API Basics

**API Basics**

API Stands for Application Programming Interface.

An API's essential function is to enable two applications to talk to each other.

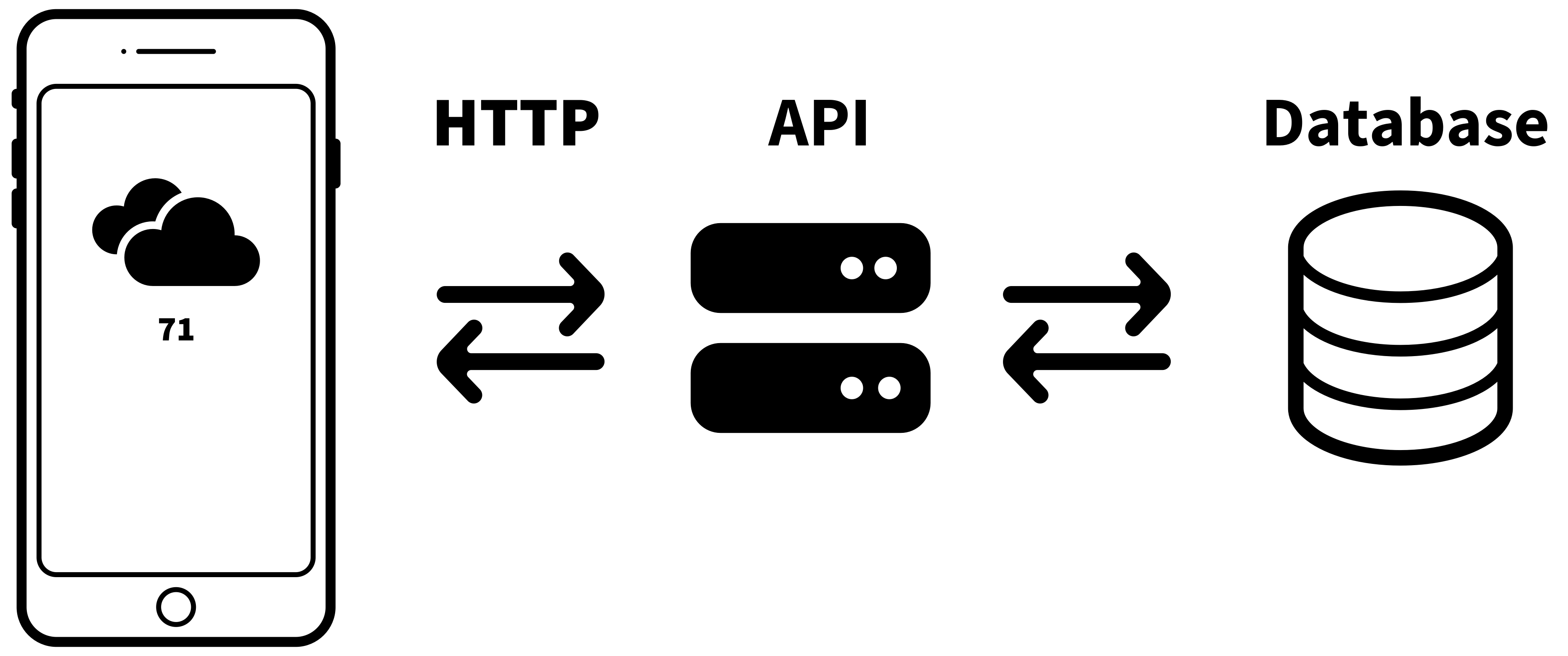
An API is like a waiter in a restaurant who takes your order and communicates it to the kitchen. The kitchen then prepares your food and the waiter brings it out when its ready.

APIs follow a pattern of Request and Response. In the restaurant example, you make a request for some food and the kitchen generally responds by giving you what you asked for.

**How APIs Work**

The diagram here gives a basic overview of how a weather API might work on your phone.

An HTTP request is sent from your mobile device to a weather API. The API fetches the data and sends it back to the mobile app (usually in JSON format). The application will then parse the data and display weather information in a user friendly format.



**API Protocols**

There are many different types of API Protocols. Each has its own advantages and disadvantages. We won't dive into the differences here, but the following is a list of the most well-known protocols:

* REST (Representational State Transfer)
* RPC (Remote Procedure Calls)
* SOAP (Simple Object Access Protocol)
* GraphQL

Many of the APIs you will encounter will use REST protocol. For More On this topic, you can visit: https://www.codecademy.com/article/what-is-rest

**API Contexts**

* Open Public APIs
  + These are APIs that anyone can access.
* Paid Public APIs
  + These are APIs that customers pay for as a service.
* Partner APIs
  + These APIs are used to pass information between two partner companies.
* Internal APIs
  + These APIs are developed purely for passing data between applications within a company.

**API Endpoints**

All APIs utilize endpoints to give users access to resources provided by the API.

You can think of Endpoints like aisles in a grocery store. Each aisle contains a sign indicating the goods you can access down that aisle.

Similarly, Endpoints are URL paths with different name endings. Each path has defined data you can access.

Base URL - This is the root address of the API. For example, https://api.example.com/.

Endpoint Path - This is appended to the Base URL to access specific resources.

For example, /users, /products, /orders.

A full endpoint URL might look like this: https://api.example.com/users - {Base URL}/{end point}

**Request Methods (Verbs)**

You will interact with APIs by making requests. There are a few different kinds of requests that you can make. Each request method aligns with a CRUD Operation.

* POST (Create)
* GET (Read)
* PUT (Update/Replace)
* PATCH (Update/Modify)
* DELETE (Delete)

**Response Codes**

When working with APIs you will frequently run into response status codes.

Response codes fall into 5 main categories:

* 100s - Information
* 200s - Success
* 300s - Redirection
* 400s - Client Error
* 500s - Server Error

To see a full list of Response Codes visit: <https://en.wikipedia.org/wiki/List_of_HTTP_status_codes>

**Parameters**

Path Parameters - These are part of the URL path and are used to identify specific resources.

For example, in /users/1, 1 is a path parameter that identifies a specific user.

Query Parameters - These are appended to the URL and provide additional information for the request.

For example, /users?role=admin filters users by the admin role. The text after ‘?’ is query parameter.

Headers - These provide metadata about the request, such as authentication tokens, content type, etc.

Body - For methods like POST and PUT, the body contains the data to be sent to the server, usually in JSON format.

**Endpoints**

GET /books - Retrieve a list of all books.

GET /books/{id} - Retrieve details of a specific book by its ID.

POST /books - Add a new book to the library.

PUT /books/{id} - Update details of a specific book.

DELETE /books/{id} - Remove a specific book from the library.