Social Media App - React Native + Node.js

A full-featured social media application built with React Native (Expo Router) and Node.js backend with SQLite database.

Features

- **Authentication**: Email-based signup and login with JWT tokens
- **Feed**: View posts from users you follow
- **Explore**: Discover new content from users you don't follow
- **User Profiles**: View your profile, stats, and manage your posts
- **Social Features**:
  - Create, edit, and delete posts
  - Like and unlike posts
  - Comment on posts
  - Follow and unfollow users
  - Search for users
- **Real-time Updates**: Pull-to-refresh on all screens
- **Persistent Storage**: SQLite database for all data
- **Secure**: Password hashing with bcrypt, JWT authentication

Project Structure

```
social-media-app/          # React Native Frontend
├── app/
│   ├── (auth)/           # Authentication screens
│   ├── (tabs)/           # Main app tabs
│   ├── _layout.tsx       # Root layout
│   └── index.tsx         # Entry point
├── components/           # Reusable components
├── contexts/             # Context providers
├── config/               # Configuration files
└── types/                # TypeScript definitions

social-media-backend/     # Node.js Backend
├── server.js             # Express server
├── database.js           # SQLite database module
└── social_media.db       # SQLite database file (created automatically)
```

Prerequisites

- Node.js (v14 or higher)
- npm or yarn
- Expo CLI (`npm install -g expo-cli`)
- iOS Simulator (Mac) or Android Emulator

Setup Instructions

1. Backend Setup

bash

```
# Navigate to backend directory
cd social-media-backend

# Install dependencies
npm install

# Start the server
npm run dev
```

The backend will run on `http://localhost:3000`

2. Frontend Setup

bash

```
# Navigate to frontend directory
cd social-media-app

# Install dependencies
npm install

# Install iOS pods (Mac only, if using bare workflow)
# cd ios && pod install && cd ..
```

3. Configure API URL

Important: Update the API URL in `/social-media-app/config/api.ts`:

1. Find your machine's local IP address:
   - Mac: Run `ifconfig | grep inet`
   - Windows: Run `ipconfig`
   - Linux: Run `hostname -I`
2. Update the API_URL:

typescript

```
// Replace with your machine's IP address
export const API_URL = 'http://192.168.1.100:3000/api';
```

4. Run the App

bash

```
# In the social-media-app directory
npx expo start

# Then press:
# - 'i' for iOS Simulator
# - 'a' for Android Emulator
# - Scan QR code with Expo Go app on physical device
```

Testing the App

Test Accounts

Create test accounts through the signup screen or use these credentials after creating them:

1. **User 1**:
   - Email: user1@example.com
   - Password: password123
   - Username: user1
2. **User 2**:
   - Email: user2@example.com
   - Password: password123
   - Username: user2

Testing Flow

1. **Sign Up**: Create a new account with email and password
2. **Login**: Use your credentials to login
3. **Create Posts**: Go to Profile tab and tap "New Post"
4. **Explore**: Search and follow other users in the Explore tab
5. **Interact**: Like and comment on posts in your feed
6. **Profile**: View your stats and manage your posts

API Endpoints

Authentication

- `POST /api/auth/signup` - Register new user
- `POST /api/auth/login` - Login user

Posts

- `GET /api/posts/feed` - Get feed posts (from followed users)
- `GET /api/posts/explore` - Get explore posts (random posts)
- `POST /api/posts` - Create new post
- `DELETE /api/posts/:postId` - Delete post
- `POST /api/posts/:postId/like` - Like/unlike post
- `POST /api/posts/:postId/comment` - Add comment

Users

- `GET /api/users/:userId/posts` - Get user's posts
- `GET /api/users/:userId/stats` - Get user statistics
- `GET /api/users/search?q=query` - Search users
- `POST /api/users/:userId/follow` - Follow/unfollow user

Common Issues & Solutions

Issue: Cannot connect to backend

Solution: Make sure you've updated the API_URL in `config/api.ts` with your machine's IP address, not localhost.

Issue: Network request failed

Solution:

1. Ensure backend is running (`npm run dev` in backend directory)
2. Check firewall settings allow connections on port 3000
3. If using physical device, ensure it's on the same network

Issue: Expo command not found

Solution: Install Expo CLI globally:

bash

```
npm install -g expo-cli
```

Issue: Metro bundler issues

Solution: Clear cache and restart:

bash

```
npx expo start -c
```

Security Notes

For production deployment:

1. **Change JWT Secret**: Update `JWT_SECRET` in `server.js`
2. **Use HTTPS**: Deploy with SSL certificates
3. **Add Rate Limiting**: Implement rate limiting for API endpoints
4. **Validate Input**: Add more robust input validation
5. **Use Environment Variables**: Store sensitive data in `.env` files
6. **Database Security**: Use proper database authentication

Database Schema

Users Table

- `id` (TEXT, PRIMARY KEY)
- `email` (TEXT, UNIQUE)
- `username` (TEXT, UNIQUE)
- `password` (TEXT, hashed)
- `created_at` (DATETIME)

Posts Table

- `id` (TEXT, PRIMARY KEY)
- `user_id` (TEXT, FOREIGN KEY)
- `content` (TEXT)
- `created_at` (DATETIME)
- `updated_at` (DATETIME)

Likes Table

- `post_id` (TEXT)
- `user_id` (TEXT)
- `created_at` (DATETIME)

Comments Table

- `id` (TEXT, PRIMARY KEY)
- `post_id` (TEXT, FOREIGN KEY)
- `user_id` (TEXT, FOREIGN KEY)
- `content` (TEXT)
- `created_at` (DATETIME)

Follows Table

- `follower_id` (TEXT)
- `following_id` (TEXT)
- `created_at` (DATETIME)

Customization

Styling

- Colors and themes: Edit styles in component files
- App theme: Modify `app.json` for splash screen and icons

Features to Add

- Push notifications
- Image/video uploads
- Direct messaging
- Stories feature
- User profile pictures
- Edit profile functionality
- Password reset
- Email verification

Deployment

Backend Deployment Options

1. **Heroku**:


bash

```
heroku create your-app-name
git push heroku main
```

2. **Railway**:

- Connect GitHub repo at railway.app
- Add environment variables
- Deploy

3. **VPS (DigitalOcean, AWS, etc.)**:

- Set up Node.js environment
- Use PM2 for process management
- Configure Nginx as reverse proxy

Frontend Deployment

1. **Expo EAS Build**:


bash

```
npm install -g eas-cli
eas build --platform all
```

2. **App Stores**:

- Follow Expo's guide for App Store and Google Play submission

Performance Optimization

1. **Backend**:
    - Implement caching (Redis)
    - Add database indexes
    - Use connection pooling
    - Implement pagination for large datasets
2. **Frontend**:
    - Implement lazy loading
    - Add image caching
    - Use React.memo for components
    - Implement virtual lists for large feeds

Contributing

Feel free to fork and submit pull requests. Areas for improvement:

- Add unit tests
- Implement end-to-end testing
- Add more robust error handling
- Improve accessibility
- Add internationalization (i18n)

License

MIT License - feel free to use this project for learning or commercial purposes.

Support

For issues or questions, please create an issue in the repository or contact the maintainer.

Note: This is a learning project demonstrating full-stack mobile development with React Native and Node.js. For production use, additional security measures and optimizations should be implemented.