

# DOCUMENTATION:UNIVERSITY ERP SYSTEM

VARUN MEHTA-2024607

MANNAT RAJ SINGH- 2024333

## 1. INTRODUCTION

Our project implements a University ERP Desktop application using Java Swing, JDBC, and MySQL.

It has 3 main dashboards-

- 1) Student
- 2) Instructor
- 3) Admin

The system follows a layered structure: Domain -> DAO -> Service -> UI for clean separation.

## 2. LAYOUT

```
ERP /  
|  
|-- .idea/  
|-- out/  
|-- src/  
|   |-- authen/  
|   |-- erp/  
|       |-- data/  
|       |-- domain/  
|       |-- dto/  
|       |-- service/
```

```

|   |   |-- ERPConnector
|   |
|   |-- Test/
|   |
|   |-- ui/
|   |   |-- Admin/
|   |   |   |-- panels/
|   |   |   |-- AdminDashboardFrame.java
|   |   |
|   |   |-- Instructor/
|   |   |   |-- panels/
|   |   |   |-- InstructorDashboardFrame.java
|   |   |
|   |   |-- Student/
|   |   |   |-- panels/
|   |   |   |-- StudentDashboardFrame.java
|   |   |
|   |   |-- ERPMain.java
|   |   |-- LoginFrame.java
|   |   |-- RoleRouter.java
|   |
|   |-- Main.java
|

```

### 3. HOW ROLES AND MAINTENANCE ENFORCED

- Every user logs in through the users\_auth table.
- After login, the system saves a session containing:  
    userId

username  
role (Admin / Instructor / Student)

- Based on the role stored in the session:
  - Admins get access to all admin features
  - Instructors only see instructor options
  - Students only see student functions
- Users can do different things based on their role:
- Admin - Can manage the entire system:
  - Add/remove users and courses
  - Assign teachers to classes
  - Control system settings and backups
- Instructor - Can manage their classes:
  - View their assigned sections
  - Calculate and enter student grades
  - Export class data
  - Class stats
- Student - Can manage their own enrollment:
  - Browse Catalog
  - Register for courses
  - Drop courses
  - Check their grades and transcript
  - View timetable

## 4. FINAL-GRADE WEIGHTING RULE

- Each section can have any number of assessment components — such as Quiz, Assignment, Project, Midsem, Endsem, Viva, etc.
- Instructors can freely add for each section.
- The component structure can vary from course to course and instructor to instructor (no fixed categories).
- Each component has an input-based weightage.
- Total weightage of all components must sum to 100 to ensure correct final grade calculation.

- If the grade is 90 or above, return “A.”
- If it is 80 or above, return “B.”
- If it is 70 or above, return “C.”
- If it is 60 or above, return “D.”
- Otherwise, return “F.”

## 5. EXTRAS ADDED

- **Profile Pages:** Both students and instructors have their own profile, showing basic details such as name, user id.
- **Account Lockout:** The account gets temporarily block for 30 seconds after 5 incorrect attempts.
- **Change Password:** Users can change their password in the login frame.
- **CSV Import/Export:** Instructors can upload and download student grades into a CSV file.
- **Backup & Restore:** The admin can backup and restore the university database.

## 6. AUTHENTICATION DATABASE TABLES

```
mysql> use auth_db;
Database changed
mysql> show tables;
+-----+
| Tables_in_auth_db |
+-----+
| users_auth         |
+-----+
1 row in set (0.00 sec)
```

```
mysql> desc users_auth;
+-----+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| user_id | int    | NO   | PRI | NULL    | auto_increment |
| username | varchar(50) | NO   |     | NULL    |
| role | enum('STUDENT','INSTRUCTOR','ADMIN') | NO   |     | NULL    |
| password_hash | varchar(255) | NO   |     | NULL    |
| status | enum('ACTIVE','LOCKED') | YES  |     | ACTIVE  |
| last_login | datetime | YES  |     | NULL    |
| failed_attempts | int    | YES  |     | 0       |
| lock_time | datetime | YES  |     | NULL    |
+-----+-----+-----+-----+-----+-----+
8 rows in set (0.00 sec)
```

## 7. ERP DATABASE TABLES

```
mysql> show tables;
+-----+
| Tables_in_univ_erp |
+-----+
| assessment_components |
| courses |
| enrollments |
| grades |
| instructors |
| notifications |
| section_labels |
| sections |
| settings |
| students |
+-----+
10 rows in set (0.02 sec)
```

```
mysql> desc assessment_components;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id | int | NO | PRI | NULL | auto_increment |
| section_id | int | NO | UNI | NULL | |
| name | varchar(50) | NO | | NULL | |
| weight | double | NO | | NULL | |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

```
mysql> desc courses;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| course_id | int | NO | PRI | NULL | auto_increment |
| code | varchar(20) | NO | UNI | NULL | |
| title | varchar(200) | NO | | NULL | |
| credits | int | NO | | NULL | |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

```
mysql> desc enrollments;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| enrollment_id | int | NO | PRI | NULL | auto_increment |
| student_id | int | NO | | NULL | |
| section_id | int | NO | | NULL | |
| status | enum('ENROLLED','DROPPED','WAITLISTED','COMPLETED') | YES | | ENROLLED | |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

```
mysql> desc grades;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| grade_id | int | NO | PRI | NULL | auto_increment |
| enrollment_id | int | NO | | NULL | |
| component | varchar(50) | NO | | NULL | |
| score | double | YES | | NULL | |
| final_grade | varchar(10) | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

```
mysql> desc instructors;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| user_id | int | NO | PRI | NULL |
| department | varchar(100) | YES | UNI | NULL |
| title | varchar(50) | YES | MUL | NULL |
+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

```
mysql> desc section_labels;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| label_id | int | NO | PRI | NULL | auto_increment |
| section_id | int | NO | UNI | NULL |
| course_id | int | NO | MUL | NULL |
| label | varchar(50) | NO | | NULL |
+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

```
mysql> desc notifications;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| notification_id | int | NO | PRI | NULL | auto_increment |
| user_id | int | NO | | NULL |
| message | text | NO | | NULL |
| created_at | timestamp | YES | | CURRENT_TIMESTAMP | DEFAULT_GENERATED |
| is_read | tinyint(1) | YES | | 0 |
+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

```
mysql> desc sections;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| section_id | int | NO | PRI | NULL | auto_increment |
| course_id | int | NO | | NULL |
| instructor_id | int | YES | | NULL |
| day_time | varchar(100) | YES | | NULL |
| room | varchar(50) | YES | | NULL |
| capacity | int | NO | | NULL |
| semester | varchar(20) | YES | | NULL |
| year | int | YES | | NULL |
| status | enum('OPEN', 'CLOSED', 'CANCELLED') | YES | | OPEN |
+-----+-----+-----+-----+-----+
9 rows in set (0.00 sec)
```

```
mysql> desc settings;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| k | varchar(100) | NO | PRI | NULL |
| v | varchar(200) | YES | | NULL |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

```
mysql> desc students;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| user_id | int | NO | PRI | NULL |
| roll_no | varchar(30) | YES | UNI | NULL |
| program | varchar(100) | YES | | NULL |
| year | int | YES | | NULL |
+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

## 8. FEW FEATURES

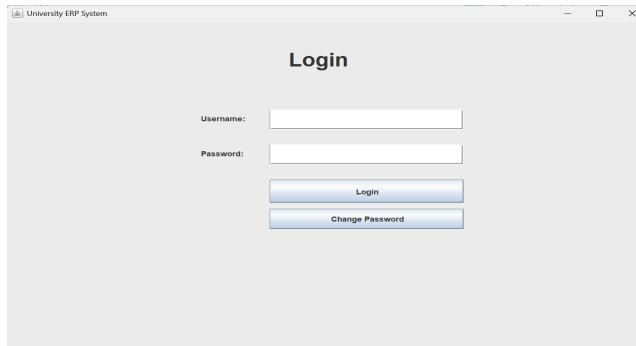


Figure 1: Login



Figure 2: Change Password

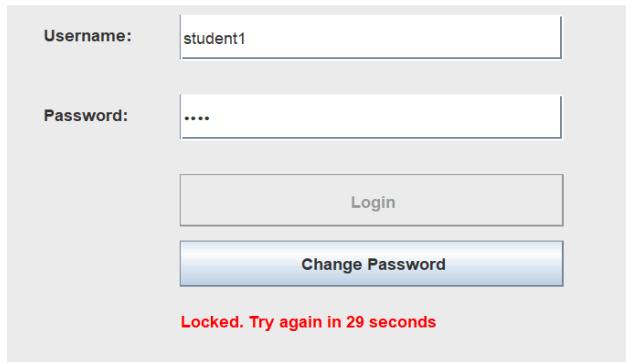


Figure 3: Lockout

# STUDENT

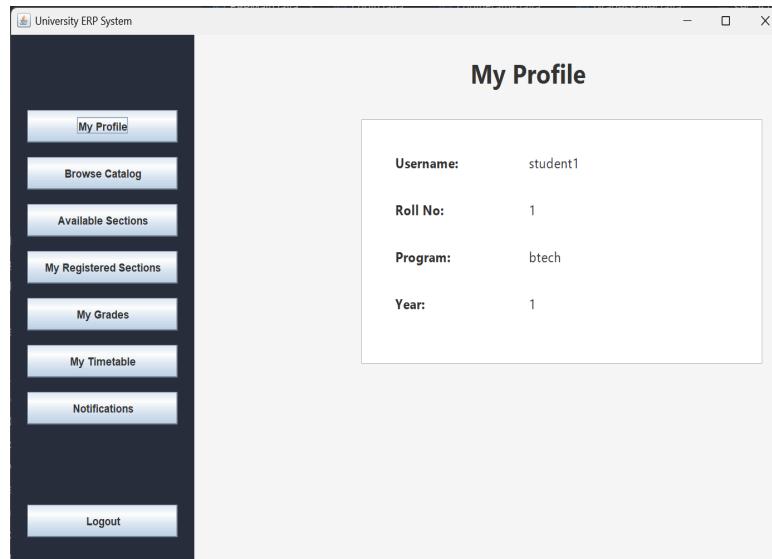


Figure 4: Student Dashboard

Browse Course Catalog			
Course Code	Title	Credits	Instructor
CSE101	Course Title 1	4	instructor1
CSE102	Course Title 2	4	
CSE103	Course Title 3	4	
CSE104	Course Title 4	4	
CSE105	Course Title 5	4	
CSE106	Course Title 6	4	
CSE107	Course Title 7	4	
CSE108	Course Title 8	4	
CSE109	Course Title 9	4	
CSE110	Course Title 10	4	
CSE111	Course Title 11	4	
CSE112	Course Title 12	4	
CSE113	Course Title 13	4	
CSE114	Course Title 14	4	
CSE115	Course Title 15	4	
CSE116	Course Title 16	4	
CSE117	Course Title 17	4	
CSE118	Course Title 18	4	
CSE119	Course Title 19	4	
CSE120	Course Title 20	4	
CSE121	Course Title 21	4	
CSE122	Course Title 22	4	
CSE123	Course Title 23	4	
CSE124	Course Title 24	4	

Figure 5: Browse Catalog

**Available Sections**

Semester: 1 ▾

Section	Course Code	Title	Capacity	Semester	Year	Status
sec_A	CSE101	Course Title 1	20	1	1	OPEN
forrest	CSE101	Course Title 1	1	1	1	OPEN

**Refresh** **Register**

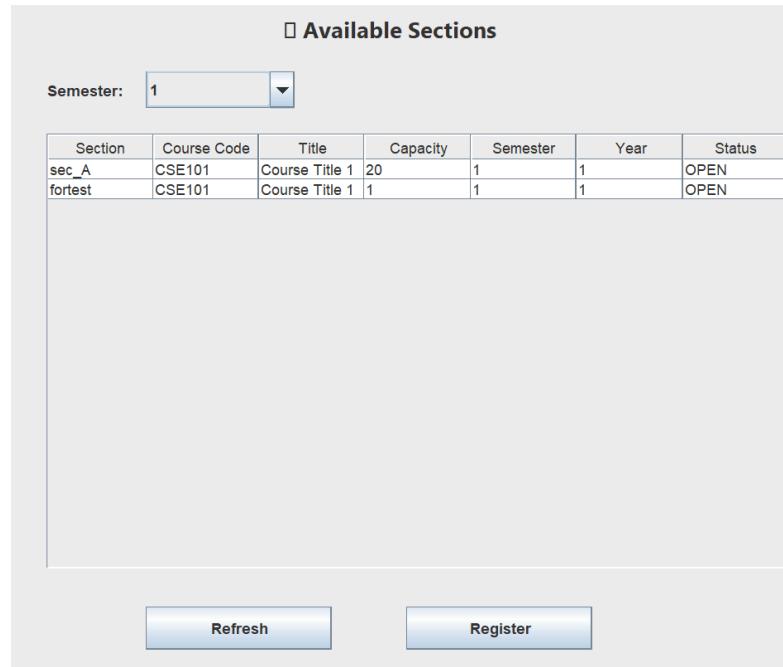


Figure 6: Register Section

**My Grades**

CSE101 ▾ Load **Final Grade: 89.5 (B)**

Component	Score	Weight
quiz	89.0	0.5
midsem	90.0	0.5

**Refresh All** **Download CSV**

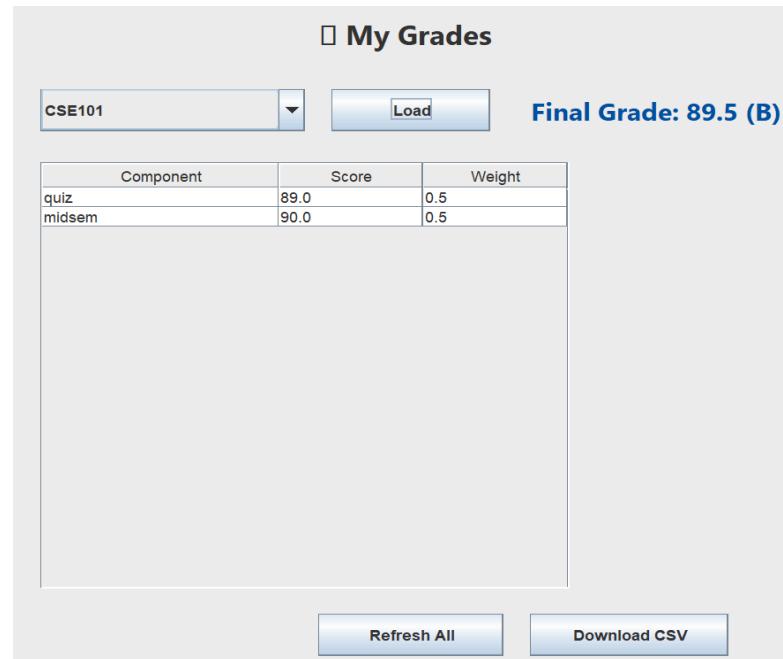


Figure 7: My grades

# INSTRUCTOR

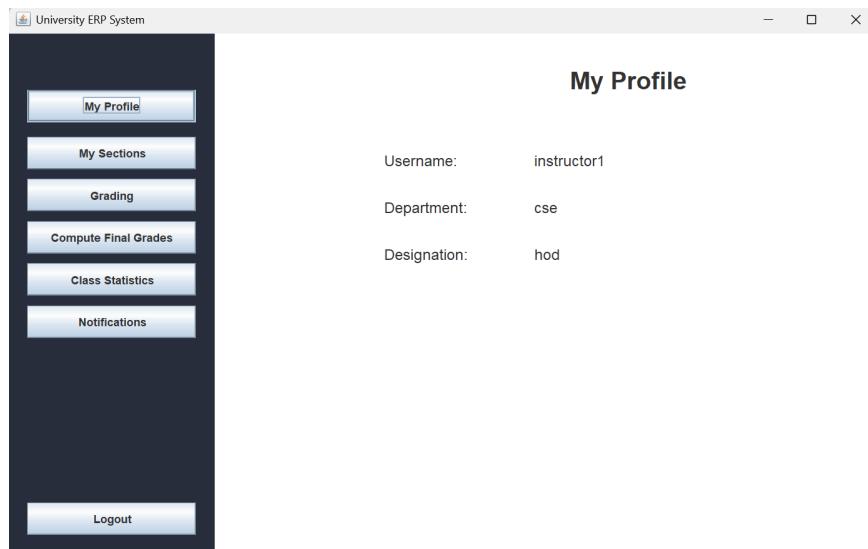


Figure 8: Instructor Profile

The screenshot shows the 'Define Assessment Component' page. The title 'Define Assessment Component' is at the top center. Below it, there are three input fields: 'Select Section:' with a dropdown menu showing 'CSE101 | sec\_A', 'Component Name:' with an empty input field, and 'Weight (e.g., 0.2):' with an empty input field. At the bottom is a large blue 'Add Component' button.

Figure 9: Assessment Component

**Enter Student Score**

Section:

Component:

Student:

Score (0 - 100):

Figure 10: Enter Score

Select Section:

Component	Avg	High	Low
quiz	89.00	89.00	89.00
midsem	90.00	90.00	90.00

Figure 11: Class Statistics

# ADMIN

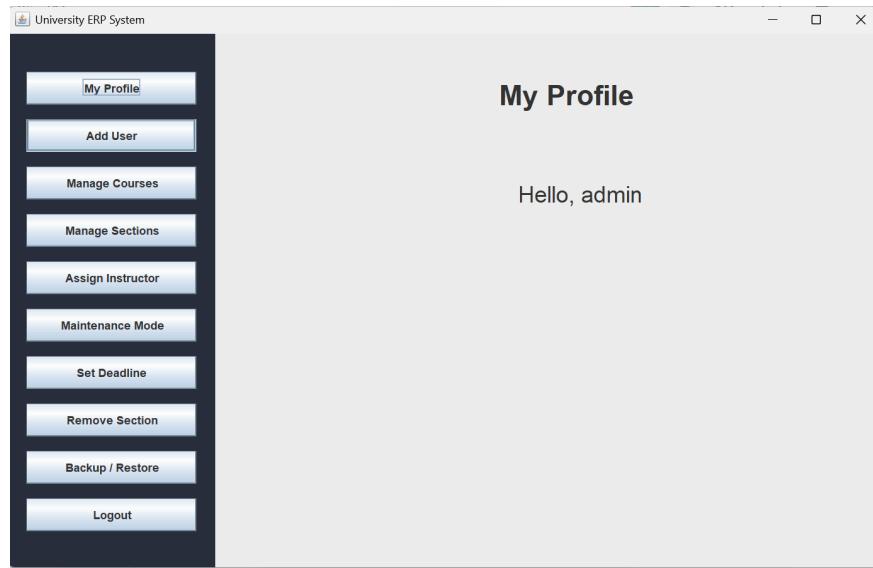


Figure 12: Admin Dashboard

### Add New User

Student     Instructor     Admin

### Add New Student

Username:

Password:

Roll No:

Program:

Year:

Figure 13: Add Users(Student / Instructor / Admin)

## Maintenance Mode

Maintenance Mode: OFF

Enable (ON)

Disable (OFF)

Figure 14: Maintenance Mode

← Back

## Set Drop/Add Deadline

Current Deadline:

Current drop deadline: 2024-11-10 18:00:00

New Deadline (yyyy-MM-dd HH:mm:ss):

Set Deadline

Figure 15: Set Deadline

## **9. CONCLUSION**

Overall, this project helped us understand how a real university ERP works internally. We implemented role-based access, separate databases for authentication and academic data, maintenance mode control, and a full registration + grading workflow. The project has scope for many more features, but even in its current form it is stable and covers all important requirements of the assignment