STUDENT MANAGEMENT PROJECT KICK OFF DOCUMENT

Project Name: STUDENT MANAGEMENT WEB APPLICATION PROJECT

Definition: The Student-Management application is a web application which is able to manage education directly by using a web browser in your computer, phone, tablet, or mobile device. There will be different types of user in the Student-Management application. Users are able to login into application.

STUDENT MANAGEMENT WEB APPLICATION CUSTOMER REQUIREMENTS

1. Student-Management Application must be a web application.
2. The application must have user friendly UI. (Non functional requirement)
3. The application must be secure. (Non functional requirement)
4. Language of the application must be English. (Non functional requirement)
5. The application must display response of user request nearby for 5 seconds. (Non functional requirement)
6. Anonymous users who will make a process should be able to register this application with their own information
7. There should be able five types of roles for authorized user: Admin, Teacher, Student, Manager, Assistant Manager
8. Each user can only have one role (a user with the teacher role can also have the advisor teacher role at the same time)
9. Registered user should be able to login to the application after registration
10. Authenticated user should be able to update its own user information
11. Authenticated user should be able to update its own password
12. User who has the admin role should be able to get all users' information
13. User who has the Admin or Manager role should be able to get a user information
14. User who has the Admin or Manager or Assistant_Manager role should be able to update ateacher or student information
15. User who has the Admin role should be able to delete a user
16. User who has the Manager role should be able to delete a user whoes role is Teacher or Student or Assistant_Manager
17. User who has the Assistant_Manager role should be able to delete a user whoes role is Teacher or Student

18. User who has the Admin role should be able to update all user

19. User who has the Manager or Assistant_Manager role should be able to update a user whoes role is Teacher or Student

20. User who has the Manager or Assistant_Manager role should be able to get a user by his/her name

21. User who has the Manager or Assistant_Manager role should be able to add a lesson Program to a user whoes role is Teacher

22. User who has the Student role should be able to add a lesson Program to him/her

23. User who has the Admin or Manager or Assistant_Manager role should be able to get the status of user whoes role is Student.

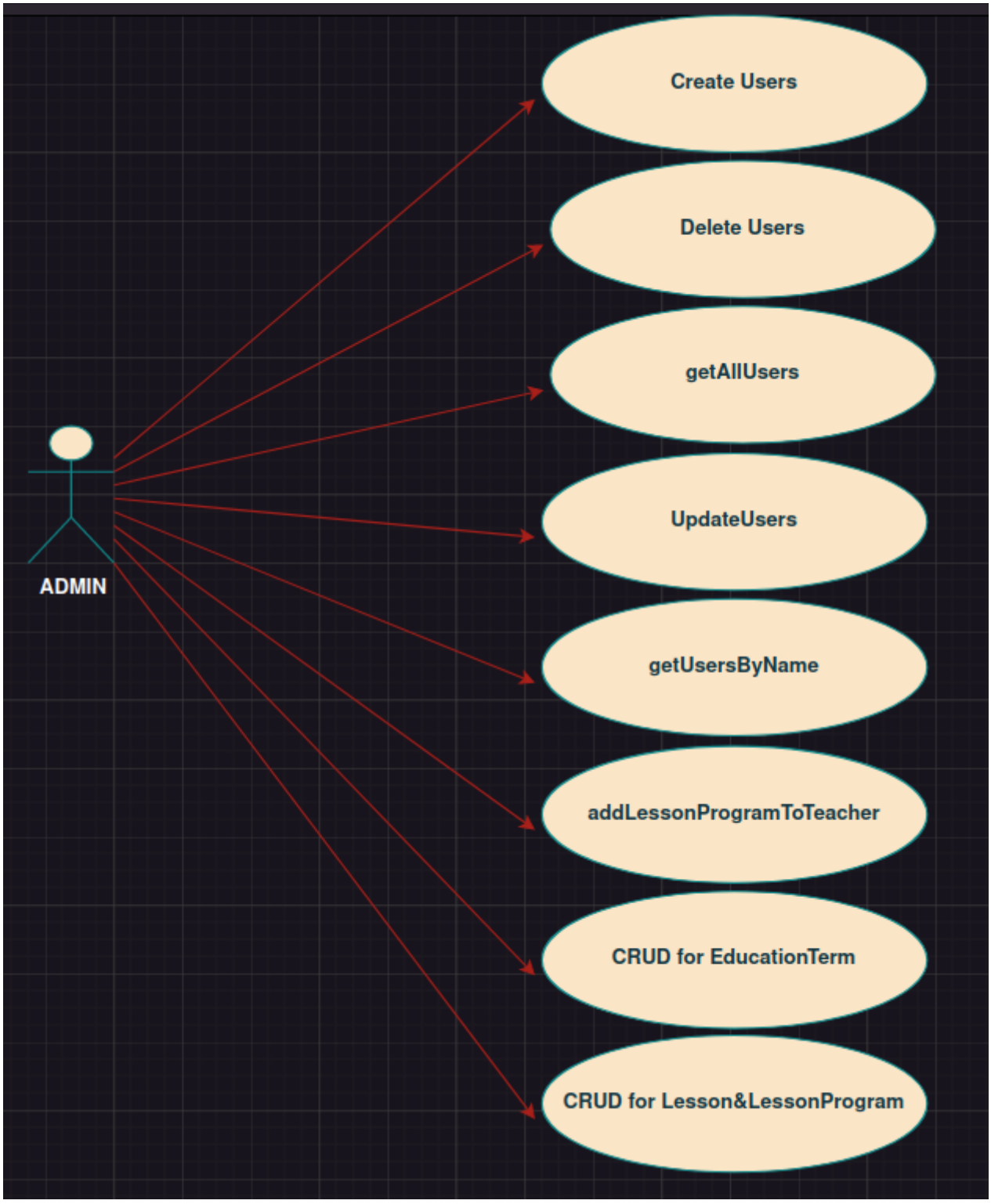
24. User who has the Admin or Teacher role should be able to get informations of his/her students

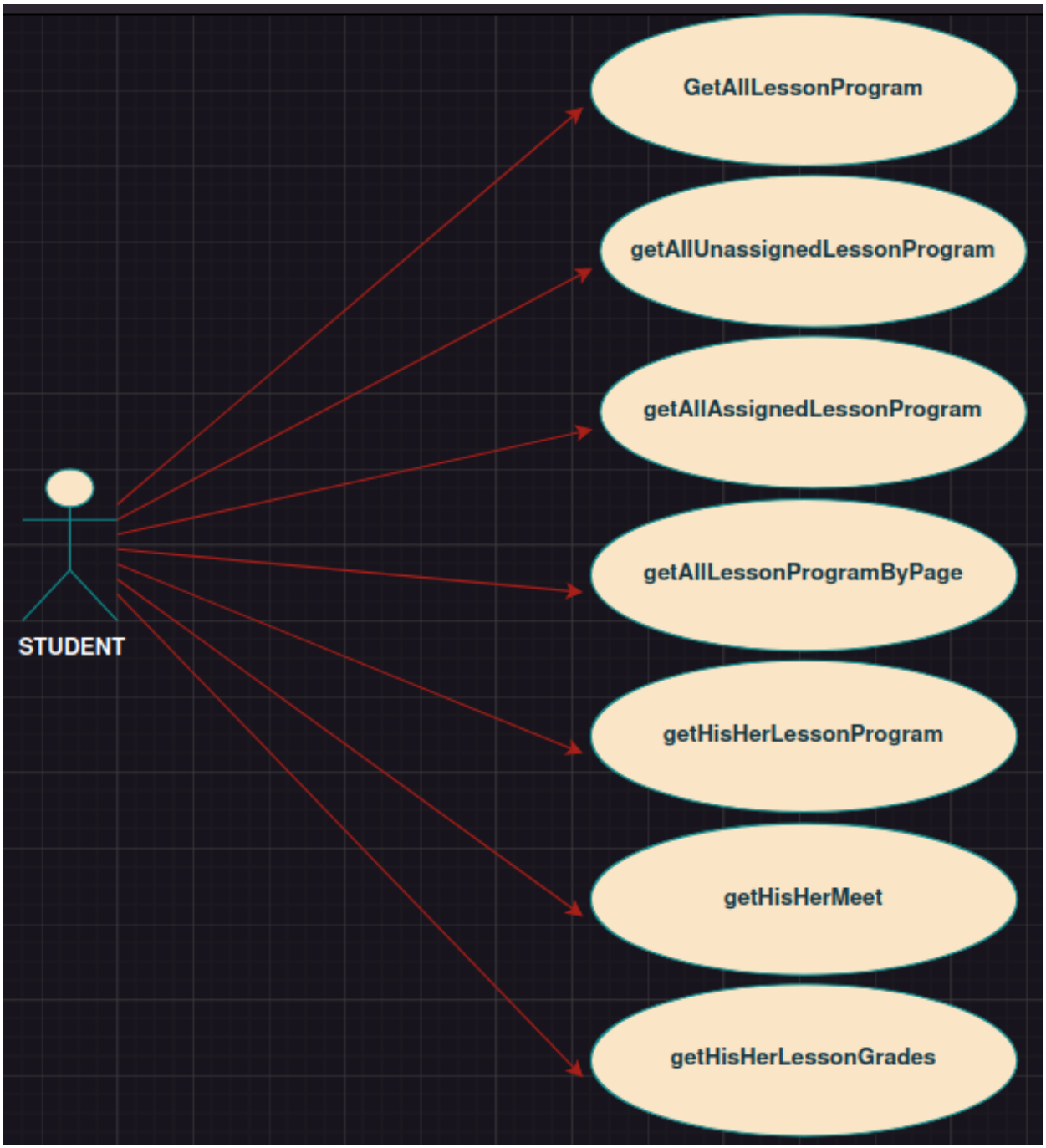
25. User who has the Admin or Manager or Assistant_Manager role should be able to save AdvisorTeacher

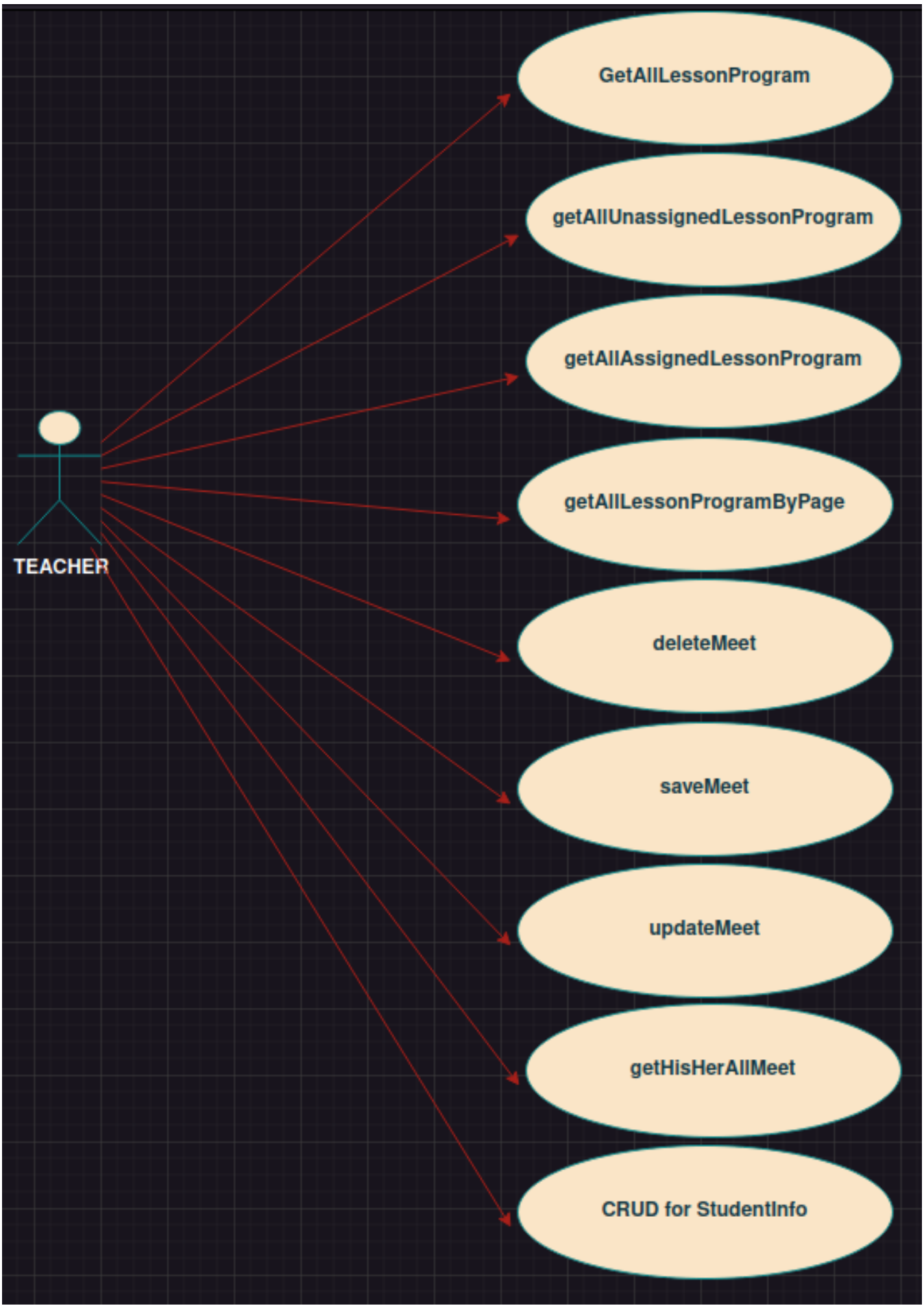
26. User who has the Admin or Manager or Assistant_Manager role should be able to get all AdvisorTeachers

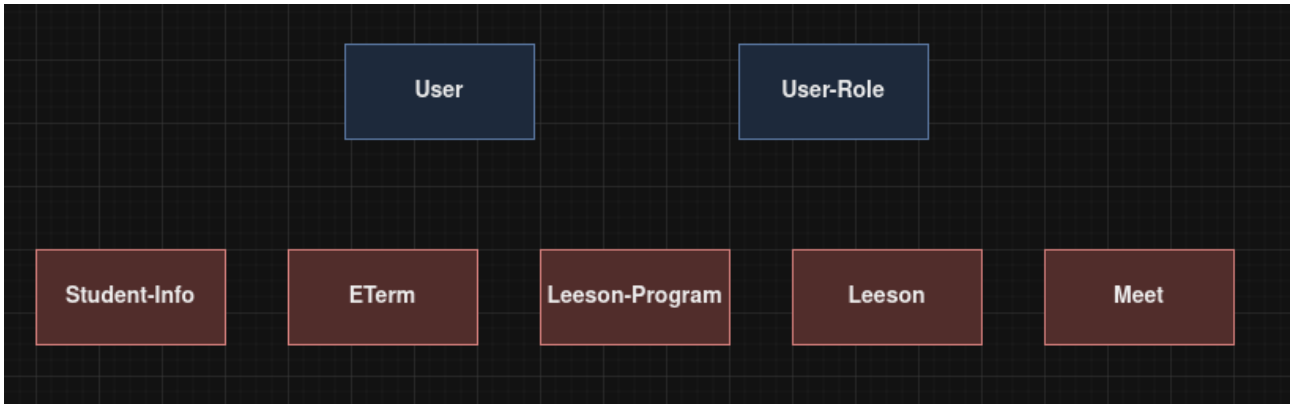
What is Use Case Diagram: In the Unified Modeling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. A UML use case diagram is the primary form of system/software requirements for a new software program underdeveloped. Use cases specify the expected behavior (what), and not the exact method of making it happen (how). Use cases once specified can be denoted both textual and visual representation (i.e. use case diagram). A key concept of use case modeling is that it helps us design a system from the end user's perspective. It is an effective technique for communicating system behavior in the user's terms by specifying all externally visible system behavior.

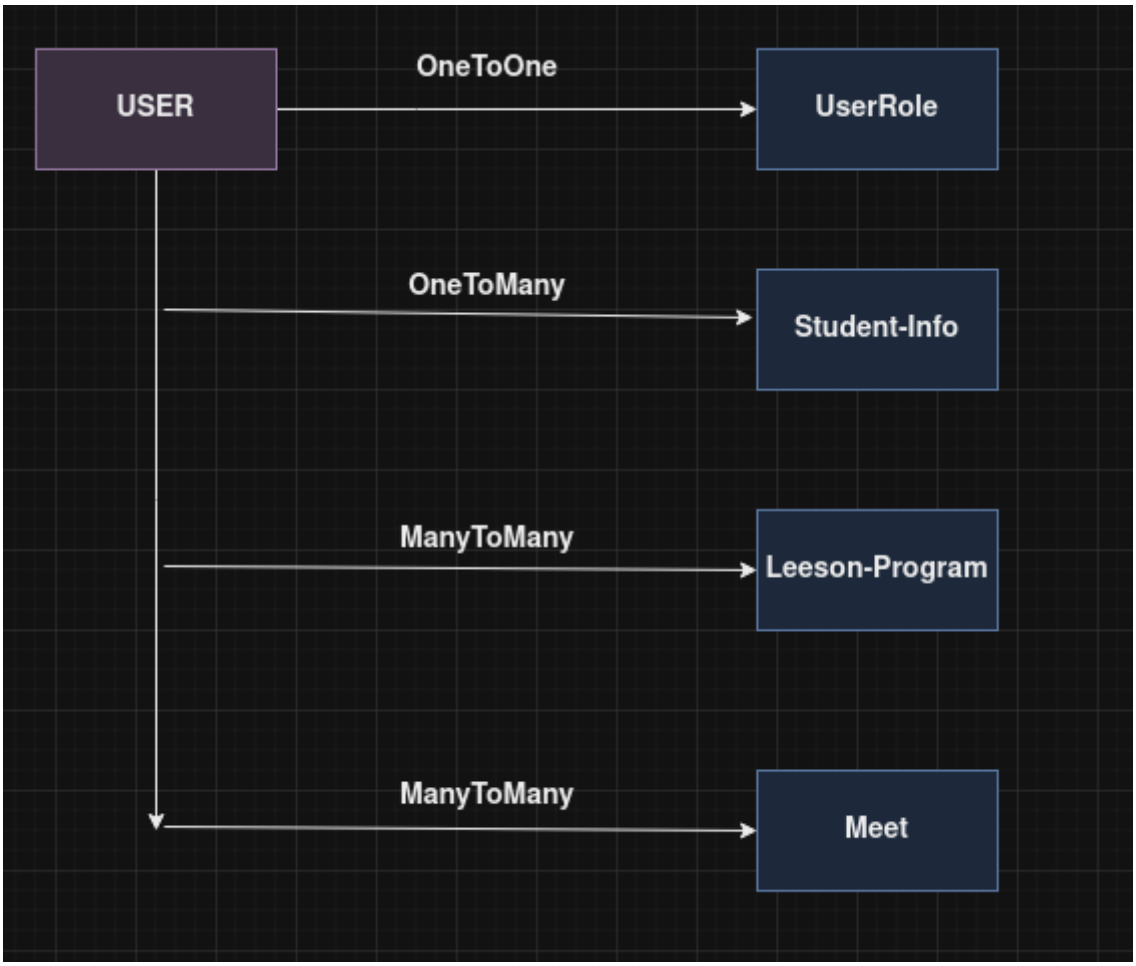
You will see usecase diagrams about StudentManagement project below :


Create Users
Delete Users
getAllUsers
UpdateUsers
getUsersByName
addLessonProgramToTeacher
CRUD for EducationTerm
CRUD for Lesson&LessonProgram
ADMIN

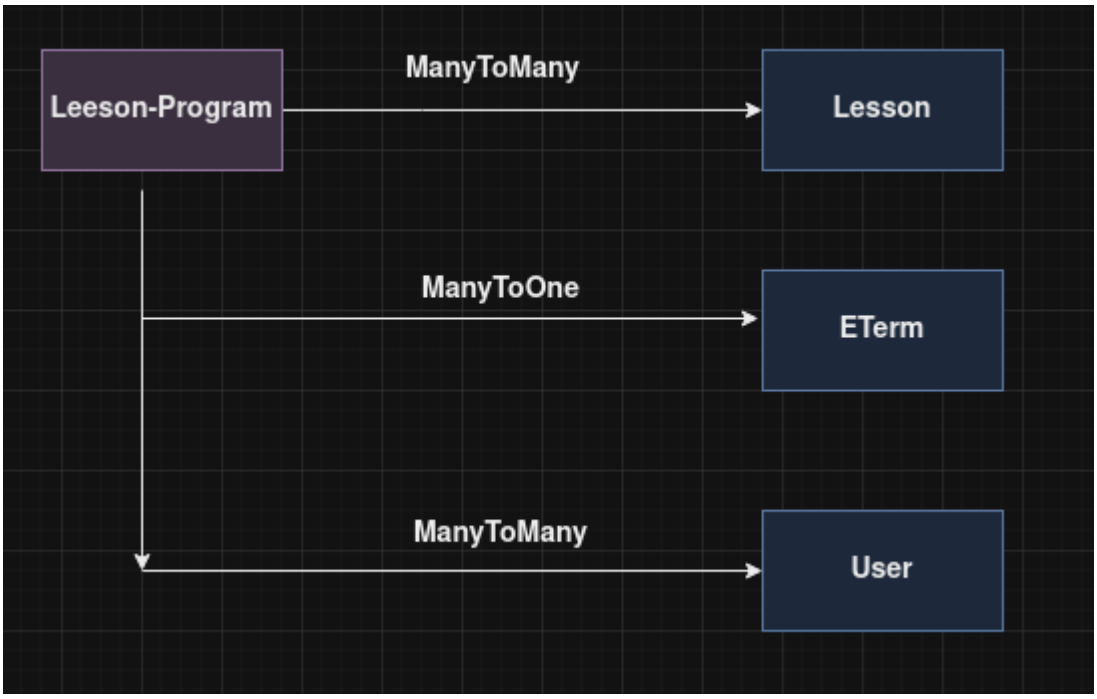

GetAllLessonProgram
getAllUnassignedLessonProgram
getAllAssignedLessonProgram
getAllLessonProgramByPage
getHisHerLessonProgram
getHisHerMeet
getHisHerLessonGrades
STUDENT

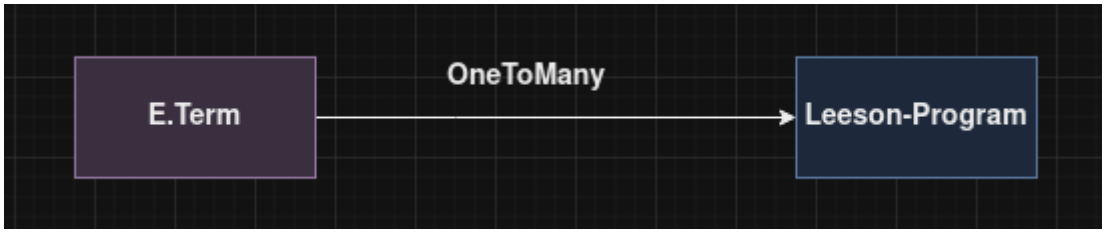

GetAllLessonProgram
getAllUnassignedLessonProgram
getAllAssignedLessonProgram
getAllLessonProgramByPage
deleteMeet
saveMeet
updateMeet
getHisHerAllMeet
CRUD for StudentInfo
TEACHER


User
User-Role
Student-Info
ETerm
Leeson-Program
Leeson
Meet


USER
OneToOne
UserRole
OneToMany
Student-Info
ManyToMany
Leeson-Program
ManyToMany
Meet


Lesson
ManyToMany
Leeson-Program

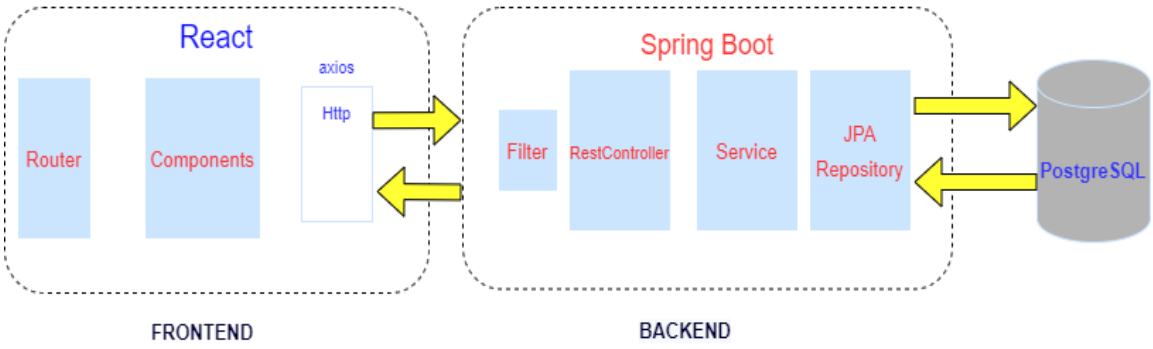

Leeson-Program
ManyToMany
Lesson
ManyToOne
ETerm
ManyToMany
User


E.Term
OneToMany
Leeson-Program


Meet
ManyToMany
User
ManyToOne
User

Frontend: The part of a web site or web application that the user interacts with directly. In this project, React Library will be used for frontend.

Backend: It is the server site the wep application. It stores and arranges data. It is the part of the web app that you can't see and interact with. In this project, Sprint boot framework will be used as a backend tecknology. Moreover PostgreSQL will be used as relational database.



Graphic4 General system structure of the application

DEVELOPMENT ENVIRONMENT, IDEs and TOOLS – INSTALLATIONS

1. FOR BACKEND

- Java 11 https://www.oracle.com/tr/java/technologies/javase/jdk11-archive-downloads.html
- Intellij IDE
- Postman https://www.postman.com/
- PostgreSQL 15 https://www.enterprisedb.com/downloads/postgres-postgresql-downloads
  Guide for installation POSTGRESQL: https://www.postgresqltutorial.com/install-postgresql/

THE TOPICS THAT YOU SHOULD REVIEW BEFORE STARTING PROJECT

FOR BACKEND (SPRING BOOT PROJECT)

1. Core Java: OOPS, classes, enums, interfaces, exception handling, collections, stream (foreach, filter, map), lambda,optional key word and other fundamentals.
2. Logging (SLF4J, Logback)
3. Regular expressions
4. What is Spring Framework.
5. What is Spring Boot Framework

6. What is Spring Security Framework. (JWT Based Security)
7. What is inversion of control , dependency injection and Spring IOC Container
8. What is JPA, Hibernate and Spring Data JPA
9. What is entity class and how to create it
10. What are OneToOne, OneToMany, ManyToOne, ManyToMany relations on hibernate.
11. How to use @JoinTable, @JoinColum annotations.
12. JPQL (Java Persistence Query Language), Basic SQL knowledge
13. What is REST API
14. How , why to use @Bean, @Autowired, @RestController, @Service, @Repository annotations.
15. Jackson annotations (@JsonIgnore, @JsonFormat etc.)
16. Controller-Service-Repository layered structure in spring boot app
17. What is @Transactional annotation in org.springframework.transaction.annotation
18. What is the Data Transfer Object and how to use it.
19. Usage of @ResponseEntity, @ResponseBody, @Request Body, @RequestMapping, @PostMapping, @GetMapping, @DeleteMapping, @PutMapping. @Valid annotations.
20. HTTP Response Status Codes. (200, 201, 400, 404, etc.)
21. Why and How to use @PathVariable and @QueryParam annotations
22. Project Lombok

GOOD

LUCK

WITH THAT