



Object Oriented Database Mini-Project

College Enrollment System

Participants:

- Ahmed Osama Mohamed Mohamed (Case)
- Abd El Rahman Hafez Abd El Rahman (Case)

Project Supervisors:

- Prof. Dr. Ahmed Saeed
- T.A: Aya Saber
- T.A: Mohamed Safwat

GitHub:

<https://github.com/Ahmed9Osama/College-OODB-Mini-Project>

Object-Oriented Database Project Documentation

1. Project Title

College Enrollment System Object-Oriented Database Project Using ObjectDB and Java

2. Project Overview

This project demonstrates an Object-Oriented Database Management System (OODBMS) using ObjectDB and Java. The system models a college database with three main entities:

- Student
- Course
- Enrollment (to manage the many-to-many relationship between students and courses).

The project highlights core functionalities such as CRUD operations (Create, Read, Update, Delete), entity relationships, and object querying.

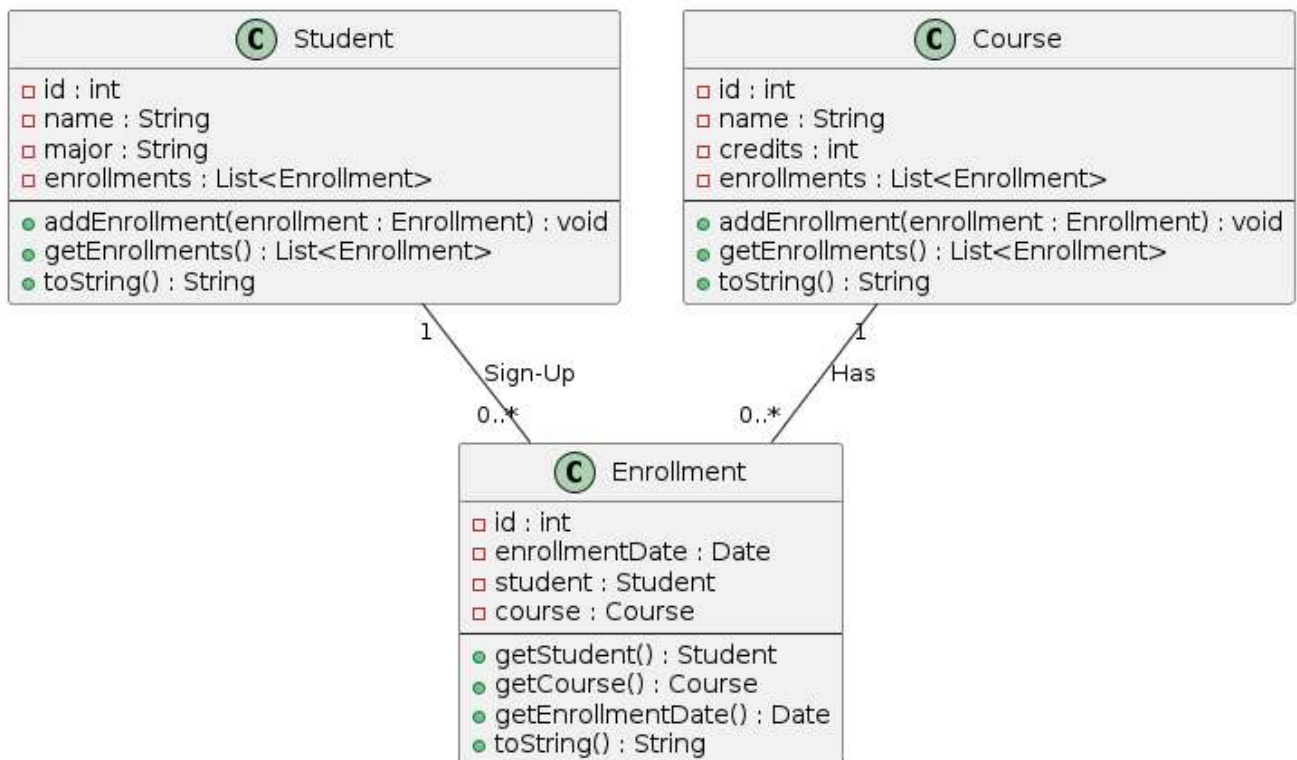
3. Objectives

1. Implement object-oriented database concepts.
2. Perform CRUD operations using JPA (Java Persistence API).
3. Manage entity relationships (One-to-Many, Many-to-One).
4. Execute queries using JPQL (Java Persistence Query Language).

4. Tools & Technologies Used

- Java: Programming language.
- ObjectDB: Object-oriented database management system.
- JPA (Java Persistence API): Persistence management for objects.
- Eclipse IDE: Development environment.
- PlantUML: For class diagram generation.

5. UML Class Diagram



@startuml

' Define the Student class

```
class Student {  
    - id : int  
    - name : String  
    - major : String  
    - enrollments : List<Enrollment>  
    --  
    + addEnrollment(enrollment : Enrollment) : void  
    + getEnrollments() : List<Enrollment>  
    + toString() : String  
}
```

' Define the Course class

```
class Course {  
    - id : int  
    - name : String  
    - credits : int  
    - enrollments : List<Enrollment>  
    --  
    + addEnrollment(enrollment : Enrollment) : void  
    + getEnrollments() : List<Enrollment>  
    + toString() : String  
}
```

' Define the Enrollment class

```
class Enrollment {  
    - id : int  
    - enrollmentDate : Date  
    - student : Student  
    - course : Course  
    --  
    + getStudent() : Student  
    + getCourse() : Course  
    + getEnrollmentDate() : Date  
    + toString() : String  
}
```

' Relationships

Student "1" -- "0..*" Enrollment : "Sign-Up"

Course "1" -- "0..*" Enrollment : "Has"

@enduml

6. System Design

1. Student Class

Attributes:

- id : int (Primary Key)
- name : String
- major : String

Relationships:

- One-to-Many with Enrollment (A student can have multiple enrollments).

Methods:

- addEnrollment(Enrollment enrollment)
- getEnrollments()
- toString()

2. Course Class

Attributes:

- id : int (Primary Key)
- name : String
- credits : int

Relationships:

- One-to-Many with Enrollment (A course can have multiple enrollments).

Methods:

- addEnrollment(Enrollment enrollment)
- getEnrollments()
- toString()

3. Enrollment Class

Attributes:

- id : int (Primary Key)
- enrollmentDate : Date
- student : Student (Many-to-One relationship)
- course : Course (Many-to-One relationship)

Methods:

- toString()

7. System Features

1. Data Persistence: The project uses ObjectDB and JPA for data persistence.

2. CRUD Operations:

- Create: Add students, courses, and enrollments.
- Read: Fetch and display data.
- Update: Modify existing entities.
- Delete: Remove entities.

3. Queries:

- Retrieve all students and their enrollments.
- Aggregate query to count total courses.

8. Code Structure

1. Student.java: Defines the Student class.
2. Course.java: Defines the Course class.
3. Enrollment.java: Defines the Enrollment class.
4. MainApp.java: Main application demonstrating CRUD operations.
5. persistence.xml: Configuration file for ObjectDB.

9. How to Run the Project

1. Setup Environment:
 - Install Java JDK.
 - Add objectdb.jar to the project classpath.
2. Run MainApp.java:
 - Compile and run the main class.
3. Expected Output:

The system displays students, courses, enrollments, and total course count.

10. Conclusion

This project successfully demonstrates the use of an object-oriented database system with JPA and ObjectDB. It highlights entity relationships, CRUD operations, and querying techniques.