Software Development Life Cycle (SDLC)

Group 1: Attendance & Payroll Management System
QuickHire Services Ltd

Phase 1: Requirements & Analysis

The project began with gathering system requirements from the HR Director of QuickHire Services Ltd.

The goal was to build a **Python-based Attendance & Payroll System** for 75 field employees using:

- **CLI-based application**
- **REST API**
- **JSON files as the database**
- **Ngrok integration for external access**
- **CSV exporting for payroll and attendance reports**

After meeting with HR, the following key requirements were identified:

Employee Management

- Add, edit, remove, and list employees
- Store hourly rate, department, role, fixed allowance, and fixed deduction
- Save employee data into employees.json

Attendance Tracking

- Employees can sign in and sign out
- System automatically timestamps time-in/time-out
- Calculates daily hours and overtime beyond 8 hours
- Saves logs into attendance.json
- HR has access to correct attendance if needed

Payroll Generation

- Monthly salary computed using:
    - Regular hours × hourly rate
    - Overtime (beyond 8 hrs/day) × 1.3 rate multiplier
    - Add allowance, subtract deduction
- Generate payslips and monthly payroll summary
- Save payroll in payroll.json

Reporting

- Export CSV files for:
    - Monthly payroll
    - Attendance history
    - Daily summary

    - Overtime report
    - Individual payslips

API Requirements

- Endpoints for employees, attendance, and payroll
- External access via Ngrok

After analyzing all needs, JSON was chosen due to simplicity, and CLI was preferred for ease of use.

Phase 2: Design

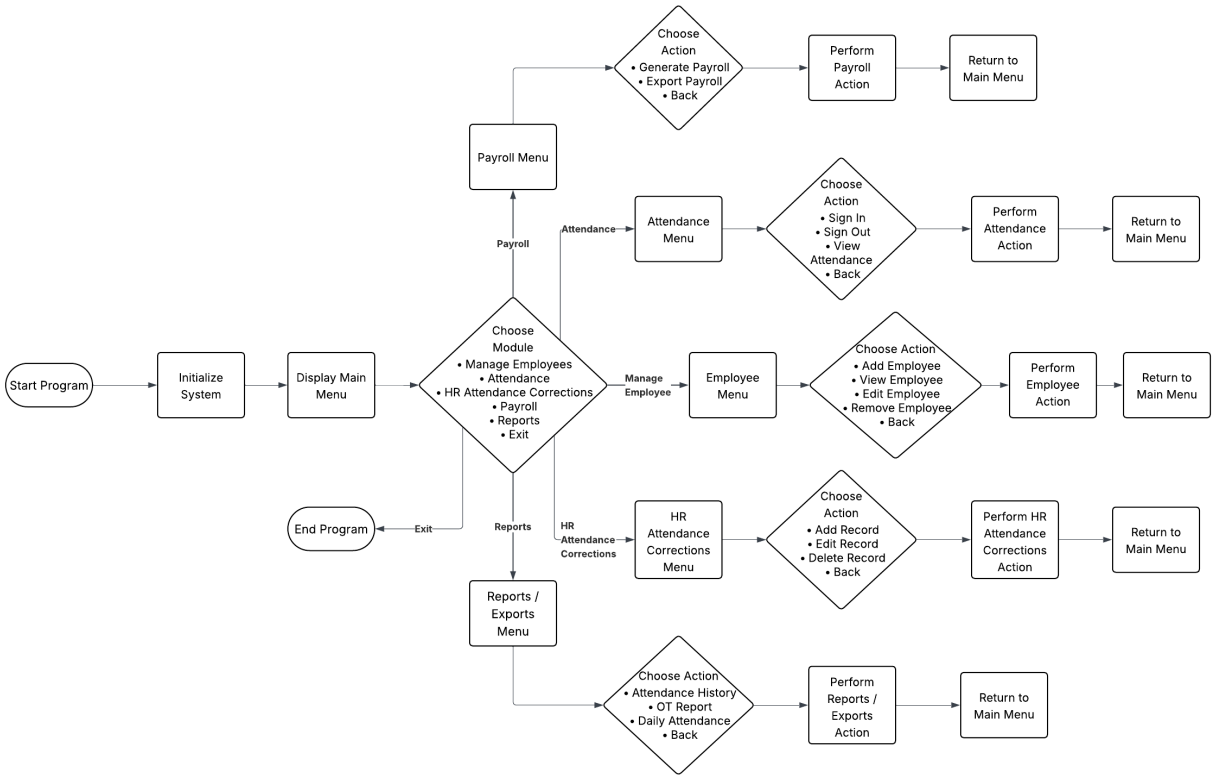
The system was designed using a modular and beginner-friendly structure with folders for database and exports:

project/
— main.py
— employees.py
— attendance.py
— payroll.py
— api.py
— utils.py
— db/
— exports/

Design Decisions

- JSON files act as a simple database
- utils.py provides reusable functions such as validation and export paths
- Each module handles a specific responsibility:
    - **employees.py** → employee operations
    - **attendance.py** → sign in/out & attendance logs
    - **payroll.py** → computation & payroll summary
- Export folder automatically created for CSV reports
- Data stored using simple lists and dictionaries

Flowchart:



Phase 3: Implementation

This phase focused on building the actual code based on the design.

Key Tasks Completed

✓ Implemented employee add/edit/remove functions

✓ Attendance sign-in/sign-out with automatic timestamps

✓ Working hour calculation with overtime separation

✓ Payroll generator with:

- Regular hours
- OT multiplier 1.3×
- Allowance & deduction

✓ CSV export functions using a shared exporter in utils.py

✓ REST API with CRUD endpoints

✓ Ngrok setup for external API access

Every module was tested individually to ensure they worked properly before connecting everything inside the CLI.

Phase 4: Testing

Testing was done through manual execution and checking JSON output files.

Tests Performed

Employee Management Testing

- Add employee → check JSON entry
- Edit employee → verify updated values
- Remove employee → ensure deletion is correct

Attendance Testing

- Sign-in logs correct date/time
- Sign-out calculates hours properly
- Overtime applied only when hours > 8

Payroll Testing

- Salary and overtime formula computed correctly
- Allowance and deduction reflected in total pay
- CSV exports stored in /exports folder
- Payroll JSON saved correctly

API Testing

- GET, POST, PUT, DELETE tested via Postman
- Verified Ngrok forwarding works

Bugs Found & Fixed

- CSV files not saving → fixed using get_export_path()
- Input validation improved
- Missing attendance record handling
- Minor formatting issues in reports

Phase 5: Deployment

Deployment steps included:

CLI Deployment

- System runs via main.py
- Requires Python installed

API Deployment

- Flask app hosted locally
- Ngrok creates a public URL for access
- HR or external staff can interact with the API remotely

Folder Setup

- /db created automatically for data
- /exports created on first export

The system is now deployable on any Windows or Linux machine with Python installed.

Phase 6: Maintenance

Ongoing maintenance will include:

Correcting and Updating Data

- Fixing attendance errors
- Adding new employees or updating rates
- Adjusting allowances/deductions

Code Improvements

- Adding more robust error handling
- Adding optional PDF export support

System Enhancements

Future upgrades may include:

- GUI version
- Monthly backup of JSON files
- User login system with roles (HR, employee)
- Automated email of payslips

The maintenance phase ensures the system remains functional, accurate, and aligned with HR needs.