AWS Cognito OTP Authentication System Documentation

Index

---

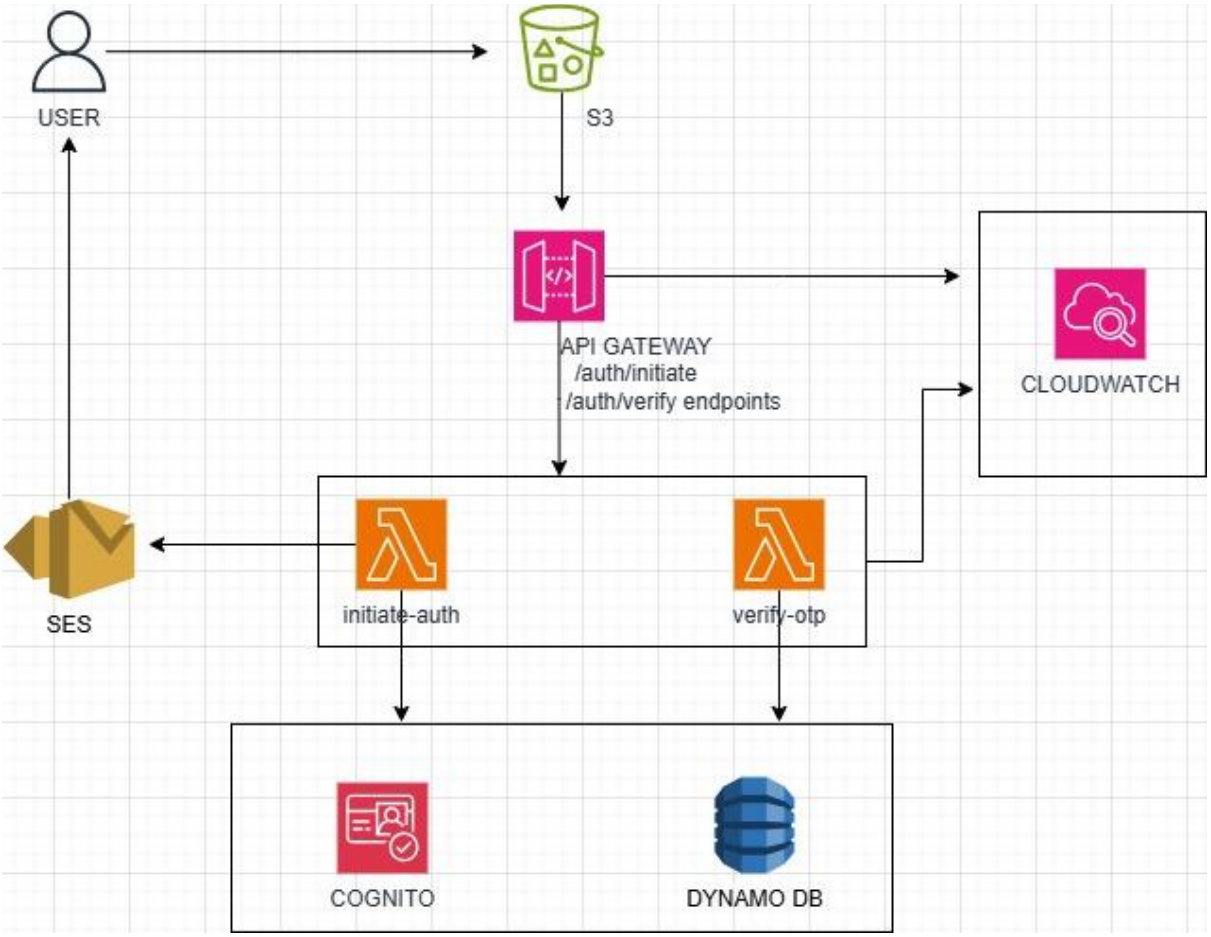
1. Overview

This documentation describes how to implement a **passwordless OTP authentication system** on AWS using **Cognito, API Gateway, Lambda, DynamoDB, SES, and S3**. Users authenticate by receiving a One-Time Password (OTP) via email, removing the need for passwords while maintaining security.

*This documentation aims to **empower cloud engineers, developers, and architects** by providing a guide to implementing a secure, scalable, and passwordless authentication system on AWS.*

---

2. Architecture Diagram



Workflow Explanation

1. User enters email on the frontend hosted on S3.
2. API Gateway /auth/initiate endpoint triggers **initiate-auth Lambda**.
    - If user does not exist, Cognito User Pool creates a new user through the lambda function.
    - Lambda generates OTP, stores it in DynamoDB with a TTL (Time to Live), and sends OTP via SES to the user.
3. User enters OTP on frontend, which calls API Gateway /auth/verify endpoint which triggers **verify- otp Lambda**.
    - **verify-otp Lambda** validates OTP against DynamoDB.
    - If valid, Cognito returns JWT tokens (IdToken, AccessToken, RefreshToken). Lambda receives the tokens from Cognito and returns the

tokens to the browser via API Gateway in the HTTP response body. Authentication is successful and user is redirected to the application.

    - OTP entry is deleted from DynamoDB.

4. Frontend stores tokens securely (short-lived in memory or httpOnly cookies).
5. Cloudwatch acts as an **operational visibility,** it tells how your authentication system is performing and helps debug issues.

3. Prerequisites

- AWS Account
- Verified email/domain in SES
- Node.js for Lambda development
- Basic knowledge of API Gateway, DynamoDB, and Cognito

---

4. Detailed Component Breakdown

Frontend (S3 Hosting)

- Files: index.html, home.html, auth.js, styles.css
- Hosted in an S3 bucket with static website hosting enabled.

API Gateway

- Routes: /auth/initiate, /auth/verify
- Connected to respective Lambda functions
- CORS enabled for frontend domain

Lambda Functions

- **initiate-auth**: Generate OTP, store in DynamoDB, send via SES, create user in Cognito if needed.
- **verify-otp**: Validate OTP, issue tokens via Cognito, remove OTP from DynamoDB.

DynamoDB

- Table: otp-storage
- Partition Key: email

- Attributes: otp, ttl
- TTL ensures automatic deletion after expiration.

SES (Email Delivery)

- Sends OTP emails to users
- Requires verified sender identity

Cognito User Pool

- Handles user management and JWT token issuance
- Integrated with Lambda to complete custom OTP-based flow

CloudWatch

- Logs Lambda execution and API Gateway requests for monitoring and debugging

---

5. Security Best Practices

- Set OTP TTL (5 minutes) and max retry attempts
- Store tokens securely (prefer httpOnly cookies)
- Rate limit API Gateway endpoints
- Monitor SES sending limits and bounce rates
- Use IAM least-privilege for all services

---

6. Troubleshooting

- **No OTP email**: Check SES sandbox mode, verify sender, and CloudWatch logs, SES emails may go to spam folder without domain authentication.
- **OTP expired**: Ensure DynamoDB TTL is set correctly.
- **Invalid token**: Verify Cognito client and pool IDs match in Lambda.
- **CORS issues**: Confirm API Gateway CORS setup matches frontend domain.

7. Proof of Concept

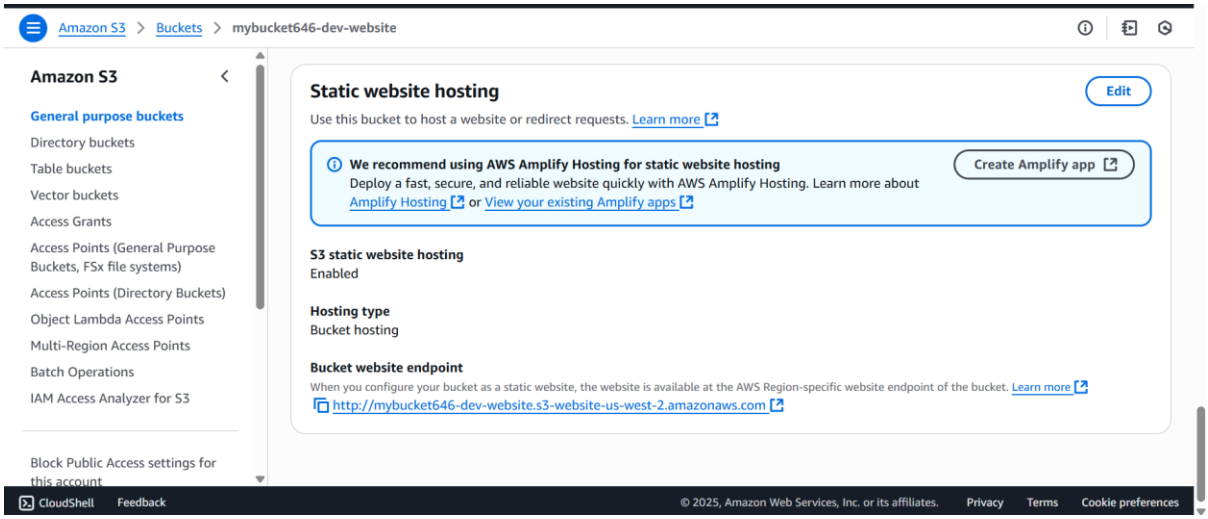
Objective

The purpose of this Proof of Concept (PoC) is to demonstrate a **passwordless login system** built with AWS Cognito, where users authenticate via **One-Time Passwords (OTP) delivered by email**, instead of traditional passwords.

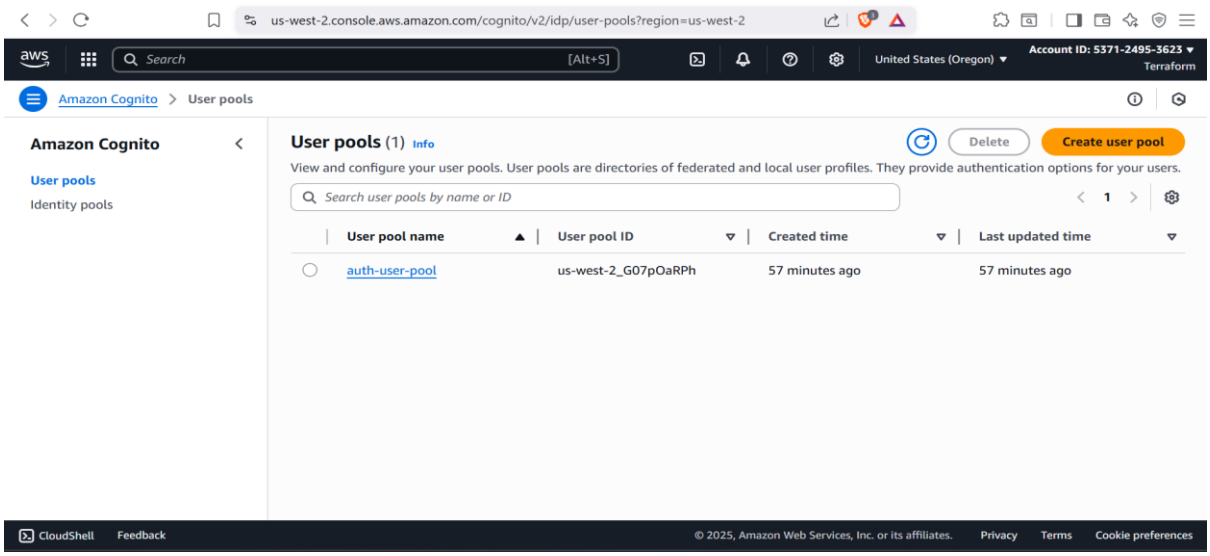
Provisioned Resources

Screenshots of the AWS resources used:

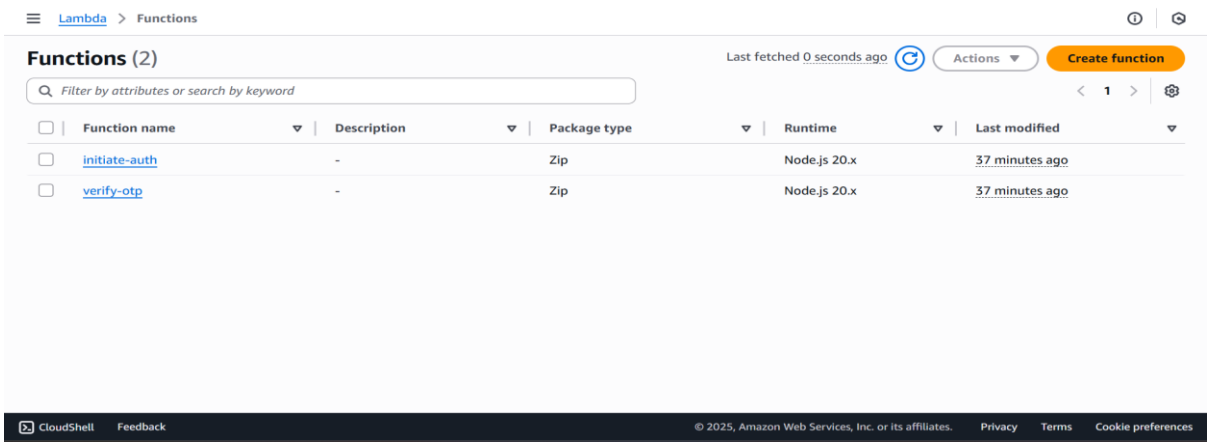
S3 bucket with website hosting enabled.



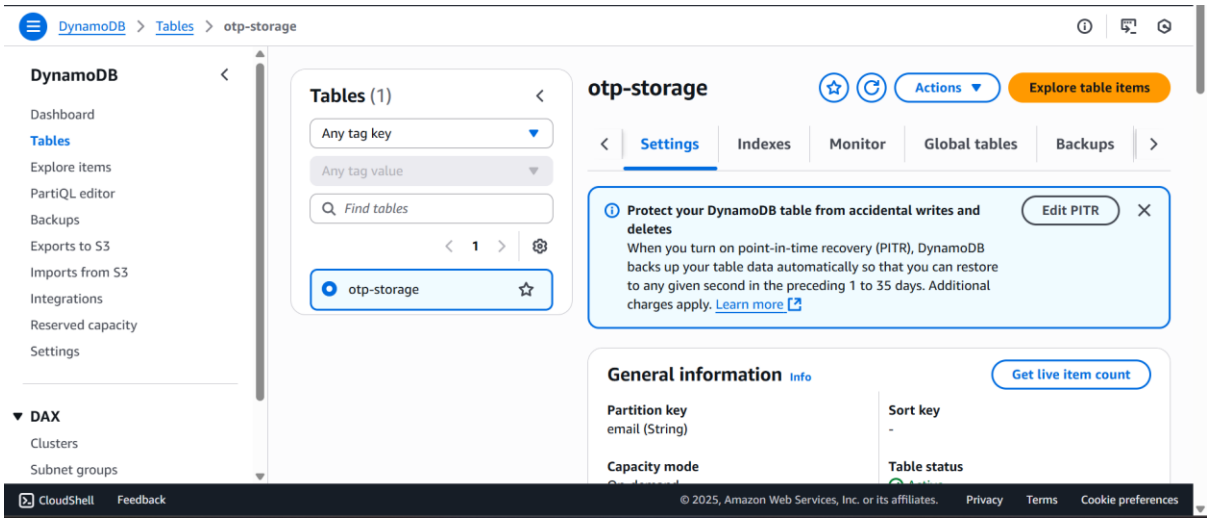
Cognito User Pool (for user identity & tokens).



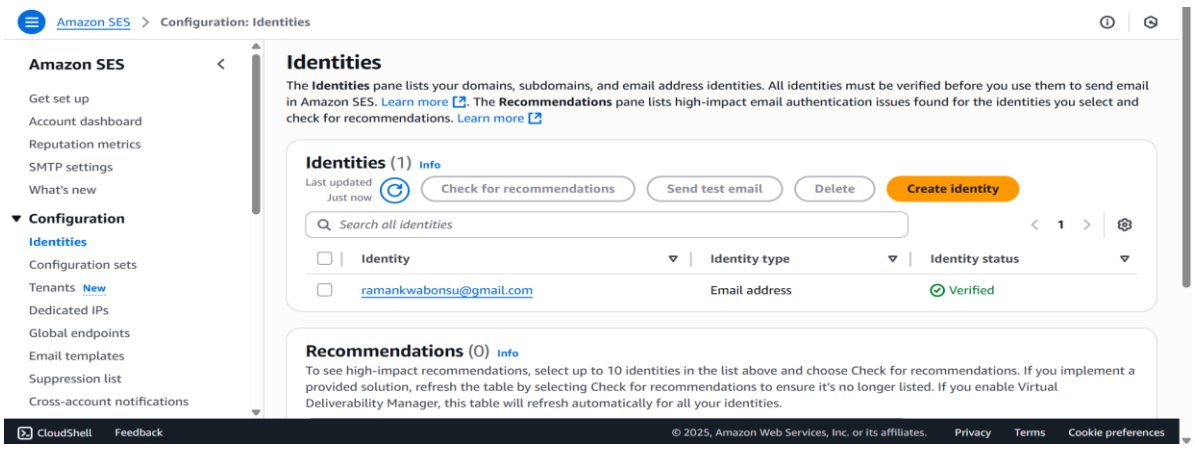
Lambda functions (initiate-auth, verify-otp).



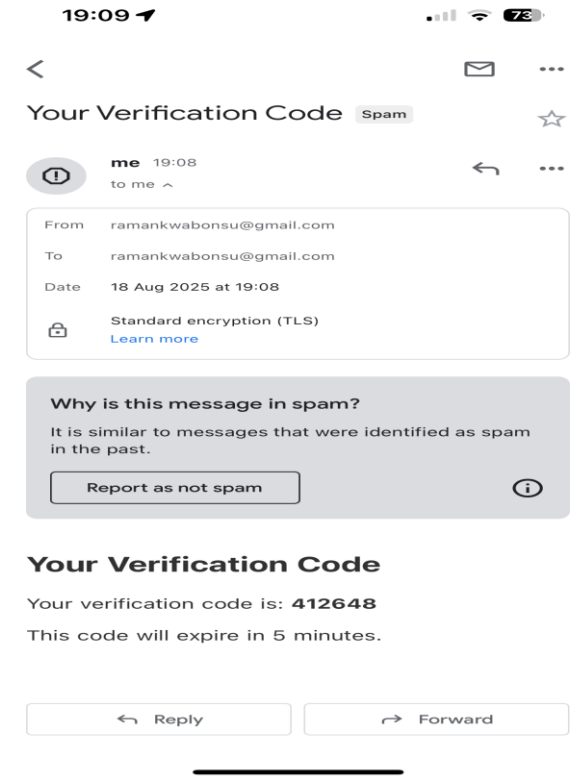
DynamoDB table



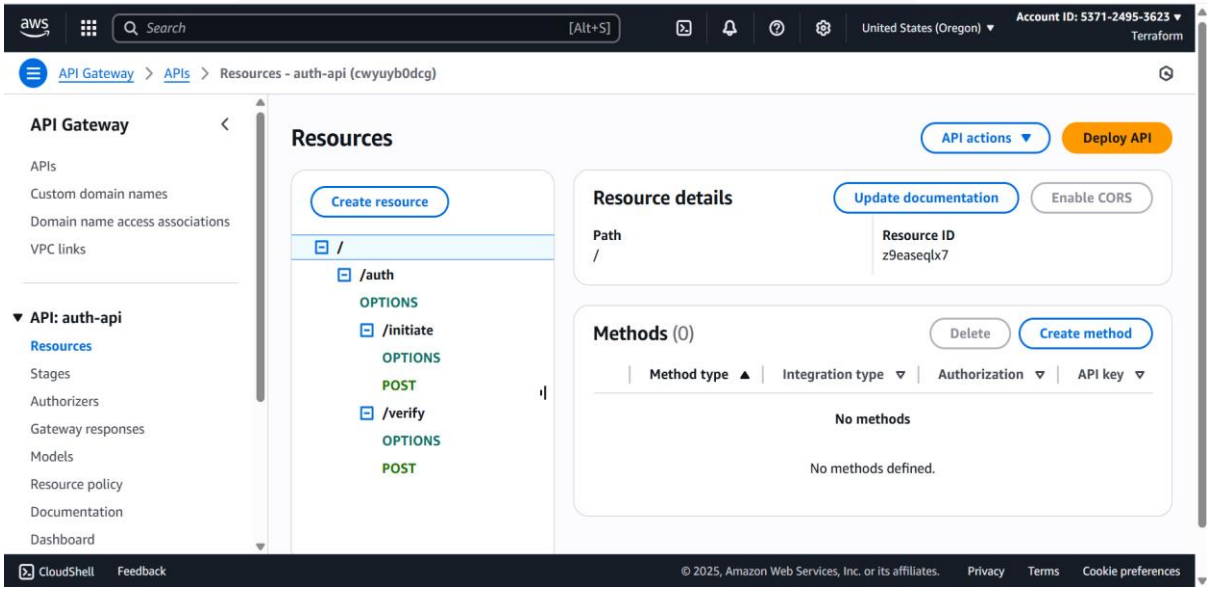
SES verified sender email.



The email was sent to the spam folder since I was using a personal email in sandbox mode which has low and limited reputation, but in production environments with an authenticated domain, this issue will be solved.



API Gateway routes connected to Lambda.



Configuration & Code.

The full implementation is available in this GitHub repo.

Demo / Validation

1. User enters email → receives OTP via SES
2. User enters OTP → validated against DynamoDB
3. Cognito issues JWT tokens → login successful

Demo video:

Watch the demo video here.