Video Platform Integration Summary

Key Issues Fixed

1. Service Layer Fixes (`video.service.js`)

- **Method Naming**: Fixed `togglePublishStatus` → `toggleVideoPublishStatus` to match component usage
- **Data Structure Handling**: Properly handles both old and new API response formats
- **Error Handling**: Enhanced error handling with meaningful messages
- **Response Parsing**: Correctly extracts video data from various response structures
- **Token Management**: Improved JWT token refresh logic

2. Component Integration Fixes

Home Component (`Home.jsx`)

- **Data Flow**: Properly extracts videos from service response
- **Pagination**: Added load more functionality
- **Error States**: Comprehensive error handling and empty states
- **Loading States**: Better loading indicators

ManageVideos Component (`ManageVideos.jsx`)

- **Authentication**: Added proper auth checks and redirects
- **Video Operations**: Fixed delete and publish toggle functionality
- **State Management**: Proper state updates after operations
- **User Feedback**: Better loading states and success/error messages

EditVideo Component (`EditVideo.jsx`)

- **Data Loading**: Proper video data fetching and form population
- **File Handling**: Improved thumbnail upload with validation
- **State Sync**: Proper synchronization between form and API
- **Error Handling**: Comprehensive error handling for all operations

Watch Component (`Watch.jsx`)

- **Media URLs**: Helper function to handle different URL formats
- **User Interactions**: Proper like/subscribe functionality
- **Comments**: Complete comment system with posting and display
- **Loading States**: Better loading and error states

Search Component (`Search.jsx`)

- **Query Handling**: Proper URL parameter extraction
- **Results Display**: Improved results formatting and empty states
- **Error Handling**: Graceful error handling for failed searches

History Component (`History.jsx`)

- **Data Fetching**: Proper watch history retrieval
- **Authentication**: Auth checks and redirects
- **Empty States**: Better empty state handling

Backend Controller Alignment (`video.controller.js`)

The controller properly handles:

- **Pagination**: Uses aggregation pipeline with pagination
- **Search**: Regex-based search in title and description
- **File Uploads**: Cloudinary integration for video and thumbnail uploads
- **Ownership Validation**: Proper user ownership checks for operations
- **Published Status**: Only shows published videos in public listings

Integration Flow

```
User Action → Component → Service → API → Controller → Database
    ↑                                  ↓
User Feedback ← State Update ← Response ← JSON Response ← Query Result
```

Key Integration Points:

1. **Authentication Flow**
   - JWT tokens in localStorage
   - Automatic token refresh
   - Proper auth checks in components
2. **Data Flow**
   - Consistent data structure handling
   - Proper error propagation
   - Loading state management
3. **File Upload Flow**
   - FormData creation in components

    - Cloudinary upload in backend
    - Progress tracking in service

4. **State Management**
    - Local component state for UI
    - Service layer for API calls
    - Proper state updates after operations

Testing Checklist

□ Videos display properly on home page

□ Upload functionality works with progress tracking

□ User can manage their videos (edit, delete, publish/unpublish)

□ Video watch page displays correctly with all features

□ Search functionality returns relevant results

□ Watch history is properly maintained

□ Authentication redirects work correctly

□ Error handling provides meaningful feedback

□ Loading states display appropriately

Environment Setup

Ensure these environment variables are set:

```env
VITE_APP_API_URI=http://localhost:8000/api/v1
```

Backend should have:

```env
CLOUDINARY_CLOUD_NAME=your_cloud_name
CLOUDINARY_API_KEY=your_api_key
CLOUDINARY_API_SECRET=your_api_secret
```

Usage Flow

1. **Home Page**: Shows all published videos
2. **Upload**: Users can upload videos with thumbnails
3. **Manage Videos**: Users see their own videos (published/unpublished)
4. **Edit Video**: Users can edit title, description, thumbnail, publish status
5. **Watch Page**: Full video player with comments, likes, subscription

6. **Search**: Find videos by title/description
7. **History**: View previously watched videos

The integration ensures a smooth user experience with proper error handling, loading states, and data consistency across all components.