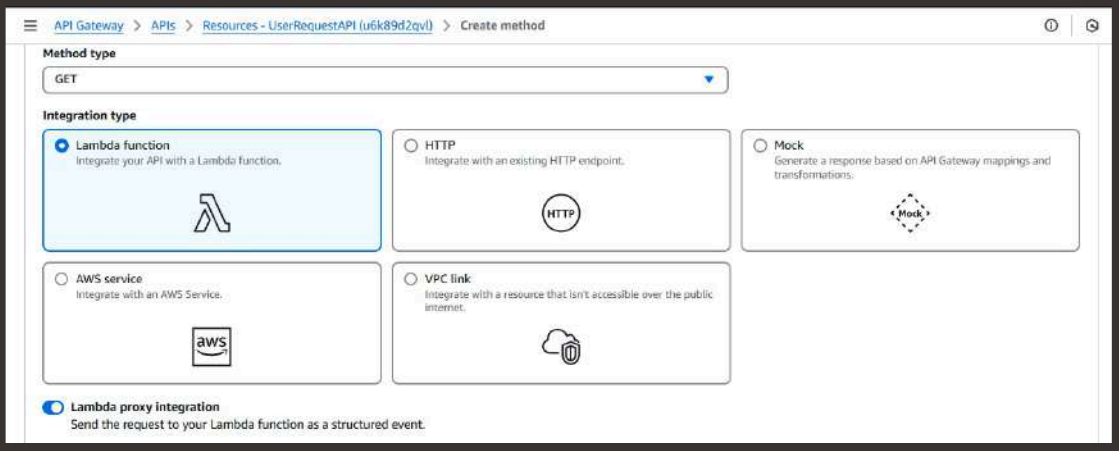
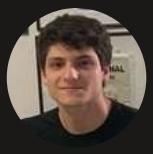
nextwork.org

APIs with Lambda + API Gateway

Nicolás Aversa





Nicolás Aversa
NextWork Student

Introducing Today's Project!

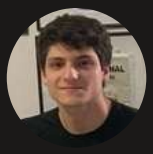
In this project, I will demonstrate how to build a serverless API by developing a Lambda function, configuring an API with API Gateway, and integrating both services to create a fully managed backend. I'm doing this project to learn about serverless architectures, which are really common nowadays.

Tools and concepts

Services I used were Lambda and API Gateway. Key concepts I learnt include Lambda functions, API resources, API types, and how these two services work together to handle requests.

Project reflection

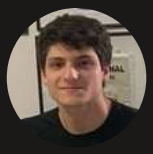
This project took me approximately one hour. The most challenging part was setting up the Lambda function. It was most rewarding to create the GET method to be able to retrieve user data.



Nicolás Aversa
NextWork Student

NextWork.org

I did this project today because I wanted to learn about serverless architecture, and use Lambda and API Gateway, which are currently among the most in-demand serverless technologies, as businesses increasingly adopt cloud-native solutions for their scalability and cost-efficiency. This project met my goals, and I'm excited to continue onto the next part of this project series!

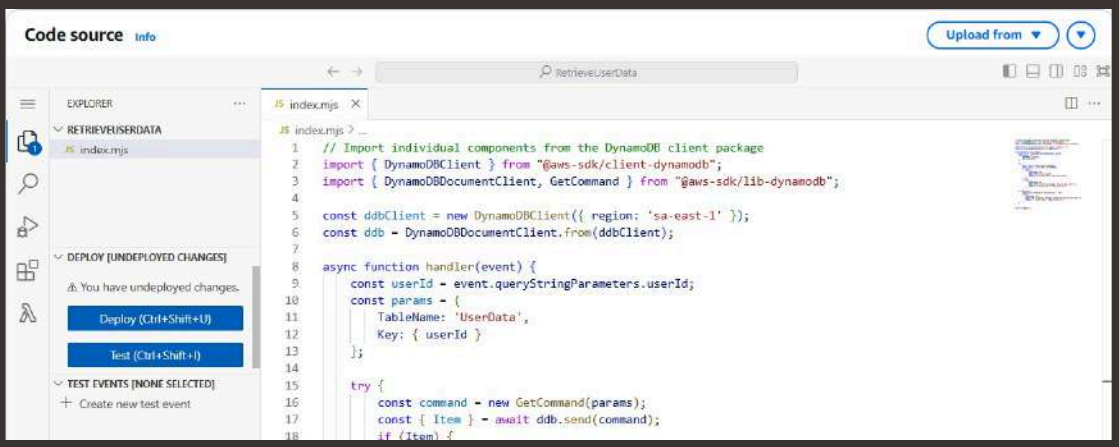


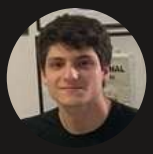
Nicolás Aversa
NextWork Student

Lambda functions

AWS Lambda is a service that lets you run code without needing to manage any computers/servers - Lambda does it for you. I'm using Lambda in this project to create a function to fetch data from a database and return it to the user.

The code I added to my function will look for specific user data based on a 'userId' and return that data. If there's an error e.g. the userId doesn't exist in the database, it returns an error message.





Nicolás Aversa
NextWork Student

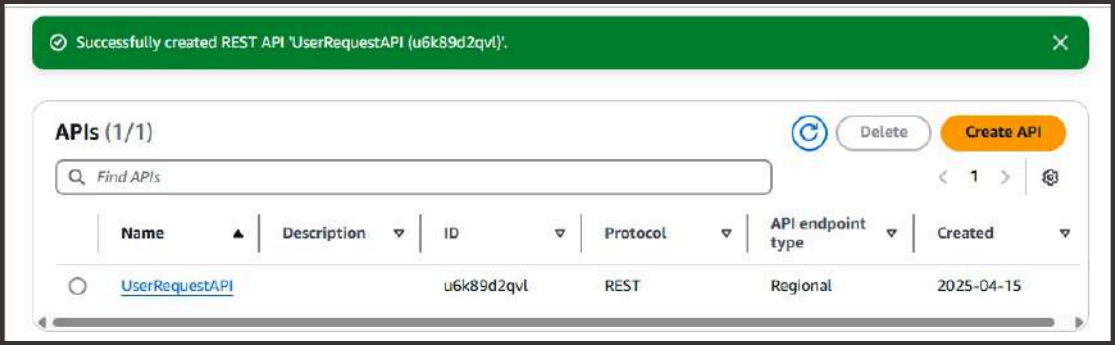
API Gateway

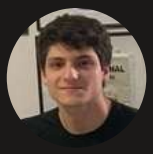
APIs are a way for different software systems to talk to each other. It's like a messenger that carries requests and responses between systems. There are different types of APIs, like REST, HTTP, and WebSocket. My API is a REST API, which uses HTTP methods to interact with resources.

Amazon API Gateway is an AWS service that makes it easy for developers to create, publish, maintain, monitor, and secure APIs at any scale. It manages incoming traffic, directing them to the correct services, and makes sure only authorized requests get through. I'm using API Gateway in this project to forward requests to my Lambda function.

When a user makes a request, API Gateway acts as the "front door" to the Lambda function. It receives requests and then forwards them to Lambda functions for processing. Lambda processes the request, then sends the response through the API Gateway back to the user.

Nicolás Aversa
NextWork Student





Nicolás Aversa
NextWork Student

NextWork.org

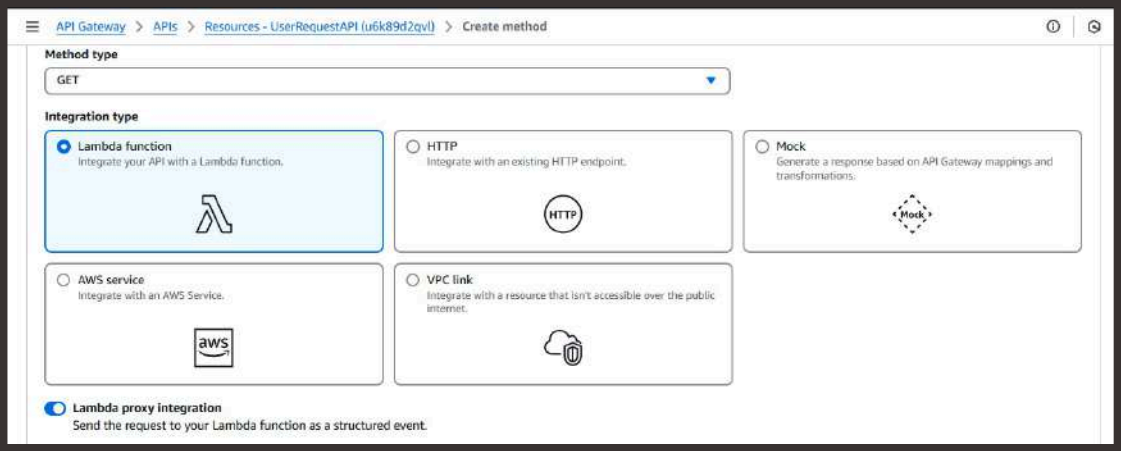
API Resources and Methods

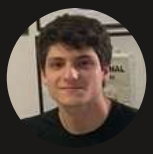
An API is made up of resources, which are individual endpoints within your API that handle different parts of its functionality.

Each resource consists of methods, which are actions you can perform on a resource. They are based on standard HTTP methods, which are different commands that let you interact with data over the internet. For example: GET to retrieve POST to add PUT to update DELETE to remove data

I created a GET method for the /users resource that when it gets called, API Gateway will pass that request to the Lambda function I set up. When the function runs, Lambda will retrieve user data in a DynamoDB table. Finally, Lambda sends the result back to API Gateway and API Gateway then passes that response on to the user.

Nicolás Aversa
NextWork Student



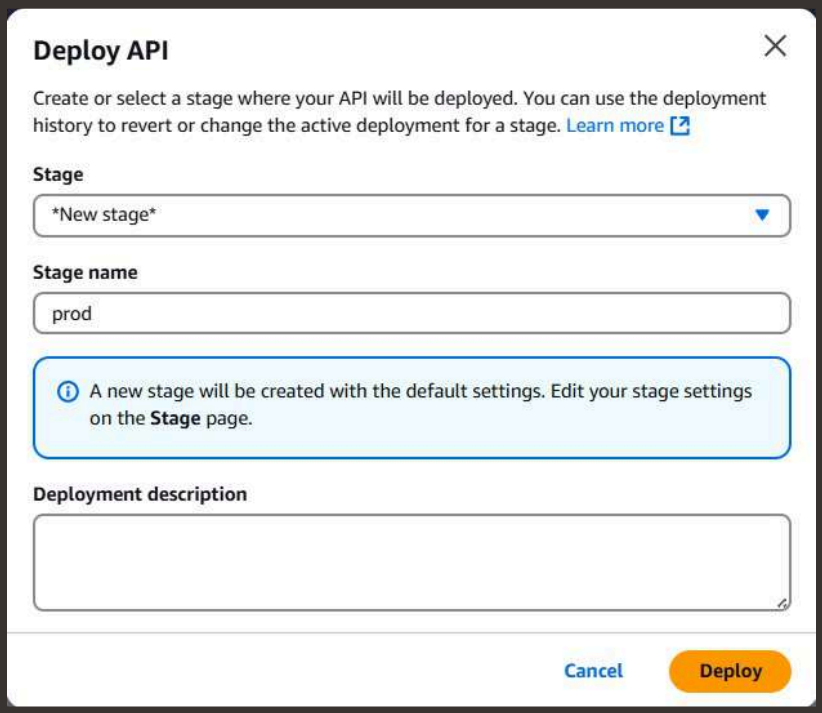


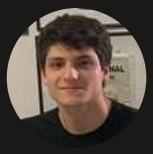
Nicolás Aversa
NextWork Student

API Deployment

When you deploy an API, you deploy it to a specific stage. A stage is a snapshot of your API at a specific point in time. I deployed to prod, which stands for production.

To visit my API, I visited my prod stage API's Invoke URL. The API displayed an error because I haven't set up my DynamoDB table yet. The API set up is all done, and I'll get to creating the DynamoDB table and connecting that with this API in the next project.





Nicolás Aversa
NextWork Student

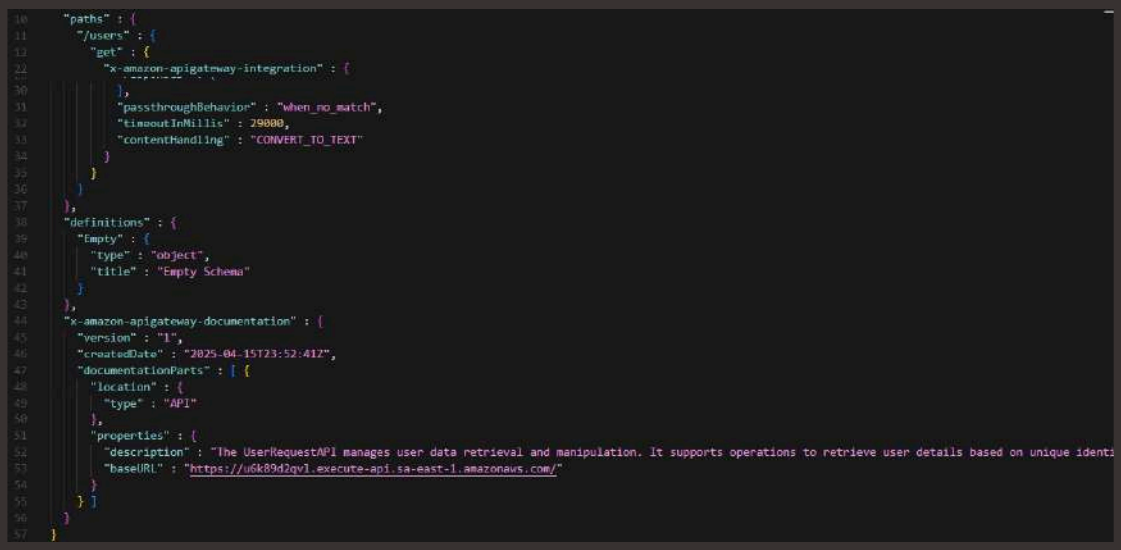
API Documentation

For my project's extension, I am writing API documentation because it provides a detailed description of an API's functionality, including its endpoints (e.g. /users), methods (e.g. GET), parameters (e.g. userId), and responses (e.g. errors or success response). It helps you understand how to use the API correctly and efficiently. You can do this in the Documentation tab of your API.

Once I prepared my documentation, I can publish it to download an updated API definition file that can be used with external tools like Swagger or OpenAPI. You have to publish your API to a specific stage because API Gateway takes a snapshot of the current state and associates it with the selected stage.

My published and downloaded documentation showed me metadata like my API's version and title, resources (/users), and methods (like GET) you can perform on these endpoints.

Nicolás Aversa
NextWork Student



NextWork.org

Everyone should be in a job they love.

Check out nextwork.org for more projects