**Yonder Technical Assignment Junior DevOps**

**Darius Petruta**

**Steps to access questions:**

1. I downloaded and installed Docker.
2. I pulled the image for SpringBoot application.
3. I opened a terminal and listed my images with **docker images**
4. I ran the yonder images using **docker run -p 30000:8080 <imageId>**
5. I opened a Chrome tab to **localhost:30000.**

**Questions and answers:**

1. Exemplify two data structures that you know and describe some situations where you would use them.

* List: Keeping track of student grades in a list for easy access and manipulation in the classroom.
* Hash table: Storing user session data in a web application where you can quickly retrieve user information based on session IDs.

1. You open a web browser and access http://www.tss-yonder.com. What is the IP address behind this website and how does the browser know how to get the correct IP?

We can get the IP address using: <https://www.nslookup.io/website-to-ip-lookup/>. It is: 172.67.73.177. Or we can use, in terminal **nslookup http://www.tss-yonder.com.**

1. Exemplify two transport protocols and think of two applications that would use each of them.

* **Transmission Control Protocol (TCP)** is a connection-oriented protocol that provides reliable, ordered, and error-checked delivery of data packets over a network. Applications that require reliable and ordered data delivery typically use TCP. Examples include:
  + Web browsing: HTTP (Hypertext Transfer Protocol) relies on TCP for transmitting web pages, ensuring that data is delivered reliably and in the correct order.
  + Email: SMTP (Simple Mail Transfer Protocol) uses TCP to transmit email messages between mail servers, ensuring that messages are delivered reliably without loss or duplication.
* **User Datagram Protocol (UDP)** is a connectionless protocol that provides faster but unreliable delivery of data packets over a network. It does not guarantee delivery or ordering of packets. Applications that prioritize speed and efficiency over reliability often use UDP. Examples include:
  + Voice over IP (VoIP): VoIP applications, such as Skype or WhatsApp calls, use UDP for real-time communication because it minimizes latency. While some data loss may occur, real-time communication is prioritized over reliability.
  + Streaming media: Streaming services like YouTube or Twitch use UDP to deliver live video and audio content. While some packets may be lost or arrive out of order, UDP's speed and low overhead are beneficial for streaming large volumes of data in real-time.

1. You wrote a chat web application in your favorite programming language. You need to host this somewhere and run it so that the entire world can start using it. Describe how you would do that and the tools you would use.

* Manual way:
  + choose a Hosting Provider
  + set up Server Infrastructure
  + database Configuration
  + domain Name and DNS Configuration
  + security
  + load Balancing and Scaling
  + monitoring and Logging
  + continuous Integration and Deployment (CI/CD)
* Serverless Platforms: Platforms like AWS, Google Cloud Functions, Azure Functions, and others allow us to deploy code without worrying about server management.

1. Now your application is famous but unfortunately it has a lot of bugs. You want only you and a couple of your friends to be able to access it until you patch it. Describe two ways you can achieve this.

We can implement IP whitelisting (and add our friends Ips there, so only they could reach the application) or disable the login for all users, and create some super-users that can login (for our friends).

1. Your application is ready for the public once again. You realize that you forgot about security and any network administrator can see the messages that a user sends or receives. How would you improve your application to prevent this? Is there any way to do this so that not even the application owner (you) can see the messages between two random users?

To improve the security of the chat application and prevent unauthorized access to user messages, especially by network administrators or even the application owner, we can implement end-to-end encryption (E2EE). End-to-end encryption ensures that only the sender and intended recipient can read the messages, making it nearly impossible for intermediaries, including network administrators and even the application owner, to access the message content.

1. What are cookies and what are they used for? Find a cookie used by http://www.tss-yonder.com and copy its name and value. What do you think is its purpose?

Cookies are small pieces of data stored on the client's browser by websites they visit. They serve various purposes, including session management, personalization, tracking user behavior, and maintaining user preferences across multiple sessions.

One common use of cookies is for session management, where a unique identifier is stored in a cookie to track a user's session on a website. This allows websites to remember user preferences, login status, and other session-specific information as the user navigates through different pages.

On <http://www.tss-yonder.com>, an example can be: **cookieconsent\_status = dismiss,** or **PHPSESSID = bbv8hkcma4jbvhmpuabfj09bdd.**

1. While writing your application you need to create more worker processes for processing some data. How can you create child processes in your favorite language? What are the possible states of a process?

Python:

**import multiprocessing**

**def worker():**

**print("Worker process")**

**if \_\_name\_\_ == "\_\_main\_\_":**

**p = multiprocessing.Process(target=worker)**

**p.start()**

**p.join()**

Possible states of a process:

* Running
* Ready
* Blocked
* Suspended (or Stopped)
* Terminated (or Finished)

1. Your application is running but it still has a few problems. Occasionally, it returns an error page. How can you find the PID of your application? What would you do to debug it?

We can use **ps aux | grep <application\_name>.**

1. What DBMS would you use to store your application data and why? How would you store the passwords of each user?

For storing application data, MongoDB would be a suitable choice DBMS due to its document-oriented nature and flexibility. MongoDB's schema-less architecture allows for storing data in JSON-like documents, making it ideal for applications with evolving data models. It offers horizontal scalability through sharding and replica sets.

Regarding the storage of user passwords, it's essential to prioritize security by following best practices. Firstly, passwords should never be stored in plaintext. Instead, they should be securely hashed using algorithms like bcrypt, which incorporate salting and key stretching techniques to mitigate common attacks such as rainbow table attacks and brute force attacks. Each user's password should be salted with a unique random value before hashing to prevent attackers from cracking multiple passwords simultaneously. By securely hashing passwords and implementing strong encryption practices, the application can safeguard user credentials and maintain the integrity of user accounts.