IZMIR UNIVERSITY OF ECONOMICS

DEPARTMENT OF SOFTWARE ENGINEERING

SE 360 PROJECT REPORT

Defne Yılmaz - 20220601078
Hüseyin Yontar - 20210601067

Contents

1 INTRODUCTION

This project provides a desktop application to ease the process for university lecturers to prepare exams and monitor students' performance.

The application provides tools for creating question banks with specific learning outcomes, generating exams tailored to these outcomes, and tracking students' success in the course material. By automating these tasks, the system saves time and enhances the accuracy and relevance of assessments.

With this application, lecturers can make decisions to improve their teaching strategies and ensure that students achieve the intended learning outcomes.

2 TECHNICAL DETAILS

This application was developed using Java Swing for the graphical user interface, providing an interactive environment for users to manage exams and questions.

Data was handled using an SQLite database, which ensures storage and retrieval of exam-related information such as questions, learning outcomes, and exam details, as well as lecturers, students and their grade information. To facilitate updates for different client socket programming is used, enabling synchronization of data to manage and display data dynamically in Swing components.

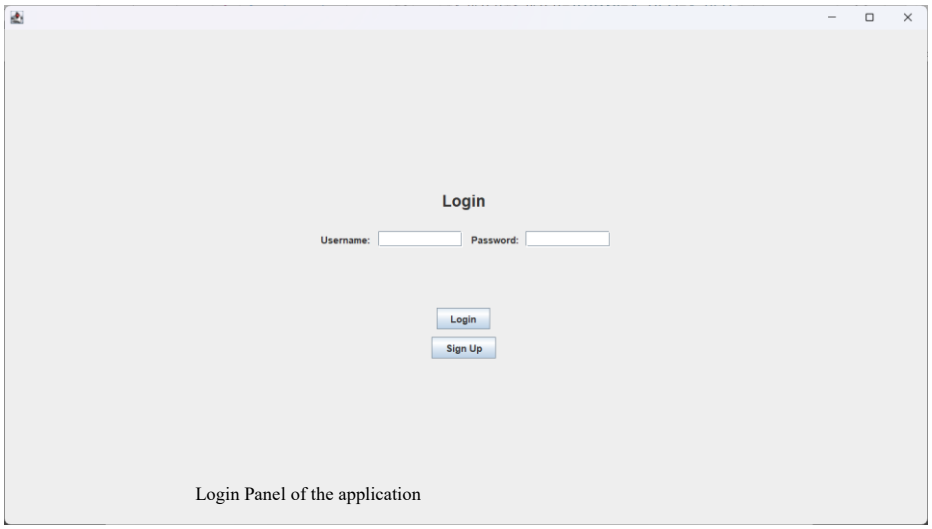
Data exchange between the server and client utilizes JSON for structured data serialization, ensuring compatibility. Also, the application uses Java’s standard I/O streams for data transfer over sockets. String operations and regular expressions help process text-based data.

3 FUNCTIONALITY

In this section, the core functionalities of the application, which include some panels are covered.

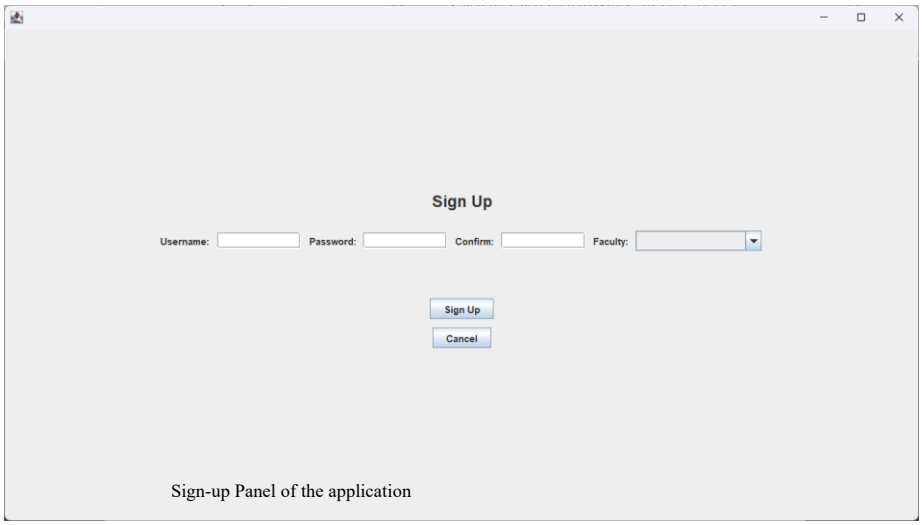
3.1 Login/Sign-Up Panel

This panel provides a user interface logging in and signing up, utilizing Java Swing’s Panel. Initially, the login screen is displayed, where users can enter their username and password to log in. If the inputs match an existing account in database, the user is authenticated, and their name and faculty are saved.



Login Panel of the application

If login fails, an error message prompts to sign up. “Sign Up” button switches to the sign-up screen, where users can create an account by entering a username, password, and selecting a faculty. The system validates the passwords and stores the new account in the database. After successful sign-up, the user is prompted to log in again.

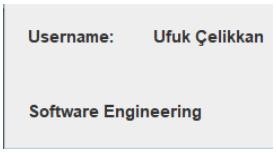


Sign-up Panel of the application

3.2 Left Panel

The left panel includes some elements for managing the user's profile and courses. It shows user information such as name and faculty, along with a list of courses the user is teaching. Additionally, the left panel provides options to add or remove courses, allowing instructors to manage their course list directly.

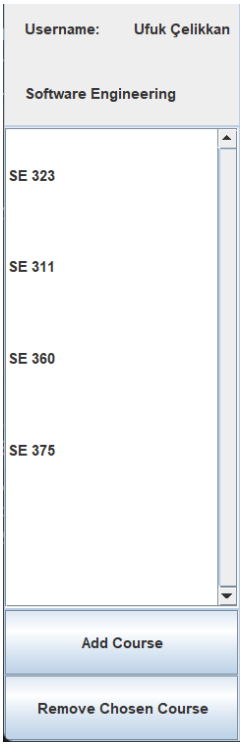
3.2.1 User Information



User Information Panel of the application

User information panel displays the logged-in user’s details. It includes their username and department. These are dynamically fetched from the Login/Sign-Up Panel.

3.2.2 Course List



Course List and Add/Remove Panel of the application

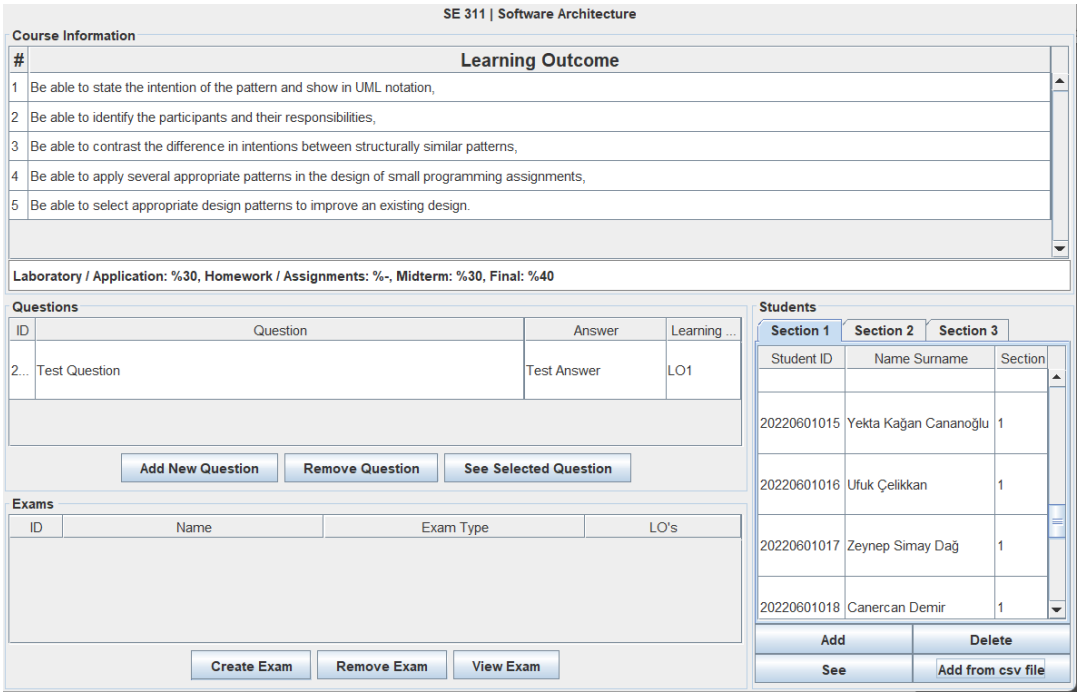
Course list panel dynamically displays a list of courses associated with the logged-in lecturer which are fetched from database.

3.2.3 Course Add/Remove

This button panel provides functionality for adding and removing courses. It includes two buttons. “Add Course” which allows the user to select a course from a list fetched from the database and add it to their course list, and “Remove Chosen Course” which removes the currently selected course from both the displayed list and the database.

3.3 Right Panel

The right panel includes some important elements for managing course-specific details. It displays the selected course's learning outcomes (LOs), along with options to create questions and exams. Additionally, it shows information about students enrolled in the course, allowing instructors to track and manage student grades directly.



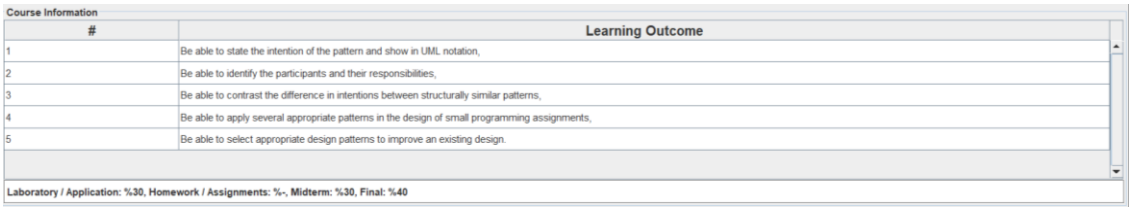
Right Panel of the application

3.3.1 Course Name

The course name panel only displays the course code and its corresponding course name. It fetches the course name from the database using the provided course code and updates the label to show both information.

3.3.2 Learning Outcomes

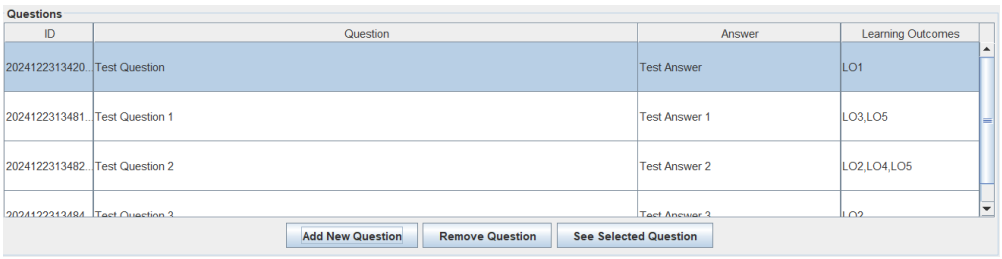
The learning outcome panel shows the learning outcomes which presents LOs in a structured table format with columns for their numbers and descriptions, and evaluation criteria for a selected course which displays course evaluation criteria such as participation, quizzes, and exams. Data comes from the database and through web scraping.



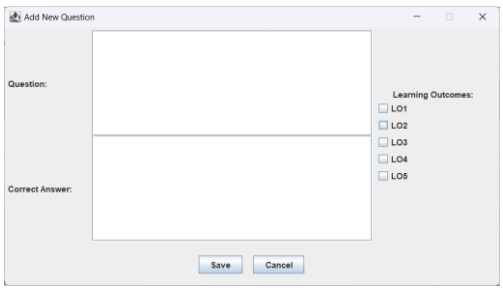
Learning Outcomes Panel of the application

3.3.3 Questions

The question panel shows a list of question for a selected course, presenting them in table with columns for their ID's, questions, answers and chosen LO's; allowing users to add, view, and remove questions linked to learning outcomes. It includes buttons for adding new questions, which opens a dialog for inputting question details and selecting relevant LOs, removing selected questions with confirmation, and viewing full details of a selected question in a popup. The panel queries the database to fetch and populate questions and LOs for the table.



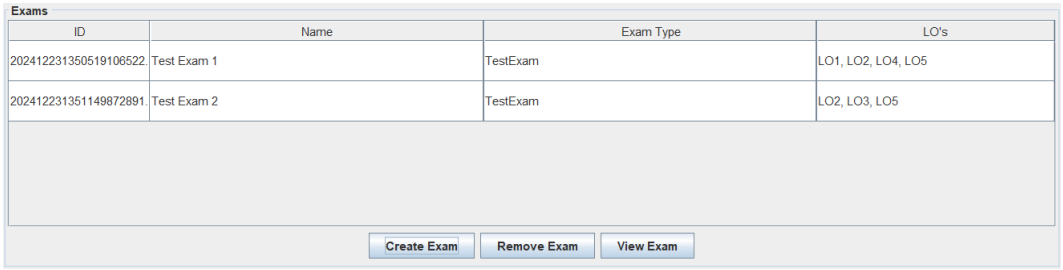
Question Panel of the application



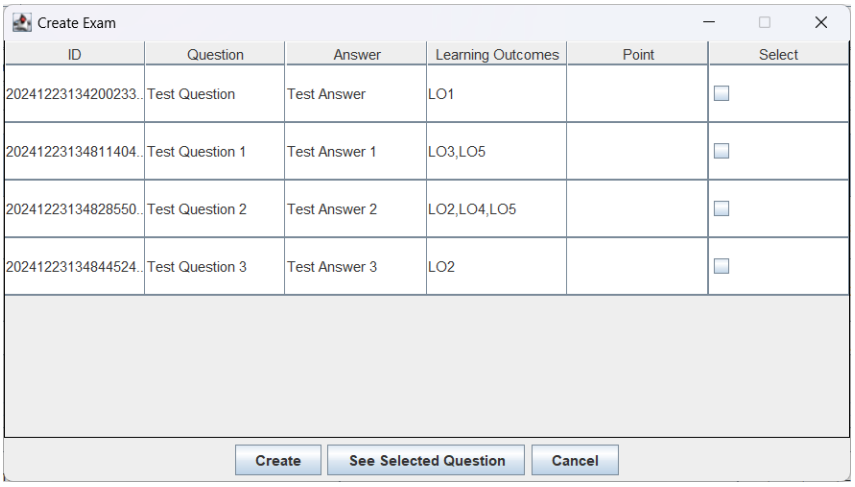
Add Question Panel

3.3.4 Exams

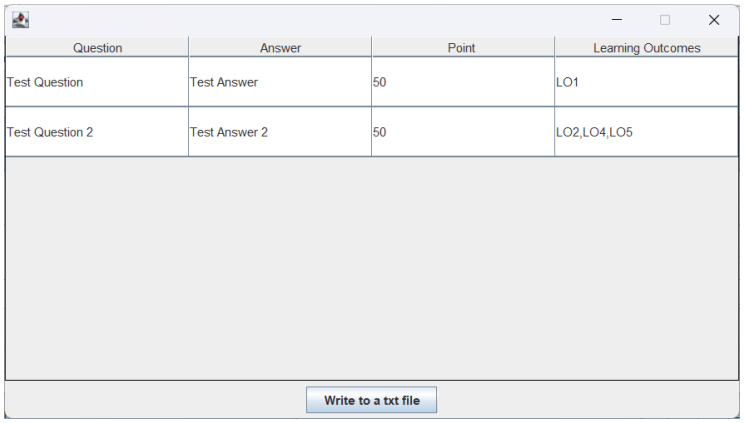
The exam panel displays a list of exams for selected course, presenting them in a table with columns of their ID's, names, types, and associated learning outcomes. The "Create Exam" button allows lecturers to select from course's questions to create exam by showing a table of questions for a selected exam, with columns for their IDs, questions, answers, learning outcomes, points, and selection status. It allows lecturers to select and set points for questions (validates that the points are between 0 and 100 and ensures the total equal to 100), then create exam. Data for both panels is fetched from the database. The "View Exam" button allows to view the questions for a selected exam in a pop-up window, with the option to export them to a text file. The "Remove Exam" button enables the deletion of exams and their associated questions from the database after user confirmation.



Exam Panel of the application

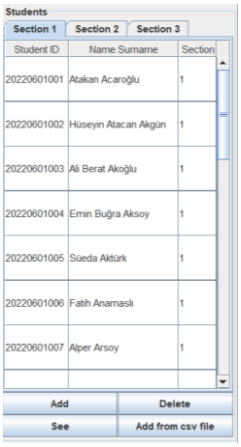


Create Exam Panel of the application



View Exam Panel of the application

3.3.5 Students



Student Panel of the application

The student panel displays a list of enrolled students for the selected course in a table, showing their IDs, names, and grades according to their section. The "Add Student" button allows lecturers to add new students to the course by selecting or entering their information, which is then validated before adding them to the database. The data is fetched from the database, ensuring that students are only added if they are not already enrolled. The "Remove Student" button allows for the deletion of students from the course, with confirmation required before removing their details from the database. Additionally, the panel enables lecturers to view and manage student enrolment across different sections. Lecturers can also set grades for each student based on their performance in specific exams they have taken, ensuring that the grades are properly assigned and recorded in the database for each question.

4 DATABASE AND NETWORK COMMUNICATION

In this section, details about the application's database and how the network communication is done using java.net sockets are provided. Briefly, the application uses SQLite for local database storage and a server program to handle client requests via sockets.

4.1 Database

SQLite is used as the local database for this application. The SQLite database file (db.sqlite) stays on the server machine. The application accesses the SQLite database using JDBC. On the first execution of the server application, the database file initialized automatically by creating required tables and populating initial data (the university's courses with CE and SE codes and learning outcomes) by scraping the university's website.

Schema Design:

Table 1: CourseInfo ("coursecode", "coursename", "evaluationcriteria")

Table 2: Courses ("*lecturername*", "coursename")

Table 3: LOs ("*coursecode*", "number", "text")

Table 4: Lecturers ("username", "password", "faculty")

Table 5: Students ("studentID", "name")

Table 6: Enrollments ("*studentID*", "coursecode", "section")

Table 7: Questions ("questionID", "*coursecode*", "question", "answer", "possiblepoint", "LO", "*examID*")

Table 8: Exams ("examID", "coursecode", "examtype", "examname", "los")

Table 9: Grades ("*studentID*", "*questionID*", "point")

4.2 Network Communication

The application uses Java.net sockets to enable communication between the server and clients. The server program listens client requests on port “12345”, queries the database, and sends responses.

The server has a multithreaded structure to handle multiple client requests concurrently. There is a main thread listens for incoming client connections and for each client connection, a new thread is created to process the request independently.

The server gets the request in string format and checks whether the request is a select statement or update statement. According to the request type executes the corresponding query and return a response to the client. The response format is List of HashMap for select queries using JSON and toList() function.

5 GITHUB LINK

https://github.com/defneeyilmaz/Exam-Analyze-App