

NOM I COGNOMS: HAMMAD JAMIL
CURS: 2025-2026
GRUP: 2º N DAW

DESENVOLUPAMENT WEB EN ENTORN CLIENT



UF4

NOTA:

Observacions: _____

INS PUIG CASTELLAR
PROFESSOR: David Tomas Gonzalez

ÍNDEX

1. Descripció general.....	2
2. Especificacions tècniques.....	3
2.1. Tecnologies principals.....	3
2.2. APIs externes utilitzades.....	4
2.3. Requisits del sistema.....	4
Per desenvolupament:.....	4
Per usuaris finals:.....	4
2.4. Estructura del projecte.....	4
3. Explicació de l'estructura de dades.....	5
3.1. Interfície Movie.....	5
3.2. Interfície LocationData.....	5
3.3. Interfície ApiResponse.....	5
3.4. Estructura de dades en l'estat de l'aplicació.....	6
4. Explicació del codi.....	6
4.1. Hook personalitzat useApi.....	6
4.2. Servei api.ts.....	7
4.2.1. fetchMovieData.....	7
4.2.2. fetchLocationInfo.....	8
4.3. Component principal App.tsx.....	9
4.4. Component MapComponent.tsx.....	11
5. Ús, exemples i proves.....	13
5.1. Instal·lació i configuració.....	13
5.2. Exemple d'ús complet.....	14
Cerca de pel·lícula.....	14
Exploració de localitzacions.....	14
Gestió de favorites.....	15
5.3. Proves i casos d'ús.....	17
Prova 1: Cerca per títol exacte.....	17
Prova 2: Gestió d'errors.....	17
Prova 3: Persistència de localitzacions favorites.....	17
Prova 4: Canvi de vista del mapa.....	18
6. Possibles millors o extensions.....	19
6.1. Millores funcionals.....	19
6.2. Noves funcionalitats.....	19

GitHub

<https://github.com/hammad2003/cinemamap-explorer>

CinemaMap Explorer - Documentació Tècnica

1. Descripció general

CinemaMap Explorer és una aplicació web interactiva que permet als usuaris descobrir on s'han filmat les seves pel·lícules favorites i aprendre sobre aquests llocs. Utilitzant una interfície intuïtiva, l'aplicació permet cercar pel·lícules, visualitzar les seves localitzacions en un mapa interactiu, i obtenir informació detallada tant de la pel·lícula com dels llocs de rodatge.

L'aplicació combina dades cinematogràfiques amb informació geogràfica i descriptiva de les localitzacions, permetent als amants del cinema explorar la dimensió geogràfica de les seves pel·lícules favorites. Els usuaris poden guardar les seves localitzacions preferides, alternar entre vistes de mapa (estàndard i satèl·lit), i obtenir detalls de Wikipedia sobre les localitzacions.

L'objectiu principal és oferir una experiència educativa i entretinguda que connecti el cinema amb la geografia real, fomentant l'interès tant per les pel·lícules com pels llocs on es van filmar.

CinemaMap Explorer

Discover where your favorite movies were filmed and learn about those places

Search for a movie...

[Movie Info](#) [Location Details](#) [Switch to Satellite](#)

Search for a movie to see information and filming locations

Created with React, TypeScript, and react-leaflet | Data from OMDB API and Wikipedia API

CinemaMap Explorer

Discover where your favorite movies were filmed and learn about those places

The Dark Knight

[Movie Info](#) [Location Details](#) [Switch to Satellite](#)

The Dark Knight
(2008)

Director: Christopher Nolan
Cast: Christian Bale, Heath Ledger, Aaron Eckhart
Genre: Action, Crime, Drama
Rating: ★ 9.0/10

Synopsis
When a menace known as the Joker wreaks havoc and chaos on the people of Gotham, Batman, James Gordon and Harvey Dent must work together to put an end to the madness.

Filming Locations
Click on a location to see it on the map

United States

Created with React, TypeScript, and react-leaflet | Data from OMDB API and Wikipedia API

2. Especificacions tècniques

2.1. Tecnologies principals

- Frontend: React 19 amb TypeScript
- Estils: CSS
- Gestió d'estat: React Hooks (useState, useEffect)
- Mapes: React-Leaflet 5.0.0 i Leaflet 1.9.4

- Peticions HTTP: Axios 1.8.4
- Ícones: React-Icons 5.5.0
- Entorn de desenvolupament: Vite 6.2.0

2.2. APIs externes utilitzades

- OMDb API: Per obtenir informació de les pel·lícules
- Nominatim API (OpenStreetMap): Per obtenir coordenades geogràfiques de les localitzacions
- Wikipedia API: Per obtenir descripcions i imatges de les localitzacions

2.3. Requisits del sistema

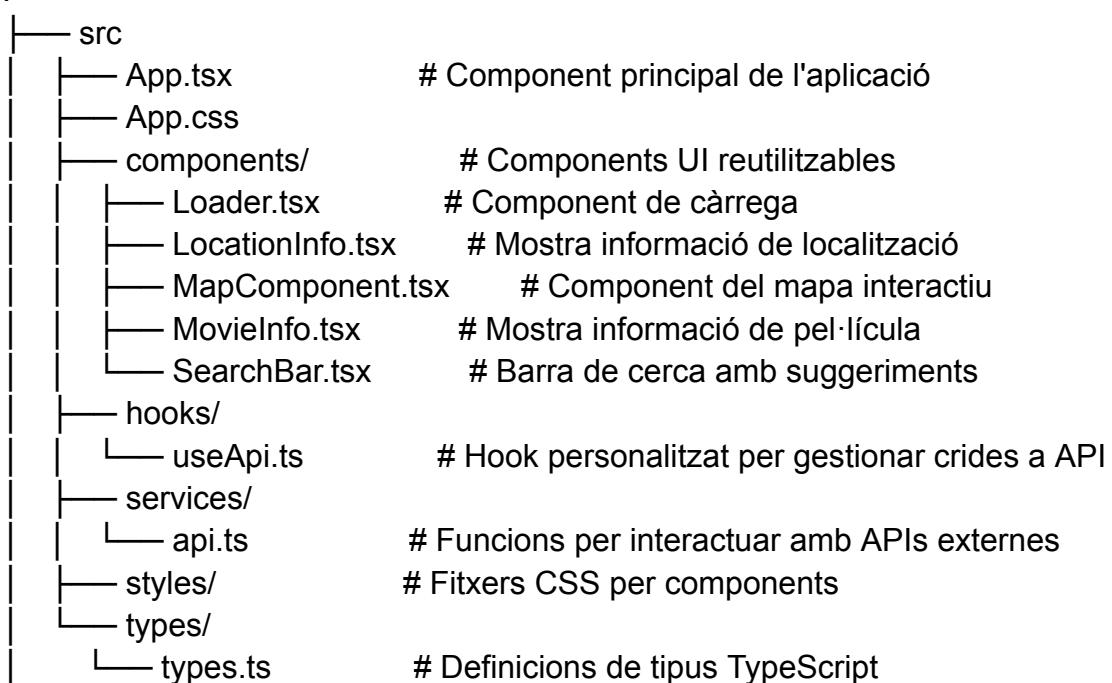
Per desenvolupament:

- Node.js (versió compatible amb React 19)
- npm
- Connexió a internet (per accedir a les APIs externes)

Per usuaris finals:

- Navegador web modern compatible amb React i Leaflet
- JavaScript habilitat
- Connexió a internet

2.4. Estructura del projecte



3. Explicació de l'estructura de dades

L'aplicació utilitza principalment dues estructures de dades principals definides al fitxer types.ts:

3.1. Interfície Movie

Aquesta interfície representa les dades d'una pel·lícula:

```
export interface Movie {  
    title: string;          // Títol de la pel·lícula  
    year: number;           // Any de llançament  
    locations: string[];    // Llista de localitzacions de rodatge  
    director?: string;      // Director (opcional)  
    actors?: string[];       // Llista d'actors principals (opcional)  
    poster?: string;         // URL del pòster (opcional)  
    plot?: string;           // Sinopsi (opcional)  
    imdbRating?: string;     // Puntuació IMDB (opcional)  
    genre?: string;          // Gènere (opcional)  
}
```

3.2. Interfície LocationData

Aquesta interfície representa la informació d'una localització geogràfica:

```
export interface LocationData {  
    name: string;           // Nom de la localització  
    lat: number;            // Latitud  
    lon: number;            // Longitud  
    description: string;    // Descripció breu  
    image?: string;          // URL d'imatge (opcional)  
    wikipediaExtract?: string; // Text extret de Wikipedia (opcional)  
    type?: 'city' | 'country'; // Tipus de localització (opcional)  
}
```

3.3. Interfície ApiResponse

Aquesta interfície genèrica s'utilitza per a gestionar les respostes de les crides a API:

```
export interface ApiResponse<T> {  
    data: T | null;          // Dades rebudes (null si no hi ha dades)
```

```

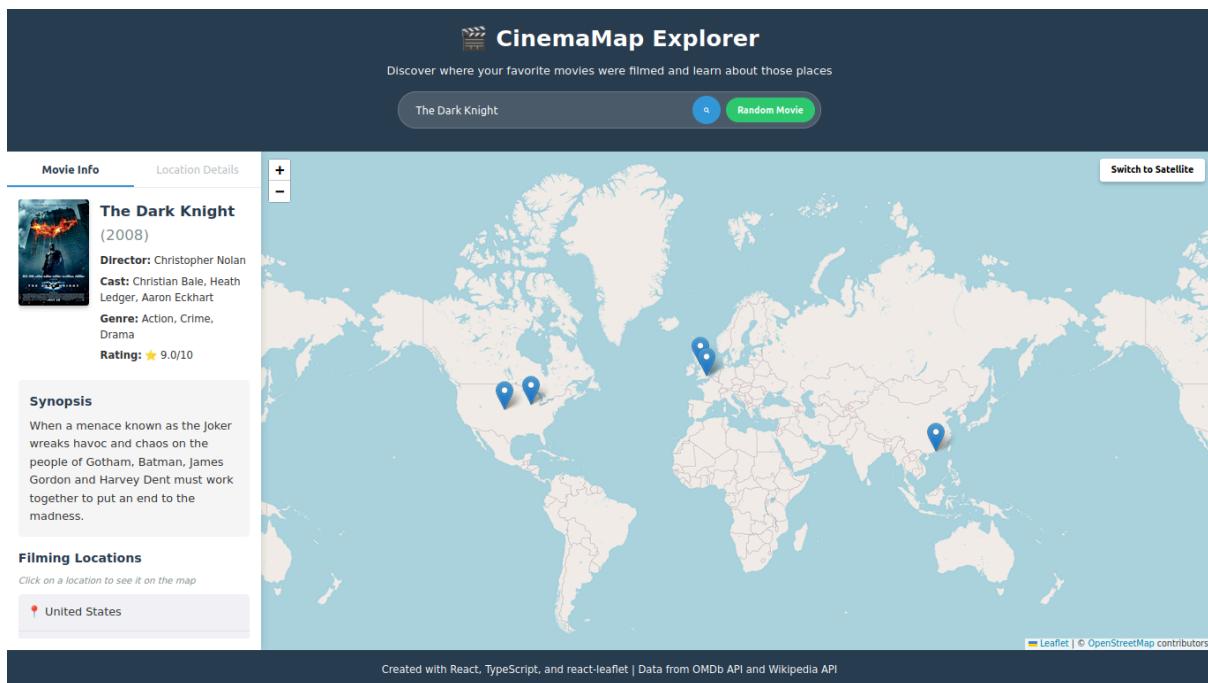
loading: boolean;      // Indicador de càrrega
error: string | null; // Missatge d'error (null si no hi ha error)
}

```

3.4. Estructura de dades en l'estat de l'aplicació

En el component App.tsx, es gestiona l'estat global de l'aplicació amb diversos estats de React:

- movieQuery: string - Consulta de cerca de pel·lícula actual
- selectedLocation: LocationData | null - Localització seleccionada actualment
- locations: LocationData[] - Totes les localitzacions de la pel·lícula actual
- mapView: 'standard' | 'satellite' - Tipus de vista del mapa
- favLocations: LocationData[] - Localitzacions guardades com a favorites
- activeTab: 'movie' | 'location' - Pestanya activa a la barra lateral



4. Explicació del codi

4.1. Hook personalitzat useApi

Aquest hook facilita les crides a API i la gestió dels seus estats:

```
// ./hooks/useApi.ts
```

```

export function useApi<T>(apiCall: () => Promise<T>, dependencies: any[] = []):
  ApiResponse<T> {
  const [data, setData] = useState<T | null>(null);
  const [loading, setLoading] = useState<boolean>(true);
  const [error, setError] = useState<string | null>(null);

  useEffect(() => {
    const fetchData = async () => {
      try {
        setLoading(true);
        const result = await apiCall();
        setData(result);
        setError(null);
      } catch (err) {
        setError(err instanceof Error ? err.message : 'Unknown error occurred');
        setData(null);
      } finally {
        setLoading(false);
      }
    };
    fetchData();
  }, dependencies);

  return { data, loading, error };
}

```

Aquest hook:

1. Inicialitza tres estats: data, loading i error
2. Executa la funció de crida a API quan els dependencies canvien
3. Gestiona automàticament els estats de càrrega i error
4. Retorna un objecte amb els tres estats

4.2. Servei api.ts

Aquest fitxer conté funcions per interactuar amb les APIs externes:

4.2.1. fetchMovieData

```

export const fetchMovieData = async (title: string): Promise<Movie> => {
  // ... implementació
}

```

```

// Cerca la pel·lícula a l'API d'OMDb
const searchResponse = await retryWithBackoff(searchMovie);

// Processa la resposta
const movieData = searchResponse.data;

// Extreu i processa les localitzacions
let locations: string[] = [];
if (movieData.Country && typeof movieData.Country === 'string') {
    locations = movieData.Country.split(/\s*,\s*/).filter(Boolean);
}

// ... més processament de localitzacions

// Retorna l'objecte Movie
return {
    title: movieData.Title,
    year: parseInt(movieData.Year) || 0,
    director: movieData.Director,
    actors: movieData.Actors?.split(/\s*,\s*/).filter(Boolean) || [],
    locations,
    poster: movieData.Poster !== 'N/A' ? movieData.Poster : undefined,
    plot: movieData.Plot,
    imdbRating: movieData.imdbRating,
    genre: movieData.Genre
};
}

```

Aquesta funció:

1. Realitza una petició a l'API d'OMDb amb el títol proporcionat
2. Extreu les localitzacions del rodatge de la resposta
3. Afegeix localitzacions conegudes per a pel·lícules populars
4. Formata i retorna un objecte de tipus Movie

4.2.2. fetchLocationInfo

```

export const fetchLocationInfo = async (locationName: string): Promise<LocationData> => {
    // ... implementació

    // Primer intenta buscar com a ciutat a Nominatim
    let nominatimResponse = await retryWithBackoff(

```

```

() => searchNominatim('city')
);

// Si no troba res, busca com a país o sense especificar tipus
// ... més implementació

// Consulta a Wikipedia per obtenir més informació
try {
    // ... implementació

    return {
        name: display_name,
        lat: parseFloat(lat),
        lon: parseFloat(lon),
        description: wikipediaResponse.data.extract || `Information about
${display_name}`,
        type: firstResult.type as 'city' | 'country',
        wikipediaExtract: wikipediaResponse.data.extract,
        image: wikipediaResponse.data.thumbnail?.source
    };
} catch (wikiError) {
    // En cas d'error, retorna informació bàsica
    return {
        name: display_name,
        lat: parseFloat(lat),
        lon: parseFloat(lon),
        description: `Location: ${display_name}. This is one of the places related to the
movie.`,
        type: firstResult.type as 'city' | 'country'
    };
}
}

```

Aquesta funció:

1. Cerca informació geogràfica d'una localització a l'API de Nominatim
2. Intenta diferents tipus de cerca (ciutat, país, genèric) si la primera falla
3. Consulta l'API de Wikipedia per obtenir informació addicional
4. Retorna un objecte LocationData amb tota la informació recopilada

4.3. Component principal App.tsx

El component principal que coordina tota l'aplicació:

```

export const App = () => {
  // Estats principals
  const [movieQuery, setMovieQuery] = useState("");
  const [selectedLocation, setSelectedLocation] = useState<LocationData | null>(null);
  const [locations, setLocations] = useState<LocationData[]>([]);
  const [mapView, setMapView] = useState<'standard' | 'satellite'>('standard');
  const [favLocations, setFavLocations] = useState<LocationData[]>([]);
  const [activeTab, setActiveTab] = useState<'movie' | 'location'>('movie');

  // Utilitza el hook useApi per obtenir dades de pel·lícules
  const { data: movie, loading: movieLoading, error: movieError } = useApi<Movie | null>(
    () => movieQuery ? fetchMovieData(movieQuery) : Promise.resolve(null),
    [movieQuery]
  );

  // Gestió de localitzacions
  useEffect(() => {
    const fetchLocations = async () => {
      if (movie && movie.locations && movie.locations.length > 0) {
        // ... implementació per obtenir dades de localitzacions
      }
    };
    fetchLocations();
  }, [movie]);

  // Manegadors d'esdeveniments
  const handleSearch = (query: string) => { /* ... */ };
  const handleLocationSelect = (location: LocationData) => { /* ... */ };
  const handleTabChange = (tab: 'movie' | 'location') => { /* ... */ };
  const toggleMapView = () => { /* ... */ };
  const addToFavorites = (location: LocationData) => { /* ... */ };
  const removeFromFavorites = (location: LocationData) => { /* ... */ };

  // Renderització de l'aplicació
  return (
    <div className="app">
      <header>
        <h1>🎬 CinemaMap Explorer</h1>
        <p>Discover where your favorite movies were filmed and learn about those places</p>
    
```

```

<SearchBar onSearch={handleSearch} loading={movieLoading} />
</header>

<div className="content">
  {/* Barra lateral amb informació */}
  <div className="sidebar">
    {/* ... pestanyes i contingut */}
  </div>

  {/* Contenidor del mapa */}
  <div className="map-container">
    {/* ... controls i component de mapa */}
  </div>
</div>

<footer>
  <p>Created with React, TypeScript, and react-leaflet | Data from OMDb API and Wikipedia API</p>
</footer>
</div>
);
};

```

El component App.tsx:

1. Gestiona l'estat global de l'aplicació
2. Coordina les crides a API mitjançant useApi i efectes
3. Implementa la cerca de pel·lícules i la gestió de localitzacions
4. Permet guardar localitzacions favorites a localStorage
5. Renderitza la interície d'usuari principal

4.4. Component MapComponent.tsx

Aquest component encapsula la funcionalitat del mapa interactiu:

```

export const MapComponent = ({
  locations,
  selectedLocation,
  onLocationSelect,
  mapView,
  favoriteLocations
}: MapComponentProps) => {
  // ... configuració d'ícones i helpers
}

```

```

// Determina el centre i zoom del mapa
const center = selectedLocation
? [selectedLocation.lat, selectedLocation.lon] as [number, number]
: locations.length > 0
? [locations[0].lat, locations[0].lon] as [number, number]
: [40, 0] as [number, number];

const zoom = selectedLocation ? 10 : locations.length > 0 ? 3 : 2;

// Determina si una localització és favorita
const isLocationFavorite = (location: LocationData) => { /* ... */ };

// Escull la icona adequada
const getMarkerIcon = (location: LocationData) => { /* ... */ };

// Configura el tipus de mapa (estàndard o satèl·lit)
const tileLayerUrl = mapView === 'standard'
? 'https://{}tile.openstreetmap.org/{z}/{x}/{y}.png'
: 'https://server.arcgisonline.com/ArcGIS/rest/services/World_Imagery/MapServer/tile/{z}/{y}/{x}';

return (
<div className="map-container">
  <MapContainer
    center={center}
    zoom={zoom}
    style={{ height: '100%', width: '100%' }}
    className="map"
  >
    <TileLayer url={tileLayerUrl} attribution={attribution} />
    <MapUpdater center={center} zoom={zoom} />

    {/* Marcadors per a cada localització */}
    {locations.map((location, index) => (
      <Marker
        key={index}
        position={[location.lat, location.lon]}
        icon={getMarkerIcon(location)}
        eventHandlers={{
          click: () => onLocationSelect(location),
        }}
    
```

```

    >
    <Popup>
        {/* ... contingut del popup */}
    </Popup>
    </Marker>
)}
```

{/* Cercle al voltant de la localització seleccionada */}

```

{selectedLocation && (
    <LayerGroup>
        <Circle
            center={[selectedLocation.lat, selectedLocation.lng]}
            pathOptions={{ color: 'red', fillColor: 'red', fillOpacity: 0.2 }}
            radius={50000} // 50km radius
        />
    </LayerGroup>
)}
```

```

</MapContainer>
</div>
);
};

```

Aquest component:

1. Renderitza un mapa interactiu amb Leaflet
2. Mostra marcadors per a cada localització de la pel·lícula
3. Utilitza icones diferents per a localitzacions seleccionades i favorites
4. Permet alternar entre vistes de mapa estàndard i satèl·lit
5. Mostra un cercle de 50km al voltant de la localització seleccionada

5. Ús, exemples i proves

5.1. Instal·lació i configuració

1. Clona el repositori i instal·la les dependències:
 - git clone <https://github.com/hammad2003/cinemamap-explorer>
 - cd cinemamap-explorer
 - npm install
2. Configura claus d'API (si és necessari):
 - L'aplicació utilitza la clau d'API d'OMDb c95ba846, però podria ser necessari canviar-la en cas que caduqui o tingui límits d'ús.

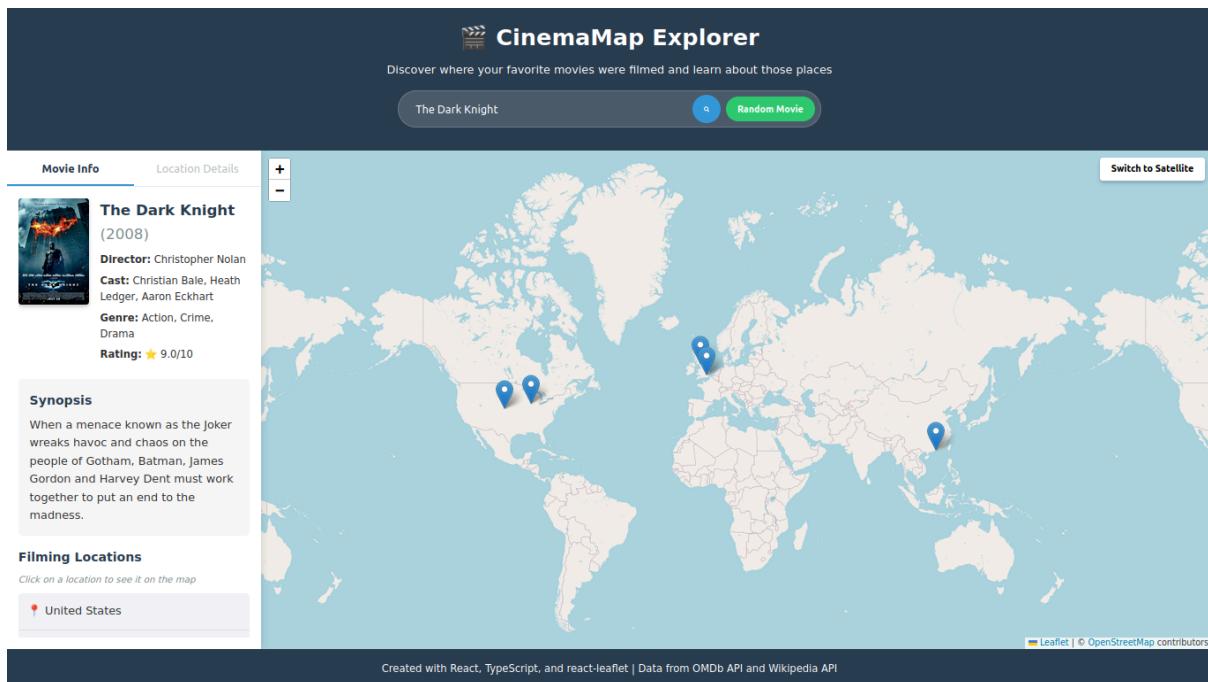
- Inicia l'aplicació en mode desenvolupament:
 - npm run dev

5.2. Exemple d'ús complet

Cerca de pel·lícula

- A la pantalla principal, escriu el nom d'una pel·lícula a la barra de cerca (p. ex., "Inception").
- Prem el botó de cerca o prem Enter.
- L'aplicació mostrarà la informació de la pel·lícula i carregarà les localitzacions al mapa.

```
// Exemple de funcionament intern del codi durant una cerca
handleSearch("Inception");
// Intern: setMovieQuery("Inception");
// Trigger: useApi efectua fetchMovieData("Inception")
// Resultat: Carrega dades de la pel·lícula i les localitzacions
```



Exploració de localitzacions

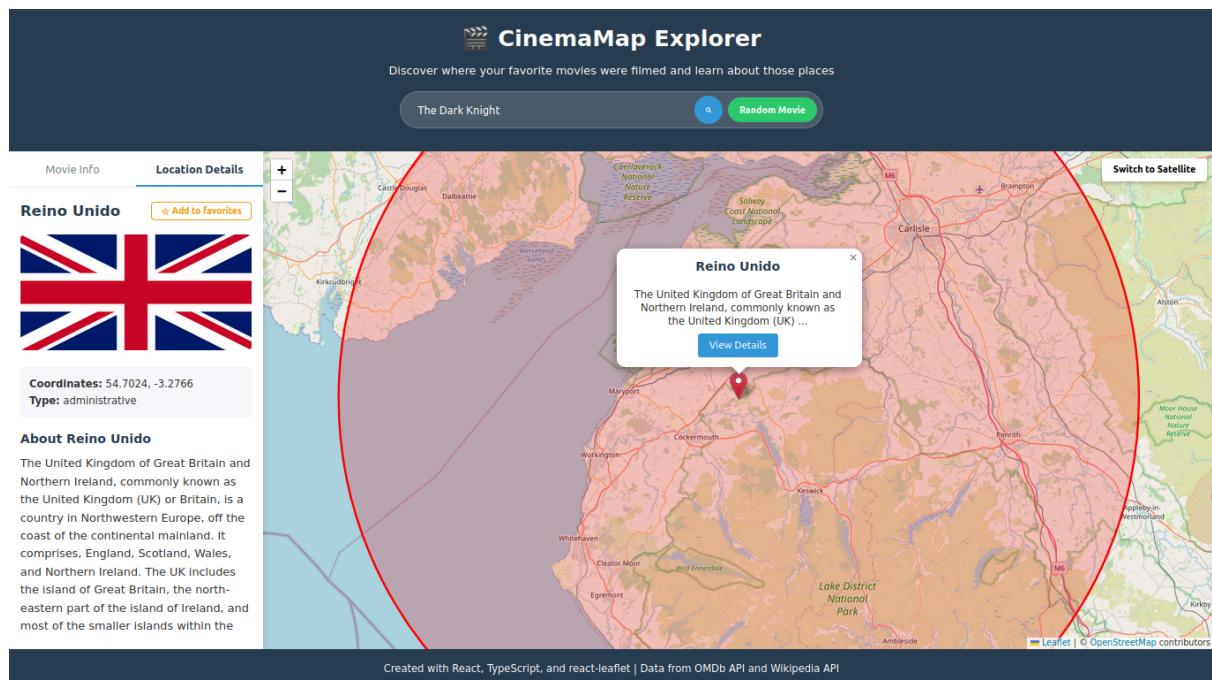
- Fes clic en una localització de la llista de la pel·lícula o directament al mapa.

- L'aplicació mostrarà informació detallada sobre la localització seleccionada i centrarà el mapa en ella.

// Exemple de flux de dades en seleccionar una localització

```
handleLocationSelect({
  name: "Paris, France",
  lat: 48.8566,
  lon: 2.3522,
  description: "..."
});
```

// Intern: setSelectedLocation(location);
// Intern: setActiveTab('location');
// Resultat: La UI mostra els detalls de París i centra el mapa



Gestió de favorites

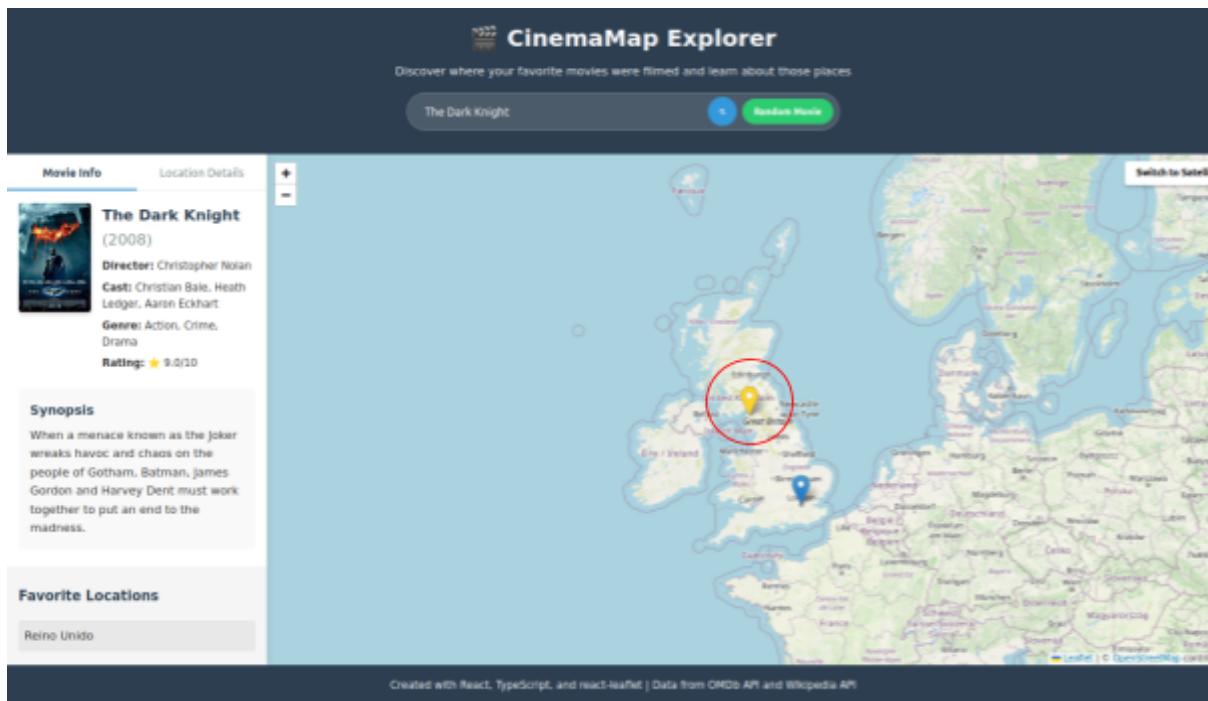
- Visualitza els detalls d'una localització.
- Fes clic al botó "Add to favorites".
- La localització apareixerà a la secció "Favorite Locations" i tindrà una icona daurada al mapa.

// Exemple d'afegir una localització a favorites addToFavorites(location);
// Intern: setFavLocations([...favLocations, location]);

```
// Intern: localStorage.setItem('favoriteLocations', JSON.stringify([...favLocations, location]));
// Resultat: La localització es guarda a favorites i persisteix entre sessions
```

The screenshot shows the CinemaMap Explorer interface. On the left, there is a sidebar with 'Movie Info' and 'Location Details' tabs. Under 'Location Details', the 'Reino Unido' location is selected, highlighted with a red box. A button labeled 'Add to Favorites' is also visible. Below this, there is a small British flag icon, coordinates (54.7024, -3.2766), and a type indicator (administrative). To the right is a map of the United Kingdom with a large red circle centered on London, indicating the search radius for movie filming locations. A tooltip for 'Reino Unido' provides a brief description of the United Kingdom. At the bottom of the map, there is a note about the data source: 'Created with React, TypeScript, and react-leaflet | Data from OMDB API and Wikipedia API'.

This screenshot is similar to the first one, but it includes a red box highlighting the 'Favorite Locations' section in the sidebar. This section contains a single item: 'Reino Unido'. The rest of the interface, including the map and the tooltip for 'Reino Unido', remains the same as in the first screenshot.



5.3. Proves i casos d'ús

Prova 1: Cerca per títol exacte

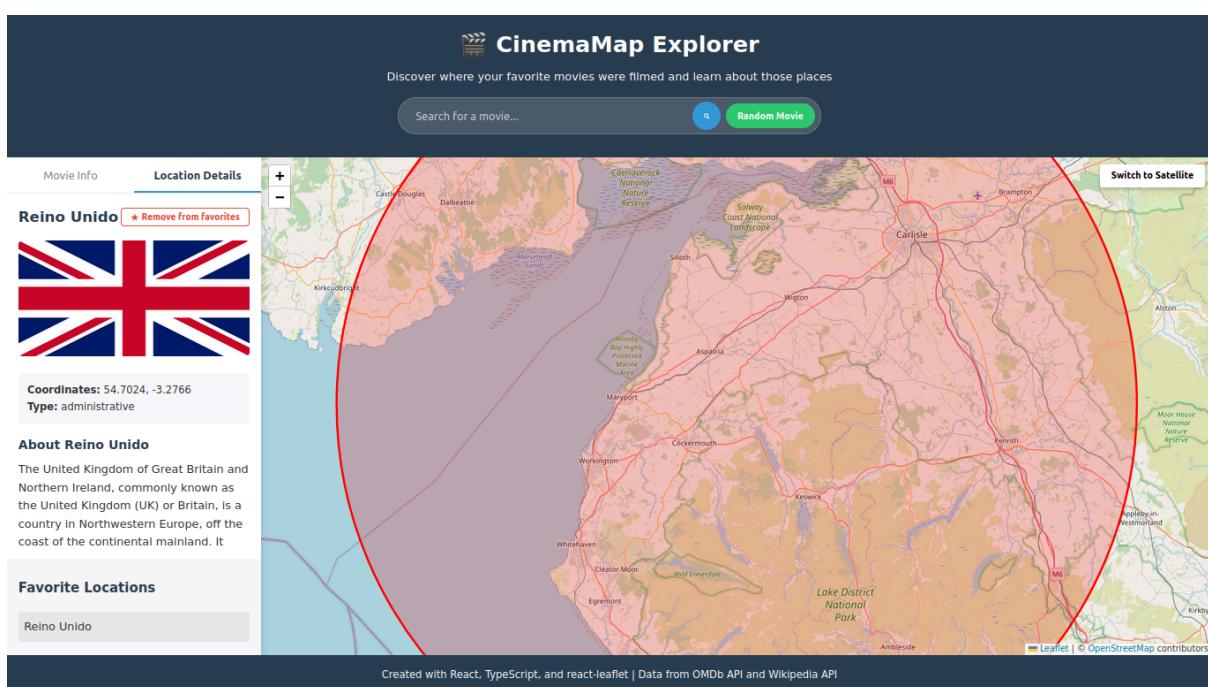
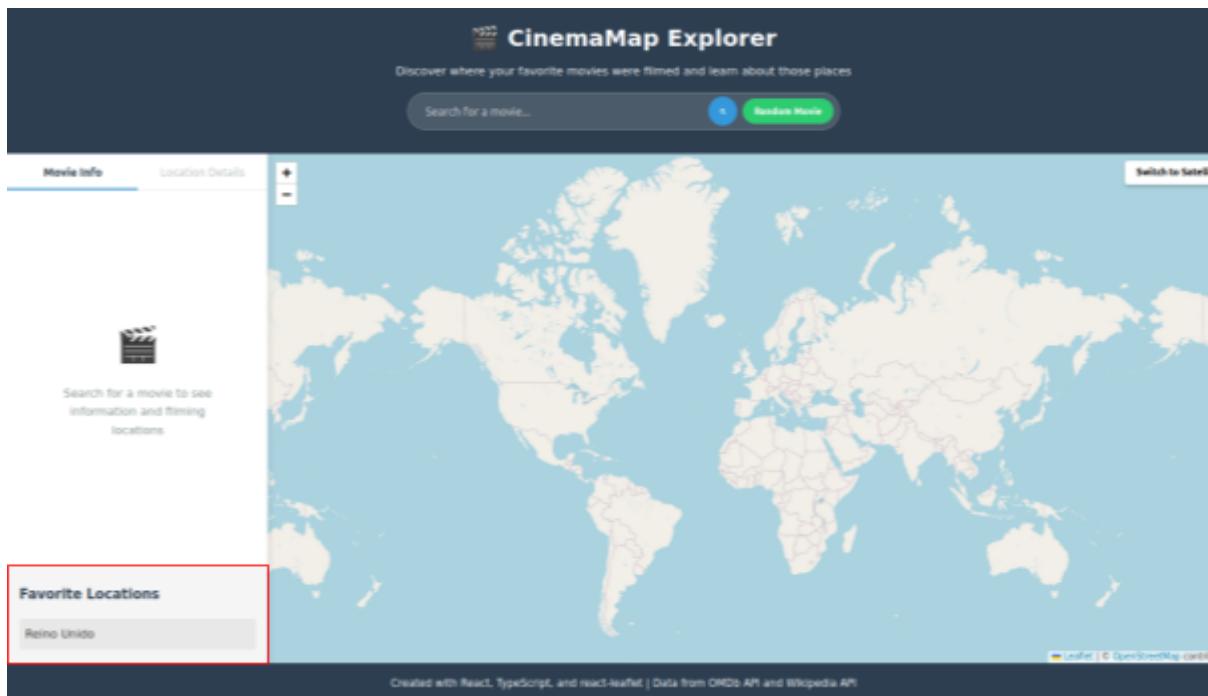
- Entrada: "The Dark Knight"
- Resultat esperat: Informació de la pel·lícula "The Dark Knight" i les seves localitzacions (Chicago, Hong Kong, London)
- Components involucrats: SearchBar, api.ts (fetchMovieData), MovieInfo

Prova 2: Gestió d'errors

- Entrada: Títol inexistent o error d'API
- Resultat esperat: Missatge d'error mostrat a la interfície
- Components involucrats: useApi, App.tsx, components d'informació

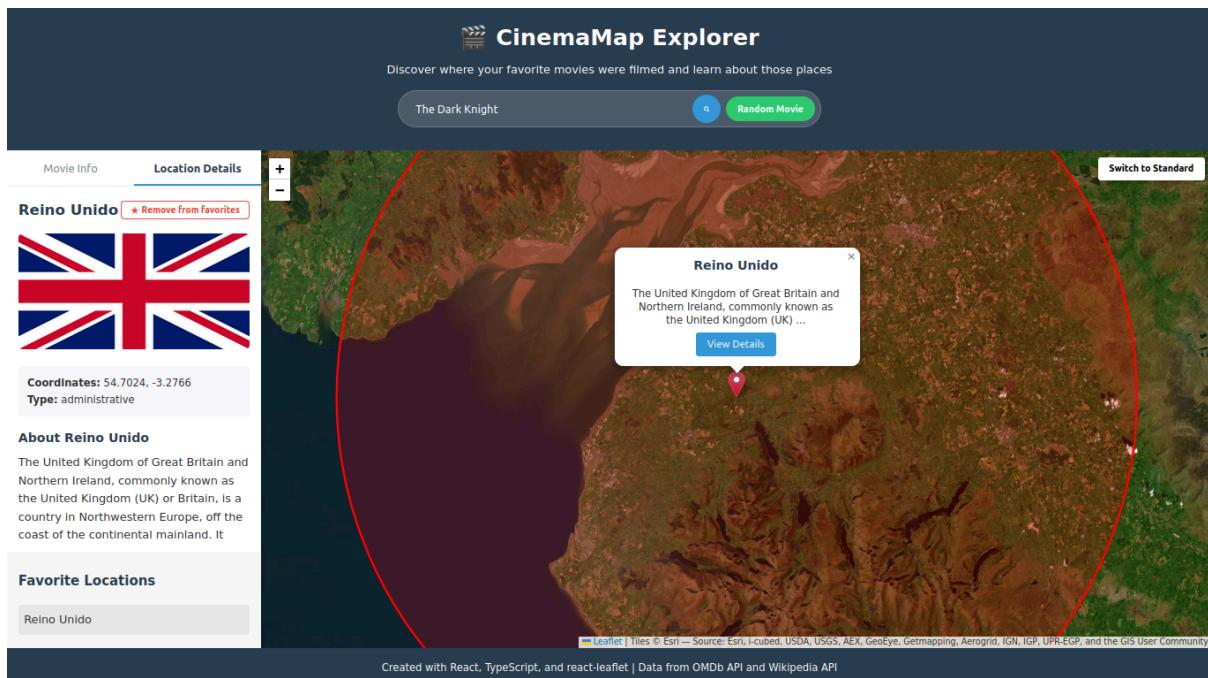
Prova 3: Persistència de localitzacions favorites

- Entrada: Afegir diverses localitzacions a favorites i recarregar la pàgina
- Resultat esperat: Les localitzacions favorites es mantenen després de recarregar
- Components involucrats: LocalStorage, App.tsx (useEffect de càrrega de favorites)



Prova 4: Canvi de vista del mapa

- Entrada: Clic al botó "Switch to Satellite"
- Resultat esperat: El mapa canvia a vista satèl·lit
- Components involucrats: App.tsx (toggleMapView), MapComponent



6. Possibles millors o extensions

6.1. Millores funcionals

- Filtratge avançat de localitzacions: Incorporar filtres per gènere, país de producció, dècada, o director de la pel·lícula, per millorar l'experiència de cerca.
- Ordenació de resultats: Permetre ordenar per proximitat, popularitat, o puntuació (per exemple, IMDb).
- Visualització cronològica: Afegir una línia temporal per veure l'evolució històrica de les localitzacions cinematogràfiques.
- Suport multilingüe: Implementar traduccions de la interfície per arribar a un públic més ampli.

6.2. Noves funcionalitats

- Perfiles d'usuari i localitzacions favorites: Permetre als usuaris registrar-se per guardar i gestionar les seves localitzacions preferides.
- Integració amb serveis externs: Mostrar vídeos de YouTube relacionats (trailers, vlogs), enllaçar amb bases de dades com IMDb, o mostrar la vista actual de la localització mitjançant Google Street View.