mukul rajpoot

WEB

ARCHITECTURE BASICS

mukul rajpoot

WHAT IS WEB ARCHITECTURE?

Web architecture is the planning and design of the technical, functional and visual components of a website - before it is designed, developed and deployed.

Web architecture involves multiple components like a database, message queue, cache, user interface, etc., all running in conjunction to form an Web Application.

mukul rajpoot

CLIENT-SERVER ARCHITECTURE

Client-server architecture is the fundamental building block of the web.

The architecture works on a request-response model. The client sends the request to the server for information and the server responds with it.

Every website you browse, be it a WordPress blog, an application like Facebook, Twitter, or your banking app, is built on the client-server architecture.

mukul rajpoot

CLIENTS

The client holds our user interface. It's written in HTML, JavaScript, CSS and is responsible for the look and feel of the application.

It can be a mobile app, a desktop or a tablet like an iPad. It can also be a web-based console, running commands to interact with the backend server.

There are a plethora of technologies that can be leveraged for writing the front-end like React, Flutter, Angular, Vue etc...

There are primarily two types of clients:

- THIN CLIENT: A thin client is a client that holds just the user interface of the application. It contains no business logic of any sort. For every action, the client sends a request to the backend server

- THICK CLIENT: The thick client holds all or some part of the business logic. For instance online games, utility apps etc...

mukul rajpoot

SERVERS

The primary task of a web server is to receive the requests from the client and provide the response after executing the business logic based on the request parameters received from the client.

There are a plethora of technologies that can be leveraged for writing the back-end Server like Node.js, Python, Java etc...

All the components of a web application need a server to run, be it a database, a message queue, a cache, or any other component. In modern application development, even the user interface is hosted separately on a dedicated server.

Often the developers use a server to render the user interface on the backend and then send the generated data to the client. This technique is known as server-side rendering.

mukul rajpoot

COMMUNICATION BETWEEN THE CLIENT AND THE SERVER

The client and the server have a request-response model. The client sends the request and the server responds with the data.
If there is no request, there is no response.

The entire communication happens over the HTTP protocol. It is the protocol for data exchange over the World Wide Web. HTTP protocol is a request-response protocol that defines how information is transmitted across the web.

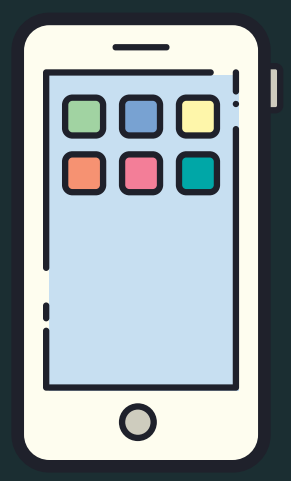
Every client has to hit a REST endpoint to fetch the data from the backend.

The backend application code has a REST-API implemented. This acts as an interface to the outside world requests.

mukul rajpoot

COMMUNICATION BETWEEN THE CLIENT AND THE SERVER

Client



Request

- GET
- POST
- PUT
- DELETE

Server

Response

Client

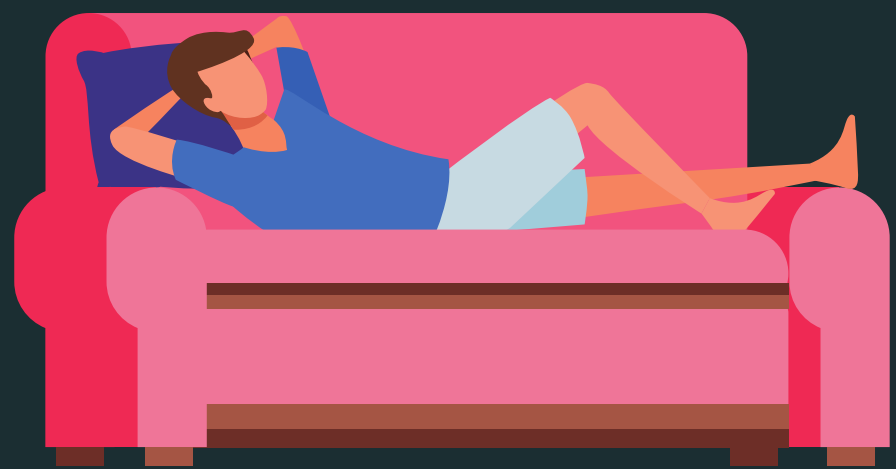
REST API

mukul rajpoot

REST stands for Representational State Transfer.

A REST API takes advantage of the HTTP methodologies to establish communication between the client and the server. REST also enables servers to cache the response that improves the application’s performance.

The communication between the client and the server is a stateless process. It means there is no information or memory carried over from the previous communications. So, every time a client interacts with the backend, the client has to send the authentication information to it as well. This enables the backend to figure out whether the client is authorized to access the data or not.



mukul rajpoot

REST API ENDPOINT

An API/REST/Backend endpoint means the URL of the service that the client could hit.

For instance, https://example.com/users/{username} is a backend endpoint for fetching the user details of a particular user from the service.

The REST-based service will expose this URL to all its clients to fetch the user details using the above stated URL.

With the availability of the endpoints, the backend service does not have to worry about the client implementation. It just calls out to its multiple clients and says, “Hey Folks! Here is the URL address of the resource you need. Hit it when you need it. Any client with the required authorization to access a resource can access it.

mukul rajpoot

APPLICATION DEVELOPMENT BEFORE THE REST API

Before the REST-based API interfaces became mainstream in the industry, we often tightly coupled the backend code with the client. Java Server Pages (JSP) is one example of this.

We would always put business logic in the JSP tags. This made code refactoring and adding new features difficult because the business logic spread across different layers.

In today’s application development landscape, there is hardly any online service implemented without a REST API.

mukul rajpoot

BYE