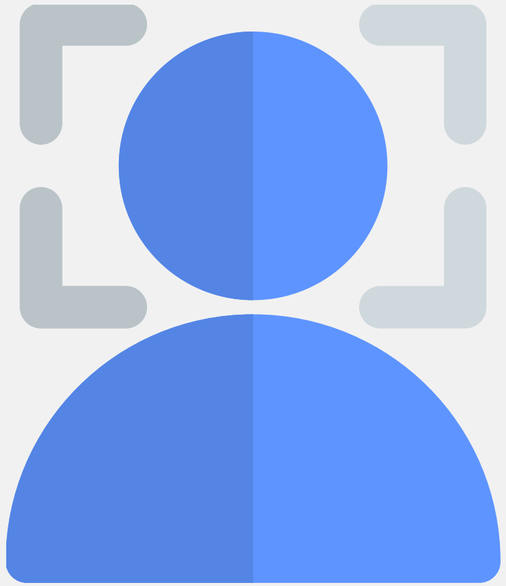


Face recognition on AWS

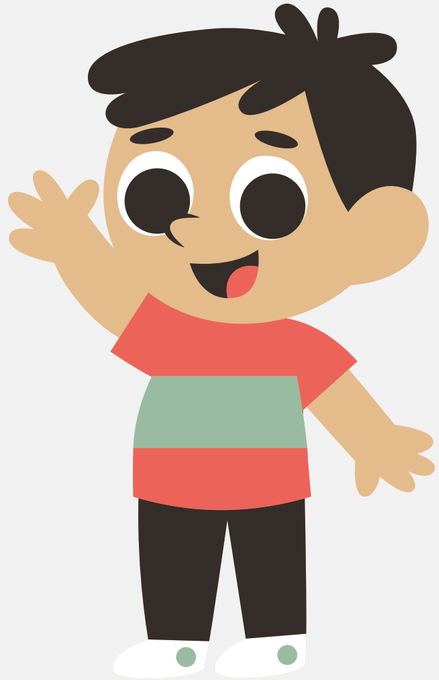
Cloud computing project (AIT)

Kaung Htet Cho
st124092
https://github.com/KaungHtetCho-22/Face-recognition-on-AWS

Recap

What is Face recognition on AWS

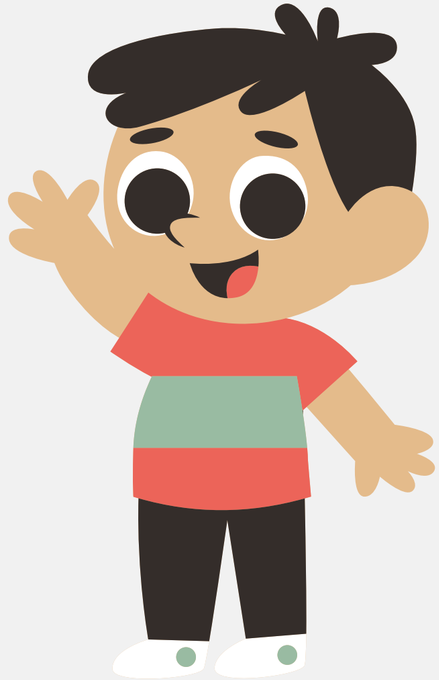
Face recognition on AWS is utilizing a service called ***Rekognition*** that uses powerful computer vision algorithms to quickly identify people's faces or other feactures in images or videos. This technology can be used for various purposes and application by personalizing user experiences



What is Face recognition on AWS

Problem statement

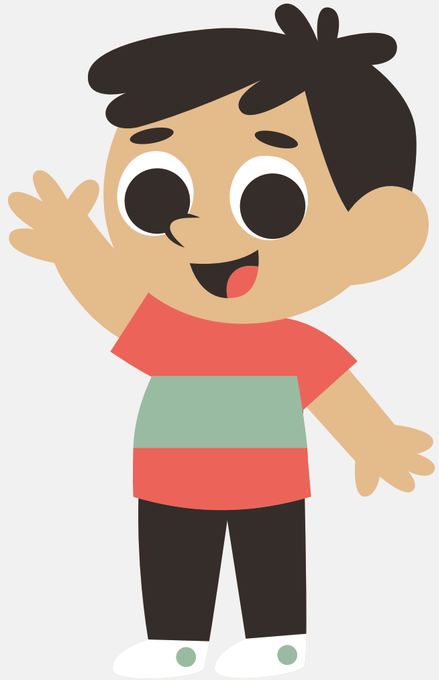
- Face recognition is important for many applications
  - Identifying and recognizing individuals based on facial features
- Manual face recognition methods have limitations
  - Time-consuming
  - Error-prone
  - Not scalable for large datasets
- Automated face recognition system is needed
  - Enhance security
  - Improve user experiences
  - Applicable across various domains



What is Face recognition on AWS

Importance of the system

- Versatile applications
  - Suitable for various domains (e.g., security, hospitality, retail)
  - Adaptable to specific business requirements
- Cost-effective and efficient
  - Utilizes pay-as-you-go cloud services
  - Eliminates need for extensive on-premises infrastructure
  - Scales based on demand, optimizing resource usage
- Automated and scalable solution
  - Leverages AWS cloud services (S3, Lambda, DynamoDB, EC2, etc)
  - Handles large volumes of data seamlessly



Literature reviews (similar systems)

- OpenCV with Python: Many projects leverage OpenCV, a popular computer vision library, combined with Python for face recognition tasks.
- Microsoft Azure Face API: Offers robust facial recognition capabilities, including detection, verification, and integration with Azure services.
- Google Cloud Vision API: Provides facial recognition services with face detection and attribute analysis, compatible with Google Cloud services.
- Custom Deep Learning Models: Utilizing convolutional neural networks (CNNs), exemplified by Facebook DeepFace and Google FaceNet, for accurate and efficient recognition.
- Hybrid Systems: Combine traditional computer vision with deep learning for enhanced accuracy and scalability, often leveraging cloud services

Objectives

Implement AWS Rekognition Service

Utilize AWS Rekognition services, specifically indexFaces and searchFacesByImage, for efficient face recognition tasks.

Automate Recognition Process

Develop a Lambda function triggered by image uploads to S3, which initiates Rekognition for face detection and stores the results in DynamoDB.

Database Integration

Integrate DynamoDB for storing Rekognition IDs and corresponding names, enabling efficient retrieval of recognized faces.

Enhance security and access control measures

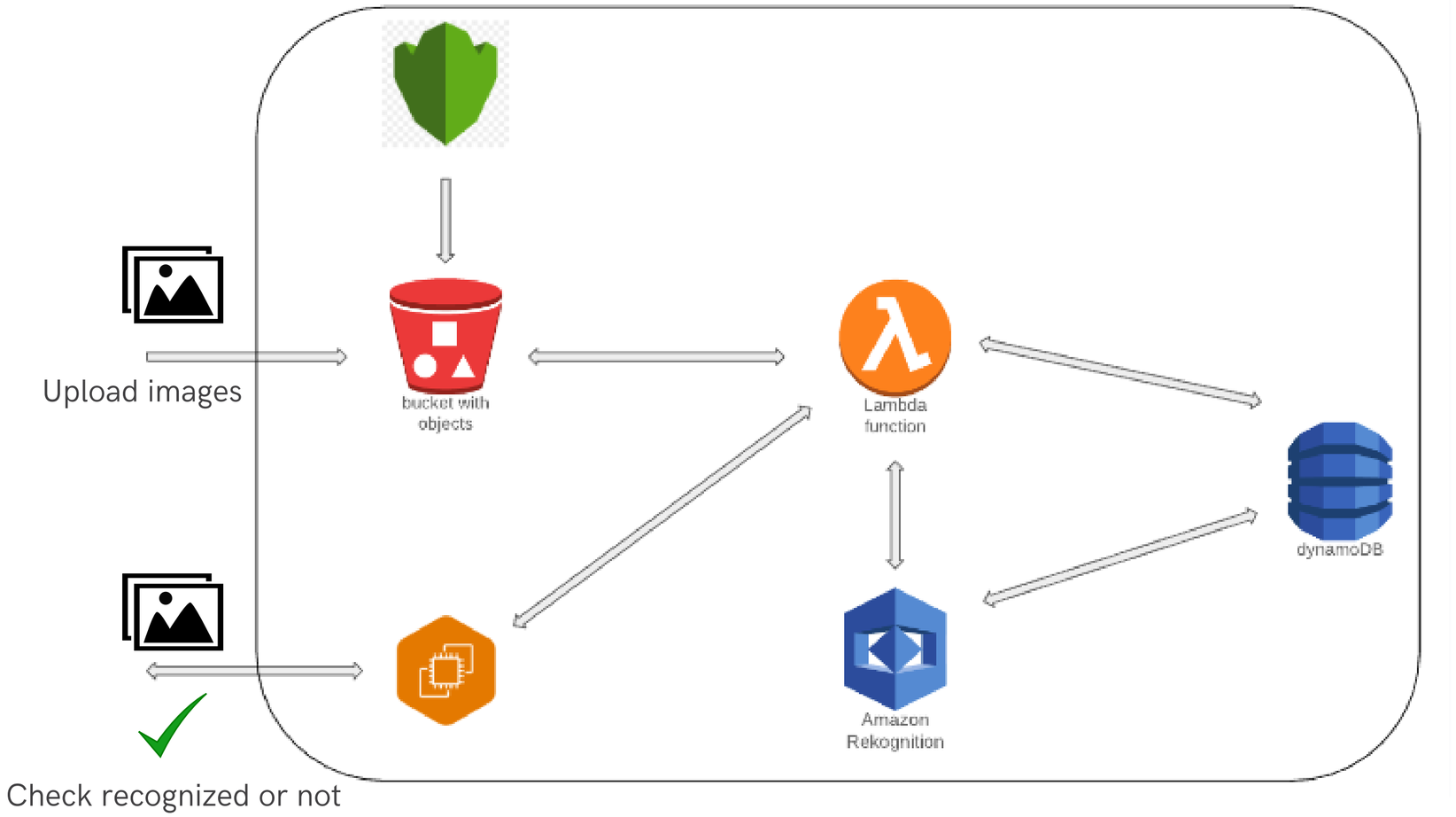
Accurately identify authorized individuals and Enable secure access to facilities, systems, or resources

Architecture

AWS services

✓	EC2	Used to host the web server for user interface deployment
✓	Lambda	Triggers face recognition processes upon image uploads to S3 and integrates with DynamoDB for storage
✓	S3	Stores uploaded images, triggering Lambda functions for face recognition
✓	DynamoDB	Stores Rekognition IDs and corresponding names for efficient retrieval during real-time recognition
✓	Rekognition	Performs face detection and recognition on images uploaded to S3, generating Rekognition IDs
✓	KMS	Ensures the security of sensitive data by encrypting and managing keys used for encryption across various AWS services, providing secure storage for data such as Rekognition IDs and user information in DynamoDB
✓	CloudFormation	Automates the deployment and management of AWS infrastructure resources by defining them in templates, enabling consistent and repeatable provisioning of resources

Overall architecture



Six pillars of well-architected design

Performance efficiency

Lambda function optimization	S3 storage optimization
Implement best practices for Lambda function optimization, such as reducing function execution time, minimizing dependencies, and maximizing concurrency settings to ensure efficient processing of image uploads and recognition tasks.	Implement lifecycle policies and storage classes in S3 to automatically move infrequently accessed images to cheaper storage tiers, reducing costs while maintaining accessibility.

Operational excellence

Automated Deployment with CloudFormation
Use CloudFormation templates to automate the deployment of your AWS infrastructure, ensuring consistency, repeatability, and reducing the risk of human error during deployments.

Security foundations

Data Encryption at Rest and in Transit	Identity and Access Management (IAM)
Implement encryption mechanisms, such as AWS Key Management Service (KMS) and SSL/TLS encryption, to encrypt data both at rest (stored in S3 and DynamoDB) and in transit (transferred between services), ensuring confidentiality and integrity	Follow the principle of least privilege by assigning IAM role and permissions to Lambda functions and EC2 instances, granting only the necessary permissions required for their respective tasks

Cost optimization

Lifecycle Policies for S3	Right-Sizing Resources
Implement lifecycle policies in S3 to automatically transition infrequently accessed objects to lower-cost storage classes, such as S3 Glacier or S3 Glacier Deep Archive, reducing storage costs while maintaining data accessibility	Analyze the resource usage patterns of your application and right-size EC2 instances, Lambda functions, and DynamoDB tables to match the workload demands, avoiding over-provisioning and under-utilization of resources.

Reliability

High Availability	Fault tolerance
The facial recognition system ensures high availability by utilizing AWS services like Amazon S3, which provides 11-9s durability and 99.99% availability for storing facial images. Ensure that the system is available and accessible to users at all times, allowing them to upload images and receive real-time recognition results without experiencing downtime or service interruptions	Deploying the system across multiple Availability Zones further minimizes the impact of any single point of failure.

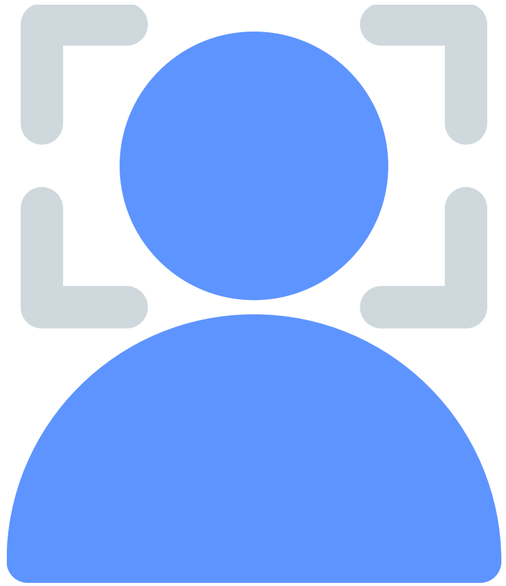
Sustainability

Serverless Architecture	On-Demand Provisioning
Utilizes serverless computing with AWS Lambda for image processing tasks. Serverless architectures eliminate the need for maintaining and provisioning servers, reducing resource wastage and energy consumption associated with idle servers	Analyze the resource usage patterns of the application and right-size of EC2 instances, Lambda functions, and DynamoDB tables to match the workload demands, avoiding over-provisioning and under-utilization of resources.

Conclusion

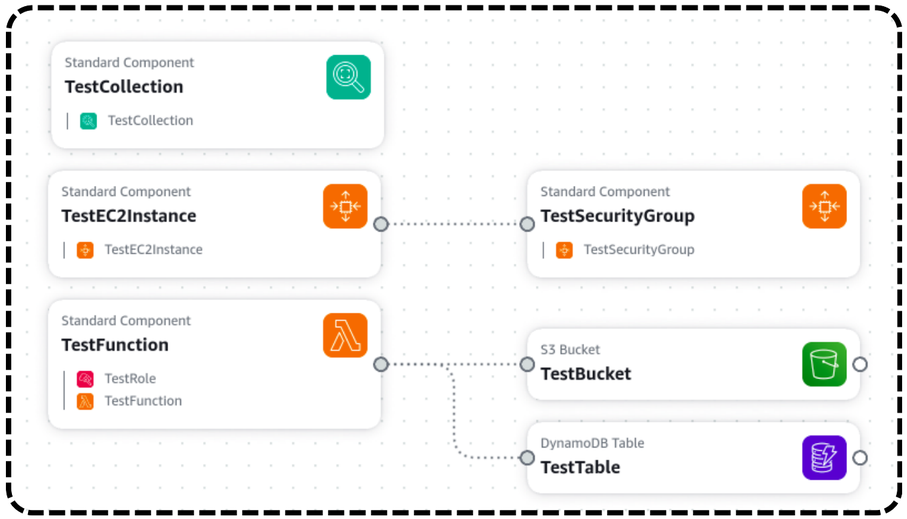
Conclusion

The real-time face recognition system, shows how cloud computing can solve difficult problems. Have made an automatic, scalable, and efficient solution that improves security, makes user experiences better, and can be used in many different areas. By constantly making the system better and adding more AWS services, this system can change how we do face recognition, making it more secure, faster, and personalized for each user.



Demo work

Infrastructure as Code (IaC)



cloudformation .yaml file

REKOGNITION

Create Collection for IndexFaces API and Searchfacesbyimage API

EC2

Create instance and security group rules

S3

Create S3 bucket define bucket policies and lifecycle rules

DYNAMODB

Save rekognition ID along with the names in a structured format

LAMBDA

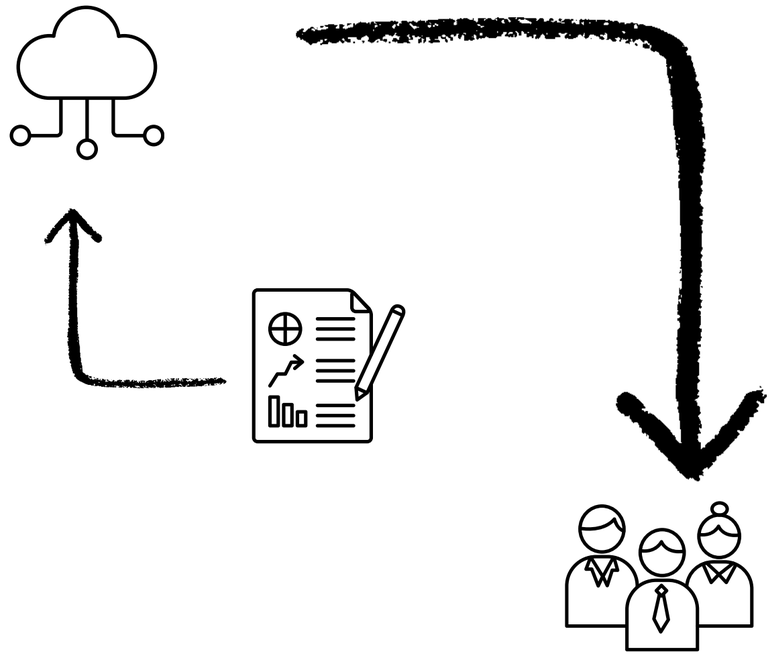
Trigger Rekognition and DynamoDB when images are uploaded

KMS

Encrypt images in S3 with SSE KMS

Web server hosted for user interfacing

- Allocate elastic ip address and associate with the ec2 instance
- Deployed app.py and test the results



Source

https://github.com/KaungHtetCho-22/Face-recognition-on-AWS

Thank you for your time

Do you have any questions?