Proyecto 1: Aplicación web tradicional

Objetivos

Objetivo general

Construir una **aplicación web tradicional con un propósito específico** utilizando para ello el framework Koa para Node.js. Esta aplicación web tendrá funcionalidades simples y que a la vez permitan una interacción fluida con el usuario, de manera tal que los estudiantes obtengan como resultado una aplicación web que perfectamente podría ser utilizada por usuarios reales.

Objetivos específicos

- Utilizar la arquitectura MVC para la construcción de una aplicación web tradicional, con ayuda de un template pre-definido construido específicamente para el curso
- Implementar manejo de usuarios (registro y sesiones)
- Implementar un frontend básico utilizando los lenguajes HTML y CSS

Contexto

Actualmente nos encontramos en una época en que nuestras interacciones se llevan a cabo mayoritariamente de forma digital, hecho que se ha visto incrementado por la crisis sanitaria en la que aún nos encontramos. Las redes sociales constituyen la plataforma que utilizamos para estas interacciones, con Instagram, TikTok, Twitter y Facebook entre las más utilizadas hoy en día en Chile.

Frecuentemente surgen nuevas redes sociales, algunas más exitosas que otras, y ocasionalmente una de ellas redefine la forma en que utilizamos estas herramientas. El año 2020, por ejemplo, fue sin dudas el año de TikTok. Así el panorama va evolucionando a medida que pasan los años.

En este proyecto los estudiantes tendrán la oportunidad de crear su propia red social, guiados por una base común que tendrá algunas restricciones, sin perjuicio de permitirles desarrollar una idea propia.

Descripción

La red social que los estudiantes deben implementar, en este proyecto, consistirá en una plataforma que permite “subir” publicaciones creadas por usuarios registrados, las cuales serán visibles para cualquier persona que acceda a la red social. La temática o enfoque de las publicaciones queda a criterio de los estudiantes (pequeños mensajes, links a fotos o videos, etc). Lo importante es que un usuario registrado pueda publicar contenido, de acuerdo a lo que su plataforma (red social) determine, es decir, el contenido publicado debe

tener **un propósito específico y definido**, que sea coherente con el resto del contenido de la plataforma.

Al ingresar a la página principal de la plataforma, se visualizarán todas las publicaciones hechas. Además, los usuarios registrados podrán colocarle “me gusta” a una publicación. Toda publicación será de acceso libre, por lo que podrá ser compartida como enlace con otras personas.

Funcionalidades

Registro de usuarios

- Un usuario sin cuenta puede registrarse en la plataforma, con lo que crea una cuenta de usuario. Para eso debe llenar un formulario de registro a definir por ustedes.

Login de usuarios

- Un usuario que posea cuenta puede iniciar sesión llenando un formulario de login. Luego de ingresar, en cada página se muestra una sección común que incluye información del usuario (nombre o correo, y link para cerrar sesión)

Crear una publicación

- Un usuario que haya iniciado sesión puede crear publicaciones llenando un formulario con los datos requeridos para una publicación
- Los campos que posea este formulario dependerán del caso particular de la temática que escoja cada grupo de proyecto para su red social.

Detalle de una publicación

- Cada publicación creada tendrá una página específica que incluya el detalle, de tal forma que sea posible compartirla en otras redes sociales o a través de otros medios digitales. Esta página tendrá una URL única y contendrá todo el detalle de la publicación.

Gestionar una publicación

- Un usuario que haya iniciado sesión puede editar la información de una publicación que haya creado previamente
- Un usuario que haya iniciado sesión puede eliminar una publicación que haya creado previamente

Feed (ingreso) de publicaciones

- En la página principal de la aplicación se deben desplegar todas las publicaciones existentes en la plataforma. Si es necesario, se deben paginar los resultados

- Al hacer click en una publicación, la aplicación debe mostrar la página de detalle de esa publicación (es decir, con su URL única).

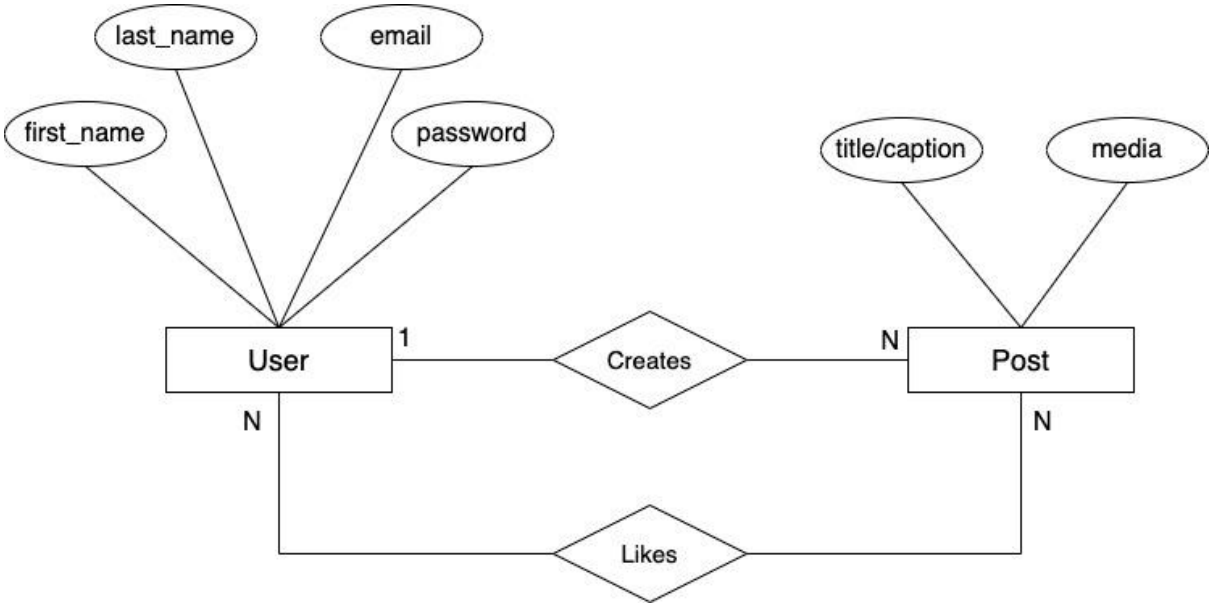
Perfil de usuario

- Un usuario que posea cuenta debe tener su propia página de perfil de usuario, donde se listen todas las publicaciones que haya creado
- Los perfiles de usuario serán públicos, por lo tanto, deben tener una URL única, al igual que las publicaciones
- De igual forma que para el feed de publicaciones, al hacer click en una publicación, la aplicación debe llevar a la página específica de esa publicación

Publicaciones guardadas

- Un usuario que haya iniciado sesión puede marcar una publicación (no propia) con un “me gusta” o “favorito”, y esta información debe ser persistida en la plataforma. En otras palabras, se puede entender como una publicación “guardada” por el usuario
- Dentro de su perfil, un usuario que haya iniciado sesión podrá ver una sección particular con todas las publicaciones que ha guardado
- Del mismo modo que para el feed de publicaciones, al hacer click en una publicación guardada, la aplicación debe llevar a la página específica de esa publicación

Modelo de datos



Entregas

Este proyecto se compone de 3 entregas: una sin nota (1.0) y dos con nota (1.1 y 1.2). Las entregas estarán separadas una de otra por un intervalo cercano a 2 semanas, a excepción de la entrega 1.2 que tendrá un plazo de 3 semanas, debido a la semana de receso de este semestre (10 al 14 de mayo).

Fechas de entregas

- **Entrega 1.0:** 15 de abril (evaluada sin nota)
- **Entrega 1.1**: 30 de abril
- **Entrega 1.2**: 20 de mayo

Forma de entrega

Todo el código será entregado únicamente a través de un **repositorio Git alojado en Github** y asociado a su grupo. El equipo docente les proveerá a los estudiantes repositorios gratuitos y privados para cada equipo.

Los ayudantes revisarán el branch main únicamente. Deben usar otras ramas durante el desarrollo y realizar merge a main sólo de lo que están seguros de querer entregar. Para la entrega 1.2 tendrán que seguir un flujo definido para hacer merge a main, mediante el uso de Pull Requests. El detalle se encuentra explicado en el enunciado de esa entrega.

Los estudiantes pueden incluir un archivo README en el repositorio del proyecto con cualquier indicación que consideren importante para el ayudante que los evaluará.

Para alojar sus aplicaciones en la Web, la plataforma soportada y recomendada será **Heroku**. Si los estudiantes desean utilizar otro proveedor pueden hacerlo, pero será bajo su propia responsabilidad.

El resultado final esperado es que tengan que realizar un deploy real de su aplicación en modo producción y que esta quede disponible para cualquier usuario de la Web, es decir, que sea pública. No cumplir con este requisito implica un castigo en la evaluación. La única excepción será la Entrega 1.0, donde sólo será necesario dejar disponibles los archivos en el repositorio correspondiente.

Entrega 1.0

Sitio web estático: HTML + CSS

En esta primera entrega los estudiantes deben concentrarse en el frontend de la aplicación. Como no existirá funcionalidad aún, lo que deben construir es un **sitio web estático, simulando las páginas principales de la aplicación final**. “Simular” se refiere en este caso a implementar páginas web ficticias y fijas (siempre se verá lo mismo cada vez que se ingrese al sitio web).

El primer paso es crear un layout (estructura) común para el sitio web y que sea compartido por todas las páginas. En general un layout incluye: header, navegación, contenido y footer. El contenido, por supuesto, es lo que varía de una página a otra, pero el resto se mantiene igual. Se puede hacer el supuesto de que un usuario ya inició sesión (y sería conveniente “nombrar” a este usuario ficticio).

En concreto, deben construir las siguientes 3 páginas web, referenciadas (“linkeadas”) entre sí:

- **Feed de publicaciones**: la lista genérica de publicaciones de la plataforma. Se debe simular un número razonable de publicaciones (más de 10). El header del sitio web debe incluir algún link al inicio, que al ser presionado, lleve a esta página con el feed.
- **Detalle de una publicación**: se requiere una sola página web. Como cada publicación debería tener su propia página, sería poco eficiente en esta etapa escribir una página web por publicación, por lo que en este caso, todo link a una publicación puede referenciar esta única página web de detalle
- **Perfil de usuario**: incluye la información pública del usuario, así como la lista de publicaciones. Deben incluir al menos 5 publicaciones asociadas a este usuario. Al igual que en el feed, es posible referenciar una única página web de detalle de publicación. La forma de acceder a esta página de perfil de usuario es haciendo click en un link que se debe encontrar en el header del sitio web (puede ser el nombre del usuario o algún ícono)

Si bien esta entrega no tiene una nota asociada, el completar lo solicitado les ayudará bastante a los estudiantes a no “arrastrar” trabajo hacia las dos entregas que sí tienen nota asociada. **Todo el trabajo que realicen en esta entrega puede ser reutilizado en la aplicación final**.

Para esta entrega, los estudiantes deben utilizar el repositorio indicado por su ayudante. **Deben incluir todos los archivos de su entrega dentro de una carpeta llamada “E1.0”**.

Cualquier duda general que tengan los estudiantes sobre el trabajo a realizar puede ser consultada en el foro del curso. Cualquier duda particular sobre su proyecto, pueden consultarla directamente a su ayudante.

Finalmente, es recomendable que trabajen en grupo y no en forma individual repartiéndose el trabajo, pues de esta manera aprenderán parte importante de lo que ocurre en el frontend. Para esto pueden organizar algunas sesiones de trabajo virtual mediante Zoom o la herramienta que más les acomode.

Entrega 1.1

CRUD de publicaciones

En esta segunda entrega los estudiantes deberán producir **la primera versión funcional de la aplicación, utilizando el template del curso**. En términos generales lo que se les pide es que tengan un CRUD (acciones, sobre un almacenamiento permanente, de: Create, Read, Update and Delete) de publicaciones, y que utilicen la asociación con usuarios.

En esta entrega **no es necesario** que la aplicación cuente con un manejo de sesión de usuarios. Las acciones que necesiten de un "usuario actualmente logueado", como por ejemplo, la creación de publicaciones, se pueden implementar mientras tanto con una selección del mismo en el formulario de creación.

En este punto no es importante que las páginas se vean del todo acabadas, aunque sería muy deseable que los estudiantes al menos pudieran respetar el layout que implementaron en la primera entrega.

Para esta entrega, los estudiantes deben utilizar el mismo repositorio indicado por su ayudante para la entrega 1.0.

Cualquier duda general que tengan los estudiantes sobre el trabajo a realizar puede ser consultada en el foro del curso. Cualquier duda particular sobre su proyecto, pueden consultarla directamente a su ayudante.

Finalmente, es recomendable que trabajen en grupo y no en forma individual repartiéndose el trabajo, pues de esta manera aprenderán todo el proceso inicial de crear una aplicación web utilizando el template del curso. Para esto pueden organizar algunas sesiones de trabajo virtual mediante Zoom o la herramienta que más les acomode.

Entrega 1.2

Aplicación final + aspectos de calidad

En esta última entrega del primer proyecto deben **finalizar su aplicación web** en cuanto a funcionalidades, incorporando manejo de sesión de usuarios y de todas las acciones que estos puedan realizar en la plataforma. Además, se evaluarán aspectos de calidad de software específicos que serán detallados más adelante.

Los requerimientos generales son los siguientes:

- Registro de usuarios y manejo de sesión (sign in/out), modificando funcionalidad existente considerando el "current user".
- Validaciones de servidor asociadas a datos y acceso de usuarios, donde sea pertinente.
- Vistas y estilos finales con CSS (SCSS). Tener en consideración que **no se deben utilizar estilos inline**.
- Cualquier funcionalidad extra que los estudiantes decidan implementar (siempre que no altere la funcionalidad requerida para este proyecto y mantenga una coherencia con lo que es su red social. No olviden usar la guía de sus ayudantes)

Los requerimientos de calidad de software son los siguientes:

- Metodología de *Code Reviews* utilizando *Pull Requests*. Los repositorios de los estudiantes estarán configurados de tal forma que sólo puedan hacer merge a main a través de la interfaz de *Pull Requests* (no será posible realizar merge directos por consola). Además se requerirá la revisión de código y aprobación de al menos un estudiante del grupo para hacer merge a main
- Análisis estático de código a través de ESLint. Los repositorios de los estudiantes estarán configurados para ejecutar ESLint cada vez que se haga un commit. Si el *build* no es exitoso, cualquier Pull Request que incluya ese commit no podrá ser *merged* en main
- Tests unitarios utilizando Jest. Los estudiantes deben incluir tests de lo siguiente:
    - Modelos (sólo métodos que no sean los típicos finders)
    - Middlewares interesantes de su aplicación
    - Cualquier función o helper que incluya su aplicación

Para esta entrega, los estudiantes deben utilizar el mismo repositorio de la aplicación web que han utilizado para las entregas anteriores.

Deben considerar, además, que **esta entrega equivale al lanzamiento de su aplicación web**, por lo que deben asegurarse de verificar detalles, como por ejemplo, incorporar datos razonables y realistas. Se les recomienda a los estudiantes **presentar un plan de pruebas a su ayudante** con anticipación a la fecha de la entrega, para que lo analice y los pueda guiar de la mejor manera, de tal forma que no entreguen su trabajo “sin saber” lo que el ayudante espera.

Cualquier duda general que tengan los estudiantes sobre el trabajo a realizar puede ser consultada en el foro del curso. Cualquier duda particular sobre su proyecto, pueden consultarla directamente a su ayudante.

Finalmente, es recomendable que trabajen en grupo y no en forma individual repartiéndose el trabajo, pues de esta manera aprenderán sobre el proceso de finalización de una aplicación web y su lanzamiento.

Evaluación

Tal como está indicado en el programa del curso, cada entrega de proyecto será evaluada con una nota de 1.0 a 7.0 siguiendo una escala discreta que se detalla a continuación:

Calificación	Nota asociada	Observaciones
A	7.0	Muy buen trabajo. La entrega excede con creces lo solicitado.
B	5.5	Buen trabajo. La entrega cumple con lo mínimo solicitado.
C	4.0	Regular. La entrega cumple parcialmente con lo mínimo solicitado.
D	2.5	Deficiente. La entrega está muy por debajo del mínimo solicitado.

En el caso de que un grupo no entregue nada, o esté fuera de plazo (habiendo utilizado la totalidad de sus cupones de atraso), la nota será de un 1.0.

Además, posterior a la calificación de cada entrega grupal, los estudiantes integrantes del grupo deberán calificarse entre sí en una **evaluación de pares**. Pueden revisar los detalles de cómo se llevará a cabo esta evaluación en el programa del curso.

Para evaluar cada entrega, todos los miembros del equipo tendrán una **reunión virtual de corrección** con el ayudante que se les haya asignado. La no asistencia (salvo justificación de peso) de alguno de los integrantes implicará un **descuento de 0.2** para ese estudiante **en el ponderador por evaluación de pares** en esa entrega (no 2 décimas).

El avance logrado en cada entrega se evaluará en base a lo subido al branch main del repositorio Github en la fecha y hora límite de la entrega, y a lo mostrado en Heroku en el momento de la reunión con el ayudante. Se verificará que el último commit en Heroku corresponda al último commit en Github antes de la fecha y hora límite.

Finalmente, las entregas 1.1 y 1.2 (luego de evaluación de pares) serán **promediadas** y eso generará la calificación de este primer proyecto (P1).

Consideraciones y restricciones

- La aplicación web debe desarrollarse en Node.js y koa **utilizando el template del curso**.
- El templating engine soportado será EJS (con HTML), pero si lo desean podrán usar otros lenguajes de templates bajo su responsabilidad.
- Podrán usar SASS/SCSS, less, stylus u otros preprocesadores de CSS.
- NO está permitido el uso de lenguajes que compilen a JavaScript, como CoffeeScript o TypeScript.
- Tampoco está permitido usar librerías de componentes HTML/CSS/JS como Twitter Bootstrap o Zurb Foundation, salvo una que otra excepción (aquí deberán consultar en el foro). Pero si quieren, ¡vean su código fuente y úsenlos de inspiración!
- **Cualquier *package* o librería adicional que quieran usar debe ser aprobado por el equipo docente. Pregunten a través del foro del curso si tienen intención de utilizar alguna**.
- Deberán usar Jest para los *tests* unitarios.
- Es recomendable escribir código en inglés. Sacarán el máximo provecho de las convenciones que puedan haber y el código quedará en armonía con el lenguaje de programación que usen (que también estará en inglés).
- La interfaz de la aplicación web puede estar en inglés o en español.