# EXPERIMENT NO. 04 Class Diagram for the project.

**Learning Objective:** Sketch Class Diagram for the project.

**AIM:** To Class Diagram for the project.

**Tools:** Draw.io, word, Canva,miro

# Theory:

A class diagram, a fundamental tool in software engineering, provides a graphical depiction of the architecture of a software system, illustrating the structure and interrelationships between classes. Each class within the diagram encapsulates both data attributes and behaviors that collectively define the functionality of the system. Attributes represent the characteristics or properties of objects belonging to a particular class, while methods define the operations that can be performed on these objects. These elements work together to form a blueprint of the system's design, enabling developers to conceptualize and communicate complex software structures effectively.

In addition to representing individual classes, class diagrams also depict the relationships between them using various types of associations. Associations signify connections and interactions between classes, while aggregations denote "whole-part" relationships, illustrating how one class encompasses or contains another. Furthermore, generalizations in class diagrams indicate inheritance hierarchies, showcasing how subclasses inherit attributes and behaviors from their parent classes. By adhering to key design principles such as abstraction, modularity, and information hiding, developers can ensure that class diagrams promote clarity, maintainability, and scalability in software development projects.

1. **Project**: Contains attributes like ID, name, description, start/end dates, and status. Methods handle project duration calculation and status updates.
2. **Task**: Attributes include ID, name, description, start/end dates, status, priority, and assigned user. Methods manage task status, priority, and user assignment.
3. **User**: Stores user details such as ID, username, password, email, and role. Methods handle authentication and role management.
4. **Team**: Holds team information like ID, name, and description. Methods manage team members.
5. **Comment**: Stores comments with attributes ID, text, creator, and creation date. Methods handle adding comments to tasks/projects.
6. **Attachment**: Manages file attachments with attributes ID, filename, type, size, uploader, and upload date. Methods handle upload/download.
7. **Dependency**: Represents task dependencies with attributes ID, dependent task ID, and dependency task ID. Methods manage task dependencies.
8. **Notification**: Sends notifications with attributes ID, message, recipient, sender, and timestamp. Methods handle notification sending.
9. **Managers (ProjectManager, TaskManager, UserManager, TeamManager)**: Manage corresponding entities (projects, tasks, users, teams) with methods for CRUD operations and entity-specific tasks (e.g., task assignment, user authentication).

**Result and Discussion:**

The class diagram for the project management system outlines essential entities and relationships, facilitating project, task, user, and team management. It incorporates attributes and methods for effective system operation. The diagram offers a clear overview of the system's architecture, aiding communication and guiding development. Managers for projects, tasks, users, and teams promote modular design, enhancing scalability and maintainability. Adherence to design principles ensures flexibility and robustness. Further refinement may optimize performance and address specific requirements, making the diagram a valuable tool for system implementation.

Top of Form

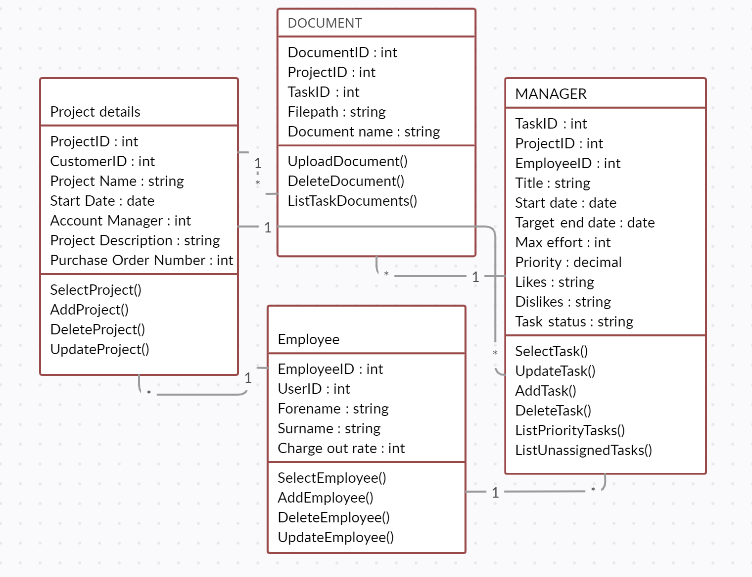
**Learning Outcomes:** The student should have the ability to

LO4.1: The class diagram for our project management system provides a visual representation of the system's structure, illustrating the entities involved such as projects, tasks, users, and teams.

LO4.2 The class diagram serves as a communication tool, improving collaboration among project stakeholders by providing a common visual language to discuss and understand the project management system.

LO4.3 As a comprehensive design documentation, the class diagram guides software development by detailing the structure, behavior, and interactions of the project management system.**Course Outcomes:** Evaluate techniques to Draw Data Flow Diagram problem.

**Design**:



**Conclusion:** the class diagram for the project management system provides a clear visualization of the system's structure, behavior, and interactions. It serves as a valuable tool for guiding software development through comprehensive design documentation and improving communication among project stakeholders. By depicting entities such as projects, tasks, users, and teams, along with their relationships, the diagram facilitates a deeper understanding of the system's functionality and promotes effective collaboration throughout the development process.

**For Faculty Use:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Correction Parameters** | **Formative Assessment [40%]** | **Timely completion of Practical [ 40%]** | **Attendance / Learning Attitude [20%]** |  |
| **Marks Obtained** |  |  |  |