CLOUD-BASED SMART FILE STORAGE & SHARING SYSTEM

A MINI GOOGLE DRIVE–LIKE CLOUD APPLICATION

DEVELOPED BY:
AISHWARYA KOPULWAR

INTERNSHIP: CLOUD COMPUTING INTERNSHIP – ELEVATE LABS (REMOTE, BANGALORE)
YEAR: 2025

LIVE LINK: https://d3dltsp1h13o1v.cloudfront.net/

PROBLEM STATEMENT

- Need for secure online file storage
- Accessing files from any device
- Sharing files easily
- Keeping data organized & safe
- Inspired by Google Drive

PROJECT OBJECTIVE

This project aims to build a system that supports:

- User Signup & Login
- JWT-based authentication
- Uploading files
- Cloud file storage
- Download / share via link
- Secure deletion
- User-specific file access

SYSTEM OVERVIEW

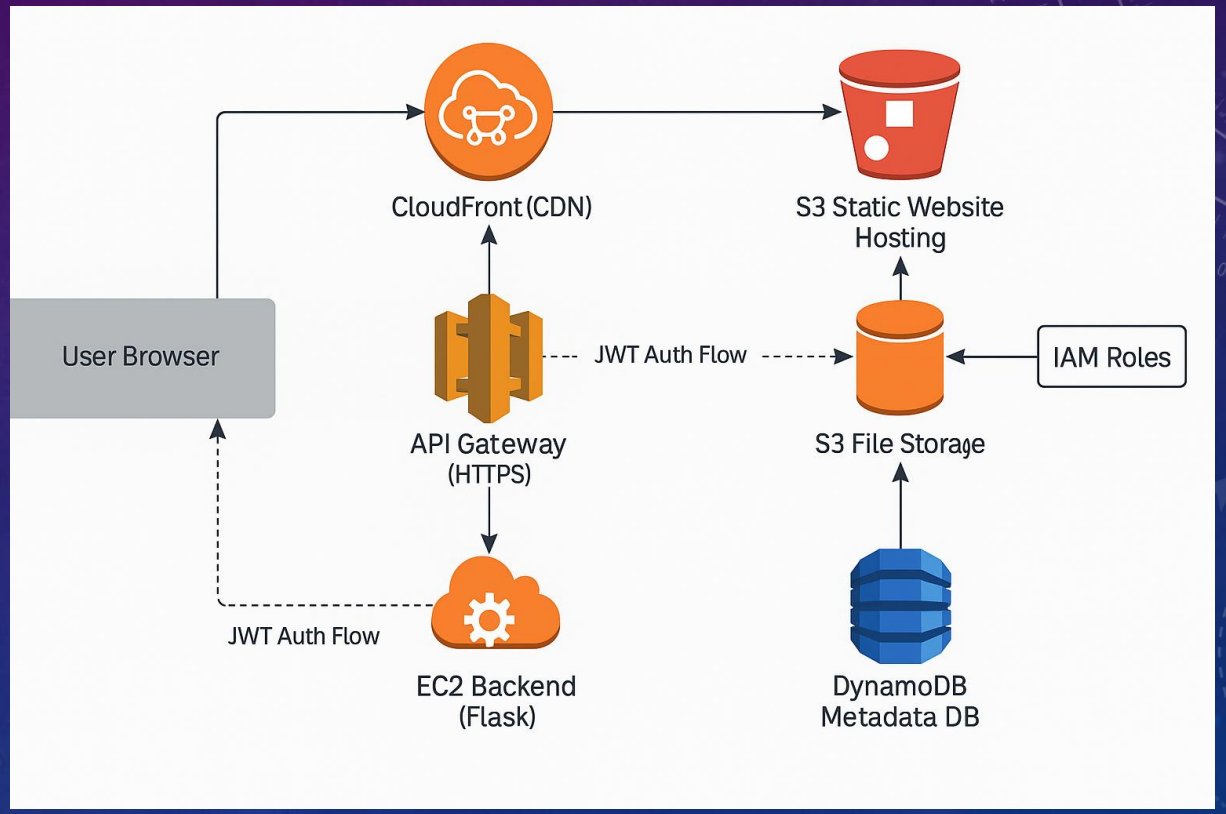
Technologies Used Across the System

- **Frontend:** HTML, CSS, JavaScript
- **Backend:** Python Flask on EC2
- **Storage:** Amazon S3
- **Database:** DynamoDB
- **API Layer:** API Gateway
- **Delivery:** CloudFront
- **Security:** JWT Auth, HTTPS, IAM Roles

ARCHITECTURE DIAGRAM

How It Works:

- User interacts with the UI
- CloudFront forwards requests
- API Gateway → EC2 backend
- EC2 → S3 for file storage
- EC2 → DynamoDB for metadata
- IAM role provides secure access
- Response returned to user



TOOLS & TECHNOLOGIES

AWS Services

- Amazon S3
- DynamoDB
- EC2
- API Gateway
- CloudFront
- **IAM (for S3 & DynamoDB permissions)**

Frameworks & Languages

- Python Flask
- HTML, CSS, JavaScript
- JWT Authentication

Deployment

- EC2 hosting
- CloudFront CDN
- S3 static hosting

FEATURES IMPLEMENTED

- **Core Features**

  JWT Authentication
  File Upload
  File Download (Pre-signed URLs)
  Copy Link
  Share Link
  Delete File
  User-Specific Access

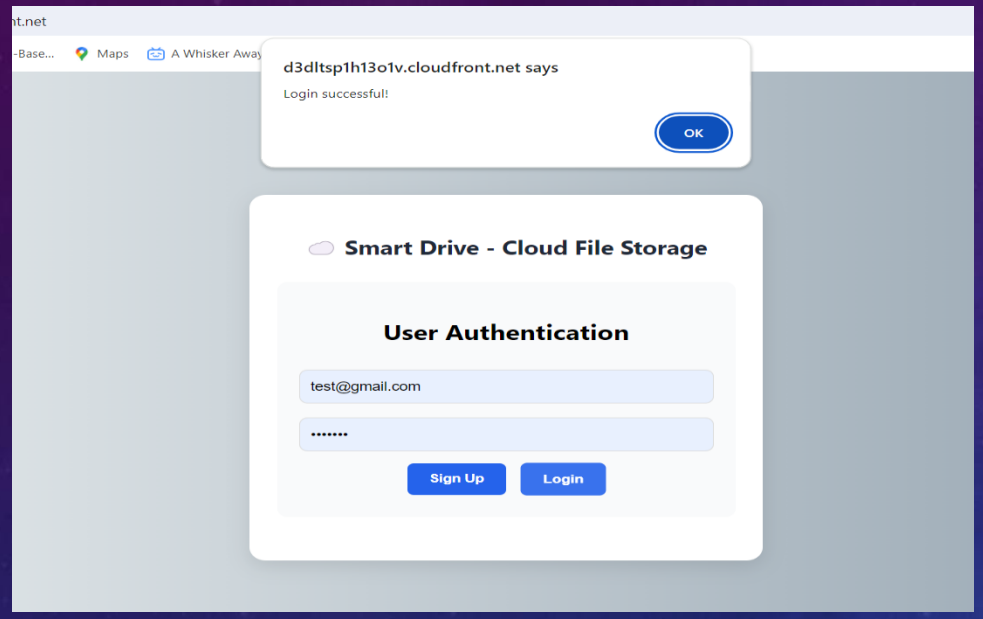
- **Cloud Features**

  HTTPS-enabled APIs
  IAM-secured backend
  CloudFront global caching

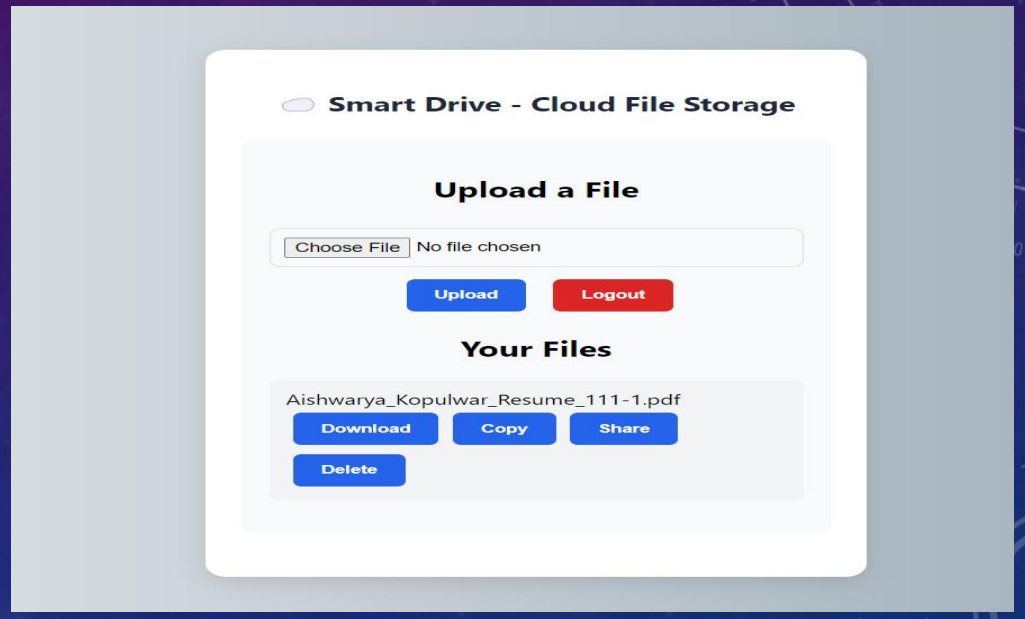
IMPLEMENTATION WORKFLOW

- **Step-by-Step Flow**
- 1 User signs up → Token generated
  2 Login → JWT verified
  3 File upload → Stored in S3
  4 Metadata saved in DynamoDB
  5 Download/Share → Pre-signed link
  6 Delete → S3 + DynamoDB updated
  7 CloudFront → Fast global delivery
  8 IAM → Secure EC2 permissions

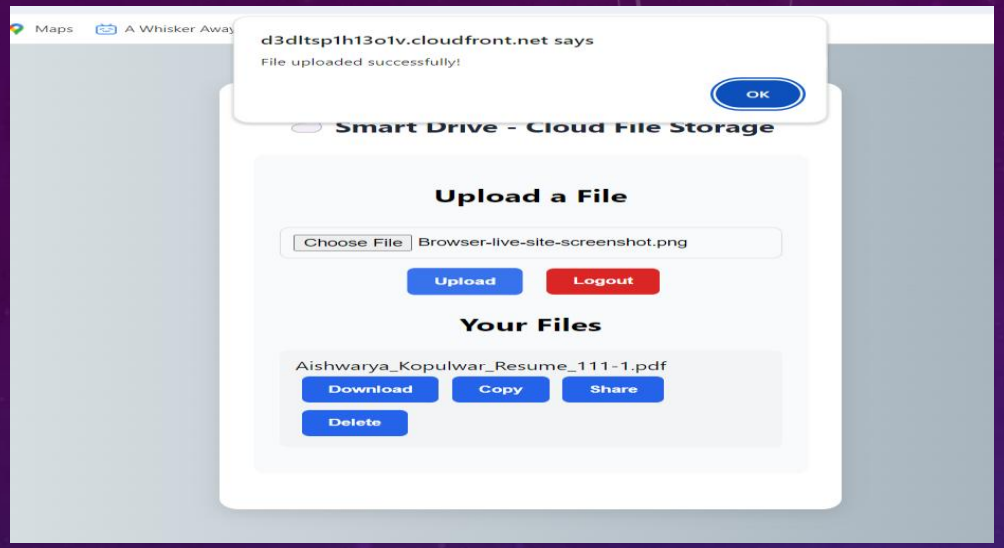
SCREENSHOTS / DEMO



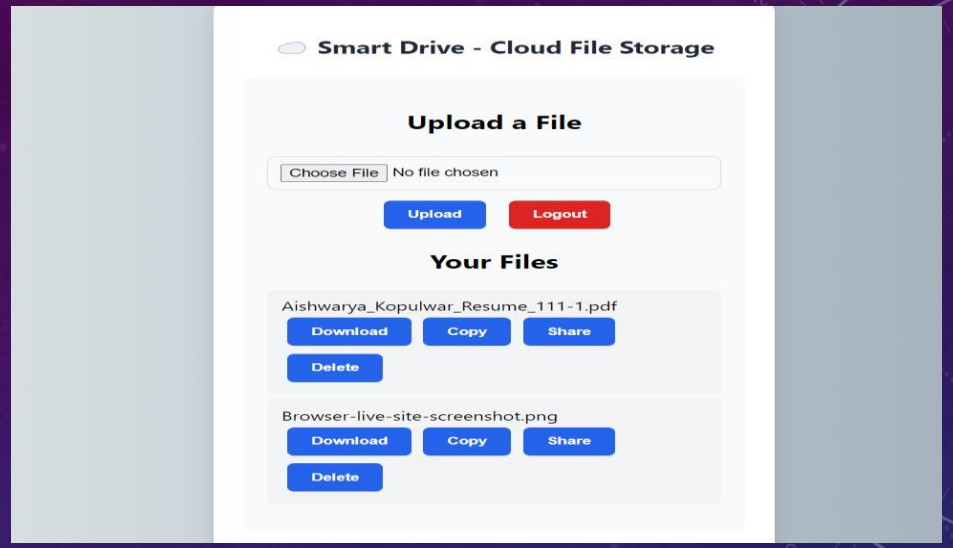
Login Page



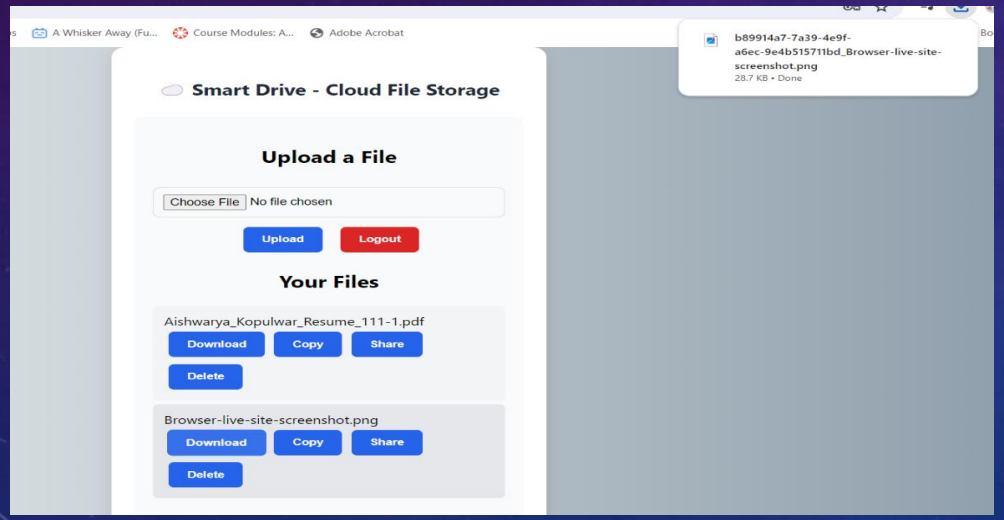
File Upload page



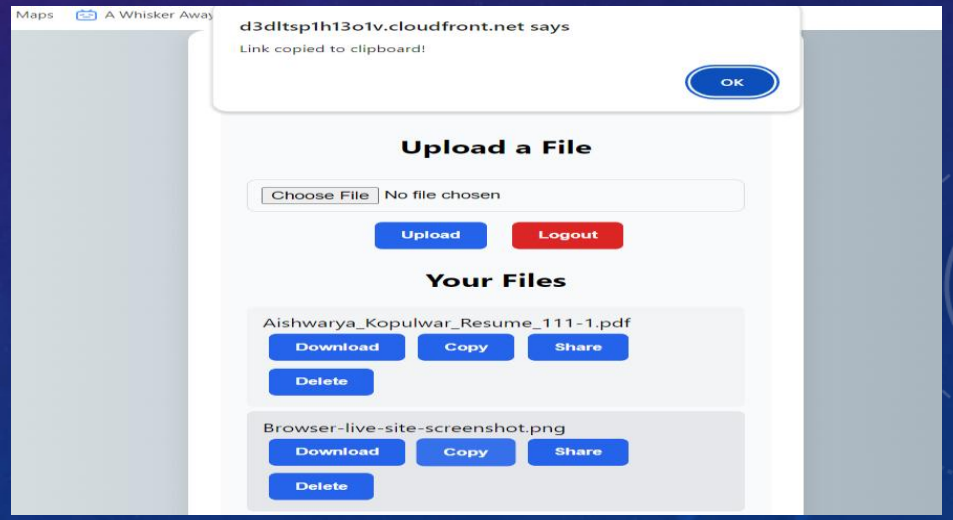
File is Uploaded



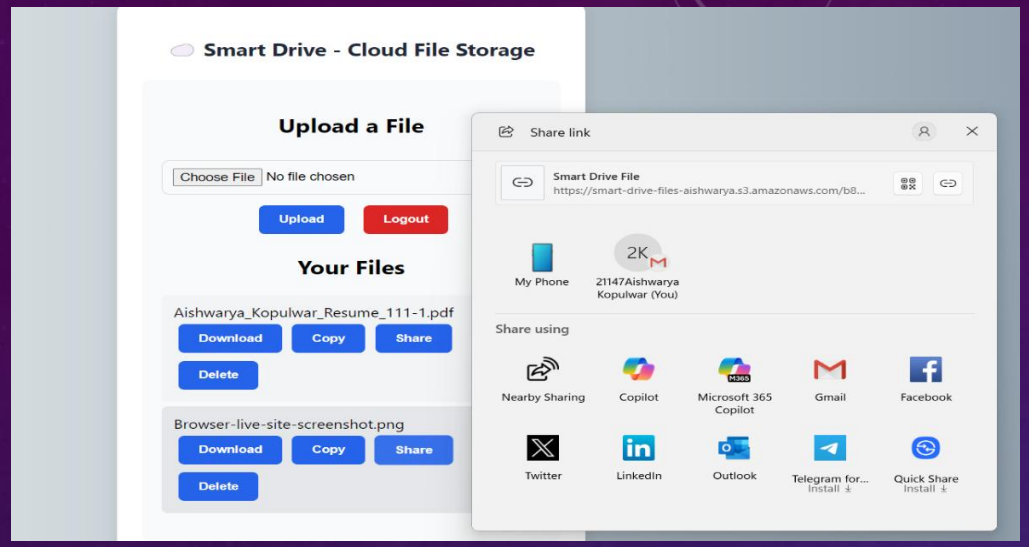
File List page



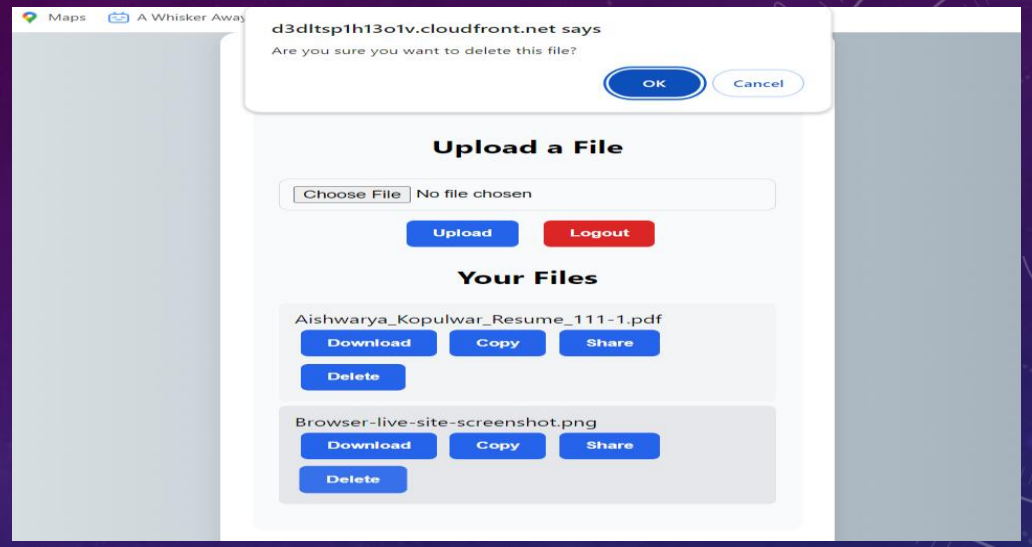
File is Downloaded



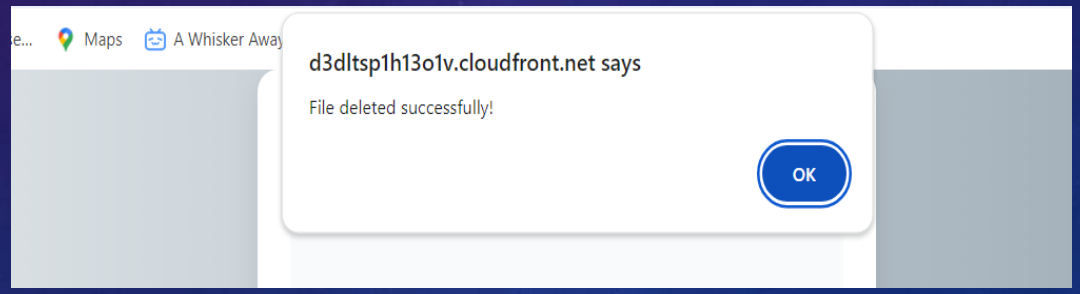
Copy Link



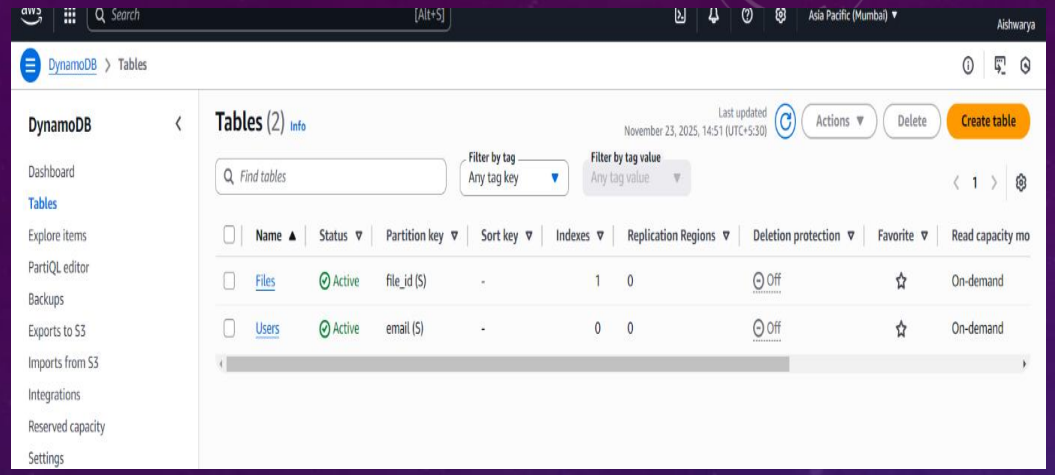
Share Option



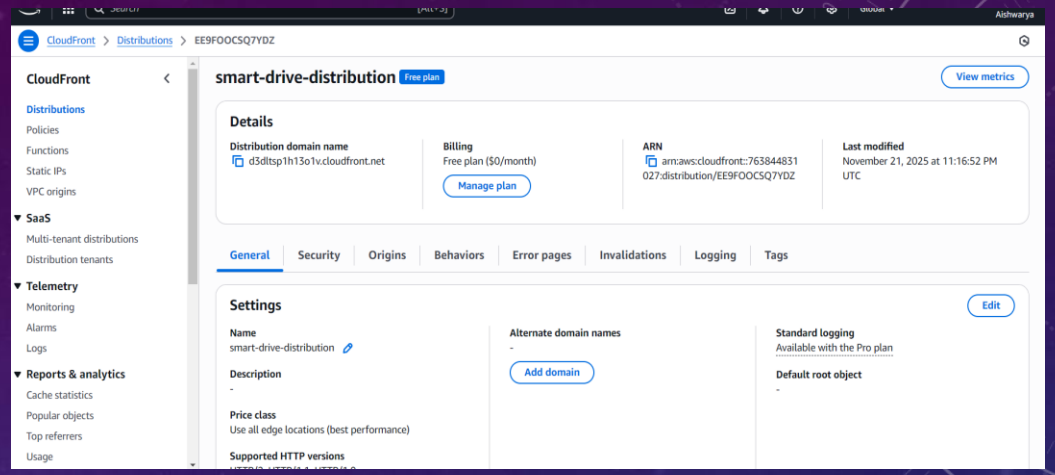
Delete Confirmation



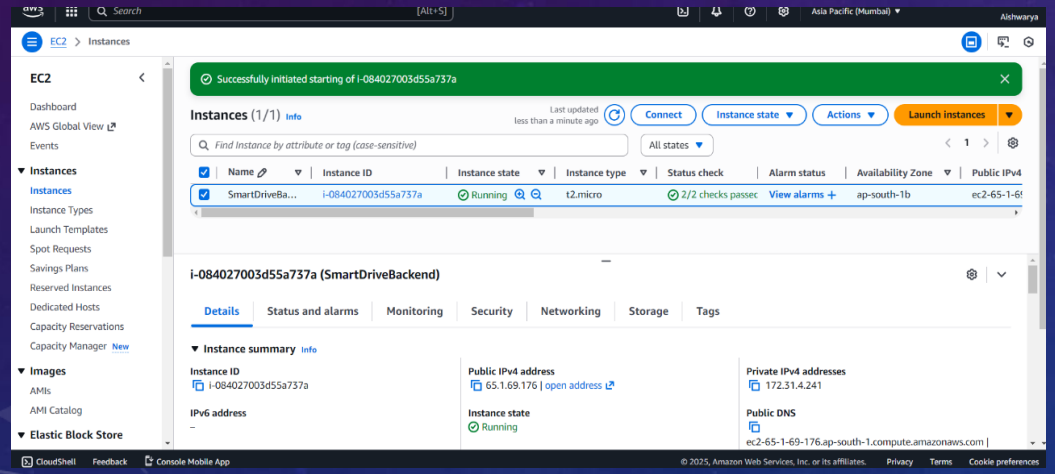
File Deleted



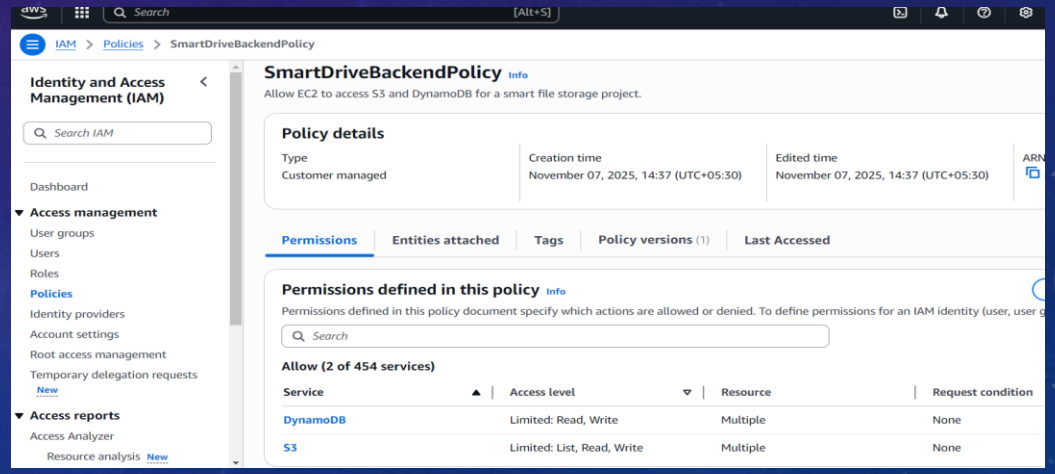
DynamoDB Tables



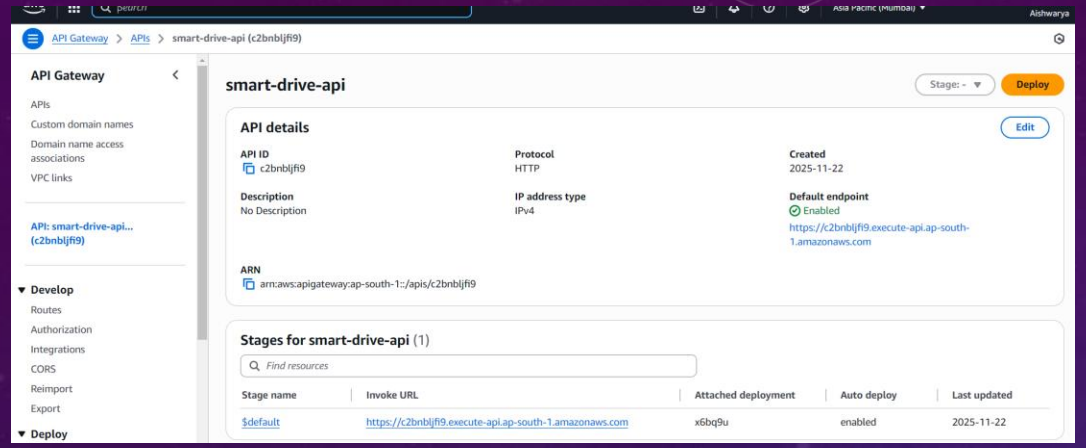
CloudFront Page



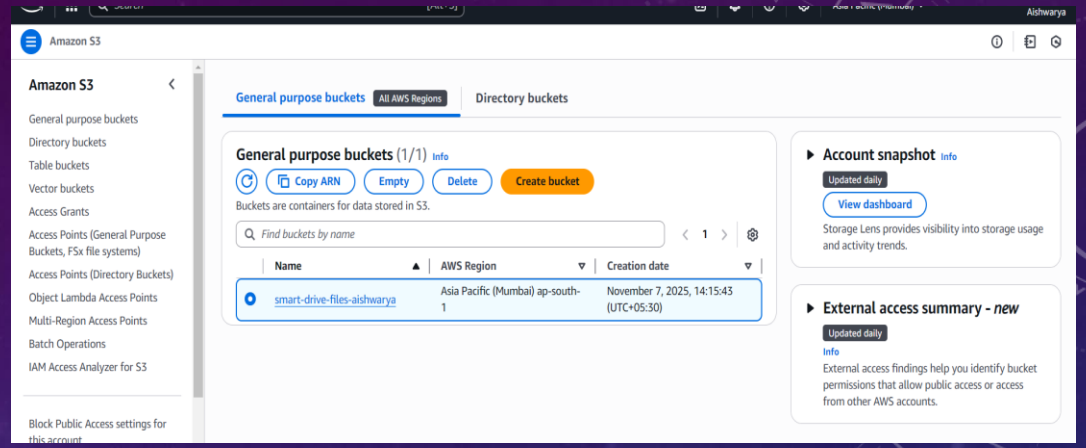
EC2 Terminal



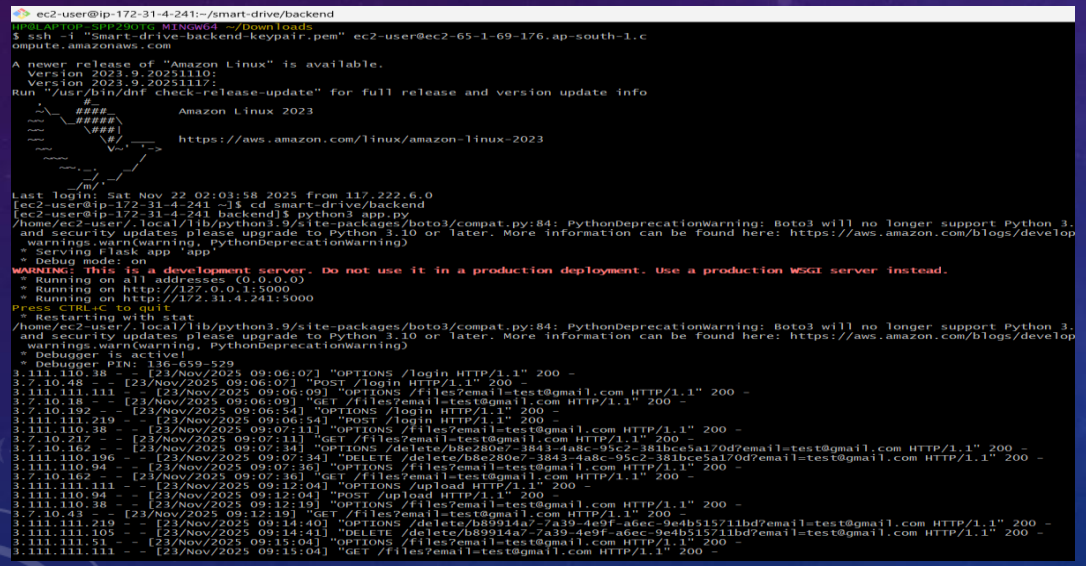
IAM Policies



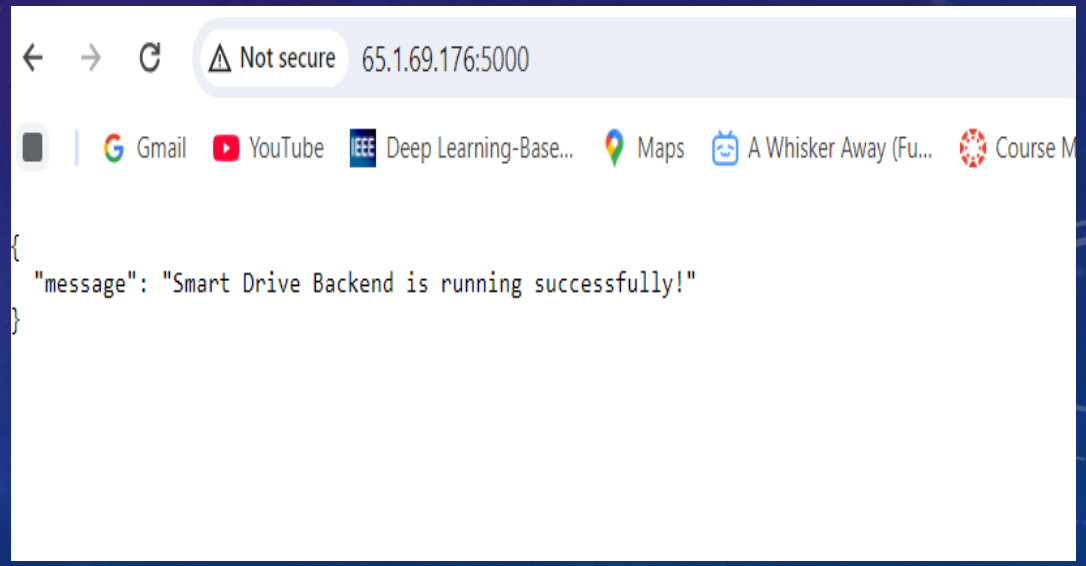
API Gateway



S3 Bucket



Flask API Terminal



Backend is Running

RESULTS

- **Achievements**
- Fully functional cloud file storage system
- Fast upload/download
- Secure JWT login
- EC2 backend + S3 storage + DynamoDB database
- CloudFront global performance
- HTTPS enabled via API Gateway
- **Live URL:**

Frontend URLs:

CloudFront (Primary – HTTPS):

https://d3dltsp1h13o1v.cloudfront.net/

S3 Static Website (Optional – HTTP Only):
http://smart-drive-files-aishwarya.s3-website.ap-south-1.amazonaws.com

Backend API (API Gateway – HTTPS):
https://c2bnbljfi9.execute-api.ap-south-1.amazonaws.com/

CHALLENGES FACED

- CORS issues during testing
- IAM permission errors
- API Gateway integration setup
- EC2 backend restart handling
- Pre-signed URL expiration
- Static hosting behavior in S3

FUTURE ENHANCEMENTS

- Multi-file upload
- File preview (PDF, images, videos)
- User roles
- Email-based sharing
- File version control
- Serverless backend using AWS Lambda
- Analytics dashboard

CONCLUSION

This project helped me learn:

- AWS Cloud Architecture
- Python Flask API development
- IAM-based secure backend
- Cloud storage & database integration
- Deployment & scaling
- End-to-end full-stack cloud application

A complete working real-world cloud project.

THANK YOU!