



International Islamic University Chittagong

Department Of Computer Science & Engineering

Project Report

Course Code : CSE - 3532

Course Title : Tools & Technologies For Internet Programming

Project Title : Task Tracker

Prepared By :

Name : AFIA IBNAT

ID No. : C231516

Section : 5DF

Semester : 5th

Supervisor :

Sara Karim,

Adjunct Lecturer,

Department of CSE,

IIUC.

● Table Of Contents ●

S.L. NO.	Contents	Page NO.
1.	Title Of The Project	3
2.	Abstract	3
3.	Literature Review	3
4.	Methodology / System Design	4
5.	Flowchart	5
6.	Implementation	6 - 10
7.	Result & Analysis	11
8.	Challenges	11
9.	Conclusion	12
10.	References	12

● Title Of The Project ●

~ Task Tracker ~

● Abstract ●

The Task Tracker is a simple and user-friendly web-based application designed to assist individuals in managing their daily tasks effectively. It provides core functionalities such as user registration, login, task creation with associated dates and days, and task deletion. Users can easily view their task list and stay organized by updating or removing completed tasks. This tool is especially useful for those who want to improve time management and productivity. By offering a clean interface and essential features, the Task Tracker serves as a helpful solution for personal task tracking and planning.

● Literature Review ●

Several task management tools and to-do list applications exist in the current digital landscape, such as Google Tasks, Todoist, Microsoft To Do. These platforms offer features like task scheduling, reminders, and collaboration. While they are effective, many of them are either too complex for basic users or require account integration with other services.

Some existing systems also come with limited free access, require subscriptions for full functionality, or are not fully customizable for personal needs. Additionally, many apps are not lightweight or web-based, making them less accessible for users with limited devices or internet speed.

The Task Tracker project addresses these limitations by providing a simple, user-focused interface, allowing basic yet essential functionalities like task creation, viewing, and deletion without unnecessary complexity. It is designed to be lightweight, easy to use, and efficient, especially for individuals who only need basic daily task management without distractions.

● Methodology / System Design ●

1. Frontend Development (Client side) :

[i] **HTML (HyperText Markup Language):** Used to create and structure the web pages, including forms for login, signup, and task input.

[ii] **CSS (Cascading Style Sheets):** Applied for designing and styling the interface, ensuring a clean and responsive user experience.

[iii] **JavaScript:** Handles client-side logic such as form validation, displaying tasks dynamically, and triggering functions like adding or deleting tasks.

2. Backend Development (Server-Side) :

The backend is built using Python Flask, a lightweight web framework --

[i] Managing HTTP requests for login, registration, and task operations.

[ii] Validating user inputs and controlling access to different features.

[iii] Communicating with the database to store and retrieve user and task data.

[iv] Sending appropriate responses back to the frontend for rendering.

3. Database Integration (Oracle DB) :

The application uses Oracle Database to securely store user credentials and task information. The backend connects to the database using SQL queries. Major operations include:

[i] **Insert:** For registering new users and adding tasks.

[ii] **Select:** For displaying tasks based on the logged-in user.

[iii] **Delete:** For removing tasks when marked as complete or unnecessary.

4. Data Flow Overview :

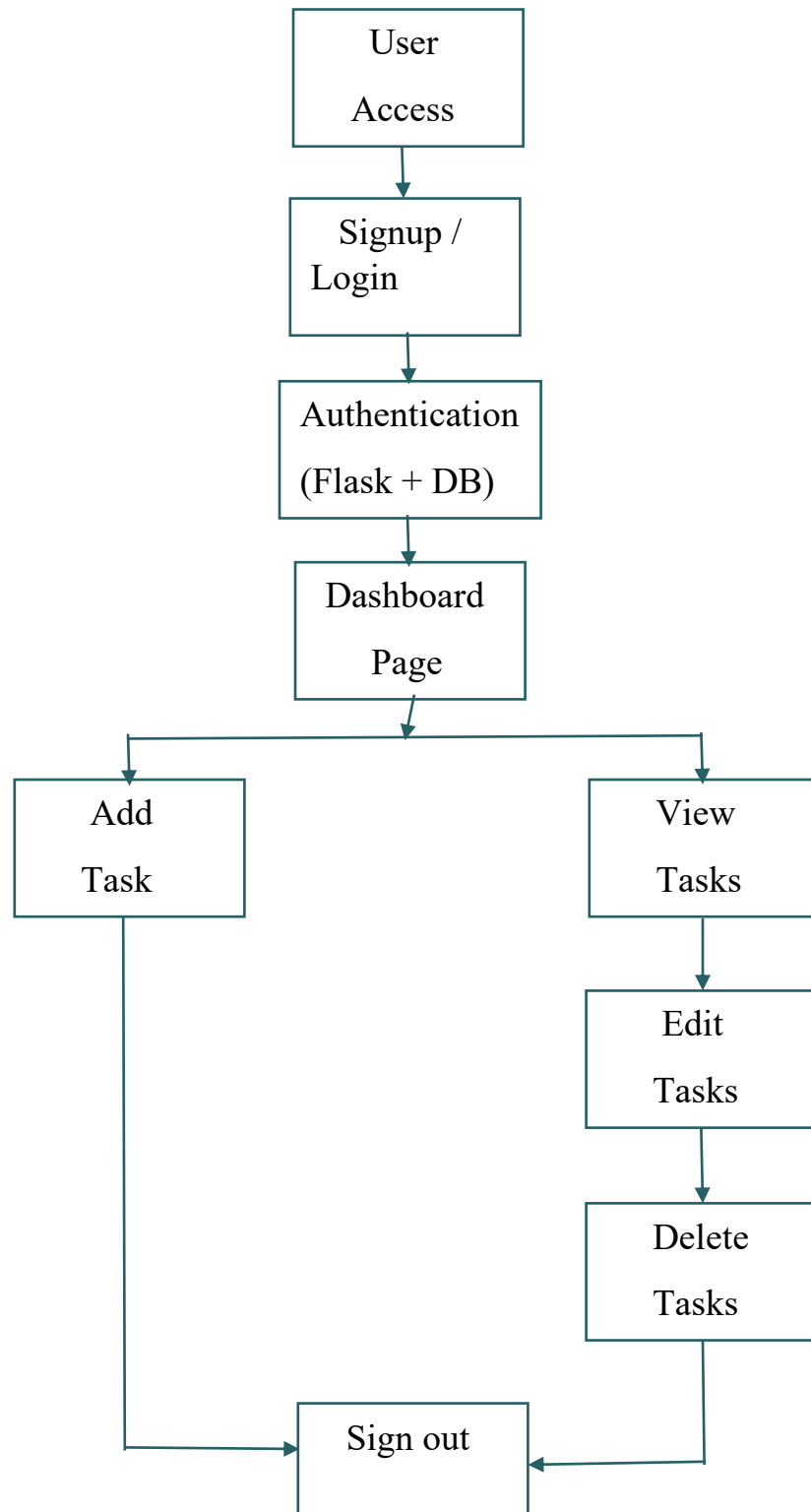
[i] User enters data on the frontend.

[ii] Frontend sends data to Flask backend via HTTP.

[iii] Flask processes the data and interacts with Oracle Database.

[iv] Flask receives results and updates the frontend accordingly.

● Flowchart ●



● Implementation ●

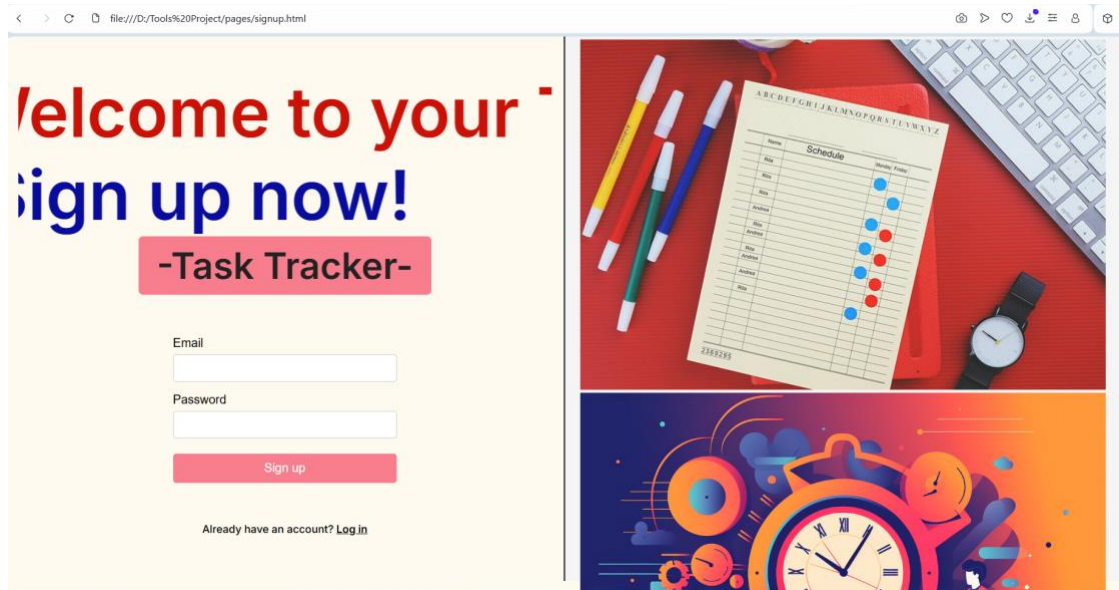
1.Code Overview :

The backend of the system is developed using Python Flask, where each route is mapped to a specific function like user registration, login, adding tasks, retrieving tasks, and deleting tasks.

- User authentication is handled with secure form validation and database checks.
- Task operations are managed using Flask routes that connect to Oracle Database using SQL queries.
- The frontend uses JavaScript for client-side validation and dynamic task display, while HTML and CSS handle the layout and styling.

2. User Interface views :

Signup Page: Where users create a new account.



Login Page: For user authentication.

file:///D:/Tools%20Project/pages/login.html

-Welcome back-

Email

Password

Log in

Don't have an account? [Sign up](#)

Dashboard: Displays task list, add task form, and sign out buttons.

file:///D:/Tools%20Project/pages/dashboard.html

My tasks

Add task Sign out

List Board

To do

Tools Exam Due on July 09, 2025

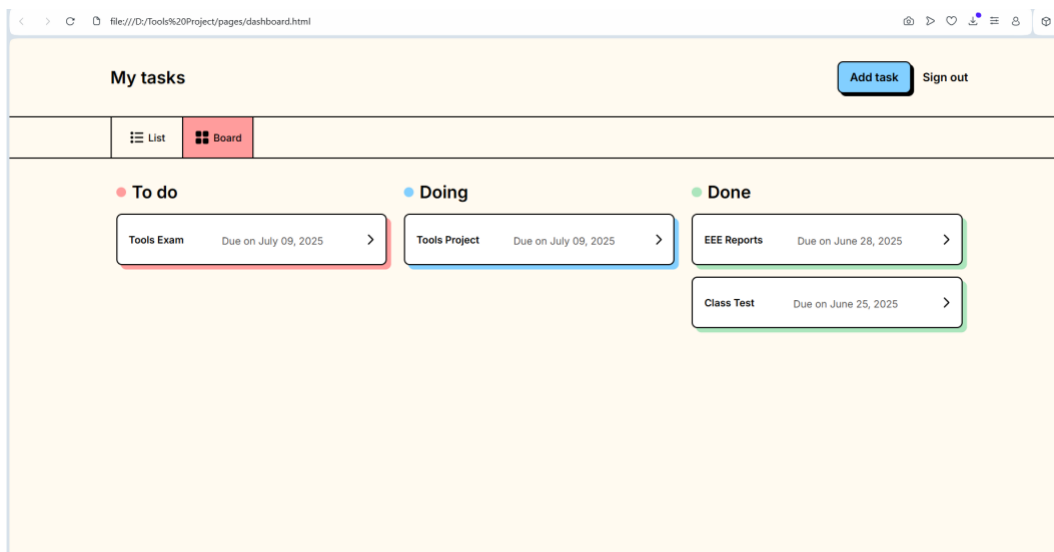
Doing

Tools Project Due on July 09, 2025

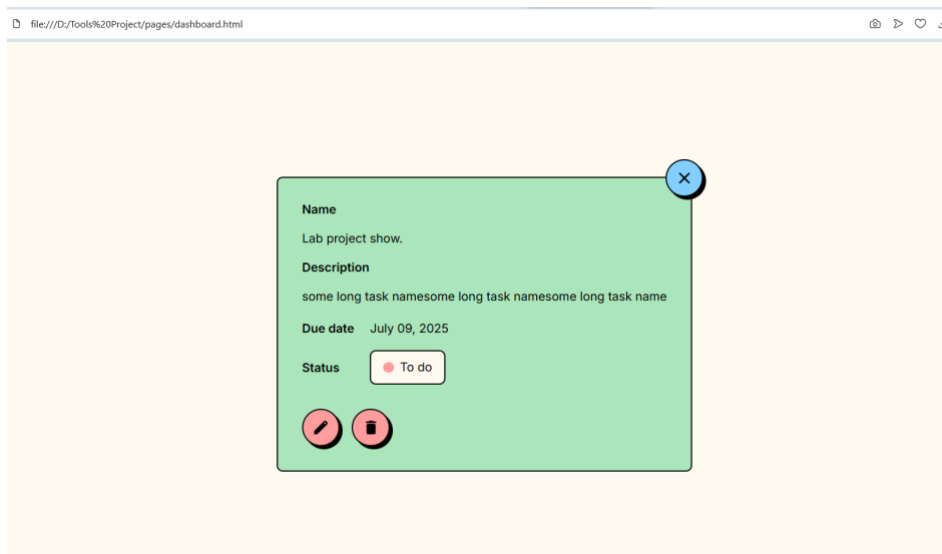
Done

EEE Reports Due on June 28, 2025

Class Test Due on June 25, 2025



Edit & Delete options :



3. System Workflow (Step-by-Step) :

Here's how the system works from the user's perspective:

- The user signs up or logs in via the web interface.
- On successful login, the user is redirected to the Dashboard.
- The user adds a task by entering task name, date, and day.
- The task is sent to the Flask backend, which stores it in Oracle DB.
- The updated task list is retrieved and displayed on the screen.
- If a user deletes a task, it is removed from the database and interface.

4. Database Table Structure :

The structure of the users table used in the Oracle Database -

USERS

Table

Data

Indexes

Model

Constraints

Grants

Statistics

UI Defaults

Triggers

Dependencies

SQL

Add Column

Modify Column

Rename Column

Drop Column

Rename

Copy

Drop

Truncate
















Create Lookup Table

Column Name	Data Type	Nullable	Default	Primary Key
EMAIL	VARCHAR2(100)	No	-	-
PASSWORD	VARCHAR2(100)	No	-	-
1 - 2				

USERS

TableDataIndexesModelConstraintsGrantsStatisticsUI DefaultsTriggersDependenciesSQL

QueryCount RowsInsert Row

EDIT	EMAIL	PASSWORD
	test@example.com	1234
	newuser@example.com	1234
	afia@gmail.com	1234
	a@gmail.com	1122
	ibnat@gmail.com	1200
	afiaibnat231516@gmail.com	test123
	she@gmail.com	3344
	and@gmail.com	6600
	any@gmail.com	0001
	you@gmail.com	0102
	shana@gmail.com	0203
	saj@gmail.com	0505
	y@gmail.com	0909
	sea@gmail.com	1100
	c231516@gmail.com	1000
row(s) 1 - 15 of 15		

Download

5. Flask Terminal Output:

The Flask development server running and logging backend activities.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\DELL> D:
PS D:\> cd "D:\Tools Project\py"
PS D:\Tools Project\py> python signup.py
* Serving Flask app 'signup'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 123-248-747
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\DELL> D:
PS D:\> cd "D:\Tools Project\py"
PS D:\Tools Project\py> python login.py
* Serving Flask app 'login'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5001
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 123-248-747
```

● Result & Analysis ●

The Task Tracker application was successfully developed and implemented. It allows users to register, log in, and manage their daily tasks by adding, viewing, and deleting tasks. The interface is user-friendly, responsive, and functions smoothly across modern browsers. Data is stored and retrieved from the Oracle database without errors. All intended features work as expected, ensuring a seamless experience for the user.

The system was tested with multiple user registrations, logins, and task operations. The test cases included:

- Valid and invalid login attempts
- Task addition with proper date and day
- Task deletion functionality
- Database updates after each operation

All tests passed successfully.

● Challenges ●

During the development of the Task Tracker application, a few challenges were encountered. Setting up the connection between Flask and Oracle Database required additional configuration and troubleshooting.

● Conclusion ●

The Task Tracker application was successfully developed to help users manage their daily tasks in a simple and efficient way. It provides essential features such as user registration, login, task addition, and deletion, all integrated with a secure Oracle database using Flask.

Through this project, I gained practical experience in web development, backend integration, and database connectivity. It also helped me improve problem-solving and debugging skills.

In the future, the system can be enhanced with additional features like task editing, reminder notifications, task prioritization, and a user-friendly interface.

● References ●

https://youtu.be/oHDSb6teQ_Q?si=6WJiS84lgYz1aHBz

<https://youtu.be/AQQi6nw3qjc?si=c4E5D5x7zp44y0DJ>

<https://youtu.be/lGAI9Z6egl0?si=MykG2pJYqUXncTYz>

<https://github.com/AhlamKhalid/Task-Management-System-HTML-CSS-Javascript>

<https://www.geeksforgeeks.org/javascript/task-scheduler-using-html-css-and-js/>

~Thank You~