Fv2580
Moeid Shariff

TODO list Application

Functional requirements

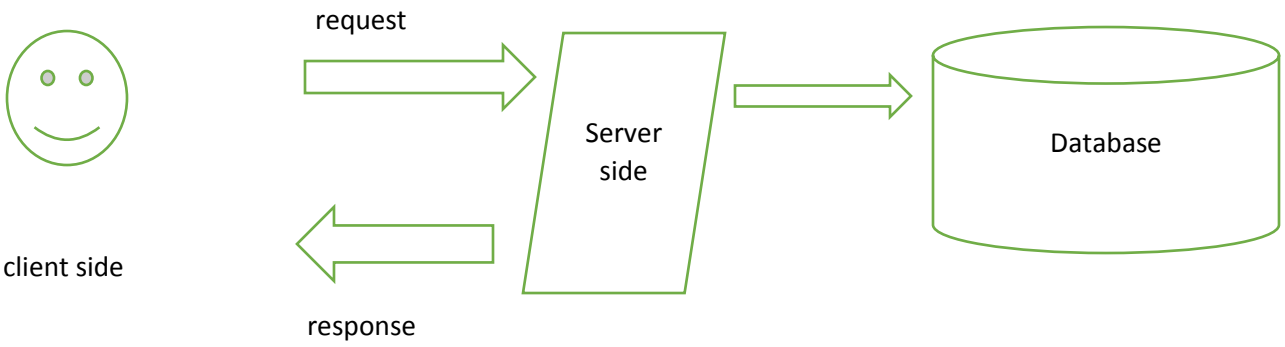
The TODO list application should have three basic operations which will be the part of functional requirements.

- The add button that will insert tasks in the database. Tasks will be inserted with the help of three text fields including TODOtask (string), Date (varchar 8 char), Time (varchar 6 char)
- The delete button that will delete individual tasks. Tasks are deleted based on their id numbers; the text field accepts integers.
- The view button displays all the tasks in the database.

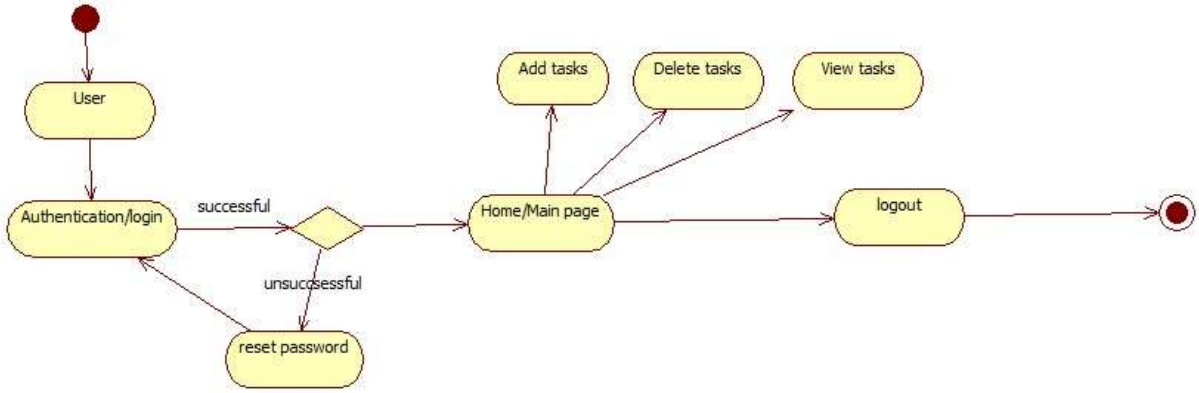
Non-functional requirements

- Usability
  - The app should be easier to navigate.
  - Future versions should support multiple languages.
- Security
  - Different access levels on the database side.
  - Passwords should include special characters, at least 8 chars, renewing policy
- Performance
  - Lower browser refresh time
  - Minimize database query load times

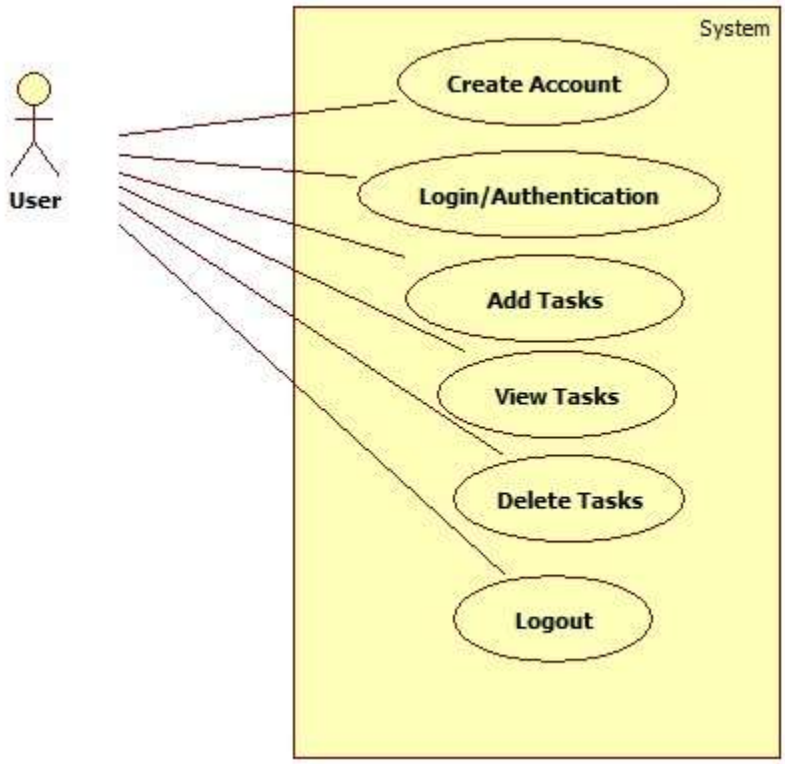
System Architecture Diagram



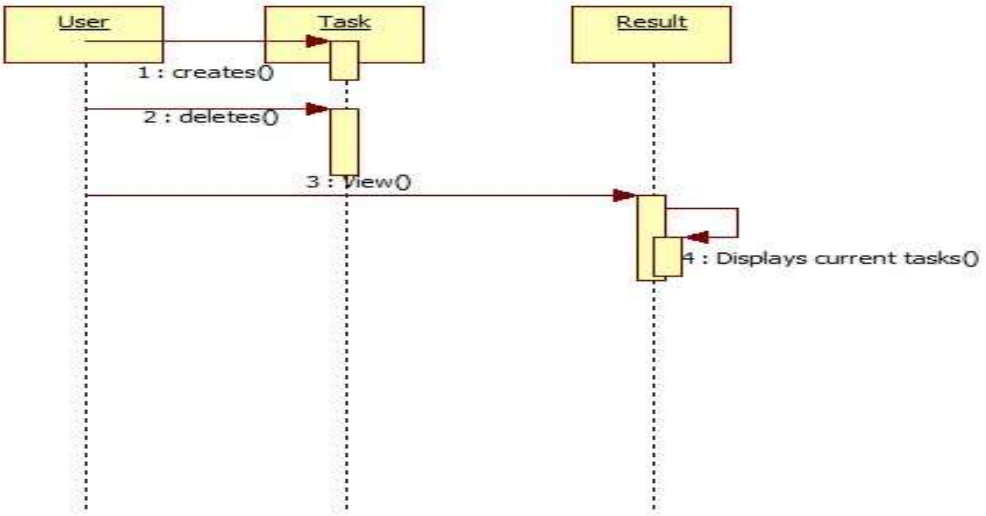
Data Flow Diagram



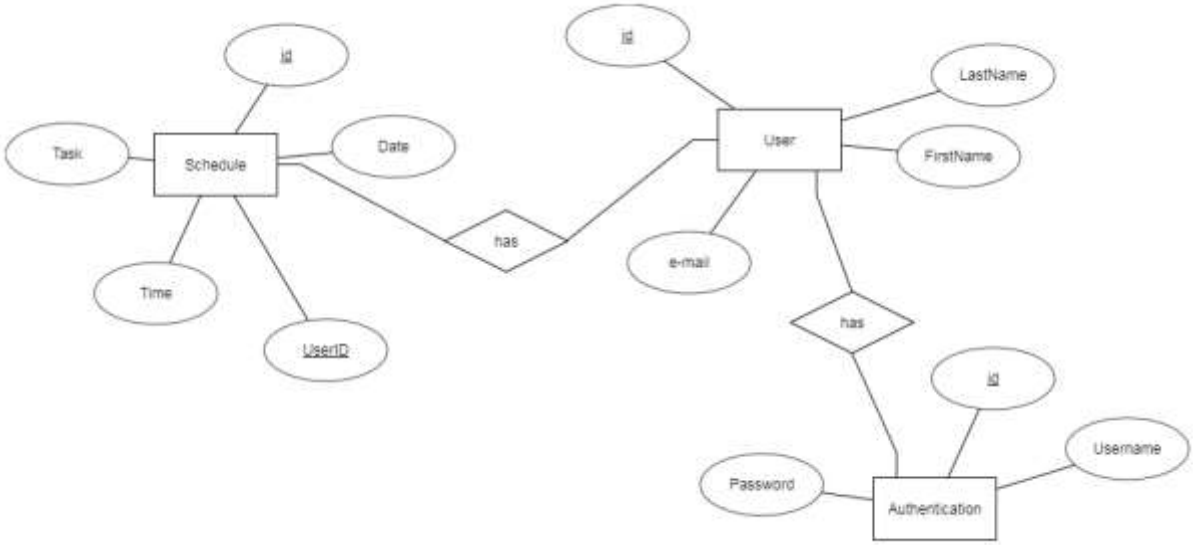
Use Case



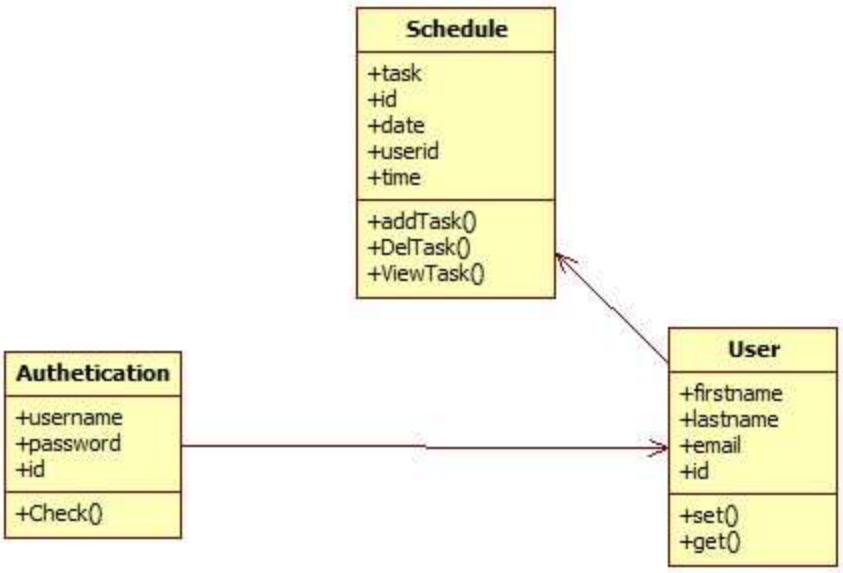
Sequence diagram



Database design



Class diagram



Test Cases

Creating a user account – To create a user account we will open the application and click on the create new user button. After that the screen should show six text fields which should ask the user for their first and last name, e-mail address, phone number (optional), Username and Password. Once the user provides the requested details, the application will display a message indicating that a user account has been successfully created.

Login/logout with a user account – The user will login with his/her credentials and if the login is successful the homepage will be displayed. In case of an incorrect password the user will be displayed a message to reset the password. To test the logout feature, user will simply click on the logout button and once the button is clicked user should be redirected to the login page.

Reset Password – Once the user clicks this button they will be asked to verify their details either by a secret question or sending a reset link to their e-mail address that they registered while creating an account.

Add Task - Add a task in the database successfully through the browser by clicking on the “add task” button in the application. There are three parameters to add a task which includes adding the description of the task in the form of text, adding the date in this format 1/1/2000 and adding a time per this format 12:00am.

Delete Task - Delete a task in the database by clicking on the delete button and inputting the task number i.e. 1,2,3. Upon successful deletion the task will be deleted from the database.

View Task - View all current tasks in the database by clicking on the view button. It should show all the current tasks available in the system.

Fv2580
Moeid Shariff

Work Breakdown

Task name: Todo list app

Estimated No of hours: 12 hours

SQL : 2-4 hours

HTML: 1 hour

PHP: 2 hours

GitHub: 1 hour

Documentation: 2-4 hours

Actual time spent: 13 hours

Notes: With regards to expandability the application can be much more robust. In its current state the operations it performs are rudimentary and limited to only adding, deleting and viewing tasks. In the documentation, I've tried to add future functionality to include a login/authentication system that will provide the users with a more customized feel of the app.

The schema (database design) includes tables that are not implemented in the code yet.

Issues: I would say documentation was the challenging part as different diagrams are required to understand what is required in them was time consuming to figure out. Also, getting back to GitHub was a learning curved as I have not used it for quite some time. At first for all the diagrams I was using Microsoft word but then I realized there were better software such as StarUML which are easier to use and better in presentation.