**Documentación Cambios Sportta**

**Ambientes de Prueba**

Sportta utiliza la metodología gitflow y maneja actualmente ambientes de prueba que comparten una misma base de datos de testing para cada uno de los proyectos, esto con el fin de evitar errores a la hora de realizar los despliegues, para la api realizada en laravel se usa el ambiente de pruebas de sportta-stage el cual se encuentra en Heroku, para el ambiente móvil se tiene el ambiente de testing también alojado en el servicio de Heroku llamado testing en el Frontend, para el móvil se tiene un ambiente de pruebas alojado en la plataforma de expo con cuenta de testing disponible en trello, todas las pruebas se deben realizar bajo estos ambientes, cada ambiente se encuentra relacionado a una rama de git en específico, se manejan actualmente 3 ramas las cuales son main, dev y testing para los proyectos móviles y web, para la api se cuenta con las ramas master,dev y testing.

Las ramas de main y master poseen relacionados los ambientes de producción, **Importante no trabajar sobre estas ramas, solo se deben hacer despliegues desde estas de actualizaciones probadas debido a que los cambios sobre estas se verán reflejados en la aplicación en producción**, las ramas de dev y testing están relacionadas a los ambientes de prueba, se recomienda no trabajar sobre la rama dev y hacer pruebas sobre la rama de testing.

**Despliegues**

Para realizar despliegues en la api solo basta con subir cambios en la rama master y los cambio se reflejaran en producción, esto debido a que git se encuentra conectado con Heroku, para subir cambios a la versión móvil es necesario crear una build tanto para ios como para Android la cual debe ser desplegada en la cuenta de producción de Expo, luego se debe descargar esa build y cargarla a las tienda de play store mediante la consola de play store y en iOS mediante la plataforma app store connect, para realizar cambios en versión web se deben enviar a Heroku ya que no se encuentra automatizado el despliegue con git, esto se hace mediante la instrucción git push Heroku <Nombre de la rama>, para realizar esta instrucción se ha debido registrar con antelación el registro del repositorio esto mediante la instrucción Heroku git: remote <Nombre del repositorios>.

**Cambios en formularios de Registro**

En la aplicación móvil se realizaron validaciones de los inputs para asegurarnos de que se reciba de manera correcta los datos y no haya ningún tipo de error al realizar búsquedas, además se agregaron los inputs de cedula de ciudadanía y fecha de nacimiento los cuales nos permitirán tener la información del usuario actualizada, en base a la fecha de nacimiento se realizó el cálculo de la edad para que se mantenga actualizada.

Por otra parte, se modifico la solicitud para enviar notificaciones en la app móvil para que lo solicite una vez el usuario concluya el registro, cabe destacar que es importante que el usuario nos otorgue estos permisos para que la aplicación funcione de una manera correcta.

**Base de datos de Prueba.**

Debido a la eliminación de los planes gratuitos de Heroku se procedió a cambiar el plan de la base de datos de prueba a un plan Basic el cual tiene un costo de 9USD, con esto Sportta asegura el ambiente de pruebas para los 3 proyectos trabajados actualmente.

**Permisos para realizar despliegues**

Se procedió a actualizar las credenciales que se encontraban vencidas para poder realizar despliegues en la iOS, estos certificados solo se pueden firmar desde una maquina MAC con la aplicación del llavero, para esto son necesarios certificados y credenciales que se encontrarán en el tablero de trello o serán proporcionadas por el personal a cargo de ellas

**Cambios en los permisos para las notificaciones push**

Sportta requiere para su optimo funcionamiento tener permiso para mandar notificaciones push a los usuarios, por esto al realizar el registro se requiere este permiso, si el usuario no les brinda los permisos a los usuarios la aplicación no podrá enviar dichas notificaciones, en la última actualización realizada a la app se configuro para solicitar este permiso una vez se complete el registro.

**Permitir actualizar edad**

Sportta inicialmente capturaba la edad de los usuarios lo cual era un error debido a que con esto no se permitía tener una edad que se pudiese actualizar de forma dinámica, por esto se realizó el cambio de almacenar la fecha de nacimiento con la cual se calculaba la edad de forma dinámica, en base a esto se configuro la aplicación para poder dar la opción de actualizar la edad del usuario esto se logra actualizando la fecha de nacimiento.

**Permitir a los usuarios administrador descargar informes en Excel**

Como actualización se trabajo en permitir a los administradores poder descargar informes de cada uno de los movimiento realizados en su establecimiento, estos informes se podrán descargar por categorías dependiendo el pago(Cancelados, Digitales, Pago en efectivo, pendientes), además de poder realizar filtros sobre los mismos por fechas y se organizaron los registros del mas reciente al mas antiguo, por otra parte se agregó la opción de cambiar el estado de un movimiento de pendiente a pago en efectivo, esto debido a que si es un movimiento de pago en efectivo se debe recibir el dinero antes de que el movimiento sea aprobado al estado de pago en efectivo, este dinero pendiente por aprobar se visualiza en su propia categoría.

Por otra parte, se pueden ver los acumulados por categorías y el total recaudado, cabe decir que los pagos es estado pendiente no se agregar a este total, también se muestra el total de dinero que el administrador ha dejado de recibir por reservas canceladas.

**Permitir a los usuarios maestro descargar información de reservas canceladas en Excel**

Para el usuario maestro se agregó la opción de poder descargar un informe de todas las transacciones canceladas, además de poder filtrar sobre las mismas y descargar esos resultados filtrados.

Por otra parte, se agregó la visualización de los montos recibos por motivo de movimientos en cada uno de sus categorías y el total de cuanto ha recibido el establecimiento.

**Revisar esquema de base de datos**

Se procedió a generar el esquema de la base de datos esto con el fin de poder visualizar que tablas existen, como se relacionan y que atributos tienen, este esquema nos permite tener una visión general de cómo se están comunicando y transfiriendo los datos facilitándonos el trabajo.

**Habilitar y deshabilitar cobro de planes**

Como nueva funcionalidad se habilito el poder cambiar el precio de los planes y tener planes gratuitos, un plan marcado como gratuito tendrá un costo de 0 COP y cuando se realice el pago no requiere que se redirija a la pasarela de pagos, un plan premium o pago requiere que se especifique un valor costo el cual debe ser mayor a 1500 COP que es el monto mínimo que se puede manejar por la pasarela de pagos Wompi.

**Accesos a gitLab y gitLab**

Para evita la dependencia de desarrolladores que ya no trabajan en la empresa se procedió a dar los accesos a el desarrollador a cargo, estos accesos corresponden a los repositorios del proyecto alojados en gitLab y GitHub, con esto se pueden realizar actualizaciones, agregar o eliminar miembros de la organización de una forma rápida.

**Generación de certificados**

Para realizar generar el build de la aplicación para la app store es necesario firmar el build con los certificados de iOS, esta firma se realiza desde el usuario de expo dedicado a producción, cabe destacar que estos certificados tienen fecha de vencimiento y se deben generar cada cierto tiempo, los certificados que se usan para iOS son:

* Certificado de distribución
* Perfil de aprovisionamiento

Para generar estos certificados nos debemos dirigir a la consola de iOS, Apple Developers entrando en el perfil de desarrollador, cabe destacar que se debe estar logueado.

<https://developer.apple.com/account/resources/devices/list>

Una vez en la consola de app store se procede a dirigir a la sección de generar certificados y se elige crear un certificado dependiendo del que se necesite, cabe destacar que para generar un certificado de distribución es necesario que sea firmado desde un dispositivo MAC, con la aplicación del llavero y usando el certificado generado en la Apple Developers, para firmarlo necesitamos la clave del ambiente en expo y la clave de la mac, así como asignarle una clave al certificado, después de generar los certificados se debe dirigir a la cuenta de producción de expo y actualizar las credenciales.

**Configuracion de consola de play store**

Para poder realizar despliegues en la tienda play store se requirió especificar las políticas de recolección de datos de la aplicación móvil, esto debido a que si la configuracion es incorrecta tendremos el inconveniente de que google no nos permitirá realizar los despliegue

**Incluir filtros para visualizar transacciones**

**Paginación**

La paginación consiste en mostrar cierta cantidad de resultados por cada página, para esto desde backend esta configurado para devolver cierta cantidad de resultados por petición, para que esto funciones en la petición se debe establecer la cantidad de resultados a devolver, si no se hace de esta manera el método devolver a 10 resultados por defecto, cabe destacar el backend también devuelve la cantidad de paginas que se van a mostrar, esto se destermina diciendo la cantidad de resultados obtenidos entre la cantidad de resultados a mostrar por vista, cada vez que se pagine desde el Frontend se envía una petición al backend donde se especifica la pagina a mostrar, el backend devolverá los resultados correspondientes al la página, cabe destacar que si sobre los resultados se han aplicado filtros, esos filtros se deben especificar cada vez que se haga una petición de paginación.

**Pasarela de pagos Wompi**

<https://drive.google.com/file/d/1y_5cHSFvNj3l_1UcmprXF81NjgymVJHt/view?usp=sharing>

**Método de pagos Wompi y Tokenizacion de tarjeta**

Para registrar la tarjeta y realizar los pagos sin necesidad de registrar la tarjeta cada que se quiera realizar un pago, se procede a tokenizar la tarjeta con los links de pagos de los establecimientos registrados, de esta manera se crea un método de pago para cada establecimiento, este método de pago contiene la información necesaria de la tarjeta y el link privado de pago del establecimiento, con esto cuando se realiza un pago por motivo de reserva el pago se redirige al establecimiento que corresponde.

**Registro de una tarjeta**

Para registrar una tarjeta se procede a tokenizar y a asociársele un link de pago por establecimiento.

**Responsive Portal Web**

El portal web se encuentra con responsive design únicamente para dispositivos móviles, se llevo a cabo haciendo uso de una función que ayuda a condicionar la forma en que se muestran los componentes dependiendo del tamaño de la ventana. La función se encuentra en el estableshimentContext y se puede hacer uso de ella en cualquier componente solo haciendo uso del useContext. Te permite calcular el tamaño de la ventana ya sea en ancho o altura:

windowDimensions.width = ancho

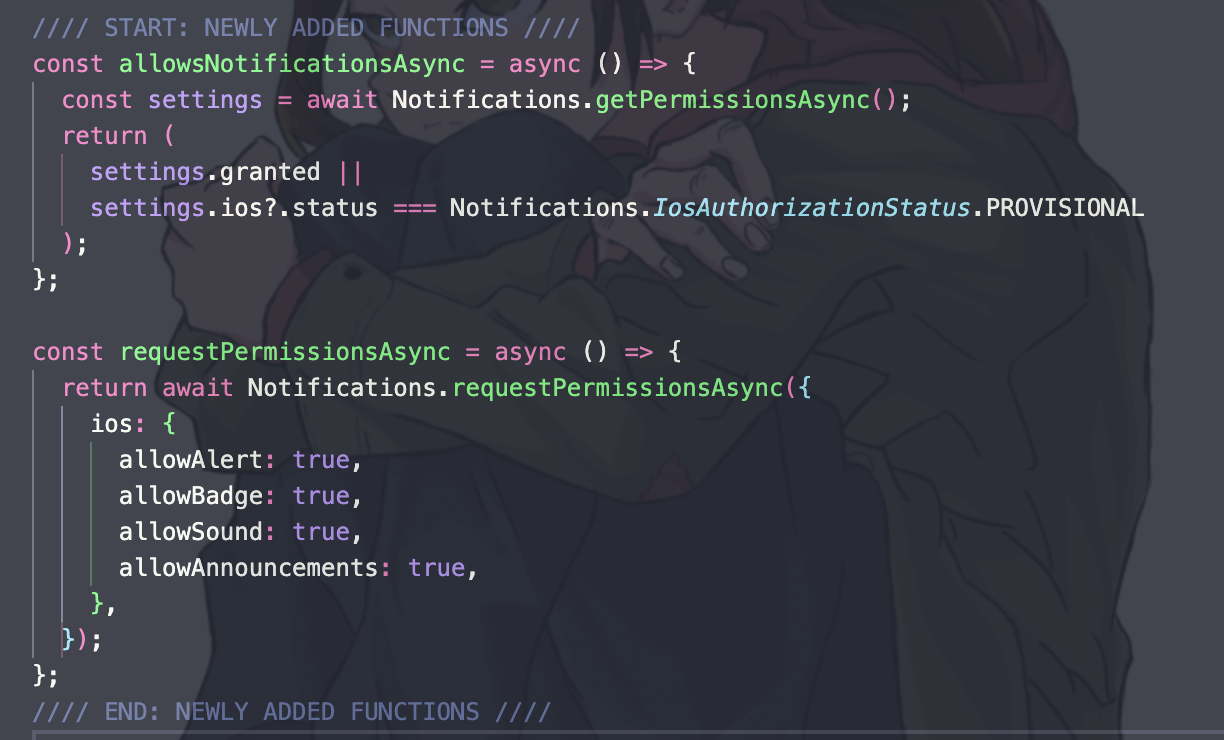
windowDimensions.height = altura

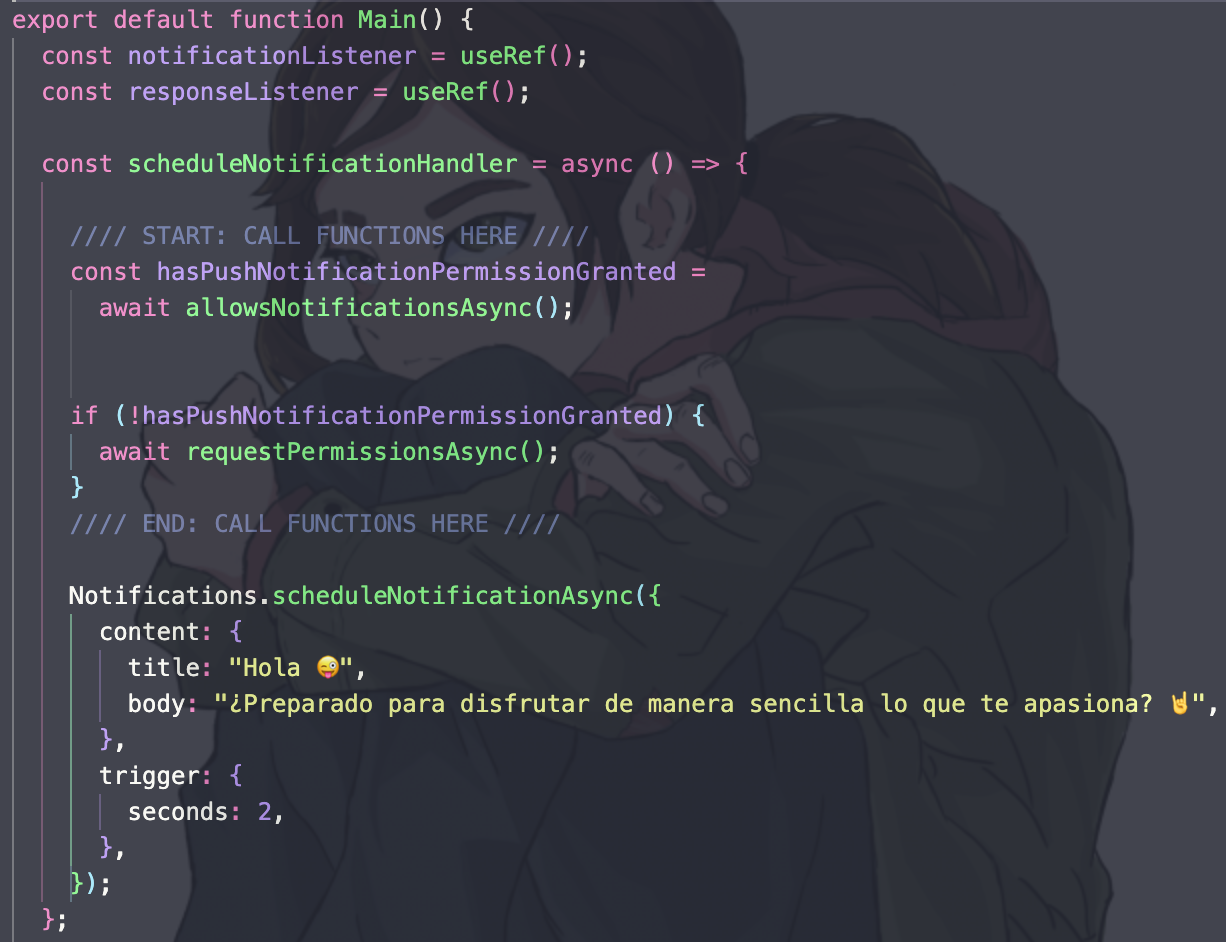
**Notificaciones Push**

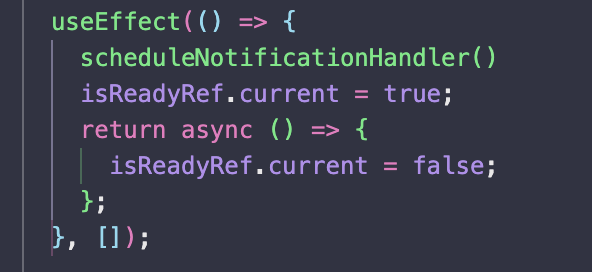
Las notificaciones push se están trabajando con Expo Notifications desde backend, el ExpoToken se genera desde frontend y en dos situaciones diferentes:  
  
1. Cuando se envía un sms al número de celular en la autenticación.

2. Cuando se inicia sesión(en caso de que el expoToken haya caducado este se generará nuevamente).

Para poder enviar notificaciones a iOS hay que agregar unos pasos extras:

1. Crear una función que verifique los permisos sobre el dispositivo para poder enviar notificaciones push. Ejemplo:





Como se puede ver en el ejemplo, primero se crean dos funciones. La primera es la que verifica los permisos del dispositivo, si fue concebido o no. La segunda solicita los diferentes permisos de una notificación push en iOS.

Después en el componente principal main.js creamos una función que básicamente lo que hará es enviar la notificación, entonces usa la primer función creada para verificar si fueron dados los permisos y sino es así entonces los solicita, finalmente envía una notificación donde:

Title: Título de la notificación push.

Body: Mensaje de la notificación push.

Trigger: Duración de la notificación push.

Haciendo uso de un useEffect() enviamos la notificación cada que se cargue el componente main.js

**Módulo de Canchas**

Este módulo cuenta con un crud básico en el cual se pueden listar las canchas con filtro por ciudad y por puntos cardinales, todos los campos son requeridos al momento de crear y actualizar canchas, estas canchas se ven reflejadas en la app móvil divididas en directorios.

**Módulo de Noticias**

Este módulo cuenta con un crud básico en el cual se pueden listar las noticias con filtro por tipo de deporte, todos los campos son requeridos al momento de crear y actualizar noticias, estas noticias se ven reflejadas en la app móvil divididas en directorios. Este módulo cuenta con un Job en el backend que verifica cada minuto si se ha cumplido con el día de expiración y si este se cumple, se elimina la noticia; de la misma forma con la publicación de noticias.

Actualmente se cuenta con los siguientes tipos de noticias:

Noticias sportta: son noticias que son creadas y actualizadas semanalmente desde el maestro.

Noticias liga: son noticias que tienen como proveedor una api pública la cual nos permite mostrar información de ligas como su tabla de posiciones y entre otras.

**Troubleshooting y errores cometidos**

Durante el desarrollo se han cometidos errores significativos ya que de alguna manera han retrasado la entrega de software funcional, a continuación algunos de ellos:  
  
- Despliegue en Android: se retrasó el despliegue en Android buen tiempo porque Google play console rechazaba la versión debido a que se solicitaba un permiso (ACCESS\_BACKGROUND\_LOCATION) que incumplía con políticas de información sensible del usuario. Google recomendó eliminar este permiso ya que no se estaba haciendo uso de el y no era vital para el core de la app.

**Actualización de SDK**

Cada 6 meses sale una nueva versión de SDK para Expo projects, asimismo se mantienen vigentes las 3 últimas versiones. Actualmente 6 de marzo de 2023, se tiene la versión 48 y están vigentes 46,47 y 48.  
Al momento de actualizar el SDK hay que tener en cuenta los siguientes puntos:

1. Tener cuidado con las dependencias usadas ya que muchas de estas se vuelven obsoletas y por lo tanto, es necesario actualizarlas o incluso buscar otras alternativas.
2. Arreglar warnings. Puede pasar que cuando compilemos el proyecto localmente este se ejecute sin ningún problema pero si ves errores en consola es importante arreglarlos ya que al momento de pasar a producción el Build puede contener errores y fallar.

**Consultas con Eloquent y Query Builder**

Para nadie es un secreto que el ORM de Laravel nos facilita la vida a los programadores gracias a la facilidad con la que podemos hacer consultas, sin embargo, no se habla mucho del tiempo de respuesta usando Eloquent. Cuando hacemos consultas sencillas como buscar un dato en especifico o una cantidad de datos relativamente pequeña, realizarlas usando Eloquent es algo factible por la seguridad y facilidad que nos ofrece. Pero cuando hablamos de consultas más complejas que abarcan una cantidad de información bastante grande a nivel de bases de datos, lo mejor es usar Query Builder con joins, tal vez es más difícil construir el query pero te evitará muchos problemas en cuanto a velocidad.

**Jobs**

Al momento de trabajar con jobs se debe realizar como buena practica la ejecución en el handle con un try catch para que de esta manera se puedan detectar errores fácilmente. Ejemplo:

Otra cosa que hay que tener en cuenta es el tipo de funciones que se ejecutarán, hay que analizar si esta es una función estática y sí realmente se está instanciando de manera correcta. En este caso particularmente, el Job no se estaba ejecutando porque la función se estaba instanciando como si fuera una función estática. Por ello en el handle para acceder a ella se usan propiedades app() y make(). Para verificar que nuestro Job está ejecutando correctamente una función podemos hacer uso del dump() para ver en consola los procedimientos con php artisan schedule:work

**Patterns**

Se encuentra patterns tanto en backend como en frontend y hace referencia a las pautas de Sportta. Cada pattern debe tener definido su ubicación dentro de las pantallas de app móvil. Se hace uso de banner y Cover. Banner es la imagen(horizontal) que aparece en diferentes secciones de la aplicación y Cover hace referencia a un PopUp que inicialmente solo se renderiza durante una reserva e ingresando al detalle de los torneos, el Cover debe ser una imagen Vertical y se debe evitar para las dos usar fondos blancos o transparentes. Al momento de hacer onPress sobre las diferentes imágenes sea popup o banner se redirige al link guardado para cada pauta y se suma 1 a la columna clics.