**TASK 1 - CASE STUDIES**

**AIM**: To study about Amazon AWS, Google Apps, Microsoft Azure

**Amazon Web Services**

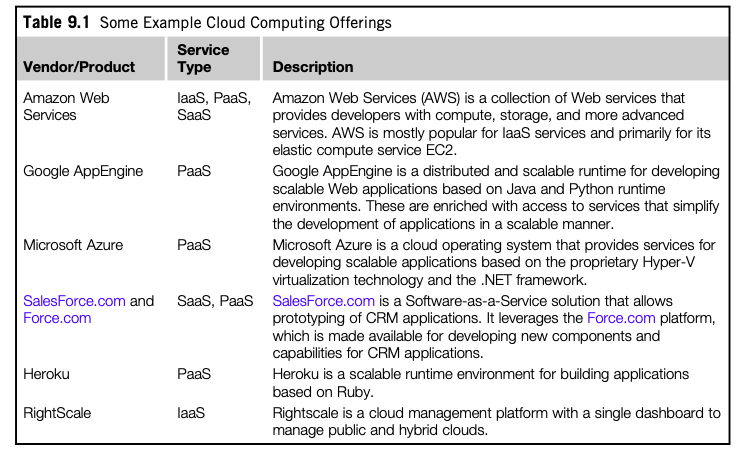


**Amazon Web Services** (**AWS**) is a [subsidiary](https://en.wikipedia.org/wiki/Subsidiary) of [Amazon](https://en.wikipedia.org/wiki/Amazon.com) that provides [on-demand](https://en.wikipedia.org/wiki/Software_as_a_service) [cloud computing](https://en.wikipedia.org/wiki/Cloud_computing) [platforms](https://en.wikipedia.org/wiki/Computing_platform) and [APIs](https://en.wikipedia.org/wiki/Application_programming_interface) to individuals, companies, and governments, on a metered pay-as-you-go basis. In aggregate, these cloud computing [web services](https://en.wikipedia.org/wiki/Web_services) provide a set of primitive abstract technical infrastructure and [distributed computing](https://en.wikipedia.org/wiki/Distributed_computing) building blocks and tools. One of these services is [Amazon Elastic Compute Cloud](https://en.wikipedia.org/wiki/Amazon_Elastic_Compute_Cloud), which allows users to have at their disposal a [virtual](https://en.wikipedia.org/wiki/Virtualization) [cluster of computers](https://en.wikipedia.org/wiki/Computer_cluster), available all the time, through the Internet. AWS's version of virtual computers emulate most of the attributes of a real computer including, hardware [central processing units](https://en.wikipedia.org/wiki/Central_processing_unit) (CPUs) and [graphics processing units](https://en.wikipedia.org/wiki/Graphics_processing_unit) (GPUs) for processing, local/[RAM](https://en.wikipedia.org/wiki/Random-access_memory) memory, hard-disk/[SSD storage](https://en.wikipedia.org/wiki/Solid-state_drive); a choice of operating systems; networking; and pre-loaded application software such as [web servers](https://en.wikipedia.org/wiki/Web_server), [databases](https://en.wikipedia.org/wiki/Database), [customer relationship management](https://en.wikipedia.org/wiki/Customer_relationship_management) (CRM), etc.

The AWS [technology](https://en.wikipedia.org/wiki/Technology) is implemented at [server farms](https://en.wikipedia.org/wiki/Server_farm) throughout the world, and maintained by the Amazon subsidiary. Fees are based on a combination of usage which is known as "Pay-as-you go" model, the hardware/OS/software/networking features chosen by the subscriber, required [availability](https://en.wikipedia.org/wiki/Availability_(system)), [redundancy](https://en.wikipedia.org/wiki/Redundancy_(engineering)), [security](https://en.wikipedia.org/wiki/Computer_security), and service options. Subscribers can pay for a single virtual AWS computer, a dedicated physical computer, or clusters of either. As part of the subscription agreement,[[5]](https://en.wikipedia.org/wiki/Amazon_Web_Services" \l "cite_note-AWSagreement-5) Amazon provides security for subscribers' system. AWS operates from many global geographical regions including 6 in [North America](https://en.wikipedia.org/wiki/North_America).[[6]](https://en.wikipedia.org/wiki/Amazon_Web_Services#cite_note-GlobalInfrastructure-6)

In 2017, AWS comprised more than 90 (165 as of 2019) services spanning a wide range including [computing](https://en.wikipedia.org/wiki/Computation), [storage](https://en.wikipedia.org/wiki/Storage_virtualization), [networking](https://en.wikipedia.org/wiki/Computer_network), [database](https://en.wikipedia.org/wiki/Database), [analytics](https://en.wikipedia.org/wiki/Analytics), [application services](https://en.wikipedia.org/wiki/Application_service_provider), [deployment](https://en.wikipedia.org/wiki/Software_deployment), [management](https://en.wikipedia.org/wiki/Systems_management), [mobile](https://en.wikipedia.org/wiki/Mobile_application_development), [developer tools](https://en.wikipedia.org/wiki/Programming_tool), and tools for the [Internet of Things](https://en.wikipedia.org/wiki/Internet_of_Things). The most popular include [Amazon Elastic Compute Cloud](https://en.wikipedia.org/wiki/Amazon_Elastic_Compute_Cloud) (EC2) and [Amazon Simple Storage Service](https://en.wikipedia.org/wiki/Amazon_Simple_Storage_Service) (Amazon S3). Most services are not exposed directly to end users, but instead offer functionality through [APIs](https://en.wikipedia.org/wiki/Application_programming_interface) for developers to use in their applications. Amazon Web Services' offerings are accessed over [HTTP](https://en.wikipedia.org/wiki/HTTP), using the [REST](https://en.wikipedia.org/wiki/Representational_State_Transfer) architectural style and [SOAP](https://en.wikipedia.org/wiki/SOAP_(protocol)) protocol.

Amazon markets AWS to subscribers as a way of obtaining large scale computing capacity more quickly and cheaply than building an actual physical server farm.[[7]](https://en.wikipedia.org/wiki/Amazon_Web_Services#cite_note-7) All services are billed based on usage, but each service measures usage in varying ways.



**Compute services**

Compute services constitute the fundamental element of cloud computing systems. The fundamen- tal service in this space is Amazon EC2, which delivers an IaaS solution that has served as a refer- ence model for several offerings from other vendors in the same market segment. Amazon EC2 allows deploying servers in the form of virtual machines created as instances of a specific image. Images come with a preinstalled operating system and a software stack, and instances can be con- figured for memory, number of processors, and storage. Users are provided with credentials to remotely access the instance and further configure or install software if needed.

**Amazon machine images**

Amazon Machine Images (AMIs) are templates from which it is possible to create a virtual machine. They are stored in Amazon S3 and identified by a unique identifier in the form of ami-xxxxxx and a manifest XML file. An AMI contains a physical file system layout with a predefined operating system installed. These are specified by the Amazon Ramdisk Image (ARI, id: ari-yyyyyy) and the Amazon Kernel Image (AKI, id: aki-zzzzzz), which are part of the configuration of the template. AMIs are either created from scratch or “bundled” from existing EC2 instances.

**What Is Amazon EC2?**

Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the Amazon Web Services (AWS) cloud. Using Amazon EC2 eliminates your need to invest in hardware up front, so you can develop and deploy applications faster. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage. Amazon EC2 enables you to scale up or down to handle changes in requirements or spikes in popularity, reducing your need to forecast traffic.

**Features of Amazon EC2**:

Virtual computing environments, known as instances

Preconfigured templates for your instances, known as Amazon Machine Images (AMIs), that package the bits you need for your server (including the operating system and additional software)

Various configurations of CPU, memory, storage, and networking capacity for your instances, known as instance types

Secure login information for your instances using key pairs (AWS stores the public key, and you store the private key in a secure place)

Storage volumes for temporary data that's deleted when you stop or terminate your instance, known as instance store volumes

Persistent storage volumes for your data using Amazon Elastic Block Store (Amazon EBS), known as Amazon EBS volumes

Multiple physical locations for your resources, such as instances and Amazon EBS volumes, known as Regions and Availability Zones

A firewall that enables you to specify the protocols, ports, and source IP ranges that can reach your instances using security groups

Static IPv4 addresses for dynamic cloud computing, known as Elastic IP addresses

Metadata, known as tags, that you can create and assign to your Amazon EC2 resources

Virtual networks you can create that are logically isolated from the rest of the AWS cloud, and that you can optionally connect to your own network, known as virtual private clouds (VPCs)

**S3**

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. This means customers of all sizes and industries can use it to store and protect any amount of data for a range of use cases, such as websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics. Amazon S3 provides easy-to-use management features so you can organize your data and configure finely-tuned access controls to meet your specific business, organizational, and compliance requirements. Amazon S3 is designed for 99.999999999% (11 9's) of durability, and stores data for millions of applications for companies all around the world.

**Benefits**

**Industry-leading performance, scalability, availability, and durability**

Scale your storage resources up and down to meet fluctuating demands, without upfront investments or resource procurement cycles. Amazon S3 is designed for 99.999999999% (11 9’s) of data durability because it automatically creates and stores copies of all S3 objects across multiple systems. This means your data is available when needed and protected against failures, errors, and threats.

**Wide range of cost-effective storage classes**

Save costs without sacrificing performance by storing data across the S3 Storage Classes, which support different data access levels at corresponding rates. You can use S3 Storage Class Analysis to discover data that should move to a lower-cost storage class based on access patterns, and configure an S3 Lifecycle policy to execute the transfer. You can also store data with changing or unknown access patterns in S3 Intelligent-Tiering, which tiers objects based on changing access patterns and automatically delivers cost savings.

**Unmatched security, compliance, and audit capabilities**

Store your data in Amazon S3 and secure it from unauthorized access with encryption features and access management tools. S3 is the only object storage service that allows you to block public access to all of your objects at the bucket or the account level with [S3 Block Public Access](https://aws.amazon.com/s3/features/block-public-access/). S3 maintains compliance programs, such as PCI-DSS, HIPAA/HITECH, FedRAMP, EU Data Protection Directive, and FISMA, to help you meet regulatory requirements. AWS also supports numerous auditing capabilities to monitor access requests to your S3 resources.

**Easily manage data and access controls**

S3 gives you robust capabilities to manage access, cost, replication, and data protection. S3 Access Points make it easy to manage data access with specific permissions for your applications using a shared data set. S3 Replication features manage data replication within the region or to other regions. S3 Batch Operations helps manage large scale changes across billions of objects. Since S3 works with AWS Lambda, you can log activities, define alerts, and automate workflows without managing additional infrastructure.

**Query-in-place services for analytics**

Run big data analytics across your S3 objects (and other data sets in AWS) with our query-in-place services. Use Amazon Athena to query S3 data with standard SQL expressions and Amazon Redshift Spectrum to analyze data that is stored across your AWS data warehouses and S3 resources. You can also use S3 Select to retrieve subsets of object data, instead of the entire object, and improve query performance by up to 400%.

**Most supported cloud storage service**

Store and protect your data in Amazon S3 by working with a partner from the AWS Partner Network (APN) — the largest community of technology and consulting cloud services providers. The APN recognizes migration partners that transfer data to Amazon S3 and storage partners that offer S3-integrated solutions for primary storage, backup and restore, archive, and disaster recovery. You can also purchase an AWS-integrated solution directly from the AWS Marketplace, which lists over 250 storage-specific offerings.

Google App Engine



Google App Engine is a system that exposes various pieces of Googles scalable infrastructure so that you can write server-side applications on top of them . Simply this is a platform which allows users to run and host their web applications on Googles infrastructure. These applications are easy to build, easy to maintain and easy to scale whenever traﬃc and data storage needed. By using Googles App Engine, there are no servers to maintain and no administrators needed. The idea is user just to upload his application and it is ready to serve its own customers. User has a choice either his product to be served by the free domain appspot.com or to allow Google Apps to serve it from domain chosen by the customer.

Google also provide the user with the option to limit the access of the application within the members of his own organization or to share it with the rest of the world. The starting packet is free of charge and additional obligation. All the user have to do is to sign up for a free account, and then to develop and publish his own application. The starting package includes up to 500MB of storage and enough CPU power and bandwidth to Google App Engine lets you run your web applications on Google’s infrastructure. App Engine applications are easy to build, easy to maintain, and easy to scale as your traﬃc and data storage needs grow. With App Engine, there are no servers to maintain: You just upload your application, and it’s ready to serve your users. You can serve your app from your own domain name (such as http://www.example.com/) using Google Apps. Or, you can serve your app using a free name on the appspot.com domain. You can share your application with the world, or limit access to members of your organization.

Google App Engine supports apps written in several program- ming languages. With App Engine’s Java runtime environment, you can build your app using standard Java technologies, including the JVM, Java servlets, and the Java programming languageor any other language using a JVM-based interpreter or compiler, such as JavaScript or Ruby. App Engine also features a dedicated Python runtime environment, which includes a fast Python interpreter and the Python stan- dard library. The Java and Python runtime environments are built to ensure that your application runs quickly, securely, and without interference from other apps on the system. With App Engine, you only pay for what you use. There are no set-up costs and no recurring fees. The resources your application uses, such as storage and bandwidth, are measured by the gigabyte, and billed at competitive rates.

You control the maximum amounts of resources your app can consume, so it always stays within your budget. App Engine costs nothing to get started. All applications can use up to 500 MB of storage and enough CPU and bandwidth to support an eﬃcient app serving around 5 million page views a month, absolutely free. When you enable billing for your application, your free limits are raised, and you only pay for resources you use above the free levels.

**HOW DOES IT WORK?**

The Application Environment With this new service provided by Google it is really easy to create reliably appli- cations which runs under heavy load and which use large amounts of data. Several key features are included in the environment

• dynamic web serving, with full support for common web technologies.

• persistent storage with queries, sorting and transactions. • automatic scaling and load balancing.

• APIs for authenticating users and sending email using Google Accounts.

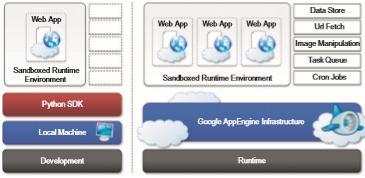
• a fully featured local development environment that simulates Google App Engine on users computer.

• task queues for performing work outside of the scope of a web request.

• scheduled tasks for triggering events at speciﬁed times and regular intervals.

Implementation of Google App Engine applications is done under Python pro- gramming language. Full Python language support along with most of the Python standard library comes with standard runtime environment. Currently Python is the only supported language by Google App Engine, but improvements to support other languages are in progress

**Service provided by GAE**



Google Cloud Computing Services Google App En- gine In the Platform as a Service (PaaS) space Google is a key player. App Engine is a platform to create, store and run applications on Googles servers using de- velopment languages as java and python. App Engine includes tools for manag- ing the data store, monitoring the site and its resource consumption, and debug- ging and logging. A user can serve the app from his own domain name (such as http://www.example.com/) using Google Apps. Or, he can serve his app using a free name on the appspot.com domain. A user can share his application with the world, or limit access to members of organization. App Engine costs nothing to get started. All applications can use up to 1 GB of storage and enough CPU and bandwidth to support an eﬃcient app serving around 5 million page views a month, absolutely free.

Google App Engine enables users to build a basic web application very quickly. Conﬁguring and setting up an application is quick and easy. The Google App Engine Architecture provides a new approach without deal- ing with web servers and load balancers but instead deploying the applications on the Google App Engine cloud by providing instance access and scalability which is showing in the ﬁgure 2. The Google App Engine Software Development Kit (SDK) provides Java and Python programming languages. The languages have their own web server application that contains all Google App Engine services on a local com- puter. The web server also simulates a secure sandbox environment. The Google App Engine SDK has APIs and libraries including the tools to upload applications. The Architecture deﬁnes the structure of applications that run on the Google App Engine.

The Sandbox All user applications operate in a secure environment. This environment has a limited access to the underlying operating system. Because of these limitations, App Engine is able to distribute applications web requests across various servers, which allows starting and stopping the servers to meet traﬃc demand. The sand- box separates the application in its own protected and reliable environment which is independent of the operating system, hardware or the physical location of the web server. Here are some of the restrictions which are included in the sandbox environment:

• An application can only access other computers on the Internet through the provided URL fetch and email services and APIs. Other computers can only connect to the application by making HTTP (or HTTPS) requests on the standard ports.

• An application cannot write to the ﬁle system and can read ﬁles, but only ﬁles uploaded with the application code. The application must use the App Engine datastore for all data that persists between requests.

• Application code only runs in response to a web request, and must return response data within 30 seconds. A request handler cannot spawn a sub- process or execute code after the response has been sent.

The Languages Runtime Your application can run in one of two runtime environments: the JAVA environ- ment, and the PYTHON environment. Each environment provides standard proto- cols and common technologies for web application development. The Java Runtime Environment You can develop your application for the Java runtime environment using com- mon Java web development tools and API standards. Your app interacts with the environment using the Java Servlet standard, and can use common web application technologies such as JavaServer Pages (JSPs). The Java runtime environment uses Java 6.

The App Engine Java SDK supports developing apps using either Java 5 or 6. The environment includes the Java SE Runtime Environment (JRE) 6 platform and libraries. The restrictions of the sandbox environment are implemented in the JVM. An app can use any JVM bytecode or library feature, as long as it does not exceed the sandbox restrictions. For instance, bytecode that attempts to open a socket or write to a ﬁle will throw a runtime exception. Your app accesses most App Engine services using Java standard APIs. For the App Engine datastore, the Java SDK includes implementations of the Java Data Objects (JDO) and Java Per- sistence API (JPA) interfaces. Your app can use the JavaMail API to send email messages with the App Engine Mail service. The java.net HTTP APIs access the App Engine URL fetch service. App Engine also includes low-level APIs for its ser- vices to implement additional adapters, or to use

directly from the application. See the documentation for the datastore, URL fetch, mail, images and Google Accounts APIs. Typically, Java developers use the Java programming language and APIs to implement web applications for the JVM. With the use of JVM-compatible compil- ers or interpreters, you can also use other languages to develop web applications, such as JavaScript, Ruby, or Scala. For more information about the Java runtime environment, see The Java Runtime Environment.

The Python Runtime Environment With App Engine’s Python runtime environment, you can implement your app using the Python programming language, and run it on an optimized Python in- terpreter. App Engine includes rich APIs and tools for Python web application development, including a feature rich data modeling API, an easy-to-use web appli- cation framework, and tools for managing and accessing your app’s data. You can also take advantage of a wide variety of mature libraries and frameworks for Python web application development, such as Django.

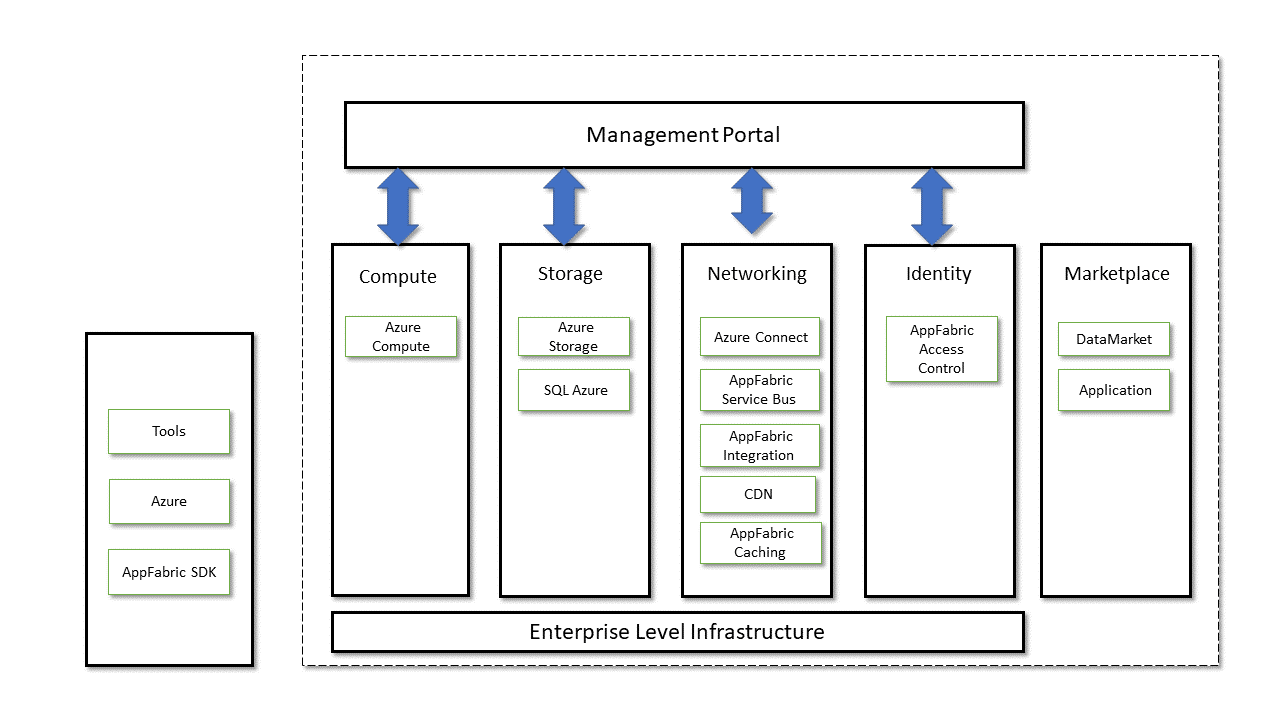
The Python runtime environment uses Python version 2.5.2. Additional support for Python 3 is being considered for a future release. The Python environment includes the Python standard library. Of course, not all of the library’s features can run in the sandbox environment. For instance, a call to a method that attempts to open a socket or write to a ﬁle will raise an exception. For convenience, several modules in the standard library whose core features are not supported by the runtime environment have been disabled, and code that imports them will raise an error. Application code written for the Python environment must be written exclusively in Python. Extensions written in the language are not supported.The Python environment provides rich Python APIs for the datastore, Google Accounts, URL fetch, and email services. App Engine also provides a simple Python web application framework called webapp to make it easy to start building applications. You can upload other third-party libraries with your application, as long as they are implemented in pure Python and do not require any unsupported standard library modules.

MICROSOFT AZURE:

Microsoft Azure (formerly Windows Azure) is a [cloud computing](https://en.wikipedia.org/wiki/Cloud_computing) service created by [Microsoft](https://en.wikipedia.org/wiki/Microsoft) for building, testing, deploying, and managing applications and services through Microsoft-managed [data centers](https://en.wikipedia.org/wiki/Data_center). It provides [software as a service (SaaS)](https://en.wikipedia.org/wiki/Software_as_a_service), [platform as a service (PaaS)](https://en.wikipedia.org/wiki/Platform_as_a_service) and [infrastructure as a service (IaaS)](https://en.wikipedia.org/wiki/Infrastructure_as_a_service) and supports many different [programming languages](https://en.wikipedia.org/wiki/Programming_language), tools and frameworks, including both Microsoft-specific and third-party software and systems.

Azure was announced in October 2008, started with codename "Project Red Dog",[[1]](https://en.wikipedia.org/wiki/Microsoft_Azure" \l "cite_note-1) and released on February 1, 2010, as "Windows Azure" before being renamed "Microsoft Azure" on March 25, 2014

**Architecture of Microsoft Azure:**



**Compute services**

Virtual machines, [infrastructure as a service](https://en.wikipedia.org/wiki/Infrastructure_as_a_service) (IaaS) allowing users to launch general-purpose [Microsoft Windows](https://en.wikipedia.org/wiki/Microsoft_Windows) and [Linux](https://en.wikipedia.org/wiki/Linux) virtual machines, as well as preconfigured machine images for popular software packages.

Most users run Linux on Azure, some of the many [Linux distributions](https://en.wikipedia.org/wiki/Linux_distribution) offered, including Microsoft's own [Linux](https://en.wikipedia.org/wiki/Linux_kernel)-based [Azure Sphere](https://en.wikipedia.org/wiki/Azure_Sphere).

App services, [platform as a service](https://en.wikipedia.org/wiki/Platform_as_a_service) (PaaS) environment letting developers easily publish and manage websites.

[Websites](https://en.wikipedia.org/wiki/Microsoft_Azure_Web_Sites), high density hosting of websites allows developers to build sites using [ASP.NET](https://en.wikipedia.org/wiki/ASP.NET), [PHP](https://en.wikipedia.org/wiki/PHP), [Node.js](https://en.wikipedia.org/wiki/Node.js), or [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) and can be deployed using [FTP](https://en.wikipedia.org/wiki/File_Transfer_Protocol), [Git](https://en.wikipedia.org/wiki/Git_(software)" \o "Git (software)), [Mercurial](https://en.wikipedia.org/wiki/Mercurial), [Team Foundation Server](https://en.wikipedia.org/wiki/Team_Foundation_Server) or uploaded through the user portal. This feature was announced in preview form in June 2012 at the Meet Microsoft Azure event.[[7]](https://en.wikipedia.org/wiki/Microsoft_Azure#cite_note-meetnew-7) Customers can create websites in PHP, ASP.NET, Node.js, or Python, or select from several open source applications from a gallery to deploy.

This comprises one aspect of the [platform as a service](https://en.wikipedia.org/wiki/Platform_as_a_service) (PaaS) offerings for the Microsoft Azure Platform. It was renamed to Web Apps in April 2015.

WebJobs, applications that can be deployed to an App Service environment to implement background processing that can be invoked on a schedule, on demand, or run continuously. The Blob, Table and Queue services can be used to communicate between WebApps and WebJobs and to provide state.

**Mobile services**

Mobile Engagement collects real-time analytics that highlight users’ behavior. It also provides push notifications to mobile devices.

HockeyApp can be used to develop, distribute, and beta-test mobile apps.

**Storage services**

Storage Services provides [REST](https://en.wikipedia.org/wiki/REST) and [SDK](https://en.wikipedia.org/wiki/Software_development_kit) [APIs](https://en.wikipedia.org/wiki/API) for storing and accessing data on the cloud.

Blob Service allows programs to store unstructured text and binary data as blobs that can be accessed by a HTTP(S) path. Blob service also provides security mechanisms to control access to data.

Queue Service lets programs communicate asynchronously by message using queues.

File Service allows storing and access of data on the cloud using the [REST](https://en.wikipedia.org/wiki/REST) APIs or the [SMB protocol](https://en.wikipedia.org/wiki/SMB_protocol).

**Data management**

Azure Search provides text search and a subset of [OData](https://en.wikipedia.org/wiki/OData" \o "OData)'s structured filters using REST or SDK APIs.

[Cosmos DB](https://en.wikipedia.org/wiki/Cosmos_DB) is a [NoSQL](https://en.wikipedia.org/wiki/NoSQL" \o "NoSQL) database service that implements a subset of the SQL SELECT statement on [JSON](https://en.wikipedia.org/wiki/JSON) documents.

Redis Cache is a managed implementation of [Redis](https://en.wikipedia.org/wiki/Redis" \o "Redis).

[StorSimple](https://en.wikipedia.org/wiki/StorSimple) manages storage tasks between on-premises devices and cloud storage

[SQL Database](https://en.wikipedia.org/wiki/SQL_Azure), formerly known as [SQL Azure](https://en.wikipedia.org/wiki/SQL_Azure) Database, works to create, scale and extend applications into the cloud using [Microsoft SQL Server](https://en.wikipedia.org/wiki/Microsoft_SQL_Server) technology. It also integrates with [Active Directory](https://en.wikipedia.org/wiki/Active_Directory) and [Microsoft System Center](https://en.wikipedia.org/wiki/Microsoft_System_Center) and [Hadoop](https://en.wikipedia.org/wiki/Apache_Hadoop" \o "Apache Hadoop).

Azure SQL Data Warehouse is a fully managed cloud data warehouse for enterprises of any size that combines lightning-fast query performance with industry-leading data security.

[Azure Data Factory](https://en.wikipedia.org/w/index.php?title=Azure_Data_Factory&action=edit&redlink=1), is a data integration service that allows creation of data-driven workflows in the cloud for orchestrating and automating data movement and data transformation.

[Azure Data Lake](https://en.wikipedia.org/wiki/Azure_Data_Lake) is a scalable data storage and analytic service for [big data](https://en.wikipedia.org/wiki/Big_data) analytics workloads that require developers to run massively parallel queries.

Azure HDInsightis a big data relevant service, that deploys [Hortonworks](https://en.wikipedia.org/wiki/Hortonworks" \o "Hortonworks) [Hadoop](https://en.wikipedia.org/wiki/Hadoop" \o "Hadoop) on Microsoft Azure, and supports the creation of Hadoop clusters using Linux with Ubuntu.

[Azure Stream Analytics](https://en.wikipedia.org/wiki/Azure_Stream_Analytics) is a [Serverless](https://en.wikipedia.org/wiki/Serverless_computing" \o "Serverless computing) scalable event processing engine that enables users to develop and run real-time analytics on multiple streams of data from sources such as devices, sensors, web sites, social media, and other applications.

**Media services**

A PaaS offering that can be used for encoding, [content protection](https://en.wikipedia.org/wiki/Content_protection), streaming, or [analytics](https://en.wikipedia.org/wiki/Analytics).

CDN

A global [content delivery network](https://en.wikipedia.org/wiki/Content_delivery_network) (CDN) for audio, video, applications, images, and other static files. It can be used to cache static assets of websites geographically closer to users to increase performance. The network can be managed by a REST based HTTP API.

Azure has 54 point of presence locations worldwide (also known as Edge locations) as of August 2018.

Developer

Application Insights

[Azure DevOps](https://en.wikipedia.org/wiki/Azure_DevOps_Services)

**Management**

Azure Automation, provides a way for users to automate the manual, long-running, error-prone, and frequently repeated tasks that are commonly performed in a cloud and enterprise environment. It saves time and increases the reliability of regular administrative tasks and even schedules them to be automatically performed at regular intervals. You can automate processes using runbooks or automate configuration management using Desired State Configuration.[[21]](https://en.wikipedia.org/wiki/Microsoft_Azure#cite_note-21)

[Microsoft SMA](https://en.wikipedia.org/wiki/Microsoft_SMA_(software))

**Functions**

Azure functions are used in [serverless computing](https://en.wikipedia.org/wiki/Serverless_computing" \o "Serverless computing) architectures where subscribers can execute code as a Function-as-a-Service ([FaaS](https://en.wikipedia.org/wiki/Function_as_a_service" \o "Function as a service)) without managing the underlying server resources.[[29]](https://en.wikipedia.org/wiki/Microsoft_Azure#cite_note-29)

Design

Microsoft Azure uses a specialized operating system, called Microsoft Azure, to run its "fabric layer":[[41]](https://en.wikipedia.org/wiki/Microsoft_Azure" \l "cite_note-41) A cluster hosted at Microsoft's data centers that manages computing and storage resources of the computers and provisions the resources (or a subset of them) to applications running on top of Microsoft Azure. Microsoft Azure has been described as a "cloud layer" on top of a number of Windows Server systems, which use Windows Server 2008 and a customized version of [Hyper-V](https://en.wikipedia.org/wiki/Hyper-V), known as the Microsoft Azure Hypervisor to provide virtualization of services.[[42]](https://en.wikipedia.org/wiki/Microsoft_Azure#cite_note-:0-42)

Scaling and reliability are controlled by the Microsoft Azure Fabric Controller, which ensures the services and environment do not fail if one or more of the servers fails within the Microsoft [data center](https://en.wikipedia.org/wiki/Data_center), and which also provides the management of the user's Web application such as memory allocation and load balancing.[[42]](https://en.wikipedia.org/wiki/Microsoft_Azure#cite_note-:0-42)

Azure provides an [API](https://en.wikipedia.org/wiki/Application_programming_interface) built on [REST](https://en.wikipedia.org/wiki/Representational_state_transfer), [HTTP](https://en.wikipedia.org/wiki/HTTP), and [XML](https://en.wikipedia.org/wiki/XML) that allows a developer to interact with the services provided by Microsoft Azure. Microsoft also provides a client-side managed class library that encapsulates the functions of interacting with the services. It also integrates with [Microsoft Visual Studio](https://en.wikipedia.org/wiki/Microsoft_Visual_Studio), [Git](https://en.wikipedia.org/wiki/Git_(software)" \o "Git (software)), and [Eclipse](https://en.wikipedia.org/wiki/Eclipse_(software)).[[43]](https://en.wikipedia.org/wiki/Microsoft_Azure#cite_note-43)[[44]](https://en.wikipedia.org/wiki/Microsoft_Azure#cite_note-44)[[45]](https://en.wikipedia.org/wiki/Microsoft_Azure#cite_note-45)

In addition to interacting with services via API, users can manage Azure services using the Web-based Azure Portal, which reached General Availability in December 2015.[[46]](https://en.wikipedia.org/wiki/Microsoft_Azure#cite_note-46) The portal allows users to browse active resources, modify settings, launch new resources, and view basic monitoring data from active virtual machines and services.

**Deployment models**

Microsoft Azure offers two deployment models for cloud resources: the "classic" deployment model and the Azure Resource Manager. In the classic model, each Azure resource (virtual machine, SQL database, etc.) was managed individually. The Azure Resource Manager, introduced in 2014,[[47]](https://en.wikipedia.org/wiki/Microsoft_Azure#cite_note-arm_vs_classic-47) enables users to create groups of related services so that closely coupled resources can be deployed, managed, and monitored together.

**Privacy**

Microsoft has stated that, per the [USA Patriot Act](https://en.wikipedia.org/wiki/USA_Patriot_Act), the US government could have access to the data even if the hosted company is not American and the data resides outside the USA.[[63]](https://en.wikipedia.org/wiki/Microsoft_Azure#cite_note-63) However, Microsoft Azure is compliant with the [E.U. Data Protection Directive (95/46/EC)](https://en.wikipedia.org/wiki/Data_Protection_Directive).[[64]](https://en.wikipedia.org/wiki/Microsoft_Azure#cite_note-64)[[65]](https://en.wikipedia.org/wiki/Microsoft_Azure#cite_note-65)[[*contradictory*](https://en.wikipedia.org/wiki/Category:Articles_contradicting_other_articles)] To manage privacy and security-related concerns, Microsoft has created a Microsoft Azure Trust Center,[[66]](https://en.wikipedia.org/wiki/Microsoft_Azure#cite_note-66) and Microsoft Azure has several of its services compliant with several compliance programs including [ISO 27001:2005](https://en.wikipedia.org/wiki/ISO_27001:2005) and [HIPAA](https://en.wikipedia.org/wiki/HIPAA). A full and current listing can be found on the Microsoft Azure Trust Center Compliance page.[[67]](https://en.wikipedia.org/wiki/Microsoft_Azure#cite_note-67) Of special note, Microsoft Azure has been granted JAB Provisional Authority to Operate (P-ATO) from the U.S. government in accordance with guidelines spelled out under the [Federal Risk and Authorization Management Program](https://en.wikipedia.org/wiki/Federal_Risk_and_Authorization_Management_Program) (FedRAMP), a U.S. government program that provides a standardized approach to security assessment, authorization, and continuous monitoring for cloud services used by the federal government.[[68]](https://en.wikipedia.org/wiki/Microsoft_Azure#cite_note-68)

Significant outages



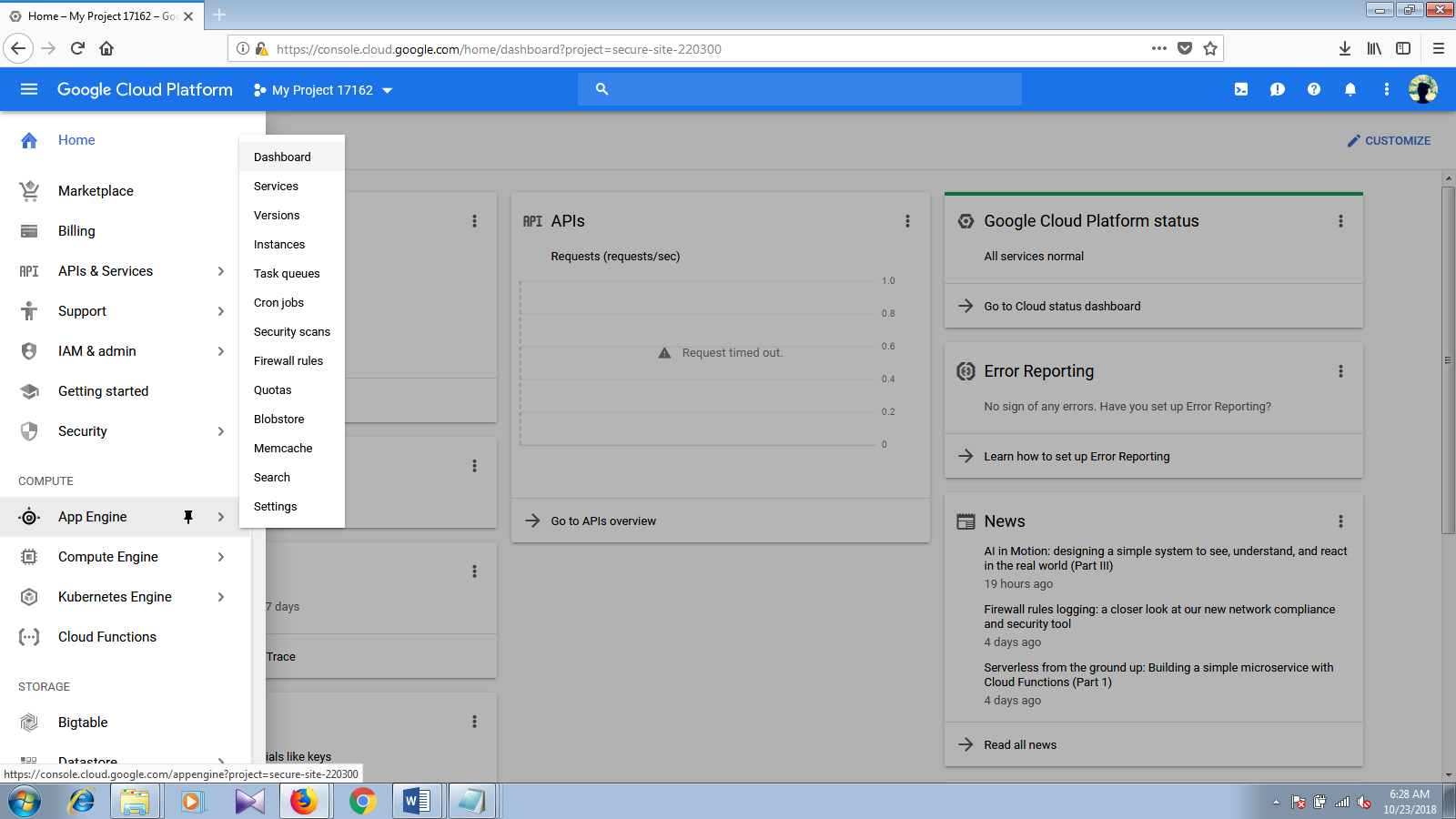
**TASK 2 - To Use Google Cloud Platform inorder to deploy an application**

**AIM –** To make use of Google App Engine in order to deploy an application/ Mobile Recharge Portal by the end user.

**PROCEDURE :**

**Step 1 :**

Open your browser and visit <https://console.cloud.google.com/>



**Step 2 :**

Create new project after accepting terms of service corresponding to the app engine.

**Step 3:**

Select an interface language through which you want to develop the app(Java, Ruby, Python, php ect.)

**Step 4:**

Select a region from which you want your app to be served. The default region will be us-central( You may choose any among the available servers)

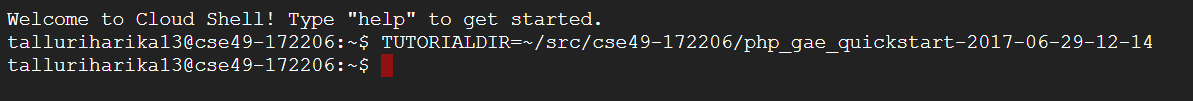
**Step 5 :**

If you are making use of app engine for the first time, it provides you with Appengine quick start that enable you to understand how to operate with the cloud. Following the instructions of quick start helps us to make use of hello world program which is already integrated with the cloud.

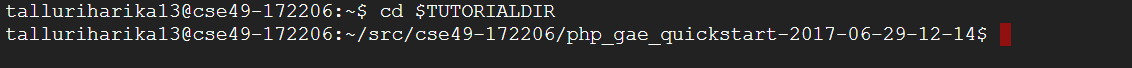
**Step 6 :**

The commands provided by the quick start are to be executed in the Cloud Shell as follows

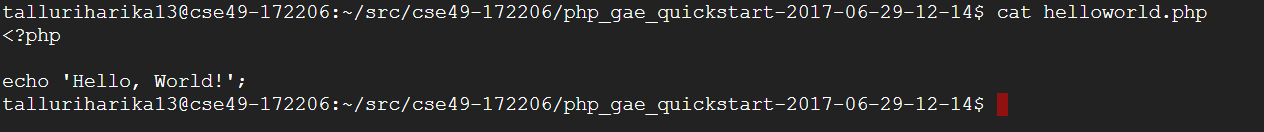
sahithi12345678@cse49-172206:~$TUTORIALDIR=~/src/cse49-172206/php\_gae\_quickstart-2017-06-29-12-14



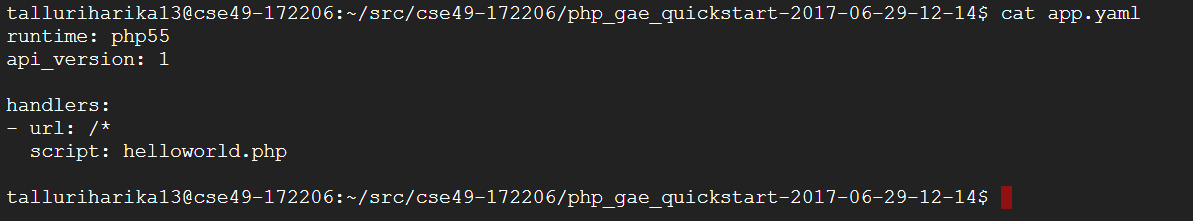
sahithi12345678@cse49-172206:~$ cd $TUTORIALDIR



sahithi12345678@cse49-172206:~/src/cse49-172206/php\_gae\_quickstart-2017-06-29-12-14$ cat helloworld.php



sahithi12345678@cse49-172206:~/src/cse49-172206/php\_gae\_quickstart-2017-06-29-12-14$ cat app.yaml

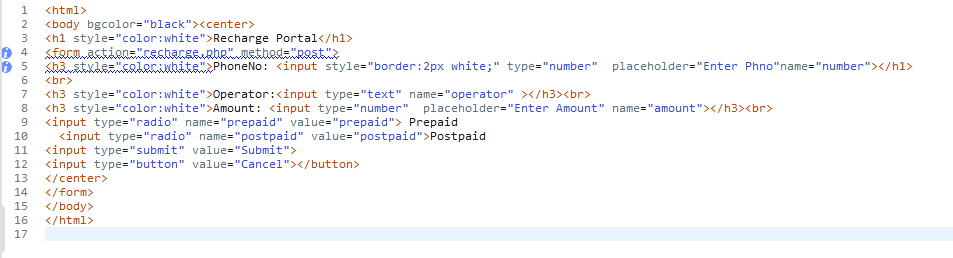


**Step 7:**

Now your helloworld program is ready and the final result can be viewed by selecting the option of web preview. We need select a port number to view the result. The default port is 8080.

**Working with the Google App Engine using HTML – PHP simple mobile recharge portal application**

**Step 1: PROGRAM :**

****

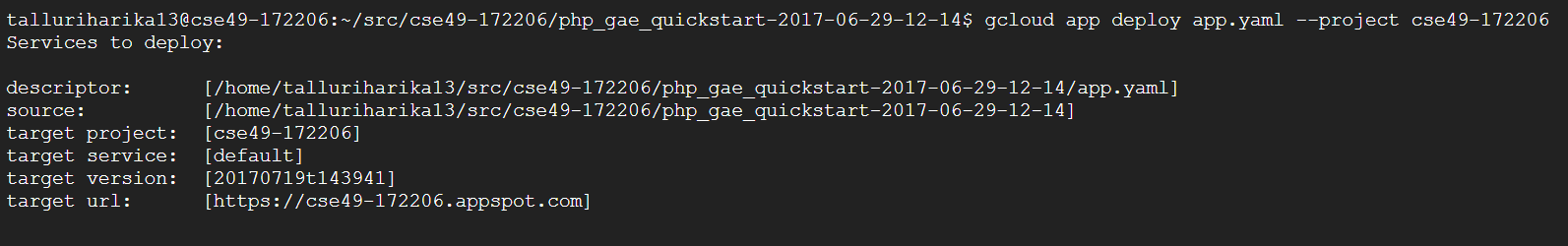
**Step 2:**

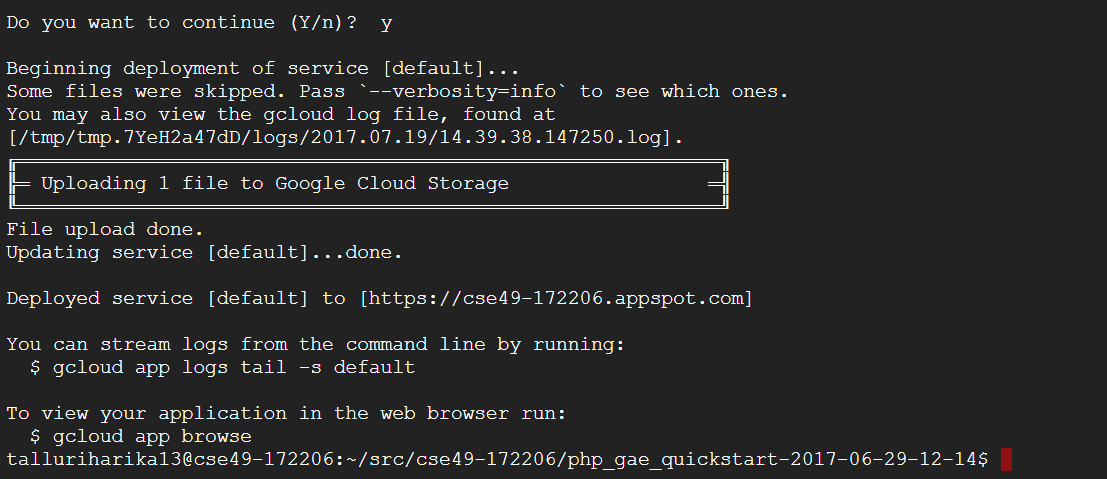
Create PHP file so that we can make use of it in order to validate the previously created HTML form

****

**Step 3:**

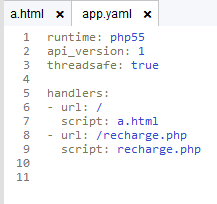
Since the two programs are ready, now we need to deploy the two programs into our project directory present in the cloud.

sahithi12345678@cse49-172206:~/src/cse49-172206/php\_gae\_quickstart-2017-06-29-12-14$ gcloud app deploy app.yaml --project 



**Step 4:**

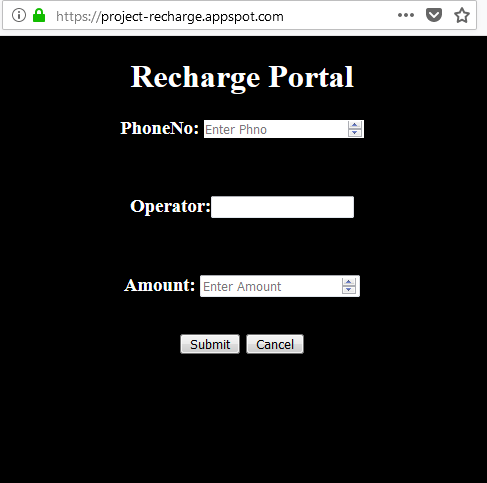
In order to run the application, we need to modify the app.yaml file as follows

****

**Now we can open the recharge portal with the url provided by the App engine.**

[**https://project-recharge.appspot.com/**](https://project-recharge.appspot.com/)

**The portal looks as follows :**

****

**When we enter the Phone No, Operator and Amount a message is been displayed as follows**



**RESULT** : Thus we have deployed a simple recharge portal using GoogleAppEngine Successfully**.**