Trabajo práctico N°5
Desarrollo Full Stack

Temas de la unidad:

a. Docker.

b. Azure.

c. Desarrollo web.

d. Máquinas virtuales.

e. Nginx.

Contexto

Imagina que has sido contratado/a por una startup tecnológica que necesita una aplicación web funcional y lista para producción. Tu rol como desarrollador/a Full Stack será fundamental para construir toda la solución desde cero. La empresa requiere un equipo ágil capaz de desarrollar tanto el backend como el frontend, asegurando la comunicación eficiente entre ambas partes y desplegando la solución final en la nube.

Objetivos principales del trabajo:

- Aplicar conocimientos reales de desarrollo web full stack.
- Aprender sobre integración y despliegue en entornos de producción.
- Entender la importancia del versionamiento de código, la contenerización y el despliegue en la nube.

Ing. JORGE, Martín – Ayudante Ad-Honorem KRAL, Franco

Puntos a desarrollar

1- Desarrollar el Backend (API CRUD).
Diseña y desarrolla un backend que exponga una API REST con operaciones CRUD (Crear, Leer, Actualizar y Eliminar). Este backend deberá estar conectado a una base de datos y responder correctamente a las solicitudes del cliente (frontend).

2 - Desarrollar el Frontend que consuma el Backend.
Crea una interfaz de usuario que consuma la API desarrollada en el paso anterior. Este frontend debe permitir al usuario interactuar con los datos (crear registros, ver listas, editar y eliminar).

3 - Escribir un Dockerfile para el Backend y otro para el Frontend.
Contenerizar ambas aplicaciones por separado. Escribe un Dockerfile para cada una, asegurando que las imágenes sean funcionales, eficientes y contengan todas las dependencias necesarias.

4 - Subir ambas imágenes a Docker Hub.
Crea una cuenta en Docker Hub (si aún no tienes una) y sube tus imágenes del frontend y backend utilizando docker push. Asegúrate de etiquetarlas correctamente (nombre_usuario/nombre_imagen:tag).

5 - Crear una máquina virtual en Azure con Ubuntu Server.
Accede a la plataforma de Microsoft Azure y crea una instancia de máquina virtual con Ubuntu Server. Esta VM actuará como tu servidor de producción.

6 - Instalar Docker y Docker Compose en la máquina virtual.
Accede a la VM por SSH y realiza la instalación de Docker y Docker Compose. Asegúrate de habilitar el servicio y verificar que esté funcionando correctamente.

7 - Descargar las imágenes desde Docker Hub.
Desde la máquina virtual, utiliza docker pull para descargar las imágenes previamente subidas a Docker Hub, tanto del frontend como del backend.

8 - Escribir un archivo docker-compose.yml para levantar los servicios.

Crea un archivo docker-compose.yml que defina tres servicios:

- Uno para el **frontend**, utilizando la imagen que creaste.
- Uno para el **backend**, también utilizando tu imagen personalizada.
- Uno para la **base de datos** (puedes usar una imagen oficial, como mysql, postgres, o mongo, según tu elección).

9 - Ejecutar el comando docker compose up -d.

Esto levantará los servicios definidos (frontend, backend y base de datos) en segundo plano. Verifica con docker ps que todos los contenedores estén corriendo correctamente.

10 - Verificar el funcionamiento de la aplicación desde el navegador

Abre un navegador web y accede a la IP pública de tu máquina virtual en el puerto donde esté expuesto el frontend. Por ejemplo: http://<IP-DE-TU-MAQUINA>:<PUERTO-FRONTEND>

Tarea opcional

1 - Aplicar un proxy inverso usando Nginx.

- Crear un archivo de configuración para Nginx que defina rutas para el frontend y el backend.
- Integrar este proxy inverso en tu archivo docker-compose.yml (si usas contenedores).
- Probar que las solicitudes sean redirigidas correctamente y que la aplicación funcione sin problemas accediendo sólo a través del proxy.

Aclaraciones importantes

- **Tema libre para el CRUD:**
  Pueden elegir cualquier temática que les resulte interesante o motivadora para desarrollar la aplicación CRUD. La idea es que trabajen en un proyecto que les guste y les permita demostrar sus habilidades.

- **Backend y frontend separados:**
  Es imprescindible que el backend y el frontend sean desarrollados como proyectos independientes, utilizando frameworks diferentes para cada uno.
  Por ejemplo, aunque Django permite crear tanto la lógica del backend como las vistas (templates) integradas, **no está permitido usar Django para ambas partes en un solo proyecto**. Deben crear un backend que exponga una API y un frontend separado que consuma dicha API.

- **Configuración de red en la máquina virtual:**
  Tengan en cuenta que para que su aplicación sea accesible desde el navegador, probablemente necesiten configurar la apertura de puertos en la máquina virtual a través del portal de Azure. Asegúrense de habilitar los puertos correspondientes para que el frontend pueda responder correctamente a las solicitudes HTTP.

Entrega

La entrega final consistirá en un **documento en formato PDF** que deberá contener la siguiente información:

1. Descripción general del proyecto
Tema elegido para la aplicación y breve explicación del propósito y funcionalidades principales.

2. Herramientas y tecnologías utilizadas
Indicar qué frameworks, lenguajes y tecnologías se utilizaron tanto en el backend como en el frontend. Mencionar también el uso de Docker, Docker Compose, Nginx (si realizaron la tarea opcional) y la plataforma de despliegue (por ejemplo, Azure).

3. Contenerización y despliegue
Incluir los Dockerfile del backend y frontend, y el archivo docker-compose.yml utilizado. Si aplicaron un proxy inverso con Nginx, explicar brevemente su configuración.

4. Conclusión
Mencionar las principales problemáticas que surgieron durante el desarrollo y cómo las resolvieron. Incluir una breve reflexión sobre lo aprendido y la experiencia general del proyecto.

5. Video demostrativo (máximo 3 minutos)
Grabar un video de hasta 3 minutos mostrando el funcionamiento de la aplicación. Puede subirse a YouTube como “oculto” y se debe incluir el enlace en el mismo documento PDF.

Notas importantes
No es necesario incluir el código completo del backend y frontend. Si lo desean, pueden destacar algunos aspectos importantes como los endpoints del backend.

Fecha de entrega: NO DEFINIDA

Ing. JORGE, Martín – Ayudante Ad-Honorem KRAL, Franco