Programación 2 >

>Tecnicatura Universitaria en Desarrollo de Software

UPATECO
UNIVERSIDAD PROVINCIAL
ADMINISTRACIÓN • TECNOLOGÍA • OFICIOS

Ministerio de Economía
y Servicios Públicos
Gobierno de Salta

academia
CIMNE IBER

Proyecto Integrador

Requerimientos generales

Se busca desarrollar una **aplicación web** para un sistema de mensajería, similar a Discord.

Dicha aplicación debe permitir registrar **usuarios**, los cuales podrán crear o unirse a uno o más **servidores**.

Un servidor es un espacio que puede contener usuarios y a su vez **canales**. Un canal puede ser creado dentro de un servidor en concreto, y únicamente por un usuario perteneciente a dicho servidor.

Cada canal define el nombre de un único **chat**, el cual es un registro histórico de los mensajes enviados por los **usuarios**.

Ejemplos:

Servidor	Canales	
Literatura Fantástica	#Club de lectura	Canal donde los amantes de la literatura fantástica pueden discutir y compartir sus lecturas.
	#Escritores emergentes	Este canal está dedicado a los escritores principiantes que desean compartir y recibir comentarios sobre sus obras.
	#Recomendaciones	Este canal está pensado para recomendar y descubrir nuevas obras de fantasía.

Servidor	Canales	
Programación Web	#Preguntas técnicas	Este canal está pensado para plantear dudas y obtener respuestas sobre programación web.
	#Proyectos compartidos	En este espacio los desarrolladores pueden compartir y discutir sus proyectos web.
	#Recursos útiles	En este canal se pueden compartir enlaces tutoriales y recursos útiles relacionados con programación web.

Especificaciones para la aplicación web

La aplicación deberá cumplir las siguientes funcionalidades:

1. **Crear usuarios**, para lo cual se solicitará el nombre del usuario, contraseña, y cualquier otro dato que considere necesario. La imagen de perfil sólo podrá elegirse posteriormente a la creación del usuario.
2. **Iniciar la sesión de un usuario**. En caso de que este no exista, deberá indicarse dicho hecho y sugerir que se registre en la aplicación.
3. Una vez iniciada una sesión, se deben mostrar tres columnas en la pantalla principal:
    a. Una primera columna debe mostrar un **listado con los servidores a los que el usuario pertenece**. Si no pertenece a ningún servidor, entonces mostrará un mensaje indicando esto. Por defecto , ningún servidor estará seleccionado. Sin embargo, al seleccionar uno, se deberá cargar una segunda columna con los **canales** que posea ese servidor. Además, esta columna debe tener un botón para **crear un servidor nuevo**.
    b. La segunda columna debe mostrar un **listado de los canales del servidor seleccionado**, si no tiene ningún canal creado muestra un

mensaje indicándolo. Por defecto , ningún canal estará seleccionado. Sin embargo, al seleccionar uno, se deberá cargar una tercera columna con los **mensajes del chat de ese canal**. Además, esta columna debe tener un botón para **crear un canal nuevo**.

   c. La tercera columna mostrará los **mensajes ordenados cronológicamente**, con el más reciente en la parte inferior del chat. Si no hay ningún mensaje en el chat mostrará un mensaje indicando este hecho. Por supuesto, esta columna debe contar con un cuadro de texto para escribir un nuevo mensaje.

4. Los mensajes de un chat sólo pueden ser modificados o eliminados por el usuario que los ha creado.
5. Debe contar con un componente que permita mostrar el **perfil del usuario logueado**. En el perfil del usuario se podrán actualizar los datos personales del usuario, incluyendo la imagen del mismo.
6. Adicionalmente, la aplicación deberá implementar manejadores de errores personalizados para los siguientes casos:
   a. 400, Bad Request.
   b. 404, Not Found.
   c. 403, Forbidden. Para aquellas peticiones donde no se tenga permisos de acceso o modificación. Por ejemplo, al intentar eliminar un mensaje del chat de otro usuario.
   d. 500, Server Error.
7. La aplicación deberá contar con un **buscador de servidores** (se buscará por el nombre del servidor). En este componente se mostrarán todos los servidores que coincidan con la búsqueda realizada, para cada resultado se debe mostrar el nombre del servidor, la descripción del servidor (si la tuviera) y la cantidad de usuarios registrados en él (siempre tiene al menos un usuario registrado, quien lo creó).
8. Se deberá administrar la sesión de un usuario, es decir, registramos la información de un usuario inicie una sesión, y el acceso a los *endpoints* de la API REST diseñada deberá estar restringida sólo a usuarios logueados.

9. Notificaciones e invitaciones a un servidor.

Condiciones de aprobación

El proyecto tendrá un puntaje entre 0 y 100 puntos, y se aprobará con un total de 60 puntos. Para alcanzar este puntaje de aprobación (60 puntos) deberán implementarse, **de manera obligatoria**, **todas las funcionalidades 1 a 5** enunciadas anteriormente. Para obtener un puntaje superior a 60, se tomarán en cuenta cualquier otra de las funcionalidades adicionales (6 a 9) implementadas.

Modalidad

1. Para el desarrollo de la aplicación web será necesario diseñar:
    - Una interfaz con la que los usuarios de la aplicación interactúen, implementada con HTML, CSS y Javascript.
    - Una REST API que pueda consumir la interfaz mediante *fetching*. La cual debe ser implementada usando **Flask,** haciendo uso del patrón de diseño MVC.
    - Una base de datos en MySQL, para manejar toda la información de la app.
2. Durante los **módulos 2** y **3** de la materia, la presentación de trabajos prácticos se realizará de manera grupal. Estos mismos grupos, de **2 a 4 integrantes** de la misma comisión, deberán presentar también el proyecto integrador según se indica en el cronograma de la materia.
3. Cada grupo deberá realizar una presentación del proyecto integrador explicando la aplicación desarrollada.
4. El desarrollo del proyecto se realizará mediante la plataforma GitHub para tener un seguimiento del avance.
5. Para la presentación del proyecto **se aceptarán los cambios efectuados hasta el miércoles 27 de septiembre**. En la misma se podrán emplear diapositivas, videos, y/o cualquier recurso audiovisual que consideren conveniente para explicar el desarrollo y funcionalidad de su proyecto.

Criterios a tener en cuenta

- No se admitirá el uso de Flask-SQLAlchemy para manejar base de datos en el proyecto integrador. Las consultas a MySQL deben realizarse mediante python-mysql.connector.
- No se admite el uso de frameworks de CSS, como Bootstrap, Tailwind.
- No se admite el uso de frameworks de JS, como React, Angular, Vue.
- No se admite el uso de frameworks de Backend que no sean Flask.
- **Se adjuntará un mockup a modo de ejemplo de la interfaz a desarrollar.**
- **La disposición de la interfaz puede diferir de la mostrada en el mockup de ejemplo, siempre y cuando se cumpla con la funcionalidades solicitadas. Por ejemplo, las columnas pueden tener una orientación distinta a la mostrada, ocultarse dentro de un menú, etc.**