Development of a Visual Project Management Application

Completed By:

Sychev A.D.

ET-212

Formulation of the problem

Development of a Visual Project Management Application

The application for visual project management is designed to assist users in organizing, managing, and tracking projects and tasks in a convenient and visual format. Such applications enable efficient planning, task assignment, tracking, as well as managing project resources and timelines.

Program Requirements:

- Graphical User Interface (GUI);
- Mouse functionality;
- Database for multi-user functionality;

Necessary Functions

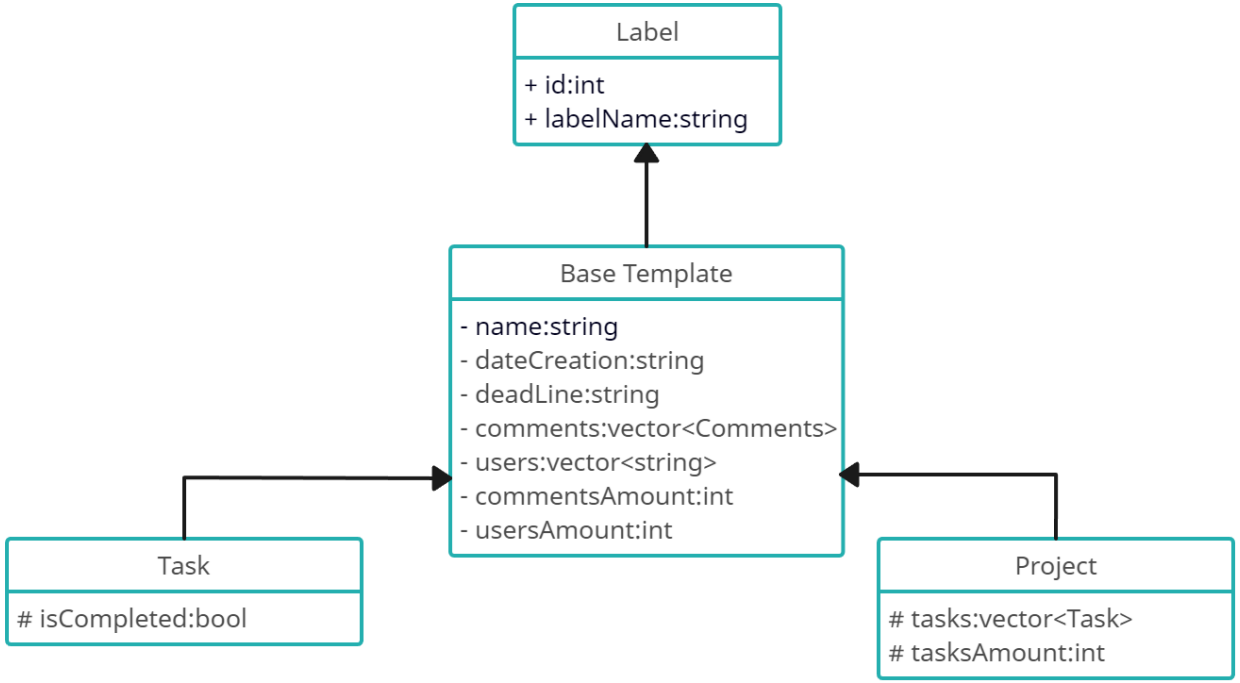
- Opportunity for user authentication;
- Opportunity to browse and create your own projects, join to other people projects;
- Opportunity to browse, change, delete any tasks in project;
- Opportunity to create special comments for every task;

Domain analysis reveals the following objects

- User. User will have name, special id, password and information about his projects;
- Project. Every project will have its own name and will have information about amount of people , informations about all its tasks, amount of tasks and deadline;
- Task. Every task will have its own name, date of creation, deadline, current state and comments;

Main classes and Hierarchy

Module: Module

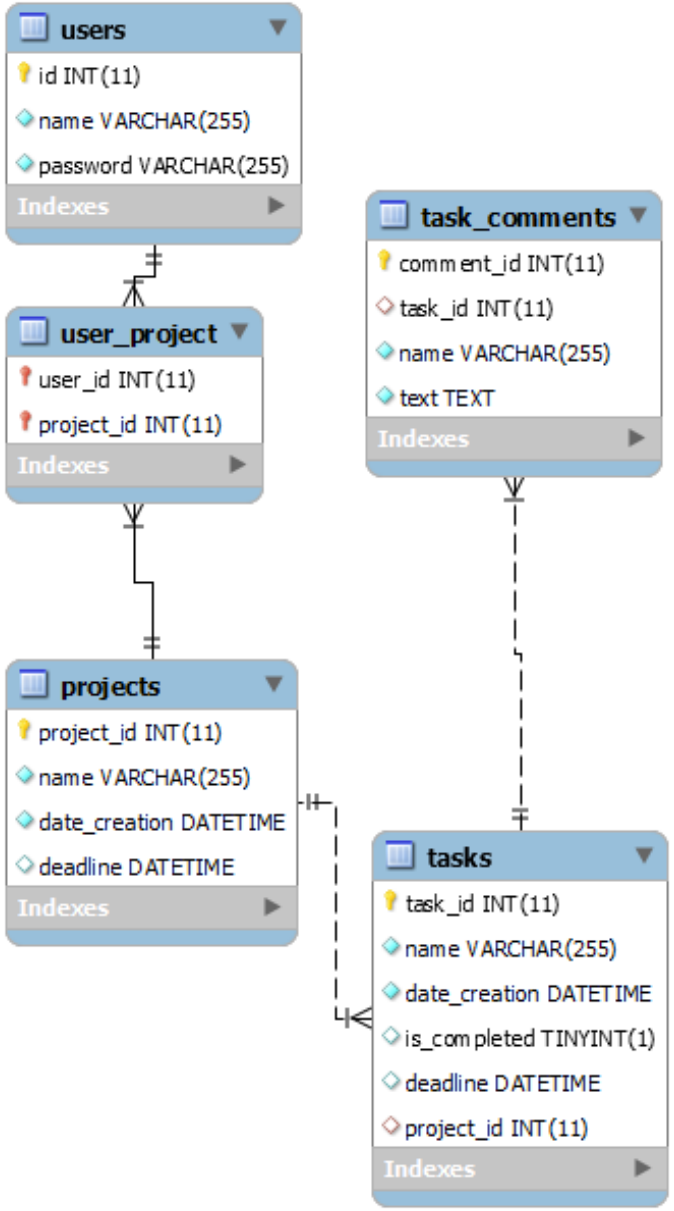


Module: Session

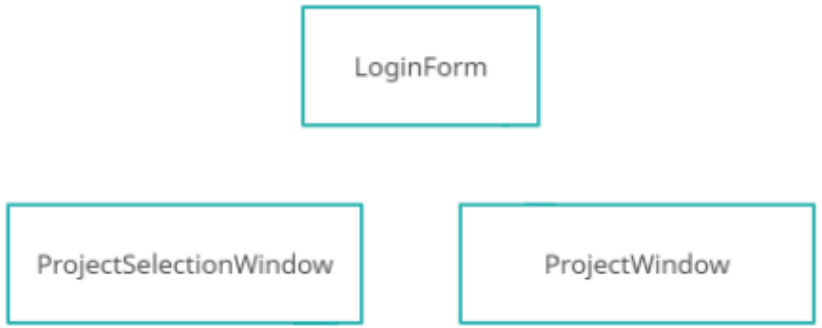
Database
- driver: sql::Driver* - url: sql::SQLString - properties: sql::Properties - conn: unique_ptr<sql::Connection> + addUserToProject(name: string, id: int): string + disconnect(): void + loadProjectsDescription(currUser: User): vector<Label> + deleteProject(id: int): void + saveProject(project: Project): void + loadProject(id: int): Project + saveTask(task: Task, id: int): void + deleteTask(id: int): void + saveComment(id: int, comment: Comment): void + loadComments(id: int): vector<Comment> + loadUsers(projectId: int): vector<string> + ConnectUser1(name: string, password: string): User

Implementation Features

For the development of the program, a database was created based on the MariaDB DBMS, which allows implementing a multi-user mode of operation. A user can work remotely in the program if they have the ability to connect to the server.

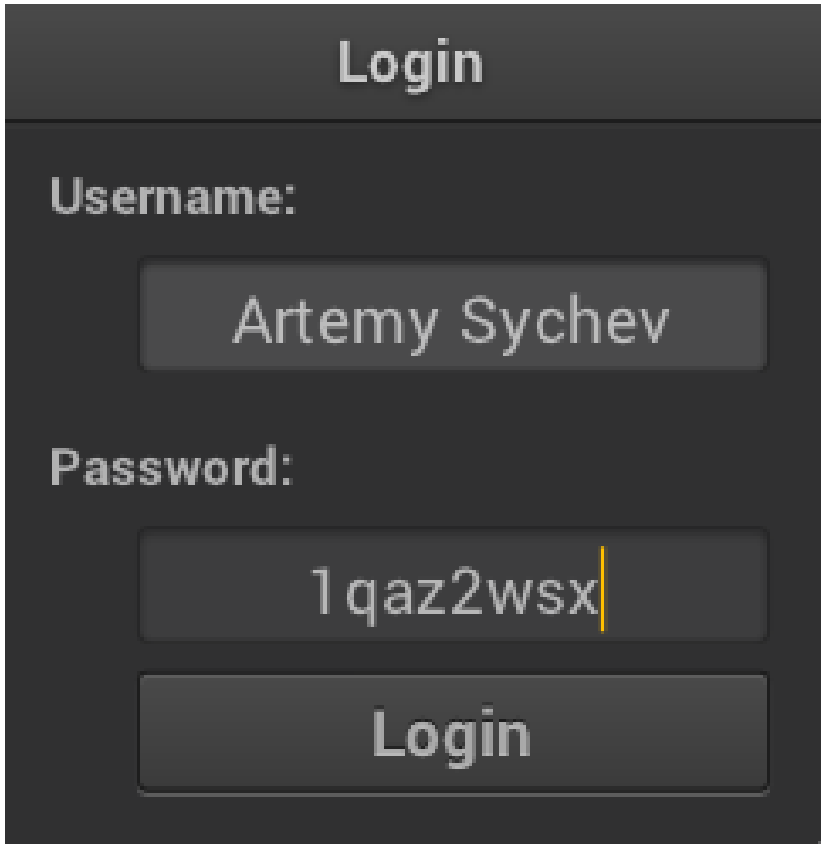


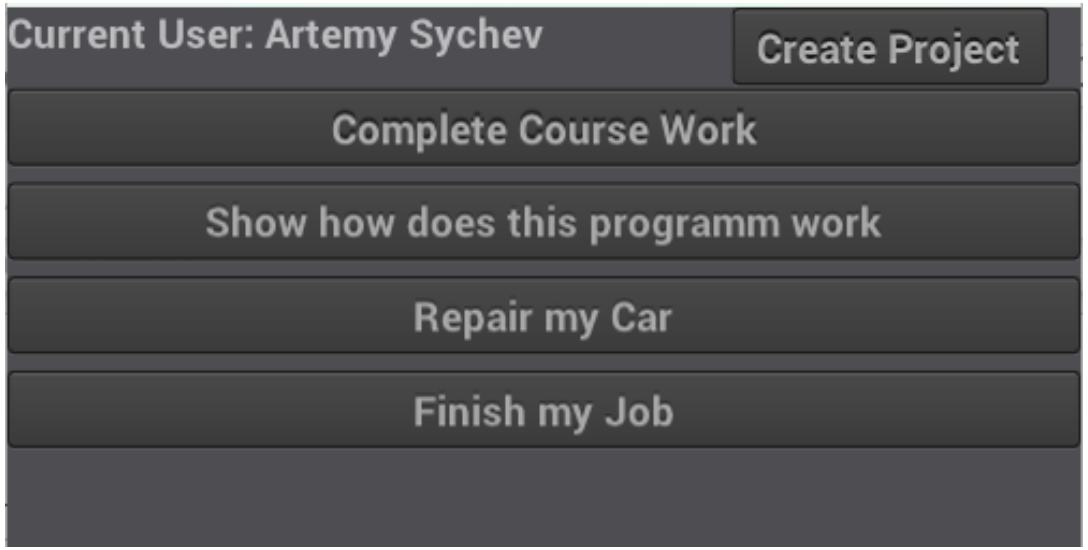
Implementation Features



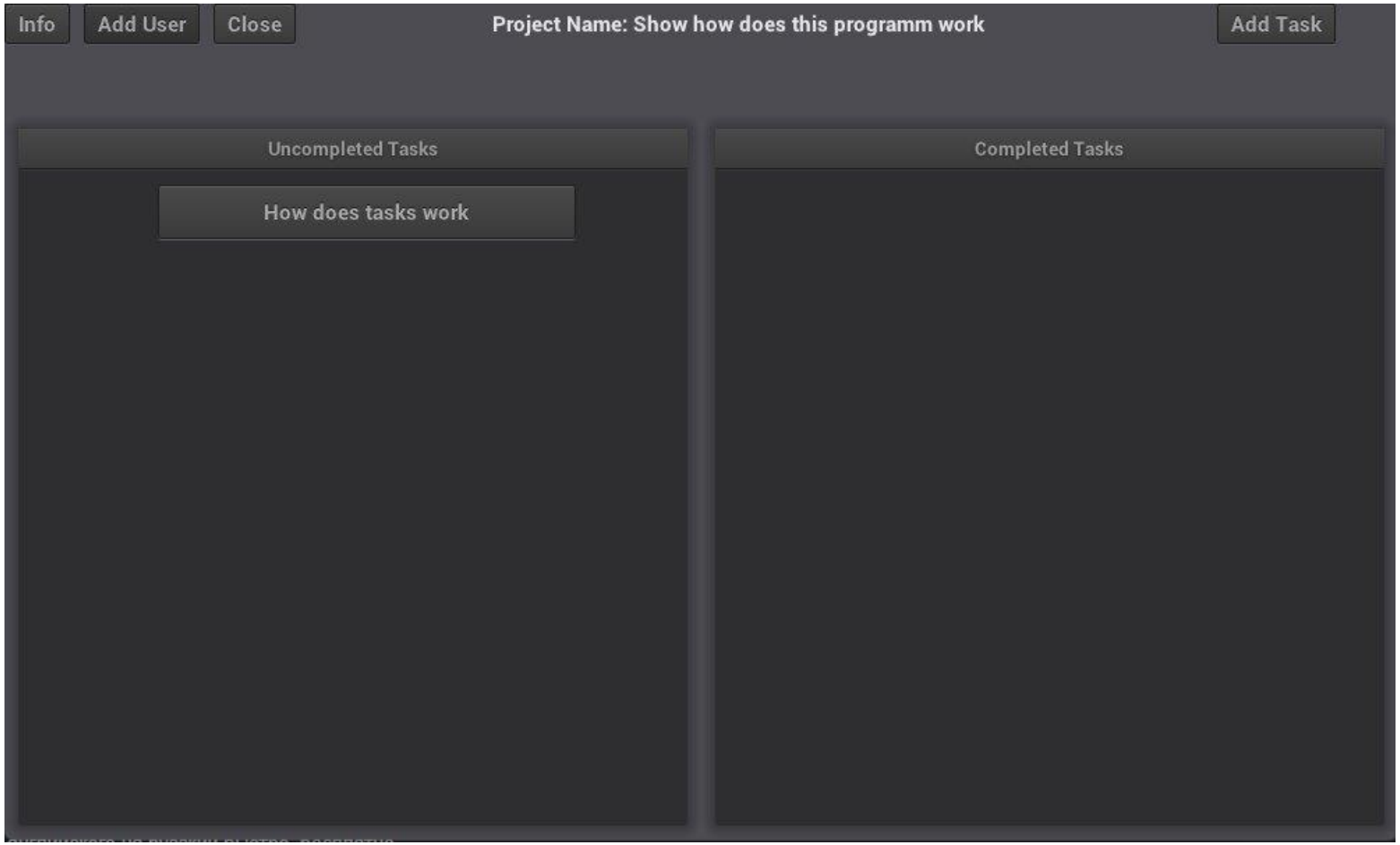
Besides the main classes, during development, I created an additional module necessary for rendering the graphical interface. **LoginForm** is a class needed for rendering the initial authentication window. **ProjectSelectionWindow** is used for rendering the window where project selection occurs. And **ProjectWindow** is necessary for rendering the window of the selected project."

Example of authorization

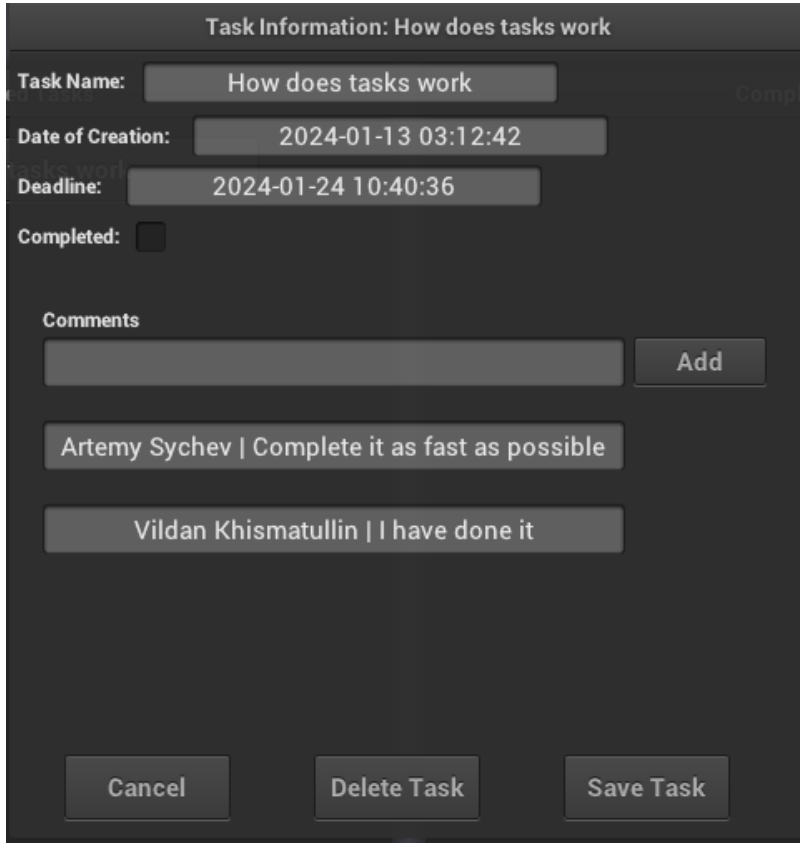


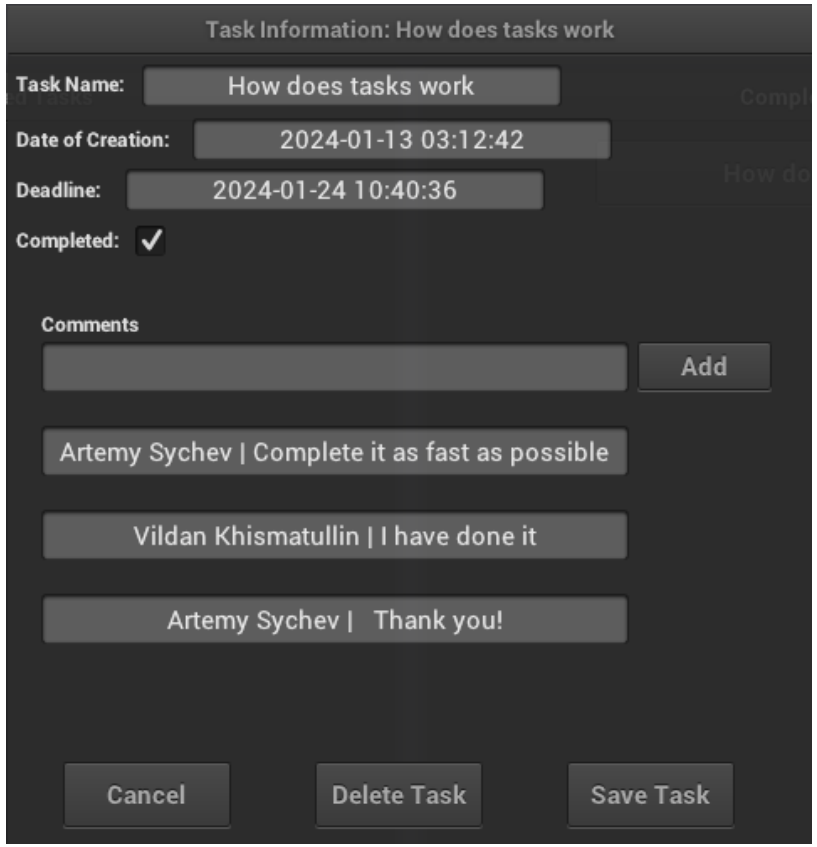


Main Menu


Info
Add User
Close
Project Name: Show how does this programm work
Add Task
Uncompleted Tasks
Completed Tasks
How does tasks work

Example of Editing a Task





Conclusion

During the coursework, domain objects were identified, and a system of classes for them was defined. The program interface was developed. After object-oriented design, the classes were implemented in C++. The developed code was tested against control tests, and necessary corrections were made to the code. Documentation describing the installation and usage of the program was developed. Thus, the goal of the work was achieved, and the tasks were solved.