Task-by-Task Guide

If you'd like a little more support while completing this project, explore this step-by-step resource to get additional hints and resources to help you along each task of this project.

Task 0 - Start with a Plan in Mind

Start With A Plan in Mind

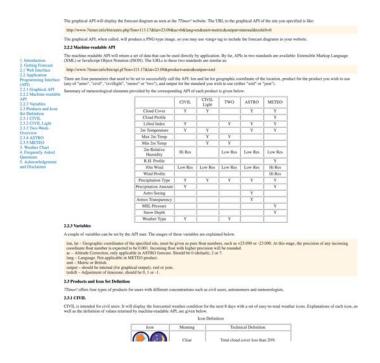
Before you begin, consider taking a step back to plan your steps. Properly planning your project, or scoping, will greatly benefit you; scoping creates structure while requiring you to think through your entire project before you begin. You should start by stating the goals for your project, then gathering the data, and considering the analytical steps required. A proper project scope can be a great road map for your project, but keep in mind that some analyses you start may become dead ends which will require you to adjust your plan.



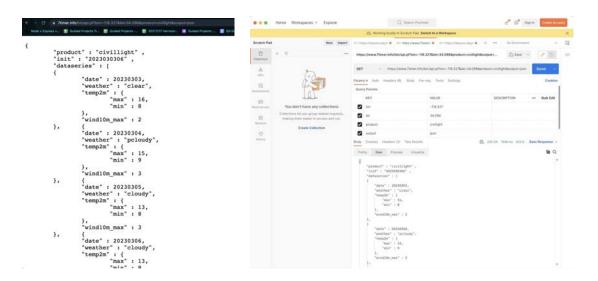
Task 1 - Explore the Weather Forecast API

The most important step is to explore the API service. What type of parameters do you want to pass along? What is the format of the results will you get back?

In this project, you will be using the <u>7Timer!</u> API. It is free and does not require any API keys to work. Follow the link and see the different levels of services provided. You will probably be more interested in the Civil types.



You can use either a web browser (or optionally use an application like <u>Postman</u>) to send requests and examine the results.



Task 2 - Apply HTML and CSS to the User Interface

Decide what input you need to gather from users to send to the external service. Do you want to use a dropdown list or find some other way to pass the latitude and longitude of a city along? (A CSV file with latitudes and longitudes of major European cities is included in the starter files).

Design the web page and user interface to accept user input and display results that you get back. If you are well versed in CSS frameworks such as Bootstrap. You may opt to use it.

Also, one of the important concepts of web design is to make your page "responsive", meaning adapting to different device screen sizes so keep that in mind.

Hints

- If you are using Visual Studio Code, the Live Server extension allows you to see the effects of your code in real time.
- Most modern web browsers (including Google Chrome) have console displays that you can
 use to debug CSS and JavaScript.
- Be sure to use images that you own or are royalty-free. Using graphics of unknown origin can lead to legal actions and undermine your credibility as a designer or developer. There are a lot of websites where you can find these. For example: Pixabay. You can also search for what you need by adding "royalty free" after the subject in any search engine. Some sites require attribution to use their images. Make sure you follow the instructions.

More Resources:

- Royalty-free graphics: Pixabay
- Free icons (requires attribution): Flat Icons
- Build Fast, Responsive sites with Bootstrap
- Debugging with the Chrome Browser

Task 3 - Write Code to Create and Send API Requests

Write the JavaScript code that builds an HTTP request and sends it to the 7Time! Server. It is recommended that you request data in JSON format since it is easier to work with JavaScript than XML.

If you are well versed in other JavaScript frameworks such as JQuery. You may opt to use it instead of plain JavaScript. Focus on connecting to the external server and getting results back in this step. Don't worry about formatting the results. Simply print it in the browser console or display the raw data on your page.

Hint

- Most modern web browsers (including Google Chrome) have console displays that you can
 use to debug CSS and JavaScript.
- Remember that HTTP API is asynchronous, meaning you will not be getting the results back right away.

Resources

- Build Fast, Responsive sites with Bootstrap
- Debugging with the Chrome Browser

Task 4 - Process Data and Generate HTML Results

Now it is time to generate the HTML and CSS code that you need to present the results in a way that the users can read. Write the necessary JavaScript to decode the data (JSON or XML) that you get back. Don't forget to handle any possible errors too.

Hint

Most modern web browsers (including Google Chrome) have console displays that you can use to debug CSS and JavaScript.

More Resources:

- Build Fast, Responsive sites with Bootstrap
- Debugging with the Chrome Browser

Task 6 - Use Fine-Tuning to Test Functionality

Thoroughly test your work and fine-tune any formatting or functionality issues. Make sure all external links work and all your JavaScript work as intended and free of errors. If possible, having someone else test your work might reveal bugs that you won't find otherwise.

Finally, we can wrap up the project. You can write a conclusion about your process and any key findings.

Hint

The main components that you will want to include:

- What did you learn throughout the process?
- Are the results what you expected?
- What are the key findings and takeaways?