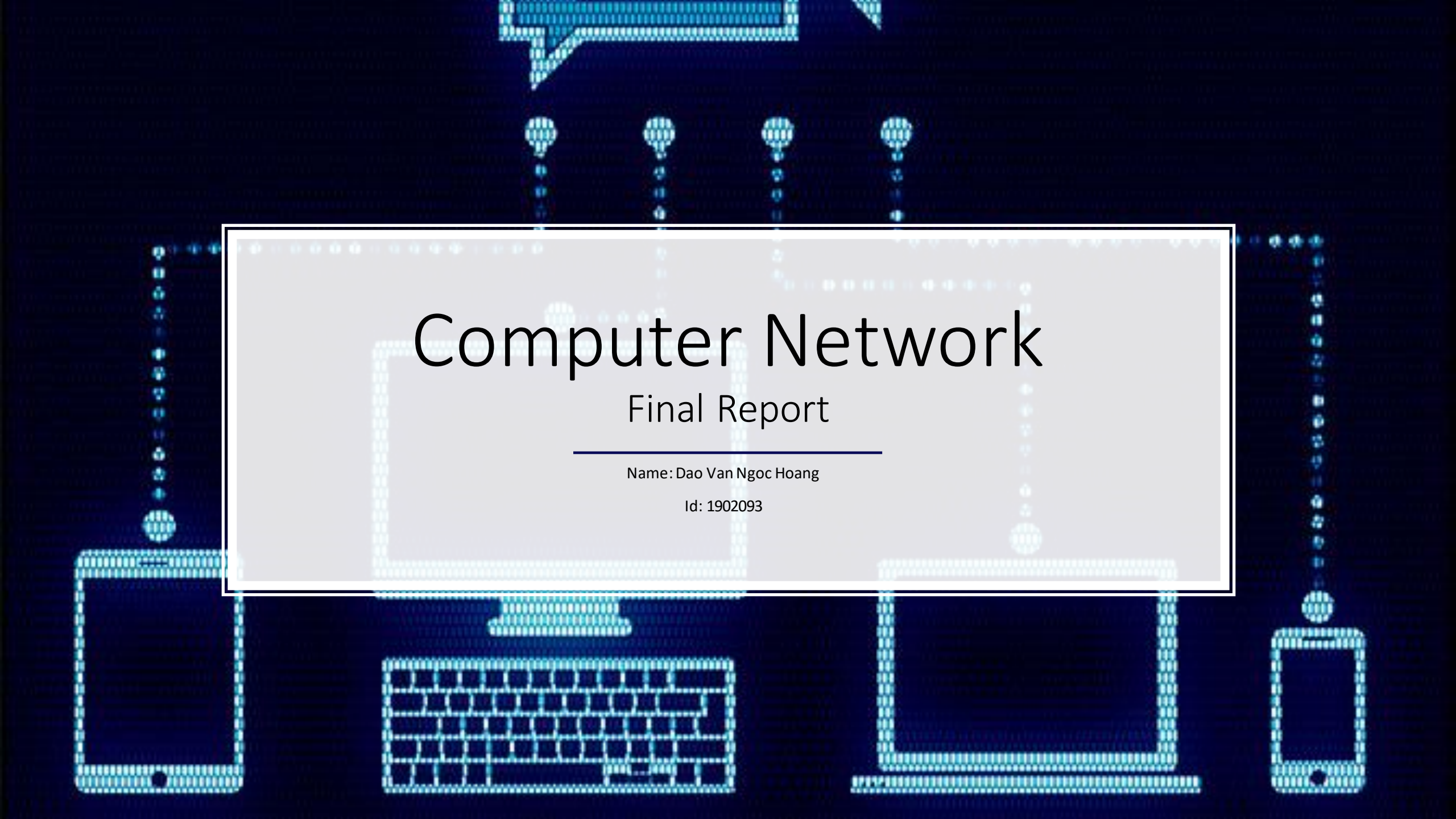


Computer Network

Final Report

Name: Dao Van Ngoc Hoang

Id: 1902093

What is my website?

An Easy IT website act like a cloud based app that allow users create file, folder and write their data via APIs.

In the commercial we have a lot of similar services, however they seems complicated and not minimal for user, so I propose a Website that provide Short APIs and easy to interact.

```
// create a new user schema
module.exports = (mongoose) => {
  // create a new schema for the user model
  const contentSchema = new mongoose.Schema(
    {
      _id: mongoose.Schema.Types.ObjectId,
      date: {
        type : Number,
        required: true
      },
      body: {
        type: String,
        required: true,
      },
      type: {
        type: String,
        required: true},
    },
    { versionKey: false }
  );


  // create a new user model
  const Content = mongoose.model("Contents", contentSchema);
  return Content;
};
```

```
module.exports = (mongoose) => {
  const fileSchema = new mongoose.Schema(
    {
      _id: {
        type: String,
        required: true,
        unique: true,
      },
      size: {
        type: Number,
        default: 0
      },
      name: {
        type: String,
        required: true,
      },
      contents: [{type: mongoose.Schema.Types.ObjectId, ref: "Contents"}],

      owner: [String],
      createdAt: {
        type: Number,
        required: true,
      },
    },
    { versionKey: false }
  );

  const Files = mongoose.model("Files", fileSchema);
  return Files;
};
```

```
module.exports = (mongoose) => {


    const folderSchema = new mongoose.Schema({
        _id: {
            type: String,
            required: true,
            unique: true,
        },
        name: {
            type: String,
            required: true
        },
        quantity: {
            type: Number,
            default: 0
        },
        owner: [String],
        size : {
            type: Number,
            default: 0
        },
        files: [
            {
                type: String,
                ref : 'Files'
            }],
        createdAt: {
            type: Number,
            required: true,
          },
        },
        { versionKey: false },

    );


    const Folder = mongoose.model("Folders", folderSchema);
    return Folder;
}
```

Models

- Because we save data into mongo dB, we need to create specific model for each user, file, folder, content object to easier to maintain.
- We use mongoose as a library to manage mongo DB.

Authentication

Using JWT (Json Web Token) to generate user's access tokens from user's information (ID) and a (Private key).

Expires after 30 minutes if user is not using services.

If in 30 minute user still using app, we generate a new access token for user.

METHODS

GET

GET – get file/folder information by date.

POST

POST – create file/folder

UPDATE

UPDATE – add content, add file in folder...

DELETE

DELETE - delete by date, delete a whole object

POST

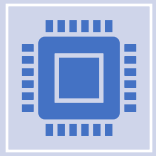
Called by the interface (Fe), create new object and send back an short API which user can interact directly with the API.

POST method is sent with a body contains:

type : file | folder => this indicate file or folder.

isPrivate : true/false => about the authentication of the file.

name : file name.



Server receive those information and generate a ID that include type file and type of authentication of the file, then return to client a APIs allow interacting with that object.

GET

Return filename, createAt, size, content.

http://localhost:5000/fffppp-1f27-a439-41aa-a77c-6da50670 to get all information of the file

http://localhost:5000/fffppp-1f27-a439-41aa-a77c-6da50670?from=2030-2-11&to=2050-4-11 to get information of a file in a specific time interval.

PUT

- Update content to file, create new file – content in folder
- http://localhost:5000/FFFppp-038e-9f31-4623-840e-73e9a1f2
- Server will get all raw data in request body and convert to text and save the content to database.

DELETE

- Delete content/files in a specific time interval or delete file/folder.
- http://localhost:5000/api/delete/FFFppp-0a15-e137-4fe7-b100-a76d69e9 if the API not contain the query of time then we will delete the object.
- http://localhost:5000/api/delete/FFFPPP-b958-ce37-4a6a-b425-776984df?from=2020-2-11&to=2022-4-11 delete information inside object with a time interval.

My understanding about Network

The Client-Server Programming Model

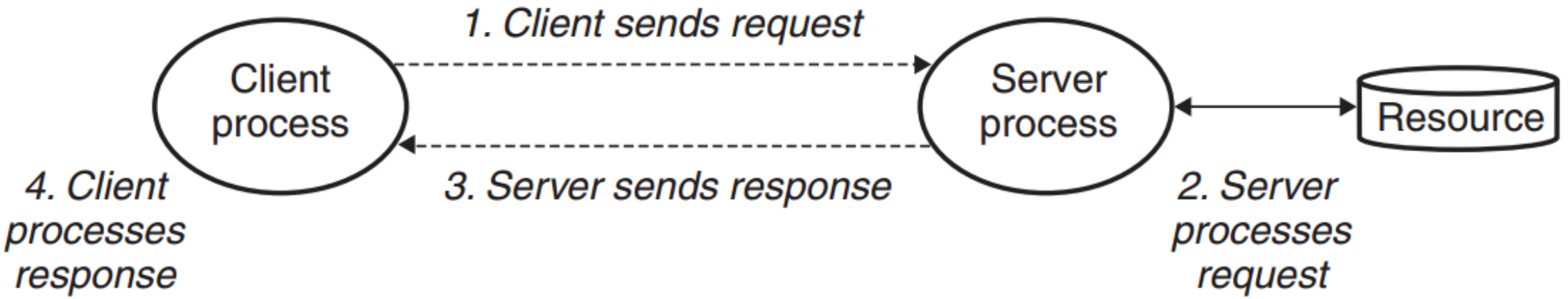


Figure 11.1 **A client-server transaction.**

DNS

- Easier to remember the server's address.
- whaleshark.ics.cs.cmu.edu.

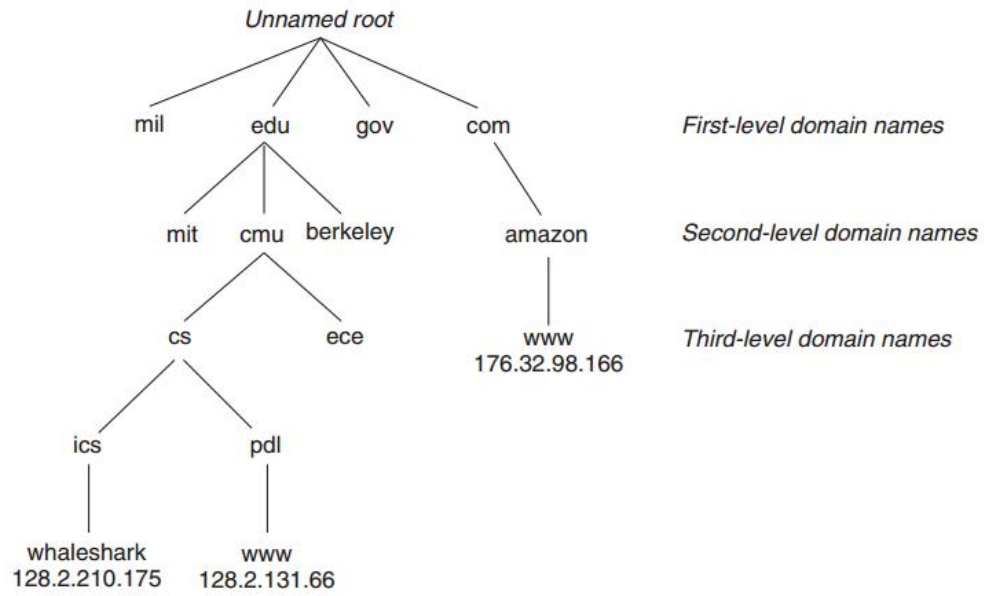
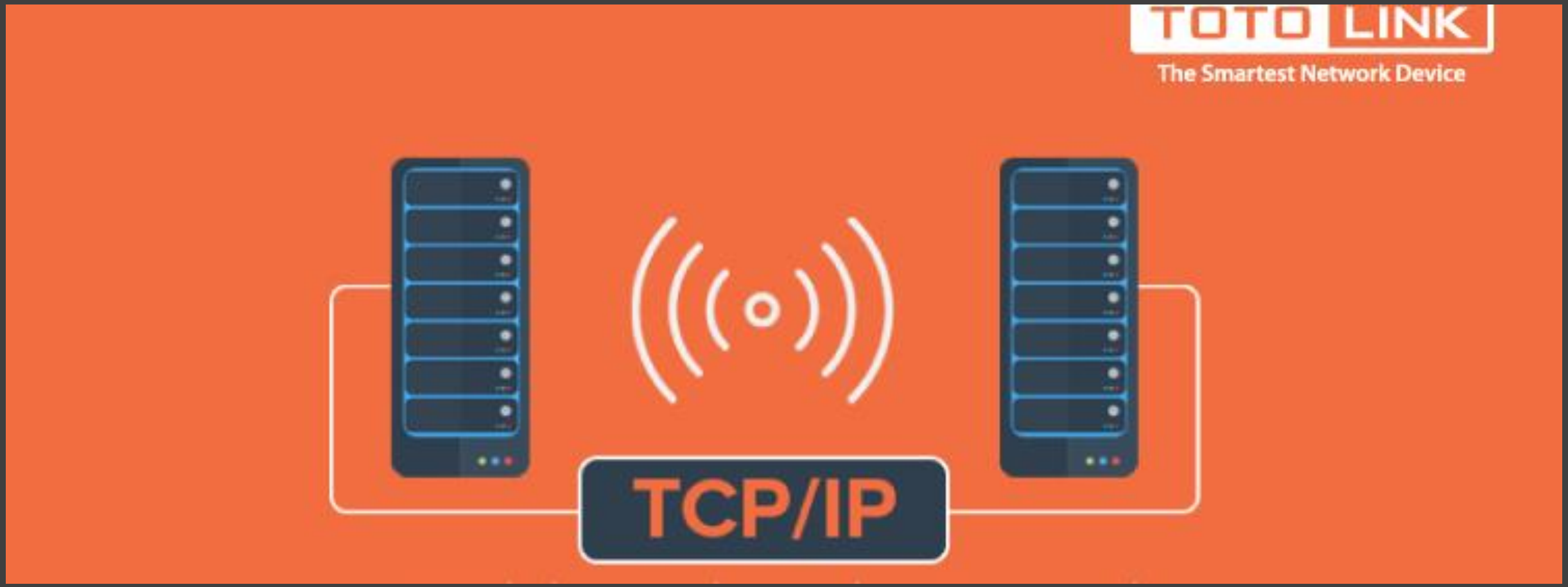


Figure 11.10 **Subset of the Internet domain name hierarchy.**

The 8 steps in a DNS lookup:

1. A user types 'example.com' into a web browser and the query travels into the Internet and is received by a DNS recursive resolver.
2. The resolver then queries a DNS root nameserver (.).
3. The root server then responds to the resolver with the address of a Top Level Domain (TLD) DNS server (such as .com or .net), which stores the information for its domains. When searching for example.com, our request is pointed toward the .com TLD.
4. The resolver then makes a request to the .com TLD.
5. The TLD server then responds with the IP address of the domain's nameserver, example.com.
6. Lastly, the recursive resolver sends a query to the domain's nameserver.
7. The IP address for example.com is then returned to the resolver from the nameserver.
8. The DNS resolver then responds to the web browser with the IP address of the domain requested initially.



TCP/IP

- Guarantee the packets are successfully arrived the destination without error or missing.

HTTP

HyperText Transfer Protocol (HTTP), the Web's application-layer protocol.

HTTP is Implemented in client-server programming model.

HTTP uses TCP as its underlying transport protocol.

non-persistent and persistent connections

- non-persistent connections, a brand-new connection must be established and maintained for each requested object and the TPC connection is terminated after a request.
- persistent connections, connection is established for a request and subsequent of requests and responses between a same server-client is sent in one persistent TCP connection, and over a period of idle time it’ll be closed.

HTTP req & res format

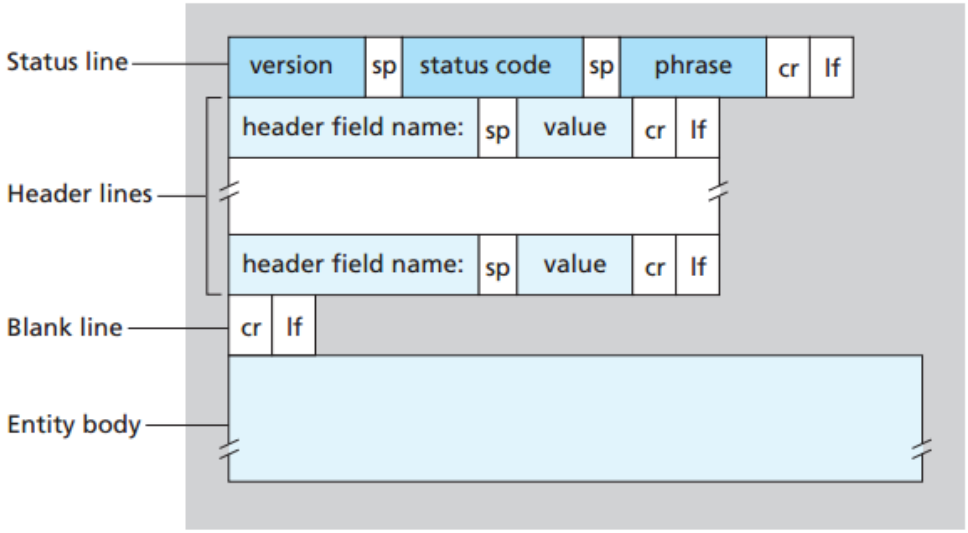


Figure 2.9 ♦ General format of an HTTP response message

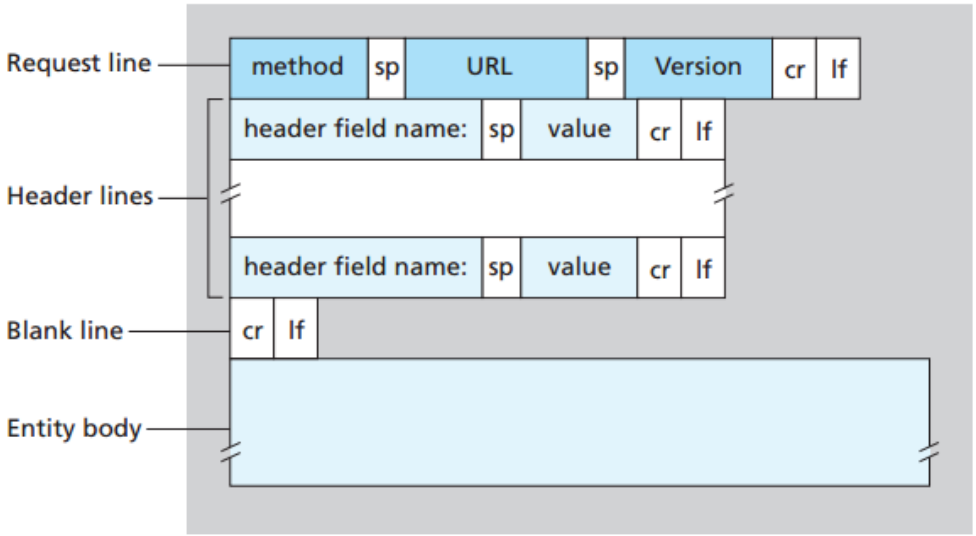


Figure 2.8 ♦ General format of an HTTP request message

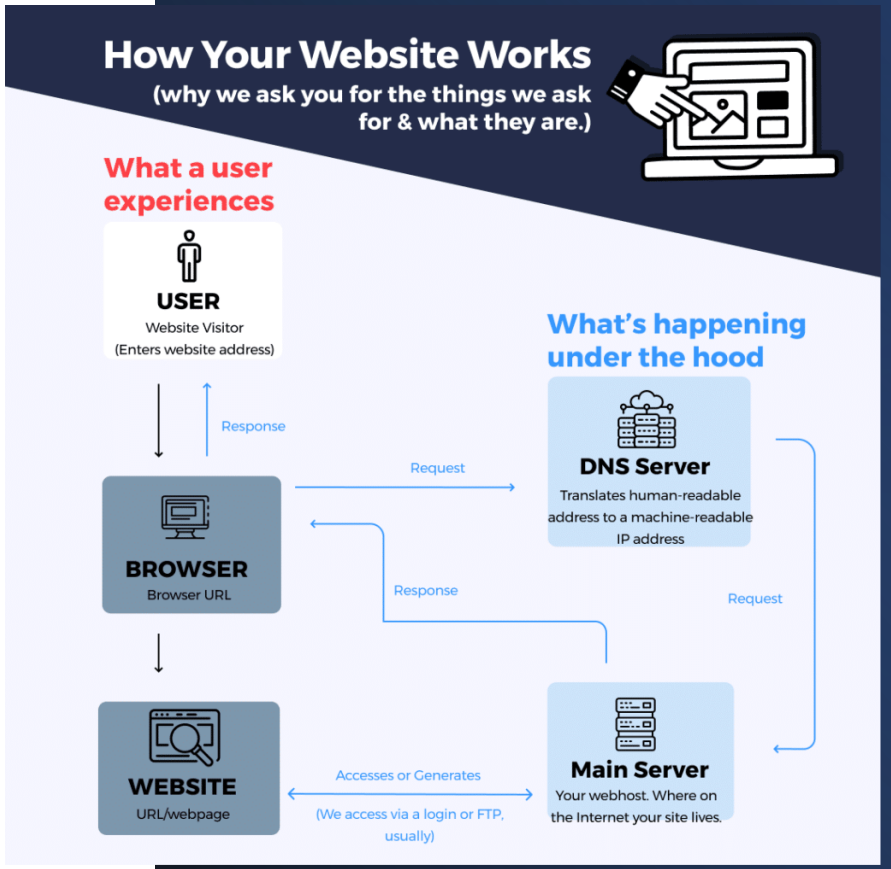
How does a website work

The browser goes to the DNS server, and finds the real address of the server that the website lives on (you find the address of the shop).

The browser sends an HTTP request message to the server, asking it to send a copy of the website to the client (you go to the shop and order your goods). This message, and all other data sent between the client and the server, is sent across your internet connection using TCP/IP.

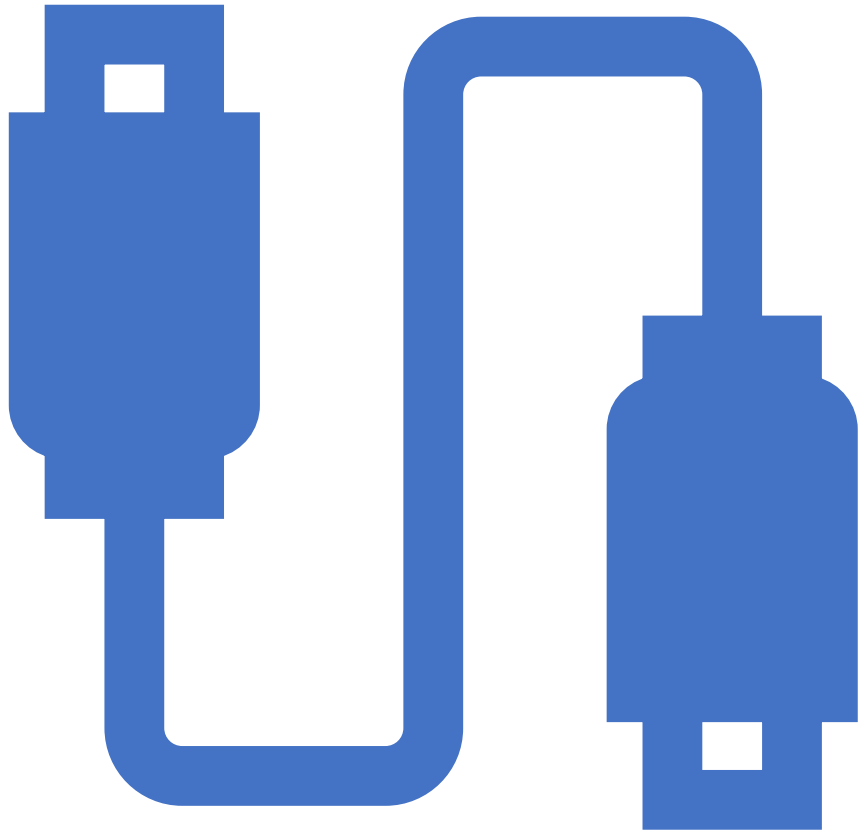
If the server approves the client's request, the server sends the client a "200 OK" message, which means "Of course you can look at that website! Here it is", and then starts sending the website's files to the browser as a series of small chunks called data packets (the shop gives you your goods, and you bring them back to your house).

The browser assembles the small chunks into a complete web page and displays it to you (the goods arrive at your door — new shiny stuff, awesome!).



Socket

- A socket is an end point of a connection. Each socket has a corresponding socket address that consists of an Internet address and a 16-bit integer port2 and is denoted by the notation **address:port.**
- the client's socket address is assigned automatically by the kernel, the server's socket address is typically some well-known port that is permanently associated with the service. For example, Web servers typically use port 80, and email servers use port 25.



Socket methods in C and Linux

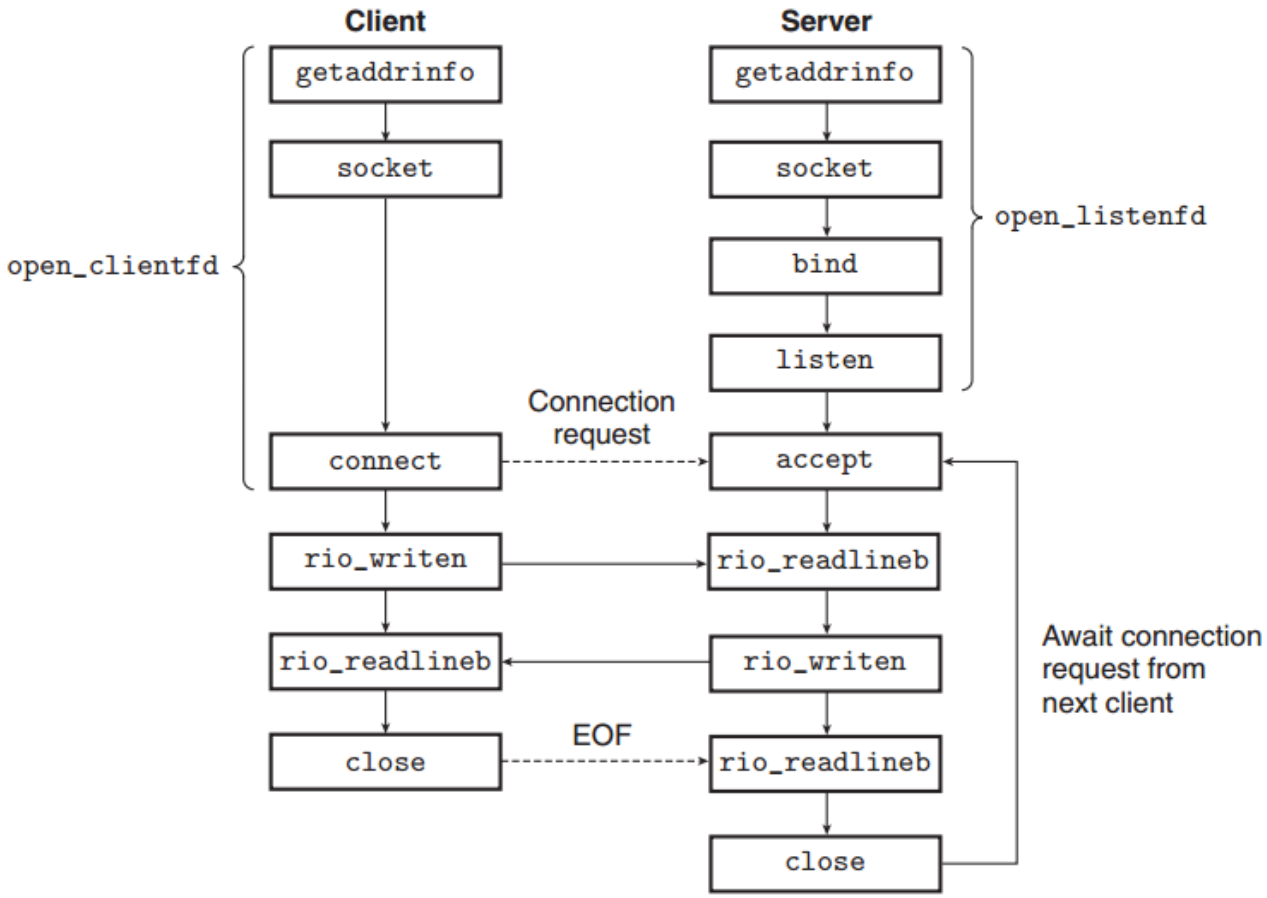


Figure 11.12 **Overview of network applications based on the sockets interface.**

References

- *What is DNS? | how DNS works | cloudflare*. (n.d.). Retrieved May 13, 2022, from https://www.cloudflare.com/learning/dns/what-is-dns/
- Randal-E.-Bryant-David-R.-OHallaron-Computer-Systems.-A-Programmers-Perspective-3rd-ed.-2016-Pearson.
- Kurose, J., & Ross, K. (2022). *Computer networking*. Harlow: Pearson.