CORY

Software Requirements Specification

1.0

November 7 2024

Sebastian varon
Team Lead

Nikola Varicak
Lead Software Engineer

Document Approval

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

Signature	Printed Name	Title	Date
SV	Sebastian Varon	Team Lead	Nov 7 2024
NV	Nikola Varicak	Lead Software Eng.	Nov 7 2024

Table of Contents

1. Introduction

The Software Requirements Specification (SRS) for the CORY Platform provides a comprehensive outline of the project's requirements. This document will serve as a foundation for the software engineering team to design and implement the CORY platform based on well-defined needs and specifications. Every functional and non-functional requirement necessary to meet the project's goals is included to ensure consistency, precision, and alignment with stakeholder expectations. Additionally, this SRS provides the contextual information and references needed to fully understand the project and scope, ensuring clear guidelines for future development stages.

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) is to document the requirements for developing the CORY platform. This SRS is intended primarily for the software engineering team to guide the design and implementation phases of the project. Secondary audiences include the project's sponsors, stakeholders, and advisors, such as Takeover 6ix Events, who may reference this document to ensure alignment with project objectives and industry expectations.

1.2 Scope

The CORY platform is a web-based solution designed to streamline the process of event staffing for electronic music events, with a particular focus on youth empowerment and volunteer engagement.

- Product Name: CORY - Connection & Organization for Raves & Youth
- Product Objectives:
    - To provide event organizers with a user-friendly platform for job posting, volunteer management, and staff hiring.
    - To create an accessible space for young people interested in gaining hands-on event experience through volunteer roles.
    - To deliver educational content focused on harm reduction, event safety, and industry best practices, enhancing the quality and safety of music events.
- Product Capabilities:
    - The platform will enable event organizers to post jobs, manage applications, and communicate with hired staff and volunteers.
    - Volunteers and paid staff will have access to job opportunities and feedback, helping them build their experience and reputation in the event industry.
    - Educational resources will be provided to users, promoting safer, more successful event experiences.
- Limitations:
    - The initial scope does not include ticketing, full-scale event management tools, or direct integration with external scheduling systems.

1.3 Definitions, Acronyms, and Abbreviations

1.4 References

The following documents and references are essential for interpreting and understanding this SRS:

1. Project Vision Document
   - Title: CORY Project Vision Document
   - Date: [Date of Document]
   - Publishing Organization: George Brown College Capstone Project
2. High-Level Requirements Document
   - Title: CORY High-Level Requirements
   - Date: [Date of Document]
   - Publishing Organization: George Brown College Capstone Project
3. Team Charter
   - Title: CORY Team Charter
   - Date: [Date of Document]
   - Publishing Organization: George Brown College Capstone Project

2. General Description

This section provides a high-level overview of the CORY platform and factors that affect its requirements. By offering context and summarizing the platform's functions, user characteristics, and constraints, this section aims to improve the understanding of the specific requirements described in subsequent sections.

2.1 Product Perspective

The CORY platform is designed as a web-based solution that integrates with event planning and staffing workflows. Positioned as a unique tool in the electronic music industry, CORY bridges the gap between event organizers and a pool of volunteers and skilled paid staff, such as AV technicians and lighting engineers. Unlike traditional event management platforms, CORY focuses on providing resources for both event success and youth development, offering educational content on harm reduction, volunteer training, and event safety.

- Integration with Other Platforms: While CORY does not initially integrate with external ticketing systems, it complements existing platforms by providing a separate staffing and educational component. Future versions may consider integrating with popular ticketing and event management systems.
- Comparison to Other Tools: Compared to general event staffing tools, CORY places a stronger emphasis on volunteer coordination, youth empowerment, and event education. The freemium

model allows for scalability, with basic functions accessible to all users and advanced features

Term	Definition
SRS	Software Requirements Specification
MVP	Minimum Viable Product – the most basic version of the product that meets user needs
Event Organizer	An individual or organization responsible for planning and executing events
Volunteer	A person offering services without pay, typically to gain experience
Freemium	A business model where basic services are free, while premium features require payment
Takeover 6ix Events	The project's sponsor, a Toronto-based event organization specializing in techno events
Harm Reduction	Strategies designed to reduce risks associated with large events

available to paid users.

2.2 Product Functions

The primary functions of the CORY platform include:

- Job Posting and Hiring: Allows event organizers to post job opportunities for both paid and volunteer roles, view applicants, and select candidates based on profiles and ratings.
- Volunteer Coordination: Manages volunteer opportunities, allowing youth to sign up, track their participation, and receive feedback.
- Real-Time Feedback and Ratings: Provides a feedback system where event organizers can evaluate staff performance, fostering accountability and skill-building for volunteers and paid staff.
- Educational Content Delivery: Offers educational resources on event management, harm reduction, and audiovisual best practices to help organizers enhance event quality and safety.
- User Profiles: Each user (organizer, volunteer, and paid staff) has a profile that reflects their skills, experience, and ratings, making it easier to find appropriate job matches.

2.3 User Characteristics

The **CORY platform** is designed for three primary user types:

- **Event Organizers**: Typically intermediate to advanced users familiar with event planning, but may have varying levels of technical skill. They are looking for streamlined solutions to find, manage, and evaluate staff quickly and efficiently.
- **Volunteers**: Often younger individuals with beginner-level experience in the event industry, seeking hands-on experience and training. They may require guidance through educational materials and resources provided on the platform.

- **Paid Staff (Freelancers)**: This group includes skilled AV technicians, lighting engineers, and other specialized staff with moderate to advanced technical skills. They require a platform where they can easily showcase their experience, apply for jobs, and receive feedback from employers.

These users' diverse skill levels influence design choices, ensuring the interface is intuitive for beginners while robust enough for professional users.

2.4 General Constraints

Several constraints affect the design and development of the **CORY platform**:

- **Time and Resources**: The project is being developed as a capstone project by a two-person team, limiting the scope and pace of development.
- **Data Privacy and Security**: Since the platform will collect personal information from volunteers (many of whom may be minors), stringent data protection measures must be implemented to ensure compliance with privacy regulations.
- **Budget Limitations**: The project operates under a limited budget, meaning that certain advanced features or third-party integrations may need to be deferred to future versions.
- **Internet and Device Accessibility**: CORY will be accessible through web browsers and optimized for various devices; however, due to resource limitations, it will not initially support native mobile applications.

2.5 Assumptions and Dependencies

Key assumptions and dependencies that affect the requirements for **CORY** include:

- **Operating Environment**: The platform is expected to operate primarily on desktop and mobile web browsers. Assumes widespread access to modern browsers like Chrome, Firefox, and Safari.
- **Sponsor Support**: The project's requirements are partially based on feedback from *Takeover 6ix Events*, which is expected to provide real-world testing and feedback.
- **Youth Engagement**: Assumes that a significant number of youth will seek volunteer roles and benefit from the platform's educational resources. The platform's success depends on youth adoption and engagement.
- **Freemium Model Viability**: Assumes that a freemium model will encourage adoption by event organizers, with potential monetization through paid features.
- **Long-term Scalability**: Requirements assume that the platform will eventually scale to support different event types and geographies, though the initial scope is limited to electronic music events in Toronto.

3. Specific Requirements

This section provides a comprehensive and detailed list of the requirements necessary to guide the design, implementation, and testing of the **CORY platform**. Each requirement is written to be clear, traceable, testable, and prioritized for ease of understanding and implementation.

3.1 External Interface Requirements

3.1.1 User Interfaces

Requirement ID: 3.1.1

- The CORY platform will have a user-friendly interface, accessible on both desktop and mobile web browsers.
- **Features**: Clear navigation for event organizers, volunteers, and paid staff.
- **Design Goals**: Consistency across pages, responsive layout, and accessibility for a range of user expertise levels.
- **Traceability**: Linked to functional requirements 3.2.1 (Job Posting and Hiring) and 3.2.2 (Volunteer Coordination).

3.1.2 Hardware Interfaces

Requirement ID: 3.1.2

- The platform will be compatible with standard web-enabled devices, including PCs, tablets, and smartphones.
- **Testing**: Must function correctly on commonly used devices and across different screen sizes.
- **Limitations**: No additional hardware requirements are necessary beyond typical user devices with internet access.

3.1.3 Software Interfaces

Requirement ID: 3.1.3

- The system will interface with commonly used web browsers (Chrome, Firefox, Safari).
- **Dependencies**: The platform will depend on browser compatibility for certain JavaScript-based interactions and responsive design.

3.1.4 Communications Interfaces

- **Requirement ID**: 3.1.4
    - **Network Requirements**: CORY must operate over standard internet protocols (HTTP/HTTPS).
    - **Security**: Communications will use HTTPS encryption to ensure secure data transfer, especially for sensitive information related to user accounts.

3.2 Functional Requirements

This section describes specific functions that the CORY platform must perform.

3.2.1.1 Introduction

- Allows event organizers to post job openings for specialized roles, view applicants, and make hiring decisions.

3.2.1.2 Inputs

- Job posting details (title, role description, requirements).
- Search filters from event organizers (e.g., skill, experience level).

3.2.1.3 Processing

- Match job postings with user profiles (volunteers and paid staff) based on skills, experience, and availability.

3.2.1.4 Outputs

- List of applicants, job matches displayed for organizers, notifications for applicants.

3.2.1.5 Error Handling

- Error message for incomplete job posting fields, alerts for connection errors when posting.

3.2.2.2 Inputs

- Volunteer registration details, event requirements from organizers.

3.2.2.3 Processing

- Matches volunteers to open roles based on experience, interest, and availability.

3.2.2.4 Outputs

- Confirmations of volunteer positions, tracking of volunteer activity, feedback received after events.

3.2.2.5 Error Handling

- Alerts for role unavailability, notifications for volunteers on updates in job status.

3.2.3 Real-Time Feedback and Ratings

3.2.3.1 Introduction

- Provides a feedback mechanism for event organizers to rate staff and volunteers after each event.

3.2.3.2 Inputs

- Feedback and ratings submitted by organizers, based on staff and volunteer performance.

3.2.3.3 Processing

- Stores and displays feedback on user profiles for future reference by organizers and staff.

3.2.3.4 Outputs

- User performance rating updates, feedback notifications to volunteers and paid staff.

3.2.3.5 Error Handling

- Alerts if feedback fields are incomplete, failure notices if feedback submission does not save.

3.3 Use Cases

3.3.1 Use Case #1: Event Organizer Posts a Job

- **Actors**: Event Organizer
- **Description**: Organizer posts a job for an upcoming event, specifying details like role, qualifications, and duration.

3.3.2 Use Case #2: Volunteer Applies for a Role

- **Actors**: Volunteer
- **Description**: Volunteer logs in, views available roles, and applies to suitable ones based on interest and qualifications.

3.4 Classes / Objects

3.4.1 User Profile

3.4.1.1 Attributes

- User ID, name, role, experience level, skills, rating.

3.4.1.2 Functions

- Update profile, view job history, receive notifications.

3.4.2 Event Job Posting

3.4.2.1 Attributes

- Job ID, title, description, required skills, status.

3.4.2.2 Functions

- Post job, edit job details, view applicants.

3.5 Non-Functional Requirements

3.5.1 Performance

- 95% of transactions (e.g., posting, applying for jobs) should be completed in under 2 seconds.

3.5.2 Reliability

- The system should maintain a 99.5% uptime, with downtime not exceeding 1 minute per day.

3.5.3 Availability

- Available 24/7, with planned maintenance notifications 24 hours in advance.

3.5.4 Security

- Data encryption for sensitive user information and compliance with GDPR and local data protection laws.

3.5.5 Maintainability

- Modular design to allow for easy updates and bug fixes.

3.5.6 Portability

- Optimized for compatibility across major web browsers and devices.

3.6 Inverse Requirements

- The platform will not support ticketing or full event management. CORY is solely focused on staffing, volunteering, and educational resources.

3.7 Design Constraints

- **Data Privacy Regulations**: Compliance with data protection standards, specifically for minors and volunteers.
- **Limited Budget**: Design choices must minimize costs and leverage free or open-source resources when possible.

3.8 Logical Database Requirements

- **Data Formats**: Structured storage for user profiles, job postings, feedback, and ratings.

- **Data Retention**: User data to be retained for 5 years post-account creation, with options for users to request deletion.

3.9 Other Requirements

- **Language Support**: Initially, English only, with future multilingual support as demand increases.
- **Feedback System**: Allows users to report issues or suggest features for continuous improvement.

4. Analysis Models

This section provides a comprehensive view of the analysis models used to derive and clarify the specific requirements in this SRS. These models—**Sequence Diagrams**, **Data Flow Diagrams (DFD)**, and **State-Transition Diagrams (STD)**—help in understanding interactions, data flow, and state changes within the CORY platform. Each model is traceable to the specific requirements listed in Section 3, ensuring consistency and alignment with the project's objectives.

4.1 Sequence Diagrams

Introduction:
The **Sequence Diagrams** illustrate the interaction flow between users and the CORY platform's components. Each diagram visualizes how users, such as event organizers, volunteers, and paid staff, interact with the system to accomplish tasks like job posting, volunteer application, and feedback submission. These diagrams align with specific functional requirements (3.2.1 - Job Posting and Hiring, 3.2.2 - Volunteer Coordination, and 3.2.3 - Real-Time Feedback).

Narrative Description:

1. **Job Posting Sequence**:
    - The **event organizer** accesses the CORY platform, navigates to the job posting section, and submits a job post.
    - The system confirms submission and stores the job details in the database.
    - Volunteers and paid staff receive notifications and can view available jobs based on filters.
2. **Volunteer Application Sequence**:
    - A **volunteer** logs into their account, searches for open roles, and applies for a position.
    - The system records the application and notifies the organizer of the new applicant.

- If accepted, the volunteer receives a confirmation, and the event role is marked as filled.

3. **Feedback Submission Sequence**:
    - After the event, the **organizer** submits feedback on volunteers and paid staff.
    - The feedback is stored in each user's profile, affecting their overall rating.
    - Users receive notifications about their new feedback.

Traceability:
Each sequence diagram is traceable to the functional requirements in Section 3, specifically:

- **3.2.1**: Job Posting and Hiring
- **3.2.2**: Volunteer Coordination
- **3.2.3**: Real-Time Feedback and Ratings

4.2 Data Flow Diagrams (DFD)

Introduction:
The **Data Flow Diagrams (DFD)** represent the flow of information within the CORY platform, showing how data moves between different components, such as user accounts, job postings, and feedback modules. DFDs provide a clear view of how data is created, stored, processed, and distributed across the system.

Narrative Description:

1. **Context Diagram (Level 0)**:
    - Shows the CORY platform as a central node with connections to key external entities: **Event Organizer**, **Volunteer**, **Paid Staff**, and **Sponsor**.
    - Depicts high-level data exchanges such as job posting, volunteer applications, and feedback collection.
2. **Level 1 DFD - Job Posting**:
    - Details the flow when an organizer creates a job posting:
        - Input data: Job details, required skills, and position status.
        - Processing: System validates and stores job data.
        - Output: Notifications sent to volunteers and paid staff.
3. **Level 1 DFD - Feedback Collection**:
    - Depicts how feedback data flows from organizers to user profiles:
        - Input data: Feedback ratings and comments from organizers.
        - Processing: System associates feedback with user profiles.
        - Output: Updated ratings and notifications to users.

4.3 State-Transition Diagrams (STD)

Introduction:
The **State-Transition Diagrams (STD)** illustrate how the CORY platform manages different states for users and event listings. They show how users move from one state to another based on actions taken, providing clarity on the platform's response to various inputs.

Narrative Description:

1. **Job Posting Lifecycle**:
    - **States**: Draft, Posted, Filled, Closed.
    - **Transitions**:
        - From **Draft** to **Posted** when the organizer finalizes and submits the job.
        - From **Posted** to **Filled** once a candidate is accepted.
        - From **Posted** or **Filled** to **Closed** when the job is no longer available or the event ends.
2. **User Account State**:
    - **States**: New, Active, Suspended, Deactivated.
    - **Transitions**:
        - From **New** to **Active** after account verification.
        - From **Active** to **Suspended** if there is a policy violation.
        - From **Suspended** or **Active** to **Deactivated** if the user opts to close their account.
3. **Application Process State**:
    - **States**: Applied, In Review, Accepted, Rejected.
    - **Transitions**:
        - From **Applied** to **In Review** once submitted.
        - From **In Review** to **Accepted** or **Rejected** based on the organizer's decision.

Traceability:
The STD models map to **Section 3.2 Functional Requirements**, with specific traceability to:

- **3.2.1**: Job Posting and Hiring
- **3.2.2**: Volunteer Coordination
- **3.2.3**: Real-Time Feedback and Ratings

5. Change Management Process

The **Change Management Process** for the **CORY Software Requirements Specification (SRS)** outlines the procedures to be followed whenever updates to the SRS are required due to changes in project scope, requirements, or stakeholder needs. This process ensures that any adjustments are documented, reviewed, and approved in a controlled manner to maintain consistency and clarity throughout the project lifecycle.

5.1 Change Submission Process

1. **Eligibility to Submit Changes**:
    - Changes may be submitted by:
        - **Core Team Members** (Developers, Project Leads)
        - **Project Sponsor** (e.g., *Takeover 6ix Events*)
        - **Academic Advisors** overseeing the capstone project
    - Users, such as **event organizers and volunteers**, may provide feedback, which core team members can evaluate and submit as potential changes.
2. **Means of Submitting Changes**:
    - Change requests should be submitted in writing using a **Change Request Form** that includes:
        - A description of the proposed change.
        - Justification for the change.
        - Any anticipated impact on functionality, design, or schedule.
    - All change requests are to be documented in the **Change Log** section of the SRS, available through the shared project repository on GitHub.

5.2 Review and Approval Process

1. **Initial Review**:
    - Each submitted change request will first be reviewed by the **Project Leads** (Sebastian and Nikola) to assess its relevance and feasibility. This assessment includes evaluating the impact on existing requirements, timelines, and resources.
2. **Stakeholder Review and Feedback**:
    - After the initial review, if deemed feasible, the change request will be shared with:
        - The **Sponsor** (*Takeover 6ix Events*) for insights on the potential impact on real-world usage.
        - **Academic Advisors** for approval if the change affects the project scope or deliverables required for the capstone project.
3. **Approval**:
    - Final approval for any change requires the consensus of the **Project Leads** and **Academic Advisors**.
    - Approved changes are marked as "Accepted" in the Change Log, and specific adjustments to requirements are outlined in an updated version of the SRS.
4. **Implementation of Changes**:

- Once approved, the core team integrates the changes into the project's current scope and timeline, updating all relevant sections of the SRS to reflect the new requirements.
- Version control is applied to each updated SRS, with each approved change leading to a new document version (e.g., v1.1, v1.2) and timestamped for reference.

5.3 Communication of Changes

1. **Team Notification**:
   - All team members are notified of approved changes through a formal **Project Update Meeting** or an official notice on the project's Slack channel.
2. **Stakeholder Notification**:
   - Significant changes impacting the project's scope, deliverables, or timeline are communicated to external stakeholders, including *Takeover 6ix Events*, to maintain alignment.

5.4 Version Control and Documentation

1. **Version Control**:
   - Each iteration of the SRS is versioned (e.g., SRS v1.0, v1.1) with detailed notes summarizing changes and updates.
   - The **Change Log** at the end of each version lists all approved changes, including the description, date, submitter, and approvers.
2. **Documentation**:
   - Updated versions of the SRS are stored in the project repository (GitHub), accessible to all team members, sponsors, and advisors.
   - Documentation includes both the updated SRS and the Change Log, ensuring full traceability and accountability for every modification.