Practical No.04

Aim: Case Study of PaaS (Facebook, Google App Engine)

Theory:

- Platform as a Service (PaaS) is a cloud computing service model that provides a platform and environment for developers to build, deploy, and manage applications without the complexities of managing underlying infrastructure.
- PaaS offerings typically include development tools, runtime environments, database management systems, and other services required for application development and deployment.
- Here are the key features and benefits of PaaS:
  1. **Development Tools**: PaaS platforms often come with a set of development tools, integrated development environments (IDEs), and frameworks to streamline application development.
  2. **Runtime Environment**: PaaS provides a runtime environment where applications can be executed, automatically handling aspects like scaling, load balancing, and infrastructure management.
  3. **Database Services:** PaaS solutions usually include managed database services, which simplify database setup, scaling, and maintenance.
  4. **Middleware:** Middleware components, such as messaging queues, caching, and authentication services, are often available within PaaS offerings.
  5. **Automated Scaling:** PaaS platforms typically offer auto-scaling capabilities, allowing applications to automatically adjust resources based on traffic and demand.
  6. **Security and Compliance**: PaaS providers often have built-in security measures and compliance certifications, simplifying security management.
  7. **Deployment and DevOps Integration:** PaaS facilitates continuous integration and continuous deployment (CI/CD) pipelines, making it easier to deliver and update applications.
  8. **Resource Management:** PaaS abstracts infrastructure management, allowing developers to focus on coding and application logic rather than server provisioning and maintenance.

- Benefits of PaaS:
  1. **Faster Development:** PaaS accelerates development by providing pre-configured development environments and tools, reducing the time required to set up infrastructure.
  2. **Scalability:** PaaS platforms offer automatic scaling, ensuring that applications can handle variable workloads without manual intervention.
  3. **Cost Efficiency:** PaaS eliminates the need for organizations to invest in and manage physical infrastructure, reducing capital expenditures and operational costs.
  4. **Reduced Complexity:** Developers can focus on writing code and building features without worrying about server and infrastructure management.
  5. **Global Reach:** Many PaaS providers offer a global network of data centers, enabling applications to be hosted close to end-users for lower latency and improved performance.
  6. **Security and Compliance:** PaaS providers often have robust security measures and compliance certifications, making it easier for organizations to meet regulatory requirements.
  7. **Collaboration:** PaaS fosters collaboration among development teams by providing shared development environments and tools.
  8. **Easier Maintenance:** PaaS providers handle routine maintenance tasks, such as patching, backups, and monitoring, freeing up IT teams to focus on strategic initiatives.
  9. **High Availability:** PaaS platforms often offer redundancy and failover capabilities to ensure high availability of applications.

Examples of popular PaaS providers include Google App Engine, Microsoft Azure App Service, Heroku, AWS Elastic Beanstalk, and IBM Cloud Foundry. Organizations choose PaaS solutions based on their specific development needs, preferred programming languages, and existing cloud infrastructure preferences. PaaS is a valuable option for businesses looking to streamline application development and deployment while benefiting from the scalability and flexibility of cloud computing.

- **Case Study: Facebook's Use of PaaS:**

Facebook, one of the largest social media platforms in the world, serves billions of users globally. It requires a robust and scalable infrastructure to manage user data, facilitate communication, and support a wide range of features and services.

- Use of PaaS:

  While Facebook does not use a traditional third-party PaaS provider like Google App Engine or Azure App Service, it has developed its internal PaaS-like solutions to address its specific needs. These solutions are custom-built to manage the scale and complexity of Facebook's operations.

- Key Components of Facebook's PaaS-like Infrastructure:

1. FBOSS (Facebook Open Switching System):

- Purpose: FBOSS is Facebook's open-source network switch software platform. While not a traditional PaaS, it plays a critical role in managing network infrastructure.
- Use Case: FBOSS helps Facebook efficiently manage its vast data center networks, ensuring high performance and reliability.

2. Tupperware:

- Purpose: Tupperware is Facebook's container orchestration platform.
- Use Case: It allows Facebook to run thousands of applications in containers across its data centers, improving resource utilization and application deployment speed.

3. FBOpen (Facebook Open Source):

- Purpose: Facebook actively contributes to open-source projects, fostering collaboration and innovation in the developer community.
- Use Case: By sharing its tools and technologies as open source, Facebook encourages the development community to improve and expand upon its solutions.

- Benefits:

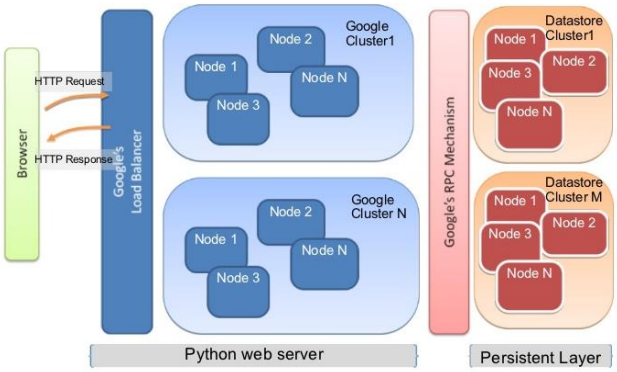
- Scalability: Facebook's custom PaaS-like solutions are designed to handle immense scale, supporting billions of users and an enormous amount of data and traffic.
- Efficiency: These solutions enable Facebook to efficiently manage its data centers, optimize resource usage, and reduce operational overhead.
- Customization: Facebook's internal PaaS allows for customization and fine-tuning of infrastructure and services to meet its unique requirements.
- Innovation: By actively contributing to open source and developing its solutions, Facebook fosters innovation within the tech community and maintains a cutting-edge technology stack.

- Challenges:

  - Complexity: Building and maintaining custom PaaS solutions at Facebook's scale is a highly complex undertaking.
  - Security: Ensuring the security and privacy of user data is a top priority, and managing these aspects at scale presents significant challenges.

- **Case Study on Google App Engine:**

App Engine Physical Deployment Diagram



Google App Engine is an industry-leading Platform as a Service (PaaS) from the company that pioneered much of the microservices technology we rely on today.
In this blog, we are going to cover Google App Engine, its features, advantages, and use cases.

Google App Engine:

Google App Engine is a fully managed serverless platform for developing and hosting web applications at a scale. Users can choose from several popular languages, libraries, and frameworks to develop their applications and then App Engine takes care of provisioning servers and scaling app instances based on demand. It is a PaaS for building scalable applications.

Google App Engine Environments

Google Cloud provides 2 environments to use App Engine, one is a standard environment with constrained environments and support for languages such as Python, Go, and node.js. The other one is the Flexible Environment where developers have more freedom such as running custom runtimes using docker, longer request & response timeout, and the ability to install custom dependencies/software, and SSH into the virtual machine.

1.) Standard Environment

It is based on the container that runs on the Google infrastructure. It provides users with the facility to easily build and deploy an application that runs under heavy load and a large amount of data. It supports the following languages: Python, JAVA, Node.js, Ruby, PHP, and Go.

Features of Standard Environment:

- Persistent storage with queries, sorting, and transactions.
- Automatic scaling and load balancing.
- Asynchronous task queues for performing work outside the scope of a request.
- Scheduled tasks for triggering events at regular intervals or specific time intervals.

2.) Flexible Environment

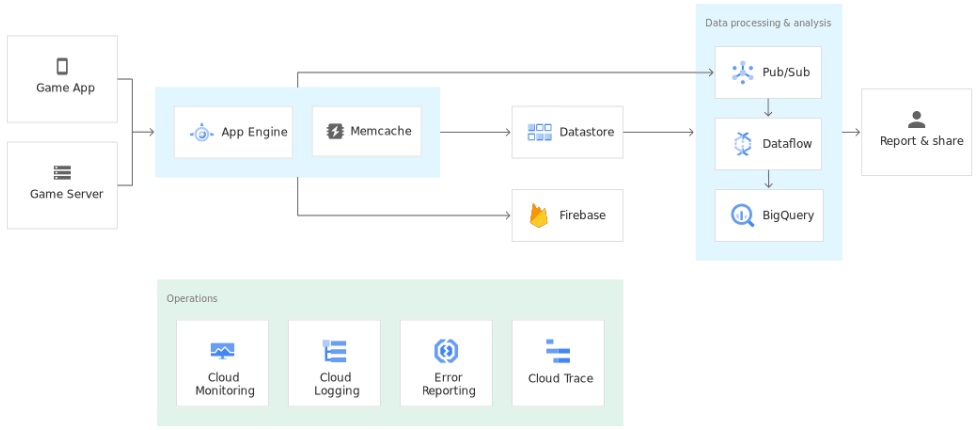
App Engine Flexible Environment allows users to concentrate on writing code. Based on Google Compute Engine, it automatically scales the app up and down and along with it also balances the load. It allows users to customize their runtime and the operating system of their virtual machines using Dockerfiles.

Features of Flexible Environment:

- Infrastructure Customization: App Engine flexible environment instances are Compute Engine virtual machines, which implies that users can take advantage of custom libraries, use SSH for debugging, and deploy their own Docker containers.
- It is an open-source community.
- Native feature support: Features such as microservices, authorization, SQL and NoSQL databases, traffic splitting, logging, etc are natively supported.
- Performance: Users can take advantage of a wide array of CPU and memory configurations.

Google App Engine Use-Cases

1. **Scalable Mobile Backends:** App Engine automatically scales the hosting environment for users who are building their first mobile application or looking to reach out to existing users via a mobile experience. It offers seamless integration with Firebase which provides an easy-to-use frontend mobile platform along with a scalable and reliable back end.



2. **Modern Web-Applications:** Quickly reach customers and end-users by deploying web apps on App Engine. With zero-config deployments and zero server management, It allows users to focus on just writing code. In addition to this, it automatically scales to support sudden traffic spikes without provisioning, patching, or monitoring.

Benefits of Google App Engine

The main benefits of Google App Engine are:

- **Open and familiar languages and tools:** Users can build and deploy apps quickly using popular languages or bring their own language runtimes and frameworks, they can also manage resources from the command line, debug source code, and run API backends easily.
- **Just add code:** App Engine protects from security threats using firewall capabilities, IAM rules, and managed SSL/ TLS certificates so that it helps users to write code without any underlying infrastructure.

- **Pay only for what you use:** It naturally scales relying upon the application traffic and expends resources just when the code is running.

Features of App Engine

Some of the prominent features of Google App Engine include:

- **Popular language:** Users can build the application using language runtimes such as Java, Python, C#, Ruby, and PHP or build their own runtimes.
- **Open and flexible:** Custom runtimes allow users to bring any library and framework to App Engine by supplying a Docker container.
- **Fully managed:** It allows users to add your web application code to the platform while it manages the infrastructure. The engine ensures that web apps are secure and running and enables the firewall to save them from malware and threats.
- **Powerful application diagnostics:** Google App engine uses cloud monitoring and cloud logging to monitor the health and performance of the app and to diagnose and fix bugs quickly it uses cloud debugger and error reporting.
- **Application versioning:** It easily hosts different versions of the app, and create development, test, staging, and production environments.
- **Application security:** Google App Engine helps safeguard the application by defining access rules with an App Engine firewall and leverage managed SSL/TLS certificates by default on the custom domain without incurring any additional cost.

- **Conclusion:** From the above practical we learned about PaaS with a case study on Facebook and Google App Engine.