Company: AMC

Products: Ticketing Software, organizations

Software Requirements Specification Template

Software Engineering

The following annotated template shall be used to complete the Software Requirements Specification (SRS) assignment.

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.

Ticketing System

Software Requirements Specification

v1.0

2/15/2024

Group 1

Khang Tonthat
Rafael Aguilar
Ethan Morales

Revision History

Date	Description	Author	Comments
<date>	<Version 1>	<Your Name>	<First Revision>
2/15/24	Version 1.0.0	Rafael Aguilar, Khang Tonthat, Ethan Morales	Created SRS Outline all the way up to Section 3.6
2/29	Version 1.0.1	Rafael Aguilar, Khang Tonthat, Ethan Morales	Completed the remaining Section 3.7 and the entirety of Section 4 of SRS

Document Approval

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

Signature	Printed Name	Title	Date
	<Your Name>	Software Eng.	
	Dr. Gus Hanna	Instructor, CS 250	

<Project Name>

Table of Contents

<Project Name>

<Project Name>

1. Introduction

The introduction to the Software Requirement Specification (SRS) document should provide an overview of the complete SRS document. While writing this document please remember that this document should contain all of the information needed by a software engineer to adequately design and implement the software product described by the requirements listed in this document. (Note: the following subsection annotations are largely taken from the IEEE Guide to SRS).
The introduction of this Software Requirements Specification

1.1 Purpose

What is the purpose of this SRS and the (intended) audience for which it is written.
The purpose of this SRS document is to collect all ideas that our group has come up with which will serve to define our software requirements. As well as building and predicting various concepts that may help or become an obstacle in the process of building the software. This document also illustrates the various requisites and prerequisites that the stakeholders may specify.

1.2 Scope

This subsection should:
(1) Identify the software product(s) to be produced by name; for example, Host DBMS, Report Generator, etc

(2) Explain what the software product(s) will, and, if necessary, will not do
This product will produce a ticketing website system in which customers can buy tickets for movies available in theaters. Providing movie options for customers in need of buying tickets. As well as providing a selection of movie theaters nearby for the customer's best accommodation. And it will be well protected from malicious outside sources which may harm customers.

This product will not have any malicious hardware which may put the customer's personal information at risk. It will not have a messy layout that may be difficult and confusing to the customer. It will run smoothly software wise to make it as fast as possible for the customer to buy tickets.

(3) Describe the application of the software being specified. As a portion of this, it should:
(a) Describe all relevant benefits, objectives, and goals as precisely as possible. For example, to say that one goal is to provide effective reporting capabilities is not as good as saying parameter-driven, user-definable reports with a 2 h turnaround and on-line entry of user parameters.
The goal is to provide a more streamlined and convenient way for user to enjoy there films without delay

(b) Be consistent with similar statements in higher-level specifications (for example, the System Requirement Specification) , if they exist.What is the scope of this software product.

Modern look, easy to use website with various services which will facilitate users into buying tickets.

Secure broadband, virus-free server that secures users information into an unbreakable information vault.

Provide a comfortable experience regarding signing up for rewards, searching for selected movies, and securing important information in a secure server.

1.3 Definitions, Acronyms, and Abbreviations

This subsection should provide the definitions of all terms, acronyms, and abbreviations required to properly interpret the SRS. This information may be provided by reference to one or more appendixes in the SRS or by reference to other documents.

Server: Manages access to a centralized service in a network.
User: Customers trying to purchase any tickets we are offering.
Software: Necessary components in order to build and create the backend compartments of the website.
Hardware: The necessary components in which the website should be able to operate.
User interface: Interface in which customers will interact with various services that are provided on the website.
Functional Requirements: Requirements necessary for product to properly function in order
Broadband: high-capacity transmission technique that can carry multiple signals and data types simultaneously over a communication medium.
Data Privacy: Protection of personal and sensitive information from unauthorized accessed from the website.

1.4 References

This subsection should:
(1) Provide a complete list of all documents referenced elsewhere in the SRS, or in a separate, specified document.
(2) Identify each document by title, report number - if applicable - date, and publishing organization.
(3) Specify the sources from which the references can be obtained.
This information may be provided by reference to an appendix or to another document.

IEEE Computer Society. (1998). IEEE Recommended Practice for Software Requirements Specifications. STD 830-1993.
Document used to understand the practice of basic software requirements specifications in order to create our own.

NATO Software Engineering. October 11th, 1968. NATO Science Committee.
Document used in order to understand the concept of a working team constructing an SRS.

Software Requirements Specification Document With Example. May 8th, 2024. Ravi Bandakkanavar.

<Project Name>

Used to understand the basics of important elements of an SRS like use cases as well as requirement specifications.

1.5 Overview

This subsection should:
(1) Describe what the rest of the SRS contains
(2) Explain how the SRS is organized.

The rest of this document contains:
General description of the product.

Specific requirements that are needed for the product to be successfully developed.

Analysis models providing examples of the finished products.

Change management process identifies the process and changes towards the main SRS.

2. General Description

This section of the SRS should describe the general factors that affect 'the product and its requirements. It should be made clear that this section does not state specific requirements; it only makes those requirements easier to understand.

This document describes many factors about problems this product may have when it comes to the building of it. As well as solutions for any of these problems. It contains a list of various solutions that users and stakeholders will have. It as well shows the requirements from various stakeholders. It furthers our knowledge and understanding of the specific requirements a stakeholder may need and want for the ticketing system. As well as a brief description of major features.

2.1 Product Perspective

This subsection of the SRS puts the product into perspective with other related products or projects. (See the IEEE Guide to SRS for more details).

The ticketing system is a component of a larger system that involves multiple interfaces in order to make it function. Products such as Ticketmaster, AMC Theaters, and Regal Cinema operate using similar systems and encounter multiple of the same functions as well as constraints due to the nature of handing out and managing tickets on a large scale.

Our product should be easily integrable onto multiple platforms or third party services, whether it be through the use of API's or compatibility that allows users to vendor their own tickets depending on the event.

Through these platforms, they should be able to seamlessly link to a payment gateway to easily vend the ticket without much hassle by making our software compatible with modern financial systems

We'll also provide data analytics for our users who are vending tickets, giving them information on how many tickets have been bought, sections/seats that have been taken, revenue, etc.

2.2 Product Functions

This subsection of the SRS should provide a summary of the functions that the software will perform.

The functions of the software will be able to perform a ticketing based where clients can come to the website and buy tickets for films they would like to view. Its functions should include scheduling, ticketing, availability, and a pop of details in regard to the film to entice the clients to buy the tickets.

Links to other software such as Instagram, Twitter, Facebook, TikTok, and YouTube to promote clients to buy tickets.

2.3 User Characteristics

This subsection of the SRS should describe those general characteristics of the eventual users of the product that will affect the specific requirements. (See the IEEE Guide to SRS for more details).

- Elderly (60+)
- Military/ Service Members/ Veterans
- Under 18 (Teens)
- Children (2-12)
- V.I.P / Movie Members

2.4 General Constraints

This subsection of the SRS should provide a general description of any other items that will limit the developer's options for designing the system. (See the IEEE Guide to SRS for a partial list of possible general constraints).

- Scalability (Handling large volumes of traffic since there are peak periods of time due to selling tickets for popular movies)
- Cookies
- Allocated memory set for the website
- Data Privacy (Handling User Data when They Input Private Information to Purchase Tickets or Logging into Accounts)
- General Performance (We want our website to be modern and sleek, but not unresponsive so that our site runs on all devices to maximize users)
- Location based (Different laws may affect what is available and allowed)

2.5 Assumptions and Dependencies

This subsection of the SRS should list each of the factors that affect the requirements stated in the SRS. These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS. For example, an assumption might be that a

<Project Name>

specific operating system will be available on the hardware designated for the software product. If, in fact, the operating system is not available, the SRS would then have to change accordingly.

- Platform Compatibility (MacOS, Windows, etc.)
-
- Internet Connectivity: User is assumed to have internet connectivity in order to access the product.
- User authentication: User is assumed to have valid credentials.
- User Training: User is assumed to be familiar with website structure.
- Data Accuracy: Data by users is assumed to be accurate and valid.

Dependencies:
- Web Server: Website depends on a functional web server in order to work properly.
- Network Infrastructure: Functionality depends on well developed network infrastructure.
- Security Software: Security depends on the functionality of security measures like detection systems.
- Payment Paywalls: Purchasing tickets depends on payment paywalls for secure transactions.
- Compatibility of Browsers: Whole system depends on compatibility with various web browsers.

3. Specific Requirements

This will be the largest and most important section of the SRS. The customer requirements will be embodied within Section 2, but this section will give the D-requirements that are used to guide the project's software design, implementation, and testing.

Each requirement in this section should be:
- *Correct*
- *Traceable (both forward and backward to prior/future artifacts)*
- *Unambiguous*
- *Verifiable (i.e., testable)*
- *Prioritized (with respect to importance and/or stability)*
- *Complete*
- *Consistent*
- *Uniquely identifiable (usually via numbering like 3.4.5.6)*

Attention should be paid to the carefuly organize the requirements presented in this section so that they may easily accessed and understood. Furthermore, this SRS is not the software design document, therefore one should avoid the tendency to over-constrain (and therefore design) the software project within this SRS.

3.1 External Interface Requirements

3.1.1 User Interfaces

- The system shall display a variety of movie options for the customer.
- The system shall display a brief summary of the movie that the customer selects.

<Project Name>

- The system shall display options of seating in the form of rows.
- Include itinerary displaying movie time and dates.
- Should display the length of the movie as well as rating and when it was first released.
- Should display a search bar for customer needs.
- Should include a sidebar with various options ranging from rewards, finding a theater/movie.
- Should display the amount of tickets a customer may want and the age range for each ticket.
- Should display the total amount of money the customer owes.
- After each purchase, a QR code should be displayed to indicate the customer's tickets.
- Should display an option regarding whether a customer may want to sign up and become a member,
- Should display a sign-in button for customers with existing accounts.

3.1.2 Hardware Interfaces

- Software should be compatible with hardware configurations
  - Configurations include: Personal Computers, Laptops, Smartphones/Tablets
- Should also support machinery such as printers so that users can print out tickets if they want them physically rather than digitally
- Should also support machinery such as scanners so that if users use a barcode or a QR scanner, their ticket will allow them access to their event

3.1.3 Software Interfaces

- Will be accessed through most web browsers, so should be compatible with multiple types of them
  - Chromium: Google Chrome, Microsoft Edge, Opera
  - Gecko: Mozilla Firefox
  - WebKit: Apple's Safari
- Users will access the website from multiple types of devices, so it's important that the website can run on many types of operating systems
  - MacOS
  - Windows
  - iOS
  - Android
- Must be integrable with outside software that helps authorize and process payment methods
  - PayPal
  - Stripe
  - Shopify

3.1.4 Communications Interfaces

- Websites should use HTTPS to help secure communication between the user and our website servers.

<Project Name>

- Website should also be capable of sending emails for tasks such as user login confirmation, ticket purchase confirmation, account updates, and special announcements

3.2 Functional Requirements

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.

3.2.1 <Online Ticket Purchasing>

3.2.1.1 Introduction

The system should allow users to browse available movies, see when they start showing, view available and taken seats, and finally make their secure online payments to purchase tickets.

3.2.1.2 Inputs

User inputs will include movie selection, preferred showtime, seat choices, and their payment information.

3.2.1.3 Processing

The system will check seat availability to prevent duplicate tickets, calculate the total cost, and process the payment securely through a payment processor

3.2.1.4 Outputs

If a user completes their purchase, the system will send them a confirmation email with either a unique QR or Barcode for their ticket(s). The website will update to take into account the tickets purchased by removing their seating from being available.

3.2.1.5 Error Handling

In the case that the user fails to purchase their ticket because their payment method declines, or the seat they desired is taken during the payment process, they will be offered a refund for any money that may have been already processed, and an error message will be displayed.

3.2.2 <Member Registration>

3.2.2.1 Introduction

Guest users are given the option to create an account when they first open our website, allowing them to register through our online ticketing system. Through the creation of an account, the user is given specific perks such as special announcements, viewing purchase history, and having their preferences saved.

3.2.2.2 Inputs

User Inputs will include the option to create an account, the input of their email, phone number, and password.

3.2.2.3 Processing

The system will check to see if their email or phone number is already linked to an account, then will create a unique user ID and register the user after sending them a confirmation email with a 30 minute window.

3.2.2.4 Outputs

Through the creation of the account, the user is given access to specific perks that are catered to help their purchasing experience and record their purchases

3.2.2.5 Error Handling

<Project Name>

In the case that the user's email or phone number is already registered, an error message is output telling the user that their account has failed to be created since they may already have an account registered, barring the creation of the account and the generation of a confirmation email.

In the case that the user fails to confirm their information through the confirmation email after 30 minutes, the system will send a reminder email asking the user to generate a new confirmation email so that they may start using their account. If they choose to ignore the reminder, their account will be deactivated until confirmation has been received.

3.3 Use Cases

3.3.1 Use Case #1: Online Ticket Purchase

3.3.1.1 Actors
- Guest User
- Registered User

3.3.1.2 Purpose
Guest users can browse available movies, select showtimes, choose seats, and make online payments for ticket purchase. Registered users can log in to access their saved information.

3.3.1.3 Scenarios
1. Guest/Registered user selects a movie, seats, and showtime.
3. The system validates seat availability and calculates the total cost.
4. User is provided the option to log into or create an account to making ticket purchasing easier
5. User is then sent to view payment processor in order to validate payment
6. User is given the option to choose their payment method (Debit, Credit)
6. Email containing ticket information is sent and the user's account is updated
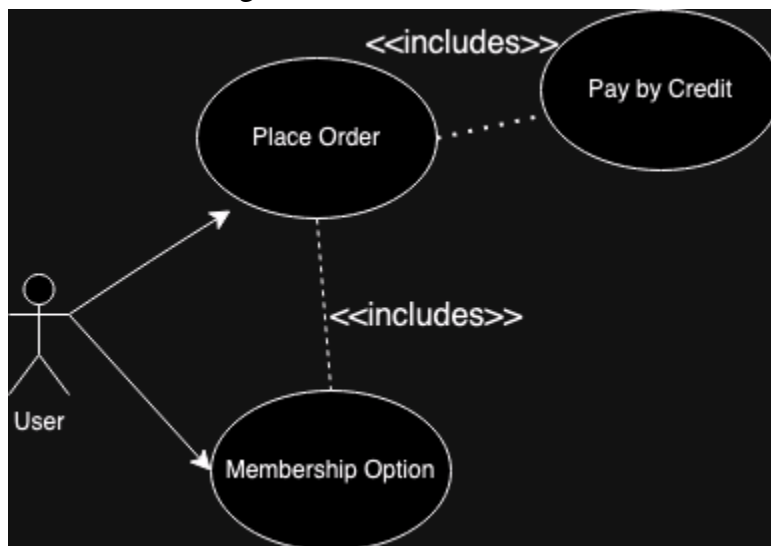
### 3.3.2 Use Case #2: Member Registration

3.3.2.1 Actors
-    Guest User

3.3.2.2 Purpose
Guest users can create an account by registering with the online ticketing system. Registration allows users to access additional features such as saving preferences, viewing purchase history, and receiving personalized recommendations.

3.3.2.3 Scenarios

1. Guest user navigates to the registration page.
2. System displays a registration form with fields such as: Name, Email/Phone number, Password, Address
3. Guest user fills in the required information.
4. the System validates the entered information for completeness and accuracy.
5. If validation is successful, the system generates a unique user ID and registers the user.
6. System sends a confirmation email to the provided email address.
7. User clicks the confirmation link in the email to verify their account.
8. Once verified, the user gains access to additional features and can log in to the system.
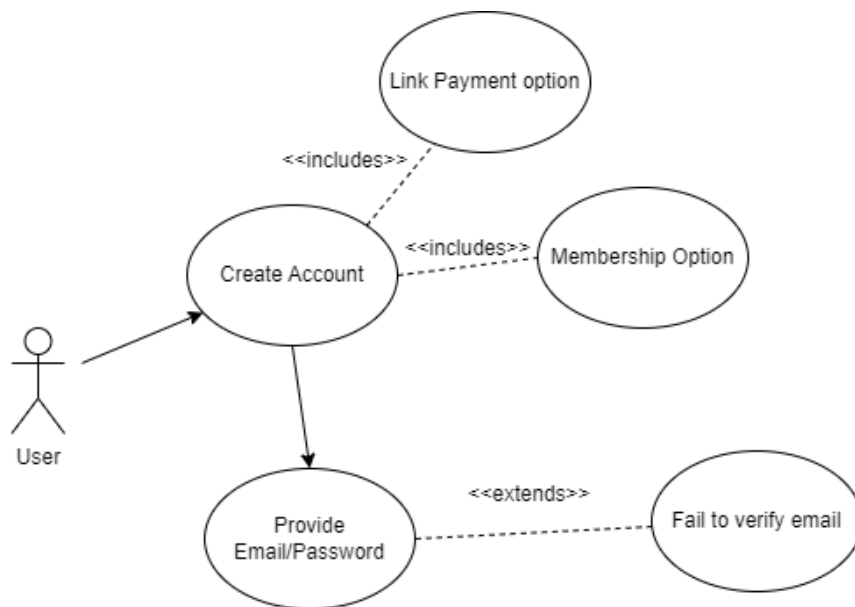
3.3.3 Use Case #3: Ticket Cancellation
3.3.3.1 Actors
   - Registered User
3.3.3.2 Description
Users are allowed to cancel any purchased tickets as long as it is before their showing, giving them a full refund of their purchase.
3.3.3.3 Scenarios
   1. User logs into system

2. User navigates to their purchase history
3. User is allowed to cancel their purchase as long as it is before their showing
4. System asks for confirmation in their cancellation
5. System will process their cancellation, update the user account, and give them their refund in due time.
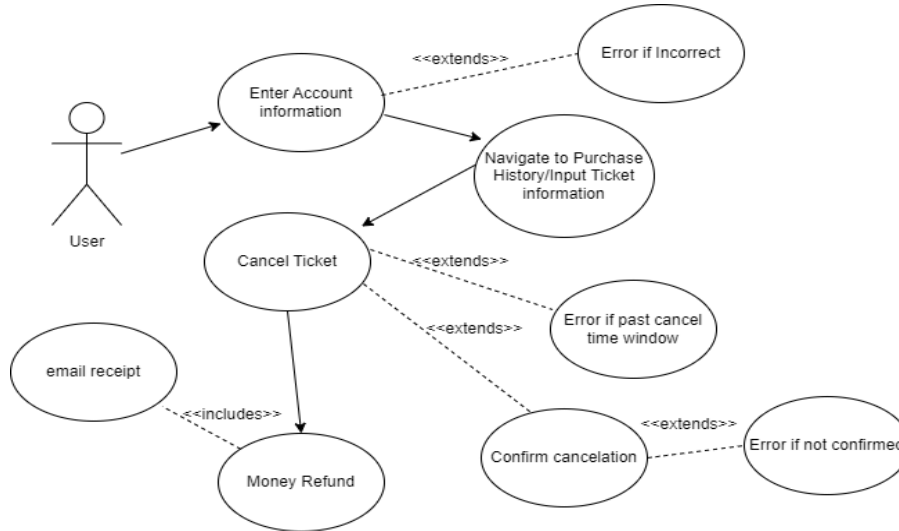
3.4 Classes / Objects

3.4.1 <Class / Object #1>: User Account

3.4.1.1 Attributes
- User ID (int): Unique Identification for each user account
- Email (string): Email address associated with the account, will be used in the case of a username
- Password (string): Encrypted password meant to serve as securing user information
- Phone number (int): User's phone number that serves as a second platform for messages other than email

3.4.1.2 Functions
- updateProfile(): Allows user to update their information, such as adding phone number
- changePassword(): Allows user to change their password
- changeEmail(): Allows user to change their email address

3.4.2 <Class / Object #2>: Ticket

3.4.2.1 Attributes
- Ticket ID (int): Unique identifier for each ticket
- MovieTitle(string): Title of movie that is corresponded to the ticket purchased
- Showtime(int): Date and Time of showing
- SeatNumber(string): Assigned seat corresponded to the ticket purchased
- Price(float): Ticket cost
- Status(bool): Indicated whether the ticket has been purchased or still available.

3.4.2.2 Functions

- generateQRCode(): Generates code corresponding to the ticket ID that allows you to scan your ticket at check-in.
- cancelTicket(): Allows the user to cancel their ticket, updating its status back to being available and prompting a refund.

3.5 Non-Functional Requirements

Non-functional requirements may exist for the following attributes. Often these requirements must be achieved at a system-wide level rather than at a unit level. State the requirements in the following sections in measurable terms (e.g., 95% of transaction shall be processed in less than a second, system downtime may not exceed 1 minute per day, > 30 day MTBF value, etc).

3.5.1 Performance

- Scalability: The system should support a minimum of 5000 users without affecting the performance of the system.
- Response Time: The system should be able to respond to a user within a maximum of 1.5 seconds.

3.5.2 Reliability

- Tolerance regarding any faults: The system should still continue to function regarding any data loss or small faults throughout the service until the problem is solved.
- Recovery: The system should be able to document various data in order to prevent any data loss from a fault.

3.5.3 Availability

- Time Available: The system should be able to operate within all times around the world. Should be operating at all times.
- World Wide Usage: The system should be able to be used only locally.

3.5.4 Security

- Authentication: Users should be able to authenticate by using secure methods. The proper authentication should be included for users to authenticate themselves.
- Data Encryption: The system should encrypt any private and sensitive information that a user may enter in order to secure and avoid any data cracks by unauthorized users.
- Access Control

3.5.5 Maintainability

- Software Maintainability: This system should have the tools and personnel in order to maintain a quality assurance over the system.
- Documentation: Documentation should be provided for the architecture of the system in case of any faults in the system.
- Hardware Maintainability: The system should function properly even in case of various hardware failures in the absence of the customer.

3.5.6 Portability
- Training Requirements: This system should be designed to accommodate the user for as less minimal training required for the use of this system.

3.6 Inverse Requirements

State any *useful* inverse requirements.

3.6.1 Duplicate Accounts
- Users shouldn't be able to create an account with an email that has already been registered.
- Inverse Requirement: The system must enforce uniqueness for email addresses as a user tries to create an account, barring them from creation if they input an already registered email.

3.6.2 Duplicate Tickets
- Users shouldn't be able to purchase a ticket that has already been purchased by another user
- Inverse Requirement: The system will track ticket status, only allowing the purchasing of a specific ticket as long as its status still deems it available, and then turning its status off once availability of the ticket is gone.

3.6.3 Unauthorized Ticket Cancellation
- Users shouldn't be allowed to cancel tickets without proper authorization and confirmation
- Inverse Requirement: User will have to log into their account in order to access their purchase history, and will implement multiple prompts to confirm whether the user intends to refund their ticket, such as re-entering their login information.

3.7 Design Constraints

Specify design constrains imposed by other standards, company policies, hardware limitation, etc. that will impact this software project.

3.8 Logical Database Requirements

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

3.9 Other Requirements

Catchall section for any additional requirements.

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.

4.1 Sequence Diagrams

4.3 Data Flow Diagrams (DFD)

4.2 State-Transition Diagrams (STD)

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

Appendices may be used to provide additional (and hopefully helpful) information. If present, the SRS should explicitly state whether the information contained within an appendix is to be considered as a part of the SRS's overall set of requirements.

Example Appendices could include (initial) conceptual documents for the software project, marketing materials, minutes of meetings with the customer(s), etc.

A.1 Appendix 1

A.2 Appendix 2