# ReactJS App con Python y AWS Lambda

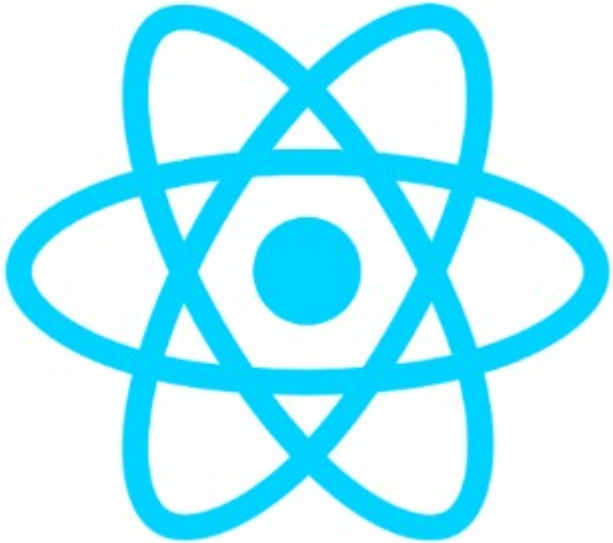
by Luis Mejia

# Agenda

- ReactJS
- Create-react-app
- Construir una ReactJS App
- ReactJS + S3
- API con python y AWS Lambda

# ReactJS

React.js es una librería JavaScript desarrollada por Facebook.



- ¿Para qué nos sirve ?
- ¿Cómo funciona?

# ReactJS: ¿Para qué nos sirve ?

- Facilitar la creación de componentes:
  - Interactivos
  - Rápidos
  - Reutilizables
  - Para interfaces de usuario.

# ReactJS: ¿Cómo funciona?

- Está construido en torno a funciones
- Toma las actualizaciones de estado de la página
- Siempre que React es informado de un cambio de estado, vuelve a ejecutar esas funciones para determinar una nueva representación virtual de la página
- Se traduce automáticamente ese resultado en los cambios del DOM necesarios para reflejar la nueva presentación de la página.

# Create-react-app

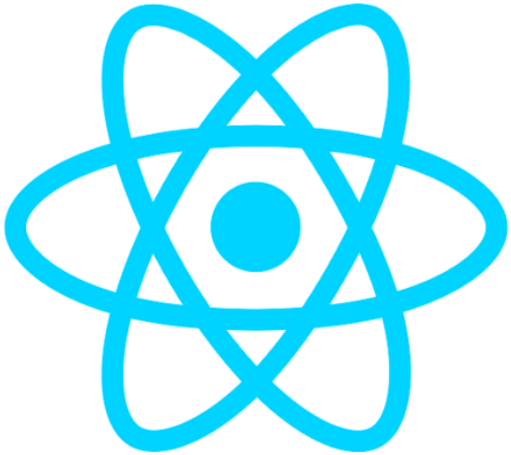
Es una herramienta (creada por desarrolladores en Facebook) que le brinda una ventaja masiva al crear aplicaciones React

- npm install -g create-react-app
- create-react-app test

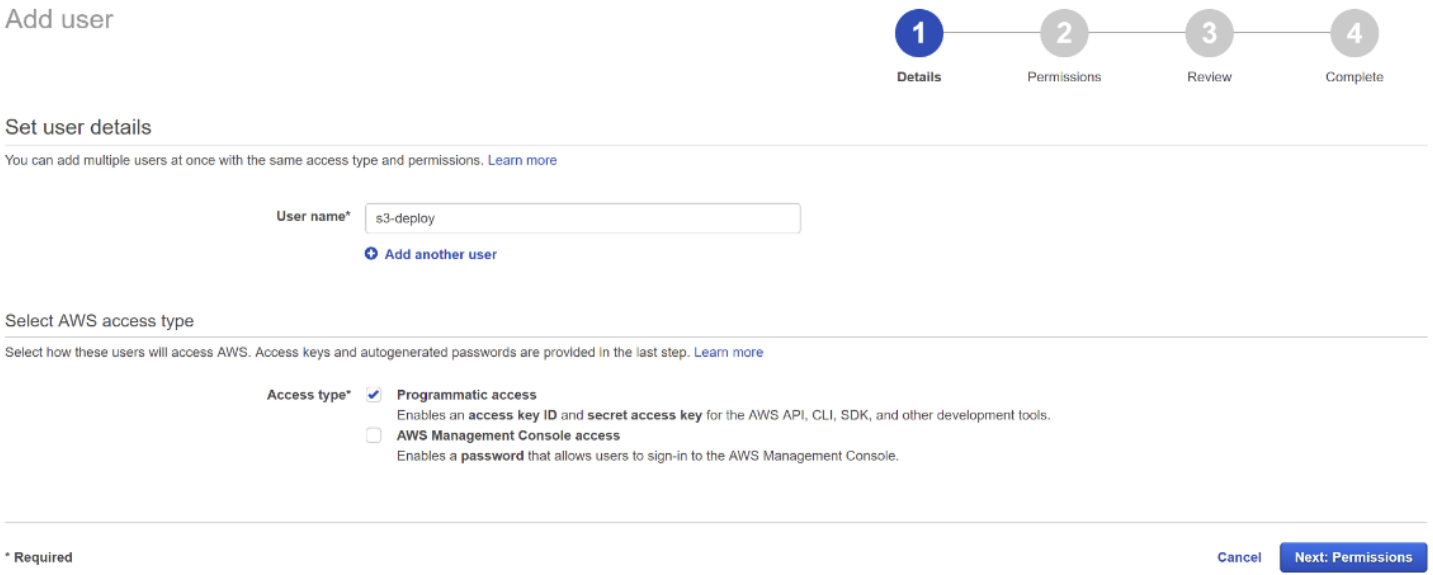
# Construir una ReactJS App

- JSX
- Componentes y Contenedores
- Redux
- API Requests

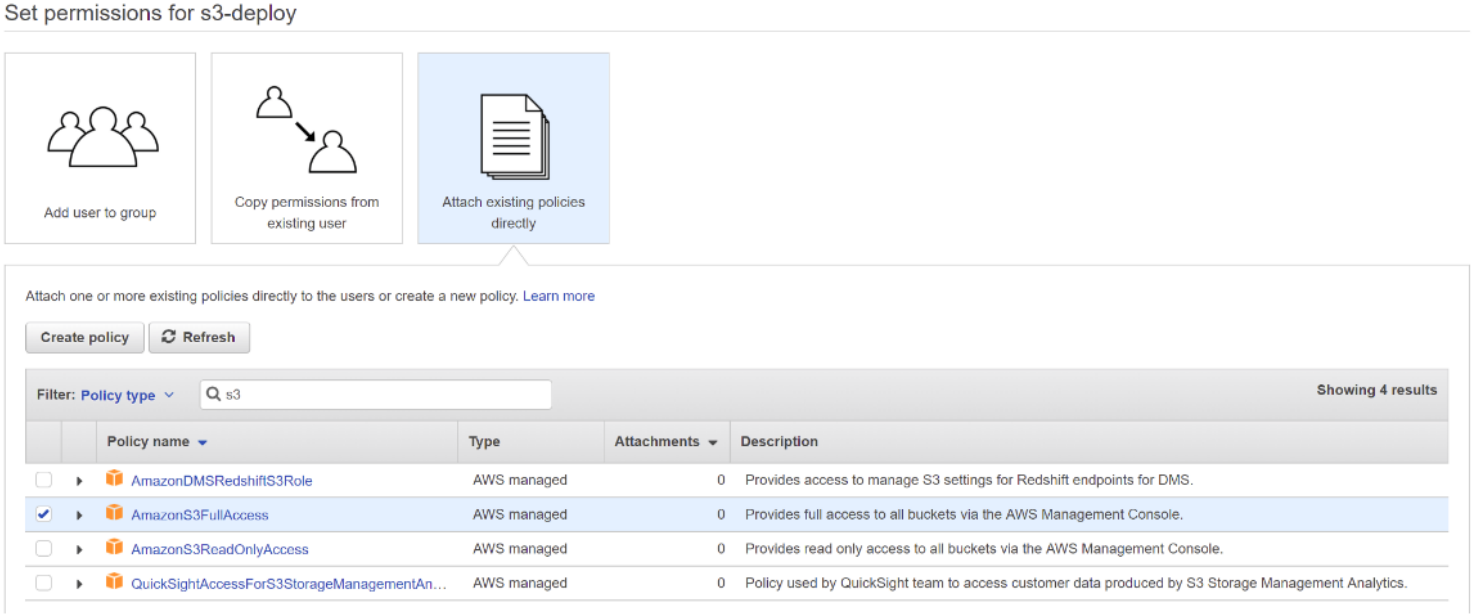
# ReactJS + S3



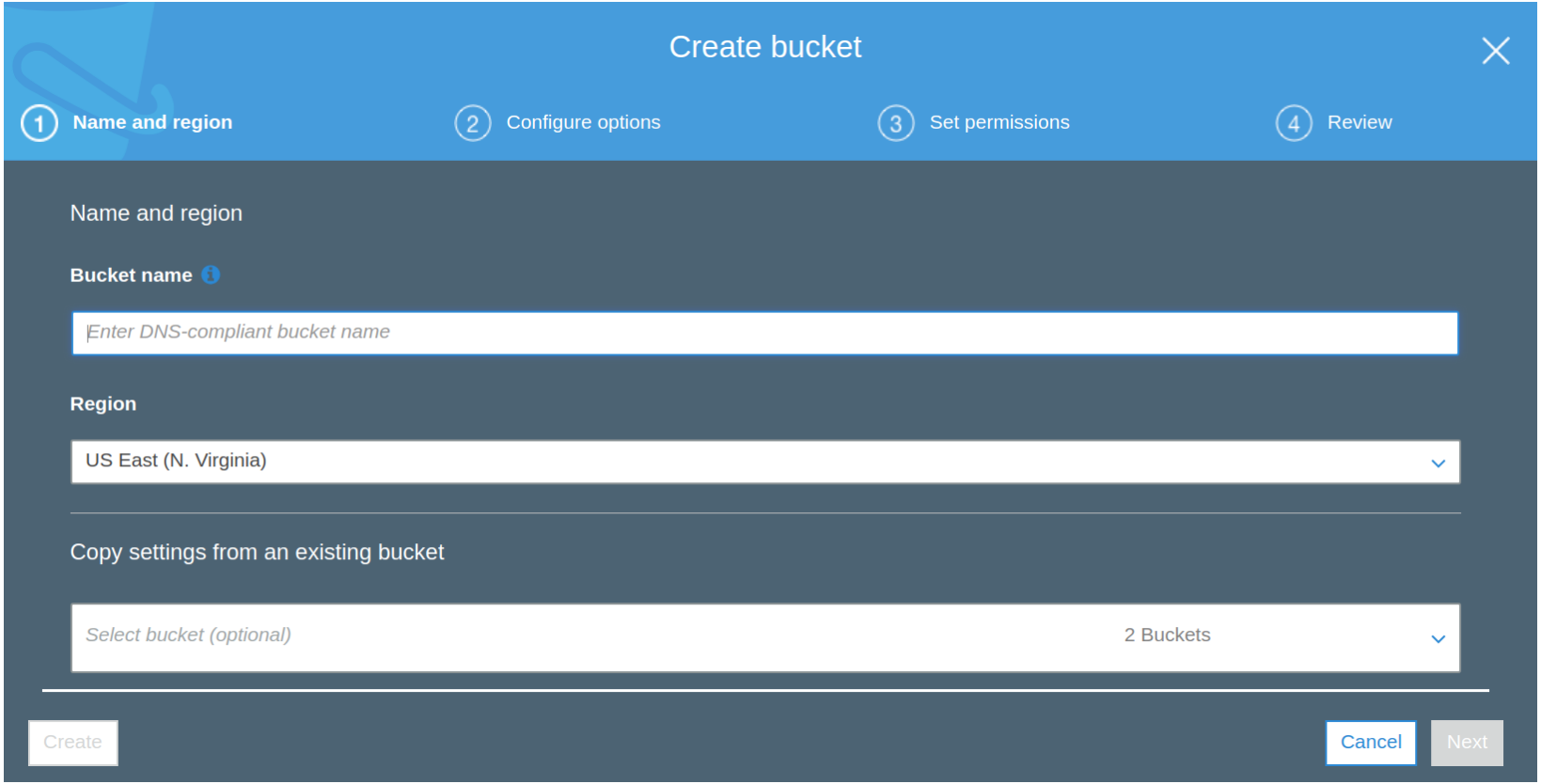
# ReactJS + S3


Add user
1
Details
2
Permissions
3
Review
4
Complete
Set user details
You can add multiple users at once with the same access type and permissions. Learn more
User name*
s3-deploy
Add another user
Select AWS access type
Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. Learn more
Access type*
Programmatic access
Enables an access key ID and secret access key for the AWS API, CLI, SDK, and other development tools.
AWS Management Console access
Enables a password that allows users to sign-in to the AWS Management Console.
* Required
Cancel
Next: Permissions

# ReactJS + S3


Set permissions for s3-deploy
Add user to group
Copy permissions from existing user
Attach existing policies directly
Attach one or more existing policies directly to the users or create a new policy. Learn more
Create policy
Refresh
Filter: Policy type
s3
Showing 4 results
Policy name
Type
Attachments
Description
AmazonDMSRedshiftS3Role
AWS managed
0
Provides access to manage S3 settings for Redshift endpoints for DMS.
AmazonS3FullAccess
AWS managed
0
Provides full access to all buckets via the AWS Management Console.
AmazonS3ReadOnlyAccess
AWS managed
0
Provides read only access to all buckets via the AWS Management Console.
QuickSightAccessForS3StorageManagementAn...
AWS managed
0
Policy used by QuickSight team to access customer data produced by S3 Storage Management Analytics.

# ReactJS + S3


Create bucket
1 Name and region
2 Configure options
3 Set permissions
4 Review
Name and region
Bucket name
Enter DNS-compliant bucket name
Region
US East (N. Virginia)
Copy settings from an existing bucket
Select bucket (optional)
2 Buckets
Create
Cancel
Next

# ReactJS + S3

## AWS Policy Generator

The AWS Policy Generator is a tool that enables you to create policies that control access to Amazon Web Services (AWS) products and resources. For more information about creating policies, see key concepts in Using AWS Identity and Access Management. Here are sample policies.

### Step 1: Select Policy Type

A Policy is a container for permissions. The different types of policies you can create are an IAM Policy, an S3 Bucket Policy, an SNS Topic Policy, a VPC Endpoint Policy, and an SQS Queue Policy.

**Select Type of Policy** S3 Bucket Policy

### Step 2: Add Statement(s)

A statement is the formal description of a single permission. See a description of elements that you can use in statements.

**Effect** Allow Deny

**Principal** *

Use a comma to separate multiple values.

**AWS Service** Amazon S3 All Services ('*')

Use multiple statements to add permissions for more than one service.

**Actions** 1 Action(s) Selected All Actions ('*')

**Amazon Resource Name (ARN)** arn:aws:s3:::com.luismejia.propublica/*

ARN should follow the following format: arn:aws:s3:::<bucket_name>/<key_name>.
Use a comma to separate multiple values.

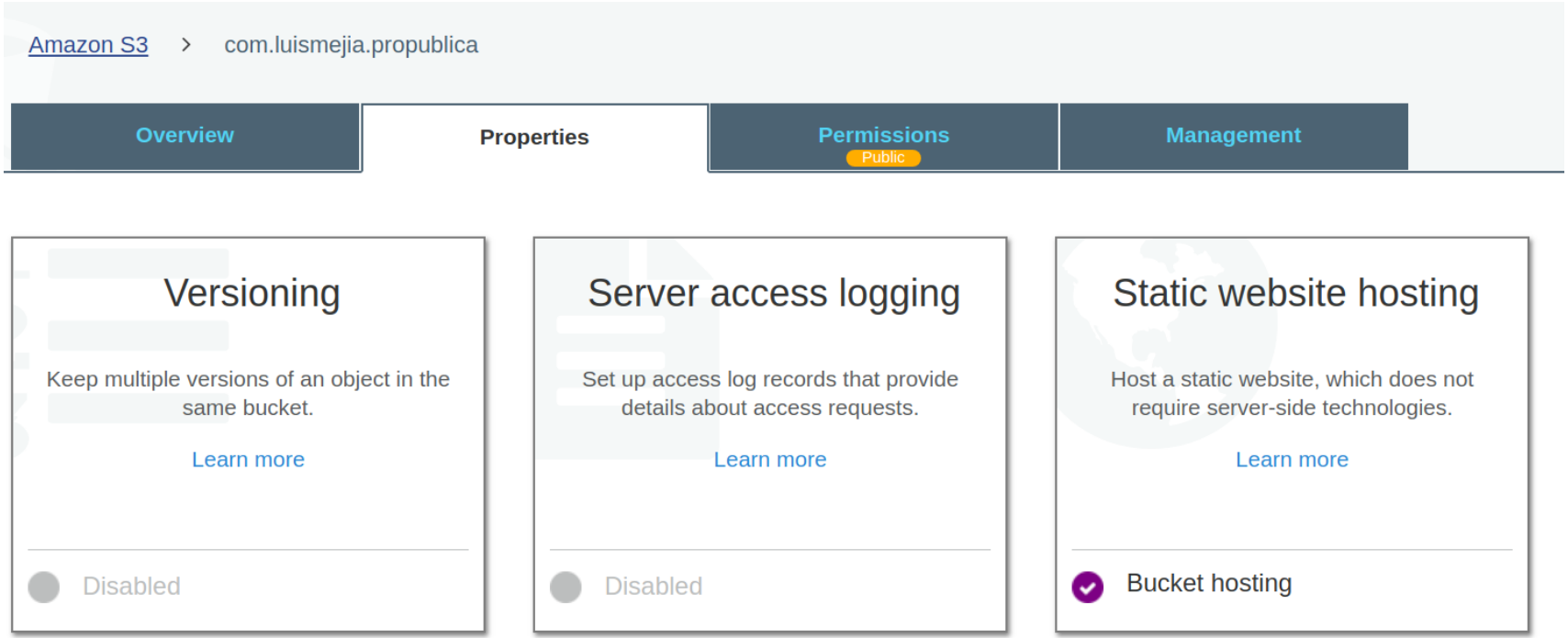
Add Conditions (Optional)

**Add Statement**

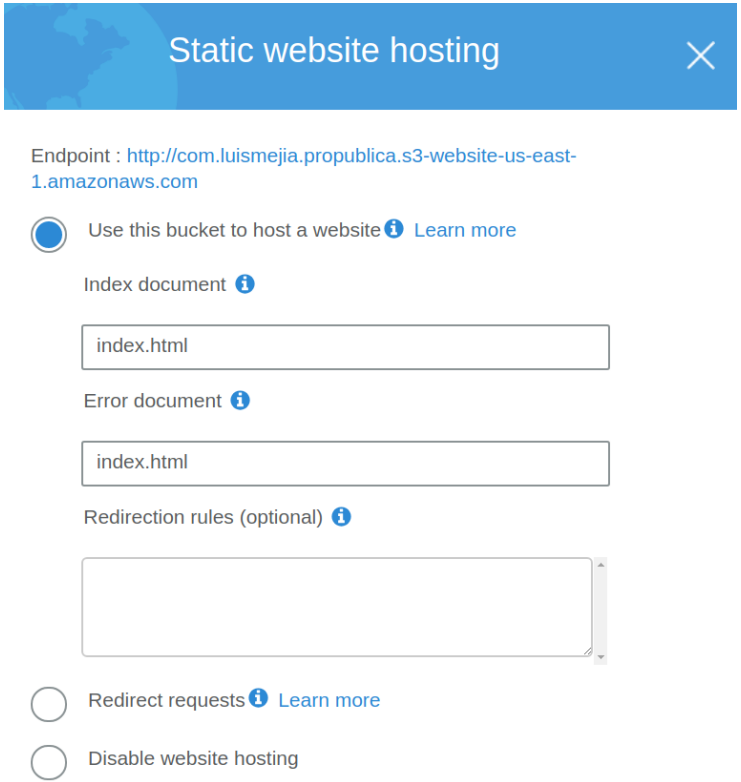
# ReactJS + S3


Amazon S3 > com.luismejia.propublica
Overview
Properties
Permissions
Public
Management
Public access settings
Access Control List
Bucket Policy
Public
CORS configuration
Bucket policy editor ARN: arn:aws:s3:::com.luismejia.propublica
Type to add a new policy or edit an existing policy in the text area below.
Delete
Cancel
Save
{
"Version": "2012-10-17",
"Id": "Policy1552627269938",
"Statement": [
{
"Sid": "Stmt1552627238432",
"Effect": "Allow",
"Principal": "*",
"Action": "s3:GetObject",
"Resource": "arn:aws:s3:::com.luismejia.propublica/*"
}
]
}

# ReactJS + S3


Amazon S3
com.luismejia.propublica
Overview
Properties
Permissions
Public
Management
Versioning
Keep multiple versions of an object in the same bucket.
Learn more
Disabled
Server access logging
Set up access log records that provide details about access requests.
Learn more
Disabled
Static website hosting
Host a static website, which does not require server-side technologies.
Learn more
Bucket hosting

# ReactJS + S3


Static website hosting
Endpoint : http://com.luismejia.propublica.s3-website-us-east-1.amazonaws.com
Use this bucket to host a website
Learn more
Index document
index.html
Error document
index.html
Redirection rules (optional)
Redirect requests
Learn more
Disable website hosting

# ReactJS + S3

```
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "deploy": "aws s3 sync build/ s3://com.luismejia.propublica",
    "test": "react-scripts test --env=jsdom",
    "eject": "react-scripts eject"
  },
```

# ReactJS + S3

## Deploy to S3

```
aws configure
npm run build
npm run deploy
```

# API con python y AWS Lambda

```
mkdir ~/my-serverless-project
cd ~/my-serverless-project
sls create -n my-serverless-project -t aws-python3

sls deploy
```