

CLOUDAUDITPRO – FULL SYSTEM ARCHITECTURE (DETAILED EXPLANATION)

This document explains how CloudAuditPro works end-to-end, including:

- Backend (FastAPI + Auth + AWS scanning)
- Frontend (React + Vite + Auth + Password reset flow)
- EC2 deployment
- S3 + CloudFront hosting
- JWT authentication
- SQLite storage
- SES email logic
- Onboarding flow
- SPA routing
- And how all pieces talk to each other

1. HIGH-LEVEL ARCHITECTURE

User → Browser → React Frontend (CloudFront CDN)
→ API Requests → FastAPI Backend (EC2)
→ AWS STS AssumeRole into Customer Account
→ SecurityHub / S3 Scans
→ Send results → Frontend
→ Optional: SES → Customer email

2. DETAILED COMPONENTS

A. FRONTEND (React + Tailwind + Vite, hosted on S3 + CloudFront)

- URL: <https://app.cloudauditpro.app>
- Hosted in S3 bucket
- Served globally by CloudFront
- Uses Vite environment variable:
VITE_API_BASE_URL = <https://api.cloudauditpro.app>
- SPA routing:
/login
/signup
/forgot-password
/reset-password
/
- CloudFront error pages forward 403/404 → /index.html so SPA works.

Frontend responsibilities:

- ✓ User login & signup
- ✓ Token storage (localStorage)
- ✓ Calling backend (/scan, /aws/s3-summary, /report/email)
- ✓ Showing outputs (SecurityHub findings, S3 summaries)
- ✓ Forgot/reset password flow

B. BACKEND (FastAPI running on EC2 behind systemd)

URL: <https://api.cloudauditpro.app>

Backend files:

- main.py → core routes + scanning + email
- auth_utils.py → hashing, JWT, password reset tokens, SES email logic
- routers/auth.py → login, signup, reset password endpoints
- routers/onboarding.py → store AWS connection info
- aws.py → STS assume role, get SecurityHub findings, S3 summary
- report.py → format reports
- db.py / models.py / schemas.py → SQLite ORM

Backend responsibilities:

- ✓ User authentication
- ✓ Session tokens (JWT)
- ✓ Database of users + reset tokens
- ✓ Scanning AWS accounts
- ✓ Sending email reports
- ✓ Password reset
- ✓ CORS configuration

C. SQLITE DATABASE

Location on EC2:

~/CloudAuditPro/backend/cloudauditpro.db

Tables:

- users
- aws_connections
- organizations
- password_reset_tokens

Used for:

- ✓ Login
- ✓ Storing user metadata
- ✓ Temporary storage of reset tokens

D. AUTH SYSTEM (JWT-based)

Signup:

POST /auth/register
→ hash password
→ store user
→ return user

Login:

POST /auth/login
→ verify email/password
→ return JWT access token

→ frontend stores token in localStorage

Protected routes:

Depends(get_current_user)

→ decode JWT

→ load user from database

Password Reset:

1. POST /auth/request-password-reset

→ create token

→ email link with ?token=

2. POST /auth/reset-password

→ verify token

→ update password

E. AWS ROLE ASSUMPTION + SCANNING

Backend uses:

STS → assume role into target AWS account

Requires:

- AWS Account ID

- IAM Role Name (CloudAuditProReadRole)

- External ID (cloudauditpro)

Client creation:

securityhub_client_from_creds

s3_client_from_creds

Scanning:

list_findings()

get_s3_security_summary()

F. SES Email Flow

Used for:

- Weekly report (/report/email)

- Password reset emails

Requirements:

- Verified SES_FROM_ADDRESS

- Backend must run in SES-approved region

G. EC2 DEPLOYMENT FLOW

- Python 3.11 venv
- Uvicorn running via systemd service
- Environment variables stored in ~/.profile or systemd Environment=
- Nginx optional (API-only mode works with CORS)

Systemd → Uvicorn → FastAPI → AWS STS → scanners → frontend

S3 bucket:
cloudatauditpro-frontend

- Origin: S3
- OAC for permissions
- ACM certificate: app.cloudauditpro.app
- Routes all requests → index.html for SPA

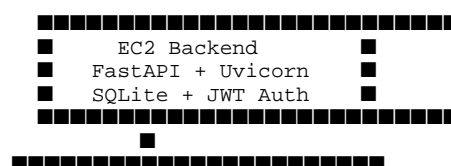
```
aws s3 sync dist/ s3://cloudatauditpro-frontend
aws cloudfront create-invalidation --paths "/*"
```

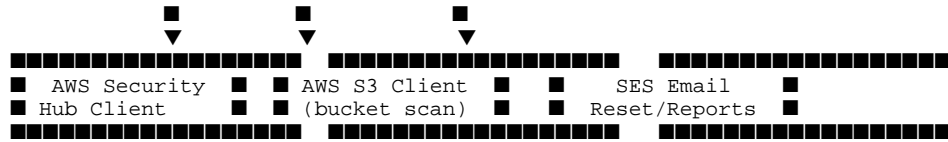
```

    █ User Browser █
    █ (React Frontend) █
    █ HTTPS (443)
    ▼
    █ CloudFront CDN (Global) █
    █ - Serves React bundle █
    █ - SPA routing for /reset-password █
    ▼
    █ S3 Frontend Bucket █
    █ app.cloudauditpro.app █

```

Browser → API call → <https://api.cloudauditpro.app>





- ▲
- STS AssumeRole into customer AWS account
- using:
- - Account ID
- - Role name
- - External ID

```

Customer AWS Account (Target Account)
IAM Role: CloudAuditProReadRole
Permissions:
- SecurityHubReadOnlyAccess
- AmazonS3ReadOnlyAccess

```

END OF DOCUMENT