

# AWS Translate Project Documentation

## Index

1. Project Overview
2. Architecture
3. Deployment & Configuration
4. Security & Permissions
5. Monitoring & Logging
6. Cost Considerations
7. Future Improvements
8. References
9. Proof of Concept

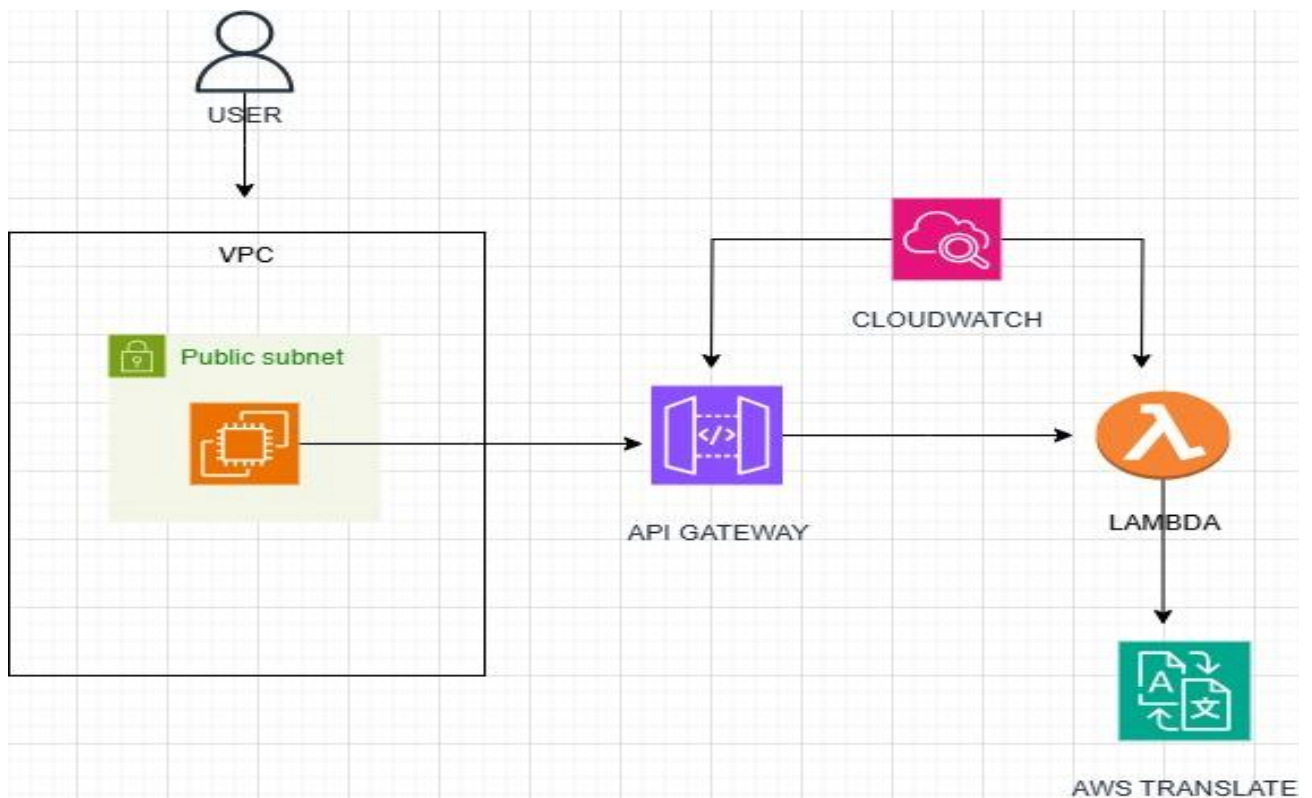
## 1. Project Overview

**Project Name:** Multilingual Translation Web App

**Description:** A web application that allows users to select a target language, input text, and receive translations using AWS Translate. The frontend can be hosted on an Amazon EC2 instance or an S3 bucket for lightweight, cost-efficient serverless application, with translation requests routed through API Gateway and processed by AWS Lambda.

**Business Use Case:** Helps businesses and users by providing real-time multilingual support for global customers.

## 2. Architecture diagram



### AWS Services Used:

**Amazon EC2** – Hosts the frontend (HTML, CSS, JS) and serves it over HTTP/HTTPS.

**Amazon API Gateway** – Exposes REST endpoints to handle translation requests from the frontend.

**AWS Lambda** – Processes requests by calling AWS Translate API.

**AWS Translate** – Provides translation services.

**Amazon CloudWatch** – Logs API and Lambda execution details.

### Workflow

- User visits the web app hosted on EC2.
- User selects target language, enters text, and clicks Translate.
- Frontend JavaScript sends request to API Gateway endpoint.
- API Gateway forwards request to Lambda.
- Lambda calls AWS Translate API.

- Translated text is sent back to API Gateway.
- API Gateway returns response to the EC2 frontend.
- Frontend displays translated text to the user.

### 3. Deployment & Configuration

Terraform manages all resources

Full code and config can be found in this [GitHub repo](#)

#### Provisioning:

- EC2 instance launched with a web server (Apache) to serve frontend files.
- API Gateway deployed with a POST method integrated with Lambda.
- Lambda configured with AWS Translate permissions.

#### Key Configuration Details:

- EC2 instance in a public subnet, secured with a Security Group allowing HTTP/HTTPS.
- API Gateway with CORS enabled to accept requests from EC2 frontend domain.
- Lambda uses IAM execution role with TranslateText permission.

### 4. Security & Permissions

#### EC2:

Security Group restricted to HTTP/HTTPS inbound only.

Key pair used for secure SSH access.

#### API Gateway:

Secured with API keys or Cognito authentication (optional).

### **IAM Roles:**

Lambda role restricted to Translate and CloudWatch logging.

### **Network Security:**

VPC rules to isolate backend resources.

## **5. Monitoring & Logging**

**EC2:** OS-level monitoring with CloudWatch Agent.

**Lambda:** Logs stored in CloudWatch for debugging.

**API Gateway:** Access and error logs enabled.

**Metrics Monitored:** API Gateway error rates, Lambda execution failures, EC2 CPU/Memory.

## **6. Cost Considerations**

**EC2:** Ongoing hourly cost depending on instance type.

**AWS Translate:** Pay per character translated.

**API Gateway:** Billed per API request.

**Lambda:** Pay per execution time.

**Optimization:** Use a small EC2 instance (t2.micro under Free Tier) for frontend; batch translations to reduce Translate calls.

## **7. Future Improvements**

- Replace EC2 with S3 static hosting + CloudFront for a fully serverless architecture.
- Add authentication via Cognito for secured access.
- Add DynamoDB to log and store translation history.
- Implement CI/CD pipeline to update frontend and backend automatically.

## 8. References

[AWS Translate Documentation](#)

[Amazon EC2 Documentation](#)

[Amazon API Gateway Documentation](#)

[AWS Lambda Documentation](#)

## 9. Proof of Concept

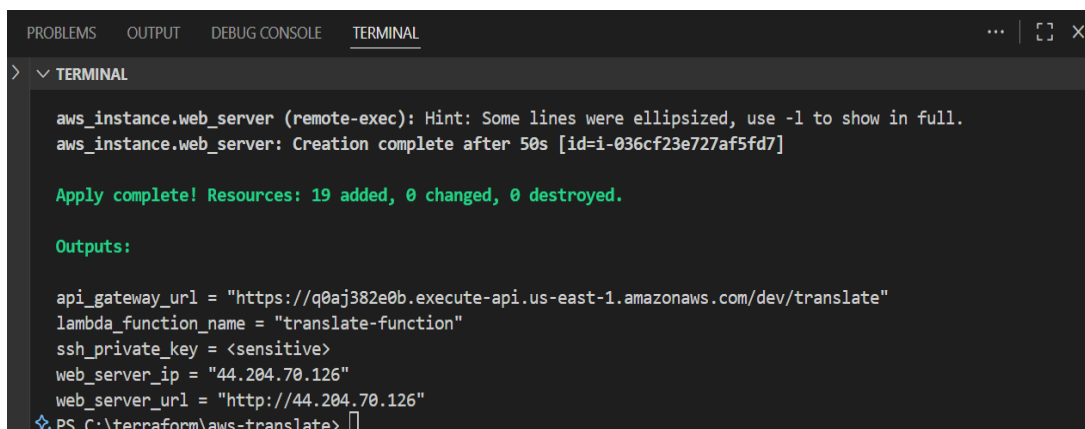
### Overview

This proof of concept demonstrates a real-time translation application built on AWS. The solution hosts a frontend on Amazon EC2, which sends translation requests via Amazon API Gateway. The requests invoke an AWS Lambda function, which uses Amazon Translate to return translated text.

This PoC validates the end-to-end workflow for live multilingual communication.

### Screenshots / Resource Verification

**After Terraform apply, the web server can be accessed through the outputs by clicking on the web server url provided.**

A screenshot of a terminal window with a dark background. The terminal shows the output of a Terraform apply command. At the top, there are tabs for 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', and 'TERMINAL', with 'TERMINAL' being the active tab. The output text includes a hint about ellipsized lines, a message stating 'Creation complete after 50s [id=i-036cf23e727af5fd7]', a green status message 'Apply complete! Resources: 19 added, 0 changed, 0 destroyed.', and a section titled 'Outputs:' in green. Below this, several key-value pairs are listed: 'api\_gateway\_url' pointing to an AWS execute-api endpoint, 'lambda\_function\_name' set to 'translate-function', 'ssh\_private\_key' marked as sensitive, 'web\_server\_ip' set to '44.204.70.126', and 'web\_server\_url' set to 'http://44.204.70.126'. The prompt at the bottom shows the user is in a PowerShell session at 'C:\terraform\aws-translate'.

## Screenshot of EC2 instance running frontend.

The screenshot displays the AWS Management Console interface for EC2 instances. The left sidebar shows the navigation menu with categories like EC2, Images, and AMIs. The main content area shows the 'Instances (1/1)' page. A table lists the instance 'translate-web-...' with ID 'i-036cf23e727af5fd7', state 'Running', and type 't2.micro'. Below the table, the 'Details' tab for the instance is selected, showing the 'Instance summary' with the instance ID, public IPv4 address (44.204.70.126), and private IPv4 addresses (172.31.81.247). The browser window below the console shows the web application running on the public IP address. The application is titled 'Language Translator' and features two dropdown menus for selecting languages (English and German), a 'Do it now' input field, a 'Translate' button, and a 'Mach es jetzt' input field.

EC2 > Instances

Instances (1/1) Info

Find Instance by attribute or tag (case-sensitive)

Instance state = running

Clear filters

Launch Instances

Name	Instance ID	Instance state	Instance type	Status check	Alarm status
translate-web-...	i-036cf23e727af5fd7	Running	t2.micro	2/2 checks passed	View alarms

i-036cf23e727af5fd7 (translate-web-server)

Details Status and alarms Monitoring Security Networking Storage Tags

Instance summary Info

Instance ID: i-036cf23e727af5fd7

Public IPv4 address: 44.204.70.126 | open address

Private IPv4 addresses: 172.31.81.247

Not secure 44.204.70.126

### Language Translator

English German

Do it now Translate Mach es jetzt

Watch the full video demo [here](#)

## Screenshot of API Gateway endpoint.

The screenshot displays the AWS API Gateway console. The top section shows the 'APIs (1/1)' list with a table containing one entry: 'translate-api' with ID 'q0aj382e0b', Protocol 'REST', API endpoint type 'Regional', and Created date '2025-09-04'. The bottom section shows the 'Resources' page for the 'translate-api', displaying the resource path '/' with methods 'OPTIONS' and 'POST'. The 'Methods (0)' section indicates no methods are defined.

**APIs (1/1)**

Name	Description	ID	Protocol	API endpoint type	Created
<a href="#">translate-api</a>		q0aj382e0b	REST	Regional	2025-09-04

**Resources**

**Create resource**

**Resource details**

Path: /

Resource ID: teh46gd2li

**Methods (0)**

No methods defined.

Watch the full video demo [here](#)

## Screenshot of Lambda and its logs in CloudWatch.

The image displays two screenshots from the AWS console. The top screenshot shows the 'translate-function' in the Lambda console. It includes a diagram with an API Gateway trigger, a description of the function, and tabs for Code, Test, Monitor, Configuration, Aliases, and Versions. The bottom screenshot shows the CloudWatch 'Log events' page for the same function. It displays a list of log events with timestamps and messages, including details about the function's execution and the translated text 'Mach es jetzt'.

**Lambda Console Screenshot:**

- Navigation: Lambda > Functions > translate-function
- Diagram: Shows the function architecture with an API Gateway trigger.
- Function Details:
  - translate-function**
  - Layers: (0)
  - API Gateway: + Add trigger
  - + Add destination
- Description:
  - Description:** -
  - Last modified:** 1 hour ago
  - Function ARN:** arn:aws:lambda:us-east-1:123456789012:function:translate-function
  - Function URL:** Info
- Tabs: Code, Test, Monitor, Configuration, Aliases, Versions
- Code source: Info, Open in Visual Studio Code, Upload from

**CloudWatch Log Events Screenshot:**

- Navigation: CloudWatch > Log groups > /aws/lambda/translate-function > 2025/09/04/[\$LATEST]30fcff18036e49e1aaec9a9a914786b7
- CloudWatch Sidebar:
  - Favorites and recents
  - Dashboards
  - AI Operations New
  - Alarms 0 0 0 0
    - In alarm
    - All alarms
    - Billing
  - Logs
    - Log groups
    - Log Anomalies
    - Live Tail
    - Logs Insights
- Log events:
  - Filter events - press enter to search
  - 1m 1h UTC timezone
  - Display
- Log events table:

Timestamp	Message
2025-09-04T20:35:17.550Z	INFO Handling POST request
2025-09-04T20:35:17.550Z	INFO Parsed request

Watch the full video demo [here](#)