SlugSync: A Centralized Event Discovery Platform for UCSC

Vihari Tejomurtula
Raksshana Harish Babu
Praneeth Mandalapu
Nisheeth Velicheti

Technical Design and System Overview

1 Abstract

University students often miss valuable academic, professional, and social opportunities due to fragmented event discovery across multiple pages, like Instagram and email lists. SlugSync is a centralized, mobile-first event discovery platform designed to cluster events across university organizations and display them in a single, searchable feed. The system consists of a Swift-based iOS client, FastAPI backend, SQL Database, and Render hosting, with role-based authentication for university students and approved event hosts. SlugSync was built with trust and integrity in mind, separating normal users from authorized club representatives who are the only ones able to create, edit, and delete events through ownership-based access controls. This document outlines the system architecture, data flow, authentication models, and design decisions.

2 Problem Motivation

As current freshmen at UCSC, we missed out on valuable opportunities simply because we did not know that they existed. At the moment, students are forced to look through fragmented Instagram pages in the hope of finding event announcements. This approach is especially limiting for incoming freshmen, who do not even know the names of the campus organizations, and therefore have no reliable way to discover their events. As a result, many high-value opportunities go unseen; not because of a lack of interest, but because of poor visibility.

3 System Overview

SlugSync is built as a client-server system designed to centralize event discovery on campus. A clean separation exists between the mobile frontend, backend services, and persistent data storage. The four main parts of SlugSync are the iOS Frontend, FastAPI Backend, a relationship-preserving database, and a cloud-based server deployment.

3.1 Frontend

Our Swift-based iOS client serves as the primary interface for students and clubs to interact with. Upon opening the app, users are presented with a registration page that prompts them to log in with their Google account. Upon registration, users view an event feed that displays all events posted in the app. A search feature enables users to filter events based on keywords, date, and event classification. Authorized users can create events and post them to the feed, and update or delete the events they created. All the processing logic is left to the backend to keep the client lightweight and so that the core logic remains enforced on all users.

3.2 Backend

Our FastAPI backend, the central coordination layer of the system, is responsible for user authentication, event processing logic, and data validation. It answers requests made by the frontend via HTTP responses containing the requested content. Club representatives are authorized to create, update, and delete only the events they own, while normal users are restricted to read-only access. By enforcing permissions at the API level, the system ensures data integrity and prevents unauthorized changes.

3.3 Database

Persistent data is stored in an SQL database, which maintains structured relations between users, events, and ownership. The database enables efficient querying, filtering, and consistency across the platform, allowing the backend to serve up-to-date event information to the client. It hosts different tables for users, events, and user-linked favorites. This separation ensures data integrity and prevents corruption.

3.4 Deployment

The backend and database are deployed on Render, a web-based cloud hosting platform that allows us to manage infrastructure, environment variables, and continuity. The frontend makes HTTP requests through this server, which relays it to the backend. The HTTP responses are then sent by the backend to the frontend via this server.

4 Authentication & Authorization Model

SlugSync distinguishes between authentication (verifying who a user is) and authorization (determining what that user is allowed to do). This separation is critical for maintaining security, data integrity, and trust in a multi-user system.

User Authentication is handled via Google OAuth, only allowing UCSC Gmails to register/log in to our app. An ID is given to the user upon login, which is sent to the backend and exchanged for a short-lived JWT. This allows the backend to authenticate users in a stateless manner.

Role-Based Authorization is implemented manually, with club representatives manually being made hosts at our discretion. Club representatives would submit a request to be made an ”event host” through a Google form, which would then be processed by us over the next 48-72 hours.

Beyond role-checks, SlugSync enforces ownership-based authorization for all write operations. While hosts are allowed to create events, they can only update or delete the events that they created. This prevents any unauthorized edits, protects against vandalism, and mirrors access control panels used in production-grade systems.

5 Data Model & Event Life Cycle

SlugSync is centered around an Event object, which represents a name, time, location, host name, and event description.

Creation: SlugSync authorizes a club representative, and once authorized, they can create events for their club. Which is then validated by the backend.

Storage: Events are stored in a relational SQL database, which preserves user-event relations and creation time stamps for filtering.

Retrieval: The event feed returns upcoming events by default, filtering out completed ones. Additionally, users can filter by keyword, category, and date.

6 Design Decisions and Tradeoffs

- We made this an iOS app as about 88% of teenagers have an Apple iPhone.
- FastAPI is a robust yet effective architecture that enables fast, scalable API development
- We are manually approving hosts right now because it aligns with SlugSync’s need for API contracts, strong data validation, and fast iteration

7 Limitations and Future Works

Future improvements to SlugSync include building an organizer-facing dashboard to manage events as well as introducing automated verification workflows to scale host approval. We also plan to implement RSVP tracking to provide turnout analytics to both organizers and students.

8 Conclusion

SlugSync addresses a commonly faced problem that all university students face: fragmented event discovery and unreliable access to event details. By centralizing event details in one place with role-based control, the system benefits both students and organizations on campus, aiming to increase engagement. This project demonstrates how a clean client-server-based architecture with strong backend validation can result in a practical, deployable solution tailored to a real user's needs.