

Who am I?

Humera Tariq

PhD, MS, MCS (Computer Science), B.E (Electrical)

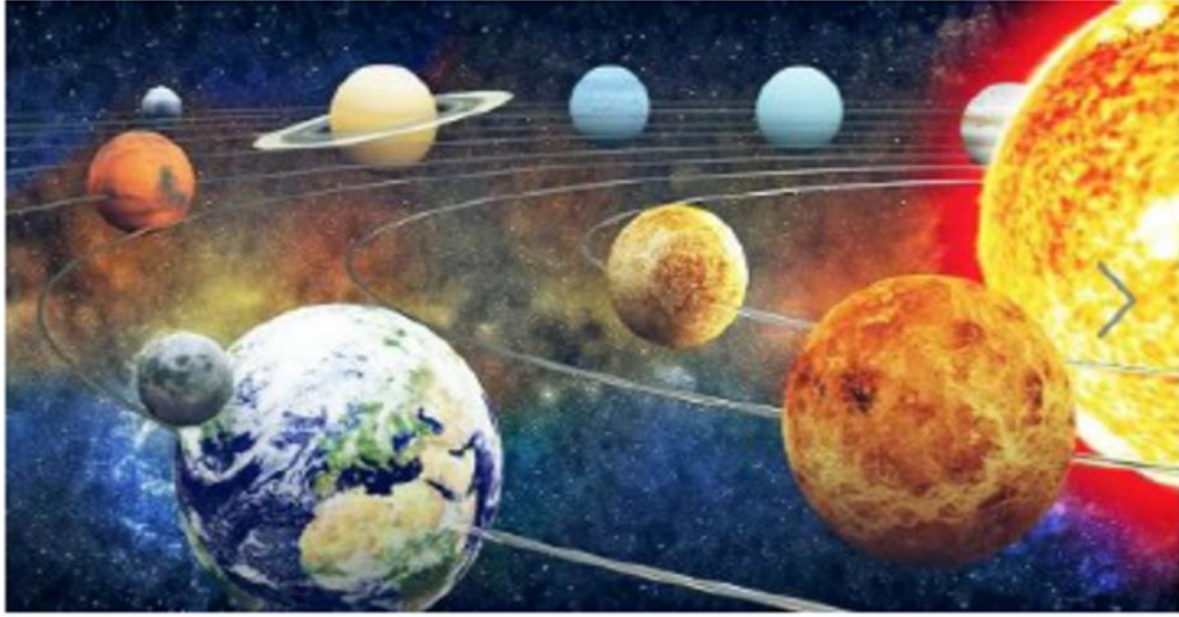
Postdoc (Medical Image Processing, Deep Neural Networks)

Email: humera@uok.edu.pk

Web: <https://humera.pk/>

Discord: <https://discord.gg/xeJ68vh9>

Starting in the name of Allah,



*the most beneficial,
the most merciful.*

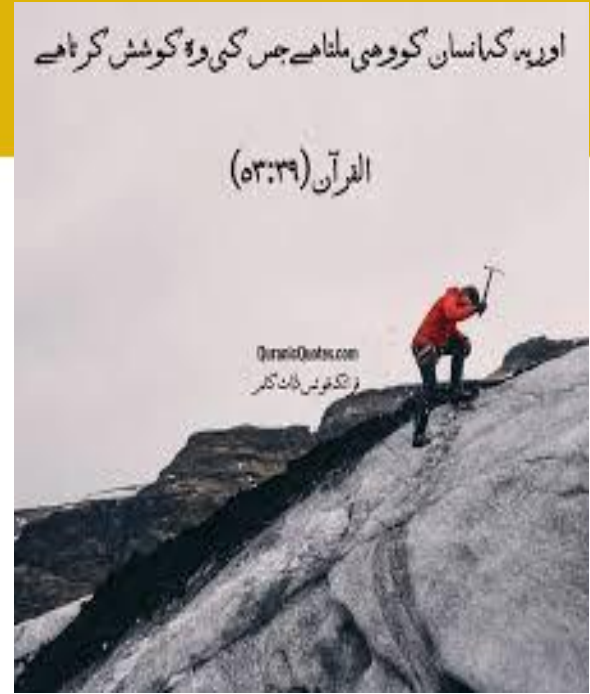
أَمْرٌ لِلْإِنْسَانِ مَا تَبَيَّنَ

کیا انسان کو ہر وہ چیز حاصل ہے جس کی اس نے تمنا کی؟



UNIVERSITY OF
KARACHI

And there is not for man except that [good] for which he strives.





Week 06

Internet Application Development

Copyright © 2025, Humera Tariq

Department of Computer Science (DCS/UBIT)

University of Karachi

January 2025



test included a survey question asking how many hours students spent studying for the test. The scatter plot and trend line below show the relationship between how many hours students spent studying and their score on the test.

The line fitted to model the data has a slope of 15.

What is the best interpretation of this slope?



The model predicts that students who scored 1



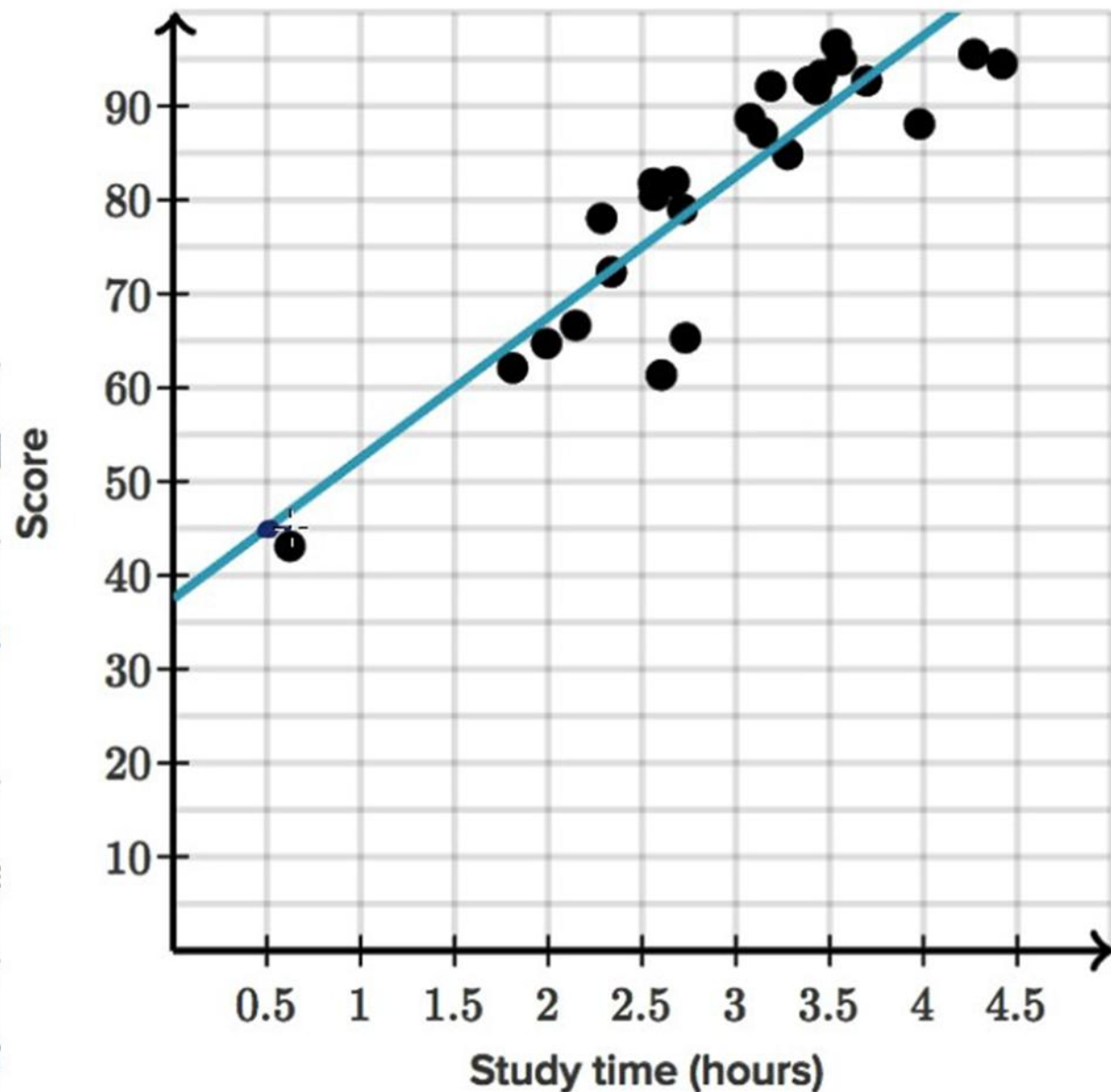
The model predicts that students who didn't study points.

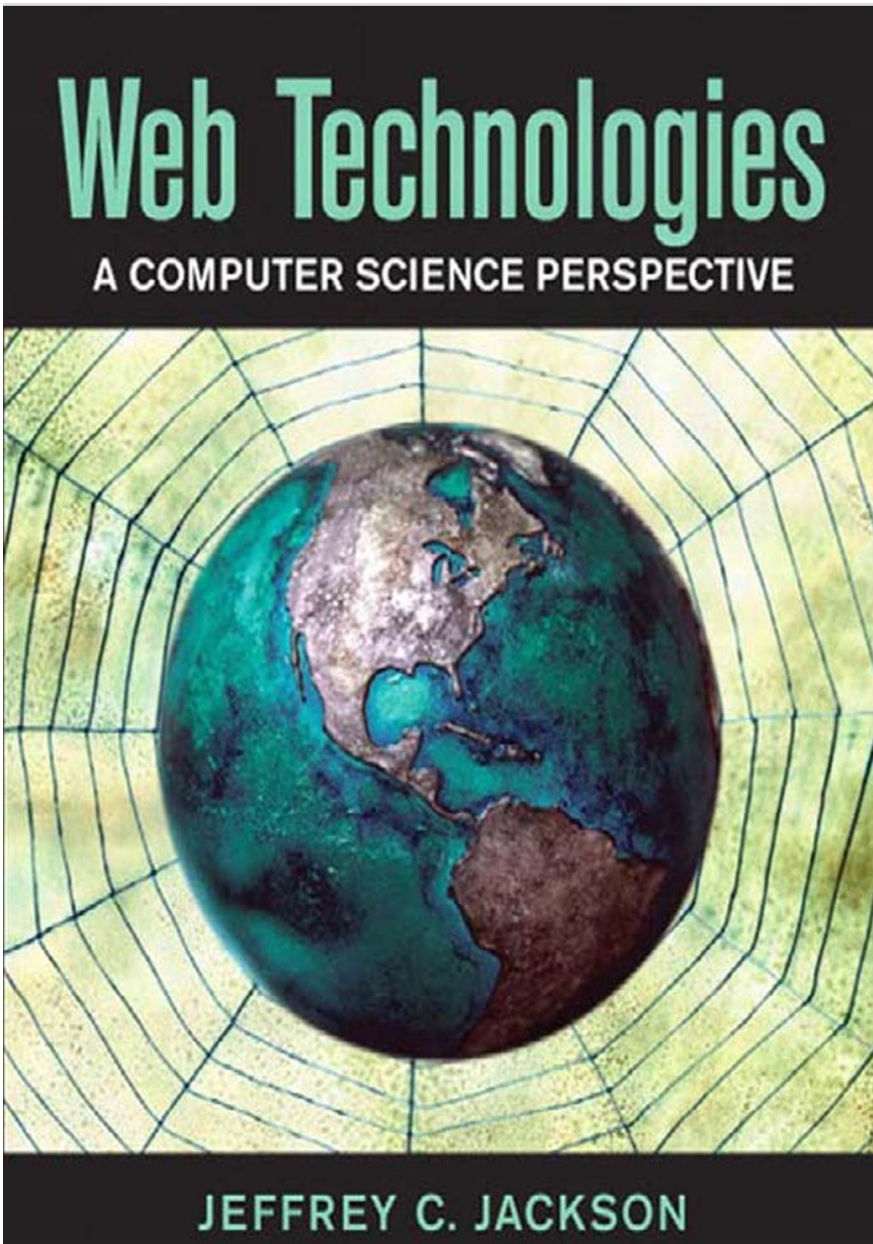


The model predicts that the score will increase



The model predicts that the study time will increase





CHAPTER 1	Web Essentials: Clients, Servers, and Communication	1
CHAPTER 2	Markup Languages: XHTML 1.0	56
CHAPTER 3	Style Sheets: CSS	121
CHAPTER 4	Client-Side Programming: The JavaScript™ Language	192

- Minimal app (4 ways)
- Modify submitted code to practice max JS concepts (Group -> individual)

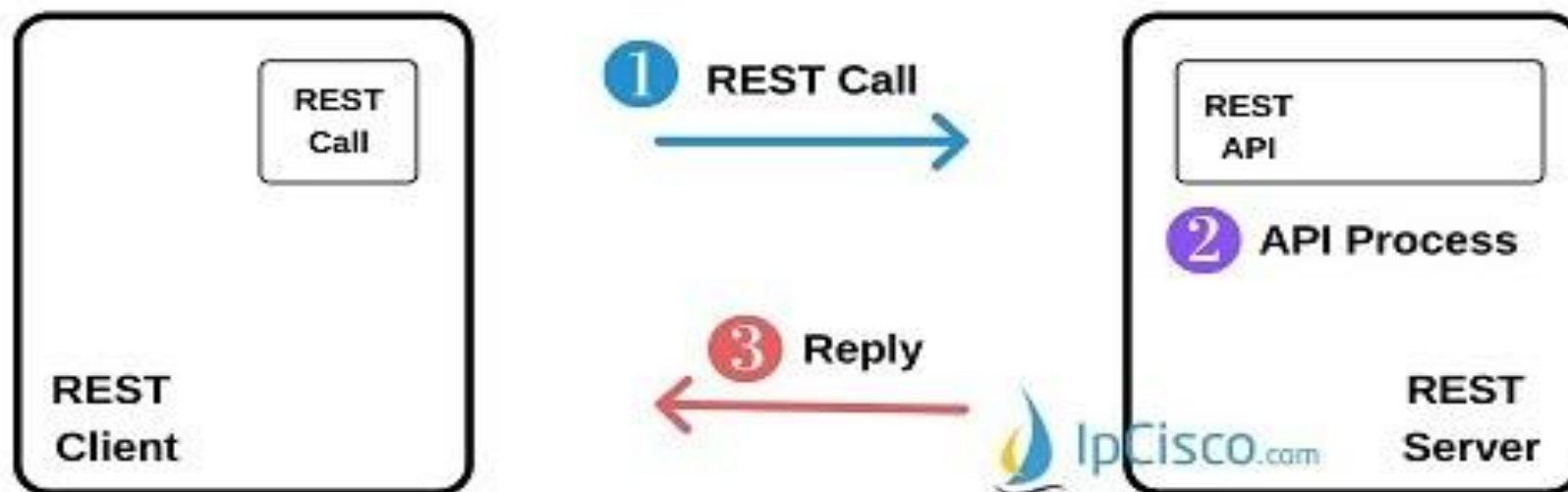
Express-Node Back-end

Building a simple back-end application with
Express, Node, and MongoDB

Need for back-end? Who accept HTTP requests?

- ✓ Back-end will connect to a _____ , get some results, and do some processing with those results.
- ✓ Back-end will expose _____ API that front-end will use to interact with the DB .
- ✓ _____ will accept HTTP requests from _____ app and use _____ operations to interact with the DB.
- ✓ Express is a minimal and flexible web application _____ for Node.js. It is designed for building web applications and _____ .

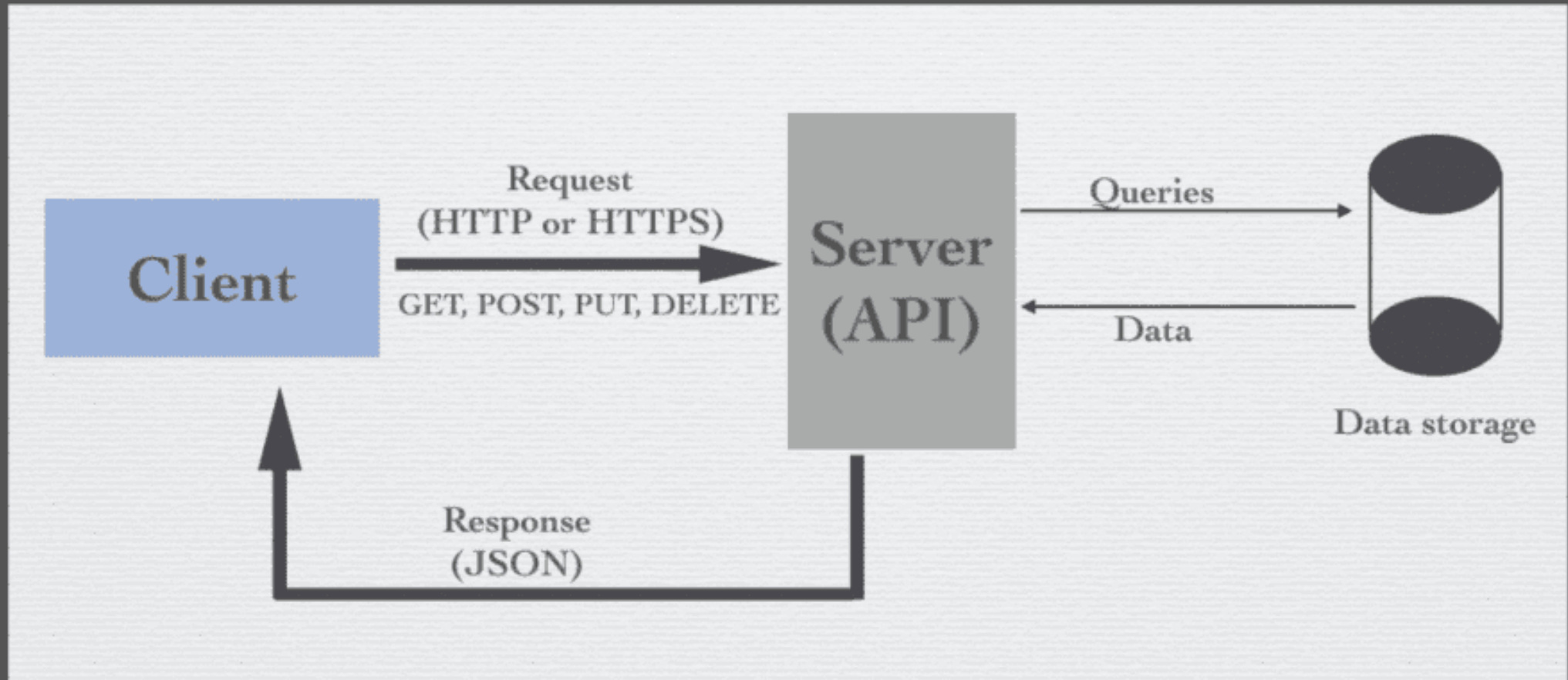
REST API Client/Server Operation



REST API Client/Server Process

- 1) REST Client starts a **REST Call**.
- 2) REST Server receives and **API process** starts.
- 3) REST Server **Replies** to the REST Call.

What is RESTful API?



What is REST ?

REST stands for _____ .

It is an _____ style for designing networked applications that relies on _____ (Hint: stateless/stateful) communication and standard _____ methods like **GET, POST, PUT, DELETE, PATCH**, etc. **RESTful APIs** follow a set of principles that make applications scalable, flexible, and easy to use.

Five (5) RESTful APIs Design Principals

- ✓ : Each **request** from a client contains all the information needed for processing, and the **server does not store client state**.
- ✓ **Architecture**: The client and server are independent; they communicate via **HTTP requests and responses**.
- ✓ **Uniform** : Consistent resource identification (**URLs**), resource manipulation via **representations (JSON, XML)**, and standard **status codes**.
- ✓ : Responses can be cached to improve performance.
- ✓ **System**: The **client** does not need to know whether it is directly communicating with the **server** or an intermediary.

The concept of a **RESTful API** is an _____ style—it's a **set of design principles** and **constraints** for how web _____ **??** (hint: **app/service**) should be built—while **Express** is just one of many tools (_____) you can use to implement that design. In other words, **RESTful API design** is independent of the technology used to _____ the server. You can build a **RESTful API** using **Express**, **Django**, **Flask**, or any other framework; the key is that you follow **REST principles** such as _____, a uniform _____, and proper use of _____ methods and codes. **Express** doesn't force you to build a **RESTful API**—it merely provides an environment to implement **RESTful design if you choose to do so**.



How to distinguish between web application and web service ?



Web service vs Web application

- ✓ RESTful APIs are web services in their purest form means the service exposes an _____ (usually via HTTP and data formats like JSON) for other software to consume.
- ✓ Web applications can—and often do—use RESTful APIs as the _____ layer between the client and the server. This separation aligns with the client-server architecture, ensuring clarity and maintainability in your project's design.



Web service vs Web application



- ✓ All **web applications** have a _____ side (which behaves like a **web service**), not every web service is a complete _____.
- ✓ Name few real-world examples where you have a **web service that isn't a full web application**—meaning it **exposes functionality via APIs** without providing a complete _____ (UI).

JSON is not an inherent requirement of the web service model ?



Web service vs Web application

Payment gateways, weather APIs, geocoding APIs, and cloud storage APIs) are considered _____ because they make HTTP requests to their API _____ for data exchange without a built-in _____. They often use JSON because it's popular today, but JSON is not an inherent requirement of the web service model.



How to distinguish between web api
and function call ?

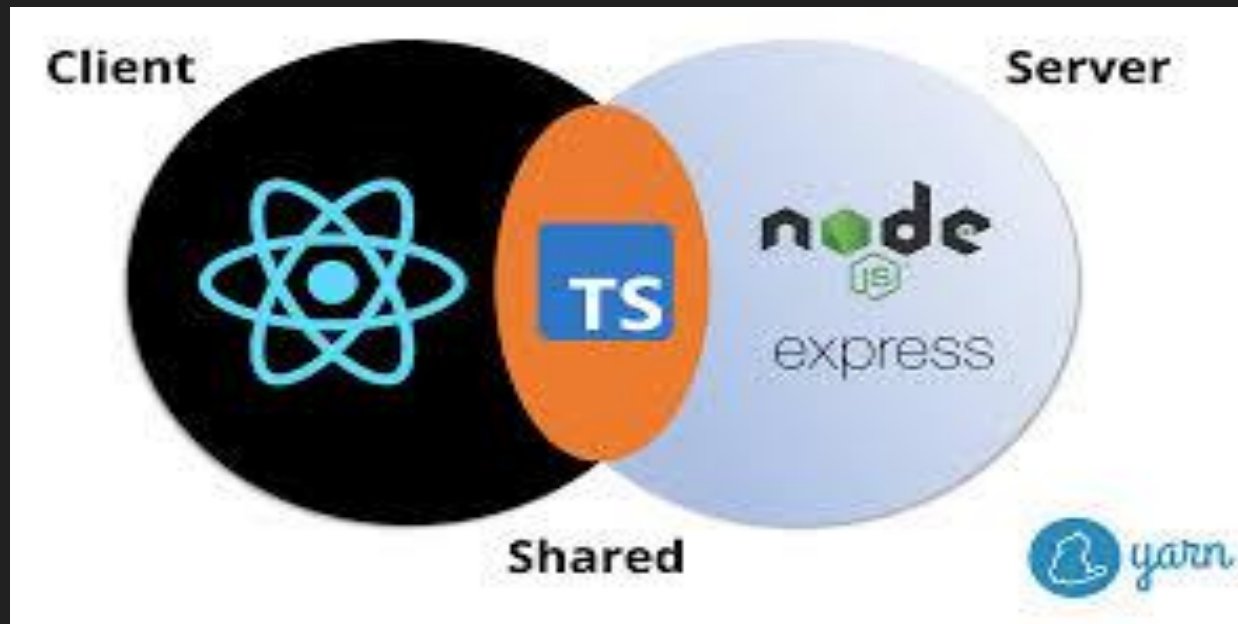
Web API vs SDK or function call

Endpoints are Fundamental: Every web API exposes endpoints (specific URLs) that represent resources or services. Clients make _____ (using methods like GET, POST, etc.) to these endpoints, and the server responds with data (commonly in JSON, XML, etc.).

In many cases, an API is a _____ without a user interface.

Additionally, the term “API” can also refer to function libraries or SDKs that expose a set of function calls. These aren’t _____ part of a networked client-server architecture but are still considered _____ because they define how different software components _____.

What we should do now !



Type script
or java
script 😊 ??

Separation Advantage

When discussing the **server side alone**, the focus is on **REST principles**—defining _____ , _____ , _____ , and ensuring _____ interactions. Later, when you **integrate the front-end**, you'll see how the **REST API** serves as the _____ bridge between the **server and the React application**. This separation lets you build and test the backend independently before connecting it with the client side.

Let's design unique REST API for a
"enhancing web development skills at
University of Karachi Department of
computer science with Humera Tariq"
service.

Let's design unique REST API for a
"enhancing web development skills at
University of Karachi Department of
computer science with Humera Tariq"
service.

Base URL

<https://api.ubit-webdev-capstone.com>

The **base URL** is a key component in our **RESTful API design**. It ensures that **both the front-end and back-end** consistently _____ and _____ **URLs**. For instance, our modern front-end (built with frameworks like **React or Angular**) will use this **base URL** to construct _____, while our **Express server** uses it to define its _____ logic. This **uniformity** is crucial for **seamless communication between client and server**.

Still thinking about Base URL!



In our **project**, the **base URL** (stored in _____ **files**) is fixed **office address** for our _____ (Hint: **server/client ?**) .

The **RESTful API** then provides the “**rooms**” or _____ inside that **building**, which the **front-end (like a React app)** _____ to retrieve data. This clear separation ensures that even as you build and deploy your **server-side logic** in a **RESTful style**, both components—**front-end and back-end**—consistently know where to send _____ (Hint: **data/request ?**) and how to _____ URLs.

Setting up back-end API and app

Setting up back-end app

```
C:\Users\humer>node --version
```

v18.18.0

```
C:\Users\humer>npm --version
```

9.8.1

```
C:\Users\humer>express --version
```

4.16.1

```
C:\Users\humer>npm install -g express-generator (optional: boilerplate code)
```

```
C:\Users\humer>nodemon -v
```

3.1.9

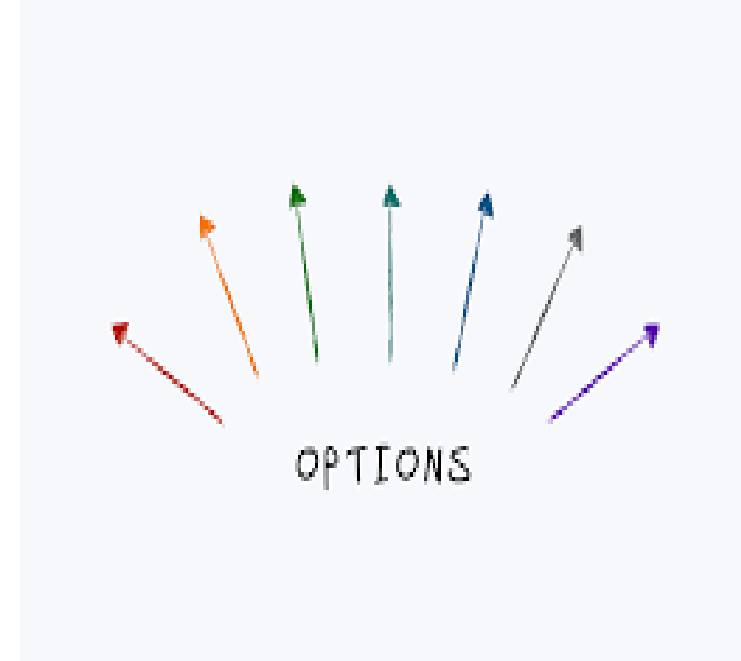
Prerequisite:

- [Install Node and Npm](#)
- [Install Mongo DB](#)

<https://nodejs.org/>

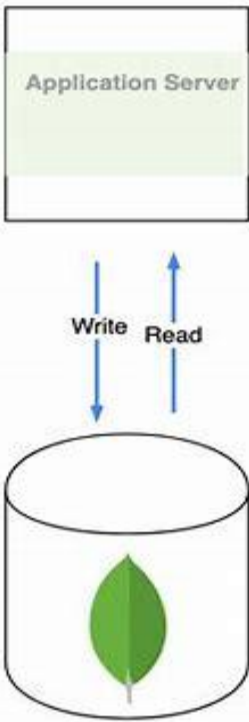
Software	Justification
Node.js	Node.js is <u>JS</u> run time environment that allows you to run JavaScript outside of the browser. It provides an environment to execute backend logic, read files, interact with databases, etc.
<u>npm</u>, <u>nodemon</u>	<u>npm install --save-dev nodemon</u> Without <u>nodemon</u> : Every time you make <u>changes</u> , you must manually restart the server
Express.js (The Web Framework for Node.js)	<ul style="list-style-type: none"> ✓ Express.js is built on top of Node.js to simplify server-side development. ✓ It provides routing, middleware support, and simplifies API handling. ✓ Without Express, you'd need to manually handle requests using Node.js' built-in http module, which is more complex. ✓ Express abstracts away low-level details, making backend development faster and cleaner.
Code Editor	VS code <u>recomended</u>
	The <u>dotenv</u> package is used to manage environment variables in a Node.js application. It allows you to store sensitive configuration data, like API keys, database URLs, or port numbers, in a .env file instead of hardcoding them in your code.
Postman or <u>cURL</u>	For API testing, optional but useful





When to Install Locally vs. Globally?

Installation Type	Command	When to Use?
Global	<code>npm install -g nodemon</code>	When you want to use a tool across all projects (e.g., Nodemon, Express Generator).
Local	<code>npm install express</code> <code>mongoose dotenv</code>	Required for every new project to ensure it has its own dependencies.



C:\Users\humer>mongod --version

'mongod' is not recognized as an internal or external command, operable program or batch file.

<https://www.mongodb.com/try/download/community>

- For **beginners or simple projects** → Suggest **MongoDB Atlas** (easier to set up, no installation).
- For **advanced development or offline access** → Suggest **local MongoDB** (better control, faster).

1. Create an Atlas Account

- Go to [MongoDB Atlas](#)
- Sign up and create a **free cluster** (M0 tier)

2. Get the Connection URI

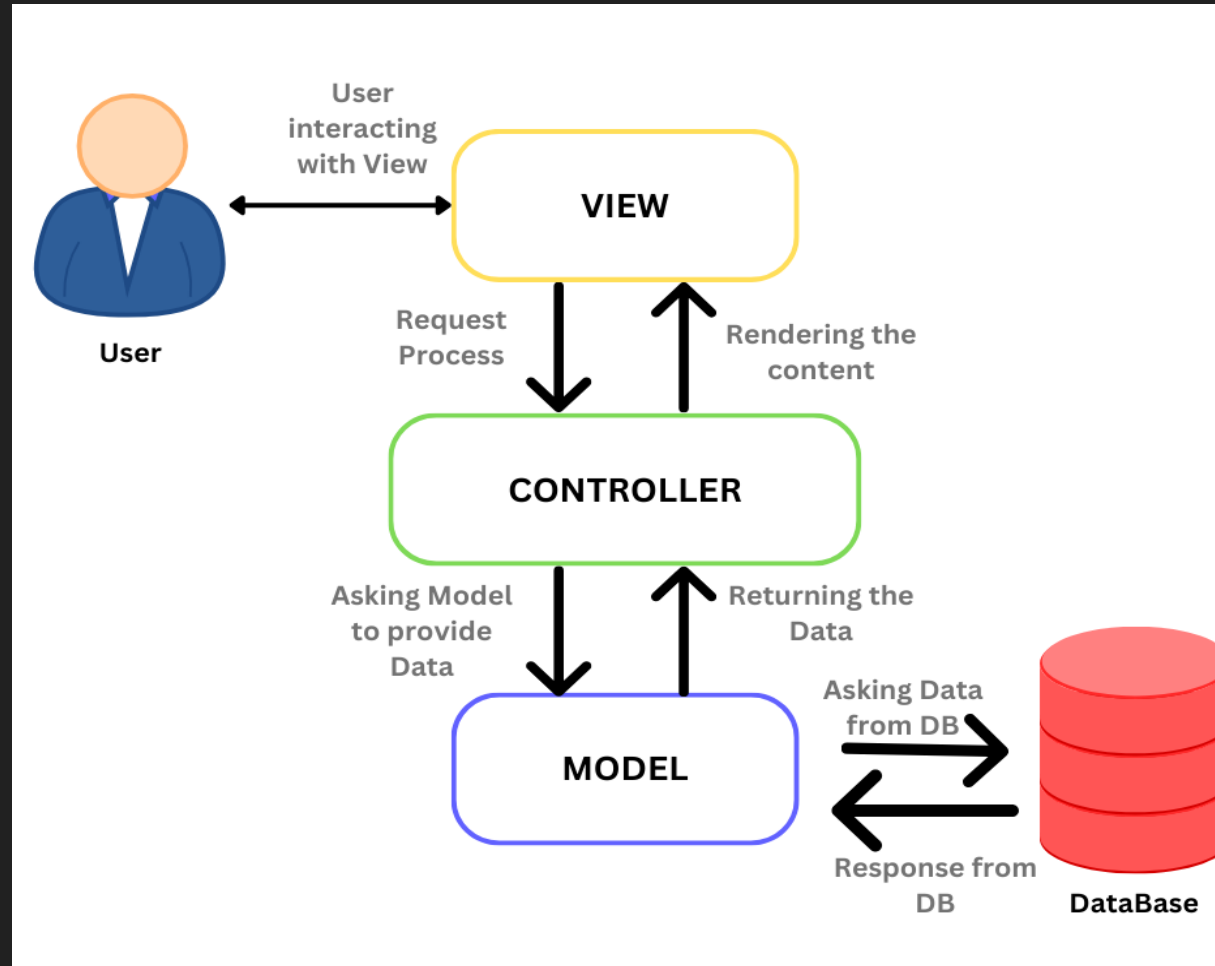
- Go to Database Deployments → Click Connect
- Choose "Connect your application"
- Copy the MongoDB connection string



3. Store Connection URI in `.env` File

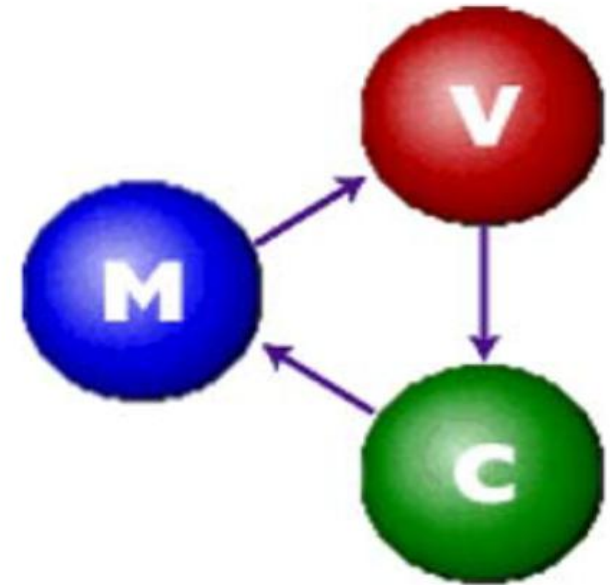
Inside your project folder, create a `.env` file:

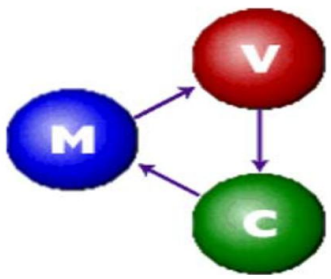
Mapping Architecture to Webapps Technology



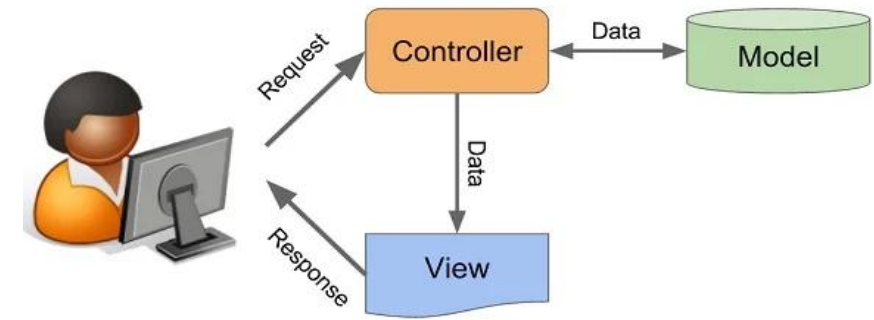
Traditional MVC architecture

Traditional MVC
architecture calls for a
visual application to be
broken up into three (3)
separate parts





Structure



4

Notify the view of a change..

5

Ask the model what has changed

2

Request changing Model's state

1

The user did an action

3

Change the view

Model

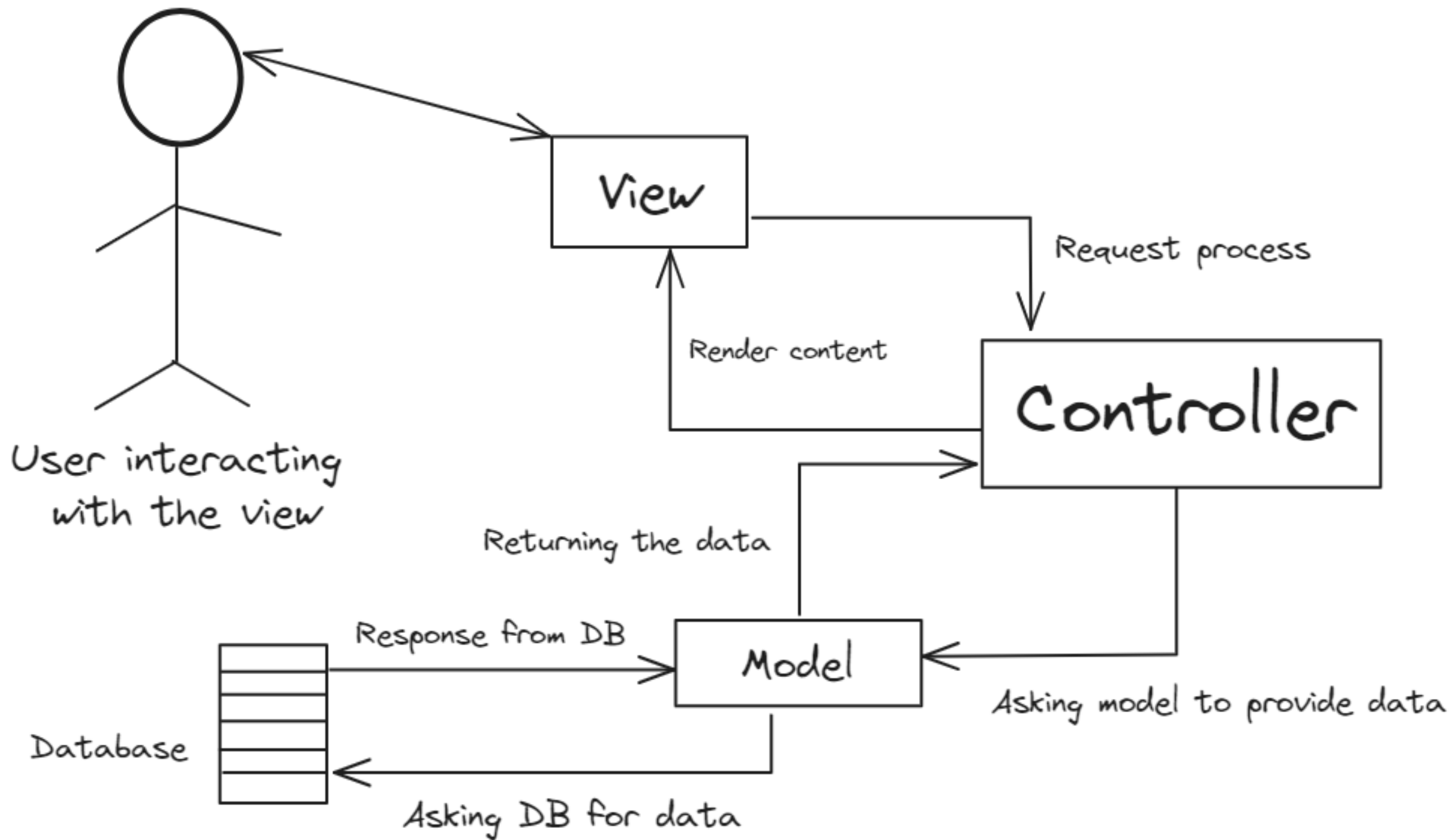
Controller

View

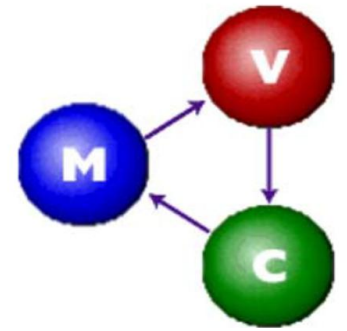
Keep the state in-sync with the data store.

Data Store

(Not part of the MVC)



Hope you are able to answer this



- ✓ A *model* represents the _____ for the application.
- ✓ The *view* is the _____ representation of that data.
- ✓ A *controller* takes _____ on the *view* and translates that to changes in the *model*.”

React as frontend and Express as backend (REST API).



```
backend-mvc/  
|— controllers/    <-- Handles business logic for each route  
|— models/         <-- Defines database schemas (M in MVC)  
|— routes/         <-- Defines REST API endpoints (acts as interface)  
|— middleware/     <-- Custom middleware (Auth, Logger, etc.)  
|— config/         <-- Database config (e.g., MongoDB connection)  
|— services/       <-- Optional: Business logic separate from controllers  
|— utils/          <-- Helper functions (e.g., hashing, token generation)  
|— public/         <-- Optional: Static files (if needed)  
|— server.js       <-- Main entry point (sets up Express)  
|— .env            <-- Environment variables (MongoDB URI, PORT)  
|— package.json    <-- Project dependencies & scripts  
|— README.md       <-- Documentation
```



🚀 The reason why `views/` is NOT included in this backend-MVC project structure is because:

- You are building a **REST API** where the backend does **not** render HTML.
- Instead, the backend sends **JSON responses** to a separate **React frontend** that handles all the UI.

Backend Type	Does it Need <code>views/</code> ?	Explanation
REST API (Backend + React Frontend)	✗ No	The backend only serves data in JSON format, while React (frontend) is responsible for rendering the UI. There is no need for the backend to generate HTML views.



- ✓ In a **traditional backend MVC setup**, the **View (V)** is responsible for rendering UI using templating engines like **EJS (_____)**, **Pug**, or **Handlebars**.
- ✓ But when using **React for the frontend**, React itself handles UI _____.
- ✓ The _____ **now only provides data via APIs (JSON responses)** instead of rendering HTML.
- ✓ In a **React + Express setup**, the **backend only serves data (Model + Controller)**, while React takes over the **View layer**.



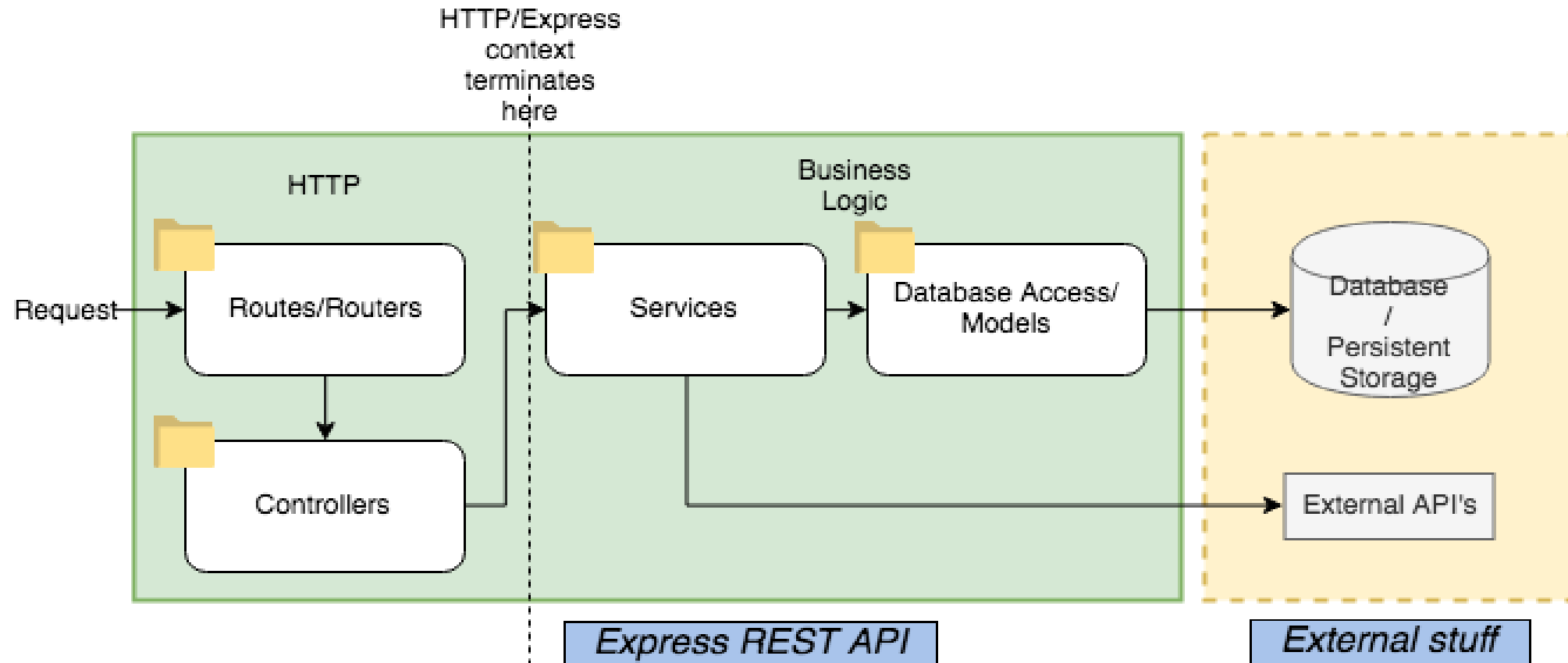
Does React (Front end app) Itself
Follow MVC?

Lets go to create a backend API in Express JS!

Best Practices for Setting Up a Backend API Project
(Node.js + Express + MongoDB Atlas)

- ✓ Install Node JS and Express JS
- ✓ Create a project folder and a server file
- ✓ Define the data model and the data array
- ✓ Create the routes and the handlers for the API
- ✓ Test the API with a tool like Postman /curl

Express as backend (REST API).



Step I

Create Project Directory & Initialize Node.js

D:\back_end> **npm init -y** # Initializes package.json with default values

EXPLORER

...

{} package.json ✕

▼ BACK_END

{} package.json

{} package.json > ...

```
1  {
2    "name": "back_end",
3    "version": "1.0.0",
4    "description": "",
5    "main": "index.js",
6    "scripts": {
7      "test": "echo \"Error: no test specified\" && exit 1"
8    },
9    "keywords": [],
10   "author": "",
11   "license": "ISC"
12 }
```

`module.exports`

Common JS

`require('./index.js')`

`export`

ES modules

`import`

VS

Which one to choose?

Common JS

This is the commonly used **module system** in [Node.js](#), where modules are imported [synchronously](#). Since modules are imported synchronously, each **task will be paused** until the module is imported entirely.

In math.js

```
function add(a, b) {  
  return a + b;  
}
```

```
function subtract(a, b) {  
  return a - b;  
}
```

```
// Exporting the functions to be used in index.js
```

```
module.exports = {  
  add,  
  subtract  
};
```

Syntax Example `module.exports`

In index.js

Syntax Example require

```
const { add, subtract } = require('./math');
```

```
console.log( add ( 5 , 3 ) );
```


```
console.log( subtract ( 10 , 4 ) );
```

THANKS
for your
ATTENTION

Any
questions



UNIVERSITY OF
KARACHI



"Don't be satisfied with stories, how things have gone with others. Unfold your own myth." ~Rumi



UNIVERSITY OF
KARACHI



Department of Compute Science (UBIT Building), Karachi, Pakistan.

1200 Acres (5.2 Km sq.)

53 Departments

19 Institutes

25000 Students



My Homeland Pakistan

