Task Management System

Team : Mironshoh Inomjonov 2110036
Dilnoza Raxmatullayeva 2110145
Damir Mirzabayev 2110015
Behzod Hakimov 2110037
Okiljon Dadakhonov 2110172
Abdullo Eshonxonov 2110065

Task Management Application Documentation

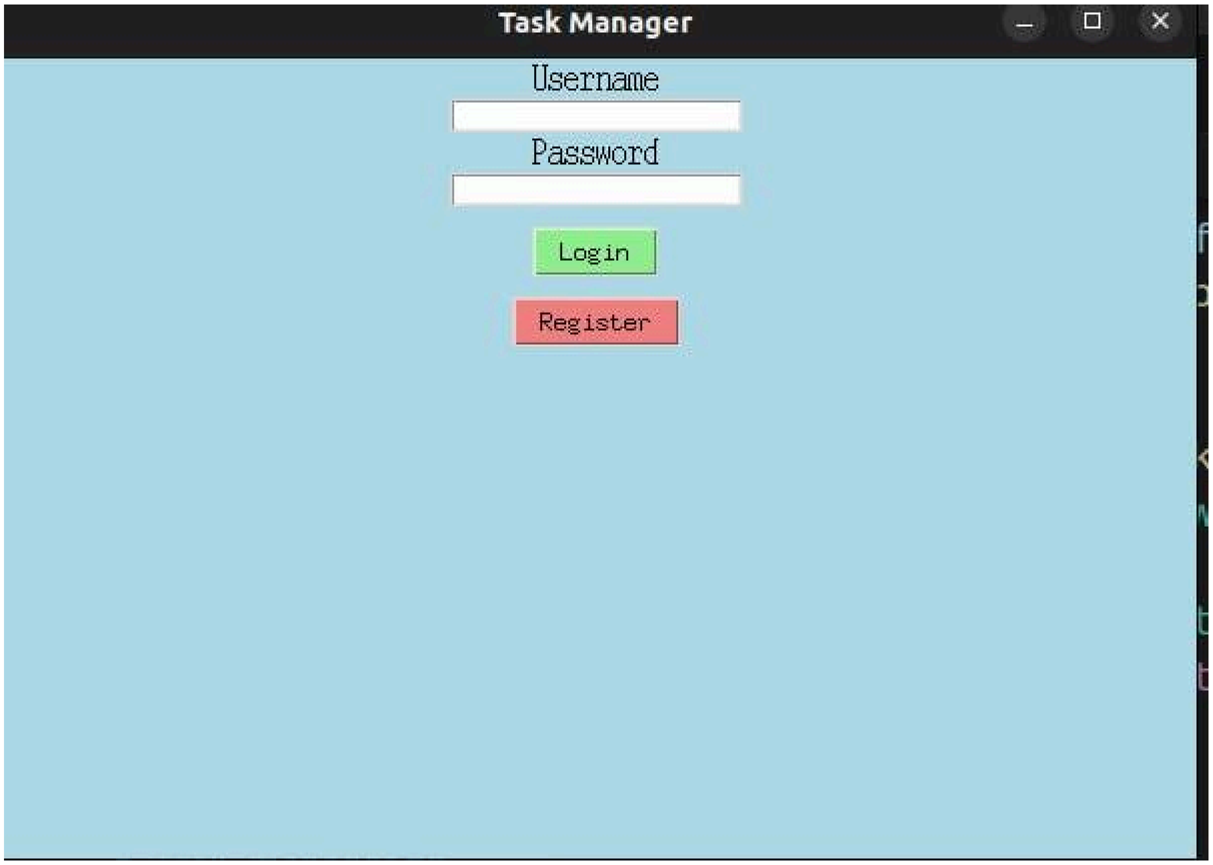
Overview

The Task Management Application is a comprehensive tool designed to facilitate the process of managing tasks within a software development team. It allows for distinct roles, namely managers and developers, to interact with the system in ways pertinent to their responsibilities. Managers can create, assign, and monitor tasks, while developers can view and update the status of tasks assigned to them.

Features

User Authentication

Registration: New users can register as either a manager or a developer by providing a username, password, and role.
Login: Users log in with their credentials. Access to different functionalities is based on the user's role.

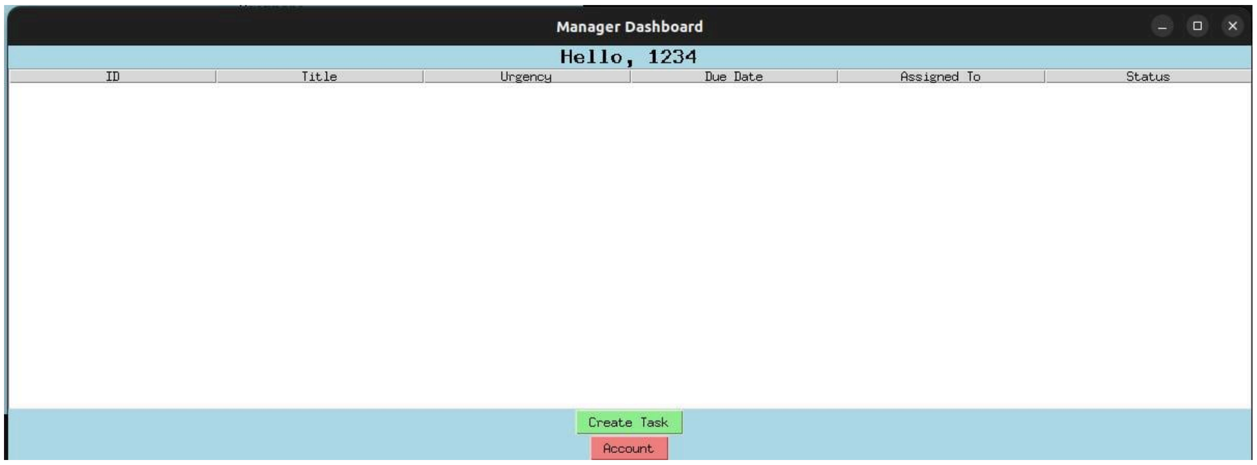


Manager View

Create Tasks: Managers can create tasks, setting properties like title, urgency level, due date, and the developer to whom the task is assigned.

View Tasks: Managers can view a list of tasks that they have created, including details such as title, urgency, due date, assigned developer, and current status.

Account Management: Access to account settings and the ability to log out.

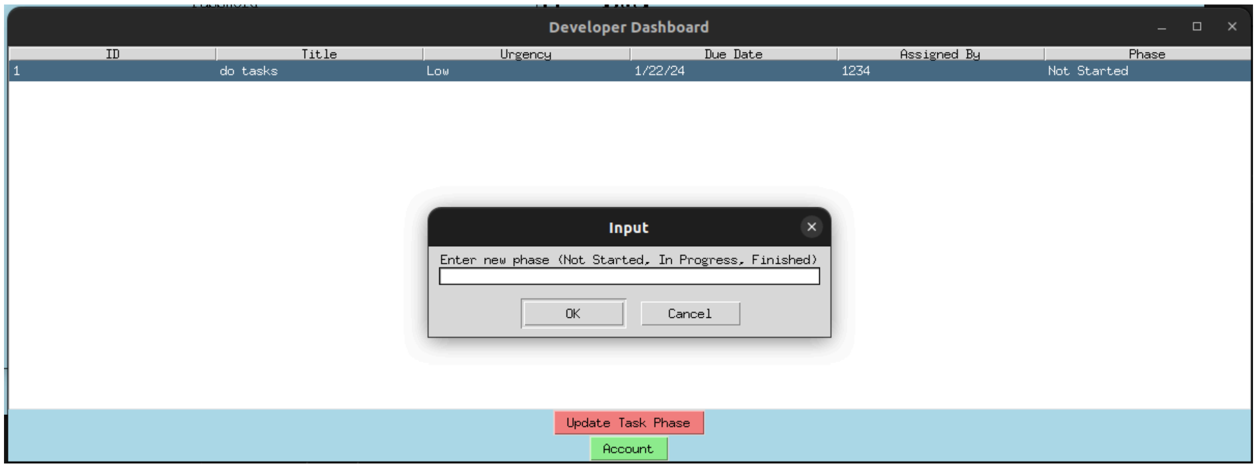


ID	Title	Urgency	Due Date	Assigned To	Status

Developer View

View Assigned Tasks: Developers can see a list of tasks that have been assigned to them, with details including task name, urgency level, due date, who assigned the task, and the current phase of the task.

Update Task Phase: Developers can update the phase of a task to reflect its current status (Not Started, In Progress, Finished) using a dropdown selection.



ID	Title	Urgency	Due Date	Assigned By	Phase
1	do tasks	Low	1/22/24	1234	Not Started

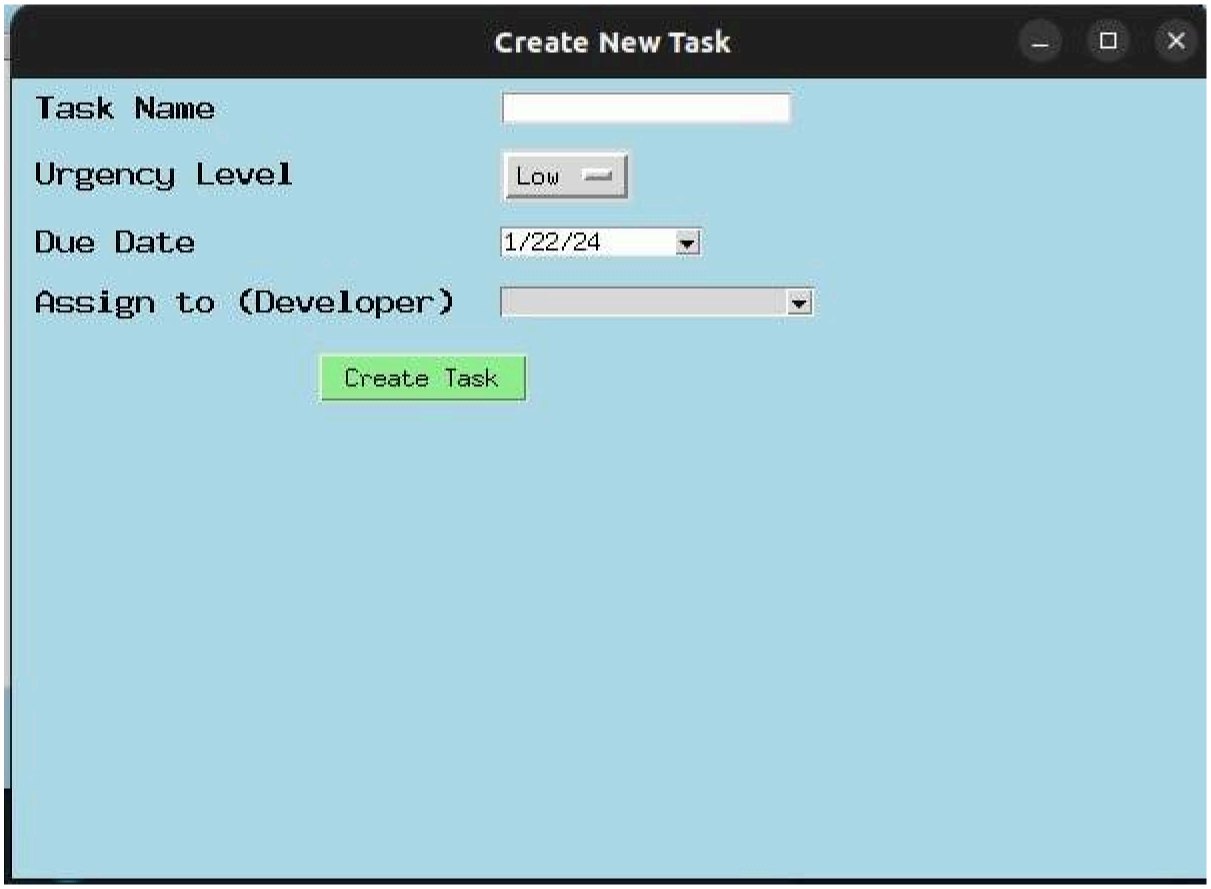
Enter new phase (Not Started, In Progress, Finished)

Task Assignment

Assigning to Developers: When creating a task, managers select a developer to assign the task to from a dropdown list of available developers. The system ensures that tasks cannot be assigned to other managers.

Task Phase Changing

Phase Update: Developers can change the phase of a task using a dropdown menu within their view, promoting clear communication of progress.


Create New Task

Task Name	
Urgency Level	Low
Due Date	1/22/24
Assign to (Developer)	

Technical Details

Backend Development for Task Management and Assignment Software

The robust backend of our task management and assignment software is engineered with SQLite, a lightweight and efficient relational database management system. This choice ensures reliable data persistence, seamless data retrieval, and optimal performance, contributing to the overall stability and functionality of the application.

Database Structure:

The backend comprises two primary tables, namely "users" and "tasks," each carefully designed to encapsulate essential information for user management and task tracking.

Users Table:

The "users" table serves as the repository for user-related data. This includes fields such as user ID, username, password (securely hashed for confidentiality), role designation, and any other pertinent details specific to user management.

The table structure is designed to support diverse user roles, ensuring flexibility and scalability as the software accommodates various stakeholders with distinct responsibilities within the task management ecosystem.
Tasks Table:

The "tasks" table is pivotal in storing information related to tasks, creating a centralized hub for task management. Fields include task ID, task name, description, assigned user, status, due date, priority, and any other attributes crucial for effective task tracking and assignment.
The table design facilitates the establishment of relationships between tasks and users, allowing for seamless assignment and tracking of responsibilities. The inclusion of status and priority fields ensures that users can efficiently manage and prioritize their tasks.
Data Integrity and Relationships:

Data integrity is maintained through the enforcement of relationships between tables. The user ID in the "users" table serves as a foreign key in the "tasks" table, establishing a clear association between tasks and their assigned users. This relational approach ensures consistency in data representation and supports the efficient retrieval of information when generating reports or conducting analyses.

SQLite for Data Persistence:

SQLite is employed as the database engine due to its serverless architecture, simplicity, and reliability. Its lightweight nature makes it well-suited for our task management application, ensuring fast query execution and low resource utilization. The use of SQLite also simplifies deployment, making it easy to integrate into various environments.

Scalability and Performance:

The backend is designed with scalability in mind, allowing the software to accommodate an increasing volume of users, tasks, and data over time. The efficient structure of the database tables and the utilization of SQLite contribute to optimal performance, ensuring that the application remains responsive even as the user base and task complexity grow.

Users Table
id: Primary key.
username: Unique username for the user.
password: Hashed password for security.
role: Role of the user, either 'manager' or 'developer'.

Tasks Table
id: Primary key.

title: Title of the task.
description: Description of the task.
urgency: Urgency level of the task (Low, Medium, High).
due_date: Due date for the task completion.
assigned_to: Foreign key referencing users(id), indicating the developer assigned to the task.
status: Current phase of the task (Not Started, In Progress, Finished).
created_by: Foreign key referencing users(id), indicating the manager who created the task.

Front end

The frontend of our task management and assignment software has been meticulously crafted using Tkinter, a powerful and versatile Python library for creating graphical user interfaces. This choice ensures that users are provided with an intuitive, seamless, and interactive experience as they navigate through various functionalities.

User Roles and Interface Customization:
Our software caters to different user roles, tailoring the interface based on specific responsibilities and permissions. Whether you're an administrator, project manager, team member, or any other designated role, the frontend dynamically adjusts to present relevant features and information. This customization enhances user efficiency by streamlining access to essential tools.

Window Architecture:
The user interface is organized into distinct windows, each serving a specific purpose in the task management workflow. These windows are thoughtfully designed to present information in a clear and structured manner, minimizing cognitive load and optimizing user experience. The window architecture ensures that users can seamlessly transition between different aspects of task management, such as project planning, task assignment, and progress tracking.

Interactive Elements:
To facilitate user interaction, a variety of interactive elements have been incorporated into the frontend. Buttons, strategically placed throughout the interface, trigger essential actions like creating tasks, assigning responsibilities, updating statuses, and generating reports. Entry fields enable users to input and edit task details efficiently, while dropdown menus provide easy selection of options, enhancing the overall usability of the application.

Role-Specific Functionalities:
Depending on their roles, users can access specific functionalities through the frontend. Administrators may have privileges to set up projects, manage user accounts, and define access levels. Project managers can create and assign tasks, track progress, and generate reports. Team members, on the other hand, have access to tasks assigned to them and can update task statuses. This role-specific approach ensures that each user interacts with the frontend in a manner aligned with their responsibilities.

Security Measures

1. Password Security:
-Hashing Algorithm: Passwords are hashed using the bcrypt hashing algorithm. Bcrypt is a robust and widely accepted cryptographic hash function designed for password hashing. This ensures that even in the event of a data breach, user passwords remain secure and cannot be easily deciphered.

-Salt Integration: To enhance security, a unique salt is generated for each password before hashing. Salting adds an extra layer of complexity, making it significantly more difficult for attackers to use precomputed tables (rainbow tables) to crack passwords.

2. Input Validation:
-Preventing Injection Attacks: User input is thoroughly validated to prevent injection attacks. This includes both client-side and server-side validation to ensure that malicious data or code cannot be injected through forms, queries, or other input fields.

-Cross-Site Scripting (XSS) Protection: Measures are in place to mitigate XSS attacks by validating and sanitizing user input before displaying it on any user interface. This helps prevent attackers from injecting malicious scripts into web pages viewed by other users.

3. Authorization Controls:
-Role-Based Access Control (RBAC): The software follows a role-based access control model, ensuring that users only have access to functionalities and data relevant to their roles within the organization. This minimizes the risk of unauthorized access and data exposure.

-Fine-Grained Permissions: Administrators can define and customize permissions at a granular level, allowing precise control over who can perform specific actions within the software. This helps in aligning access privileges with organizational requirements.

4. Data Encryption:
-Data in Transit: All data transmitted between the client and server is encrypted using secure protocols such as HTTPS. This ensures the confidentiality and integrity of the information as it travels across networks.

-Data at Rest: Sensitive data stored in databases is encrypted to protect against unauthorized access in case of a data breach or physical theft of storage devices.

Usage

Upon launching the application, users are greeted with a login window. New users can navigate to the registration window to create an account. After logging in, the user is presented with either the manager or developer dashboard based on their role. Managers can create new tasks and view tasks they have assigned, while developers can view their assigned tasks and update their statuses as needed.

Conclusion

This Task Management Application streamlines the process of assigning and tracking tasks within a development team, ensuring clarity of responsibilities and progress tracking. Its user-friendly interface and secure backend make it a reliable tool for teams of all sizes.