School registering system

Student (**student_id**, first_name, last_name, department) - Ethan

Section (**section_id,** year, semester, course_id, courseName) - Sankalp

StudentCredentials (**student_id**, username, password) - Sankalp

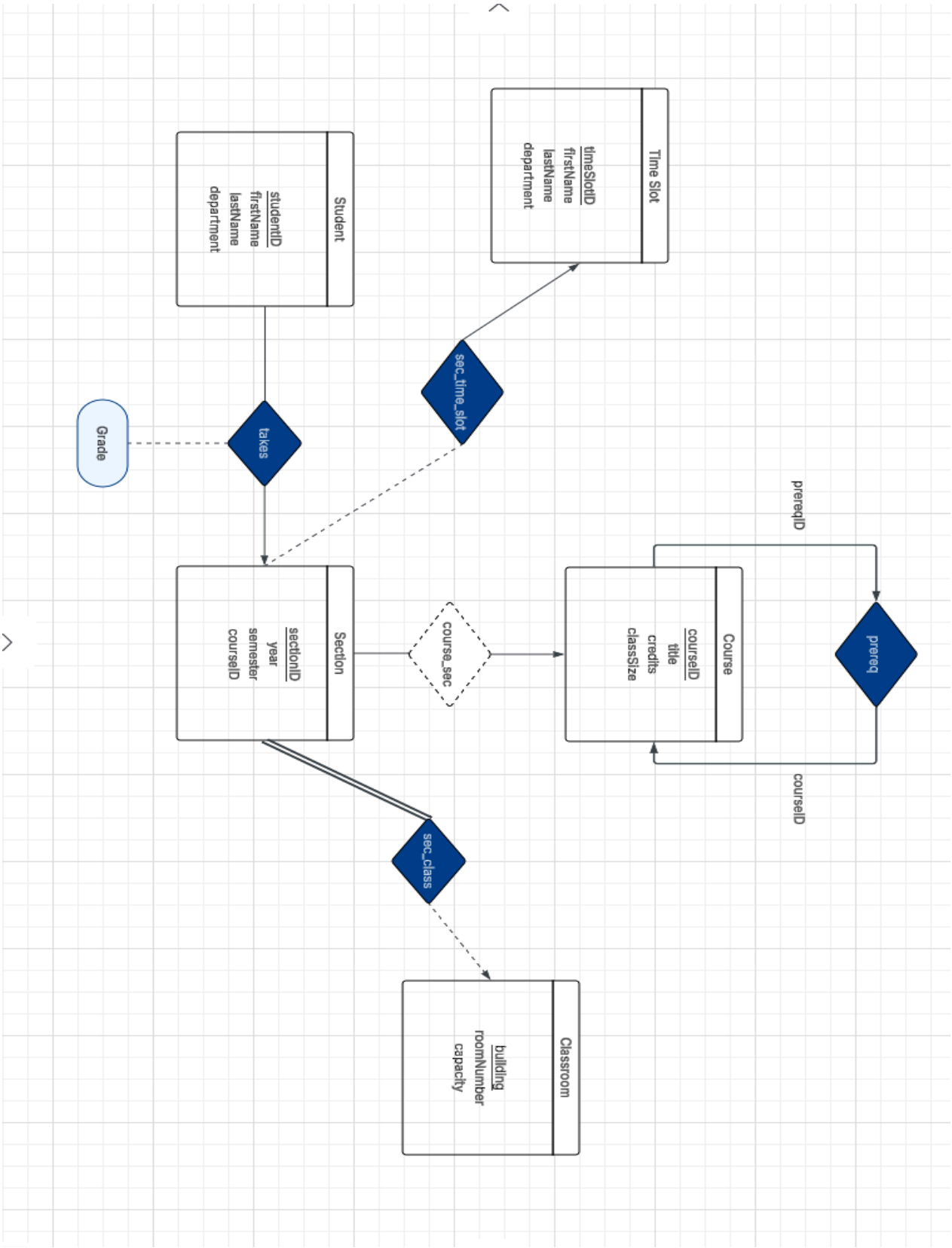
Course (**course_id,** courseName, credits, prereq) - Ethan

sect_timeSlot (**timeslot_id, section_id**, st art_time, end_time, day(s)) - Nat

Classroom (**classroom_id**, **section_id**, building, roomNumber, capacity) - Nat

Takes (**course_id, section_id**, grade, **student_id**) - Sankalp

(E-R) Diagram:


Time Slot
timeSlotID
firstName
lastName
department
Student
studentID
firstName
lastName
department
sec_time_slot
Grade
takes
prereqID
prereq
Course
courseID
title
credits
classSize
Section
sectionID
year
semester
courseID
course_sec
courseID
sec_class
Classroom
building
roomNumber
capacity

Detailed Description of the Relational Database Schema

The relational database schema for CMPT_391_P01 is designed to efficiently manage a college registration system by organizing data into normalized and logically structured tables. This design promotes data integrity, reduces redundancy, and allows for optimized query performance through the use of stored procedures, indexes, and views.

1. Core Tables and Relationships

- **Student**(StudentID, first_name, last_name, department): Stores student profiles. StudentID is the primary key.

- **Course**(CourseID, courseLabel, courseName, credits, prereq): Stores all course data. Includes a self-referencing foreign key to handle prerequisites.

- **Instructor**(InstructorID, FirstName, LastName, Department): Contains instructor records.

- **Section**(SectionID, ClassroomID, CourseID, CrseYear, Semester, CrseName, Capacity, InstructorID): Represents course offerings each semester. Foreign keys link to Course, Classroom, and Instructor.

- **Classroom**(ClassroomID, Building, RoomNumber, Capacity): Stores classroom information.

- **Sect_TimeSlot**(TimeSlotID, SectionID, StartTime, EndTime, DayOfWeek): Defines scheduled class Times.

- **Takes**(CourseID, SectionID, Grade, StudentID): A junction table representing enrollment and completed courses. Includes primary key on CourseID, SectionID, and StudentID.

- **Cart**(SectionID, StudentID, CourseID, CourseName): Temporarily stores courses a student selects prior to final registration.

- **StudentCredentials**(StudentID, Username, StudentPassword): Manages authentication and login.

2. Normalization and Data Integrity

The schema applies principles of normalization up to 3NF:

- Each table has a primary key and contains atomic values.
- Redundant data is minimized (e.g., Instructor and Student are distinct entities).
- Foreign key constraints enforce referential integrity (e.g., Takes references Student, Course, and Section).

3. Query Optimization Features

Views:

- **vw_SectionAvailability**: Aggregates section-level data (e.g., instructor, seat availability).
- **vw_StudentSchedule**: Provides student schedules in a normalized form.
- **vw_CourseCompletion**: Lists completed courses to help in transcript generation.

Indexing:

- **Clustered indexes on views** (e.g., vw_StudentSchedule and vw_CourseCompletion) enable fast lookups and grouping.

Stored Procedures:

- Examples include `RegisterStudentToSection`, `GetCompletedCoursesForStudent`, and `sp_ValidateStudentLogin`.
- These encapsulate logic for validations, filtering by semester, and transaction safety.

4. Design Rationale

The schema supports modularity and scalability. Each entity and interaction is explicitly defined to support:

- Multi-semester course registration.
- Validation of prerequisites and schedule conflicts.
- Efficient grade tracking and history retrieval.
- Logical UI integration with minimal SQL complexity on the front end.

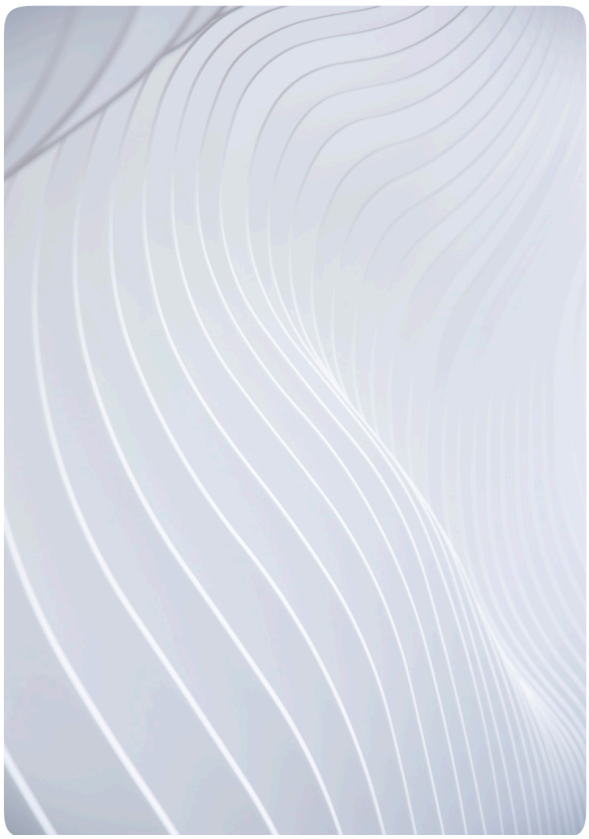
5. Support for Efficient Query Processing

- Filtering and joining are efficient due to foreign keys and indexed views.
- Schemabound views and transactions are used to ensure consistency.
- Stored procedures simplify logic reuse and help prevent SQL injection.

Conclusion

The database schema for CMPT_391_P01 is designed for robustness, data consistency, and ease of extensibility, while enabling smooth interaction with a .NET-based frontend UI through well-defined stored procedures and dynamic views.

Screen Mock-ups:



SecureCourse

Welcome

Email

Example@email.com

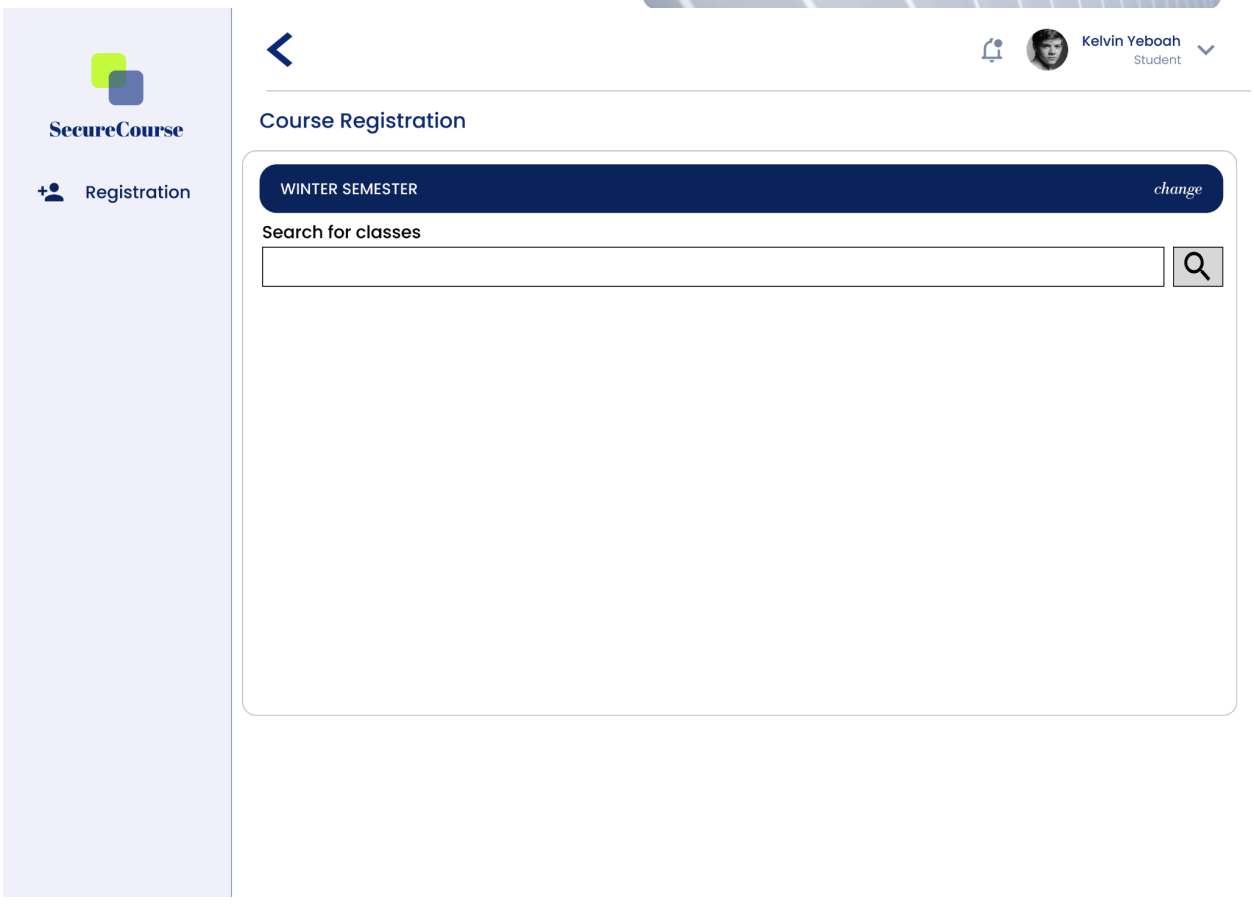
Password

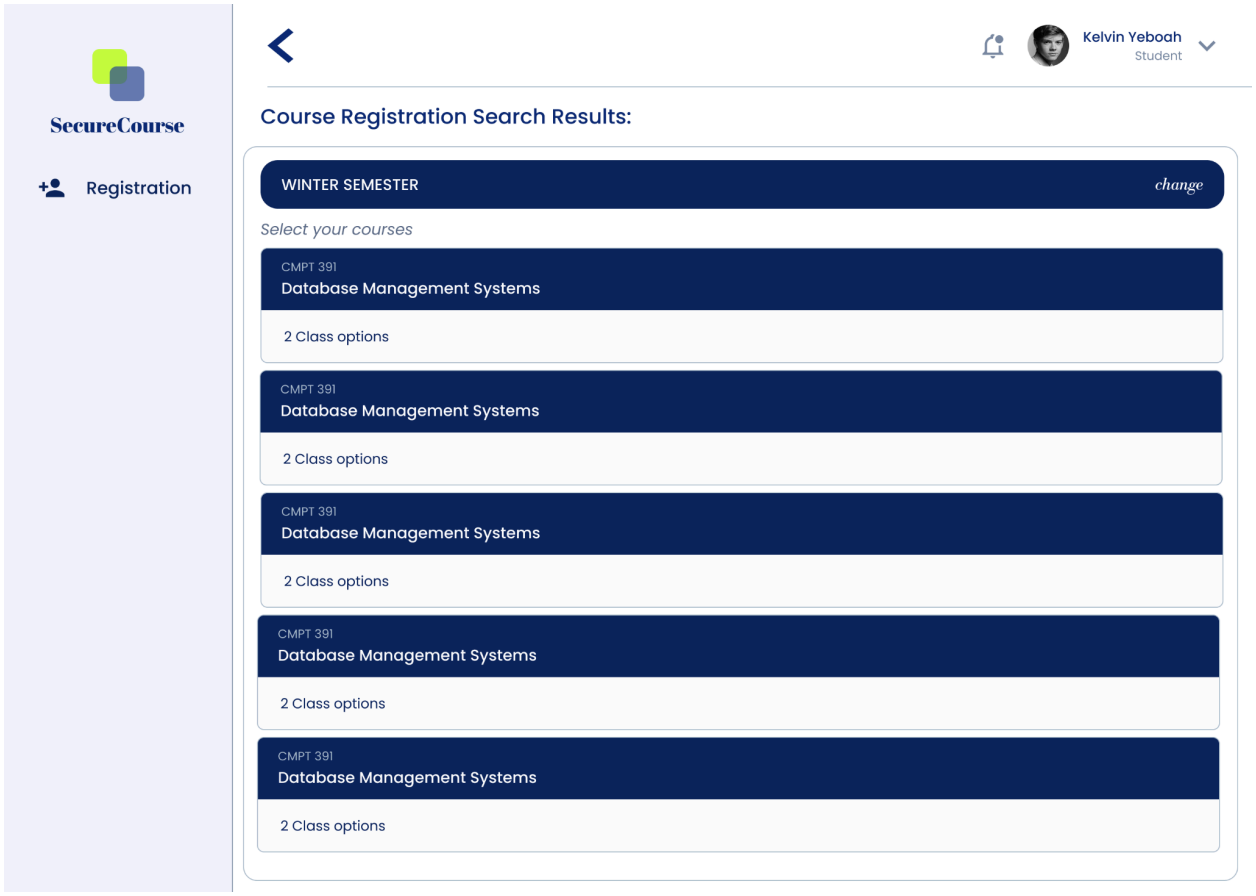
at least 8 characters

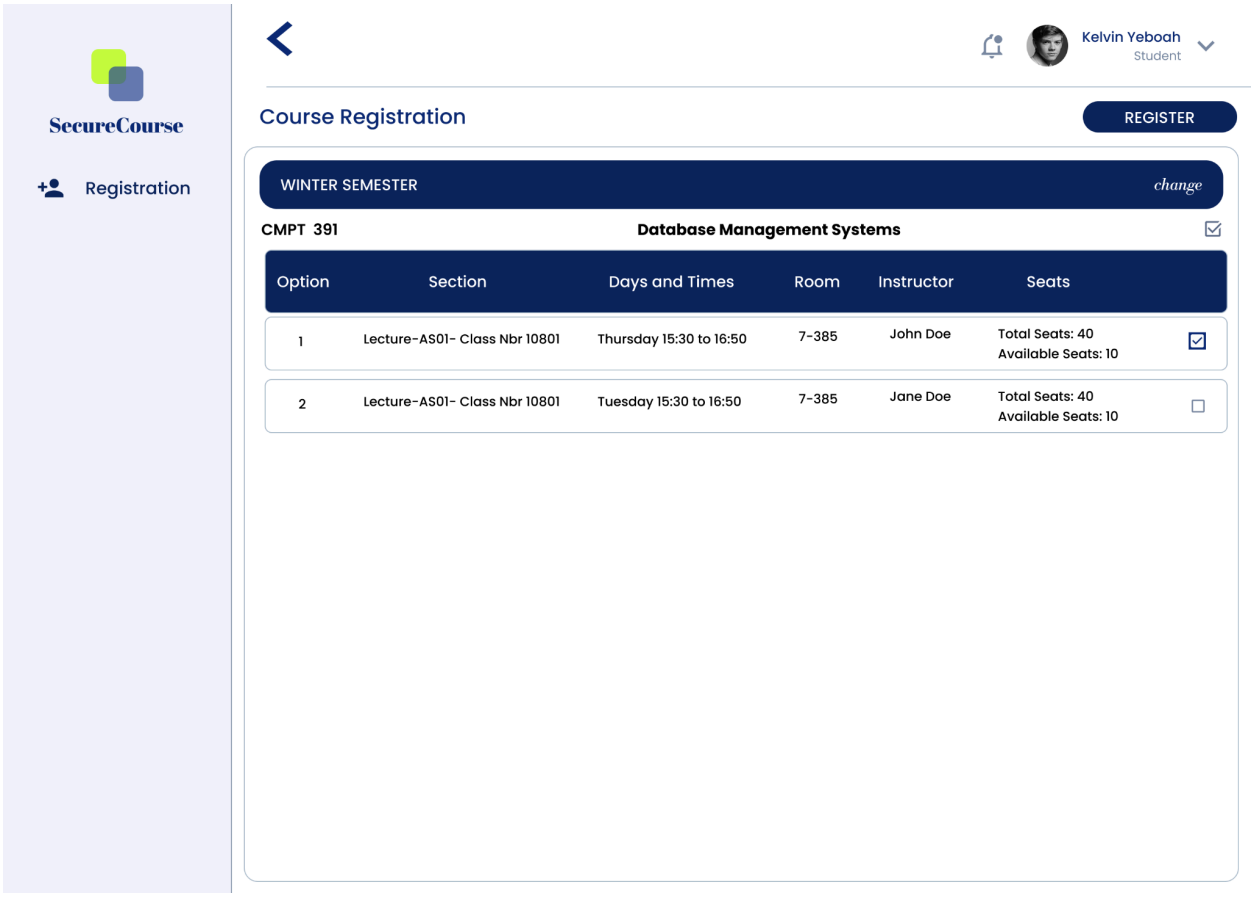
Forgot Password?

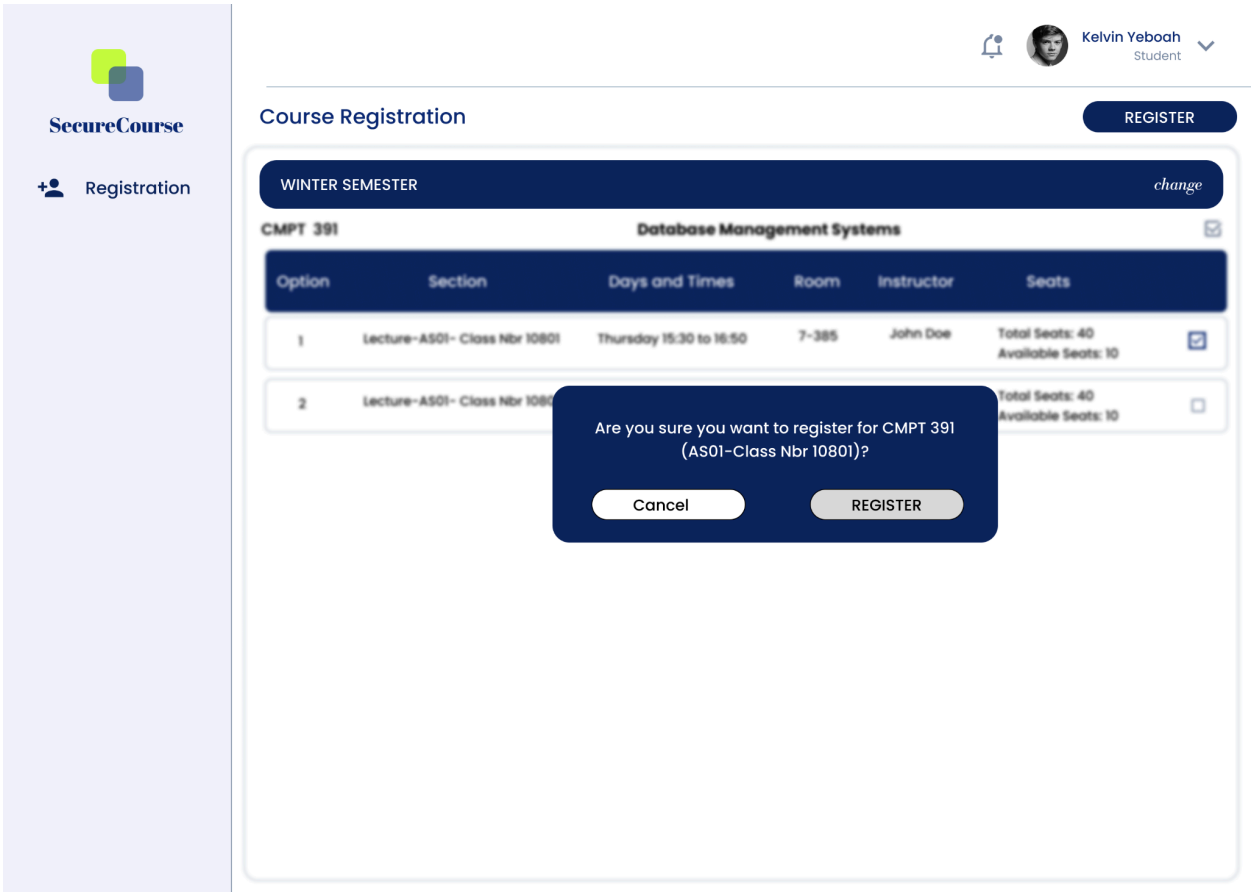
Sign in

Don't you have an account? Sign up




SecureCourse
Registration
Kelvin Yeboah
Student
Course Registration Search Results:
WINTER SEMESTER
change
Select your courses
CMPT 391
Database Management Systems
2 Class options
CMPT 391
Database Management Systems
2 Class options
CMPT 391
Database Management Systems
2 Class options
CMPT 391
Database Management Systems
2 Class options
CMPT 391
Database Management Systems
2 Class options


SecureCourse
Registration
Kelvin Yeboah
Student
Course Registration
REGISTER
WINTER SEMESTER
change
CMPT 391
Database Management Systems
Option
Section
Days and Times
Room
Instructor
Seats
1
Lecture-AS01- Class Nbr 10801
Thursday 15:30 to 16:50
7-385
John Doe
Total Seats: 40
Available Seats: 10
2
Lecture-AS01- Class Nbr 10801
Tuesday 15:30 to 16:50
7-385
Jane Doe
Total Seats: 40
Available Seats: 10


SecureCourse
Registration
Kelvin Yeboah
Student
Course Registration
REGISTER
WINTER SEMESTER
change
CMPT 391
Database Management Systems
Option
Section
Days and Times
Room
Instructor
Seats
1
Lecture-AS01- Class Nbr 10801
Thursday 15:30 to 16:50
7-385
John Doe
Total Seats: 40
Available Seats: 10
2
Total Seats: 40
Available Seats: 10
Are you sure you want to register for CMPT 391 (AS01-Class Nbr 10801)?
Cancel
REGISTER