**Interview Questions**

**Can you explain a recent project you worked on and the specific role you played in**

**its development?**

**“Real-Time User Interaction Web App”**

For this web application, I developed a back-end server using Flask, a Python web framework and utilized an SQL database to store user data and article information. The project was hosted on Digital Ocean, and I used SSH keys for secure access to my server.

In the NGINX configuration, I implemented security measures to prevent common web exploits and to maintain the confidentiality and integrity of the data exchange between the client and server.

Specifics:

Firewall rules: The first part of the file lists firewall rules that allow or deny traffic from certain services. Rules have been set up for Nginx and OpenSSH, as well as for Postfix, a popular open-source mail transfer agent. Only specific IP addresses are allowed to access the server via SSH (port 22). Additionally, HTTP and HTTPS traffic is allowed from anywhere for the Nginx server (ports 80 and 443).

HTTPS and Redirection: The Nginx server is configured to listen on port 443, which is the standard port for HTTPS traffic. All HTTP traffic (port 80) to noahwilliamshaffer.com and www.noahwilliamshaffer.com is redirected to HTTPS to ensure secure, encrypted communication. Any other requests on port 80 are returned with a 404 error.

SSL/TLS settings: The Nginx server uses certificates from Let's Encrypt, a free, automated, and open certificate authority. TLSv1, TLSv1.1, TLSv1.2, and TLSv1.3 are enabled, but older, insecure protocols like SSLv3 are disabled to protect against vulnerabilities like POODLE. The ssl\_prefer\_server\_ciphers online ensures that the server's preferred order of ciphers is used rather than the client's. server\_tokens off; is used to prevent Nginx from displaying the version number in error messages and server headers, which can be valuable information for potential attackers. Access and error logs are enabled. They can be useful for monitoring and detecting any attempted security breaches. Gzip compression is enabled for various types of files. This is not a direct security measure, but it improves the efficiency of data transfer between the server and clients. The proxy\_set\_header directives ensure that the real client IP addresses are passed to the proxied servers. This is useful for logging and can also be critical for applications that need to know the client's IP address. A basic mail proxy configuration is included with authentication, which can add another layer of security by requiring users to authenticate before they can use the mail server. The configuration uses a Unix socket for communication with a proxied server (http://unix:/home/marlee/ProjectFiles/myproject.sock). Unix sockets can offer better performance and security than TCP sockets on the same machine.

To ensure secure navigation between various parts of the website, I employed Flask's routing system. This allowed me to map URLs to Python functions, enabling users to go from one web page to another securely.

The routes are listed below:

@app.route("/UserProfiles", methods=["GET", "POST"]): This route manages the user profiles. When a GET or POST request is made to /UserProfiles, the user\_profiles() function is executed. It retrieves data from two SQLite databases (likedArticles.db and dislikedArticles.db), collects the email from the session data, checks if the email belongs to an admin, and finally returns the relevant HTML template. If the user is an admin, it returns the UserProfiles.html template, otherwise, it redirects to the ErrorAdmin.html page.

@app.route("/database"): This route displays all articles stored in the Art table in the database. It fetches all rows from the Art table and renders the database.html template with these rows.

@app.route("/login"): The login route redirects the user to an external OAuth login page. After the user logs in successfully, the OAuth service will redirect the user back to the /callback route.

@app.route("/callback", methods=["GET", "POST"]): This route is intended to be the callback

URL for the OAuth2 service. When the service redirects back to this URL, it processes the OAuth2 tokens, saves the user information in the session, and adds the email to a database.

@app.route("/logout"): The logout route clears the session data and redirects the user to the logout URL of the OAuth service. After logging out, the user is redirected back to the /home route.

@app.route("/Admin"): This route is for administrators. It fetches user data from the user's table in the users.db SQLite database, checks if the current user is an admin, and then renders the Admin.html template with user data if the user is an admin. If not, it redirects the user to the home page.

@app.route("/Profile", methods=["GET", "POST"]): This route renders the Profile.html page, which displays a user's profile.

@app.route("/", methods=["GET", "POST"]): This route maps to the home page of the website and renders the home.html template.

@app.route("/disliked", methods=["GET", "POST"]): This route handles user interactions with articles they dislike. If the user submits a POST request, the function extracts the disliked article's details and stores them in a database. It then fetches all articles from the Art table and displays them on the news.html page.

@app.route("/liked", methods=["GET", "POST"]): This route functions similarly to the "/disliked" route but handles the articles that users like instead.

@app.route("/news", methods=["GET", "POST"]): This route displays the news articles stored in the Art table in the database on the news.html page.

Lastly, if \_\_name\_\_ == "\_\_main\_\_": app.run(debug=True) is a Python idiom.

For user authentication, I integrated Auth0, a robust and easy-to-implement authentication and authorization platform. This allowed users to log in securely with a system that follows best practices for handling user credentials. Once a user was logged in, I tracked them using their unique user ID.

To further personalize the user experience, I implemented a feature where users could like or dislike articles. I used their unique IDs to keep track of these preferences. This information was stored securely in our SQL database.

Leveraging APScheduler's BackgroundScheduler, I implemented an automatic refresh system in my Python back-end. Every 12 hours, a task ran in the background clearing the SQL tables with the article data, then repopulated them with the latest top articles from HackerNews API. Additionally, 'like' and 'dislike' user data were cleared every three days for efficient operation and to keep user preferences current. Care was taken to ensure these clearings didn't disrupt the user experience. This approach kept the application fresh, efficient, and responsive to users' recent interactions.

On the front-end side, I used JavaScript and HTML to create an interactive and dynamic user interface. For instance, the color of the like and dislike buttons changed based on user interaction, giving instant visual feedback.

For each user, I created a personalized page that showed their liked and disliked articles. This allowed users to review their previous preferences and revisit articles of interest.

Overall, I focused on creating a secure, user-friendly web application that provided personalized and up-to-date content from a popular news source. I used a combination of Python, Flask, SQL, JavaScript, HTML, and third-party services to accomplish this task, emphasizing secure and efficient practices throughout the project.

On many of the projects I’ve been in, I've taken a leadership role where I created groups and meetings. I normally identify the approach we should take, get people motivated to work, and ask the necessary questions. I never hesitate to find the answers our group needs. With most projects, I attempt to set our group up for success, and the work is evenly distributed. I am extremely proud of myself for the project I’ve referenced because, due to the circumstances, my partner was unable to contribute due to an internship she had taken in Dallas, Texas. I took on the project, and I wrote and implemented almost all of the code. Furthermore, the additional demands on me drove me to put more work in than I ever had on a previous project. I went above and beyond what was expected of me in case I was to get stuck. I wanted to get extremely ahead as early as I could. Because of that feeling of no one to fall back on, I led the entire class in my implementation. I was creating new material and helping others in the class. I learned more during this project than I have in any other project. I took absolute responsibility and accountability for it. Because I had no one to take a leadership role in my project, I aspired to be a class leader. I helped my friends and those who asked, encouraging them and helping them learn while not simply providing code snippets. I talked to them through my way of thinking. It was an incredibly rewarding experience and inspired a lot of growth in me. It made me incredibly confident in my skills to work independently, but it also allowed me to help others.

**Describe a situation where you encountered a technical roadblock during a project.**

**How did you identify and resolve the issue?**

During the project, I had many minor roadblocks. I pushed through the majority of them but I remember one simple issue that I was stuck on and ignored for a very long time because I could simply work around it. I needed how to figure out how to get the user ID from the Python side of the code to correctly identify users based on their likes and dislikes. I had already implemented it on the HTML side so it should have been incredibly straightforward to use the session I was passing into the HTML to get the details I needed. It was the syntax of the lines I needed on the Python side that was the issue. The HTML side was easy to implement. I was so ahead of everyone in the class at this point that by the time everyone had caught up to this point in the project I had essentially finished everything else. After dedicating several hours scrounging the internet for help I found the answer on the auth0 homepage in their information section. I identified that this issue was going to be a problem but instead of putting the rest of the project on pause, I decided to just complete everything else and thought I may be able to find the solution passively or from a colleague in the future. This didn’t end up happening, but after I was way ahead of schedule I dedicated the data to finding the answer and it took an additional few days to have the project refined and completed fully.

**What programming languages, frameworks, and tools are you most comfortable**

**working with, and why?**

I love working in Python because of all the modules and tools available in it. I’ve been coding in languages like Java and C++ for many years, but as comfortable as I am with them, I love the power that Python provides in its utility. Beyond the project I previously mentioned, I also used Pandas and Sklearn for data visualization and have become very comfortable with the language. I love Python's utility. Because of the previous project, I mentioned compounded with a data systems class where we focused sql exclusively, I got extremely comfortable working with and utilizing SQL tables. Flask was the framework I used with this project so I have a lot of experience using it to bring a web application to production. I made an entire e-commerce application in C using the .Net framework, so I’d say it's a framework I'm very comfortable with. I have a lot of experience with javascript as well, using it to aid the functionality of my html web pages. Some data science tools I’m comfortable with and have used extensively are pandas, NumPy, SciPy, Matplotlib, Seaborn, and Scikit-Learn to perform visualizations and data diagnosis. I am comfortable with AWS, Azure, and Docker. I routinely use Git from the command line, GitHub, and GitLab for many of my projects.

**Have you ever had to optimize code for performance? If so, how did you approach**

**it, and what were the outcomes?**

For this particular project, I used a variety of methods to optimize the performance which include. Before diving into optimization, thoroughly review the codebase to understand the current implementation and identify potential bottlenecks. Use tools like Python's cProfile module to identify time-consuming functions. For the front end, the browser's performance tab can be used to analyze which tasks take a long time to execute. For Flask routes, be careful about database queries. Redundant or inefficient queries can greatly reduce performance. Use an ORM like SQLAlchemy's lazy='joined' to reduce the number of queries. Enable gzip compression in your Nginx configuration. It reduces the size of the data that's being transferred between your server and your users. This can speed up the load times of your web pages. Implement caching at various levels. For the backend, frequently accessed data that doesn't change often can be cached to reduce database queries. On the front end, service workers can be used to cache static files. On the front, use tools to minify and combine your CSS and JavaScript files. This can reduce the number of HTTP requests and the size of the files that the client needs to download. If the application has heavy traffic, consider using Nginx's load-balancing features to distribute traffic across multiple servers. Serve your static files like CSS, JavaScript, and images through a Content Delivery Network (CDN) to reduce latency for users in different geographical locations. While not necessarily code optimization, I used AWS to monitor the traffic and log it. I looked for suspicious activity and ensured the security was in a good state.

**How do you ensure the reliability and stability of your code? Share your strategies**

**for testing and debugging.**

For the previously mentioned project, we used a variety of methods to ensure the reliability and stability of the code. We began with manual testing. Manual testing is beneficial to catch issues automated tests might miss. I would manually test the application across different browsers and screen sizes to ensure consistent behavior and appearance. I had friends log in and use my website from their devices to break it. I would then use Python's built-in pdb debugger for the backend and browser developer tools for the front end. Debugging involves replicating the issue, understanding the cause, and then applying the fix. For some issues, I would implement proper error handling is crucial for reliability. I'd ensure the application fails gracefully if something goes wrong. The logging system I implemented would enlighten me on the issue that was occurring. This is a rare but often necessary step that I implement as a last resort. I used JMeter to perform load testing to understand how the system behaves under significant stress and identify any potential bottlenecks. For the Flask routes and associated logic, I would write unit tests using a framework-like unit test in Python's standard library. These tests would cover all the functions and routes, ensuring that individual units of code work as expected.

**Can you discuss a time when you collaborated with a cross-functional team or**

**multiple stakeholders to deliver a software solution? How did you handle**

**conflicting requirements or opinions?**

During my final year at university, I had the opportunity to collaborate on a capstone project that involved developing a health-tracking application. Our team was a mix of computer science majors, UI/UX designers, and health science students. This collaboration made our team cross-functional and allowed us to have multiple perspectives on the project.

As it usually happens in a diverse team, conflicting opinions and requirements arose. For example, there was a time when the UI/UX designers proposed an intricate design that would make the application aesthetically appealing but could have potentially compromised its performance due to the complexity of the animations and transitions they wanted to implement.

On the other hand, we, the developers, were more focused on performance, reliability, and functionality. We were concerned that the design could impact the speed and overall performance of the application, especially for users with low-end devices.

To handle this situation, we held a meeting to discuss our concerns openly. We explained the technical limitations and how a complex design could affect the performance of the app. The designers, in return, explained their viewpoints and the importance of the design in creating an engaging and user-friendly interface.

We reached a compromise by adopting a simpler design that was both visually appealing and less demanding on the app's performance. The developers also agreed to optimize the code as much as possible to accommodate some of the more complex elements of the design.

This experience taught me the importance of clear communication and negotiation in a cross-functional team. I learned that it's essential to respect the expertise of colleagues from different disciplines and understand that everyone's primary goal is to create the best product possible.

Resolving such conflicts in requirements or opinions can sometimes be challenging, but it's always a learning opportunity and, in the end, can lead to a better outcome.

**Describe a situation where you had to make a trade-off between delivering a**

**project on time and achieving a technically superior solution. How did you handle**

**this situation?**

A situation where I had to make a trade-off to deliver a functioning product on time was during the third. increment of an ecommerce app in c# using the .Net Framework. I ended up getting a completely functional product at the cost of settling for a solution that wasn’t clean. I submitted it on time, and I found out that only 7 students in the class of 60+ had submitted it on the deadline. My submission was fully functional, but the instructor ended up extending the due date by 2 or 3 weeks. I got bonus points for my early submission but I had to rework my implementation for the next increment which made the next one extraordinarily difficult. Because I was afraid to take the time to sit back and model the project effectively due to the deadline, it caused further problems down the line. It was a great learning experience and I still did well in the class. I think getting products functional by the deadline is the most important objective even after that experience. But depending on the leniency and context given to me, I could be made to prioritize either delivering on time or a superior solution.

**Have you ever faced a situation where you had to refactor a large codebase? How**

**did you approach the refactoring process to ensure minimal disruption to the**

**existing functionality?**

For the third increment of my e-commerce app in C# in the .Net framework, I previously mentioned how I sacrificed a good solution to ensure complete functionality by the deadline. For the next increment, My previous solution made it impossible to continue with what was being asked without completely refactoring it. Instead of going completely freeform in my approach, I used the MVC method. MVC stands for Model-View-Controller, a design pattern that separates the application logic into three interconnected components. This pattern is particularly useful in applications where you want to separate the data and business logic from the user interface. The Model represents the application's data structure. It typically interacts with the database, retrieving and storing information. It contains all of the business logic that governs how data can be created, retrieved, updated, and deleted. For instance, in my e-commerce application, a model might represent the products, with properties such as the name, price, quantity, and so on. The View is what the user sees and interacts with - the user interface. It displays data from the model to the user and sends user commands (e.g., button clicks, selections) to the controller. The view does not process any data itself - it simply formats and displays data from the model. The Controller acts as the intermediary between the Model and the View. It processes all the user requests, usually from user input via the View. For example, if a user clicks a button to delete an item from the store when purchased, the Controller will process this action by sending a command to the Model to delete the appropriate data. Then it updates the View to reflect this change. I started with this schema on a completely blank app to refactor all the functionality in my previous implementation. It was arduous but it made the next functionality in the next increment easy to achieve. Between the third and fourth increments, I would like to say 75% of the time was refactoring my entire code base and then 25% was the new implementation. So I have experience and understanding of that process.

**How do you stay updated with the latest industry trends and technologies? Can you**

**provide examples of how you have implemented new technologies or techniques in**

**your work?**

I try to keep up to date in a variety of ways. Sites like Hacker News, TechCrunch, and The Next Web are excellent sources for the latest news and developments in the tech world. Platforms such as Coursera, Udacity, and edX provide courses on trending topics. Webinars hosted by tech companies and communities also offer insight into new technologies and best practices.

Websites like StackOverflow and GitHub, and communities like Reddit's /r/programming, have active discussions on a variety of topics, which can help stay abreast of new developments.

Podcasts like "The Daily" by the New York Times and YouTube channels like "TechLead" and "Siraj Raval" provide valuable insights. Local tech meetups and larger conferences provide opportunities to learn from experienced professionals and network with peers.

As for implementing new technologies in my work, I'd like to mention a project I completed during my final semester. It involved developing a web application using the MERN stack (MongoDB, Express.js, React.js, and Node.js), which was a relatively new technology stack at that time. I chose the MERN stack after attending a local tech meetup where a speaker talked about the benefits of using this particular stack for web development. The application turned out to be highly performant and scalable and received positive feedback from the project evaluator. I learned not only about these new technologies but also how to effectively apply them to solve real-world problems. I am constantly looking out for new tools, languages, and frameworks that can help optimize my workflow, enhance the quality of my work, and broaden my skill set. I believe this continuous learning mindset is crucial for any professional in the field of computer science.

**Describe a time when you had to troubleshoot and fix a critical production issue**

**under time pressure. How did you approach the problem, and what was the**

**Outcome?**

A time when my team and I had to fix a large critical issue under time constraints was during the final increment of our operating systems class. This project was to create a fully-fledged operating system with multi-threading capability and background processes in C. We had major issues transferring our project and installing the requirements. There was a compatibility issue with our system that took us back almost two weeks. These are two of my best friends now because this class was so incredibly difficult we would spend 3+ hours on a discord call every day. My friend who is now employed was doing the calculations on how we only need a 20% on this increment to pass the class and seeing as we only have a week left we should just scrap home of finishing. But I took hold and said no, when we go home tonight we get in call and we don’t leave until we have at least a third of the increment realized. We spent almost 12 hours on that call. I identified all the issues and places where we were stuck and emailed questions so we could have the answers for the next day. The progress we were making was addictive and it felt like we could finish it in full. After we got out of class we knew to just jump into the call and start working. And because we never ended up getting stuck, we just slowly made our way to finishing the project on time to full completion. We ended up getting an A in one of the most notoriously difficult classes that FSU has to offer. The pressure of having only a third of the time we should have had prompted us to dial in and work hard. The progress we made was crazy and we enjoyed the grind. Even though we server project was the biggest testament to my skills in comp sci, I believe that project was my greatest show of leadership, especially under stress.