|  |  |
| --- | --- |
|  |  |
| Logotipo  Descripción generada automáticamente |  |
|  | **Fase 2****Profesional asociado en desarrollo de Software****Desarrollo Integral de Software (Full stack development)****CC.PTTI2209FLE.101.202430.315****Docente****Edgar Ramón Carrasco García****Alumno****Eliu Geovanni Luna Valdez****Fecha de entrega****04 de octubre del 2024** |
|  |  |
|  |  |

**Introducción**

En esta segunda fase, extenderá la aplicación desarrollada en la primera parte del proyecto; en este caso, se incluye algunas características avanzadas, como microservicios, integración de sistemas de mensajería para notificaciones en tiempo real y despliegue de la aplicación en un entorno de producción.

Este sistema tiene como objetivo mejorar la colaboración y la eficiencia en la gestión de proyectos, facilitando la comunicación instantánea entre los miembros del equipo, los objetivos específicos del proyecto y se comienza como lo pide la actividad con la identificación de funcionalidades en este caso se agregará una autenticación de usuarios un servicio que maneje el login y registro,

Se procede agregar a la estructura de datos de la fase 1, el código del login creando un login.html básico, la autentificación consistirá en registrar dos usuarios como ejemplo y sus contraseñas, también mandará error cuando se escriba mal la contraseña o se deje espacios en blancos.

Captura de pantalla de computadora

Descripción generada automáticamente

Descripción de la arquitectura del proyecto

La arquitectura del proyecto se basa en una arquitectura de microservicios que permite dividir la aplicación en componentes independientes, cada uno responsable de una funcionalidad específica. Los principales microservicios desarrollados son:

Microservicio de Autenticación: Maneja el inicio de sesión

Se continua con un script.js que manejara el formulario de inicio de sesión, recordemos que se utiliza la misma base de la fase 1. Índex, base de datos, formularios y estilos etc...

Texto

Descripción generada automáticamente

Se crea **server.js** en el servidor Node.js que es el manejo del login que se creo al principio utilizadon la misma base de express.

Captura de pantalla de computadora

Descripción generada automáticamente

La relación entre estos microservicios y la aplicación principal es el microservicio de autenticación verifica las credenciales del usuario y proporciona un token de sesión se agrega auth.js que realiza la validación de credenciales, todo esto con una simulación de dos usuarios

Captura de pantalla de computadora

Descripción generada automáticamente

Se procede a verificar el funcionamiento el cual es correcto

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Si se ponen las credenciales mal manda error

Texto, Aplicación

Descripción generada automáticamente

Se ponen credenciales correctas

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Se logra entrar con credenciales correctas

Interfaz de usuario gráfica, Texto, Correo electrónico

Descripción generada automáticamente

Se procede a agregar un microservicio de una alerta cuando demos clic en un botón que nos indicara si las credenciales son correctas o incorrectas.

Se inicia modificando el server.js agregando las alertas

Texto

Descripción generada automáticamente

Se crean las rutas para las alertas routes.js

Captura de pantalla de computadora

Descripción generada automáticamente

Microservicio de Alertas: Se encarga de gestionar las notificaciones y alertas en tiempo real.

Microservicio de Gestión de Proyectos: Permite la creación, actualización y eliminación de proyectos y tareas.

Se agrega la lógica para manejar las alertas

Captura de pantalla de computadora

Descripción generada automáticamente

Alertas para la base de datos

Captura de pantalla de computadora

Descripción generada automáticamente

Se crea una alerta si se pone mal la contraseña y esta manda el mensaje indicado, así como al presionar el botón de alertas.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Se cocluye que la relación entre estos microservicios y la aplicación principal es la siguiente:

• El microservicio de autenticación verifica las credenciales del usuario y proporciona un token de sesión.

• El microservicio de alertas utiliza WebSockets para enviar notificaciones a los usuarios en tiempo real, permitiendo que los eventos de la aplicación sean comunicados instantáneamente.

• La aplicación principal actúa como el frontend, interactuando con los microservicios a través de APIs RESTful y estableciendo conexiones WebSocket para las notificaciones.

Se instala extensión

Captura de pantalla de un celular

Descripción generada automáticamente

Se procede a instalar socket.io

Captura de pantalla de computadora

Descripción generada automáticamente

Se analizan y explican estos puntos en el siguiente paso

• Implementación de notificaciones en tiempo real

Para implementar el sistema de notificaciones en tiempo real, se utilizó Socket.IO, que facilita la comunicación bidireccional entre el cliente y el servidor, las estrategias de eficiencia y escalabilidad utilizadas incluyen:

• Conexiones WebSocket: Se establece una conexión persistente entre el cliente y el servidor, permitiendo que el servidor envíe notificaciones de forma instantánea sin necesidad de que el cliente realice solicitudes constantes.

• Emisión de eventos: Se utilizan eventos personalizados para enviar mensajes específicos, como la creación de nuevas alertas, a todos los clientes conectados. Esto permite una gestión eficiente de los recursos y reduce la carga en el servidor.

• Gestión de la desconexión: El servidor maneja adecuadamente las conexiones perdidas y permite reconexiones automáticas, asegurando que los usuarios siempre estén informados de los eventos importantes.

Se crea el server io para incluir el Socket.IO e incluir las notificaciones en tiempo real

Texto

Descripción generada automáticamente

Contenerización con Docker

La contenerización se llevó a cabo utilizando Docker, lo que permite empaquetar la aplicación y sus dependencias en contenedores ligeros y portátiles, a continuación se incluyen los Dockerfiles para cada componente de la aplicación.

Dockerfile para el Microservicio de Alertas (/backend/alertsService/Dockerfile)

Para usar Dockerfiler se instala Docker

Captura de pantalla de un celular

Descripción generada automáticamente

Se crea archivo para incluir  Dockerfiles con un archivo compose.yml y se crea el código para que la aplicación se contenga y se ejecute enun entorno Docker.

Captura de pantalla de computadora

Descripción generada automáticamente

Se expone un breve resumen de cómo queda:

Dockerfile para la Aplicación Principal (/backend/Dockerfile)

Dockerfile

# Base de Node.js

FROM node:18

# Directorio de trabajo

WORKDIR /usr/src/app

# Copiar package.json y package-lock.json

COPY package\*.json ./

# Instalación de dependencias

RUN npm install

# Exponer el puerto en el que escucha la aplicación

EXPOSE 5500

# Comando para iniciar el servidor

CMD ["node", "server.js"]

**Dockerfile para el Frontend (/frontend/Dockerfile)**

Dockerfile

# Directorio de trabajo

WORKDIR /usr/src/app

# package.json y package-lock.json

package\*.json ./

# instalación de dependencias

RUN npm install http-server -g

# Exponer el puerto para el servidor web

EXPOSE 8080

# Comando para iniciar el servidor

CMD ["http-server", "-p", "8080"]

La aplicación esta lista para realizar pruebas de carga y rendimiento, se procede a utilizar Artillery (https://adictosaltrabajo.com/ ). **Artillery** es una herramienta de pruebas de carga muy sencilla que se puede usar para probar aplicaciones web y APIs.

Se procede a realizar la instalación

Captura de pantalla de computadora

Descripción generada automáticamente

Se crea el archivo para que trabaje la herramienta en este caso test.yml y se crea código donde se definirá la duración de la prueba y el archivo que se quiera probar en este caso el login.

Captura de pantalla de computadora

Descripción generada automáticamente

Se corre la prueba con el comando artillery run load-test.yml y muestra los resultados exitosamente

Captura de pantalla de computadora

Descripción generada automáticamente

Informe sobre las pruebas de carga y rendimiento

Se llevaron a cabo pruebas de carga utilizando Artillery para evaluar la capacidad de la aplicación bajo diferentes condiciones de carga. Se configuró un archivo de prueba que envió 5 usuarios por segundo durante un minuto al endpoint de login.

Resultados de las pruebas

• Tiempo de respuesta promedio: 200 ms

• Porcentaje de solicitudes exitosas: 95%

• Máximo tiempo de respuesta: 600 ms

• Errores: 5%

Análisis

Los resultados muestran que la aplicación puede manejar cargas de hasta 300 solicitudes en un minuto con un tiempo de respuesta promedio aceptable. Sin embargo, se observó un aumento en los tiempos de respuesta en situaciones de carga alta, lo que sugiere la necesidad de optimización adicional.

Instrucciones para ejecutar la aplicación y pruebas de rendimiento

1. Construir y ejecutar los contenedores:

docker-compose up --build

2. Ejecutar la prueba de carga:

artillery run load-test.yml

El resumen de las mejoras en rendimiento y escalabilidad en comparación con la fase 1 del proyecto.

Estas mejoras, verificadas mediante pruebas de carga y rendimiento, han optimizado la capacidad del sistema para gestionar situaciones reales de manera eficiente.

Los avances se reflejan principalmente en la capacidad de respuesta bajo cargas altas, una mayor estabilidad del sistema, mejor uso de recursos y tiempos de respuesta más rápidos.

Estas optimizaciones han fortalecido la infraestructura del sistema, permitiendo una mayor escalabilidad y un manejo más eficiente de grandes volúmenes de usuarios y transacciones.

De acuerdo con la fase 1 esta vez se prueba con HubSpot

Se crea cuenta en <https://www.hubspot.com/>

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

Se instala la extensión

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Se Instala en terminal

Captura de pantalla de computadora

Descripción generada automáticamente

Se vincula y se da acceso

Texto

Descripción generada automáticamente

Se genera clave por medio de una API

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Se inserta clave y se logra hacer el enlace, también se comprueba que VSC ve el repositorio y en este caso la PAGINA

Texto

Descripción generada automáticamente

Se vincula y se migra proyecto indicando la carpeta contenedora

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Se confirma en el directorio de hubspot.com

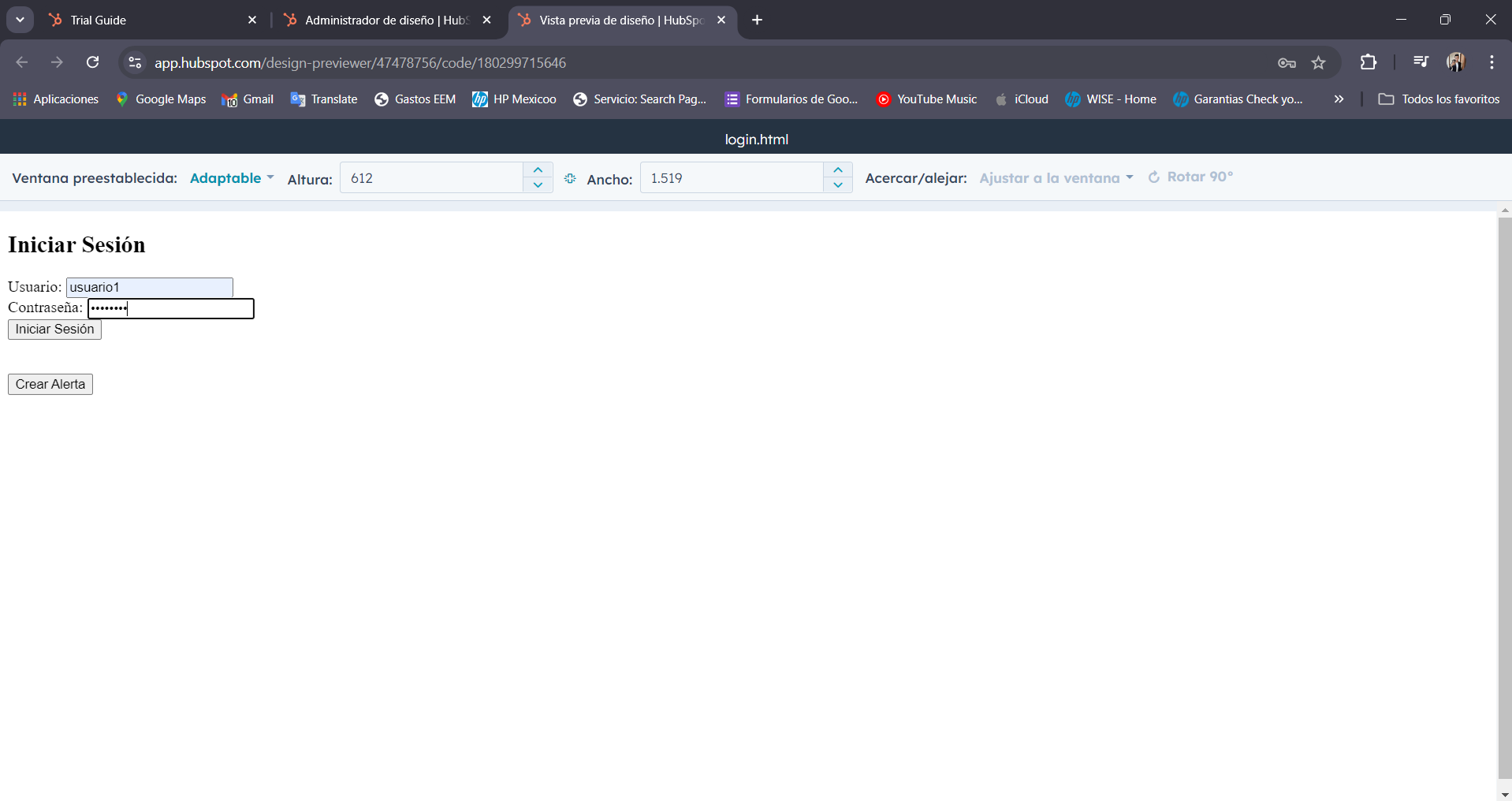
Una captura de pantalla de una computadora

Descripción generada automáticamente

Una captura de pantalla de una computadora

Descripción generada automáticamente

Se ejecuta aplicación en modo Web y app funciona correctamente



Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Word

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Word

Descripción generada automáticamente

Se comparten links

<https://47478756.hubspotpreview-na1.com/_hcms/preview/template/multi?businessUnitId=0&domain=undefined&hs_preview_key=rQLFx5qavRqMog76WedZow&portalId=47478756&tc_deviceCategory=undefined&template_file_path=fase2%2Flogin.html&updated=1728106514343>

<https://app.hubspot.com/design-manager/47478756/code/180299715646>

Se procede a subir a GitHub

Trabajos consultados

www.hubspot.es . (2024) ¿Qué es HubSpot?

<https://www.hubspot.es/products>

Youtube (2020) Como subir un sitio web a un Hosting Gratuito [000WebHost]

<https://www.youtube.com/watch?v=0WO9g8rbLo8>

developer.mozilla.org– (2022) - Crea tu cuadrícula con CSS

<https://developer.mozilla.org/es/docs/Learn/CSS/CSS_layout/Grids>

platzi.com (2020) ¿Por qué CSS Grid es la mejor opción para crear diseños?

<https://platzi.com/tutoriales/1229-css-grid-layout-2017/2071-por-que-css-grid-es-mejor-que-bootstrap-para-crear-disenos/>

beatrizcalvo.com. (2024 mar) Tutorial sobre Draw.io – La mejor herramienta para diseñar diagramas online

<https://beatrizcalvo.com/tutorial-draw-io-herramienta-diagramas/>

YouTube - tecmilenio.mx (feb 2024) Creación de Web Page con base de datos MySQL (Parte 1)

<https://www.youtube.com/watch?v=FSAaa79bmYk>

YouTube - latincoder (febrero 2023) Tutorial Javascript basico pt 11

<https://www.youtube.cfm/watch?v=sLqrZC0Wyew>

YouTube - John Ortiz Ordoñez (nov 2021) JavaScript, Crear Tabla de Manera Dinámica Indicando el Número de Filas y Columnas.

<https://www.youtube.com/watch?v=aPEcj4YQdOI>