TECHNICAL QUESTIONNAIRE

1.What is asynchronous JavaScript? How is it different from synchronous JavaScript?

Synchronous Javascript is simple and straight-forward, it basically means that the code starts executing line-by-line from the start, till the end. Asynchronous starts off similarly but it branches into different threads when it runs into an asynchronous function, (i.e) two different functions will be running at the same time.

```
let a = 1;
let b = 2;
console.log(a);
console.log(b);
```

OUTPUT:
1
2
This is a simple example of synchronous code, ‘a’ prints first followed by ‘b’.

```
let a =1;
let b = 2;
setTimeOut(() => console.log(“Asynch”), 100);   // an Asynch function that delays by 100 ms
console.log(a);
console.log(b);
```

OUTPUT:
1
2
Asynch

Notice two things:

1. The prints did not follow the flow of the program
2. The other two console logs happened while setTimeOut function was running in another thread, therefore instead of getting a 100ms delay and then ‘Asynch’, ‘1’ and ‘2’ as output, we got ‘Asynch’ in the end (after an overall 100ms delay).

2.What is an API? Explain in brief how it works and how will you create an API with technical details.

An API or Application Programming Interface is a piece of software that allows other softwares to communicate and attain data from other sources. APIs are what made the modern web possible. In web application the user has access only to the front end, to access information from the back end, we will use an API and display the response on the front-end. In layman terms APIs take in a user request, process it and returns a response.

An API can be used to get various types of information, like the weather, device's vibration mode, GPS location, user's information in a database, etc.

A real life example would be online flight booking. How is it that a single third party website like BookMyTrip has multiple airlines and all their details? This is possible by using an API, the third party website has an API that fetches data from all the airline databases and presents that to the user.

Back-end languages like Node.js can be used to make an API.

Technical Details of how an API works

A small summary of what we will be doing, a RESTful API is the norm of API making since the early 2000s, It takes in a request in a structured manner which is sent to the server, the server returns a response in another structure which can be decoded, most importantly we can take in the received JSON data and display it to our user in the front end using simple JS methods

1. The request is made in a structured manner, the state line contains information of what type of request method it is (POST, GET, etc), and the URI from which the required data can be obtained. Below the start line, we have headers which has meta data about the request (i.e) what sort of response we want (JSON) and proof of authorisation that tells the server that it can trust us with it's data

2. The server accepts our request, runs it's own code to get data from a database and returns a response message

3. The top of the response message contains info about how the request-response was (was the request in the wrong format? Was there some sort of server error? Or did everything execute accordingly?). The body of the response has the data payload

3. What is the difference between a library and a framework? Is React a library or a framework?

Both libraries and frameworks are reusable pieces of code written by other people which can be included in your project to make your life easier. The key difference between the two lies in the word “Inversion of control”.

When we use a library,

- We are in control, we decide when to call the library functions, when to use, what to use and so on.
- The flow of our code is completely up to use and the libraries are simply extra additions in our code that makes a particular task simple.

When we use a framework,

- We are not in control of how our code looks, the framework dictates the flow of our code. It calls our functions, in order words the control is inverted.

An example of a library would be JQuery. React on the other hand is a framework.

When we use React to make an application, we plug our code into React’s built-in functions, the entire structure of our code is different because React requires it to be different.

4. List some databases that are currently being used. Explain the advantages and disadvantages of each.

1. MongoDB:

Pros:

- It is highly scalable, sharding is simple and easy and doesn’t require a database design expert
- It is fast, easy to set up and has a lot of documentation
- It goes hand in hand with JS stacks like MERN and MEAN, because of it’s popularity, it’s easy to find technical support

Cons:

- It doesn't handle transaction of data at an optimum level leading to cases of lost data
- It occupies a lot of memory
- Joining different documents has proven to be a very tedious task
- It has a limited data size of 16MB per document.

2. **PostgreSql:**

Pros:

- Free, Open Source, highly expandable and uses normal SQL syntax
- Compatible with any variety of complex data types and also supports JSON
- Excellent language support
- Highly secure and scalable

Cons:

- It's slow (slower than MYSQL).
- It isn't popular, many apps don't support PostgreSQL

5. What are the advantages and disadvantages of using custom styles over existing styling frameworks? Which one do you prefer among the two?

Advantages:

- Complete freedom to make the page look as we want it to.
- Can incorporate animations, different effects and so on.
- Reduces unnecessary code and improves speed.

Disadvantages:

- Time-consuming and requires proper knowledge to execute standard features like nav-bar, responsive designs, etc.
- Browser-compatibility is a big issue.

I prefer using vanilla css for projects that focuses on the front-end. For projects that focus more on the back-end like a CRM, I prefer using a framework. In our project (All-in-one Medical App), I think the best way to proceed is to use a framework to create the initial prototype and then turn our focus back onto the front-end

6. Which project do you want to join? Why do you want to join this project specifically? What is your approach for solving this problem statement? What all tech stacks will you utilize for this project? What challenges will we be facing during the entire project and how would you solve them? Explain everything in detail.

I'm hoping to join the All-in-one Medical App, I'm aspiring to become a full-stack developer by the end of this year and I believe that this project has the right amount of front and back-end to help me achieve my goals, also this project specifically is very similar to an SIP that I had proposed.

My approach to solving the problem is to create an elaborate nested database.

In our project we will be dealing with two types of data,

1. That is provided by the hospitals (appointment schedule, list of doctors etc.)
2. That is provided by the user in our website (their history of ailments, their comments and reviews, etc.)

Every Hospital will have it's own database with a table filled with data from the hospital which can be obtained by using an API, and another table filled with our users data like comments, reviews and medical history.

A parent database will have a table connecting all these databases to it's hospital and another table with all the user data like name, address, phone, email and such.

1. When a user enters his ailments, we analyse that ailment and categorise it under a particular discipline of medicine like cardiology

2. We will search through the parent database to find hospitals with a tag of cardiology, all the hospitals will be displayed.

3. Once a user chooses the hospital of his choice we will enter that hospitals database and pull out their information like appointment schedules, list of doctors. Along with this our other table filled with comments and reviews will also be displayed depending on which doctor the user chooses.

4. If the user leaves a comment it will be stored in the specific hospitals database.

WHICH TECH STACK?

I think the MERN techstack might be a good option because it's really popular right now, therefore making it easy to hire new people if necessary. A popular techstack has the added advantage of lots of tech support and documentation, it will also receive constant updates that'll eventually make our lives easier.

CHALLENGES:

1. Database design is going to play a huge role in this project (the solution is what I've written as my approach)

2. Cyber-Security is also very important since we're dealing with sensitive data

3. Our target audience is old people who most likely struggle with current technology, making UX/UI design a key factor in the success of the project

4. To get data from hospitals we will need to make deals and such with both private and government hospitals, since both of these institutes run by different business models, our model must somehow smoothly include both.

5. Chances are that multiple hospitals will have their own style of database management, we will have to write multiple APIs to get data from all these hospitals

PROJECT: https://github.com/DK-GitHub-DK/CFI_WebOps_App