|  |  |
| --- | --- |
|  | |
| **LEONARDO DA VINCI** ESCUELA SUPERIOR DE ARTE MULTIMEDIAL | |
| A-1331 | |
|  | |
| **CARRERA:** | |
| Tecnicatura Superior en Diseño y Programación Web | |
| **ASIGNATURA:** | |
| Aplicaciones Híbridas | |
| **EXAMEN:** | |
| Parcial 2 | |
|  | |
|  | |
|  | |
|  | |
| **AÑO LECTIVO:** | |
| 2024 | |
| **DOCENTES:** | |
| MARCOS GALBÁN, Camila Belén |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**CONSIGNAS EXAMEN PARCIAL 2 – APLICACIONES HÍBRIDAS**



**Objetivo:**

Desarrollar un proyecto fullStack con MERN.

**Requisitos del Proyecto:**

Para este segundo parcial, deberán desarrollar un proyecto con la temática de su elección.

A continuación, se detallan los requisitos y lo que se espera del proyecto:

**Backend (API REST)**

* **Separación del Backend y el Frontend**: El Backend debe ser una API REST.
* **Autenticación JWT**: Implementar autenticación mediante JSON Web Tokens (JWT).
* **Base de Datos**: Utilizar una base de datos (MongoDB u otra).
* **Gestión de Usuarios**: El sistema debe contar con usuarios.
* **Entidades**: Además de los usuarios, deben existir al menos otras dos entidades.
* **CRUD**: Implementar operaciones CRUD para las entidades.

**Frontend (React.js)**

* **Componentes Funcionales y Hooks**: Utilizar componentes funcionales   
  y hooks para la creación de componentes.
* **División de Componentes**: Mantener una clara división de componentes.
* **Enrutamiento**: Utilizar Routers para las distintas páginas.
* **Separación de Vistas y Lógica de APIs**: Mantener las vistas separadas   
  de la lógica de interacción con las APIs.
* **Manejo del Estado**: Gestionar correctamente el estado de la aplicación.

**Criterios de Evaluación**

**Backend**

* Implementación completa de la API REST.
* Correcta división en controladores, routers, middlewares y esquemas.
* Identificación adecuada de recursos en las URIs.
* Uso y modelado correcto de la base de datos.

**Frontend**

* Uso de componentes funcionales y hooks.
* División clara y estructurada de componentes.
* Implementación de enrutamiento para múltiples páginas.
* Separación de la lógica de APIs de las vistas.
* Gestión adecuada del estado de la aplicación.

**Ambos**

* Validación de datos tanto en el Backend como en el Frontend.
* Implementación de autenticación con JWT.
* Archivo README con información detallada sobre cómo   
  levantar el proyecto y de qué trata.

**Requisitos Mínimos**

* Funcionalidades de login y registro con contraseñas encriptadas.
* Uso de API en el Backend (Router, Controller, Middleware).
* Implementación de JWT.
* Uso de React.js (Context, Router, Componentes, State, Hooks).
* Validación de datos en Backend y Frontend.
* Consumo de API en el Frontend.

**Importante**

* El Backend y el Frontend pueden estar en el mismo repositorio  
  o en repositorios separados.
* Entregar aparte el/los archivos .env.
* Para el backend se puede optar por utilizar el del primer parcial  
  o desarrollar uno nuevo.