AP Project Report

University ERP

Nalin Gupta, Tanish Jindal
Roll Number: 2024362, 2024579

1 Introduction

1.1 Project Overview

The University ERP System is a desktop application designed to streamline academic operations for a university. The system provides comprehensive functionality for managing users, courses, sections, enrollments, and grades while maintaining strict access control and data security.

1.2 Key Features

- **Role-Based Access Control**: Three distinct user roles (Admin, Instructor, Student) with specific permissions
- **Secure Authentication**: Password hashing using bcrypt with no plaintext password storage
- **Dual Database Architecture**: Separate databases for authentication and ERP data
- **Maintenance Mode**: System-wide read-only mode for students and instructors
- **Modern Dark UI**: Contemporary interface using FlatLaf with sidebar navigation

1.3 Technology Stack

- **Language**: Java JDK 21
- **UI Framework**: Java Swing with FlatLaf (Dark Theme)

- **Layout Manager**: MigLayout
- **Database**: MySQL (auth_db and erp_db)
- **Password Hashing**: jBCrypt
- **Logging**: SLF4J with Logback
- **Testing**: JUnit 5 with Mockito

2 Compilation and Execution

2.1 Prerequisites

- Java Development Kit (JDK) 21 or higher
- MySQL Server 8.0 or higher
- Maven 3.6 or higher (for dependency management)

2.2 Database Setup

1. **Configure database credentials**

   Open `src/resources/application.conf` and update the database user, password, and MySQL `bin` directory to match your local setup:

   ```
   db {
     host = "localhost"
     port = "3306"
     user = "root"
     password = "YOUR_PASSWORD"
   }

   mysql_paths {
     # Example for macOS:
     # bin_directory = "/usr/local/mysql/bin/"
     # Example for Windows:
     # bin_directory = "C:/Program Files/MySQL/MySQL
         Server 8.0/bin/"
   }
   ```

2. Start MySQL server

3. Execute the seed script:

```
mysql -u root -p < small_seed.sql
```

OR

```
mysql -u root -p < large_seed.sql
```

(for a larger seed data)

4. This creates two databases: `auth_db` and `erp_db`

2.3 Compilation

Navigate to the project root directory and execute:

```
mvn clean compile
```

2.4 Execution

Run the application using Maven:

```
mvn exec:java -Dexec.mainClass="edu.univ.erp.ui.Main"
```

Or execute the compiled JAR:

```
java -jar target/university-erp-1.0.jar
```

Or run the project by creating an `IntelliJ Maven` Project, and running the `Main.java` file from there.

2.5 Default Credentials

Username	**Password**	**Role**
admin1	pass123	Admin
inst1	pass123	Instructor
stu1	pass123	Student

3 System Architecture

3.1 Database Schema

3.1.1 Auth Database (auth_db)

Column	Type	Description
user_id	INT	Primary key, auto-increment
username	VARCHAR(50)	Unique username
role	ENUM	Student, Instructor, or Admin
password_hash	VARCHAR(255)	bcrypt hashed password
status	VARCHAR(20)	Account status (Active/Inactive)
last_login	TIMESTAMP	Last successful login time

Table 1: users_auth Table Structure

3.1.2 ERP Database (erp_db)

Table	Purpose
students	Student profiles (user_id, roll_no, program, year)
instructors	Instructor profiles (user_id, name, department)
courses	Course catalog (course_id, code, title, credits)
sections	Course sections (section_id, course_id, instructor_id, day_time, room, capacity, semester, year)
enrollments	Student registrations (enrollment_id, student_id, section_id, status)
grades	Assessment scores (grade_id, enrollment_id, component, score, final_grade)
settings	System settings (setting_key, setting_value)

Table 2: ERP Database Tables

3.2 Final Grade Input and Computation

The ERP system allows instructors to enter the assessment scores and grading thresholds for each course section. All component scores entered by the instructor are assumed to already be **scaled out of 100**. The ERP then computes the final grade based on simple addition and the grading thresholds provided by the instructor.

3.2.1 Assessment Components

Each course uses the following three components, each with a score between 0 and 100:

- **Quiz**

- **Mid-Term Exam**
- **End-Term Exam**

The instructor enters these component scores directly in the Gradebook interface. No automatic weighting or scaling is applied.

3.2.2 Final Score Calculation

The ERP computes the final numeric score using:

$$\text{FinalScore} = \text{Quiz} + \text{MidTerm} + \text{EndSem}$$

The maximum possible final score is 100.

3.2.3 Instructor-Defined Grading Thresholds

Unlike systems with fixed grading schemes, this ERP allows each instructor to define their own grading thresholds. During grade entry, the instructor provides the minimum score required for each letter grade.
An example of instructor-provided thresholds:

Letter Grade	**Minimum Score Required**
A+	70+
A	60+
B	50+
C	40+
D	30+
F	Below 30

These values are stored in the `grades` table along with the student's final score.

3.2.4 Letter Grade Assignment Logic

Once the thresholds are stored, the ERP assigns the final letter grade using the following procedure:

1. Compute the student's total score (0–100).
2. Compare the score against the instructor-defined thresholds.
3. Assign the highest letter grade for which the score meets the minimum requirement.

Formally, if the instructor provides thresholds:

$$T_{A+}, T_A, T_B, T_C, T_D$$

then the assigned grade G is:

$$G = \begin{cases} A+, & \text{if FinalScore} \geq T_{A+} \\ A, & \text{if FinalScore} \geq T_A \\ B, & \text{if FinalScore} \geq T_B \\ C, & \text{if FinalScore} \geq T_C \\ D, & \text{if FinalScore} \geq T_D \\ F, & \text{otherwise} \end{cases}$$

3.2.5 Validation

Before computing final grades, the following checks are performed:

- The sum of the components must add upto 100.
- Threshold values must be strictly decreasing (e.g., $T_{A+} \geq T_A \geq T_B$).
- All thresholds must be numeric and non-negative.

3.2.6 Integration With the Gradebook

After computation:

- The letter grade is shown in the instructor UI,
- Saved to the `grades` table,
- Displayed to students in the Student Dashboard.

This design allows flexibility while keeping the grading workflow simple and transparent for instructors.

4 Role-Based Access Control

4.1 Access Rules Implementation

The system enforces strict access control through the `AccessControl` class:

```
public class AccessControl {
    public boolean canModifySection(User user, int
       sectionId) {
        if (user.role() == Role.Admin) return true;
        if (user.role() == Role.Instructor) {
            // Verify instructor owns this section
            return sectionRepo.isInstructorForSection(
                user.userId(), sectionId);
        }
        return false;
    }
}
```

4.2 Maintenance Mode Enforcement

When maintenance mode is ON:

- **Banner Display**: Yellow warning banner appears on all dashboards
- **Write Blocking**: All data modification operations are blocked
- **Read Access**: View operations remain available
- **Admin Override**: Admins can still disable maintenance mode

5 Authentication and Security

5.1 Password Security

5.1.1 Hashing Implementation

Passwords are secured using bcrypt with work factor 10:

```
public class PasswordHasher {
    private static final int WORK_FACTOR = 10;

    public static String hash(String password) {
        return BCrypt.hashpw(password,
            BCrypt.gensalt(WORK_FACTOR));
    }

    public static boolean checkPassword(String plaintext
       ,
                                                     String hashed) {
        return BCrypt.checkpw(plaintext, hashed);
    }
}
```

5.1.2 Security Features

- **No Plaintext Storage**: Only bcrypt hashes stored in database
- **Automatic Salting**: bcrypt handles salt generation
- **Configurable Work Factor**: Can increase computational cost
- **Session Management**: Single active session per user
- **Change Password**: Users can update passwords securely

5.2 Database Separation

5.2.1 Auth DB (auth_db)

- Contains only authentication data
- Accessed through `UserAuthRepository`
- Handles login verification
- Stores password hashes
- Tracks login attempts (removed lockout in final version)

5.2.2 ERP DB (erp_db)

- Contains all application data
- No password or authentication information
- Linked to auth_db via `user_id`
- Accessed through domain-specific repositories

5.3 Login Flow

1. User enters username and password in LoginWindow
2. `AuthApi.login()` called with credentials
3. `AuthService.login()` validates credentials:
    - Query `auth_db` for user record
    - Verify password hash using bcrypt
    - If valid, update `last_login`
4. Load user profile from `erp_db` based on role
5. Create `User` object with profile data
6. Store in `SessionManager` singleton
7. Route to appropriate dashboard

5.4 Test Infrastructure

- **Framework**: JUnit 5
- **Mocking**: Mockito with MockitoExtension
- **Coverage**: Service layer and repository layer
- **Approach**: Unit tests with dependency injection

6 Additional Features

6.1 Implemented Features

1. **Auto-refresh on Tab Switch**: Panels refresh data when selected
2. **Comprehensive Logging**: SLF4J logging throughout application
3. **Input Validation**: Client-side and server-side validation
4. **Error Recovery**: Graceful handling of database errors

6.2 Bonus Features Attempted

- **Change Password Dialog**: Fully implemented with validation. Available to all users.
- **Grades CSV Export**: Students can export their final grades to a CSV file

7 Known Limitations

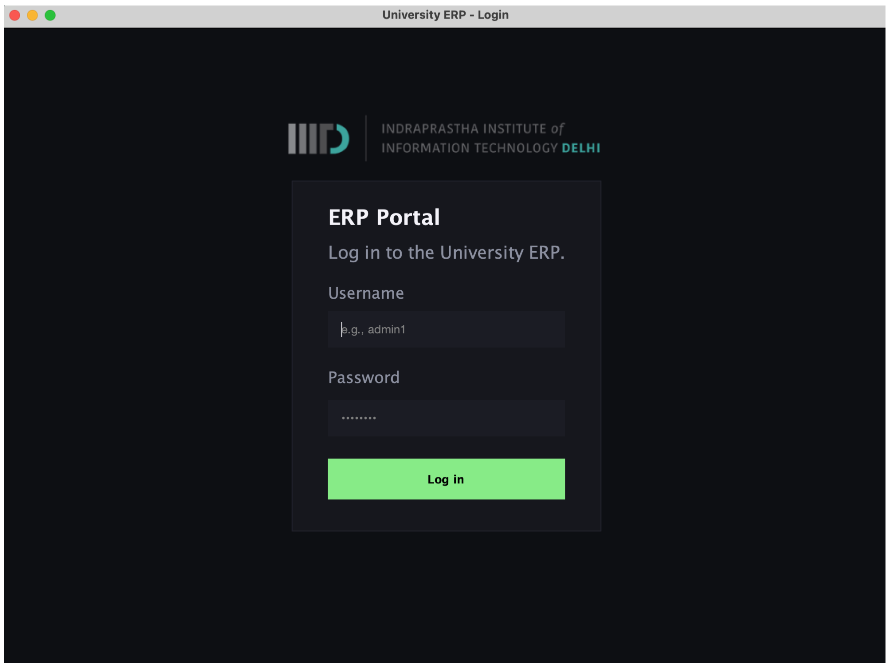
7.1 Current Limitations

- **Single Session**: Users cannot log in from multiple locations
- **No Backup/Restore**: Manual database backup required
- **No Email Notifications**: Manual communication required

8 Screenshots

This section presents key user interface screens of the ERP application for the three roles: Student, Instructor, and Admin.

8.1 Login Window



University ERP - Login

INDRAPRASTHA INSTITUTE of INFORMATION TECHNOLOGY DELHI

ERP Portal

Log in to the University ERP.

Username

e.g., admin1

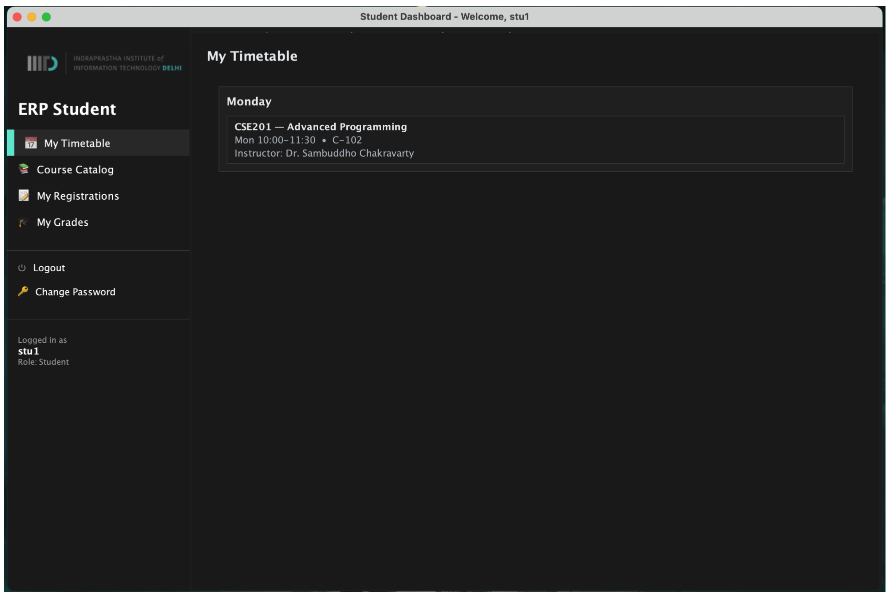
Password

••••••••

Log in

8.2 Student Dashboard

8.2.1 Student Timetable



Student Dashboard - Welcome, stu1

INDRAPRASTHA INSTITUTE of INFORMATION TECHNOLOGY DELHI

ERP Student

- My Timetable
- Course Catalog
- My Registrations
- My Grades

- Logout
- Change Password

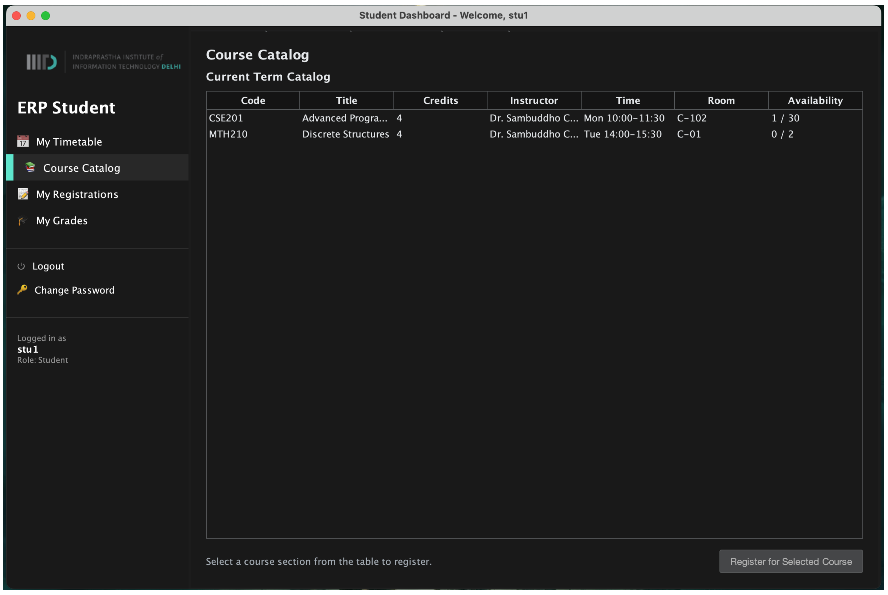
Logged in as
stu1
Role: Student

My Timetable

Monday

CSE201 — Advanced Programming
Mon 10:00–11:30 • C–102
Instructor: Dr. Sambuddho Chakravarty

8.2.2 Course Catalog


Student Dashboard - Welcome, stu1
INDRAPRASTHA INSTITUTE of INFORMATION TECHNOLOGY DELHI
ERP Student
My Timetable
Course Catalog
My Registrations
My Grades
Logout
Change Password
Register for Selected Course

Logged in as
stu1
Role: Student

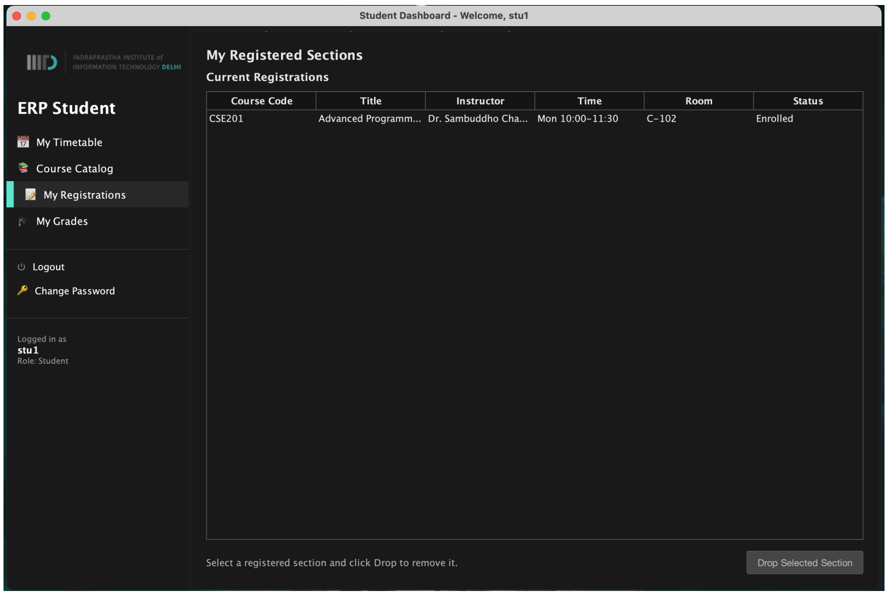
Course Catalog

Current Term Catalog

Code	Title	Credits	Instructor	Time	Room	Availability
CSE201	Advanced Progra...	4	Dr. Sambuddho C...	Mon 10:00-11:30	C-102	1 / 30
MTH210	Discrete Structures	4	Dr. Sambuddho C...	Tue 14:00-15:30	C-01	0 / 2

Select a course section from the table to register.

8.2.3 My Registrations



Logged in as
stu1
Role: Student

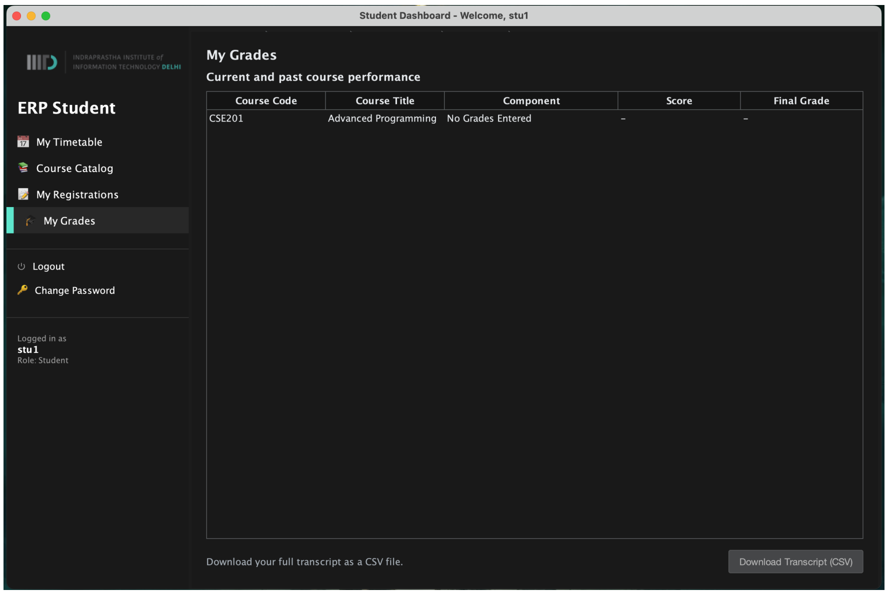
My Registered Sections

Current Registrations

Course Code	Title	Instructor	Time	Room	Status
CSE201	Advanced Programm...	Dr. Sambuddho Cha...	Mon 10:00-11:30	C-102	Enrolled

Select a registered section and click Drop to remove it.

8.2.4 My Grades


Student Dashboard - Welcome, stu1
INDRAPRASTHA INSTITUTE of INFORMATION TECHNOLOGY DELHI
ERP Student
My Timetable
Course Catalog
My Registrations
My Grades
Logout
Change Password
Logged in as
stu1
Role: Student
Download Transcript (CSV)

My Grades

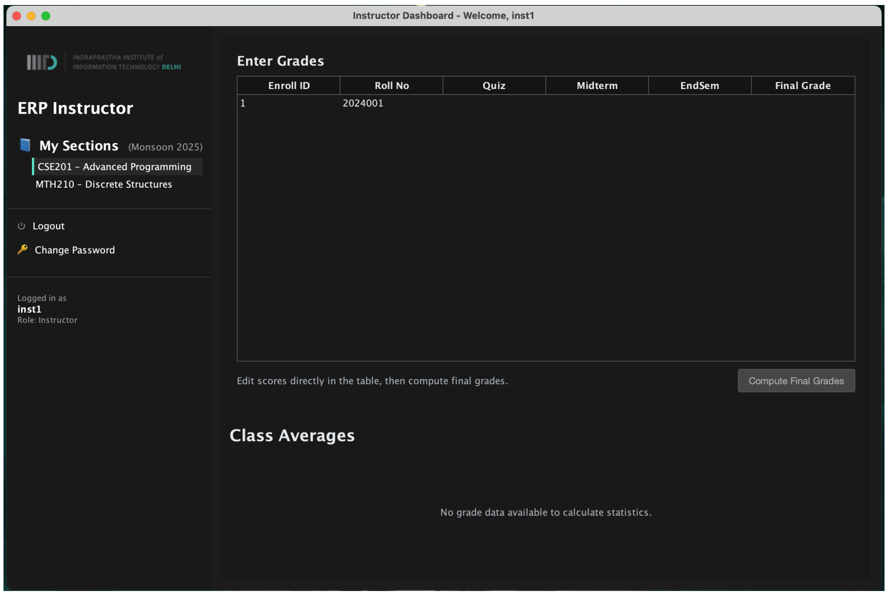
Current and past course performance

Course Code	Course Title	Component	Score	Final Grade
CSE201	Advanced Programming	No Grades Entered	–	–

Download your full transcript as a CSV file.

8.3 Instructor Dashboard

8.3.1 Enter Grades


Instructor Dashboard - Welcome, inst1
INDRAPRASTHA INSTITUTE of INFORMATION TECHNOLOGY DELHI
ERP Instructor
My Sections (Monsoon 2025)
CSE201 – Advanced Programming
MTH210 – Discrete Structures
Logout
Change Password
Logged in as
inst1
Role: Instructor
Compute Final Grades

Enter Grades

Enroll ID	Roll No	Quiz	Midterm	EndSem	Final Grade
1	2024001				

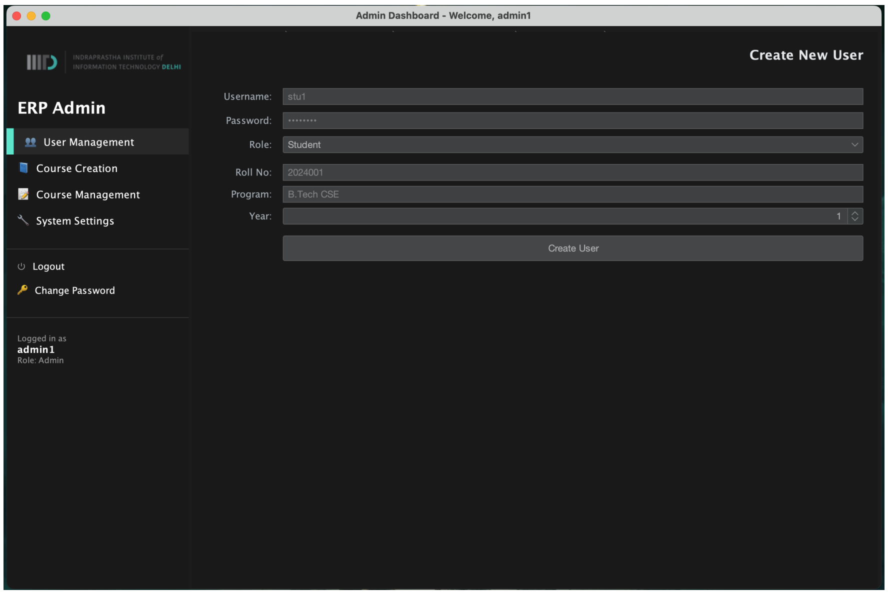
Edit scores directly in the table, then compute final grades.

Class Averages

No grade data available to calculate statistics.

8.4 Admin Dashboard

8.4.1 User Management



Admin Dashboard - Welcome, admin1

ERP Admin

- User Management
- Course Creation
- Course Management
- System Settings
- Logout
- Change Password

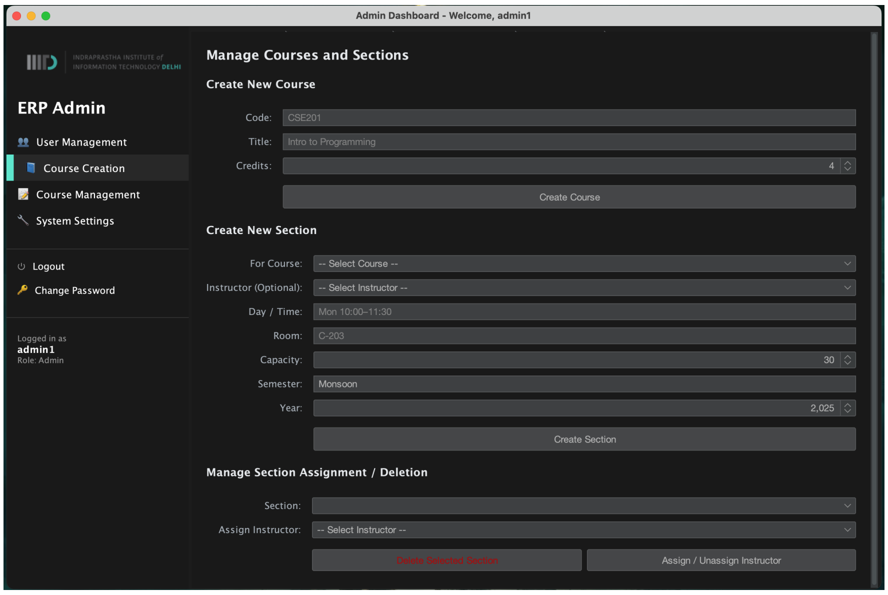
Logged in as
admin1
Role: Admin

Create New User

Username:	stu1
Password:	••••••••
Role:	Student
Roll No:	2024001
Program:	B.Tech CSE
Year:	1

Create User

8.4.2 Course Creation



Admin Dashboard - Welcome, admin1

ERP Admin

- User Management
- Course Creation
- Course Management
- System Settings
- Logout
- Change Password

Logged in as
admin1
Role: Admin

Manage Courses and Sections

Create New Course

Code:	CSE201
Title:	Intro to Programming
Credits:	4

Create Course

Create New Section

For Course:	-- Select Course --
Instructor (Optional):	-- Select Instructor --
Day / Time:	Mon 10:00–11:30
Room:	C-203
Capacity:	30
Semester:	Monsoon
Year:	2,025

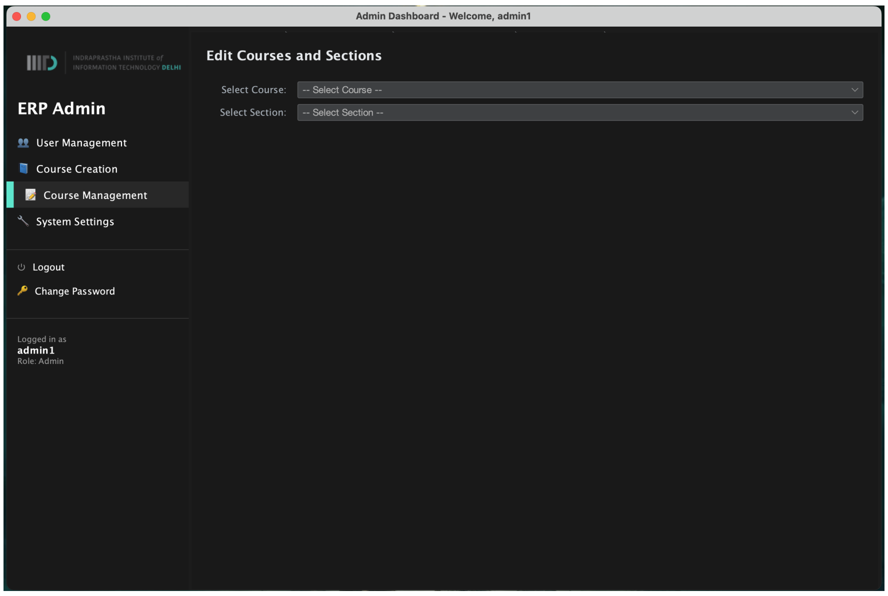
Create Section

Manage Section Assignment / Deletion

Section:	
Assign Instructor:	-- Select Instructor --

Delete Selected Section | Assign / Unassign Instructor

8.4.3 Course Management



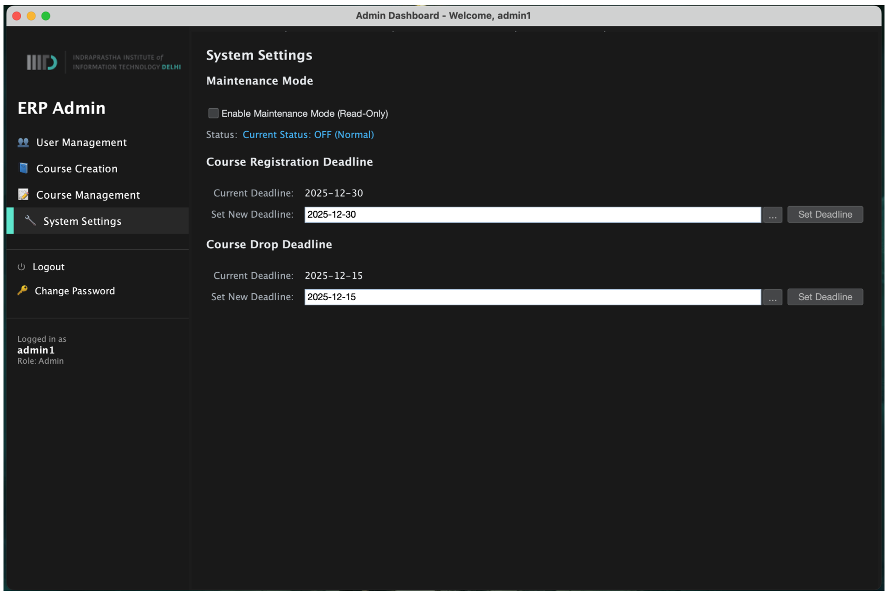
Edit Courses and Sections

Select Course: -- Select Course --

Select Section: -- Select Section --

Logged in as
admin1
Role: Admin

8.4.4 System Settings



System Settings

Maintenance Mode

☐ Enable Maintenance Mode (Read-Only)

Status: Current Status: OFF (Normal)

Course Registration Deadline

Current Deadline: 2025-12-30

Set New Deadline: 2025-12-30

Course Drop Deadline

Current Deadline: 2025-12-15

Set New Deadline: 2025-12-15

Logged in as
admin1
Role: Admin

A References

1. Oracle Java Documentation. https://docs.oracle.com/en/java/

2. FlatLaf Look and Feel. https://www.formdev.com/flatlaf/

3. MigLayout Documentation. http://www.miglayout.com/

4. jBCrypt Library. https://www.mindrot.org/projects/jBCrypt/

5. JUnit 5 User Guide. https://junit.org/junit5/docs/current/user-guide/

6. Mockito Documentation. https://site.mockito.org/

7. MySQL Reference Manual. https://dev.mysql.com/doc/

8. SLF4J Manual. http://www.slf4j.org/manual.html