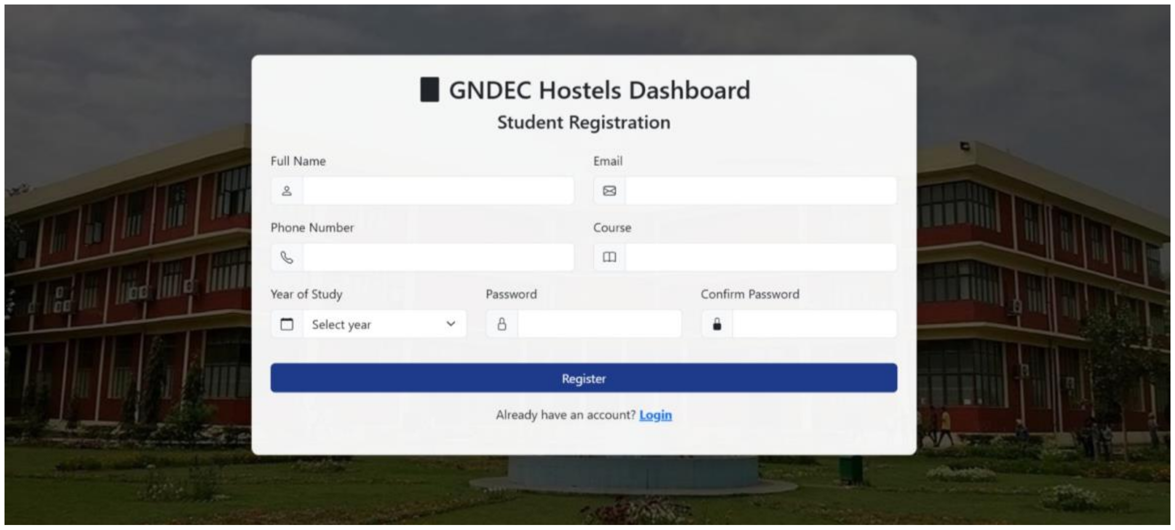
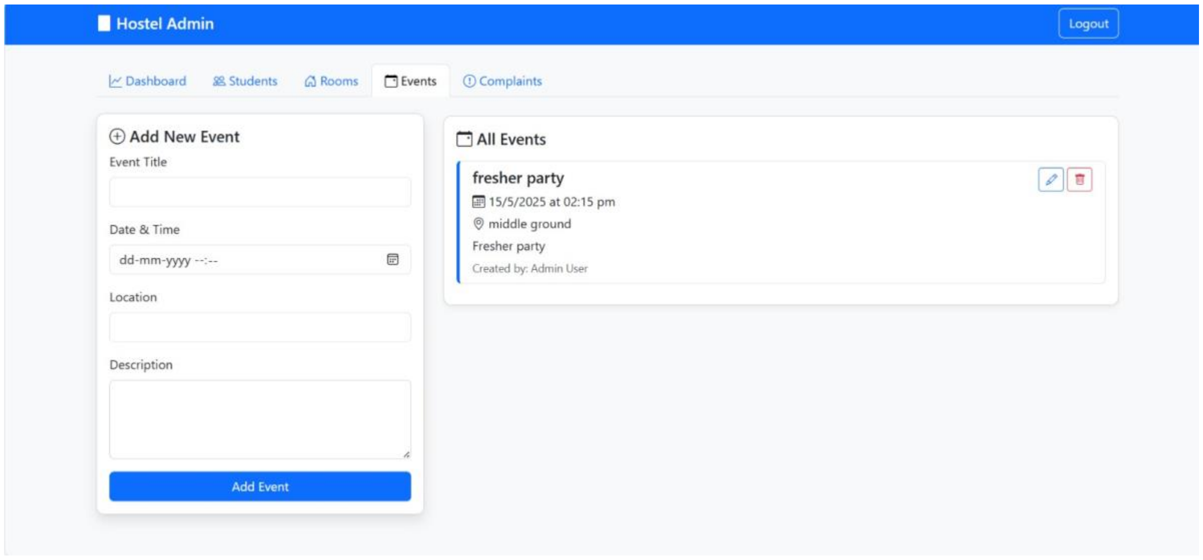
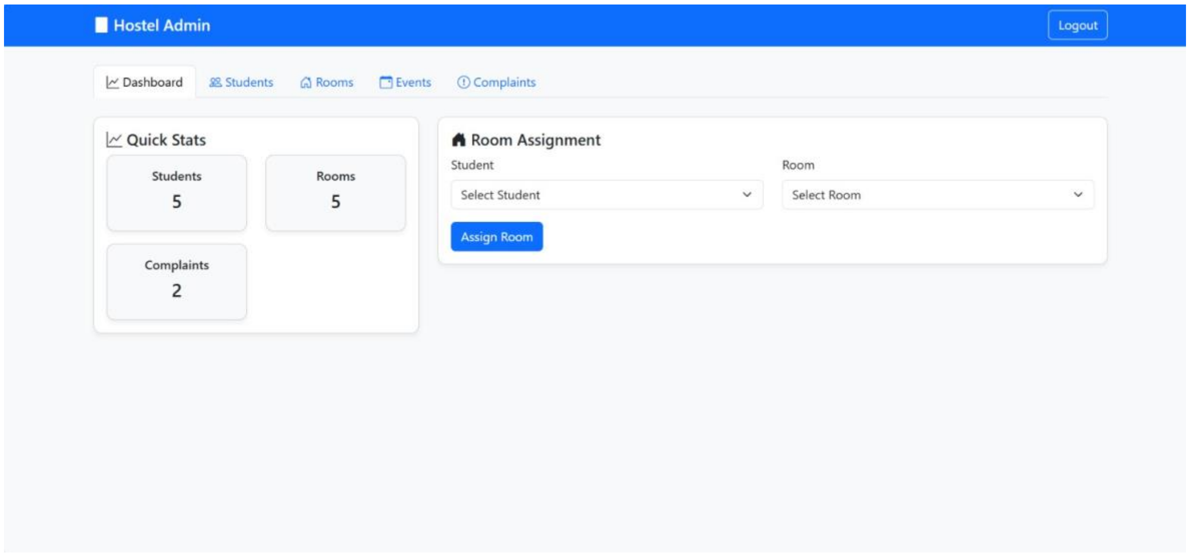


Figure 4.2: Output Screen 2



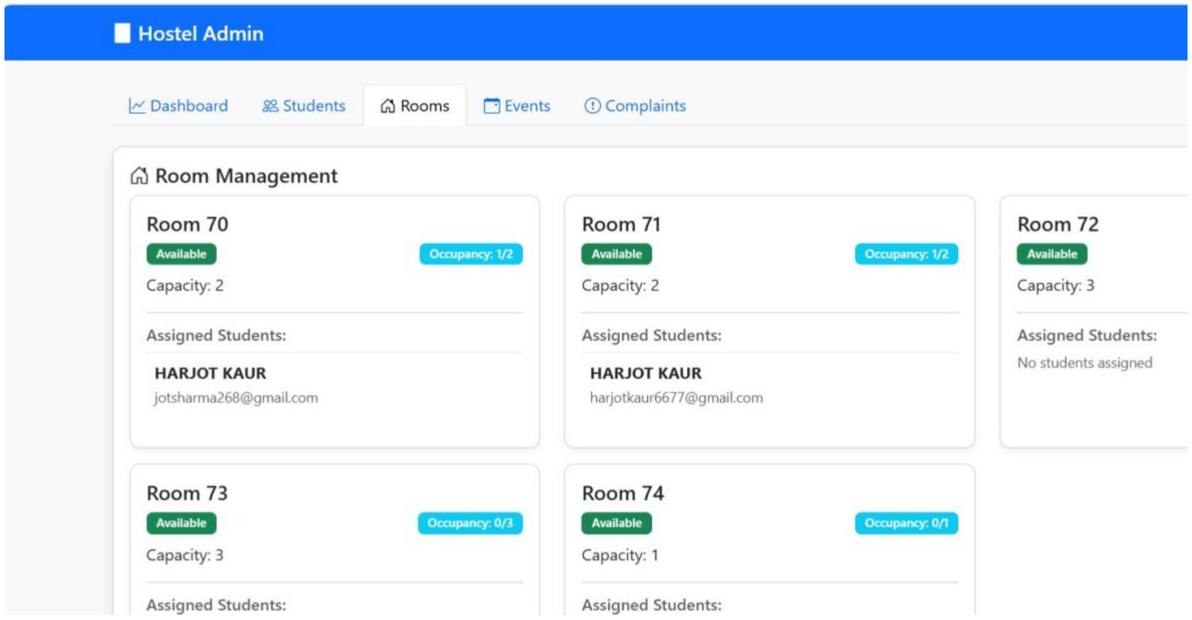
fresher party
15/5/2025 at 02:15 pm
middle ground
Fresher party
Created by: Admin User

Figure 4.3: Output Screen 3



Students 5
Rooms 5
Complaints 2

Figure 4.4: Output Screen 4


Hostel Admin

Dashboard | Students | Rooms | Events | Complaints

Room Management

Room 70
Available
Occupancy: 1/2
Capacity: 2
Assigned Students:
HARJOT KAUR
jotsharma268@gmail.com

Room 71
Available
Occupancy: 1/2
Capacity: 2
Assigned Students:
HARJOT KAUR
harjotkaur6677@gmail.com

Room 72
Available
Capacity: 3
Assigned Students:
No students assigned

Room 73
Available
Occupancy: 0/3
Capacity: 3
Assigned Students:

Room 74
Available
Occupancy: 0/1
Capacity: 1
Assigned Students:

Figure 4.5: Output Screen 5

Chapter 5 Conclusion and Future Scope

The Hostel Dashboard project successfully provides a centralized and user-friendly platform for managing hostel-related activities such as student registration, room allocation, complaints, and announcements. It streamlines the administrative workflow, reduces paperwork, and enhances communication between hostel authorities and residents. With real-time data access and a responsive interface, the dashboard improves transparency, efficiency, and overall hostel management experience.

The Hostel Dashboard is a user-friendly web application designed to simplify hostel management tasks such as student registration, room allocation, complaint handling, and announcements. It features real-time updates, role-based access for students and administrators, and a dashboard with visual insights on occupancy and issue resolution. Built with React.js for the frontend and styled using Tailwind CSS, the application uses TypeScript for improved code quality. The backend is powered by Express.js, with authentication and real-time updates.

The Hostel Dashboard significantly improves the efficiency and transparency of hostel management by digitizing core processes like room allocation and complaint resolution. It reduces manual work, minimizes errors, and ensures faster communication between students and administrators. With real-time updates, secure access, and a centralized platform, it enhances user convenience and operational control. Overall, the system fosters a more organized, responsive, and student-friendly hostel environment.

Future Scope

The current implementation serves as a solid foundation for further innovation and scalability. Potential enhancements include:

- Online Fee Payment Module : Allow students to pay hostel and mess fees directly through the portal using UPI, credit/debit cards, or net banking.
- Mobile App Version : Develop Android/iOS apps for students and wardens for easier access, push notifications, and faster communication.
- Complaint and Maintenance Tracking System : Add a ticketing system where students can raise issues (e.g., plumbing, electricity) and track resolution status.
- Visitor Management System : Enable students to pre-register visitors, and wardens to approve or deny visit requests with time-based access logs.

REFERENCES

[1] Abhishek Pundir, Akarsh Singh, Tanvisha Varshney, Tanvi Singh, Ayushi Gupta “Smart Dashboard For Managing The Hostel Activities”, Volume 9, Issue 6 , June 2021 | ISSN: 2320-2882 [Online]. Available : https://ijcrt.org/papers/IJCRT2106484.pdf

[2] Kartik Chaudhri, Riddhi Kevat "Study of Digitalized Hostel Management System", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN :2456-3307, Volume 7 , Issue 2, pp. 366-371, March-April 2021.

[Online]Available:https://www.researchgate.net/publication/351311910_Study_of_Digitalized_Hostel_Management_System

[3] Prof. Shyamsundar Magar , Ms. Sakshi Said , Mr. Rohit Jadhav , Mr. Shashikant Jadhav “Hostel Management System and Aggregation” Volume 8, Issue 10, October 2021| ISSN : 2349-5162[Online]Available:https://www.researchgate.net/publication/356579821_Hostel_Management_System_and_Aggregation

[4] Official website of GNDEC : https://gndec.ac.in/