Web Development Foundations

Chitram Dasgupta

July 8, 2025

Contents

1 What is a Web Application?

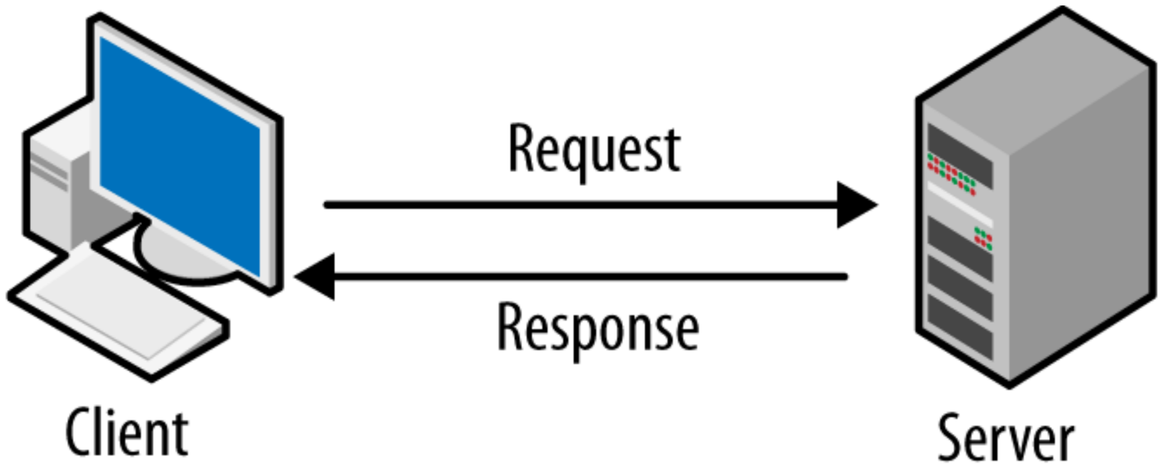
A web application is a software program that runs in a web browser and allows users to interact with it over the internet.

- Unlike static websites, web apps can accept user input and change dynamically.
- Examples: Gmail, Twitter, online banking apps.
- A web app typically has:
  - A **frontend** (client side): What the user sees and interacts with.
  - A **backend** (server side): Handles logic, database access, and responses.

2 Client-Server Architecture

Web applications follow the **client-server** model.

- The **client** is usually a web browser.
- The **server** is a computer that listens for requests and sends back responses.



- Communication happens over HTTP(S).
- The server may also talk to a database to fetch/store data.

3 Basics of Protocols

3.1 What is a protocol?

A protocol is a set of rules for how two systems communicate.

- We need protocols so that devices and software can understand each other.
- Just like human language needs grammar, computers need protocols.

3.2 Examples of Web Protocols

- HTTP/HTTPS: Transfer of data over the web.
- TCP/IP: Lower-level transport protocols.
- DNS: Domain name to IP address translation.

4 Basics of APIs

API stands for **Application Programming Interface**.

- APIs let two systems talk to each other.
- Web APIs are usually exposed over HTTP.

4.1 REST APIs (Representational State Transfer)

- Based on resources (e.g., `/users`, `/recipes`)
- Uses HTTP methods:
  - `GET` : retrieve data
  - `POST` : create data
  - `PUT` or `PATCH` : update data
  - `DELETE` : delete data
- Uses HTTP status codes to indicate the result of a request:
  - `200 OK`: The request was successful.
  - `201 Created`: A resource was successfully created.
  - `400 Bad Request`: The request was malformed or invalid.
  - `401 Unauthorized`: Authentication is required or failed.
  - `403 Forbidden`: The user is authenticated but not allowed.
  - `404 Not Found`: The requested resource doesn't exist.
  - `500 Internal Server Error`: Something went wrong on the server.

4.2 GraphQL

- Alternative to REST
- You send a query describing the data you want
- The server returns *exactly* that data (no more, no less)

5 Example: Recipe App API (REST)

Let's say we're building an app where users can share cooking recipes.

5.1 Resources

- `User`: Can sign up, log in
- `Recipe`: Belongs to a user
- `Ingredient`: Belongs to a recipe

5.2 Endpoints

- `POST /signup` : Register a user
- `POST /login` : Authenticate and return a token
- `GET /recipes` : List all recipes
- `POST /recipes` : Create a recipe (requires authentication)
- `GET /recipes/:id` : Get a specific recipe
- `POST /recipes/:id/ingredients` : Add an ingredient

5.3 Example JSON Response

```json
{
  "id": 1,
  "title": "Pasta",
  "ingredients": [
    { "name": "Pasta", "quantity": "200g" },
    { "name": "Tomatoes", "quantity": "2" }
  ],
  "author": "chitramdasgupta@example.com"
}
```

6 Motivation for HTML, CSS, and JS

6.1 HTML (HyperText Markup Language)

- Defines **structure** of the webpage
- E.g., headings, paragraphs, links, forms

6.2 CSS (Cascading Style Sheets)

- Defines **presentation and style**
- E.g., colors, fonts, layout, spacing

6.3 JavaScript

- Adds **interactivity**
- E.g., clicking a button shows/hides content, form validation, etc.