1. API Choice

* Browse through the [provided list](https://github.com/appbrewery/public-api-lists) and choose an API of interest. This choice should be guided by the potential to retrieve, manipulate, and present data in a meaningful and interactive way. I recommend choosing an API that does not require authentication and is CORS enabled. ([What is CORS?](https://medium.com/@electra_chong/what-is-cors-what-is-it-used-for-308cafa4df1a))

**Have decided to use a Game of Thrones API -** [**https://anapioficeandfire.com/**](https://anapioficeandfire.com/)

**No authentication is required on this API, so it only supports GET requests.**

[**https://github.com/joakimskoog/AnApiOfIceAndFire/wiki/Characters**](https://github.com/joakimskoog/AnApiOfIceAndFire/wiki/Characters) **is the root path for pulling up information on characters. Essentially want to generate a ‘what game of thrones character am I’ website which makes a request to get a particular character at random from this API.**

**Look at the ‘get a specific character’ section of** [**https://github.com/joakimskoog/AnApiOfIceAndFire/wiki/Characters**](https://github.com/joakimskoog/AnApiOfIceAndFire/wiki/Characters) **- essentially want to use math.floor(math.random) to generate a character number at random and add it to the API url to make the request. Don’t need to know the exact amount of characters in order to set this up – I know that they go at least up to 823 so can set it up for the first 500 for example. I HAVE FIGURED OUT THAT THERE ARE 2134 CHARACTERS IN THE API.**

**What data do I want to pull in about the character to show on the screen? I think the following: -**

* **Name**
* **Gender**
* **Born**
* **Died**
* **Aliases**

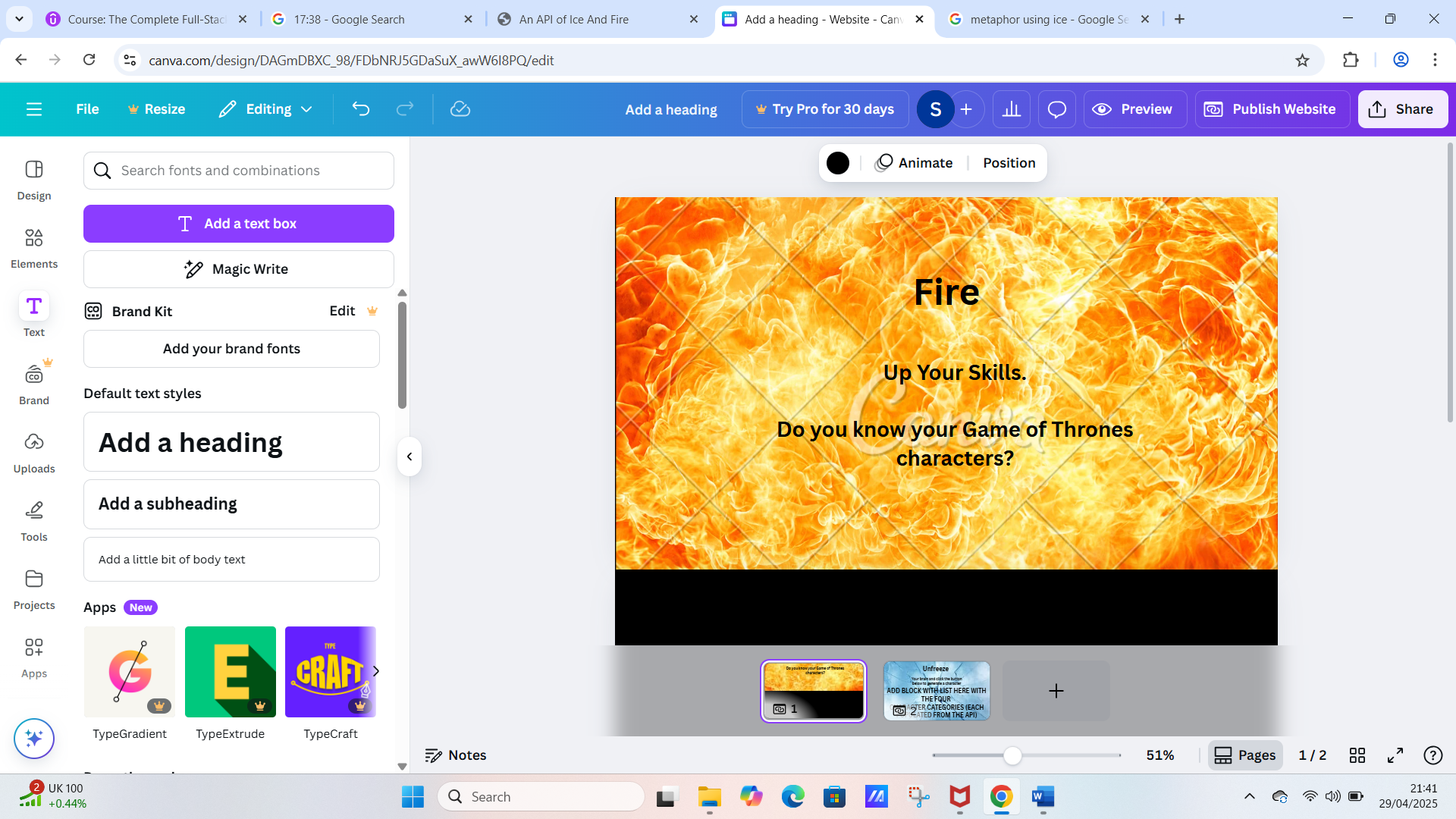
**Can I use this information to populate a table?**

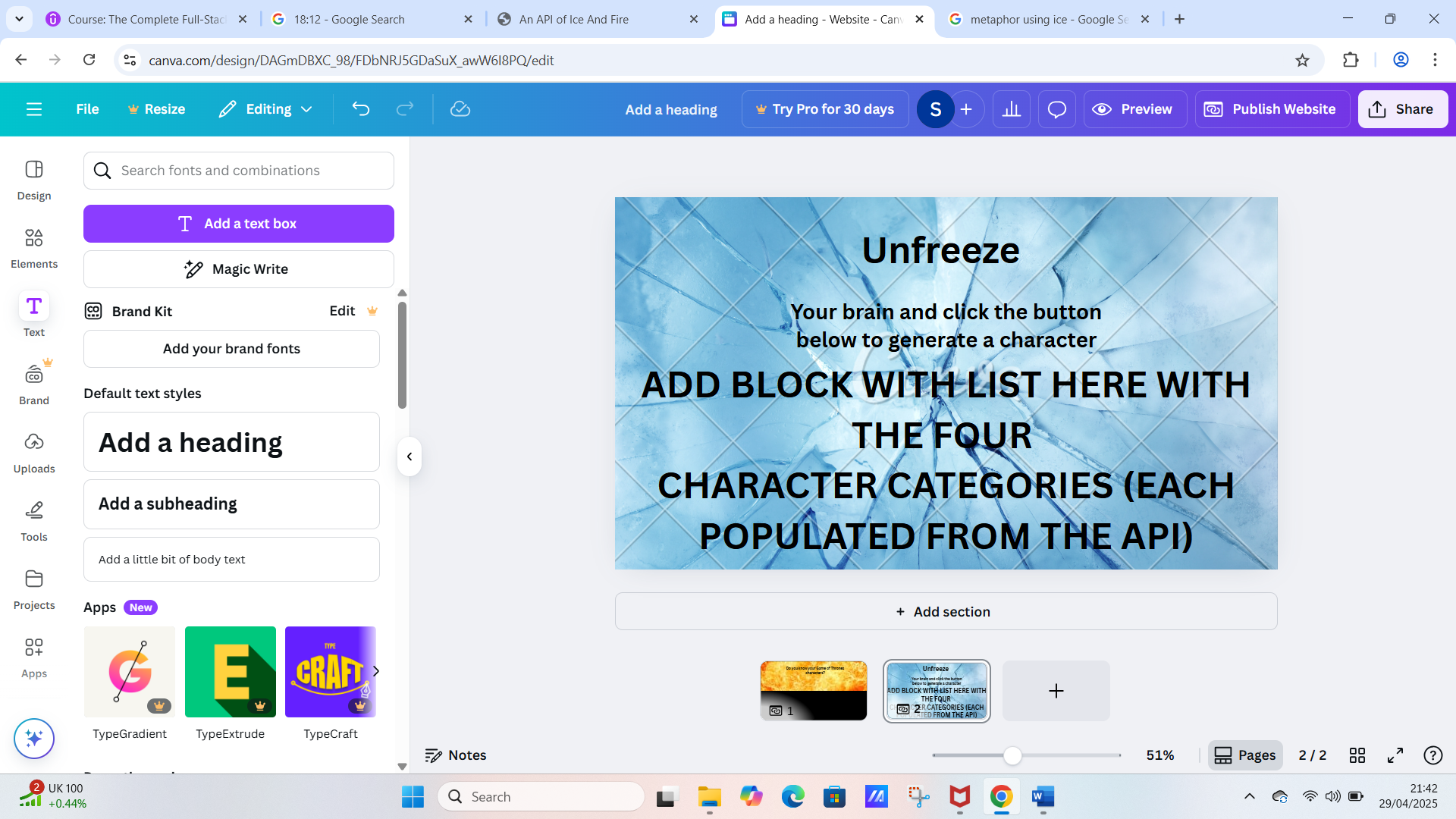
**I could have a static image that is game of thrones related, such as a background of an image of ice and one of fire, which I could use as my background in some way.**

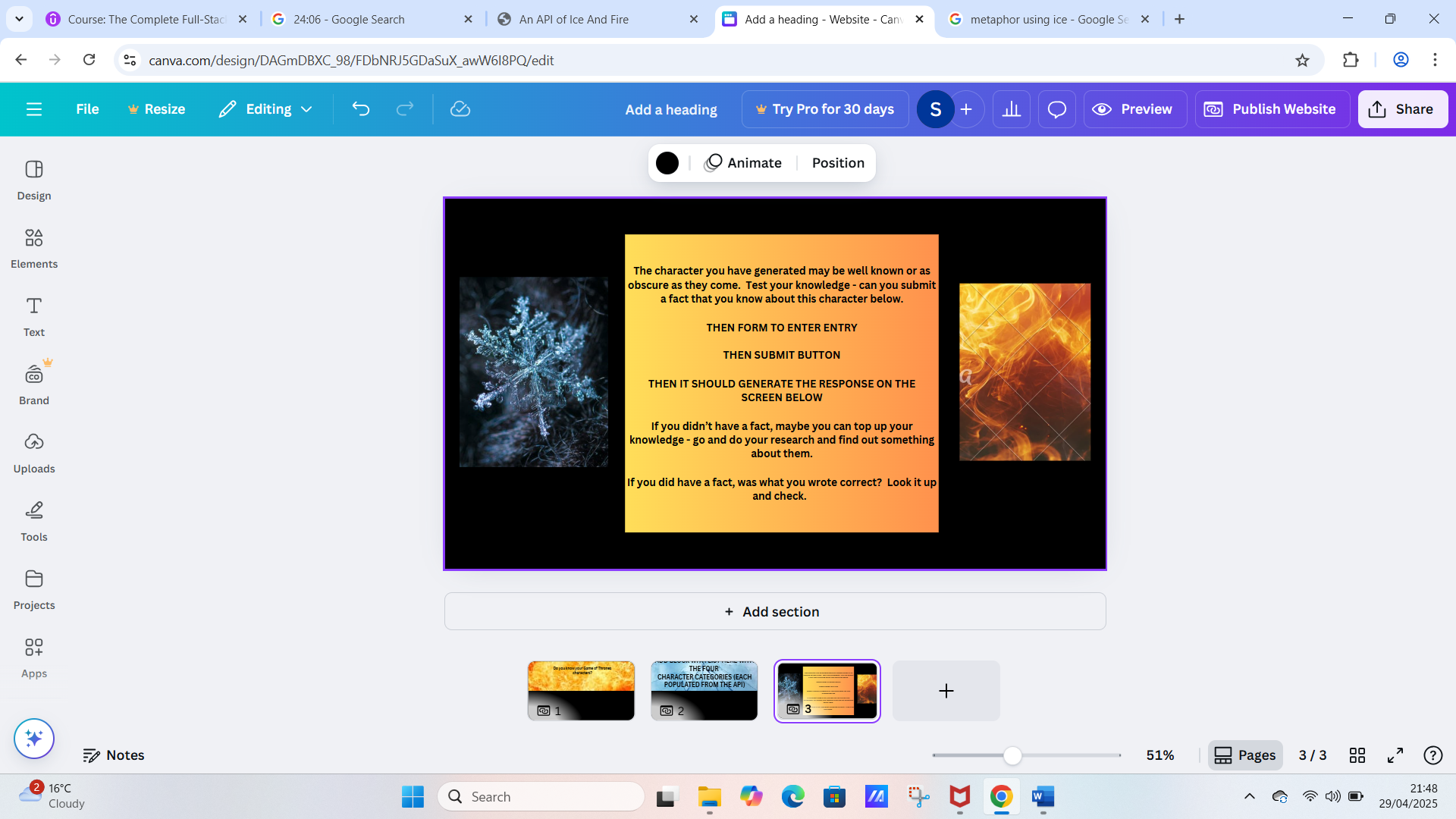
**Then I could add a form where the user could do something regarding the character, such as …. It’s a test your knowledge of GoT characters (including very obscure ones) challenge. There will be a form where it gets people to write down a fact about the person and then go and check it’s accuracy – or if they didn’t know a fact to go and look one up.**

2. Project Planning

* Think through your project, researching the chosen API, its features, what data it will provide, and how it will be used in your web application.







NEED IT TO WORK THAT WHEN YOU PRESS THE NEW CHARACTER BUTTON THEIR FORM BOX CLEARS AS WELL.

**Then I also want a header and footer as partials. The footer can just contain my email address and copyright statement. On the header, could I link to an external site to help people in their research – e.g.** [**https://www.cosmopolitan.com/entertainment/tv/g12022444/minor-game-of-thrones-characters/**](https://www.cosmopolitan.com/entertainment/tv/g12022444/minor-game-of-thrones-characters/)

3. Project Setup

* Set up a new Node.js project using Express.js.
* Include Axios for making HTTP requests.
* Include EJS for templating.
* Ensure that the project has a structured directory and file organization.

4. API Integration

* Implement at least a GET endpoint to interact with your chosen API.
* Use Axios to send HTTP requests to the API and handle responses.

**BUILD THE HEADER AND FOOTER AND INDEX FILES, ADDING IN THE IMAGES THAT I WANT, AND BUILD THE FUNCTIONALITY IN THE GET AND POST ROUTES AS I GET TO THE RELEVANT SECTIONS IN THE INDEX.EJS.**

**APP.GET ROUTE IS NOW COMPLETE. ADD THE ICE IMAGE INTO THE INDEX.EJS AND SUPERIMPOSE THE BOX, TO BE POPULATED WITH API DATA, ON TOP OF IT.**

5. Data Presentation

* Design the application to present the retrieved data in a user-friendly way. Use appropriate HTML, CSS, and a templating engine like EJS.

6. Error Handling

* Ensure that error handling is in place for both your application and any API requests. You can console log any errors, but you can also give users any user-relevant errors.

7. Documentation

* Include comments throughout your code to explain your logic.

8. Code Sharing

* Use what you have learnt about GitHub to commit and push your project to GitHub so that you can share it with other students in the Q&A area, I'd love to see what you've build too! You can tweet at me @yu\_angela
* Include a Readme.md file that explains how to start your server, what commands are needed to run your code. e.g. npm i  and then nodemon index.js

**HAVE AN ISSUE WITH IT JUST BEING A CONSTANTLY SPINNING BALL WHILE TRYING TO GENERATE A CHARACTER. HAVE REALISED COULD SET THE FUNCTION UP TO JUST TARGET IDS THAT I KNOW GENERATE CHARACTERS PLEASE SEE NOTES BELOW FROM CHAT GPT. I HAVE ALSO MADE A LIST BELOW OF IDS THAT I KNOW GENERATE CHARACTERS.**

**Character IDs that I know generate reasonably well-known characters -**

can i set it to generate a character from a subset that I know are on the sytem

**ChatGPT said:**

Yes, absolutely — that's a great idea and a very effective fix! 🎯

If you already know a subset of valid character IDs (i.e., those that exist and have names), you can **avoid slow, failed API calls altogether** by picking from only those known-good ones.

**✅ Here's how to do it:**

**1. Create a list of valid character IDs**

For example:

js

CopyEdit

const knownCharacterIds = [

16, 27, 38, 45, 148, 232, 339, 583, 957, 1052

// Add more IDs here that you’ve confirmed work

];

**2. Modify your generateCharacter function to pick from that list:**

js

CopyEdit

async function generateCharacter() {

const baseUrl = "https://anapioficeandfire.com/api/characters";

// Pick a random ID from the known list

const randomIndex = Math.floor(Math.random() \* knownCharacterIds.length);

const characterId = knownCharacterIds[randomIndex];

try {

const response = await axios.get(`${baseUrl}/${characterId}`);

const character = response.data;

return {

name: character.name || "Unknown",

gender: character.gender || "Unknown",

DOB: character.born || "Unknown",

died: character.died || "Unknown",

nicknames: character.aliases.length > 0 ? character.aliases : ["None"]

};

} catch (error) {

console.error("Error fetching character:", error.message);

// Return a fallback if something fails unexpectedly

return {

name: "No character found",

gender: "Unknown",

DOB: "Unknown",

died: "Unknown",

nicknames: ["None"]

};

}

}

**✅ Benefits of this approach:**

* **Fast**: No waiting or retrying.
* **Stable**: Fewer API failures or unexpected 404s.
* **Controlled**: You can curate a good set of characters (popular ones, etc.).

Would you like help building a good list of known character IDs that work well (e.g., popular ones like Jon Snow, Arya Stark, etc.)?

4o

Top of Form

Bottom of Form