**Front and Back**

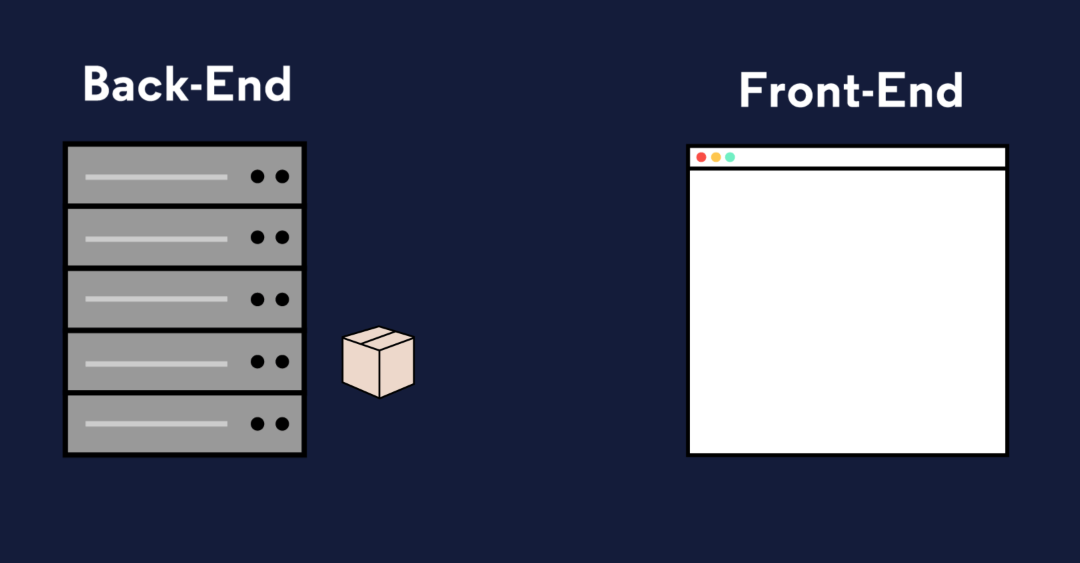
In this lesson, we’ll explain what makes up the back-end of a web application or website. The back-end can feel very abstract, but it becomes clearer when we explain it in terms of the front-end! To oversimplify a bit, the front-end is the parts of a webpage that a visitor can interact with and see.

Various tools and frameworks can be used to make a webpage, but, at its core, the front-end is composed of JavaScript, CSS, HTML, and other *static assets*, such as images or videos. Static assets are files that don’t change. When a visitor navigates to a webpage, these assets are sent to their browser.

Visiting a simple website is like ordering delivery from a restaurant: we place an order for our meal, and, once it’s delivered to us, we have it entirely in our possession. In this analogy, we can think of the front-end as everything that’s dropped off with the delivery: the containers, the utensils, and the food itself.

You’ll sometimes hear front-end development referred to as *client-side* development. Our instinct might be to think of the client as the human visitor or user of a website, but when referring to the client in web development, we’re usually referring to the non-human requester of content. In the case of visiting a website, the client is the browser, but in other circumstances, a client might be another application, a mobile device, or even a “smart” appliance!

While the front-end is the part of the website that makes it to the browser, the back-end consists of all the behind-the-scenes processes and data that make a website function and send resources to clients.



# So What is the Back-end?

When a user navigates to google.com, their request specifies the URL but not the filename for today’s [Google Doodle](https://en.wikipedia.org/wiki/Google_Doodle). The web application’s back-end will need to hold the logic for deciding which assets to send. Moreover, modern web applications often cater to the specific user rather than sending the same files to every visitor of a webpage. This is known as dynamic content.

When we eat at a restaurant, we’ll order to our tastes, make substitutions, etc; the result is a dining experience unique to us. Aside from that, there’s a lot happening behind the scenes to make a restaurant work: ingredients are ordered from suppliers, new menus are designed, and employees’ schedules are created. Similarly, to make a web application that runs smoothly, the back-end is doing a lot more than simply sending assets to browsers.

The collection of programming logic required to deliver dynamic content to a client, manage security, process payments, and myriad other tasks is sometimes known as the “application” or application server. The application server can be responsible for anything from sending an email confirmation after a purchase to running the complicated algorithms a search engine uses to give us meaningful results.

# Storing Data

You’ve probably heard that data is a big deal. By some measures, 90% of the world’s data has been generated in just the past two years! From a stored credit card number on an e-commerce site to the timestamp when you hit pause on Netflix, modern web applications collect a lot of data. For that data to be useful, it has to be organized and stored somewhere.

The back-ends of modern web applications include some sort of database, often more than one. Databases are collections of information. There are many different databases, but we can divide them into two types: [relational databases](https://www.codecademy.com/articles/what-is-rdbms-sql) and [non-relational databases (also known as NoSQL databases)](https://en.wikipedia.org/wiki/NoSQL). Whereas relational databases store information in tables with columns and rows, non-relational databases might use other systems such as key-value pairs or a document storage model. SQL, **S**tructured **Q**uery **L**anguage, is a programming language for accessing and changing data stored in relational databases. Popular relational databases include [MySQL](https://www.mysql.com/) and [PostgreSQL](https://www.postgresql.org/) while popular NoSQL databases include [MongoDB](https://www.mongodb.com/) and [Redis](https://redis.io/).

In addition to the database itself, the back-end needs a way to programmatically access, change, and analyze the data stored there.

