Vendor Management Tool – Project Documentation

---

1. Introduction

The Vendor Management Tool is a standalone client-side web application built using HTML, CSS, and vanilla JavaScript. The tool enables users to store, manage, and analyze vendor-related information using browser localStorage. It also supports CSV import/export for data portability, making it suitable for academic evaluations or lightweight business workflows.

2. Objectives

- Provide an intuitive UI to manage vendors.

- Allow CRUD (Create, Read, Update, Delete) operations.

- Enable CSV import/export functionality.

- Ensure data persistence using browser localStorage.

- Provide fully client-side architecture with no external libraries.

3. Methodology

3.1 Architecture Overview

The application is structured into modular components:

- index.html: UI layout and structure.

- styles/: Contains CSS for consistent visual design.

- js/: Contains modular JavaScript files for managing state, storage, CSV handling, and UI updates.

- assets/: Optional placeholder icons or images.

3.2 Data Storage Methodology

The tool uses `localStorage` for storing all vendor data during runtime:

- Data is serialized into JSON before storage.

- Updates trigger immediate localStorage synchronization.

- On page load, stored data is retrieved and rendered.

3.3 CSV Import/Export Methodology

CSV Import:

- User uploads a CSV file.

- The file is parsed using FileReader API.

- Each row is converted to a vendor object.

- Data is validated and merged into localStorage.

CSV Export:

- Vendor list is converted into CSV format.

- A downloadable blob is generated.

- Users can save their current dataset as a CSV file.

3.4 UI and Interaction Workflow

- Vendor form allows adding or editing entries.

- Vendor table dynamically renders all vendor records.

- Edit and delete buttons are bound to corresponding JS methods.

- Search and filtering are optional enhancements for quick navigation.

3.5 Error Handling Methodology

- Input validation for required fields.

- Try–catch blocks around CSV parsing.

- Fallback defaults if localStorage data is corrupted.

4. Conclusion

This Vendor Management Tool demonstrates practical use of client-side technologies to create a structured, persistent, and interactive data management system—ideal for technical evaluations, training, or small-scale administrative workflows.