1st Ed.

Exploring Technology's Essence

Beyond Computers

~ ***Oladosu Emmanuel Oluwafunso***

(Pioneer GoogleDSC-Babcock Web Track Lead 2022)

(Former GoogleDSC-Babcock Co-Lead 2023 *Stepped down*)

(Established the Application Development Track at GoogleDSC-Babcock 2023)

(Established the Backend Systems Track at GoogleDSC-Babcock 2023)

(Svelte Society Nigeria Lead *since* 2023)

Online Version

PDF Versions

ABSTRACT

Initially planned as a two day event, circumstances led me to transform this into an online book. However, this adaptation offers distinct advantages. As an online book, I can update and enhance its content whenever new information becomes available. Readers can revisit and reference it at their convenience, which is a significant benefit. The only drawback is that some topics may necessitate face-to-face discussions for better comprehension. I will strive to present complex ideas as simply and directly as possible.

Why did I invest my time in writing this? The term "technology" has been misconstrued by the influence of social media (especially Twitter 🙄). Many Nigerians don't take the initiative to investigate and understand the true meanings of words; instead, they make hasty assumptions based on how they observe others using these terms. To drive Nigeria's progress, it's essential to grasp the full scope of "Technology." It extends far beyond computers; it encompasses a much broader spectrum. When you hear "tech," the immediate association is often with computers, and nowadays, it's even narrowed down to just software applications. This narrow view reveals our ignorance to the world, and it's evident why our progress remains stunted. This write-up is intended for those who genuinely have a passion for technology. If you're only in it for financial gain and job prospects, focusing on trendy tools that become obsolete over time, then you may want to stop reading and proceed to learn React or Flutter. This book is for the true change-makers.

I've stepped back from assisting others, as even in group chats, despite members expressing a desire for more interaction and discussions, I've received countless private messages telling me to be silent and stop engaging, helping, and discussing,

as it's deemed stupid to share information, this I find rather funny. It seems there's an unspoken rule that suggests the more reserved you are, the more respect you earn. While I'm indifferent to these norms, this online book is the only medium through which I will share my insights from now on. I've retired from my expressive ways. Nevertheless, if you encounter any issues within the book, feel free to reach out to me on Twitter @lit.nuel or by clicking or tapping my name beneath the title. Alternatively, you can submit a pull request to the suggestions.md file. Prepare for an enlightening journey!

Suggested soundtrack in this edition for those friends who enjoy reading accompanied by music.

TABLE OF CONTENTS

If you're reading this for the first time, I strongly encourage you not to skip any section. Take your time to go through it, as there's no need to rush. This book is intentionally brief. It's worth noting that the online version may not display page numbers. However, you can easily search for specific content using the "Ctrl + F" or "Cmd + F" function.

DEFINITIONS

I initially contemplated naming this section "Building Blocks" or perhaps "Ingredients," but the choice of title is inconsequential. I trust you enjoy the captivating image of the flames. Fire! Haha, all right. My fundamental belief is in the value of learning through the grasp of concepts. While many people may be eager to dive into practical applications immediately, I prefer to first understand the "what," "why," and occasionally, the "how" (with "how" referring to the origin and principles, not just the method of application). In most cases, comprehending the "what" and "why" is adequate to provide one with a strong foundation for any endeavour.

To illustrate this point, I often use the analogy of two bricklaying apprentices. The first apprentice has no concept of what a house is, perhaps having never seen one. Yet, they are tasked with building one. He observes the master and receives guidance throughout the construction process, becoming skilled at replicating the task over time. However, when faced with a new challenge or a task that deviates from the familiar, he tends to struggle. While he may eventually find a solution through trial and error, it is unlikely to be efficient. On the other hand, the second apprentice begins with an understanding of what a house is and why it is needed. He has seen various styles of houses and has examined the fundamental components of a typical house, such as bedrooms, a living room, a dining area, and a bathroom. He has even delved into finer details, understanding the significance of door positioning and ventilation, among other things. Guided by the master, he constructs a house. With this solid foundational knowledge, the second apprentice is better equipped to solve new challenges swiftly, drawing upon his deep understanding of what a building entails.

In the end, the second apprentice excels significantly, emphasising the importance of grasping the core concepts. This approach aligns with the fundamental purpose of education, though it may not always be executed perfectly. Nevertheless, it highlights why I still value formal education, as it provides the "why" behind what I'm learning and why I'm there. While this deviates slightly from the main topic, "School no be scam." I want to clarify that success doesn't exclusively hinge on formal education. My intention is to emphasise the significance of concept-based learning as a catalyst for growth. It provides a sturdy foundation, akin to the idea of "standing on the shoulders of giants" as famously articulated by Isaac Newton.

1.1 Science

Why begin here, you might wonder? Were you expecting us to dive into the topic of information-processing boxes right away? Well, that's precisely why you find yourself reading this now. Science is a systematic way of exploring and understanding the natural world around us. It's like a toolkit that helps us make sense of the mysteries of the universe. At its core, science involves a process of observation, questioning, experimentation, and analysis. First, scientists observe something in the world, like the way plants grow or the behaviour of celestial bodies. They ask questions about what they see, formulating hypotheses or educated guesses about how things work. Then, they conduct experiments to test these hypotheses, collecting data and evidence along the way.

The key to science is being objective and open-minded, always ready to revise our understanding based on new evidence. This process is often called the scientific method. It helps us build a body of knowledge that's reliable and consistent. Science is incredibly diverse, covering various fields such as biology, chemistry, physics, astronomy, and many more. These fields each have their own set of principles and methods, but they all share the common goal of uncovering the fundamental laws that govern the natural world. By breaking down complex phenomena into smaller, more understandable parts, science enables us to make better decisions, solve problems, and improve our lives in countless ways. In essence, science is a powerful tool that empowers us to better understand and navigate the world we live in. Well, you see, science isn't just some mysterious academic magic or a bunch of laboratory explosions and all those things you might not have been curious about. I get it, you're more interested in enjoying good food, earning money, and having a good time – what we call "jaiye." The truth is, not everyone is deeply concerned or driven by the intricacies of the world, and hey, I'm a Nigerian too, so I totally get where you're coming from! At this point, it should be clear to you that the author of this book is a 20-year-old.

1.2 Technology

Before I begin it's important to note that technology isn't limited to just computers. It goes beyond apps, robots, AI, operating systems, and hardware (Haha, got you at the last one!), even electrical devices and electricity (since you're not yet convinced). I began by emphasising that it's not solely about computers, but your mind may have gravitated towards what you're accustomed to. Have you considered the impact of technology in medicine? At this point, you might be starting to see the broader picture. You might even find yourself wishing you'd paid more attention during those Basic Science classes in Primary and Secondary School – I certainly don't regret it.

Technology, in its simplest form, is the application of scientific knowledge for practical purposes (Yes, applying science). It's like a giant toolbox humanity has been building over centuries to solve problems, make life easier, and connect us in unimaginable ways. The history of technology is a journey that spans millennia. It started with the earliest humans using tools like stone blades and fire to survive. Over time, we've seen monumental innovations, from the invention of the wheel and the printing press to the steam engine and electricity. These breakthroughs marked major technological revolutions, shaping the world as we know it.

As time marched on, technology evolved in remarkable ways. We moved from mechanical contraptions to the digital age, with computers, smartphones, and the internet transforming the way we work, communicate, and live. Today, technology comes in various types and forms, each serving specific purposes. Information technology, for instance, powers our digital world, encompassing computers, software, and networks. Biotechnology involves manipulating living organisms for medical and scientific advances. Aerospace technology propels us into the skies and beyond, while nanotechnology deals with the tiniest particles, opening new frontiers in materials and medicine. Transportation technology, from the invention of the wheel to self-driving cars, has revolutionised how we move around. Renewable energy technology addresses our growing environmental concerns, harnessing the power of the sun, wind, and water. The list goes on, and the types and forms of technology continue to expand. In our modern age, technology touches every aspect of our lives, from healthcare and entertainment to transportation and communication. It's a dynamic force that shapes our world, and understanding its history and the various branches it encompasses is essential for navigating the ever-evolving landscape.

1.3 Engineering

I can't help but chuckle at this. Exploring the myopic perspective of engineering, even among educated Nigerians, is quite an interesting topic to delve into. I'm willing to bet that when you initially thought of engineering, images of physical creations like roads, bridges, and machines immediately came to mind. But, hey, don't worry, you're not alone in this misconception – even foreigners often grapple with comprehending the full scope of engineering, especially those who haven't had the opportunity for formal education, unlike us fortunate individuals.

Engineering is the **creative** process of DESIGNING, DEVELOPING and MAINTAINING tangible or intangible solutions to real world problems. You're welcome to express your frustration. Engineering is a vast field encompassing diverse specialisations such as aerospace, environmental, biomedical, and software engineering, each focusing on solving unique challenges. Engineering heavily relies on technology, and there are various forms and types of technology integrated into engineering practices. Mechanical technology plays a pivotal role in designing machines and devices that power our world. Electrical and electronics technology drives our digital age, enabling communication and automation. Civil engineering utilises construction technology to build infrastructure like bridges and buildings. Materials technology explores new materials for various applications. Biotechnology merges biology and technology to develop medical advancements. Information technology is the backbone of modern computing and data management.

1.3.1 Designing

Designing is like creating a blueprint for a project. It's where engineers use their creativity and knowledge to plan how something should be built. They consider what it should look like, how it will function, and what materials should be used. Imagine it as the detailed sketch before building a house – it guides the entire construction process. It may come as a surprise that when I mentioned design, I wasn't specifically referring to graphics or visual design. In engineering, it's apparent that visual design is merely one facet of the broader field of design. Design involves not only how a product looks but also how it works, the materials used, its structural integrity, and its ability to meet certain criteria or standards. This includes aspects such as ergonomics, manufacturability, safety, and efficiency, which go beyond visual aesthetics.Various forms of visual design are employed within engineering to communicate ideas and conceptual plans effectively. Design, even when not exclusively related to engineering, encompasses far more than just visualisation and graphics. It encompasses the process of intentional planning and creation with a specific purpose in mind. This is an area where some educational institutions may have overlooked the comprehensive nature of design.

1.3.2 Developing

Let's expedite through the remaining steps so I can finally catch some well-deserved shut-eye. Just for context, I'm composing this at the rather early hour of 3 am. Following the meticulous design phase, engineers transition into the development stage. Here's where the blueprint truly comes to life. They roll up their sleeves, whether building physical structures, writing lines of code, or assembling complex projects, all with the shared goal of breathing life into the original design. This dynamic phase is where ideas transform into reality, and it's a crucial step in the journey of creation.

1.3.3 Maintaining

Once a project is up and running, engineers must keep it in good shape. Maintenance involves regular checks, repairs, and updates to ensure that the system or structure continues to work as intended. Think of it like taking care of your car – changing the oil, fixing any issues to keep it running smoothly.

TECHNOLOGY

2.0 Definition

Technology is the application of science. As I mentioned before, this concept isn't limited to computers alone. Technology encompasses a wide array of fields and industries. Medicine is a form of technology, where advanced tools and techniques are employed for healthcare. Automobiles are a prime example of technological innovation, and even everyday aspects of life like housing and food processing involve technology for improvement and efficiency. Technology is truly ubiquitous, shaping our world in countless ways beyond just the digital realm. Let's set aside the narrower definition of 'tech' often seen on social media, which tends to confine it to application software development. Instead, let's be mindful that technology goes well beyond those boundaries. It's completely okay to interact with, accept, and perhaps even play along with these misunderstandings, but it's essential to be aware of the actual facts.

2.1 Some Forms of Technology

2.1.1 Computing Technology (Information Communication Technology)

Computing technology, also known as Information Communication Technology (ICT), forms the backbone of our interconnected and digitised world.

2.1.2 Biotechnology

Biotechnology involves using biological systems, organisms, or derivatives to develop or create new products or processes for various industries, including medicine, agriculture, and environmental management. It aims to harness biological processes for practical purposes e.g. Genetic Engineering, Pharmaceutical Technology, Bioinformatics.

2.1.3 Transportation Technology

From automobiles and aeroplanes to high-speed trains and even autonomous vehicles, transportation technology keeps the world moving efficiently. It's responsible for making travel faster, safer, and more convenient, revolutionising how we connect with the world e.g Automotive Technology, Aviation Technology, High-speed Rail Technology, Maritime Technology.

2.1.4 Energy Technology

Harnessing power from various sources. This includes traditional energy sources like fossil fuels, as well as renewable technologies such as solar, wind, and hydropower. These technologies help generate electricity, heat homes, and drive the engines of industry e.g. Traditional Energy Sources, Renewable Energy Technology, Smart Grid Technology.

2.1.5 Manufacturing Technology

The tools, processes, and systems used to create goods. From 3D printing and CNC machines to automated assembly lines, these technologies improve production efficiency, product quality, and customization options.

2.1.6 Agricultural Technology

Agricultural technology has revolutionised the way we produce food. It involves improving yields, and food quality e.g. Precision Agriculture, Genetic Modification in Agriculture.

2.1.7 Environmental Technology

As environmental concerns grow, so does the importance of environmental technology. This includes solutions for waste management, pollution control, renewable energy, and conservation efforts. These technologies aim to mitigate the environmental impact of human activities e.g. Waste Management Technology, Pollution Control Technology, Renewable Energy Solutions.

2.1.8 Space Technology

Encompasses the equipment and systems required for space exploration and satellite communication. It involves rockets, space probes, satellite networks, and telescopes, pushing the boundaries of our understanding of the universe e.g. Rocket Technology, Satellite Technology, Space Exploration Systems.

2.2 Computing Technology

In the realm of computing technology, we delve into the multifaceted landscape of hardware, software, and networks that collectively shape our modern technological ecosystem. At its core, computing technology involves the development and use of systems that facilitate the processing and transmission of information. Hardware, comprising devices like computers, servers, and smartphones, provides the physical infrastructure, while software, ranging from operating systems to applications, dictates how these devices function. The intricate web of networks, both wired and wireless, serves as the connective tissue, enabling seamless communication and data exchange on a global scale. In essence, this domain explores the dynamic interplay between hardware and software, emphasising the pivotal role of computing technology in driving innovation, communication, and problem-solving across diverse sectors, from business and education to healthcare and entertainment. Understanding computing technology is not just an exploration of gadgets and codes; it is an immersion into the transformative force that underpins the digital age, shaping the way we live, work, and connect with the world around us.

2.2.1 Information Technology

Information Technology, often abbreviated as IT, is a broad field that revolves around the acquisition, storage, processing, and dissemination of information. In essence, IT is the application of computers to store, retrieve, transmit, and manipulate data. This multifaceted discipline encompasses a range of activities, from managing complex databases and developing software applications to maintaining network infrastructure. One of the key pillars of IT is data management. IT professionals design and implement databases to store vast amounts of information efficiently. This data can include anything from customer records to financial transactions. IT specialists are responsible for ensuring the security and integrity of this data, implementing backup systems, and developing protocols for data retrieval.

2.2.2 Communication Technology

Communication Technology is a specialised field within the broader spectrum of computing technology, focusing on the means and methods by which information is transmitted and received. It encompasses a wide range of technologies that facilitate the exchange of data, voice, and multimedia content across various channels and platforms. At its core, Communication Technology involves the development and management of systems that enable seamless communication. This includes traditional methods like telephony and broadcasting, as well as modern innovations such as satellite communication, fibre optics, and wireless networks. These technologies allow for real-time interaction on a local or global scale. One of the pivotal elements of Communication Technology is the internet. The internet serves as a vast network connecting individuals, businesses, and organisations worldwide. Email, instant messaging, social media, and video conferencing are all products of Communication Technology that have revolutionised the way we connect and share information. Telecommunications engineers and professionals play a vital role in developing and maintaining the infrastructure that supports communication systems. This involves designing and optimising networks to ensure reliability, speed, and accessibility. Additionally, the field encompasses the study of protocols, data compression techniques, and encryption methods to secure the transmission of information.

Communication Technology is not only about hardware and infrastructure; it also includes software applications and platforms that enable communication. From messaging apps to video conferencing software, these tools contribute to the ease and efficiency of information exchange. Communication Technology is the driving force behind our interconnected world. It shapes how we communicate, collaborate, and share ideas, fostering a global network where information flows seamlessly across borders and cultures.

COMPUTING

3.0 Definition

Computing refers to the process of using and manipulating computer technology to perform various tasks, solve problems, and process data. It involves the use of hardware, software, algorithms, and data to execute computations, store and retrieve information, and carry out specific functions. At its core, computing involves the use of digital devices, to process information and perform operations. Computing has become an integral part of numerous industries and aspects of daily life, influencing how we work, communicate, and solve problems in the modern world.

3.1 Jargon

3.1.1 Compute

"Compute" is a verb that refers to the process of performing calculations or carrying out mathematical operations using a computer or computational device. It involves using the processing power of a computer to solve problems, analyse data, or execute specific tasks. The term is often associated with the ability of a computer to perform mathematical or logical operations quickly and accurately.

3.1.2 Computer

A computer is an electronic device that processes data and performs tasks according to a set of instructions called programs. It can store, retrieve, and process data, making it a versatile tool, from simple calculations to complex simulations.

3.1.3 Computer Science

Computer science is the academic discipline that explores the theory, development, and application of computers and computational systems. It encompasses a wide range of topics, including algorithms, data structures, programming languages, software development, artificial intelligence, and the design and analysis of algorithms. Computer scientists work on advancing the understanding of computation and developing innovative solutions to complex problems.

3.1.4 Computer Engineering

Computer engineering is a branch of engineering that integrates principles of computer science and electrical engineering to design and develop computer systems. Computer engineers focus on the development of both hardware and software components, including microprocessors, memory systems, and communication interfaces. They work on creating efficient and reliable computer systems that meet the evolving needs of technology. Computer engineering spans the design of individual computer components to the architecture of entire computing systems.

3.2 Two-Sides

3.2.1 Hardware

Hardware refers to the physical components of a computer system or any electronic device. These tangible components are the machinery and equipment that make up the computer and enable it to function. Key hardware components include the central processing unit (CPU), which serves as the brain of the computer, memory (RAM) for temporary data storage, storage devices such as hard drives or SSDs for long-term data storage, input devices like keyboards and mice, output devices such as monitors and printers, and various internal and external peripherals. Hardware interacts with software to execute commands and perform tasks, forming the essential foundation of any computing system. The design, manufacturing, and configuration of hardware components play a crucial role in determining a computer's capabilities, performance, and overall functionality.

3.2.2 Software

Software, in contrast, is intangible and consists of a set of instructions, programs, and data that directs the operation of a computer or electronic device. It is written in programming languages and includes operating systems, application software, utility programs, and system software. Software serves as the intermediary between users and hardware, translating commands into actions that the tangible components can execute. Its intangible nature allows for flexibility, adaptability, and continuous modification to meet evolving user needs in computing systems. The collaboration between tangible hardware and intangible software is fundamental to the effective functioning of electronic devices.

SOFTWARE

4.1 Definition of terms

4.1.1 Program

A set of instructions to be executed by a computer to perform a particular task.

4.1.2 Programming

The process of specifying a set of instructions (program) that a computer can interpret and execute.

4.1.3 Programming Language

A programming language is a formal system comprising a set of rules and syntax used to write computer programs.

4.1.4 Software

Software refers to a collection of related programs.

4.1.5 Software Development

It involves a series of systematic steps to produce high-quality software that meets specified requirements and addresses user needs.

4.1.6 Software Engineering

Software engineering is a disciplined approach that employs principles, methodologies, and best practices to the DESIGN, DEVELOPMENT, testing, and MAINTENANCE of software.

4.2 Types/Categories

Software is categorised into system and application software. I will just provide a basic view.

4.2.1 System Software

A type of software designed to facilitate the operation and functioning of computer hardware. It provides a platform for running application software and manages various hardware components. Key functions of system software include controlling the computer's hardware, managing memory and file systems, providing security features, and facilitating communication between hardware and application software.

I. Operating Systems: They act as an intermediary between the computer hardware and application software, facilitating communication and resource allocation. Examples include Windows, macOS, Linux, and Android. Operating systems handle tasks like process management, memory management, file system management, and user interface interactions.

II. Utilities: This category of systems software provides essential support for the computer infrastructure. Utilities play a crucial role in assisting the operating system with various tasks, including ensuring data security, to maintain the efficient operation of the computer.

III. Drivers: Integrated into the operating system, facilitates communication between the operating system and a hardware device. The driver instructs the system on how to interact with the hardware. Hardware devices like printers, scanners, card readers, and modems often require a driver to function properly.

IV. Firmware: Operates a hardware device by providing instructions on how it should perform. It collaborates with device drivers to ensure that hardware functions as anticipated. Unlike a driver, which is part of the operating system, firmware is embedded directly in the hardware itself.

V. Programming language processors/translators: Facilitate the execution of programs written in programming languages. These processors play a crucial role in translating human-readable code into machine code or an intermediate code that a computer's central processing unit (CPU) can understand and execute. There are three main types of programming language processors: Compilers, Interpreters and Assemblers.

4.2.2 Application Software

Sure, before we start, I want to emphasise for the millionth time in my life that apps aren't just for mobile devices. Many individuals express a desire to transition into mobile development, driven by the aspiration to create 'apps'. This way of thinking is wrong and quite common, especially in Nigeria. Have you ever heard of a web app? Yes, that student portal you use and even facebook.com are examples of web apps. Pretty surprising, huh? Looks like someone might have missed a few lessons with their computer science teacher back in primary school. You're familiar with desktop apps too, right? No comments.

Apps are software required to perform specific tasks. They are dependent on the Operating system of the device. Consequently, the differentiation among app types hinges on the frontend or the specific methodology employed for frontend development. The backend operates on the web, hosted in the cloud, appearing distinct from the user-facing app or frontend. Consider this: applications like calculators, word processors, and offline games lack a backend but still offer functionality. Where does this functionality reside? It resides on the frontend, thanks to client-side processing – an aspect familiar to those who've delved into books and paid attention in class. The component of an application that users directly engage with is primarily influenced by client-side processing. Contrary to the notion that the frontend is merely about aesthetics and design, individuals who have experience in building standalone frontend applications understand its central role. Feeling provoked? Go ahead, express your frustration because it challenges the misconceptions you didn't bother fact-checking.

Applications can be broadly categorised into various types based on their platforms:

I. Desktop & TV Apps: For Windows, applications are built with C# using the .NET framework. MacOS relies on Swift or Objective-C for application development. For Samsung Smart TVs using Tizen, developers often work with JavaScript and C/C++. On the other hand, LG Smart TVs running webOS primarily involve JavaScript in the development process.

II. Mobile (Phones, Tablets, Wearables) Apps: For Android OS, applications are crafted using Java or Kotlin. Conversely, iOS, Apple's mobile operating system, employs Swift or Objective-C for its applications.

III. Cross Platform (Web, Hybrid) Apps: Cross-platform apps aim to minimise dependency on a specific operating system, allowing them to be more versatile and accessible across different devices and platforms.

    A. Web applications find their foundation in JavaScript, the language for web development. As a versatile client-side scripting language, JavaScript underpins the dynamic and interactive nature of web applications. Popular frameworks like React.js, Angular and Vue.js are often employed to streamline the development process.

        1. A backend application is a web app, constructed with server-side technologies like Java, Python, JavaScript, GoLang, or PHP, orchestrates data interactions. Hosted on servers, whether on-premises or in the cloud through services like AWS, Azure, or Google Cloud, it manages data processing, storage, and ensures security, scalability, and reliability. Beyond being an information repository, the backend facilitates communication, offering APIs for seamless interaction between frontends (web or native applications). In essence, it serves as a central hub, providing consistent functionality for mobile, desktop, and web applications, promoting efficiency in development and updates.

    B. Cross-platform desktop applications, seeking universality, find expression in languages like Java and frameworks using Javascript like Electron.js.

    C. For cross-platform mobile applications, JavaScript serves as a versatile language, utilised in frameworks such as React Native and Cordova, and Dart programming language for Flutter.

IV. Console Apps: Console apps are software applications designed to run on gaming consoles such as PlayStation, Xbox, or Nintendo consoles. These apps may include games, entertainment applications, or utilities tailored for the gaming environment. Console apps are developed to work specifically on the operating system or platform associated with the gaming console.

Applications can be broadly categorised into three types based on their functions:

I. Frontend (standalone) applications, also known as client-side applications, are responsible for user interfaces and run on native devices (mobile, desktop, etc) or directly in web browsers. They utilise technologies like JavaScript, Java, Swift, Objective-C e.t.c. Examples include word processors, calculators, notepads, offline games and many more.

II. Backend applications, on the other hand, are web applications that operate on the server side, managing data, processing logic, and interacting with databases. They expose APIs for communication with frontend applications and are constructed using server-side languages like Java, PHP, JavaScript and Python.

III. Full-stack applications encompass both frontend (mobile, desktop and web) and backend (web) components, providing end-to-end solutions. Unlike standalone frontend applications, full-stack counterparts manage data processing, server-side logic, and intricate business operations. They oversee user authentication, database management, and more. Notable examples encompass social media giants like Facebook, Instagram, and Twitter, content streaming platforms such as Netflix, YouTube, and Spotify, e-commerce behemoths like Amazon and eBay, and real-time messaging apps like WhatsApp and Telegram.

SOFTWARE ENGINEERING

5.0 Definition

DESIGN, DEVELOPMENT and MAINTENANCE of software systems. No, I don't mean system software. In a general context, a system is a collection of interacting or interrelated components working together to achieve a common objective or purpose. It's crucial to distinguish system software, which oversees computer systems, from a software system – a compilation of interconnected software components. Feeling confused? Give it another read, no need for that puzzled expression. It should click now, you'll see.

5.1 Breakdown

Let's break down software engineering into its key phases: design, development, and maintenance, making it simple to understand.

5.1.1 Software Design

Design is like creating a blueprint before building a house. In software engineering, it's about planning how the software will look and function. Here, we decide on the software's structure, features, and how users will interact with it. It's like sketching the layout before turning it into a real application.

I. Conceptual Design: Establishing the overall structure, architecture, and high-level functionality of the software system. During this phase, designers outline the fundamental concepts, major features, and the interaction between system components. It's a conceptual roadmap for the software.

II. Detailed Design: Delving into the specifics (low-level) of each component defined in the conceptual design. It involves creating detailed specifications for coding and implementation. Designers work on detailed algorithms, data structures, data flow diagrams, and interface specifications. This phase provides a comprehensive guide for developers to follow during coding.

III. Interface Design: Creating the user interface (UI) elements that users will engage with during interactions with the software. This focuses on the visual aspects, designing layouts, buttons, forms, and other elements. The goal is to ensure a user-friendly and aesthetically pleasing interface.

IV. Experience Design: Involves crafting every touchpoint a user has with the software, from their initial interaction to the completion of their tasks. It considers the emotions, perceptions, and overall satisfaction of the user throughout the entire process. Interaction design within Experience Design deals with the responsiveness and intuitiveness of user interactions. It ensures that users can easily navigate and accomplish tasks. Experience designers deeply understand user needs, behaviours, and motivations. They go beyond the surface to design solutions that resonate with users on a deeper level, ensuring the software meets their expectations and provides value.

5.1.2 Software Development

Development is like building a house using a blueprint. In software, it's about writing the actual code that brings the design to life. Developers write lines of code, create functions, and ensure that the software works as intended. It's the hands-on process of turning ideas into a functional program.

I. Coding: is the heart of the development phase, where developers translate the design and specifications into actual lines of code. Developers write code using programming languages, creating functions, algorithms, and logic based on the design specifications.

II. Implementation: involves putting the coded components together to create the complete software application. Developers integrate different modules, ensuring that they work seamlessly together. This phase transforms the individual pieces of code into a functioning whole.

III. Testing: is a crucial aspect of development to ensure the software functions correctly and meets the specified requirements. Testing the software for bugs, errors, and adherence to design specifications. This phase identifies and resolves any issues before the software is deployed.

IV. Debugging: is the process of finding and fixing errors or issues identified during testing. Developers identify and eliminate bugs in the code to enhance the software's stability and performance. Debugging is an iterative process until the software meets quality standards.

V. Optimization: focuses on improving the efficiency and performance of the software. Developers fine-tune the code, enhance algorithms, and optimise resource usage to ensure the software runs smoothly and meets performance requirements.

5.1.3 Software Maintenance

Maintenance is like taking care of the house after it's built. In software, it's about ensuring the program continues to work well over time. This phase involves fixing any issues that arise, updating the software to adapt to changes, and sometimes adding new features. It's like keeping the house in good condition, making sure everything runs smoothly.

5.2 Forms of Software Engineering

5.2.1 Application Engineering

'Tech Twitter' predominantly centres around this form. If I speak I am in trouble.

5.2.2 System Software Engineering

Involves the development, maintenance, and optimization of essential software that enables and manages the functionality of computer hardware and provides a platform for application software.

5.2.3 Data Engineering

Handling the design and management of data infrastructure and databases.

5.2.4 Machine Learning Engineering

Developing software solutions that involve machine learning algorithms and artificial intelligence.

5.2.5 Cloud (Computing) Software Engineering

Developing and managing applications that leverage cloud computing services involves the utilisation of cloud-based resources, platforms, and services to build, deploy, and operate software applications. Cloud computing offers a scalable and flexible infrastructure, allowing developers and businesses to offload computational tasks, storage, and other services to remote servers accessible over the internet. This approach replaces traditional on-premises infrastructure.

5.2.6 Automation Engineering

Implementing automated processes to enhance efficiency in software development and operations.

5.3 Application Software Development

Let's narrow it down to the implementation (development) aspect and steer away from the broader engineering perspective. I mean, it was originally intended to be an event for the Google DEVELOPER Student Club in Babcock for a reason. If you've been following along from the beginning, you should already have an understanding of this subsection. So, I'll proceed to delve into its fundamental types.

5.3.1 Front-end Development

There's a lot of misinformation and misunderstanding surrounding this role, leading to disrespect and confusion. Individuals often express sentiments such as, "There's no logic involved in the frontend; I'm switching to the backend." (someone who can't create a simple to-do list application.) Let's take a step back before the internet era. How were software applications structured? They were predominantly offline! And what handled the processing (logic)? It was the frontend! How? Through client-side processing! Fast forward to the present day. Have you ever created a calculator, especially on the web? What's the purpose of JavaScript – colours and boxes? Now, consider the frontend of native applications, both for mobile and desktop. Is it built with HTML 😭? No, it involves programming languages such as Java, C#, Objective-C, etc. Have you ever developed a word processor using C# or a desktop game with Java? No comments.

Front-end development is commonly known as client-side development. To be candid, they wouldn't have labelled it as "frontend" in the past because there was no backend; applications were straightforward. Then the internet emerged, demanding information sharing, and voila, we now have two distinct components of applications. Many people will still argue that client-side processing is still considered backend, just not on the web but within the application itself. This often confirms their misunderstanding of these concepts, and it's a perspective commonly voiced by web developers who have not delved into building desktop applications before.

Understanding the significance of the frontend is crucial because it constitutes the actual application. To illustrate, consider a building analogy: the frontend is the construction, while the backend is akin to the electrical and plumbing systems, though not all structures require these. Picture a gazebo – not a perfect analogy, but it conveys the point. The frontend varies across platforms. Take Spotify, for example, which has web, mobile, and desktop apps. Although the look, feel, and codebase differ, the accessed information remains consistent. Your account is universal across platforms; there's no need for a new one on each app. The backend functions like an external application stored in the cloud on the server. In contrast, the frontend is downloaded or installed on the user's device or accessed via the browser, depending on the OS and browser environment. In the case of Spotify, multiple apps access data from the server's backend. The frontend is the tangible application for user interaction. It's important to note that web apps, like the backend, require an internet connection, as they are stored on a server in the cloud. However, advancements like Progressive Web Apps (PWAs) offer alternatives, but we won't delve into that here. So, the frontend goes beyond aesthetics; consider Java Swing or JavaFX developers.

5.3.2 Back-end Development

The primary purpose of the backend is to enable 'Data Sharing.' While I didn't claim that this is its sole function, I'm not inclined to engage in debates with individuals unwilling to explore educational resources. If you comprehend this concept, let's progress. Picture this: you've successfully constructed your frontend, but now you aim to enable users to interact and share information online – posting, creating profiles, and collaborating. This is when a Backend becomes essential. Recall when I mentioned that the frontend involves client-side processing? Well, the backend engages in server-side processing because it constitutes the application component stored on a server (another computer). Your computer, the client, simply retrieves information from the server over the internet. So, the next time someone asserts that there's no logic on the front-end, you can correct that misconception. Indeed, the backend is crucial, especially for web applications. Web applications hosted on the internet primarily focus on data sharing and collaboration. The backend plays a pivotal role in facilitating these functionalities, managing data storage, processing user requests, and ensuring seamless interactions. It serves as the engine behind the scenes, powering the data-driven and collaborative nature of web applications. Without a robust backend, the dynamic and collaborative features that users experience on the frontend would not be possible.

Backend developers are responsible for designing and implementing the server-side logic, which serves as the core functionality driving applications. This encompasses handling user requests, processing data, and executing the underlying business logic. Integral to backend development is effective database management, with developers working with relational databases (like MySQL, PostgreSQL) or NoSQL databases (like MongoDB) to store and retrieve data efficiently. A critical aspect involves the creation of Application Programming Interfaces (APIs), allowing different software systems to communicate and enabling smooth interaction between the frontend and backend. Security is paramount, and backend developers implement secure user authentication and authorization mechanisms, managing user sessions, encrypting sensitive data, and ensuring access control. Backend developers configure servers and deploy applications to ensure they run seamlessly in a production environment. Considerations for scalability, load balancing, and server optimization are integral to this process. Frameworks such as Django (Python), Express (Node.js), Ruby on Rails (Ruby), Spring (Java) and Laravel (PHP) are commonly employed to streamline the coding process and adhere to best practices.

Prioritising security, backend developers implement measures to protect against common vulnerabilities like SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF). Ensuring applications can scale to accommodate growing user bases is a key responsibility. Backend developers optimise code, manage database performance, and implement caching strategies to enhance overall application performance. The workflow involves understanding project requirements, implementing server-side logic, managing databases, developing APIs, ensuring security, conducting rigorous testing, and deploying and maintaining the backend infrastructure. Challenges in backend development include managing scalability, addressing security concerns, handling database complexities, and effective collaboration with frontend developers for smooth integration.

EPILOGUE

If, at any point, you found the content offensive, it might be worth reflecting on the possible influence of your personal ego. I'm sure you've been wondering why a substantial portion of this book focuses on computing. The purpose is twofold: first, to dispel the societal misconception that technology is solely about computers and to expand your perspective, shedding light on the true essence of technology. Second, it aims to reintroduce you to computing after resetting your initial perception of tech. Consider this process as establishing a solid foundation for your understanding. Why is it geared towards Applications? Well, as of now (Nov, 2023), I'm in my final year pursuing a degree in Software Engineering, with a specific interest in web application engineering. So, that should clarify why I delved deeply into software-related topics. The next updates from me will likely reach you through your TV and the Internet.

I understand that there might be confusion surrounding the job titles of Software Developer and Software Engineer. In certain companies, there is a division of roles where individuals specialise in the design and planning aspects of the engineering process. This includes roles like Software Architects, Software Requirements Engineers, and UI/UX Designers and Researchers. Their responsibility is to conceptualise and plan, leaving the implementation, coding, and development to the title of Software Developers. In some cases, companies may not have a clear organisational structure and use the term Software Developers for individuals who handle the entire process from design to maintenance (Software Engineering), potentially leading to underpayment. However, the ideal role of a Software Engineer should encompass activities from design through development to maintenance for a comprehensive understanding and contribution to the entire software life cycle.

Professionals in diverse engineering fields consistently immerse themselves in technology to innovate, design, and tackle intricate challenges. While the term "tech bros" might typically evoke images of software development or startup culture, individuals spanning various engineering domains share a unified commitment: utilising technology to propel advancements in their specific areas. Whether it's Civil Engineers shaping infrastructure, Mechanical Engineers optimising machinery, or Environmental Engineers addressing sustainability challenges, the common thread persists. From Chemical and Computer Engineers to Biomedical and Structural Engineers, the label of technology enthusiasts applies universally across disciplines.

If you are currently reading the pdf copy, I will admonish you to check the online version for updates.

Nov, 2023 Edition