Software Requirement Specifications (SRS)

BOOK IT

Blockchain Based Online Ticketing System

Version 1.0

Project Code	**F20-031**
Supervisor	Dr. Faisal bin Ubaid
Co-Supervisor	
Project Manager	Muhammad Faizan
Project Team	Muhammad Faizan Arsalan Ashraf Saad Ahmed Raja
Submission Date	

Document History

Version	Name of Person	Date	Description of change
1.0	Muhammad Faizan	01-11-2023	*Document Created and Added scope*
1.0	Saad Raja	04-11-2022	*Added functional requirements*
1.0	Arsalan Ashraf	07-11-2022	*Added Use-Case and non-functional requirements.*

Distribution List

Name	**Role**
Dr. Faisal bin ubaid	Supervisor
Muhammad Faizan	Project manager
Saad Raja	Group Member
Arsalan Ashraf	Group Member

Revision History

Version	Sign-off Authority	Reason For Change	Sign-off Date
1.0	Muhammad Faizan	initial draft	

Document Sign-Off

Version	Sign-off Authority	Project Role	Sign-off Date
1.0	Dr. Faisal bin ubaid	Supervisor	
1.0	Muhammad Faizan	Project Manager	

Project Manager: _____ **Supervisor**: _____

Table of Contents

1. Introduction 6
1.1. Purpose 6
1.2. Document Conventions 6
1.3. Project Scope 6
1.4. References 6
2. Over-All Description 6
2.1. Product Perspective 6
2.2. User Classes and Characteristics 6
2.3. Operating Environment 7
2.3.1. Hardware Platform 7
2.3.2. Operating System 7
2.3.3. Blockchain Infrastructure 7
2.3.4. Database Management System (DBMS) 7
2.3.5. Network Environment 7
2.3.6. Web Browsers 7
2.4. System Constraints 7
2.4.1. Software Constraints 7
2.4.2. Hardware Constraints 7
2.4.3. Cultural Constraints 8
2.4.4. Legal Constraints 8
2.4.5. Environmental Constraints 8
2.4.6. User Constraints 8
2.4.7. Performance Constraints 8
2.4.8. Security Constraints 8
2.4.9. Regulatory Constraints 8
2.4.10. Business Constraints 8
2.5. Assumptions and dependencies 9
2.5.1. Assumptions 9
2.5.2. Dependencies 9
3. FUNCTIONAL REQUIREMENTS & SYSTEM FEATURES 9
3.1. User Registration and Authentication 9
3.1.1. Description 9
3.1.2. Functional Requirement 10
3.2. Search for Bus Routes and Schedules 10

3.2.1. Description 10
3.2.2. Functional Requirement 10
3.3. Seat Selection and Booking 10
3.3.1. Description 10
3.3.2. Functional Requirement 10
3.4. Transaction 10
3.4.1. Description 10
3.4.2. Functional Requirement 10
3.5. Real-Time Availability Updates 11
3.5.1. Description 11
3.5.2. Functional Requirement 11
3.6. Booking Confirmation 11
3.6.1. Description 11
3.6.2. Functional Requirement 11
3.7. Booking Management 11
3.7.1. Description 11
3.7.2. Functional Requirement 11
3.8. User Support and Assistance 11
3.8.1. Description 11
3.8.2. Functional Requirement 11
4. Use Case Requirement 12
4.1. Use Case Diagram 12
4.2. Use Cases 12
4.2.1. Use Case: Bus Details 12
4.2.2. Use Case: Driver Profile 12
4.2.3. Use Case: Bus Schedule 12
4.2.4. Use Case: Payment 13
4.2.5. Use Case: Customer Info 13
4.2.6. Use Case: Booking/Reservation 13
4.2.7. Use Case: Database Backup 14
4.2.8. Use Case: Reports 14
5. External Interface Requirements 14
5.1. User Interfaces 14
5.2. Software Interfaces 14
5.3. Hardware Interfaces 15
5.4. System Interfaces 15
5.5. Communication Interface 15

6. Quality attributes 16
6.1. Usability 16
6.2. Availability 16
6.3. Security 16
6.4. Reliability 16
7. Internationalization and localization requirements 16
8. Non-Functional Requirements 16
8.1. Security Requirements 16
8.2. Performance Requirements 16
8.3. Scalability Requirements 17
8.4. Reliability 17
8.5. Usability 17
8.6. Compatibility 17
8.7. Maintainability 17
8.8. User Support 17

1. Introduction

1.1. Purpose

The Software Requirement Specification Document is created with the objective of comprehensively documenting the requirements and specifications for a blockchain-based online bus ticketing system, focusing on improving user experience and security. The system empowers users to search for buses based on location, date, time, and type, allowing them to reserve seats, select preferred payment methods, and save their preferences for future bookings. It enhances the customer experience through features like loyalty programs, group booking discounts, and vacation packages. Blockchain integration ensures secure transactions, while real-time bus tracking and performance evaluations aid decision-making. Automated confirmation calls and 24/7 customer support instill user confidence. This project aims to deliver a robust, user-friendly, and secure platform by delving into blockchain technology, software development, and the intricacies of the travel industry.

1.2. Document Conventions

The Software Requirement Specification document is written in Calibri (Body) font of size 11, Calibri (Light) font size of 14 for Headings and Times new Roman font of size 14 for Sub-Headings. Each functional requirement has its own priority according to its priority mentioned.

1.3. Project Scope

The scope of the system aims to offer a secure, transparent, and efficient way to book bus tickets. By implementing blockchain technology, it ensures the integrity of transactions and enhances the overall user experience.

1.4. References

The Software Requirement Specification template that is used in building the document is referenced as fallowing:

- ISO/IEC/IEEE
- Robertson and Robertson.

2. Over-All Description

2.1. Product Perspective

We're creating a new way to buy bus tickets online that's safer and more convenient. The old way had problems like fraud and was not very user-friendly. Our new system will let you book tickets easily, choose your seat, and pay the way you prefer. It will also use special technology called blockchain to make transactions secure, and you'll be able to track your bus in real-time. This project is all about making bus ticketing easier and safer for both passengers and bus companies.

2.2. User Classes and Characteristics

The user of the system is anyone that has a device and internet connection. The user just has to be familiar enough to use the system in his/her device. The one who wants to book the ticket is the main user class of this system.

2.3. Operating Environment

2.3.1. Hardware Platform

The system is designed to run on a variety of hardware platforms, including desktop computers, laptops, smartphones, and tablets. Users can access the system through web browsers.

2.3.2. Operating System

The system should be compatible with multiple operating systems, such as Windows, macOS, Linux, Android, and iOS, to ensure broad accessibility for users.

2.3.3. Blockchain Infrastructure

The blockchain component of the system operates on a distributed network of nodes that run blockchain software. These nodes could be hosted on a variety of operating systems and hardware configurations, depending on the chosen blockchain technology (e.g., Ethereum, Hyperledger, etc.).

2.3.4. Database Management System (DBMS)

The system relies on a DBMS to store and manage user data, ticketing information, and transaction records. (MongoDB).

2.3.5. Network Environment

The system operates in a networked environment, requiring internet connectivity for users to access the platform. It must be compatible with various network technologies, including 4G, 5G, Wi-Fi, and wired connections.

2.3.6. Web Browsers

Users access the system via web browsers, so the system must be compatible with a wide range of browsers, such as Google Chrome, Mozilla Firefox, Microsoft Edge, Safari, and others.

2.4. System Constraints

2.4.1. Software Constraints

- The system must be compatible with various web browsers and mobile operating systems, limiting the choice of technologies and programming languages used in development.
- It may need to integrate with third-party software, such as payment gateways and blockchain networks, necessitating adherence to their APIs and data formats.

2.4.2. Hardware Constraints

- The software must run on a range of hardware platforms, which may have different processing power, memory, and display capabilities.
- Mobile applications need to support various device configurations and screen sizes.

2.4.3. Cultural Constraints

- Language and localization requirements may vary based on the cultural preferences of the user base, requiring multi-language support and cultural adaptability.
- Consideration of cultural norms and user expectations is essential when designing the user interface and content.

2.4.4. Legal Constraints

- Compliance with data protection regulations, privacy laws, and industry-specific legal requirements (e.g., payment processing regulations) is mandatory.
- Adherence to licensing agreements and intellectual property rights for third-party components and services used within the system.

2.4.5. Environmental Constraints

- The system may be accessed in various physical environments, including noisy or crowded settings. It must be designed to accommodate these conditions, potentially requiring enhanced usability features, such as larger buttons or voice-based interactions.

2.4.6. User Constraints

- User demographics and preferences influence design choices. For instance, if the system caters to children, it may require a more graphical and intuitive user interface.
- Accessibility constraints may apply to accommodate users with disabilities, ensuring compliance with accessibility standards and regulations.

2.4.7. Performance Constraints

- The system must perform efficiently and respond quickly, considering varying network conditions and device capabilities.
- Scalability constraints must be addressed to handle increasing user loads and transaction volumes.

2.4.8. Security Constraints

- The system must meet stringent security requirements, including encryption, authentication, and authorization mechanisms to protect user data and financial transactions.

2.4.9. Regulatory Constraints

- Compliance with local, national, and international regulations, including taxation, transportation, and consumer protection laws, is essential.

2.4.10. Business Constraints

- Budgetary constraints, time constraints, and resource limitations may affect the project's scope, schedule, and feature set.

2.5. Assumptions and dependencies

2.5.1. Assumptions

Assumption: Users have access to the internet.
Rationale: The system relies on online connectivity for booking tickets, accessing schedules, and processing transactions.
Assumption: Users have access to web browsers.
Rationale: The system is designed to be accessible through web browsers, requiring users to have compatible devices.
Assumption: Users have basic computer or mobile device literacy.
Rationale: The system assumes that users have a basic understanding of how to operate web browsers
Assumption: Users will provide accurate and up-to-date personal information during registration.
Rationale: Accurate user data is crucial for reservations, communication, and loyalty program participation.
Assumption: Third-party payment gateways and blockchain networks are reliable and available.
Rationale: The system relies on external payment and blockchain services for transactions and security.

2.5.2. Dependencies

Dependency: Availability of third-party payment gateways.
Rationale: The system depends on external payment gateways to process financial transactions, making their availability and reliability essential.
Dependency: Reliability and scalability of the Ethereum blockchain network (if applicable).
Rationale: The system's blockchain component relies on the external blockchain network for secure and transparent transactions.
Dependency: Network and internet service providers.
Rationale: The system's functionality depends on the availability and reliability of internet connectivity and network services provided by external providers.
Dependency: External geolocation services (if used for real-time bus tracking).
Rationale: The real-time bus tracking feature may rely on external geolocation services for accurate location data.

3. FUNCTIONAL REQUIREMENTS & SYSTEM FEATURES

3.1. User Registration and Authentication

The system will provide user registration and authentication.

3.1.1. Description

This feature will provide Secure user registration and login processes. Give Multi-factor authentication for added security and have Password reset and recovery options.

3.1.2. Functional Requirement

Users should have the ability to register and create accounts using their email addresses or mobile numbers. This registration process allows for personalization and ensures that users can log in securely. In case of forgotten passwords or account recovery, the system should offer password reset and recovery options to assist users in accessing their accounts.

3.2. Search for Bus Routes and Schedules

This System should provide user to search for bus routes and schedules.

3.2.1. Description

Users can search for available bus routes, operators, and schedules based on departure and destination locations, date, and time preferences.

3.2.2. Functional Requirement

The system should provide a robust search feature that enables users to search for available bus routes, operators, and schedules. Users can specify their departure and destination locations, select a date, and set their preferred departure time. This search function helps users find the most suitable bus options.

3.3. Seat Selection and Booking

3.3.1. Description

Users can select their preferred seats from a seating layout, and once chosen, the booking process begins, allowing users to confirm their seat choices.

3.3.2. Functional Requirement

Users should have the ability to choose their preferred seats from a visual seating layout. Once they've selected their seats, the booking process should be initiated. Users should be able to review their choices before confirming the booking. This feature ensures that users can select seats that best suit their preferences.

3.4. Transaction

The System should provide its users a transaction feature.

3.4.1. Description

Users have the flexibility to pay for their tickets using different methods like credit cards, debit cards, and digital wallets. The payment process is both secure and easy to use, ensuring a smooth and worry-free transaction experience with block-chain integration.

3.4.2. Functional Requirement

The system should offer flexibility in payment methods. Users should be able to pay for their tickets using various options, such as credit cards, debit cards, and digital wallets. The payment processing system should be secure with block-chain

integration, user-friendly, and capable of accommodating different payment preferences.

3.5. Real-Time Availability Updates

The system should provide real-time availability updates to its user.

3.5.1. Description

The system provides real-time updates on seat availability and pricing, ensuring users receive the latest information.

3.5.2. Functional Requirement

To keep users well-informed, the system must provide real-time updates on seat availability and pricing. These updates should be continually available as users make decisions about their bookings. It's crucial that users have access to the latest information to make informed choices.

3.6. Booking Confirmation

The system should provide the booking confirmation notification through email or messages.

3.6.1. Description

Upon successful payment, users receive booking confirmation with a unique reference number and e-ticket details.

3.6.2. Functional Requirement

Once users have successfully completed the payment process, the system should promptly generate booking confirmations. Users should receive confirmation emails or messages containing a unique reference number and detailed e-ticket information. This confirmation is vital for users to have a record of their booking.

3.7. Booking Management

The system should provide users with the ability to access and manage their bookings.

3.7.1. Description

Users can access and manage their bookings, including rescheduling or canceling tickets, if permitted.

3.7.2. Functional Requirement

To cater to changing circumstances, the system should allow users to access and manage their bookings. Users may need to reschedule their trips or cancel tickets, so the platform should facilitate these processes efficiently. This feature ensures user flexibility.

3.8. User Support and Assistance

The system should provide 24/7 user support and assistance.

3.8.1. Description

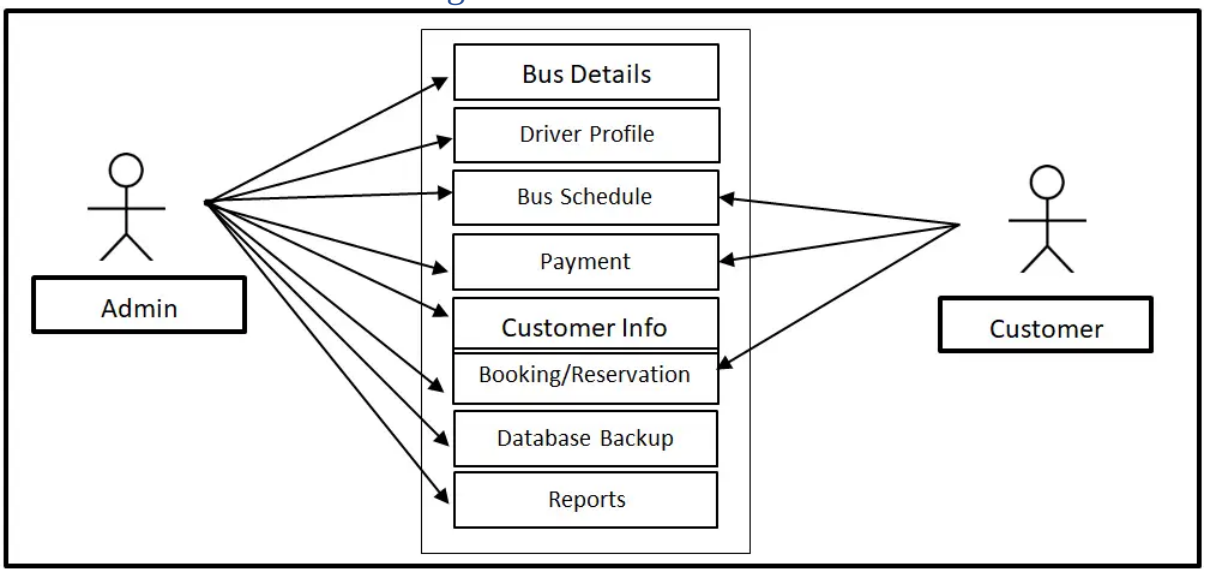
Round-the-clock customer support is available to address inquiries, issues, and provide assistance.

3.8.2. Functional Requirement

To ensure users receive the support they may need, the system should offer round-the-clock customer support and assistance. Users should be able to reach out for help with inquiries, issues, and general assistance, contributing to high customer satisfaction and problem resolution.

4. Use Case Requirement

4.1. Use Case Diagram



4.2. Use Cases

4.2.1. Use Case: Bus Details

Actor(s): Admin

Description: This feature is used to manage the details of the buses registered in the system.

Successful Completion:

1. The admin will register new buses using this feature, for old buses, this is used to update their details.
2. The admin can search, add, update and remove bus details.

Alternative: The admin can access all bus details.

Precondition: New bus for registration, existing bus for updating.

Post condition: accepted bus registration and updated bus details.

4.2.2. Use Case: Driver Profile

Actor(s): Admin

Description: This feature is used to manage the personal profile of the drivers registered in the system.

Successful Completion:

1. The admin can register new driver using this feature, for old drivers, this is used to update their personal profile.
2. Admin can search, add, update and remove a driver's data or profile.

Alternative: The admin can access all of the driver's profile.

Precondition: New drivers for registration, existing drivers for updating.

Post condition: accepted driver registration and updated driver's profile.

4.2.3. Use Case: Bus Schedule

Actor(s): Admin and Customer.

Description: This feature is used to manage the schedule of buses.

Successful Completion:

1. Customer can set a schedule to ride a bus. They will be notified if the schedule was approved, cancelled or rescheduled.

2. Admin can manage, accept, cancel or reschedule the appointment based on the availability of the bus schedules.

Alternative: None.
Precondition:

1. Customers will need to login first in order to access the feature.
2. Admin will need to login also to manage the schedule of buses.

Post condition: updated list of bus schedules.

4.2.4. Use Case: Payment

Actor(s): Admin and Customer
Description: This feature is used to manage the payment of the customers.
Successful Completion:

1. Customers can upload a proof of payment (receipt, deposit slip, etc).
2. Admin can verify the payment done by the client.

Alternative: None.
Precondition:

1. Customers will need to login first in order to access the feature.
2. Admin will need to login also to manage the payment transactions.

Post condition: updated list of verified customer payment.

4.2.5. Use Case: Customer Info

Actor(s): Admin
Description: This feature is used to manage the personal profile of the customers registered in the system.
Successful Completion:

1. The admin can register new customer using this feature, for old customers, this is used to update their personal profile.
2. Admin can search, add, update and remove a customer data or profile.

Alternative: The admin can access all of the customer's profile.
Precondition: New clients for registration, existing clients for updating.
Post condition: accepted client registration and updated client's profile.

4.2.6. Use Case: Booking/Reservation

Actor(s): Admin and Customer.
Description: This feature is used to manage the bookings/reservation of the customers.
Successful Completion:

1. Customers can use this feature to book for buses.
2. Admin can verify the bookings of the customers.

Alternative: None.
Precondition:

1. Customers will need to login first in order to access the feature.
2. Admin will need to login also to manage the bookings/reservations.

Post condition: updated list of bookings/reservations.

4.2.7. Use Case: Database Backup

Actor(s): Admin

Description: This feature is used to manage the backup database of the system.

Successful Completion:

1. The admin can add, edit, update database backup information.

Alternative: None.

Precondition: Admin will create and connect the backup database.

Post condition: new backup database.

4.2.8. Use Case: Reports

Actor(s): Admin

Description: This feature is used to view and print the reports in the system.

Successful Completion:

1. Admin can view, print and export the report of the system.

Alternative: None.

Precondition: Admin will need to login to access the reports.

Post condition: hard and soft copy of the report of the system.

5. External Interface Requirements

5.1. User Interfaces

The blockchain-based online bus ticketing system aims to provide accessibility to all users via the internet. It will have two distinct user interfaces: one for regular users and another for administrators.

- Users can browse the system's database without the need to log in. However, they must log in to perform additional transactions. These transactions include reserving bus seats, making payments, and managing their personal preferences.
- Administrators are required to log in to access their functionalities. They will have the authority to add or remove bus routes and manage information about buses as necessary. Additionally, administrators can monitor the inventory of available bus seats and the quantity in stock.

5.2. Software Interfaces

- **Payment Gateway Interfaces**: To facilitate secure online payments, the system will interface with external payment gateway services.

- **User Authentication Interfaces**: These interfaces allow the system to verify and authenticate user credentials during login and transaction processes.

- **Database Management System (DBMS)**: The system will have an interface with a DBMS to store and retrieve data related to bus routes, bookings, user profiles, and more.

- **Blockchain Integration Interfaces**: It will have interfaces with blockchain networks and nodes for data synchronization and transaction validation.

5.3. Hardware Interfaces

The System does not have any hardware requirements.

5.4. System Interfaces

- **User-Booking Module Interface**: This interface allows the user-facing module to communicate with the booking and reservation system, enabling users to search for buses, select seats, and make reservations.

- **Payment-Processing Interface**: The payment module interfaces with external payment gateways to securely process payments for ticket bookings.

- **Database-System Interface**: This interface connects the system with the database management system (DBMS) to store and retrieve data related to bus routes, bookings, user profiles, and other relevant information.

- **Blockchain Integration Interface**: There will be interfaces between the system and the blockchain network or nodes to facilitate data synchronization and transaction validation.

- **User Authentication and Authorization Interface**: This interface ensures that the system can authenticate and authorize users to perform various actions within the platform, including booking tickets and managing their accounts.

5.5. Communication Interface

- **User-System Communication**: This interface enables users to interact with the system through the user interface. Users can search for bus routes, make reservations, and receive booking confirmations.
- **Payment Gateway Communication**: The system communicates with external payment gateway services to process secure online payments for ticket bookings.
- **Email and Messaging Communication**: Communication interfaces with email and messaging services are used to send automated notifications and confirmations to users, keeping them informed about their bookings and updates.
- **Database Communication**: The system communicates with the database management system (DBMS) to store and retrieve data, such as bus route information, user profiles, and transaction records.
- **Blockchain Network Communication**: If blockchain technology is integrated for secure transactions or other purposes, communication interfaces with blockchain networks and nodes are used to synchronize data and validate transactions.
- **Administrator-System Communication**: Administrators can communicate with the system through their interface to manage content, add or remove bus routes, and monitor the system's inventory.

6. Quality attributes

6.1. Usability

The system should be user-friendly with an intuitive interface that allows users to easily navigate, search for routes, and book tickets.

6.2. Availability

The system should be accessible 24/7 to accommodate users from different time zones and travel schedules.

6.3. Security

Secure payment processing, data encryption with block-chain integration, and user authentication are essential to protect user data and financial information.

6.4. Reliability

Users should be able to rely on the system to provide accurate and up-to-date information about routes, schedules, and seat availability.

7. Internationalization and localization requirements

For the bus ticket booking system to cater to a global audience, it should be designed to detect users' geographic locations through GPS positioning. Based on their locations, the system will customize the displayed information. For instance, if a user is in the USA, the system will primarily show bus routes and ticket prices in Dollars and adjust delivery charges according to relevant local or state regulations. To fulfill internationalization and localization requirements, users will be asked to grant location access or manually input their current location, ensuring that the system functions in alignment with their specific location and preferences. This approach enables the system to provide a personalized experience for users around the world.

8. Non-Functional Requirements

8.1. Security Requirements

The system’s security requirements are as follow:

- The system should employ strong encryption to protect sensitive user data, payment information, and transactions.
- User authentication and authorization mechanisms should be in place to prevent unauthorized access.

8.2. Performance Requirements

The Systems performance requirements are following:

- The system should provide fast response times for search queries and booking processes, ensuring that users can complete transactions quickly.
- Response times should be consistent under varying loads, and the system should handle peak usage efficiently.

8.3. Scalability Requirements

Scalability Requirements of The System are as follows:

- The system must be scalable to handle an increasing number of users and booking transactions, particularly during peak travel times or special events.
- Scalability should be achieved without degrading the system's performance.

8.4. Reliability

- Availability: The system should be available 99.9% of the time, excluding scheduled maintenance windows.
- Fault Tolerance: The system should be designed to withstand server failures without impacting ongoing transactions.

8.5. Usability

- User Interface Consistency: The user interface should maintain consistency across different browsers and devices.
- Accessibility: The system should comply with accessibility standards (e.g., WCAG) to ensure usability for users with disabilities.

8.6. Compatibility

- Cross-Browser Compatibility: The system should be compatible with the latest versions of major web browsers, including Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari.
- Mobile Responsiveness: The platform should provide a user-friendly experience on various mobile devices.

8.7. Maintainability

- Code Modularity: The system's codebase should be modular, allowing for easier maintenance and future updates.
- Documentation: Comprehensive documentation for code, architecture, and system functionalities should be maintained for developers and administrators.

8.8. User Support

- 24/7 Customer Support: The system should provide round-the-clock customer support to assist users with inquiries, issues, and general assistance.
- User Support Response Time: Customer support requests should be acknowledged within 1 hour, and resolution times should be communicated to users.