Dewan Hassan

CSC 4996

**TODO Application Documentation**

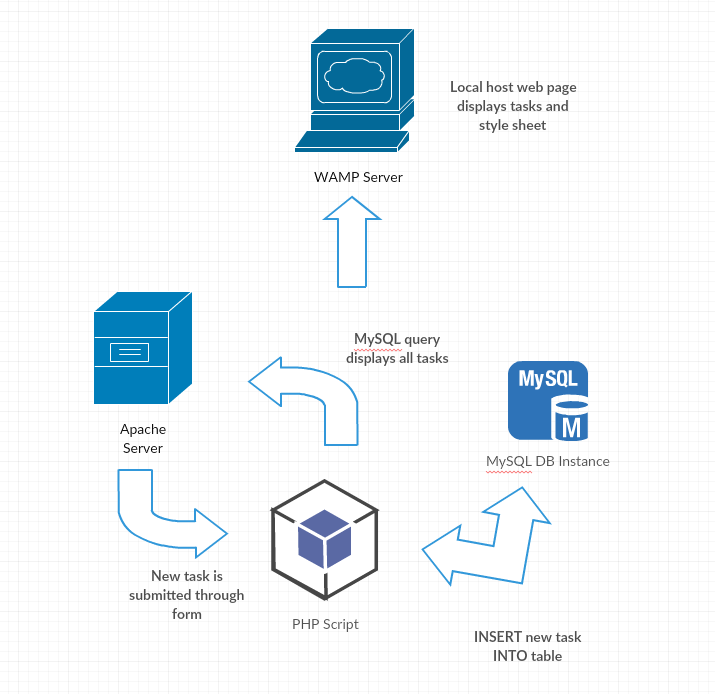
**Functional Requirements:**

1. Users are able to input a task into a form field, and have it appear on the web page once submitted.
2. Users can view existing tasks already inputted.
3. Users can delete tasks after they have been completed.

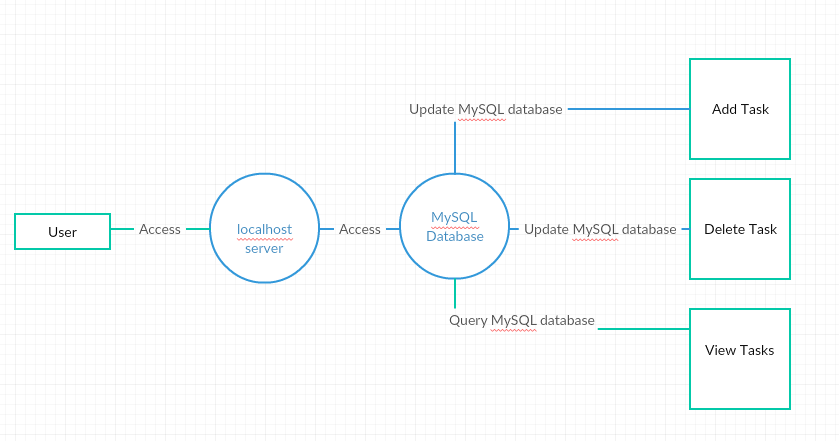
**Non-Functional Requirements:**

1. Accessibility
   1. Any user can access the TODO application that is hosted on the local account. No log in authentication is required.
   2. These users can view, add, and delete tasks on the application.
2. Capacity
   1. Any number of tasks can be added to the TODO application, although it is subject to limitations set by phpmyAdmin.
3. Resilience
   1. Tasks will remain visible on the task list for any length of time. However, after deletion of the task it can no longer be recovered.
   2. Updates to the task list occur on phpmyAdmin and reflect the state of the database following adding or deleting tasks.
4. Supportability
   1. Tasks can support an input of up to 255 characters.
   2. The form field cannot be blank when adding a task.

**System Architecture Diagram**



**Data Flow Diagram**



**Use Case**

Add Task

1. User types new task into form field.
2. User submits task into website by pressing “Add Task” button.

Delete Task

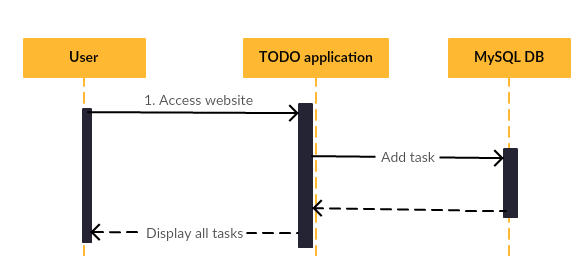
1. User views existing tasks when accessing website.
2. User deletes selected task by pressing ‘X’ under action.

View Task

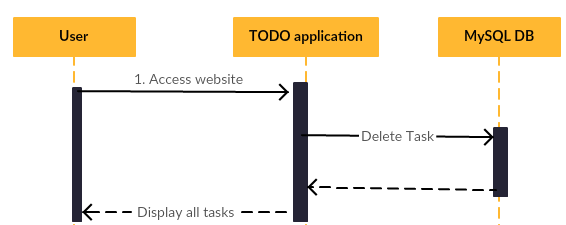
1. User access website and can view existing tasks.
2. Tasks automatically update following add or delete actions.

**Sequence Diagram**

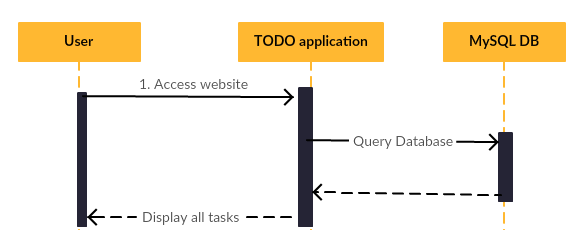
**Add Task**



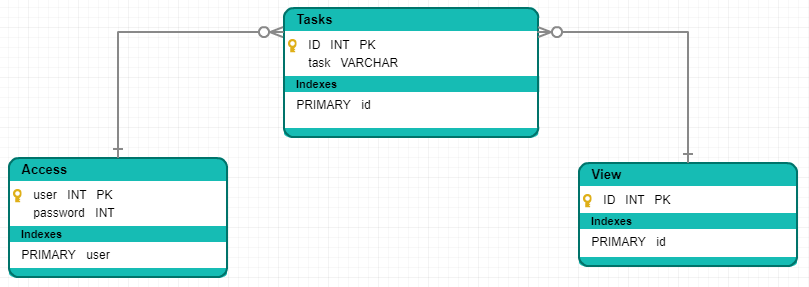
**Delete Task**



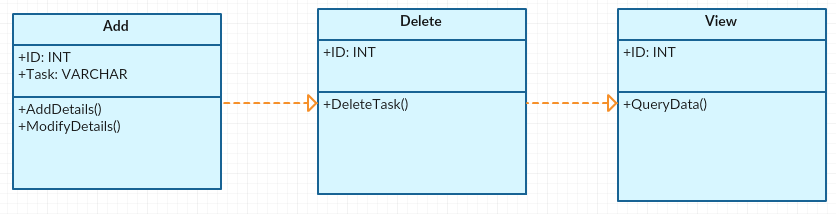
**View Task**



**Database Diagram**



**Class Diagram**



**Test Case**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case** | | | | |
| **Step** | **Procedure** | **Expected Result** | **Actual Result** | **Status** |
| Add Task | Add a task into the form. | The web page will update with the task inputted. | The task was inputted; however, the tasks are not always in order. | Pass |
| Delete Task | Press the “Done” button to remove a task from the list. | The web page will remove the selected task. | The task was removed. | Pass |
| View Task | Users will view a static image of the current task list. | The task list will be displayed. | The tasks were displayed. | Pass |

Task: Download WAMP & learn how to use it

Estimated No of hours: 1

Actual time spent: 1.5 hours

Notes: There are different types of servers and versions, using WAMP 3.0.6

Issues: Finding the correct version to download

Task: Learn PHP basics and how to connect MySQL database

Estimated No of hours: 2

Actual time spent: 5 hours

Notes: PHP basics and syntax is very similar to other languages, learning how to integrate MySQL database took a lot of time.

Issues: Input in form field did not input database. Switched to $\_POST method.

Task: Set up new table, “tasks”

Estimated No of hours: 1

Actual time spent: 1.5 hours

Notes: Using table descriptors, added placeholders.

Issues: None.

Task: Add task functionality

Estimated No of hours: 1

Actual time spent: .5 hours

Notes: Used echo statement to print, and stored to MySQL database.

Issues: Updating database and keeping numbering in order.

Task: Delete task functionality

Estimated No of hours: 1

Actual time spent: 1.5 hours

Notes: Tasks were deleted according to the ID number originally assigned.

Issues: None.

Task: .css stylesheets

Estimated No of hours: 1

Actual time spent: 3 hours

Notes: Played with border, fonts, and sizing. Kept it simple, black and white interface.

Issues: Adjusting the done button to aligned to the center.