



## Homework 6

**PEPE (Project Elaborating Program with Extension)  
Proposal**

**Software Engineering Principle  
Software Engineering Program,  
Department of Computer Engineering,  
School of Engineering, KMITL**

67011235 Paphavee Yanmook  
67011287 Ramida Laphasphokin  
67011352 Theepakorn Phayonrat

# PEPE (Project Elaborating Program with Extension)

## Description

PEPE (Project Elaborating Program with Extension) is a web-based collaborative project management system designed to support software development teams throughout the entire Software Development Life Cycle (SDLC). It provides an integrated environment that enables teams to plan, organize, track, and manage software projects in a structured, transparent, and efficient manner.

The system is specifically targeted at students, academic project groups, and small software development teams who require a lightweight yet powerful tool to manage their projects without the complexity of enterprise-level platforms.

## Objectives

PEPE aims to:

- Improve team collaboration and communication.
- Provide a centralized platform for project planning and documentation.
- Enable efficient task tracking, progress monitoring, and issue management.
- Assist teams in meeting deadlines through proper scheduling and milestone control.
- Provide insights into team productivity through basic analytics and reporting.

# Key Features

## 1. Project Planning

- Create and manage multiple projects.
- Define project scope, goals, milestones, and deadlines.
- Visualize timelines using Gantt charts.

## 2. Task Management

- Break projects into tasks and subtasks.
- Assign tasks to team members.
- Set priorities, deadlines, and dependencies.
- Track task status: To Do, In Progress, Testing, Completed.

## 3. Collaboration & Communication

- Built-in comment system for tasks.
- Real-time notifications for task updates and deadlines.
- Activity feed to monitor project progress.

## 4. Reporting & Analytics

- Progress reports per project or per member.

## 5. User & Role Management

- Roles: Admin, Project Manager, Team Member.
- Access control based on roles.

## 6. Issue & Bug Tracking (Optional)

- Report bugs or issues via tickets.
- Assign severity levels.
- Track resolution status.

## 7. Visual Studio Code Integration with Extension (Optional)

- Interpret **TASK** from comment in code.
- Assign **TASK** to **@USER** or **@ROLE** via external task specification file in Markdown format.
- Interpret **TASK** from comment in code.

## Benefits

- Reduces dependency on scattered tools like spreadsheets and chat apps.
- Improves project transparency and accountability.
- Enhances learning experience for students in team-based projects.
- Saves time by automating routine project management activities.

## Target Users

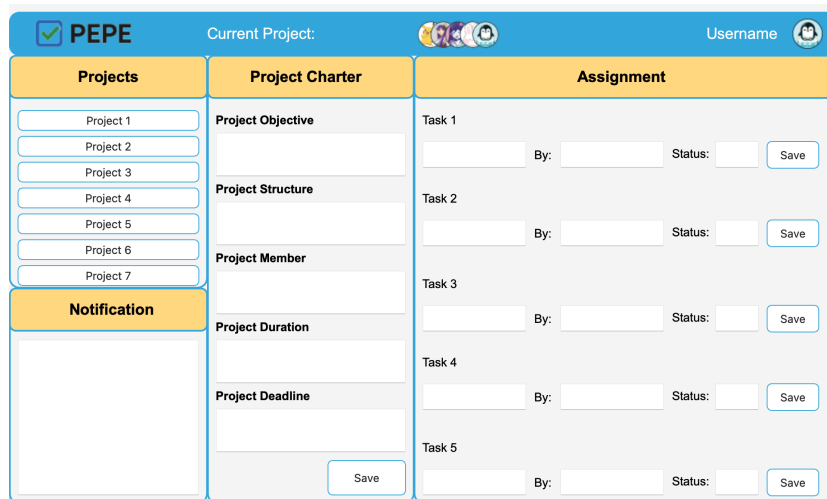
- University students working on capstone or group projects.
- Small development teams and startups.
- Academic instructors supervising multiple student teams.

## Conclusion

PEPE serves as a simple yet comprehensive project management solution that bridges the gap between theory and practice in software engineering. By providing essential planning, tracking, and collaboration features in one unified platform, it empowers teams to deliver projects on time with higher quality and better coordination.

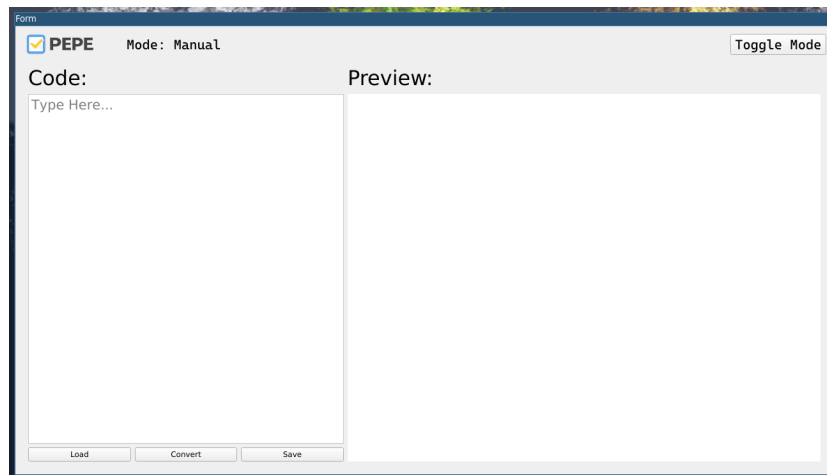
## Demonstration UI:

### Main Page



The screenshot shows the main interface of the PEPE application. It features a blue header bar with the PEPE logo, a 'Current Project:' label, a row of five circular icons, and a 'Username' label with a user profile icon. Below the header, the interface is divided into three main sections: 'Projects', 'Project Charter', and 'Assignment'. The 'Projects' section on the left contains a list of seven projects (Project 1 to Project 7) and a 'Notification' section below it. The 'Project Charter' section in the middle contains four input fields labeled 'Project Objective', 'Project Structure', 'Project Member', and 'Project Duration', followed by a 'Project Deadline' field and a 'Save' button. The 'Assignment' section on the right contains five task entries (Task 1 to Task 5), each with a 'By:' field, a 'Status:' field, and a 'Save' button.

### Task Assignment Page



The screenshot shows the 'Task Assignment Page' in the PEPE application. It features a blue header bar with the PEPE logo, a 'Mode: Manual' label, and a 'Toggle Mode' button. Below the header, the interface is divided into two main sections: 'Code:' and 'Preview:'. The 'Code:' section on the left contains a large text area with the placeholder text 'Type Here...'. The 'Preview:' section on the right is a large empty area. At the bottom of the interface, there are three buttons: 'Load', 'Convert', and 'Save'.

Form

☒ **PEPE** Mode: Sync Toggle Mode

Preview:

Form

☒ **PEPE** Mode: Manual Toggle Mode

**Code:**

How type of language (We tell them step by step what we do to get the result.)  
 Relational Algebra is a Procedural Language which can tell how do we do step by step to get the result.

### Relational Calculus

- Domain Relational Calculus
- Query By Example (QBE) (created by IBM) is based on this.
- Tuple Relational Calculus
- Structured English Query Language (SEQUEL) (created by IBM) is based on this. (Later changed to SQL (Structured Query Language))
- QUEL (created by UCB) which was ran on Ingres later to be ancestor of Postgres.

> [!IMPORTANT]  
 > For a DBMS to be called as a relational

**Preview:**

**Data Manipulation**

What type of language (We tell the language what we want.) Relational Calculus is a Non-Procedural Language which can tell the result we want. How type of language (We tell them step by step what we do to get the result.) Relational Algebra is a Procedural Language which can tell how do we do step by step to get the result.

**Relational Calculus**

- Domain Relational Calculus
  - Query By Example (QBE) (created by IBM) is based on this.
- Tuple Relational Calculus
  - Structured English Query Language (SEQUEL) (created by IBM) is based on this. (Later changed to SQL (Structured Query Language))
  - QUEL (created by UCB) which was ran on Ingres later to be ancestor of Postgres.

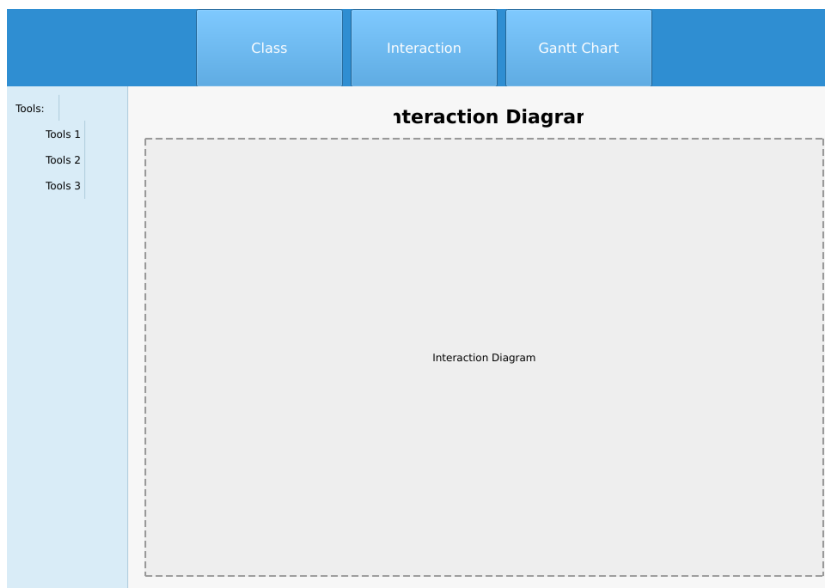
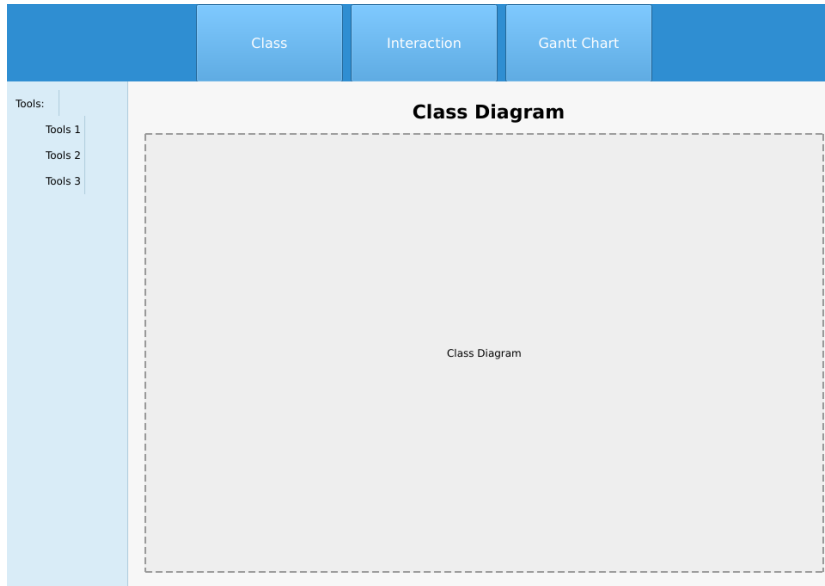
[!IMPORTANT]  
 For a DBMS to be called as a relational DB, they must: - Support Relation as Data Structure - Support Primary Key and Foreign Key - Support Relational Algebra and Relational Calculus (AKA SQL)

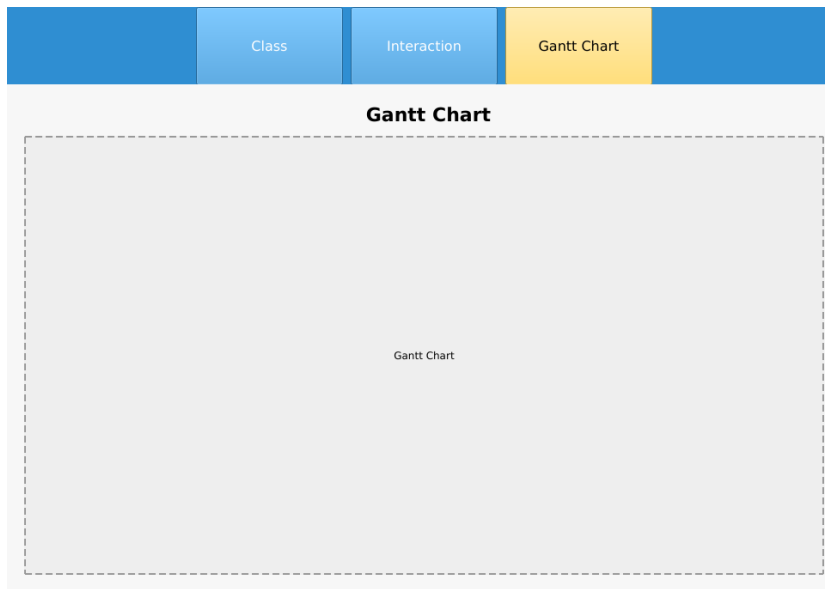
[!NOTE]  
 A relational complete language is a language at least as powerful (equivalent) as the Relational Algebra or the Relational Calculus.

**Database Analysis (Normalization)**

Using techniques called normalization to check whether your DB is good enough for the

## Design Page





## User Settings Page

**Account:**

Username: Raph  
Email: raph@example.com  
Phone: +66 1234 5678  
Account ID: 12345  
Status: Active

**Edit Account Information:**

Username:

Email:

Phone:

Account ID (Read-only):

PushButton



## Processes

No.	Label	Name	Day(s) to complete	Predecessor
1	<i>START</i>	Start the project (01/01/2025)	0	-
2	$T_1$	Prepare Project Topic	2	<i>START</i> (1)
3	$T_2$	Define Requirements	4	<i>START</i> (1)
4	$T_3$	Define Core Features	3	<i>START</i> (1)
5	$T_4$	Decide Technology Stack	2	<i>START</i> (1)
6	$M_1$	Finished Planning Phase	2	$T_1, T_2, T_3, T_4$ (2;3;4;5)
7	$T_5$	Design UI Layout	3	$M_1$ (6)
8	$T_6$	Design and Normalize Database Model	4	$M_1$ (6)
9	$T_7$	Design API	6	$M_1$ (6)
10	$M_{2.1}$	Finished Designing Phase (Client Side)	0	$T_5$ (7)
11	$M_{2.2}$	Finished Designing Phase (Server Side)	0	$T_6, T_7$ (8;9)
12	$T_8$	Implement Client Side	18	$M_{2.1}$ (10)
13	$T_9$	Implement Server Side	22	$M_{2.2}$ (11)
14	$M_{3.1}$	Finished Development Phase (Client Side)	0	$T_8$ (12)
15	$M_{3.2}$	Finished Development Phase (Server Side)	0	$T_9$ (13)
16	$T_{10}$	Test Client Side	2	$M_{3.1}$ (14)
17	$T_{11}$	Test Server Side	2	$M_{3.2}$ (15)
18	$M_4$	Finished Testing Phase	0	$M_{3.2}$ (16;17)
19	$T_{12}$	Deploy Client Side	1	$T$ (18)
20	$T_{13}$	Deploy Server Side	1	$M_{3.2}$ (18)
21	<i>FINISH</i>	Finished the project	0	$T_{11}, T_{13}$ (19;20)

## Tasks and Milestones

### 1. Planning Phase

- Prepare Project Topic
- Define Requirements
- Define Core Features
- Decide Technology Stack

### 2. Designing Phase

- Design UI Layout
- Design and Normalize Database Model

- Design API

### 3. Development Phase

- Implement Client Side
  - Main Page
  - User Uettings Page
  - Task Assignment Page
    - \* Live Markdown Preview
    - \* Simple Editor
    - \* Save File Button
  - Design page
    - \* Gantt chart
    - \* Class diagram
    - \* Interaction diagram
- Implement Server Side
  - Implement Program API
  - Implement Database

### 4. Testing Phase

- Test Client Side
- Test Server Side

### 5. Deployment Phase

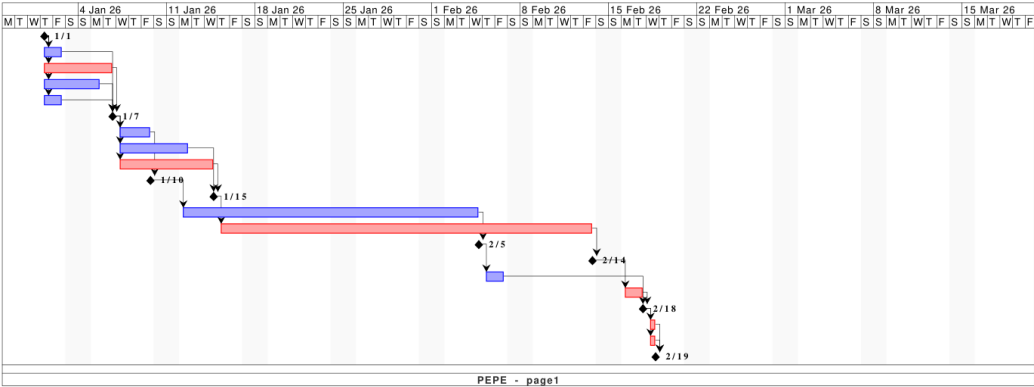
- Deploy Server Side
- Deploy Server Side

Tasks Table

		Name	Duration	Start	Finish	Predecessors
1		START	0 days	1/1/26, 8:00AM	1/1/26, 8:00AM	
2		Prepare Project Topic	2 days	1/1/26, 8:00AM	1/2/26, 5:00PM	1
3		Define Requirements	4 days	1/1/26, 8:00AM	1/6/26, 5:00PM	1
4		Decide Core Features	3 days	1/1/26, 8:00AM	1/5/26, 5:00PM	1
5		Decide Tech Stack	2 days	1/1/26, 8:00AM	1/2/26, 5:00PM	1
6		Finished Planning P...	0 days	1/6/26, 5:00PM	1/6/26, 5:00PM	2;3;4;5
7		Design UI Layout	3 days	1/7/26, 8:00AM	1/9/26, 5:00PM	6
8		Design and Normaliz...	4 days	1/7/26, 8:00AM	1/12/26, 5:00PM	6
9		Design API	6 days	1/7/26, 8:00AM	1/14/26, 5:00PM	6
10		Finished Designing ...	0 days	1/9/26, 5:00PM	1/9/26, 5:00PM	7
11		Finished Designing ...	0 days	1/14/26, 5:00PM	1/14/26, 5:00PM	8;9
12		Implement Client Side	18 days	1/12/26, 8:00AM	2/4/26, 5:00PM	10
13		Implement Server Side	22 days	1/15/26, 8:00AM	2/13/26, 5:00PM	11
14		Finished Developme...	0 days	2/4/26, 5:00PM	2/4/26, 5:00PM	12
15		Finished Developme...	0 days	2/13/26, 5:00PM	2/13/26, 5:00PM	13
16		Test (Client)	2 days	2/5/26, 8:00AM	2/6/26, 5:00PM	14
17		Test (Server)	2 days	2/16/26, 8:00AM	2/17/26, 5:00PM	15
18		Finished Testing Ph...	0 days	2/17/26, 5:00PM	2/17/26, 5:00PM	16;17
19		Deploy Client	1 day	2/18/26, 8:00AM	2/18/26, 5:00PM	18
20		Deploy Server	1 day	2/18/26, 8:00AM	2/18/26, 5:00PM	18
21		FINISH	0 days	2/18/26, 5:00PM	2/18/26, 5:00PM	19;20

PEPE - page1

Gantt Chart



# Tasks Network

