

CLOUD COMPUTING

MINI PROJECT

PRESENTED BY

ANANYA SACHAN: 22070126010

APARNA IYER: 22070126017

HEVARDHAN S.: 22070126046

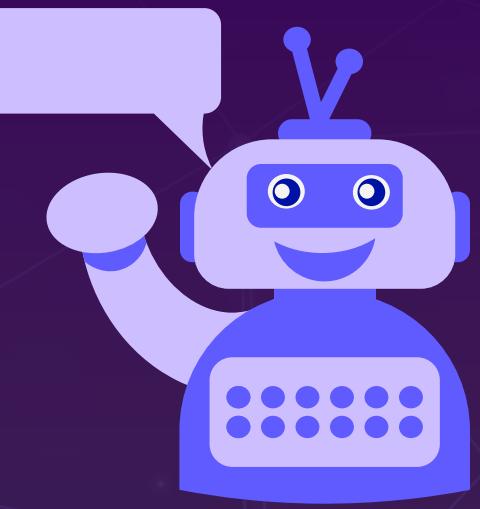
RIYA SHUKLA: 22070126090



INTRODUCTION

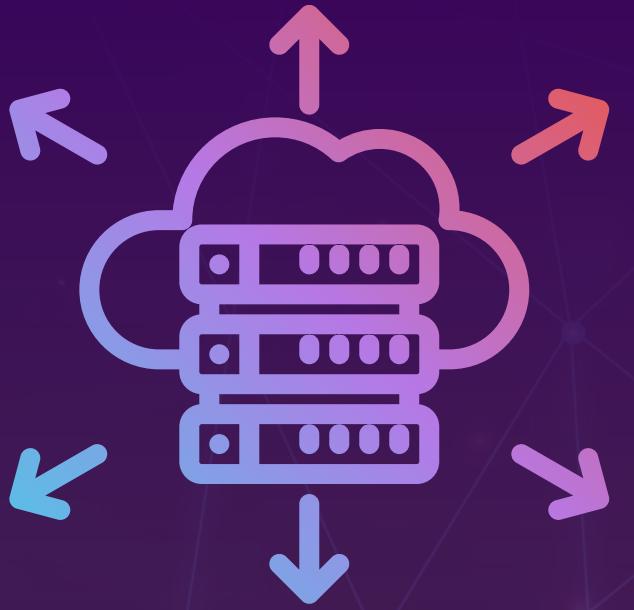


The project is centered around building a scalable and resilient infrastructure for hosting a web application on AWS. By leveraging cloud-native concepts such as EC2 Auto Scaling, Elastic Load Balancing, and EC2 instances, the setup aims to maintain consistent performance under fluctuating traffic conditions.

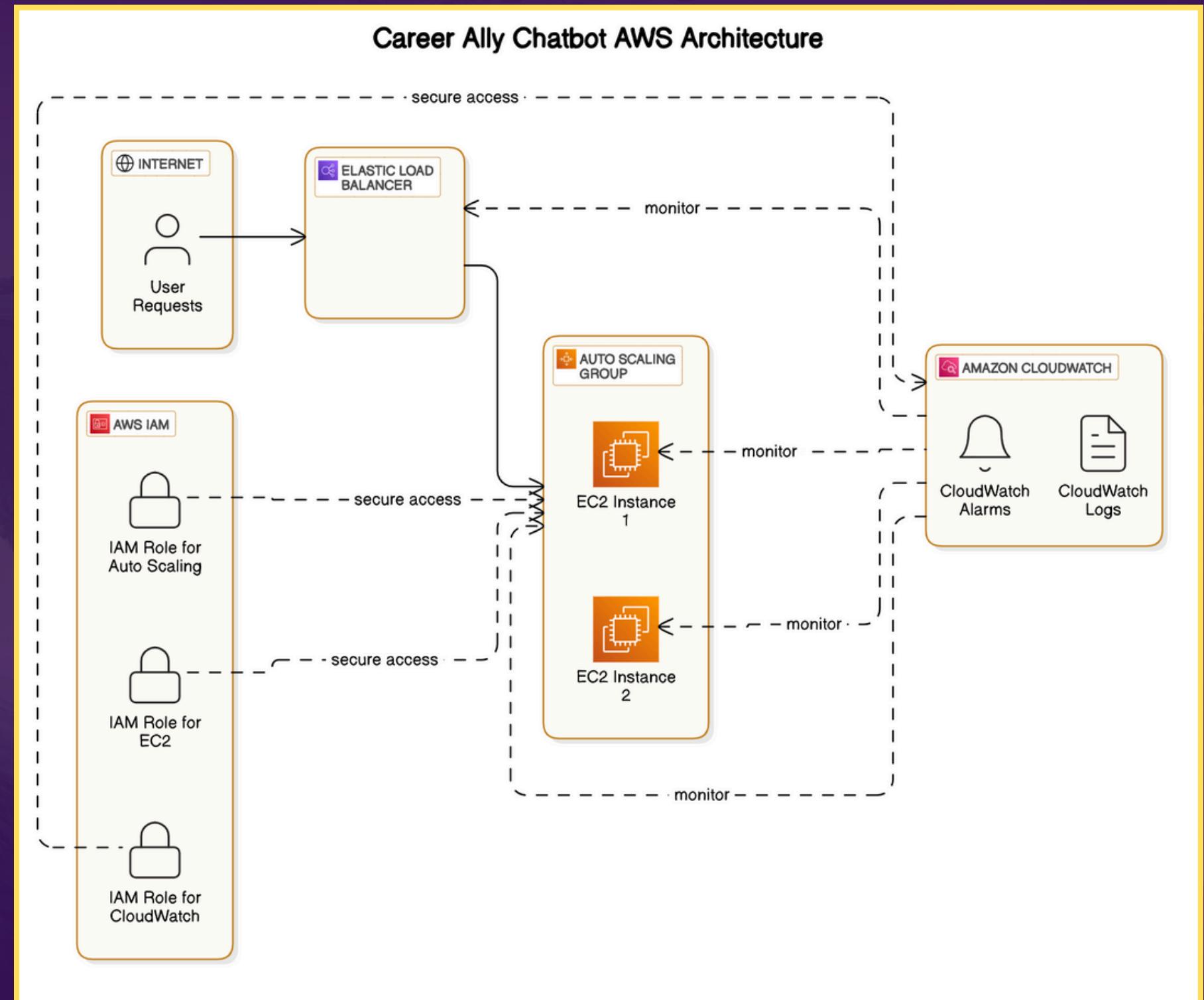


PROBLEM STATEMENT

The goal of this project is to design and implement a scalable, resilient, and highly available infrastructure on Amazon Web Services (AWS) to host a web application. As the traffic to the application fluctuates over time, it is crucial to ensure that the infrastructure can dynamically scale to handle both peak loads and periods of low demand without compromising performance, reliability, or cost-efficiency.



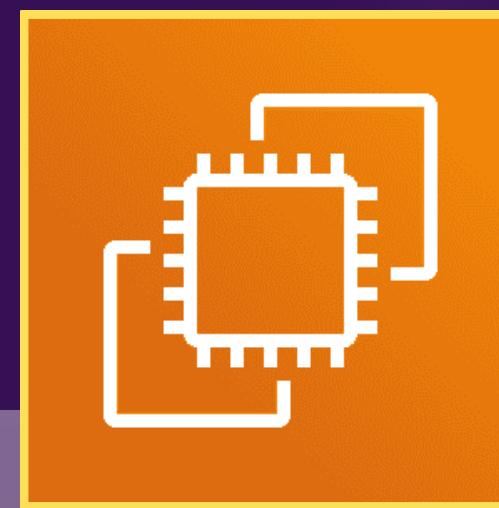
ARCHITECTURE



AWS SERVICES USED

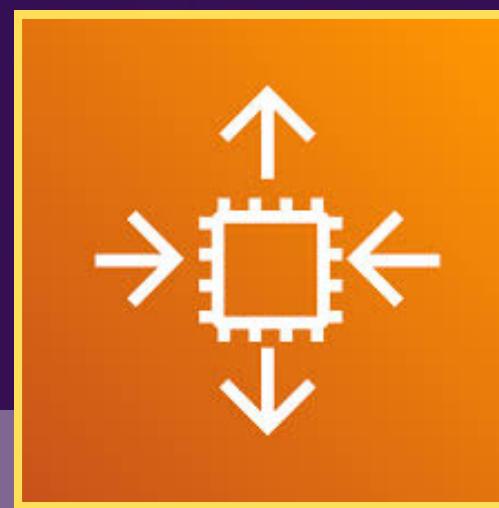
EC2 INSTANCES

Provides resizable virtual machines (VMs) (instances) for computing needs. Used to run the application software for Career Ally.



AUTOSCALING GROUP

To automatically adjust the number of EC2 instances in response to traffic demand, keeping the application available and cost-effective.

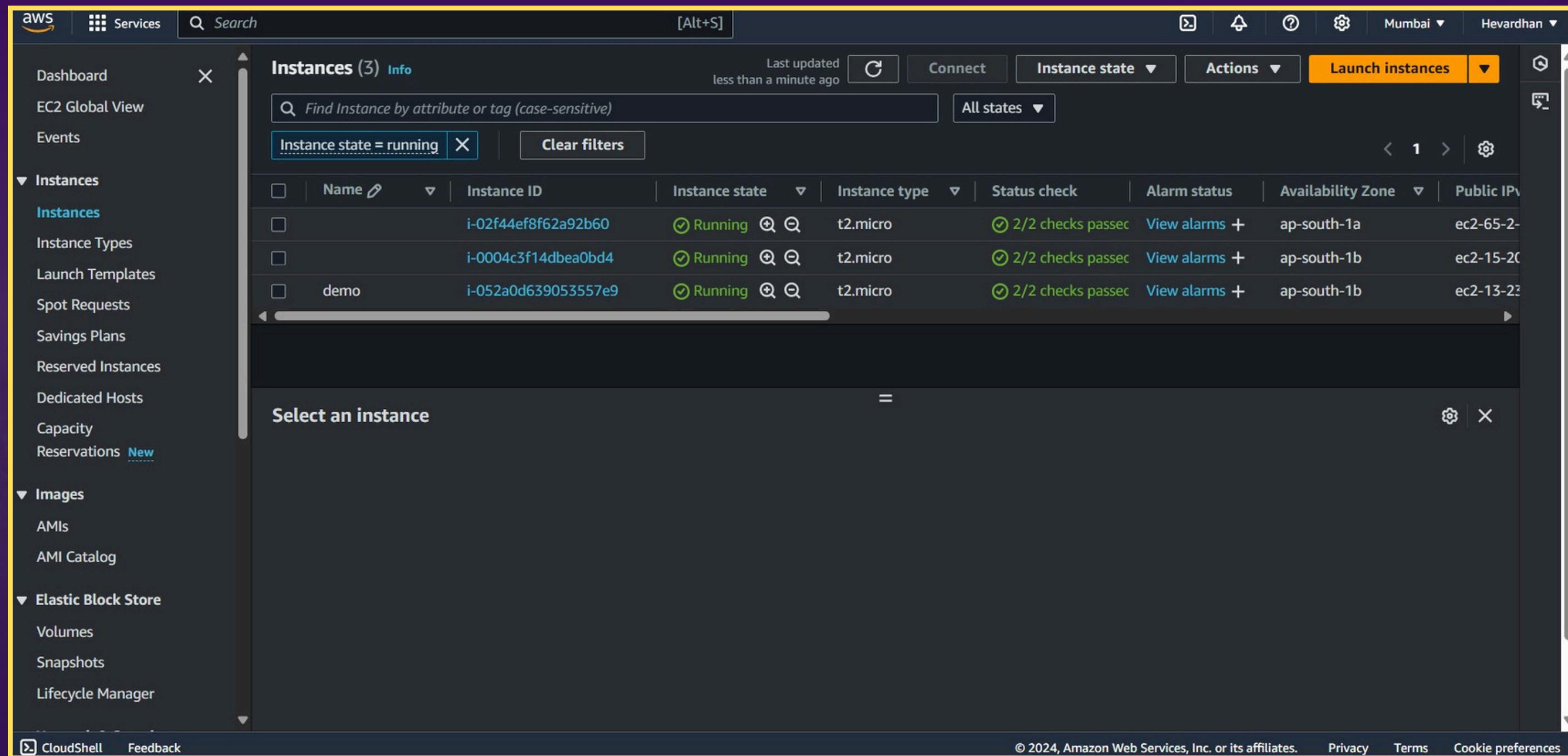


ELASTIC LOAD BALANCING

To distribute incoming traffic across multiple EC2 instances or in one or more availability zones, enhancing fault tolerance and load distribution.



IMPLEMENTATION



The screenshot shows the AWS EC2 Instances page with a yellow border around the main content area. The left sidebar contains navigation links for Dashboard, EC2 Global View, Events, Instances (with sub-links for Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity, and Reservations), Images (with sub-links for AMIs and AMI Catalog), and Elastic Block Store (with sub-links for Volumes, Snapshots, and Lifecycle Manager). The main content area displays a table of three EC2 instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
	i-02f44ef8f62a92b60	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1a	ec2-65-2-
	i-0004c3f14dbea0bd4	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1b	ec2-15-20-
demo	i-052a0d639053557e9	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1b	ec2-13-23-

A modal window titled "Select an instance" is open at the bottom, containing the text "Select an instance" and a close button.

EC2 Instances

IMPLEMENTATION

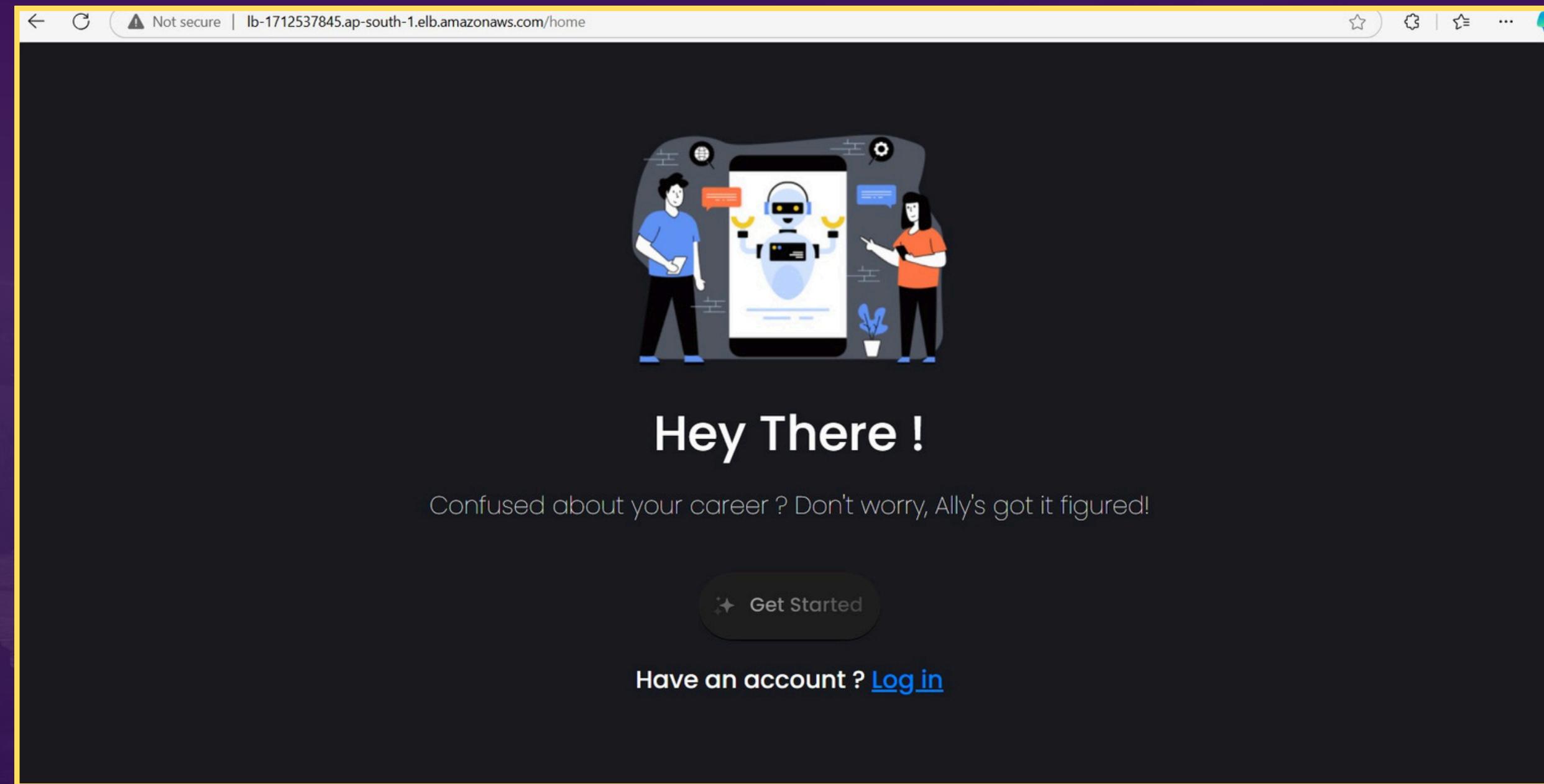
The screenshot shows the AWS CloudWatch Alarms interface. On the left, a sidebar lists various monitoring services: Favorites and recents, Dashboards, Alarms (with 0 alerts, 0 triggers, and 3 pending confirmations), In alarm, All alarms (selected), Logs, Metrics, X-Ray traces, Events, Application Signals (New), Network monitoring, and Insights (0 alerts). The main content area displays the 'Alarms (2)' section. It includes a search bar and filters for Alarm state, Alarm type, and Actions status. Two alarms are listed:

Name	State	Last state update (UTC)	Conditions	Actions
Instamnce	Insufficient data	2024-11-07 16:28:40	GroupTotalInstances <= 3 for 1 datapoints within 5 minutes	Actions enabled
hijk	Insufficient data	2024-11-07 16:28:32	GroupTotalInstances >= 4 for 1 datapoints within 5 minutes	Actions enabled Warr...

At the top of the main content area, there are three notifications: a green success message about creating an alarm, a blue info message about pending SNS subscriptions, and a green success message about deleting an alarm. The top navigation bar includes the AWS logo, a search bar, and account information for Mumbai and Hevardhan.

Alarms under CloudWatch

IMPLEMENTATION



Career Ally Landing Page



CONCLUSION



This project successfully deployed a scalable, reliable web application on AWS, using EC2 instances, Auto Scaling, and Elastic Load Balancing to handle variable traffic loads efficiently. The use of IAM provided robust access control, ensuring secure interactions with cloud resources. AWS's flexible infrastructure allowed us to optimize resource usage, balancing performance with cost-effectiveness.

Overall, this project demonstrates the advantages of AWS for building resilient applications and provides a strong foundation for future enhancements.

THANK YOU!