

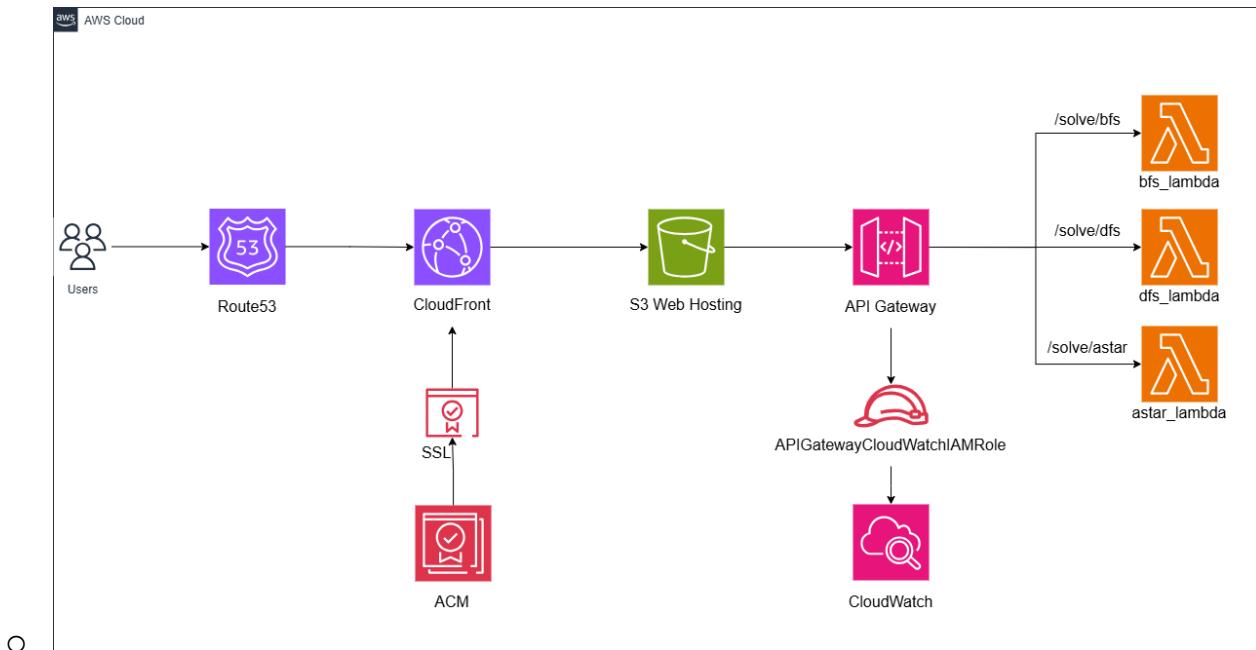
AWS Project – Serverless Graph Search Visualizer

Project Overview

- This project is a cloud-native interactive web application designed to visualize Graph Search Algorithms (BFS, DFS, A*) in a grid-based maze environment. The system is built entirely using Cloud Computing technologies on the AWS platform, leveraging a **Serverless Architecture** to ensure high scalability, performance, and cost-efficiency (Pay-as-you-go model).

Solution Architecture

- The system design follows a decoupled architecture, separating the Frontend from the Backend. The application interface is hosted statically and distributed globally via a CDN, while the algorithmic logic is executed via cloud functions triggered through a secure API.



Frontend & Hosting Components

- **Amazon S3 (Simple Storage Service)**

- Used for hosting the static website files (`index.html`, `style.css`, `script.js`).
- **Configuration:** "Block Public Access" is enabled to ensure security. Access is restricted solely to CloudFront using **Origin Access Control (OAC)** policies.
- **Purpose:** Secure, durable, and highly available storage for frontend assets.

maze-solver-website-s3 [Info](#)

Objects | Metadata | Properties | Permissions | Metrics | Management | Access Points

Actions ▾ [Create folder](#) [Upload](#)

Objects (2) [Copy S3 URI](#) [Copy URL](#) [Download](#) [Open](#) [Delete](#) [Actions](#) [Create folder](#) [Upload](#)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	index.html	html	December 11, 2025, 21:26:27 (UTC+02:00)	7.4 KB	Standard
<input type="checkbox"/>	script.js	js	December 11, 2025, 21:26:27 (UTC+02:00)	7.0 KB	Standard

○

- **Amazon CloudFront (CDN)**

- **Function:** Caches content at Edge Locations worldwide to minimize latency and improve load times.
- **Security:** Provides a secure layer (HTTPS) using an SSL certificate and manages access to the S3 origin.
- **Integration:** Configured with the S3 Bucket as the origin source.

mazesolver-dist [View metrics](#)

Details

Distribution domain name [dpyylirikv8oy6.cloudfront.net](#)

Billing Pay-as-you-go [Switch to a plan](#)

ARN [arn:aws:cloudfront::491085402769:distribution/E1MRPVYLXTCU6E](#)

Last modified December 11, 2025 at 9:40:54 PM UTC

General | Security | Origins | Behaviors | Error pages | Invalidations | Logging | Tags

Settings

Name mazesolver-dist [Edit](#)

Description -

Price class Use all edge locations (best performance)

Supported HTTP versions HTTP/2, HTTP/1.1, HTTP/1.0

Alternate domain names [zaher.online](#) [Route domains to CloudFront](#)

Custom SSL certificate [zaher.online](#)

Security policy TLSv1.2_2021

Standard logging [Off](#)

Cookie logging [Off](#)

Default root object -

○

- **Amazon Route53 (DNS)**

- Used for Domain Name System management (`zaher.online`).
- **Function:** Routes user traffic to the CloudFront distribution using an **Alias Record**, ensuring a branded and professional entry point.

Hosted zone details

Records (4)

Record name	Type	Routing policy	Differences	Alias
<code>zaher.online</code>	A	Simple	-	Yes
<code>zaher.online</code>	NS	Simple	-	No
<code>zaher.online</code>	SOA	Simple	-	No
<code>_88017392d55e45bde93365b599e6f0b3.zaher.online</code>	CNAME	Simple	-	No

Backend & Compute Components

- **Amazon APIGateway (HTTP API)**
 - Acts as the unified entry point for the backend, receiving requests from the frontend and routing them to the appropriate Lambda function.
 - **Routes:** Defined paths for each algorithm:
 - POST `/solve/bfs`
 - POST `/solve/dfs`
 - POST `/solve/astar`
 - **CORS: Cross-Origin Resource Sharing** is enabled to allow the browser to securely send API requests from the frontend domain.

APIs (1/1)

Name	Description	ID	Protocol	API endpoint type	Created	Security policy	API status
<code>maze_api</code>		<code>bgmrmmybcpe</code>	HTTP	Regional	2025-12-11	-	-

The screenshot shows the AWS API Gateway interface for the 'maze_api'. It lists three routes:

- /solve**: POST method, AWS Lambda function.
- /bfs**: POST method, AWS Lambda function.
- /dfs**: POST method, AWS Lambda function.

- **AWS Lambda (Compute)**

- Represents the core logic of the system. Python (Flask-based logic) is used to execute the pathfinding algorithms. A **Microservices Pattern** was adopted by creating a separate function for each algorithm:
 - **bfs-lambda**: Calculates the shortest path using Breadth-First Search.
 - **dfs-lambda**: Explores paths using Depth-First Search.
 - **astar-lambda**: Finds the optimal path using the A* Heuristic.

The screenshot shows the AWS Lambda Functions list. It displays three functions:

Function name	Description	Package type	Runtime	Type	Last modified
bfs-lambda	-	Zip	Python 3.14	Standard	20 hours ago
astar-lambda	-	Zip	Python 3.14	Standard	20 hours ago
dfs-lambda	-	Zip	Python 3.14	Standard	21 hours ago

- **AWS Lambda Layer**

- To adhere to the **DRY (Don't Repeat Yourself)** principle, shared logic and helper functions (such as `get_neighbors` and `reconstruct_path`) were encapsulated in a separate **Lambda Layer** named `maze-modules`.
 - **Benefit:** Reduces code redundancy, simplifies maintenance, and ensures consistency across all functions.

Data Flow Workflow

1. The user navigates to <https://zaher.online>.
2. **Route 53** resolves the domain and routes traffic to **CloudFront**.
3. **CloudFront** serves the cached frontend assets from **S3** to the user.
4. The user clicks "Simulate"; the JavaScript client sends a POST request containing the start node, end node, and wall data.
5. **API Gateway** receives the request and routes it to the specific **Lambda Function** (e.g., bfs-lambda).
6. The **Lambda** function imports shared logic from the **Layer**, executes the algorithm, and returns the result (Visited Nodes & Path) as JSON.
7. The Frontend processes the response and visualizes the animation in real-time.

Key Technical Benefits

- **Serverless:** No infrastructure management (No EC2 instances to patch or maintain).
- **Cost-Effective:** Costs are incurred only when code is running (Pay-as-you-go).
- **Scalability:** The system automatically scales to handle thousands of concurrent requests via AWS Lambda.
- **Performance:** High availability and low latency achieved through CloudFront's global edge network.

Source Code & Live Demo

The complete project source code (Frontend & Backend) is available on GitHub. Additionally, a persistent **Live Demo** has been deployed on Vercel to ensure accessibility for review purposes after decommissioning the AWS resources for cost optimization.

- **GitHub Repository:** [GitHub](#)
- **Live Demo (Vercel):** [Live Demo](#)

