CODER HOUSE

Clase 36. Programación Backend

Twilio & OWASP

OBJETIVOS DE LA CLASE

- Aprender a enviar mensajes de Whatsapp desde aplicaciones en Node.
- Conocer acerca de OWASP y los riesgos de seguridad de aplicaciones más importantes.
- Comprender los riesgos del OWASP Top 10.

CRONOGRAMA DEL CURSO

Clase 35

Envío de mensajes y Seguridad

Clase 36

Twilio & OWASP

Clase 37

Creación de proyectos

CODER HOUSE

TWILIO WHATSAPP

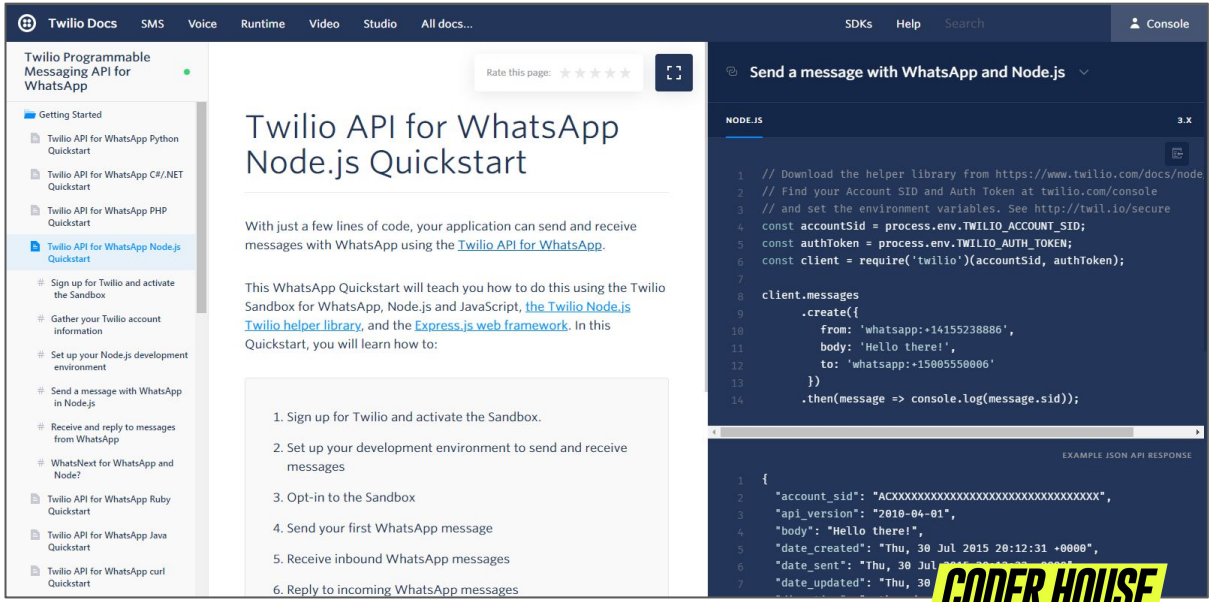
CODER HOUSE

¿De qué se trata?

- De forma similar a como vimos el envío de SMS a través de Twilio en la clase pasada, se puede enviar y recibir mensajes de Whatsapp también mediante Twilio.
- Funciona como intermediario entre la cuenta de Whatsapp y nosotros.
- Podemos enviar los mensajes desde nuestra app de Node con el módulo de Twilio para Node que utilizamos la clase pasada. Pero esta vez lo vamos a configurar para que envíe mensajes de Whatsapp.

¿De qué se trata?

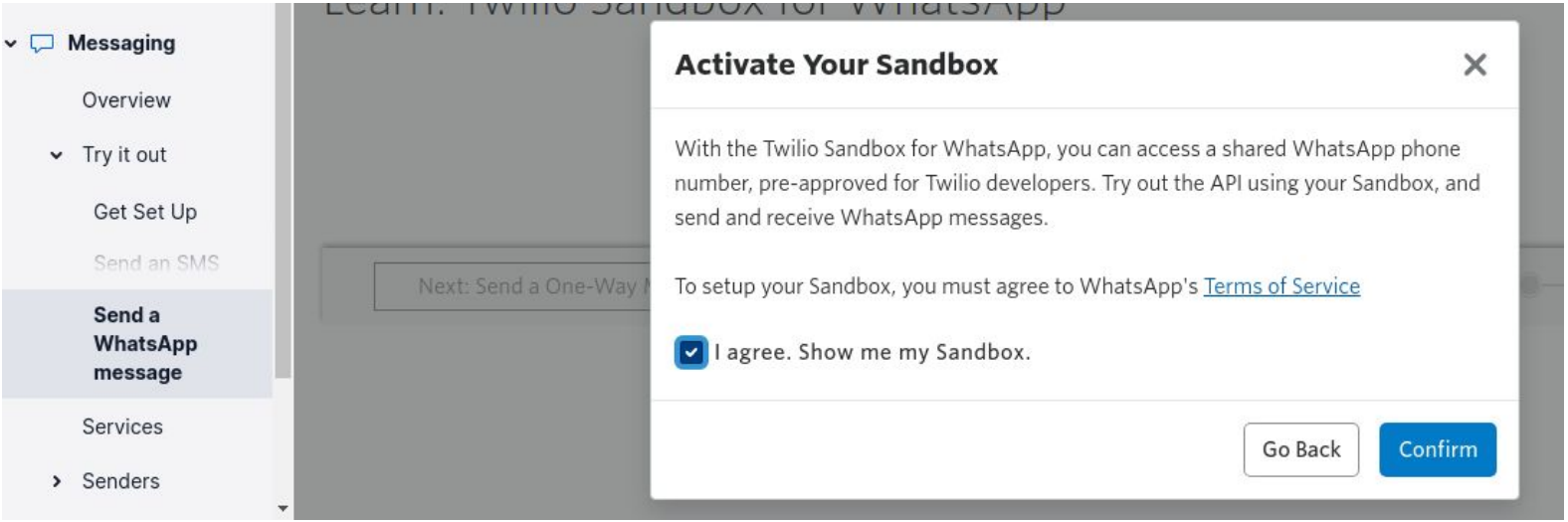
Entrando a Twilio para Whatsapp podremos ver la documentación que nos permitirá empezar a usarlo.



Comenzar a utilizar a Twilio-Whatsapp

Ejemplo en vivo

1. Ingresamos de nuevo a la cuenta de Twilio que nos creamos la última actividad, volvemos al menú de mensajes, y activamos nuestro sandbox (espacio de pruebas).



Comenzar a utilizar a Twilio-Whatsapp

Ejemplo en vivo

2. El cliente para realizar la prueba va a tener que enviar un mensaje con un código que suele ser ***join alguna-cosa*** al número que nos ofrece Twilio.

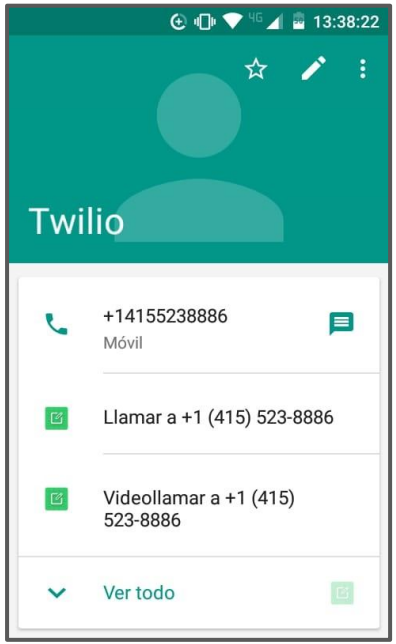
CODER HOUSE

Comenzar a utilizar a Twilio-Whatsapp

Ejemplo en vivo

3. Vamos entonces a nuestro Whatsapp y enviamos el mensaje.

De esta forma ya nos queda configurado para recibir los mensajes que enviemos desde nuestro proyecto.



CODER HOUSE

Configurarlo en nuestro proyecto

Ejemplo en vivo

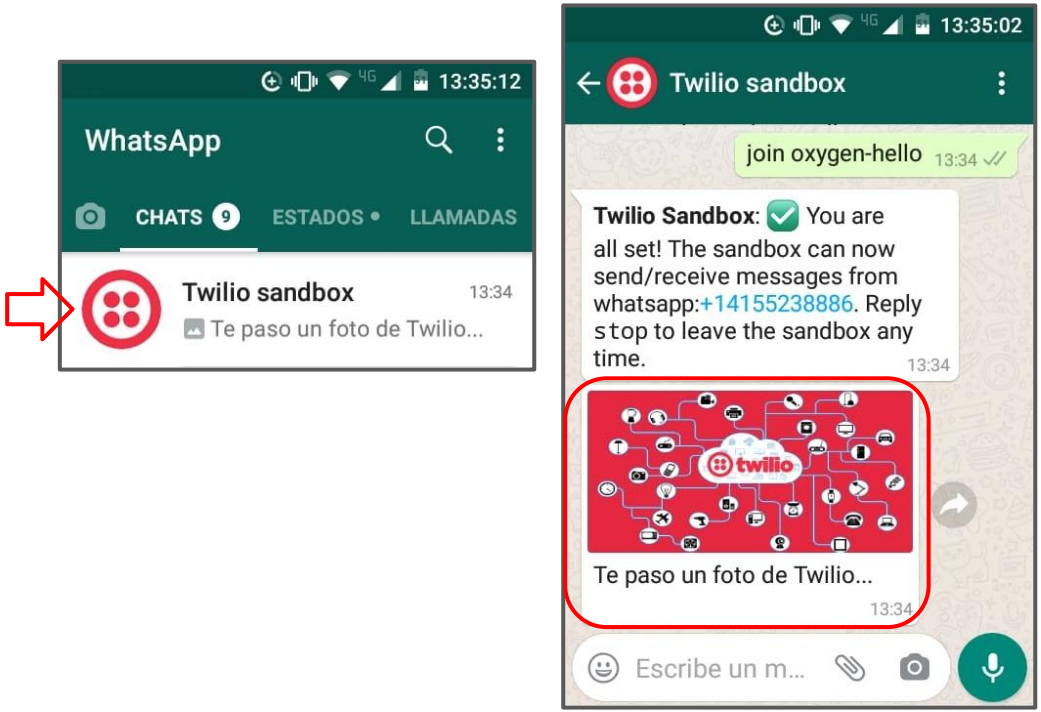
4. De igual forma que vimos la clase pasada con SMS, configuramos Twilio, especificando antes del número que es mensaje de Whatsapp.

```
const options = {
   body: 'Hola soy un WSP desde Node.js!',
   mediaUrl: [ 'https://www.investingmoney.biz/public/img/art/xl/18012019161021Twilio-IoT.jpg' ],
   from: 'whatsapp:+14155238886',
   to: 'whatsapp:+5491100000000'
}
```

👉 En este ejemplo, enviamos con *mediaUrl* un adjunto en el mensaje.
Esto es opcional, si no se desea enviar adjuntos.

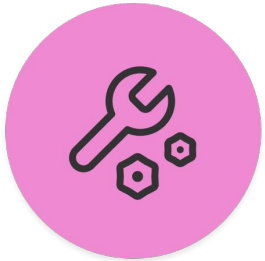
Configurarlo en nuestro proyecto

Ejemplo en vivo



5. Vamos nuevamente a nuestro Whatsapp y podemos corroborar que haya llegado el mensaje que enviamos (una vez que ejecutamos el código de nuestro proyecto).

CODER HOUSE



ENVIAR WHATSAPP DESDE NODE

Tiempo: 10 minutos

CODER HOUSE

Enviar Whatsapp desde Node

Desafío generico

Tiempo: 10 minutos

Realizar en Node.js un programa llamado 'enviarWapp' que permita enviar mensajes de texto a la red social Whatsapp.

- ➔ Configurar en la cuenta en Twilio la operación de Whatsapp SandBox (https://www.twilio.com/console/sms/whatsapp/sandbox/) habilitando a través de este servicio, el número telefónico del Whatsapp en el que se quieren recibir los mensajes.
- ➔ El número telefónico destino y el mensaje a enviar se le pasarán a la aplicación por línea de línea de comandos.
- ➔ Verificar que el mensaje de Whatsapp llegue al número indicado y que la operación se refleje en la consola de Twilio.

NOTA: Twilio brinda un número gratis para el envío de Whatsapp que viene con un monto en USD. Cada vez que enviemos un mensaje, se descontará el costo de la operación de dicho monto.

BREAK

¡5/10 MINUTOS Y VOLVEMOS!

OWASP

CODER HOUSE

¿Qué es?

- ***OWASP*** es Open Web Application Security Project (Proyecto abierto de seguridad de Aplicaciones Web).
- Es un proyecto de código abierto dedicado a determinar y combatir las causas que hacen que el software sea inseguro.
- La Fundación OWASP es un organismo sin fines de lucro que apoya y gestiona los proyectos e infraestructura de OWASP. La comunidad OWASP está formada por empresas y organizaciones educativas y particulares.

¿Qué es?

- Es un nuevo tipo de entidad en el mercado de seguridad informática.
- Está libre de presiones corporativas y eso facilita que OWASP proporcione información imparcial y práctica sobre seguridad de aplicaciones informáticas.
- No está afiliado a ninguna compañía tecnológica, si bien apoya el uso informado de tecnologías de seguridad.
- Recomienda enfocar la seguridad de aplicaciones informáticas considerando todas sus dimensiones: personas, procesos y tecnologías.

OSWAP TOP 10

CODER HOUSE

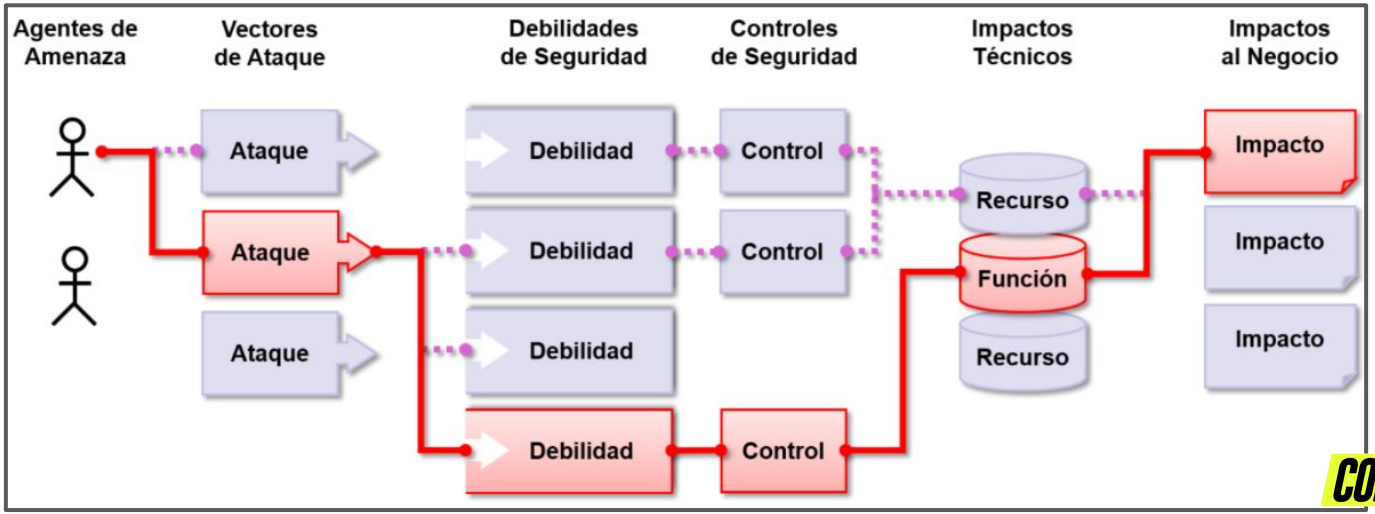
¿De qué se trata?

- OWASP Top 10 es un documento que recopila los diez riesgos de seguridad más importantes (críticos) en aplicaciones web según la organización OWASP.
- Esta lista se suele actualizar y publicar cada 3 o 4 años.
- El objetivo es crear conciencia acerca de la seguridad en aplicaciones mediante la identificación de algunos de los riesgos más críticos que enfrentan las organizaciones.
- La última versión publicada a la fecha es del 2017.

👉 Puedes consultar la última versión en Current Version (2017)

¿Cuáles son los riesgos en seguridad de aplicaciones?

- Los atacantes pueden, potencialmente, utilizar diferentes rutas a través de su aplicación para perjudicar su negocio u organización.
- Cada uno de estos caminos representa un riesgo que puede o no ser suficientemente grave como para merecer atención.



CODER HOUSE

Veamos cada uno de los aspectos destacados en el informe 2017...

CODER HOUSE

A1:2017 Inyección

- Las fallas de inyección, como consultas SQL, NoSQL, o casi cualquier fuente de datos, ocurren cuando se envían datos no confiables a un intérprete, como parte de un comando o consulta.
- Los datos dañinos del atacante pueden engañar al intérprete para que ejecute comandos involuntarios o acceda a los datos sin la debida autorización.
- Estos defectos son muy comunes, particularmente en código heredado.
- Una inyección puede causar divulgación, pérdida o corrupción de información, pérdida de auditabilidad, o denegación de acceso.
- El impacto al negocio depende de las necesidades de la aplicación y de los datos.

A2:2017 Pérdidas de autenticación

- Las funciones de la aplicación relacionadas a autenticación y gestión de sesiones son implementadas incorrectamente, permitiendo a los atacantes comprometer usuarios y contraseñas, token de sesiones, o explotar otras fallas de implementación para asumir la identidad de otros usuarios (temporal o permanentemente).
- Los errores de pérdida de autenticación son comunes debido al diseño y la implementación de la mayoría de los controles de acceso.
- Los atacantes solo tienen que obtener el acceso a unas pocas cuentas o a una cuenta de administrador para comprometer el sistema. Dependiendo del dominio de la aplicación, esto puede permitir robo de identidad, lavado de dinero y la divulgación de información sensible protegida legalmente.

A3:2017 Exposición de datos sensibles

- Muchas aplicaciones web y APIs no protegen adecuadamente datos sensibles, tales como información financiera como tarjetas de crédito o información personal como de salud. Los atacantes pueden robar o modificar estos datos protegidos inadecuadamente para llevar a cabo fraudes con tarjetas de crédito, robos de identidad u otros delitos.
- Los datos sensibles requieren métodos de protección adicionales, como el cifrado en almacenamiento y tránsito.
- El error más común es simplemente no cifrar los datos sensibles. Cuando se emplea criptografía, es común la generación y gestión de claves, algoritmos, cifradores y protocolos débiles.

A4:2017 Entidades externas XML (XXE)

- Muchos procesadores XML antiguos o mal configurados evalúan referencias a entidades externas en documentos XML. Las entidades externas pueden utilizarse para revelar archivos internos mediante la URI o archivos internos en servidores no actualizados, escanear puertos de la LAN, ejecutar código de forma remota y realizar ataques de denegación de servicio (DoS).
- Estos defectos se pueden utilizar para extraer datos, ejecutar una solicitud remota desde el servidor, escanear sistemas internos, realizar un ataque de denegación de servicio y ejecutar otro tipo de ataques.
- El impacto al negocio depende de las necesidades de la aplicación y de los datos.

A5:2017 Pérdida de control de acceso

- Las restricciones sobre lo que los usuarios autenticados pueden hacer no se aplican correctamente. Los atacantes pueden explotar estos defectos para acceder, de forma no autorizada, a funcionalidades y/o datos, cuentas de otros usuarios, ver archivos sensibles, modificar datos, cambiar derechos de acceso y permisos, etc.
- Las debilidades del control de acceso son comunes debido a la falta de detección automática y a la falta de pruebas funcionales efectivas por parte de los desarrolladores de aplicaciones.
- El impacto técnico incluye atacantes anónimos actuando como usuarios o administradores; usuarios que utilizan funciones privilegiadas o crean, acceden, actualizan o eliminan cualquier registro.

A6:2017 Configuración de seguridad incorrecta

- La configuración de seguridad incorrecta es un problema muy común y se debe en parte a establecer la configuración de forma manual, ad hoc o por omisión (o directamente por la falta de configuración).
- Son ejemplos: cabeceras HTTP mal configuradas, mensajes de error con contenido sensible, falta de parches y actualizaciones, frameworks, dependencias y componentes desactualizados, etc.
- Los atacantes a menudo intentarán explotar vulnerabilidades sin parchear o acceder a cuentas por defecto, etc. para obtener acceso o conocimiento del sistema o del negocio.
- Ocasionalmente, estos errores resultan en un completo compromiso del sistema.

A7:2017 Secuencia de comandos en sitios cruzados (XSS)

- Los XSS ocurren cuando una aplicación toma datos no confiables y los envía al navegador web sin una validación y codificación apropiada; o actualiza una página web existente con datos suministrados por el usuario utilizando una API que ejecuta JavaScript en el navegador.
- Permiten ejecutar comandos en el navegador de la víctima y el atacante puede secuestrar una sesión, modificar los sitios web, o redireccionar al usuario hacia un sitio malicioso.
- Existen herramientas automatizadas que permiten detectar y explotar las tres formas de XSS, y también se encuentran disponibles kits de explotación gratuitos.
- Es la segunda vulnerabilidad más frecuente en OSWAP.

A8:2017 Deserialización insegura

- Estos defectos ocurren cuando una aplicación recibe objetos serializados dañinos y estos objetos pueden ser manipulados o borrados por el atacante para realizar ataques de repetición, inyecciones o elevar sus privilegios de ejecución.
- En el peor de los casos, la deserialización insegura puede conducir a la ejecución remota de código en el servidor.
- Algunas herramientas pueden descubrir defectos de deserialización, pero con frecuencia se necesita ayuda humana para validarlo.
- Lograr la explotación de deserialización es difícil por lo que no es algo tan frecuente en las aplicaciones web. Sin embargo no se debe desvalorizar su impacto por la ejecución remota de código.

A9:2017 Componentes con vulnerabilidades conocidas

- Los componentes como bibliotecas, frameworks y otros módulos se ejecutan con los mismos privilegios que la aplicación. Si se explota un componente vulnerable, el ataque puede provocar una pérdida de datos o tomar el control del servidor.
- Las aplicaciones y API que utilizan componentes con vulnerabilidades conocidas pueden debilitar las defensas de las aplicaciones y permitir diversos ataques e impactos.
- Estos defectos están muy difundidos. El desarrollo basado fuertemente en componentes de terceros, puede llevar a que los desarrolladores no entiendan qué componentes se utilizan en la aplicación o API y, mucho menos, mantenerlos actualizados.
- Dependiendo del activo que se está protegiendo, este riesgo puede ser incluso el principal de la lista.

A10:2017 Registro y monitoreo insuficientes

- El registro y monitoreo insuficiente, junto a la falta de respuesta ante incidentes permiten a los atacantes mantener el ataque en el tiempo, pivotear a otros sistemas y manipular, extraer o destruir datos.
- Los estudios muestran que el tiempo de detección de una brecha de seguridad es mayor a 200 días, siendo típicamente detectado por terceros en lugar de por procesos internos.
- Los atacantes dependen de la falta de monitoreo y respuesta oportuna para lograr sus objetivos sin ser detectados.
- Los ataques más exitosos comienzan con la exploración de vulnerabilidades.
- Permitir que el sondeo de vulnerabilidades continúe puede aumentar la probabilidad de una explotación exitosa.

Cambios desde la anterior versión (2013)

- Dos diferenciadores clave sobre las versiones anteriores son las *notables devoluciones de la comunidad* y la *gran cantidad de datos recopilados* de docenas de organizaciones, siendo posiblemente la mayor cantidad de datos jamás reunidos en la preparación de un estándar de seguridad de aplicaciones.
- Esto nos da la confianza de que el nuevo OWASP Top 10 aborda los riesgos de seguridad de aplicaciones más impactantes que enfrentan las organizaciones en la actualidad.
- Se agregaron los nuevos riesgos A8 y A10 que no se consideran en las versiones anteriores.

CODER HOUSE

Recomendaciones para mejorar la seguridad de nuestra App

- Se recomienda establecer y utilizar procesos de seguridad repetibles y controles estándar de seguridad.
- Para ayudar a organizaciones y desarrolladores a reducir los riesgos de seguridad de sus aplicaciones de un modo rentable, OWASP ha producido un gran número de recursos gratuitos y abiertos, que los podemos utilizar para gestionar la seguridad de nuestras aplicaciones.
- A continuación, en la siguiente diapositiva, se muestran algunos de los muchos recursos que OWASP ha producido para ayudar a los desarrolladores a generar aplicaciones web y APIs seguras.

Recomendaciones para mejorar la seguridad de nuestra App

- **Requisitos de Seguridad en aplicaciones**: OWASP recomienda utilizar el Estándar de Verificación de Seguridad en Aplicaciones de OWASP (ASVS).
- **Arquitectura de seguridad en aplicaciones**: Es mucho más rentable diseñar la seguridad desde el principio.
- **Controles de Seguridad Estándar**: Construir controles de seguridad fuertes y usables es difícil. Un conjunto de controles estándar de seguridad simplifican radicalmente el desarrollo de aplicaciones y APIs seguras.

Recomendaciones para mejorar la seguridad de nuestra App

- **Ciclo de vida de desarrollo seguro**: Para mejorar el proceso que una organización utiliza para crear aplicaciones y APIs, OWASP recomienda el Modelo de Garantía de la Madurez del Software (SAMM). Este modelo ayuda a las organizaciones a formular e implementar estrategias para el software seguro, adaptado a los riesgos específicos para un negocio u organización.
- **Educación de la Seguridad en Aplicaciones**: El proyecto educacional de OWASP proporciona material de formación para ayudar a educar a los desarrolladores en seguridad en aplicaciones web.

CODER HOUSE

TERCERA ENTREGA DEL PROYECTO FINAL

Deberás entregar el avance de tu aplicación eCommerce Backend correspondiente a la tercera entrega de tu proyecto final.

CODER HOUSE

TERCERA ENTREGA DEL PROYECTO FINAL

Formato: link a un repositorio en Github con el proyecto cargado.
Sugerencia: no incluir los node_modules

>>Se debe entregar:

- Un menú de registro y autenticación de usuarios basado en passport local, guardando en la base de datos las credenciales y el resto de los datos ingresados al momento del registro.
    - El registro de usuario consiste en crear una cuenta en el servidor almacenada en la base de datos, que contenga el email y password de usuario, además de su nombre, dirección, edad, número de teléfono (debe contener todos los prefijos internacionales) y foto ó avatar. La contraseña se almacenará encriptada en la base de datos.
    - La imagen se podrá subir al servidor y se guardará en una carpeta pública del mismo a la cual se tenga acceso por url.

CODER HOUSE

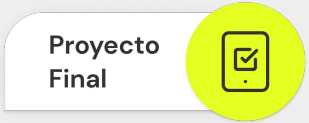
TERCERA ENTREGA DEL PROYECTO FINAL

Formato: link a un repositorio en Github con el proyecto cargado.
Sugerencia: no incluir los node_modules

- Un formulario post de registro y uno de login. De modo que, luego de concretarse cualquiera de estas operaciones en forma exitosa, el usuario accederá a su home.
    - El usuario se logueará al sistema con email y password y tendrá acceso a un menú en su vista, a modo de barra de navegación. Esto le permitirá ver los productos totales con los filtros que se hayan implementado y su propio carrito de compras e información propia (datos de registro con la foto). Además, dispondrá de una opción para desloguearse del sistema.
    - Ante la incorporación de un usuario, el servidor enviará un email al administrador con todos los datos de registro y asunto 'nuevo registro', a una dirección que se encuentre por el momento almacenada en una constante global.

TERCERA ENTREGA DEL PROYECTO FINAL

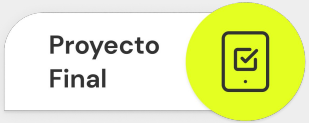
Formato: link a un repositorio en Github con el proyecto cargado.
Sugerencia: no incluir los node_modules



- Envío de un email y un mensaje de whatsapp al administrador desde el servidor, a un número de contacto almacenado en una constante global.
    - El usuario iniciará la acción de pedido en la vista del carrito.
    - Será enviado una vez finalizada la elección para la realizar la compra de productos.
    - El email contendrá en su cuerpo la lista completa de productos a comprar y en el asunto la frase 'nuevo pedido de ' y el nombre y email del usuario que los solicitó. En el mensaje de whatsapp se debe enviar la misma información del asunto del email.
    - El usuario recibirá un mensaje de texto al número que haya registrado, indicando que su pedido ha sido recibido y se encuentra en proceso.

TERCERA ENTREGA DEL PROYECTO FINAL

Formato: link a un repositorio en Github con el proyecto cargado.
Sugerencia: no incluir los node_modules



>>Aspectos a incluir:

- ➔ El servidor trabajará con una base de datos DBaaS (Ej. MongoDB Atlas) y estará preparado para trabajar en forma local o en la nube a través de la plataforma PaaS Heroku.
- ➔ Habilitar el modo cluster para el servidor, como opcional a través de una constante global.
- ➔ Utilizar alguno de los loggers ya vistos y así reemplazar todos los mensajes a consola por logs eficientes hacia la misma consola. En el caso de errores moderados ó graves el log tendrá además como destino un archivo elegido.
- ➔ Realizar una prueba de performance en modo local, con y sin cluster, utilizando Artillery en el endpoint del listado de productos (con el usuario vez logueado). Verificar los resultados.

CODER HOUSE

¿PREGUNTAS?

¡MUCHAS GRACIAS!

Resumen de lo visto en clase hoy:

- Envío de mensajes de Whatsapp desde Node con Twilio
- OWASP, riesgos de seguridad y las OWASP Top 10.

OPINA Y VALORA ESTA CLASE

#DEMOCRATIZANDOLAEDUCACIÓN

CODER HOUSE