FPT Education
FPT UNIVERSITY

Assignment 2: MusicTracker App

1. PROJECT OVERVIEW

MusicTracker is a mobile application that helps users manage and track their favorite music albums. Users can add new albums, view a list of stored albums, edit album details, delete albums, and mark albums as favorites. The application uses AsyncStorage to store data locally. It is designed to run on both Android and iOS platforms.

The data for each album includes:

1. **id** (string): A unique identifier for the album (e.g., 1, 2, 3).
2. **name** (string): The name of the album (e.g., "Thriller", "Back in Black").
3. **artist** (string): The artist of the album (e.g., "Michael Jackson", "AC/DC").
4. **releaseDate** (string): The release date of the album in YYYY-MM-DD format (e.g., "1982-11-30").
5. **genre** (string): The genre of the album (e.g., "Pop", "Rock").
6. **trackCount** (number): The number of tracks in the album (e.g., 9, 10).
7. **price** (number): The price of the album (e.g., 15.99, 20.00).
8. **rating** (number): The rating of the album (e.g., 4.5, 5.0).
9. **favorite** (boolean): Whether the album is marked as a favorite (e.g., true/false).

2. MAIN FEATURES

General Requirements:

- **For this practice exam, a 'Resource' folder with image files and a sample data file is provided. Image files are mandatory for your app development.**
- **Project folder** must be named: MMA301FA24_StudentCode.
- **Development Tool**: Use Visual Studio Code as your development IDE.
- **Technology**: Use NPM and EXPO to build your application.
- **Data**: Album data must be stored and managed using AsyncStorage.
- **Navigation:** Use Bottom Tabs to navigate between screens: Home, Add Album, Favorites.
- **UI Enhancement:** Utilize supporting libraries such as icons, styles, and optimization tools to make the app's interface attractive and engaging.

3. SUMMARY OF IMPLEMENTATION STEPS

App with 4 bottom navigation tabs:

🏠 **Home** – View album list, search, edit, delete, mark as favorite
➕ **Add Album** – Add a new album
❤️ **Favorites** – View favorite albums
💰 **Stats** – View total expense for favorite albums

Task 1: Create Album Management Screen (Home Screen)

Create the Home Screen to display the list of albums, including the following fields: *(2)Album name, (3)Artist, (4)Release date, (5)Genre, (7)Price, (8)Rating.*

Requirements:

- The album list must be stored in AsyncStorage and displayed using FlatList .

- Add a Bottom Tab Navigator with icons for Home, Add, and Favorite, navigating to the respective screens when clicked.
- Each album must have edit and delete buttons. Prompt the user to confirm before deleting an album.
- When the application is run for the first time, the data will be empty. Data can be added by clicking the Add button.
- Display an image banner on the Home Screen

Task 2: Add New Albums (Add Album Screen)

Create the Add Album Screen where users can add new Albums, including

User Input Fields

Allow the user to input the following album data:

1. **Album Name** (`TextInput`)
2. **Artist** (`TextInput`)
3. **Release Date** (`TextInput` or Date Picker)
4. **Genre** (`TextInput` or Dropdown)
5. **Track Count** (`Number Input`)
6. **Price** (`Number Input`)
7. **Rating** (`Slider` or Number Input`)

System-Handled Fields (Do Not Ask User to Enter)

- **`id`**: Automatically assigned using **auto-increment logic**
  → e.g., based on the last album's ID in AsyncStorage
- **`favorite`**: Set to `false` by default when creating a new album

```
const newAlbum = {
  id: lastId + 1,
  name,
  artist,
  releaseDate,
  genre,
  trackCount,
  price,
  rating,
  favorite: false, // default value
};
```

Requirements

- After the user presses the **"Save"** button:
  - Validate all fields (not empty, numbers valid)
  - Create a new album object with auto-incremented ID and `favorite: false`
  - Save the new album to **AsyncStorage**
  - Immediately reflect the update on the **Home Screen**
- Pressing the **"Back"** button should return to the **Home Screen**

Benefits of This Clarification

- ✅ Simplifies the user experience (no need to set ID or favorite manually)
- ✅ Prevents incorrect input for system-generated fields
- ✅ Matches standard design principles for controlled app data flow

Task 3: Show Detail Information

Create the Album Detail Screen to display album information, including: *(2)Album name, (3)Artist, (4)Release date, (5)Genre,* (*6)Track Count, (7)Price, (8)Rating.*

Requirements:

- When the user taps on the name or Artist of an Album on the Home Screen, navigate to the Album Detail Screen to display and allow editing of the album details, and save the changes.
.
- Press the Back button to return to the Home Screen.

Task 4: Edit Albums (Update Album Screen)

Create the Update Album Screen to display album information, including: *(2)Album name, (3)Artist, (4)Release date, (5)Genre,* (*6)Track Count, (7)Price, (8)Rating.*

Requirements:

- When the user taps the edit button or icon of an album on the Home Screen, navigate to the Update Album Screen to display and allow editing of the album details, and save the changes.
.
- Press the Back button to return to the Home Screen.

Task 5: Display Favorite Albums (Favorites Screen)

Create the Favorites screen to show a list of favorite albums. The list should include *(2)Album name, (3)Artist, (4)Release date, (5)Genre*

Requirements:

- Add a favorite feature for albums with a heart icon on each album on the Home Screen that updates the favorite status when clicked, allowing users to mark or unmark albums as favorites.
- Display the favorite albums stored in AsyncStorage.

Task 6: Search Functionality (in Home Tab)

Add a search feature to the Home Screen that allows users to search for albums by name.

Task 7: Stats Tab – Favorite Album Spending Summary

- Display the **total number** of favorite albums
- Calculate and display the **total money spent** on favorite albums (sum of price)
- Display the **most expensive favorite album**, including:

- Album name
- Artist
- Price

- **Implementation:**

- Filter albums where `favorite === true`
- Use `reduce` to sum `price` values

- **Example UI (Stats Tab)**

```
❤️ Favorite Albums Summary
---------------------------
Total Favorites: 4
Total Spent: $68.97

💎 Most Expensive Album:
- Name: Thriller
- Artist: Michael Jackson
- Price: $25.99

📈 Genre: Pop | Average Rating: 4.8
```

Task 8: Unified Screen: AlbumDetailForm.js

Create a **single reusable screen** that handles all the following actions:

- View album details
- Edit album details
- Update favorite status
- Delete an album

This reduces code duplication, simplifies navigation, and unifies the user experience.

Behavior Based on Mode

`mode`	Triggered From	Fields Editable	Buttons Shown	Description
`view`	Tap album name or artist	✅ Yes	Back, Favorite	View and update favorite status
`edit`	Tap ✏️ icon or edit button	✅ Yes	Save, Delete, Back	Update full album info
`favorite`	Tap album from Favorite tab	✅ Yes	Favorite (toggle), Back	Toggle favorite status only
`delete`	Tap 🗑️ delete icon	❌ No	Confirm Delete	Confirm and remove album
(default)	No mode	✅ Yes	Save, Back	Fallback for safety or adding future extensions

Form Fields

Include the following inputs in the form (enabled or disabled based on mode):

- ID (Read-only or auto-generated)
- Album Name (`TextInput`)
- Artist (`TextInput`)
- Release Date (`TextInput` or date picker)
- Genre (`TextInput` or `Dropdown`)
- Track Count (`Numeric input`)
- Price (`Numeric input`)
- Rating (`Slider` or `Numeric input`)
- Favorite (`Switch` or `Toggle Button`)

Navigation Parameters

Pass necessary data using React Navigation:

```
 navigation.navigate('AlbumDetailForm', {
  mode: 'edit',        // 'view', 'edit', 'favorite', 'delete'
  album: albumObject, // album details if editing or viewing
});
```

Buttons and Logic by Mode

Button	When Visible	Action
Save	`edit` mode	Validate form → Save to AsyncStorage → Navigate back to Home
Delete	`edit` mode	Show confirm → Delete from AsyncStorage → Navigate back to Home
Favorite	`view`/`favorite`	Toggle `album.favorite` → Save → Stay or go back
Back	All modes	Navigate back to Home
Cancel	Optional	Discard changes and go back

Form Behavior Notes

- Fields should be **conditionally editable** based on the mode:
  - In `view` or `favorite` mode, make fields read-only except for the `favorite` toggle.
  - In `edit` mode, all fields should be editable.

Example:

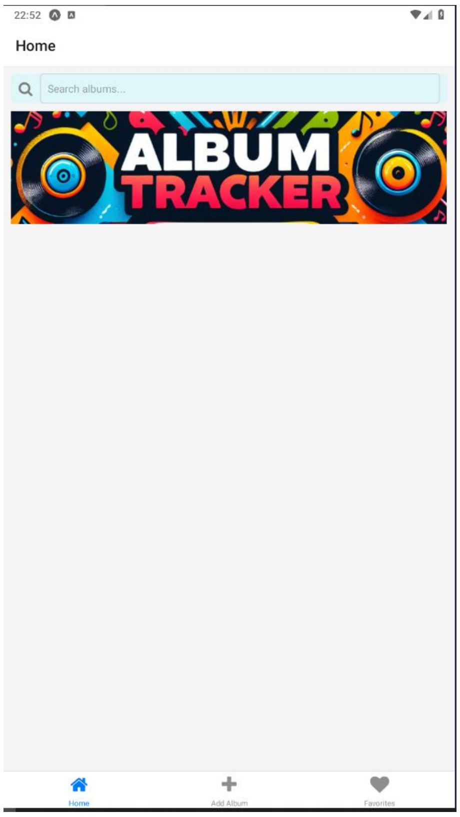
```
<TextInput
  value={album.name}
  editable={mode === 'edit'}
  onChangeText={(text) => setAlbum({ ...album, name: text })}
/>
```

Advantages

- ✅ **Reusable**: One component handles multiple use-cases
- ✅ **Efficient**: Less boilerplate and navigation complexity
- ✅ **Maintainable**: Easier to add validations, styles, and future enhancements
- ✅ **Consistent UX**: Same interface layout and behavior across all interactions

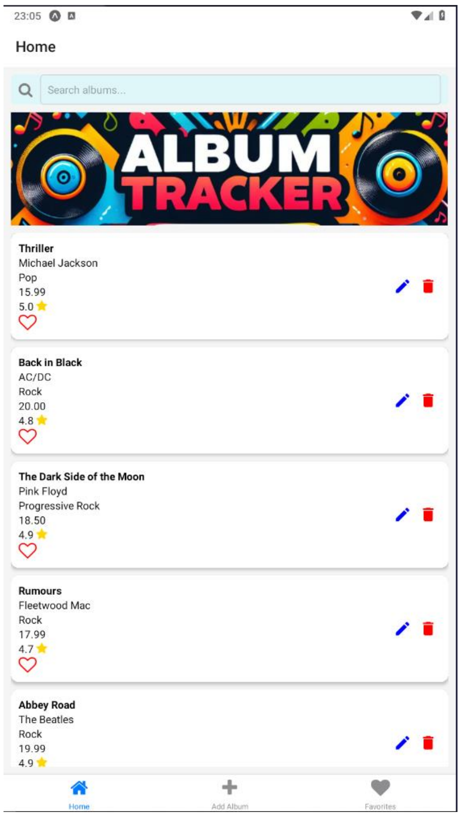
Task 9: Additional Requirements

- Validate input fields and provide immediate feedback for add, update, delete, and mark as favorite actions, including confirmation before deletion
- Ensure the interface is visually appealing and modern, using appropriate colors, icons, and layouts.


22:52

Home

Search albums...


23:05

Home

Search albums...

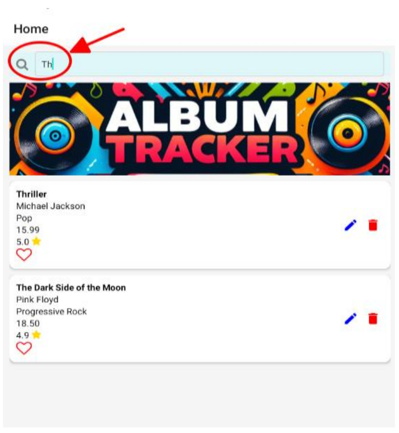
Thriller
Michael Jackson
Pop
15.99
5.0

Back in Black
AC/DC
Rock
20.00
4.8

The Dark Side of the Moon
Pink Floyd
Progressive Rock
18.50
4.9

Rumours
Fleetwood Mac
Rock
17.99
4.7

Abbey Road
The Beatles
Rock
19.99
4.9

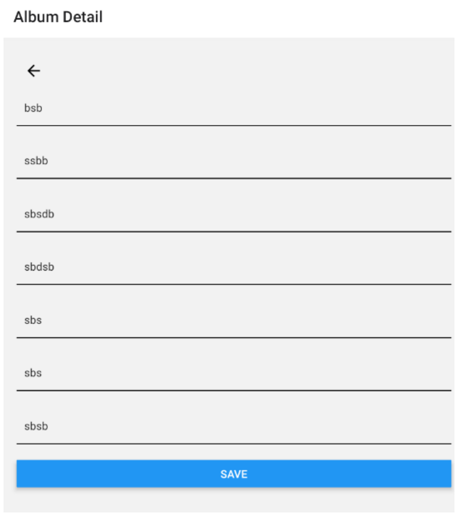

Home

Th

Thriller
Michael Jackson
Pop
15.99
5.0

The Dark Side of the Moon
Pink Floyd
Progressive Rock
18.50
4.9

Figure 1: Home Screen **Figure 2:** The Search Fucntion


Album Detail

bsb

ssbb

sbsdb

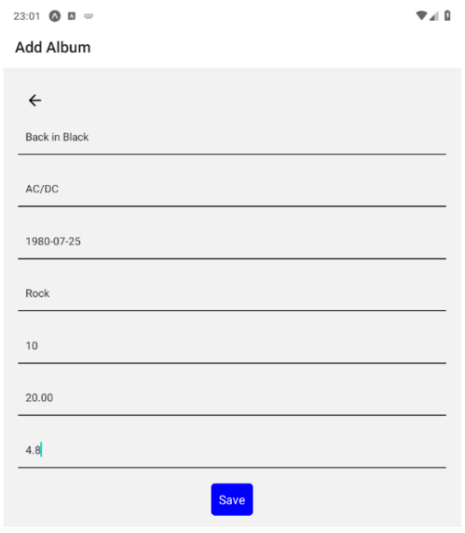
sbdsb

sbs

sbs

sbsb

SAVE


23:01

Add Album

Back in Black

AC/DC

1980-07-25

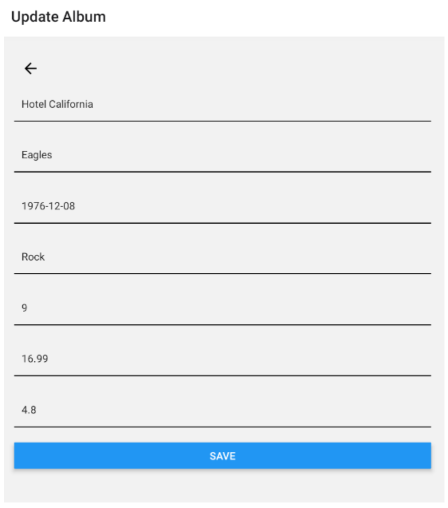
Rock

10

20.00

4.8

Save


Update Album

Hotel California

Eagles

1976-12-08

Rock

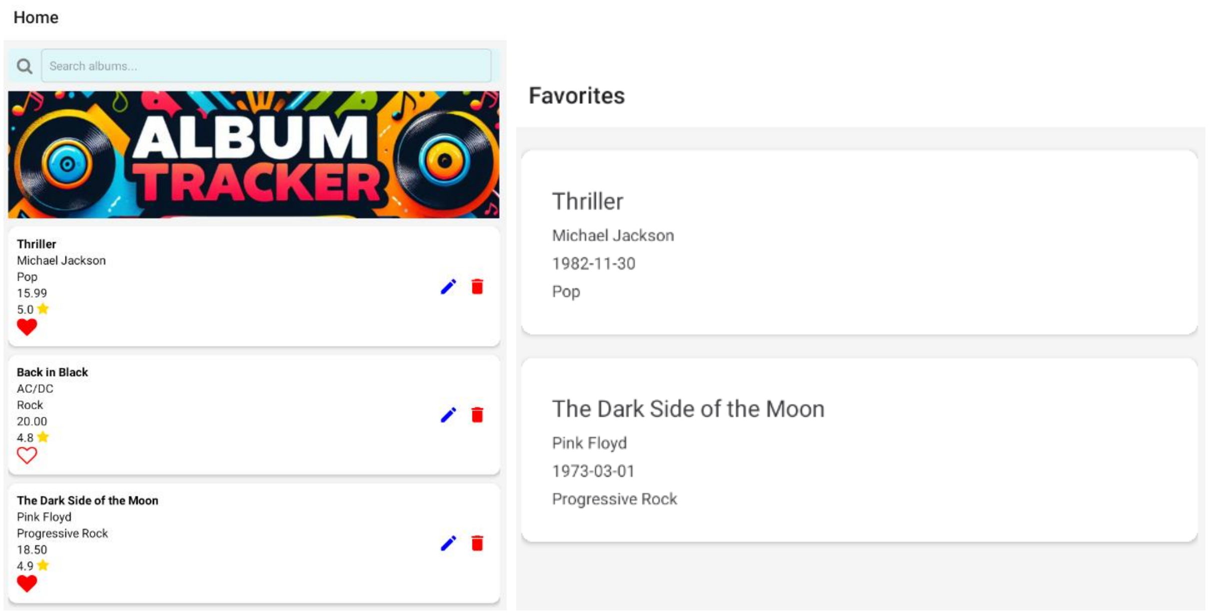
9

16.99

4.8

SAVE

Figure 3: The Album Details/Add/Update Screen


Home

Search albums...

Thriller
Michael Jackson
Pop
15.99
5.0 ★

Back in Black
AC/DC
Rock
20.00
4.8 ★

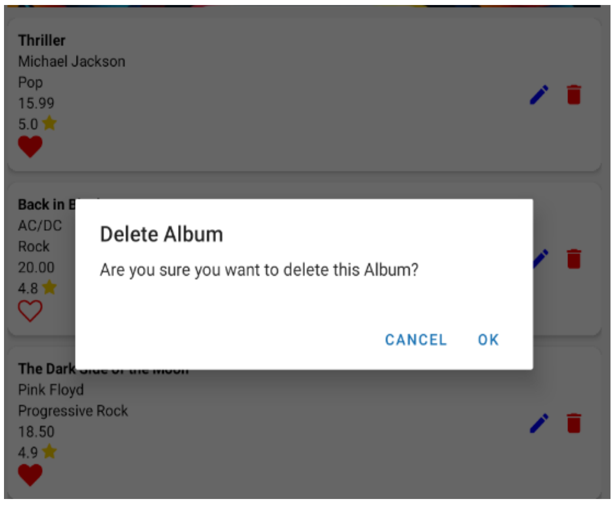
The Dark Side of the Moon
Pink Floyd
Progressive Rock
18.50
4.9 ★

Favorites

Thriller
Michael Jackson
1982-11-30
Pop

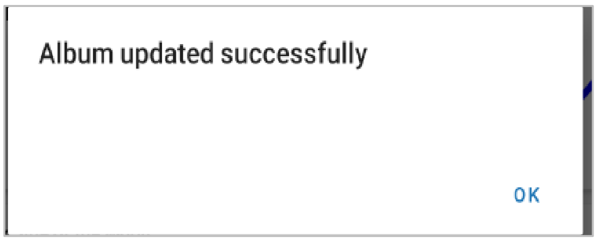
The Dark Side of the Moon
Pink Floyd
1973-03-01
Progressive Rock

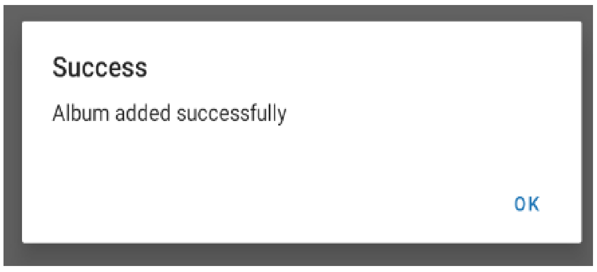
Figure 4: The Favorites Screen

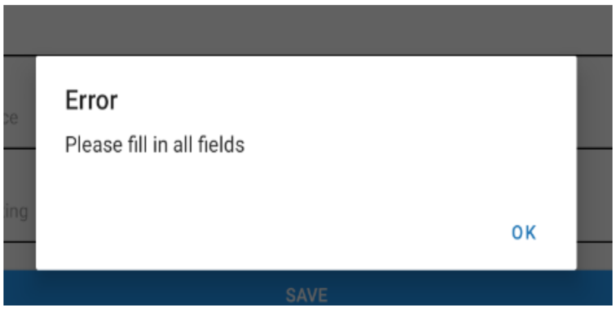

Thriller
Michael Jackson
Pop
15.99
5.0 ★

Delete Album
Are you sure you want to delete this Album?
CANCEL OK

The Dark
Pink Floyd
Progressive Rock
18.50
4.9 ★


Album updated successfully
OK


Success
Album added successfully
OK


Error
Please fill in all fields
OK
SAVE

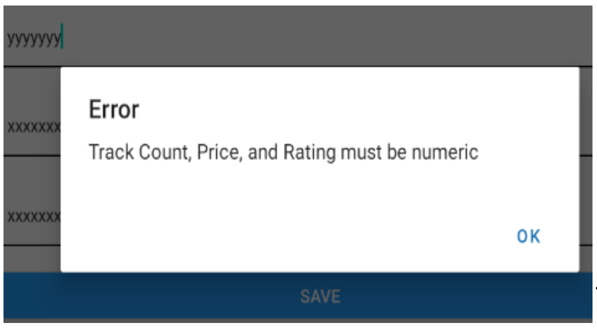

yyyyyyy
Error
Track Count, Price, and Rating must be numeric
OK
SAVE

Figure 5: The confirmation dialog before the delete action and inform after insert/update.

4. HÌNH THỨC NỘP BÀI

- **Gửi bài tập dưới dạng file nén (.zip/.rar)**, gồm:
    - **Chụp screenshoot toàn màn hình: Thư mục, comment tác giả đầu mỗi file source code, cấu trúc vscode đầy đủ**.
    - **source code**
    - **File tài liệu báo cáo (.PDF hoặc .DOCX)**.

- **Đặt tên file nén theo format:**
    - `[MãSV]_[Tên]_assignment2.zip`

      Ví dụ: `SE12345_NguyenVanA_ assignment2.zip`

- **Hạn chót nộp bài:** theo lịch Edunext

5. LƯU Ý QUAN TRỌNG

X **Bài nộp không đầy đủ hoặc thiếu file báo cáo sẽ bị trừ điểm.**
X **Mọi hành vi sao chép code sẽ bị xử lý theo quy định của nhà trường.**
X **Sinh viên cần kiểm tra kỹ lưỡng trước khi nộp bài.**
- **Được sử dụng AI để phân tích và thực hiện bài.**