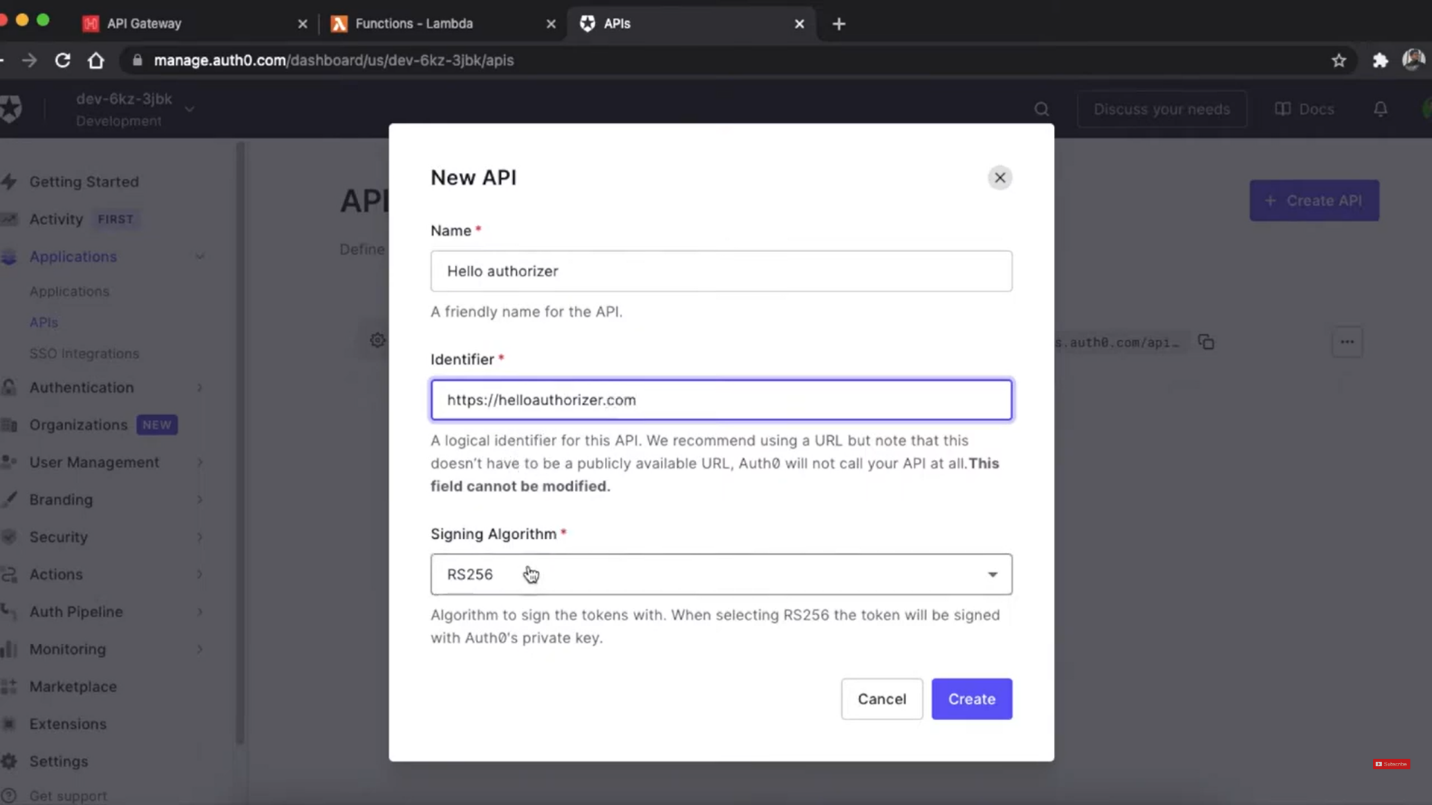
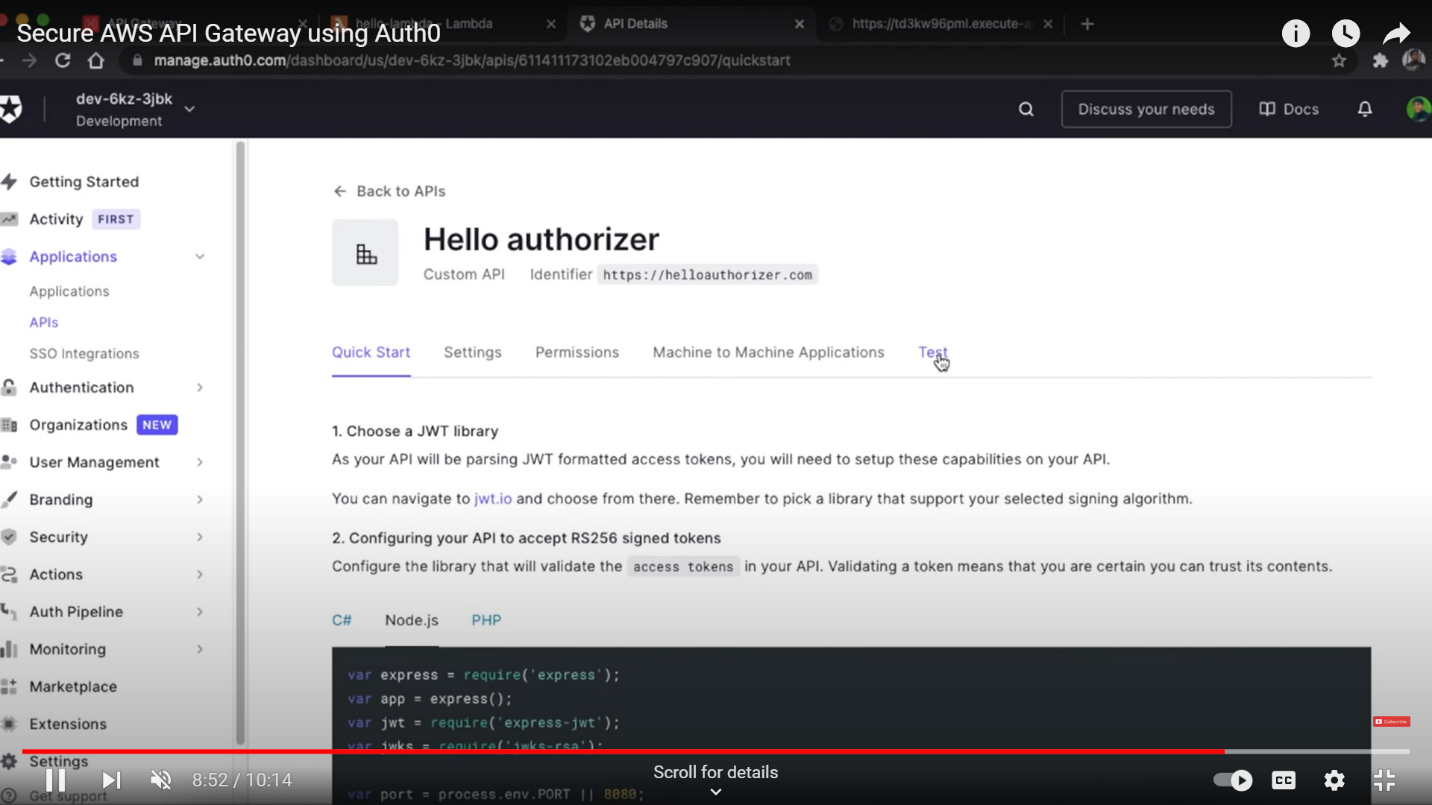
**Securing an AWS API Gateway using Auth0**

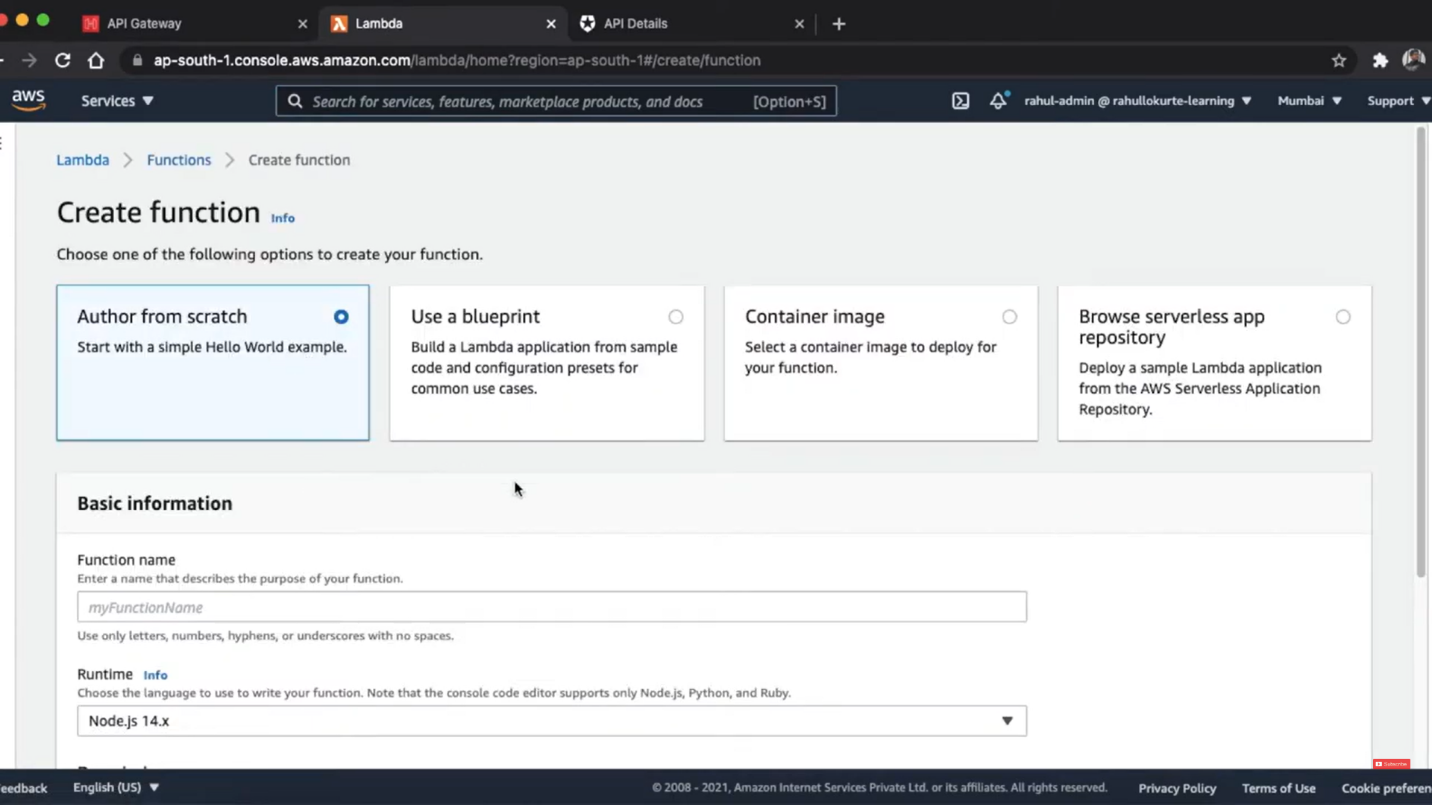
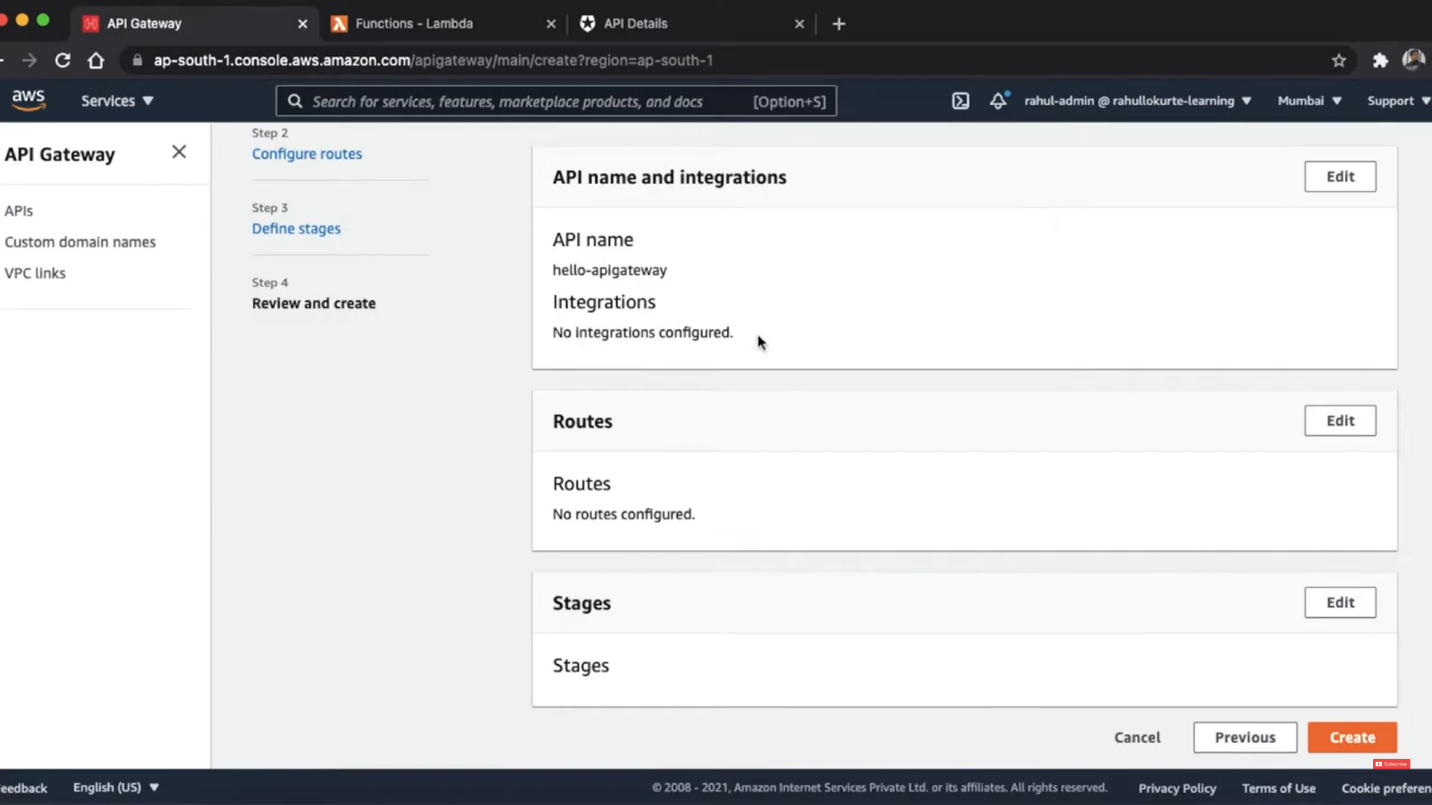
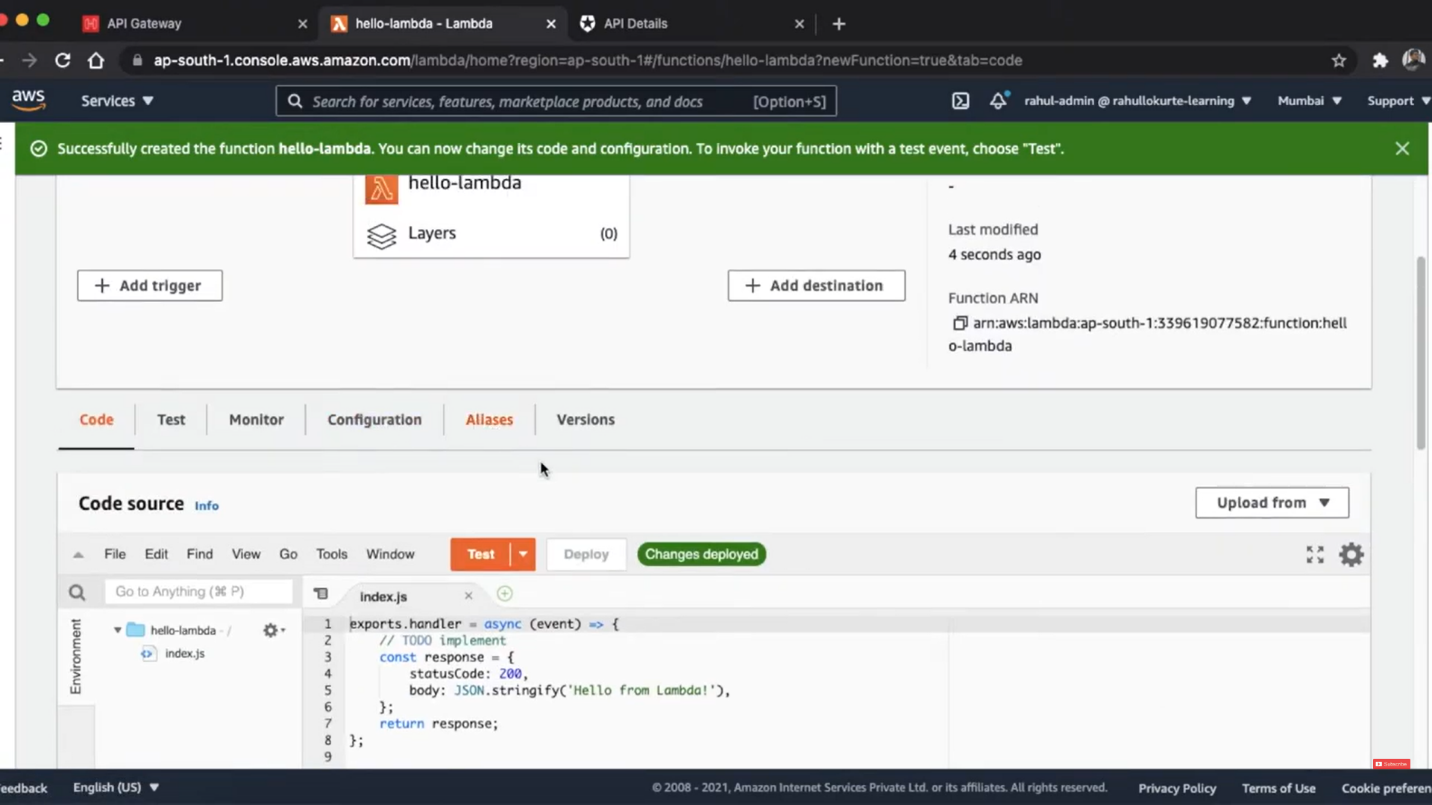
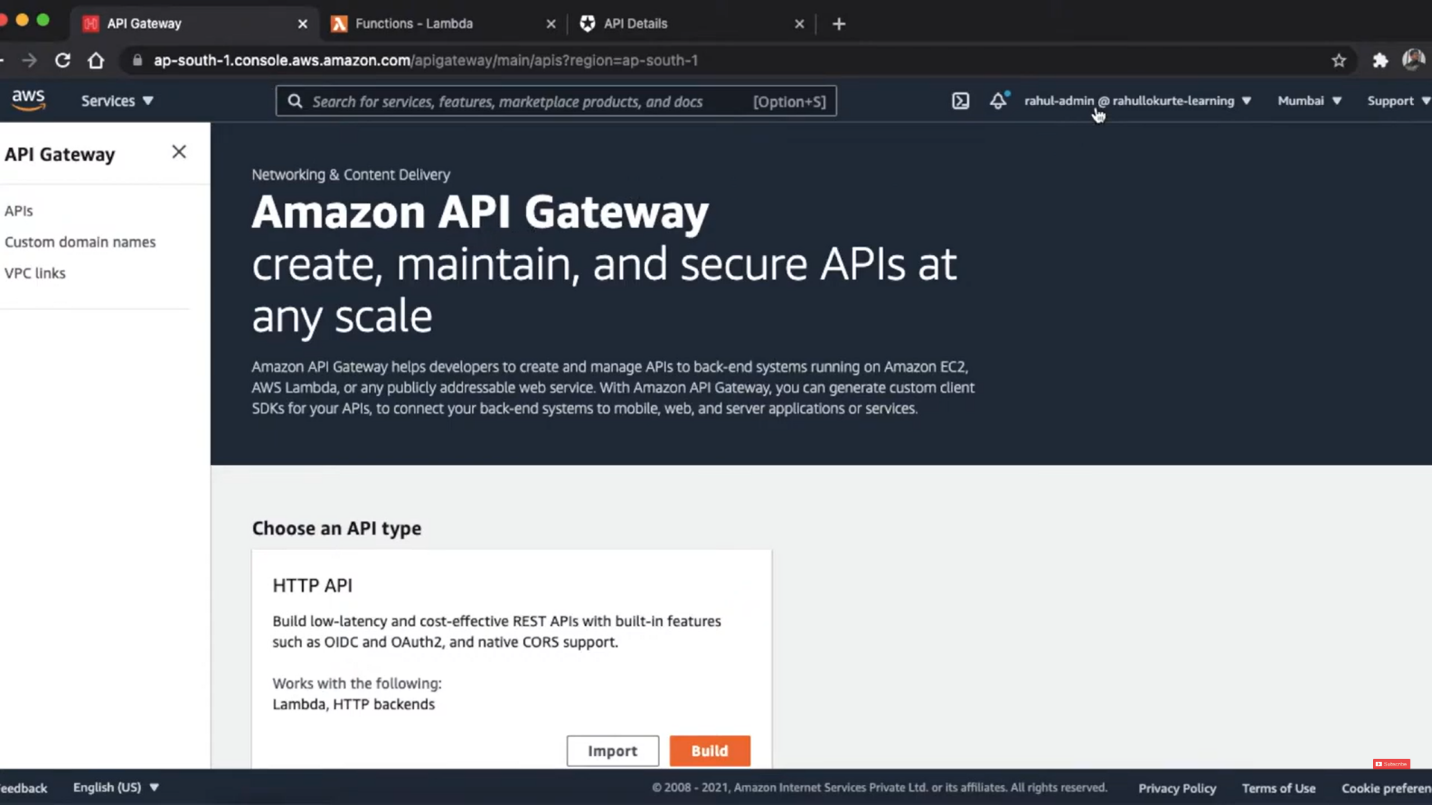
**1. Create an Auth0 Application:**

* Log in to your Auth0 dashboard and create a new Auth0 application. 
* Note down the "Client ID" and "Client Secret" from your Auth0 application settings. 

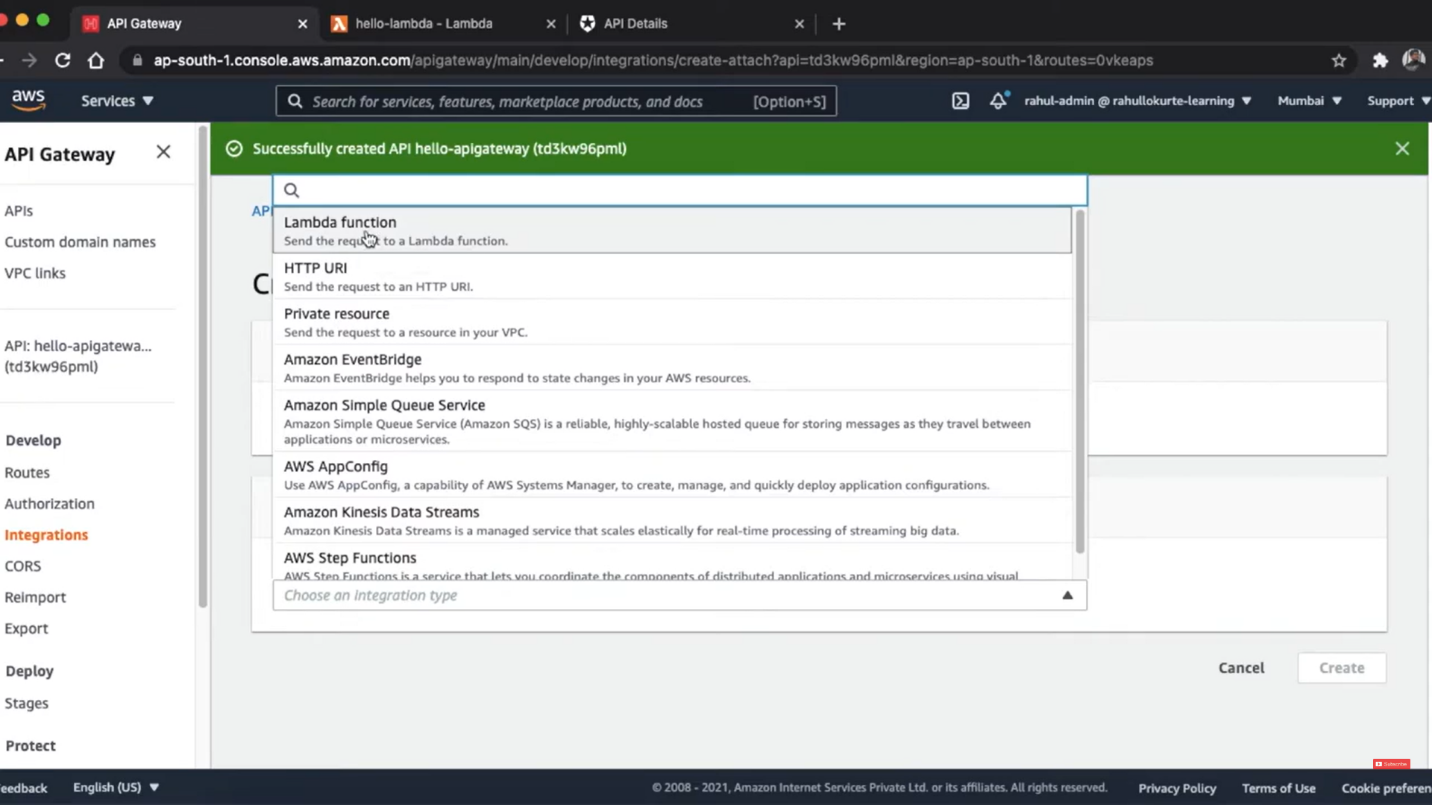
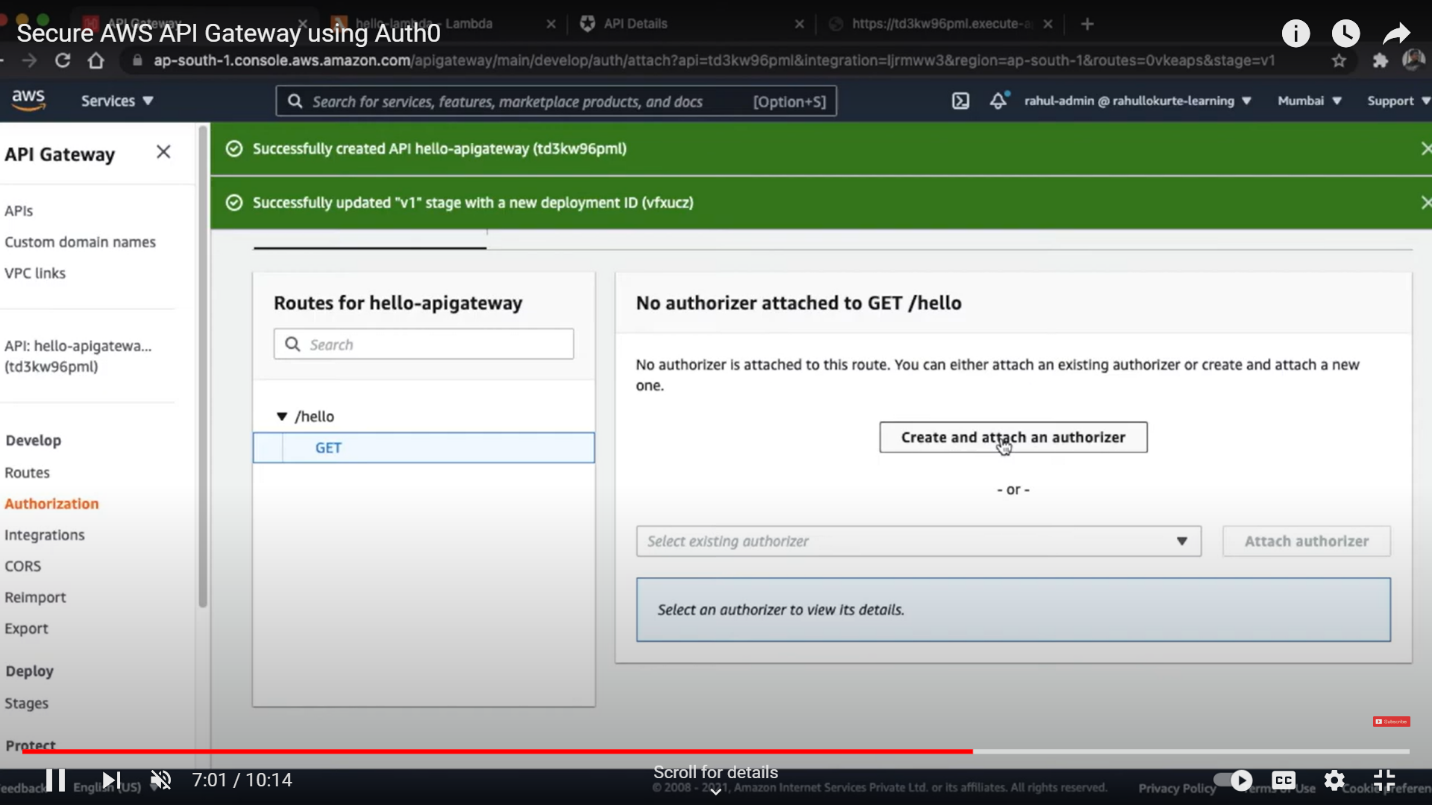
**2. Configure Auth0 for API Authentication:**

* In your Auth0 dashboard, go to the "APIs" section and create a new API.
* Define the "Identifier" for your API, typically in the format `https://your-api-domain.com`.

**3. Set Up Auth0 as an Authorizer in API Gateway:**

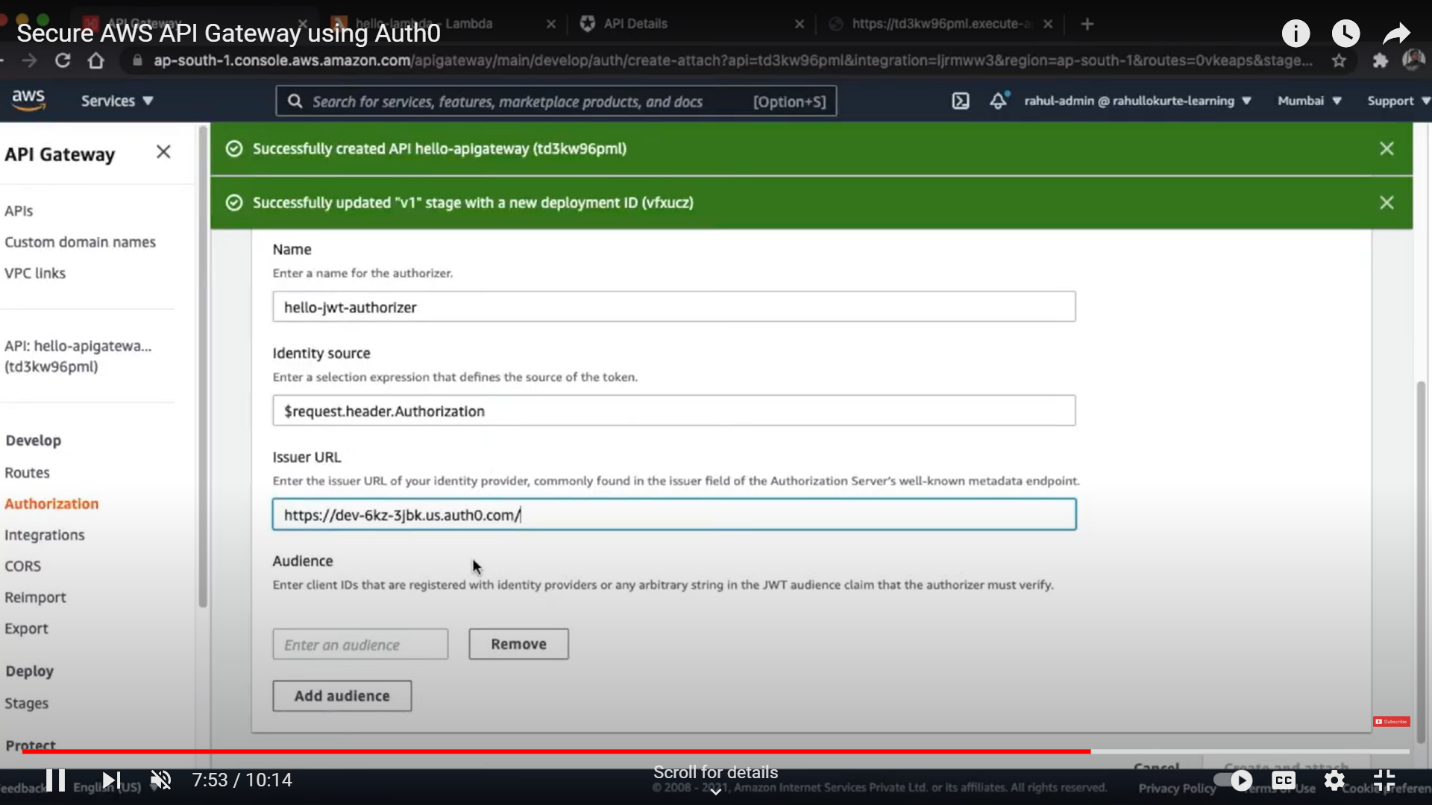
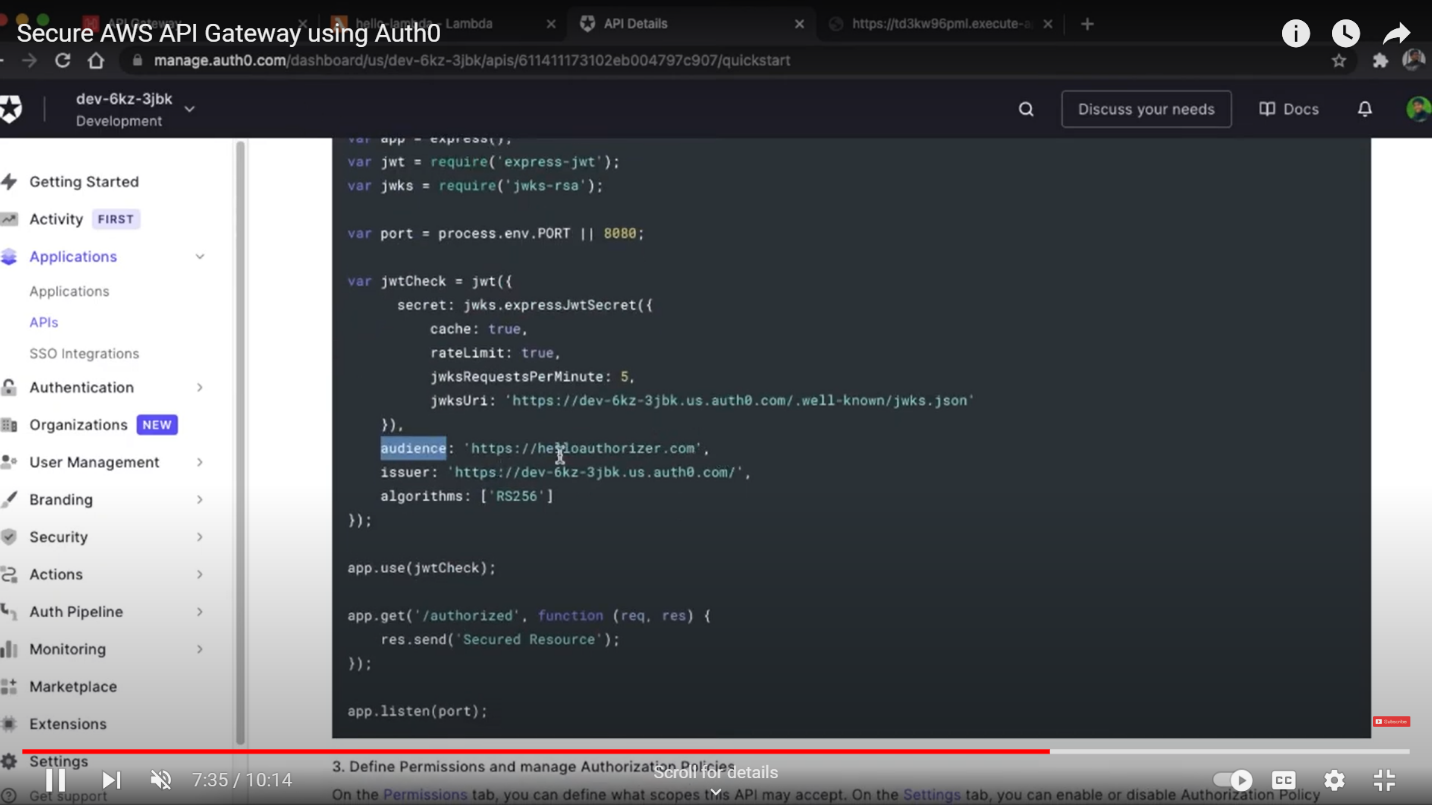
* Open the AWS Management Console and navigate to API Gateway.
* Create or select your API in API Gateway.
* Under your API, go to "Authorizers" and create a new Authorizer.
* Choose "Cognito" as the Authorizer type.
* Provide the Auth0 API Identifier as the "Provider ARN" and configure the other settings. 

**4. Protect API Resources:**

* For each resource or method in your API that you want to protect, set the Auth0 Authorizer as the method's authorizer.
* Configure the Authorization header in your method's integration request to forward the token to your backend. 

**5. Validate JWT Tokens in Your Backend**:

* In your backend code (e.g., Lambda function, serverless function), validate the JWT tokens received in the Authorization header against Auth0.
* Use Auth0 SDKs or manually verify the tokens to ensure their integrity and authenticity.



**6. Handle Token Expiry and Refresh:**

* Implement token expiry handling in your backend. If tokens have expired, you may need to refresh them using Auth0's token refresh endpoint. 