



Desarrollo de Aplicaciones Informáticas

Examen de NextJS con Sockets

Docente: Ing. Pablo Morandi

Ayudante: Matías Marchesi

Curso: 5to año - Informática

Duración: 2 horas cátedra (80 min)

Fecha: 23/10/25

TEMA 3

Leer con atención antes de empezar el examen:

Material permitido: Apuntes de clase, documentación oficial, búsquedas en Google SIN IA. El uso de la IA implica que **el examen sea anulado automáticamente** sin excepción y con nota 1(unos).

Formato de entrega: Subir carpeta completa del proyecto en un archivo .RAR y con el formato "5x_NEXT_APELLIDO" a Google Classroom.

POR FAVOR, respetar el formato de entrega.

IMPORTANTE: la carpeta **/node_modules** y la carpeta **./next** **NO DEBEN SER ENTREGADAS**. De ser así, **se restará 1 (uno) punto** de la nota final del examen sin excepción.

Sistema de Tareas Colaborativas en Tiempo Real

Desarrollar una aplicación de gestión de tareas compartidas que utilice **Socket.IO** para comunicación en tiempo real, **Router** para navegación, **Conditional Rendering** para mostrar diferentes estados de la interfaz y **useState** y **useEffect** para manejo de datos.

A modo de recordatorio, para crear un proyecto de NextJS utilizamos `npx create-next-app@latest` o en su defecto pueden utilizar un proyecto ya existente teniendo en consideración de borrar todo lo que no esté relacionado a la evaluación al momento de la entrega.

Ejercicio 1: Pregunta Teórica sobre CLSX

1a) ¿Para qué se utiliza la librería CLSX en React? Que ventajas tiene en comparación con CSS tradicional? Responda esta pregunta como comentario en la página `/inicio` del proyecto

Ejercicio 2: Página de Inicio con Router

Crear una página de inicio en `/inicio` que cumpla con los siguientes requisitos:

Requisitos

1. Formulario de ingreso:

- Input para el nombre del usuario (`username`) (mínimo 3 caracteres)
- Input para el ID de alumno (`alumnoId`). Este ID es el nro que a usted le corresponde de la lista del curso ordenada alfabéticamente.
 - Esta variable se utilizara luego para el evento `join_tareas` como id de sala.
- Botón "Ir a Tareas"

2. Validación y estados:

- Mostrar mensaje de error si el nombre tiene menos de 3 caracteres
- El mensaje de error debe mostrarse usando **Conditional Rendering**

3. Navegación con Router:

- Al hacer clic en "Ir a Tareas", redirigir a `/tareas`
- Pasar `username` y `alumnoId` como parámetros de ruta o query string
 - URL de ejemplo: `http://localhost:3000/tareas?username=pepito&alumnoId=4`

Ejercicio 3: Creación de componentes Tarea y NuevaTarea

3a) Crear un componente `Tarea.js` que:

1. Reciba por props: `texto` (string), `prioridad` (number) y `creador` (string)

2. Conditional Rendering:

- Si prioridad es mayor a 3: mostrar "Prioridad alta"
- Si prioridad es 3 o menor, mostrar "Prioridad normal"

3b) Crear un componente `NuevaTarea.js` que:

1. Reciba por props: `onChangeTarea` (function), `onClickCrearTarea` (function) y `onChangePrioridad` (function)

2. El componente debe tener:

- Un título "Crear nueva tarea"
 - Un Input para ingresar el texto de la tarea
 - Un Input numérico para ingresar la prioridad (valores del 1 al 5)
 - Un Botón "Crear Tarea" para enviar la tarea por el socket.
-

Ejercicio 4: Página de Tareas con Socket.IO

Crear una página de tareas en `/tareas` que implemente comunicación en tiempo real.

Nota : para probar el proyecto, deben abrirse 2 instancias con `npm run dev` , una en el puerto 3000 y otra en el puerto 3001 para poder probar todas las funcionalidades requeridas en este punto.

Nota : para probar el proyecto, utilice la consola del navegador con F12 para ver lo que llega de los eventos y eventuales errores.

Backend Provisto

Servidor: `http://10.1.5.137:4000`

Eventos Socket.IO

Eventos que ENVÍAS al servidor:

Evento	Descripción	Datos
<code>join_tareas</code>	Unirse a tu sala de tareas	<code>{ alumnoId }</code>
<code>crear_tarea</code>	Crear una nueva tarea	<code>{ creador, texto, prioridad }</code>

Eventos que RECIBES del servidor:

Evento	Descripción
<code>joined_OK_tareas</code>	Confirmación de ingreso a sala
<code>nueva_tarea</code>	Actualización cuando hay una nueva tarea
<code>tareas_completas</code>	Se completaron 5 tareas

Estructura del objeto Tarea

```
{
  id: number,
  texto: string,
  prioridad: number,
  creador: string,
  cantidadTareas: number,
  timestamp: string
}
```

Requisitos

1. Conexión con Socket.IO

- Poner la ip del backend en el hook de useSocket `http://10.1.5.137:4000`
 - Obtener `username` y `alumnoId` desde los parámetros de ruta que vienen desde `/inicio`
-

2. Unirse a la sala de tareas

- **Botón** "Unirse a la sala de tareas"
 - Al hacer clic, emitir evento `join_tareas` con `{ alumnoId }`
 - Escuchar evento `joined_OK_tareas` para validar que se unio a la sala.
-

3. Crear tareas (Componente `NuevaTarea`)

- **Input de texto** para ingresar el texto de la tarea
 - **Input numérico** para ingresar la prioridad (1-5)
 - **Botón "Crear Tarea"**
 - **Validación:** La prioridad debe estar entre 1 y 5, sino mostrar `alert` con mensaje de error
 - Si es válido, emitir evento `crear_tarea` con `{ creador, texto, prioridad }`
-

4. Escuchar actualizaciones en tiempo real

Evento `nueva_tarea` :

- Guardar los datos de la tarea actual en un estado y mostrarlos con el componente `Tarea` .
- Esto permite que ambos clientes vean las tareas del otro en tiempo real

Evento `tareas_completas` :

- Reiniciar los estados necesarios
- Mostrar mensaje de finalización

Nota : para reiniciar flujo, presionar boton `Unirse a la sala de tareas`

5. Conditional Rendering

Mostrar :

- **"Se completaron todas las tareas! Por favor presiona "Unirse a sala de tareas" para iniciar nuevo flujo"** cuando llegue el evento `tareas_completas`
 - **"Número de sala: {alumnoId}"** cuando se haya conectado exitosamente
 - Deshabilitar botón "Crear Tarea" si no hay sala activa
-

Ejercicio 5: Historial de Tareas

Agregar funcionalidad para mostrar el historial de las últimas 5 tareas creadas.

Requisitos

1. Crear un estado `historial` (array) que almacene las nuevas tareas
2. Mostrar las últimas 5 tareas con: creador, texto, prioridad y timestamp
3. Si no hay tareas en el historial, mostrar: "No hay tareas creadas."