**Practical-5 Platform as a service using AWS**

**Name: Heeta Parmar**

**Roll no: A049**

**1)Platform as a Service (PaaS)**

Platform as a Service (PaaS) is a cloud computing model that provides a ready-to-use environment for developers to build, run, and manage applications without worrying about the underlying infrastructure. PaaS offers pre-configured tools, libraries, and frameworks that simplify the development and deployment process, helping developers focus on coding and application logic rather than server management and infrastructure maintenance. Examples of PaaS include AWS Elastic Beanstalk, Google App Engine, and Microsoft Azure App Service.

**2)AWS Elastic Beanstalk**

AWS Elastic Beanstalk is an easy-to-use PaaS solution that automates the deployment, scaling, and management of applications in the AWS Cloud. With Elastic Beanstalk, developers can deploy applications written in popular programming languages (like Python, Java, Node.js, PHP, Ruby, and more) by simply uploading code. Beanstalk handles tasks such as provisioning the necessary infrastructure, managing load balancing, auto-scaling, monitoring, and more. It’s particularly useful for fast deployments and quick iterations while maintaining flexibility over the underlying AWS services.

**3)Components of Elastic Beanstalk**

Elastic Beanstalk comprises several key components that work together to create and manage the environment in which applications run. These include:

1. Application: A container that holds different environments. Each environment runs a version of the application, allowing for testing and production environments within the same application container.

2. Environment: The specific infrastructure resources and settings used to run an application version. Each environment is associated with a URL, load balancer, and other resources that Elastic Beanstalk configures based on the application’s needs.

3. Environment Configuration: Settings applied to the environment, such as instance type, security groups, scaling policies, and software versions. These configurations can be updated as the application evolves.

4. Application Version: A deployable version of the code. Multiple versions can be stored and rolled back if necessary.

5. Environment Tier: The type of environment, either Web Server (used for web applications) or Worker (for asynchronous, background tasks).

6. Elastic Beanstalk CLI (EB CLI): A command-line tool that facilitates the management of applications and environments directly from the command line.

**4) IAM (Identity and Access Management)**

AWS Identity and Access Management (IAM) is a security service that enables users to securely control access to AWS services and resources. With IAM, users can manage permissions and access controls for various AWS resources, ensuring that only authorized individuals or applications can perform specific actions. Key IAM components include:

1. Users: Individual user accounts representing a person or service.

2. Groups: Collections of users to simplify permission management. Permissions applied to a group apply to all members.

3. Roles: Roles allow users or services to assume permissions temporarily, ideal for cross-account or temporary access.

4. Policies: JSON-based documents that define permissions and can be attached to users, groups, or roles to control access.

IAM plays a crucial role in Elastic Beanstalk by securing access to application environments and associated resources, allowing developers to manage permissions granularly.