Part 1:  
Q1:

| **Entity** |  | **Key Attributes** |
| --- | --- | --- |
| **Student** |  | student\_id (PK), name, gender, dob, email, phone, address |
| **Subject** |  | subject\_id (PK), subject\_name, credit, description |
| **Class** |  | class\_id (PK), class\_name, subject\_id (FK), semester, year |
| **Enrollment** |  | enrollment\_id (PK), student\_id (FK), class\_id (FK), registration\_date |
| **Attendance** |  | attendance\_id (PK), enrollment\_id (FK), attendance\_date, status |
| **Grade** |  | grade\_id (PK), enrollment\_id (FK), score |
| **Notification** |  | notification\_id (PK), student\_id (FK), content, date |

Q2:  
A black background with white text

AI-generated content may be incorrect.

Q3:

A black background with white text

AI-generated content may be incorrect.

Q4:

Not Ennrolled -> Enrlled -> Attending -> Completed-> Graded-> Notified

A screenshot of a phone

AI-generated content may be incorrect.

Part 2:

Q1:

| **Layer** | **Class / Table Name** | **Description** |
| --- | --- | --- |
| **Client Side** | RegisterSubjectComponent | UI component that allows a student to view available classes and register. |
|  |  | Sends a POST request to the server with the selected class. |
| **Server Side** | EnrollmentController | Handles HTTP POST requests for registering a student into a class. |
| *(Controller)* |  | Receives data from the client and forwards it to the service layer. |
| **Server Side** | EnrollmentService | Contains business logic for registration: |
| *(Service)* |  | - Checks if student already registered. |
|  |  | - Validates class availability. |
|  |  | - Creates new enrollment if valid. |
| **Server Side** | EnrollmentRepository | Interface for database operations related to the Enrollment entity. |
| *(Repository)* |  | Uses ORM (e.g., JPA) to save/retrieve enrollments. |
| **Database** | Student | Stores student information: student\_id, name, email. |
|  | Subject | Stores subject details: subject\_id, subject\_name, credit. |
|  | Class | Represents a class of a subject: class\_id, subject\_id, semester, year. |
|  | Enrollment | **Join table** for students and classes: |
|  |  | enrollment\_id (PK), student\_id (FK), class\_id (FK), registration\_date. |

Q2:

A screenshot of a computer

AI-generated content may be incorrect.  
Part 3:

Q1:

Pattern Name: Singleton

Pattern Family: Creational

Q2:

class Singleton:

private static instance = null

private constructor():

if instance == null:

instance = new Singleton()

return instance

Q3:

In a large application, you only want one global configuration object or one database connection manager. Creating multiple instances may lead to inconsistent settings or too many database connections.

Singleton ensures that only one shared instance exists during runtime.