**Project title:-**

Nim, a turn-based strategy game

**Introduction:-**

A basic Nim starts with two players and multiple heaps each containing multiple objects. Heaps are also called heaps and items are sometimes called stones.

Each player must take at least one stone in turn, but can take multiple stones as long as they all come from the same stack. You can empty the stack, effectively removing it from play. If the player cannot move, the game ends. Of course, as long as there are stones, each player can take them and move. Therefore, the end condition can be translated as the game ending when there are no more stones left.

In his normal Nim, the loser is the player who gets stuck. In combinatorial game theory, the state in which the last player to move in a normal game wins is called the normal state. In Nim, the player who cannot move wins instead. This corresponds to the loss of the player who took the last stone.

**Concept note:-**

Imagine creating an online application that allows users to play Nim, a turn-based strategy game. Nim has heaps of three objects. Two players take turns removing any number of objects from a single stack. The object of the game is to force other players to remove the last object.

(1) Amazon DynamoDB and for state storage

As part of my application, I need to save the state of an existing game. I also need to notify the user at various points in the game. Notify when a user invites you to a new game, when it's your turn, and when a winner is determined.

Amazon DynamoDB is used to store existing game state and persist it between rounds. Amazon SNS is used to notify players at important points in the game.

(2) Amazon SNS for SMS messaging

When creating a turn-based game, you need a way to contact your users and inform them about important events. This includes notifications that a new game has started, that it's your turn, or that the game has ended.

We use Amazon Simple Notification Service (Amazon SNS) to notify users. Amazon SNS is a fully managed messaging system that enables pub/sub functionality and direct messaging to SMS and email.

(3) Amazon Cognito for user authentication

I am using Amazon Cognito User Pools. Allow users to register through your application. After registering, you can log in via the client and receive his ID token. This ID token can be passed to your application as a header to authenticate the user.

(4) AWS Lambda for compute

we contribute our application code to AWS Lambda, a serverless computing service. With AWS Lambda, we don't have to worry about server management or capacity planning. Simply upload the code you want to run and AWS will execute it when the configured event triggers occur.

(5)Amazon API Gateway for HTTP routing.

**Literature review:-**

(1) DynamoDB is a fully managed NoSQL database that provides fast, consistent performance at any scale. It has a flexible billing model, tight integration with infrastructure as code, and a hands-off operational model.[1]

(2) Amazon DynamoDB is ideal for systems that require a high speed in data reading and writing.[2]

There could be many instances where we can use DynamoDB. Below are some of them:

* Real-time bidding
* Shopping carts
* High I/O needs
* Unstructured data from mobile applications and gaming software
* Content management

(3) SNS is a web service that coordinates and manages the delivery or sending of messages to subscribing endpoints or clients. Subscribers (i.e. web servers, email addresses, SQS queues, AWS Lambda functions), when subscribing to a topic, can use any of the supported protocols (i.e. SQS, HTTP/S, email, SMS, Lambda) Consume or receive messages or notifications via .[3]

(4) With Amazon Simple Notification Service (SNS), there are no upfront fees, no required minimum commitments, and no long-term contracts. You pay only for what you use, based on type of topic used. SNS topics are defined as Standard topics or FIFO topics. You will be charged at the end of the month for your usage.[4]

(5) In the case where you have multiple simultaneous events, Lambda simply spins up multiple copies of the function to handle the events. In other words, Lambda can be described as a type of function as a service (FaaS).[5]

(6) An API gateway is middleware that sits between an API endpoint and backend services, transmitting client requests to an appropriate service of an application. It’s also an architectural pattern, which was initially created to support microservices.[6]

(7) When you are building an application, you know that your users have probably more than one device. Being able to sync your user’s profile information whether save game data or any other kind of information is really important, so the users can have a great experience with your application whenever and wherever they use regardless of which device they use.

But building the back end you need to support that kind of storage and synchronization is a lot of work as you have to build it, deploy it and manage the infrastructure that runs on. Wouldn’t it be great if you could stay focused on riding your app without having to build your own back end just to support syncing and storing user’s data. That is where Amazon Cognito comes into the picture, which makes it really easy for you to manage user data for your apps across multiple mobiles or connected devices.[7]

References:-

[1] https://aws.amazon.com/getting-started/hands-on/purpose-built-databases/dynamodb/

[2] <https://blog.knoldus.com/a-simple-introduction-to-amazon-dynamodb/>

[3] https://jayendrapatil.com/aws-sns-simple-notification-service/

[4] https://aws.amazon.com/sns/pricing/

[5] https://www.bmc.com/blogs/aws-lambda/

[6] https://www.altexsoft.com/blog/api-gateway/

[7] https://k21academy.com/amazon-web-services/amazon-cognito/