SWE481 – Advanced Web Applications Engineering
First Semester 2025
SmartSchedule

SmartSchedule

#	Student's Name	Student's ID
1	Rand Albesher	442201693
2	Shahad Jaber Awaji	443201091
3	Deemah Alajaleen	443200931
4	Layan Almunasser	443201067

Supervised By:

Dr. Asma Alussayen

SWE481 – Advanced Web Applications Engineering
First Semester 2025
SmartSchedule

Introduction

The SmartSchedule project aims to develop an intelligent system for managing course schedules and examinations within the department. The system provides an interactive platform that enables real-time collaboration between scheduling and load committees, faculty members, and students, with version tracking and AI-driven recommendations. It relies on synthetic data for simulation and is delivered as a prototype to demonstrate how scheduling can be made more flexible, transparent, and efficient

GitHub Repository

Smart schedule github link

Draft Schedule

Link to draft schedule

Requirements Breakdown

- Functional Requirements

• Students

1. The student shall be able to provide elective course preferences.
2. The student shall be able to view their schedules.
3. The student shall be able to view exam timetables.
4. The student shall be able to give comments on preliminary schedules.
5. The student shall be able to give Feedback on preliminary schedules.

• Faculty Members

6. The faculty member shall be able to view their preliminary schedules.
7. The faculty member shall be able to view their courses

8. The faculty member shall be able to view their sections
9. The faculty member shall be able to view their exams
10. The faculty member shall be able to provide feedback on their schedules
11. The faculty member shall be able to provide feedback on their assignments.
12. The faculty member shall be able to update their availability
13. The faculty member shall be able to update their teaching preferences

- **Scheduling Committee**

14. The registrar shall be able to enter student counts per level into SmartSchedule.
15. The scheduling committee member shall be able to view student counts per level.
16. The scheduling committee member shall be able to view the default number of students per section.
17. The scheduling committee member shall be able to modify the default number of students per section.
18. The registrar shall be able to enter irregular-student data, including the
    18.1student's name
    18.2remaining courses from past levels
    18.3 courses needed to prevent falling behind.
19. The scheduling committee member shall be able to view irregular-student data entered by the registrar.
20. The scheduling committee member shall be able to send notifications to the registrar requesting irregular-student data.
21. The scheduling committee member shall be able to enter scheduling information from other departments and colleges before generating the preliminary schedule.
22. The scheduling committee member shall be able to create a survey to gather students' elective course preferences.
23. The scheduling committee member shall be able to publish the survey.

24. The scheduling committee member shall be able to set survey start and end dates.
25. The scheduling committee member shall be able to view survey responses.
26. The scheduling committee member shall be able to create a schedule.
27. The scheduling committee member shall be able to share versions of schedules with the load committee and students.
28. The scheduling committee member shall be able to receive feedback on schedules from the load committee and students.
29. The scheduling committee member shall be able to modify schedules.
30. The scheduling committee member shall be able to submit a final version of schedules to the deanship of registration.
31. The scheduling committee member shall be able to delete schedules.
32. The scheduling committee member shall be able to create an exam timetable.
33. The scheduling committee member shall be able to share versions of exam timetables with the students.
34. The scheduling committee member shall be able to receive feedback on exam timetables and students.
35. The scheduling committee member shall be able to modify exam timetables.
36. The scheduling committee member shall be able to submit a final version of exam timetables to the deanship of registration.
37. The scheduling committee member shall be able to delete exam timetables.
38. The scheduling committee member shall be able to add new scheduling rules.
39. The scheduling committee member shall be able to modify existing scheduling rules.
40. The scheduling committee member shall be able to delete existing scheduling rules.

- **Load Committee**

41. Load Committee members shall be able to view the schedule
42. Load Committee members shall be able to add feedback on the schedule .

- **User**

43. The user shall be able to sign up and create an account.
44. The user shall be able to log in using their credentials.
45. The user shall be able to log out.

- **System**

46. The system shall provide role-based authentication (student, faculty, committee).
47. The system shall provide notifications for relevant updates or deadlines.
48. The system shall auto-save changes in real time.
49. The system shall generate intelligent schedule recommendations using AI APIs.
50. The system shall allow search of schedules.
51. The system shall allow filtering of schedules.

System Architecture

We selected the Three-tier Architecture because it perfectly fits the needs of our system. This architecture divides the application into three clear layers:

1. **Presentation Layer**: Responsible for user interface and user interactions.
2. **Application Layer:** Responsible for the business logic and system services.
3. **Data Layer:** Responsible for storing and managing data.

This architecture was chosen for several reasons:

- **Clear separation of concerns:** each layer has its own responsibility, making the system easier to design and maintain.
- **Maintainability:** updates in one layer do not affect the others.
- **Scalability:** allows adding new clients in the future (such as a mobile application) without affecting backend or database design.
- **Reliability:** ensures data consistency and smooth system performance.
- **Flexibility**: supports integration with future services (e.g., AI recommendations or external university systems).

Frameworks and Technologies

- **Frontend: React.js**
  Provides a responsive and dynamic user interface with a component-based architecture, which speeds up development and simplifies maintenance.
- **Backend: Node.js with Express**
  Delivers high performance and supports real-time operations such as schedule updates and feedback, with easy integration of RESTful APIs.
- **Database: PostgreSQL**
  A relational database that guarantees strong data integrity, efficient handling of structured data (students, courses, schedules), and advanced querying capabilities.

Entity Relationship Diagram

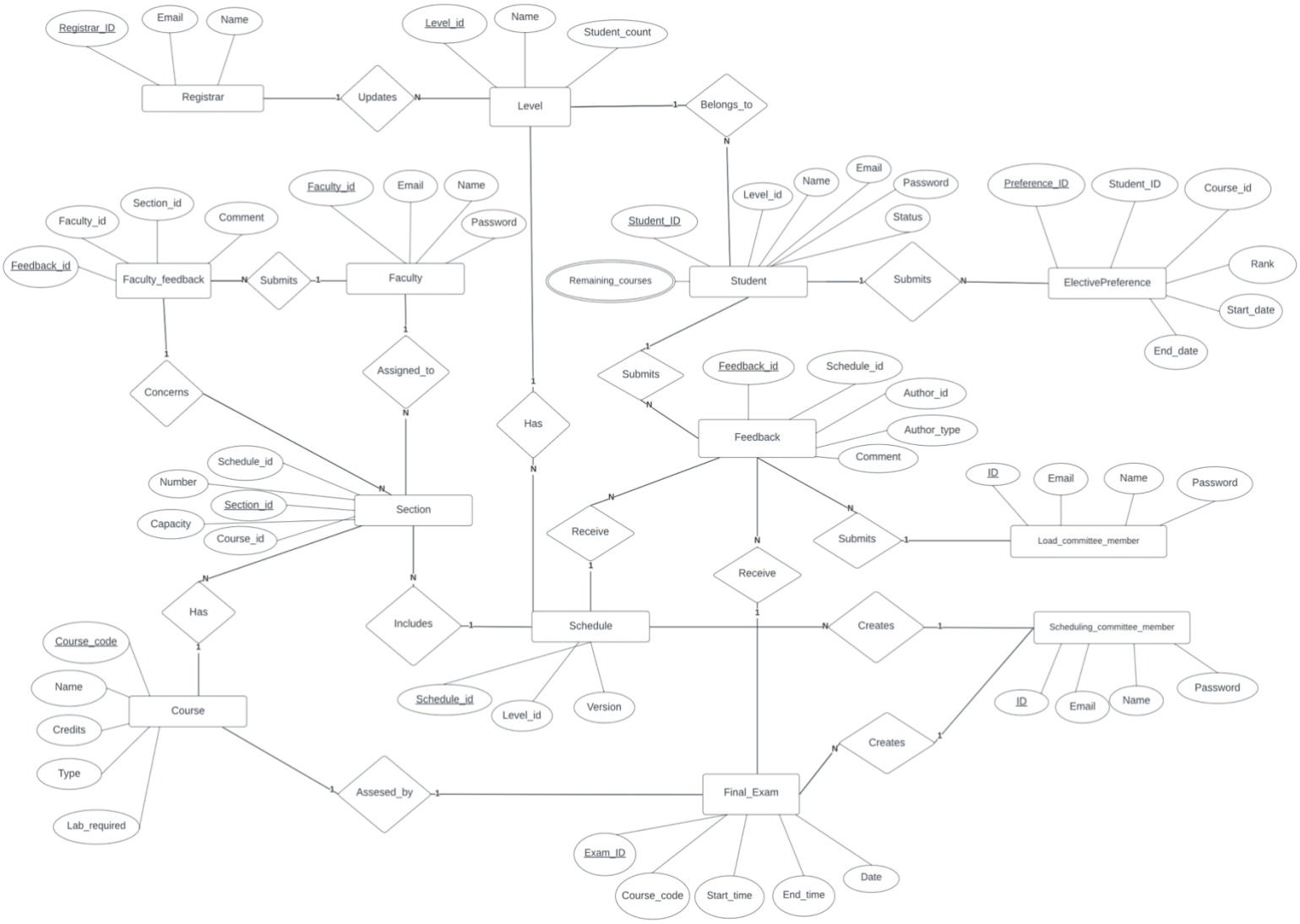

Figure 1 Entity–Relationship (ER) Diagram of the SmartSchedule System

The entity relationship diagram shown above illustrates the structure of the SmartSchedule system by presenting the main entities, their attributes, and the relationships that connect them.

1. Entities

- **Student** : defined by Student_ID and including attributes Level_ID, Name, Email, Password, Status, and Remaining_courses.

- **Level**: defined by Level_ID and including attributes Name and Student_count.

- **Registrar**: defined by Registrar_ID and including attributes Name and Email.
- **ElectivePreference**: defined by Preference_ID and including attributes Student_ID, Course_ID, Rank, Start_date, and End_date.
- **Schedule**: defined by Schedule_ID and including attributes Level_ID and Version.
- **Course**: defined by Course_code and including attributes Name, Credits, Type, and Lab_required.
- **Section**: defined by Section_ID and including attributes Schedule_ID, Course_ID,Number, and Capacity.
- **Faculty**:defined by Faculty_ID and including attributes Name, password and Email.
- **Faculty Feedback**: defined by Feedback_ID and including attributes Faculty_ID, Section_ID, and Comment.
- **Feedback**: defined by Feedback_ID and including attributes Schedule_ID, Author_ID, Author_type, and Comment.
- **Final Exam**: defined by Exam_ID and including attributes Course_code, Date, Start_time, and End_time.
- **Scheduling Committee Member**: defined by ID and including attributes Name, Email, and Password.
- **Load Committee Member**: defined by ID and including attributes Name, Email, and Password.

2. Relationships

Here, we explain how the entities are connected describing the main interactions and roles they play in the system:

- The Registrar updates Levels.
- Levels have Students and Schedules.
- Students submit Elective Preferences and Feedback.
- The Scheduling Committee Member creates Schedules and Final Exams.
- Schedules include Sections.
- Courses have Sections and exactly one Final Exam.
- Faculty members are assigned to Sections and can submit Faculty Feedback concerning their assignments.
- Both Students and Load Committee Members provide Feedback on Schedules and Final Exams.

Schema Diagram

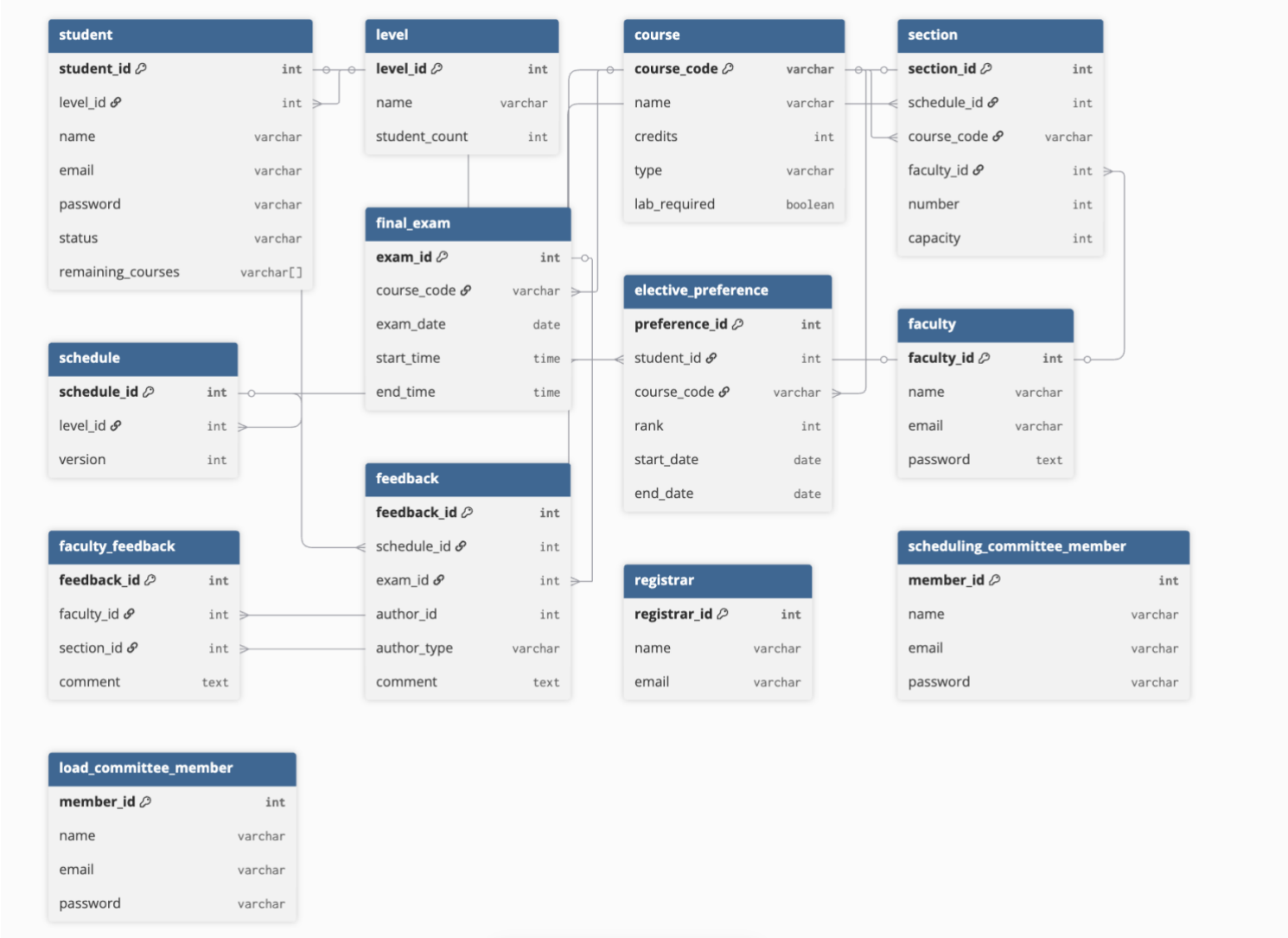

Figure 2 Relational Database Schema Diagram for the Smart Schedule System

Note: PK = Primary Key, FK = Foreign Key. PKs uniquely identify records within a table, while FKs establish relationships between tables.

1.Registrar

Attributes:

- registrar_id (PK): unique identifier.
- name: registrar's name.
- email: registrar's email.

2.Level

Attributes:

- level_id (PK): unique identifier for an academic level.
- name: label of the level.
- student_count: number of students registered in each level.

3.Student

Attributes:

- student_id (PK).
- level_id (FK → Level).
- name.
- email.
- password.
- status: regular/irregular.
- remaining_courses: list (array) of remaining courses.

4.Faculty

Attributes:

- faculty_id (PK).
- name.
- email.
- password.

5.Load Committee Member

Attributes:

- member_id (PK).
- name.
- email.
- password.

6.Scheduling Committee Member

Attributes:

- member_id (PK).
- name.
- email.
- password.

7.Course

Attributes:

- course_code (PK).
- name.
- credits.
- type: Required or Elective.
- lab_required: boolean flag.

8.Section

Attributes:

- section_id (PK).
- course_code (FK → Course).
- schedule_id (FK → Schedule).
- faculty_id (FK → Faculty).
- Number:5-digit section number.
- Capacity: default 25 students.

9.Schedule

Attributes:

- schedule_id (PK).
- level_id (FK → Level).
- version – version number for revisions.

10.Elective Preference

Attributes:

- preference_id (PK).
- student_id (FK → Student).
- course_code (FK → Course).
- rank: ranking among electives.
- start_date, end_date: survey open and close.

11.Feedback

Attributes:

- feedback_id (PK).
- schedule_id (FK → Schedule, nullable).
- exam_id (FK → Final Exam, nullable).
- author_id.
- author_type: Student or Load Committee.
- comment.

12.Faculty Feedback

Attributes:

- feedback_id (PK).
- faculty_id (FK → Faculty).
- section_id (FK → Section).
- comment.

13.Final Exam

Attributes:

- exam_id (PK).
- course_code (FK → Course).
- exam_date.
- start_time.
- end_time.

Seeded Data (Sample Records)

To validate the database schema, we seeded each table with sample records. Below are screenshots showing examples from some key tables. These records are for demonstration purposes, as the actual database contains seeded data for all tables.

1. Student Table

student_id int4	level... i...	name varchar	email varchar	passw... varc...	status varchar	remaining_courses text[]
442187654	3	Mona Faisal	442187654@student.ksu.edu.sa	mona321	Regular	["SWE497"]
443201067	1	John Doe	443201067@student.ksu.edu.sa	pass123	Regular	["CS101","CS203"]
444198452	1	Sara Ali	444198452@student.ksu.edu.sa	sara456	Irregular	["CS205","CS301","CS401"]
445209873	2	Khalid Omar	445209873@student.ksu.edu.sa	khalid789	Irregular	["CS201","CS305"]

Figure 3 Sample records from the student table

2. Course Table

course_code varchar	name varchar	credits int4	type varchar	lab_required bool
CENX303	Computer Communications & Networks	3	Required	FALSE
CSC111	Computer Programming (1)	3	Required	TRUE
CSC212	Data Structures	3	Required	FALSE
SWE211	Introduction to Software Engineering	3	Required	FALSE
SWE312	Software Requirements Engineering	3	Required	FALSE
SWE481	Advanced Web Applications Engineering	3	Elective	FALSE
SWE485	Selected Topics in Software Engineering	3	Elective	FALSE
SWE497	Graduation Project II	3	Required	FALSE

Figure 4 Sample records from the Course table

3. Section Table

section_id int4	schedule_id int4	course_code varchar	facul... i...	number int4	capacity int4
6	6	CSC111	NULL	10101	25
7	6	SWE211	NULL	10234	25
8	6	CSC212	NULL	20345	25
9	7	SWE312	NULL	20456	25
10	8	SWE481	NULL	30567	25

Figure 5 Sample records from the Section table

3. Faculty Table

member_id int4	name varchar	email varchar	password varchar
1	Dr. Hanan Yousif	hanan.yousif@ksu.edu.sa	sched123
2	Dr. Saleh Omar	saleh.omar@ksu.edu.sa	sched456

Figure 6 Sample records from the faculty table