**a. What is the entire cycle of events that follows when you type in the url of a webpage?**

Ans : On typing the URL of a webpage, the browser looks up the IP address for that particular domain name present in the URL, by checking for it in the DNS(Domain Name Server). The browser then sends a HTTP(Hypertext Transfer Protocol) request to the server corresponding to that particular IP. The browser sends back a HTTP response. The browser then starts to render the HTML packets received from the HTTP response. The browser, while rendering the HTML, sends other requests too for the required embedded objects(images, css, Javascript). After the webpage has been loaded, the browser continues to send Asynchronous update requests to the server as required.

**b. How are urls mapped to website hosted on a server in Bangalore?**

Ans : After getting the domain name, it searches for the IP address using DNS query :

To find the DNS record, the browser checks four caches.

● First, it checks the browser cache. The browser maintains a repository of DNS records for a fixed duration for websites you have previously visited. So, it is the first place to run a DNS query.

● Second, the browser checks the OS cache. If it is not in the browser cache, the browser will make a system call (i.e., *gethostname* on Windows) to your underlying computer OS to fetch the record since the OS also maintains a cache of DNS records.

● Third, it checks the router cache. If it’s not on your computer, the browser will communicate with the router that maintains its’ own cache of DNS records.

● Fourth, it checks the ISP cache. If all steps fail, the browser will move on to the ISP. Your ISP maintains its’ own DNS server, which includes a cache of DNS records, which the browser would check with the last hope of finding your requested URL.

(Ref: <https://medium.com/@maneesha.wijesinghe1/what-happens-when-you-type-an-url-in-the-browser-and-press-enter-bb0aa2449c1a>)

The browser then initiates a TCP connection with the server associated with that IP and sends an HTTP request. The rest of the procedure has been explained in part ‘a’.

**d. What does setting up a server even mean?**

Ans : A server is a piece of software or hardware which accepts requests and sends responses. Setting up a server means to install the required piece of software on a hardware so that the hardware can respond with the required data(E.g - It can be Mail Server, Application Server, Game Server, DNS, Web Server, etc) when a request is thrown as it.

**e. Get an overall idea of how a server resolves a request? [ Google Apache and NginX, and see what role they play].**

Ans : Apache is an open-source, cross-platform web server software. A Web server software connects the database framework or file-server(backend) to our frontend GUI(webpage/android app,etc).

I couldn’t find good structured reading material for the basics of Apache and Nginx. I Googled them only to find their short descriptions and then they directly dived in relevant coding stuff. It would be great if you could point me to the right direction.

**f. Suppose you built a html page for the first time [you should if you haven’t already :P],**

**and you are very enthusiastic to show it to your friend on his device. How would you**

**do this if you are not allowed to share your code with him or upload your website. [**

**Every laptop is a server. There’s actually a place like 127.0.0.1)**

Ans : Firstly in my method I’ll need **Node.js** to be installed in my Windows system. Then I’ll follow the following steps :

1. Open command prompt in Administrator mode.
2. cd to the required directory containing the requisites for the html file.
3. Now create a http-server(local) using “npm”
4. **npm install http-server -g**
5. Run **http-server** command. It runs our web app in the default port 8080.
6. Now if we go to our localhost in our browser we can access it. Site : **127.0.0.1:8080**
7. Now we have to make it public and accessible to the internet.
8. We can do it in a number of ways, but the easiest will be to use Port Forwarding.
9. If I’m connected to some primary router for internet, I’ll have to dive into the router configuration and forward the **port 80** to the LAN IP address(can be easily found through **ipconfig,**  as we use static IP inside the insti) of my workstation, kind of the same way, we create a FTP server(I’ve created an active FTP server through LAN, for my wingies to share movies and stuff :P) on our PC, or kind of the same way DC++ works on the insti level intranet.
10. Now anyone with my WAN/Global IP address, which can be found by just going to [<http://www.whatismyip.com/>] can access the website that we just hosted !

**2. You want to introduce the feature that the books are searchable by the author? What will be the best way to accommodate these. Keep in mind there is additional load on the server when you are querying the tables to get the information.**

**2a. You want to minimize the number of queries. The simplest way is to fetch all the books and match the author for each of them. Can you do any better?**

Ans : We can use defer() and only() as there are database columns [Book Name, Author Name are the only columns that we’ll need] that we know thatwe won't need, so we avoid loading them. If we do use them, the ORM(Object Relational Mapping) will have to go and get them in a separate query, making this an additional load on the server.

So we won’t fetch all the books, we’ll just get the required columns from the dataframe. That’ll speed up the process a lot.

**queryset = Book.objects.filter(name\_author=author).values(‘str\_book’)**

This will give us a list of dictionaries, with the names of the books, with the given “author” name.

I’ve implemented the method that will do the author to book mapping, in the **models.py**

**3. [BONUS] Whenever the user wants to browse the list of all books available, you want to display them based on his interests, and not randomly. A similar model is implemented for the events in InstiApp (you have to dive into the repo here ).**

**a. Can you figure out the approximate calculations behind it and understand which**

**factors are given the most weightage?**

Ans : Mainly this problem is dealt with loyalty points/bonuses a person gets for doing things related to his/her interests : (Here I’ve referred to **prioritizer.py** while answering)

* Apply bonus to **events starting soon** with **Gaussian Distribution Bonus,** depending on the time for which the person has registered for the event.
* Apply **linear decay** in weight for events that have a **long time to start.**
* Apply **inverse** relation penalty to weight of **long running events**.
* Apply a **high bonus** for **finishing** an event.
* Apply a bonus for following a body and also set a **MAX\_BONUS(he/she won’t get more than this just for following the body)** for the specific user, to remove redundancy.

Therefore, we can draw some conclusions :

* How fast the user registers or gets interested towards a specific event dictates his/her interests. This should be given a good amount of priority.(Similarly, if a user readily registers for a new book to be borrowed, he/she should be given bonus points for that genre of the book)
* When the user takes part and completes an event, it should be given a huge bonus towards his/her interest. And in future such related events can be suggested to him/her.(This is analogous to borrowing and reading a book !)

**b. Since the books do not follow any hierarchy, how would you decide the order?**

Ans : We can introduce another entity like a list of “**tags**”in the **book class** to specify the tags or genre names related to a specific book. And similarly we need to introduce an **interest** field(This will contain the genres and the points/bonuses the user has in these genres, so that we get a priority order) in the **User class**, so that we can match the users with their interests.

All references :

For coding part :

<https://docs.djangoproject.com>

<https://books.agiliq.com/projects/django-orm-cookbook/en/latest/select_some_fields.html>

<https://github.com/wncc/IITBapp/>

<https://github.com/wncc/InstiApp>

<https://www.webforefront.com/django/setuprelationshipsdjangomodels.html>

For Network part :

<https://en.wikipedia.org/wiki/Server_(computing)>

<https://computer.howstuffworks.com/what-is-network-server.htm>

<https://www.geeksforgeeks.org/tcp-and-udp-server-using-select/>

<https://docs.microsoft.com/en-us/IIS-Administration/>

<https://httpd.apache.org/docs/2.4/urlmapping.html>

<https://nodejs.org/en/>

<https://complete-concrete-concise.com/web-tools/how-to-change-the-apache-port-in-xampp/>

<https://stackoverflow.com/>

<https://networkengineering.stackexchange.com>