**StudentMarksheet Application Document**

# Use Case: Student Marksheet Management System

**🛠 Technologies Used**

| **Technology** | **Purpose** |
| --- | --- |
| **C# (.NET 8)** | Backend development for Web API |
| **WPF (Windows Presentation Foundation)** | UI framework for the application |
| **Entity Framework Core** | ORM for database interactions |
| **SQL Server** | Database to store student marks |
| **JSON Server** | Mock database to simulate API responses |
| **Newtonsoft.Json** | JSON serialization & deserialization |
| **HttpClient** | API communication for fetching & posting data |

# Functional Overview

**📝 Objective**

The **StudentMarksheet Management System** is designed to:

1. Fetch student marksheets from an external API and display them in a data grid on the UI.
2. Allow users to upload new student marksheets via a CSV file.
3. Store and manage data in a SQL database, saving records in StudentMarks and Marks tables.
4. Ensure a seamless connection between the frontend (WPF UI), backend API, and the database.

# Detailed Functional Flow

# 1. Fetch Student Marksheet from an External API

**📌 Trigger:**

* When the user opens the application, it **calls the Web API** (http://localhost:5000/api/StudentMarkSheet/fetch) to fetch student records.

**🛠 Process:**

1. The **WPF application** sends a GET request to the Web API.
2. The **Web API (StudentMarkSheetUploadAPI)** fetches data from **JSON Server (http://localhost:3001/students)**.
3. The **fetched data is deserialized and displayed** in the **UI Grid**.

**💾 Stored Data:**

* The student records are **stored in memory** and **also saved in SQL tables**:
  + StudentMarks (stores student details)
  + Marks (stores individual subject marks)

# 2. Upload a New Student Marksheet (CSV Upload)

**📌 Trigger:**

* The user **clicks the "Upload CSV" button** on the WPF application to add new student records.

**🛠 Process:**

1. **The user selects a .csv file** containing student details.
2. **The WPF application reads the CSV file** and converts it into a **list of student objects**.
3. **New student data is sent to the Web API (POST request to http://localhost:5000/api/StudentMarkSheet/upload)**.
4. The **Web API processes the new records**:
   * If the **student exists**, it **ignores the record**.
   * If the **student is new**, it **adds the record to SQL Database (StudentMarks, Marks)**.
5. The Web API **forwards the new students to JSON Server** to **update the mock database**.

**💾 Stored Data:**

* The **new student data is inserted into the SQL database** (StudentMarks, Marks tables).
* The **updated student list is also added to JSON Server**.

# 3. Save & Retrieve Student Data in SQL Database

**📌 Tables Used:**

| **Table Name** | **Description** |
| --- | --- |
| StudentMarks | Stores student details (Roll Number, Name, Total Marks, Status, etc.) |
| Marks | Stores individual subject marks linked to StudentMarks |

**🛠 Process:**

* **Entity Framework Core (EF Core)** is used for database operations.
* Once the student marksheet is fetched from Json server, it is also saved in the database in parallel to displaying it on UI.
* Before inserting into the table, it checks if the student detail is already existing by comparing the roll number, If not, then proceeds to insert into the table.

# 4. Application & Deployment Structure

**📌 Application Layers**

| **Layer** | **Technology Used** | **Responsibilities** |
| --- | --- | --- |
| **Frontend (UI)** | **WPF (.NET)** | Displays student records, allows CSV upload |
| **Backend API** | **ASP.NET Core Web API** | Fetches & updates student records |
| **Database Layer** | **SQL Server + Entity Framework Core** | Stores student data |
| **Mock API Layer** | **JSON Server** | Simulates an external API response |

**📌 Deployment Setup**

| **Component** | **Runs On** | **Purpose** |
| --- | --- | --- |
| **WPF App (StudentMarkSheet.exe)** | Windows PC | UI for fetching & uploading marksheets |
| **Web API (StudentMarkSheetUploadAPI.dll)** | Localhost (http://localhost:5000) | Handles API requests from WPF App |
| **JSON Server (json-server --watch student\_marksheet\_updated.json)** | Localhost (http://localhost:3001) | Mock API to simulate student records |
| **SQL Server (StudentMarksDB)** | Local DB | Stores permanent student records |

# Summary

| **Functionality** | **Technology Used** | **Data Storage** |
| --- | --- | --- |
| **Fetch Student Data** | HttpClient (WPF → API) | JSON Server & SQL Database |
| **Display in UI Grid** | DataGrid (WPF UI) | ObservableCollection |
| **Upload CSV File** | OpenFileDialog (WPF) | SQL Database & JSON Server |
| **Store in Database** | EF Core (Web API) | StudentMarks, Marks Tables |
| **Mock API Simulation** | JSON Server | student\_marksheet\_updated.json |