

El objetivo de esta tarea es desarrollar una aplicación completa en tres partes, desplegarla correctamente en las plataformas correspondientes y conectar los distintos componentes para una solución funcional. La tarea se dividirá en tres ejercicios, y cada uno deberá incluir evidencias claras del desarrollo y despliegue.

- **Ejercicio 1: Creación de una aplicación básica en React o Angular y despliegue en Vercel.**

- **Requisitos:**

- Crear una aplicación básica en React o Angular que permita realizar a través de botones (por ejemplo) realizar peticiones a una API.
- La aplicación debe ser de una sola página (SPA) con un diseño sencillo pero funcional.

- **Despliegue:**

- Desplegar la aplicación en **Vercel**.
- Proporcionar la URL pública del despliegue.

- **Evidencias:**

- Capturas de pantalla del código de la aplicación.
- Capturas del proceso de despliegue en Vercel.
- URL pública de la aplicación.

- **Ejercicio 2: Creación de una API REST con Node.js y despliegue en Render**

Desarrollar una API REST sencilla con **Node.js** y **Express** que permita gestionar tareas.

- **Requisitos:**

- La API debe incluir las siguientes rutas, si fuese un ejemplo de tareas:
 - GET /tareas: Devuelve una lista de tareas.
 - POST /tareas: Crea una nueva tarea (debe incluir un título y una descripción).

➤ DELETE /tareas/:id: Elimina una tarea por su ID.

▪ **Despliegue:**

- Desplegar la API en **Render**.

▪ **Evidencias:**

- Capturas de pantalla del código de la API.
- Capturas del proceso de despliegue en Render.
- URL pública de la API.

• **Ejercicio 3: Conexión con una base de datos MongoDB alojada en Mongo Atlas**

Modificar la API REST desarrollada en el ejercicio anterior para que utilice una base de datos MongoDB alojada en **Mongo Atlas**.

▪ **Requisitos:**

- Configurar una base de datos y colección en Mongo Atlas.
- Cambiar la implementación de la API para que:
 - GET /tareas obtenga las tareas desde MongoDB.
 - POST /tareas cree tareas en MongoDB.
 - DELETE /tareas/:id elimine tareas desde MongoDB.
- Usar la librería **Mongoose** para facilitar la conexión y manipulación de datos.

▪ **Despliegue:**

- Actualizar la API desplegada en Render para que utilice la conexión a MongoDB.

▪ **Evidencias:**

- Capturas de pantalla del código actualizado.
- Capturas de la configuración de la base de datos en Mongo Atlas.

- Capturas del proceso de la realización de consulta, creación y eliminación a la API con comunicación a MongoDB.
- URL pública de MongoDB.

Entrega final

1. Un documento (PDF) con las evidencias de los tres ejercicios:
 - Breve explicación del proceso seguido.
 - Capturas de código, despliegue y pruebas.
 - URLs públicas de los despliegues.
2. Código fuente organizado en un repositorio de GitHub con carpetas para:
 - La aplicación frontend.
 - La API REST backend.

Rúbrica

1. Funcionamiento de los ejercicios (40%)

- La aplicación frontend funciona correctamente: 10%.
- La API REST funciona correctamente: 10%.
- La conexión a MongoDB funciona y permite operaciones CRUD: 20%.

2. Despliegue (30%)

- La aplicación frontend está correctamente desplegada en Vercel: 10%.
- La API REST está correctamente desplegada en Render: 10%.
- La conexión con MongoDB desde Mongo Atlas está configurada y funcional: 10%.

3. Documentación y evidencias (20%)

- Capturas claras del código, despliegue y pruebas: 10%.
- Explicación detallada del proceso seguido: 10%.

4. Organización y limpieza del código (10%)

- Código limpio y bien organizado: 10%.