Software Requirements Specification Template

CptS 322—Software Engineering

9 February 2005

The following annotated template shall be used to complete the Software Requirements Specification (SRS) assignment of WSU-TC CptS 322. The instructor must approve any modifications to the overall structure of this document.

**Template Usage:**

Text contained within angle brackets (‘<’, ‘>’) shall be replaced by your project-specific information and/or details. For example, <Project Name> will be replaced with either ‘Smart Home’ or ‘Sensor Network’.

Italicized text is included to briefly annotate the purpose of each section within this template. This text should not appear in the final version of your submitted SRS.

This cover page is not a part of the final template and should be removed before your SRS is submitted.

**Acknowledgements:**

Sections of this document are based upon the IEEE Guide to Software Requirements Specification (ANSI/IEEE Std. 830-1984). The SRS templates of Dr. Orest Pilskalns (WSU, Vancover) and Jack Hagemeister (WSU, Pullman) have also be used as guides in developing this template for the WSU-TC Spring 2005 CptS 322 course.

SHIFTSMART

Software Requirements Specification

<Version1.0>

<November 6,2024>

Meet Patel: Team leader

Manvi Gumber

Harin Reddy

Shyam Patel

Prepared for

WSU-TC CptS 322—Software Engineering Principles I

Instructor: A. David McKinnon, Ph.D.

Spring 2005

# Revision History:

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
| November 01,24 | Introduction and General Description | Manvi Gumber and Shyam Patel |  |
| November 02,24 | Specific Requirements | Meet Patel and Harin Reddy |  |
| November 04,2024 | Analysis Model | MMHS |  |
| November 6,2024 | SRS Template | Manvi Gumber and Shyam Patel |  |

# Document Approval:

The following Software Requirements Specification has been accepted and approved by the following:

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature** | **Printed Name** | **Title** | **Date** |
| Meet Patel | Meet Patel | Team Lead | November 3,2024 |
| Harin Reddy | Harin Reddy | Software Developer | November 3,2024 |
| Manvi Gumber | Manvi Gumber | UI/ UX Designer | November 3,2024 |
| Shyam Patel | Shyam Patel | Data Analyst | November 3,2024 |

**Table of Contents**

Revision History ii

Document Approval ii

1. Introduction 1

1.1 Purpose 1

1.2 Scope 1

1.3 Definitions, Acronyms, and Abbreviations 1

1.4 References 1

1.5 Overview 1

2. General Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Characteristics 2

2.4 General Constraints 2

2.5 Assumptions and Dependencies 2

3. Specific Requirements 2

3.1 External Interface Requirements 3

3.1.1 User Interfaces 3

3.1.2 Hardware Interfaces 3

3.1.3 Software Interfaces 3

3.1.4 Communications Interfaces 3

3.2 Functional Requirements 3

3.2.1 <Functional Requirement or Feature #1> 3

3.2.2 <Functional Requirement or Feature #2> 3

3.3 Use Cases 3

3.3.1 Use Case #1 3

3.3.2 Use Case #2 3

3.4 Classes / Objects 3

3.4.1 <Class / Object #1> 3

3.4.2 <Class / Object #2> 3

3.5 Non-Functional Requirements 4

3.5.1 Performance 4

3.5.2 Reliability 4

3.5.3 Availability 4

3.5.4 Security 4

3.5.5 Maintainability 4

3.5.6 Portability 4

3.6 Inverse Requirements 4

3.7 Design Constraints 4

3.8 Logical Database Requirements 4

3.9 Other Requirements 4

4. Analysis Models 4

4.1 Sequence Diagrams 5

4.3 Data Flow Diagrams (DFD) 5

4.2 State-Transition Diagrams (STD) 5

5. Change Management Process 5

A. Appendices 5

A.1 Appendix 1 5

A.2 Appendix 2 5

# 1. Introduction

## The Software Requirements Specification (SRS) for the ShiftSmart Scheduling Platform provides a comprehensive description of the system, its purpose, and its specifications. This document serves as a foundational guide for software engineers and other stakeholders involved in developing and implementing the platform, detailing functional and non-functional requirements. The ShiftSmart Scheduling Platform is designed to streamline workforce management, including scheduling, payroll processing, and compliance adherence.

## Purpose

The purpose of this SRS is to define the requirements for the ShiftSmart Scheduling Platform, intended to be used by software engineers, HR specialists, managers, payroll administrators, and relevant project stakeholders. This document outlines the expected functionalities and capabilities to ensure a clear understanding of the system's intended use and behavior.

## 1.2 Scope

1. **Software Product Identification**

The software product to be produced is named **ShiftSmart**, which includes the following components:

* **Automated Shift Scheduling Module**
* **Employee Management Module (CRUD Operations)**
* **Shift Management System for Real-Time Swaps and Call-offs**
* **Payroll and Invoice Generation System**
* **Notification System for Schedule Updates**
* **Analytics Dashboard for Performance and Cost Analysis**

1. **Product Functionality**

**What ShiftSmart will do:**

* **Automated Shift Scheduling:** The system will automatically generate employee shifts based on availability, preferences, and labor laws.
* **CRUD Operations for Employees:** The system will allow HR and managers to Create, Read, Update, and Delete employee records.
* **Real-Time Shift Swaps and Call-offs Management:** Employees will be able to swap shifts or call off work in real-time, with notifications sent to the relevant parties.
* **Payroll and Invoice Generation:** The system will calculate payroll based on hours worked and generate invoices accordingly.
* **Real-Time Notifications:** The system will send real-time notifications to employees regarding schedule updates and changes.
* **Analytics Dashboard:** The system will provide an analytics dashboard to assess employee performance, attendance, and labor costs, enabling data-driven decision-making.

**What ShiftSmart will not do:**

* The system will not manage external third-party integrations (such as external payroll services or HR platforms) unless explicitly stated.
* ShiftSmart will not perform any functions related to employee recruitment or onboarding beyond employee record management.

1. **Application of the Software**

ShiftSmart is designed to streamline the scheduling and management processe within organizations that require dynamic employee shift management. Its application includes:

1. **Benefits, Objectives, and Goals**

* **Efficiency in Shift Scheduling:** ShiftSmart aims to reduce the time spent on manual shift scheduling by automating the process based on employee availability, resulting in a decrease in scheduling conflicts.
* **Enhanced Employee Satisfaction:** By enabling real-time shift swaps and call-offs, employees have greater control over their work schedules, which can lead to improved job satisfaction and retention.
* **Accurate Payroll Processing:** The payroll and invoice generation capabilities will ensure timely and accurate compensation for employees, minimizing errors and administrative overhead.
* **Proactive Communication:** Real-time notifications will keep employees informed of schedule changes, reducing no-shows and increasing overall operational efficiency.
* **Data-Driven Insights:** The analytics dashboard will provide valuable insights into employee performance and labor costs, supporting management in making informed decisions to optimize workforce management.

1. **Consistency with Higher-Level Specifications**

The goals and functionalities described in this SRS are consistent with the overarching System Requirement Specification (SRS) outlined for the ShiftSmart project. They align with the project's mission to enhance workforce management efficiency through automation and data analytics.

## 1.3 Definitions, Acronyms, and Abbreviations

## ShiftSmart: The name of the project focused on automating employee shift scheduling.

## Sriven Security: The industry partner associated with the ShiftSmart project.

## SRS: Software Requirements Specification, a document that describes the intended purpose and environment for software under development.

## CRUD: Create, Read, Update, Delete - basic operations for managing data.

## UI: User Interface - the means by which a user interacts with a software application.

## UAT: User Acceptance Testing - a phase in the software development process where the software is tested in the real world by the intended audience.

## Payroll System: Software that manages the payment of salaries, including wage calculations and disbursement.

## Notification System: A system designed to send alerts and messages to users regarding important updates or changes.

## Analytics Dashboard: A user interface that visually displays key performance indicators and other important metrics related to the workforce.

## Integration: The process of combining various systems or software to work together seamlessly.

## Compliance: Adhering to laws, regulations, and guidelines relevant to the project and the industry.

## Dependency: A relationship between two tasks where one task relies on the completion of another.

## Risk Management: The process of identifying, assessing, and mitigating risks that could impact the project.

## Stakeholder: Any individual, group, or organization that can affect or be affected by a project.

## Gantt Chart: A type of bar chart that represents a project schedule, showing the start and finish dates of various elements.

## RAM: Responsibility Assignment Matrix - a tool used to identify the roles and responsibilities of team members for specific tasks in a project.

**1.3 Glossary :**

|  |  |  |
| --- | --- | --- |
| | **Term** | **Definition** | | --- | --- | |
| |  |  | | --- | --- | | **Shift Scheduling** | **The process of efficiently assigning workers to specific time slots or shifts, ensuring optimal staffing and compliance with labor regulations.** | |
| |  |  | | --- | --- | | **Worker Scheduling Application** | **The software designed to automate and streamline employee shift assignments, improving operational efficiency and employee satisfaction.** | |
| |  |  | | --- | --- | | **Employee Portal** | **A platform for employees to view their schedules, request time off, and receive notifications regarding shifts and other updates.** | |
| |  |  | | --- | --- | | **Payroll** | **The system used for tracking hours worked by employees and generating payslips or invoices accordingly.** | |
|  |
| |  |  | | --- | --- | | **Invoice** | **A financial document detailing hours worked and payment due, automatically generated by the system for payroll and client billing purposes.** | |
| |  |  | | --- | --- | | **Shift Swaps** | **The process by which employees can exchange or replace their assigned shifts with other available shifts or workers, subject to approval.** | |
| |  |  | | --- | --- | | **Real-time Alerts** | **Instant notifications sent to users (via email or SMS) regarding changes in schedules or other updates related to shift management.** | |
| |  |  | | --- | --- | | **Manager Dashboard** | **A management interface allowing supervisors to oversee schedules, assign shifts, and analyze analytics related to workforce performance.** | |
| |  |  | | --- | --- | | **Automated Notifications** | **Alerts generated by the system that inform employees and managers of updates, approvals, and changes to their schedules in real-time.** | |
| |  |  | | --- | --- | | **Advanced Analytics** | **The capability of the application to generate reports on workforce performance, attendance, labor costs, and scheduling efficiency to inform decision-making.** | |
| |  |  | | --- | --- | | **User-friendly Interface** | **A design feature that ensures the application is intuitive and easy to navigate for both employees and managers, enhancing user experience and adoption.** | |
| |  |  | | --- | --- | | **Stakeholder** | **Any individual or group with an interest in the project, including managers, employees, and clients, who will be affected by the scheduling application.** | |
| |  |  | | --- | --- | | **Company Owner** | **The individual responsible for the overall success of the company, setting strategic direction, overseeing operations, and making high-level decisions.** | |
| |  |  | | --- | --- | | **Scheduling Manager** | **The person overseeing employee shifts and scheduling, managing schedules, approving shift swaps, and monitoring attendance.** | |
| |  |  | | --- | --- | | **Employees** | **Workers who receive shift assignments via the platform, responsible for viewing assigned shifts, inputting availability, and requesting time off or shift swaps.** | |
| |  |  | | --- | --- | | **Payroll Administrators** | **Individuals responsible for processing payroll and client invoices, ensuring legal compliance and accurate financial management.** | |
| |  |  | | --- | --- | | **Security Company Management** | **Represents overall business interests and project success, ensuring the project aligns with company goals and providing final approval.** | |
| |  |  | | --- | --- | | **HR Department** | **Focuses on employee relations, payroll, and compliance needs, providing input on payroll management and ensuring adherence to labor regulations.** | |
| |  |  | | --- | --- | | **Development Team (MMHS)** | **Responsible for developing the Worker Scheduling platform, implementing features, integrating feedback from stakeholders, and ensuring system delivery.** | |
| |  |  | | --- | --- | | **End Users (Security Managers/Guards)** | **Individuals who will use the system daily, providing feedback during testing to ensure the system meets practical needs for shift scheduling and management.** | |

## 1.4 References

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version number** | **Title** | **Author** | **Date** | **Source / Location** |
| *1.0* | Project Plan | *MMHS* | *September 30th, 2024* | Sprint 2 |
| *1.0* | Project Vision Document | *MMHS* | *October 15,2024* | Sprint 1 |

## 1.5 Overview

// System Environment Diagram – You can refer the diagram in the zip file (It is a pen paper based diagram)

The Worker’s Scheduling Platform has 5 Actors (Employee, Manager, Payroll Administrator, Admin) and 4 module Employee Portal Module which is connected to Employee actor as they can use the platform to view their schedules, update availability, and request time off; Manager Portal where use the platform to view their schedules, update availability, and request time off. Unlike temporary workers, they often have more predictable assignments and may require consistent roles across shifts; Payroll Administrator accesses the payroll platform Module to process payroll based on the hours worked, attendance, and overtime recorded for each worker. This role ensures accurate compensation by pulling relevant shift and attendance data, simplifying the payroll process and Admin who has full access to the system, responsible for managing worker availability, creating and assigning shifts, and overseeing role and site management. This role ensures that each shift is adequately staffed and that workers receive timely notifications about their assignments.

## 2. General Description

## This section provides a high-level overview of the Worker Scheduling Application, highlighting factors that influence its design and requirements. It aims to clarify the application’s context, purpose, and primary constraints, serving as a foundation for understanding the specific requirements that follow.

## 2.1 Product Perspective

## The Worker Scheduling Application is designed to replace traditional manual scheduling methods with an automated, real-time scheduling system that integrates employee availability, shift assignments, payroll, and compliance management. Positioned as a comprehensive workforce management solution, this application supports industries with dynamic staffing needs, like security and retail. The application will integrate seamlessly with existing HR systems, enhancing current processes without overhauling them.

## This application serves as part of a broader suite of HR and workforce management tools, relying on the organization’s HR and payroll systems for employee data and payroll processing. It acts as a bridge, enabling improved communication between employees and managers, while providing a user-friendly platform for shift management and payroll generation.

## 2.2 Product Functions

## The primary functions of the Worker Scheduling Application include:

## Automated Shift Scheduling: Assigns shifts based on employee availability, business needs, and compliance with labor regulations, reducing manual errors and improving efficiency.

## Real-Time Notifications: Provides instant notifications for shift assignments, updates, approvals, and last-minute changes, ensuring effective communication.

## Employee Management: Allows CRUD (Create, Read, Update, Delete) operations on employee profiles, availability data, and shift preferences.

## Shift Swapping and Time-Off Management: Empowers employees to request shift swaps or time-off, pending manager approval, thus promoting flexibility.

## Payroll and Invoice Generation: Tracks working hours, generates payroll data, and invoices based on hours worked, reducing reliance on manual calculations.

## Advanced Analytics and Reporting: Offers insights into labor costs, shift efficiency, and attendance patterns, enabling data-driven decision-making.

## Compliance Tracking: Ensures schedules comply with local labor laws, including overtime, breaks, and rest periods.

## 2.3 User Characteristics

## The application will be used by different user groups, each with varying technical proficiencies and specific needs:

## Company Owners: Responsible for high-level decision-making and overseeing company operations. They require clear, high-level reports and summaries on labor costs, productivity, and efficiency.

## Scheduling Managers: Individuals managing the daily workforce operations, including assigning shifts, handling time-off requests, and approving shift swaps. They need an intuitive interface and real-time visibility into schedules and employee availability.

## Employees: Workers who view their schedules, submit availability, request time-off, and initiate shift swaps. They need a straightforward, mobile-friendly interface to access their schedules and request changes conveniently.

## Payroll Administrators: Handle payroll processing and invoicing. They require secure access to hours worked and payroll data to ensure accurate payments.

## Supervisors: They are responsible for overseeing and managing their assigned employees' schedules, approving shift swaps, monitoring attendance, and ensuring that employees adhere to their assigned shifts. They act as intermediaries between HR and employees, handling real-time scheduling needs.

## 2.4 General Constraints

## The development of the Worker Scheduling Application is subject to the following constraints:

## Project Timeline: The application must be developed within a set timeframe, which may limit the inclusion of additional features or enhancements.

## Compliance with Labor Laws: The system must comply with relevant labor regulations, requiring detailed design consideration for various jurisdictions.

## Network Reliability: As a cloud-based application, the system relies on stable internet connectivity to deliver real-time notifications and updates.

## Data Privacy and Security: Compliance with data protection regulations is mandatory, imposing strict requirements on the collection, storage, and handling of employee data.

## 2.5 Assumptions and Dependencies

**Assumptions:**

* **Employee Availability**: Employees will keep their availability updated in the system for accurate scheduling.
* **Manager Approvals**: Managers will promptly review and approve shift requests to avoid conflicts.
* **Payroll Integration**: The payroll system will integrate smoothly for accurate payroll processing.
* **Reliable Notifications**: Notifications will be delivered in real-time without major delays.
* **User Training**: Employees and managers are trained and comfortable using the application.
* **Stable Internet**: Users will have steady internet access for real-time updates and notifications.
* **Legal Compliance**: The system will adhere to current local labor laws.
* **Data Accuracy**: All data entered, including work hours, is accurate and up-to-date.
* **User Feedback**: Users will provide feedback during testing to enhance functionality.
* **Scalability**: The system can scale to accommodate growth in employees and locations.
* **Resource Availability**: Development resources, including personnel and hardware, will be available throughout the project’s lifecycle.
* **Stakeholder Participation**: Timely feedback from all stakeholders, including business managers and end-users, will be provided to ensure alignment with business needs.

**Dependencies:**

* **Accurate Data Input**: It is assumed that employee availability, payroll data, and other necessary information inputted into the system will be accurate and kept up to date.
* **User Training**: Users, especially managers and employees, will receive training to use the platform effectively.
* **Compatibility with Existing Systems**: The application will be compatible with current HR and payroll systems, ensuring seamless data integration.
* **Reliable Internet Access**: Users are expected to have access to stable internet to utilize cloud-based features effectively.
* **Notification Providers**: Real-time notifications depend on reliable third-party providers.
* **Employee Data Management**: Accurate and current employee data is essential for scheduling.
* **Labor Law Compliance**: The system must be updated for any changes in local labor laws.
  1. **Requirements Specification**

This section describes specific features of the software project. If desired, some requirements may be specified in the use-case format and listed in the Use Cases Section.

**3.1**

**Functional Requirement 1: Automated Shift Scheduling**

* Introduction: This feature enables automatic scheduling of employee shifts based on their availability and role requirements. It minimizes manual input and errors, ensuring that shifts are filled efficiently.
* Inputs: Employee availability required skills for shifts, and predefined shift time slots.
* Processing: The system processes employee availability against required shifts, assigns employees to shifts based on availability and suitability, and generates the schedule accordingly.
* Outputs: A completed schedule listing shifts with assigned employees, viewable by managers and employees.

### Functional Requirement 2: Shift Swap Scheduling

* Introduction: This feature allows employees to request shift swaps with colleagues and managers to approve or deny these requests.
* Inputs: Employee-initiated shift swap requests, including preferred swap shifts and reason for swapping.
* Processing: The system validates the swap request based on shift requirements and availability of both employees and notifies the manager for approval.
* Outputs: Updated schedule reflecting the approved swap (or, in case of denial, a notification sent to the employee with the reason).

### Functional Requirement 3: Payroll Generation

* Introduction: This feature calculates payroll based on hours worked, shifts rates, and overtime.
* Inputs: Shift schedules, hours worked per employee, hourly rates, and overtime rates (if applicable).
* Processing: The system calculates each employee’s payroll based on hours and rates and generates payroll reports.
* Outputs: Detailed payroll reports available to the payroll manager and summaries accessible to each employee.

### Functional Requirement 4: Availability Management

* Introduction: This feature allows employees to set and update their availability, helping managers in creating conflict-free schedules.
* Inputs: Employee availability entries (day and time preferences for shifts)
* Processing: The system stores and updates availability information for each employee.
* Outputs: An updated availability record that managers can access when assigning shifts.

### Functional Requirement 5: Real Time Notifications

* Introduction: Allows employees to report workplace incidents for HR/manager action.
* Inputs: Incident details (description, location, date).
* Processing: Logs the incident for further investigation.
* Outputs: Incident report for HR/manager follow-up.

### Functional Requirement 6: Leave Management Reporting

* Introduction: Manages employee leave requests and approvals.
* Inputs: Leave request with type, duration, and reason.
* Processing: Updates schedule and employee availability based on leave approval or denial.
* Outputs: Adjusted schedule and availability records.

### Functional Requirement 7: Reporting and Analytics

* Introduction: Provides analytics on scheduling, attendance, and payroll trends.
* Inputs: Historical data (shifts, attendance, payroll).
* Processing: Analyzes data to generate insights and metrics.
* Outputs: Dashboards and reports with key metrics for managers and HR.

### Functional Requirement 8: Employee Self-Service Portal

* Introduction: Allows employees to view schedules, update availability, and access payroll information.
* Inputs: Employee login and action selection.
* Processing: Retrieves and displays relevant data based on selected actions.
* Outputs: Employee-specific data (schedule, availability, payroll) in portal.

### Functional Requirement 9: Manager Dashboard

* Introduction: Provides a comprehensive overview for managers, showing shifts, availability, and scheduling insights.
* Inputs: Shift data, availability records, manager inputs.
* Processing: Displays real-time shift assignments and employee availability.
* Outputs: Interactive dashboard for efficient scheduling.

### Functional Requirement 10: Payroll Generation

* Introduction: This feature allows employees to set and update their availability, helping managers in creating conflict-free schedules.
* Inputs: Employee availability entries (day and time preferences for shifts)
* Processing: The system stores and updates availability information for each employee.
* Outputs: An updated availability record that managers can access when assigning shifts.

### Functional Requirement 11: Role-Based Access Control

* Introduction: Ensures data security by restricting access based on user roles.
* Inputs: User roles and permissions.
* Processing: Verifies role-based permissions before granting access.
* Outputs: Restricted access levels based on roles.

**3. Use Cases**

**3.1 General Use Cases**

**3.1.1 Use Case 1:** Update Profile

* **Actors:** Employee
* **Description:** Employee updates their personal profile information.
* **Preconditions:** Employee is logged in.
* **Postconditions:** Employee's profile is updated in the system.
* **Main Flow:**
  1. Employee navigates to the "Profile" section.
  2. Employee edits personal information (name, phone number, address, etc.).
  3. Employee submits the changes.
  4. System validates and saves the updated profile information.

**3.1.2 Use Case 2:** Submit Availability

* **Actors:** Employee
* **Description:** Employee submits their weekly availability for shifts.
* **Preconditions:** Employee is logged in.
* **Postconditions:** Employee's availability is recorded for the upcoming week.
* **Main Flow:**
  1. Employee navigates to the "Availability" section.
  2. Employee enters their weekly availability.
  3. Employee submits the availability.
  4. System saves the availability data.

**3.2 Employee Use Cases**

**3.2.1 Use Case 3:** Request Leave

* **Actors:** Employee
* **Description:** Employee submits a leave request.
* **Preconditions:** Employee is logged in.
* **Postconditions:** Leave request is recorded and awaits manager approval.
* **Main Flow:**
  1. Employee navigates to the "Leave Request" section.
  2. Employee fills in leave details (start date, end date, reason).
  3. Employee submits the request.
  4. System saves the request and notifies the manager.

**3.2.2 Use Case 4:** Submit Incident Report

* **Actors:** Employee
* **Description:** Employee submits an incident report related to work.
* **Preconditions:** Employee is logged in.
* **Postconditions:** Incident report is recorded in the system.
* **Main Flow:**
  1. Employee navigates to the "Incident Report" section.
  2. Employee enters details about the incident (description, date).
  3. Employee submits the report.
  4. System saves the report and notifies relevant parties (e.g., Manager).

**3.2.3 Use Case 5:** Get Pay Stubs

* **Actors:** Employee
* **Description:** Employee retrieves past pay stubs.
* **Preconditions:** Employee is logged in.
* **Postconditions:** Employee can view and download pay stubs.
* **Main Flow:**
  1. Employee navigates to the "Pay Stubs" section.
  2. Employee selects a pay period.
  3. System retrieves and displays pay stub details.
  4. Employee downloads the pay stub if needed.

**3.6.1 Use Case 6**: Dispute Payroll

* **Actors**: Employee
* **Description:** Employee disputes a payroll amount they believe isincorrect.
* **Preconditions:** Employee is logged in and has access to payroll information.
* **Postconditions:** Payroll dispute is recorded and forwarded for review.
* **Main Flow:**
  1. Employee navigates to the "Payroll" section.
  2. Employee selects the payroll entry they wish to dispute.
  3. Employee submits a dispute form with details (reason for dispute).
  4. System records the dispute and notifies the payroll manager.

**3.7.1 Use Case 7:** Send Chat Message

* **Actors:** Employee
* **Description:** Employee sends a message in a chat group.
* **Preconditions:** Employee is logged in and part of a chat group.
* **Postconditions:** Message is sent and stored in the chat group.
* **Main Flow:**
  1. Employee selects a chat group to communicate in.
  2. Employee types the message in the chat input box.
  3. Employee sends the message.
  4. System saves the message and displays it in the chat for all members.

**3.8.1 Use Case 8:** View Notifications

* **Actors:** Employee
* **Description:** Employee views their notifications for updates and alerts.
* **Preconditions:** Employee is logged in.
* **Postconditions:** Employee can see a list of notifications.
* **Main Flow:**
  1. Employee navigates to the "Notifications" section.
  2. System retrieves and displays all relevant notifications.
  3. Employee can click on notifications for more details if needed.

**3.9.1 Use Case 9:** Claim Open Shift

* **Actors:** Employee
* **Description:** Employee claims an open shift that is available for picking.
* **Preconditions:** Employee is logged in and there are open shifts available.
* **Postconditions:** Employee is assigned to the claimed shift.
* **Main Flow:**
  1. Employee navigates to the "Available Shifts" section.
  2. Employee reviews open shifts and selects one to claim.
  3. Employee confirms the claim.
  4. System updates the shift assignment and notifies the manager.

**3.10.1 Use Case 10:** Retrieve Shift History

* **Actors:** Employee
* **Description:** Employee retrieves a record of all assigned past and upcoming shifts.
* **Preconditions:** Employee is logged in.
* **Postconditions:** Employee can view a list of all shifts, with details for each shift.
* **Main Flow:**
  1. Employee navigates to the "Shift History" or "My Shifts" section.
  2. System retrieves all shifts associated with the employee, including past and upcoming shifts.
  3. System displays the list of shifts, including details such as date, time, site, and assigned manager.
  4. Employee can select individual shifts for more details if needed (e.g., site location, shift start and end times, total hours).

**3.3 HR Use Cases**

**3.3.1 Use Case 11:** Add Employee

* **Actors:** HR
* **Description:** HR adds a new employee to the system.
* **Preconditions:** HR is logged in and has the necessary permissions.
* **Postconditions:** New employee's details are stored in the system.
* **Main Flow:**
  1. HR navigates to the "Add Employee" section.
  2. HR enters the new employee’s details (name, email, phone, role, etc.).
  3. HR submits the information.
  4. System validates and saves the new employee details.

**3.3.2 Use Case 12:** View Employee

* **Actors:** HR
* **Description:** HR views details of an existing employee.
* **Preconditions:** HR is logged in and has the necessary permissions.
* **Postconditions:** HR can view the profile of the selected employee.
* **Main Flow:**
  1. HR navigates to the "Employee Directory" section.
  2. HR selects an employee from the list.
  3. System retrieves and displays the employee’s profile.

**3.13.1 Use Case 13:** Delete Employee

* **Actors:** HR
* **Description:** HR deletes an employee's record from the system.
* **Preconditions:** HR is logged in and has the necessary permissions.
* **Postconditions:** Employee’s record is removed from the system.
* **Main Flow:**
  1. HR navigates to the "Employee Directory" or "View Employees" section.
  2. HR searches for or selects the employee to be deleted.
  3. HR initiates the delete action.
  4. System prompts HR to confirm the deletion.
  5. Upon confirmation, the system removes the employee’s record from the database.
  6. HR receives confirmation that the employee has been deleted.

**3.14.1 Use Case 14:** Assign Employee to a Manager

* **Actors:** HR
* **Description:** HR assigns an employee to a specific manager forreporting purposes.
* **Preconditions:** HR is logged in and has the necessary permissions; employee and manager profiles exist in the system.
* **Postconditions:** Employee is assigned to the selected manager.
* **Main Flow:**
  1. HR navigates to the "Assign Manager" section or accesses an employee's profile.
  2. HR selects an employee and then selects a manager from a list of available managers.
  3. HR submits the assignment.
  4. System updates the employee’s record to reflect the assigned manager.
  5. HR receives confirmation that the assignment has been successfully saved.

**3.15.1 Use Case 15:** Process Payrolls

* **Actors:** Payroll Manager
* **Description:** Payroll Manager processes payrolls for employeesbased on their work hours, leaves, and other factors.
* **Preconditions:** Payroll Manager is logged in and has the necessary permissions; payroll data for the employees (e.g., hours worked, leave taken) is available in the system.
* **Postconditions:** Payroll records are processed, and employees' pay amounts are calculated and stored in the system.
* **Main Flow:**
  1. Payroll Manager navigates to the "Process Payrolls" section.
  2. System displays a list of employees with their payroll information, including hours worked, pay rate, and any deductions or bonuses.
  3. Payroll Manager reviews and verifies each employee's payroll data.
  4. Payroll Manager initiates the payroll processing action.
  5. System calculates each employee's final pay amount and updates the payroll records.
  6. Payroll Manager receives confirmation that payrolls have been processed successfully.

**3.16.1 Use Case 16: View Payroll History**

* **Actors: Payroll Manager**
* **Description:** Payroll Manager views the payroll history for all employees, including past payroll records and payment statuses.
* **Preconditions:** Payroll Manager is logged in and has the necessary permissions; processed payroll records are available in the system.
* **Postconditions:** Payroll Manager can view historical payroll records.
* **Main Flow:**
  1. Payroll Manager navigates to the "Payroll History" section.
  2. System displays a list of historical payroll records for all employees, including details such as employee name, pay period, hours worked, deductions, and payment status.
  3. Payroll Manager can filter or search for specific employees or pay periods.
  4. Payroll Manager views details of individual payroll records as needed.

**3.17.1 Use Case 17:** Respond to Payroll Disputes

* **Actors:** Payroll Manager, Employee
* **Description:** Payroll Manager reviews and responds to payrolldisputes raised by employees regarding their pay or deductions.
* **Preconditions:** Payroll Manager is logged in and has the necessary permissions; employees have submitted payroll disputes.
* **Postconditions:** Disputes are reviewed and resolved, with employees receiving notifications of the outcome.
* **Main Flow:**
  1. Payroll Manager navigates to the "Payroll Disputes" section.
  2. System displays a list of active payroll disputes submitted by employees.
  3. Payroll Manager selects a dispute to review, viewing details such as employee name, disputed payroll period, reason for dispute, and any supporting documents provided.
  4. Payroll Manager investigates the dispute, makes necessary adjustments if required, and documents the resolution.
  5. Payroll Manager marks the dispute as resolved.
  6. System notifies the employee of the dispute outcome.

**3.4 Manager Use Cases**

**3.4.1 Use Case 18:** Approve Leave Requests

* **Actors:** Manager
* **Description:** Manager reviews and approves or rejects leave requests.
* **Preconditions:** Manager is logged in and has the necessary permissions.
* **Postconditions:** Leave request is marked as approved or rejected.
* **Main Flow:**
  1. Manager navigates to the "Leave Requests" section.
  2. Manager reviews leave details.
  3. Manager approves or rejects the request.
  4. System updates the leave status and notifies the employee.

**3.5.2 Use Case 19:** View Employee Availability

* **Actors:** Manager
* **Description:** Manager views the availability of employees for scheduling shifts.
* **Preconditions:** Manager is logged in and has necessarypermissions; employees have submitted their availability.
* **Postconditions:** Manager can see a list of available employees for specific dates or times.
* **Main Flow:**
  1. Manager navigates to the "Employee Availability" section.
  2. Manager selects a date or shift time.
  3. System displays a list of employees who are available during the selected time.
  4. Manager views availability details for scheduling purposes.

**3.5.3 Use Case #20:** Create Sites

* **Actors:** Manager
* **Description:** Manager creates a new site location in the system.
* **Preconditions**: Manager is logged in and has necessary permissions.
* **Postconditions**:New site is created and saved in the system.
* **Main Flow:**
  1. Manager navigates to the "Manage Sites" section.
  2. Manager enters the site details (e.g., location, contact info).
  3. Manager submits the information.
  4. System saves the new site in the database.

**3.5.4 Use Case 21:** Add Shifts

* **Actors:** Manager
* **Description:** Manager adds shifts for specific dates and times at assigned sites.
* **Preconditions:** Manager is logged in and has necessarypermissions; site information is available.
* **Postconditions:** New shifts are created and stored in the system.
* **Main Flow:**
  1. Manager navigates to the "Manage Shifts" section.
  2. Manager selects a site and enters shift details (e.g., date, start time, end time).
  3. Manager submits the shift information.
  4. System saves the new shift in the database.

**3.5.5 Use Case 22:** Assign Shifts to Employees

* **Actors:** Manager
* **Description:** Manager assigns employees to specific shifts.
* **Preconditions:** Manager is logged in and has necessary permissions; shifts and employee availability data are in the system.
* **Postconditions:** Employees are assigned to shifts and receive notifications.
* **Main Flow:**
  1. Manager navigates to the "Assign Shifts" section.
  2. Manager selects a shift and views the list of available employees.
  3. Manager assigns one or more employees to the shift.
  4. System updates the shift assignments and sends notifications to the assigned employees.

**3.5.6 Use Case 23:** Get Employee Reports

* **Actors:** Manager
* **Description:** Manager views reports on attendance or incidents.
* **Preconditions**: Manager is logged in and has necessarypermissions; reports have been submitted by employees or system-generated.
* **Postconditions:** Manager can view detailed reports for employee management.
* **Main Flow:**
  1. Manager navigates to the "Employee Reports" section.
  2. System displays a list of available reports.
  3. Manager selects a report to view details.
  4. System shows the report, including details like incidents or performance metrics.

**3.5.7 Use Case 24: Create Chat Group**

* **Actors:** Manager
* **Description:** Manager creates a chat group for employees at specific sites to facilitate communication.
* **Preconditions:** Manager is logged in and has necessary permissions.
* **Postconditions:** New chat group is created, and members are added to the group.
* **Main Flow:**
  1. Manager navigates to the "Chat Groups" section.
  2. Manager selects a site and enters details for the group (e.g., group name, purpose).
  3. Manager selects employees to add to the group.
  4. System creates the chat group and adds selected members.

**3.5.8 Use Case 25: Open Shift for Claim**

* **Actors: Manager, Employee**
* **Description:** Manager opens a shift to allow employees to claim it voluntarily.
* **Preconditions:** Manager is logged in and has necessarypermissions; shift is created but unassigned.
* **Postconditions:** Shift is marked as open, and employees receive notifications.
* **Main Flow:**
  1. Manager navigates to the "Manage Shifts" section.
  2. Manager selects an unassigned shift and marks it as "Open for Claim."
  3. System updates the shift status and notifies available employees of the open shift.
  4. Employee claims the shift, and system updates the assignment if approved.

## 3.5 Non-Functional Requirements

## ->Non-Functional Requirements

The non-functional requirements for a system are typically constraints on the functional requirements – that is, not what the system does, but how it does it (e.g. how quickly, how efficiently, how easily from the user’s perspective, etc.).

### Non-functional requirements may exist for any of the following attributes – Performance, Reliability, Availability, Security, Maintainability, Portability.

### 4.1 Performance

* Requirement: The system should process 95% of transactions (such as shift scheduling updates, payroll calculations, and shift swap approvals) in less than one second.
* Justification: High Performance is crucial to provide a smooth and responsive experience for users, especially during peak scheduling times.

### 4.2 Reliability

* Requirement: The system should achieve at least 99.9% uptime, ensuring downtime does not exceed one minute per day.
* Justification: Since the application is essential for managing work schedules and payroll, high reliability is necessary to minimize disruption to employees and managers.

### 4.3 Availability

* Requirement: The system must be available 24/7, allowing managers and employees to access their schedules, request swaps, or update availability at any time.
* Justification: Flexibility in access ensures that employees can manage their schedules and availability outside of working hours, enhancing usability and satisfaction.

### 4.4 Security

* Requirement: The application should support multi-factor authentication (MFA) for all users and encrypt sensitive data (e.g., payroll details and employee information) both at rest and in transit.
* Justification: Security is essential for protecting sensitive data and ensuring compliance with data protection regulations, which is particularly important for payroll and personal information.

### 4.5 Maintainability

* Requirement: The codebase should be modular, with clearly documented components and standardized naming conventions to facilitate future maintenance and updates.
* Justification: A maintainable design ensures that the system can adapt to future requirements, such as additional scheduling features or changes in payroll calculations.

### 4.6 Portability

* Requirement: The application should be accessible on multiple platforms, including web browsers and mobile devices, without requiring specific configurations or software.
* Justification: Supporting multiple platforms enables broader accessibility, allowing employees and managers to access the system on the go, which is essential for remote work environments.

Inputs: Employee availability entries (day and time preferences for

Often these requirements must be achieved at a system-wide level rather than at a unit level. *Inputs: Employee-initiated shift swap requests, including preferred swap shifts and reason for swapping.*

95% of transaction shall be processed in less than a second, system downtime may not exceed 1 minute per day, etc).

## 5.0 Logical Database Requirements

Will a database be used? If so, what logical requirements exist for data formats, storage capabilities, data retention, data integrity, etc?

### 5.1 Database Usage:

* Yes, a database will be used to store all the necessary information for your Worker Scheduling Application.
* The system will require both relational (for structured data like employee details, shifts, payroll) and non-relational (for unstructured data like communications, logs, etc.) storage.

### 5.2 Data Formats

Relational Data:

* Tables for employees, shifts, sites, payroll, and shift requests will follow a structured format (e.g., SQL tables with predefined columns).
* Fields like employee name, role, and shift time should have specific data types (e.g., VARCHAR for text, DATETIME for time-related fields).

Non-relational Data:

* Logs and communication data could be stored in a document-oriented format (e.g., MongoDB), where each document might represent a message or log entry with dynamic fields.

### 5.3 Storage Capabilities

* The system needs to support scalability as data grows, especially with employee records, shifts, and logs.
* PostgreSQL can be used for structured data (employee details, shift data) due to its robust support for relational integrity, complex queries, and scalability.
* MongoDB will handle unstructured data like communications, notifications, and logs.

### 5.4 Data Retention

Retention Policies:

* Employee data (name, role, availability) should be retained indefinitely.
* Shift data may need to be archived periodically (e.g., after one year), as long-term shift history might not be needed.
* Communication and logs can be retained for a specific period (e.g., 6 months) or based on operational requirements.
* The system should include automated retention features that can archive or delete data after specific intervals to comply with internal policies or legal regulations.

### 5.5 Data Integrity

* Referential Integrity: The database should maintain referential integrity, ensuring that all relationships between entities (e.g., employee shifts, payroll data) are consistent and accurate.
* ACID Transactions: Ensure that database operations for shifts, employee scheduling, and payroll generation are transactional, meaning they are atomic, consistent, isolated, and durable.
* Data Validation: Before insertion or modification, data should be validated to ensure it meets format and business rules (e.g., shift start time cannot be later than the end time).

# 4. Analysis Models

List all analysis models used in developing specific requirements previously given in this SRS. Each model should include an introduction and a narrative description. Furthermore, each model should be traceable the SRS’s requirements.

**// Note - All the diagrams are placed in the ZIP file due to spacing and image quality issue, the names are as follow.**

## 4.1 Sequence Diagrams

## 4.2.1 Activity Diagram

## 4.2.2 Activity Diagram – 2

## 4.2.3 Activity Diagram – 3

## 4.3 Data Flow Diagrams (DFD)

## 4.4 Data Flow Diagrams (DFD)

## 4.5.1 Entity Relationship Diagram – Class based

## 4.5.2 Entity Relationship Diagram Main

## 4.6 Use Case Diagram names as MHHS-1

## 4.7 UML Diagram

# 5. Change Management Process

Identify and describe the process that will be used to update the SRS, as needed, when project scope or requirements change. Who can submit changes and by what means, and how will these changes be approved.

# A. Appendices

### A.1 Appendix 1 – Conceptual Documents

This appendix would contain initial conceptual ideas or drafts for your **ShiftSmart** system. It could include:

* **Initial mockups** or **wireframes** of the system interface, such as:
  + Dashboard for HR, Managers, and Employees.
  + Employee availability input screens.
  + Shift assignment screens for managers.
* **Initial flow diagrams** or **process flows** showing the interactions between HR, Managers, Payroll, and Employees.
* **User stories** or **early use cases** describing how each user role (HR, Manager, Employee) interacts with the system.

### A.2 Appendix 2 – Marketing Materials (If applicable)

If applicable, you may include documents related to the **market positioning** or **future vision** of the system. Examples:

* **Product overview** or **feature highlights** that summarize the benefits and key features of the ShiftSmart system.
* **Competitive analysis** comparing ShiftSmart to other scheduling systems in the market.
* **Target audience or user personas** that detail the characteristics of the primary users (HR, Managers, Employees).

### A.3 Appendix 3 – Meeting Minutes

Include the minutes from meetings with your industry partner (**Sriven Security**), professors, or internal team discussions:

* **Meeting minutes** from discussions with Sriven Security, detailing feedback on the project requirements.
* **Feedback and decisions** regarding features like the optional notification system or the integration of payroll management.
* **Action items** from meetings about how to improve the system or adjust requirements.

### A.4 Appendix 4 – Glossary of Terms

If your project uses technical terms that are important to understand, include definitions here. For example:

* **HR**: Human Resources – the department responsible for managing employee records.
* **Shift Assignment**: The process of assigning employees to shifts at specific work sites.
* **Availability**: The time slots when an employee is available to work.
* **Payroll Data**: Information related to employee pay, such as hours worked, leave taken, and salary.

### A.5 Appendix 5 – References

Provide a list of external documents, standards, or guidelines referenced throughout the SRS:

* **Legal requirements**: Any regulations concerning employee scheduling, labor laws, or data privacy.
* **Technology documentation**: Links or references to frameworks, tools, or platforms (e.g., Node.js, MongoDB, Express.js) used in the development of the ShiftSmart system.
* **Industry standards**: Any industry-specific guidelines for scheduling or HR management.

### A.6 Appendix 6 – Diagrams and Visuals

Include detailed **diagrams** and **visuals** that are referenced in the main SRS but are too large to include in the main body. For example:

* **Data Flow Diagrams (DFD)**: Include your **Level 1** and **Level 0 DFD** diagrams showing the interactions between different user roles (HR, Manager, Employee) and the system.
* **Entity-Relationship (ER) Diagram**: Show the relationships between entities in your database (e.g., Employee, Shift, Site, Assignment).
* **System Architecture Diagram**: Include a diagram showing the system’s architecture, how different components (database, front end, back end) interact.