INSTITUT ESTEVE TERRADAS I ILLA

DESARROLLO DE APLICACIONES WEB

TRABAJO DE FINAL DE GRADO

CREACIÓN DE UN BOT CALENDARIO

EN DISCORD

Jesús Serrano Pérez

Cornellá de Llobregat, 2021

Indice

Capitulo 1. Introducción

JUSTIFICACIÓN DEL PROYECTO

La intención de este trabajo final es la implementación de un calendario completamente funcional en la plataforma social Discord, que permite la creación e implementación de aplicaciones y bots a través de su API.

Con dicha aplicación, se podrá agilizar la comunicación en grupos, ya sea en un ámbito más profesional o informal, puesto que no haría falta el uso de aplicaciones externas para poder crear y gestionar eventos, pues se podrá hacer desde la misma plataforma que ya están usando.

FORMULACIÓN DEL PROBLEMA

Durante la pandemia de COVID-19, nos hemos percatado de un problema que llevamos arrastrando desde hace ya mucho tiempo, y es que tendemos a utilizar herramientas inconexas entre ellas, o que no acaban de cumplir con la función para la cuál fueron creadas.

Un ejemplo sobre este problema, es el utilizar Discord cómo medio de comunicación entre docentes y estudiantes, pero en vez de utilizar las herramientas que proporciona la plataforma, se utiliza un calendario externo (Moodle), que complica la gestión de tareas por parte del alumnado al no tener recordatorios, o también se ha optado múltiples veces por dejar escrito en un mensaje la fecha de una entrega, sin ningún tipo de recordatorio.

Es así cómo llegué a la idea de solventar dicho problema creando una aplicación con las herramientas de desarrollo que proporciona Discord, la principal plataforma de comunicación utilizada durante dicha pandemia.

OBJETIVOS

El principal objetivo de este trabajo consiste en desarrollar una aplicación calendario. Dicha aplicación consistirá en:

1. Un bot en la plataforma Discord.
2. Una página web dinámica que permita gestionar los eventos del bot.

El bot: La principal funcionalidad del bot consistirá en la creación, visualización y alerta de eventos, de la forma más cómoda para el usuario posible.

La página web: La principal funcionalidad de la plataforma web, será visualizar de forma más cómoda los eventos creados, a demás de poder editar y eliminar estos mismos. También proporcionará el enlace de invitación del bot para añadirlo a cualquier servidor dónde la persona tenga los permisos necesarios.

Ambas plataformas compartirán una base de datos, la cuál será la encargada de guardar los eventos, usuarios, y la información necesaria tanto para el bot, cómo la para el sitio web.

METODOLOGÍA

Para llevar a cabo dicho proyecto, se hará uso de tecnologías ágiles, específicamente el método Scrum, ya que permite poder ver el estado del proyecto de forma ágil y eficaz, permitiendo hacer los cambios pertinentes a la programación con tal de poder crear el mejor producto posible dentro del marco de tiempo establecido.

RESULTADOS ESPERADOS

Se espera que este proyecto una vez finalizado, pueda ser de utilidad para todo aquél que requiera de una herramienta para gestionar eventos dentro de Discord.

Capitulo 2. Uso del bot

USO EN SERVIDOR

Cualquier persona puede usar el bot, aunque para ello primero hay que añadirlo al servidor. Esto se puede realizar fácilmente desde el botón “añadir a Discord” que se puede encontrar en la página web https://telek.tk. Importante recordar que para añadir un bot a un servidor, se debe tener permisos de administrador en dicho servidor.

Una vez el bot se encuentre dentro, se puede utilizar con el prefijo “~cal”. En caso de no saber el prefijo, se puede consultar mencionando al bot[1].

USO EN MENSAJES PRIVADOS

Al igual que en servidor, el bot también se puede utilizar en mensajes privados. Para ello, hay que hacer clic secundario sobre este en un servidor, y seleccionar “Mensaje”.

Todos los comandos son comunes, es decir, funcionan igual en servidor que en privado. También es necesario utilizar el prefijo para hacer uso del bot.

COMANDOS: PING

El bot dispone de la opción de ping. Este comando devuelve la latencia existente entre la API de Discord y el bot.

COMANDOS: AYUDA

El comando de ayuda proporciona un listado con todos los comandos disponibles, agrupados en 3 secciones: utilidades, eventos y todo.

Utilidades agrupa los comandos ping, ayuda, invitacion y version.
Eventos agrupa los comandos añadir y eventos
Todo es un único comando que muestra todos los comandos disponibles en un único mensaje.

COMANDOS: INVITACION

El comando de invitación devuelve el enlace necesario para poder añadir el bot a un nuevo servidor. Al igual que desde la página web, es necesario tener permisos de administrador en el servidor de destino para poder efectuar esta operación.

COMANDOS: VERSION

Un comando muy simple que devuelve la versión actual sobre la que se está ejecutando el bot. Útil para comprobar que la versión en producción sea la correcta.

---

[1] Para mencionar a alguien en Discord, hay que escribir @Nombre#Discriminador. En el caso del bot, @EzCal#8072. También se puede hacer con la ID de usuario <@!826102310492831804>.

COMANDOS: AÑADIR

El comando añadir permite crear un nuevo evento, con los datos pertinentes a la base de datos. Cuando llegue la hora designada en el evento, se enviará un mensaje en el mismo lugar dónde fue creado, recordando dicho evento.

COMANDOS: EVENTOS

El comando eventos obtiene todos los eventos del servidor, y los muestra en formato de lista, indicando la fecha, el título del evento y la descripción de este.

Capitulo 3. Uso de la página web

LANDING PAGE

La “landing page”, o página inicial es la primera que veremos por lo general. Desde aquí, podemos iniciar sesión, añadir el bot a un servidor, e ir a las secciones de “documentación” y “sobre nosotros”.

DOCUMENTACIÓN

La pagina de documentación nos permite entender de forma muy resumida en qué consiste el proyecto, y contiene enlaces a los repositorios del bot, de la página web, y a la wiki del bot. También muestra las tecnologías empleadas en el proyecto.

SOBRE NOSOTROS

La sección “sobre nosotros” hace un resumen en el concepto del proyecto, y el propósito de este. También muestra múltiples vías de contacto.

LOGIN

La opción de login, permite a los usuarios iniciar sesión en el sitio web utilizando su cuenta de Discord, por lo que no tendrá que hacer un registro en el sitio web. Primero redirige al sitio web de Discord, y luego es redirigido de nuevo al sitio web, sólo que esta vez a la sección de eventos.

EVENTOS

En la sección de eventos o “dashboard”, podemos visualizar todos los servidores a los que pertenecemos, a demás de una sección al principio del todo con eventos privados. Al entrar en cualquiera de estas secciones, se cargará la vista de calendario.

CALENDARIO

En la vista de calendario, se muestran todos los eventos dentro de un calendario, a demás del nombre de servidor o de usuario en caso de ser los eventos privados.

Las vistas del calendario se pueden alternar entre mensual, semanal y diario.

Capitulo 4. Backlog

SPRINT 1

El sprint 1 comenzó el 26 de Abril, y acabó el 30 de Abril.

Total de horas planificadas: 25h
Total de horas reales: 18h

Tareas planificadas:

- Documentación sobre DiscordJS (6h)
- Crear entornos de producción y desarollo (2h)
- Crear base de datos (3h)
- Crear aplicación en Discord e inicializar el bot (1h)
- Crear comandos para el bot (6h)
- Hacer que el bot funcione en servidor y mensajes privados (1h)
- Implementación del evento de avisos (6h)

Tareas pendientes: Nada

SPRINT 2

El sprint 2 comenzó el 3 de Mayo, y acabó el 7 de Mayo.

Total de horas planificadas: 25h
Total de horas reales: 21h

Tareas planificadas:

- Traducir comandos del bot a español y arreglar comando de ayuda (1h)
- Crear landing page para el sitio web (6h)
- Social Login (Discord) (2h)
- Enlace de invitación del bot a Discord (2h)
- Crear wireframes (2h)
- A partir del token proporcionado por Discord, conseguir los datos de usuario (OAuth2) (6h)
- Crear página principal con un enlace a cada servidor en el que esté el bot (6h)

Tareas pendientes:

- Crear página principal con un enlace a cada servidor en el que esté el bot (6h)

SPRINT 3

El sprint 3 comenzó el 10 de Mayo, y acabó el 14 de Mayo.

Total de horas planificadas: 24h
Total de horas reales: 21h

Tareas planificadas:

- Crear página principal con un enlace a cada servidor en el que esté el bot (2h)
- Añadir información del usuario al header (3h)
- Añadir calendario a la página (2h)
- Añadir calendario con los eventos personales (5h)
- Añadir eventos al calendario del servidor (3h)
- Crear página de información (about us) (2h)
- Crear documentación (3h)
- Crear página de documentación (página web) (2h)
- Añadir información del usuario que ha creado el evento a la página (2h)

Tareas pendientes:

- Crear documentación (3h)
- Añadir información del usuario que ha creado el evento a la página (2h)

SPRINT 4

El sprint 4 comenzó el 17 de Mayo, y acabó el 19 de Mayo.

Total de horas planificadas: 20h
Total de horas reales: 12h

Tareas planificadas:

- CRUD eventos web (6h)
- Crear documentación (3h)
- Añadir información del usuario que ha creado el evento a la página (2h)
- Preparar presentación (6h)
- Subida final a producción (3h)

Tareas pendientes:

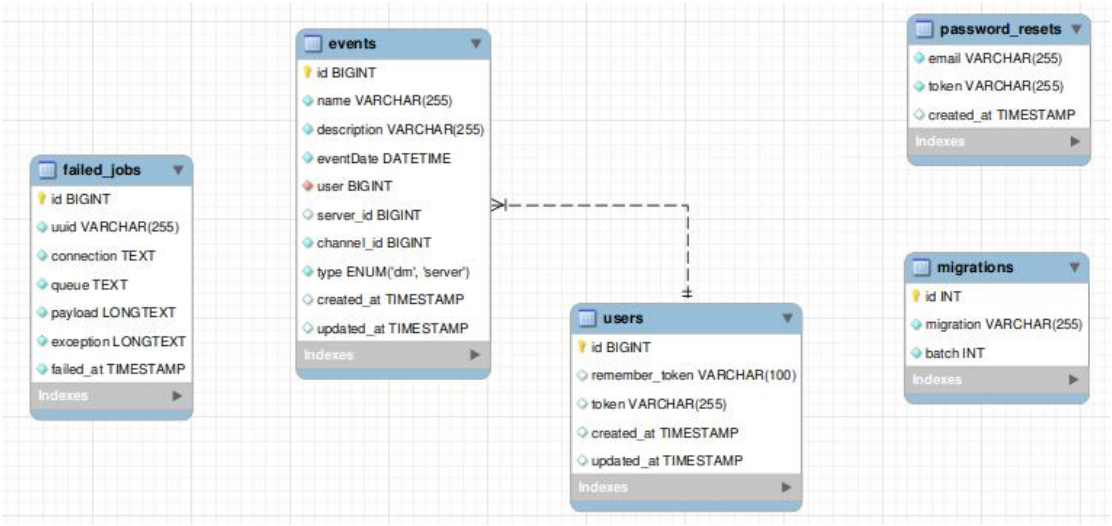
- CRUD eventos web (6h)
- Añadir información del usuario que ha creado el evento a la página (2h)

Capitulo 5. Base de datos

TIPO DE BASE DE DATOS

La base de datos funciona sobre MySQL, y ha sido creada utilizando Laravel. La base de datos se encuentra en la misma máquina, junto con el bot y la página web.

ESQUEMA



Esquema de la base de datos MySQL.

Capitulo 6. Manual de producción

PONER EL PROYECTO EN PRODUCCIÓN

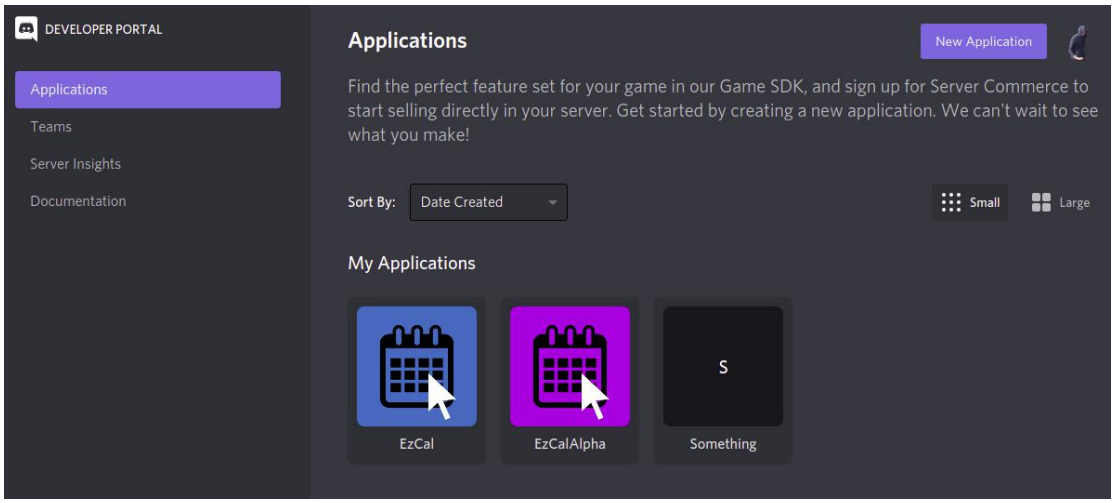
El proyecto completo se encuentra dividido en dos repositorios diferentes, uno con el código fuente del bot, y otro con la plataforma web.
Lo primero de todo, necesitaremos descargarnos ambos repositorios. Para ello, abrimos una terminal con git, nos movemos a carpeta de destino utilizando el comando cd, y a continuación, ejecutamos las siguientes sentencias:

git clone https://github.com/JesusSePe/EzCal.git
git clone https://github.com/JesusSePe/laracal

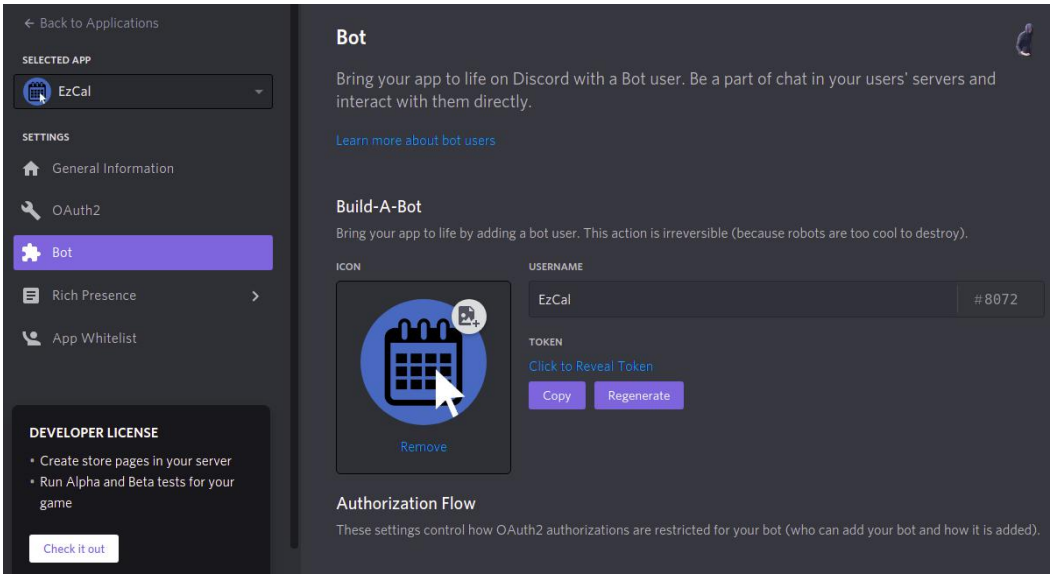
Una vez hecho esto, tendremos ambos repositorios instalados en nuestra máquina local. Lo siguiente que necesitamos, es tener una base de datos funcionando, para ello, instalamos MySQL en la máquina en caso de ser necesario, y accedemos al sistema gestor de base de datos, dónde crearemos una nueva base de datos con “CREATE DABATASE calendario”.

A continuación, necesitaremos crear una aplicación desde el portal de desarrollo de Discord (https://discord.com/developers/applications), y seguimos los siguientes pasos:

1. Click en “Nueva Aplicación”
2. Introducimos el nombre de nuestra aplicación (NO es el mismo nombre que tendrá el bot)
3. Hacemos click sobre nuestra nueva aplicación
4. Nos vamos a la pestaña “Bot”
5. Hacemos click en “Añadir Bot”
6. Editamos el nombre del bot y su imagen (opcional)



Portal de desarrolladores de Discord.



Pestaña bot de una aplicación. En la imagen se pueden ver los campos de icono, nombre y token del bot.

Ya creada la base de datos, y con nuestra aplicación en Discord creada, nos dirigimos a la carpeta EzCal, y duplicamos el archivo “index.js.example”, con el nuevo nombre de “index.js”.

Una vez clonado, editamos la sección de base de datos, empezando en la línea 16 del archivo “index.js”. Hay que cambiar los siguientes valores:

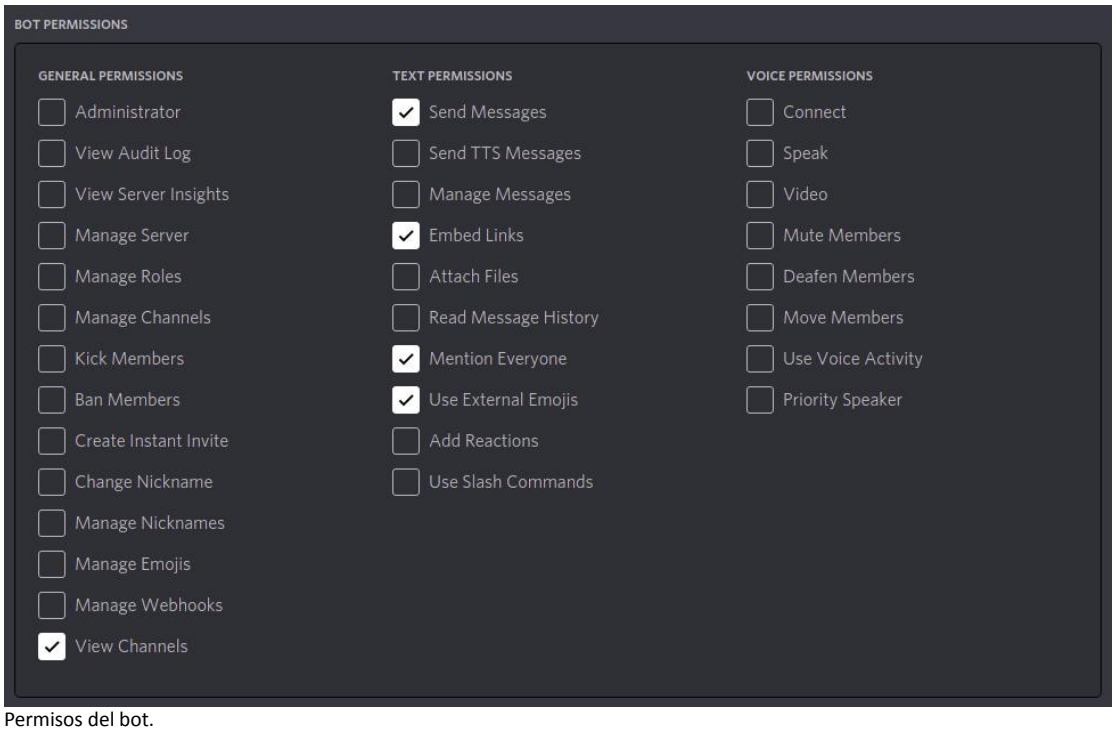
host : ‘localhost’,
database: ‘calendario’,
user: [nombre del usuario de la base de datos],
password: [contraseña del usuario de la base de datos]

```
// DB
const mysql = require('mysql2');
const pool = mysql.createPool({
    host     : 'host',
    database : 'DB',
    user     : 'user',
    password : 'password',
    waitForConnections: true,
    connectionLimit: 10,
    queueLimit: 0,
    supportBigNumbers: true,
    bigNumberStrings: true
});
```

Sección de conexión a base de datos del bot

También habrá que editar el enlace de invitación (línea 12 en index.js) por el enlace proporcionado en la sección de OAuth2, al seleccionar “bot” en “scopes” al final del todo. Los permisos necesarios son:

- View Channels
- Send Messages
- Embed Links
- Mention Everyone
- Use External Emojis



Permisos del bot.

Y por último, en la línea 53, hay que cambiar dónde pone “ID” por la id de nuestro bot, podemos encontrarla en el portal de desarrollo, en “General Information”, es la application id.

Al igual que con index.js, habrá que duplicar y editar el archivo config.js.example, dentro del cuál introduciremos el token de nuestro bot para poder acceder a Discord. Dicho token se consigue en la pestaña bot de nuestra aplicación en el portal de desarrollo de Discord, en la sección “Token”, justo debajo del nombre del bot. Copiamos este, y lo pegamos en el archivo dónde pone “DISCORD BOT TOKEN”.

Con todo esto, estamos listos para poner el bot a funcionar una vez tengamos la base de datos (se hará más adelante) con el comando node index.js, o en el caso de que queramos que se mantenga en pié indefinidamente, podemos usar pm2 start index.js (es necesario instalar pm2 previamente para poder ejecutar este comando).

Con el bot funcionando, es hora de poner a funcionar el sitio web. En estas instrucciones se especifica cómo poner a funcionar el sitio web en local, en caso de querer publicarlo, será necesario obtener un dominio, y modificar la configuración de Nginx/Apache2 para que muestre los archivos de la carpeta “laracal/public”.

Lo primero que haremos, será clonar el archivo “.env.example”, y en el clon editar el nombre a “.env”. Una vez hecho esto, editaremos los datos entre las líneas 11 a 15.

El host será localhost, la base de datos calendar, y especificamos el nombre y contraseña del usuario de la base de datos.

Ahora, en una terminal ejecutamos “composer update”, para generar los archivos faltantes, este proceso puede tardar un poco.

Crearemos una nueva clave de aplicación con el comando “php artisan key:generate”, migramos todo a la base de datos con “php artisan migrate” y ya podemos iniciar el bot si queremos.

Con todo esto, nos quedaría actualizar los enlaces del sitio web por los nuestros. El sitio web lo podemos ejecutar en local mediante el comando “php artisan serve”.

Capitulo 7. Documentación API

API DE DISCORD EN EL BOT

El bot necesita conectarse a la API de Discord, y para ello utiliza la librería de DiscordJS. Los principales endpoints que utiliza son:

- Client: Crea una nueva conexión con la API de Discord, y recibe todos los eventos que puedan afectar al bot.
- Ready: Este evento notifica al bot cuando la conexión con Discord se ha establecido correctamente y puede empezar a operar.
- Message: Este evento notifica al bot cuando llega un nuevo mensaje de cualquier persona. Luego el bot filtra que no provenga de otro bot, y que dicho mensaje empieza con el prefijo especificado, para así asegurar que el bot sólo funciona cuando debe.

API DE DISCORD EN EL SITIO WEB

En el caso del sitio web, se utiliza la API de Discord con OAuth2, el mecanismo de verificación para evitar cualquier uso no autorizado de la API. Los principales endpoints que utiliza son:

- oauth2/authorize: Primer paso de verificación. El usuario debe autorizar que la aplicación acceda a sus datos.
- oauth2/token: Segundo paso de verificación. Se envían los datos privados de la aplicación para autentificar la conexión, y devuelve un token de usuario para poder hacer las siguientes llamadas a la API.
- users/@me: Devuelve la información del usuario que se ha registrado.
- users/@me/guilds: Devuelve la información de los servidores que forma parte el usuario registrado.

Capitulo 8. Manual de usuario

INTRODUCCIÓN

Este es un bot calendario cuyo propósito es el de facilitarte la vida a la hora de organizar tus eventos, tanto privados cómo con amigos. No necesitas ningún tipo de aplicación externa, pues funciona desde Discord. Para utilizarlo, simplemente añade el bot a tu servidor favorito, y empieza a usarlo con el prefijo ~cal.

COMANDOS

Ping:

El bot dispone de la opción de ping. Este comando devuelve la latencia existente entre la API de Discord y el bot.

Ayuda:

El comando de ayuda proporciona un listado con todos los comandos disponibles, agrupados en 3 secciones: utilidades, eventos y todo.

Utilidades agrupa los comandos ping, ayuda, invitacion y version.
Eventos agrupa los comandos añadir y eventos
Todo es un único comando que muestra todos los comandos disponibles en un único mensaje.

Invitacion:

El comando de invitación devuelve el enlace necesario para poder añadir el bot a un nuevo servidor. Al igual que desde la página web, es necesario tener permisos de administrador en el servidor de destino para poder efectuar esta operación.

Version:

Un comando muy simple que devuelve la versión actual sobre la que se está ejecutando el bot. Útil para seguir el desarrollo del bot.

Añadir:

El comando añadir permite crear un nuevo evento, con los datos pertinentes a la base de datos. Cuando llegue la hora designada en el evento, se enviará un mensaje en el mismo lugar dónde fue creado, recordando dicho evento.

Eventos:

El comando eventos obtiene todos los eventos del servidor, y los muestra en formato de lista, indicando la fecha, el título del evento y la descripción de este.

CREANDO EVENTOS

A la hora de crear un nuevo evento, el bot nos irá guiando con mensajes. Aquí los pasos a seguir:

1. ~cal añadir Nombre del evento
2. Descripción del evento
3. Fecha del evento en formato MM/DD/YYYY. Nota: El bot guarda la hora en la zona horaria GMT +00:00, por lo que habrá que variar la hora con tal de que se ajuste a la hora deseada. Ejemplo: Si queremos un evento a las 21:30 en la zona horaria de Europa Central, habrá que poner que el evento es a las 23:30 (GMT +02:00).
4. Evento guardado!

VISUALIZANDO EVENTOS DESDE EL SITIO WEB

Podemos ver todos los eventos desde nuestro bot con el comando ~cal eventos, pero también podemos ver estos eventos en formato calendario desde el sitio web. Para ello, hay que entrar en https://telek.tk, hacer click en login, iniciar sesión con Discord, y seleccionar el servidor deseado para ver los eventos. Con esto, ya puedes ver todos tus eventos de una forma cómoda y simple.

Capitulo 9. Wireframes

Haciendo click en este enlace se pueden visualizar los wireframes en la plataforma Figma.

Capitulo 10. Problemas

TIEMPO

El principal problema ha sido los recortes de tiempo respecto a las inicialmente propuestas 100 horas de proyecto, pues ha habido un festivo en la primera semana, y las fechas de entrega también se han adelantado.
Todo esto ha contribuido a que se tengan que hacer algunos recortes en el proyecto respecto a lo inicialmente propuesto.

DOCUMENTACIÓN

Otro de los problemas encontrados es que se han utilizado nuevas tecnologías, por lo que se ha tenido que invertir mucho tiempo en documentación. Algunas de estas tecnologías han sido la API de Discord, algunas funciones específicas de Node, y Cron para las tareas del bot.

ZONAS HORARIAS

A la hora de insertar los eventos, MySQL convierte la hora a GTM +00:00. No se ha encontrado con el tiempo disponible una solución viable y dinámica.

Capitulo 11. Lineas futuras

CRUD

Una de las principales características que se podrían implementar a futuro, es la edición y el borrado de eventos desde el bot y el sitio web, a demás de la creación de nuevos eventos desde esta.

PAGO

Un posible método para poder financiar el proyecto, sería crear comandos “premium”, por los que haya que hacer una aportación económica con tal de poder utilizarlos. Otra opción, sería implementar la opción de pagar para poder crear más de una cierta cantidad de eventos.

INFORMACIÓN DE USUARIOS

Por último, una característica que se podría implementar en el futuro, consistiría en obtener la información de los usuarios que han creado los eventos y mostrarla en el sitio web, junto con la descripción del evento.