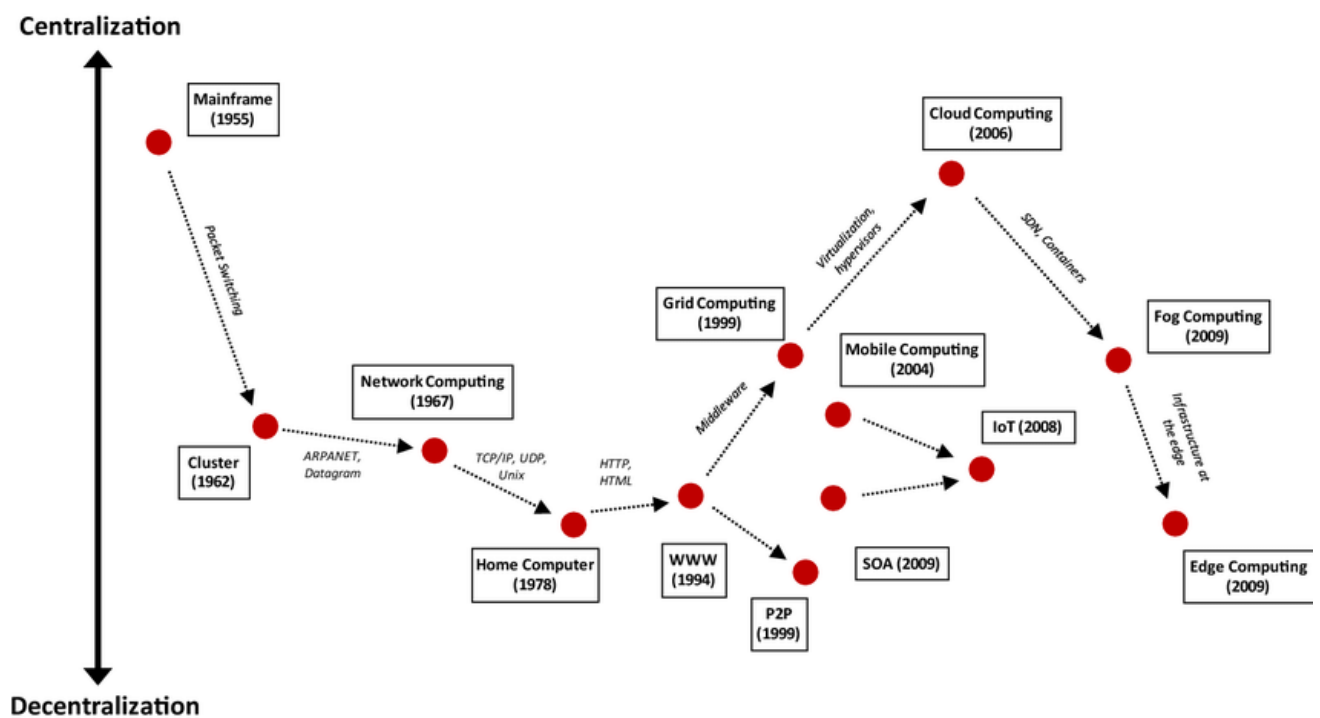


Contents

History and evolution of distributed computing system.....	2
Enterprise applications and enterprises architectures	9
Enterprise Applications	9
Enterprise Architectures	12
Microservice Architecture.....	15
Microservice Framework	18
Justification for selected framework.....	20
References.....	22

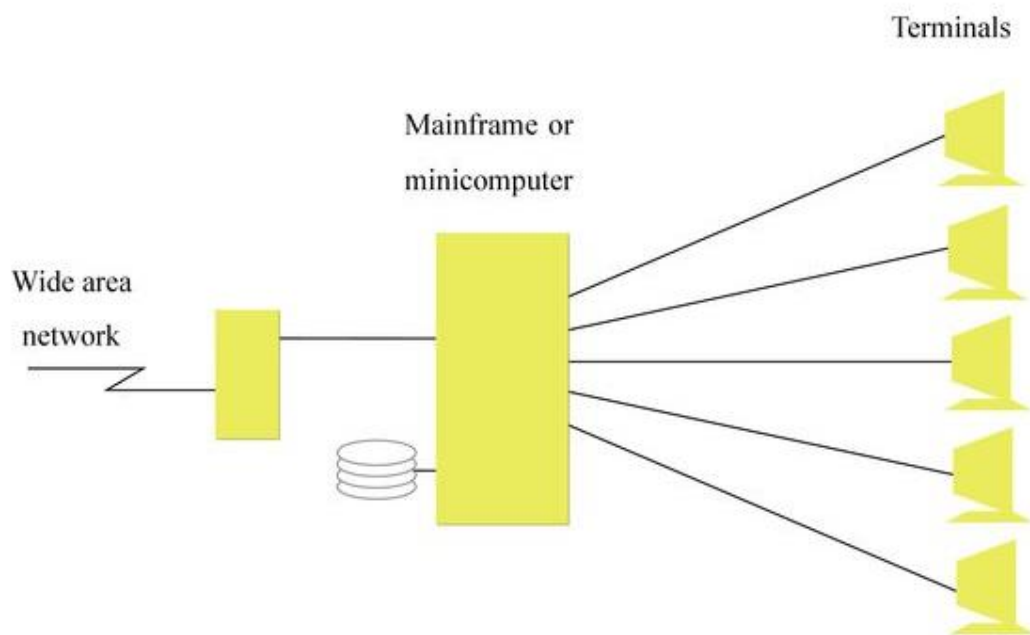
History and evolution of distributed computing system

Distributed computing system referring to a computer concept that multiple computer systems or devices working together to perform a task or solving a problem. A single task or problem will be separated into many parts and handled by computers (also known as nodes) that connected. (Rouse, 2017) The distributed computing system evolved from centralization to decentralization throughout the time and came out with the decentralized system such as edge computing or internet of things that commonly used nowadays. (aryasinghbjc, 2022) This is because the workload to handling a task will be divided among several nodes and allowing them to operate concurrently and share resources. The performance, reliability and scalability will be increased significantly as each of the node in the system will contributes its memory, storage capacities and process power. The image below shows the evolution of the system where the brief introduction will be provided for each of the phases. (Lindsay et al., 2021)



Mainframe (1960 to 1967)

Mainframe machines work as a centralized processing units that providing a single time-sharing system to multiple users that interacts with teletype terminals. It able to efficiently distribute multiple resources over a single medium across the clients when the clients connected to the server and make any requests. For example, IBM system/360 that announced on 7 April 1964 were widely used by organizations such as governments and banks as the system achieved the features such as 32-bit words, microcode, byte and standardized interfaces able to process large amounts of data and provide centralized computing services to multiple terminals. (Shirriff, n.d.)



Cluster Networks (1967 to 1974)

Cluster Networks worked by connecting groups of computers or workstations that tightly linked with high-speed local-area network where each node operates on the same operating system. The ARPANET is the best example of cluster network as it was the first network that able to connect more than 2 computers where it able to connect to several universities and research centres with packet switching. It is the basis for the internet that we are commonly used nowadays as it brought the technologies such as TCP/IP (establish connection among ARPANET computers), packet-switched networks (enable rapid delivery of information to a desire destination) and interface message processors (IMPs) that act as specialized routers for data exchange. (Cveticanin, 2023)

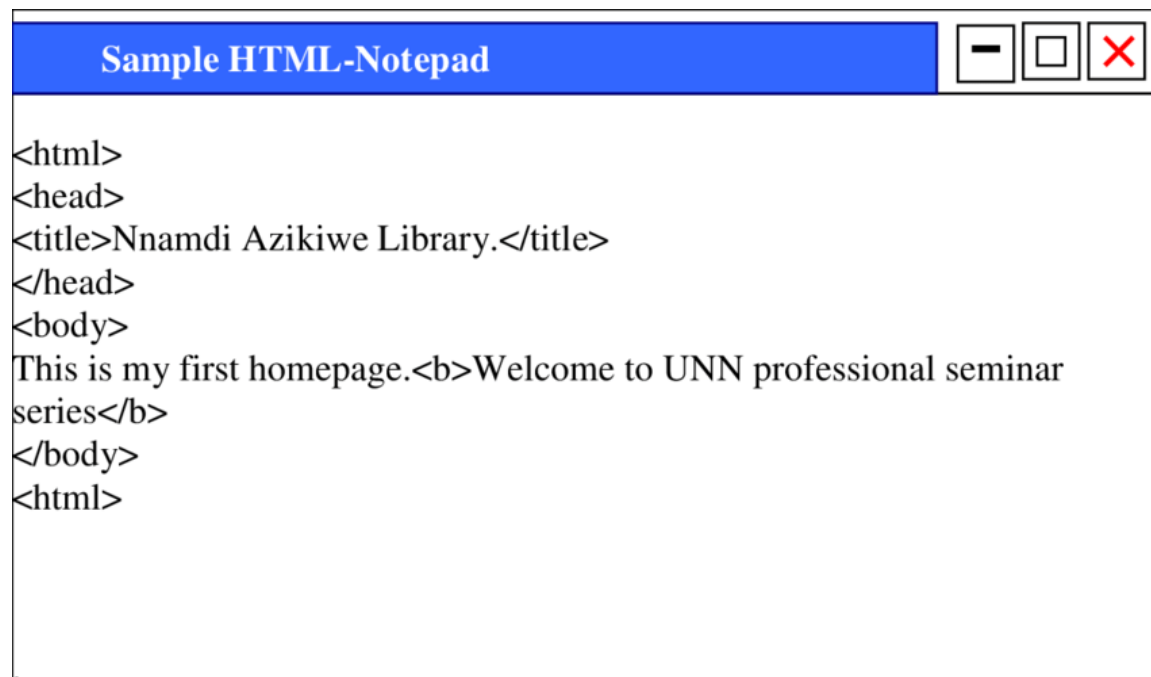
Internet and home PCs (1974 to 1985)

The development of the internet happens in this phase where TCP/IP was introduced and transform the internet into number of interconnected backbones, connecting local networks to the larger internet. As a result, the number of hosts that connected to the network increase drastically which caused the centralised naming system such as HOSTS.TXT impossible to scale sufficiently. Therefore, Domain Name System (DNSs) were created in 1985 to convert the host domain names into IP addresses. (Lindsay et al., 2021)

At the same time, the first personal computer had been invented by IBM known as the Acord that equipped with an Intel chip, 2 floppy disks and colour monitor. In addition, other personal computers such as Commodore 64 and Apple II also invented continuously. It allows the consumers to use the computers at home where each of the computer had GUI-based that used WIMP (windows, icons, menus, and pointers). (Calvello, 2019)

World Wide Web (1985 to 1996)

The distributed computing was recolonized by the World Wide Web as it provided a standardized platform for information access and sharing on a worldwide scale. It was invented by Tim Berners-Lee when working at CERN during 1990 that built on the top of the internet. It consisted of HTTP that allow data transfer between a client and a server, URL or URI that work as a unique universal identifier to access the specific Web component, and the HTML which is the common format to publish a web documents. (contributors, 2023)



```
<html>
<head>
<title>Nnamdi Azikiwe Library.</title>
</head>
<body>
This is my first homepage.<b>Welcome to UNN professional seminar
series</b>
</body>
</html>
```

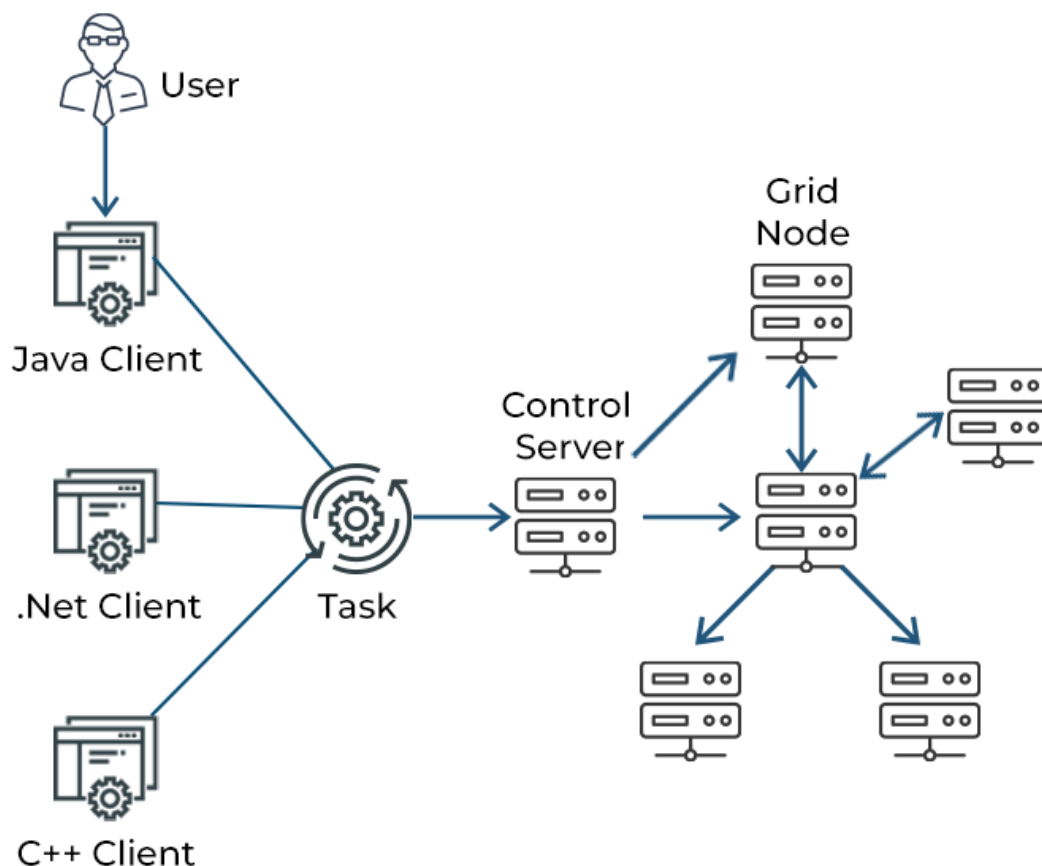
Peer to Peer (P2P), Grids & Web Services (1994 to 2000)

P2P is a decentralized network communications architecture where the peers represent the computer system that connected to each other. The advantages of the peer-to-peer communication is that there is no central coordinator which each peer in the network will act as a client and server. For example, Napster file-sharing service is fully utilized the P2P network by allowing users to share the music file directly with each other. (Vasilevski, 2020)

In a grid computing system, numerous computers collaborate via a network to address significant and challenging issues. To complete tasks that are too challenging for a single computer, they cooperate as a virtual supercomputer. With this method, users and applications can more easily access and utilise a large amount of processing power and resources as if they were all a part of a single large machine. (Kanade, 2022)



HOW GRID COMPUTING WORKS



The development of Web Services makes it possible to further separate the system interface from its Web implementation. Web Services mediated communication via brokerage services rather than enabling direct contact between clients and servers. Standardized interfaces were made available for accessing and using distributed computing resources by web services like Amazon Web Services (AWS).

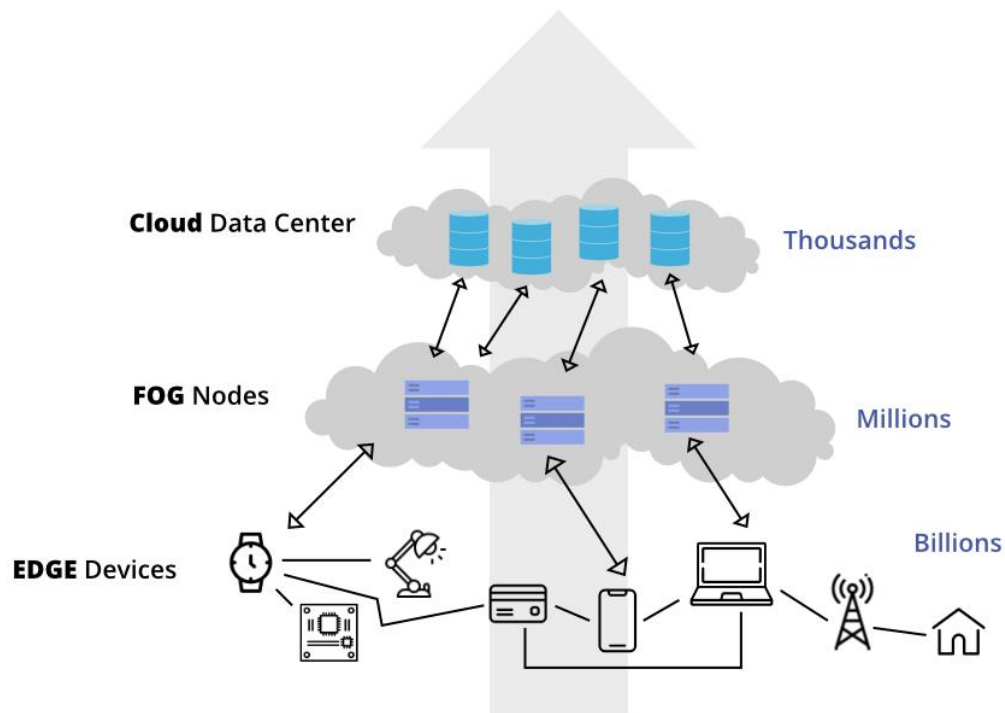
Cloud, Mobile and IoT (2000–2010)

Through cloud computing, computing resources, and services able to be manage online over the internet without the use of building on physical hard drive or own server. It ensures that the resources able to access by anyone around the world. The example of cloud providers such as Amazon EC2, Microsoft Azure, and Google Cloud offer subscription-based services that providing all the computing resources that require.

Mobile computing allowing user to transfer data through wireless networks and become common in nowadays. The most popular mobile computing devices include smart cards, smartphones, and tablets. For example, mobile banking app that allow user to check balances and perform financial transactions preventing the need for physical visit to bank.

The IoT refers to the interconnected device that able gather, exchange and analyse data which each of the device equipped with sensors. For example, the smart home system that popular nowadays where the devices such as smart lights, security cameras and smart thermostats that allow remote management through a mobile application. (aryasinghbjc, 2022)

Fog and edge computing (2010-present)



The edge computing is the practice of processing data and computation within local network of the device. The locally processed data will transfer to the cloud in real-time which reduce the lengthy data transfer to centralized cloud servers. For example, Autonomous vehicle edge computing devices, like self-parking automobiles, gather information from cameras and sensors on the vehicle, evaluate it, and make judgements in milliseconds.

An addition to cloud computing is fog computing. It is a layer that exists between the cloud and the edge. Fog nodes receive the massive volumes of data that edge computers send to the cloud and sort through the vital information. Then the fog nodes move the vital data to the cloud to be stored and either delete or keep the unnecessary data on their own for later analysis. This is how fog computing transfers critical data rapidly while conserving a lot of space on the cloud. (Satyabrata_Jena, 2022)

Enterprise applications and enterprises architectures

Enterprise Applications

Enterprise system (ES) have been recognised in both industry and academics as a desirable technological investment for operation managers as it able to improve the business performance. ES consist of wide range of applications, which included but are not limited to Enterprise Resource Planning (ERP), Supply Chain Management (SCM), and Customer Relationship Management (CRM) systems. (Hendricks et al., 2006)

Enterprise Resource Planning (ERP)



ERP is an application that streamline business operation and provide internal controls, and insight towards the business. It completes the action by utilizing a central database that compiles input from different departments such as manufacturing, accounting, sales, supply chain management and human resource. This is because businesses frequently suffer from the problem where information being dispersed across various platforms which makes it difficult to execute operation and make informed decisions. For example, different teams can use separate spreadsheet for tracking expenses which resulted in the difficulty to locate important data and restrict access to it. Due to the lack of a centralised platform to store all the information of the business, huge amount of time will be wasted in searching for relevant documents and redundancies of work will keep occur due to information need to be up to date across multiple teams. An ERP system handles these problems by compiling data into a centralised database, giving managers and workers access to data from all relevant departments. As a result, costs

are reduced, and since less time is spent looking for the needed data, productivity increases. (McCue, 2022)

Supply Chain Management (SCM)



Supply chain management (SCM) is the process of controlling the flow of materials, information, and money related to a good or service. It includes a number of phases, from obtaining raw materials through shipping the finished product to the intended location. To make it easier for all parties engaged in the development, delivery, and tracking of goods and services to work together, SCM systems in use today combine physical material handling with software solutions. This comprises retailers, wholesalers, transportation and logistics companies, suppliers, and manufacturers. Procurement, product lifecycle management, supply chain planning (such as inventory planning and maintaining corporate assets and production lines), logistics (including transportation and fleet management) and order management are just a few of the many tasks that fall under the umbrella of SCM operations. Additionally, SCM can be used to control global suppliers and coordinate cross-border production processes to manage operations related to international trade. (oracle, 2023) Supply chain management (SCM) varies for each company based on its goals, constraints, and strengths. There are six primary SCM models:

Continuous Flow Model: Suitable for mature industries with consistent customer demand.

Agile Model: Ideal for companies facing unpredictable demand or customer-order products, prioritizing flexibility.

Fast Model: Emphasizes quick product turnover and capitalizing on trends with short life cycles.

Flexible Model: Suited for companies with seasonality, allowing easy scaling of production based on varying demand.

Efficient Model: Focuses on optimizing supply chain processes to gain a competitive advantage in industries with tight profit margins.

Custom Model: Tailored SCM approach for highly specialized industries with unique requirements. (FERNANDO, 2022)

Customer Relationship Management (CRM) systems.



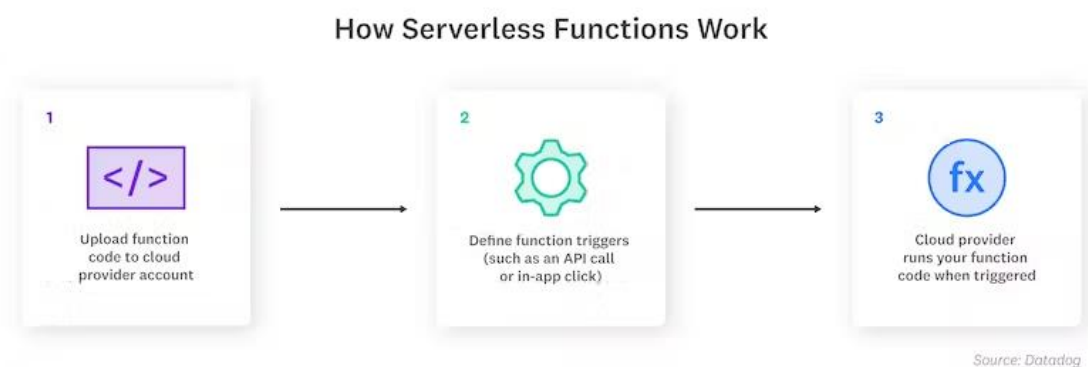
CRM system allowing sales, marketing and customer service departments to have a thorough understanding of the business company base. This is because CRM focus on utilizing the customer data to enhance the customer experience and improve relationship by centralized and manage the data such as purchase history and preferences of customers. For example, personalized recommendations can be provided if a customer consistently purchase any specific brand clothes which helps in strengthening the customer-brand bond. CRM software is a crucial tool for sales and marketing, allowing them to segment clients, construct buyer profiles, produce insights, and collect information from many sources, including social media. Customer interactions are converted into useful data, enabling efficient sales and marketing initiatives. Additionally, CRM helps customer service groups by allowing case management for resolving post-sale product difficulties, thus enhancing customer relationships. It improves the customer relationship cycle overall by supplying essential data and facilitating personalised interactions from prospecting to nurturing devoted and returning clients. It gives organisations the ability to maximise their efforts in marketing, sales, and customer service, which enhances client loyalty and satisfaction. (Mai, 2022)

Enterprise Architectures

Due to their success in solving the issues of contemporary software development, a number of application architecture patterns and strategies have grown in prominence in recent years. Today's prevalent application architectural trends include:

Serverless Architecture

Serverless Architecture allow developers to built and deploy applications without having to worry about managing the underlying infrastructures. This is because developers able to write code as discrete functions that triggered by events using the well-liked serverless architecture known an Function as a Service (FaaS). These functions are deployed to a cloud provider, which manages execution by provisioning servers as needed. It allow developers to concentrate on writing and deploying the application code as the server management had been take care of for them.



In 2014, Amazon introduced the AWS Lambda as the first mainstream FaaS platform that still the main options for developers nowadays to build serverless applications. Other than that, there are other platform such as Google Cloud Functions (GCF) and Azure Functions.

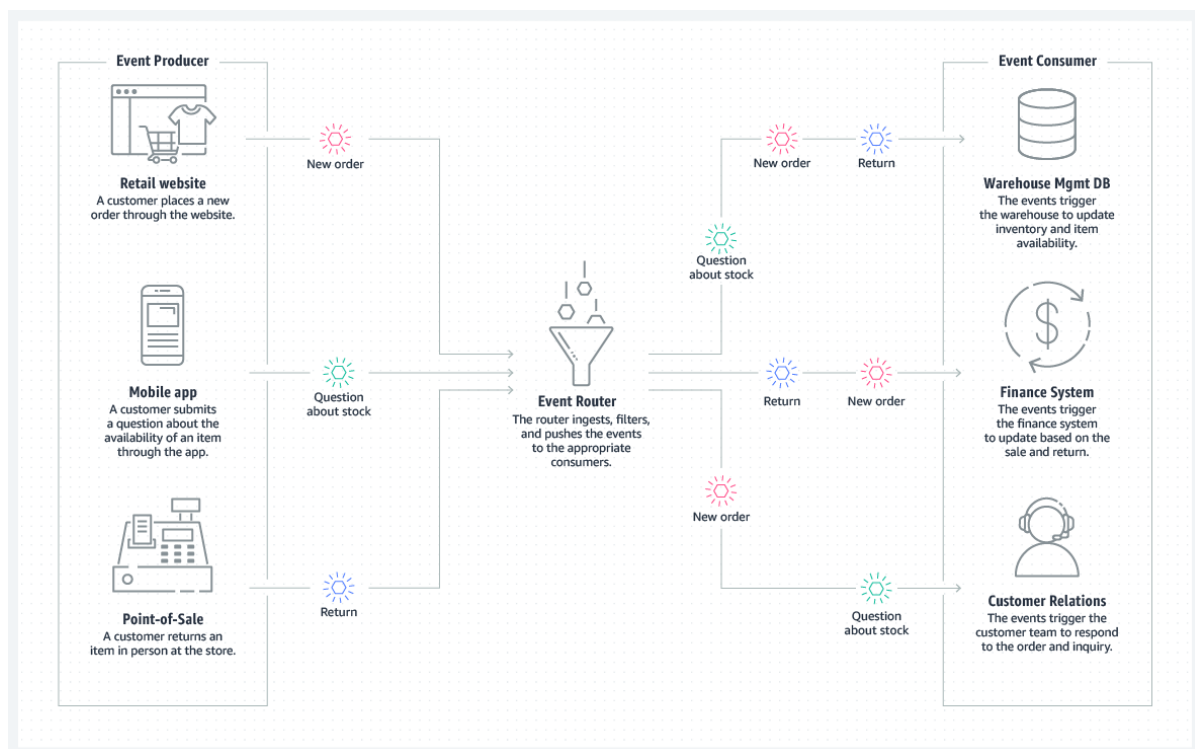
Containerization and Orchestration

Containerized architecture allows each of the software that built have its own dependencies to be packaged in an independent unit call container that ensure that it able to run in its own environment that configured. Containers able to run without errors and will not face any incompatibilities in various settings compared to traditional software development when moving to different environment. It helps developers to manage the automate software development and deployment by specified the required infrastructure in a configuration file and deploy it when required. The orchestration allowing developers to deploy large numbers

of containers that build and manage within large scale. It handles application load and environment metrics, identifies and recovers from problems, controls application configuration, handles storage and networking, and improves security. The example of container orchestration platforms such as Kubernetes, Docker Swarm and Apache Mesos. (Containerized Architecture: Components and Design Principles, 2023)

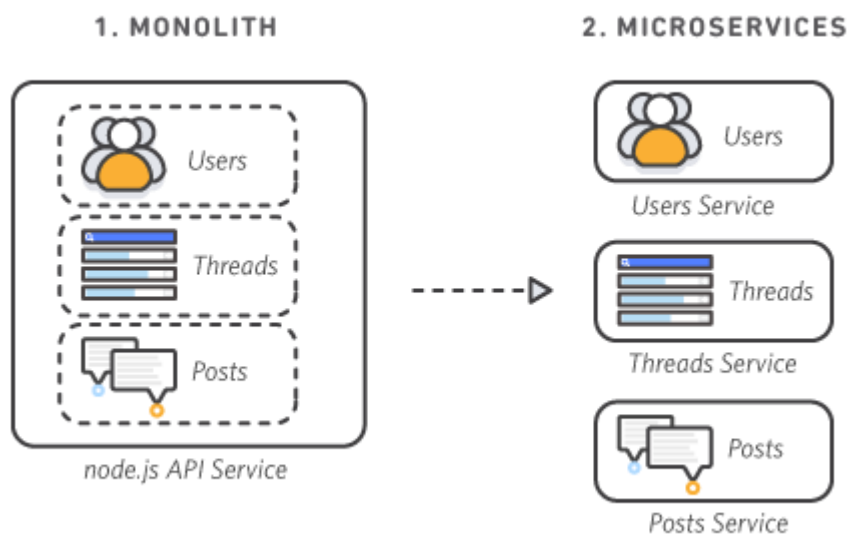
Event-driven architecture

Event-driven architecture is a system design method that allows for the recording, transmission, and processing of events via decoupled architecture. It is able to share information and execute the tasks without the need to know what happens after, resulting in enhanced flexibility and scalability. This is because the event procedures can directly send the notification to event router without knowing who the recipients are, and the events will be distributed accordingly to the relevant consumers. The asynchronous event processing will be conducted by consumers which lead to the real-time data processing and independent operation. (Roddewig, 2022)



Microservice Architecture

Microservice architecture is a method where software is divided into small and independent services that communicate over API, message broker and service mesh. Each of the service is handle by a small team that come with different programming background and knowledge available which increase the speed of development and scalability. By comparing to monolithic architecture that require all processes are tightly bind and run as a single service, microservice allow for individual scaling by developed in a single service according to the business capabilities as a single function. As they are independently run, each service can be updated, deployed, and scaled to meet demand for specific functions of an application. (AWS, 2023)



Performance

Microservice have a better performance as each of the individual service can be scale independent and optimize resources allocation. It able to utilize more resource to increase the performance by using the technologies such as load balancing, caching, and asynchronous communication. In addition, microservice will be prioritize business functionality over technologies as it able to focus on development of specific services without require building entire application from scratch. This allow the application to have continuous deployments and release as each of the functionalities can be divided among development teams and worked on simultaneously. (appdynamics, 2023)

Scalability

As microservice architecture allow each of the service to be developed and runs independently, it able to seamless addition, removal, updating and scaling of individual microservices without causing disruption to other services. Developers have the freedom to complete these activities independently, allowing for more efficient resource allocation. For example, a microservice for purchasing item in e-commerce applications experiencing increased demand due to special promotions can be easily scale up to accommodate surge without affecting the other services that will cause the huge increment of cost with monolithic architecture. When the demand falls, the microservice can be scaled back to free up resources for other areas of the system. (GitLab, 2022)

Reliability

By using microservice architecture, the detection of error and resolution for the performance issue will become straightforward. This is because the fault isolation provided by individual modules ensure that a single failure of service will not have any effect toward the whole application. In addition, the risk of downtime will be reduced as developers have the option to make changes or rollback on a specific module without redeploying the whole application. (appdynamics, 2023)

Precondition : Need Graph that compose Vertices and Edge with direction

Input : Graph $G(E, V)$

Output : Graph Microservices

BEGIN

Loop each Control in Graph G

IF Control.RelatedCount() ≥ 2 **THEN**

FOR i= 0 to Control.RelatedCount() **DO**

 CallRate \leftarrow Control.CallMethodRate(Entity)

 NameRate \leftarrow Control.CompareMethodRate(Entity)

 Avg \leftarrow Average(CallRate, NameRate)

 Entity \leftarrow SetMaxRateMainEntity(Control, Avg)

ENDFOR

ELSE THEN

 Entity \leftarrow SetMainEntity(Control)

ENDIF

END Loop

Loop each Entity in Graph G

IF Entity.hasMainEntity() **THEN**

 Microservices \leftarrow ConstructMicroservice(Control, Entity)

ENDIF

END Loop

Loop each M in Microservices

IF M.Control.RelateCount() ≥ 1 **THEN**

 Microservices \leftarrow CreateNewBoundary(M.Source, M.Target)

 Microservices \leftarrow CreateNewRelation(M.Source, M.Target, M.Method)

ENDIF

END Loop

RETURN Microservices

End

Append is the pseudocode for microservice algorithms.

Microservice Framework

Spring Boot is one of the users friendly for beginner and finest frameworks in Java for microservice development. It is an open-source framework that developed by Pivotal Team that renowned for the simplicity and massive features to build stand-alone and production ready application. (tutorialspoint, 2023) It also equipped with built-in functionalities such as auto-configuration, starter dependency and security that speed up the development process. As it has a large community for developers, assistance can be got in short period of time to resolve the errors that occurred during development. (yuvraj10, 2022)

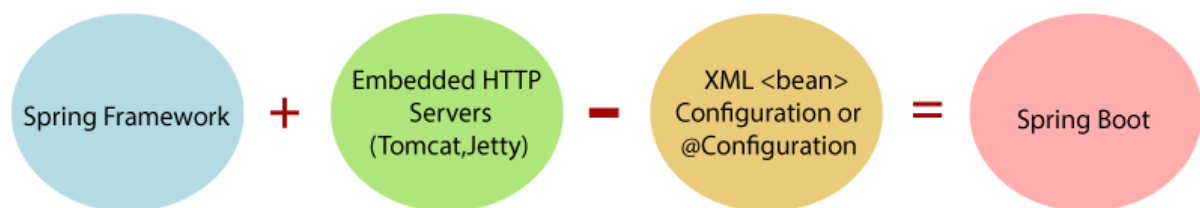
In Spring Boot, the application can be divided into four primary layers (Gunkar, 2019):

Presentation Layer: The frontend part of the application which includes user interface that also known as views.

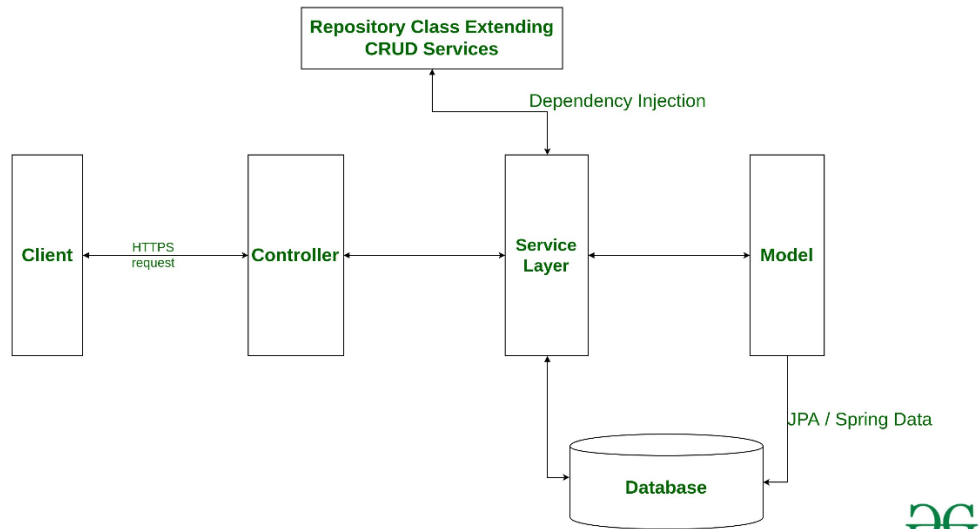
Data Access Layer: Handling the operations that related to data retrieval and storage from database such as CRUD.

Service Layer: Consist of service classes that provided business logic and act as the connection between presentation layer and data access layer.

Integration layer: Integrates with multiple web services available over the internet. It often involves the use of XML messaging systems to communicate with external services.



Spring Boot flow architecture



1. The client can make a HTTP request such as GET, PUT, POST or DELETE.
2. The request will be forwarded to the controller which help to map the correct server logic that map with the request that received.
3. The Service layer is where the business logic takes place. Spring boot executes all logic on database data that is assigned to the spring boot model class via Java Persistence Library (JPA).
4. A JSP page will return at the end of the whole request when no error occurred.

Justification for selected framework

As a web-based Property APU System will be required to develop in Part 2, Spring Boot will be one of the good options for the selected framework. This is because Spring Boot provide an easy method to develop Spring-based applications using Java language which meet the programming language requirements for the project. (Pedamkar, 2023) Other than that, below are the details explanation for the selection of framework:

- **Auto configuration**

Comparing to traditional java development method that required a lot of manual configurations and adding the dependencies into the application, Spring Boot able to increase the speed of development as database connection, session factory, data source, security, and other infrastructure-related tasks can be auto-configure. Spring Boot takes a convention-over-configuration approach, with many of these configurations automatically configured able to allow development to start quickly. (AmiyaRanjanRout, 2023) For example, entityManagerFactory and transactionManager beans will be load to the configuration class. (baeldung, 2023)

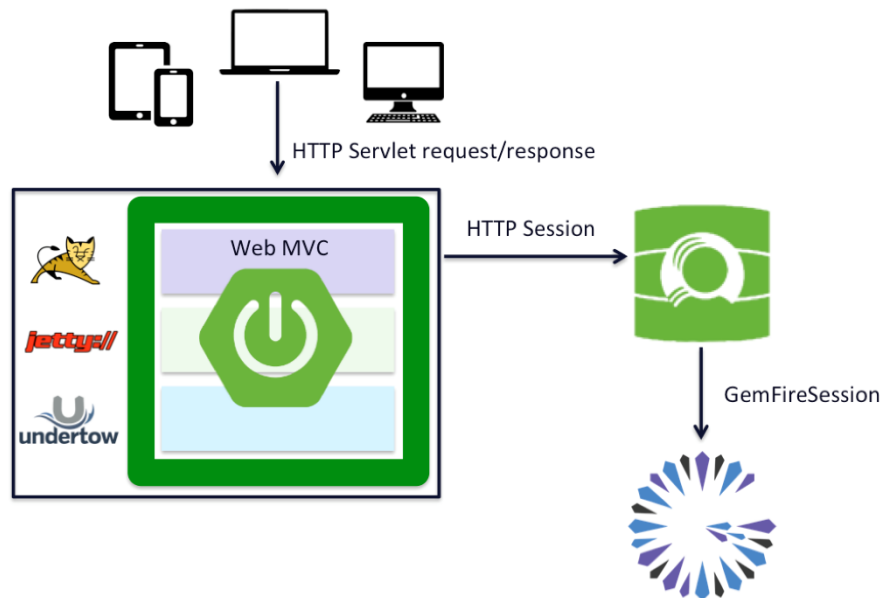
```
@Bean
@ConditionalOnBean(name = "dataSource")
@ConditionalOnMissingBean
public LocalContainerEntityManagerFactoryBean entityManagerFactory() {
    LocalContainerEntityManagerFactoryBean em
        = new LocalContainerEntityManagerFactoryBean();
    em.setDataSource(dataSource());
    em.setPackagesToScan("com.baeldung.autoconfigure.example");
    em.setJpaVendorAdapter(new HibernateJpaVendorAdapter());
    if (additionalProperties() != null) {
        em.setJpaProperties(additionalProperties());
    }
    return em;
}
```

```
@Bean
@ConditionalOnMissingBean(type = "JpaTransactionManager")
JpaTransactionManager transactionManager(EntityManagerFactory entityManagerFactory) {
    JpaTransactionManager transactionManager = new JpaTransactionManager();
    transactionManager.setEntityManagerFactory(entityManagerFactory);
    return transactionManager;
}
```

- **Suitable for web application**

Spring Boot is designed for web application as provided quick development and rapid prototyping as several starter features such as REST APIs, security, database connection and more will be provided. It allows developers to have simple access towards Embedded HTTP servers like Jetty and Tomcat that can test the web application easily.

It helps developers to save the time as the web application can be test without spending much time on boilerplate code or configuration. (Padamkar, 2023)



- **Managing dependencies**

Managing dependencies in traditional Java development can be difficult and time-consuming. Developers must actively look for, download, configure, and resolve library version conflicts, which might cause compatibility concerns. Spring Boot makes this process easier by utilising build systems such as Maven or Gradle and supplying startup dependencies. Starters encapsulate all required libraries and configurations, resolving dependencies and maintaining versions automatically. Including a beginning dependency puts up the essential components for specific functionalities, such as web applications or database connectivity. The dependency management in Spring Boot saves time, lowers errors, and lets developers to focus on application logic. (Prad, 2020)

References

- AmiyaRanjanRout. (2023). *7 Major Reasons to Choose Spring Boot For Microservices Development*. Retrieved from geeksforgeeks: <https://www.geeksforgeeks.org/why-to-choose-spring-boot-for-microservices-development/>
- appdynamics. (2023). *What Are The Benefits of Microservices Architecture?* Retrieved from appdynamics: <https://www.appdynamics.com/topics/benefits-of-microservices>
- aryasinghbjc. (22 11, 2022). *Evolution of Distributed Computing Systems*. Retrieved from geeksforgeeks.org: <https://www.geeksforgeeks.org/evolution-of-distributed-computing-systems/>
- AWS. (2023). *What are Microservices?* Retrieved from AWS: <https://aws.amazon.com/microservices/>
- baeldung. (4 5, 2023). *Create a Custom Auto-Configuration with Spring Boot*. Retrieved from baeldung: <https://www.baeldung.com/spring-boot-custom-auto-configuration>
- Calvello, M. (8 7, 2019). *A Complete History of Computers: From the 1800s to Now*. Retrieved from g2: <https://www.g2.com/articles/history-of-computers#computers-from-the-1980-1990s>
- Containerized Architecture: Components and Design Principles*. (2023). Retrieved from aquasec: <https://www.aquasec.com/cloud-native-academy/container-security/containerized-architecture/>
- contributors, M. (8 6, 2023). *World Wide Web*. Retrieved from developer.mozilla.org: https://developer.mozilla.org/en-US/docs/Glossary/World_Wide_Web
- Cveticanin, N. (6 5, 2023). *ARPANET: The Project That Launched the Internet*. Retrieved from dataprot: <https://dataprot.net/articles/what-is-arpanet/>
- FERNANDO, J. (7 7, 2022). *Supply Chain Management (SCM): How It Works and Why It Is Important*. Retrieved from investopedia: <https://www.investopedia.com/terms/s/scm.asp>
- GitLab. (29 9, 2022). *What are the benefits of a microservices architecture?* Retrieved from gitlab: <https://about.gitlab.com/blog/2022/09/29/what-are-the-benefits-of-a-microservices-architecture/#:~:text=The%20benefit%20of%20a%20microservice,the%20resilience%20of%20the%20infrastructure.>
- Gunkar, R. (26 8, 2019). *Introduction to Spring Boot*. Retrieved from geeksforgeeks: <https://www.geeksforgeeks.org/introduction-to-spring-boot/>
- Kanade, V. (19 1, 2022). *What Is Grid Computing? Key Components, Types, and Applications*. Retrieved from Spiceworks: <https://www.spiceworks.com/tech/cloud/articles/what-is-grid-computing/>
- Mai, T. (2 12, 2022). *ERP VS CRM VS SCM: WHY YOU SHOULD COMBINE THESE DIFFERENT SYSTEMS*. Retrieved from magenest: <https://magenest.com/en/erp-vs-crm-vs-scm/>

- McCue, I. (12 8, 2022). *What Is ERP (Enterprise Resource Planning)?* Retrieved from netsuite: <https://www.netsuite.com/portal/resource/articles/erp/what-is-erp.shtml>
- oracle. (2023). *What is SCM (Supply Chain Management)?* Retrieved from oracle: <https://www.oracle.com/my/scm/what-is-supply-chain-management/#:~:text=At%20the%20most%20fundamental%20level,product%20at%20its%20final%20destination.>
- Pedamkar, P. (14 3, 2023). *What is Spring Boot?* Retrieved from educba: <https://www.educba.com/what-is-spring-boot/>
- Prad, R. (30 12, 2020). *Building Microservices Application Using Spring Boot & Cloud.* Retrieved from sayonetech: <https://www.sayonetech.com/blog/spring-boot-microservices-new-age-framework-your-apps/>
- Roddewig, S. (15 2, 2022). *What Is Event-Driven Architecture? Everything You Need to Know.* Retrieved from hubspot: <https://blog.hubspot.com/website/event-driven-architecture>
- Rouse, M. (15 6, 2017). *Distributed Computing System.* Retrieved from techopedia: <https://www.techopedia.com/definition/7/distributed-computing-system>
- Satyabrata Jena. (22 11, 2022). *Difference Between Edge Computing and Fog Computing.* Retrieved from geeksforgeeks: <https://www.geeksforgeeks.org/difference-between-edge-computing-and-fog-computing/>
- Shirriff, K. (n.d.). *Ken Shirriff's blog.* Retrieved from righto: <https://www.righto.com/2019/04/iconic-consoles-of-ibm-system360.html>
- tutorialspoint. (2023). *Spring Boot - Introduction.* Retrieved from tutorialspoint: https://www.tutorialspoint.com/spring_boot/spring_boot_introduction.htm
- Vasilevski, A. (2020). *Introduction to Peer to Peer (P2P) Network.* Retrieved from codespot: <https://www.codespot.org/introduction-to-peer-to-peer-network/>
- yuvraj10. (8 8, 2022). *geeksforgeeks.* Retrieved from 5 Best Java Frameworks For Microservices: <https://www.geeksforgeeks.org/5-best-java-frameworks-for-microservices/>
- Lindsay, D., Gill, S. S., Smirnova, D., & Garraghan, P. (2021). The evolution of distributed computing systems: from fundamental to new frontiers. *Computing*. <https://doi.org/10.1007/s00607-020-00900-y>
- Hendricks, K. B., Singhal, V. R., & Stratman, J. K. (2006). The impact of enterprise systems on corporate performance: A study of ERP, SCM, and CRM system implementations. *Journal of Operations Management*, 25(1), 65–82. <https://doi.org/10.1016/j.jom.2006.02.002>