

Movie Searching App

The Enhanced Text Extractor Tool is a software application designed to accurately extract and enrich text from diverse sources such as documents, images, and web pages. It employs advanced techniques including optical character recognition (OCR) and data enrichment algorithms to ensure high accuracy and reliability in text extraction. The tool supports multiple file formats and offers a user-friendly interface for easy navigation and customization. With its ability to extract and enrich text efficiently, the Enhanced Text Extractor Tool is a valuable resource for organizations and individuals looking to digitize, process, and analyze textual content effectively.

Table of contents (TOC)

- ☐ Requirements
- ☐ FrontEnd
- ☐ BackEnd
- ☐ Usage Instructions.
- ☐ OMDB API

Requirements

- React: A JavaScript library for building user interfaces.
- React Router: A routing library for React applications, allowing for navigation and routing between different components.
- Axios: A promise-based HTTP client for making requests to APIs and handling responses.
- Bootstrap: A front-end framework for building responsive and mobile-first websites. It provides pre-styled components and a grid system for layout.
- React Router DOM: A routing library for React applications specifically designed for the web. It provides components for handling routing in the browser.
- Node.js: A JavaScript runtime environment that allows you to run JavaScript on the server side.
- Express.js: A web application framework for Node.js that simplifies the process of building web applications and APIs.

Command to install : npm install react react-router-dom

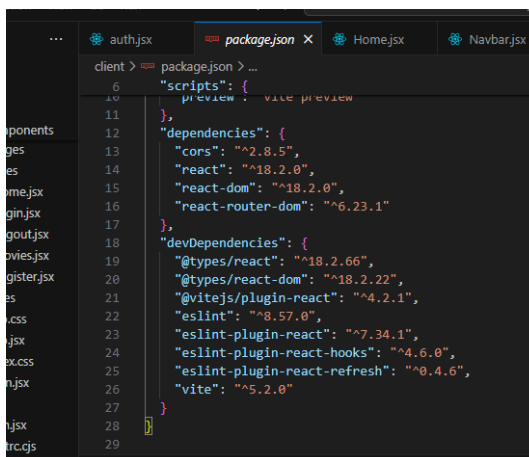
FrontEnd Dependencies:

React.js: React is the main frontend library you're using. It allows you to create reusable UI components and manage the application's state efficiently. React uses a component-based architecture, making it easier to build complex UIs.

React Router: React Router is a library for handling routing in React applications. It enables navigation between different pages or views in a single-page application (SPA) without the need for page reloads. This is useful for creating a multi-page experience within a single-page application.

Axios: Axios is a promise-based HTTP client for making requests to APIs from the browser. You're using Axios to handle HTTP requests to your backend server, such as fetching data or submitting forms.

CSS: You can also use custom CSS (Cascading Style Sheets) to style your components further or override Bootstrap styles as needed. CSS allows you to customize the appearance of your application and make it visually appealing.



```
client > package.json > ...
  6  "scripts": {
  7    "preview": "vite preview"
  8  },
  9  "dependencies": {
 10    "cors": "^2.8.5",
 11    "react": "^18.2.0",
 12    "react-dom": "^18.2.0",
 13    "react-router-dom": "^6.23.1"
 14  },
 15  "devDependencies": {
 16    "@types/react": "^18.2.66",
 17    "@types/react-dom": "^18.2.22",
 18    "@vitejs/plugin-react": "^4.2.1",
 19    "eslint": "^8.57.0",
 20    "eslint-plugin-react": "^7.34.1",
 21    "eslint-plugin-react-hooks": "^4.6.0",
 22    "eslint-plugin-react-refresh": "^0.4.6",
 23    "vite": "^5.2.0"
 24  }
 25 }
```

BackEnd Dependencies:

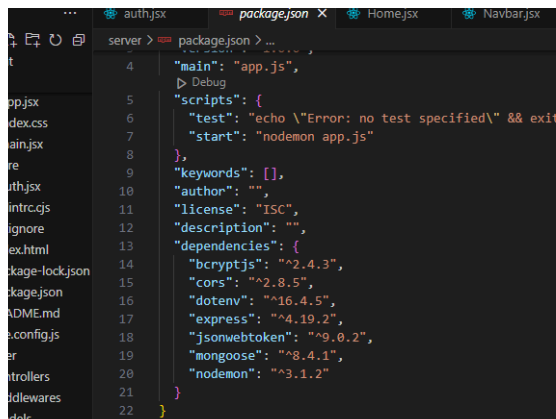
Node.js: Node.js is a JavaScript runtime environment that allows you to run JavaScript on the server side. It provides an event-driven architecture and non-blocking I/O operations, making it suitable for building scalable and high-performance server-side applications.

Express.js: Express.js is a minimalist web application framework for Node.js. It provides a robust set of features for building web servers and APIs, including routing, middleware support, and template engines. Express simplifies the process of handling HTTP requests and responses, making it easier to build RESTful APIs and web applications.

Database: Depending on your project requirements, you might be using a database to store and manage data. Common choices for databases in Node.js applications include MongoDB (a NoSQL database), MySQL, PostgreSQL, or SQLite. You'll use a database to store user information, application data, or any other relevant data.

Middleware: Express.js allows you to use middleware functions to perform tasks like parsing request bodies, authenticating users, and handling errors. Middleware functions can be chained together to create a pipeline that processes incoming requests before passing them to route handlers.

RESTful APIs: If your project involves client-server communication, you'll likely be building RESTful APIs with Express.js. RESTful APIs use HTTP methods (such as GET, POST, PUT, DELETE) to perform CRUD (Create, Read, Update, Delete) operations on resources. Express.js makes it easy to define routes and handle incoming requests to perform these operations.



```
{
  "name": "server",
  "version": "1.0.0",
  "main": "app.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1",
    "start": "nodemon app.js"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": "",
  "dependencies": {
    "bcryptjs": "^2.4.3",
    "cors": "^2.8.5",
    "dotenv": "^16.4.5",
    "express": "^4.19.2",
    "jsonwebtoken": "^9.0.2",
    "mongoose": "^8.4.1",
    "nodemon": "^3.1.2"
  }
}
```

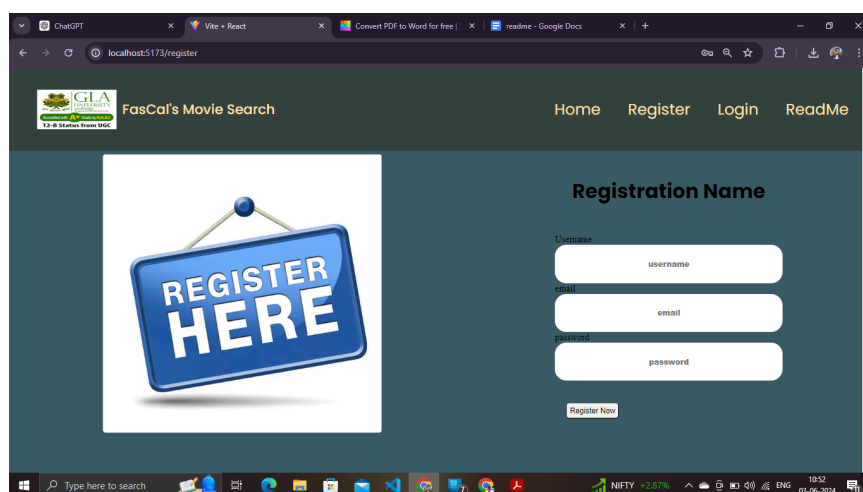
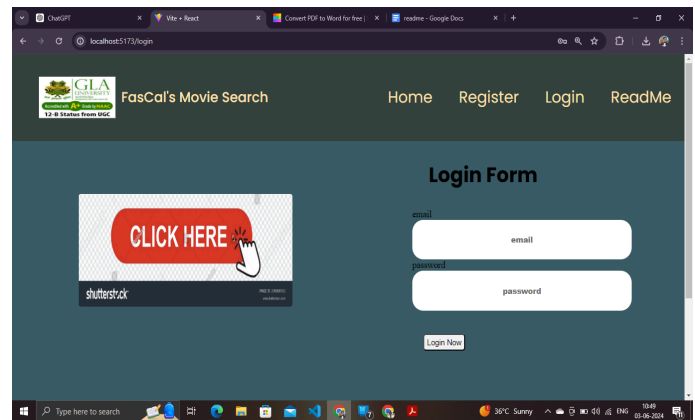
TIP: Run the FrontEnd and BackEnd Servers.

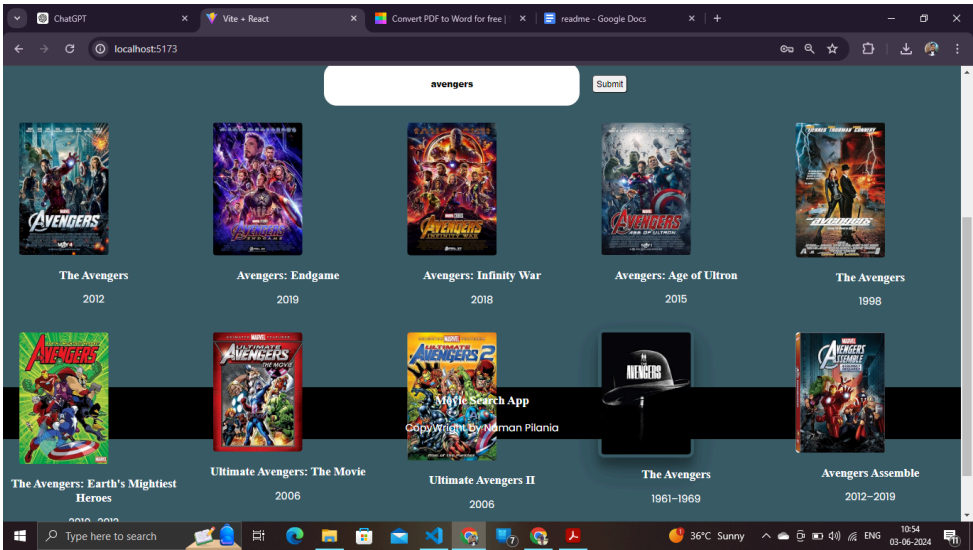
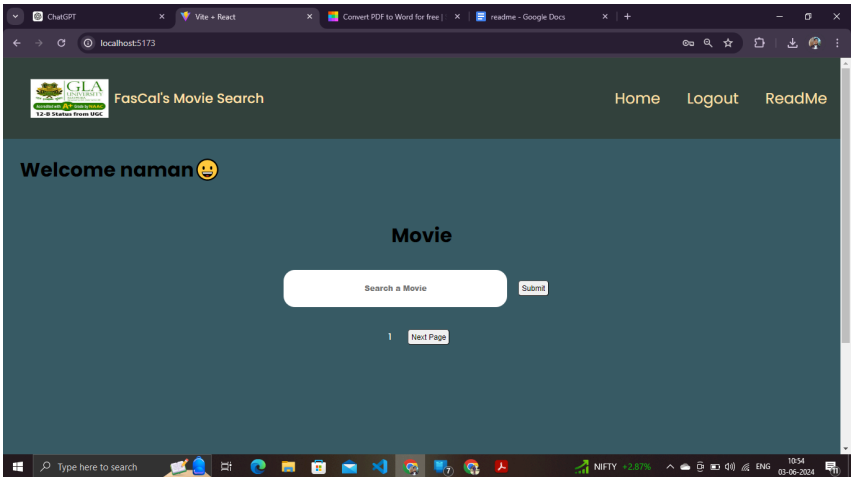
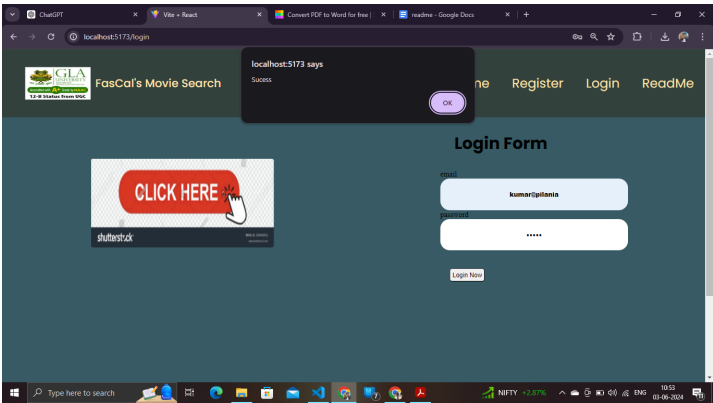
Usage Instructions

The Landing page will show you the LoginScreen.(This app is designed as such that you will only proceed when you are authenticated.

1. Proceed to Login (If Registered). OR Proceed to SignUp
2. Login.
3. In Prompt. or Search box, type the favorite movie title your want to search.
4. This will generate the Output. and a Next Button will be there
5. After logout, you will again redirected to login page and the search bar will get disappear..

This WEB Application also has Navbar with About Section in it. Visit there to see about the Application Details.





OMDB API: -

The OMDb (Open Movie Database) API is a web service that provides access to a vast amount of movie, TV show, and actor information. It's a popular resource for developers who want to integrate movie-related data into their applications.

Key Features of the OMDb API

- **Movie and TV Show Information:** Provides detailed information about movies and TV shows, including titles, release years, genres, directors, actors, plot summaries, language, country, and awards.
- **Poster Images:** Returns URLs to poster images of movies and TV shows.
- **Ratings:** Includes ratings from various sources like IMDb, Rotten Tomatoes, and Metacritic.
- **Search Functionality:** Allows searching for movies and TV shows by title. Supports paging through search results.
- **ID-Based Lookups:** Enables lookup of detailed information using specific IDs, such as IMDb IDs.
- **Flexible Output Formats:** Supports both JSON and XML response formats.

How to Use the OMDb API

Get an API Key:

To use the OMDb API, you need to obtain an API key. You can get a free API key by signing up on the OMDb API website.

Making Requests:

You can make requests to the API using the HTTP GET method.

Example Api : `http://www.omdbapi.com/?s=Inception&apikey=YOUR_API_KEY`

Search Result: In JSON format:

```
{
  "Search": [
    {
      "Title": "Inception",
      "Year": "2010",
      "imdbID": "tt1375666",
      "Type": "movie",
      "Poster": "https://someposterurl.com"
    },
  ],
  "totalResults": "2",
  "Response": "True"
}
```

