

# CampGround Site

---

## Team Details

- Khush Gandhi - 612303058
- Hardik Mutha - 612303066
- Suswan Joglekar - 612303074
- Tanmay Karad - 612303083

## Problem Statement

Campers often find it difficult to locate verified and well-documented campgrounds online, as most booking platforms focus mainly on hotels or resorts. This creates a gap for campground owners to digitally list and manage their sites, and for users to book them efficiently.

The project aims to design and implement a **CampGround Booking Database System** that enables campers, campground owners, and administrators to interact within a structured relational database. The system supports campground listings, bookings, amenities, reviews, and approval workflows, demonstrating various entity relationships and database operations.

## Objectives

1. To provide a **unified platform** for discovering, listing, and booking campgrounds.
2. To design a normalized relational database (up to 3NF) for a campground booking platform.
3. To demonstrate entity relationships such as one-to-many (Users → Campgrounds), many-to-many (Campgrounds ↔ Amenities), and one-to-one (Campground ↔ Rules).
4. To implement CRUD operations (Create, Read, Update, Delete) for users, campgrounds, bookings, reviews, and notifications.
5. To support admin-controlled approval for new campground listings.
6. To include additional database concepts like composite keys, foreign keys, constraints, and triggers (if applicable) for maintaining data integrity.

# Functional Requirements

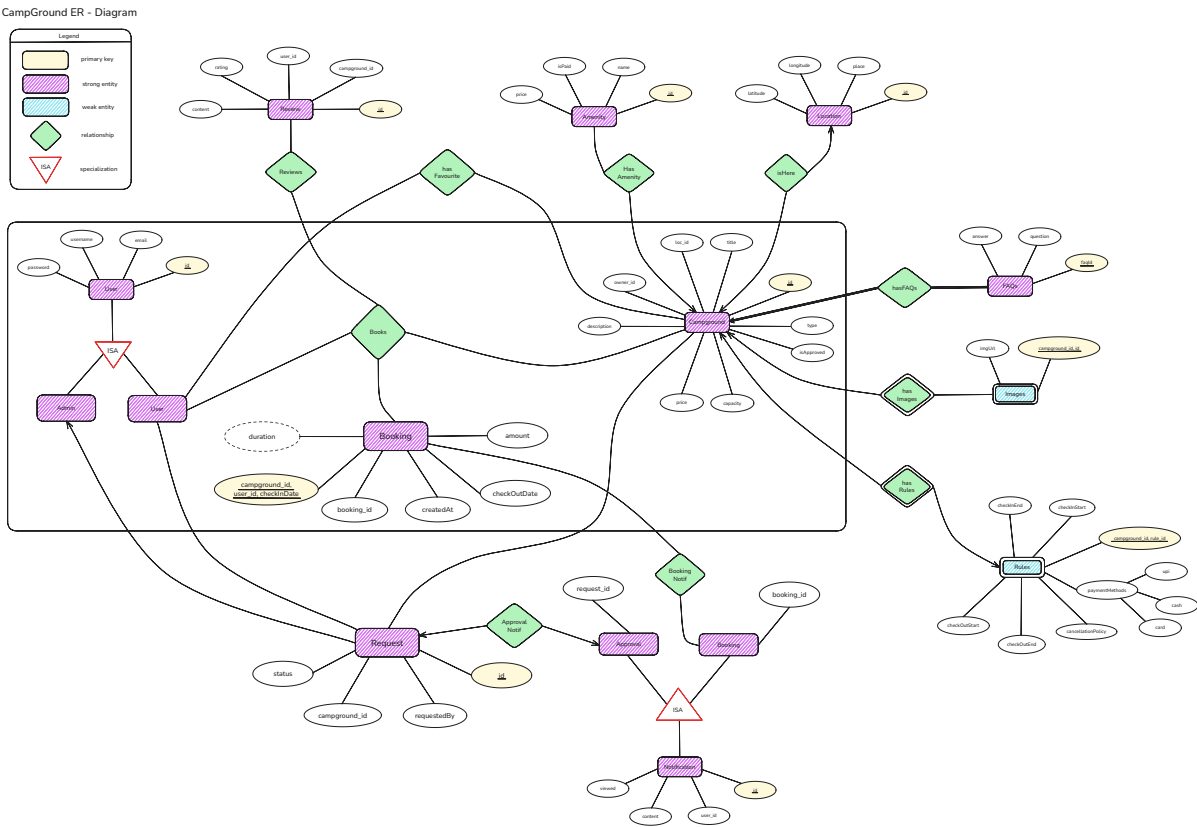
## User Roles:

- **Admin:**
  - Approves or rejects new campground listings (Request).
  - Manages user accounts (Users).
  - Oversees reviews and system notifications (Notifications).
- **Campground Owner:**
  - Creates, updates, or deletes campground listings (Campground).
  - Links amenities to their camps (HasAmenity).
  - Sets pricing, rules, and availability (Rules, Campground).
  - Views and manages booking requests.
- **User (Camper):**
  - Registers and browses approved campgrounds.
  - Books campgrounds and receives notifications (Booking, BookingNotif).
  - Posts reviews and ratings (Review).
  - Marks favorites for quick access (HasFavourite).

## Core Functions:

- **Campground Management:** Owners can create detailed listings with descriptions, price, capacity, and location.
- **Approval System:** Admin verifies and approves campground listings before they appear publicly.
- **Booking Module:** Users can book available campgrounds with date validation and payment options (cash, upi, card).
- **Amenity Mapping:** Campgrounds can offer free or paid amenities linked through an association table.
- **Review & Rating:** Users can rate and review their experience, maintaining a feedback record.
- **Notification System:** Users and owners receive booking and approval alerts.
- **FAQ & Rules:** Common questions and campground-specific policies are stored for reference.

ER Diagram



# Relational Schemas

## 1. Users

- (id PK, username U, role, email, password)

## 2. Location

- (id PK, place, longitude, latitude, UNIQUE(longitude, latitude))

## 3. Campground

- (id PK, title, description, capacity, type, locId FK → Location(id), ownerId FK → Users(id), isApproved, price)

## 4. Request

- (id PK, requestedBy FK → Users(id), campgroundId FK → Campground(id), status)

## 5. Amenity

- (id PK, name U, isPaid, price)

## 6. Review

- (id PK, userId FK → Users(id), campgroundId FK → Campground(id), content, rating)

## 7. Booking

- (userId FK → Users(id), bookingId U, campgroundId FK → Campground(id), checkInDate, checkOutDate, createdAt, amount, PRIMARY KEY(userId,campgroundId,checkInDate))

## 8. HasAmenity

- (amenity\_id FK → Amenity(id), campgroundId FK → Campground(id), PRIMARY KEY(amenity\_id,campgroundId))

## 9. HasFavourite

- (userId FK → Users(id), campgroundId FK → Campground(id), PRIMARY KEY(campgroundId,userId))

## 10. Images

- (id, imgUrl, campgroundId FK → Campground(id), PRIMARY KEY(id,campgroundId))

## 11. Notifications

- (id PK, content, userId FK → Users(id), viewed)

## 12. ApprovalNotif

- (notifId PK FK → Notifications(id), requestId FK → Request(id))

### 13. **BookingNotif**

- (notifId PK FK → Notifications(id), bookingId FK → Booking(bookingId))

### 14. **FAQs**

- (faqId PK, question, answer)

### 15. **Rules**

- (campgroundId PK FK → Campground(id), checkInStart, checkInEnd, checkOutStart, checkOutEnd, cancellationPolicy, cash, upi, card)

# Functional Dependencies

## 1. Users

- $id \rightarrow \{username, role, email, password\}$  (Primary Key)
- $username \rightarrow \{id, role, email, password\}$  (UNIQUE)

## 2. Location

- $id \rightarrow \{place, longitude, latitude\}$  (Primary Key)
- $\{longitude, latitude\} \rightarrow \{id, place\}$  (UNIQUE)

## 3. Campground

- $id \rightarrow \{title, description, capacity, type, locId, ownerId, isApproved, price\}$  (Primary Key)

## 4. Request

- $id \rightarrow \{requestedBy, campgroundId, status\}$

## 5. Amenity

- $id \rightarrow \{name, isPaid, price\}$  (Primary Key)
- $name \rightarrow \{id, isPaid, price\}$  (UNIQUE)

## 6. Review

- $id \rightarrow \{userId, campgroundId, content, rating\}$  (Primary Key)

## 7. Booking

- $\{userId, campgroundId, checkInDate\} \rightarrow \{bookingId, checkOutDate, createdAt, amount\}$  (Primary Key)
- $bookingId \rightarrow \{userId, campgroundId, checkInDate, checkOutDate, createdAt, amount\}$  (UNIQUE)

## 8. Images

- $\{id, campgroundId\} \rightarrow \{imgUrl\}$  (Primary Key)

## 9. Notification

- $id \rightarrow \{content, userId, viewed\}$  (Primary Key)

## 10. FAQs

- $faqId \rightarrow \{question, answer\}$  (Primary Key)

## 11. Rules

- $campgroundId \rightarrow \{checkInStart, checkInEnd, checkOutStart, checkOutEnd, cancellationPolicy, cash, upi, card\}$  (Primary Key)