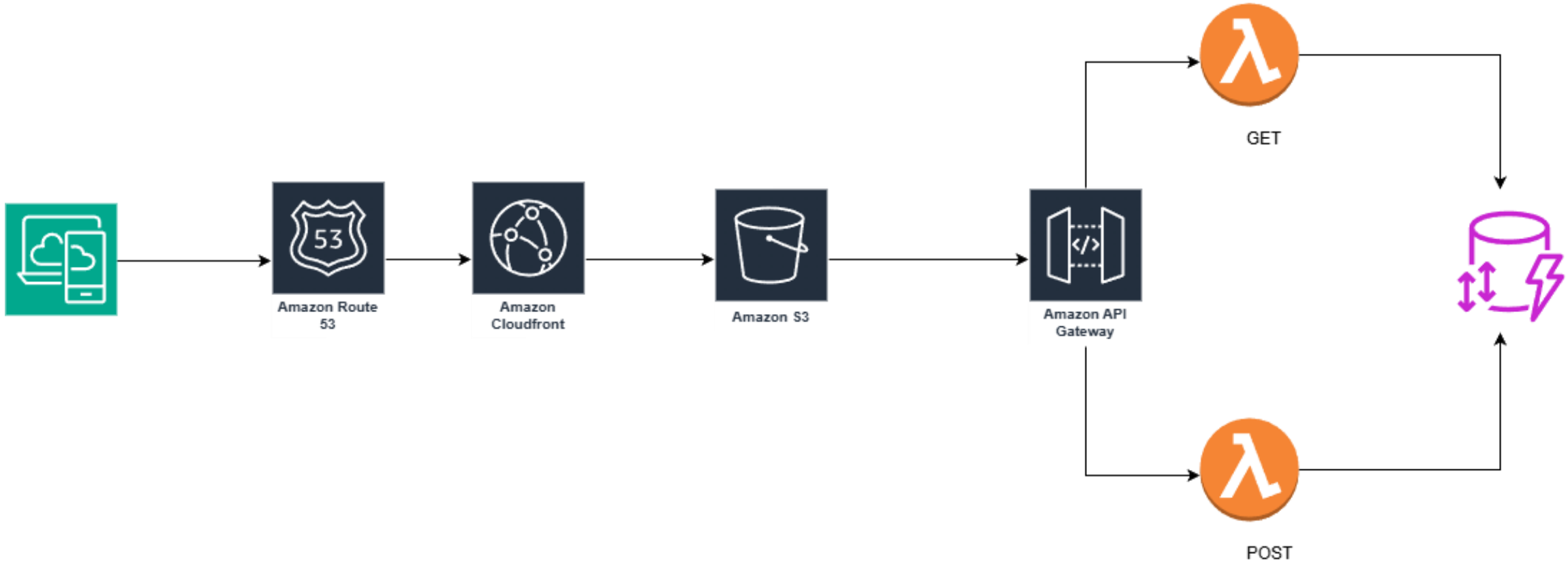
Deploying a Serverless Web Application on AWS Using S3, CloudFront, API Gateway, Lambda, DynamoDB, and Custom Domain Integration

AGENDA

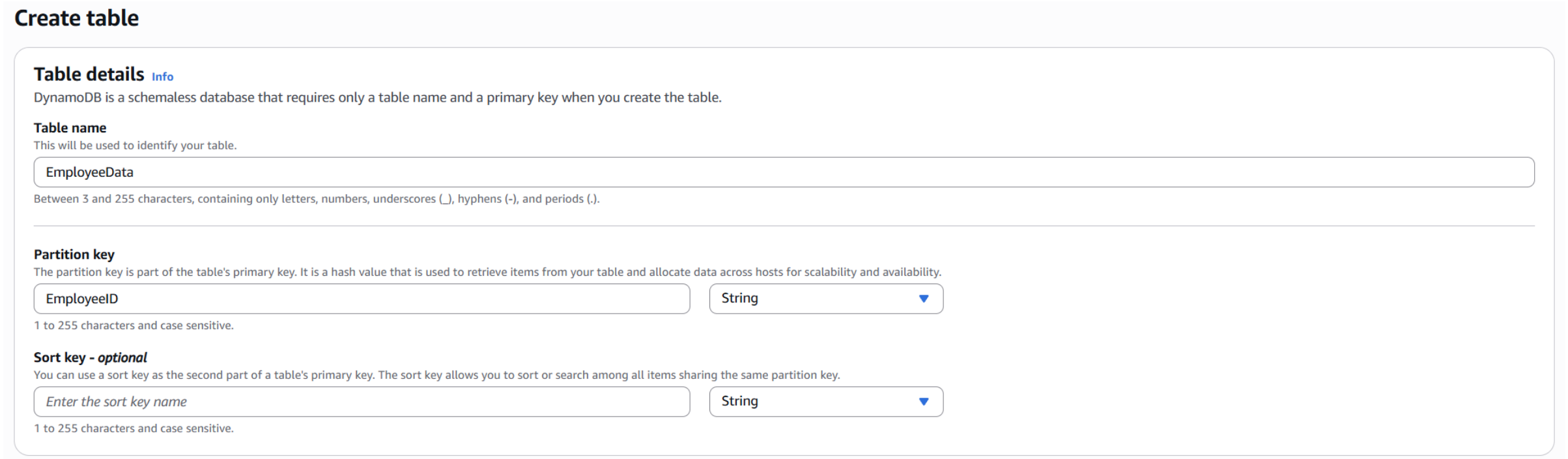
- Deploy Lambda Functions for GET and POST
- Set Up DynamoDB Table
- Configure API Gateway
- Host Frontend on S3
- Set Up CloudFront CDN
- Troubleshooting & Common Issues


Amazon Route 53
Amazon Cloudfront
Amazon S3
Amazon API Gateway
GET
POST

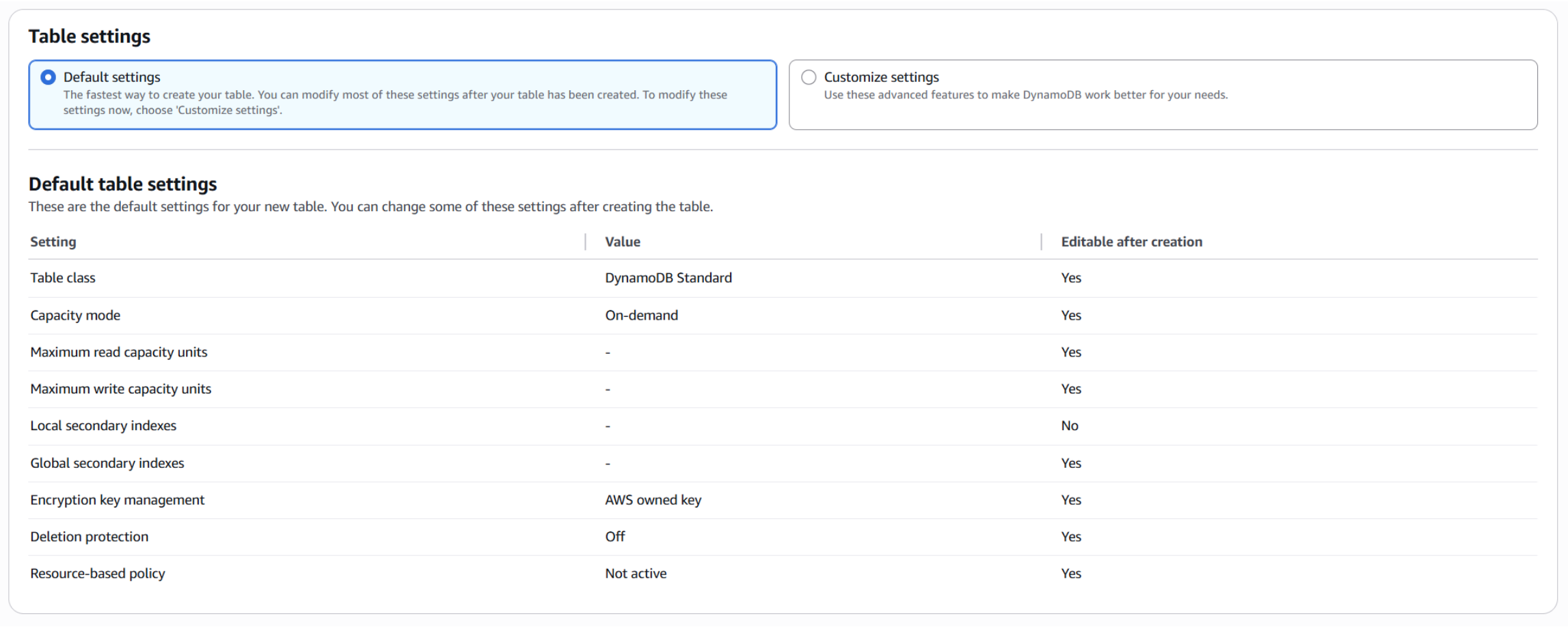
Set Up DynamoDB Table

- Open AWS Console → search “DynamoDB”
- Click “Create table”
- Enter Table Name
- Set Partition Key (e.g., EmployeeID – String)
- Leave Sort Key OFF (optional)
- In Table settings, keep Default settings
- Keep Table class as “DynamoDB Standard”
- Click “Create table”

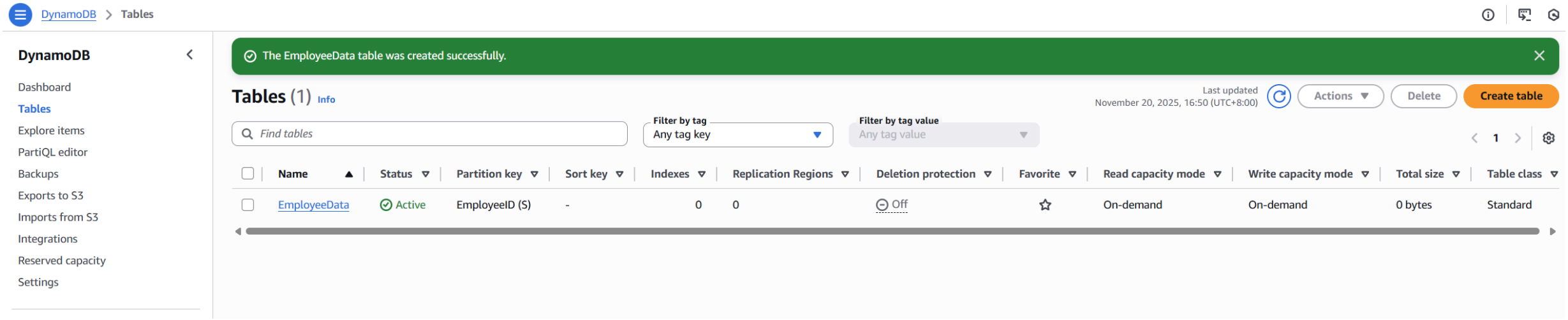
Set Up DynamoDB Table



Set Up DynamoDB Table



Set Up DynamoDB Table



Deploy Lambda Functions

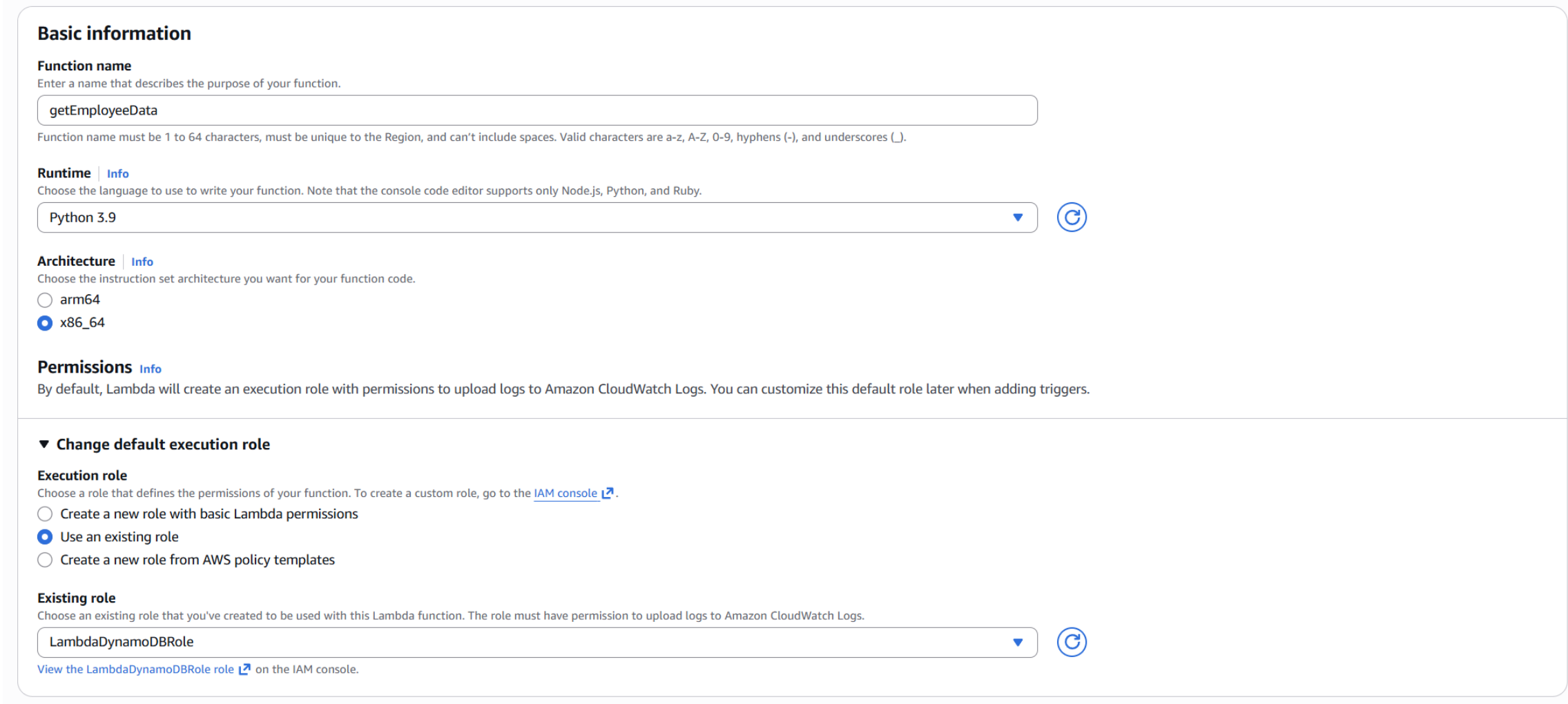
- Go to AWS Console → Open Lambda
- Click Create function
- Select Author from scratch
- Enter function name
- Choose runtime (Python/Node.js, etc.)
- Select or create an execution role
- Click Create function
- Add code → Deploy → Test

Deploy Lambda Functions for GET and POST

Creating a Lambda Function – getEmployeeData

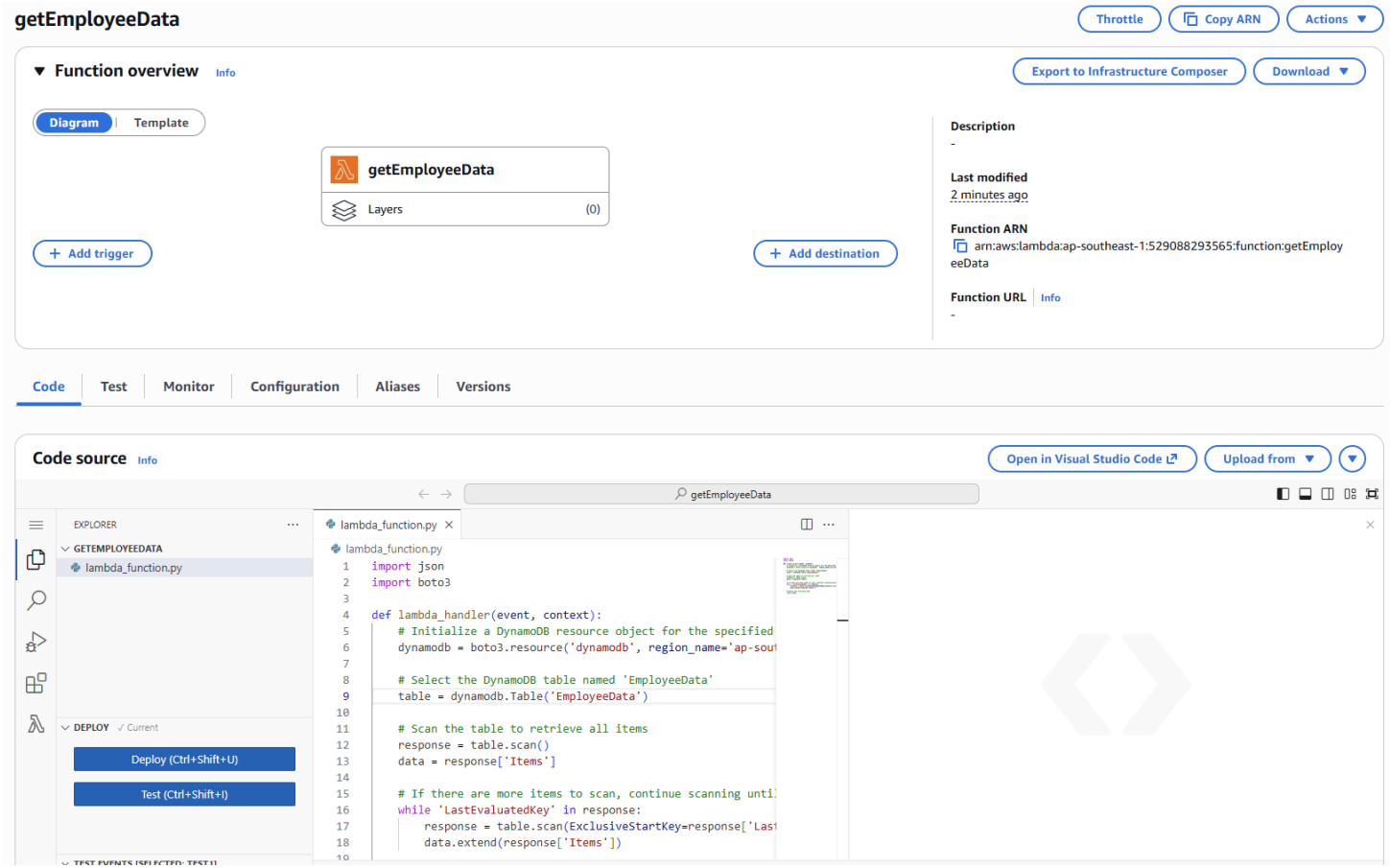
- **Function Name:** getEmployeeData
- **Runtime:** Python 3.9
- **Architecture:** x86_64
- **Execution Role:** Custom IAM Role
  - Create role in IAM console
  - Attach the role to Lamda
  - Ensure the role has permissions for CloudWatch logging and DynamoDB access
- **Click on Create Functions**

Deploy Lambda Functions for GET and POST



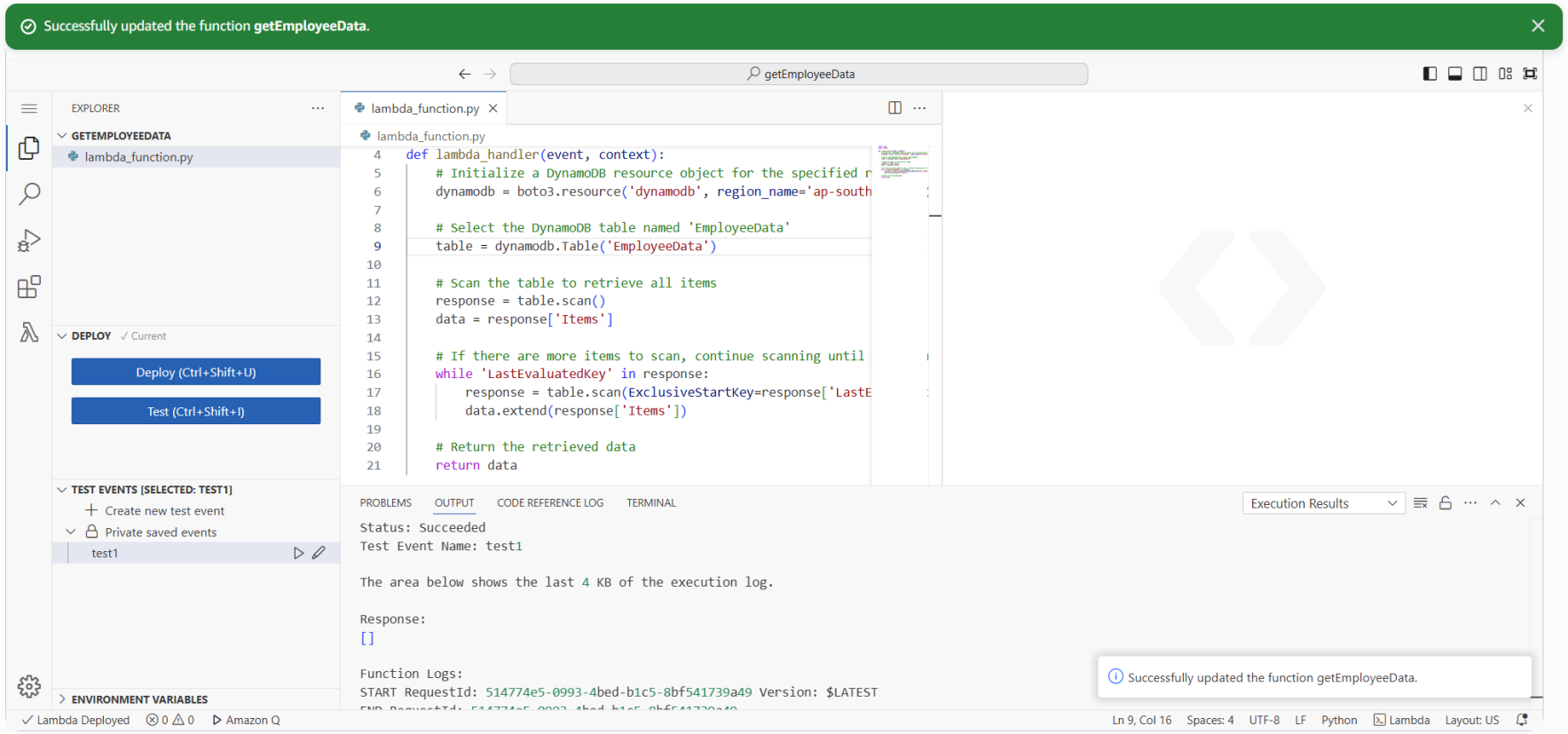
Deploy Lambda Functions for GET and POST

Deploying Python Code on AWS Lamda (Serverless Deployment)



Deploy Lambda Functions for GET and POST

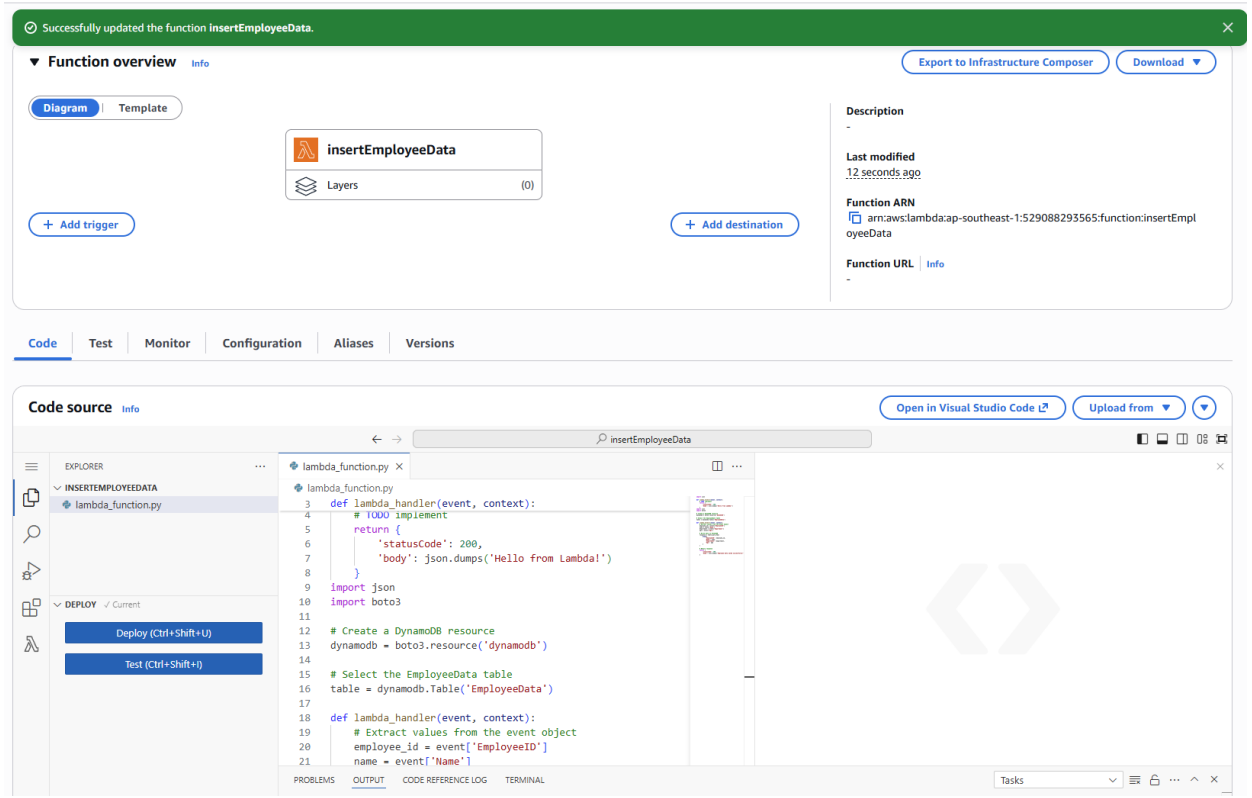
Lamda Test Execution



Deploy Lambda Functions for GET and POST

Creating a Lambda Function – insertEmployeeData

Remark: *Follow the steps on pages 8–11 to create the insertEmployeeData Lambda function*

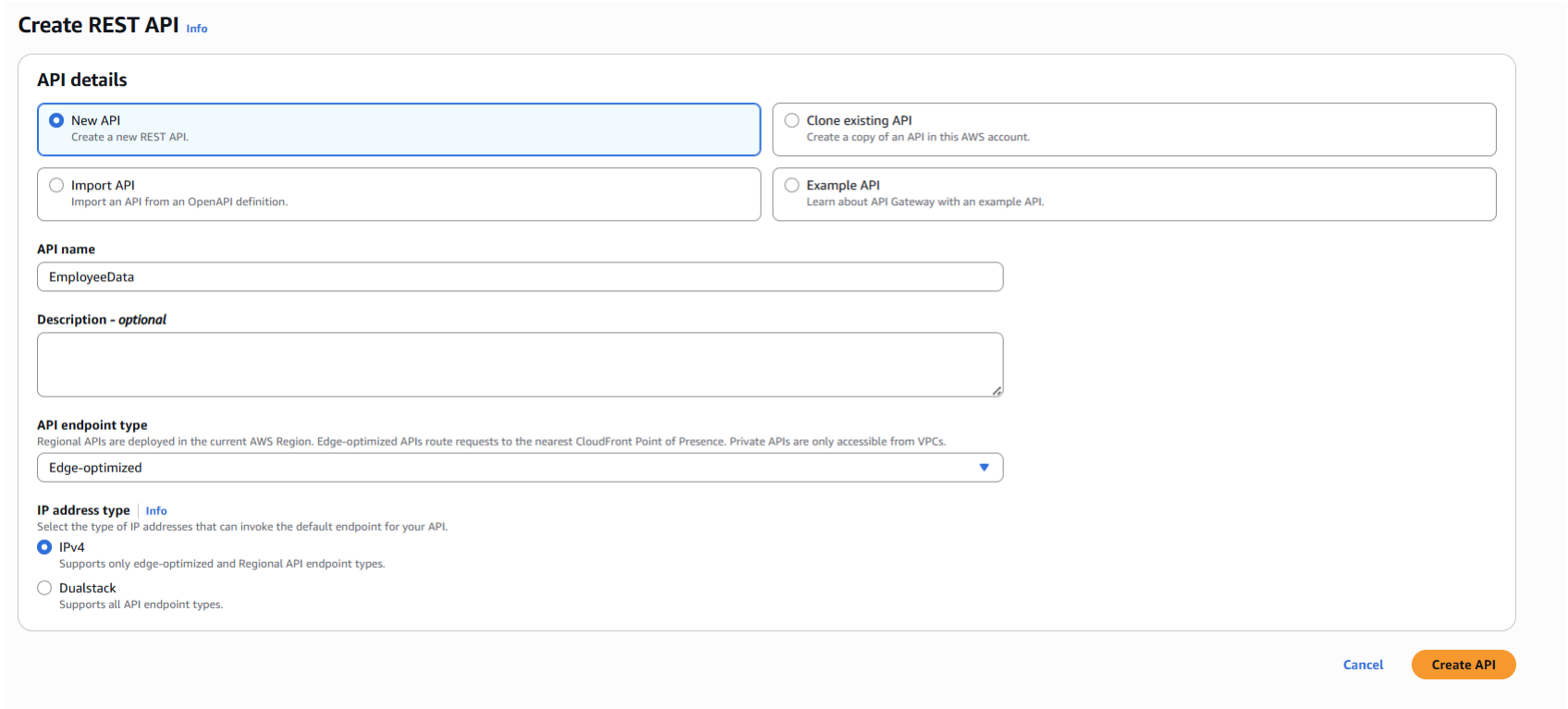


Configure API Gateway

- Open AWS Console → search “API Gateway”
- Click “Create API”
- Choose “REST API (Build)”
- Select “New API” → Give API Name
- Click “Create API”
- Create a Resource
- Create a Method (GET or POST)
- Choose Integration Type: Lambda Function
- Select your Lambda → Click “Save”
- Deploy API → Create a new Stage (e.g., prod)
- Copy the Invoke URL and test your REST API

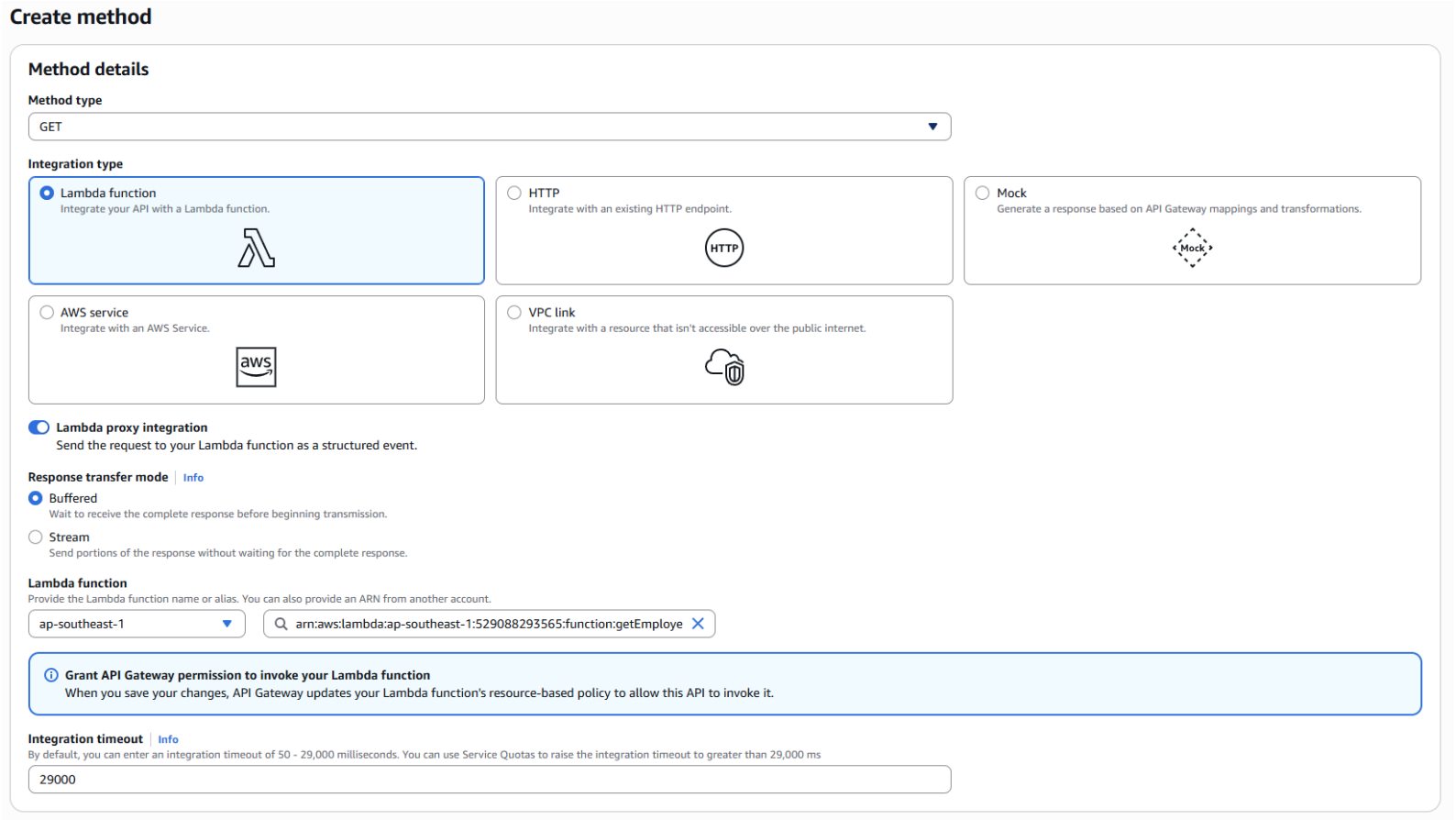
Configure API Gateway

- Create an API and select REST API



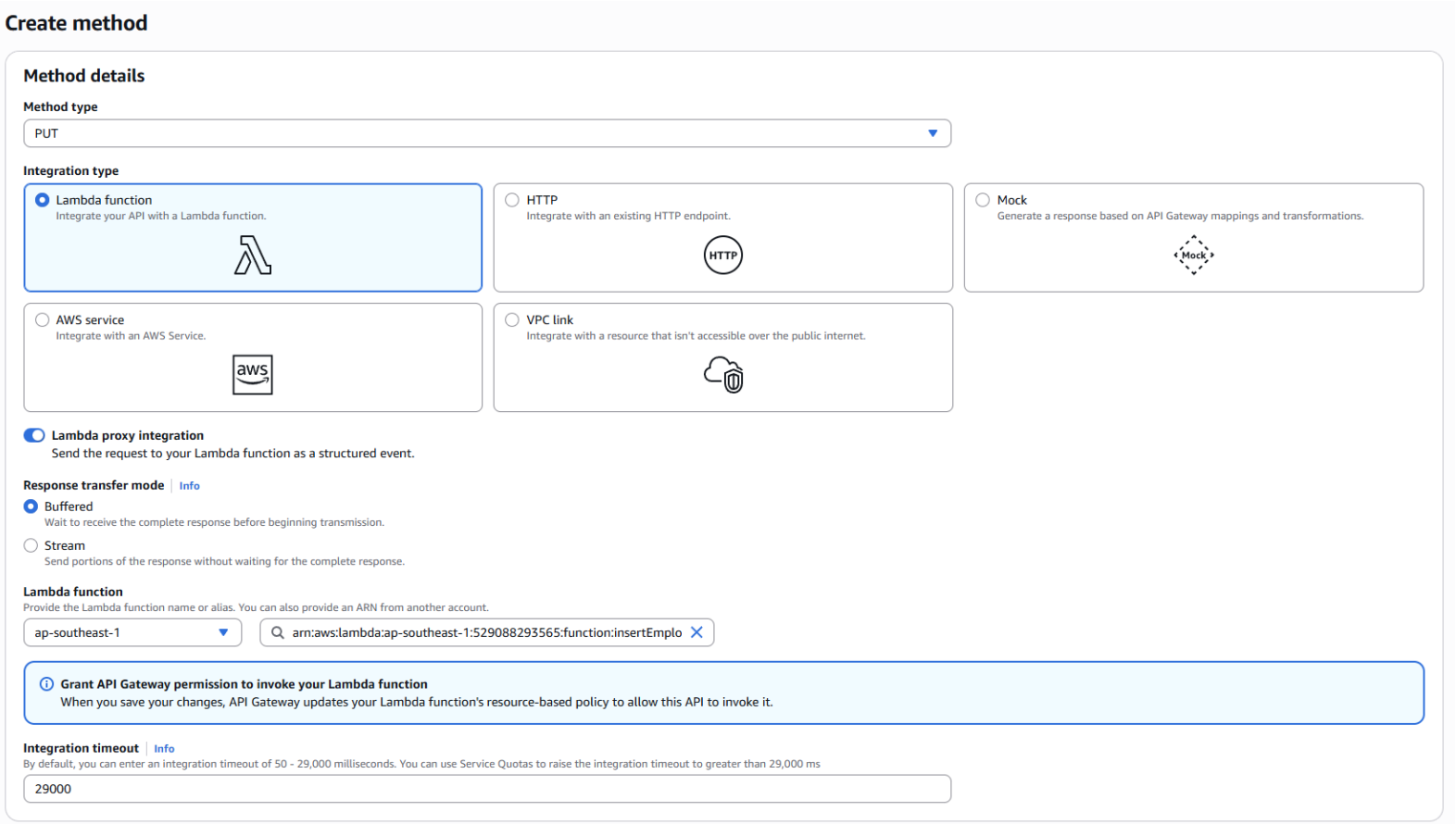
Configure API Gateway

- Create a GET method



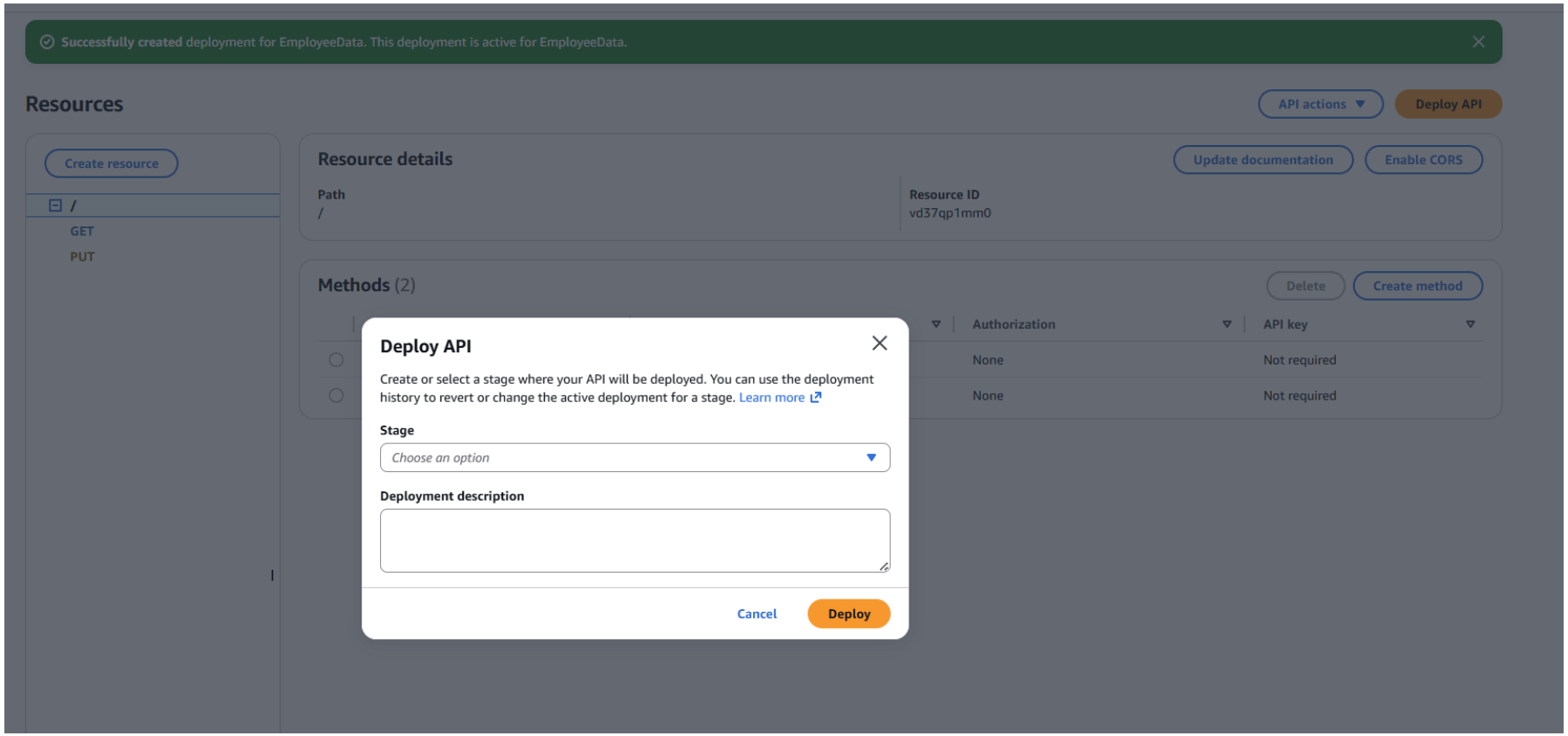
Configure API Gateway

- Create a PUT method



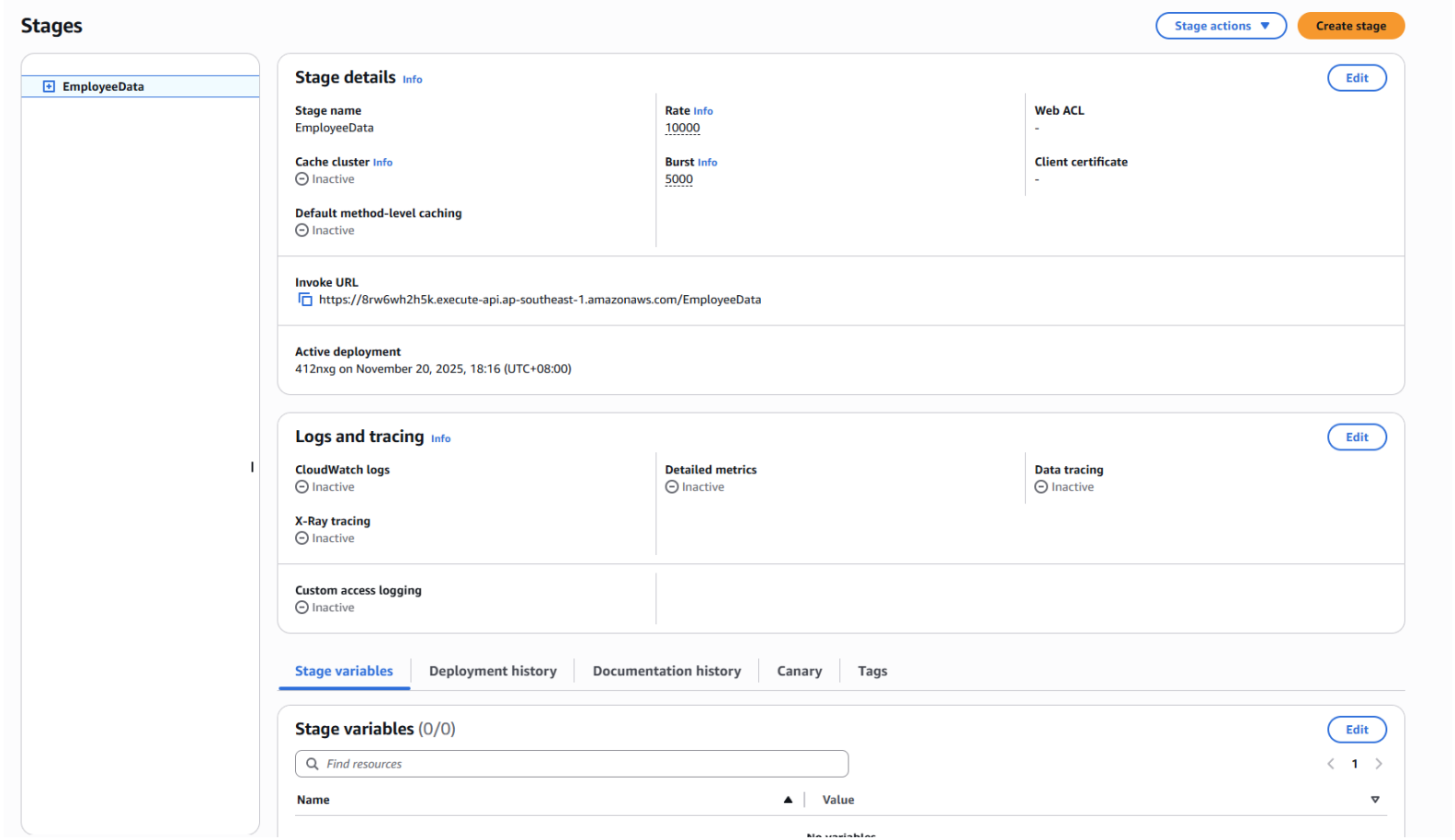
Configure API Gateway

- Deploy the API to a stage



Configure API Gateway

- API Deployed

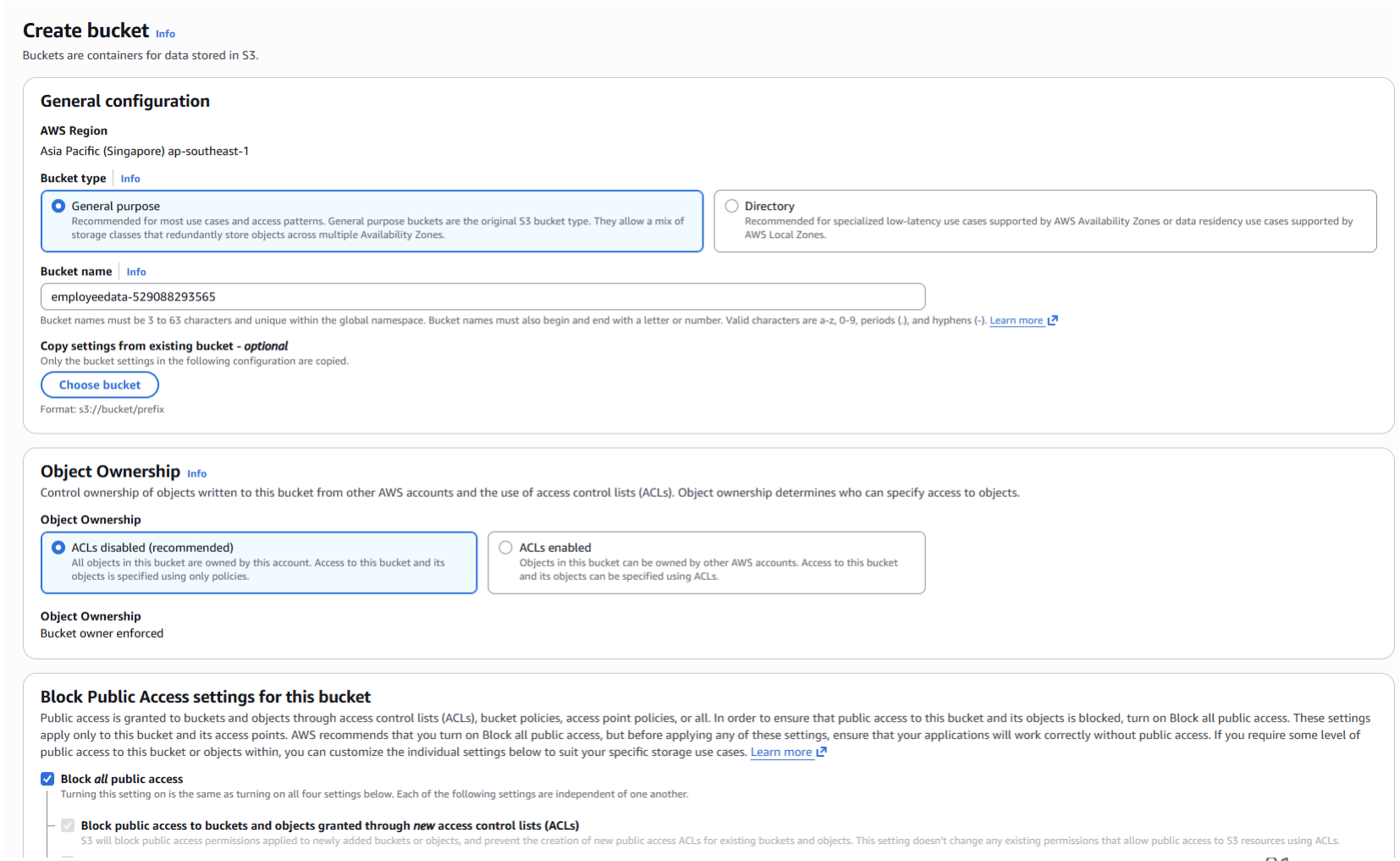


Host Frontend on S3

- Open AWS Console → search “S3”
- Create bucket → Uncheck “Block all public access”
- Upload your build files (index.html, JS)
- Go to Permissions → Add Bucket Policy to allow public read
- Go to Properties → Enable “Static website hosting”
- Set Index document = index.html
- Save changes
- Copy the S3 Website Endpoint URL and access your site

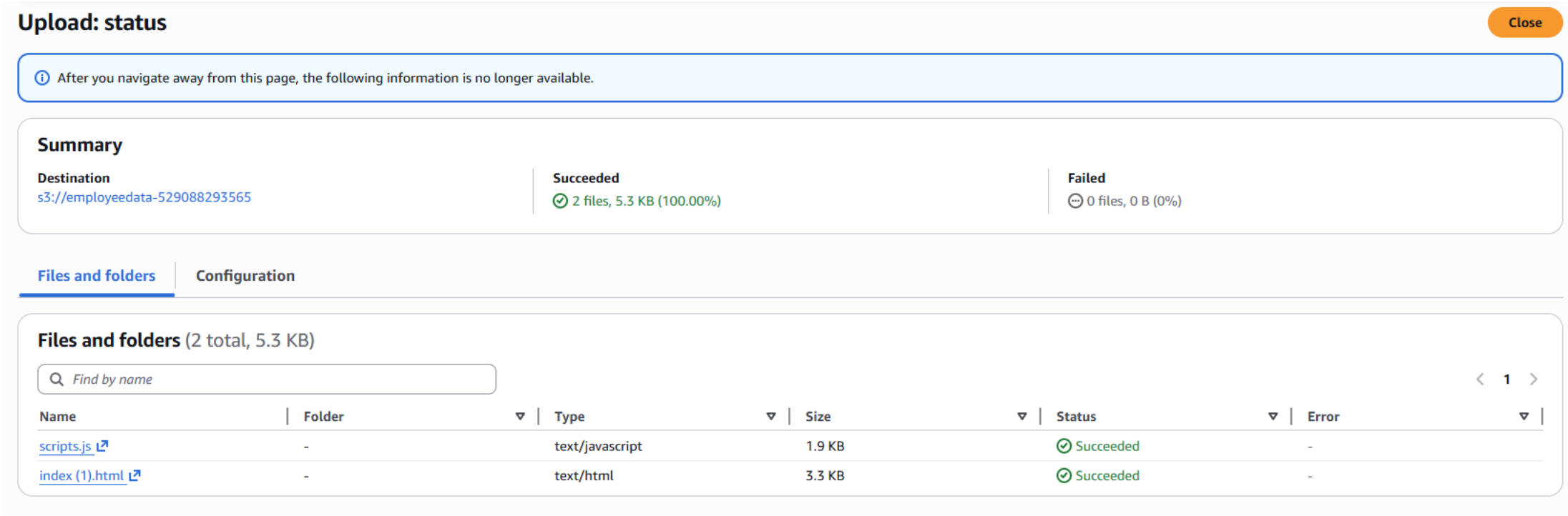
Host Frontend on S3

- Create an S3 bucket



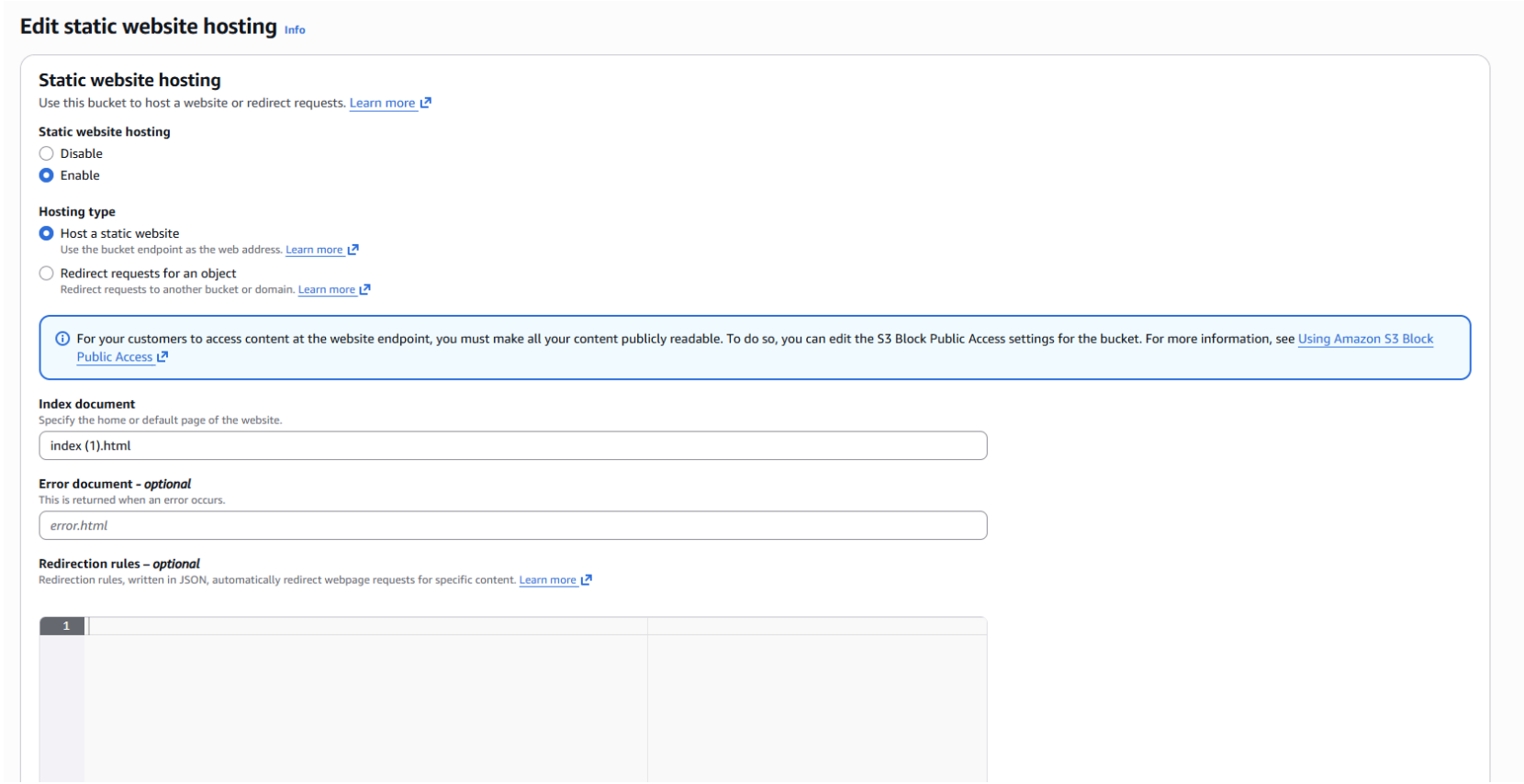
Host Frontend on S3

- Upload the index.html and frontend JavaScript files to the S3 bucket



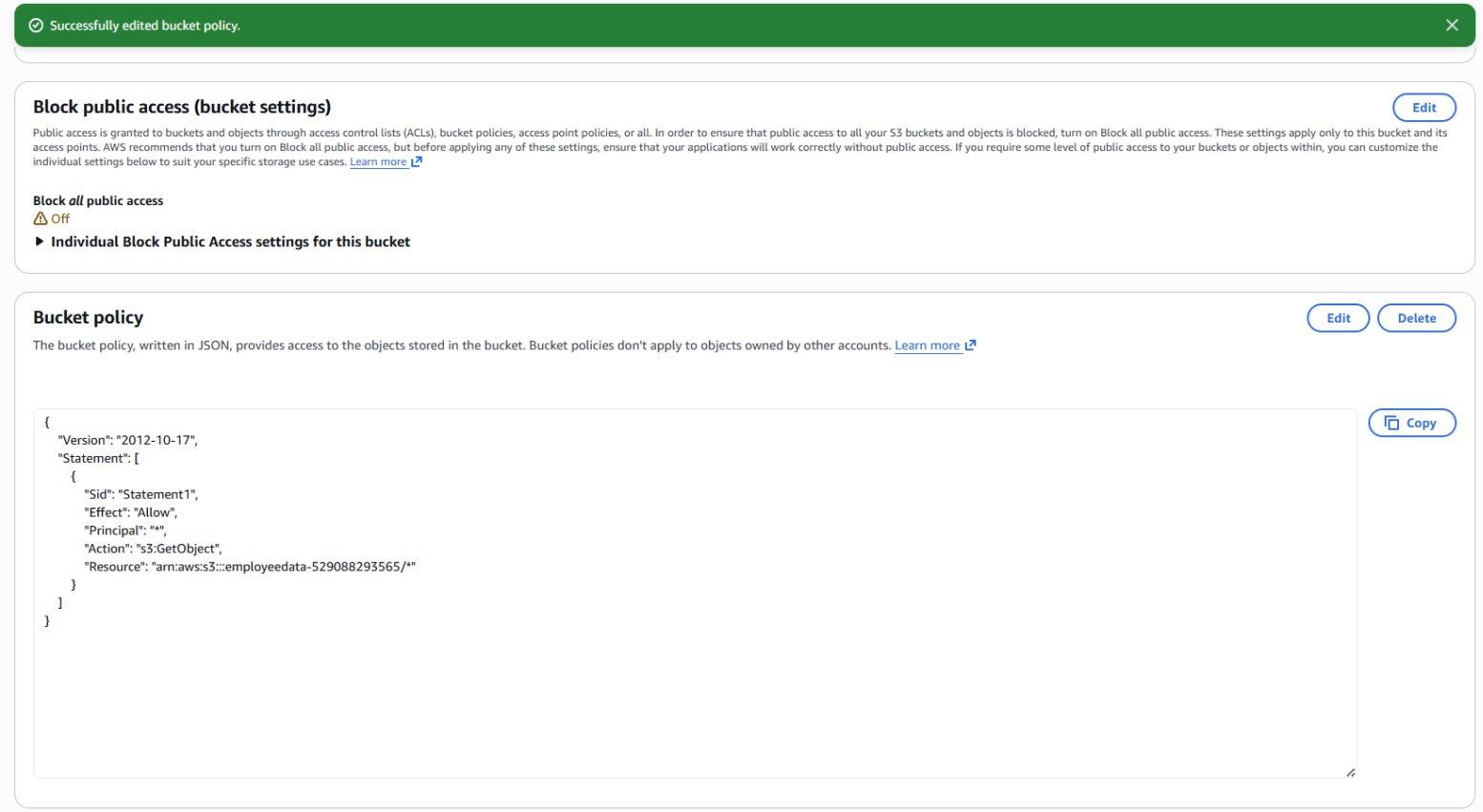
Host Frontend on S3

- Enable Static website hosting



Host Frontend on S3

- Edit the bucket policy under Permissions

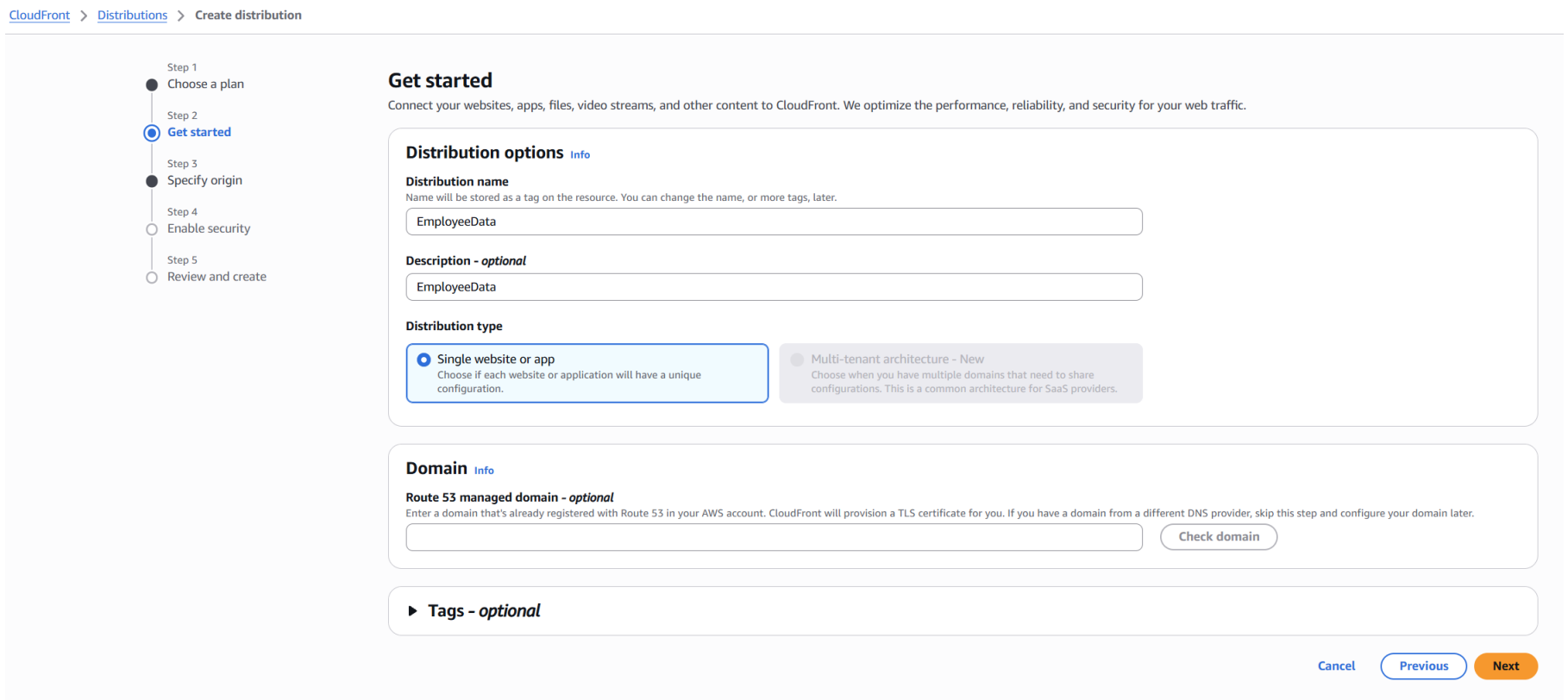


Set Up CloudFront CDN

- Go to CloudFront → Create Distribution
- Choose Distribution Type: Single Page Web Application (or Web)
- Select your S3 bucket as the Origin
- Set Origin Access = OAC to secure S3
- Click Create OAC → Copy policy → Paste into S3 bucket policy
- Set Default Root Object = index.html
- Leave other settings as default and Create Distribution
- Wait for deployment, then copy the CloudFront Domain Name

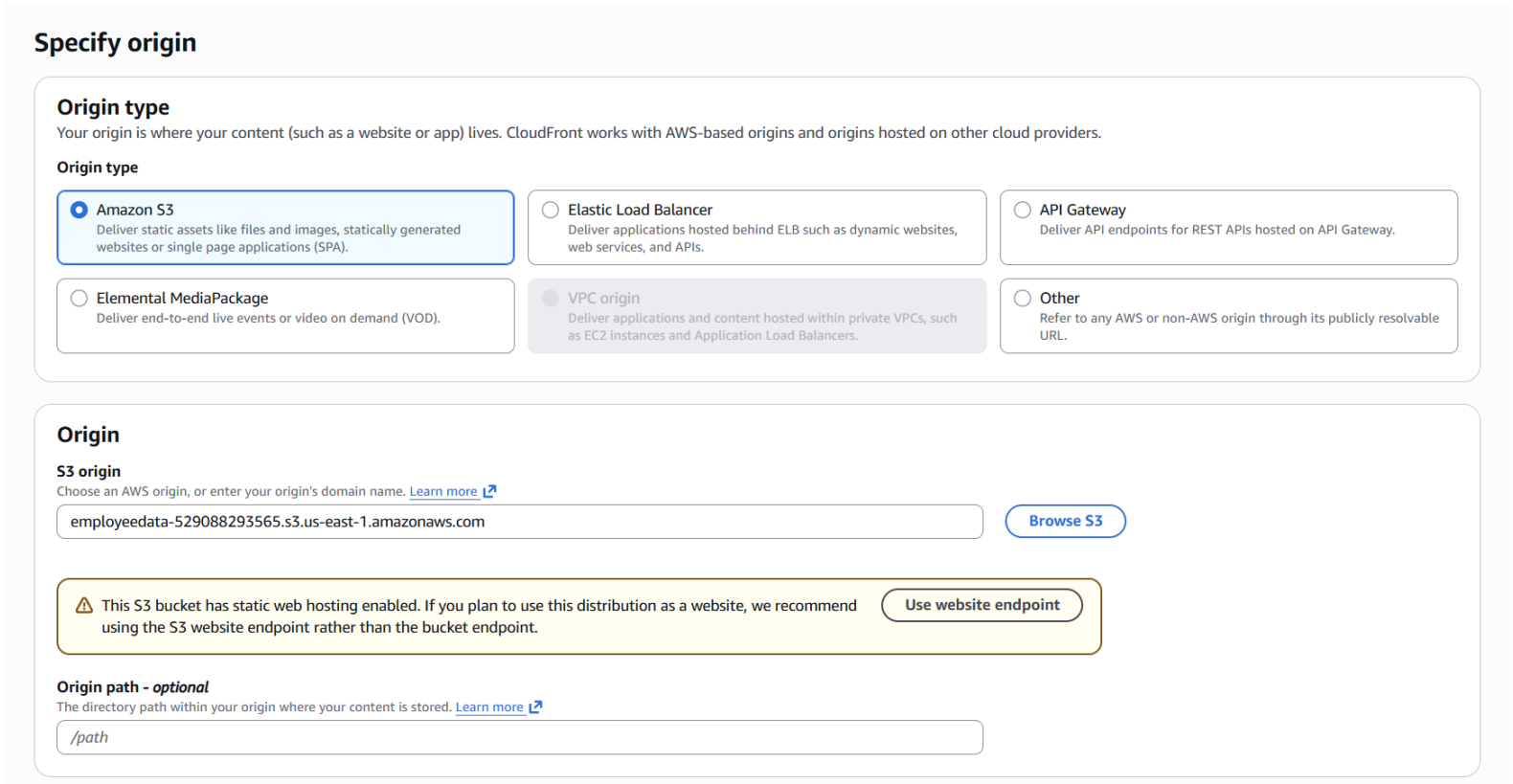
Set Up CloudFront CDN

- Create a CloudFront distribution.



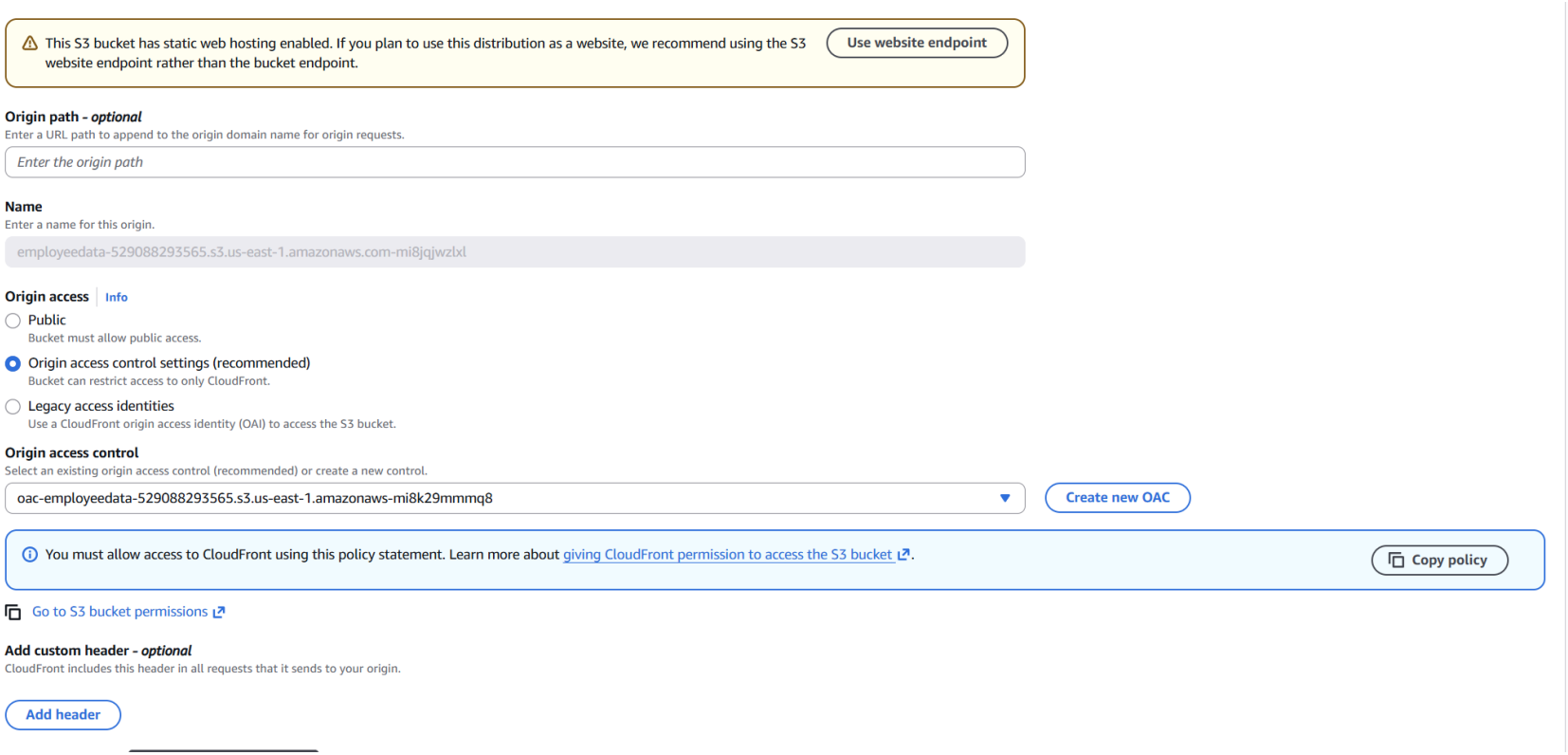
Set Up CloudFront CDN

- Selected S3 bucket as the *origin* for CloudFront



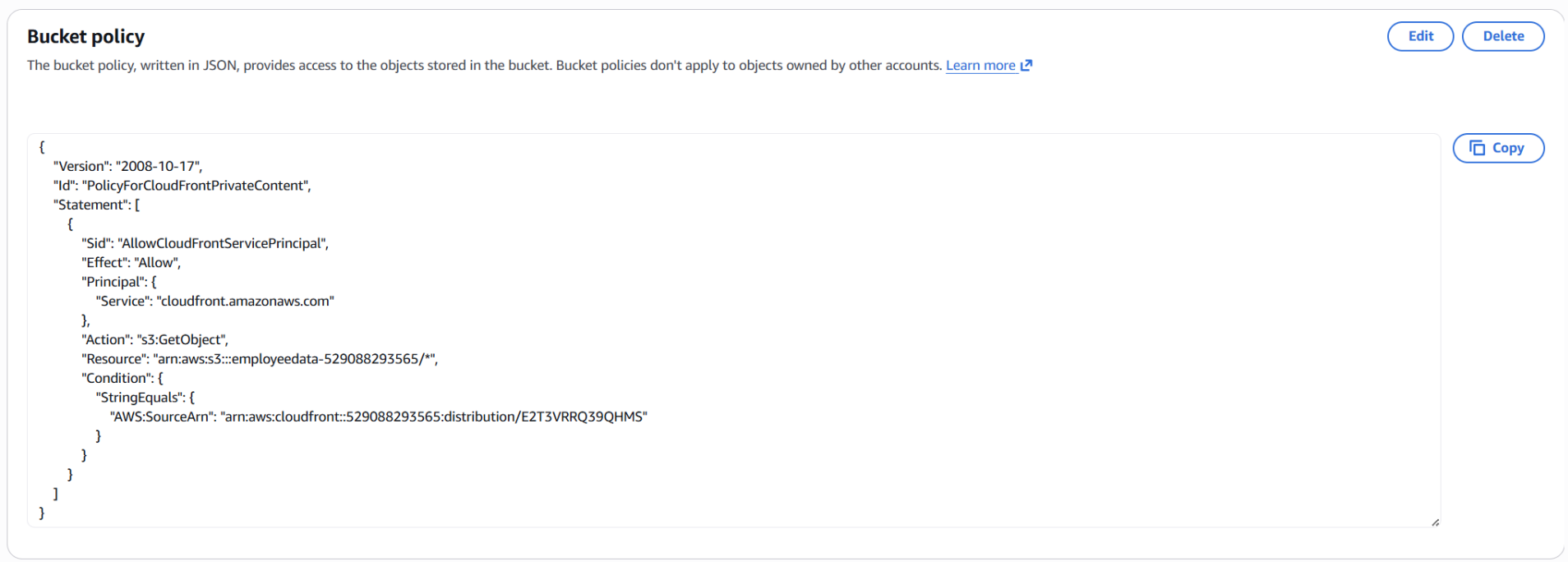
Set Up CloudFront CDN

- Create a new CloudFront Origin Access Control (OAC) to allow CloudFront to securely access my S3 bucket.



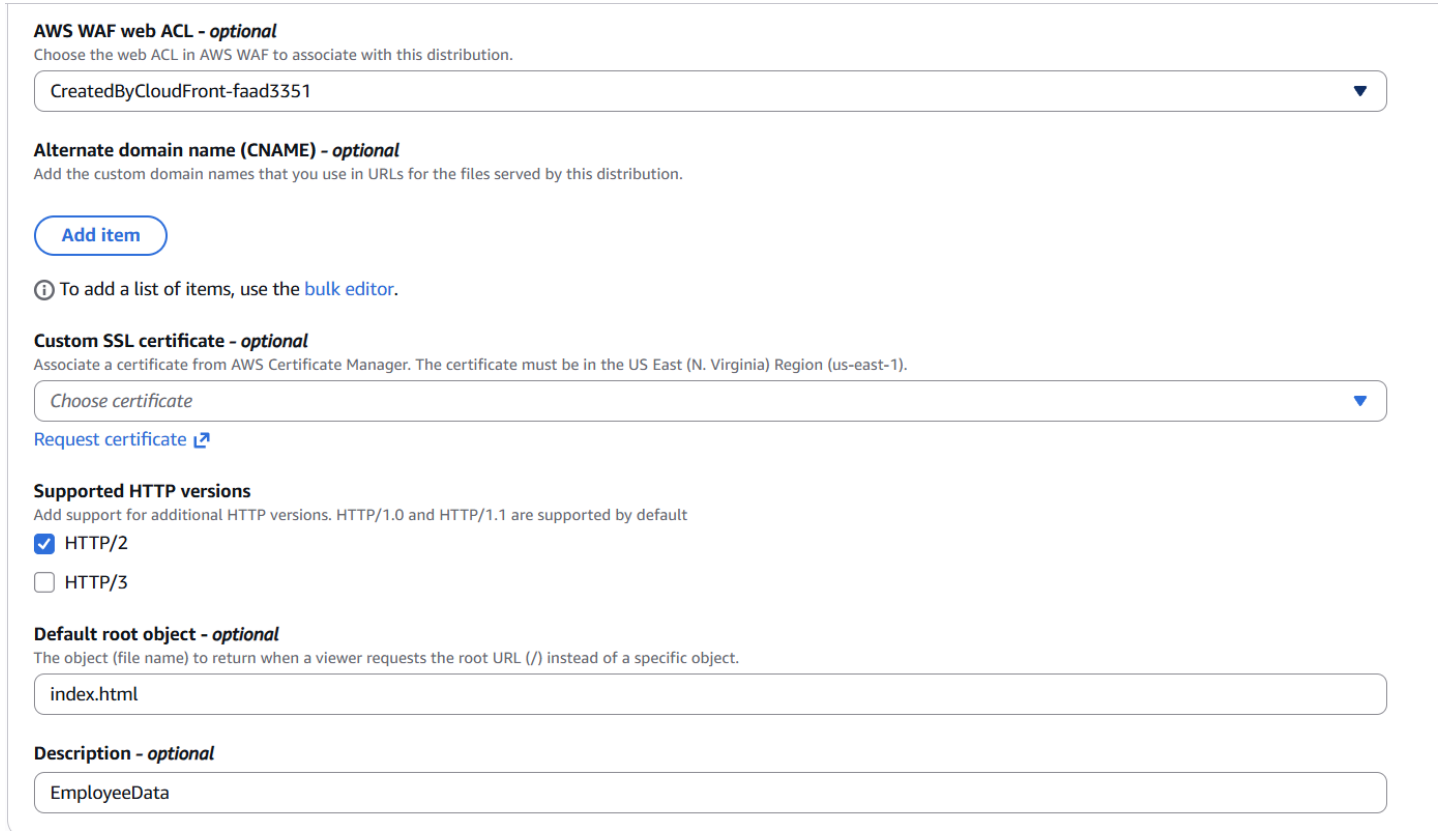
Set Up CloudFront CDN

- Copy the generated OAC bucket policy and paste it into the S3 bucket policy to allow CloudFront access



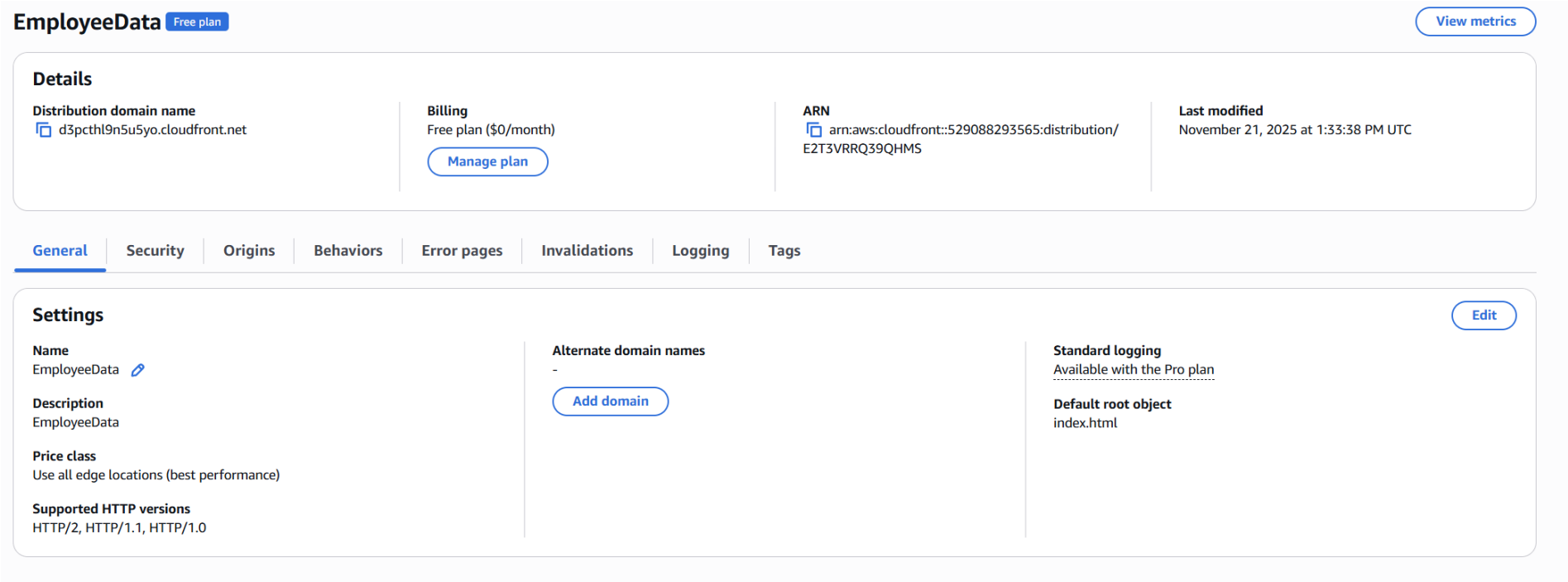
Set Up CloudFront CDN

- Set Default Root Object = index.html



Set Up CloudFront CDN

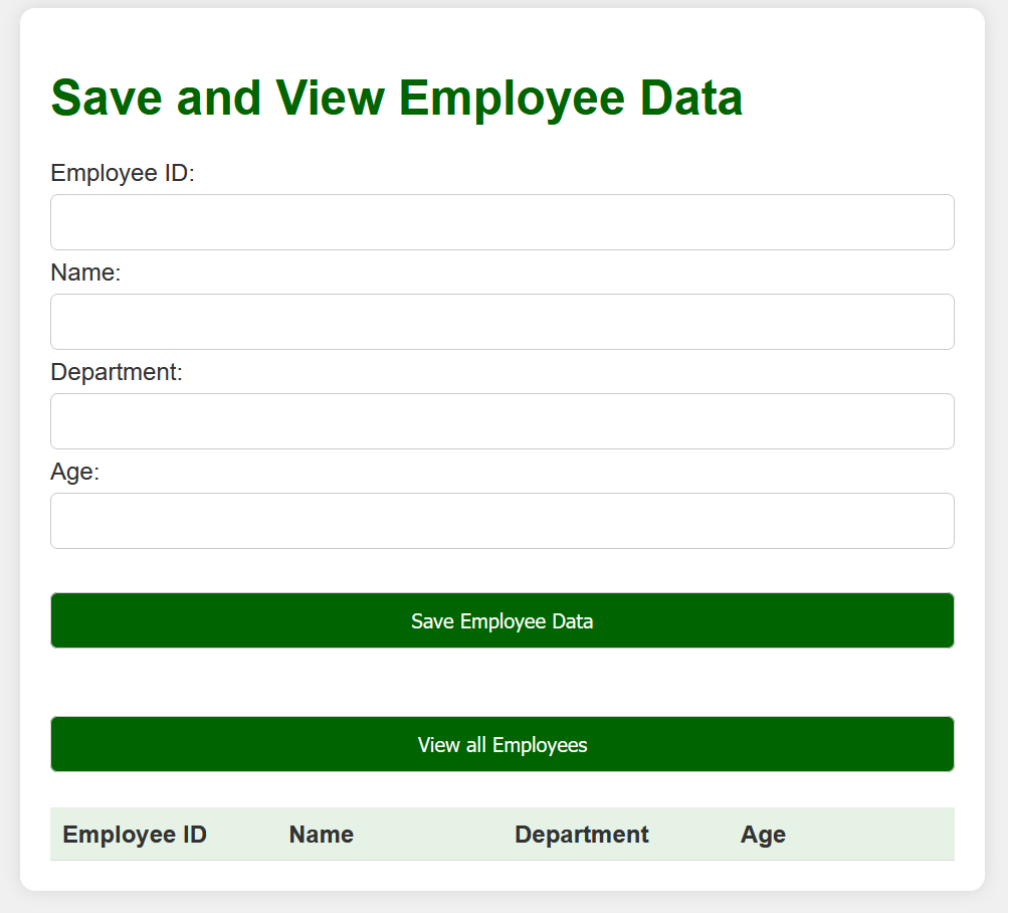
- CloudFront is deployed. Copy the domain name and open it in your browser.



Set Up CloudFront CDN

- CloudFront is deployed. Copy the domain name and open it in your browser.

API stores employee data in DynamoDB and fetches all records on "View All Employees."



Troubleshooting & Common Issues

❑ 403 Forbidden

- OAC not configured correctly
- S3 bucket policy misconfigured
- Lambda IAM role missing required permissions
- API CORS Misconfiguration

❑ 404 Not Found

- Default Root Object was not configured in CDN
- API called with wrong resource path → Endpoint not found → 404 error.