

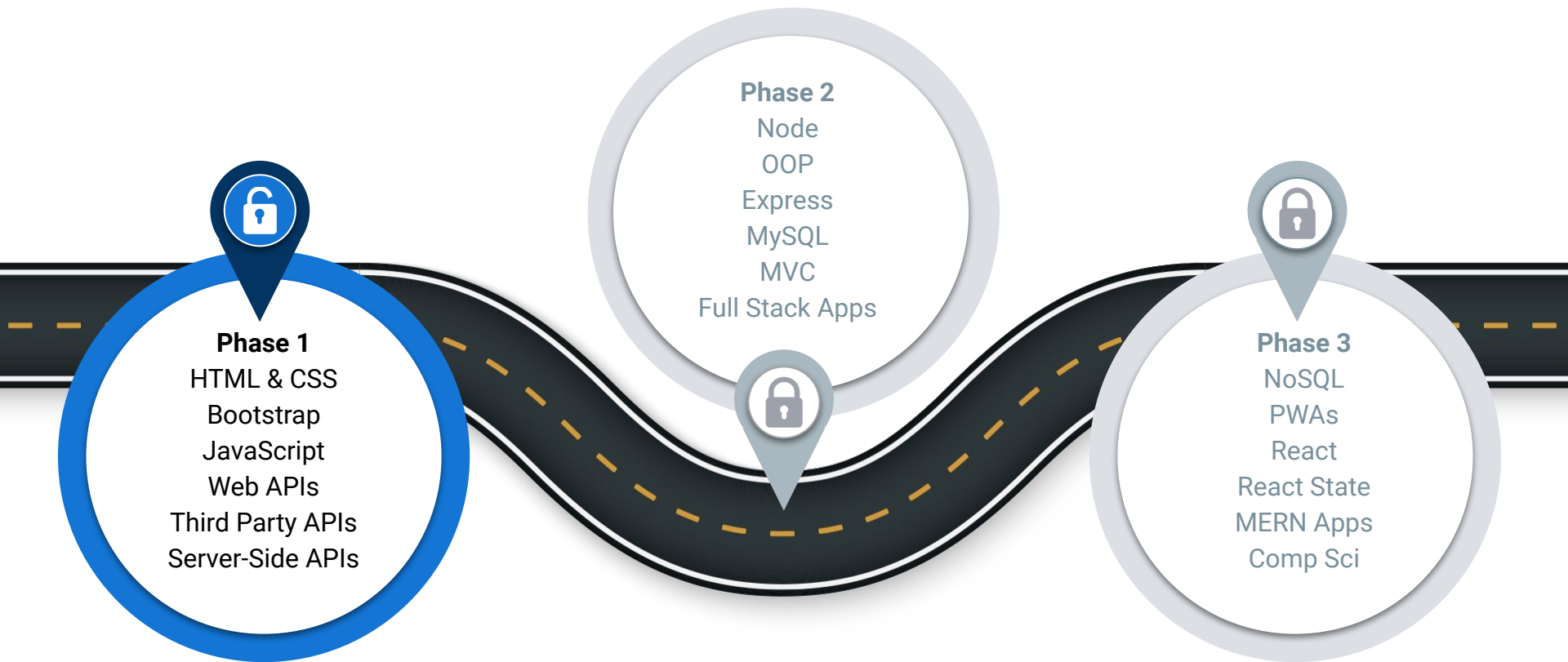
Coding Boot Camp

Module 06



# The Big Picture

---



# Boot Camp Pointers

---

Before we get into this content, remember that you have a lot of different support systems available to you:



# This Week: Server-Side APIs

---

By the end of this week, you will learn how to:



Explain the difference between a client-side API and a server-side API



Explain the client-server model and request-response pattern



Explain and implement the differences between HTTP GET requests using XMLHttpRequest, jQuery AJAX, and the fetch API



Explain HTTP response codes and handle response metadata with fetch API



Parse JSON to dynamically generate HTML



Explain the benefits and challenges of working with asynchronous JavaScript



Explain and implement query string parameters



## This Week's Assignment

---

How will you use this week's  
content in your next assignment?



## Career Connection

---

How will you use this week's  
content in your career?

# Tips for Success: Server-Side APIs

---

Keep these tips in mind:

01

Each API's documentation is going to be different. **Read it carefully!**

02

Don't forget to be mindful of your error messages. **They may help you uncover an unexpected bug.**

03

Googling error messages can lead you to what other developers have done to resolve issues!

The background of the slide is a dark charcoal gray, featuring a series of parallel diagonal lines that create a sense of depth and movement. The lines are slightly lighter in color than the background, giving the impression of a 3D effect, like a series of overlapping planes or a stylized architectural pattern.

# Let's Code!

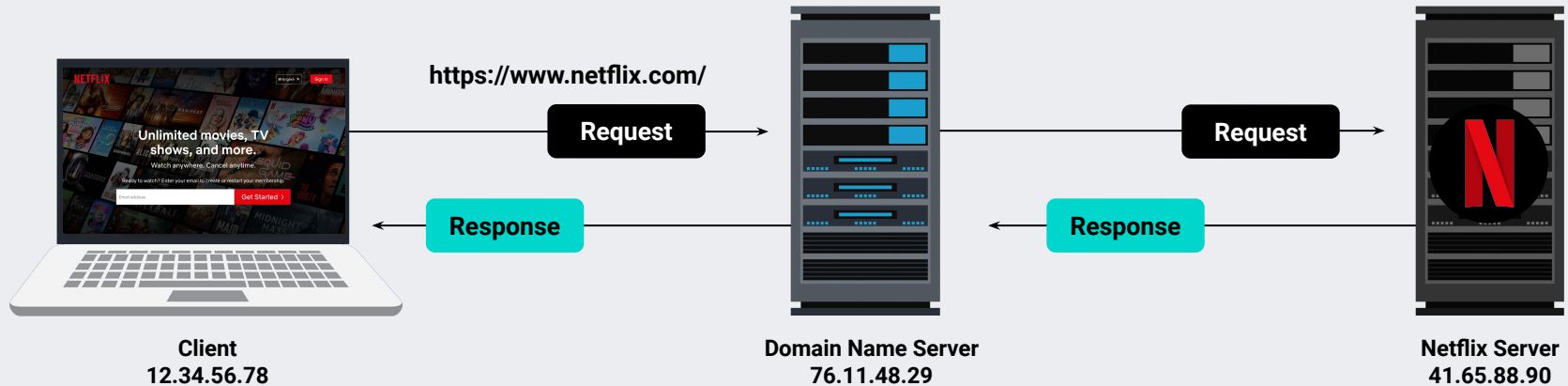




**Where do web applications live?**

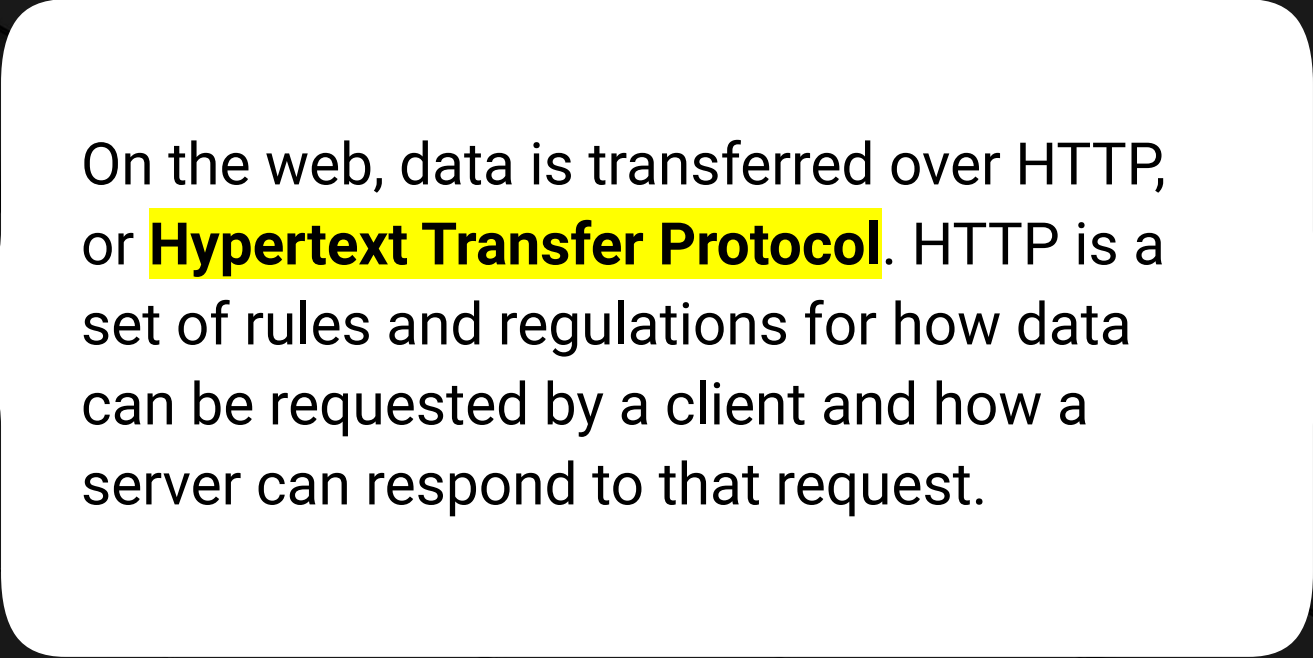
# Web Applications Live on Servers

Web servers are typically nothing more than specialized computers running software with the specific task of waiting for an internet request to come in and ask for data in return.





**How are these requests made?**



On the web, data is transferred over HTTP, or **Hypertext Transfer Protocol**. HTTP is a set of rules and regulations for how data can be requested by a client and how a server can respond to that request.



**Across all internet-connected devices,  
we constantly make HTTP requests to  
web servers for different types of data  
constantly, like the following examples.**

# Requesting Data Over HTTP

---

Across all internet-connected devices, we constantly make HTTP requests to web servers for different types of data constantly, like the following:



Visiting deployed applications at `<username>.github.io`.



A phone or watch automatically updating the weather forecast.



Using a media streaming service.



Using HTML `<link>` and `<script>` tags to incorporate Bootstrap, jQuery, or any other third-party API into an application.



**Can we use data from other  
servers in an application?**

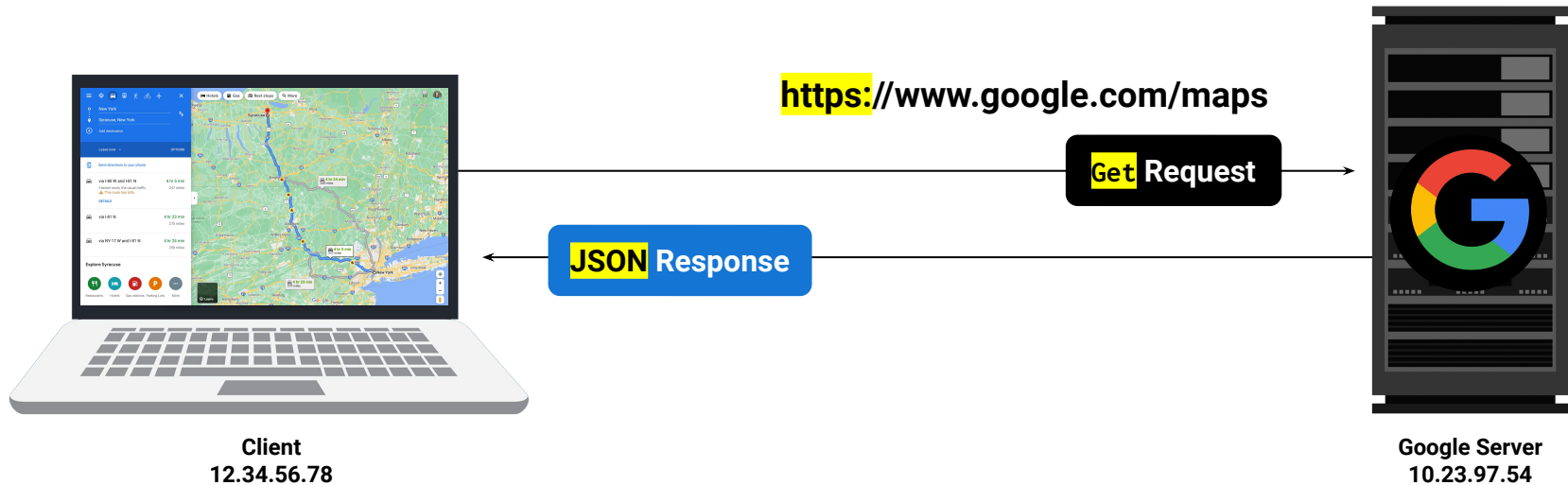


**Yes, we can! Just as we've used third-party APIs to make an application's functionality and design easier to maintain, we can use specific functionality to request data over HTTP and use that data in an application.**



# JSON (JavaScript Object Notation)

This data usually comes in the form of a special type of JavaScript object known as JSON (JavaScript Object Notation).



# JSON (JavaScript Object Notation)

---

With this data, we can do any of the following in an application:



Retrieve weather data to display in an application.



Use Google Maps to help create a trip itinerary.



Manage Spotify or YouTube playlists.



Control lights, alarms, and other devices.



And much, much more!



**How can we learn to use and  
implement these types of APIs?**

# How to Learn Server-Side APIs

---

Like other APIs we've used in the past, the implementation of server-side APIs depends on what solution that API provides. Some are very simple, while others are complex and powerful, so it's up to us to determine which parts to use.



# How to Learn Server-Side APIs

---

You can try the following strategies to learn more about specific APIs:



Read the official documentation and practice with the provided examples.



Reverse-engineer finished code to see how something was accomplished.



Build something from scratch and debug it using the Chrome DevTools.



Ask questions!



# Instructor Demonstration

---

## Mini Project