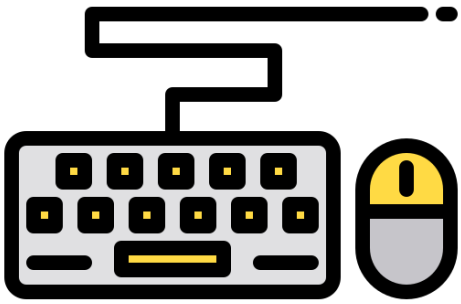
Institut Puig Castellar

Santa Coloma de Gramenet



DisControl

CFGS Desarrollo de Aplicaciones Multiplataforma

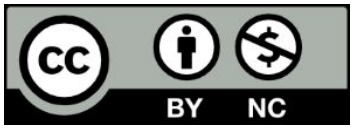
Daniel Coto Vázquez

Jonatan Sánchez Sánchez

2º DAM

El Puig Castellar

29 Mayo 2020



Datos del proyecto y resumen

Resumen

Este proyecto consiste en crear dos aplicaciones, una para dispositivos móviles y otra para escritorio, en las que podamos usar nuestro dispositivo móvil como control remoto para nuestro PC, dándonos así la posibilidad de controlar el ratón y el teclado desde la pantalla del móvil.

Esta idea surge de la falta de una aplicación sencilla de utilizar y gratuita para poder controlar de forma remota el PC que no se limite a una solo plataforma en cuanto a dispositivo móvil, rompiendo así la barrera entre Android e IOS.

Con esto puedes facilitar el trabajo de las personas que necesitan controlar el PC a una determinada distancia o no pueden tener un contacto directo con el mismo y a personas que por discapacidad no puedan tener un uso normal del PC.

Para ello se utiliza Flutter para desarrollar la aplicación móvil puesto que nos permite el desarrollo en paralelo tanto para Android como para IOS, en cuanto a comunicación entre dispositivos se utiliza Firebase Realtime, una opción de Firebase que nos permite la comunicación en tiempo real entre el dispositivo móvil y el PC. En cuanto a la aplicación de escritorio se utiliza Java para su desarrollo y JavaFX para la interfaz de usuario.

El objetivo del proyecto es crear una aplicación que sea rápida, fácil de entender y usar y que funcione en cualquier dispositivo. De esta forma la aplicación no se cierra a ningún grupo de usuarios, cualquier persona podrá identificar qué hace cada componente en pantalla con un simple vistazo haciendo así que el periodo de aprendizaje sea casi inexistente.

Palabras Clave

Java, Flutter, Firebase, JavaFX, Android, IOS, PC

Key Words

Java, Flutter, Firebase, JavaFX, Android, IOS, PC

Índice

Introducción

Este proyecto consiste en comunicar dos aplicaciones, una que estará funcionando en un dispositivo móvil y otra en un ordenador. Para comunicarlas se utiliza Firebase, más concretamente su función de Realtime Database con la cual no hay retraso entre la acción que ocurre en el móvil y la respuesta que da el ordenador a esa acción. Gracias al uso de Firebase no es obligatorio que nuestro dispositivo móvil y el ordenador que queramos controlar estén en la misma red, lo cual evita todo tipo de problemas, desde necesitar estar en la misma red ambos dispositivos a tener que depender de una red Wi-Fi en el móvil en vez de usar datos móviles.

De esta forma se ofrece la posibilidad de usar el pc de forma remota, facilitando el uso del mismo a la gente que no puede estar físicamente delante manejando teclado y ratón. En este grupo de gente encontramos tanto gente con alguna discapacidad que le impida un uso corriente del PC como gente que por motivos de seguridad necesita controlar el pc desde una determinada distancia, ya sea en laboratorios o fábricas.

Sí, existen otras aplicaciones en el mercado que cumplen con el objetivo de nuestro proyecto, un ejemplo sería Monect. Esta aplicación permite entre muchas otras cosas controlar el PC desde un dispositivo móvil o desde otro PC. Además añade muchas otras funciones como utilizar el micrófono del dispositivo con el que controlamos el PC o dibujar en la pantalla del PC.

Monect es un ejemplo de una aplicación muy completa y que cubre muchas necesidades en una sola aplicación, pero esto también puede resultar estresante para el usuario ya que cuenta con demasiadas opciones, esto puede generar el abandono de la aplicación por parte de los

usuarios, sobre todo en usuarios de una edad más avanzada en la que se produce un choque tecnológico más fuerte.

Por lo tanto, usando Monect como inspiración en nuestro proyecto queremos ofrecer una aplicación potente pero con menos funcionalidades y más limpia, usando una interfaz simple y agradable para el usuario haciendo de esta forma que resulte fácil de aprender a usar, así el público que considera compleja una aplicación como Monect puede usar la nuestra .

La aplicación para móviles está programada en Flutter, ya que eso nos permite desarrollar nuestra aplicación tanto para Android como para IOS. Flutter también nos permite usar Firebase.

La aplicación para escritorio está programada en Java, ya que nos permite usar Firebase. También usamos JavaFX para crear la interfaz del login y facilitar la compresión del usuario.

Contexto y justificación del proyecto

La idea de hacer una aplicación dedicada a gente con movilidad reducida viene de la necesidad de que todo el mundo tenga la opción de utilizar el ordenador de la forma más cómoda posible. Las aplicaciones que son parecidas tienen un problema, para que se conecten con el ordenador necesitan estar en la misma red. La solución que nosotros vemos es usar Firebase Realtime Database, ya que no influye para nada que estén o no en la misma red.

Objetivos

Nuestro objetivo principal es poder controlar el ordenador con el móvil, con una interfaz sencilla y sin la obligación de tener el ordenador en la misma red que el móvil.

También nos permite introducirnos y aprender en el manejo de Flutter.

Objetivos Específicos

- Crear una aplicación funcional para dispositivos móviles usando Flutter.
- Crear una aplicación funcional para ordenador usando Java y JavaFX.
- Poder utilizar la aplicación en dispositivos Android y IOS.
- Usar Firebase para conectar las aplicaciones de PC y móvil.
- Hacer una interfaz cómoda y sencilla de usar para gente con movilidad reducida.

- Tener el mínimo delay (retraso) posible en la comunicación a través de Firebase Realtime Database

Si el rendimiento de Firebase como enlace entre las aplicaciones no es óptimo, optamos por usar Bluetooth como enlace.

Planificación del proyecto

Metodología de trabajo

Para gestionar las dimensiones del proyecto, hemos utilizado una serie de programas para facilitarnos el trabajo.

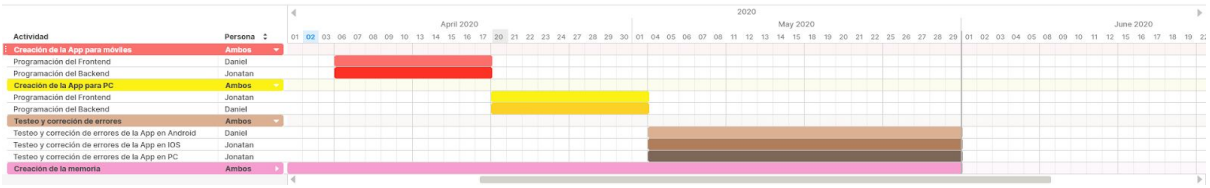
- Trello: Trello es un software de navegador que te permite organizar todas las diferentes tareas utilizando una pizarra. (Igual que en la metodología Scrum). Además de poder colocar las diferentes tareas en una pizarra, tienes unos “checkbox” en cada tarea para ver los avances que se van haciendo.

- Google Drive y GitHub: Las herramientas de google drive y GitHub se utiliza para poder tener diferentes archivos del proyecto en la nube, ya sean imágenes o el propio ejecutable del juego para poder descargar.

Planificación inicial

Dada la situación que estamos pasando y que las condiciones no son las más óptimas, la planificación inicial de nuestro proyecto es orientativa y está sujeta a cambios.

A continuación está nuestro diagrama de Gantt. Para verlo en mejor calidad está el enlace a la imagen en la bibliografía.



Recursos

Para nuestro proyecto necesitamos dos tipos de recurso, Software y Hardware.

Software

El software necesario para nuestro proyecto es:

- Android Studio - Programa para la creación de la aplicación móvil
- IntelliJ IDEA - Programa para la creación de la aplicación de escritorio
- Scene Builder - Programa para la creación de la interfaz gráfica de la aplicación de escritorio.

Todo el software es gratuito.

Hardware

El hardware necesario para nuestro proyecto está compuesto de un dispositivo móvil y un ordenador, ambos dispositivos son usados para hacer tests de las aplicaciones.

En nuestro caso usamos nuestros propios dispositivos, con lo que no tenemos la necesidad de comprar ningún dispositivo extra.

Estudio Económico y Presupuestario

Para realizar el presupuesto inicial, se decidió dividirlo en tres partes: Equipo, Instalaciones y Personal.

En la parte de equipo, hemos introducido todo el material de trabajo necesario para llevar a cabo el proyecto, tanto software como hardware

.

El apartado de instalaciones se componen los gastos generados por los locales y lugares de trabajo que se han utilizado durante la elaboración del proyecto.

Por último los gastos de personal donde se estima lo que cobrara los empleados por las horas de proyecto. Para hacer la estimación de lo que cobrará cada desarrollador hemos realizado una búsqueda por diferentes webs de empleo. Y hemos realizado una media de lo que deberá cobrar por todo el trabajo. En el anexo está el enlace a la web con los salarios.

Equipo	Precio
Ordenador	1657,17 €
Móvil	318,99€
Material de Ofimática	0€

Instalaciones	Precio
Local(Casa propia)	0€

Total Equipo:	0€

Total Instalaciones:	0€

Personal	Precio * 99 horas

Apartado	Precio
Equipo	0€

Programador Junior	1.299€
Programador Junior	1.299€

Instalaciones	0€
Personal	2.598€

Total Proyecto:	2.598€

Total Equipo:	2.598€

Descripción del proyecto

Tecnologías

A continuación se explican las tecnologías a usar en este proyecto, junto con las ventajas y desventajas que conllevan usarlas:

Flutter

Flutter es un framework de código abierto desarrollado por Google para crear aplicaciones nativas de forma fácil, rápida y sencilla. Su principal ventaja radica en que genera código 100% nativo para cada plataforma, con lo que el rendimiento y la UX es totalmente idéntico a las aplicaciones nativas tradicionales.

Utilizamos Flutter porque nos permite hacer que la aplicación se pueda usar tanto en Android como en IOS sin problemas, esto hace que no nos cerremos a un tipo de cliente en concreto y a

su vez hace que se pierda la “obligación” de usar una plataforma en concreta, ya que funciona en ambas plataformas.

Además Flutter se encuentra en alza pese a su corto tiempo de vida y apunta maneras para superar a Android (Java + XML) en el desarrollo de aplicaciones.

Sin embargo esto también genera un inconveniente ya que es más fácil encontrar errores de Flutter y a su vez poseemos una menor formación que nos limita en su uso.

Java

Java es un lenguaje de programación orientado a objetos.

Nos ofrece la posibilidad de establecer una conexión con Firebase en tiempo real que nos permite recibir la información que genera la aplicación móvil sin retraso haciendo que la experiencia de uso no se vea afectada.

También es el encargado de pasar las órdenes que damos desde el móvil a órdenes que puede interpretar el ordenador para mover el ratón o pulsar determinadas teclas del teclado.

Por lo tanto Java se utiliza exclusivamente en la aplicación de escritorio junto con JavaFX para la interfaz gráfica.

Firebase

Firebase es una plataforma ubicada en la nube, integrada con Google Cloud Platform, que usa un conjunto de herramientas para la creación y sincronización de proyectos que serán dotados de alta calidad, haciendo posible el crecimiento del número de usuarios y dando resultado también a la obtención de una mayor monetización.

Firebase Realtime Database

Firebase Realtime Database es una base de datos alojada en la nube. Los datos se almacenan en formato JSON y se sincronizan en tiempo real con cada cliente conectado. Cuando compilas apps multiplataforma con nuestros SDK de iOS, Android y JavaScript, todos tus clientes comparten una instancia de Realtime Database y reciben actualizaciones automáticamente con los datos más recientes.

La tecnología de Realtime nos permite hacer que la comunicación entre dispositivos sea instantánea y no haya retraso, pero en el caso de que esta comunicación se vea afectada o no se establezca/configure correctamente puede generar inconvenientes en el uso de la aplicación haciendo que su uso se vea reducido o sea inutilizable.

JavaFX

JavaFX es un conjunto de paquetes de gráficos y medios que permite a los desarrolladores diseñar, crear, probar, depurar e implementar aplicaciones de cliente enriquecido que operan de forma consciente en diversas plataformas.

Su uso se centra en la interfaz de usuario de la aplicación de escritorio ya que se utiliza junto a Java, así pues el objetivo de utilizarlo es facilitar el uso de la aplicación y mejorar la experiencia del usuario al no tener que verse obligado a usar un terminal o modificar el propio código dependiendo del equipo en el que se utilice.

Fases del Proyecto

Análisis y Diseño

En la Fase de análisis buscamos establecer las dimensiones del proyecto y definir las diferentes funcionalidades que tiene.

Gracias a esta fase llegamos a la conclusión de que la aplicación tenía que tener un diseño sencillo, que favorece su uso en personas menos acostumbradas a este tipo de aplicaciones, y, que quizá no son las más comunes en el dia a dia.

Por lo tanto, todo el análisis hecho previamente influye directamente en el diseño de la aplicación y en el desarrollo.

Funcionalidades

Las funcionalidades que se realizan en la aplicación son las siguientes:

- Mover el ratón:
    - Arrastrar: te permite mover el ratón en cualquier dirección.
- Escribir:
    - Escribir: te permite introducir texto.

El hacer click en los diferentes botones de la aplicación, te permite realizar las siguientes acciones:

- Borrar
- Hacer un salto de línea
- Usar las flechas de dirección del ratón

Diagramas

Los siguientes diagramas reflejan la estructura del código de la aplicación de escritorio y los estados por los que pasa la ejecución de la aplicación, tanto en el dispositivo móvil como en el ordenador.

Diagrama de clases de la aplicación de escritorio:

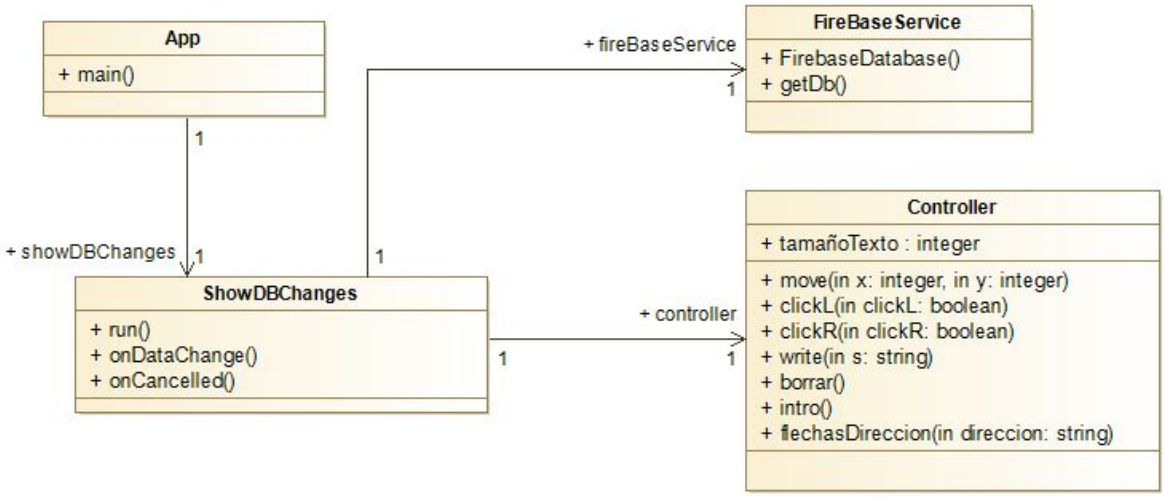


Diagrama de estados de la aplicación de escritorio:

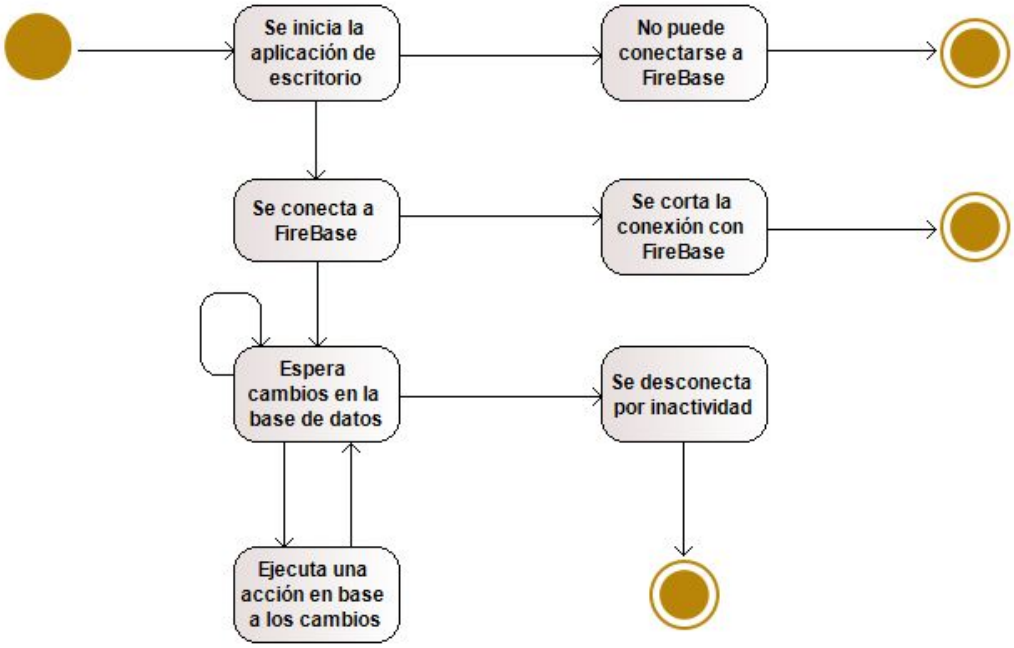
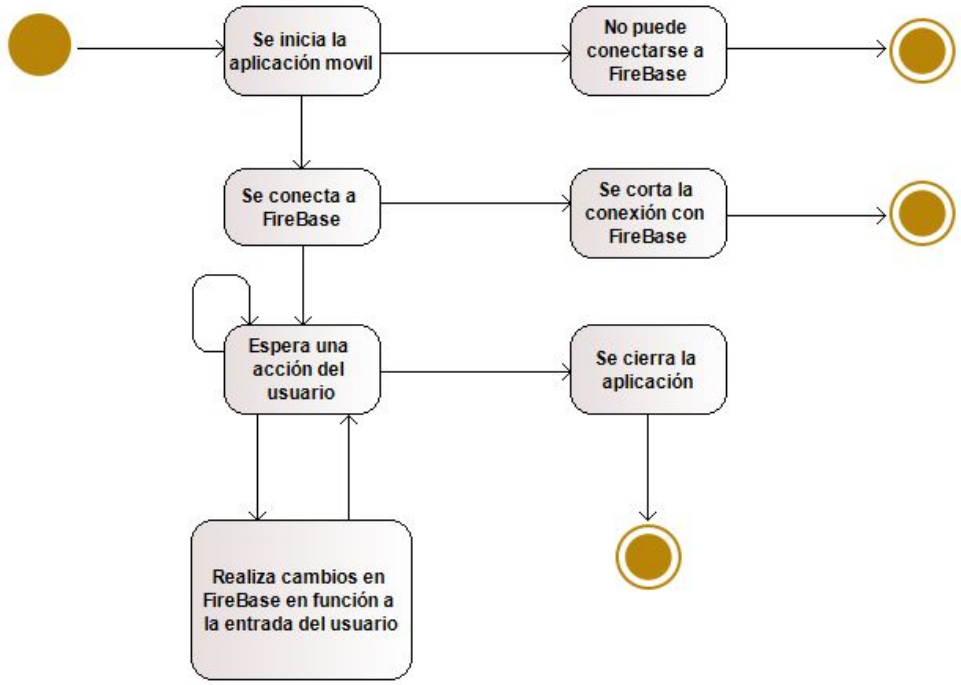
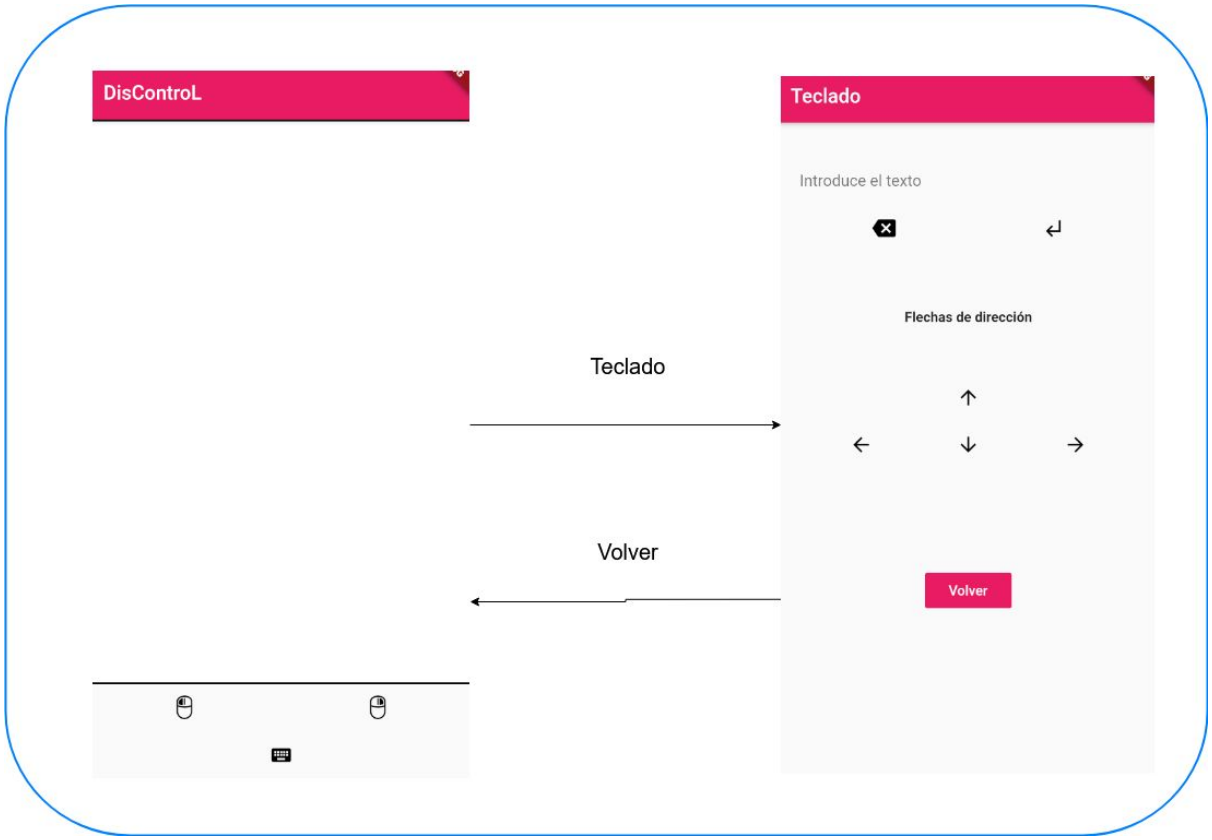


Diagrama de estados de la aplicación móvil:



Interfaz

La forma más sencilla de explicar el funcionamiento de la interfaz es utilizando un mock up de la aplicación. Los mockUps son fotomontajes que permiten a los diseñadores gráficos mostrar a cliente como quedará el diseño de la aplicación. En este caso, el mock up mostrará las diferentes pantallas de la aplicación y cómo se relacionan entre ellas.



Desarrollo

El código de las aplicaciones está subido a GitHub, el enlace está puesto en el apartado “Bibliografía” de la memoria.

Aplicación Escritorio

La aplicación se divide en varias clases. Una de ellas se encarga de todo lo relacionado con la base de datos, luego otra clase procesa los cambios de la base de datos y elige qué método ejecutar, por último tenemos una clase con los métodos que se ejecutarán en función a los cambios en la base de datos.

App

Es la clase principal de la aplicación, es lo que se ejecutará al iniciar la aplicación.

ShowDBChanges

Se encarga de crear un objeto de la clase Controller y pasarle los parámetros requeridos para cada método. Para ello comprueba constantemente el estado de Firebase Realtime y cuando detecta un cambio se ejecuta, lo que hace que envíe información a los métodos de la clase Controller correspondientes, por lo tanto si ha detectado un cambio en la posición del ratón en la base de datos solo ejecutará el método correspondiente al movimiento del ratón.

Para ello utiliza condicionales que comprueban que ha cambiado en la base de datos, de no comprobarlo se ejecutarán todos los métodos, por lo tanto ha de ser aquí donde se decida qué método ejecutar y no en la clase controller.

FirebaseService

Esta clase se encarga de la conexión con Firebase, su función es establecer una conexión con la base de datos, crear una instancia y devolverla. Esta instancia es la que usa la clase ShowDBChanges.

Controller

Aquí es donde están programadas las acciones que se harán según los cambios que se hayan realizado en la base de datos.

Para ello se utiliza la clase Robot, que es propia de Java, que se utiliza para generar las entradas del usuario. De esta forma podemos simular el movimiento del ratón del usuario y gracias a al método move mover el ratón en nuestro PC.

Esta clase consta de un método dedicado a cada acción que puede realiza el usuario con la aplicación móvil, para ello se le pasan los parámetros desde la clase ShowDBChanges dependiendo de los cambios que se producen en la base de datos.

Aplicación móvil

La app para móviles se divide en 2 apartados, la ventana principal, en la que tenemos el touchpad y los botones de los clicks del ratón, y la ventana donde tenemos las opciones del teclado.

Ventana Principal

La ventana principal contiene el touchpad, en el cual realizamos la acción de mover el ratón, y los botones que realizan el click del ratón.

El touchpad obtiene la distancia recorrida que realizamos y actualizamos la base de datos con esos datos. Para realizar esta acción se utiliza el widget GestureDetector. En el widget le definimos que cuando el touchpad se actualiza, envíe la distancia recorrida a la base de datos en FIrebase.

Los botones que realizan la acción de click actualizan los campos de la base de datos con un “true”, seguido de un “false” para resetear el valor de la base de datos y así sea más sencillo el saber cuando la aplicación debe realizar la acción.

También contiene un botón el cual te permite navegar a la ventana del teclado.

Teclado

En el apartado del teclado contiene lo siguientes campos:

- Campo de texto (TextField).
- Botones “Borrar” y “Salto de línea”.

- Botones de dirección.

El campo de texto actualiza la base de datos en tiempo real. Las ventajas y desventajas que esto conlleva son las siguientes.

Al escribir un carácter, el carácter escrito no reemplaza al carácter existente en la base de datos, si no que se concatenan. La ventaja de esto es que si borramos algún carácter también se borra en la base de datos. La desventaja es que en la aplicación de escritorio tenemos que filtrar la cadena de la base de datos y quedarnos solo con el último carácter.

Los botones de “Borrar”, “Salto de línea” y los botones de dirección actualizan el valor de sus propios campos de la base de datos en el momento que hacemos click en ellos. Actualizan la base de datos de la misma manera que los botones utilizados para hacer los click del ratón explicados en el apartado anterior.

También contiene un botón el cual te permite volver a la ventana de inicio.

Base de datos de Firebase:

En Firebase se guarda un objeto con una estructura de árbol, partiendo de la raíz hacia las ramas encontramos 3 caminos, uno en el que se registra el movimiento del ratón, otro para registrar los clicks y las teclas de dirección y un tercero en el que se registra toda la información relacionada con el teclado.

Cambios respecto a la planificación inicial y consecuencias de los cambios

El cambio más importante ha sido la no implementación de un inicio de sesión en la aplicación móvil, esto ha hecho que no sea necesaria la interfaz de JavaFX ya que no aporta alguna función más allá de una interfaz en la que rellenar campos.

Esto se debe a que la interfaz de JavaFX se utiliza para recoger la entrada del usuario y buscar en Firebase dicho usuario, al no tener un inicio de sesión en Firebase no es necesario buscar el usuario que se introduzca en la interfaz de JavaFX, haciendo por lo tanto innecesario el uso de JavaFX.

También repercute en la arquitectura de la aplicación, ya que al usar JavaFX contariamos con un Modelo Vista Controlador (MVC) para gestionar los archivos de la interfaz y el código de la aplicación.

Pruebas

Funcional

Antes de la versión final se han hecho diferentes pruebas para comprobar que todas las funcionalidades de las aplicaciones funcionan correctamente. A continuación dejo un listado de las pruebas que se han realizado.

- Test funcionamiento de Firebase.
- Test de funcionamiento del uso del ratón.
- Test de funcionamiento del uso del teclado.
- Test de funcionamiento del uso de los botones de borrar y se salto de línea.
- Test de funcionamiento del uso de los botones de las flechas de dirección.

Corrección de errores

Los principales errore que hemos tenido durante el desarrollo de la aplicación han sido el uso del ratón y el teclado.

El problema principal del ratón era en el desplazamiento, ya que cuando hacíamos el desplazamiento, no se continuaba desde la posición en la que estaba el ratón, si no que el ratón automáticamente se desplaza a una posición de la pantalla y luego hace el movimiento que nosotros le indicamos.

El problema residía en el método que usábamos para recibir los datos del movimiento, ya que al hacer el movimiento, recibiamos la posición de la pantalla cuando realmente los datos que se necesitan para ejecutar bien el movimiento es recibir la distancia que hemos recorrido.

En el teclado lo problemas son más complejos.

El principal problema del uso del teclado está en el momento de borrar el primer carácter que introducimos cuando escribimos, nos es imposible borrarlo. Eso lo hemos solucionado añadiendo un botón que permita borrar.

El segundo problema reside en los saltos de línea. En el teclado del móvil no aparece el botón de salto de línea, por lo tanto lo pusimos nosotros aparte.

El tercer error, consiste en los caracteres especiales. La aplicación de escritorio no escribe ciertos caracteres especiales, la razón reside en el método utilizado para escribir, ya que para escribir ciertos caracteres necesitas varias combinaciones de botones. En nuestro caso, por falta de tiempo, no hemos podido introducir todas las combinaciones posible, por lo que ciertos caracteres no se pueden escribir.

Conclusiones

El utilizar tecnologías como Flutter, una tecnología que apenas habíamos tocado, y Firebase, que ofrece un servicio como Firebase Realtime de forma gratuita, hace que el proyecto sea más accesible y no limita al uso de Android o IOS ya que nos ofrece la posibilidad de crear aplicaciones para ambas plataformas.

Por lo tanto podemos decir que estas tecnologías tienen un hueco asegurado entre las más usadas en los próximos años.

Post Proyecto

Debido a que el proyecto tiene unas horas limitadas, las funcionalidades planteadas anteriormente que no hemos podido implementar serán implementadas en un futuro. Las funcionalidades son las siguientes:

- Añadir una interfaz a la aplicación de escritorio.
- Añadir un inicio de sesión.
- Ampliación de las funciones del teclado.

Bibliografía

Github del proyecto:
https://github.com/DisControl-Project

Web con los salarios:
https://www.indeed.es/salaries/programador-junior-Salaries?period=monthly

Precio por componentes del PC indicado en el apartado de recursos:
https://www.pccomponentes.com/configurador/B3B459Bc1

Dispositivo móvil usado en el proyecto:
https://www.amazon.es/gp/product/B07T5CCP8T/ref=ppx_yo_dt_b_asin_title_o00_s00?ie=UTF8&psc=1

Diagrama de Gantt:
https://drive.google.com/open?id=1XyG6LV4E8juELJdINn0cIGfjIWv45zwW