DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

IITI Student Community Application

CS 416/616: Service-Oriented Systems

Name	Roll Number
Krish Agrawal	210001034
Rahul	210001056
Vikas Maurya	210001076
Agrima Bundela	210002009
Adarsh Mishra	2402101004
Akshat Jain	2430201002

Course Instructor: Prof. Abhishek Srivastava

Project Report

GitHub Repository
Current Application

April 25, 2025

1 Overview of the project

The IIT Indore Student Community App is a centralized platform designed to enhance student engagement and streamline communication across the campus. It offers features such as ride-sharing, discussion forums, general announcements, and club-student interaction. The app provides a comprehensive suite of tools for managing event updates, registering for activities, navigating the campus, and engaging in student-led discussions. Additionally, it supports e-commerce for club merchandise and promotes a vibrant community experience. Designed for the IIT Indore student community, the app aims to improve campus connectivity and the overall student experience.

2 Product Features

- **Centralized Event Updates**: Aggregates all campus events in one place with filter and search capabilities.
- **Personalized Notifications**: Tailors alerts based on user interests and preferences.
- **Event Registration & Calendar Syncing**: Simplifies enrollment for events and integrates with Google Calendar.
- **Community Discussion Board**: Facilitates conversations and feedback among students and clubs.
- **E-Commerce Module**: Provides a secure platform for clubs to manage and sell merchandise.
- **Ride-Sharing Module**: Connects students for organized, eco-friendly traveling between campus and off-campus locations.

3 Tech Stack

Platforms: Android and iOS

- **Flutter**: For cross-platform mobile app development.
- **Node.js**: Backend cloud functions for handling notifications.
- **React.js**: Frontend for the Admin Dashboard.
- **Firebase Firestore**: For real-time data storage and synchronization.
- **Firebase Authentication**: For secure user login via Google OAuth 2.0 with institute credentials.
- **Firebase Storage**: For user-uploaded media like discussion post images.

4 APIs Used

- **Local Calendar Service**: For syncing registered events to users' calendars.
- **Firebase Cloud Messaging (FCM)**: For delivering personalized push notifications.
- **Google Maps API**: For event location tagging and campus navigation.
- **Google Place API**: For getting place name from search.

- **Google Geocode API**: For getting place name from coordinates.
- **Razorpay API**: For secure in-app payments for club merchandise purchases.
- **Google OAuth 2.0**: For secure, institution-based authentication and authorization.

5 Screenshots

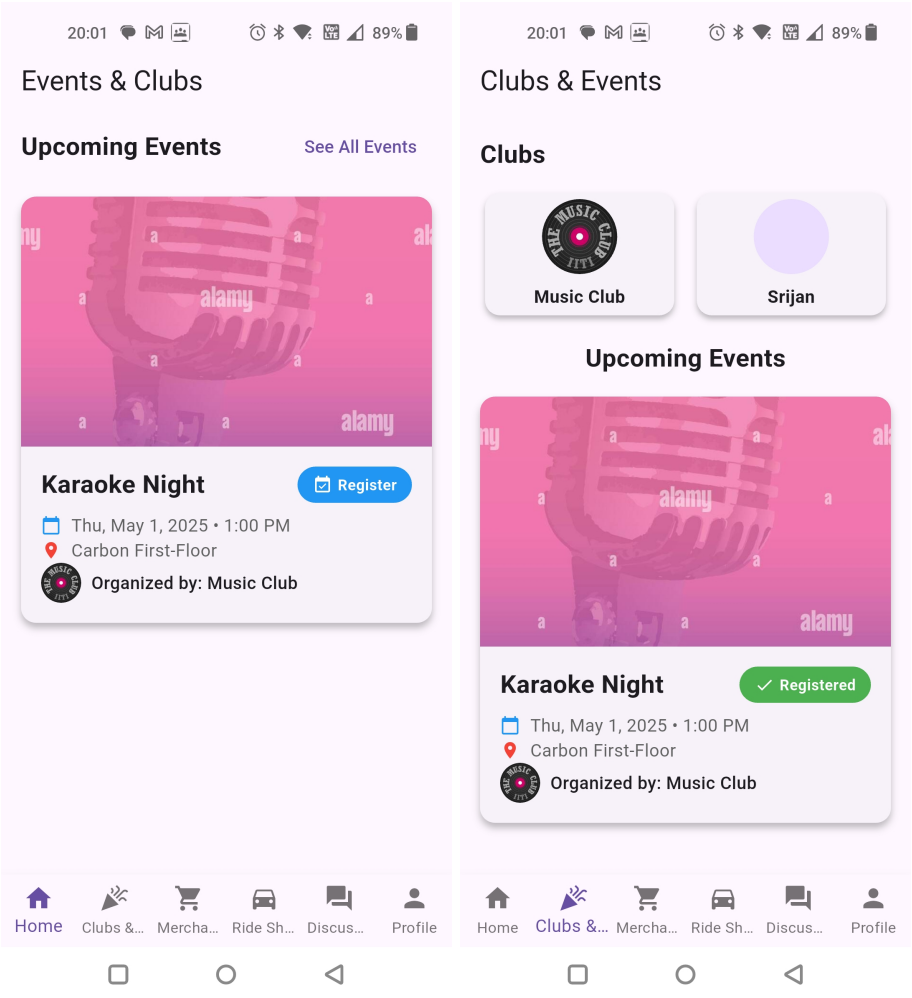


Figure 1: Home Page and Clubs & Events Screen

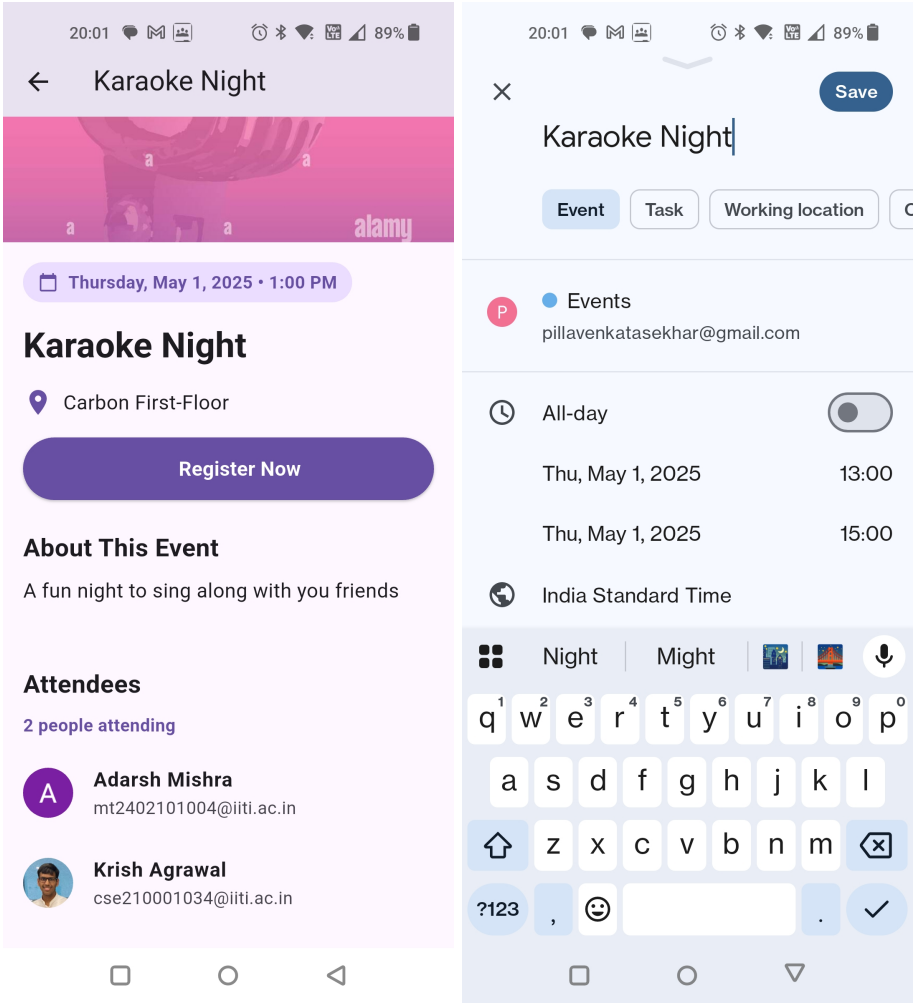


Figure 2: Events Details and Post Registration Google Calendar Link

6 High-Level Workflow

6.1 Events and Clubs Module

The following steps outline how the "Events and Clubs Module" operates:

1. **Subscribed Events Section:** In the home tab, users see a list of upcoming events from clubs they have subscribed to. This section ensures personalized content delivery based on user preferences.

2. **Navigation to All Events:** Users can click "See All Events" to navigate to the dedicated "Clubs Events" tab. This tab displays all available events across different clubs.

3. **Clubs Section:** The "Clubs Events" tab includes a grid layout showcasing all clubs. Users can explore club-specific details, including their upcoming events.

4. **Event Discovery:** Events are displayed in chronological order, ensuring that users can easily find relevant activities. Filters such as date or club affiliation may be applied to refine results.

5. **Real-Time Updates:** The module uses Firebase Firestore's live query feature to display real-time updates for newly added events or changes in event details.

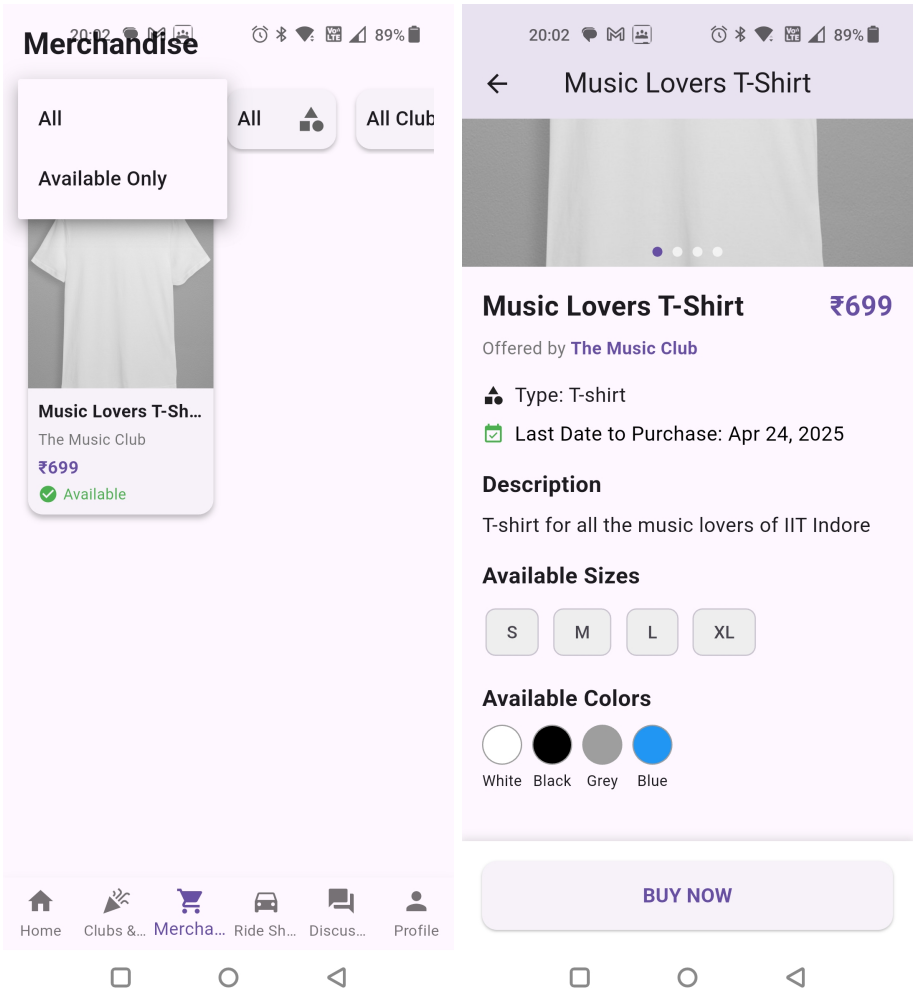


Figure 3: Merchandise and Merchandise Details Page

6. **Event Details Page:** When users select an event, they are navigated to the event details page. This page provides comprehensive information about the event, including:
   - Event name, date, time, location, and description.
   - Registration options (e.g., "Register Now").
   - Attendee list showing who has registered.
   - Interactive map for visualizing event location (if coordinates are available).
7. **User Interaction:** Users can register for events directly from the event details page. Registration status is updated in real-time in Firestore.
8. **Club-Specific Events Section:** Users can view all upcoming events organized by a specific club. This section filters events based on organizer ID and displays them in chronological order.
9. **Eco-Friendly Impact:** By promoting participation in community-driven activities, this module fosters collaboration and engagement among students.

6.2 Notification Service

The following describes how the notification system works at a high level:

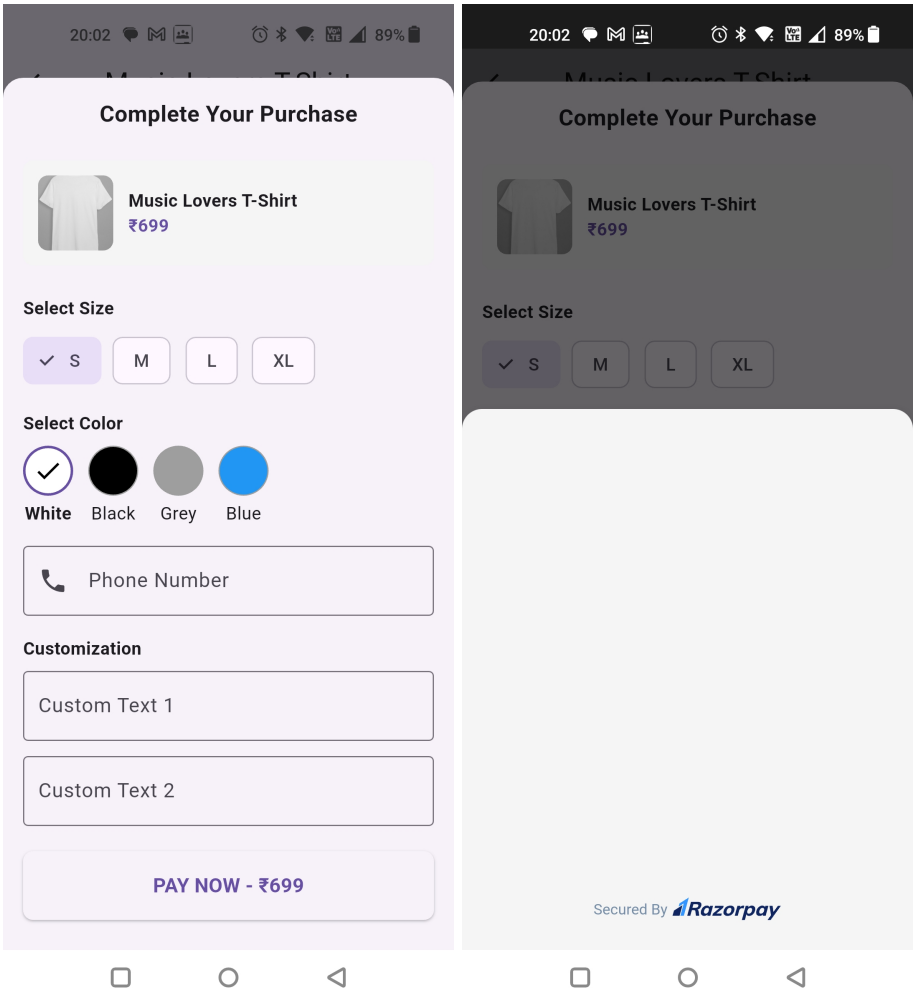


Figure 4: Purchase Form and RazorPay API Integration

1. **User Preferences:** Through settings, users specify their notification preferences, such as subscribing to a particular club.
2. **Preference Storage:** These preferences are stored in Firebase Firestore under each user's profile. Additionally, users are subscribed to relevant Firebase Cloud Messaging (FCM) topics based on their preferences (e.g., `club_sports`).
3. **Event Creation:** When a new event is created in the system (e.g., by a club organizer), it is added to Firestore. A Firebase Cloud Function is triggered automatically upon event creation.
4. **Notification Preparation:** The Cloud Function retrieves event details (e.g., name, location, time) and formats a notification message. It determines which FCM topics or individual users should receive the notification based on their preferences.
5. **Notification Delivery:** The formatted notification is sent via FCM either to specific topics (e.g., all subscribers of a club) or directly to individual user tokens if topic delivery fails.
6. **User Interaction:** When users receive a notification:
    - If the app is in the foreground, a local notification is displayed using Flutter Local Notifications.

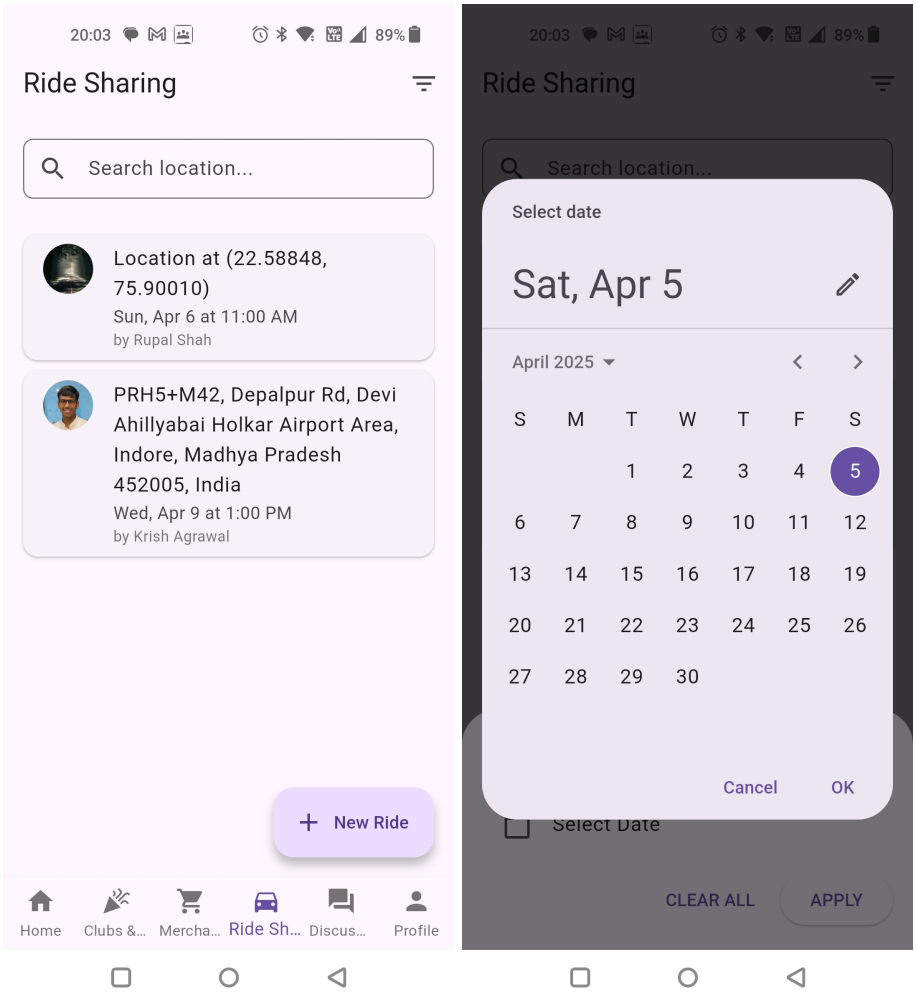


Figure 5: Ride Sharing and Filter with Date

- If the app is in the background or closed, tapping the notification navigates the user to the relevant event details screen.

7. **Read Status Tracking:** Notifications include metadata such as event IDs. When users interact with a notification, it is marked as "read" in Firestore for tracking purposes.

8. **Token Management:** FCM tokens are refreshed periodically and updated in Firestore to ensure seamless delivery of notifications.

6.3 Community Discussion Board

The following steps outline how the "Community Discussion Board" operates:

1. **Post Creation:** Users can create new posts by entering content into a text field. Posts are stored in Firebase Firestore along with metadata such as timestamp and user information.

2. **Post Display:** Posts are displayed in chronological order (latest first). Each post includes the author's name/email, timestamp, and content.

3. **Commenting Functionality:** Users can add comments to posts. Comments are stored in Firestore under the respective post's collection.

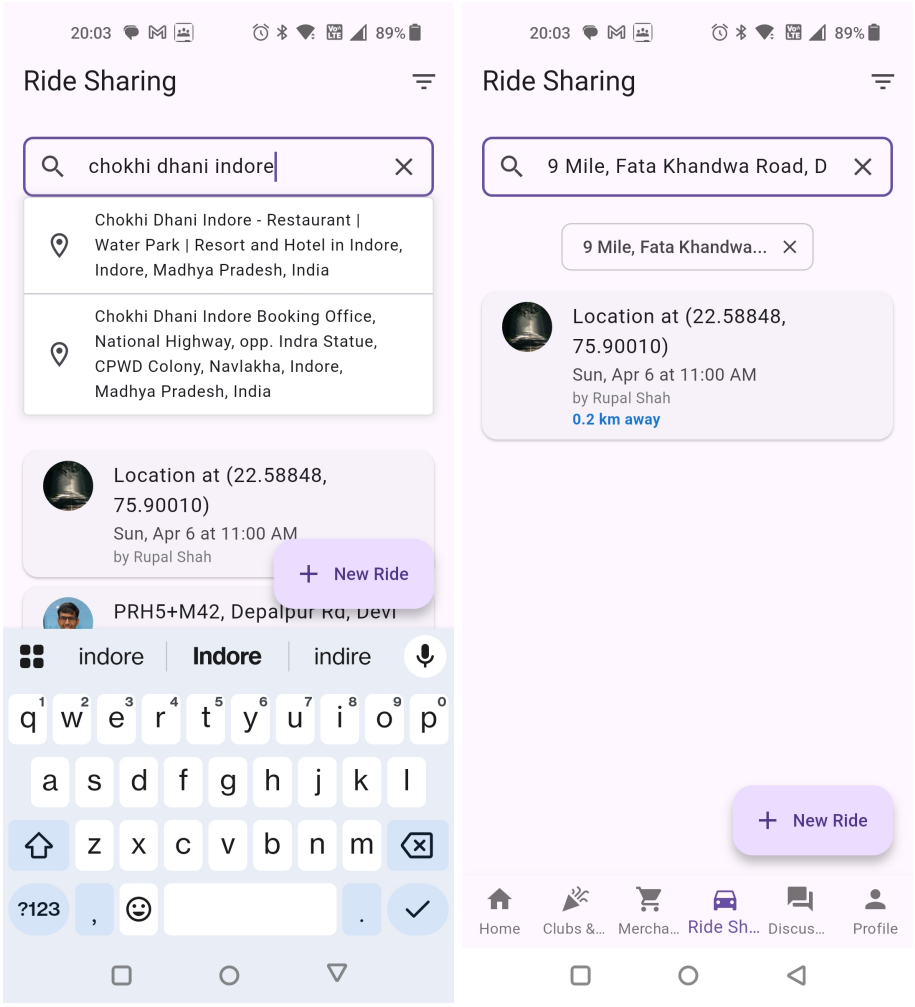


Figure 6: Ride Sharing Places and within 5 km search

4. **Post Management:** Users can edit or delete their own posts. Edits update the post content in real-time, while deletions remove the post permanently from Firestore.

5. **Real-Time Updates:** New posts or comments are reflected instantly using Firestore's live query feature.

6. **Community Engagement:** The module encourages interaction among users through discussions. Future enhancements may include likes/reactions or threaded replies.

7. **Error Handling:** The system handles errors gracefully by displaying appropriate messages when operations fail (e.g., network issues).

6.4 Merchandise Module

The following steps outline how the "Merchandise Module" operates:

1. **Product Browsing:** Users can view all available merchandise in a grid layout. Each product card displays essential details such as name, price, availability, and club affiliation.

2. **Filtering Options:** Users can apply filters to narrow down their search:

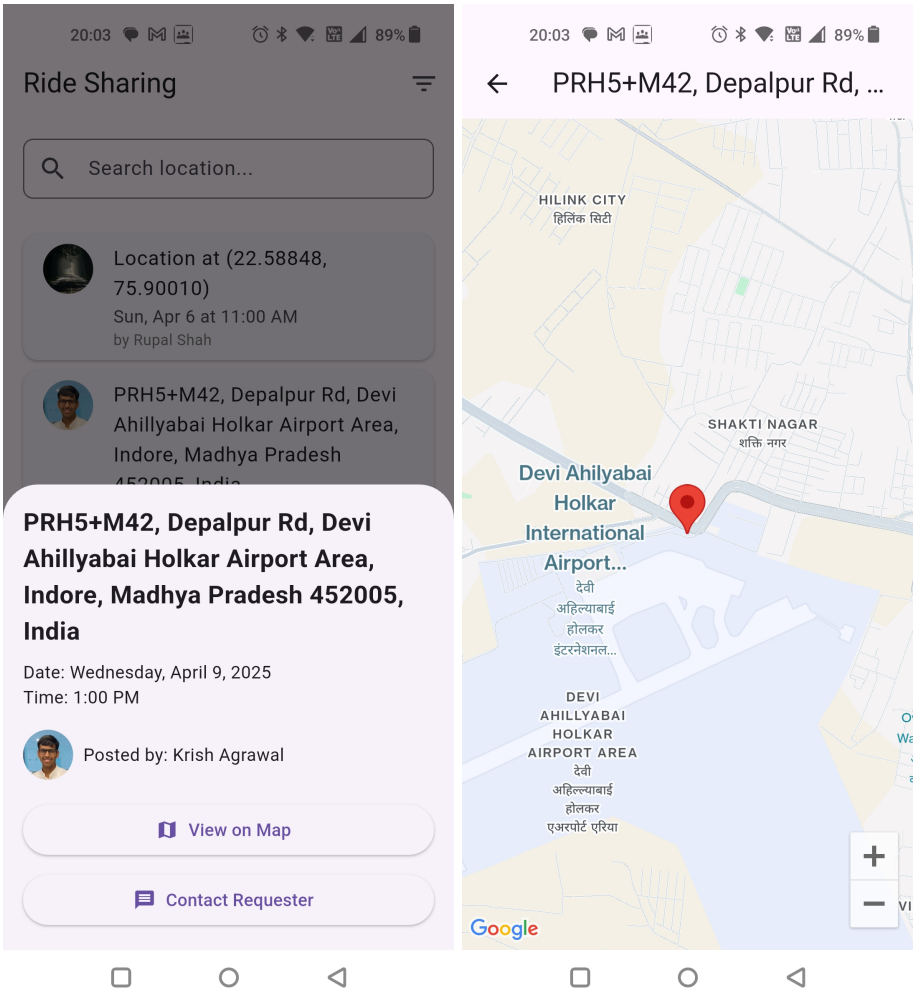


Figure 7: Ride Sharing Details and View Map

- Availability: Show only available products.
- Type: Filter by categories such as clothing, accessories, etc.
- Club: Show products offered by specific clubs.

3. **Product Details:** When a user selects a product, they are navigated to the product details page. This page includes:
   - Product images (carousel for multiple images).
   - Name, price, description, and club offering the product.
   - Available sizes and colors (if applicable).
   - Purchase deadline and availability status.
   - Customization options (e.g., text or design preferences).
4. **Purchase Process:** Users can initiate the purchase process from the product details page:
   (a) Select size, color, or customization options (if applicable).
   (b) Enter contact details (e.g., phone number).
   (c) Proceed to payment using Razorpay integration.

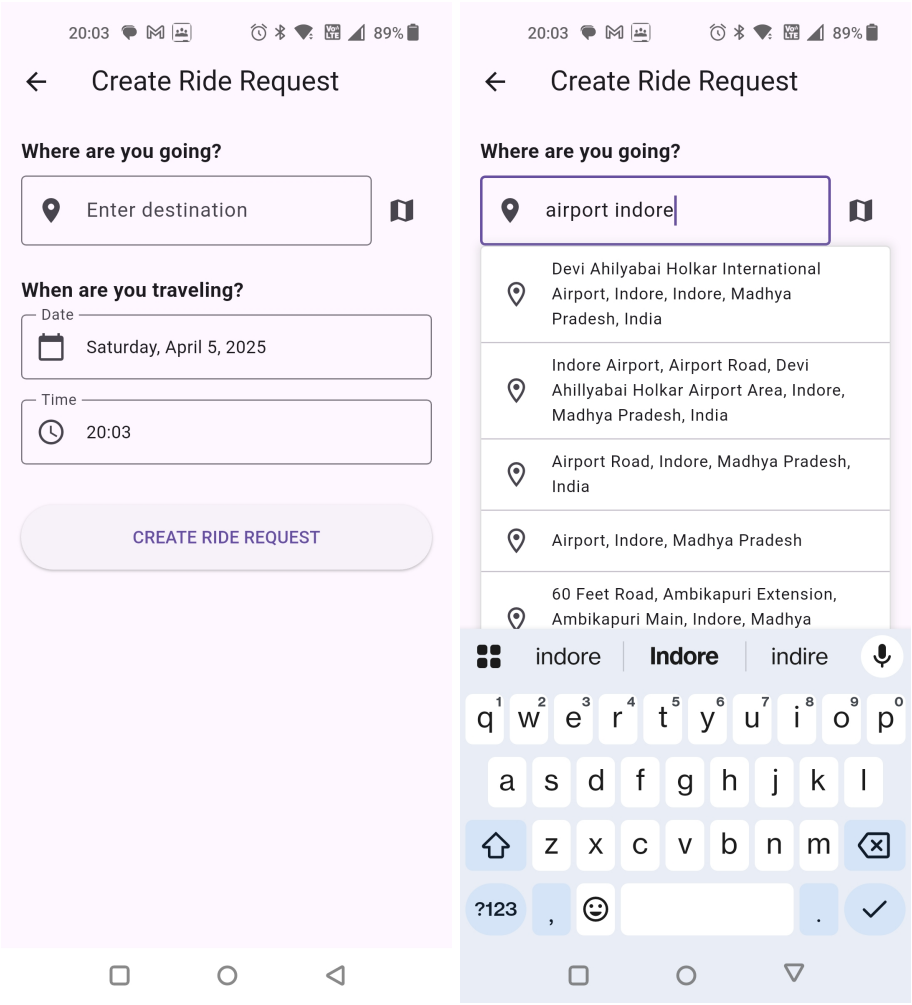


Figure 8: Ride Request and Ride Request Places API Integration

(d) Upon successful payment, the order is saved in Firestore with details such as user ID, product ID, and payment status.

5. **Real-Time Updates:** Any changes in product availability or new merchandise additions are reflected in real-time using Firebase Firestore's live query feature.

6. **Error Handling:** The system handles errors gracefully by displaying appropriate messages for issues such as network failures or payment errors.

6.5 Ride-Sharing Module

The following steps outline how the Ride-Sharing Module operates:

1. **User Interaction:** Users can post a ride offer or request a ride through a user-friendly interface. They provide details such as destination, date, time, and pickup location.

2. **Location Handling:** The system uses Google Places API to suggest locations as users type. Users can also select locations on an interactive map. The selected location is stored as geographic coordinates.

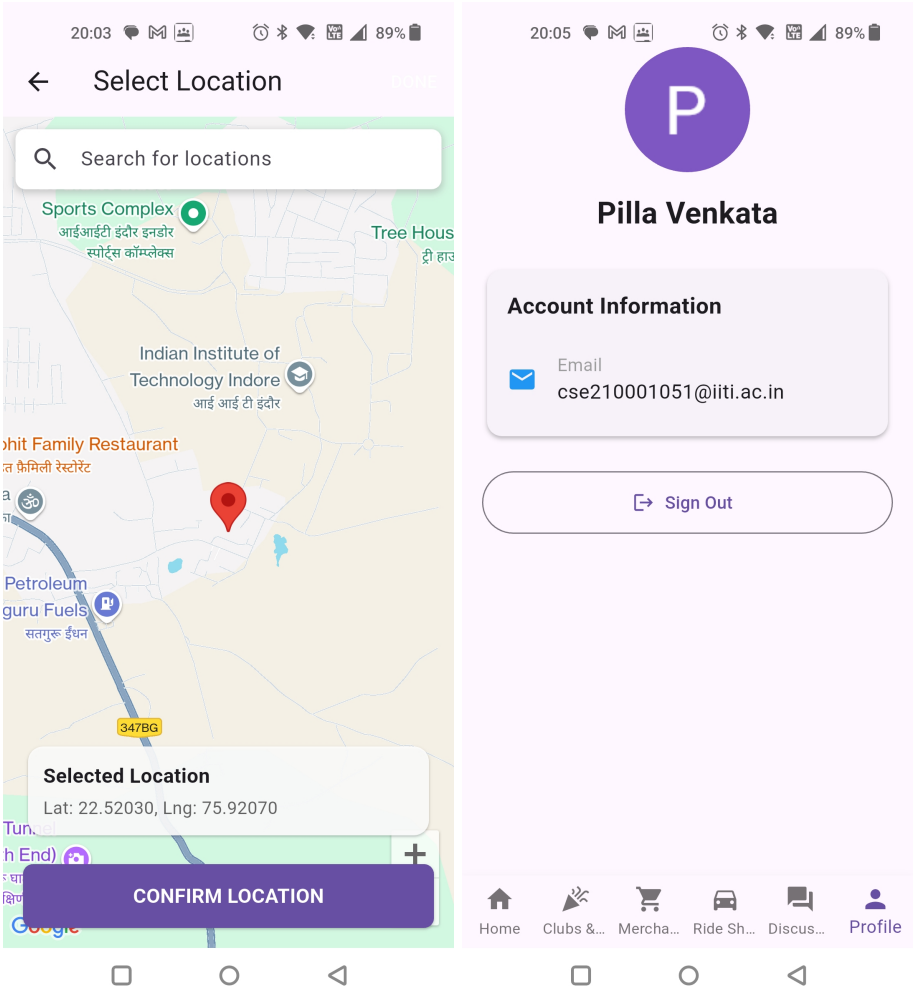


Figure 9: Ride Request Select on Map and Profile Tab

3. **Data Storage:** Ride offers and requests are stored in Firebase Firestore with details such as user ID, location, date, time, and coordinates.
4. **Ride Matching:** When a user searches for rides:
   - The system retrieves all available rides from Firestore.
   - It filters rides based on date, time proximity, and route similarity using geographic coordinates.
   - Matching results are displayed in a list with relevant details (e.g., distance from the user's location).
5. **Coordination:** Users can view ride details and contact the ride poster for further coordination. Contact options may include messaging or calling (to be implemented).
6. **Real-Time Updates:** Changes to ride offers or requests (e.g., new rides posted) are reflected in real-time using Firestore's live query feature.
7. **Filters:** Users can apply filters such as date or location proximity to narrow down search results.
8. **Eco-Friendly Impact:** By encouraging shared rides, the module reduces the number of vehicles on the road, contributing to a greener environment.

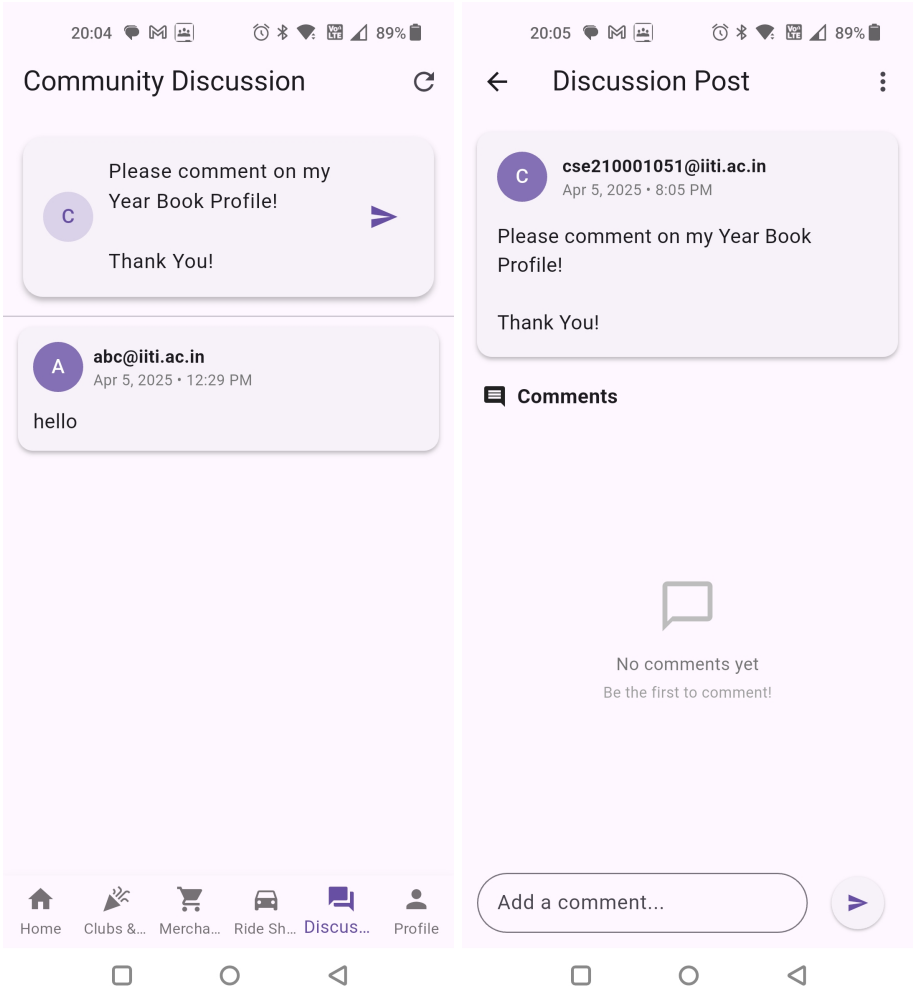


Figure 10: Discussion Board and Discussion Post

7 Admin Dashboard

7.1 Overview

This admin dashboard provides comprehensive management tools for:

- **Events Management**
- **Merchandise Management**
- **Clubs Management**

Technology Stack: React.js + Firebase (Firestore, Storage, Authentication)

7.2 Key Functionalities

7.2.1 Events Management

Feature	Description	Tech Used
Create Events	Add events with details (name, date, location, images)	Firestore + Storage
Edit/Delete Events	Modify or remove existing events	`updateDoc() + deleteDoc()`
Image Uploads	Event posters with progress tracking	Firebase Storage + `uploadBytesResumable()`
RSVP System	Users can join/leave events	`arrayUnion + arrayRemove`
CORS Handling	Prevents upload errors with proper CORS config	`gsutil cors set`

7.2.2 Merchandise Management

Feature	Description	Tech Used
Add Products	Create merchandise items (T-shirts, hoodies, etc.)	Firestore + Storage
Multi-Image Upload	Support for multiple product images	`Promise.all()`
Inventory Tracking	Sizes, colors, pricing, and stock management	Firestore arrays + numbers
Edit/Delete Products	Update product details or remove items	`updateDoc() + deleteDoc()`

7.2.3 Clubs Management

Feature	Description	Tech Used
Club Registration	Add new clubs with logos & banners	Firestore + Storage
Subscription System	Users can subscribe/unsubscribe	`arrayUnion + arrayRemove`
Image Optimization	Auto-upload for club photos/banners	Firebase Storage
Edit/Delete Clubs	Modify club details or remove clubs	`updateDoc() + deleteDoc()`

7.3 Technical Implementation

7.3.1 Firebase Integration

Service	Usage
Firestore	Real-time database for events, merch, clubs
Storage	Handles image uploads (events, merch, clubs)
Authentication	Secures admin access & user interactions

7.3.2 State Management

- React Hooks (`useState`, `useEffect`)
- Real-time Updates (`onSnapshot`)
- Form Handling (Controlled components)

7.3.3 Error Handling

- Try-Catch Blocks for all Firebase operations
- User Feedback (Alerts, progress bars)
- Validation (Prevents `undefined` Firestore writes)

7.3.4 Responsive UI

- Tailwind CSS for styling
- Mobile-friendly tables & forms

7.4 Security

Measure	Implementation
Firestore Rules	Restrict writes to admins
Storage Rules	Allow uploads only for authenticated users
CORS Policy	Configured for `localhost` and production domains

7.5 Performance Optimizations

- Lazy Loading (If implemented)
- Image Compression (Before upload)
- Batched Writes (For bulk operations)

7.6 Future Improvements

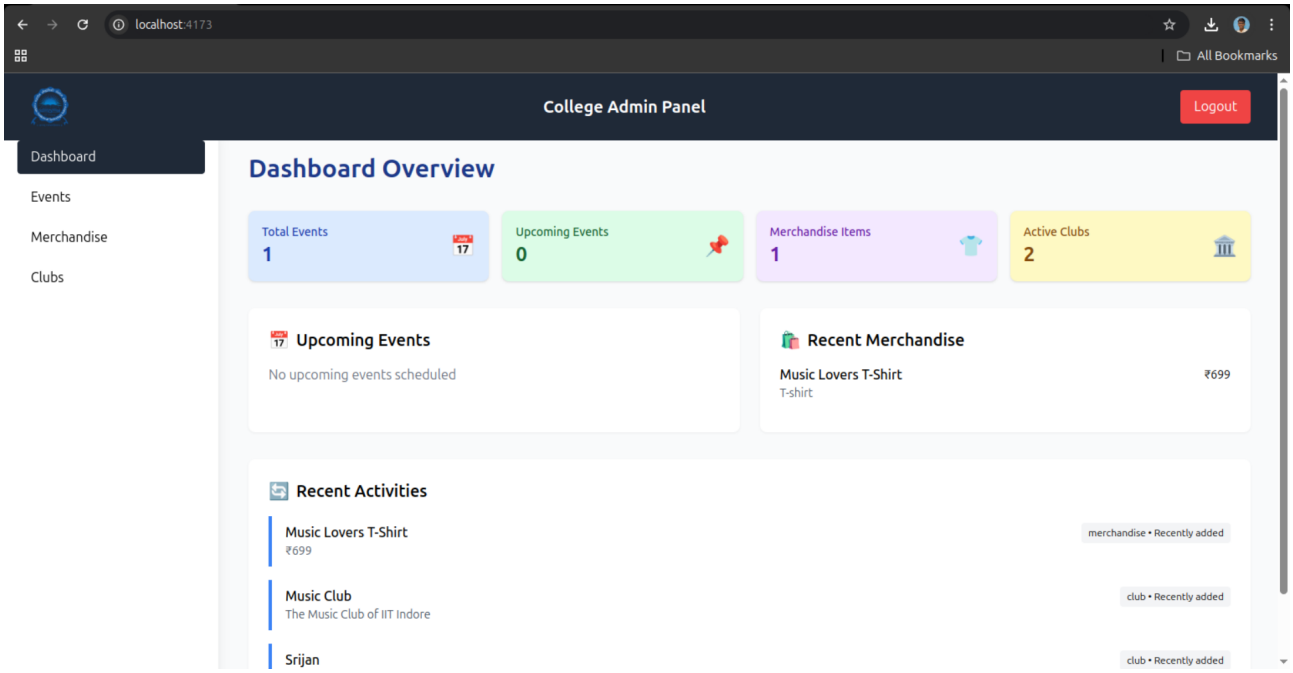
- **Admin Authentication** (Restrict dashboard to admins)
- **Export Data** (CSV/Excel reports)
- **Analytics Dashboard** (Views, subscriptions, sales)
- **Bulk Actions** (Delete/update multiple items)

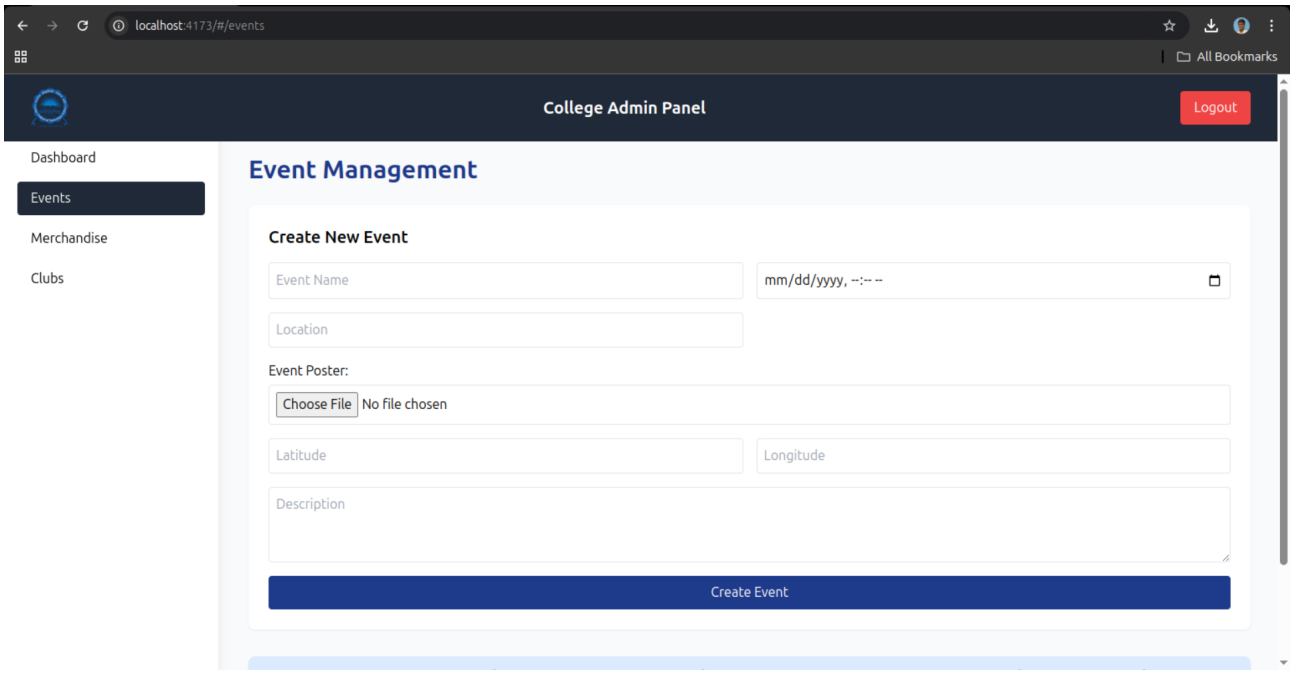
7.7 How to Test

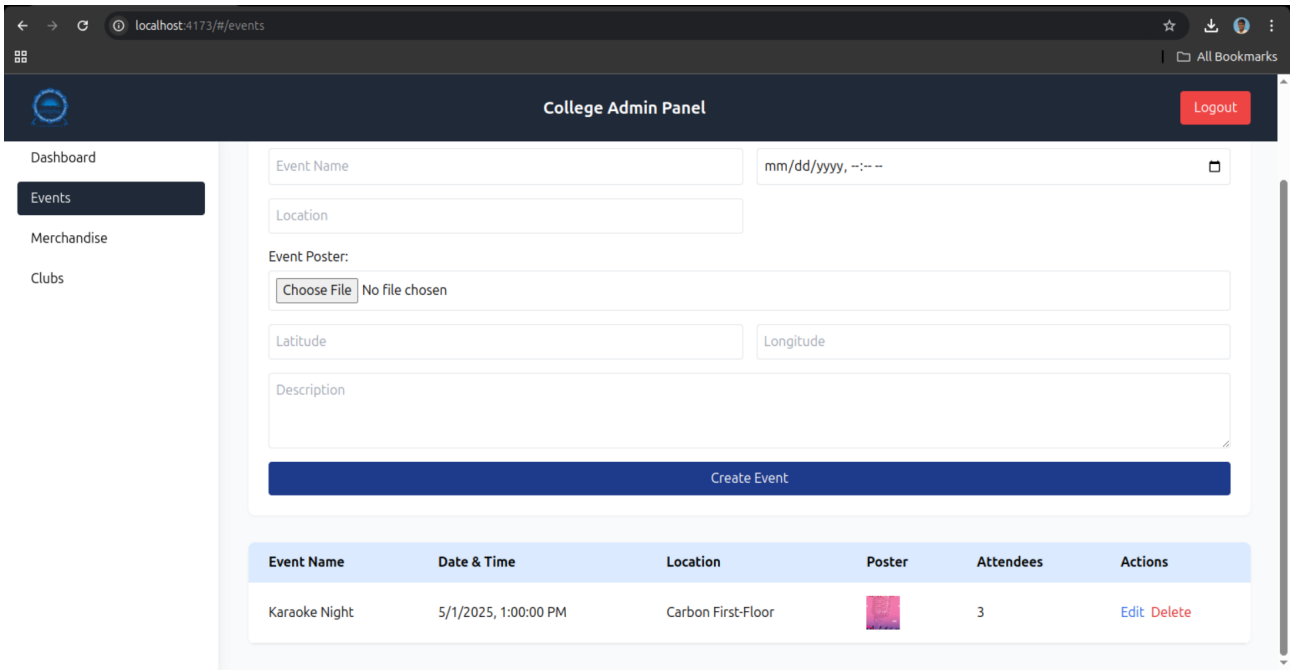
1. **Events** → Create, edit, delete events + test RSVP
2. **Merchandise** → Add products with multiple images
3. **Clubs** → Register clubs + test subscriptions

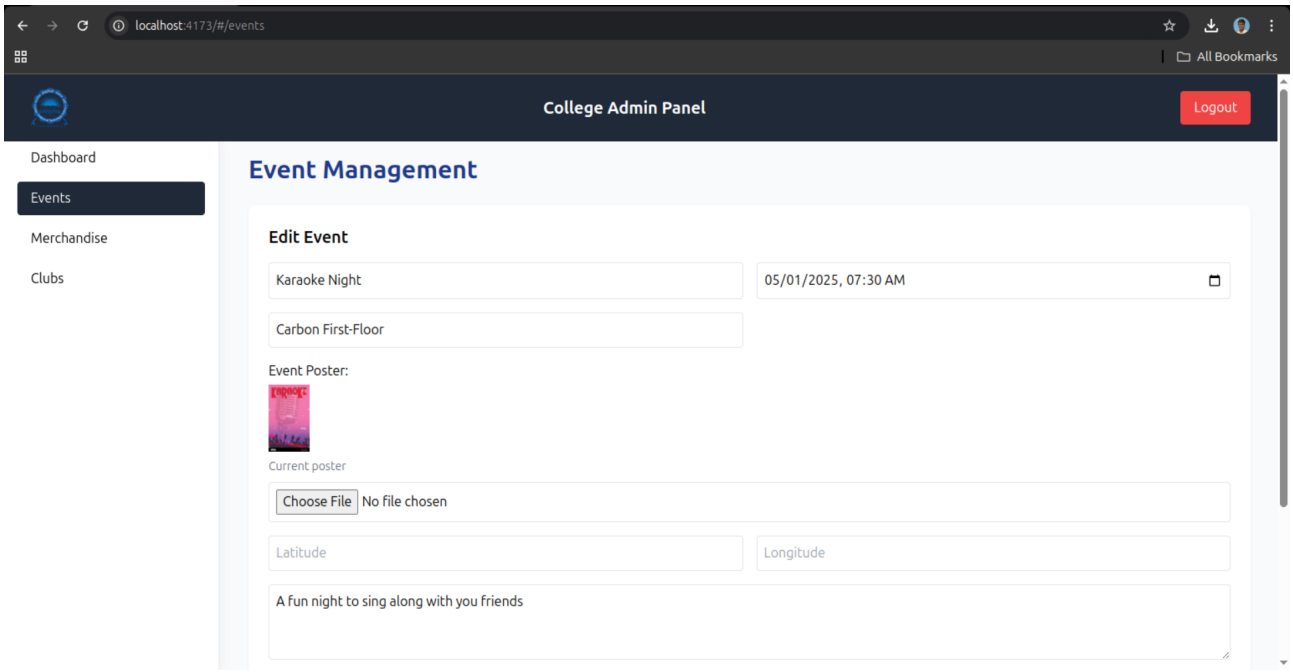
Note: Ensure Firebase CORS is configured (`gsutil cors set`).

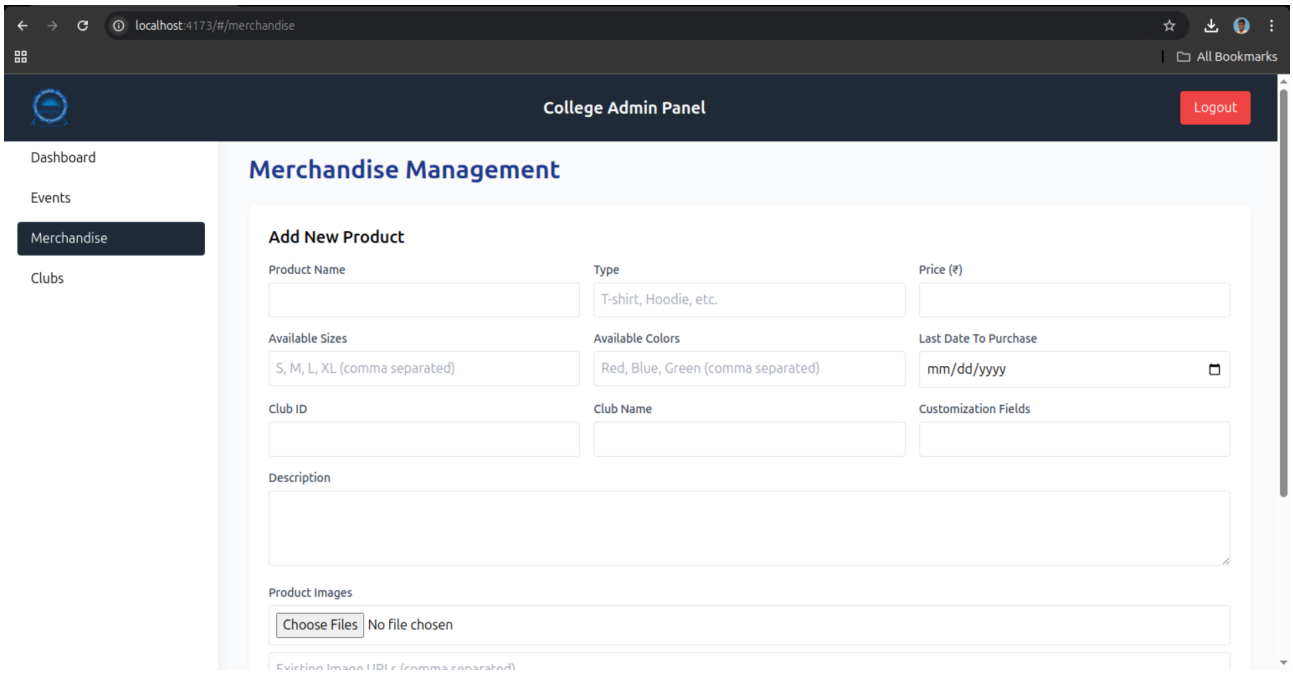
7.8 Screenshots

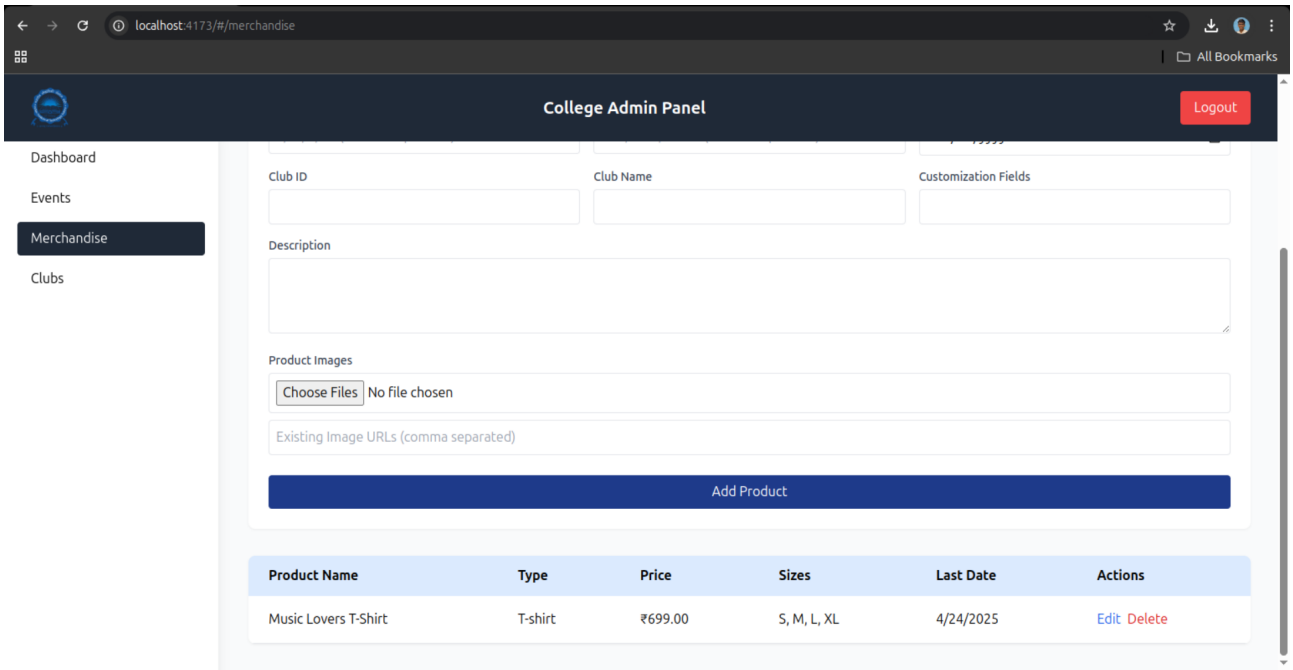


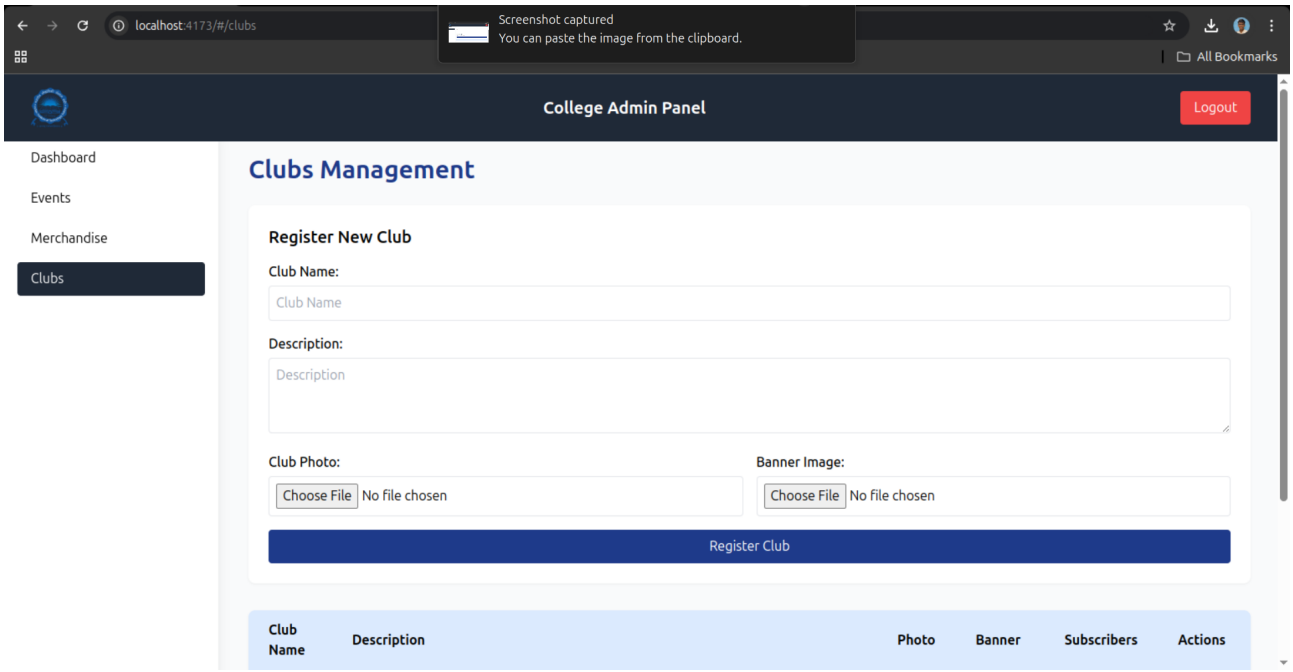


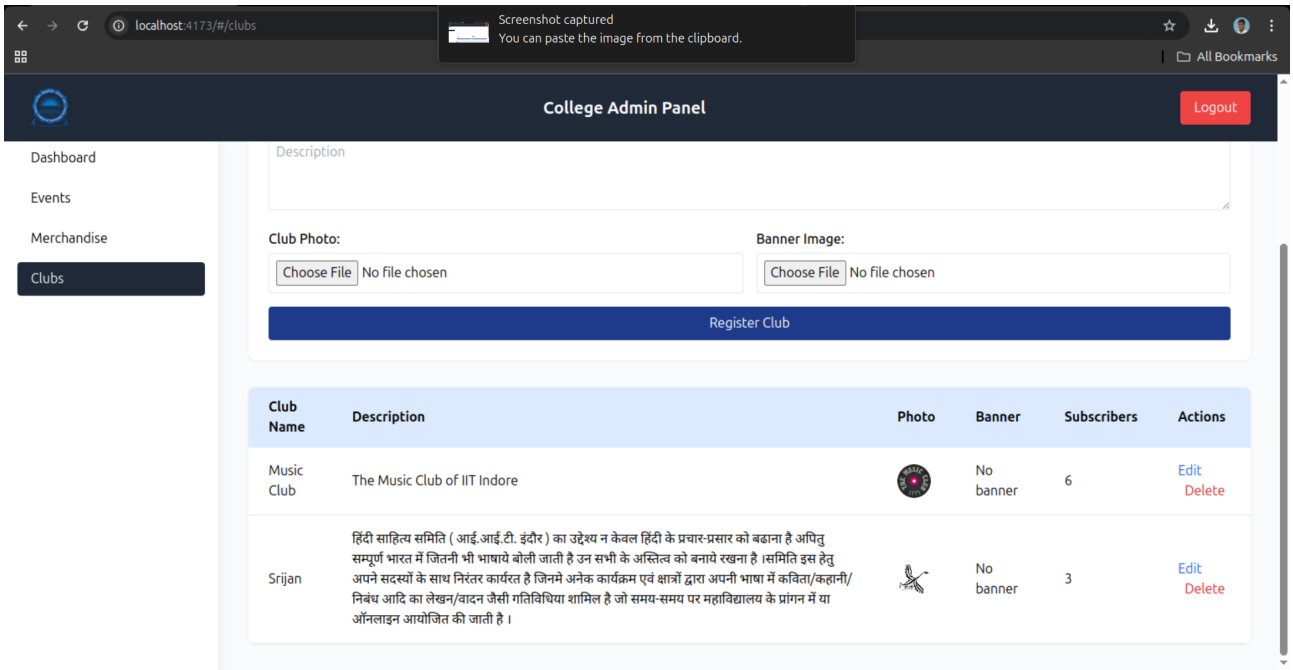

localhost:4173/#/events
All Bookmarks
College Admin Panel
Logout
Dashboard
Events
Merchandise
Clubs
Event Name
mm/dd/yyyy, --:-- --
Location
Event Poster:
Choose File No file chosen
Latitude
Longitude
Description
Create Event
Event Name
Date & Time
Location
Poster
Attendees
Actions
Karaoke Night
5/1/2025, 1:00:00 PM
Carbon First-Floor
3
Edit Delete


localhost:4173/#/events
All Bookmarks
College Admin Panel
Logout
Dashboard
Events
Merchandise
Clubs
Event Management
Edit Event
Karaoke Night
05/01/2025, 07:30 AM
Carbon First-Floor
Event Poster:
Current poster
Choose File No file chosen
Latitude
Longitude
A fun night to sing along with you friends


localhost:4173/#/merchandise
All Bookmarks
College Admin Panel
Logout
Dashboard
Events
Merchandise
Clubs
Merchandise Management
Add New Product
Product Name
Type
T-shirt, Hoodie, etc.
Price (₹)
Available Sizes
S, M, L, XL (comma separated)
Available Colors
Red, Blue, Green (comma separated)
Last Date To Purchase
mm/dd/yyyy
Club ID
Club Name
Customization Fields
Description
Product Images
Choose Files
No file chosen


localhost:4173/#/merchandise
All Bookmarks
College Admin Panel
Logout
Dashboard
Events
Merchandise
Clubs
Club ID
Club Name
Customization Fields
Description
Product Images
Choose Files
No file chosen
Existing Image URLs (comma separated)
Add Product
Product Name
Type
Price
Sizes
Last Date
Actions
Music Lovers T-Shirt
T-shirt
₹699.00
S, M, L, XL
4/24/2025
Edit
Delete


localhost:4173/#/clubs
Screenshot captured
You can paste the image from the clipboard.
All Bookmarks
College Admin Panel
Logout
Dashboard
Events
Merchandise
Clubs
Clubs Management
Register New Club
Club Name:
Club Name
Description:
Description
Club Photo:
Choose File No file chosen
Banner Image:
Choose File No file chosen
Register Club
Club Name
Description
Photo
Banner
Subscribers
Actions


localhost:4173/#/clubs
Screenshot captured
You can paste the image from the clipboard.
All Bookmarks
College Admin Panel
Logout
Dashboard
Events
Merchandise
Clubs
Description
Club Photo:
Choose File No file chosen
Banner Image:
Choose File No file chosen
Register Club
Club Name
Description
Photo
Banner
Subscribers
Actions
Music Club
The Music Club of IIT Indore
No banner
6
Edit
Delete
Srijan
हिंदी साहित्य समिति (आई.आई.टी. इंदौर) का उद्देश्य न केवल हिंदी के प्रचार-प्रसार को बढाना है अपितु सम्पूर्ण भारत में जितनी भी भाषाये बोली जाती है उन सभी के अस्तित्व को बनाये रखना है ।समिति इस हेतु अपने सदस्यों के साथ निरंतर कार्यरत है जिनमे अनेक कार्यक्रम एवं क्षात्रों द्वारा अपनी भाषा में कविता/कहानी/ निबंध आदि का लेखन/वादन जैसी गतिविधिया शामिल है जो समय-समय पर महाविद्यालय के प्रांगन में या ऑनलाइन आयोजित की जाती है ।
No banner
3
Edit
Delete