Yuno System Documentation

1. Project Overview

Yuno is a mobile application for students to engage with their selected schools, share educational content (videos, images, text), view class updates, participate in community Q&A, and manage academic tasks via a planner. It features a TikTok-like vertical feed, restricted to educational content, with a local-first architecture using Expo SQLite for offline functionality and Supabase for cloud syncing and media storage. The app is built with Expo SDK 51, using Expo Router for navigation, and emphasizes Yuno branding (e.g., blue/white color scheme).

2. System Architecture

2.1 Technology Stack

- **Frontend**: React Native with Expo SDK latest, TypeScript, Expo Router.
- **Database**:
    - **Local**: Expo SQLite eg.(post.db) for offline storage of users, posts, schools, updates, and community data. You can create different ones for different things
    - **Cloud**: Supabase Postgres with Row Level Security (RLS) for centralized data and authentication.
- **Media Storage**: Supabase Storage for videos, images, and avatars; local storage via expo-file-system.
- **Video/Audio**: expo-video for video playback and recording.
- **State Management**: Zustand for lightweight state management (e.g., user, feed, planner).
- **Networking**: expo-network for connectivity detection; Supabase for real-time updates.
- **Notifications**: expo-notifications for push notifications.

2.2 Local-First Architecture

- **Offline Mode**: All CRUD operations use SQLite; media stored locally via expo-file-system.
- **Syncing**: Sync queue in SQLite (or AsyncStorage) tracks changes via last_synced timestamps. Syncs with Supabase when online.
- **Conflict Resolution**: Use server precedence or created_at timestamps to resolve conflicts.

2.3 Navigation

- **Structure**: Expo Router with a tab-based root layout (app/(tabs)/), stack navigators for secondary flows (e.g., app/post/), and modals (app/modals/).
- **Tabs**: Home, Discover, Upload, Planner, Profile.
- **Stacks**: Post creation, authentication, community, class feed, settings, profiles, search, messages.

- **Modals**: Comments, share, report.

3. Database Structure

3.1 Local Database: Expo SQLite

- **File**: yuno.db (via expo-sqlite).
- **Purpose**: Store data locally for offline access.
- **Tables**:

1. **Users**
    - **Fields**:
        - id: TEXT (UUID, Primary Key, synced with Supabase Auth)
        - email: TEXT (Unique)
        - username: TEXT (Unique)
        - full_name: TEXT
        - avatar_url: TEXT (Local path or Supabase URL)
        - last_synced: INTEGER (Unix timestamp)
    - **Constraints**: Unique email, username.
2. **Schools**
    - **Fields**:
        - school_id: TEXT (UUID, Primary Key)
        - name: TEXT
        - location: TEXT
        - description: TEXT
        - logo_url: TEXT
        - last_synced: INTEGER
    - **Constraints**: Unique school_id.
3. **User_Schools**
    - **Fields**:
        - id: INTEGER (Primary Key, auto-increment)
        - user_id: TEXT (FK: Users.id)
        - school_id: TEXT (FK: Schools.school_id)
        - joined_at: INTEGER
        - last_synced: INTEGER
    - **Constraints**: Composite unique key (user_id, school_id).
4. **Posts**
    - **Fields**:
        - post_id: TEXT (UUID, Primary Key)
        - user_id: TEXT (FK: Users.id)
        - school_id: TEXT (FK: Schools.school_id)
        - content_type: TEXT (‘video’, ‘image’, ‘text’)
        - content_url: TEXT (Local path or Supabase URL)
        - caption: TEXT
        - tags: TEXT (JSON string, e.g., ["math", "homework"])
        - is_educational: INTEGER (Boolean)
        - created_at: INTEGER

        - last_synced: INTEGER
    - **Constraints**: FKs on user_id, school_id.
5. **Class_Updates**
    - **Fields**:
        - update_id: TEXT (UUID, Primary Key)
        - school_id: TEXT (FK: Schools.school_id)
        - user_id: TEXT (FK: Users.id)
        - title: TEXT
        - content: TEXT
        - created_at: INTEGER
        - last_synced: INTEGER
    - **Constraints**: FKs on school_id, user_id.
6. **Comments**
    - **Fields**:
        - comment_id: TEXT (UUID, Primary Key)
        - post_id: TEXT (FK: Posts.post_id)
        - user_id: TEXT (FK: Users.id)
        - content: TEXT
        - created_at: INTEGER
        - last_synced: INTEGER
    - **Constraints**: FKs on post_id, user_id.
7. **Likes**
    - **Fields**:
        - like_id: TEXT (UUID, Primary Key)
        - post_id: TEXT (FK: Posts.post_id)
        - user_id: TEXT (FK: Users.id)
        - created_at: INTEGER
        - last_synced: INTEGER
    - **Constraints**: FKs on post_id, user_id.
8. **Community_Help**
    - **Fields**:
        - help_id: TEXT (UUID, Primary Key)
        - user_id: TEXT (FK: Users.id)
        - school_id: TEXT (FK: Schools.school_id)
        - group_id: TEXT (FK: Community_Groups.group_id)
        - question: TEXT
        - status: TEXT (‘open’, ‘answered’, ‘closed’)
        - created_at: INTEGER
        - last_synced: INTEGER
    - **Constraints**: FKs on user_id, school_id, group_id.
9. **Community_Answers**
    - **Fields**:
        - answer_id: TEXT (UUID, Primary Key)
        - help_id: TEXT (FK: Community_Help.help_id)
        - user_id: TEXT (FK: Users.id)

        - content: TEXT
        - created_at: INTEGER
        - last_synced: INTEGER
    - **Constraints**: FKs on help_id, user_id.
10. **Community_Groups**
    - **Fields**:
        - group_id: TEXT (UUID, Primary Key)
        - school_id: TEXT (FK: Schools.school_id)
        - name: TEXT
        - description: TEXT
        - member_count: INTEGER
        - created_at: INTEGER
        - last_synced: INTEGER
    - **Constraints**: FK on school_id.

3.2 Cloud Database: Supabase Postgres

- **Purpose**: Centralized storage for authentication, media, and syncing.
- **Tables**: Mirror SQLite structure with UUIDs and RLS.
- **Storage Buckets**:
    - avatars: User profile photos.
    - school_logos: School logos.
    - post_content: Videos and images.
- **RLS Policies**:
    - Users access own data (id = auth.uid()).
    - Posts/updates accessible to school members.
    - Media uploads restricted to authenticated users.

3.3 Local File Storage

- **Directory**: FileSystem.documentDirectory + 'yuno_media/'.
- **Purpose**: Store videos/images locally for offline access.
- **Sync**: Upload to Supabase Storage when online, update content_url.

4. System Workflow

4.1 User Flow

1. **Authentication** (app/auth/):
    - User signs up/logs in (login.tsx, signup.tsx) via Supabase Auth.
    - Onboarding (onboarding.tsx) guides school selection.
    - Data stored in Users and User_Schools (local and cloud).
2. **Home Feed** (app/(tabs)/index.tsx):
    - Displays posts from user's schools (Posts table).
    - Offline: Uses local SQLite data.
    - Online: Syncs with Supabase, prioritizing recent posts.
3. **Content Creation** (app/post/):
    - Record/upload video (create.tsx) using expo-av.
    - Edit caption/tags (edit.tsx), preview (preview.tsx), post.
    - Media saved locally, synced to Supabase Storage.

4. **Community** (app/community/):
    - Join/view groups (index.tsx, [groupId]/index.tsx).
    - Post/view Q&A ([groupId]/thread/[threadId].tsx).
    - Data stored in Community_Groups, Community_Help, Community_Answers.
5. **Planner** (app/(tabs)/planner.tsx):
    - Manage tasks/exams in Planner table (local SQLite, synced).
    - Notifications for due dates.
6. **Class Feed** (app/classfeed/):
    - View school announcements (Class_Updates).
    - Offline access via SQLite.
7. **Profile** (app/(tabs)/profile.tsx, app/profile/[username]/):
    - View/edit user profile, posts, saved content.
    - Access other users' profiles.

4.2 Technical Workflow

- **Initialization**:
    - Initialize Expo project (npx create-expo-app).
    - Set up SQLite (app/api/sqlite/db.ts).
    - Configure Supabase (app/api/supabase/client.ts).
- **Local-First**:
    - CRUD operations use SQLite.
    - Sync queue tracks changes (last_synced).
    - expo-network detects connectivity for syncing.
- **Build/Deploy**:
    - EAS Build for iOS/Android.
    - OTA updates for JavaScript changes.
    - App Store/Google Play submission via EAS Submit.

4.3 Moderation Workflow

- **Client-Side**: Keyword filtering for is_educational flag in Posts.
- **Server-Side**: Supabase functions for manual review of flagged content.
- **Reporting**: Users report posts via app/modals/report.tsx.

5. Folder Structure

The provided structure is adopted with minor adjustments for clarity and alignment with Yuno's requirements:
text

```
yuno/
├── app/
│   ├── _layout.tsx
│   ├── (tabs)/
│   │   ├── _layout.tsx
│   │   ├── index.tsx
│   │   ├── discover.tsx
│   │   ├── upload.tsx
│   │   ├── planner.tsx
│   │   ├── profile.tsx
│   ├── post/
│   │   ├── create.tsx
│   │   ├── edit.tsx
│   │   ├── preview.tsx
│   │   └── [postId]/index.tsx
│   ├── auth/
│   │   ├── login.tsx
│   │   ├── signup.tsx
│   │   ├── forgot-password.tsx
│   │   ├── onboarding.tsx
│   ├── community/
│   │   ├── index.tsx
│   │   ├── [groupId]/index.tsx
│   │   └── [groupId]/thread/[threadId].tsx
│   ├── classfeed/
│   │   └── index.tsx
│   ├── settings/
│   │   ├── index.tsx
│   │   └── notifications.tsx
│   ├── profile/
│   │   └── [username]/index.tsx
│   ├── search/
│   │   └── index.tsx
│   ├── messages/
│   │   ├── index.tsx
│   │   └── [chatId].tsx
│   ├── modals/
│   │   ├── comments.tsx
│   │   ├── share.tsx
│   │   ├── report.tsx
│   ├── api/
│   │   ├── supabase/
│   │   │   ├── client.ts
│   │   │   ├── auth.ts
│   │   │   ├── posts.ts
│   │   │   ├── schools.ts
│   │   │   ├── updates.ts
│   │   │   ├── community.ts
│   │   ├── sqlite/
│   │   │   ├── db.ts
│   │   │   ├── schema.ts
│   │   │   ├── queries/
│   │   │   │   ├─── users.ts
│   │   │   │   ├─── posts.ts
│   │   │   │   ├─── schools.ts
│   │   │   │   ├─── updates.ts
│   │   │   │   ├─── community.ts
│   │   │   │   ├─── planner.ts
│   │   ├─── sync.ts
│   ├─── components/
│   │   ├─── video/
│   │   │   ├─── VideoPlayer.tsx
│   │   │   ├─── MuteButton.tsx
│   │   ├─── feed/
│   │   │   ├─── PostCard.tsx
│   │   │   ├─── UpdateCard.tsx
│   │   ├─── ui/
│   │   │   ├─── Button.tsx
│   │   │   ├─── Input.tsx
│   │   │   ├─── Avatar.tsx
│   │   ├─── layout/
│   │   │   ├─── TabBar.tsx
│   │   │   ├─── Header.tsx
│   │   ├─── community/
│   │   │   ├─── QuestionCard.tsx
│   │   │   ├─── AnswerCard.tsx
│   ├─── hooks/
│   │   ├─── useAuth.ts
│   │   ├─── useUpload.ts
│   │   ├─── useFeed.ts
│   │   ├─── useCommunity.ts
│   │   ├─── usePlanner.ts
│   ├─── context/
│   │   ├─── AuthContext.tsx
│   │   ├─── ThemeContext.tsx
│   ├─── store/
│   │   ├─── index.ts
│   │   ├─── slices/
│   │   │   ├─── user.ts
│   │   │   ├─── feed.ts
│   │   │   ├─── planner.ts
│   ├─── constants/
│   │   ├─── colors.ts
│   ├─── utils/
│   │   ├─── media.ts
│   │   ├─── validation.ts
│   │   ├─── format.ts
│   │   ├─── date.ts
│   ├─── types/
│  │   ├─── user.d.ts
│  │   ├─── post.d.ts
│  │   ├─── comment.d.ts
│  │   ├─── planner.d.ts
│  │   ├─── community.d.ts
│   ├─── assets/
│  │   ├─── fonts/
│  │   ├─── icons/
│  │   ├─── images/
│  │  │   ├─── yuno_logo.png
 ├─── app.json
 ├─── tsconfig.json
 ├─── babel.config.js
 ├─── package.json
 ├─── .eslintrc.js
 ├─── .prettierrc
 ├─── README.md
```

7. Screen Implementations

Below are detailed implementations for each screen, including UI elements, behavior, and dependencies, as specified.

7.1 Main Tab Screens (app/(tabs)/)

1. **Home Feed (index.tsx)**
   - **UI Elements**:
     - Fullscreen VideoPlayer.tsx (autoplay, expo-video , expo-audio).
     - Text: Username, caption, hashtags.
     - School badge (from Schools).
     - Buttons: Like, Comment, Share, Bookmark (components/ui/).
     - Audio info (sound name, optional).
     - Follow button.
     - Scroll indicator (via react-native-gesture-handler).
     - Custom TabBar.tsx.
   - **Behavior**:
     - Swipe up/down for next/prev video (react-native-gesture-handler).
     - Pull to refresh (FlatList).
     - Double tap to like (store in Likes).
     - Long press for report/save (app/modals/).
   - **Dependencies**: expo-video, react-native-gesture-handler, expo-sqlite, supabase/supabase-js.
2. **Discover (discover.tsx)**
   - **UI Elements**:
     - Search bar (Input.tsx).

      - Trending topics list.
      - Popular schools/departments (Schools table).
      - Horizontal scrolls for hashtags, subjects, creators (FlatList).
      - Vertical grid of video thumbnails (expo-av).
   - **Behavior**:
      - Filter feed by hashtag/school.
      - Auto-rotate trending topics.
      - Save recent searches (AsyncStorage).
   - **Dependencies**: expo-sqlite, supabase/supabase-js, @rneui/themed.
3. **Upload (upload.tsx)**
   - **UI Elements**:
      - Buttons: Record Video, Upload from Gallery (Button.tsx).
      - Optional: Create from slides/images.
      - Cancel button.
   - **Behavior**:
      - Navigates to app/post/create.tsx.
      - Requests camera/media permissions (expo-image-picker).
      - Compresses media (expo-av).
   - **Dependencies**: expo-av, expo-image-picker, expo-file-system.
4. **Planner (planner.tsx)**
   - **UI Elements**:
      - Timetable (cards or calendar, FlatList or third-party calendar).
      - Add task/exam/assignment button.
      - Subject badges.
      - Notification indicators (expo-notifications).
      - Toggle: List/Calendar view.
   - **Behavior**:
      - Swipe between days.
      - Tap to edit tasks.
      - Long press to reschedule.
      - Store in Planner table.
   - **Dependencies**: expo-sqlite, expo-notifications.
5. **Profile (profile.tsx)**
   - **UI Elements**:
      - Profile picture, name, school, bio (Users, User_Schools).
      - Follow/Followers count.
      - Edit Profile button.
      - Tabs: Posts, Saved, Class Feed (FlatList).
      - Settings icon (navigates to app/settings/).
   - **Behavior**:
      - Infinite scroll for posts.
      - Tappable followers/following.
      - Share profile via app/modals/share.tsx.
   - **Dependencies**: expo-sqlite, supabase/supabase-js, @rneui/

themed.

7.2 Secondary & Stack Screens

1. **Post Creation (app/post/create.tsx)**
   - **UI Elements**:
     - Camera recorder (expo-av).
     - Upload video button (expo-image-picker).
     - Sound selector (optional).
     - Next button (to edit.tsx).
   - **Behavior**:
     - Record or upload video, save locally (expo-file-system).
     - Store in Posts table.
   - **Dependencies**: expo-av, expo-image-picker, expo-file-system.
2. **Post Edit (app/post/edit.tsx)**
   - **UI Elements**:
     - Caption input (Input.tsx).
     - School/class picker (Schools).
     - Hashtag input.
     - Visibility toggle (public/private/school-only).
     - Post button.
   - **Behavior**:
     - Save edits to Posts table.
     - Navigate to preview.tsx.
   - **Dependencies**: expo-sqlite, @rneui/themed.
3. **Post Preview (app/post/preview.tsx)**
   - **UI Elements**:
     - Video preview (expo-av).
     - Change music/trim options.
     - Post settings summary.
     - Publish button.
   - **Behavior**:
     - Preview video, confirm settings.
     - Sync to Supabase Storage/Posts.
   - **Dependencies**: expo-av, supabase/supabase-js.
4. **Single Post (app/post/[postId]/index.tsx)**
   - **UI Elements**:
     - Full video playback (VideoPlayer.tsx).
     - Like, comment, share buttons.
     - Comment input/thread (Comments table).
     - Report button (app/modals/report.tsx).
     - Related content (FlatList).
   - **Behavior**:
     - Load post/comments from SQLite or Supabase.
     - Real-time comment updates (supabase.channel).
   - **Dependencies**: expo-av, expo-sqlite, supabase/supabase-js.

5. **Community List (app/community/index.tsx)**
    - **UI Elements**:
        - Search bar.
        - Join/Create group button.
        - Group list (Community_Groups, by topic/course).
        - Member count, active threads.
    - **Behavior**:
        - Filter groups by topic.
        - Navigate to group page.
    - **Dependencies**: expo-sqlite, supabase/supabase-js.
6. **Community Group (app/community/[groupId]/index.tsx)**
    - **UI Elements**:
        - Group name, info (Community_Groups).
        - Thread list (Community_Help).
        - Post new thread button.
        - Member avatars.
    - **Behavior**:
        - Load group threads.
        - Navigate to thread details.
    - **Dependencies**: expo-sqlite, supabase/supabase-js.
7. **Community Thread (app/community/[groupId]/thread/[threadId].tsx)**
    - **UI Elements**:
        - Question details (Community_Help).
        - Answer list (Community_Answers).
        - Answer input.
    - **Behavior**:
        - Post/view answers.
        - Real-time updates.
    - **Dependencies**: expo-sqlite, supabase/supabase-js.
8. **Class Feed (app/classfeed/index.tsx)**
    - **UI Elements**:
        - Filter by department/year.
        - Announcement cards (Class_Updates).
        - Link to poster profile.
        - Bookmark/save option.
    - **Behavior**:
        - Load updates from SQLite/Supabase.
        - Save bookmarks locally.
    - **Dependencies**: expo-sqlite, supabase/supabase-js.
9. **Auth Screens (app/auth/)**
    - **UI Elements**:
        - Email, password inputs (login.tsx, signup.tsx).
        - Google OAuth button.
        - Forgot password link (forgot-password.tsx).

        - School selection (onboarding.tsx).
    - **Behavior**:
        - Authenticate via Supabase Auth.
        - Store user data in Users, User_Schools.
    - **Dependencies**: supabase/supabase-js, @rneui/themed.
10. **Settings (app/settings/)**
    - **UI Elements** (index.tsx):
        - Profile settings, notifications toggle, privacy, about, logout.
    - **UI Elements** (notifications.tsx):
        - Toggles for likes, comments, follows, class updates.
    - **Behavior**:
        - Update user settings in Users.
        - Configure notifications (expo-notifications).
    - **Dependencies**: expo-sqlite, supabase/supabase-js, expo-notifications.
11. **Other Profiles (app/profile/[username]/index.tsx)**
    - **UI Elements**:
        - Profile picture, name, school, bio.
        - Follow button.
        - Post grid.
    - **Behavior**:
        - Load user data from SQLite/Supabase.
        - Infinite scroll for posts.
    - **Dependencies**: expo-sqlite, supabase/supabase-js.
12. **Search (app/search/index.tsx)**
    - **UI Elements**:
        - Search bar.
        - Results for users, posts, groups.
    - **Behavior**:
        - Filter results from SQLite/Supabase.
        - Save search history (AsyncStorage).
    - **Dependencies**: expo-sqlite, supabase/supabase-js.
13. **Messages (app/messages/)**
    - **UI Elements**:
        - Chat list (index.tsx).
        - Chat conversation ([chatId].tsx).
    - **Behavior**:
        - Real-time messaging via Supabase.
        - Store messages locally (new Messages table).
    - **Dependencies**: supabase/supabase-js, expo-sqlite.
14. **Modals (app/modals/)**
    - **Comments (comments.tsx)**: Comment input, thread display.
    - **Share (share.tsx)**: Share post via link.
    - **Report (report.tsx)**: Report post/user form.
    - **Dependencies**: expo-sqlite, supabase/supabase-js.