KLANS — Finalized MongoDB Schema Document (Final Version)

---

1. User Model

```
const UserSchema = new mongoose.Schema({
  phone: { type: String, required: true, unique: true },
  name: { type: String },
  avatar: { type: String },
  referralCode: { type: String, unique: true },
  referredBy: { type: mongoose.Schema.Types.ObjectId, ref: 'User', default:
null },
  onboardingStatus: { type: String, enum: ['invited', 'registered', 'active'],
default: 'invited' },
  klansJoined: [{ type: mongoose.Schema.Types.ObjectId, ref: 'Klan' }],
  klansAdminOf: [{ type: mongoose.Schema.Types.ObjectId, ref: 'Klan' }],
  role: { type: String, enum: ['user', 'admin', 'superadmin', 'moderator'],
default: 'user' },
  inviteCodeUsed: { type: String },
  createdAt: { type: Date, default: Date.now },
  updatedAt: { type: Date }
});
```

Sample Add:

```
db.users.insertOne({
  phone: "+919876543210",
  name: "Sumit Kaila",
  avatar: "https://example.com/avatar.jpg",
  referralCode: "SUMIT123",
  onboardingStatus: "active",
  role: "user",
  createdAt: new Date()
});
```

---

2. OTP Model

```
const OTPSchema = new mongoose.Schema({
  phone: { type: String, required: true },
  otp: { type: String, required: true },
  purpose: { type: String, enum: ['login', 'registration', 'reset'], default:
'login' },
  otpExpiresAt: { type: Date, required: true },
  createdAt: { type: Date, default: Date.now }
});
```

Sample Add:

```
db.otps.insertOne({
  phone: "+919876543210",
  otp: "123456",
  purpose: "registration",
  otpExpiresAt: new Date(Date.now() + 5*60*1000),
  createdAt: new Date()
});
```

---

3. Cluster Model

```
const ClusterSchema = new mongoose.Schema({
  name: { type: String, required: true },
  description: { type: String },
  icon: { type: String },
  createdBy: { type: mongoose.Schema.Types.ObjectId, ref: 'User' },
  categories: [{ type: mongoose.Schema.Types.ObjectId, ref: 'Category' }],
  createdAt: { type: Date, default: Date.now }
});
```

---

4. Category Model

```
const CategorySchema = new mongoose.Schema({
  name: { type: String, required: true },
  description: { type: String },
  icon: { type: String },
  clusterId: { type: mongoose.Schema.Types.ObjectId, ref: 'Cluster' },
```

```
  createdBy: { type: mongoose.Schema.Types.ObjectId, ref: 'User' },
  uxSchema: { type: mongoose.Schema.Types.Mixed },
  isActive: { type: Boolean, default: true },
  createdAt: { type: Date, default: Date.now }
});
```

---

5. KLAN Model

```
const KlanSchema = new mongoose.Schema({
  name: { type: String, required: true },
  description: { type: String },
  location: {
    city: { type: String },
    state: { type: String },
    coordinates: { type: [Number], index: '2dsphere' }
  },
  categoryId: { type: mongoose.Schema.Types.ObjectId, ref: 'Category' },
  clusterId: { type: mongoose.Schema.Types.ObjectId, ref: 'Cluster' },
  createdBy: { type: mongoose.Schema.Types.ObjectId, ref: 'User' },
  members: [{ type: mongoose.Schema.Types.ObjectId, ref: 'User' }],
  admins: [{ type: mongoose.Schema.Types.ObjectId, ref: 'User' }],
  privacy: { type: String, enum: ['public', 'invite-only'], default: 'public' },
  theme: { type: mongoose.Schema.Types.Mixed },
  uxSchemaOverride: { type: mongoose.Schema.Types.Mixed },
  inviteOnlyCode: { type: String },
  createdAt: { type: Date, default: Date.now },
  updatedAt: { type: Date }
});
```

---

6. Post Model

```
const PostSchema = new mongoose.Schema({
  clanId: { type: mongoose.Schema.Types.ObjectId, ref: 'Klan', required: true },
  createdBy: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required:
true },
  type: { type: String, enum: ['text', 'image', 'poll', 'event', 'carpool',
'custom'], default: 'text' },
  content: { type: String },
  media: [{ type: String }],
  meta: { type: mongoose.Schema.Types.Mixed },
  likes: [{ type: mongoose.Schema.Types.ObjectId, ref: 'User' }],
```

```
  commentsCount: { type: Number, default: 0 },
  createdAt: { type: Date, default: Date.now }
});
```

---

7. Comment Model

```
const CommentSchema = new mongoose.Schema({
  postId: { type: mongoose.Schema.Types.ObjectId, ref: 'Post', required: true },
  createdBy: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required:
true },
  text: { type: String, required: true },
  createdAt: { type: Date, default: Date.now }
});
```

---

8. Event Model

```
const EventSchema = new mongoose.Schema({
  clanId: { type: mongoose.Schema.Types.ObjectId, ref: 'Klan', required: true },
  title: { type: String, required: true },
  description: { type: String },
  date: { type: Date, required: true },
  location: { type: String },
  maxParticipants: { type: Number },
  registeredUsers: [{ type: mongoose.Schema.Types.ObjectId, ref: 'User' }],
  createdBy: { type: mongoose.Schema.Types.ObjectId, ref: 'User' },
  createdAt: { type: Date, default: Date.now }
});
```

---

9. Moderation Model

```
const ModerationSchema = new mongoose.Schema({
  targetType: { type: String, enum: ['post', 'comment', 'user'], required:
true },
  targetId: { type: mongoose.Schema.Types.ObjectId, required: true },
  action: { type: String, enum: ['remove', 'edit', 'warn', 'ban'], required:
true },
  reason: { type: String, required: true },
  moderatedBy: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required:
```

```
true },
  status: { type: String, enum: ['pending', 'reviewed', 'resolved'], default:
'pending' },
  createdAt: { type: Date, default: Date.now }
});
```

This document now contains the **final KLANS MongoDB schema** with: - Updated OTP schema with purpose. - Moderator role in User model. - Moderation model for admin dashboard.

This will be your master reference before proceeding to pilot setup.