



**Clase 34.** Programación Backend

# ***Productos Cloud: AWS***



## ***OBJETIVOS DE LA CLASE***

- Presentar AWS (Amazon Web Services) y crear una cuenta.
- Recorrer e implementar a partir de la plataforma Elastic Beanstalk de AWS.
- Desarrollar un proyecto de Node en la plataforma de AWS.

# ***CRONOGRAMA DEL CURSO***

Clase 33



Productos cloud: Heroku

Clase 34



Productos cloud: AWS

Clase 35



Envío de mensajes y  
Seguridad

# **PLATAFORMA AWS**



# *¿Qué es?*

- Amazon Web Services (AWS) es una plataforma en la nube muy adoptada y completa.
- Ofrece más de 200 servicios integrales de centros de datos a nivel global. Muchas empresas lo utilizan y con esto reducen los costos, aumentan su agilidad e innovan de forma más rápida.



En esta clase utilizaremos la **plataforma *Elastic Beanstalk*** de AWS para implementar aplicaciones NodeJS en la nube.

👉 Ten en cuenta que al crearnos una cuenta en AWS nos piden una tarjeta de crédito. Sin embargo, si hacemos las pruebas y luego lo borramos será sin cargo.



Para crear una cuenta en AWS, la plataforma solicitará los datos de una tarjeta de crédito.

Si hacemos las pruebas y luego lo borramos no generará cargos. De todos modos, a fines de este, curso *no será obligatorio* que lo hagas.

☞ Si decides no hacerlo, presta atención al proceso porque puede resultarte sumamente enriquecedor a futuro.



# Plataforma Elastic Beanstalk



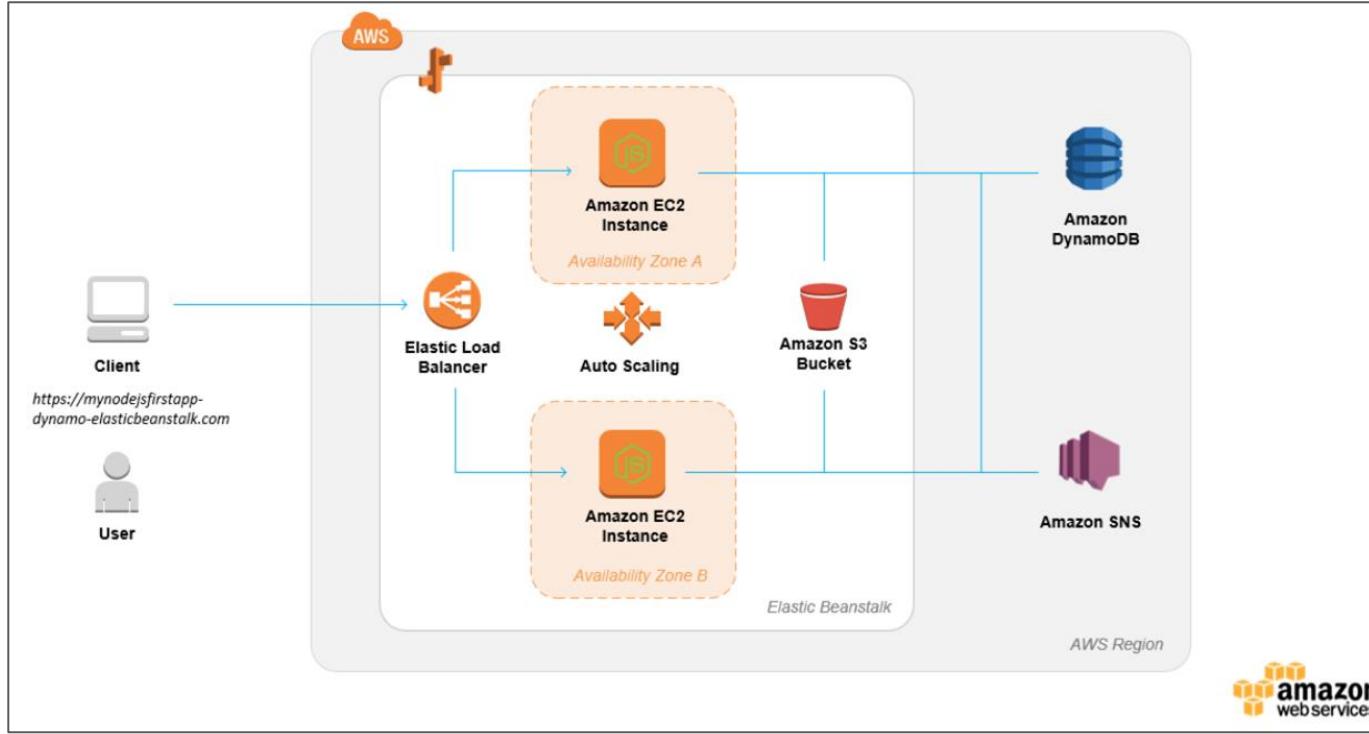
- Administra de manera automática la implementación de nuestra aplicación (desde el aprovisionamiento de la capacidad, el balanceo de carga y el auto escalamiento hasta la monitorización del estado) ingresando únicamente el código.
- Ajusta el escalado de la aplicación automáticamente en función de las necesidades específicas de las aplicaciones. Para ello utiliza una configuración de *Auto Scaling* que se puede adaptar con facilidad.



Usaremos también **Amazon DynamoDB** que es un servicio de base de datos NoSQL rápido y flexible, completamente administrado en la nube, compatible con modelos de almacenamiento de valor de clave y de documentos.



# Diagrama de arquitectura de Elastic Beanstalk



CODER HOUSE

# ***CREAR NUESTRA CUENTA EN AWS***

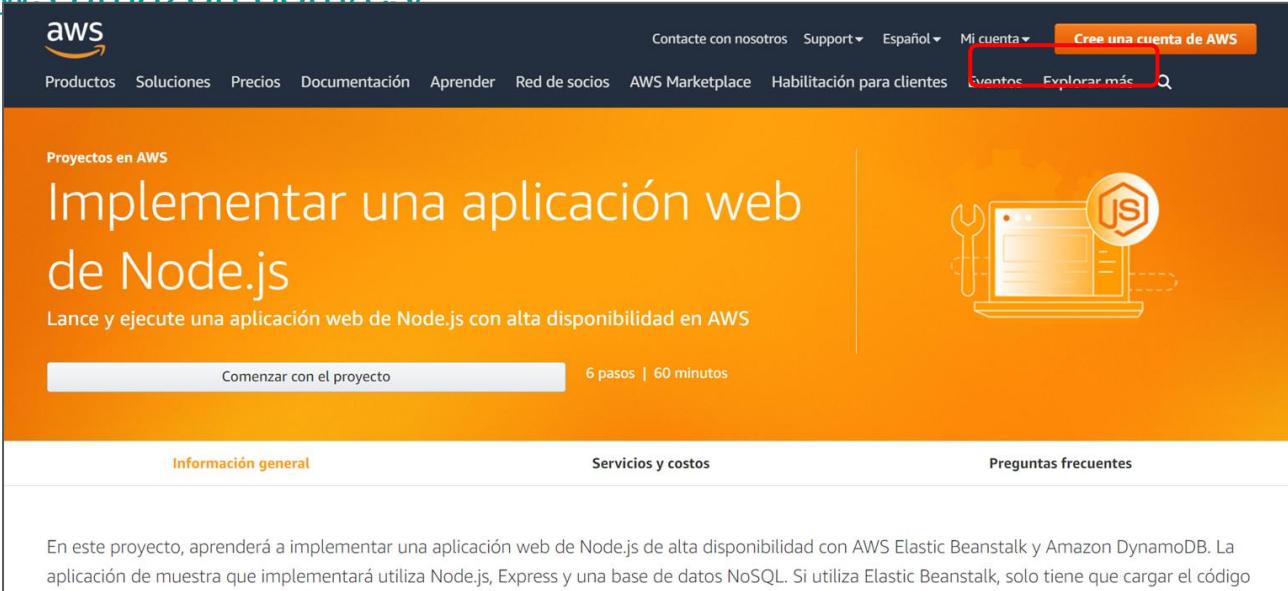


# Crear cuenta en AWS

Ejemplo  
en vivo



- Para crear una cuenta en AWS entramos a  
<https://aws.amazon.com/es/>



The screenshot shows the AWS homepage in Spanish. At the top right, there is a red box highlighting the "Cree una cuenta de AWS" (Create an AWS account) button. Below the header, there is a large orange banner for a project titled "Implementar una aplicación web de Node.js". The banner includes a "Comenzar con el proyecto" (Start with the project) button and a "6 pasos | 60 minutos" (6 steps | 60 minutes) link. At the bottom of the banner, it says: "En este proyecto, aprenderá a implementar una aplicación web de Node.js de alta disponibilidad con AWS Elastic Beanstalk y Amazon DynamoDB. La aplicación de muestra que implementará utilizará Node.js, Express y una base de datos NoSQL. Si utiliza Elastic Beanstalk, solo tiene que cargar el código".



# Crear cuenta en AWS

Ejemplo  
en vivo



- Completamos el mail, contraseña y elegimos el nombre de la cuenta. Luego continuamos al siguiente paso.

The screenshot shows the AWS sign-up process. At the top, there's a large 'aws' logo. Below it, a promotional message encourages users to explore AWS products with a free account. It includes a link to [aws.amazon.com/free](http://aws.amazon.com/free) and an illustration of a hand holding three 3D cubes. The main form is titled 'Registrarse en AWS'. It contains four input fields: 'Dirección de correo electrónico', 'Contraseña', 'Confirmar la contraseña', and 'Nombre de la cuenta de AWS'. A red box highlights the 'Continuar (paso 1 de 5)' button. Below the form, a link says 'Iniciar sesión en una cuenta de AWS existente'.

Explore los productos de la capa gratuita con una cuenta de AWS nueva.

Para obtener más información, visite [aws.amazon.com/free](http://aws.amazon.com/free).

Registrarse en AWS

Dirección de correo electrónico

Utilizará esta dirección de correo electrónico para iniciar sesión en su nueva cuenta de AWS.

Contraseña

Confirmar la contraseña

Nombre de la cuenta de AWS

Elija un nombre para la cuenta. Podrá cambiarlo en la configuración de la cuenta después de registrarse.

Continuar (paso 1 de 5)

Iniciar sesión en una cuenta de AWS existente



# Crear cuenta en AWS

Ejemplo  
en vivo



- Completamos nuestros datos de nombre y dirección y continuamos.

**aws**

## Registrarse en AWS

**Ofertas de la capa gratuita**

Todas las cuentas de AWS pueden explorar 3 tipos diferentes de ofertas gratuitas, en función del producto utilizado.

**Siempre gratis**  
Nunca vence

**12 meses gratis**  
Inicio a partir de la fecha de registro inicial

**Pruebas**  
Inicio a partir de la fecha de activación del servicio

**Información de contacto**

¿Cómo tiene previsto utilizar AWS?

Empresarial: para su trabajo, escuela u organización

Personal: para sus propios proyectos

¿A quién debemos contactar para consultar sobre esta cuenta?

Nombre completo

Número de teléfono

Introduzca el código de país y el número de teléfono.

+1 222-333-4444

País o región

Argentina

Dirección

Apartamento, suite, unidad, edificio, planta, etc.

Ciudad

Estado, provincia o región

Código postal

He leído y acepto los términos del [Contrato de usuario de AWS](#).

**Continuar (paso 2 de 5)**



# Crear cuenta en AWS

Ejemplo  
en vivo



- Luego, debemos poner los datos de una tarjeta de crédito. Pero si los proyectos que implementamos luego los eliminamos, no nos cobran nada.

Verificación segura

No cobraremos el uso que esté por debajo de los límites de la capa gratuita de AWS. Retenemos temporalmente 1 USD/EUR como transacción pendiente por un periodo de 3 a 5 días para verificar su identidad.



Registrarse en AWS

Información de facturación

Número de tarjeta de crédito o débito

VISA MASTERCARD AMEX DISCOVER

AWS acepta todas las tarjetas de crédito y débito principales. Para obtener más información sobre las opciones de pago, consulte nuestras [preguntas frecuentes](#)

Fecha de vencimiento

Mes Año

Nombre del titular de la tarjeta

Dirección de facturación

Utilizar mi dirección de contacto  
Araoz 276  
Buenos Aires Buenos Aires 14114  
AR

Utilizar una nueva dirección

**Verificar y continuar (paso 3 de 5)**

Es posible que se le redirija al sitio web de su banco para autorizar el cargo de verificación.



# Crear cuenta en AWS

Ejemplo  
en vivo



- Ahora debemos poner nuestro número de teléfono para que nos pueda enviar AWS un SMS como verificación de identidad.
- Luego, ponemos el código de verificación que nos llega por SMS y continuamos.

aws

### Registrarse en AWS

#### Confirme su identidad

Para poder utilizar la cuenta de AWS, debe verificar su número de teléfono. Cuando continúe, el sistema automatizado de AWS se comunicará con usted para proporcionarle un código de verificación.

Código de país o región

Argentina (+54)

Número de teléfono móvil

11

Comprobación de seguridad

353nap

Escriba los caracteres como se indica arriba

Enviar SMS (paso 4 de 5)

aws

### Registrarse en AWS

#### Confirme su identidad

Verificar código

Continuar (paso 4 de 5)

¿Tiene algún problema? A veces, se necesitan hasta 10 minutos para recibir el código de verificación. Si ha transcurrido más tiempo del mencionado, [vuelva a la página anterior e intételo de nuevo](#).



# Crear cuenta en AWS

Ejemplo  
en vivo



- Finalmente, elegimos el plan de soporte que queremos, en este caso el gratuito, y luego finalizamos el registro.

Registrarse en AWS

Seleccionar un plan de soporte

Elija un plan de soporte para su cuenta personal o empresarial. [Compare planes y ejemplos de precio](#). Puede cambiar su plan en cualquier momento desde la consola de administración de AWS.

Soporte de nivel Basic: gratis

- Recomendado para los usuarios nuevos que recién comienzan a utilizar AWS
- Acceso de autoservicio las 24 horas del día, los 7 días de la semana a los recursos de AWS
- Solo para problemas de facturación y cuentas
- Acceso a Personal Health Dashboard y Trusted Advisor

Soporte Developer: a partir de 29 USD al mes

- Recomendado para desarrolladores que experimentan con AWS
- Acceso por correo electrónico a AWS Support durante el horario laboral
- Tiempos de respuesta de 12 horas (horario laboral)

Soporte Business: a partir de 100 USD al mes

- Recomendado para ejecutar cargas de trabajo de producción en AWS
- Soporte técnico las 24 horas, los 7 días de la semana por correo electrónico, teléfono y chat
- Tiempos de respuesta de 1 hora
- Conjunto completo de recomendaciones de prácticas de Trusted Advisor

¿Necesita soporte de nivel Enterprise?

A partir de los 15 000 USD por mes, tendrá tiempos de respuesta de 15 minutos y una experiencia de consejo con un director técnico de cuenta asignado. [Más información](#)

**Finalizar registro**



# Crear cuenta en AWS

Ejemplo  
en vivo



- De esta forma nos queda creada la cuenta, ya podemos seguir para crear una App en AWS.
- Nos llega también un mail de bienvenida a AWS.

The screenshot shows the AWS welcome email. At the top, there's a navigation bar with links like 'Productos', 'Soluciones', 'Precios', 'Documentación', 'Aprender', 'Red de socios', 'AWS Marketplace', 'Habilitación para clientes', 'Eventos', 'Explorar más', and a search icon. On the right, there's a button labeled 'Inicie sesión en la consola'. The main content features a blue rocket launching from a white cloud. Below it, the text reads '¡Felicitaciones!' and 'Gracias por registrarse en AWS.' A message states, 'Estamos activando la cuenta. Tardaremos solo unos minutos. Recibirá un correo electrónico cuando hayamos acabado.' At the bottom, there's a yellow button with the text 'Ir a la consola de administración de AWS' and a link to 'Iniciar sesión en otra cuenta o Entrar en contacto con el departamento de ventas'.

# ***CREAR NUESTRA APP EN ELASTIC BEANSTALK***



# Crear App en Elastic Beanstalk

- Para comenzar la implementación debemos iniciar sesión en la consola de AWS entrando al siguiente link [Consola AWS](#)

The screenshot shows the AWS console homepage with a specific project highlighted. The project title is "Implementar una aplicación web de Node.js". It includes a brief description: "Lance y ejecute una aplicación web de Node.js con alta disponibilidad en AWS". Below the title are two buttons: "Comenzar con el proyecto" and "6 pasos | 60 minutos". At the top right of the screenshot, the "Inicie sesión en la consola" button is highlighted with a red box.

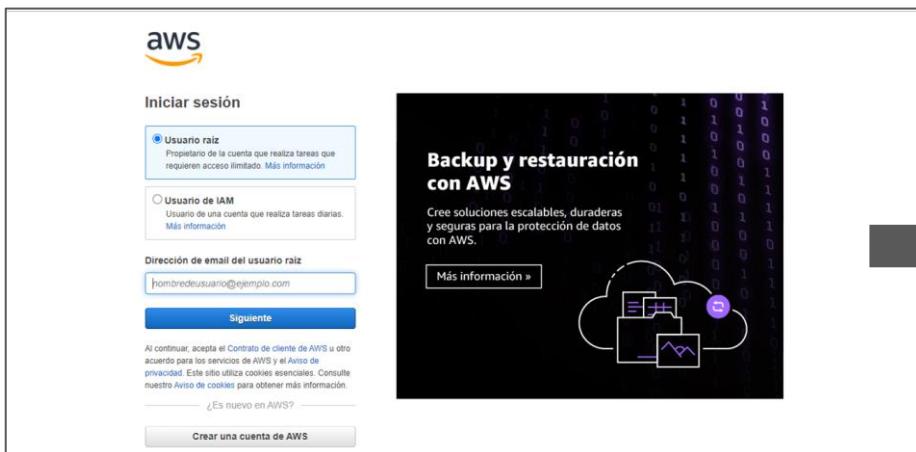
En este proyecto, aprenderá a implementar una aplicación web de Node.js de alta disponibilidad con AWS Elastic Beanstalk y Amazon DynamoDB. La aplicación de muestra que implementará utiliza Node.js, Express y una base de datos NoSQL. Si utiliza Elastic Beanstalk, solo tiene que cargar el código y Elastic Beanstalk administrará de manera automática la implementación, desde el aprovisionamiento de la capacidad, el equilibrio de carga y el escalado automático hasta la monitorización del estado de la aplicación. Elastic Beanstalk ajusta el escalado de la aplicación automáticamente en función de la necesidad específica de la aplicación, para lo que usa una configuración de Auto Scaling que se puede adaptar con facilidad. Amazon

**CODER HOUSE**

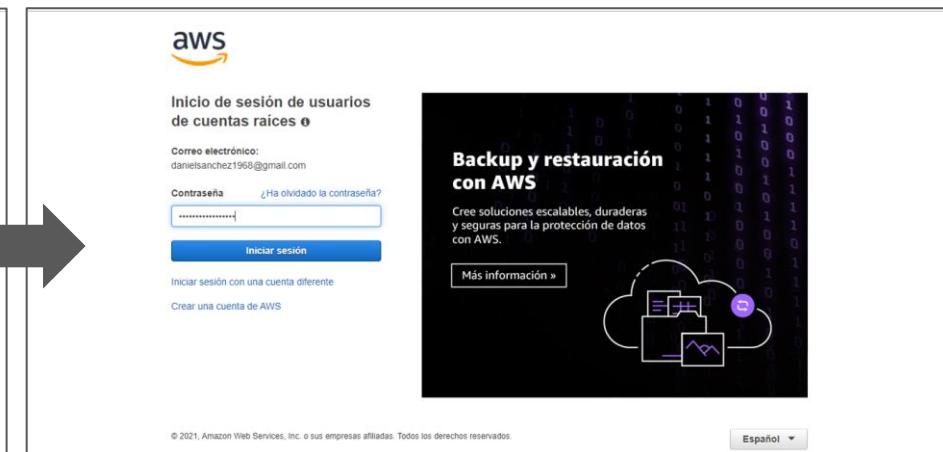
# Crear App en Elastic Beanstalk

Ejemplo  
en vivo

- Luego, elegimos la opción de usuario raíz, ponemos mail y contraseña e ingresamos.



The screenshot shows the AWS sign-in interface. It features a large 'aws' logo at the top left. Below it, there are two radio button options: 'Usuario raíz' (selected) and 'Usuario IAM'. A tooltip for 'Usuario raíz' states: 'Propietario de la cuenta que realiza tareas que requieren acceso ilimitado. Más información'. The 'Usuario IAM' option is described as 'Usuario de una cuenta que realiza tareas diarias. Más información'. There is also a field for 'Dirección de email del usuario raíz' containing 'hombradeusuario@example.com'. A prominent blue 'Siguiente' button is at the bottom. To the right of the sign-in form is a dark sidebar with the title 'Backup y restauración con AWS' and a subtext: 'Cree soluciones escalables, duraderas y seguras para la protección de datos con AWS.' It includes a 'Más información »' link and a cloud icon with binary code.



The second screenshot shows the continuation of the sign-in process. The 'aws' logo is at the top left. The 'Iniciar sesión de usuarios de cuentas raíces' section is visible, with a 'Correo electrónico:' field containing 'daniel.sanchez1968@gmail.com' and a 'Contraseña:' field with a placeholder '\*\*\*\*\*'. A link 'Ha olvidado la contraseña?' is next to the password field. Below these are 'Iniciar sesión' and 'Iniciar sesión con una cuenta diferente' buttons. At the bottom are links for 'Crear una cuenta de AWS' and '¿Es nuevo en AWS?'. The sidebar on the right remains the same as in the first screenshot.

# Crear App en Elastic Beanstalk

Ejemplo  
en vivo



- En la consola de AWS elegimos la opción de Elastic Beanstalk y luego vamos al botón de crear aplicación.

The screenshot illustrates the process of creating an application on AWS Elastic Beanstalk. It consists of two main parts: the AWS Management Console interface and the AWS Elastic Beanstalk service page.

**AWS Management Console (Left):**

- The title bar says "Consola de administración de AWS".
- The sidebar under "Servicios de AWS" shows "Elastic Beanstalk" highlighted with a red box.
- The main content area displays a "Manténgase conectado a sus recursos de AWS en cualquier lugar" section with a mobile app icon.
- Below it is an "Explore AWS" section featuring "Amazon Redshift".
- At the bottom, there are links for "Lance una máquina virtual", "Cree una aplicación web", and "Diseñe con servidores virtuales".

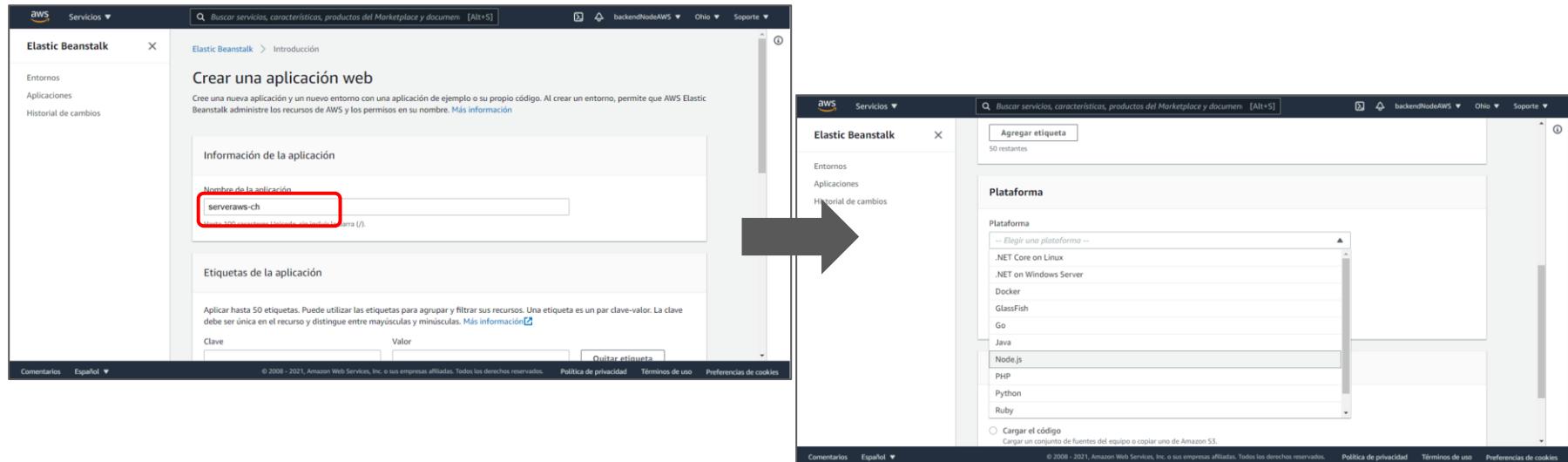
**AWS Elastic Beanstalk (Right):**

- The title bar says "AWS Elastic Beanstalk".
- The main heading is "AWS Elastic Beanstalk: Administración integral de aplicaciones web".
- A prominent red box highlights the "Empiece" button, which says "Implemente fácilmente su aplicación web en minutos de inicio".
- Below the button is a "Create Application" button.
- The "Cómo funciona" section explains the service's automatic implementation, scaling, and monitoring capabilities.
- At the bottom, there are links for "Introducción", "Política de privacidad", "Términos de uso", and "Preferencias de cookies".

# Configurar App en Elastic Beanstalk

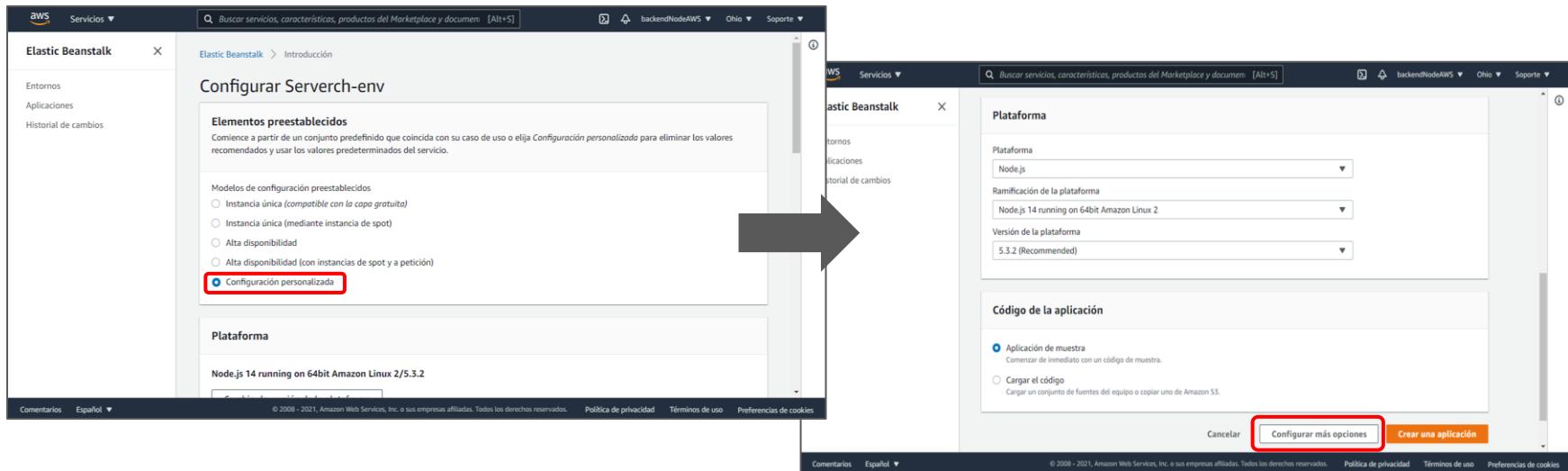
Ejemplo  
en vivo

- Elegimos el nombre de la aplicación, en este caso serveraws-ch y luego la plataforma que se corresponde al lenguaje utilizado, que es NodeJs.



# Configurar App en Elastic Beanstalk

- En código de aplicación seleccionamos aplicación de prueba y luego vamos a configurar más opciones y elegimos la configuración personalizada.



# Configurar App en Elastic Beanstalk

Ejemplo  
en vivo

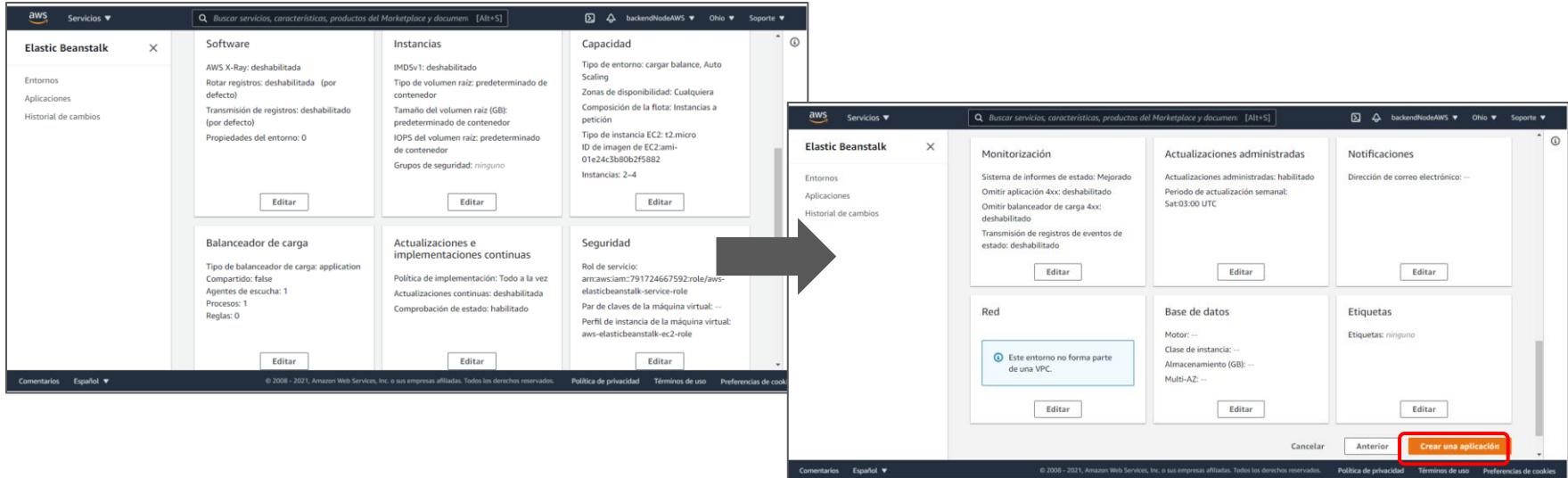
- Luego, modificamos la capacidad, configurando el tipo de entorno como carga balanceada. En el balanceador de carga configuramos las instancias con un mínimo de 2 y un máximo de 4.

The image shows two screenshots of the AWS Elastic Beanstalk console. The left screenshot displays the application configuration page for 'backendNodeAWS' in the 'Ohio' region. It shows the 'Plataforma' section with 'Node.js 14 running on 64bit Amazon Linux 2/5.3.2'. The 'Capacidad' section is highlighted with a red box, showing 'Tipo de entorno: instancia individual' and 'Instancia a petición'. The right screenshot shows the 'Modificar la capacidad' (Modify capacity) dialog box. This dialog also has a red box around the 'Capacidad' section, which includes a dropdown menu set to 'Carga balanceada', and input fields for 'Min.' (set to 2) and 'Máx.' (set to 4). Both screenshots include a large black arrow pointing from the 'Capacidad' section in the left panel to the 'Capacidad' section in the right dialog box.

# Configurar App en Elastic Beanstalk

Ejemplo  
en vivo

- Vamos hasta abajo de la página y clickeamos el botón de crear una aplicación con estos cambios que hicimos en la configuración.



# Crear App en Elastic Beanstalk

Ejemplo  
en vivo



- Elastic Beanstalk suele tardar unos minutos en crear el entorno y cada uno los recursos que administra.

The screenshot shows the AWS Elastic Beanstalk console with the following details:

- Servicios:** BackendNodeAWS, Ohio, Soporte
- Elastic Beanstalk:** Entornos, Aplicaciones, Historial de cambios
- serverch:** Versiones de la aplicación, Configuraciones guardadas
- Serverch-env:** Creando Serverch-env. Este proceso tardará unos minutos. ...
- Log messages:**
  - 1:53pm Created CloudWatch alarm named: awseb-e-zr7mpc4str-stack-AWSEBCloudwatchAlarmLow-AAQIMY3JAYQG
  - 1:53pm Created CloudWatch alarm named: awseb-e-zr7mpc4str-stack-AWSEBCloudwatchAlarmHigh-1KCMRLN3Z3YEN3
  - 1:53pm Created Auto Scaling group policy named: arn:aws:autoscaling:us-east-2:791724667592:scalingPolicy:30b96ae-371b-4874-8958-832d5701f870:autoScalingGroupName/awseb-e-zr7mpc4str-stack-AWSEBAutoScalingGroup-1TW6HJUTJJYBT:policyName/awseb-e-zr7mpc4str-stack-AWSEBAutoScalingScaleUpPolicy-E1N4KFk3F265
  - 1:53pm Created Auto Scaling group policy named: arn:aws:autoscaling:us-east-2:791724667592:scalingPolicy:ae384d33-4f7e-4a72-ad47-e5d3fa256d19:autoScalingGroupName/awseb-e-zr7mpc4str-stack-AWSEBAutoScalingGroup-1TW6HJUTJJYBT:policyName/awseb-e-zr7mpc4str-stack-AWSEBAutoScalingScaleDownPolicy-QPFRWC3WQGZE
  - 1:53pm Waiting for EC2 instances to launch. This may take a few minutes.
  - 1:53pm Created Auto Scaling group named: awseb-e-zr7mpc4str-stack-AWSEBAutoScalingGroup-1TW6HJUTJJYBT
  - 1:52pm Created security group named: awseb-e-zr7mpc4str-stack-AWSEBSecurityGroup-1GHV1HNLWU229
  - 1:51pm Created security group named: sg-03317617228e4d638



## Elastic Beanstalk cuenta con una serie de recursos:

- **Instancia EC2**
- **Grupo de seguridad de la instancia**
- **Balanceador de carga**
- **Grupo de seguridad del balanceador de carga**
- **Grupo de Auto Scaling**
- **Bucket de Amazon S3**
- **Alarmas de Amazon CloudWatch**
- **Scaling aumenta o reduce los recursos**
- **Pila de AWS CloudFormation**
- **Nombre de dominio**

¡Veámoslos en detalle!



# *Recursos del entorno de Elastic Beanstalk*



- **Instancia EC2:**

Máquina virtual de Amazon Elastic Compute Cloud (Amazon EC2) configurada para ejecutar aplicaciones web en la plataforma que elija. Cada plataforma ejecuta un conjunto específico de software, archivos de configuración y scripts compatibles con una determinada versión de lenguaje, marco de trabajo y contenedor web (o una combinación de estos). La mayoría de las plataformas utilizan Apache o nginx como un proxy inverso que se sitúa delante de la aplicación web, reenvía las solicitudes a esta, administra los recursos estáticos y genera registros de acceso y errores.



# Recursos del entorno de Elastic Beanstalk



- ***Grupo de seguridad de la instancia:***

Grupo de seguridad de Amazon EC2 configurado para permitir el tráfico entrante en el puerto 80. Este recurso permite que el tráfico HTTP procedente del balanceador de carga llegue a la instancia EC2 en la que se ejecuta la aplicación web. De forma predeterminada, el tráfico no está permitido en otros puertos.

- ***Balanceador de carga:***

Balanceador de carga de Elastic Load Balancing que está configurado para distribuir solicitudes a las instancias que se ejecutan en la aplicación. También permiten que las instancias no estén expuestas directamente a Internet.

- ***Grupo de seguridad del balanceador de carga:***



# *Recursos del entorno de Elastic Beanstalk*



- ***Grupo de Auto Scaling:***

Está configurado para reemplazar una instancia si termina o deja de estar disponible.

- ***Bucket de Amazon S3:***

Ubicación de almacenamiento para el código fuente, los registros y otros artefactos que se crean al utilizar Elastic Beanstalk.

- ***Alarmas de Amazon CloudWatch:***

Dos alarmas de CloudWatch que monitoriean la carga recibida por las instancias y que se activan si la carga es demasiado alta o demasiado baja. Cuando se activa una alarma, en respuesta, el grupo de Auto Scaling aumenta o reduce los recursos.



# Recursos del entorno de Elastic Beanstalk

Ejemplo  
en vivo



- **Plataforma de AWS CloudFormation:**

Elastic Beanstalk utiliza AWS CloudFormation para lanzar los recursos del entorno y propagar los cambios de configuración.

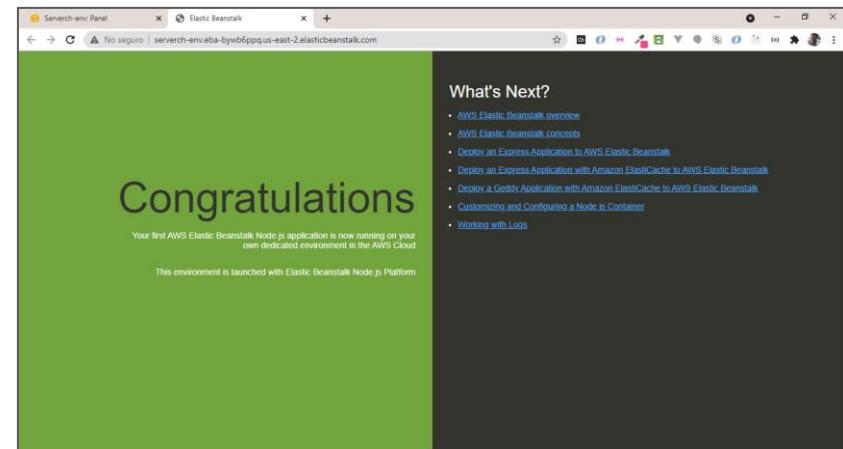
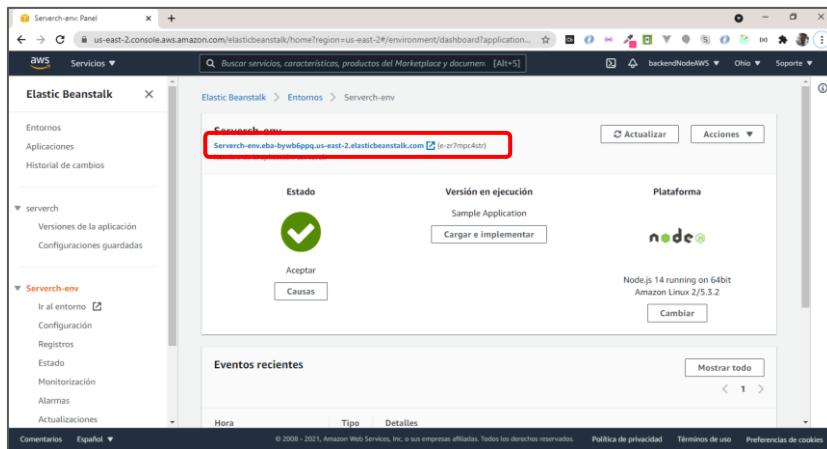
- **Nombre de dominio:**

Nombre de dominio que direcciona el tráfico a la aplicación web con el formato subdominio.region.elasticbeanstalk.com.

# Configuración de nuestra App

Ejemplo  
en vivo

- Una vez finalizada la instalación, haciendo click en el link seleccionado ya podemos ir a nuestra aplicación, donde veremos una ventana como la mostrada en la imagen.



# ***CONFIGURACIÓN DE NUESTRA APP***



# Configuración de nuestra App

- Vamos a Servicios > IAM > Roles, seleccionamos el rol asociado a instancias EC2, y le asociamos las siguientes políticas para usar base de datos y servicios de mensajería:
  - AmazonDynamoDBFullAccess
  - AmazonSNSFullAccess

The screenshot shows two side-by-side views of the AWS IAM service.

**Left View (Roles List):** Shows a list of roles. The role `aws-elasticbeanstalk-ec2-role` is selected and highlighted with a red box. Other roles listed include `aws-elasticbeanstalk-service-role`, `AWSServiceRoleForAutoScaling`, `AWSServiceRoleForElasticLoadBalancing`, `AWSServiceRoleForSupport`, and `AWSServiceRoleForTrustedAdvisor`.

**Right View (Role Permissions):** Shows the permissions tab for the selected role. A blue button labeled `Asociar políticas` (Associate policies) is highlighted with a red box. Below it, a table lists the applied policies:

Nombre de la política	Tipo de política
AmazonDynamoDBFullAccess	Política administrada por AWS
AWSElasticBeanstalkWebTier	Política administrada por AWS
AmazonDynamoDBFullAccesswithDataPipeline	Política administrada por AWS
AWSElasticBeanstalkMulticontainerDocker	Política administrada por AWS
AWSElasticBeanstalkWorkerTier	Política administrada por AWS
AmazonSNSFullAccess	Política administrada por AWS

# Configuración de nuestra App - DynamoDB

Ejemplo  
en vivo

- Luego, ingresamos a Servicios > DynamoDB > Tablas. Ahí creamos las tablas que vamos a necesitar en nuestra base de datos para la aplicación. Elegimos el nombre (en este caso: ‘product-inventory’), la clave primaria (‘productId’) que figura como *partition key*, y clickeamos el botón de crear.

The image consists of three screenshots of the AWS DynamoDB console, illustrating the step-by-step process of creating a new table named 'product-inventory'.

- Screenshot 1:** Shows the 'DynamoDB' service dashboard. A red box highlights the 'crear tabla' (Create Table) button in the top-left corner of the main content area.
- Screenshot 2:** Shows the 'Create a table in DynamoDB' wizard. The table name is 'product-inventory'. The primary key is defined as 'productId' (partition key). Under 'Configuration of the table', it shows the 'Default' configuration selected. A note at the bottom states: 'No tiene la función necesaria para habilitar Auto Scaling de forma predeterminada. Consulte Documentación.'
- Screenshot 3:** Shows the final step of the wizard. The table name is 'product-inventory', the primary key is 'productId', and the configuration is 'Default'. The 'Create' button is highlighted with a red box. A note at the bottom states: 'No tiene la función necesaria para habilitar Auto Scaling de forma predeterminada. Consulte Documentación.'

**CODER HOUSE**

# Configuración de nuestra App - DynamoDB

Ejemplo en vivo



- Una vez creada la tabla, ya está lista para crear elementos en ella.

The screenshot displays two side-by-side views of the AWS DynamoDB console. On the left, the 'Información general' tab of the 'registro-usuarios' table is shown, indicating it is being created ('Se está creando la tabla'). It includes sections for 'Alertas recientes', 'Detalles de Kinesis Data Stream' (with a note about using Amazon Kinesis Data Streams to capture changes at the level of Kinesis data), and 'Detalles del flujo de DynamoDB' (with fields for 'Flujo habilitado', 'Nombre del flujo', and ARN). On the right, the 'Elementos' tab is selected, showing a search bar and a table header for 'Examen: [Tabla] tabla\_usuarios: email'. A search input field contains 'email' with a magnifying glass icon. A tooltip at the bottom right explains that an item consists of one or more attributes, each with a name, type, and value. Both screenshots show the same navigation sidebar on the left, which includes links for 'DynamoDB', 'Panel', 'Tablas', 'Copias de seguridad', 'Capacidad reservada', 'Exportaciones a S3', 'Editor PartiQL', 'Preferencias', 'DAX', 'Panel', 'Clústeres', 'Grupos de subredes', 'Grupos de parámetros', 'Eventos', and 'Pruebe la vista previa de la nueva consola'. The bottom of the screen shows standard AWS footer links: 'Comentarios', 'Español', '© 2006 - 2021, Amazon Web Services, Inc. o sus empresas afiliadas. Todos los derechos reservados.', 'Política de privacidad', and 'Términos de uso'.

# Configuración de nuestra App - SNS

Ejemplo  
en vivo

- Luego vamos Servicios -> SNS -> Tópicos y creamos un tópico nuevo: notificaciones

The screenshot shows the AWS SNS service configuration interface. On the left, there's a sidebar with navigation links: Dashboard, Topics (which is selected and highlighted in blue), Subscriptions, Mobile (with Push notifications, Text messaging (SMS), and Origination numbers), and Marketplace. The main content area has a search bar at the top with the text 'sns'. Below it, the results for 'sns' are displayed under three categories: Services, Features, and Documentation.

- Services:** Shows 'Simple Notification Service' with a sub-description 'SN3 managed message topics for Pub/Sub'. This item is also highlighted with a red box.
- Features:** Shows 'Events' and 'ElastiCache feature'.
- Documentation:** Shows 'Sns\_Samples.cs - AWS Code Sample' and a 'Catalog' link.

To the right, a large modal window titled 'Create topic' is open. It contains a form with a 'Topic name' field containing the value 'notificaciones'. A red box highlights the 'Next step' button. Below the button, there are links to 'Start with an overview' and 'Pricing'.

# Configuración de nuestra App - SNS

Ejemplo  
en vivo

- Elegimos la configuración estándar y aceptamos al final de la página.

The screenshot shows the 'Create topic' wizard in the Amazon SNS console. The top navigation bar includes 'Amazon SNS > Topics > Create topic'. The main section is titled 'Create topic' and has a 'Details' tab selected. Under the 'Type' section, 'Standard' is chosen and highlighted with a red box. The 'Standard' type description lists: Best-effort message ordering, At-least once message delivery, Highest throughput in publishes/second, and Subscription protocols: SQS, Lambda, HTTP, SMS, email, mobile application endpoints. Below the type selection is a 'Name' field containing 'notificaciones', which is also highlighted with a red box. At the bottom right of the form are 'Cancel' and 'Create topic' buttons, with 'Create topic' also highlighted with a red box.

▶ **Delivery retry policy (HTTP/S) - optional**  
The policy defines how Amazon SNS retries failed deliveries to HTTP/S endpoints. To modify the default settings, expand this section. [Info](#)

Amazon SNS > Topics > Create topic

## Create topic

### Details

Type [Info](#)  
Topic type cannot be modified after topic is created

FIFO (first-in, first-out)

- Strictly-preserved message ordering
- Exactly-once message delivery
- High throughput, up to 300 publishes/second
- Subscription protocols: SQS

Standard

- Best-effort message ordering
- At-least once message delivery
- Highest throughput in publishes/second
- Subscription protocols: SQS, Lambda, HTTP, SMS, email, mobile application endpoints

Name

notificaciones

Maximum 256 characters. Can include alphanumeric characters, hyphens (-) and underscores (\_).

Cancel **Create topic**

# Configuración de nuestra App - SNS

Ejemplo  
en vivo

- Debería aparecer una confirmación de la creación del tópico, con un ARN asociado. Debemos copiar ese código para usarlo más adelante:

The screenshot shows the 'notificaciones' topic details in the Amazon SNS console. A green success message at the top states: "Topic notificaciones created successfully. You can create subscriptions and send messages to them from this topic." Below this, the 'notificaciones' topic is listed under 'Topics'. The 'Details' section shows the following information:

Name	Display name
notificaciones	-
ARN	Topic owner
arn:aws:sns:us-east-1:786671515187:notificaciones	786671515187
Type	
Standard	

A red box highlights the ARN value: arn:aws:sns:us-east-1:786671515187:notificaciones.

# Configuración de nuestra App - SNS

Ejemplo  
en vivo

- Luego vamos a suscripciones, y creamos una nueva suscripción. Esto nos permitirá recibir notificaciones cada vez que se publique un nuevo evento en el tópico creado:

The screenshot shows the Amazon SNS Subscriptions page. On the left, there's a sidebar with links: Dashboard, Topics, **Subscriptions** (which is highlighted with a red box), and a Mobile section with Push notifications, Text messaging (SMS), and Origination numbers. The main area has a header "Amazon SNS > Subscriptions". Below it, there's a table titled "Subscriptions (1)" with columns: ID, Endpoint, Status, Protocol, and Topic. At the top of this table area, there are five buttons: Edit, Delete, Request confirmation, Confirm subscription, and a prominent orange "Create subscription" button, which is also highlighted with a red box. A search bar is located above the table.

# Configuración de nuestra App - SNS

Ejemplo  
en vivo

- Seleccionamos el tópico al cual suscribirnos, el medio de suscripción (email en este caso) e ingresamos el destinatario (nuestra dirección de

Amazon SNS > Subscriptions > Create subscription

## Create subscription

**Details**

**Topic ARN**

**Protocol**  
The type of endpoint to subscribe

**Endpoint**  
An email address that can receive notifications from Amazon SNS.

**CODER HOUSE**

# Configuración de nuestra App - SNS

Ejemplo  
en vivo



- La suscripción está creada pero para que esté activa debemos confirmarla mediante un link enviado a la dirección de la suscripción:

Subscription to notificaciones created successfully.  
The ARN of the subscription is arn:aws:sns:us-east-1:786671515187:notificaciones:b28391c2-5eef-4fe7-ab9c-7d513737f5b7.

Amazon SNS > Topics > notificaciones > Subscription: b28391c2-5eef-4fe7-ab9c-7d513737f5b7

Subscription: b28391c2-5eef-4fe7-ab9c-7d513737f5b7

Details	
ARN	arn:aws:sns:us-east-1:786671515187:notificaciones:b28391c2-5eef-4fe7-ab9c-7d513737f5b7
Status	Pending confirmation
Protocol	EMAIL
Endpoint	[REDACTED]@gmail.com
Topic	notificaciones

AWS Notification - Subscription Confirmation Inbox ×

AWS Notifications <no-reply@sns.amazonaws.com>  
to me ▾

You have chosen to subscribe to the topic:  
arn:aws:sns:us-east-1:786671515187:notificaciones

To confirm this subscription, click or visit the link below (If this was in error no action is necessary)  
[Confirm subscription](#)

Please do not reply directly to this email. If you wish to remove yourself from receiving all future SNS subscription confirmation emails, please unsubscribe.

# Confirmar suscripción



- De esta forma, con esta página que se nos abre queda confirmada nuestra suscripción.



Simple Notification Service

## Subscription confirmed!

You have successfully subscribed.

Your subscription's id is:

arn:aws:sns:us-east-1:786671515187:notificaciones:b28391c2-5eef-4fe7-ab9c-7d513737f5b7

If it was not your intention to subscribe, [click here to unsubscribe](#).



# BREAK

¡5/10 MINUTOS Y VOLVEMOS!

# ***CREAR NUESTRO PROYECTO***



# Configurar nuestro proyecto de prueba

Trabajaremos ahora sobre una aplicación de pruebas. Nuestro proyecto será un CRUD bien sencillo. Comenzamos con el package.json en donde incluiremos express y el SDK de AWS:

```
{  
  "name": "ejemplo-aws",  
  "author": "coderhouse",  
  "version": "1.0.0",  
  "description": "",  
  "main": "app.js",  
  "scripts": {  
    "start": "node ."  
  },  
  "keywords": [],  
  "license": "ISC",  
  "dependencies": {  
    "aws-sdk": "^2.874.0",  
    "express": "^4.17.1"  
  }  
}
```



# Configurar nuestro proyecto de prueba

Luego, incluimos requerimientos, instanciamos y configuramos nuestras dependencias:

```
const express = require('express');
const AWS = require('aws-sdk');

AWS.config.update({
  region: 'us-east-1'
});

const sns = new AWS.SNS()
const SNS_TOPIC_ARN = 'arn:aws:sns:us-east-1:786671515187:notificaciones'

const dynamodb = new AWS.DynamoDB.DocumentClient();
const TABLE_NAME = 'product-inventory';

const app = express();
app.use(express.json());

app.get('/', (req, res) => { res.send('test api') })
```



# Configurar nuestro proyecto de prueba

Agregamos los controladores a las rutas de nuestros endpoints CRUD:

```
app.get('/api/productos', async (req, res) => {
  const params = {
    TableName: TABLE_NAME
  }
  try {
    const productos = await scanDynamoRecords(params);
    res.json(productos);
  } catch (error) {
    console.error('Ocurrió un error: ', error);
    res.sendStatus(500);
  }
})
```

```
async function scanDynamoRecords(scanParams) {
  try {
    let dynamoData = await dynamodb.scan(scanParams).promise();
    const items = dynamoData.Items
    while (dynamoData.LastEvaluatedKey) {
      scanParams.ExclusiveStartKey = dynamoData.LastEvaluatedKey;
      dynamoData = await dynamodb.scan(scanParams).promise();
      items.push(...dynamoData.Items);
    }
    return items;
  } catch (error) {
    throw new Error(error);
  }
}
```



# Configurar nuestro proyecto de prueba

```
app.post('/api/productos', (req, res) => {
  const params = {
    TableName: TABLE_NAME,
    Item: req.body
  }
  dynamodb.put(params).promise()
    .then(() => {
      console.log('se guardó')
      const prod = JSON.stringify(req.body)
      return sns.publish({
        Message: `nuevo producto agregado! ${prod}`,
        Subject: 'nuevo producto',
        TopicArn: SNS_TOPIC_ARN
      }).promise()
    })
})
```

```
.then(data => {
  console.log('se notificó')
  console.log(data)

  const body = {
    Operation: 'SAVE',
    Message: 'SUCCESS',
    Item: req.body
  }
  res.json(body);
})

.catch(error => {
  console.error('Ocurrió un error: ', error);
  res.status(500).end();
})
})
```



# Configurar nuestro proyecto de prueba

```
app.put('/api/productos/:id', (req, res) => {
  const item = {
    ...req.body,
    productId: req.params.id
  }
  const params = {
    TableName: TABLE_NAME,
    Item: item
  }
```

```
dynamodb.put(params).promise()
  .then(() => {
    const body = {
      Operation: 'UPDATE',
      Message: 'SUCCESS',
      Item: item
    }
    res.json(body);
  })
  .catch(error => {
    console.error('Ocurrió un error: ', error);
    res.sendStatus(500);
  })
})
```



# Configurar nuestro proyecto de prueba

```
app.delete('/api/productos/:id', (req, res) => {
  const params = {
    TableName: TABLE_NAME,
    Key: {
      'productId': req.params.id
    },
    ReturnValues: 'ALL_OLD'
  }
})
```

```
dynamodb.delete(params).promise()
  .then(response => {
    const body = {
      Operation: 'DELETE',
      Message: 'SUCCESS',
      Item: response
    }
    res.json(body);
  })
  .catch(error => {
    console.error('Ocurrió un error: ', error);
    res.sendStatus(500);
  })
})
```



# Configurar nuestro proyecto de prueba

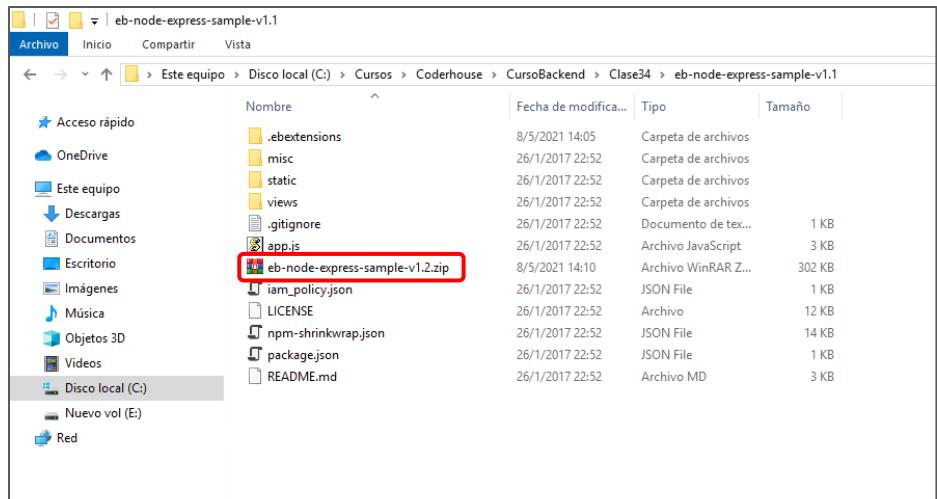
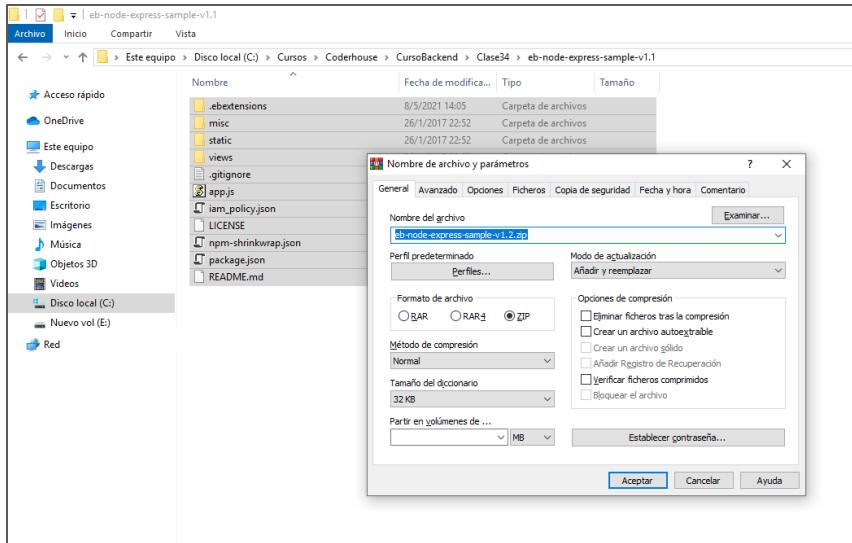
Finalmente, agregamos el código para conectar nuestro servidor al puerto 8080 o al que corresponda según las variables de entorno:

```
const port = process.env.PORT || 8080;
const server = app.listen(port, () => {
  console.log(`Escuchando en puerto ${server.address().port}`);
})
server.on('error', error => { console.log(error) })
```

# **CARGAR NUESTRO PROYECTO EN LA PLATAFORMA DE ELASTIC BEANSTALK**

# Configurar nuestro proyecto de prueba

- Creamos un zip con todos los archivos de nuestro proyecto.



# Implementar nuestro proyecto



- Volvemos a la página de nuestra aplicación en AWS y vamos al botón de carga e implementar. Clickeamos en elegir archivo para cargar nuestro proyecto.

The screenshot shows the AWS Elastic Beanstalk console for the environment 'Serverch-env'. The main view displays the application 'Serverch-env' with the URL 'Serverch-env.eba-bywb6ppq.us-east-2.elasticbeanstalk.com'. The 'Estado' section shows a green checkmark icon, indicating the application is running. The 'Versión en ejecución' section shows 'Sample Application' and a 'Cargar e implementar' button, which is highlighted with a red box. Below these sections, there are buttons for 'Aceptar' and 'Causas'. The 'Plataforma' section shows 'node.js' and 'Node.js 14 running on 64bit Amazon Linux 2/5.3.2'. At the bottom, there's a section for 'Eventos recientes' with a 'Mostrar todo' button. The left sidebar lists 'Entornos', 'Aplicaciones', 'Historial de cambios', and specific sections for 'serverch' and 'Serverch-env' (which is currently selected). The 'Serverch-env' section includes links for 'Ir al entorno', 'Configuración', 'Registros', 'Estado', 'Monitorización', 'Alarms', and 'Actualizaciones'.

# Implementar nuestro proyecto



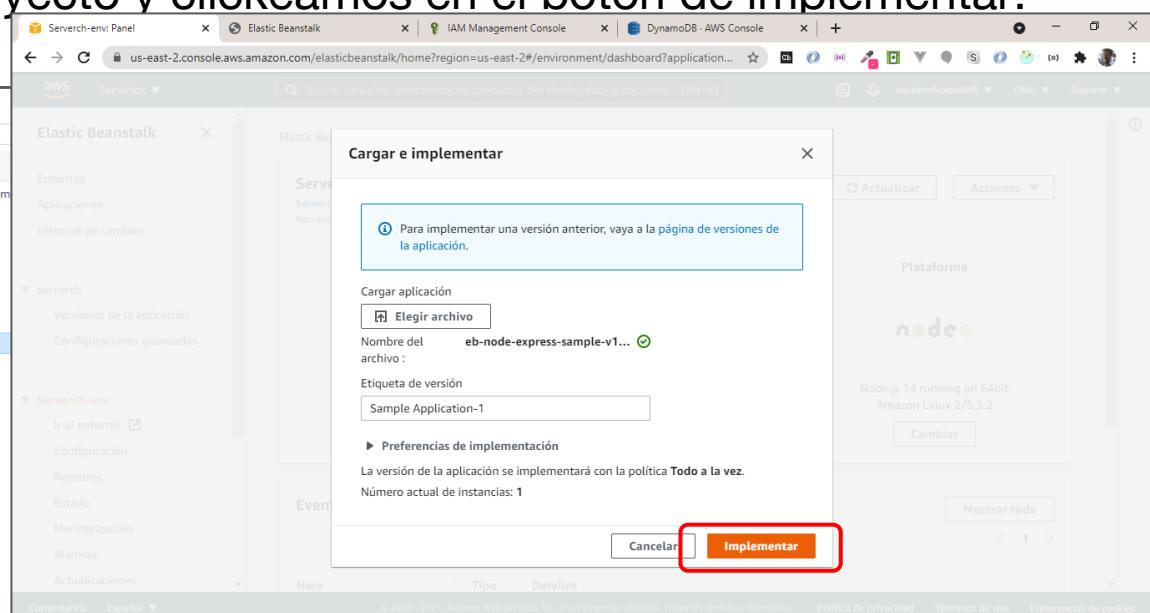
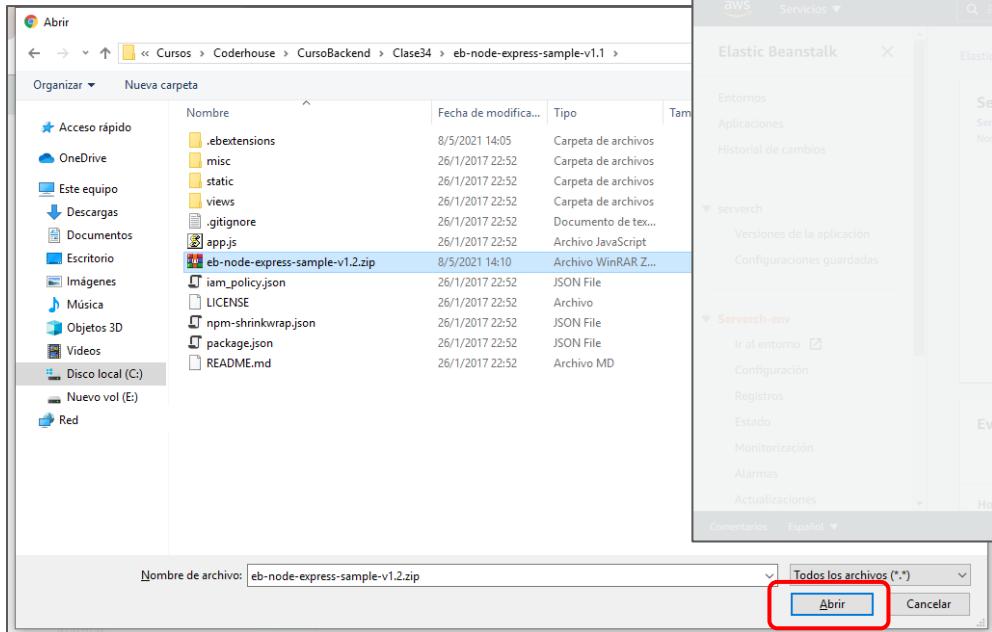
- Volvemos a la página de nuestra aplicación en AWS y vamos al botón de carga e implementar. Clickeamos en elegir archivo para cargar nuestro proyecto.

The screenshot shows the AWS Elastic Beanstalk interface. On the left, there's a sidebar with sections like 'Entornos', 'Aplicaciones', 'Historial de cambios', 'serverch', 'Versiones de la aplicación', and 'Configuraciones guardadas'. The main area displays the environment 'serverch-env' with details about the platform ('Plataforma node.js') and the Node.js version ('Node.js 14 running on 64bit Amazon Linux 2/5.3.2'). A modal window titled 'Cargar e implementar' is open. It contains a note: 'Para implementar una versión anterior, vaya a la página de versiones de la aplicación.' Below this is a section labeled 'Carga aplicación' with a red box around the 'Elegir archivo' (Select file) button. There's also a 'Etiqueta de versión' (Version label) input field and a 'Preferencias de implementación' (Deployment preferences) section. At the bottom of the modal are 'Cancelar' and 'Implementar' buttons. The status bar at the bottom of the browser window shows 'us-east-2.console.aws.amazon.com' and other service links.

# Implementar nuestro proyecto



- Elegimos nuestro zip de proyecto y clickeamos en el botón de implementar.



**CODER HOUSE**

# Implementar nuestro proyecto



- Una vez que se actualiza el entorno de Elastic Beanstalk vemos la notificación que se implementó una nueva versión en la instancia.

The screenshot shows the AWS Elastic Beanstalk console for the 'Serverch-env' environment. A prominent message at the top states: 'Elastic Beanstalk está actualizando su entorno. Para cancelar esta operación, seleccione Anular la operación actual en el menú desplegable Acciones. Ver los eventos'. Below this, the application details are shown: 'Estado' (Deployment in progress), 'Versión en ejecución' (Sample Application), and 'Plataforma' (Node.js 14 running on 64bit Amazon Linux 2/5.3.2). On the left sidebar, under 'Serverch-env', there are sections for 'Versiones de la aplicación' and 'Configuraciones guardadas'. At the bottom, there are links for 'Comentarios' and 'Español'.

The screenshot shows the AWS Elastic Beanstalk console for the 'Serverch-env' environment. It displays deployment history and recent events. The deployment history shows: 'Node.js 14 running on 64bit Amazon Linux 2/5.3.2'. The 'Eventos recientes' section lists the following events:

Hora	Tipo	Detalles
08-05-2021 14:21:41 UTC-0300	INFO	Deploying new version to instance(s).
08-05-2021 14:21:39 UTC-0300	INFO	Environment health has transitioned from Ok to Info. Application update in progress (running for 45 seconds).
08-05-2021 14:20:46 UTC-0300	INFO	Environment update is starting.
08-05-2021 13:54:53 UTC-0300	INFO	Successfully launched environment: Serverch-env
08-05-2021 13:54:55 UTC-0300	INFO	Application available at Serverch-env.eba-byw6ppg.us-east-2.elasticbeanstalk.com.

At the bottom, there are links for 'Comentarios' and 'Español'.

# Implementar nuestro proyecto



- Y luego, chequeamos la notificación de que se actualizó correctamente el entorno.

The screenshot shows the AWS Elastic Beanstalk console for the environment 'Serverch-env'. The main interface displays the application's status as 'Estado' (Green with checkmark), 'Versión en ejecución' (Sample Application), and 'Plataforma' (Node.js 14 running on 64bit Amazon Linux 2/5.3.2). On the left sidebar, under the 'serverch' section, there is a link to the recent event log. The 'Eventos recientes' table shows two entries:

Hora	Tipo	Detalles
08-05-2021 14:22:15 UTC-0300	INFO	Environment update completed successfully.
08-05-2021 14:22:15 UTC-0300	INFO	New application version was deployed to running EC2 instances.

# Verificación



- Ya podemos utilizar el link del entorno para probar nuestra api REST usando nuestro cliente HTTP preferido (por ejemplo, postman)

The screenshot shows the AWS Elastic Beanstalk console. On the left, there's a sidebar with navigation links like 'Entornos', 'Aplicaciones', 'Historial de cambios', and sections for 'serverch' and 'Serverch-env'. The 'Serverch-env' section is expanded, showing options like 'Ir al entorno', 'Configuración', 'Registros', 'Estado', 'Monitorización', 'Alarms', and 'Actualizaciones'. The main panel displays the environment details: 'Estado' (green checkmark), 'Versión en ejecución' (Sample Application), 'Plataforma' (node.js), and 'Eventos recientes'. A red box highlights the URL 'Serverch-env.eba-bw6b6ppq.us-east-2.elasticbeanstalk.com' in the top bar.

# **ELIMINAR NUESTRA APP DE LA PLATAFORMA ELASTIC BEANSTALK**

# Remover el entorno



- Vamos a acciones y elegimos terminar el entorno. En el cartel que nos aparece, ponemos el nombre del entorno y clickeamos en terminar.

The screenshot shows the AWS Elastic Beanstalk console interface. On the left, there's a sidebar with navigation links like 'Entornos', 'Aplicaciones', 'Historial de cambios', and 'serveraws-ch'. Under 'serveraws-ch', 'Serverawsch-env' is selected, with sub-links 'Ir al entorno', 'Configuración', 'Registros', 'Estado', 'Monitorización', 'Alarms', 'Actualizaciones administradas', 'Eventos', and 'Etiquetas'. The main content area shows the 'Serverawsch-env' environment details. A red box highlights the 'Acciones' button in the top right corner of the main panel. Another red box highlights the 'Terminar el entorno' option in the dropdown menu that appears when clicking 'Acciones'. To the right, a modal window titled 'Confirmar terminación del entorno' is displayed. It asks '¿Desea terminar Serverawsch-env permanentemente? Esta acción no se puede deshacer.' Below it, a list of environment details is shown: 'Capa: Servidor web', 'Plataforma: Node.js 14 running on 64bit Amazon Linux 2/5.3.2', 'Versión: Sample Application-2', and 'Última modificación: 09-05-2021 09:09:29 UTC-0300'. At the bottom of the modal, a red arrow points to a text input field where 'Serverawsch-env' is typed, and a red box highlights the 'Terminar' button.

# Eliminar la App creada



- Chequeamos que se removió correctamente la instancia del entorno.
- Luego vamos a acciones y elegimos eliminar aplicación.
- Recordamos que vamos a eliminar la aplicación por el tema del cobro de la cuenta de AWS.

Serveraws-ch-env  
Serveraws-ch-env.eba-jnpewce.us-east-2.elasticbeanstalk.com (c-9f8tkmz12u)  
Nombre de la aplicación: serveraws-ch

Estado: Desconocido

Versión en ejecución: Sample Application-2

Plataforma: node.js 14 running on 64bit Amazon Linux 2/5.3.2

Eventos recientes:

Hora	Tipo	Detalles
09-05-2021 09:17:25 UTC-0300	INFO	Removed instance [i-0b5d4ee8f2b880e3b] from your environment.
09-05-2021 09:16:25 UTC-0300	INFO	Environment health has transitioned from Ok to Info. Terminate in progress (running for 11 seconds).
09-05-2021 09:16:15	INFO	Warning for EC2 instances to terminate. This may take a few minutes.

Elastic Beanstalk > Aplicaciones

Todas las aplicaciones

Nombre de la aplicación	Entornos	Fecha de creación	Última modificación	ARN
serveraws-ch-env	Serveraws-ch-env	09-05-2021 08:33:11 UTC-0300	09-05-2021 08:33:11 UTC-0300	arn:aws:elasticbeanstalk:us-east-2:2791724667592:application/serveraws-ch

# Eliminar la App creada



- Ponemos el nombre de la aplicación y luego eliminar.
- Luego, debemos escribir la palabra delete y clickear en el botón eliminar.

The screenshot displays two side-by-side screenshots of the AWS Management Console.

**Left Screenshot (Elastic Beanstalk):** Shows the "Todas las aplicaciones" (All Applications) page. A modal window titled "Confirmar eliminación de la aplicación" (Confirm application deletion) is open. It asks if you want to permanently delete "serveraws-ch". Below the question, it says "Si continúa con esta acción, se terminarán los siguientes entornos:" (If you continue with this action, the following environments will be terminated: "Serverawsch-env"). A red arrow points to the input field where "serveraws-ch" is typed, and another red arrow points to the "Eliminar" (Delete) button at the bottom right of the modal.

**Right Screenshot (DynamoDB):** Shows the "Tables" section of the DynamoDB console. A modal window titled "Eliminar tabla" (Delete table) is open, asking "¿Está seguro de que desea eliminar esta tabla: registro-usuarios?". It includes checkboxes for "Eliminar todas las alarmas de CloudWatch para esta tabla" (Delete all CloudWatch alarms for this table) and "Crear una copia de seguridad antes de eliminar esta tabla" (Create a backup before deleting this table). Below the checkboxes is a text input field containing "delete", with a red arrow pointing to it. At the bottom right of the modal is a red-bordered "Eliminar" (Delete) button.

# Eliminar la App creada



- Finalmente nos redirige de nuevo a la página inicial de AWS.

The screenshot shows the AWS Elastic Beanstalk landing page. At the top, there's a navigation bar with the AWS logo, a search bar, and dropdown menus for 'Servicios', 'backendNodeAWS', 'Ohio', and 'Soporte'. The main title 'AWS Elastic Beanstalk' is prominently displayed, followed by the subtitle 'Administración integral de aplicaciones web.' Below this, a description explains that Elastic Beanstalk is a service for deploying and scaling web applications developed with Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker. A 'Create Application' button is visible. To the left, a sidebar lists 'Entornos', 'Aplicaciones', and 'Historial de cambios'. On the right, sections include 'Empezar' (with a sub-note about implementing the application in minutes), 'Precios' (noting no additional charges for Beanstalk), 'Cómo funciona' (explaining the automatic deployment process), 'Beneficios y características' (mentioning ease of use, control over resources, and launching applications), and 'Introducción' (with a 'Lanzar una aplicación web' button). At the bottom, there are links for 'Comentarios', 'Español', and legal notices.



# ***IMPLEMENTAR PROYECTO EN AWS***

*Tiempo: 10 minutos*



# Implementar proyecto en AWS

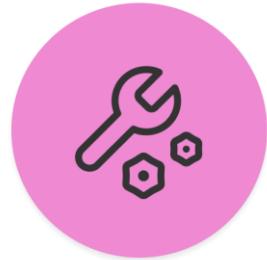
Tiempo: 10 minutos

Crear una cuenta en AWS para trabajar con la plataforma Elastic Beanstalk.

Vamos a implementar la primera parte del proyecto de referencia en AWS:

<https://aws.amazon.com/es/getting-started/hands-on/deploy-nodejs-web-app/>

- Crear un proyecto sobre la plataforma Elastic Beanstalk que contenga un proxy reverso nginx con balanceador de carga y de 2 a 4 instancias de servidor en autoscaling.
- Implementar el servidor con el código de ejemplo.
- Cuando el servidor se ponga online, verificar la página devuelta en su ruta raíz.



# ***CONTINUAR CON LA IMPLEMENTACIÓN***

*Tiempo: 10 minutos*



# Continuar con la implementación

Tiempo: 10 minutos

Continuando con lo realizado en el desafío anterior, y utilizando el código de ejemplo suministrado en <https://aws.amazon.com/es/getting-started/hands-on/deploy-nodejs-web-app/>

- Dar permisos a la aplicación para tener acceso a los servicios de DynamoDB y SNS en la plataforma AWS.
- Crear una tabla en DynamoDB que contenga una clave email tipo string, registrando su nombre dentro del archivo de configuración ubicado en el código de ejemplo.
- Configurar el mail por el cual vamos a recibir notificaciones en el archivo de configuración correspondiente en el código de ejemplo.



# Continuar con la implementación

Tiempo: 10 minutos

- Setear el proxy para las static files en la configuración del código de ejemplo.
- Zipear el código de ejemplo y subirlo a la plataforma Elastic Beanstalk.
- Verificar que nos llegue el mail de suscripción. Registrarse al servicio de notificaciones por SNS, cliqueando el link que viene en este email.
- Una vez que la aplicación se ponga online, mediante la página que proveé en su ruta raíz, registrar dos usuarios, comprobando que se guarden en la tabla creada en DynamoDB y lleguen los mails correspondientes avisando por cada nueva registración realizada.

# *Continuar con la implementación*

Desafío  
generico



Tiempo: 10 minutos

- Realizar un cambio en el código, puede ser en un archivo de vista, zipeando esta nueva versión y subiéndola a la plataforma. Comprobar que este nuevo cambio esté disponible online y todo funcione correctamente.
- Luego de probar el desempeño del servidor, realizar la finalización del entorno y de la aplicación, sin olvidar borrar la tabla creada en DynamoDB.



# ***DESPLEGAR NUESTRO PROYECTO EN LA NUBE***

Retomemos nuestro trabajo para implementar nuestro proyecto en Heroku

# IMPLEMENTAR NUESTRO PROYECTO EN HEROKU

**Formato:** link a un repositorio en Github con el proyecto cargado.

**Sugerencia:** no incluir los node\_modules

Desafío  
entregable



## >> Consigna:

Crear un proyecto en Heroku.com para subir el servidor que venimos realizando, reformando todo lo necesario para su correcto funcionamiento en la nube.

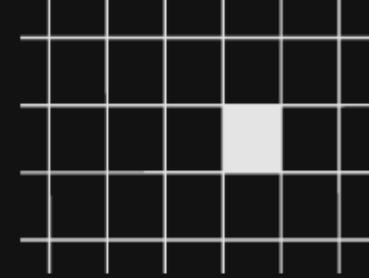
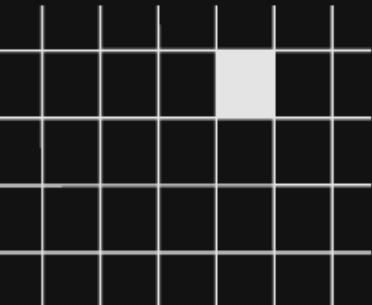
Subir el código a Heroku.com, sin olvidar incluir el archivo .gitignore para evitar subir los node\_modules. Comprobar que el proyecto inicie de manera correcta en la nube. Verificar que en su ruta raíz se encuentre la página pública del servidor.

El servidor debe seguir funcionando en forma local.

Realizar un cambio a elección en alguna vista, probar en forma local y subir nuevamente el proyecto a Heroku, verificando que la nueva reforma esté disponible online.

Revisar a través de una consola local, los mensajes enviados por nuestro servidor en Heroku a su propia consola.

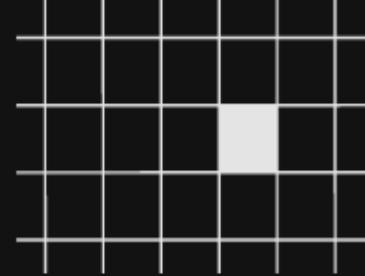
*¿PREGUNTAS?*



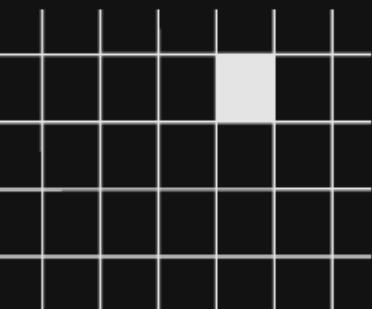
# ***MUCHAS GRACIAS!***

Resumen de lo visto en clase hoy:

- AWS y su plataforma Elastic Beanstalk.
- Creación de una cuenta y una App en esta plataforma.
- Implementación de nuestro proyecto en la misma.



***OPINA Y VALORA ESTA CLASE***



**#DEMOCRATIZANDOLAEDUCACIÓN**