Creating a fully functional website as described involves multiple stages: front-end design, back-end development, and integration with third-party platforms. I’ll guide you through the process of creating a basic prototype. Given the complexity of this project, we’ll start with the key components that will make it functional.

We can break down the project into these steps:

1. **Frontend Development** (User Interface)
2. **Backend Development** (Server-side logic and API calls)
3. **Third-party API Integration** (Fetching course data)
4. **Deployment**

Since you're looking to create a website, let's focus on setting up the **frontend** and **backend** structure, as well as a basic **course search** functionality. You can later expand it to integrate real-time data from platforms like Udemy or Coursera.

**1. Frontend: React App**

Let's set up a React app that allows users to search for courses.

**Step 1: Set up React App**

You’ll need to install **Node.js** and **npm** if you don’t already have them installed.

# Create a new React app

npx create-react-app course-aggregator

cd course-aggregator

npm start

**Step 2: Create Basic Structure**

Now, we’ll create a simple structure for your app:

* **Components**: SearchBar, CourseCard, CourseList, CourseDetail, etc.
* **Pages**: HomePage, SearchResultsPage, etc.

Here’s a simple version of the **SearchBar** component:

// src/components/SearchBar.js

import React, { useState } from "react";

const SearchBar = ({ onSearch }) => {

const [query, setQuery] = useState("");

const handleSearch = () => {

if (query) {

onSearch(query);

}

};

return (

<div className="search-bar">

<input

type="text"

placeholder="Search for a course"

value={query}

onChange={(e) => setQuery(e.target.value)}

/>

<button onClick={handleSearch}>Search</button>

</div>

);

};

export default SearchBar;

And the **HomePage** will include this search bar:

// src/pages/HomePage.js

import React, { useState } from "react";

import SearchBar from "../components/SearchBar";

import CourseList from "../components/CourseList";

const HomePage = () => {

const [courses, setCourses] = useState([]);

const searchCourses = (query) => {

// You will later make an API call here to fetch courses

console.log(`Searching for courses on: ${query}`);

};

return (

<div className="home-page">

<h1>Find Your Perfect Online Course</h1>

<SearchBar onSearch={searchCourses} />

<CourseList courses={courses} />

</div>

);

};

export default HomePage;

**CourseList** Component (just for layout):

// src/components/CourseList.js

import React from "react";

import CourseCard from "./CourseCard";

const CourseList = ({ courses }) => {

return (

<div className="course-list">

{courses.map((course, index) => (

<CourseCard key={index} course={course} />

))}

</div>

);

};

export default CourseList;

**CourseCard** Component (simplified):

// src/components/CourseCard.js

import React from "react";

const CourseCard = ({ course }) => {

return (

<div className="course-card">

<h2>{course.title}</h2>

<p>{course.platform}</p>

<p>{course.price}</p>

<p>{course.rating}</p>

<p>{course.duration}</p>

<button>View Course</button>

</div>

);

};

export default CourseCard;

**Step 3: Styling**

You can use **CSS** or a library like **Material-UI** or **Bootstrap** to style the website.

**2. Backend: Express API**

Now we’ll set up a basic **Express** server to handle course data requests.

**Step 1: Set up Express**

In the root directory (outside the React app), initialize a new Node project:

# Initialize a new Express app

mkdir backend

cd backend

npm init -y

npm install express axios cors

Create the backend server in backend/server.js:

// backend/server.js

const express = require("express");

const cors = require("cors");

const axios = require("axios");

const app = express();

const port = 5000;

app.use(cors()); // Allow cross-origin requests

// Dummy API endpoint for course search

app.get("/api/courses", async (req, res) => {

const query = req.query.query;

// Simulated response: You will later fetch actual data from platforms

const courseData = [

{

title: `Learn ${query} on Udemy`,

platform: "Udemy",

price: "$29.99",

rating: "4.5",

duration: "10 hours",

},

{

title: `Master ${query} on Coursera`,

platform: "Coursera",

price: "$39.99",

rating: "4.7",

duration: "12 hours",

},

];

res.json(courseData);

});

app.listen(port, () => {

console.log(`Server running at http://localhost:${port}`);

});

**Step 2: Fetching Data from the Backend**

In your **React app**, use Axios or Fetch to call the backend API.

Modify the searchCourses function in HomePage.js:

// src/pages/HomePage.js

import React, { useState } from "react";

import axios from "axios";

import SearchBar from "../components/SearchBar";

import CourseList from "../components/CourseList";

const HomePage = () => {

const [courses, setCourses] = useState([]);

const searchCourses = async (query) => {

try {

const response = await axios.get(`http://localhost:5000/api/courses?query=${query}`);

setCourses(response.data);

} catch (error) {

console.error("Error fetching course data", error);

}

};

return (

<div className="home-page">

<h1>Find Your Perfect Online Course</h1>

<SearchBar onSearch={searchCourses} />

<CourseList courses={courses} />

</div>

);

};

export default HomePage;

**3. Integration with Third-party APIs**

For this prototype, we’re using mock data, but in a real-world app, you'd integrate APIs from platforms like **Udemy** or **Coursera**.

Here’s a simplified example of how you might fetch courses from Udemy's public API (note: Udemy API might require authentication):

const fetchUdemyCourses = async (query) => {

const response = await axios.get(`https://www.udemy.com/api-2.0/courses/`, {

params: { search: query },

headers: {

"Authorization": `Bearer YOUR\_UDEMY\_API\_KEY`,

},

});

return response.data;

};

You would call this function in the Express backend and send the results to the frontend.

**4. Deployment**

To deploy your app:

* **Frontend**: You can deploy the React app using **Vercel**, **Netlify**, or **GitHub Pages**.
* **Backend**: You can deploy the Express server on **Heroku**, **DigitalOcean**, or **AWS**.

**Final Thoughts:**

This is a basic prototype to get you started. Once you have the core functionality in place, you can focus on improving:

* User authentication (allow users to save courses).
* Real-time data fetching from platforms (via APIs or web scraping).
* A more polished design and additional features (like course reviews, ratings, etc.).

Let me know if you’d like further help with any specific part of this process!