Software Requirements Specification

Version 1.0

September 19, 2025

EventHive

Interactive Event Discovery & Chat Platform

Aditya Bhardwaj-23106044
Yogesh Rana-23106034
Nikshay Kataria-23106048
Rohit Tripathy-23106005

Submitted in partial fulfillment of the requirements of
DSN 5003 Software Engineering course.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) is to provide a complete and unambiguous definition of the requirements for the "EventHive - Interactive Event Discovery & Chat Platform." This document outlines the system's functional capabilities, non-functional constraints, and all interactions with external interfaces. It is intended to serve as the single source of truth for all stakeholders, including the development, quality assurance, and project management teams. This SRS will guide the platform's design, implementation, testing, and future evolution, ensuring that the final product aligns with the project's vision of combining interactive map-based event discovery with a real-time, multi-layered community chat system for the Indian market.

1.2 Scope of Project

This software system will be a comprehensive platform designed to manage the complete event lifecycle, from discovery and ticketing to post-event community engagement. The core of the platform is the integration of two primary functionalities: a visual, map-based event discovery engine and a deeply integrated, multi-layered chat ecosystem.

The scope of the project encompasses the following key areas:

Smart Event Discovery: The system will provide an interactive map interface for users to visually explore and discover local events. This will be supported by advanced search and filtering capabilities, location-based recommendations, and AI-driven suggestions.

Community-Driven Content: The platform will empower users to create and publish their own events, with a special verification system to establish trust for established organizations like colleges, NGOs, and clubs.

Integrated Ticketing: The system will facilitate seamless registration for both free and paid events, incorporating a secure payment gateway with support for India's Unified Payments Interface (UPI).

Multi-Layered Chat System: A central feature is the communication network, which includes temporary, event-specific chatrooms; persistent, hyper-local city-level groups;

and broader state-wise communities, complete with anonymity options to encourage open dialogue.

Post-Event Engagement: The platform will extend the event experience through an "After-Event Hub," a temporary private chat space for attendees to connect, share memories, and provide feedback after an event concludes.

The system is designed specifically for the Indian user base and will operate as a standalone web application accessible on modern browsers across desktop and mobile devices.

1.3 Glossary

Table 1.1: Glossary of Terms

Term	Definition
API	Application Programming Interface: A set of rules and protocols for building and interacting with software applications.
CDN	Content Delivery Network: A geographically distributed network of proxy servers and their data centers, which provide high availability and performance by distributing the service spatially relative to end-users.
Collaborative Filtering	An algorithm used by recommendation systems to make predictions about the interests of a user by collecting preferences from many users.
EIR	External Interface Requirement: A requirement that describes how the software must interact with external components, such as third-party APIs or hardware.
FR	Functional Requirement: A statement that defines a specific function or behavior the system must be capable of performing.
Geolocation	The process of identifying the real-world geographic location of an object, such as a mobile device.
JWT	JSON Web Token: A compact, URL-safe standard for creating access tokens that assert some number of claims, used for secure authentication.
NFR	Non-Functional Requirement: A statement that defines the quality attributes and constraints of the system, such as performance, security, and reliability.
OAuth 2.0	An authorization framework that enables applications to obtain limited access to user accounts on an HTTP service, such as "Login with Google."
UPI	Unified Payments Interface: An instant real-time payment system developed by the National Payments Corporation of India, facilitating inter-bank transactions.

1.4 Overview of the Document

The next chapter, the Overall Description section, of this document gives a high-level, narrative overview of the functionality and purpose of the product. It is intended for a broad audience, including project managers, clients, and other stakeholders, to ensure a

shared understanding of the system's strategic goals. This section describes the informal requirements in accessible language and is used to establish a context for the technical requirements specification in the next chapter by focusing on the product's vision, scope, and target users.

The third chapter, the Requirements Specification section, of this document is written primarily for the developers, testers, and system architects. It describes in precise and technical terms the details of the product's functionality and operational constraints. This section breaks down the system into a set of specific, testable, and unambiguous requirements, including both functional (what the system must do) and non-functional (how well it must perform) criteria, which serve as the official blueprint for the engineering team.

Both sections of the document describe the same software product in its entirety but are intended for different audiences and thus use different language. The Overall Description creates a shared vision among all stakeholders by explaining the "what" and "why" of the project in broad strokes. In contrast, the Requirements Specification provides the technical team with a clear and actionable "contract" to guide the detailed work of design, implementation, and verification.

Chapter 2

Overall Description

2.1 Product Perspective

EventHive is a new, self-contained, and independent social discovery platform. It is not an extension of an existing product but a novel service designed to create a unique market niche by merging the functionalities of event management platforms like BookMyShow and community-building applications like Discord or Reddit. The system operates as a central hub that connects users with events and with each other, creating a dynamic ecosystem for social interaction.

The platform's architecture relies on a collection of internal services and integrations with numerous external, third-party systems. Its primary interfaces are with its end-users (attendees and organizers) via a dedicated web application. However, its functionality is critically dependent on external services for mapping (Google Maps), payments (Razorpay), cloud storage (AWS S3), real-time notifications (Firebase), and more. This positions EventHive as an integration-heavy platform that orchestrates various technologies to deliver a cohesive user experience.

2.2 Product Functions

The system will provide the following core functions to its users:

Event Discovery and Personalization: Users will be able to visually explore events on an interactive map, search for specific events using advanced filters, and receive per- sonalized recommendations. The system will tailor the user experience based on their location, stated interests, and past activity.

Event Management and Publishing: The platform will provide tools for any user to create and manage public events. A verification system will be in place to lend credibility to events hosted by recognized organizations, enhancing trust within the community.

Community Engagement and Communication: The system will facilitate multi-level conversations through event-specific chatrooms for pre-event coordination, city-level groups for local discussions, and state-wise forums for broader cultural exchange, with options for anonymous participation.

Transaction Management: The platform will offer a fully integrated ticketing system that handles both free registrations and paid ticket purchases through secure UPI and digital wallet transactions.

Gamified User Experience: To encourage sustained engagement, the system will incorporate gamification elements such as achievement badges, usage streaks, community leaderboards, and rewards for successful event organizers.

Safety and Moderation: The platform will prioritize user safety with features like one-click reporting, AI-powered content moderation, and granular privacy controls.

2.3 User Characteristics

The intended users of this system are technologically proficient individuals in India who are active participants in both online and offline social activities. The target audience can be segmented into three primary groups:

Urban Youth (18-30): This group includes college students and young professionals who are avid users of mobile and web applications. They are seeking novel ways to discover social events, connect with like-minded peers, and build their personal and professional networks. They are comfortable with digital payments and expect a seamless, intuitive user experience.

Community Organizers: This segment comprises individuals and entities such as college clubs, NGOs, local businesses, and independent creators who organize events. They require a platform to effectively promote their events, manage attendees, and build a lasting community around their activities.

Social Connectors: This group consists of individuals who are keen on exploring cultural diversity and connecting with people from different regions across India. They are drawn to the platform's city- and state-level chat networks as a means for cultural exchange and building national connections.

2.4 User Classes and Characteristics

The system will support four distinct user classes, each with a specific set of permissions and capabilities. This tiered structure is essential for maintaining platform integrity, content quality, and user safety.

1. **Standard User (Attendee):** This is the primary consumer of the platform. Their main activities revolve around discovering events, participating in communities, and attending events.

2. **Community Organizer:** This is a standard user who has the additional capability to create and manage public events. This role is open to any user, fostering a community-driven content model.
3. **Verified Organizer:** This is a trusted community organizer, such as a college, NGO, or established club, that has undergone a vetting process by the system administrators. Their status is visibly marked to enhance user trust in their events.
4. **Administrator:** This is a privileged user responsible for the overall management, moderation, and health of the platform.

The key permissions for each user class are detailed in Table 2.1.

Table 2.1: User Classes and Key Permissions

User Class	**Description**	**Key Permissions**
Standard User (Attendee)	General user of the platform for event discovery and social interaction.	Register/Login; Browse, search, and fil- ter events; Purchase tickets; Join event, city, and state chat groups; Manage personal profile and privacy settings; Use anonymous mode in designated chats; Report/block users.
Community Organizer	A standard user who creates and manages public events.	All Standard User permissions; Create, edit, and manage their own events; View attendee lists for their events; Access basic organizer tools.
Verified Organizer	A trusted entity that has been vetted by administrators.	All Community Organizer permissions; Display a verification badge on profile and events; Access to hierarchical chat channels for organizers; Access to enhanced analytics and promotional tools.
Administrator overseer	A system with full platform management capabilities.	Manage all user accounts (suspend, ban, verify); Oversee content mod- eration flags and user reports; Man- age event categories and platform-wide settings; View system-wide analytics; Manage the organizer verification pro- cess.

The following diagrams illustrate the primary interactions for the main user roles.

Figure 2.1: Use Case Diagram for Standard User

Figure 2.2: Use Case Diagram for Event Organizer

2.5 Assumptions and Dependencies

The successful design, implementation, and operation of the EventHive platform are contingent upon several key assumptions and external dependencies.

2.5.1 Assumptions

- **User Permissions:** It is assumed that users will grant the application necessary permissions, including access to their device's location for map-based discovery and consent to receive push notifications for real-time updates.
- **Data Formats:** The system assumes that all integrated third-party APIs will provide data in a consistent and machine-readable format (e.g., JSON) and that their schemas will remain stable or provide versioned updates.
- **UPI Adoption:** The platform's business model for paid events assumes that the target user base has access to and is proficient with UPI-based payment methods.
- **User Conduct:** It is assumed that the combination of AI moderation and community reporting will be sufficient to maintain a safe and positive environment, with administrative intervention required only for escalated cases.

2.5.2 Dependencies

Third-Party Services: The platform's core functionality is heavily dependent on the availability, performance, and API consistency of numerous external services. These include:

- **Google Maps API & Google Places API:** Essential for map rendering, event plotting, and location-based searches.
- **Payment Gateway (Razorpay/Stripe):** Critical for all ticketing transactions.
- **Cloud Infrastructure (AWS/Google Cloud):** Required for hosting, database management, and file storage.
- **Authentication Providers (Google, Facebook):** Necessary for OAuth-based social logins.
- **Communication Services (FCM, SendGrid, Twilio):** Required for push no- tifications, emails, and SMS.

- **AI Services (OpenAI API):** Used for content moderation and potentially for recommendation algorithms.

Network Connectivity: The system requires a stable and reasonably fast internet connection for both the end-user's device and the backend servers to ensure real-time functionality, especially for the chat and map features.

Hardware Capabilities: Users are expected to have modern smartphones or computers with GPS capabilities (for location services) and a web browser that supports modern web standards (e.g., WebSockets for real-time chat).

Chapter 3

Requirements Specification

This section provides the detailed requirements for the EventHive platform. It includes functional requirements, non-functional requirements, and specifications for external interfaces.

3.1 Functional Requirements

Functional requirements specify the explicit actions the system must be able to perform.

3.1.1 User Account and Profile Management

FR-1.1: The system shall allow a new user to register by providing an email address and password or by using an external OAuth 2.0 provider (Google, Facebook).

FR-1.2: The system shall enforce that all usernames are unique and shall provide a real-time availability checker during the registration process.

FR-1.3: Upon successful login, the system shall issue a secure JSON Web Token (JWT) to manage the user's session and authenticate subsequent API requests.

FR-1.4: The system shall provide an interface for authenticated users to view and edit their personal profile, which includes their username, profile picture, and a list of interests for personalization.

FR-1.5: The system shall prompt for and handle location permissions upon the user's first interaction with the map interface.

FR-1.6: The system shall maintain and display a history of events a user has registered for or attended.

3.1.2 Event Discovery and Management

FR-2.1: The system's primary interface shall be a dynamic, interactive map that displays nearby events as clickable pins.

FR-2.2: The system shall automatically detect the user's current location to center the map view. It shall also provide an interface for the user to manually search for or select a different location.

FR-2.3: Clicking an event pin on the map shall display a preview pop-up containing the event banner, title, date, time, and price.

FR-2.4: The system shall provide an advanced search functionality allowing users to find events by text query.

FR-2.5: The system shall allow users to filter the displayed events by category (e.g., music, workshop), date range, price (free/paid), distance from their location, and popularity.

FR-2.6: The system shall implement a recommendation engine using collaborative filtering algorithms to suggest events to users based on their interests and past activity.

3.1.3 Event Creation and Organizer Functions

FR-3.1: Any authenticated user shall be able to create a new event by providing mandatory details, including title, description, date, time, location, category, and pricing information.

FR-3.2: The event creation form shall allow organizers to upload rich media content, such as a banner image and additional photos for the event description.

FR-3.3: The system shall provide an application process for established entities (e.g., colleges, NGOs) to apply for "Verified Organizer" status.

FR-3.4: Events created by a Verified Organizer shall be prominently marked with a verification badge to signify trust.

FR-3.5: The system shall provide a dedicated dashboard for organizers to manage their created events, view attendee lists, and monitor engagement metrics.

3.1.4 Ticketing and Payment System

FR-4.1: The system shall support both free event registration and paid ticket purchasing.

FR-4.2: For paid events, the system shall securely integrate with an external payment gateway (e.g., Razorpay) to process transactions via UPI and other digital wallets.

FR-4.3: Upon successful registration or payment, the system shall generate a unique digital ticket (e.g., with a QR code) for the user.

FR-4.4: Users shall be able to access and view their purchased tickets within a dedicated section of their profile.

3.1.5 Community and Chat System

FR-5.1: The system shall automatically create a temporary, event-specific chatroom for every published event.

FR-5.2: Only users who have registered for an event shall be granted access to its corresponding chatroom.

FR-5.3: Event chatrooms shall support a hierarchical channel structure, allowing organizers to create separate, restricted channels for announcements and a general channel for public discussion.

FR-5.4: Event chatrooms shall be automatically archived 48 hours after the event concludes and will become read-only.

FR-5.5: For events with more than 10 registered attendees, the system shall automatically create a private "After-Event Hub" group chat that remains active for 48 hours post-event.

FR-5.6: The system shall host persistent, public city-level chat groups for users to engage in hyper-local discussions.

FR-5.7: The system shall host persistent, public state-wise chat groups as part of the "Global Indian Chat Network."

FR-5.8: In city-level and state-wise chat groups, users shall have the option to enable an "Anonymous Mode," which will replace their username and profile picture with a generic identifier.

3.1.6 Gamification and Rewards

FR-6.1: The system shall implement a coupon system to reward organizers who host events with over 100 attendees.

FR-6.2: The system shall award users with achievement badges for milestones such as attending a certain number of events, participating in different types of events, or being an active contributor in community chats.

FR-6.3: The system shall track and display usage streaks for users who engage with the app on consecutive days.

FR-6.4: The system shall display public, non-personally identifiable leaderboards ranking community engagement on a city-wise and state-wise basis.

3.1.7 Safety, Moderation, and Trust System

FR-7.1: All chat interfaces shall include a one-click function for users to report a specific message or block another user.

FR-7.2: The system shall implement an AI-powered content moderation service to automatically scan and flag messages containing prohibited keywords or exhibiting harmful behavior patterns.

FR-7.3: The system shall provide users with granular privacy settings to control the visibility of their profile information and event activity.

FR-7.4: The system shall implement a verification process to grant trust badges to reliable community members and organizers, which will be displayed on their profiles.

3.1.8 Administrator Functions

FR-8.1: The system shall provide a secure, role-based administrative interface accessible only to authorized personnel.

FR-8.2: Administrators shall have the ability to view, manage, suspend, or delete any user account.

FR-8.3: Administrators shall have access to a moderation queue to review and act upon all user-generated reports and AI-flagged content.

FR-8.4: Administrators shall be able to manage the "Verified Organizer" application process, including approving or denying requests.

FR-8.5: The system shall provide an analytics dashboard for administrators to monitor key platform metrics, such as daily active users, events created, and overall system health.

3.2 Non-Functional Requirements

Non-functional requirements define the quality attributes and constraints of the system.

3.2.1 Performance

NFR-1.1: The interactive map interface, including the plotting of all events within a 10km radius of the user, shall load in under 3 seconds on a standard 4G mobile connection.

NFR-1.2: Real-time chat messages shall be delivered to all clients in the same chat room with a maximum latency of 500ms under normal network conditions.

NFR-1.3: The server-side response time for all critical user-facing API endpoints (e.g., fetching event details, user profile) shall average below 300ms.

NFR-1.4: The real-time communication infrastructure shall be capable of supporting a minimum of 1,000 concurrent users across all chat rooms without significant performance degradation.

3.2.2 Security and Privacy

NFR-2.1: All user passwords stored in the database must be salted and hashed using a strong, industry-standard cryptographic algorithm (e.g., bcrypt with a cost factor of at least 12).

NFR-2.2: All data transmitted between the client application and the server shall be encrypted using Transport Layer Security (TLS) 1.2 or higher.

NFR-2.3: The system shall provide an option for users to enable Two-Factor Authentication (2FA) for their accounts to enhance security.

NFR-2.4: The system must be designed to comply with data protection principles analogous to GDPR, ensuring user data is handled securely and transparently.

NFR-2.5: All direct or private messages between users shall be protected with End-to-End Encryption.

3.2.3 Reliability and Availability

NFR-3.1: The core backend services (authentication, event discovery, chat) shall maintain a service availability of at least 99.8%.

NFR-3.2: The system shall perform automated, regular backups of the primary PostgreSQL database to prevent data loss, with a defined recovery point objective (RPO) of 24 hours.

NFR-3.3: The system's backend architecture must include error handling and retry mechanisms to gracefully manage transient failures from external API dependencies without causing a system-wide outage.

3.2.4 Usability

NFR-4.1: The new user onboarding process, from initiating registration to viewing the main event map for the first time, shall be completable in under 90 seconds.

NFR-4.2: The application's user interface shall be responsive and provide an optimized experience for both desktop and mobile web browsers.

NFR-4.3: The application must adhere to web accessibility standards (WCAG 2.1 Level AA), providing support for screen readers, keyboard navigation, and dynamic font sizes.

3.2.5 Maintainability

NFR-5.1: All source code must adhere to the formatting and linting rules defined in the project's ESLint and Prettier configurations.

NFR-5.2: All new functional components must be accompanied by a suite of unit and integration tests, with the goal of maintaining a minimum of 80% code coverage.

NFR-5.3: All public-facing API endpoints must be documented using the OpenAPI 3.0 specification, with interactive documentation available via a Swagger UI.

3.3 External Interface Requirements

This section describes how the EventHive system will interact with external components.

3.3.1 User Interface

EIR-1.1: The primary user interface shall be a responsive web application built using the Next.js framework. It must be compatible with the latest two major versions of modern web browsers, including Google Chrome, Mozilla Firefox, and Apple Safari.

EIR-1.2: The user interface components shall be implemented using the Tailwind CSS framework and the shadcn/ui component library.

3.3.2 Hardware Interfaces

EIR-2.1: The system shall interface with the client device's Global Positioning System (GPS) hardware through the browser's standard Geolocation API to determine the user's precise location for map centering and location-based searches.

3.3.3 Software Interfaces

EIR-3.1: The system shall interface with the Google Maps JavaScript API and Google Places API via RESTful HTTP requests to render maps, display custom event markers, and provide location search and autocomplete functionality.

EIR-3.2: The system shall interface with a designated payment gateway API (e.g., Razorpay or Stripe) via secure RESTful calls to create payment orders, process UPI and wallet transactions, and verify payment status.

EIR-3.3: The system shall use the OAuth 2.0 protocol to interface with Google and Facebook for delegated user authentication.

EIR-3.4: The system shall interface with a cloud storage provider API (e.g., AWS S3 or

Cloudinary) to upload, store, and retrieve user-generated media content such as profile pictures and event banners.

EIR-3.5: The system shall interface with the OpenAI API (or an equivalent service) via RESTful calls to submit message content for automated moderation and analysis.

EIR-3.6: The system shall interface with Firebase Cloud Messaging (FCM) to send real-time push notifications to registered user devices.

EIR-3.7: The system shall interface with the SendGrid API for sending transactional emails (e.g., registration confirmation, password resets) and the Twilio API for sending SMS notifications.

3.3.4 Communications Interfaces

EIR-4.1: All standard client-server communication for data retrieval and submission (e.g., API calls) shall use the secure HTTPS protocol.

EIR-4.2: Real-time, bidirectional communication for the chat system shall be established and maintained using the WebSocket protocol, managed by the Socket.io library on both the client and server.