**CST8285 Assignment 2 – Full Documentation**

**Project Title:** Task Management Web Application  
**Course:** CST8285 – Web Programming  
**Institution:** Algonquin College  
**Submission Date:** August 10, 2025

**Team Members**

* **Backend Developer:** Phil Maxwell-Mgbudem
* **Frontend Developer:** Ayo Adebayo

**1. Web Map (Sitemap**A screenshot of a computer

Description automatically generatedA screenshot of a computer screen

Description automatically generated

**Figure 1: Application Web Map (Sitemap Diagram)**

index.php (Landing Page)

├── pages/

│ ├── login.php (Login Form)

│ ├── register.php (User Registration)

│ └── tasks.php (User Dashboard)

│ ├── insert\_task.php (Add Task - PHP + AJAX)

│ ├── fetch\_tasks.php (Fetch Task List - PHP + AJAX)

│ └── delete\_task.php (Delete Task - PHP + AJAX)

├── scripts/

│ └── validate.js (Client-side JS form validation)

├── server/

│ ├── login\_user.php

│ ├── insert\_user.php

│ ├── connect.php

│ ├── logout.php

├── database/

│ └── create\_tables.sql (DB schema)

├── styles/

│ └── style.css

└── documentation/

├── readme.txt

├── summary.txt

└── flow.txt

**2. Wireframes**

**SCREENSHOTS OF YOUR WEBSITE UI**

A screenshot of a login form

Description automatically generatedA screenshot of a login form

Description automatically generated

**3. Code Attribution Table**

| **Team Member** | **Responsibilities** |
| --- | --- |
| **Phil Maxwell-Mgbudem** | Backend development (PHP scripts, login/registration, task DB ops, sessions) |
| **Favour Adebayo** | Frontend design (CSS, animations), JavaScript validation, testing, mobile view |

**4. Special Coding Considerations**

* JavaScript handles all **client-side form validation** (no HTML5 validation used)
* Passwords are stored using **password\_hash()** (secure hashing)
* **Prepared statements** used for all SQL queries to prevent SQL injection
* **Sessions** control dashboard access (authenticated users only)
* AJAX used to **add, fetch, and delete** tasks dynamically
* **CSRF protection** implemented by checking session state
* **Modern responsive UI** using Flexbox, Grid, and glass morphism styling

**5. System Execution Flow (User → PHP → DB → PHP → User)**

A. Registration Flow:

- User fills form → JS validates → PHP (insert\_user.php) → DB insert → session → dashboard

B. Login Flow:

- User submits login form → JS validation → PHP checks hash → session created → dashboard

C. Task Flow:

- Dashboard loads → AJAX calls fetch\_tasks.php → PHP queries DB → tasks returned as JSON

- Add Task: JS validates → insert\_task.php inserts to DB → AJAX updates view

- Delete Task: delete\_task.php removes from DB → AJAX updates UI

A screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generated

**6. README Overview**

**CST8285 Assignment 2 - Task Management Web Application**

**Team Information:**

* Lead Developer: Phil Maxwell-Mgbudem
* Review & Testing: Favour Adebayo
* Course: CST8285 – Web Programming
* Institution: Algonquin College

**Project Overview**

This is a full-featured task manager web app using **HTML**, **CSS**, **JavaScript**, **PHP (procedural)**, and **MySQL**. No frameworks used. It demonstrates real-world use of form validation, database integration, and user authentication.

**Key Features:**

* User registration & login system
* Secure password hashing
* Add/delete/manage tasks
* AJAX-based filtering/search
* Responsive UI with gradients and glass morphism
* Full JavaScript validation
* Session authentication
* SQL injection prevention

**Technology Stack:**

* **Frontend:** HTML5, CSS3, Vanilla JavaScript
* **Backend:** PHP (procedural)
* **Database:** MySQL with mysqli
* **Server:** XAMPP (Apache + MySQL)

**📁 Project File Structure:**

pgsql

CopyEdit

task-manager/

├── index.php

├── pages/

│ ├── register.php

│ ├── login.php

│ └── tasks.php

├── scripts/

│ └── validate.js

├── server/

│ ├── connect.php

│ ├── insert\_user.php

│ ├── login\_user.php

│ ├── insert\_task.php

│ ├── fetch\_tasks.php

│ ├── delete\_task.php

│ └── logout.php

├── styles/

│ └── style.css

├── database/

│ └── create\_tables.sql

└── documentation/

├── readme.txt

├── summary.txt

└── flow.txt

**⚙️ Installation Instructions:**

1. Install **XAMPP**
2. Start **Apache** and **MySQL**
3. Copy folder to /htdocs/
4. Import create\_tables.sql in **phpMyAdmin**
5. Open http://localhost/task-manager

**7. Summary of Application Functionality**

**User Registration**

* User enters name, email, password
* JS validates input in real-time
* Data submitted to insert\_user.php
* PHP hashes password, checks duplicates
* User is logged in and redirected to dashboard

**User Login**

* User enters email and password
* JS validation runs
* Data goes to login\_user.php
* PHP verifies hash, starts session
* Redirects on success or shows error

**📋 Task Dashboard**

* Tasks fetched via fetch\_tasks.php using AJAX
* Add task form submits via JS → insert\_task.php
* Delete task uses delete\_task.php via AJAX
* Filter and search use input + dropdowns to narrow list

**Database Structure**

**users Table:**

* id, name, email, password, created\_at
* Passwords hashed, email is unique

**tasks Table:**

* id, user\_id, title, description, priority, due\_date, created\_at
* Foreign key user\_id references users.id
* **Cascade delete**: if user is deleted, tasks are removed

**Security Summary**

* **Hashed passwords** (password\_hash)
* **Prepared statements** for DB ops
* **Sessions** protect all user pages
* **Client + server validation**
* **CSRF protection** via session checks

**UI Features**

* Glass morphism: blurred card containers
* Purple-to-blue gradient background
* Fully responsive for desktop & mobile
* Real-time validation + error messages
* Task priorities color-coded (e.g., red for High)

**8. System Flow Summary**

**Registration:**

1. JS validates inputs
2. PHP hashes and stores user
3. Session started → redirect to dashboard

**Login:**

1. JS checks form
2. PHP verifies password
3. Session → dashboard or error shown

**Tasks:**

1. Dashboard loads AJAX tasks
2. Add task → validated → PHP inserts → list refreshes
3. Delete task → PHP deletes → list updates