

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

Actividad

Número 2

Seguridad en la

arquitectura

MEAN Stack

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

ÍNDICE

1. Introducción: 3
- 2. Dimensiones y requisitos de seguridad 4**
- 3. Lista de salvaguardas: 16**
- 4. Abordar el diseño de seguridad de la aplicación web: 16**
- 5. Esquema de protección online 19**
- 6. Conclusiones: 20**

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

1. Introducción:

Objetivos

- Elaborar la lista de requisitos y salvaguardas de seguridad de una aplicación web.
- Abordar el diseño de seguridad de un supuesto real de una aplicación web con tecnologías concretas.
- Incorporar un esquema de protección online.

MEAN Stack:

MEAN Stack es un conjunto de tecnologías que se utilizan en el desarrollo web de aplicaciones completas. El acrónimo MEAN representa las iniciales de las siguientes tecnologías:

- MongoDB: Es una base de datos NoSQL orientada a documentos. Utiliza documentos en formato JSON para almacenar datos, lo que permite una gran flexibilidad en el esquema de la base de datos.
- Node.js: Es un entorno de tiempo de ejecución de JavaScript construido sobre el motor de JavaScript V8 de Chrome. Permite ejecutar JavaScript en el servidor y facilita la creación de aplicaciones web en tiempo real y basadas en eventos.
- Express.js: Es un framework de aplicaciones web para Node.js. Proporciona una capa de abstracción sobre Node.js que facilita la creación de API web y el manejo de solicitudes y respuestas HTTP.
- Angular: Es un framework de desarrollo de aplicaciones web de código abierto desarrollado por Google. Angular permite construir aplicaciones web de una sola página (SPA) y proporciona una estructura sólida y modular para el desarrollo de front-end.

Arquitectura de MEAN Stack:

En la arquitectura de MEAN Stack, MongoDB se utiliza como base de datos para almacenar los datos de la aplicación. Express.js se encarga de gestionar las rutas y las solicitudes HTTP, proporcionando una API para comunicarse con el cliente. Angular se utiliza en el lado del cliente para crear interfaces de usuario interactivas y dinámicas que consumen los datos de la

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

API proporcionada por Express.js. Por último, Node.js actúa como el servidor web que ejecuta la aplicación y coordina la comunicación entre la base de datos, el servidor y el cliente.

MEAN Stack es una combinación de MongoDB, Express.js, Angular y Node.js, que se utilizan juntos para desarrollar aplicaciones web modernas y escalables. Cada componente desempeña un papel importante en la arquitectura global y permite a los desarrolladores construir aplicaciones de extremo a extremo utilizando JavaScript en todos los niveles.

2. Dimensiones y requisitos de seguridad

La seguridad de una aplicación web es esencial para proteger la información de los usuarios y la integridad del sistema. En el caso de una aplicación web que utiliza Google Chrome y MEAN STACK (MongoDB, Express, Angular y Node.js), existen varias dimensiones de seguridad que deben considerarse para garantizar la protección adecuada.

RS-01 Autenticación: Autenticación con varios factores	
Versión	1.0 - 20-5-2023
Descripción	Implementar la autenticación de varios factores para mejorar la seguridad de la aplicación. Esto implica requerir más de un método de autenticación para verificar la identidad del usuario
Importancia	Alta
Prioridad	Alta
Estado	Implementado
Tecnología	AngularJS, Node.js, MongoDB.
Recomendaciones	<ul style="list-style-type: none"> ➤ Implementar las librerías Angular2-authy para Angular o Speakeasy para Node.js, que proporcionan funcionalidades de autenticación de dos factores ➤ Para MongoDB, utilizar la funcionalidad de almacenamiento seguro de contraseñas proporcionada por Bcrypt para almacenar de forma segura las contraseñas de los usuarios.

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

RS-02 Autorización: Implementación de roles y permisos	
Versión	1.0 - 20-5-2023
Descripción	El sistema debe permitir asignar roles y permisos a los usuarios para controlar su acceso a diferentes funcionalidades y recursos
Importancia	Alta
Prioridad	Alta
Estado	Implementado
Tecnología	AngularJS, Node.js, MongoDB
Recomendaciones	<ul style="list-style-type: none"> ➤ Para AngularJS, utilizar el módulo Angular-acl para implementar la gestión de roles y permisos de autorización en el lado del cliente. ➤ Para Node.js, utilizar el middleware Express-jwt junto con Jsonwebtoken para implementar la autenticación basada en tokens y autorización basada en roles en el lado del servidor. ➤ Para MongoDB, utilizar los mecanismos de control de acceso integrados, como el uso de roles y usuarios de MongoDB, para definir permisos a nivel de base de datos.

RS-03 Gestión de sesiones: Gestión segura de sesiones	
Versión	1.0 - 20-5-2023
Descripción	El sistema debe gestionar las sesiones de forma segura para evitar ataques de sesión y garantizar la privacidad del usuario
Importancia	Alta
Prioridad	Alta
Estado	Implementado
Tecnología	Express.js, Node.js

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

Recomendaciones	<ul style="list-style-type: none"> ➤ Para Express.js, utilizar la librería Express-session para manejar las sesiones de forma segura, configurando correctamente las opciones de sesión, como la firma de cookies y el almacenamiento seguro de la sesión. ➤ Se recomienda utilizar tokens JWT (JSON Web Tokens) para almacenar información de sesión en el cliente y validar la autenticidad de los tokens en el servidor.
------------------------	---

RS-04 Confidencialidad: Implementación de cifrado en la comunicación (TLS/SSL)	
Versión	1.0 - 20-5-2023
Descripción	La aplicación debe utilizar un cifrado seguro (TLS/SSL) en la comunicación entre el cliente y el servidor para garantizar la confidencialidad de los datos.
Importancia	Alta
Prioridad	Alta
Estado	Implementado
Tecnología	Node.js, Express.js
Recomendaciones	<ul style="list-style-type: none"> ➤ Para la comunicación cifrada se puede utilizar el protocolo TLS (Transport Layer Security), que es una tecnología estándar para proteger la comunicación entre el cliente y el servidor. Express.js tiene soporte nativo para TLS. ➤ Utilizar la librería Bcrypt en Node.js para el cifrado seguro de contraseñas

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

	<ul style="list-style-type: none"> ➤ Para Node.js y Express.js utilizar la librería Helmet para habilitar encabezados de seguridad, incluyendo la configuración de TLS/SSL para cifrar la comunicación.
--	--

RS-05 Integridad: Implementación de tokens anti-CSRF (Cross-Site Request Forgery)	
Versión	1.0 - 20-5-2023
Descripción	Prevenir ataques CSRF mediante la generación y verificación de tokens únicos en cada solicitud que modifique el estado del servidor
Importancia	Alta
Prioridad	Alta
Estado	Implementado
Tecnología	Node.js, Express.js, MongoDB
Recomendaciones	<ul style="list-style-type: none"> ➤ Para Node.js y Express.js utilizar el middleware Express-validator para validar y sanitizar los datos de entrada del usuario, evitando ataques de inyección de código y otros. ➤ Utilizar la librería Crypto de Node.js para implementar funciones de hash y firma digital que permitan verificar la integridad de los datos ➤ Para MongoDB: Implementar las opciones de seguridad proporcionadas por MongoDB, como la autenticación de cliente y la conexión segura mediante TLS/SSL.

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

RS-06 Logging y trazabilidad: Registro de eventos y auditoría	
Versión	1.0 - 20-5-2023
Descripción	Implementar un sistema de registro de actividad en la aplicación para registrar todas las acciones realizadas por los usuarios y detectar posibles anomalías o actividades maliciosas
Importancia	Media
Prioridad	Media
Estado	Implementado
Tecnología	Express.js, Node.js.
Recomendaciones	<ul style="list-style-type: none"> ➤ Para Express.js y Node.js, se puede utilizar la librería Winston para registrar eventos y generar logs de forma estructurada, permitiendo su análisis y seguimiento.

RS-07 Validación de entrada y salida en el código fuente: Implementación de validación de entrada y salida de datos en el código fuente.	
Versión	1.0 - 20-5-2023
Descripción	Implementar medidas de seguridad para evitar la inyección de código malicioso en la aplicación mediante la validación de la entrada y salida de datos en el código fuente
Importancia	Alta
Prioridad	Alta
Estado	Implementado
Tecnología	AngularJS, Node.js.
Recomendaciones	<ul style="list-style-type: none"> ➤ Para AngularJS, se puede utilizar el módulo ngSanitize para realizar la sanitización de los datos antes de

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

	<p>mostrarlos en las vistas, evitando ataques XSS (Cross-Site Scripting).</p> <ul style="list-style-type: none"> ➤ Para Node.js, se puede utilizar la librería Helmet para implementar cabeceras de seguridad y prevenir ataques comunes, como XSS y CSRF (Cross-Site Request Forgery). ➤ Utilizar la librería Express-validator en Node.js para validar las entradas de usuario y prevenir ataques de inyección
--	--

RS-o8 Manejo de errores y excepciones: Manejo seguro de errores y excepciones	
Versión	1.0 - 20-5-2023
Descripción	Implementar un sistema de manejo de errores y excepciones para detectar y manejar errores en la aplicación de forma segura y efectiva
Importancia	Media
Prioridad	Media
Estado	Implementado
Tecnología	Node.js, Express.js
Recomendaciones	<ul style="list-style-type: none"> ➤ Para Express.js y Node.js, se puede utilizar el middleware Errorhandler para capturar y manejar errores de forma centralizada, proporcionando mensajes de error amigables para los usuarios y registrando información relevante para su posterior análisis.

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

Rs-09 Configuración segura del cliente: Establecer una configuración segura en Google Chrome	
Versión	1.0 - 20-5-2023
Descripción	Establecer una configuración segura en Google Chrome (última versión) para proteger al usuario y la aplicación web contra riesgos de seguridad y ataques conocidos.
Importancia	Alta
Prioridad	Alta
Estado	Implementado
Tecnología	Google Chrome
Recomendaciones	<ul style="list-style-type: none"> ➤ Configurar el navegador para deshabilitar complementos y extensiones innecesarios, habilitar actualizaciones automáticas, usar bloqueadores de publicidad y habilitar la navegación segura. Promover las buenas prácticas de seguridad entre los usuarios finales, como la actualización del navegador y la educación sobre la prevención de la ingeniería social y el phishing

RS-10 Configuración segura del servidor de aplicaciones: Configuración segura del servidor de aplicaciones (Node.js, Express.js).	
Versión	1.0 - 20-5-2023
Descripción	Se deben establecer configuraciones de seguridad adecuadas en el servidor de aplicaciones (Node.js, Express.js) para proteger contra ataques y riesgos de seguridad.
Importancia	Alta

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

Prioridad	Alta
Estado	Implementado
Tecnología	Node.js, Express.js
Recomendaciones	<ul style="list-style-type: none"> ➤ Para Express.js y Node.js utilizar Helmet.js que es una librería que ayuda a proteger la aplicación web de diversas vulnerabilidades, como XSS, inyecciones SQL, sniffing, clickjacking, entre otros. Se recomienda configurar la aplicación web para utilizar Helmet.js para una mayor protección.

RS-11 Cabeceras de seguridad: Configuración adecuada de cabeceras de seguridad	
Versión	1.0 - 20-5-2023
Descripción	Configurar las cabeceras de seguridad en el servidor para prevenir ataques comunes, como XSS (Cross-Site Scripting), inyección de contenido no deseado y clickjacking.
Importancia	Alta
Prioridad	Alta
Estado	Implementado
Tecnología	Node.js, Express.js
Recomendaciones	<ul style="list-style-type: none"> ➤ Utilizar la librería Helmet en Express.js para configurar las cabeceras de seguridad de manera sencilla y efectiva. Esta librería proporciona configuraciones predeterminadas seguras y permite habilitar o deshabilitar cabeceras específicas según las necesidades de la aplicación.

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

RS-12 Configuración segura del servidor de gestor de bases de datos: Configuración segura del servidor y de MongoDB	
Versión	1.0 - 20-5-2023
Descripción	Configurar el servidor y MongoDB de manera segura para prevenir accesos no autorizados y ataques a la base de datos.
Importancia	Alta
Prioridad	Alta
Estado	Implementado
Tecnología	Node.js, MongoDB
Recomendaciones	<ul style="list-style-type: none"> ➤ Node.js: Asegurarse de mantener el servidor actualizado con las últimas versiones de Node.js para obtener las correcciones de seguridad más recientes. ➤ MongoDB: Configurar adecuadamente la autenticación y autorización en MongoDB para evitar el acceso no autorizado a la base de datos. Además, aplicar los principios de seguridad recomendados por MongoDB para proteger los datos almacenados. ➤ Utilizar Mongoose.js para establecer configuraciones seguras en el servidor de MongoDB, como autenticación, autorización, encriptación, etc.

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

RS-13 Esquema de protección online: Implementar un esquema de protección en línea, como CAPTCHA	
Versión	1.0 - 20-5-2023
Descripción	<ul style="list-style-type: none"> ➤ Implementar un mecanismo de protección en línea para prevenir ataques de bots y automatizados, como el uso de CAPTCHA
Importancia	Alta
Prioridad	Alta
Estado	Implementado
Tecnología	AngularJS, Node.js
Recomendaciones	<ul style="list-style-type: none"> ➤ Para AngularJS, se puede utilizar la librería Angular-google-recaptcha para implementar reCAPTCHA como una medida de protección en línea. Esto ayudará a distinguir entre usuarios legítimos y bots. ➤ Para Node.js, se pueden utilizar librerías como Express-recaptcha o Node-captcha para integrar reCAPTCHA en el servidor y validar las solicitudes de los clientes. ➤ Utilizar servicios de protección online como Cloudflare o AWS Shield, que ofrecen capacidades de mitigación de ataques DDoS y protección avanzada de capa de aplicación.

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

RS-14 – Disponibilidad: Implementar mecanismos para garantizar la disponibilidad del servicio	
Versión	1.0 - 20-5-2023
Descripción	Implementar medidas que aseguren la disponibilidad del servicio web, evitando ataques de denegación de servicio (DoS) y mitigando posibles fallas del servidor.
Importancia	Media
Prioridad	Media
Estado	Implementado
Tecnología	Node.js, Express.js
Recomendaciones	<ul style="list-style-type: none"> ➤ Utilizar herramientas de mitigación de ataques DoS, como Cloudflare, que pueden ayudar a filtrar el tráfico malicioso y mantener la disponibilidad del servicio. ➤ Implementar técnicas de límite de velocidad (rate limiting) para controlar el número de solicitudes que un cliente puede realizar en un período de tiempo determinado.

RS-15 Política de contraseñas: Implementar una política de contraseñas seguras	
Versión	1.0 - 20-5-2023
Descripción	Establecer una política de contraseñas seguras para los usuarios, que incluya requisitos de longitud, complejidad y renovación periódica de contraseñas.
Importancia	Alta
Prioridad	Alta

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

Estado	Implementado
Tecnología	Node.js, MongoDB
Recomendaciones	<ul style="list-style-type: none"> ➤ En Node.js, utilizar la librería Password-validator para definir una política de contraseñas seguras, verificando criterios como longitud mínima, uso de caracteres especiales, números, letras mayúsculas y minúsculas, etc. ➤ En MongoDB, utilizar la funcionalidad de almacenamiento seguro de contraseñas proporcionada por Bcrypt para almacenar las contraseñas de los usuarios de forma encriptada y protegida

RS-16 Token anti-CSRF: Implementar protección contra ataques de falsificación de solicitudes entre sitios (CSRF) mediante tokens	
Versión	1.0 - 20-5-2023
Descripción	Implementar un mecanismo de token anti-CSRF para proteger las solicitudes HTTP de la aplicación
Importancia	Alta
Prioridad	Alta
Estado	Implementado
Tecnología	AngularJS, Node.js
Recomendaciones	<ul style="list-style-type: none"> ➤ Para AngularJS, utilizar el módulo Angular-token-auth o Angular-csrf para generar y gestionar tokens anti-CSRF en las solicitudes realizadas por el cliente.

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

	<ul style="list-style-type: none"> ➤ En Node.js, se recomienda el uso de la librería Csurf para validar los tokens anti-CSRF en el servidor antes de procesar las solicitudes recibidas.
--	---

3. Lista de salvaguardas:

- Implementar políticas de contraseñas seguras y almacenamiento adecuado de contraseñas en la base de datos.
- Utilizar HTTPS/TLS para la transmisión segura de datos.
- Aplicar parches y actualizaciones regulares tanto en el servidor como en el cliente.
- Validar y sanitizar la entrada de datos en el servidor para prevenir inyecciones de código.
- Utilizar técnicas de encriptación adecuadas para proteger información sensible almacenada en la base de datos.
- Limitar el acceso a recursos y funcionalidades solo a usuarios autorizados mediante roles y permisos.
- Implementar mecanismos de monitoreo y detección de intrusiones en tiempo real.
- Realizar auditorías y pruebas de seguridad de forma regular para identificar posibles vulnerabilidades.
- Mantener registros de actividad y eventos relevantes para la trazabilidad y análisis de incidentes.
- Proporcionar capacitación y concienciación en seguridad a los desarrolladores y usuarios de la aplicación.

4. Abordar el diseño de seguridad de la aplicación web:

Autenticación:

- Implementar un flujo de autenticación robusto que incluya verificación de identidad con múltiples factores.
- Utilizar técnicas de hashing y almacenamiento seguro de contraseñas.
- Aplicar políticas de bloqueo de cuentas por intentos de inicio de sesión fallidos.

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

Autorización:

- Definir y asignar roles de usuario con niveles de acceso adecuados.
- Implementar controles de acceso basados en roles en las rutas y funciones de la aplicación.

Gestión de sesiones:

- Utilizar cookies seguras y tokens de sesión con tiempo de expiración adecuado.
- Implementar mecanismos de renovación de tokens para mantener las sesiones activas.

Confidencialidad:

- Aplicar cifrado adecuado para proteger datos sensibles en reposo y en tránsito.
- Configurar correctamente la política de seguridad de la aplicación para evitar fugas de información.

Integridad:

- Implementar mecanismos de detección y prevención de ataques de manipulación de datos.
- Utilizar firmas digitales y hash de integridad para verificar la integridad de los datos.

Logging y trazabilidad:

- Registrar eventos y acciones importantes en la aplicación.
- Establecer niveles de registro adecuados y asegurar el almacenamiento seguro de los registros.

Validación de entrada y salida en el código fuente:

- Aplicar validación de entrada en todos los formularios y campos de la aplicación para prevenir ataques de inyección.

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

- Implementar sanitización y escape de salida para evitar ataques de XSS (Cross-Site Scripting).

Manejo de errores y excepciones:

- Configurar adecuadamente los mensajes de error para no exponer información sensible.
- Implementar mecanismos de manejo de excepciones para evitar filtraciones de información y errores del sistema.

Configuración segura del cliente:

- Configurar correctamente las políticas de seguridad del navegador para prevenir ataques como clickjacking y contenido mixto inseguro.

Configuración segura del servidor de aplicaciones:

- Aplicar las mejores prácticas de configuración de seguridad en el servidor web y la base de datos.
- Mantener los componentes del MEAN stack actualizados con las últimas versiones y parches de seguridad.

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

5. Esquema de protección online

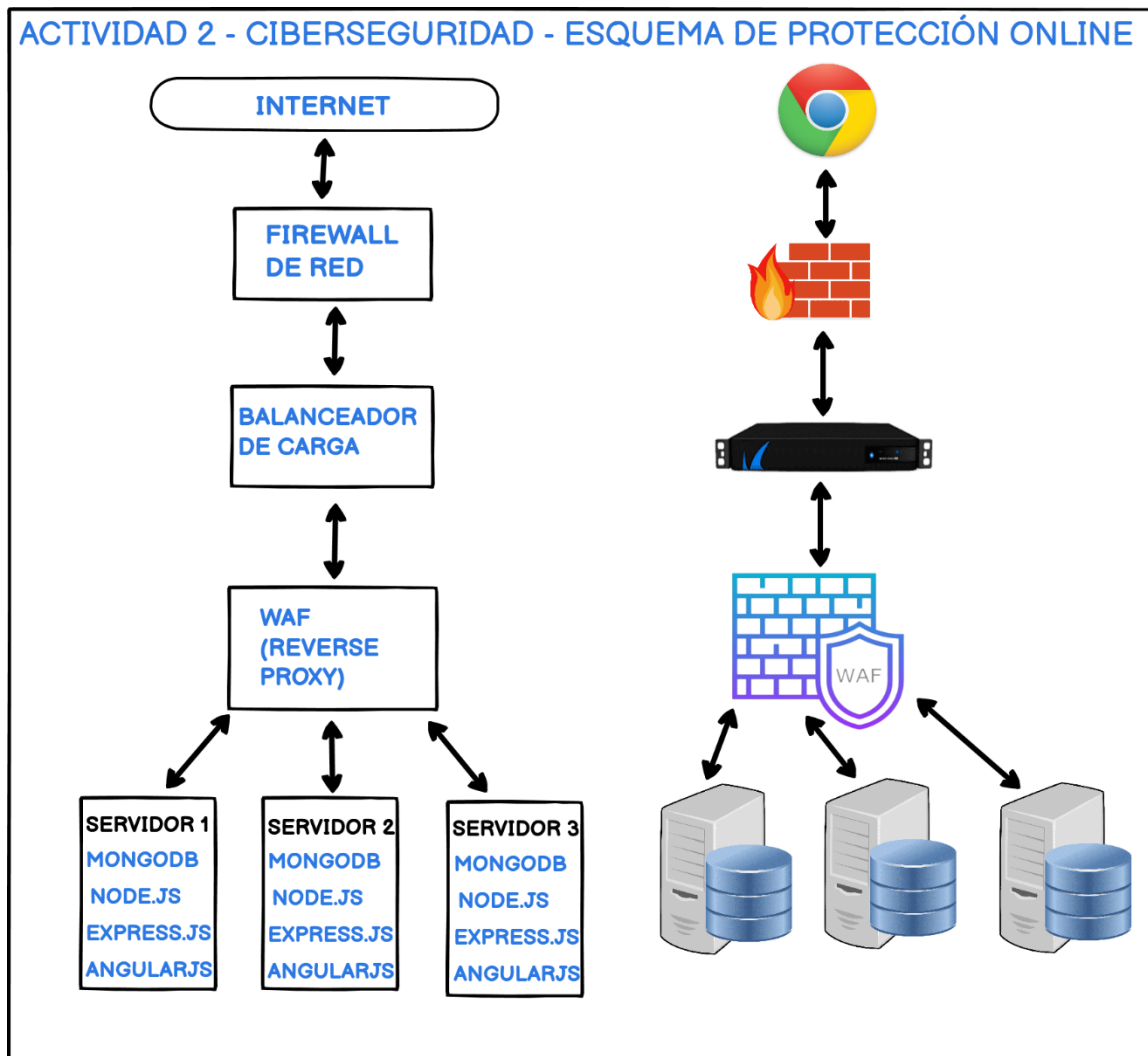


Figura 1- Esquema de protección online diseñado en Balsamiq

Funcionamiento de cada componente del esquema de seguridad:

- Internet (Chrome): Este es el punto de entrada desde el cual los usuarios acceden a la aplicación web.
- Firewall de Red: El firewall de red es la primera línea de defensa entre Internet y la infraestructura de tu aplicación. Su función principal es controlar y filtrar el tráfico entrante y saliente para proteger los servidores de posibles amenazas y ataques maliciosos.

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

- **Balanceador de Carga:** El balanceador de carga distribuye el tráfico de manera equitativa entre los servidores disponibles. Su objetivo es mejorar el rendimiento y la disponibilidad de la aplicación, evitando que un solo servidor se sobrecargue. Además, proporciona una capa adicional de seguridad al ocultar la estructura interna de la infraestructura y actuar como punto de entrada único.
- **WAF (Web Application Firewall) o Proxy Inverso:** El WAF, también conocido como proxy inverso, es un componente de seguridad que examina el tráfico entrante para detectar y bloquear posibles ataques a la aplicación web. Proporciona una protección adicional contra vulnerabilidades conocidas y desconocidas, así como contra ataques de denegación de servicio (DDoS) y otros ataques web.
- **Servidores:** Los servidores representados en el diagrama son tres instancias de servidores Mean Stack (MongoDB, Express.js, AngularJS, Node.js). Estos servidores alojan la lógica de la aplicación, la base de datos y la interfaz de usuario. Cada servidor puede procesar solicitudes individuales y se pueden escalar horizontalmente según sea necesario.
- Todos estos componentes deben ser redundantes para evitar fallos.

El esquema muestra una arquitectura de protección online para una aplicación Mean Stack. Desde el punto de entrada de Internet, el tráfico pasa a través del firewall de red, luego se distribuye de manera equitativa entre los servidores mediante el balanceador de carga. A continuación, el tráfico pasa por el WAF o proxy inverso, donde se realizan controles de seguridad adicionales antes de llegar a los servidores Mean Stack. Esta configuración ayuda a proteger la aplicación contra amenazas y asegura un funcionamiento eficiente y seguro.

6. Conclusiones:

- Se elaboró una lista de requisitos y salvaguardas de seguridad de una aplicación web.
- Se abordó el diseño de seguridad de un supuesto real de una aplicación web con tecnologías concretas.
- Se incorporó un esquema de protección online.

Asignatura	Datos del alumno	Fecha
Ciberseguridad Web	Apellidos: Santamaría Cherrez	22/5/2023
	Nombre: Jorge Patricio	

Rúbrica

Seguridad en la arquitectura MEAN Stack	Descripción	Puntuación máxima (puntos)	Peso %
Criterio 1	Compleitud de requisitos	3	30 %
Criterio 2	Adecuación de requisitos a las tecnologías	3	30 %
Criterio 3	Esquema de protección <i>online</i>	3	30 %
Criterio 4	Calidad de la memoria	1	10 %
		10	100 %