Software Requirements Specification (SRS)

Worker Management System

Team OG

November 2025

Contents

1 Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) is to provide a clear, complete, and detailed description of the **Worker Management System (WMS)**. This document serves as the authoritative reference for all stakeholders—including customers, administrators, workers, developers, and evaluators—regarding the system's expected behavior, constraints, and functionalities.

The SRS outlines functional and non-functional requirements and provides diagrams to support understanding of the system's interactions. It establishes a foundation for the design, implementation, testing, and maintenance of the application.

1.2 Scope

The Worker Management System is a terminal-based platform designed to connect customers requesting home services (such as cleaning or household chores) with available workers who possess the required skills. The system consists of three separate applications: a Customer Client (C++), an Admin Console (Java), and a Worker Console (Java). These independently running applications communicate through shared JSON storage files.

The system supports:

- Customer registration, authentication, and service request creation.
- Scheduling of services with time constraints (minimum one-hour advance).
- Worker filtering based on locality, skills, gender preference, availability, and conflict with worker calendar.
- Admin review, assignment, and rejection of service requests.
- Worker review of assigned tasks, acceptance/rejection, and profile management.

1.3 Definitions

- **Immediate Request:** A service that must be allocated immediately and occurs shortly after booking.
- **Scheduled Request:** A service that the customer chooses to schedule at a future time, with at least one hour lead time.
- **Worker Calendar:** A structured list of existing job time-slots assigned to a worker.
- **Service Status Workflow:** The system uses the following natural-language stages:
  Waiting for Admin Review → Waiting for Worker Confirmation → Assigned → Completed
  Alternative outcomes include:
  Unable to Assign (no suitable workers available),
  Rejected by Worker.

1.4 Overview

This document begins with an overall description of system characteristics and continues with detailed functional requirements, external interface descriptions, use-case diagrams, and supporting non-functional requirements.

2 Overall Description

2.1 Product Perspective

The Worker Management System uses a modular architecture consisting of three separate terminal applications. Each application functions as an independent client, reading from and writing to JSON-based persistent storage files. There is no central server; instead, synchronization occurs through sequential writes and manual refresh by the user.

This design enables low complexity while still maintaining clear separation between user roles. Customers initiate service requests, admins manage those requests, and workers handle assigned tasks. JSON files serve as the central data repository, simulating a lightweight database.

2.2 Product Functions

The key high-level functions of the system include:

- **Handling Service Requests:** Customers can create immediate or scheduled service requests. Each request includes selected tasks, locality, gender preference, and requested time slots.
- **Worker Filtering:** The admin console applies multiple filtering criteria to find suitable workers for a job, including availability, skills, locality, and gender preference.
- **Assignment and Status Management:** Admins assign workers to requests, and workers must confirm acceptance before a task is finalized.
- **Worker Calendar Management:** Workers maintain availability schedules and pre-existing calendar entries to prevent conflicts.

2.3 User Characteristics

- **Customers** are typical end-users with minimal technical knowledge. They interact with simple menu-driven terminal interfaces.
- **Admins** have moderate technical skills and are responsible for decision-making regarding worker assignments.
- **Workers** use a simple interface to review and respond to assigned jobs.

2.4 Operating Environment

The system operates in a standard terminal environment on any machine with:

- C++ compiler (e.g., g++)
- Java Runtime Environment (JRE)
- JSON parser libraries

2.5 Design & Implementation Constraints

- JSON file access must occur sequentially to avoid corruption.
- The system follows SOLID principles and modular design.
- C++ and Java must interact with JSON using independent parsing libraries (nlohmann/json, Gson, or Jackson).

2.6 Assumptions and Dependencies

- Terminal operations occur sequentially; no two modules write at the same time.
- Workers and admins regularly refresh data to see updates.

3 External Interface Requirements

3.1 User Interfaces

The system uses text-based menus. Users navigate through numbered options, with prompts guiding them through input requirements such as selecting works, entering locality, specifying time windows, and updating profile fields.

3.2 Hardware Interfaces

The system requires no special hardware beyond a computer with a keyboard and screen.

3.3 Software Interfaces

Each module interacts with the following JSON files:

- `customers.json` — holds customer profiles.
- `workers.json` — stores worker details, skills, availability, and calendar.
- `services.json` — stores service requests and status transitions.

3.4 Communications Interfaces

Inter-process communication occurs through JSON files stored in the same directory. No network-based communication is used.

4 System Features

This section describes the core functionalities offered by the Worker Management System. Each feature is grouped by user role and provides a clear explanation of the user's capabilities and how the system is expected to behave. The goal is to capture functional requirements in a way that is easy to understand and test, without referencing design or implementation details.

4.1 Customer Module

4.1.1 Register and Login

The customer module provides an interface for users to create a new account or log in to an existing one. Customers provide basic details such as name, address, locality, gender, and password. After login, the customer is directed to their main dashboard where they can request services or manage their account. The system ensures that only authenticated users can access customer-specific functionalities.

4.1.2 Create Service Request

Customers can request home services such as cleaning by selecting one or more tasks from a predefined list. They may choose an immediate service or schedule one for a future time. The system allows the customer to specify gender preference, address, and a time window for the service. Scheduled requests must follow a minimum advance-time rule (e.g., at least one hour before the requested time). Once submitted, the request is recorded and awaits administrative review.

4.1.3 View Current Bookings

Customers can check the status of all ongoing or pending service requests they have made. This includes requests waiting for admin assignment, requests awaiting worker confirmation, and requests already assigned to a worker. Each booking shows the relevant details such as service tasks, time slot, and current status.

4.1.4 View Service History

The system provides a list of all services completed for the customer in the past. Each entry includes the date, time, selected tasks, and worker details. This helps customers understand their usage patterns and access their prior service records easily.

4.1.5 View Rejected Services

Customers may also view all service requests that were rejected by the admin or declined by workers. For each rejected request, the system displays the rejection reason, enabling the customer to resubmit a corrected or revised request if desired.

4.1.6 Rebook Previous Service

Customers may select any previously completed service and quickly create a new service request based on it. The system allows them to reuse the same tasks and preferences, while still giving the option to modify date, time, and other relevant details before resubmission.

4.1.7 Edit Customer Profile

Customers can update their personal information, such as name, address, locality, password, and preferences. Any changes made here will be used for future service requests and administrative processing.

4.2 Admin Module

4.2.1 Admin Login

The admin module begins with an authentication step in which the administrator logs in using predefined credentials. Only authenticated admins may access the administrative dashboard and perform management or assignment operations.

4.2.2 View Pending Service Requests

The admin is presented with a list of all service requests that customers have submitted and that are currently waiting for administrative action. Each request clearly displays the customer details, selected works, gender preference, locality, and scheduled time. This view acts as the administrator's task queue.

4.2.3 Filter Workers

The admin can apply a filtering process to identify workers who are suitable for a given service request. The filtering criteria include:

- matching skills for selected tasks,
- compatibility with customer's gender preference,
- locality match,
- availability of the worker during the requested time window,
- absence of schedule conflicts in the worker's existing calendar.

This filtered list helps the admin determine the most appropriate worker(s) for a job.

4.2.4 Assign Worker

After reviewing the filtered list, the admin can assign one or more workers to the service request. Once the admin confirms the assignment, the request is forwarded to the worker module for acceptance. The status of the service request updates to reflect the assignment and pending worker confirmation.

4.2.5 Reject Request

If the admin determines that no workers are suitable for the service request, or if the request cannot be fulfilled for any other reason, the admin may formally reject it. The system records the rejection along with a note explaining the reason. The customer can later view this in their rejected services screen.

4.2.6 Manage Worker Profiles

The admin can add new workers, modify existing worker details, or update worker skill sets, locality, and availability hours. This feature allows the administrator to maintain an accurate and up-to-date record of the workforce, ensuring better assignments in the future.

4.3 Worker Module

4.3.1 Worker Login

Workers log in to access their personalized dashboard. After authentication, the worker is shown a list of assigned tasks that require their acceptance or rejection, as well as other profile-related options.

4.3.2 View Assigned Services

Workers can view all service requests that have been assigned to them but are still awaiting their response. Each assignment includes details such as the requested tasks, customer location, and scheduled time window. This helps the worker evaluate feasibility before committing to the job.

4.3.3 Accept Job

If the worker agrees to take up the assigned service, they can accept the job. Upon acceptance, the system updates the service request status to indicate that the worker has taken responsibility. The assigned time slot is also added to the worker's calendar to prevent future scheduling conflicts.

4.3.4 Reject Job

Workers may reject a service request if they are unavailable, uncomfortable with the request parameters, or for any other personal or logistical reason. When a job is rejected, the request's status is updated, and the admin is notified so that reassignment can be attempted.

4.3.5 Edit Worker Profile

Workers are allowed to update their personal information, including their name, locality, password, and skill set. These updates help ensure that future assignments are accurate and relevant based on worker capabilities and availability.

4.3.6 View Calendar

The worker module also provides access to the worker's own booking calendar. The calendar lists upcoming accepted jobs along with their dates and times. Workers can use this overview to manage their time effectively and avoid miscommunication regarding their availability.

5 Other Non-Functional Requirements

5.1 Performance Requirements

The system must respond promptly to user interactions. File reading and writing operations must complete within 1–2 seconds for JSON files of typical size (up to a few thousand records). Filtering logic in the admin module must efficiently process lists of workers.

5.2 Safety

Since JSON files act as the data repository, it is crucial that only one module writes at a time to avoid corrupting the files. Sequential terminal usage or file-level locking mechanisms are assumed.

5.3 Security Requirements

The system should ensure confidentiality of sensitive information by hashing passwords and preventing unauthorized access. Role-based access is implicit: customers cannot alter worker data, and workers cannot manipulate service requests beyond acceptance or rejection.

5.4 Software Quality Attributes

- **Maintainability:** Strict adherence to SOLID principles ensures the codebase can evolve.
- **Portability:** The Customer and Admin modules are implemented in C++ and Java, both of which run on multiple operating systems. JSON is used as the common data format, enabling any module to be rewritten or migrated to another language without changing the underlying data structure.
- **Reliability:** Status transitions and calendar validation maintain data consistency.
- **Usability:** Terminal flows use clear prompts and menu-driven structures.