



# Implementing WildRydes

*Using AWS services*



# AWS services in use

- **S3** (hosting the static website, which later becomes dynamic).
- **Cognito** (creating a user pool).
- **DynamoDB** (records unicorn requests).
- **Lambda** (back-end coding for unicorn requests and interacts with front-end app).
- **IAM** (for role permissions).
- **API Gateway** (RESTful API accessible in the public internet).

# Overview

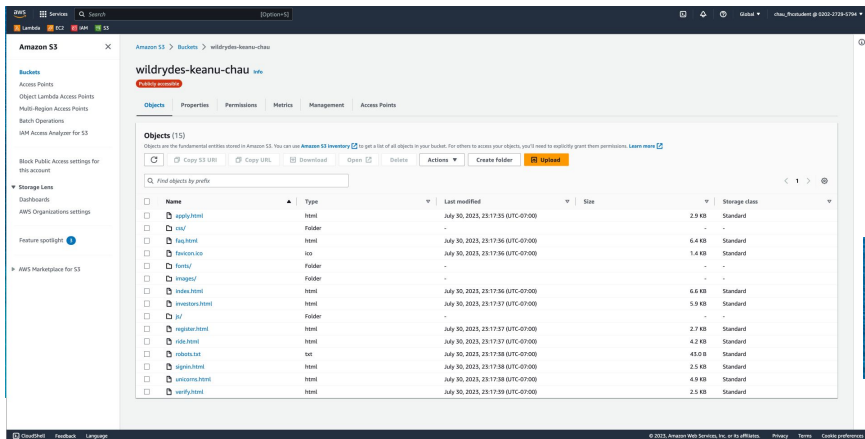
Our **S3** storage will store all our front-end files that users interact with and host a static website that we will later configure to a dynamic one. After that, we create a user pool for customers using **Cognito** that we'll use for our website. We also use **IAM** to create a role that grant access for **Lambda** and **DynamoDB**. When the user logs in, it creates and authentication token that can be authenticated with **API Gateway**. Then **Lambda** and **DynamoDB** handles our requests by **Lambda** providing the back-end code for an API request and **DynamoDB** recording the request for a unicorn. This is all done easily with AWS services, without the need of costly physical hardware.

# S3 Storage

From the talented developing team we got Javascript, HTML, CSS, image, and other files to be able to easily host the static website in **S3**. By this time, all the front-end has been developed.

We create a root domain bucket to store our files, make it publicly accessible, and turn it into a static website. Users will have a public URL that can be modified later using **Route 53**.

We later edit a **config.js** file using **Cognito** and **API Gateway** information that is specific to our user pool and API.



S3 bucket

config.js file edited to contain our user pool and API data

```
window.config = {
  cognito: {
    userPoolId: 'us-west-2_d86hep4gP', // e.g. us-east-2_uXboG5pab
    userPoolClientId: '2con6ge6q6u0vof8ucva59j4p', // e.g. 25ddkmj4v6hfsfvruhpf7n4hv
    region: 'us-west-2' // e.g. us-west-2
  },
  api: {
    invokeUrl: 'https://zvrwpn15g.execute-api.us-west-2.amazonaws.com/prod' // e.g. https://tft5alun5d.execute-api.us-west-2.amazonaws.com/prod
  }
};
```

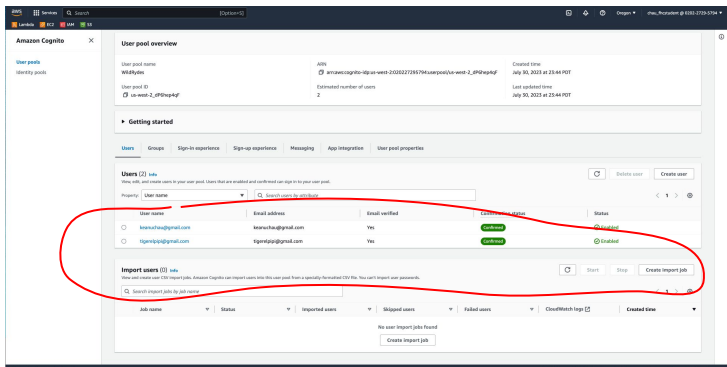


# Amazon Cognito and IAM

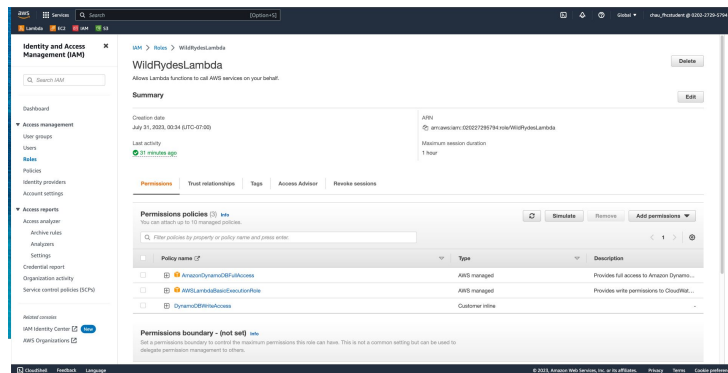
We create a user pool to manage our user's accounts with **Cognito**. We put this information on our **config.js** file in S3. Usual user information and password is provided and then user get sent a verification message. MFA is recommended. When logging in WildRydes, a Javascript function from **S3** talks with **Cognito** and an authorization token is provided and can be tested using **API Gateway**.

**AWS** mainly just charges for monthly active users.

We also use **IAM** to create a role that allows for **Lambda** and **DynamoDB** access.



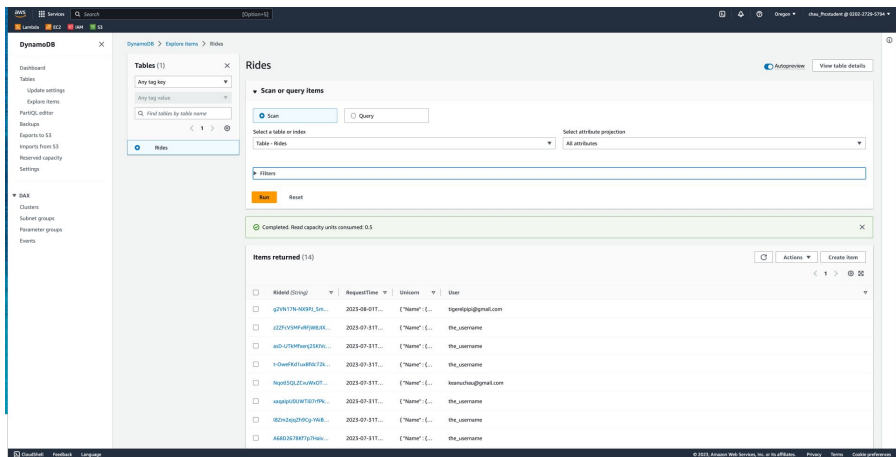
Cognito showing user information



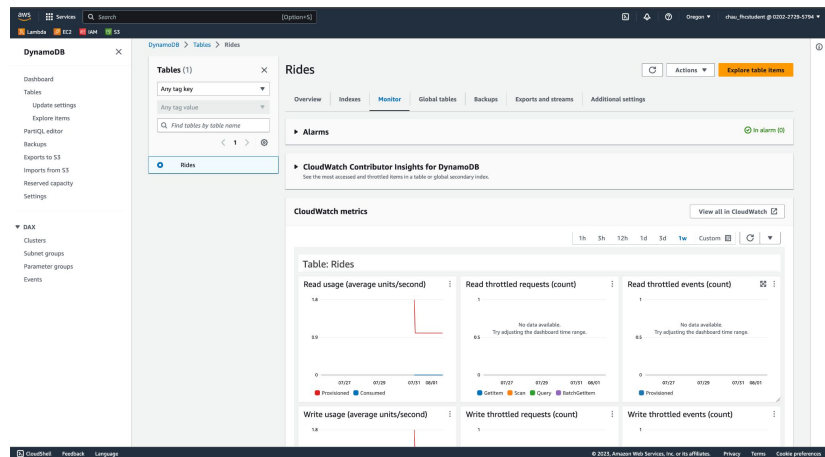
IAM WildRydes role

# DynamoDB

We create a table using **DynamoDB** in order to record unicorn requests. Note that the **IAM** role includes recording requests in this table. When the user logs in and requests a unicorn, the **Lambda** function will record the request in the table.



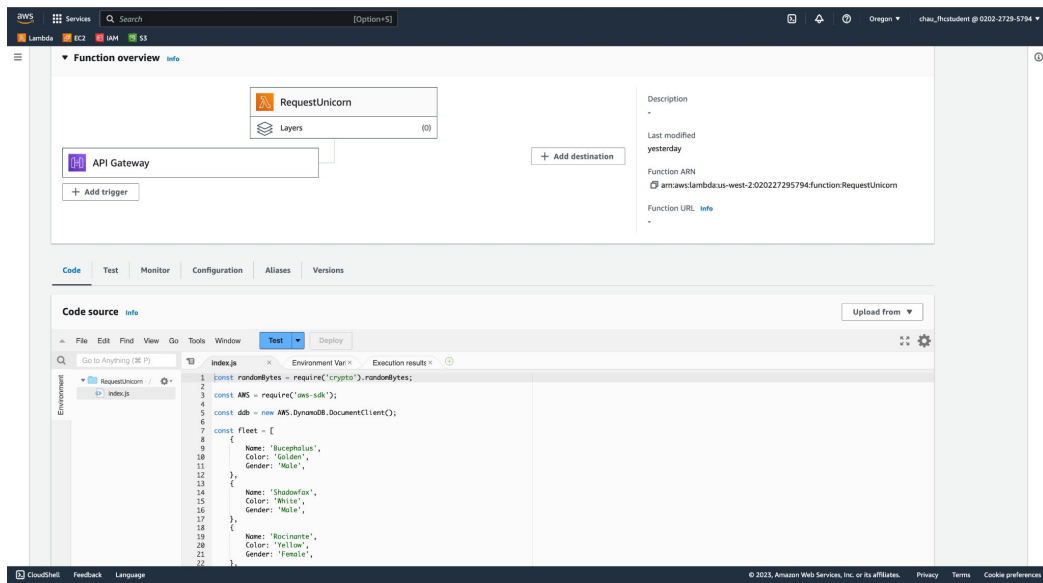
DynamoDB table showing items from requests



DynamoDB showing CloudWatch metric of the table

# Lambda

The function runs in response to HTTP requests (API requests) from the user logging in and requesting a unicorn. It will connect with a RESTful API in **API Gateway** in order to process the request, send a unicorn, and record it in **DynamoDB**. It uses the **IAM** role we created earlier and uses Node.js 16.x for the Runtime.

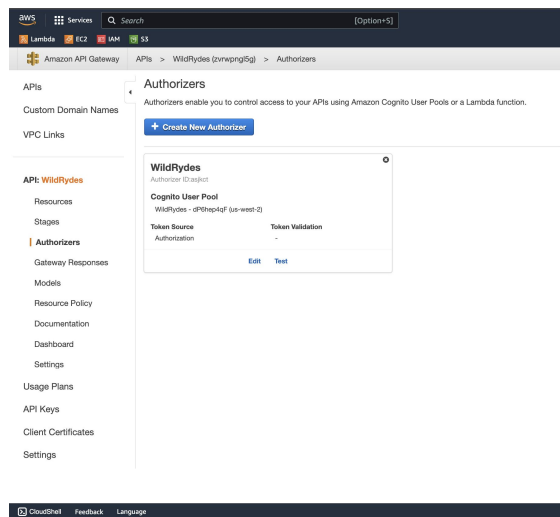


Lambda function showing it is connected with API Gateway

# API Gateway

A RESTful API is needed to connect with the **Lambda** function and is exposed to the public internet user requests. A page we already put out in **S3** interacts with the RESTful API by jQuery's ajax() method to make the remote request which then calls the **Lambda** function which sends the unicorn per the users request.

It uses **Cognito** to authorize the user that is calling the API using authentication tokens.



API Gateway showing that the user pool created in Cognito is being used as an authorizer.

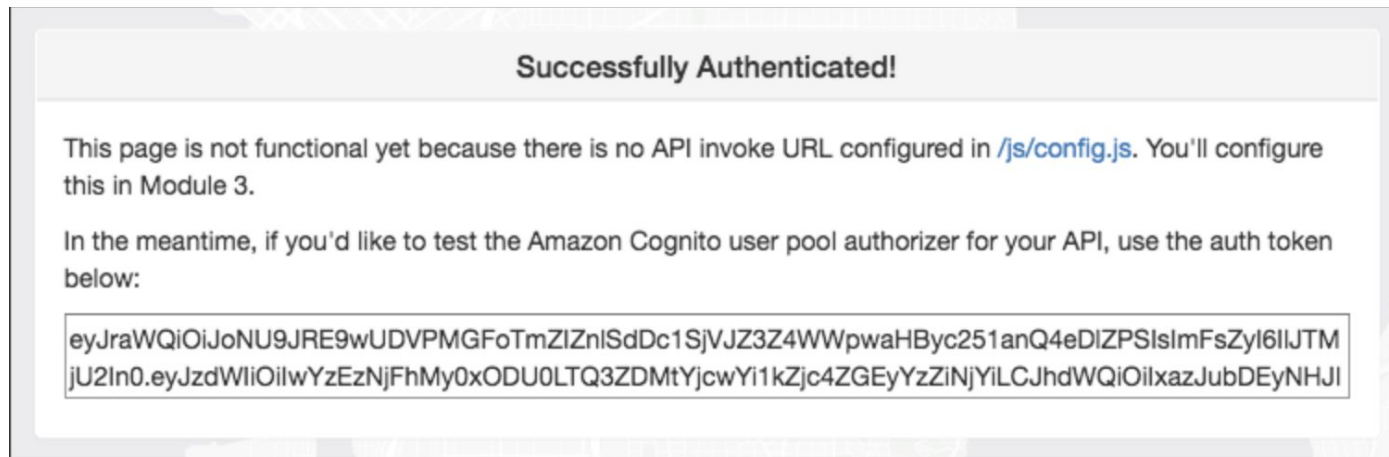


# Testing at every step

**S3:** After uploading all the files from the developing team we tested the now static website by using the provided URL to check if it was active.



**Cognito:** We logged in as a user to test if we got a verification code and see if we could register and then sign in. Verification message was successful.



This should appear after because we have not created our RESTful API yet.

**Lambda:** We configured a test event using Lambda then tested the code we got from our developing team and got correct execution results.

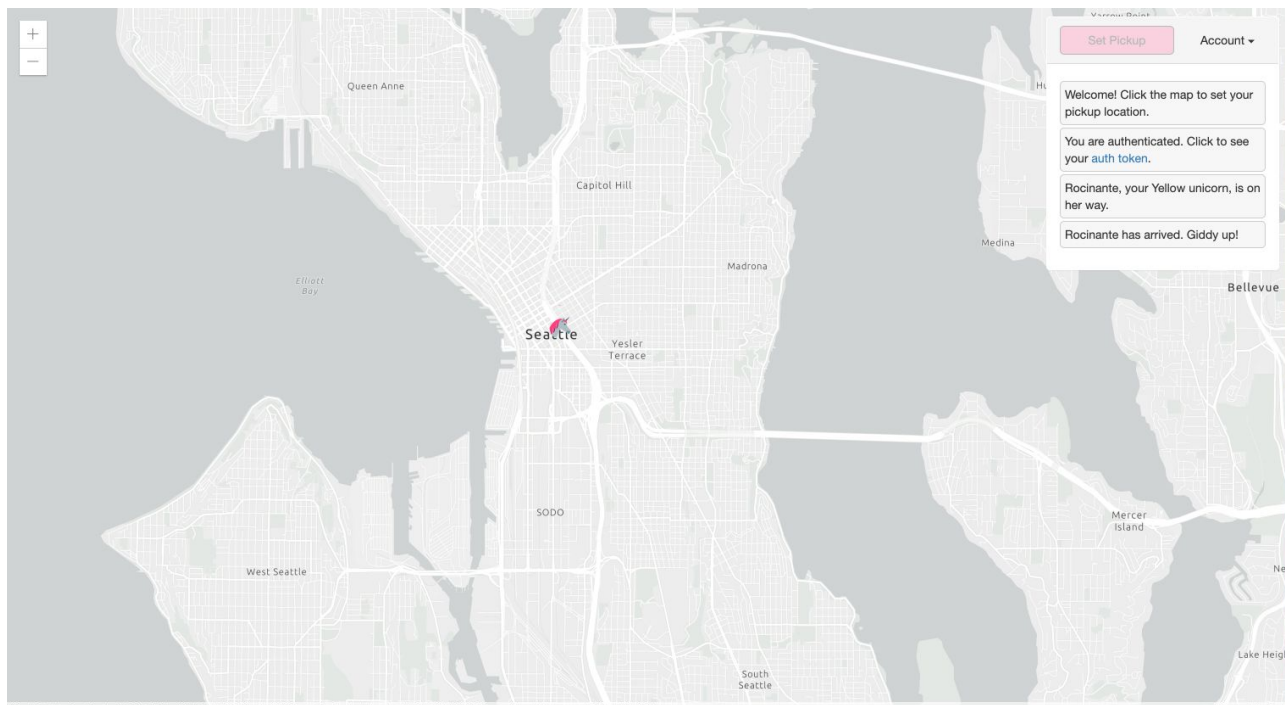
The screenshot displays the AWS Lambda console interface. At the top, there are tabs for 'code source' and 'Info'. Below these is a menu bar with 'File', 'Edit', 'Find', 'View', 'Go', 'Tools', and 'Window'. To the right of the menu bar are 'Test' and 'Deploy' buttons. A search bar labeled 'Go to Anything (⌘ P)' is present. The left sidebar shows a file explorer with a folder 'Request Unicorn' and a file 'index.js'. The main area has tabs for 'index.js', 'Environment Var', and 'Execution result'. The 'Execution result' tab is active, showing the following details:

- Test Event Name:** TestRequestEvent
- Response:**

```
{
  "statusCode": 201,
  "body": "{\"RideId\":\"iHmcgbWfijr1FS36Hlqhgw\",\"Unicorn\":{\"Name\":\"Rocinante\",\"Color\":\"Yellow\",\"Gender\":\"Female\"}\",
  \"headers\": {
    \"Access-Control-Allow-Origin\": \"*\"
  }
}
```
- Function Logs:**

```
START RequestId: f134c655-2ea9-4175-b6a0-1da243a7a4c1 Version: $LATEST
2023-08-02T01:09:13.008Z    f134c655-2ea9-4175-b6a0-1da243a7a4c1    INFO    Received event ( iHmcgbWfijr1FS36Hlqhgw ): {
  path: '/ride',
  httpMethod: 'POST',
  headers: {
    Accept: '*/*',
    Authorization: 'eyJraWQiOiJLTzRVMWZs',
    'content-type': 'application/json; charset=UTF-8'
```

**API Gateway:** With the API ready, testing the whole website is ready since we completed all necessary steps. We request a ride from a location so that it calls the API which runs the **Lambda** function and sends the unicorn. It also then records the request in **DynamoDB**. Website runs successfully.



# Cost Analysis Overview

aws pricing calculator

FeedbackLanguage: English ▼Contact Sales

Estimate summary Info

Upfront cost  
180.00 USD

Monthly cost  
627.83 USD

Total 12 months cost  
**7,713.96 USD**  
Includes upfront cost

Getting Started with AWS  

Get started for free

Contact Sales

Groups Info

My Estimate  
(Total Services: 5)  
▼ WildRydes X

WildRydes

DuplicateDeleteMove toCreate groupAdd supportAdd service

Find resources

	Service Name ▼	Status ▼	Upfront cost ▼	Monthly cost ▼	Description ▼	Region ▼	Config Sum... ▼
<input type="checkbox"/>	Amazon Cognito	-	0.00 USD	500.00 USD	-	US West (Oregon)	Advanced secur...
<input type="checkbox"/>	Amazon Simple...	-	0.00 USD	54.02 USD	-	US West (Oregon)	S3 Standard sto...
<input type="checkbox"/>	AWS Lambda	-	0.00 USD	12.42 USD	-	US West (Oregon)	Architecture (x8...
<input type="checkbox"/>	Amazon Dynam...	-	180.00 USD	26.39 USD	-	US West (Oregon)	Table class (Sta...
<input type="checkbox"/>	Amazon API Ga...	-	0.00 USD	35.00 USD	-	US West (Oregon)	REST API requ...

By service:

Privacy | Site terms | Cookie preferences | © 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Cost Analysis by service

Cognito: 10,000 user estimate

Edit Amazon Cognito [info](#)

Choose a location type [info](#)

Region

Choose a Region

US West (Oregon)

Cognito User Pools [info](#)

The Cognito Your User Pool feature has a free tier of 50,000 MAUs for users who sign in directly to Cognito User Pools and 50 MAUs for users federated through SAML 2.0 based identity providers.

**Number of monthly active users (MAU)**  
A user is counted as a MAU if, within a calendar month, there is an identity operation related to that user, such as sign-up, sign-in, token refresh or password change.

10000

**Advanced security features**  
If you enable advanced security features for Amazon Cognito, additional prices apply for monthly active users. This includes audit mode.

Enabled

**Percent of monthly users who sign in through SAML or OIDC federation**  
Enter the % of users who sign in through SAML or OIDC federation instead of a User Pool or with social identity providers.

0

%

Cognito MAU cost (Monthly): 500.00 USD

Total Upfront cost: 0.00 USD

Total Monthly cost: 500.00 USD

Cancel

Update

S3: 10 million request estimate

Edit Amazon Simple Storage Service (S3) [info](#)

The calculations below exclude Free Tier discounts.

S3 Standard storage

1

Unit

GB per month

**How will data be moved into S3 Standard?**  
Automatically calculates PUT, COPY, POST costs for moving data into S3 Standard initially. To compare the cost of current storage in S3 Standard to lifecycleing this data to another storage class, you can specify that your storage is already stored in S3 Standard while selecting Lifecycle under the new storage class to capture the upfront cost of moving your data.

The specified amount of data is already stored in S3 Standard

**PUT, COPY, POST, LIST requests to S3 Standard**  
Ongoing monthly number of PUT, COPY, POST or LIST requests

10000000

**GET, SELECT, and all other requests from S3 Standard**  
Ongoing monthly number of GET, SELECT and all other requests

10000000

**Data returned by S3 Select**  
Ongoing monthly volume of data returned by S3 Select requests

Value	Unit
-------	------

S3 Standard cost (Monthly): 54.02 USD

Total Upfront cost: 0.00 USD

Total Monthly cost: 54.02 USD

Cancel

Update



# Cost Analysis by service

**Lambda**: 500 ms request duration

**DynamoDB**: 130 byte avg. item size

Edit AWS Lambda [Info](#)

Architecture

x86

Number of requests

10

Unit

million per month

Duration of each request (in ms)

Duration is calculated from the time your code begins executing until it returns or otherwise terminates.

500

Amount of memory allocated

Enter the amount between 128 MB and 10 GB

Value

128

Unit

MB

Amount of ephemeral storage allocated

Enter the amount between 512 MB and 10,240 MB. The first 512 MB are at no additional charge, you only pay for any additional storage that you configure for the function.

Value

512

Unit

MB

Total Upfront cost: 0.00 USD

Total Monthly cost: 12.42 USD

Show Details

Cancel

Update

Edit Amazon DynamoDB [Info](#)

Table class

Select table class

Standard

Data storage

The calculations in this section exclude AWS Free Tier discounts.

Data storage size

1

Unit

GB

Average item size (all attributes)

130

Unit

Byte

Show calculations

Total Upfront cost: 180.00 USD

Total Monthly cost: 26.39 USD

Show Details

Cancel

Update

# Cost Analysis by service

API gateway: RESTful API, 10 million request estimate

## Edit Amazon API Gateway [Info](#)

### REST APIs [Info](#)

Select the units, number, and frequency for REST API requests based on expected volume. The calculations below exclude free tier discounts.

#### REST API request units

The unit will apply to the number of requests you will specify below.

millions

#### Requests

10

#### Unit

per month

#### Cache memory size (GB)

Optional. Choose cache memory size in GB. Dedicated cache memory is billed at an hourly rate.

None

► Show calculations

Total Upfront cost: 0.00 USD

Total Monthly cost: 35.00 USD

Show Details ▼

Cancel

Update