Workbook – Damian Petrov

Question 1

> Identify and explain common and important components and concepts of web development markup languages.

Web development markup languages, primarily HTML (Hypertext Markup Language), serve as the foundation for creating the structure and content of web pages. HTML utilises tags to define various elements on a webpage, such as headings (ranging from `<h1>` to `<h6>`), paragraphs (`<p>`), lists (`<ul>`, `<ol>`, `<li>`), links (`<a>`), images (`<img>`), and more. These tags encapsulate content and provide a hierarchical structure to the webpage (Mozilla Developer Network, n.d.).

In addition to tags, HTML elements can have attributes, which offer additional information or modify their behaviour. For instance, the `href` attribute is used for hyperlinks, while the `src` attribute specifies image sources. HTML documents follow a specific structure, including an opening `<html>` tag, a `<head>` section for metadata and resources, and a `<body>` section for the visible content that is to be rendered by the browser (W3Schools, n.d.).

HTML5 introduced semantic elements such as `<header>`, `<nav>`, `<section>`, `<article>`, `<footer>`, which provide meaningful structure to the content, improving accessibility for users and enhancing search engine optimisation (SEO) (MDN Web Docs, n.d.). Moreover, HTML forms (`<form>`) facilitate user input and interaction, incorporating elements like text fields (`<input type="text">`), checkboxes (`<input type="checkbox">`), radio buttons (`<input type="radio">`), and more.

Accessibility and SEO are crucial considerations in web development. Semantic HTML elements contribute to accessibility by structuring content meaningfully, while adding descriptive `alt` attributes to images enhances accessibility for users with screen readers. Meta tags, including `<title>`, `<meta description>`, and `<meta keywords>`, impact SEO by providing search engines with relevant information about the webpage's content (Search Engine Land, n.d.).

Although CSS (Cascading Style Sheets) is not a markup language, it complements HTML by styling the presentation and layout of web pages. CSS selectors target HTML elements, while properties define their visual appearance, such as color, size, typography, and layout. Responsive design techniques, enabled by CSS, adapt webpage layout and styling based on viewport size and device characteristics. Additionally, JavaScript enhances web interactivity and dynamic behaviour through DOM manipulation, event handling, and AJAX for asynchronous data exchange between the browser and server.

References:

- MDN Web Docs. (n.d.). HTML. Retrieved from https://developer.mozilla.org/en-US/docs/Web/HTML
- Mozilla Developer Network. (n.d.). HTML. Retrieved from https://developer.mozilla.org/en-US/docs/Web/HTML
- Search Engine Land. (n.d.). What is SEO? Retrieved from https://searchengineland.com/guide/what-is-seo

- W3Schools. (n.d.). HTML Attributes. Retrieved from https://www.w3schools.com/html/html_attributes.asp

---

Question 2

> Define the features of the following technologies that are essential in terms of the development of the internet:
>
> - Packets
> - IP addresses (IPv4 and IPv6)
> - Routers and routing
> - Domains and DNS
>
> Explain how each technology has contributed to the development of the internet.

The development of the internet has been shaped by key technologies and concepts that form its foundational infrastructure. In this response, we explore four essential technologies—packets, IP addresses (IPv4 and IPv6), routers and routing, and domains and DNS (Domain Name System)—examining their features, historical significance, and current relevance to the development of the internet.

Packets:

In network communication, packets serve as the basic units of data transmission. They contain the actual data (payload) and additional details such as the source and destination addresses. The internet employs a strategy of dividing data into smaller packets for efficient transmission. Each packet traverses the network independently and may take different routes to its destination. Once they reach their destination, the packets are reassembled to form the original data. This method of packet-switching is more efficient and reliable than the traditional circuit-switched networks. It allows for dynamic data routing along the most efficient paths, maintaining robust communication even if certain links or nodes fail (Techopedia, n.d.). Packets revolutionised communication with the advent of packet-switched networks in the late 1960s. ARPANET, the precursor to the modern internet, adopted packet switching in 1969, marking a significant milestone (History of the Internet, n.d.). Today, packet switching remains the backbone of the internet, enabling efficient utilisation of network resources and reliable delivery across vast distances for various internet services (BBC Bitesize, n.d.).

IP addresses (IPv4 and IPv6):

Internet Protocol (IP) addresses are vital for internet communication, enabling devices to find and exchange data with each other. The modern internet uses both IPv4 and IPv6 addresses, with the latter gaining traction to accommodate the surge of connected devices and new technologies (Cisco, 2020). IP addresses emerged alongside the early internet. IPv4 addresses, composed of 32 bits, were the initial

standard for identifying devices on the internet and spurred the growth of networked computing. However, the rapid increase in connected devices led to a shortage of IPv4 addresses, necessitating the creation of IPv6 in the late 1990s. IPv6 addresses, with their 128 bits, provide a significantly larger address space to cater to the increasing number of devices. Today, both IPv4 and IPv6 are used concurrently, with IPv6 underpinning cutting-edge technologies like IoT and 5G networks (Microsoft, n.d.). In a typical scenario, when an individual uses their smartphone to surf the internet, the Domain Name System (DNS) translates the domain name of a website, such as 'www.example.com,' into its corresponding IP address. The web browser then connects to the web server hosting the website, using either IPv4 or IPv6 connectivity based on availability. Data packets, which include the user's device's source IP address and the web server's destination IP address, are then sent over the internet. The web server processes the request and returns a response, which comprises resources like HTML, CSS, and JavaScript, split into data packets. The web browser assembles the web page from these packets, ultimately presenting it to the user. Throughout this process, IPv4 and IPv6 addresses play a pivotal role in identifying the source and destination of data packets, facilitating smooth communication and interaction on the internet.

Routers and Routing:

The task of choosing and navigating paths within and across digital networks is known as routing (Cisco, 2021). This function is performed by a unique device known as a router. A router, whether tangible or intangible, aids in the exchange of data between various packet-switched digital networks (IBM, 2022). It operates by scrutinising the destination IP address of a data packet, figuring out the optimal path for it to reach its endpoint, and then dispatching it accordingly. Upon receiving a data packet, a router investigates the packet's destination IP address and contrasts it with entries in its routing table to determine the packet's subsequent hop. Routing tables hold directives for dispatching data to particular network destinations, often taking into account elements like cost. These tables essentially act as a rule set that dictates the most effective method of directing traffic to any given IP address. For instance, a standard domestic router guides outbound traffic to its Internet Service Provider (ISP) along a default route. Furthermore, routers frequently carry out Network Address Translation (NAT), which obscures the private IP addresses of devices within a local network by replacing them with a single communal public IP address. NAT is beneficial in preserving globally valid IP addresses and bolstering network security. The notion of routing has its roots in the creation of the earliest computer networks. The Interface Message Processor (IMP), which was devised for the ARPANET in the late 1960s, exhibited functionality akin to contemporary routers, thereby illustrating the fundamental concepts of routing (Internet Society, 2029).

Domains and DNS (Domain Name System):

Domains serve as easily readable names that people use to locate and access resources on the internet, including websites, servers, and services. Each domain is associated with a unique IP address, a numerical identifier used by computers to find and connect to the desired resource. The Domain Name System (DNS) acts as a hierarchical and distributed naming system, converting domain names into corresponding IP addresses (Gill, 2018). When a user enters a domain name into their web browser, such as "www.example.com," the browser sends a request to DNS servers to obtain the IP address associated with that domain. DNS resolution involves a series of steps, starting from the root DNS servers and traversing through authoritative DNS servers for the specified domain until the IP address is successfully resolved. This translation process ensures that users can access internet resources using user-friendly domain

names rather than having to remember complex numerical IP addresses (Müller, 2019). Since its inception in the early 1980s, the Domain Name System has been a cornerstone of internet functionality, providing an essential service for internet navigation and accessibility. It simplifies the process of locating and accessing online resources by translating human-readable domain names into machine-readable IP addresses. Additionally, DNSSEC (Domain Name System Security Extensions) enhances security and trust in the domain system by providing mechanisms to authenticate DNS data and ensure its integrity (ICANN, 2020). Overall, DNS plays a critical role in facilitating seamless communication and interaction on the internet, making it an indispensable component of the modern digital landscape.

References:

- Techopedia. (n.d.). "Packet." Retrieved from Techopedia: https://www.techopedia.com/definition/4026/packet
- History of the Internet. (n.d.). "Packet Switching." Retrieved from History of the Internet: https://www.historyoftheinternet.com/packet-switching/
- BBC Bitesize. (n.d.). "How are Data Packets Routed on the Internet?" Retrieved from BBC Bitesize: https://www.bbc.co.uk/bitesize/guides/zg8mhyc/revision/3
- Cisco. (2020). "IPv6: The Foundation of Next Generation Internet." Retrieved from Cisco: https://www.cisco.com/c/en/us/solutions/enterprise-networks/ipv6.html
- Microsoft. (n.d.). "Introduction to IPv6." Retrieved from Microsoft Docs: https://docs.microsoft.com/en-us/troubleshoot/windows-server/networking/introduction-to-ipv6
- Cisco. (2021). "What is Routing?" Retrieved from Cisco: https://www.cisco.com/c/en/us/solutions/what-is-routing.html
- IBM. (2022). "What is a Router?" Retrieved from IBM Cloud: https://www.ibm.com/cloud/learn/what-is-a-router
- Internet Society. (2029). "A Brief History of the Internet." Retrieved from Internet Society: https://www.internetsociety.org/internet/history-internet/brief-history-internet/
- Gill, P. (2018). "The Domain Name System." Retrieved from Cloudflare: https://www.cloudflare.com/learning/dns/what-is-dns/
- Müller, S. (2019). "Understanding DNS - Beginners Guide to DNS." Retrieved from Cloudflare: https://www.cloudflare.com/learning/dns/what-is-dns/
- ICANN. (2020). "Domain Name System Security Extensions (DNSSEC)." Retrieved from ICANN: https://www.icann.org/resources/pages/dnssec-what-is-it-why-important-2019-03-05-en

---

Question 3

> Define the features of the following technologies that are essential in terms of the development of the internet:
>
> - TCP
> - HTTP and HTTPS
> - web browsers (requests, rendering and developer tools)
>
> Explain how each technology has contributed to the development of client and server communication over the internet (50 - 150 words for each technology)

The internet, as we know it today, is a complex ecosystem of interconnected technologies that facilitate communication, data transfer, and information exchange on a global scale. At the heart of this digital landscape are foundational protocols and technologies that govern how data is transmitted, received, and interpreted across networks. In this discussion, we explore three pivotal components of internet communication: Transmission Control Protocol (TCP), HyperText Transfer Protocol (HTTP) and its secure counterpart HTTPS, and the role of web browsers. These technologies form the backbone of modern internet communication, shaping the way we interact with online resources and enabling the seamless exchange of information. We delve into the functionalities of TCP, the significance of HTTP and HTTPS in web communication, and the pivotal role of web browsers as gateways to the vast expanse of the internet. Through understanding these fundamental elements, we gain insight into the mechanisms that underpin the functioning of the internet and its transformative impact on various facets of our lives.

TCP:

TCP (Transmission Control Protocol) is like a postman for the internet, but it's much more complex. It's a transport protocol used on top of IP to ensure reliable transmission of packets. It includes mechanisms to solve problems that arise from packet-based messaging, such as lost packets, out-of-order packets, duplicate packets, and corrupted packets. TCP uses a three-way handshake to establish a connection, sequence and acknowledgement numbers to keep track of data transmission, and a timeout mechanism to detect lost packets (Cisco, 2021).

TCP, which was developed in the 1970s, has been a fundamental part of internet communication. TCP makes our internet connections stable and reliable, and has been instrumental in the development of the internet by providing a reliable, ordered, and error-checked delivery of a stream of bytes between applications running on hosts communicating via an IP network. (TechTarget, n.d.).

HTTP and HTTPS:

HTTP and HTTPS are the languages that computers use to talk to each other on the web. HTTP (HyperText Transfer Protocol) is the foundation of data communication on the World Wide Web. It is a set of rules for transferring hypertext requests and information between the server and browser. HTTPS (HTTP Secure) is the secure version of HTTP, encrypted to increase the security of data transfer. This is particularly important when users transmit sensitive data, like credit card information. These protocols have

revolutionised the internet by enabling the creation of dynamic, user-friendly, and secure websites (Mozilla, n.d.).

HTTP was developed by Tim Berners-Lee and his team between 1989-199. It established the rules for transferring hypertext requests and information between the server and browser. HTTPS was developed to enhance the security of data transfer, particularly when users transmit sensitive data. These protocols have revolutionised client-server communication on the internet by facilitating the creation of dynamic, user-friendly, and secure websites. (Cloudflare, n.d.).

Web Browsers:

Web Browsers are your gateway to the internet. They have evolved significantly, bringing forth a wide range of features and capabilities for web development. They interpret and render HTML, CSS, and JavaScript to display web pages. Browsers also make requests to servers and handle responses, often using HTTP or HTTPS. Modern browsers come with developer tools for testing and debugging. These tools, along with advancements in HTML, CSS, and JavaScript support, have contributed to the interactive and dynamic nature of today's web (Apple, n.d.).

All these technologies - TCP, HTTP/HTTPS, and Web Browsers - work together to make the internet what it is today.

References:

- Cisco. (2021). Transmission Control Protocol (TCP). Retrieved from Cisco: https://www.cisco.com/c/en/us/solutions/what-is-routing.html
- TechTarget. (n.d.). Transmission Control Protocol (TCP). Retrieved from SearchNetworking: https://searchnetworking.techtarget.com/definition/TCP
- Mozilla. (n.d.). What is HTTP? Retrieved from Mozilla Developer Network (MDN): https://developer.mozilla.org/en-US/docs/Web/HTTP
- Cloudflare. (n.d.). Why HTTPS Matters. Retrieved from Cloudflare: https://www.cloudflare.com/learning/ssl/why-is-https-important/
- Apple. (n.d.). Web Technologies Overview. Retrieved from Apple Developer: https://developer.apple.com/documentation/webkit/web_technologies

---

Question 4

> Describe the features of interpreters and compilers and how they are different.

When it comes to transforming high-level language into machine code that computers can comprehend, both interpreters and compilers serve the purpose. However, their methods and characteristics differ:

Interpreters

- **Translation Process:** Interpreters process the program incrementally, one statement at a time (TutorialsPoint, n.d.).
- **Analysis and Execution Time:** While interpreters are typically quicker at analysing the source code, the total execution time tends to be slower compared to compilers.
- **Memory Efficiency:** Interpreters don't generate object code, making them more memory-efficient (GeeksforGeeks, n.d.).
- **Error Detection:** Interpreters excel at error detection as they translate code line by line during execution (Programiz, n.d.).
- **Flexibility:** Languages that use interpreters are generally more flexible than those that use compilers. (Codecademy, n.d.).
- **Examples:** Languages such as JavaScript, Python, and Ruby employ interpreters.

Compilers

- **Translation Process:** Compilers examine the entire program and translate it as a whole into machine code (Studytonight, n.d.).
- **Analysis and Execution Time:** Compilers may take longer to analyse the source code, but the total execution time is usually faster compared to interpreters (GeeksforGeeks, n.d.).
- **Memory Usage:** Compilers produce object code which requires further linking, thus they consume more memory.
- **Error Detection:** A compiler lists all errors after the compilation process. If there are errors in your code, it won't compile (Programiz, n.d.).
- **Security:** Compilers contribute to enhancing the security of applications(Techopedia, n.d.).
- **Examples:** Languages such as C, C++, and Java utilise compilers.

In conclusion, whether to use an interpreter or a compiler depends on the specific needs of the programming task. Interpreters are typically used for scripting languages and in development environments where programs are small and short-lived. Conversely, compilers are used for system programming and application programming where programs are large and performance optimisation is crucial.

References:

- Codecademy. (n.d.). Learn Python: Syntax. Retrieved from https://www.codecademy.com/learn/learn-python-syntax

- GeeksforGeeks. (n.d.). Compiler vs Interpreter. Retrieved from https://www.geeksforgeeks.org/compiler-vs-interpreter-2/
- Programiz. (n.d.). Interpreter vs Compiler. Retrieved from https://www.programiz.com/article/difference-compiler-interpreter
- Studytonight. (n.d.). Compiler and Interpreter. Retrieved from https://www.studytonight.com/compiler
- Techopedia. (n.d.). Compiler. Retrieved from https://www.techopedia.com/definition/1208/compiler
- TutorialsPoint. (n.d.). Compiler Design. Retrieved from https://www.tutorialspoint.com/compiler_design/

Question 5

> Identify TWO commonly used programming languages and explain the benefits and drawbacks of each.

Python:

Python, a high-level and interpreted programming language, is often lauded for its straightforwardness and readability. Its syntax is uncluttered and intuitive, making it a prime choice for novices venturing into the world of programming. Python's adaptability is another of its strong suits. It finds applications in diverse domains such as web and game development, machine learning, data analysis, and scientific computing. This wide-ranging applicability is largely due to the extensive array of libraries and frameworks that Python's vibrant and active developer community has contributed (Python Software Foundation, n.d.).

However, Python is not without its limitations. Being an interpreted language, Python's execution speed may lag behind that of compiled languages like C++ or Java. While this may not pose a problem for small-scale projects or scripts, it could become a bottleneck for larger, more intricate applications. Moreover, Python is not the go-to choice for mobile app development. Although it is feasible to develop mobile apps with Python, it is not as efficient owing to performance constraints and the absence of native support.

Conversely, Java, a statically-typed, object-oriented programming language, is renowned for its "write once, run anywhere" ethos. This implies that Java code can be written on any device, compiled into low-level machine code, and then executed on any platform equipped with a Java Virtual Machine (JVM). This platform-agnostic nature is one of Java's key advantages (Oracle, n.d.).

Java:

Java is also sturdy. Its compilers are capable of identifying numerous issues that would only surface during execution in other languages. This preemptive error detection makes Java a dependable option for building large-scale enterprise applications. Additionally, Java naively supports multithreading, enabling the creation of programs that can execute multiple tasks concurrently, thereby enhancing performance.

However, Java is not devoid of shortcomings. Its syntax is more verbose compared to languages like Python, which can make it harder to read and write. This verbosity can result in prolonged development cycles and a higher likelihood of errors. Furthermore, Java applications may exhibit slower startup times compared to applications written in languages like C++ or Go (Red Hat, n.d.).

In summary, both Python and Java come with their unique sets of strengths and weaknesses. The choice between the two is often dictated by the specific needs of the project, the environment in which the software will operate, and the proficiency of the development team. A thorough understanding of these aspects can aid in selecting the most suitable tool for the task at hand.

References:

- Python Software Foundation. (n.d.). Python. Retrieved from https://www.python.org/
- Oracle. (n.d.). Java. Retrieved from https://www.oracle.com/java/
- Red Hat. (n.d.). Java. Retrieved from https://www.redhat.com/en/topics/java

---

Question 6

> A hypothetical client has sent you an email (shown below), asking for you to build them a website. Write an appropriate, professional email response that shows your understanding of the client's needs for the website, as well as an understanding of appropriate technologies or tools needed to build the website yourself.

```
Hello there!

My name is Alex, and I'm the director of the Super Awesome Museum (SAM).
We display a variety of interesting artefacts, objects, and paraphernalia
about all sorts of things from all over the world.

I'm writing to you because the SAM needs a website. The museum is new in
the city, we're fully funded and don't sell our items but we just need to
encourage people to visit the museum.

We would need a website that showcases some of our interesting exhibits
and items, helps people find their way to the museum, and helps people
contact the museum.

We don't know much about this website stuff - does this sound like
something that you can do?

Looking forward to hearing from you,

Alex
```

```
Director

Super Awesome Museum
```

My email response:

Dear Alex,

I hope this message finds you well. I am thrilled to hear about the Super Awesome Museum (SAM) and its mission to bring fascinating artefacts and objects to the public.

I understand that as a new museum in the city, it's crucial to have a strong online presence to attract visitors. Rest assured, creating a website that meets your needs is definitely within my capabilities.

Here's a brief overview of what I propose:

1. **Exhibit Showcase:**
   We can create a visually appealing gallery to highlight some of the unique exhibits and items in the museum. This could include high-quality images, descriptions, and perhaps even interactive elements to engage visitors.

2. **Location and Directions:**
   To help people find their way to the museum, we can integrate a map feature with clear directions from various points in the city. This could also include information on public transportation options, parking facilities, and accessibility features.

3. **Contact Information:**
   A dedicated contact page will allow visitors to reach out to the museum for any inquiries. This could include a form for email inquiries, as well as phone numbers and operating hours.

4. **About Us Page:**
   To help visitors understand more about the museum's mission and history, we can create an 'About Us' page. This could also include profiles of key staff members, like yourself, and information about the museum's funding.

In terms of technology, I propose using a modern web development framework such as React for the front-end, coupled with a reliable back-end technology like Node.js. This will ensure the website is fast, responsive, and can be easily updated with new exhibits or information.

React is a JavaScript library developed by Facebook and is widely used for building user interfaces. It allows us to create reusable UI components, which makes development more efficient and easy to maintain moving forward.

Node.js is designed to build scalable network applications and is very efficient in handling multiple requests simultaneously, making it a great choice for a dynamic website like yours. Furthermore, using Node.js allows us to use JavaScript on both the front-end and back-end, which can improve development speed, consistency and maintenance.

I also recommend we implement a Content Management System (CMS). This will allow your team to easily update and edit content on your website, like adding new exhibitions or updating contact information. This

will reduce reliance on developer time and give you more control to keep content up-to-date and relevant in a timely manner.

I believe these technologies will allow us to create a robust, efficient, and engaging website for the Super Awesome Museum.

To discuss this further, I suggest we set up a meeting. I am available on the following days and times next week:

- Monday, 10:00 AM - 12:00 PM
- Wednesday, 2:00 PM - 4:00 PM
- Friday, 9:00 AM - 11:00 AM

Please let me know which time works best for you, or if there's another time that you'd prefer.

I look forward to discussing this project further with you and getting started on bringing the Super Awesome Museum online.

Best regards,

Damian Petrov
Full-stack Developer

Web Company Co.
PH: 9905 0093
E: damian@wcc.coom

Question 7

> Think back to a scenario or situation in your own software development projects or work.
>
> Explain how you would do things differently if you had a chance to go through that scenario again, using an appropriate reflective cycle or reflection technique.

Reflection is a vital component of personal and professional development, particularly in the rapidly evolving field of software development (Schön, 1983). By using Gibbs' Reflective Cycle, a well-established model for structured reflection, we can gain deeper insights into our experiences and use them to continuously improve. This cycle encourages us to describe the experience, evaluate our performance, analyse the situation, and then formulate an action plan for future improvement (Gibbs, 1988).

I recently completed a portfolio website project using HTML, CSS, and a touch of JavaScript. In line with Gibbs' Reflective Cycle, I will be reflecting on this project, identifying areas of growth and self-improvement. The importance of reflection cannot be overstated. It allows us to understand our strengths, recognize areas we need to improve, and ultimately, become better developers. Let's delve into this reflective journey.

Description: As a full-stack development student, I recently completed a portfolio website project using HTML, CSS, and JavaScript. The website showcases my development projects and articles related to web

development.

Feelings: I feel proud of what I've accomplished so far, but I also recognize that there are areas where I can improve. I am excited about the prospect of learning and growing in these areas. Evaluation: The project was successful in terms of achieving its primary goal - to showcase my work. However, there were several opportunities for growth and self-improvement that I identified.

Analysis — Opportunities for Growth:

- **Advanced JavaScript:** I realised the need to learn and apply advanced JavaScript concepts and libraries, such as React or Vue.js, to create dynamic content and improve the overall user experience. Responsive Design: I recognised the importance of using CSS frameworks like Bootstrap or learning more about media queries to ensure the website is fully responsive and provides an optimal viewing experience across all devices.
- **SEO Optimisation:** I understood the need to research and apply SEO best practices, such as proper use of meta tags, keywords, and alt text for images, to increase the visibility of the website on search engines. Performance Optimisation: I identified the need to implement techniques for improving website performance, such as minifying CSS and JavaScript files, optimising images, and leveraging browser caching.

Analysis – Opportunities for Self-Improvement:

- **Learning New Technologies:** I acknowledged the importance of continuously learning and adapting to new technologies and best practices in the ever-evolving field of web development.
- **Testing Methodologies:** I realised the need to incorporate more rigorous testing methodologies, including cross-browser and cross-platform testing, usability testing, and accessibility testing.
- **Version Control:** I understood the importance of using version control systems like Git for better code management. Making regular commits with clear messages, creating branches for new features, and using merge requests for code review.
- **Time Management:** I recognised the need to improve time management skills to ensure that adequate time is allocated for each aspect of the project, from design and development to testing and deployment.

Conclusion: By focusing on these areas, I believe I can significantly enhance my skills and the quality of my future projects.

Action Plan: Moving forward, I plan to dedicate time to learning advanced JavaScript, improving my understanding of responsive design, SEO, and performance optimisation. I will also focus on learning new technologies, improving my testing methodologies, using version control effectively, and managing my time better.

References:

- Schön, D. A. (1983). The Reflective Practitioner: How Professionals Think in Action. Harvard Business Review. Retrieved from https://hbr.org/1983/01/the-reflective-practitioner-how-professionals-think-

in-action

- Gibbs, G. (1988). Learning by Doing: A Guide to Teaching and Learning Methods. Oxford Polytechnic.

Question 8

> A large part of career growth as an information technology professional happens through networking and workshops, often found at online or in-person events or workshops.
>
> Create an action plan that identifies several relevant networking opportunities for you to participate in or attend, and add some information about what you expect to gain or grow through each item in the action plan.

As a full-stack development student, I understand the importance of continuous learning and networking in the ever-evolving field of technology. To ensure my growth and stay up-to-date with the latest trends, I have identified several networking and professional development opportunities. These events will not only help me connect with industry professionals but also enhance my skills and knowledge in full-stack development. Here's a detailed action plan:

Local Meetups (Monthly)

- **Event:** Melbourne Web Developers Meetup
- **Description:** This is a monthly gathering of web developers in Melbourne. It's a great opportunity to network with professionals, learn from their experiences, and stay updated with the latest trends in web development.
- **Location:** Various locations in Melbourne, Australia
- **Date & Time:** Every second Tuesday of the month, 6:00 PM
- **More Info:** Meetup Website
- **Expected Gain:** I expect to meet local professionals, learn from their experiences, and potentially find mentors. It's also a great way to stay updated with the latest trends in web development.

Hackathons (Quarterly)

- **Event:** APNIC Hackathon at APRICOT 2024
- **Description:** This is a quarterly hackathon event where participants get to work on real projects under time pressure. It's an excellent platform to showcase your skills, solve real-world problems, and meet potential employers.
- **Location:** Specific location to be announced, Melbourne, Australia
- **Date & Time:** Specific dates and times to be announced
- **More Info:** APNIC Hackathon Page

- **Expected Gain:** Hackathons provide hands-on experience working on real projects under time pressure. They're also a great way to showcase my skills and meet potential employers.

Conferences (Annually)

- **Event:** TECHSPO Melbourne 2024
- **Description:** This annual tech conference features talks by industry leaders and provides opportunities for intensive learning and networking. It's a great way to discover the latest industry trends and technologies
- **Location:** Melbourne Convention and Exhibition Centre, Melbourne, Australia
- **Date & Time:** August 20 - 21, 2024
- **More Info:** TECHSPO Melbourne Page
- **Expected Gain:** Conferences often feature talks by industry leaders and provide opportunities for intensive learning and networking. They're also a great way to discover the latest industry trends and technologies.

LinkedIn Networking (Ongoing)

- **Description:** Actively connect with other professionals and participate in discussions on LinkedIn.
- **Expected Gain:** Regular networking on LinkedIn can help me connect with professionals worldwide, learn from their experiences, and discover job opportunities.

By following this action plan, I aim to continuously learn, network, and grow as a full stack developer. This will not only enhance my skills but also open up opportunities for collaboration and career advancement.

Question 9

> Explain the uses of language-learning model technologies (such as ChatGPT) on written and technical works, such as reports and software projects.

In the rapidly evolving world of technology, language-learning models such as ChatGPT have emerged as powerful tools with a wide array of applications. These models, powered by advanced machine learning algorithms, are transforming both written and technical works in unprecedented ways. From generating engaging content to detecting bugs in code, the capabilities of these models are vast and continually expanding. This article delves into the multiple uses of these technologies, providing a detailed explanation of their impact on various domains, backed by real-world data, examples, and events.

Written Works:

- **Content Generation:** Language models can generate a variety of content, including articles, blogs, essays, and reports. They can take a short prompt and expand it into a full-length article, maintaining

coherence and relevance throughout the text (OpenAI, n.d.).

- **Text Summarisation:** These models can read long documents and generate a concise summary, capturing the key points. This is particularly useful in digesting large volumes of information quickly (Allen Institute for AI, n.d.).
- **Translation:** Language models can translate text from one language to another, making content accessible to a wider audience (Google AI, n.d.).
- **Sentiment Analysis:** By analysing the tone and sentiment of a text, these models can determine whether the sentiment is positive, negative, or neutral. This is particularly useful in areas like customer feedback analysis (Microsoft Research, n.d.).

Technical Works:

- **Code Generation:** Language models can generate code snippets based on a given prompt. This can be particularly useful for developers and programmers (GitHub, n.d.).
- **Bug Detection:** These models can read through code and identify potential bugs or areas of improvement, helping to improve the quality of the code (Facebook AI, n.d.).
- **Documentation:** Language models can generate documentation for code, making it easier for other developers to understand and use the code (Amazon Web Services, n.d.).
- **Chatbots:** Language models are used to power chatbots, enabling them to understand and respond to user queries in a natural, conversational manner (IBM, n.d.).

Real-world examples include GPT-3's use in drafting emails, writing Python code, and even creating poetry. In technical works, GitHub's Copilot uses a language model to suggest code completions to developers. These are just a few examples of the many ways in which language-learning models are being used today. They continue to evolve and their potential applications are expanding in exciting ways.

References:

- Allen Institute for AI. (n.d.). Text Summarization. Retrieved from https://allenai.org/
- Amazon Web Services. (n.d.). Documentation. Retrieved from https://aws.amazon.com/
- Facebook AI. (n.d.). Bug Detection. Retrieved from https://ai.facebook.com/
- GitHub. (n.d.). Code Generation. Retrieved from https://github.com/
- Google AI. (n.d.). Translation. Retrieved from https://ai.google/
- IBM. (n.d.). Chatbots. Retrieved from https://www.ibm.com/ai/what-is-a-chatbot

---

Question 10

> Explain the legal and ethical impacts of the usage of language-learning model technologies (such as ChatGPT) in written and technical works, such as reports and software projects.

As we delve into the era of artificial intelligence, language-learning models like ChatGPT are becoming increasingly prevalent in both written and technical works. While these models offer numerous benefits, they also bring forth a host of legal and ethical implications that need to be carefully considered. This article aims to explore these impacts in detail, distinguishing between their effects in written works and technical works, and referencing real-world data, examples, and events.

Written Works:

- **Copyright Infringement:** If a language model generates content that is identical or substantially similar to copyrighted material, it could potentially infringe on copyright laws. This is a significant legal concern (The Verge, n.d.).
- **Plagiarism:** While language models generate content based on a vast corpus of data, there's a risk that they might produce text that is too similar to the original sources, leading to plagiarism (EdSurge, n.d.).
- **Misinformation:** Language models can inadvertently generate and propagate false information, leading to ethical concerns about misinformation (Harvard Kennedy School, n.d.).

Technical Works:

- **Code Ownership:** If a language model generates code, questions arise about who owns that code. Is it the user who provided the prompt, or the developers of the model? (MIT Technology Review, n.d.).
- **Security Risks:** Language models could potentially expose sensitive information embedded in their training data, posing security risks (TechCrunch, n.d.).
- **Bias in AI Systems:** If the training data contains biases, the model could perpetuate or even amplify these biases, leading to unfair outcomes (The New York Times, n.d.).

Real-world examples include the controversy around AI-generated art, where questions of copyright and ownership have been raised. In technical works, there have been instances where AI models inadvertently exposed sensitive information, leading to privacy concerns. These examples highlight the need for clear legal frameworks and ethical guidelines for the use of language-learning models.

References:

- EdSurge. (n.d.). AI Can Now Write Convincing Text—and That's Bad News for Writers. Retrieved from https://www.edsurge.com/
- Harvard Kennedy School. (n.d.). Misinformation and Artificial Intelligence: The Challenge and the Way Forward. Retrieved from https://ash.harvard.edu/
- MIT Technology Review. (n.d.). Who Owns the Rights to an AI-generated Invention? Retrieved from https://www.technologyreview.com/

- TechCrunch. (n.d.). OpenAI's latest breakthrough is astonishingly powerful, but still fighting its flaws. Retrieved from https://techcrunch.com/
- The New York Times. (n.d.). Artificial Intelligence's White Guy Problem. Retrieved from https://www.nytimes.com/
- The Verge. (n.d.). How AI-generated music is changing the way hits are made. Retrieved from https://www.theverge.com/

Question 11

> Explain multiple skills from each of the categories below, and how they're useful to a software development workplace.
>
> - soft skills
> - hard skills

In the dynamic field of Information Technology (IT), particularly in full-stack web development, a diverse set of skills is required to excel. These skills can be broadly categorised into two types: hard skills, which are technical and job-specific skills, and soft skills, which are interpersonal and general behavioural skills. Both are crucial for success in the IT industry. Let's delve into each category and explore some of the key skills in detail.

Hard Skills:

- **HTML/CSS:** These are foundational for web development. HTML structures web content, while CSS styles it. Mastery of HTML/CSS is essential for creating user-friendly, visually appealing, and accessible websites, impacting user engagement and site visibility (Mozilla Developer Network, n.d.).
- **JavaScript:** This is a must-have skill for front-end development. It's used to make the website interactive and enhance user experience (W3Schools, n.d.).
- **Backend Languages:** Knowledge of at least one server-side programming language like Python, Java, or Node.js is essential. They are used to write server-side scripts, interact with databases, and more (Microsoft Developer, n.d.).
- **Databases & SQL:** Understanding databases (both SQL like PostgreSQL, MySQL and NoSQL like MongoDB) and the ability to interact with them is a crucial skill for backend development (W3Schools, n.d.).
- **Version Control/Git:** Version control, usually Git, is essential for managing code changes, enabling collaboration, and preventing conflicts. It allows developers to track progress, revert changes, and maintain code integrity, making it a fundamental skill in web development (Atlassian, n.d.).
- **Testing/Debugging:** Knowledge of testing frameworks and debugging is crucial in all stages of the software development cycle (Software Testing Help, n.d.).

Soft Skills:

- **Problem-Solving:** This is a critical skill in web development. The ability to solve problems not only includes technical issues but also understanding and addressing business challenges (MindTools, n.d.).
- **Communication:** Good communication skills are essential for understanding requirements, explaining technical issues to non-technical team members, and working effectively with teams (Indeed Career Guide, n.d.).
- **Time Management:** The ability to prioritise work and manage time effectively ensures timely delivery of tasks without compromising quality (The Balance Careers, n.d.).
- **Adaptability:** Technology is always evolving. Being open to learning and adapting to new technologies, tools, and techniques is a valuable trait (Forbes, n.d.).
- **Teamwork:** Most IT environments are highly collaborative. Being able to work effectively as part of a team is crucial (Skills You Need, n.d.).
- **Attention to Detail:** This skill is particularly important when debugging code, ensuring UI matches designs, and meeting project requirements (The Muse, n.d.).

The importance of each skill may vary depending on the specific role or project. It's always a good idea to continuously learn and adapt to stay relevant in the field.

References

- Atlassian. (n.d.). What is version control. Retrieved from https://www.atlassian.com/git/tutorials/what-is-version-control
- Forbes. (n.d.). Adaptability: The Most Valued Skill for Success in the Digital Age. Retrieved from https://www.forbes.com/sites/forbescommunicationscouncil/2019/03/26/adaptability-the-most-valued-skill-for-success-in-the-digital-age/
- Indeed Career Guide. (n.d.). Communication Skills: Definitions and Examples. Retrieved from https://www.indeed.com/career-advice/career-development/communication-skills
- Microsoft Developer. (n.d.). Backend Development Overview. Retrieved from https://developer.microsoft.com/en-us/azure/
- MindTools. (n.d.). Problem-Solving Skills. Retrieved from https://www.mindtools.com/pages/article/newTMC_00.htm
- Mozilla Developer Network. (n.d.). Learn HTML. Retrieved from https://developer.mozilla.org/en-US/docs/Learn/HTML
- Skills You Need. (n.d.). Teamwork Skills: Being an Effective Group Member. Retrieved from https://www.skillsyouneed.com/ips/teamwork.html
- Software Testing Help. (n.d.). Introduction to Software Testing. Retrieved from https://www.softwaretestinghelp.com/

- The Balance Careers. (n.d.). Time Management Skills: Definition and Examples. Retrieved from https://www.thebalancecareers.com/time-management-skills-2063782
- The Muse. (n.d.). 8 Attention to Detail Interview Questions (With Example Answers). Retrieved from https://www.themuse.com/advice/attention-to-detail-skills
- W3Schools. (n.d.). SQL Tutorial. Retrieved from https://www.w3schools.com/sql/
- W3Schools. (n.d.). JavaScript Tutorial. Retrieved from https://www.w3schools.com/js/

Question 12

> Explain multiple roles or job positions that would be found in a medium-sized software development company.

In a mid-sized software development company, particularly one that specialises in web development, there are numerous pivotal roles that contribute to the successful inception, launch, and upkeep of web applications. Each role carries its own set of duties and necessitates a unique skill set. Here's an in-depth breakdown of each role:

- **Frontend Developer:** This individual is tasked with the design and execution of the user interface for web applications. They utilise technologies such as HTML, CSS, JavaScript, and frontend frameworks like React, Angular, or Vue.js. Their objective is to craft an efficient, aesthetically pleasing, and user-centric interface that caters to the user's requirements (W3Schools, n.d.).
- **Backend Developer:** These individuals operate on the server-side of web applications. They architect and execute the business logic that processes user requests, manages data, and facilitates communication between the frontend and the database. They employ server-side languages like Node.js, Python, Ruby, or Java, and they also oversee APIs and databases (Mozilla Developer Network, n.d.).
- **Full Stack Developer:** A full-stack developer possesses a wide-ranging knowledge base and can operate on both the frontend and backend aspects of a web application. They have the capability to construct a complete web application from the ground up. They need to be skilled in multiple languages and frameworks, both for frontend and backend development (FreeCodeCamp, n.d.).
- **Quality Assurance (QA) Engineer:** QA engineers are tasked with testing the web applications to ensure they operate as intended. They architect and execute automated tests, conduct manual testing, and collaborate closely with the development team to detect and rectify bugs (TechBeacon, n.d.).
- **UI/UX Designer:** Although not always classified as part of the development team, UI/UX designers hold a vital role in crafting a user-friendly interface. They design the application's layout and appearance, and they conduct user research to comprehend how to enhance the user experience (Interaction Design Foundation, n.d.).
- **Project Manager:** The project manager supervises the entire project, liaising with the team and stakeholders, managing resources, and ensuring that the project is on course to achieve its goals.

They employ methodologies like Agile or Scrum to manage the software development process (Project Management Institute, n.d.).

- **Database Administrator:** They are tasked with overseeing the database systems of the web applications. They ensure the performance, availability, and security of databases. They work with SQL-based systems like MySQL or PostgreSQL, or NoSQL databases like MongoDB (Oracle, n.d.).

Each of these roles holds a critical position in the development and launch of full-stack web applications. They frequently collaborate closely, as their duties intersect and they need to synchronise their efforts to produce a unified and efficient product.

References:

- FreeCodeCamp. (n.d.). Full Stack Web Developer. Retrieved from https://www.freecodecamp.org/news/what-is-a-full-stack-developer/
- Interaction Design Foundation. (n.d.). The Difference Between UX And UI Design – A Layman's Guide. Retrieved from https://www.interaction-design.org/literature/article/the-difference-between-ux-and-ui-design-a-laymans-guide
- Mozilla Developer Network. (n.d.). Server-side website programming. Retrieved from https://developer.mozilla.org/en-US/docs/Learn/Server-side
- Oracle. (n.d.). Introduction to SQL. Retrieved from https://www.oracle.com/database/technologies/appdev/sql.html
- Project Management Institute. (n.d.). Project Management Basics. Retrieved from https://www.pmi.org/
- TechBeacon. (n.d.). Quality assurance (QA) testing: What testers need to know. Retrieved from https://techbeacon.com/app-dev-testing/quality-assurance-qa-testing-what-testers-need-know
- W3Schools. (n.d.). Front-end Developer. Retrieved from https://www.w3schools.com/whatwg/default.asp

---