Dormitory Automation System: A Role-Based Web Application

Sudenaz Güldal
Dept. of Information System
Engineering of Kocaeli University
Kocaeli, Turkey
sudenaz.guldal03@gmail.com
https://github.com/sudenazguldal

Melike Kır
Dept. of Information System
Engineering of Kocaeli University
Kocaeli, Turkey
melike1kir@gmail.com
https://github.com/melike-klr

Merve Gazioğlu
Dept. of Information System
Engineering of Kocaeli University
Kocaeli, Turkey
gazioglumerve4@gmail.com
https://github.com/merwf

Abstract - This paper presents a web-based dormitory automation system designed to manage student entry-exit logs, permission requests, payment records, and room assignments. The system aims to enhance security and streamline dormitory operations through role-based interfaces for security personnel and student affairs staff. Students can be added, marked passive (instead of deleted), and reassigned to different rooms when needed. The backend is powered by a MySQL database designed according to 5NF normalization rules and utilizes advanced features such as indexes, views, and triggers. The project was developed using XAMPP and Visual Studio Code as the primary development environment.

Keyword - (Dormitory automation, entry-exit system, permission tracking, room assignment, payment tracking, MySQL, web application, 5NF, trigger, view, index, XAMPP, Visual Studio Code)

I. Introduction

The need to enhance security in university dormitories, keep track of student leave requests, and manage payment records in an organized manner has been the primary motivation behind the development of this project. Traditional methods such as pen-and-paper for logging entry-exit and leave processes are time-consuming and prone to human error. Therefore, digitizing these processes was set as a key objective.

The developed web-based system offers separate interfaces for security personnel and student affairs staff, facilitating task distribution and workflow. Security personnel manually record students' entry and exit transactions through the system, while student affairs staff review and either approve or reject incoming leave requests. Additionally, the integrated payment module allows student affairs personnel to verify outstanding balances and approve or deny permissions accordingly in a digital environment.

Instead of utilizing biometric authentication systems (such as fingerprint scanning), a simpler solution has been deliberately chosen. This decision stems from the fact that the dormitory where the system is to be implemented is small in scale, housing only 12 students. In such a setting, investing in fingerprint scanners and their maintenance is unnecessary. Therefore, a digital but manually operated system offers a more cost-effective and practical solution.

II. Problem Definition

Existing dormitory entry-exit and permission systems are often centralized, costly, and inflexible. For small dorms, they become overly complex or expensive. Similarly, payment-tracking features typically lack the customization needed for specific dorm requirements.

This project provides a lightweight, web-based solution tailored to smaller dormitories. Core functions such as entry-exit logging, permission approvals, and payment verification are managed through a simple, role-based interface accessible to both security personnel and student affairs staff.

As a result:

- Entry and exit records can be tracked with ease,
- Leave requests can be processed efficiently,
- Payment status can be monitored and recorded digitally

III. Related Research

During the development of this project, several decisions were made regarding the choice of programming language and database design. PHP and MySQL were selected for their open-source nature, ease of use, and suitability for small to medium-sized web applications.

The database was designed with a focus on proper normalization, targeting compliance with the Fifth Normal Form (5NF). Separate tables were created for students, permissions, entry-exit logs, and payments to eliminate data redundancy and maintain consistency.

MySQL triggers were implemented to automate checks such as preventing multiple simultaneous entry or exit records and validating permission dates. Additionally, views were created to support simplified reporting and querying, while indexes were applied to frequently accessed columns to improve performance.

These design decisions were made after reviewing documentation and examples on platforms such as Stack Overflow, W3Schools, and GeeksforGeeks. Similar small-scale automation projects were also examined to gather ideas on user interface design and task distribution.

IV. System Flowchart

The dormitory automation system operates based on role-specific panels for security personnel and student affairs staff. The general flow of the system is outlined below:

1. Login Phase

- User navigates to the login page.
- Login credentials are verified.
- If invalid → error message is shown.
- If valid → user is redirected to the appropriate panel based on their role.

2. Student Registration (Student Affairs Role)

- A new student is added to the system.
- The student is assigned to an appropriate room and bed.
- Upon request, room changes can be processed.

3. Security Panel Operations

- Security personnel selects a student from the list.
- The system checks if the student is active and whether they have a valid permission.
- If on leave → a warning is shown.
- If not → entry or exit is logged.
- Duplicate entry or exit attempts are blocked.

4. Student Affairs Panel Operations

- Permission requests are listed and reviewed.
- Staff evaluates leave dates and payment status.
- Requests are approved or rejected accordingly.
- Payment status is updated and stored.
- Room change requests are processed if applicable.

5. Database Operations

- All actions are recorded in related tables: students, ***entry_exit_logs***, permissions, payments, ***permission_approved_by***, **room_assignments**, beds
- Views are used for simplified reporting.
- Triger ensure validation and consistency.

(See Fig. 1 on the next page.)

V. Software Architecture

The dormitory automation system follows a layered architecture and is developed primarily using **PHP** for backend logic and **MySQL** for data management. The application consists of three main layers:

1. Presentation Layer

- **Technologies:** HTML, CSS (no JavaScript used)
- **Structure:**
  - /public: login and landing pages
  - /assets/css, /assets/img: visual design and logo
  - /includes: shared layout elements (e.g., header.php, footer.php)
- This layer provides the user interface, rendered dynamically through **PHP scripts** based on user roles and permissions.

2. Application Layer (PHP Business Logic)

- **Primary Language:** PHP
- **Directories:**
  - /security: handles entry-exit logic via PHP
  - /students_affair: manages student registration, permission handling, payments, and room assignments
- **Functionality:**
  - Each module uses **PHP** to receive, validate, and process form submissions ($_POST)
  - Role-based access is enforced through auth_check.php
  - Conditional logic, redirects, and user feedback are fully handled in PHP
- This layer encapsulates the **core business rules** of the system.

3. Data Access Layer (MySQL + PDO in PHP)

- **Connection File:**
  - config/db.php creates a secure **PDO-based** MySQL connection in PHP
- **Database Design:**
  - Tables normalized up to **5NF**
  - **Triggers** enforce business logic like date validation and duplicate entry blocking
  - **Views** support role-based reporting
  - **Indexes** boost query performance on critical columns
- All data interactions occur through **PHP Prepared Statements**, ensuring security and maintainability.

This architecture ensures that the **entire logic flow is handled server-side with PHP**, while MySQL ensures reliable data persistence and integrity. The system is modular, maintainable, and tailored for role-based operations.

VI. Entity-Relationship Diagram

(See Fig. 2 on the next page.)

VII. References

[1] Stack Overflow. https://stackoverflow.com/
[2] W3Schools. HTML, CSS, PHP, SQL Tutorials. https://www.w3schools.com/
[3] GeeksforGeeks. Database and SQL Tutorials. https://www.geeksforgeeks.org/
[4] PHP Documentation. https://www.php.net/manual/en/
[5] MySQL Documentation. https://dev.mysql.com/doc/
[6] XAMPP Documentation. https://www.apachefriends.org/index.html
[7] Visual Studio Code. https://code.visualstudio.com/
[8] IEEE Author Templates. https://www.ieee.org/conferences/publishing/templates.html
[9] phpMyAdmin Documentation. https://docs.phpmyadmin.net/

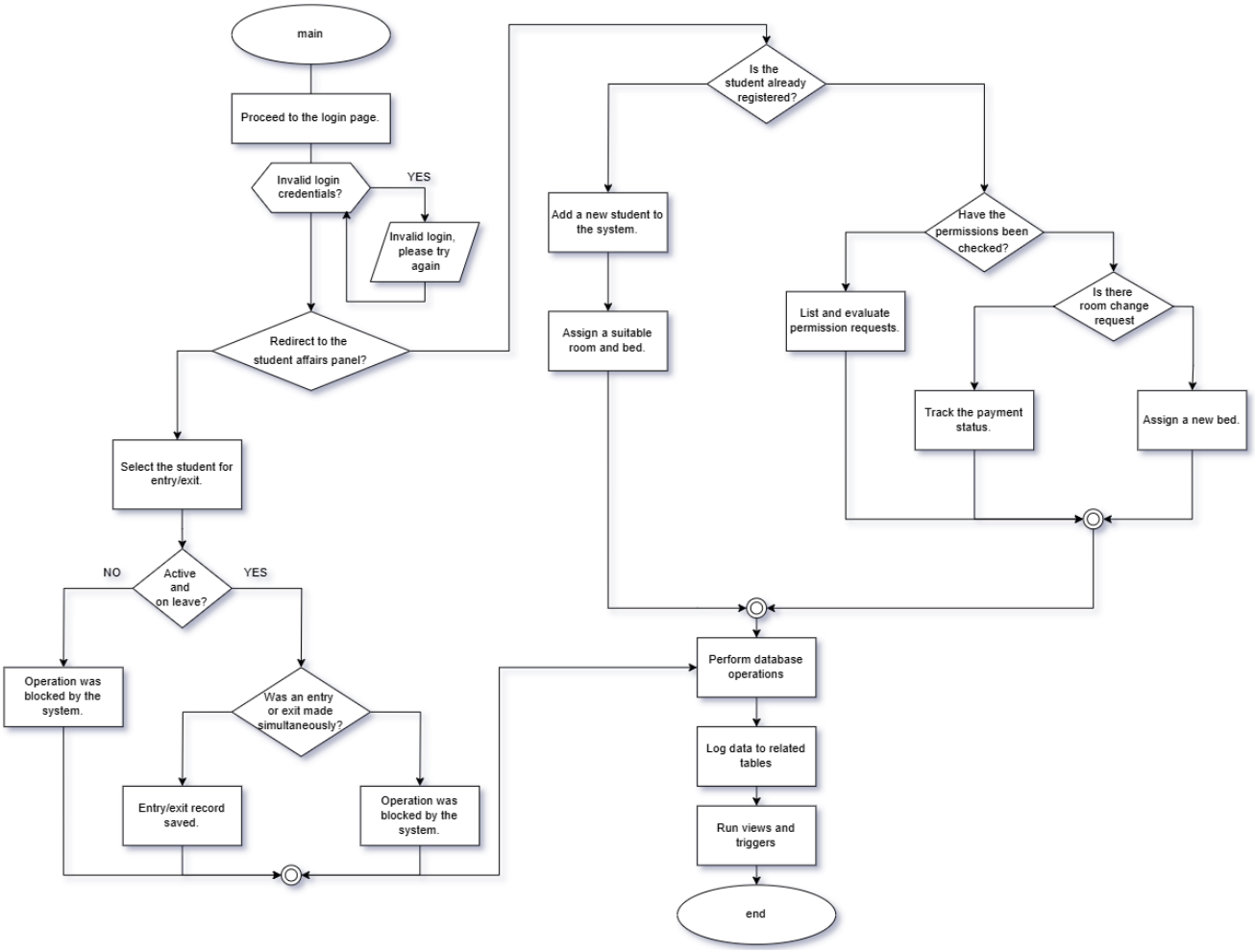

Figure 1. System Flowchart

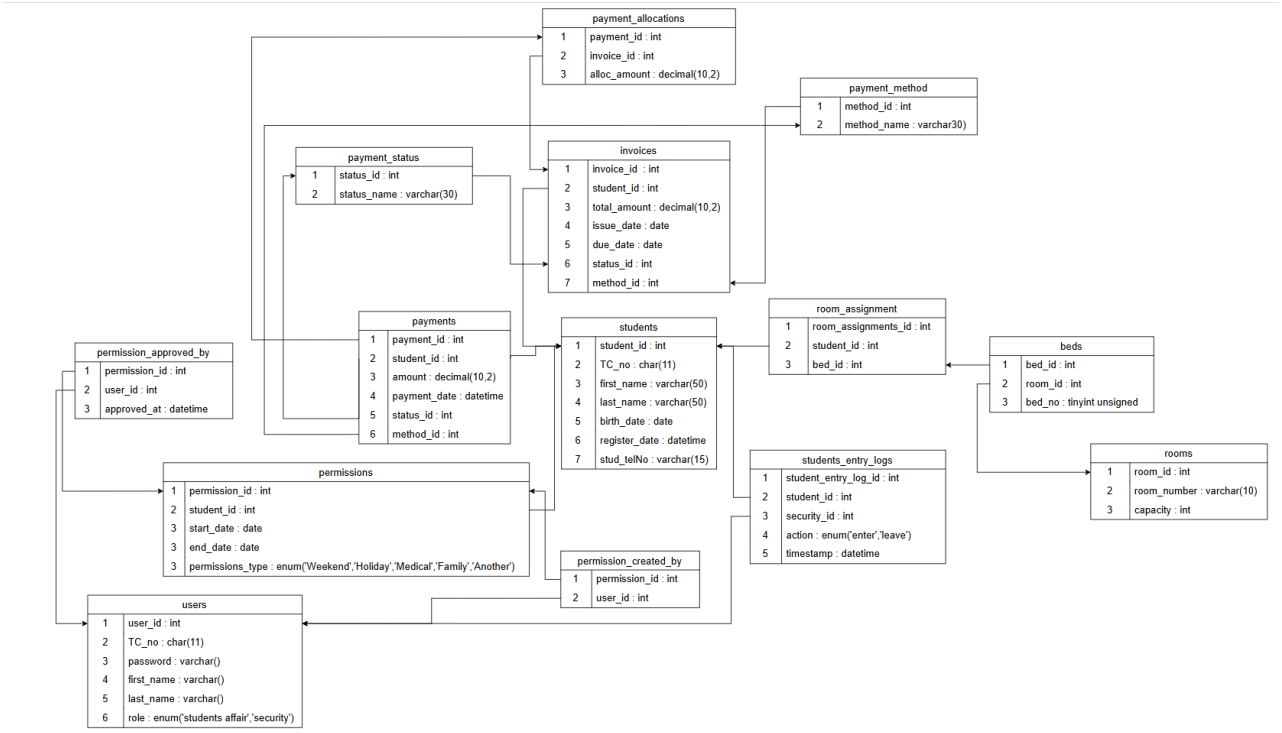

Figure 2. Simplified ER Diagram