Movie Explorer - Documentation

Project Overview

The **Movie Explorer** is a Unity-based application that displays movie details, including titles, descriptions, genres, categories, cast, and poster images. It fetches movie data dynamically from **The Movie Database (TMDb) API** and presents related movies for enhanced user experience.

Table of Contents

---

1. Project Structure

The project follows an organized structure for maintainability and scalability.

```
/Assets

|── /Scripts

|   ├── MovieDetailsManager.cs

|   ├── MovieItem.cs

|   ├── Movie.cs

|   ├── IMovieService

|   ├── Panel Manager

|   ├── TMDbAPIManager
```

```
│   ├── MovieSuggestionManager

│   ├── MovieSeriesManager

│   ├── APIKeyManager

│   ├── MovieListManager

│   ├── MovieData

│── /Prefabs

│   ├── MovieItem_Suggestion.prefab

│   ├── MovieItem.prefab
```

- **Scripts** → Contains logic for handling movie data, API calls, and UI updates.
- **UI** → Holds prefabs and UI elements used for displaying movie information.

---

2. Dependencies & Setup

Requirements:

- **Unity 6 (6000.9.38f1)**
- **Newtonsoft.Json** (for handling JSON data)
- **TMDb API Key** (for fetching movie details)

Installing Dependencies:

1. Install Newtonsoft.Json

1. Go to **Window** → **Package Manager**.
2. Click **Add package from the git URL**.
3. Enter: `https://github.com/jilleJr/Newtonsoft.Json-for-Unity.git`

2. Set up TMDb API Key

1. Sign up at **The Movie Database (TMDb)**.

2. Navigate to **Settings > API** and get an API Key.
3. Store it in **Unity PlayerPrefs**:

PlayerPrefs.SetString("TMDbAPIKey", "your_api_key_here");

PlayerPrefs.Save();

---

3. API Integration

The app interacts with **TMDb API** to fetch movie details and related movies.

Endpoints Used:

Feature	API Endpoint
Get Movie Details	`/movie/{movie_id}`
Get Related Movies	`/movie/{movie_id}/similar`
Get Poster Image	`https://image.tmdb.org/t/p/w500/{poster_path}`

API Authentication

The API key is retrieved from PlayerPrefs:

string apiKey = PlayerPrefs.GetString("TMDbAPIKey", "");

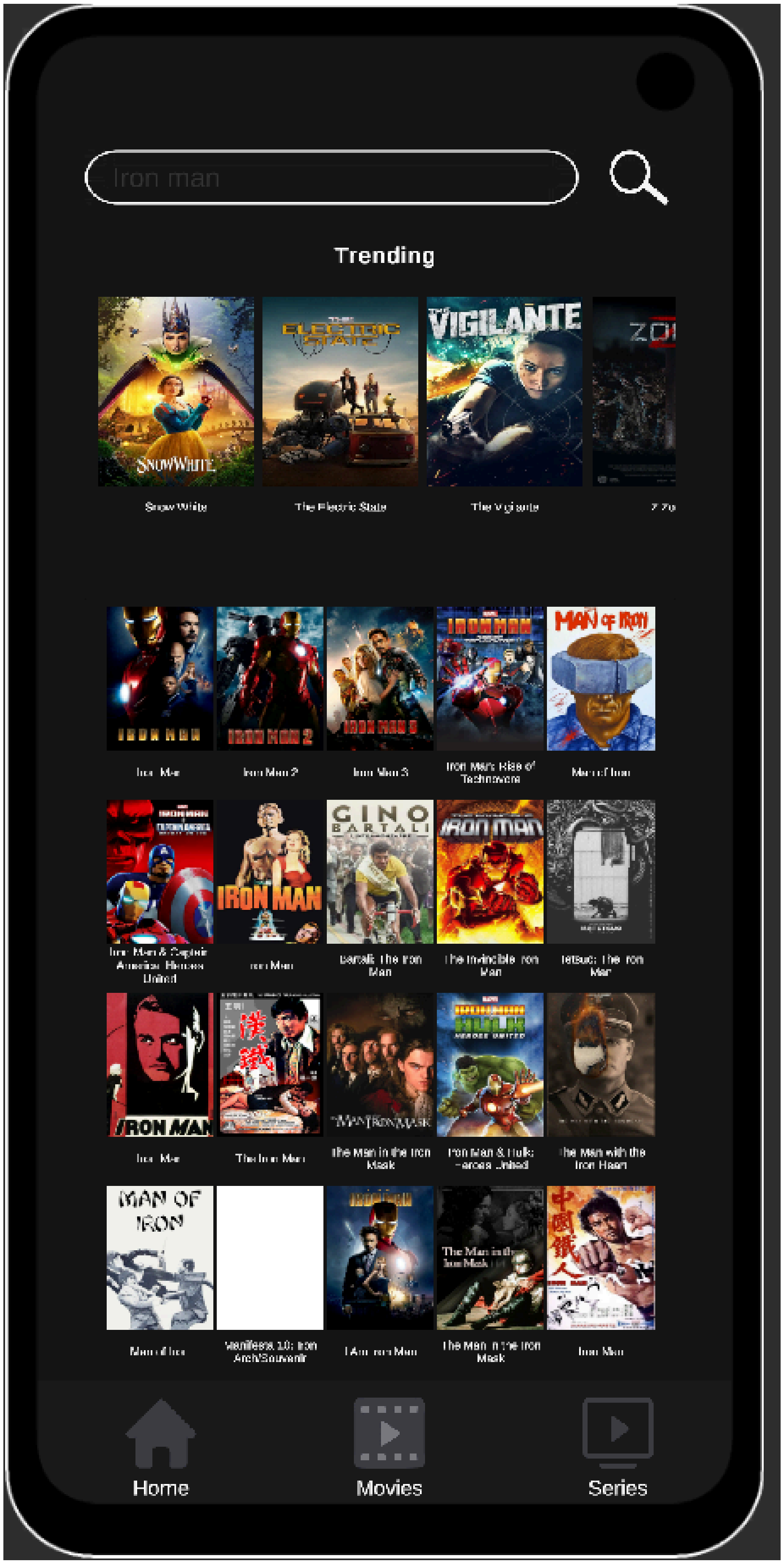
If the key is missing, the app logs an error and prevents API calls.

---

4. How the Application Works

Step 1: Load the homepage with movie suggestions

- The user can select or search for a movie.


Iron man
Trending
Snow White
The Electric State
The Vigilante
Z Zo
Iron Man
Iron Man 2
Iron Man 3
Iron Man: Rise of Technovore
Man of Iron
Iron Man & Captain America: Heroes United
Iron Man
Bartali: The Iron Man
The Invincible Iron Man
Tetsuo: The Iron Man
Iron Man
The Iron Man
The Man in the Iron Mask
Iron Man & Hulk: Heroes United
The Man with the Iron Heart
Man of Iron
Manifesto 10: Iron Arch/Souvenir
I Am Iron Man
The Man in the Iron Mask
Iron Man
Home
Movies
Series

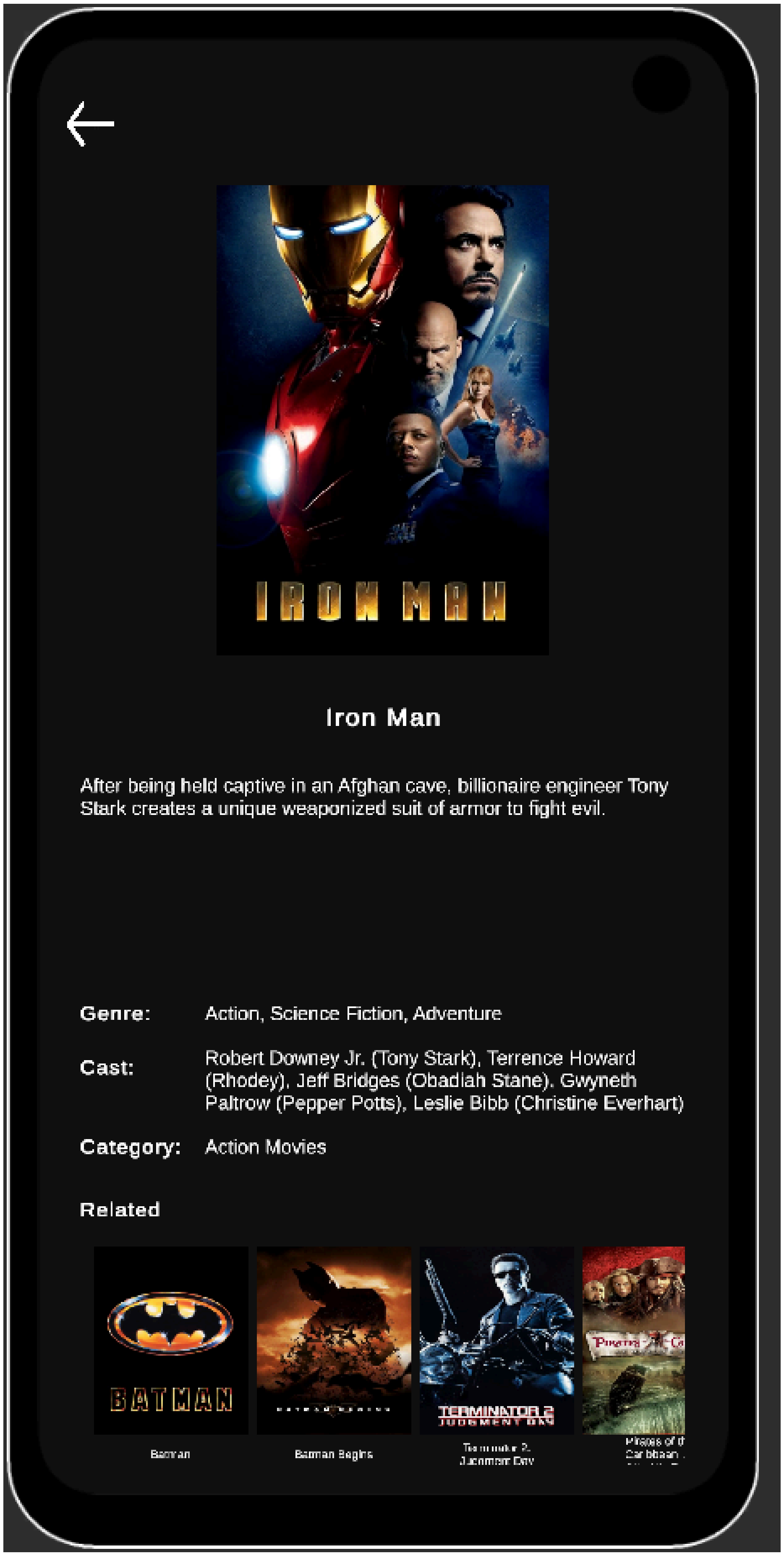
Step 2: Load Selected Movie

- The user selects the movie.
- `MovieDetailsManager` retrieves its details from the **TMDb API**.

Step 3: Fetch & Display Movie Details

- The script formats movie data (title, description, genres, and cast).

- The **poster image** is fetched asynchronously using `UnityWebRequestTexture.GetTexture()`.



Iron Man

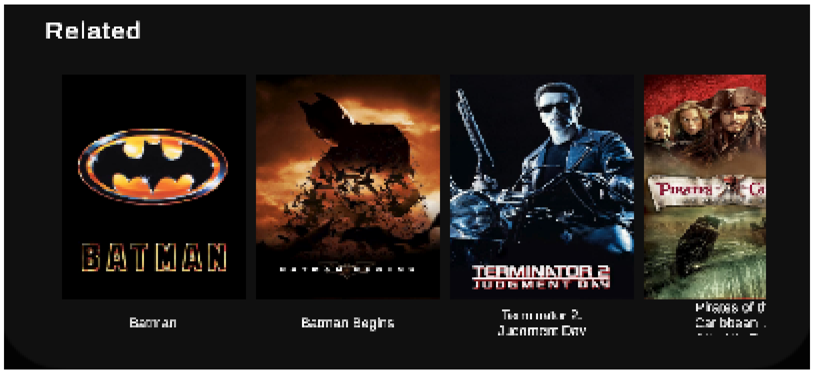
After being held captive in an Afghan cave, billionaire engineer Tony Stark creates a unique weaponized suit of armor to fight evil.

Genre:	Action, Science Fiction, Adventure
Cast:	Robert Downey Jr. (Tony Stark), Terrence Howard (Rhodey), Jeff Bridges (Obadiah Stane), Gwyneth Paltrow (Pepper Potts), Leslie Bibb (Christine Everhart)
Category:	Action Movies

Related

Step 4: Fetch & Display Related Movies

- Calls **TMDb API** for similar movies.
- Limits the display to **six related movies**.
- Updates UI dynamically by instantiating `MovieItem_Suggestion` prefabs.



---

5. Known Limitations & Debugging

1. Unauthorized API Error (401)

Issue: `Failed to fetch related movies: HTTP/1.1 401 Unauthorized`

Fix:

- Ensure the API key is **correctly set** in PlayerPrefs.
- Verify the API key has not expired.
- Confirm the correct **v3 API endpoint** is being used.

2. Slow Image Loading

Issue: Images take time to load.

Fix:

- Implement **caching** for frequently accessed movie posters.
- Use `AsyncOperation` to preload images in the background.

3. Limited Related Movies

Issue: Only six related movies are displayed.

Fix:

- Modify `PopulateRelatedMovies` to display more items dynamically.
