**Web Attendance System Report**

**Chapter 1 – Introduction**

**1.1 Project Background**

In many small businesses and cafés, attendance tracking and work hour calculations are still conducted manually using handwritten or spreadsheet-based methods. These traditional approaches are prone to errors, inefficiencies, and administrative burden, especially when managing multiple employees and shifts.

To address this challenge, the project aims to automate the attendance process by developing a **web-based attendance management system**. This system eliminates the need for manual time tracking and enables real-time monitoring and record-keeping of employee work hours through an intuitive and accessible web interface.

**1.2 Objectives**

The primary goal of this project is to build a **web-based attendance system** that automates employee check-in/check-out operations and calculates total work time by day, month, or other timeframes.

Key objectives include:

* Providing a secure login system for both employees and managers.
* Automating the shift recording process with configurable start and end times.
* Enabling real-time attendance updates and historical data review.
* Allowing managers to export and review shift reports and logs.
* Offering local server hosting with limited network access for data privacy and reliability.

**1.3 Scope of the Project**

The Web Attendance System focuses on the automation of attendance tracking and employee time management. It supports two primary user roles—**Employees** and **Managers**—each with distinct functionalities.

The system operates on a **local network** (LAN), ensuring secure data access and real-time synchronization between client browsers and the local FastAPI server. It integrates **Google Sheets API** for automatic data export and real-time updates, enhancing accessibility for reporting and analysis.

However, the system is limited to LAN-based operation and does not yet support external remote access or cloud deployment.

**Chapter 2 – System Design and Implementation**

**2.1 System Overview**

The Web Attendance System is a full-stack web application designed to streamline attendance management through an automated, browser-based platform.

Employees can log in, start and end shifts, modify their work sessions, and leave notes for managerial review. Managers can monitor live attendance, access detailed logs, and export attendance records to Excel or Google Sheets.

The backend server handles authentication, shift tracking, and data storage, while the frontend offers a user-friendly interface for both employee and managerial operations.

**2.2 Technologies Used**

**2.2.1 Frontend (Web Application)**

The frontend was developed using:

* **HTML, CSS, JavaScript, and ReactJS** for responsive and dynamic UI.
* **QR code access** to simplify login through direct link scanning.
* **Real-time updates** using asynchronous requests to the FastAPI backend.

**2.2.2 Backend Server**

The backend was implemented using:

* **FastAPI (Python)** as the primary web framework for API endpoints.
* **JWT (JSON Web Tokens)** for secure authentication and role-based access control.
* **SQLite3** for local data storage, maintaining records of employees, shifts, and logs.
* **Google Sheets API** for real-time synchronization and automatic report generation.
* **Local hosting** through a LAN-based FastAPI server to ensure private access within the workplace.

**2.2.3 Data Management**

The system uses a hybrid data approach:

* **SQLite3** stores permanent records locally.
* **Google Sheets (via API)** acts as a cloud-linked mirror for real-time viewing and backup.
* **Cache files** store tokens and temporary session data to improve performance.

**2.3 System Architecture**

The system follows a **three-layer architecture** consisting of:

1. **Presentation Layer (Frontend)** – React-based web interface for employees and managers.
2. **Application Layer (Backend)** – FastAPI services handling authentication, attendance logic, and synchronization.
3. **Data Layer** – Local SQLite3 database and Google Sheet integration for data storage and reporting.

Core entities include:

* **User**: Stores login credentials and role (Employee or Manager).
* **Shift**: Contains start time, end time, and notes for each session.
* **Log**: Records actions and system updates.

This architecture ensures modularity, easy maintenance, and scalability for future cloud migration.

**2.4 Implementation**

**2.4.1 Development Process**

The project followed an **agile iterative process**:

1. **Planning & Design** – Defined user roles, requirements, and database schema.
2. **Backend Development** – Implemented FastAPI server with JWT authentication and shift management endpoints.
3. **Frontend Development** – Designed responsive web pages for both roles.
4. **Integration & Testing** – Connected frontend and backend using REST APIs and performed validation.
5. **Deployment** – Hosted on a local machine with LAN access.

**2.4.2 Key Implementation Features**

* **Automatic attendance tracking:** Shifts start and end based on user actions or time rules.
* **Editable shifts:** Employees can modify end times under valid conditions.
* **Real-time synchronization:** Google Sheets is updated instantly using the API.
* **Manager dashboard:** Allows export to Excel and viewing of live employee activity.
* **Security:** JWT tokens ensure that only authenticated users can access protected endpoints.

**2.4.3 Challenges Faced**

Key challenges included:

* Ensuring **network consistency** for real-time synchronization in a LAN environment.
* Managing **Google API authentication tokens** and refresh cycles.
* Implementing **accurate time tracking** across different browsers and systems.
* Designing a **clean UI** that remains accessible on both desktop and mobile browsers.

**Chapter 3 – Results and Evaluation**

**3.1 Application Features**

* **Employee Functions:**
  + Log in/out securely.
  + Start and end shifts with flexible scheduling.
  + Modify end times when necessary.
  + Add notes to each session.
  + View monthly attendance history.
  + Change account password.
* **Manager Functions:**
  + View all employee shifts in real time.
  + Export attendance data as Excel or Google Sheets.
  + Reset or delete employee accounts.
  + Review daily logs for tracking and auditing.
* **Server Functions:**
  + Operates within the local network only.
  + Saves attendance sheets monthly.
  + Updates Google Sheets automatically.

**3.2 Limitations**

* The system currently supports **LAN-only access**, with no internet or cloud connectivity.
* **Manual setup** of Google API credentials is required for synchronization.
* **No biometric or GPS tracking** integration yet.
* **Limited scalability** for large organizations without external database migration.

**3.3 Future Work**

Planned improvements include:

* Cloud deployment with global access.
* Role expansion for administrators and HR managers.
* Enhanced analytics dashboards for attendance trends.
* Integration with biometric or GPS-based verification.
* Mobile web optimization and offline mode support.

**Chapter 4 – Team Contributions**

*(Section intentionally left blank for customization by the team)*

**Chapter 5 – Conclusion**

The Web Attendance System effectively automates the employee attendance process, providing a simple, secure, and efficient solution for small businesses and cafés. By combining FastAPI, React, and Google Sheets integration, the system offers real-time data management and report generation with minimal infrastructure requirements.

The project successfully achieves its goal of reducing manual work, preventing calculation errors, and enhancing transparency between employees and managers. Future expansions such as cloud hosting, multi-role support, and AI-powered analytics can further elevate its practicality and scalability.

**References**

1. FastAPI Documentation – <https://fastapi.tiangolo.com/>
2. Google Sheets API – https://developers.google.com/sheets/api
3. React Documentation – https://react.dev/
4. JSON Web Token (JWT) RFC 7519 – Internet Engineering Task Force (2015)
5. SQLite3 Documentation – https://www.sqlite.org/docs.html