DBMS ASSIGNMENT:

“EFFECTIVE TIME MANAGEMENT FOR PROJECT”

**By: Joel C Raju**

# DOMAIN DESCRIPTION:

Effective time management is crucial for the success of any project. It involves planning, organizing, and allocating resources in a way that optimizes productivity, meets deadlines, and achieves project goals. Here are key principles and practices for effective time management in projects:

Time management in project management is essential to dividing resources effectively to ensure teams produce high-quality work within the agreed-upon deadline.

# BACKGROUND STUDY:

To understand the “Effective time management for project” domain, I started to read about the domain and what it’s trying to achieve. Also learnt how the relevance of the domain during the development process.

To learn more about the domain, I tried out Nulab and Kanban Board which are time management tools that help to organize and prioritize tasks to achieve the team goal faster. It also increases productivity among the team by reducing the confusion among team members regarding the tasks assigned.

**PROJECT WORKFLOW:**

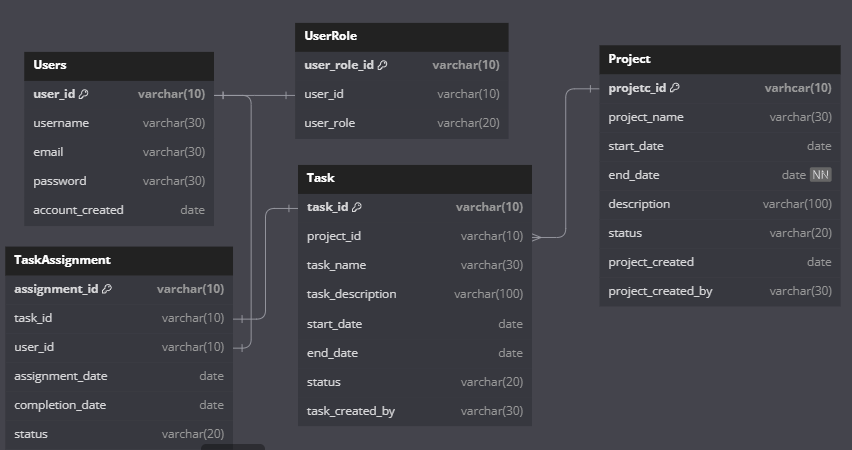
From the background study, A team is allocated a board in which there will be tasks written in cards. The Team members will be allocated the tasks along with deadline and must set the task status according to their progress. This way the team will work efficiently while keeping track of everyone’s task.

# REQUIREMENT ANALYSIS:

The requirement analysis involves understanding and delineating the needs, objectives, and functionalities related to effective time management for projects. Requirement analysis for the “Effective time management for projects” domain are:

* **Task Allocation Mechanism:** Allocating Tasks to Teams: This reveals a specific requirement for a system that allows easy allocation and tracking of tasks within a team.
* **Task Status Tracking:** This indicates a requirement for a system that supports real-time task status updates and tracking.
* **Efficient Team Collaboration:** Reducing Confusion, this points to a requirement for collaborative tools that enhance team communication and coordination.
* **Deadline Setting:** This allows team to follow a timeline and complete the work before reaching the deadline.

# E-R DIAGRAM:



# DETAILED EXPLAINATION:

* **Project Table (Project):**
  + **projetc\_id**: Primary key, VARCHAR(10), not null - Unique identifier for the project.
  + **project\_name**: VARCHAR(30), not null - Name of the project.
  + **start\_date**: DATE, not null - The start date of the project.
  + **end\_date**: DATE - The end date of the project.
  + **description**: VARCHAR(100) - Description of the project.
  + **status**: VARCHAR(20) - Status of the project.
  + **project\_created**: DATE, not null - Date when the project was created.
  + **project\_created\_by**: VARCHAR(30), not null - User who created the project.
* **Task Table (Task):**
  + **task\_id**: Primary key, VARCHAR(10), not null - Unique identifier for the task.
  + **project\_id**: Foreign key referencing Project.project\_id, VARCHAR(10), not null - Identifier of the project to which the task belongs.
  + **task\_name**: VARCHAR(30), not null - Name of the task.
  + **task\_description**: VARCHAR(100) - Description of the task.
  + **start\_date**: DATE, not null - The start date of the task.
  + **end\_date**: DATE - The end date of the task.
  + **status**: VARCHAR(20) - Status of the task.
  + **task\_created\_by**: VARCHAR(30), not null - User who created the task.
* **Users Table (Users):**
  + **user\_id**: Primary key, VARCHAR(10), not null - Unique identifier for the user.
  + **username**: VARCHAR(30), not null - User's username.
  + **email**: VARCHAR(30), not null - User's email address.
  + **password**: VARCHAR(30), not null - User's password.
  + **account\_created**: DATE, not null - Date when the user account was created.
* **TaskAssignment Table (TaskAssignment):**
  + **assignment\_id**: Primary key, VARCHAR(10), not null - Unique identifier for the task assignment.
  + **task\_id**: Foreign key referencing Task.task\_id, VARCHAR(10), not null - Identifier of the task being assigned.
  + **user\_id**: Foreign key referencing Users.user\_id, VARCHAR(10), not null - Identifier of the user to whom the task is assigned.
  + **assignment\_date**: DATE, not null - Date when the task assignment was created.
  + **completion\_date**: DATE - Date when the task was completed.
  + **status**: VARCHAR(20), not null - Status of the task assignment.
* **UserRole Table (UserRole):**
  + **user\_role\_id**: Primary key, VARCHAR(10), not null - Unique identifier for the user role.
  + **user\_id**: Foreign key referencing Users.user\_id, VARCHAR(10), not null - Identifier of the user associated with the role.
  + **user\_role**: VARCHAR(20), not null - Role assigned to the user.

# NORMALIZATION PROCESS:

### Normalization is a database design process that aims to minimize data redundancy and dependency by organizing tables in a way that reduces or eliminates data anomalies. The standard normal forms are First Normal Form (1NF), Second Normal Form (2NF), and Third Normal Form (3NF).

### 

### Let's analyze how the provided tables align with normalization principles:

### **First Normal Form (1NF):**

* All tables have atomic (indivisible) values in each field.
* There are no repeating groups or arrays in any field.

**Second Normal Form (2NF):**

* In the **Task** table:
  + All non-key attributes are fully functionally dependent on the primary key (**task\_id**).
  + The **project\_id** is a foreign key, and other attributes are dependent on the entire primary key.
* In the **TaskAssignment** table:
  + All non-key attributes are fully functionally dependent on the primary key (**assignment\_id**).
  + Both **task\_id** and **user\_id** are foreign keys, and other attributes are dependent on the entire primary key.
* In the **UserRole** table:
  + All non-key attributes are fully functionally dependent on the primary key (**user\_role\_id**).
  + The **user\_id** is a foreign key, and other attributes are dependent on the entire primary key.

**Third Normal Form (3NF):**

* In the **Project** table:
  + The **project\_created\_by** field is dependent only on the primary key (**projetc\_id**).
  + There are no transitive dependencies.
* In the **Task** table:
  + The **task\_created\_by** field is dependent only on the primary key (**task\_id**).
  + There are no transitive dependencies.
* In the **TaskAssignment** table:
  + The **status** field is dependent only on the primary key (**assignment\_id**).
  + There are no transitive dependencies.
* In the **Users** table:
  + No transitive dependencies; all non-key attributes depend directly on the primary key (**user\_id**).
* In the **UserRole** table:
  + The **user\_role** field is dependent only on the primary key (**user\_role\_id**).
  + There are no transitive dependencies.

Overall, the provided tables appear to be in at least 3NF, as there are no transitive dependencies, and each non-key attribute is dependent on the whole primary key. This structure helps in reducing redundancy and maintaining data integrity.