Universidad Católica del Norte
Facultad de Ingeniería y Ciencias Geológicas
Departamento de Ingeniería de Sistemas y Computación

Ayudantes:
★ David Araya
★ Marcelo Céspedes

Introducción al Desarrollo Web/Móvil

Taller I

Primer acercamiento a los frameworks

Sistema Dumbo

Descripción:

Los sistemas de gestión de usuarios son fundamentales para las empresas que comienzan a dar sus primeros pasos en el mundo digital, es por eso por lo que la empresa **Dumbo Supermercados Ltda.** ha convocado a los estudiantes de **Introducción al Desarrollo Web/Móvil** de la Universidad Católica del Norte para realizar su nuevo sistema de gestión de clientes.

La empresa solicita a los estudiantes trabajar en su sistema **únicamente web**, para ello deber armar una **API RESTful** que permita la comunicación entre el **Frontend** y el **Backend**. Para ello la empresa solicita a los estudiantes seleccionar los **frameworks** de desarrollo de manera **personal sin limitaciones**.

La empresa indica a los estudiantes que los clientes tienen: **Nombres**, **Apellidos**, Número de **Identificación** (RUT para chilenos y DNI para extranjeros), **correo electrónico** y cantidad de **puntos** obtenidos.

El sistema debe tener un módulo de autenticación, donde los administradores de los locales puedan iniciar sesión para visualizar los datos de los usuarios en una tabla tipo **CRUD**.

Las funciones solicitadas son las siguientes:

1. **Iniciar sesión:** El sistema solicita usuario y contraseña, si las credenciales coinciden se retorna un JWT con su RUT o DNI de cliente.
2. **Visualizar usuarios:** El sistema debe permitir visualizar a todos los usuarios del sistema a excepción del administrador. Se deben desplegar en forma de tabla todos los datos indicados anteriormente de los usuarios.
3. **Crear un usuario:** El sistema debe permitir a los administradores crear un usuario nuevo en el sistema, para ello se debe solicitar los datos de los usuarios

anteriormente mencionados. En caso de que alguno de los campos sea inválido o esté vacío, el sistema debe arrojar un **mensaje de error** al usuario indicando el problema.

4. **Editar un usuario:** El sistema debe brindar la posibilidad de editar los datos de un usuario seleccionándolo en la tabla de visualización. Los datos a editar son todos los del usuario menos el **número de identificación** (RUT/DNI). En caso de que al intentar editar un usuario quede un campo inválido o vacío, el sistema debe desplegar un **mensaje de error** indicando el problema al administrador.

5. **Eliminar un usuario:** El sistema debe permitir al administrador eliminar un usuario del sistema. Cuando el usuario a eliminar sea seleccionado, el sistema debe solicitar una doble verificación al administrador preguntando si está seguro de realizar la acción, las respuestas posibles son "**SI**" y "**NO**", en caso de negarse, el usuario **no es eliminado**, en caso contrario, se debe eliminar del sistema.

6. **Buscar un usuario:** El sistema debe permitir al administrador **buscar** un usuario específico por **número de identificación** (RUT/DNI) o **correo electrónico** dentro de la misma vista. La búsqueda realiza una actualización de la tabla de visualización utilizando el principio de la función contains.

7. **Cerrar sesión:** En esta opción, el sistema debe permitir al administrador **cerrar su sesión** actual, volver a la **pantalla de inicio** de sesión y borrar el JWT. Se debe validar que no pueda volver a ingresar al **CRUD de usuarios** sin antes volver a iniciar sesión.

Consideraciones:

- La creación y edición de usuarios debe seguir las reglas:
    - El correo debe ser único en el sistema
    - El correo debe ser válido (expresión regular).
    - El RUT/DNI debe ser único en el sistema.
    - Los puntos obtenidos no pueden ser negativos
    - Todos los campos son obligatorios (no pueden estar vacíos).
- Al iniciar el proyecto debe generar una estructura inicial que incluya tanto Backend como Frontend en una sola carpeta.
- Se debe usar GitHub para versionar su trabajo, debe ser solo un repositorio para ambos componentes del taller.
- Se debe desarrollar la interfaz de usuario utilizando el framework escogido para el Frontend, además se debe integrar un framework o biblioteca de CSS como Bootstrap o Tailwind CSS para el diseño.
- La aplicación debe ser responsiva utilizando CSS y media queries o mediante Bootstrap o Tailwind CSS.
- La aplicación debe tener un diseño agradable para la vista de un usuario, debe tener un diseño elegante y vistoso. No se permite el uso de templates de terceros.
- Se deberá utilizar Json Web Token (JWT) para la autenticación y autorización.
- Las contraseñas del sistema deben estar encriptadas utilizando BCrypt o algún formato similar.
- Las credenciales del único administrador del sistema son:
    - Usuario: Ochietto
    - Contraseña: Jaqamain3pals

Entregables:

La entrega del taller debe contener:

1. Ambos componentes, Backend y Frontend en un mismo repositorio de **GitHub**.
2. Archivo **.json** de **Postman** que haya utilizado para realizar pruebas del Backend.
3. Archivo **README.md** en la raíz del proyecto con las instrucciones para levantar los proyectos de Frontend y Backend. Estas instrucciones deben estar dirigidas a un usuario que **no tiene las tecnologías** ocupadas **instaladas** en su máquina personal. La aplicación se probará en una máquina virtual de Windows 10 sin ningún programa instalado previamente.

Condiciones de entrega:

- La fecha de entrega del taller es a más tardar el viernes **24 de noviembre a las 23:59 hrs**.
    - No se aceptarán entregas fuera de plazo o cuyos "commit" de GitHub se realicen fuera de la fecha de entrega.
- La resolución del taller es de manera individual.
- Consultas sobre el taller serán respondidas vía Campus Virtual UCN y por correo electrónico a los ayudantes de la asignatura:
    - David Araya: david.araya@alumnos.ucn.cl
    - Marcelo Céspedes: marcelo.cespedes@alumnos.ucn.cl
- La entrega del taller será por medio de GitHub subiendo la URL del repositorio **privado** a la entrega en Campus Virtual e invitando a los ayudantes como colaboradores a su repositorio.
- La copia del taller será sancionada con nota 1,0 y los antecedentes del caso serán reportados a jefatura de carrera y a registro curricular.
- Si la aplicación no es posible ejecutarla siguiendo los pasos establecidos en el entregable del README.md, no se procederá a calificar y se colocará la nota mínima 1,0.

Penalizaciones:

- El no utilizar JWT implicará un descuento de 20 décimas.
- El no utilizar la convención de commits implicará un descuento de 20 décimas.
- El no encriptar contraseñas implicará un descuento de 20 décimas.
- No documentar implicará un descuento de 30 décimas.
- No invitar al ayudante David al repositorio privado antes de la entrega tendrá un descuento de 20 décimas.

Rúbrica de evaluación:

- La Tabla I corresponde a la rúbrica del taller, la nota del taller será calculada sobre el puntaje total con un 60% de exigencia.

Criterio	Excelente (7,0 - 6,0)	Aceptable (6,0 - 3,5)	Requiere Mejorar (3,5 - 1,0)
Interfaz Web	La interfaz web corresponde al menos al 85% de lo requerido por el taller.	La interfaz web corresponde entre un 60% y 85% de lo requerido por el taller.	La interfaz web corresponde a menos del 60% de lo requerido por el taller.
API	Se construye todos los EndPoint correspondientes para dar funcionalidad a los requerimientos del taller.	Falta al menos un EndPoint correspondiente para dar funcionalidad a los requerimientos del taller.	No se establecen los EndPoint para dar funcionamiento a lo requerido por el taller
Estructura del Taller	La estructura del proyecto contempla un proyecto independiente para el backend, un proyecto independiente para el frontend. Los dos proyectos se encuentran en una sola carpeta versionada en git.	La estructura del proyecto contempla un proyecto independiente para el backend, un proyecto independiente para el frontend. Los dos proyectos se encuentran en carpetas separadas versionada en git.	No se establece una estructura clara del proyecto. No se encuentra en un solo repositorio de git
Responsividad de la Página	La aplicación web es completamente responsiva y se ve bien en diferentes tamaños de pantalla.	La aplicación web es en su mayoría responsiva, pero podría mejorar en algunos aspectos.	La aplicación web no es responsiva y no se ajusta bien a diferentes tamaños de pantalla.
Implementación de Login	Los usuarios pueden iniciar sesión con su dirección de correo electrónico y contraseña. Se verifica que las credenciales proporcionadas sean válidas antes de permitir el acceso. Se maneja adecuadamente el caso de inicio de sesión fallido con mensajes de error claros.	Los usuarios pueden iniciar sesión pero no se verifica y maneja adecuadamente el inicio de sesión. No se muestran mensajes cuando el inicio de sesión falla.	No existe un login. No se verifican adecuadamente las credenciales ingresadas por el usuario. No se maneja la sesión del usuario en el sistema.

Implementación de CRUD de Usuarios	Se ha implementado un CRUD completo para la gestión de usuarios con todas las operaciones (Crear, Leer, Actualizar, Eliminar, Buscar) y es completamente funcional.	Se implementó de 3 a 4 funcionalidades requeridas en la gestión de usuarios.	Se implementó menos de 3 funcionalidades del CRUD.
Git	La aplicación web se encuentra versionada correctamente con GIT en GITHUB, Bitbucket o cualquier otro versionador de código. Se muestra un trabajo bien organizado en el repositorio. Se encuentra un README.md donde se indican claramente los pasos y requerimientos para levantar la aplicación. ***Nota: Si la aplicación no se puede ejecutar siguiendo los requerimientos y pasos establecidos NO se procederá a calificar el taller obteniendo la nota mínima de 1.	La aplicación web se encuentra versionada correctamente con GIT en GITHUB, Bitbucket o cualquier otro versionador de código, pero el repositorio puede no estar bien organizado.	La aplicación web NO se encuentra versionada correctamente con GIT en GITHUB, Bitbucket o cualquier otro versionador de código. No se muestra un repositorio.

Pesos de cada criterio y su evaluación

Componente	Descripción	Peso
Interfaz Web	El sistema web contiene lo requerido, implementando todos los requerimientos indicados en el enunciado.	10%
API	Se construyen todos los endpoints correspondientes para dar funcionalidad a los requerimientos del sistema.	10%
Estructura del Taller	La estructura del proyecto corresponde a proyectos independientes tanto para Backend y para Frontend. Se encuentran en una sola carpeta versionada en GitHub.	5%
Responsividad	La aplicación web es completamente responsiva y se ve bien en distintos tamaños de pantalla.	20%
Autenticación	El administrador puede iniciar sesión con sus credenciales correspondientes y se verifica que estas sean válidas antes de permitir el acceso. Se manejan los errores en casos fallidos con mensajes de errores hacia el usuario.	20%

CRUD de clientes	Se implementa un CRUD completo para la gestión de clientes con todas las operaciones indicadas en el enunciado.	30%
Uso de Git	La aplicación se encuentra versionada correctamente con Git en GitHub o BitBucket. Se evidencia un trabajo organizado en el repositorio. Se encuentra el README.md con las instrucciones para levantar el proyecto.	5%

Good Luck! 😊 🍀