Name: Harpreet Narwal

SAP ID: 500087230

Roll No: R2142201826

Subject: Cloud Application Development

Q: As a final output of the project, that you are expected to upload your designed applications on public cloud (AWS/Azure) and hence need to analyse and explain which application platform will you be following, and why?

Answer:

Application Platform followed:

Cloud Platforms: Chess engines will be deployed on cloud platforms, i.e. Amazon Web Services (AWS). This allows for scalable and flexible deployment, and is often used for large-scale chess engines.

While using the AWS cloud to deploy the application (Chess engine) the services required will be:

1. EC2: This service will provide the scalable computing capacity in the cloud. EC2 instance will help in running the chess engine.
2. S3 (Single Storage Service): This service will be used to store the chess engine and related resources, such as opening book and endgame table bases.
3. Amazon RDS (Relational Database Service): It will be used to store information about games played using the chess engine, and Amazon RDS to set up a database.
4. Amazon CloudFront: This service will be used to deliver content such as the chess engine’s interface, to users with low latency and high transfer speed.
5. AWS Lambda: It will be used to run parts of chess engine in a server-less environment, also AWS Lambda function can be used to perform specific tasks.
6. Amazon API Gateway: This service will be used to create, manage, and publish APIs to allow users to access the chess engine’s functionality.

The Architecture used by the chess engine:

Chess engine typically uses the MIMD (Multiple Instruction, Multiple Data) architecture.

In MIMD architecture, multiple processors run independently, each with its own instruction stream and working on its own data. This is well suited for a chess engine as each processor can be assigned a portion of the search tree to explore in parallel, reducing the total time required to find the best move.

With MIMD, multiple processors can work on different parts of the search tree simultaneously, allowing for a much faster calculation of the best move. This is especially useful for large and complex search trees, where a single processor would take a long time to explore the entire tree.

MIMD can be implemented using multiple threads, multiple cores, or multiple computers working together in a network. The specific implementation will depend on the requirements of your engine and the resources you have available.