Índice

1.Introducción

En este primer capítulo se muestra la motivación del proyecto realizado, tanto así como la descripción del sistema actual junto al alcance del proyecto. También se adjunta la organización de dicho proyecto así como el resto de sus características

1.1 Motivación

Ante la creciente visibilización y necesidad creada sobre la salud mental en nuestros días, reunimos inquietudes suficientes en el grupo de proyectos informáticos en tercero de ingeniería informática como para intentar ofrecer una herramienta de ayuda útil para los usuarios que decidan utilizarla

1.2 Descripción del sistema actual

En esta aplicación mostramos una forma de ayuda a las personas o profesionales que la necesiten. Consta de diferentes partes.

- La primera es un inicio de sesión donde podrás loguearte con tu usuario y contraseña.

una vez dentro en el menú se te mostrarán varias herramientas para tu disposicion: diario test resultados

- Diario, es un lugar donde podrás escribir que tal fue tu dia, como te sentiste que quieres mejorar etc
- Test, en este apartado podrás seleccionar entre un test general o varios específicos. Se realizarán preguntas las cuales al finalizar (y esperar una revisión) se te enviaran los resultados
- Resultados. A este apartado llegan los resultados de tus test, en los cuales aparecerá una calificación orientativa de tu grado de necesidad, junto algun consejo que quizás te sea útil

1.3 Alcance del proyecto

Nuestro alcance es algo claro, queremos que esta aplicación sea utilizada por usuarios con inquietudes sobre su salud mental que busquen un desarrollo

positivo, tanto así como para psicólogos los cuales deseen ofrecer esta herramienta a sus pacientes para realizar un seguimiento personalizado.

1.4 Organización del documento

La documentación se divide en 10 Apartados y cada uno con una serie de subapartados los cuales son:

- introducción: En este apartado se hablará principalmente sobre una pequeña introducción de nuestro proyecto, también hablaremos sobre lo que nos motivó a realizar este proyecto además de a dónde queremos llegar con el proyecto (alcance del mismo)

- Planificación: la que nos hemos organizado para realizar el proyecto, los costes necesarios para la realización del mismo y también un último subapartado sobre las dificultades que han podido surgir a lo largo de la realización del proyecto.

- Análisis de requisitos: Este apartado se centra en el análisis de los requisitos tanto funcionales como no funcionales y los requisitos de la información además también de la explicación de los diferentes casos de uso.

- Diseño del sistema: Se especifica el diseño del proyecto o aplicación así como también las bases de las arquitecturas tanto física lógica como la de diseños. En este apartado también se mostrará el cómo se ve la interfaz del usuario en nuestra aplicación y una breve explicación de cómo funciona la misma.

- Implementación del sistema: Dividido en 3 subapartados en los cuales se hablará principalmente sobre todo lo que engloba la parte de programación del proyecto, siendo esto el lenguaje utilizado, las bibliotecas, el framework de desarrollo, etc. Además también contendrá el código fuente utilizado y las diferentes comprobaciones llevadas a cabo del código.

- Pruebas del sistema: Las pruebas tienen como objetivo la comprobación de la integración del sistema de información

globalmente, para verificar el funcionamiento correcto de las interfaces entre los distintos subsistemas que lo componen.

- Trabajo de investigación: Se hablará sobre la búsqueda de la información llevada a cabo para la realización de las preguntas de los cuestionarios, los sistemas de puntuación y las ayudas/consejos hacia los problemas de los usuarios.

- Manual del usuario: Aquí se realizará la guía o manual del usuario, el cual se trata de un documento de comunicación de cómo utilizar la aplicación destinada al usuario.

- Manual de instalación y explotación: Se mostrará los pasos que habrá que llevar a cabo para la instalación de la aplicación.

- Conclusiones: conclusiones a las que hemos llegado con la realización de todo el proyecto y todo aquello que sacamos en claro tras acabarlo.

- Bibliografía: En este apartado se pondrán todos aquellos links de páginas web de las cuales nos hayamos ayudado para la realización de las distintas partes del proyecto.

2. Planificación

2.1 Metodología del desarrollo

El Departamento de programación, concretamente el orientado al desarrollo del back-end, está conformado por 4 miembros, uno de los cuales ocupa la posición de “jefe de Departamento” y se hace responsable de que el equipo trabaje adecuadamente, siguiendo las fechas de control establecidas por el jefe de proyecto.

El equipo realiza una división de trabajo entre los miembros, donde se reparten varias funciones por individuo para su desarrollo. La metodología de trabajo seguida durante el proyecto consistía en varios controles semanales por parte del jefe de departamento, donde se indican los avances logrados o

las dudas surgidas a lo largo de los días, manteniendo así una dinámica de trabajo colaborativa entre todos los integrantes.

A lo largo del tiempo de desarrollo, el equipo se ha enfrentado numerosos desafíos y problemas en la realización de las funciones debido a numerosos cambios en las tecnologías implementadas. Dichos desafíos/problemas consisten: en el aprendizaje de un lenguaje de programación desde 0 (Primero PYTHON y, más adelante, PHP), el cambio de lenguaje de programación pasadas varias fechas (de PYTHON a PHP), múltiples cambios de base de datos (Comenzando con una base de datos relacional, MySQL, cambiando posteriormente a una no relacional, Firestore Realtime Database, y por último a Firebase Cloud Store) y por ende la implementación del lenguaje para el acceso a estas.

El Departamento de Front-End , está formado por 3 personas, uno de los cuales es el Jefe del Departamento, el cual es el encargado de dirigir esta parte o sección del mismo proyecto.

El equipo encargado del Front-End comenzó su metodología de trabajo mediante el aprendizaje de la utilización del framework Ionic , para su posterior utilización e implantación del código del mismo para realizar toda la parte correspondiente a HTML y CSS, básicamente a nivel visual.

Este equipo desarrolló de manera satisfactoria todo el contenido necesario para que a nivel visual e interactivo funcionase el proyecto, no obstante durante el proceso de aprendizaje tuvimos varios retrasos con respecto al calendario esperado debido a confusiones dentro del mismo y errores de factor humano.

El Departamento de información está formado por 2 integrantes de los cuales, uno es el Jefe de departamento, el cual se encarga de comunicar lo que se iba realizando en el departamento de información a los demás departamentos.

Este departamento llevó a cabo principalmente la búsqueda de información en diferentes sitios web, libros relacionados con la psicología y además consultamos a psicólogos para que nos ofrecieran su ayuda para poder realizar los cuestionarios de la mejor manera posible. También realizamos la búsqueda de pequeñas ayudas hacia aquellas personas con problemas diagnosticados por los test.

2.2 Planificación del proyecto

Nuestro proyecto comienza con el planteamiento de ideas sobre qué tema escoger para la asignatura de Proyectos informáticos. Decidimos cambiar del bloque sobre las adicciones hacia el bloque de medidas tóxicas y enfermedades mentales.
Una vez fijado el tema solicitamos en el aula FS03 para reunirnos.
En nuestra primera reunión decidimos el nombre de la aplicación “SELF” , organización de los bloques de trabajo: Framework, Información, Base de datos, Diseño. Y subapartados como creación del grupo de Pruebas.

2.3 Organización y recursos humanos

Una vez desarrollada la idea del proyecto se eligió a un representante, el cual ocuparía el cargo de jefe de proyecto (Rafael Becerra Villalobos) entonces, el grupo (formado por 9 miembros) se dividió en 3 departamentos: Recolección y Documentación de información (Fernando Candón Berenguer y Charbel Boueri Martinez), Programación orientada al Back-End (Rafael Becerra Villalobos, Juan Manuel Díaz Díaz, Juan García Candón y Francisco Mercado Espinosa) y Programación orientada al Front-End (Juan Jesús Milán Real, Acaymo Sánchez Ramírez y Alberto Contreras Puerto), cada representado por un jefe de departamento (Subrayados).

En los últimos departamentos nombrados se establecerían roles entre los integrantes de los mismos, destacando una persona de cada departamento que se encargaría de la documentación general de su respectivo grupo y, dos personas que realizarían las pruebas de software de la parte del Back-End.

A continuación, se citan concretamente las partes realizadas por cada miembro en su respectivo departamento:

Recopilación y Documentación de información:

- Ambos integrantes realizarían de manera conjunta los siguientes apartados: las funciones del bloque de información y documentación, la planificación completa del proyecto, la búsqueda de información sobre asuntos legales y normativas de las aplicaciones (uso legal), la documentación psicológica sobre enfermedades mentales, la

documentación psicológica sobre el uso de test de evaluación en pacientes y la creación de test para evaluación de pacientes con enfermedades mentales. Añadiendo la redacción de las actas de reuniones por la parte de Fernando, además de cumplir con las funciones de jefe de departamento.

-

Programación orientada a Back-End:

- Juan Manuel, se encargaría de realizar: las funciones relacionadas con los cuestionarios de la aplicación, las pruebas de caja blanca de todas las funciones del Back-End y la documentación de las pruebas del software.

- Rafael, se encargó de realizar la función de eliminar usuario de la base de datos y autenticación de Firebase, además de cumplir con las responsabilidades de los roles que ejercía dentro del proyecto.

- Juan, desarrolló las funciones relacionadas con el diario y la obtención de resultados por parte del usuario; y la realización de las pruebas de caja negra de todas las funciones del Back-End.

- Francisco, desarrolló las funciones de Inicio de sesión y registro de usuario; y realizó la documentación del departamento de Back-End.

Programación orientada a Front-End:

- Acaymo, desarrolló para la parte de Fron-End: la preparacion del framework ionic y angular, realizo los arreglos de incompatibilidades con ionic, capacitor y cordova, desarrollo la parte Scss y Html de la aplicación, la adaptación a terminales iPhone y Android, el diseño visual de la aplicación (usando Undraw para la mayoria de ilustraciones y fontawesome para algunos iconos), la preparación del código para futuras versiones y exportaciones a otros dispositivos, la preparación del código para la exportación a Android studio para compilarlo, realizar una apk funcional para cada iteración (3 a 8 segundos en compilarse) y el enrutamiento de la aplicación usando lazyload y Typescript.

- Alberto, realizó la búsqueda de información acerca del framework, la documentación general del Front-End y ayudó en gran medida a la documentación general del proyecto.

- Juan Jesús, se ha encargado de la creación y gestión de la base de datos, de la unión del Front-End y el Back-End y de mitigar los errores derivados de ello. Además de cumplir con las funciones de jefe de departamento. Juan Jesús ha sido la cabeza del desarrollo de la aplicación.

2.4 Costes

El coste final que se estimó al principio de este proyecto, se calculó a la baja suponiendo salarios precarios de no más de 10€ la hora. Por tanto, el resultado que salió en ese momento fue de unos 2700€ más o menos. Además, la estimación media de horas de trabajo fue inferior de lo que originalmente se pensó, siendo estas de unas 35 horas de media.

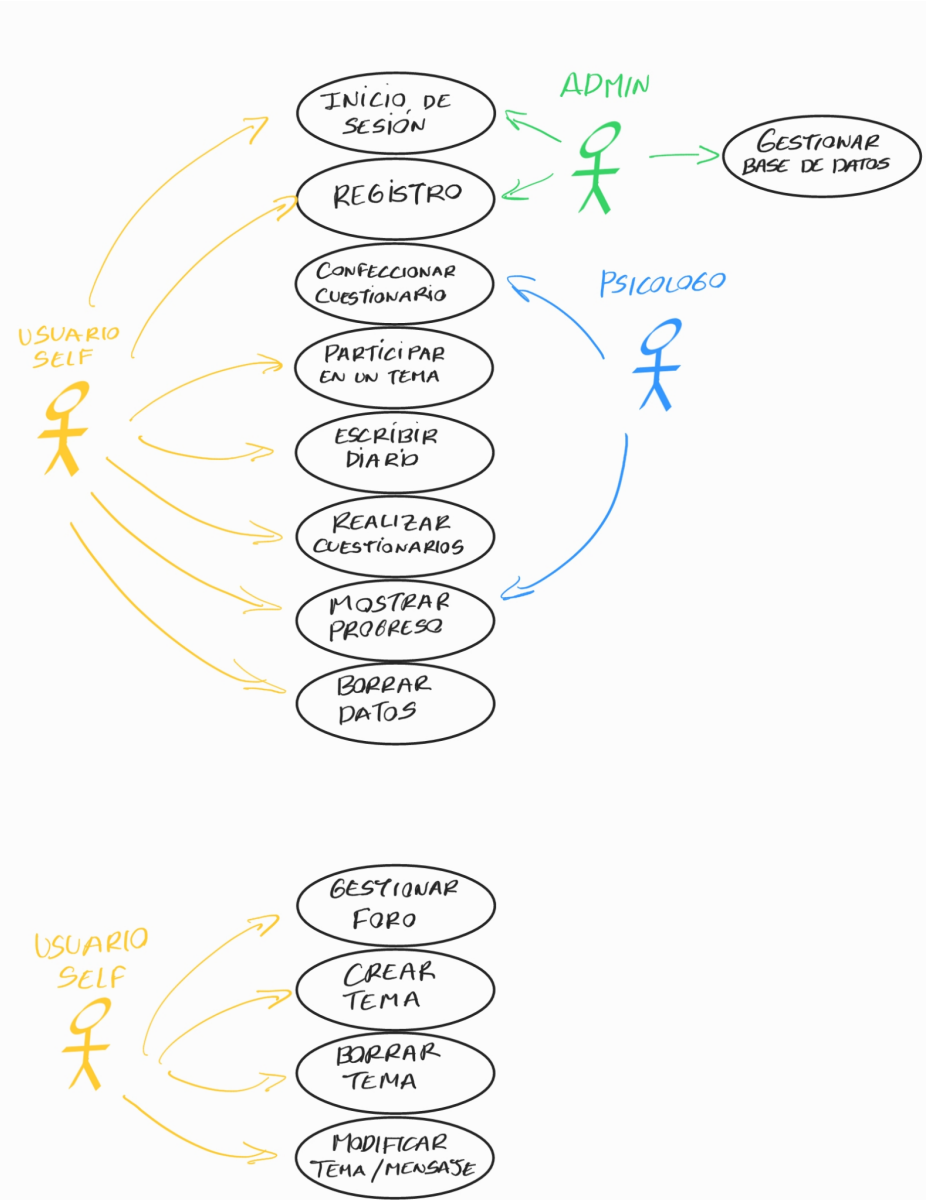
Tras la finalización de este proyecto, se ha decidido sobrevalorar, lo que hubiesen sido, los salarios de los trabajadores. Asimismo, el cálculo de la media de horas trabajadas ha ascendido exponencialmente a 60 por desarrollador provocando, por ello, una subida del coste a 12570€. Un 460% más de lo originalmente previsto.

2.5 Gestión de riesgos

A lo largo de la realización del proyecto hemos llegado a encontrar 4 inconvenientes que han podido afectar a la realización del mismo, uno de ellos, el cual es el de menos impacto, es la falta de asistencia por parte de algunos participantes a ciertas reuniones las cuales tenían una fecha concreta. Otro de los inconvenientes de impacto menor sería aquellos eventos ajenos al grupo de trabajo, es decir, no poder centrarse en su parte del trabajo que se le impuso debido a inconvenientes que no tenía nada que ver con el trabajo. Y hay dos inconvenientes que sí pudieron haber afectado mucho a la realización del proyecto, que son los posibles conflictos internos entre miembros del equipo de trabajo y por último la falta de recursos, lo cual podría hacer incapaz el trabajo para ciertos miembros del grupo a causa de la falta de materiales (software y hardware).

3. Análisis de requisitos

3.1 Catalogo de actores



3.2 Requisitos de información

Aquí se procederá a desplegar los casos de usos que hacen referencia al backend.

Casos de Gestión del Usuario

Caso de Uso: Registro Usuario.

Actor Principal: Usuario.

Escenarios:

- **Escenario principal(éxito)**: El usuario se registra con éxito.
- **Escenario alternativo (error 1):** Huecos vacíos en el registro.
- **Escenario alternativo (error 2):** Formato email incorrecto.
- **Escenario alternativo (error 3):** Apodo ya registrado.
- **Escenario alternativo (error 4):** Correo ya registrado.
- **Escenario alternativo (error 5):** Correo y confirmación de correo diferentes.
- **Escenario alternativo (error 6):** Contraseña y confirmación de contraseña diferentes.
- **Escenario alternativo (error 7):** Longitud contraseña menor que 6 caracteres.
- **Escenario alternativo (excepción):** El usuario cancela el registro.

Descripción de escenarios:

Escenario Principal:

1- El sistema muestra el menú de registro de usuarios.

2- El usuario rellena los datos que se solicitan para el registro y pulsa registro (Apodo, fecha de nacimiento, email, confirmación del email, Localización, Estudios, contraseña y confirmación de la contraseña).

3- El sistema comprueba que los datos introducidos son correctos.

4- El sistema registra al usuario en la base de datos.

5- El sistema redirecciona al usuario a la pestaña de Inicio de sesión.

Extensiones:

3a- El sistema comprueba que, al menos, un campo del menú de registro este vacío.

1- El sistema notifica de ello al usuario y lo reenvía al menú de registro.

2- Volvemos paso 1.

3b- El sistema comprueba que el formato del email es invalido.

1- El sistema notifica de ello al usuario y lo reenvía al menú de registro.

2- Volvemos al paso 1.

3c- El sistema comprueba que el apodo introducido ya está registrado en la base de datos.

1- El sistema notifica de ello al usuario y lo reenvía al menú de registro.

2- Volvemos al paso 1.

3d- El sistema comprueba que el correo introducido ya está registrado en la base de datos.

1- El sistema notifica de ello al usuario y lo reenvía al menú de registro.

2- Volvemos al paso 1.

3e- El sistema comprueba que el correo y la confirmación de correo introducidos no coinciden.

1- El sistema notifica de ello al usuario y lo reenvía al menú de registro.

2- Volvemos al paso 1.

3f- El sistema comprueba que la contraseña y la confirmación de contraseña introducidos no coinciden.

1- El sistema notifica de ello al usuario y lo reenvía al menú de registro.

2- Volvemos al paso 1.

3g- El sistema comprueba que la longitud de la contraseña es menor que 6 caracteres.

1- El sistema notifica de ello al usuario y lo reenvía al menú de registro.

2- Volvemos al paso 1.

1-2a El usuario cancela el registro.

Caso de Uso: Inicio de Sesión Usuario.

Actor Principal: Usuario.

Escenarios:

- **Escenario principal(éxito):** El usuario inicia sesión con éxito.

- **Escenario alternativo (error 1):** Huecos vacíos en el inicio de sesión.
- **Escenario alternativo (error 2):** Formato email incorrecto.
- **Escenario alternativo (error 3):** Contraseña o correo incorrecto.
- **Escenario alternativo (excepción)**: El usuario cancela el inicio de sesión.

Descripción de escenarios:

Escenario Principal:

1- El sistema muestra el menú de inicio de sesión.

2- El usuario rellena los datos que se solicitan para el inicio de sesión y pulsa "entrar" (Mail y contraseña).

3- El sistema comprueba que los datos introducidos son correctos.

4- El sistema inicia la sesión del usuario y lo redirecciona a la pestaña de Home del usuario.

Extensiones:

3a- El sistema comprueba que, al menos, un campo del menú del inicio de sesión está vacío.

1- El sistema notifica de ello al usuario y lo reenvía al menú de inicio de sesión.

2- Volvemos paso 1.

3b- El sistema comprueba que el formato del email es invalido.

1- El sistema notifica de ello al usuario y lo reenvía al menú de inicio de sesión.

2- Volvemos al paso 1.

3c- El sistema comprueba que el correo y/o la contraseña introducidos son incorrectos.

1- El sistema notifica de ello al usuario y lo reenvía al menú de inicio de sesión.

2- Volvemos al paso 1.

1-2a El usuario cancela el inicio de sesión.

Caso de Uso: Borrar

Actor Principal: Usuario.

Escenario principal (éxito): El usuario borra su cuenta.

Escenario de excepción: El usuario cancela la operación.

Escenario de Error: Se le cae la conexión al usuario durante la operación.

Descripción de escenarios:

-**Escenario principal:**

1-El usuario accede a la funcionalidad *borrar.*

2.-Pulsa el botón de Borrar.

3.-Se procede a borrar los datos de la base de datos.

Extensiones:

2ª.- El usuario desea cancelar la operación.

3ª.-Se pierde la conexión durante la operación:

1.- Se cancela la operación.

Casos de uso de Visualización de Resultados

Caso de Uso: Resultados

Actor Principal: Sistema.

Escenario principal (éxito): El usuario accede a la pestaña Resultados.

Escenario excepción: El usuario no tiene realizado ningún cuestionario.

Escenario de Excepción: El usuario no tiene conexión a internet.

Descripción de escenarios:

Escenario principal:

1- Se muestra la pestaña de Resultado.

2.-El usuario accede a dicha página.

3.- Si es la primera vez que accede, el sistema crea un fichero .json local

4.-El sistema accede a la base de datos.

5.- El sistema descarga la información sobre los resultados de los cuestionarios

6.- Se le muestra al usuario por pantalla dichos resultados.

7.- El sistema almacena una copia local de los resultados

Extensiones:

2ª.- El usuario no ha realizado ningún cuestionario.

En ese caso el sistema no mostrará nada.

4ª.- El usuario no tiene conexión a internet:

El sistema accede a la copia local almacenada, con el resultado de cuestionarios hechos anteriormente.

Casos de uso de Gestión de Diario:

Caso de Uso: Crear Archivo Diario

Actor Principal: Sistema.

Escenario principal (éxito): Acceder a la funcionalidad Diario por primera vez.

Descripción de escenarios:

Escenario principal:

1- Se muestra la pestaña de Diario.

2.-El usuario accede a esta pestaña por primera vez.

3.-El sistema crea un archivo .json vacío.

Caso de Uso: Actualizar Página.

Actor Principal: Usuario.

Escenario Principal (éxito): El usuario accede a Diario, actualiza la página (únicamente puede actualizar la página el mismo día en que la escribe).

Escenario de Error: No se almacena la actualización de la página.

Escenario de Excepción: El usuario intenta actualizar una página de un día anterior.

Descripción de Escenarios:

1. Se muestra la pestaña de Diario.
2. El usuario accede a Diario.
3. Accede a la página correspondiente con el día.
4. Actualiza la información de la página.
5. El sistema almacena los cambios en el fichero .json
6. El sistema actualiza la Base de Datos

Caso de Uso: Escribir Página

Actor Principal: Usuario

Escenario principal (éxito): El usuario accede a la pestaña Diario. Se le muestra una página para escribir en ella

Escenario Error: No se muestra página para escribir en ella.

Escenario Excepción: El usuario ha llegado al límite de páginas creadas.

Escenario Excepción: No hay conexión a internet, por lo que no se puede actualizar la Base de Datos.

Descripción de escenarios:

Escenario principal:

1-Se muestra la pestaña de Diario.

2.-El usuario accede a dicha ventana.

3.- El usuario escribe en la página del Diario.

4.-El sistema almacena en el archivo .json la información de dicha página junto con la fecha.

5.- El sistema sube los datos a la base de datos.

Extensiones:

4a.- El sistema no almacena la página porque ha llegado al límite de tamaño del archivo.

1.- El sistema comprueba que hay un número insuficiente de preguntas del tipo del cuestionario, muestra el error y se cancela la generación del cuestionario.

5ª.- No hay conexión a la Base de Datos.

1. El sistema comprueba que hay conexión a internet.

1ª En caso de que haya sube el archivo a la Base de Datos.

1bº En caso de que no haya almacena la copia local, y posteriormente sube toda la información del archivo entero, de esta manera se podrá usar el diario sin conexión durante los días que se quiera.

Casos de uso Rellenar Cuestionario:

Caso de Uso: Rellenar cuestionario:

Actor Principal: Usuario.

Escenario principal (éxito): Rellenar cuestionario Ok (el usuario contesta a todas las preguntas del cuestionario y los datos se envían correctamente).

Escenario alternativo (error 1): Usuario envía el cuestionario sin contestar alguna de las preguntas

Escenario alternativo (excepción): Cancelar cuestionario

Descripción de escenarios:

Escenario principal:

1-Se muestra el menú de cuestionarios.

2.-El usuario pulsa en rellenar cuestionario.

3.-El sistema muestra una serie de preguntas

4.-El usuario contesta a las preguntas del cuestionario

5.-El usuario pulsa en enviar cuestionario

6.-El sistema comprueba que todas las preguntas han sido respondidas

7.-El sistema muestra un mensaje informándole al cliente que se han guardado sus respuestas y que el resultado del cuestionario se mostrará cuando un especialista las haya evaluado.

8.-Se muestra la pantalla inicial del caso de uso.

Extensiones:

6a.-Falta por contestar una o más preguntas.

1.- El sistema comprueba que hay alguna pregunta que no se han contestado y solicita al usuario que la responda. (paso 4)

3-6a.-Cancelar cuestionario

1.-El usuario selecciona la opción Salir del cuestionario

2.-El sistema cancela el cuestionario.

Caso de Uso: Generar Cuestionarios:

Actor Principal: Administrador

Escenario principal (éxito): Cuestionario generado correctamente (El cuestionario se genera con preguntas del mismo tipo y se almacena correctamente en la Base de Datos).

Escenario alternativo (error 1): No existen preguntas de ese tipo

Escenario alternativo (excepción): Cancelar generación de cuestionario

Descripción de escenarios:

Escenario principal:

1-Se muestra el menú de administración.

2.-El usuario pulsa en generar cuestionario.

3.-El sistema genera un cuestionario con preguntas aleatorias cuyo tema sea el mismo que el del cuestionario

4.-El cuestionario se almacena correctamente

Extensiones:

3a.-Faltan preguntas del mismo tipo que el cuestionario

1.- El sistema comprueba que hay un número insuficiente de preguntas del tipo del cuestionario, muestra el error y se cancela la generación del cuestionario.

2-4a.-Cancelar cuestionario

1.-El administrador selecciona la opción Salir de generar cuestionario

2.-El sistema cancela la generación del cuestionario.

3.3 Reglas de negocio

Ya que no deseamos anuncios invasivos, nuestros target principal no son los usuarios libres que entren en la aplicación para consultarse a sí mismos. Son gabinetes de psicólogos, empresas externas o instituciones como Junta de Andalucía, que tienen un mayor alcance y pueden permitirse el desembolso de compra y mantenimiento de la aplicación.
Los beneficios de la aplicación saldría de la venta de licencias a gabinetes de psicólogos para que puedan usarlos de manera privada con sus pacientes y a través del mantenimiento de la aplicación de aquellos a quienes hayan adquirido una licencia profesional de SELF.

También se pueden canalizar beneficios con la venta de permisos para postear cuestionarios públicos y privados.
Se pueden llegar a pagar hasta 300€ por una plantilla de cuestionarios en el mundo de la psicología y, por tanto, una fuente de ingresos que puede ser explotada.
El precio en mercado por licencia privada de uso será de 199€ por institución privada pequeña. En caso de ser una organización como podría ser la junta de Andalucía o una universidad, entonces el precio ascendería en función de las mismas y del contrato prestado.

Por una licencia de cuestionarios se deberá abonar 39,95€ por persona física. Para publicitar a una mayor escala SELF, se darán licencias gratuitas a psicólogos de gran relevancia y/o importancia.

3.4 Estudio de alternativas

Existen otras alternativas tecnológicas en el mercado relacionadas también con aplicaciones de salud mental. Sin embargo, no existe ninguna aplicación que sea idéntica o, al menos, similar que pueda darnos una verdadera cara de como han realizado otros equipos independientes este trabajo.

Algunas apps y webs dedicadas a este trabajo son:

- PACIFICA: Dedicada específicamente a combatir la ansiedad a través de terapias de relajación.
- SPIRE: La cual es capaz de detectar el estado de ánimo de una persona y ayudarle a controlar su inteligencia emocional.
- Mood 24/7: Esta aplicación es la más parecida que hemos encontrado a la nuestra. Trata de intentar ayudarte en tu estado de ánimo diario a través de diferentes preguntas.
- MOODTUNE: Según dicen los propios desarroladores: “Si un juego es más tu estilo, trata con esta herramienta desarrollada por expertos en salud mental. MoodTune, usa tareas que te ayudan a controlar la depresión y la ansiedad, fue desarrollada después de casi 10 años de investigación.”

Hay muchas más aplicaciones en internet que hablan y se encargan de tratar este tema desde distintos puntos de vista; sin embargo, tal y como se ha podido apreciar con estos ejemplos, no se ha podido encontrar ninguna app idéntica a SELF. Lo que sí que se ha podido hallar en la red es algún estudio relacionado con nuestro tema (visto desde el punto de vista informático).

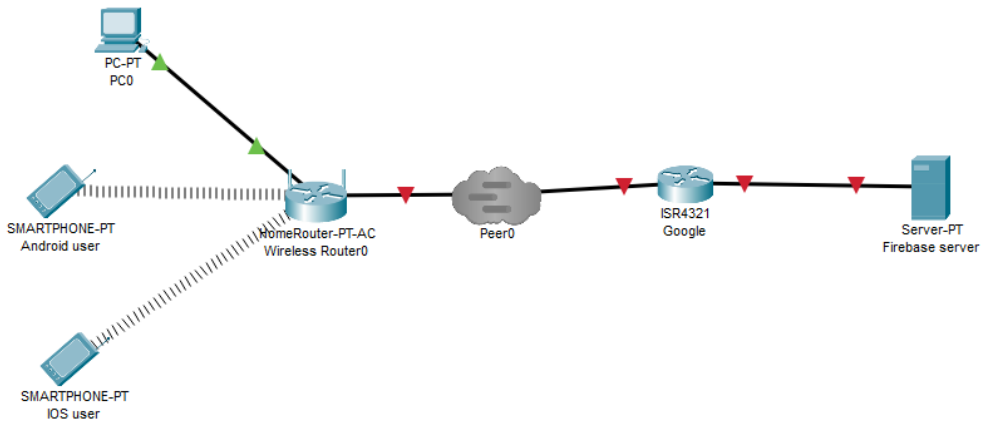
4. Diseño del sistema

Este capítulo describe el proceso de diseño de la aplicación. Concretamente centra su atención en la base de arquitectura realizada tanto física como lógica o de diseños

4.1 Diseño de la arquitectura

4.1.1 Arquitectura física

El programa necesita de un servidor firebase que esté online siempre para poder almacenar y tratar las peticiones que se realizan por parte de los usuarios y administradores que usan la aplicación web o APK. Como tal no se necesita ningún otro aspecto físico para que la aplicación pueda funcionar de manera online, sin embargo, es totalmente necesario que haya algún sistema que pida peticiones al servidor si lo que se pretende es que el servidor actúe ante estas peticiones. Dicho de otro modo, no hace falta ninguna otra estructura física complementaria para el funcionamiento normal de la aplicación si no se pretende realizar ninguna interacción con el servidor online.
Se adjunta el esquema del hardware.



Arquitectura física real

El hardware utilizado para crear esta aplicación ha sido de un total de 9 ordenadores con conexión a internet y con acceso al server de google.

4.1.2 Arquitectura lógica

Al inicio de este proyecto, se propusieron muchas ideas sobre los softwares que se iban a usar y los lenguajes de programación y entornos que podrían ser utilizados para la realización de la aplicación final. Incluso se comenzó a trabajar activamente en el desarrollo con estos candidatos, pero, tras informarnos, reunirnos y esclarecer activamente sobre cómo iba a funcionar nuestra base de datos,

decidimos hacer un cambio radical en cuanto a la arquitectura lógica se refiere utilizando, finalmente, todo lo que se aprecia a nivel software del proyecto.

Básicamente empezamos con python 8.6 como backend, html como frontend junto con IONIC como framework y una base de datos relacional (mysql). Tras investigar un poco más sobre si las herramientas que estábamos cursando eran las correctas o no, decidimos utilizar una base de datos no relacional (firebase) el cual se conectaría con el backend a través de las funciones que programaríamos con PHP.
Usamos Ionic debido a que la solución que planteamos a la necesidad solicitada por el cliente, pasa por el desarrollo de un aplicación híbrida , además de la propia versatilidad del framework Ionic con respecto a la competencia en su campo.
Usamos Firebase por la rapidez y sencillez del desarrollo de la base datos , además de la misma rapidez en la carga y subida de datos.
Por tanto, la arquitectura lógica final con la que se terminó de desarrollar SELF son:
A nivel de Backend:

- PHP.
- Javascript.

A nivel de servidor:

- Servidor de google con sistema Firebase database and authentication.

A nivel de frontend :

- Framework Ionic basado en Angular.
- HTML y CSS.

4.1.3 Arquitectura de diseños y diseño de componentes

Self está dividido en dos partes fundamentales (a nivel de desarrollo) llamadas Backend y Frontend.
Por resumir rápidamente, el backend es toda la parte que no ves en la página web o en la aplicación del móvil. Es todo el apartado que se encarga de realizar las funciones que hacen que el programa funcione internamente (todo lo referente al código fuente del mismo). Mientras que el frontend es todo lo contrario a lo anterior, este es todo aquello que ves e interactúas con ello. Puede tener algo de programación como HTML pero siempre orientada al entorno gráfico.

Tras tener claro estas definiciones, ahora procederemos a explicar como funciona y se conecta cada parte individualmente y en conjunto. Por lo anteriormente mencionado, se procede a separar este apartado en tres partes:

-¿Cómo funciona el backend?

El backend tiene varios módulos los cuales sirven para una tarea específica cada uno. Algunos contienen una serie de funciones extras de las que deberían (en teoría) pero; sin embargo, necesarias para que el programa esté completo y totalmente funcional.

Los módulos de los que estamos hablando son exactamente:

- conexionBDyAU: Este módulo se encarga de realizar la conexión con nuestro servidor específico de Firebase y de realizar ciertas comprobaciones de autenticación cuando un usuario se registra. Como por ejemplo que no haya dos usuarios iguales. Aparte, se encarga de que no haya cadenas vacías y que los correos y los usuarios estén bien escritos.
- loginBDyAU: Esta parte del sistema de logueo y registro es encargada únicamente de loguear a los distintos usuarios con su respectivo usuario o correo y contraseña. Como es de esperar, hace las comprobaciones necesarias con el servidor para que nadie acceda con un usuario que no existe o con una contraseña que no se corresponda con un usuario existente.
- registroBDyAU: Este módulo hace referencia a su propio nombre. Su tarea es la de subir el nuevo usuario registrado a una whitelist que el propio servidor trae denominada “Autentication” y a un apartado diferente de la BD donde se almacenará todo lo relacionado con el progreso del usuario con el programa, los cuestionarios y su diario. Tal y como debería de esperarse, este módulo revisa que el usuario que se registra no introduzca un nickname y un correo que ya estén subidos para que no haya datos duplicados.

Aclaración: Los usuarios que salen aquí no son usuarios reales.

- GestionDiarioLocal: Esta función se encarga únicamente de gestionar el diario de manera local (memoria del teléfono) en caso de que la conexión con el servidor se interrumpa. Para ello, cada vez que un usuario accede a su apartado del diario desde la aplicación móvil, se descarga una copia completa de la totalidad del diario pudiendo así gestionar y visualizar por completo esta colección sin la necesidad de una conexión a internet. Si el usuario vuelve a establecer una relación directa con el servidor, entonces se volverá a organizar toda la información en la nube a través de peticiones desde el cliente.

- GestionDiarioBD: Al igual que el anterior, sirve para la gestión y preparación del diario del usuario con la gran diferencia de que esta es la que controla cómo se “mueve” el diario y las funcionalidades básicas de este.
  El diario por si solo no podría funcionar con GestionDiarioLocal.php pero sí con GestionDiarioBD.php pues es el núcleo del funcionamiento.

- Borrar: El módulo hace exactamente lo que dice el título del mismo, borra los datos completos del usuario si así lo desea la persona que está logueada en ese momento.
  Lo borra tanto del authentification como de un apartado donde se almacena todo el progreso del mismo.

- Cuestionarios_preguntas: Esta es una parte vital del backend pues es la encargada de generar, de subir y de recibir todos los datos que tengan que ver con la creación de los cuestionarios, el progreso del usuario con respecto a estos o la ayuda recibida tras analizar automáticamente los datos.
  Cuando un usuario entre en su apartado de progreso, el módulo cuestionarios_preguntas consultará a la base de datos los datos que existen sobre él y se los devolverá por pantalla.

-¿Cómo funciona el frontend?

Podemos dividir la estructura del frontend en diferentes modulos que estan compuestos por un fichero .html el cual utiliza una serie de módulos como componentes del mismo , un ejemplo de estos seria el header (cabecera de la app) ya que sino sería redundante escribir este mismo header en los diferentes módulos, el fichero .scss que sirve para el formateo de la información y componentes que utilizará el fichero .html.

-¿Cómo se conectan estas dos partes?
Estas dos partes se conectan mediante un servidor web y la base de datos Firebase , realizando unas peticiones al mismo mediante unos formatos json. El “pegamento” usado para unir estas dos partes ha sido javascript y Java.

4.2 Diseño de la interfaz de usuarios

- Calendario: Es un módulo el cual muestra un calendario para que el usuario pueda controlar el número de días que lleva registrado en la app y a partir de ahí evaluar sus procesos y acciones dentro de la misma app.
- Chat : En este módulo el usuario podrá hablar con diferentes usuarios pero de manera individual.
- Cuestionario : En este módulo se realizarán los diferentes cuestionarios en relación a los posibles problemas detectados al usuario dentro de la aplicación , para que vaya tomando las diferentes orientaciones con respecto a su evolución.
- Diario : En este módulo el usuario podrá escribir lo que el usuario desee con respecto a cómo se siente , que realiza… pudiendo además adjuntar archivos multimedia a dicha publicación dentro de su perfil siendo siempre de manera privada.
- Forgotpassword : Este módulo sirve para poder enviar un mail a tu correo electrónico asociado a tu cuenta dentro de la app, en el que te ofrecen crear una nueva contraseña.

- Foro : En este módulo se trata, como su nombre indica de un chat dentro de la aplicación para que los usuarios que estén registrados en la aplicación puedan comunicarse, opinar e incluso debatir sobre ciertos temas en relación a sus problemas con la autoestima y etc…
- Help : En este módulo el usuario podrá consultar el servicio técnico de la aplicación para cualquier tipo de duda o mediante una batería de preguntas que resuelvan problemas en relación al uso de la misma.
- Home : En este módulo el usuario volverá a su pantalla de inicio , la cual es la que se muestra cuando se inicia sesión dentro de la aplicación.

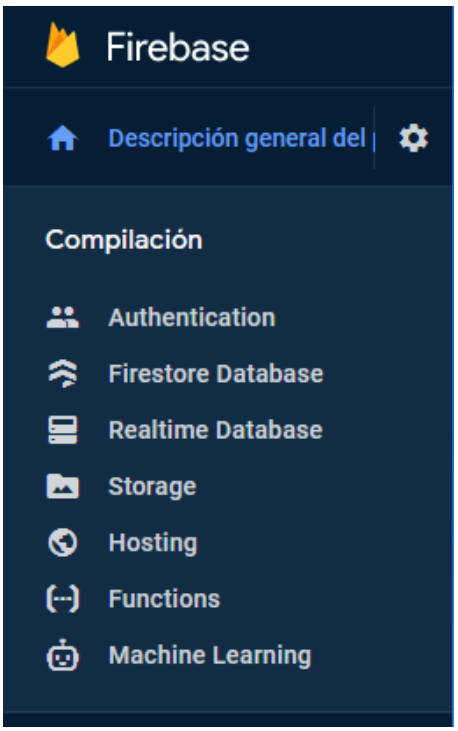
- Juego : En este módulo el usuario podrá utilizar un juego interactivo dentro de la aplicación, el cual está basado en el uso de un avatar personalizable.
- Login : Es el módulo encargado de la realización del inicio de sesión dentro la aplicación del usuario que lo solicite.
- Notifications: Es el módulo encargado de enviar las diferentes notificaciones y/o alertas dentro de la aplicación para el usuario.
- Options : Es el módulo encargado para ejecutar los diferentes cambios dentro de la aplicación de manera local para el usuario en cuanto a cómo quiere visualizar esta misma.
- Registro : Es aquel módulo encargado de la realización del registro del usuario dentro de la aplicación con los datos que este introduzca .
- Resultados : Es el módulo encargado de mostrar los progresos y acciones que el usuario va haciendo en el sistema en cuanto los cuestionarios realizados y las puntuaciones obtenidas.

4.3 Diseño de datos

En el Diseño de datos vamos a explicar cómo tenemos organizado la base de datos paso por paso, la tecnología que presenta, el sistema que corre por dentro y las reglas y acciones que realiza la base.
Lo primero a destacar es algo que ya hemos mencionado anteriormente en varias ocasiones, estamos hablando de la tecnología que mueve todo el sistema informático el cúal es Firebase.
Firebase de Google es una plataforma en la nube para el desarrollo de aplicaciones web y móvil. Está disponible para distintas plataformas (iOS, Android y web), con lo que es más rápido trabajar en el desarrollo.
Esta base de datos es no-relacional (también conocida como nosql) el cual quiere decir que plantea modelos de datos específicos de esquemas flexibles que se adaptan a los requisitos de las aplicaciones que requieran una mayor flexibilidad de manejo de datos.
El simple hecho de haber escogido una base de datos de estas características y no una relacional tradicional, como podría ser mysql, es por varios motivos:

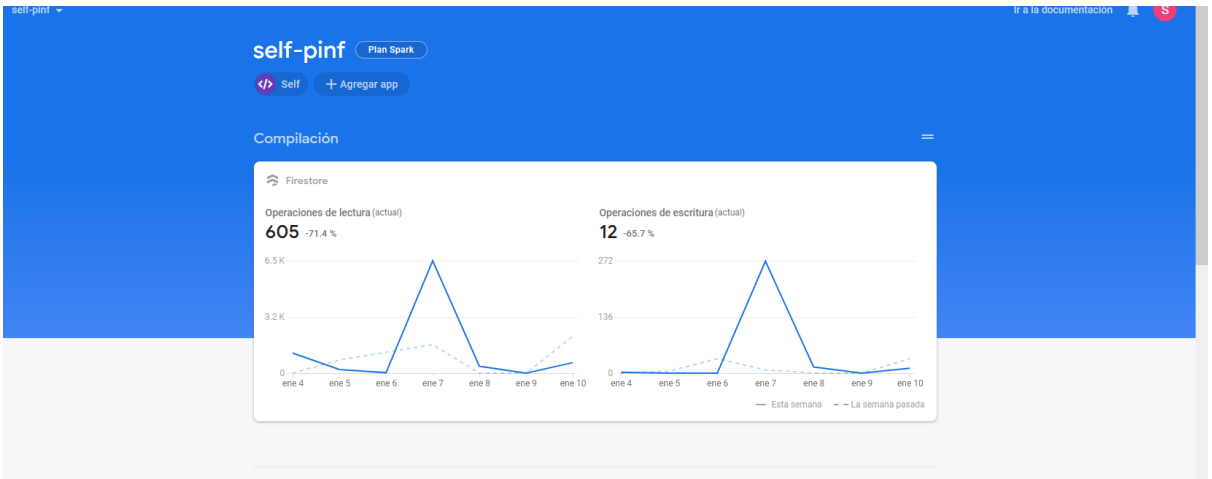
1. Las aplicaciones más modernas están comenzando a usar modelos no relacionales.
2. La idea de utilizar una base de datos no relacional implica también una mejora en el entorno de programación pues las peticiones de backend se pueden realizar de una manera más sencilla sin necesidad de tener que depender de ninguna tabla, esquema o claves primarias.
3. Al empezar a ser usadas más frecuentemente, se están creando sistemas y tecnologías punteras las cuales incluyen muchas funciones, plugins y facilidades para la conexión con la base y la aplicación desarrollada. Ejemplo: IONIC, Django, Larabel o Cordova.
4. Firebase trae funciones auxiliares y plugins que ayudan a la prevención de ataques de SQL, robo de datos y cifrado de los mismos incrementando exponencialmente la seguridad de la información sensible que se almacena.

Teniendo claro ya el por qué de Firebase, ahora vamos a ver como se ve Firebase por dentro y como funciona en sí.



Esta imagen nos enseña en que subapartados se subdivide todo el sistema. De todo lo que se aprecia, a nosotros solo nos interesa explicar los tres primeros apartados pues estos son los únicos que se han usado activamente.

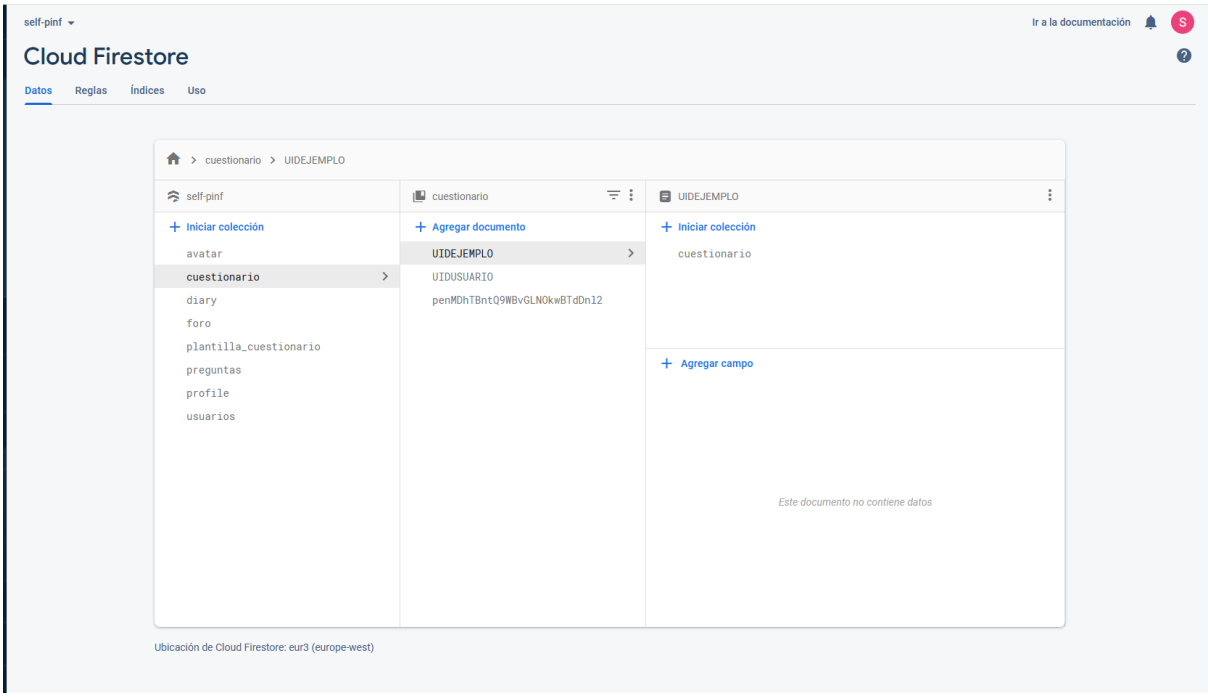
- Descripción general: Esta pestaña ofrece todo el resumen en cuanto a peticiones al servidor, operaciones de lectura y/o y compilación nos referimos. Si lo que queremos es consultar la actividad de nuestros usuarios con respecto al uso de la aplicación, este es el lugar para visualizarlo.



- Autentication: Esta sección es la encargada de dar de alta a los usuarios que se registren a través de la aplicación. Literalmente estamos hablando de que esta parte es una whitelist a nivel usuario. Nosotros como administradores podemos eliminar y editar toda la información que nos plazca, sin embargo, lo único que no podemos tocar y visualizar, son las claves de los usuarios pues estas se encuentran encriptadas en hashes de AES 256, cumpliendo así con lo que dictamina la UE con respecto al almacenamiento de este tipo de información sensible.

- Firestore database: Aquí están almacenadas todas las reglas, los parámetros y las variables que componen la base de datos.
  Este es el apartado más importante pues es aquí donde se suben, se consultan, se escriben y se modifican todos los datos de la aplicación. Estos datos van desde las preguntas de los cuestionarios, el progreso del usuario, la información adicional de este último hasta todo lo que el sujeto escribe en su diario. Incluso los temas, las respuestas y los propietarios de todos los mensajes del foro van almacenados aquí.

  Toda la información que se almacena aquí está catalogada en lo que llamamos UIDs. Las UIDs son cadenas que sirven para identificar textos, usuarios y otros tipos de datos con el fin de tener una manera de referenciarlos y conseguir un acceso específico. Un UID puede estar repetido en varias subsecciones del database si lo que pretendemos es que hagan referencia a un mismo usuario o actor del sistema.
  Ejm: Para sincronizar la información de un usuario con lo que escribe en el diario o el progreso de los cuestionarios que va realizando.



Cada colección de la BD tiene, respectivamente, los siguientes enlaces y documentos:

- Avatar->UID->apariencia
  ->money
  ->nick
  ->sala_actual
  ->sala_personal
  ->user

- Cuestionario->UID->Preguntas

- diary->UID->paginas

- foro->UID->owner
  ->content
  ->title

- Plantilla_cuestionario->UID->creado_por
  ->created_at
  ->disponible
  ->n_preguntas
  ->tipo
  ->updated_at

- Preguntas->UID->creado_por
  ->pregunta
  ->tipo
  ->valor

- profile->UID->email

- usuarios->UID->birth-date
  ->e-mail
  ->location
  ->nickname
  ->studies

5. Implementación del sistema

5.1 Entorno tecnológico

Ionic : Es un framework el cual utiliza tecnologías HTML, CSS JavaScript y Typescript, además es libre y de código abierto, este a su vez está basado en Angular, React y Vue, usamos este framework para la programación del front-end ya que con él podemos desarrollar aplicaciones híbridas y la propia facilidad.

Firebase de Google: Es una plataforma en la nube para el desarrollo de aplicaciones móvil y web, está además posee una base de datos proporcionada por Google en tiempo real, la cual sirve para la carga y descarga de datos.

Esta base de datos es NoSQL que se comunica o utiliza mediante JSON, por lo cual tenemos una rapidez en los datos muy elevada, la cual es posible mediante la técnica del “Lazy Load”.

Lazy Load: Es un patrón de diseño el cual está caracterizado por no cargar datos innecesarios cuando se adentra en una web y/o aplicación permitiendo una velocidad de carga de datos superior.

PHP: Es un lenguaje de programación orientado al desarrollo , debido a que la ejecución de este lenguaje se realiza mediante servidores, usamos este lenguaje debido a la facilidad de uso y la continua mejora incluyendo lo intuitivo que es por sí mismo.

5.2 Calidad de código

Para garantizar la calidad de nuestro código, hemos realizado exhaustivos análisis sobre este sometiendolo a diversas pruebas de sistema (tratadas en el punto 6) como:

- Pruebas Unitarias.
- Pruebas de Caja Negra.
- Pruebas de Caja Blanca.

A parte de estas pruebas, durante el desarrollo del software el equipo de programación ha estado cuidando el duplicado de código, añadiendo además un buen nivel de comentarios de manera que cualquier persona del equipo pueda entender el código de otro miembro del equipo. A través de esta metodología, disminuimos la complejidad del software desarrollado, que es otro punto importante a tratar para asegurar la calidad del software.

Por último, aparte de las pruebas del sistema, el código ha estado en constante prueba en entornos simulados para corroborar la funcionalidad y eficiencia del mismo.

6. Pruebas del sistema

6.1 Pruebas unitarias

Las pruebas de software se dividen en dos: Pruebas de caja negra y blanca. La gran diferencia entre estas es que, en las de caja negra, nuestra única intención es meter una serie de datos y ver qué salidas proporciona la función en cuestión. Esto incluye el hecho de si nos esperábamos ese tipo de salida de datos o no. Sin embargo, las pruebas de caja blanca van más allá de lo anterior pues también se encargan de medir si el programa funciona o no línea por línea.
Sabiendo esta definición, se procede a integrar todas las pruebas realizadas:

6.1.1 Pruebas de caja negra

Descripción: Descripcion de la funcion

Llamada: Llamada con dato de entrada especifico

Valor de la variable: Valor que de la variable que almacena el return de la funcion.

Funciona: El programa continua con su ejecución o *crashea*.

Nota:

En *False* nos referimos a un valor de entrada correcta pero no el esperado.

En caso de *TRUE el valor es el esperado.*

En caso de *NULL* la función no recibe nada.

Cuestionarios

1._ almacenarCuestionarios()

Descripción: Almacena datos de cuestionario y preguntas en la Base de Datos.

Prueba	Descripción	Llamada	Valor Variable	Funciona
1	Petición HTTP	*(_SESSION[‘verifier_user_id], “NULL”)*	*NULL*	NO
2	Petición HTTP	*(_SESSION[‘verifier_user_id], “FALSE”)*	*FALSE*	SI
3	Petición HTTP	*(_SESSION[‘verifier_user_id], “TRUE”)*	*TRUE*	SI

2._RecibirDatosCuestionario()

Descripción: Devuelve los datos del cuestionario en forma de array.

Prueba	Descripción	Llamada	Valor Variable	Funciona
1	Petición HTTP	*(_POST[‘id_cuestionario’], “NULL”)*	*FALSE*	SI
2	Petición HTTP	*(_POST[‘id_cuestionario’], “FALSE”)*	*FALSE*	SI
3	Petición HTTP	*(_POST[‘id_cuestionario’], “TRUE”)*	TRUE	SI
4	Petición HTTP	*(_POST[‘created_at’], “NULL”)*	FALSE	SI
5	Petición HTTP	*(_POST[‘created_at’], “FALSE”)*	FALSE	SI
6	Petición HTTP	*(_POST[‘created_at’], “TRUE”)*	TRUE	SI
7	Petición HTTP	*(_POST[‘tipo], “NULL”)*	FALSE	SI
8	Petición HTTP	*(_POST[‘tipo], “FALSE”)*	FALSE	SI
9	Petición HTTP	*(_POST[‘tipo], “TRUE”)*	TRUE	SI

3._RecibirDatosPreguntas($uid, $uid_cuestionario)

Descripción: Devuelve el resultado del cuestionario.

Prueba	Descripción	Llamada	Valor Variable	Funciona
1	UID del usuario	*($uid, “NULL”)*	TRUE	SI
2	UID del usuario	*($uid, “FALSE”)*	TRUE	SI

3	UID del usuario	*($uid, “TRUE”)*	TRUE	SI
4	ID de cuestionario	*($id_cuestionario, “NULL”)*	TRUE	SI
5	ID de cuestionario	*($id_cuestionario, “FALSE”)*	TRUE	SI
6	ID de cuestionario	*($id_cuestionario, “TRUE”)*	TRUE	SI
7	Peticion HTTP	*(_POST[‘pregunta’], “NULL”)*	TRUE	SI
8	Peticion HTTP	*(_POST[‘pregunta’], “FALSE”)*	TRUE	SI
9	Peticion HTTP	*(_POST[‘pregunta’], “TRUE”)*	TRUE	SI
10	Peticion HTTP	*(_POST[‘resultado”], “NULL”)*	TRUE	SI
11	Peticion HTTP	*(_POST[‘resultado’], “FALSE”)*	TRUE	SI
12	Peticion HTTP	*(_POST[‘resultado’], “TRUE”)*	TRUE	SI

4._SubirDatosPreguntas($datos, $uid, $id_cuestionario)

Descripción: Se almacenan los datos de las preguntas en la base de datos.

Prueba	Descripción	Llamada	Valor Variable	Funciona
1	Datos Cuestionario	*($id_cuestionario, “NULL”)*	*NULL*	SI
2	Datos Cuestionario	*($id_cuestionario, “FALSE”)*	*FALSE*	SI
3	Datos Cuestionario	*($id_cuestionario, “*TRUE*”)*	TRUE	SI
4	ID de cuestionario	*($id_cuestionario, “NULL”)*	NULL	NO
5	ID de cuestionario	*($id_cuestionario, “FALSE”)*	FALSE	NO
6	ID de cuestionario	*($id_cuestionario, “FALSE”)*	TRUE	NO

Login

1._ LoginBDyAU.php

Descripción: Inicia sesión y devuelve el UID del usuario

Prueba	Descripción	Llamada	Valor Variable	Funciona
1	Valor devuelto por una funcion: *validarCadena()*	*(_POST[‘identificador’], “NULL”)*	*TRUE*	SI
2	Valor devuelto por una funcion: *validarCadena()*	*(_POST[‘identificador’], “FALSE”)*	*TRUE*	SI
3	Valor devuelto por una funcion:	*(_POST[‘identificador’], “TRUE”)*	TRUE	SI

	validarCadena()			
4	Valor devuelto por una funcion: *getIserByEmail("$email")*	*$email = NULL*	NULL	SI
5	Valor devuelto por una funcion: *getIserByEmail("$email")*	*$email = FALSE*	FALSE	SI
6	Valor devuelto por una funcion: *getIserByEmail("$email")*	*$email = TRUE*	TRUE	SI
7	Valor devuelto por una funcion: *signInWithEmailAndPassword($email, $password)*	*$email = NULL* *$password = NULL*	NULL	SI
8	Valor devuelto por una funcion: *signInWithEmailAndPassword($email, $password)*	*$email = NULL* *$password =FALSE*	NULL	SI
9	Valor devuelto por una funcion: *signInWithEmailAndPassword($email, $password)*	*$email = NULL* *$password = TRUE*	NULL	SI
10	Valor devuelto por una funcion: *signInWithEmailAndPassword($email, $password)*	*$email = TRUE* *$password = NULL*	NULL	SI

11	Valor devuelto por una funcion: *signInWithEmailAndPassword($email, $password)*	*$email = FALSE* *$password = NULL*	NULL	SI
12	Valor devuelto por una funcion: *signInWithEmailAndPassword($email, $password)*	*$email = FALSE* *$password = TRUE*	FALSE	SI
13	Valor devuelto por una funcion: *signInWithEmailAndPassword($email, $password)*	*$email = TRUE* *$password = FALSE*	FALSE	SI
14	Valor devuelto por una funcion: *signInWithEmailAndPassword($email, $password)*	*$email = TRUE* *$password = TRUE*	TRUE	SI

Registro

2._ registroBDyAU.php

Descripcion: El usuario se registra

Prueba	Descripción	Llamada	Valor Variable	Funciona
1	Valor devuelto por una funcion: *validarCadena()*	*$cadena*	*TRUE*	SI
2	Valor devuelto por una funcion: *validarCadena()*	*$cadena*	*TRUE*	SI
3	Valor devuelto por una funcion:	*$cadena*	TRUE	SI

	validarCadena()			
4	Valor devuelto por una funcion: *Empty()*	*Empty(*NULL*)*	FALSE	SI
5	Valor devuelto por una funcion: *Empty()*	*Empty(FALSE)*	FALSE	SI
6	Valor devuelto por una funcion: *Empty()*	*Empty(TRUE)*	TRUE	SI
7	Valor devuelto por una funcion: *CompararUsuarios*	*$email = FALSE or TRUE* *$nick = TRUE or FALSE*	FALSE	SI
8	Valor devuelto por una funcion: *CompararUsuarios*	*$email = TRUE* *$nick = TRUE*	TRUE	SI
9	Valor devuelto por una funcion: *comprobarCadenas()*	*$password = TRUE or FALSE* *$conPass = FALSE or TRUE*	FALSE	SI
	lor devuelto por una funcion: *comprobarCadenas()*	*$password = TRUE* *$conPass = TRUE*	TRUE	SI
10	Valor devuelto por una funcion: *createUser()*	*$password = TRUE* *$conPass = TRUE*	TRUE	SI

11	Valor devuelto por una funcion: *createUser()*	*$userProperties = NULL or FALSE*	FALSE	NO
12	Valor devuelto por una funcion: *createUser()*	*$userProperties = TRUE*	TRUE	SI

3._ ConexionBD.php

Descripcion: Funciones Auxiliares para poder conectarnos a la Base de datos y realizare el login y registro correctamente.

Prueba	Descripción	Llamada	Valor Variable	Funciona
1	*validarCadena()*	*$cadena*	*TRUE*	SI
2	*validarCadena()*	*$cadena*	*TRUE*	SI
3	*validarCadena()*	*$cadena*	*TRUE*	SI
4	*comprobarCadenas()*	*$password = TRUE or FALSE* *$conPass = FALSE or TRUE*	FALSE	SI
5	*comprobarCadenas()*	*$password = TRUE* *$conPass = TRUE*	TRUE	SI

6	*comprobarCadenas()*	*$password = NULL* *$conPass =NULL*	NULL	NO
7	*CompararUsuarios()*	*$email = FALSE or TRUE* *$nick = TRUE or FALSE*	FALSE	SI
8	*CompararUsuarios()*	*$email = TRUE* *$nick = TRUE*	TRUE	SI
9	*subirDatosRegistro($collection, $datos, $uid)*	*$collection = NULL* *$datos =* *$uid =*	NULL	NO
10	*subirDatosRegistro($collection, $datos, $uid)*	*$collection = FALSE* *$datos =* *$uid*	NULL	NO
11	*subirDatosRegistro($collection, $datos, $uid)*	*$collection = TRUE* *$datos = TRUE* *$uid = TRUE*	TRUE	SI
12	*subirDatosRegistro($collection, $datos, $uid)*	*$collection = TRUE* *$datos = TRUE* *$uid = TRUE or FALSE*	TRUE	SI

4._ ConexionesAU.php

Gestión del Diario

1._ GestionDiarioLocal.php

CrearDiario

Descripcion: Crear archivo Diario cuando el usuario accede por primera vez a *Diario*.

Prueba	Descripción	Llamada	Valor Variable	Funciona
1	Petición HTTP	*(_SESSION[‘verifier_user_id], “NULL”)*	*NULL*	NO
2	Petición HTTP	*(_SESSION[‘verifier_user_id], “FALSE”)*	*FALSE*	SI
3	Petición HTTP	*(_SESSION[‘verifier_user_id], “TRUE”)*	*TRUE*	SI
4	Valor devuelto por la funcion: *Fopen()*	*Fopen(ruta fichero, modo apertura)*	NULL O FALSE	NO
5	Valor devuelto por la funcion: *Fopen()*	*Fopen(ruta fichero, modo apertura)*	TRUE	SI

CrearPaginaLOCAL()

Descripcion: Crea una pagina y lo almacena en un archivo *.json*

Prueba	Descripción	Llamada	Valor Variable	Funciona

1	Petición HTTP	*(_SESSION[‘verifier_user_id], “NULL”)*	*NULL*	NO
2	Petición HTTP	*(_SESSION[‘verifier_user_id], “FALSE”)*	*FALSE*	SI
3	Petición HTTP	*(_SESSION[‘verifier_user_id], “TRUE”)*	*TRUE*	SI
4	Valor devuelto por la funcion: *File_get_contents()*	*File_get_contents(ruta fichero, modo apertura)*	NULL O FALSE	NO
5	Valor devuelto por la funcion: *File_get_contents()*	*File_get_contents(ruta fichero, modo apertura)*	TRUE	SI
6	Valor devuelto por la funcion: *Array_merge()*	*Array_merge($array1, $array2)*	NULL O FALSE	NO
7	Valor devuelto por la funcion: *Array_merge()*	*Array_merge($array1, $array2)*	TRUE	SI
8	Valor devuelto por la funcion: *File_put_contents()*	*File_put_contents(ruta)*	NULL O FALSE	NO
9	Valor devuelto por la funcion: *File_put_contents()*	*File_put_contents(ruta)*	TRUE	SI
10	Valor devuelto por la funcion: *Is_connected()*	*Is_connected()*	NULL OR FALSE	SI
11	Valor devuelto por la funcion: *Is_connected()*	*Is_connected()*	TRUE	SI

12	Valor devuelto por la funcion: *RecibirDatosClienteLOCAL*	*$var = RecibirDatosClienteLOCAL()*	NULL	SI
13	Valor devuelto por la funcion: *RecibirDatosClienteLOCAL*	*$var = RecibirDatosClienteLOCAL()*	FALSE	SI
14	Valor devuelto por la funcion: *RecibirDatosClienteLOCAL*	*$var = RecibirDatosClienteLOCAL()*	TRUE	SI

Descripción: devuelve el contenido de la pagina indicada por la fecha

Prueba	Descripción	Llamada	Valor Variable	Funciona
1	Petición HTTP	*(_SESSION[‘verifier_user_id], “NULL”)*	*NULL*	NO
2	Petición HTTP	*(_SESSION[‘verifier_user_id], “FALSE”)*	*FALSE*	SI
3	Valor devuelto por la funcion: *File_get_contents()*	*File_get_contents(ruta fichero, modo apertura)*	NULL O FALSE	NO
4	Valor devuelto por la funcion: *File_get_contents()*	*File_get_contents(ruta fichero, modo apertura)*	TRUE	SI

5	Valor devuelto por la funcion: *File_get_contents()*	*File_get_contents(ruta fichero, modo apertura)*	NULL O FALSE	NO
6	$fecha	*DescargarPaginaLOCAL($fecha)*	FALSE	SI
7	$fecha	*DescargarPaginaLOCAL($fecha)*	TRUE	SI
9	$fecha	*DescargarPaginaLOCAL($fecha)*	NULL	NO

ActualizarPaginaLOCAL()

Descripción: Actualiza pagina del Diario que ya este creada.

1	Petición HTTP	*(_SESSION[‘verifier_user_id], “NULL”)*	*NULL*	NO
2	Petición HTTP	*(_SESSION[‘verifier_user_id], “FALSE”)*	*FALSE*	SI
3	Petición HTTP	*(_SESSION[‘verifier_user_id], “TRUE”)*	*TRUE*	SI
4	Valor devuelto por la funcion: *File_get_contents()*	*File_get_contents(ruta fichero, modo apertura)*	NULL O FALSE	NO
5	Valor devuelto por la funcion: *File_get_contents()*	*File_get_contents(ruta fichero, modo apertura)*	TRUE	SI
6	Valor devuelto por la funcion: *Array_merge()*	*Array_merge($array1, $array2)*	NULL O FALSE	NO

7	Valor devuelto por la funcion: *Array_merge()*	*Array_merge($array1, $array2)*	TRUE	SI
8	Valor devuelto por la funcion: *File_put_contents()*	*File_put_contents(ruta)*	NULL O FALSE	NO
9	Valor devuelto por la funcion: *File_put_contents()*	*File_put_contents(ruta)*	TRUE	SI
10	Valor devuelto por la funcion: *Is_connected()*	*Is_connected()*	NULL OR FALSE	SI
11	Valor devuelto por la funcion: *Is_connected()*	*Is_connected()*	TRUE	SI
12	Valor devuelto por la funcion: *RecibirDatosClienteLOCAL*	*$var = RecibirDatosClienteLOCAL( )*	NULL	SI
13	Valor devuelto por la funcion: *RecibirDatosClienteLOCAL*	*$var = RecibirDatosClienteLOCAL( )*	FALSE	SI
14	Valor devuelto por la funcion: *RecibirDatosClienteLOCAL*	*$var = RecibirDatosClienteLOCAL( )*	TRUE	SI

RecibirDatos.php

CrearJSONcuestionarios()

descripción: Crea un archivo con extensión. json para almacenar el resultado de los cuestionarios.

Prueba	Descripción	Llamada	Valor Variable	Funciona
1	Petición HTTP	*(_SESSION[‘verifier_user_id], “NULL”)*	*NULL*	NO
2	Petición HTTP	*(_SESSION[‘verifier_user_id], “FALSE”)*	*FALSE*	SI
3	Petición HTTP	*(_SESSION[‘verifier_user_id], “TRUE”)*	*TRUE*	SI
4	Valor devuelto por la funcion: *Fopen()*	*Fopen(ruta fichero, modo apertura)*	NULL O FALSE	NO
5	Valor devuelto por la funcion: *Fopen()*	*Fopen(ruta fichero, modo apertura)*	TRUE	SI

DescargarResultadoCuesitonarios()

descripción: Descarga el resultado de los cuestionarios, los sube a la base de datos y almacena una copia Local

Prueba	Descripción	Llamada	Valor Variable	Funciona
1	Petición HTTP	*(_SESSION[‘verifier_user_id], “NULL”)*	*NULL*	NO
2	Petición HTTP	*(_SESSION[‘verifier_user_id], “FALSE”)*	*FALSE*	SI
3	Petición HTTP	*(_SESSION[‘verifier_user_id], “TRUE”)*	*TRUE*	SI

10	Valor devuelto por la funcion: *Is_connected()*	*Is_connected()*	NULL OR FALSE	SI
11	Valor devuelto por la funcion: *Is_connected()*	*Is_connected()*	TRUE	SI
6	Valor devuelto por la funcion: *Array_merge()*	*Array_merge($array1, $array2)*	NULL O FALSE	NO
7	Valor devuelto por la funcion: *Array_merge()*	*Array_merge($array1, $array2)*	TRUE	SI
6	Valor devuelto por la funcion: *Array_push()*	*Array_push ($array1, $array2)*	NULL O FALSE	NO
7	Valor devuelto por la funcion: *Array_push()*	*Array_push ($array1, $array2)*	TRUE	SI
4	Valor devuelto por la funcion: *File_get_contents()*	*File_get_contents(ruta fichero, modo apertura)*	NULL O FALSE	NO
5	Valor devuelto por la funcion: *File_get_contents()*	*File_get_contents(ruta fichero, modo apertura)*	TRUE	SI
8	Valor devuelto por la funcion: *File_put_contents()*	*File_put_contents(ruta)*	NULL O FALSE	NO

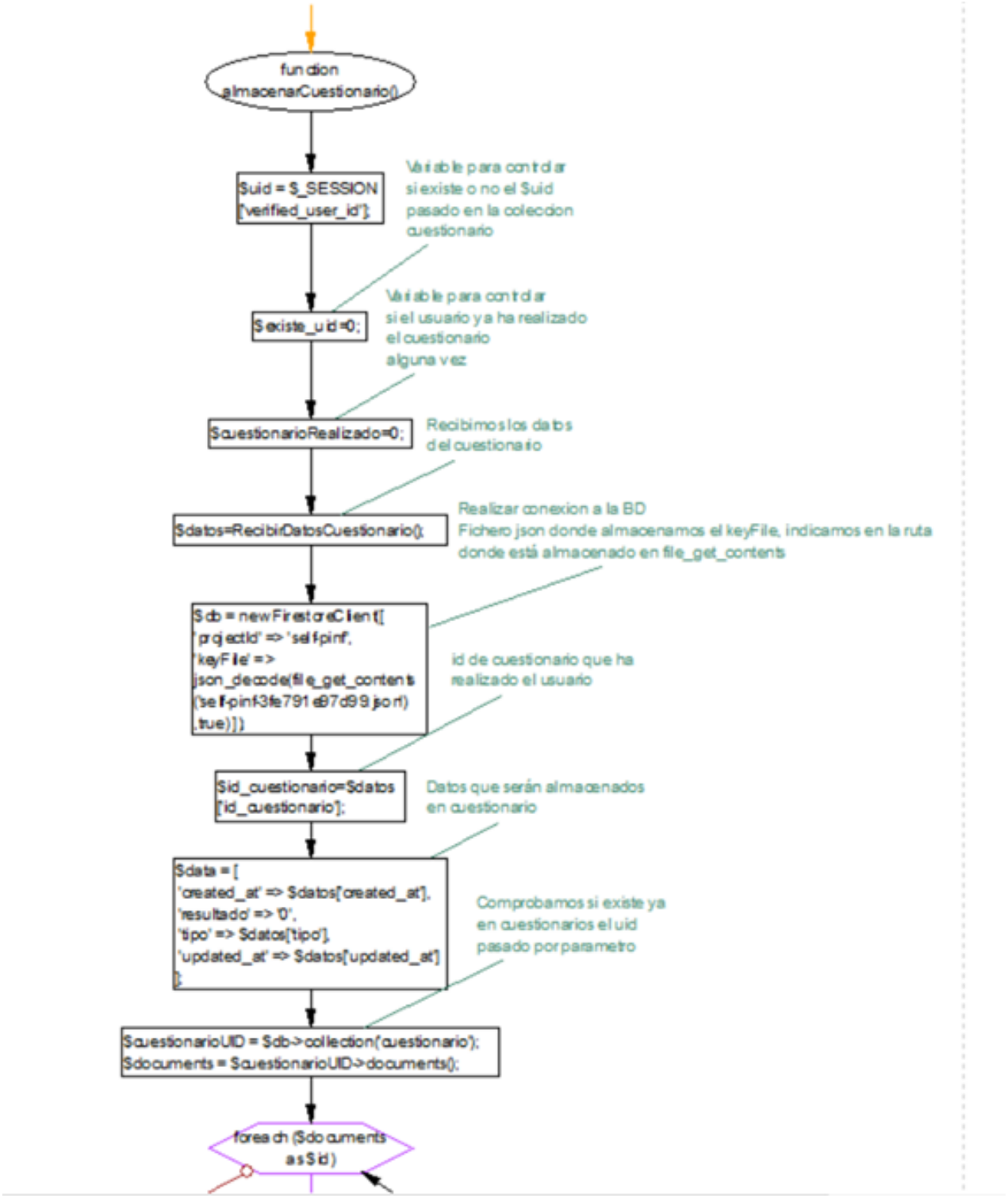
9	Valor devuelto por la funcion: *File_put_contents()*	*File_put_contents(ruta)*	TRUE	SI
10	Valor devuelto por la funcion: *Object_array()*	*Object_array($objecto)*	NULL or FALSE	NO
11	Valor devuelto por la funcion: *Object_array()*	*Object_array($objecto)*	TRUE	SI

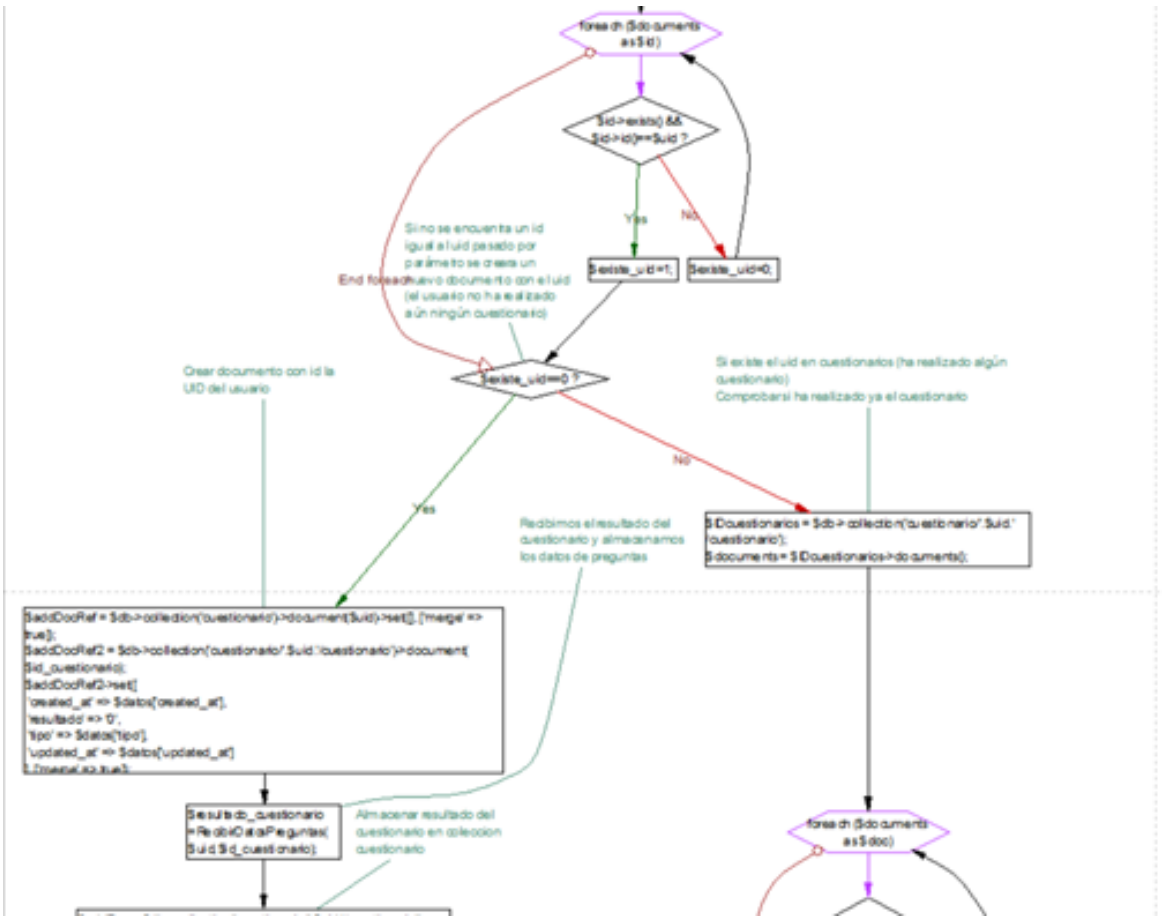
6.1.2 PRUEBAS DE CAJA BLANCA

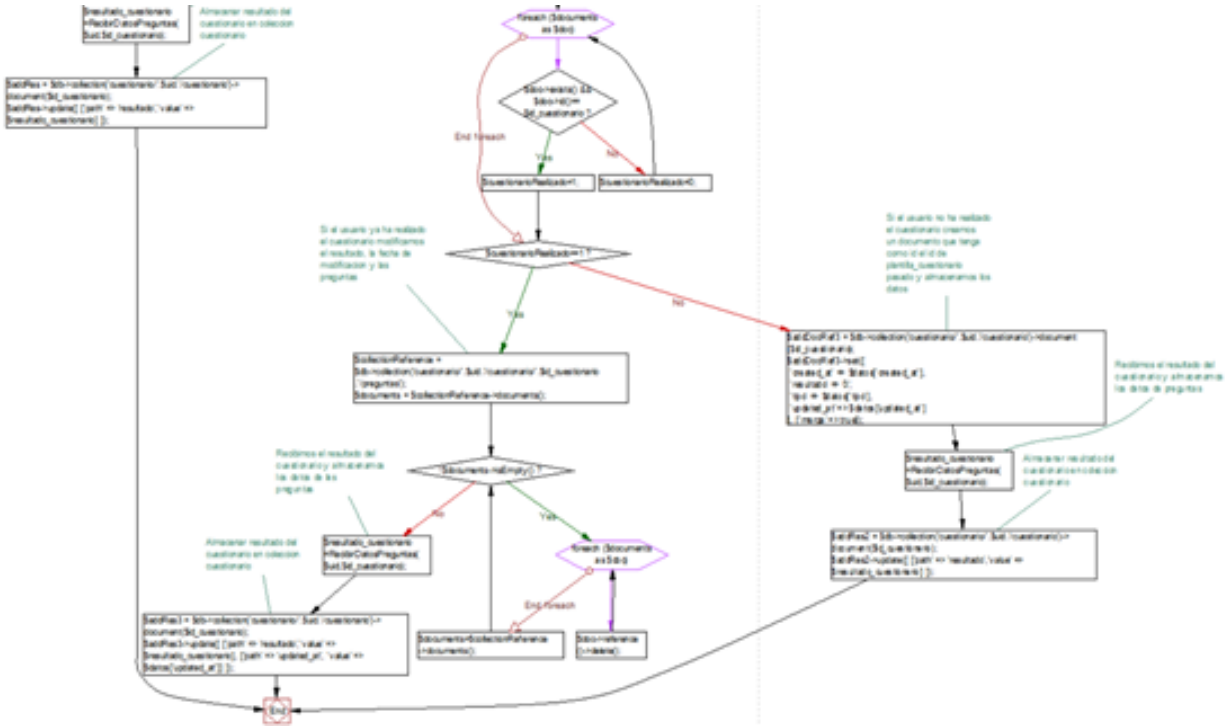
Se realizará la prueba de ruta básica a cada función del back end, para ello se genera el diagrama de flujo de cada función, se calculará la complejidad ciclomática de cada función y obtendremos las rutas independientes, e indicamos que valores tiene que tomar el código para que se ejecute cada una de estas las rutas.

- **Cuestionarios:**

-**almacenarCuestionarios()**




foreach ($documents as $id)
$id->exists() && $id->id()==$uid ?
Yes
No
$existe_uid=1;
$existe_uid=0;
Si no se encuentra un id igual al uid pasado por parámetro se crea un nuevo documento con el uid (el usuario no ha realizado aún ningún cuestionario)
End foreach
$existe_uid==0 ?
Crear documento con id la UID del usuario
Si existe el uid en cuestionarios (ha realizado algún cuestionario)
Comprobarsi ha realizado ya el cuestionario
Yes
No
Recibimos el resultado del cuestionario y almacenamos los datos de preguntas
$addDocRef = $db->collection('cuestionarios')->document($uid)->set([], ['merge' => true]);
$addDocRef2 = $db->collection('cuestionarios/' $uid.'/cuestionarios')->document($id_cuestionario);
$addDocRef2->set([
'created_at' => $datos['created_at'],
'resultados' => '0',
'tipo' => $datos['tipo'],
'updated_at' => $datos['updated_at']
], ['merge' => true]);
$resultado_cuestionario =RecibeDatosPreguntas($uid, $id_cuestionario);
Almacenar resultado del cuestionario en coleccion cuestionario
$documents= $Cuestionarios->documents();
foreach ($documents as $doc)


$resultado_cuestionario =RecibeDatosPreguntas($uid,$id_cuestionario);
Almacenar resultado del cuestionario en coleccion cuestionario
$addRes = $db->collection('cuestionarios/' $uid.'/cuestionarios')->document($id_cuestionario);
$addRes->update([['path' => 'resultado', 'value' => $resultado_cuestionario]]);
foreach ($documents as $doc)
$doc->exists() && $doc->id()== $id_cuestionario ?
End foreach
Yes
No
$cuestionarioRealizado=1;
$cuestionarioRealizado=0;
Si el usuario ya ha realizado el cuestionario modificamos el resultado, la fecha de modificacion y las preguntas
$cuestionarioRealizado== 1 ?
Yes
No
Si el usuario no ha realizado el cuestionario creamos un documento que tenga como id el id de plantilla_cuestionario pasado y almacenamos los datos
$addDocRef = $db->collection('cuestionarios/' $uid.'/cuestionarios')->document($id_cuestionario);
$addDocRef->set([
'created_at' => $datos['created_at'],
'resultado' => '1',
'tipo' => $datos['tipo'],
'updated_at'=>$datos['updated_at']
], ['merge' =>true]);
Recibimos el resultado del cuestionario y almacenamos los datos de preguntas
$resultado_cuestionario =RecibeDatosPreguntas($uid,$id_cuestionario);
Almacenar resultado del cuestionario en coleccion cuestionario
$addRes = $db->collection('cuestionarios/' $uid.'/cuestionarios')->document($id_cuestionario);
$addRes->update([['path' => 'resultado','value' => $resultado_cuestionario]]);
$colleccionReferencia = $db->collection('cuestionarios/' $uid.'/cuestionarios/' $id_cuestionario /'preguntas');
$documents = $colleccionReferencia->documents();
$documents->isEmpty() ?
No
Yes
Recibimos el resultado del cuestionario y almacenamos los datos de las preguntas
$resultado_cuestionario =RecibeDatosPreguntas($uid,$id_cuestionario);
Almacenar resultado del cuestionario en coleccion cuestionario
$addRes = $db->collection('cuestionarios/' $uid.'/cuestionarios')->document($id_cuestionario);
$addRes->update([['path' => 'resultado', 'value' => $resultado_cuestionario], ['path' => 'updated_at', 'value' => $datos['updated_at']]]);
foreach ($documents as $doc)
End foreach
$documents->colleccionReferencia ->documents();
$doc->reference ->delete();
End

Nº Nodos Predicados: 5

V(G) = Nº Nodos Predicados + 1 = 5 +1 = 6

6 Rutas: 1-2-3-5-2-6-7

1-2-3-4-2-6-8-9-10

1-2-3-4-2-6-8-9-11-8-12-13

1-2-3-4-2-6-8-9-11-8-12-14

1-2-3-4-2-6-8-12-14

1-2-6-7

C1: $existe_uid=0

C2: $existe_uid=1 && $cuestionariorealizado=0

C3: $existe_uid=1 && $doc->id()==id_cuestionario && $cuestionarioRealizado=1

C4: $existe_uid=1 && $doc->id()!=id_cuestionario && $cuestionarioRealizado=0

C5: $cuestionario_realizado=0

C6: $existe_uid=0

-RecibirDatosCuestionarios()

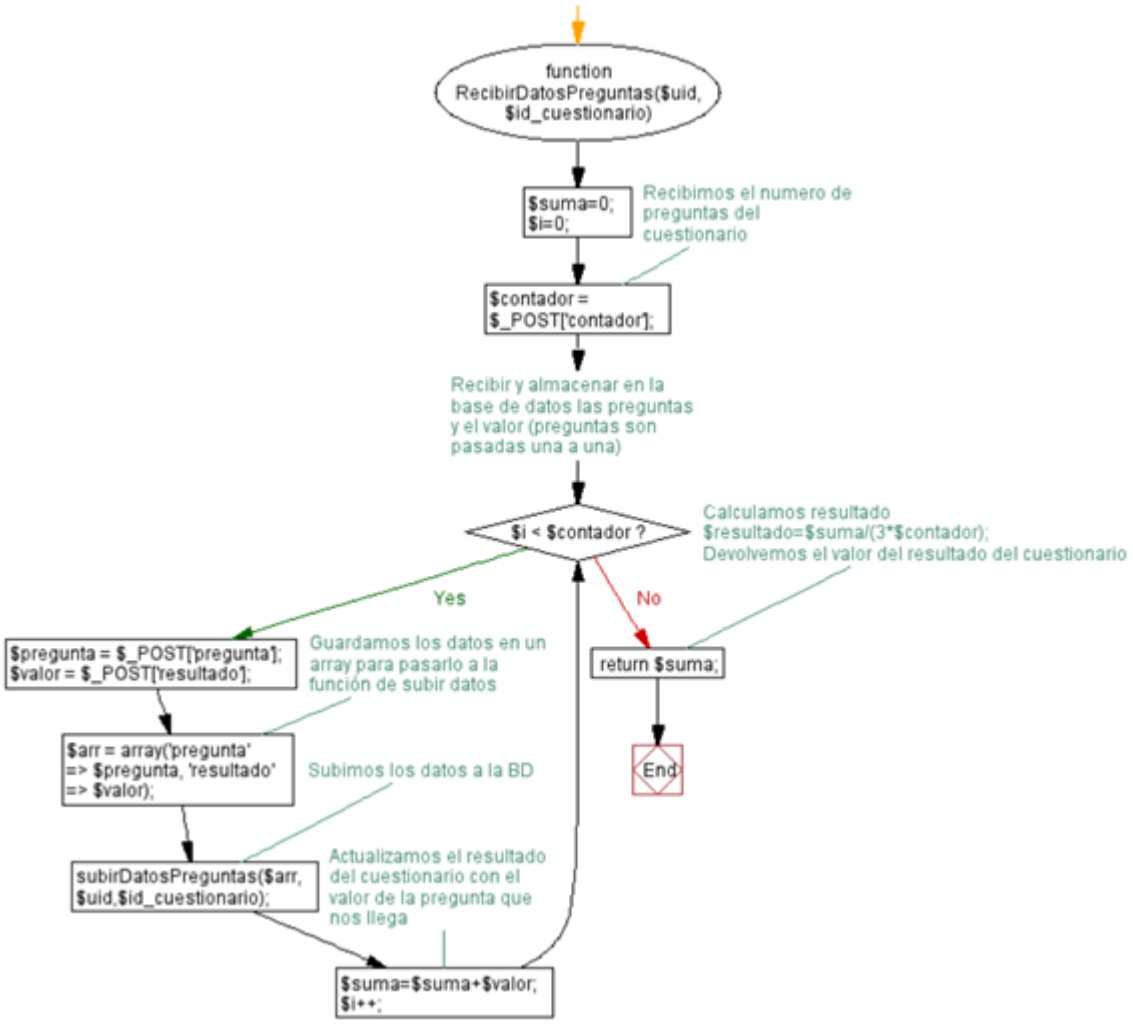
Nº Nodos Predicados: 0

V(G) = Nº Nodos Predicados + 1 = 0 +1 = 1

1 Rutas: 1-2-3

C1: Se reciben datos desde front end, $arr contiene los datos recibidos

-RecibirDatosPregunta($uid, $id_cuestionario)



Nº Nodos Predicados: 1

V(G) = Nº Nodos Predicados + 1 = 1 + 1 = 2

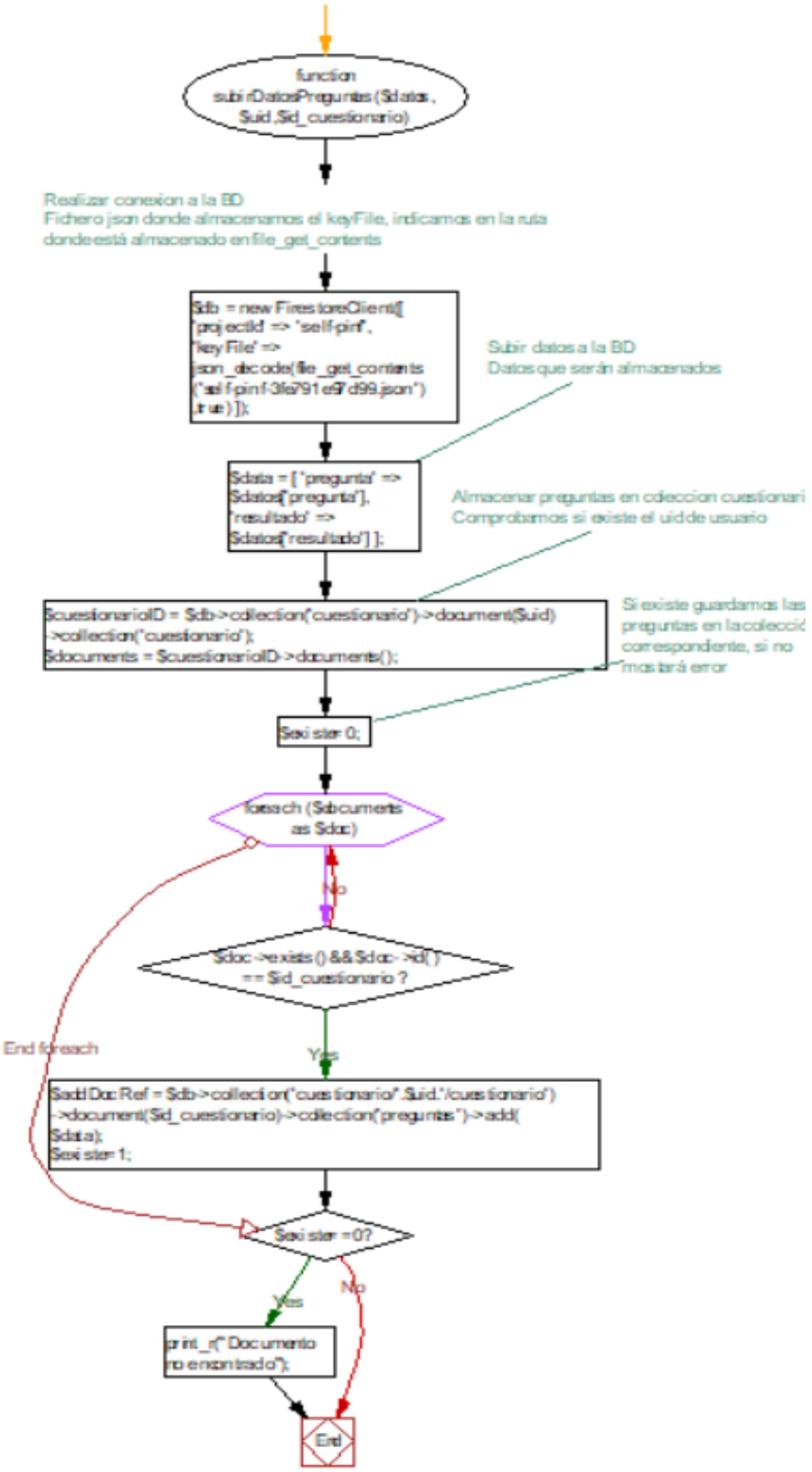
2 Rutas: 1-2-3-4-5-6-7-3-8-9

1-2-3-8-9

C1: contador > 0 (se recibe alguna pregunta)

C2: contador==0 (no se recibe ninguna pregunta)

-subirDatosPreguntas($datos, $uid, $id_cuestionario)



Nº Nodos Predicados: 2

V(G) = Nº Nodos Predicados + 1 = 2 + 1 = 3

3 Rutas: 1-2-3-4-5-6-7-8-9-10

1-2-3-4-5-6-7-8-10

1-2-3-4-5-8-9-10

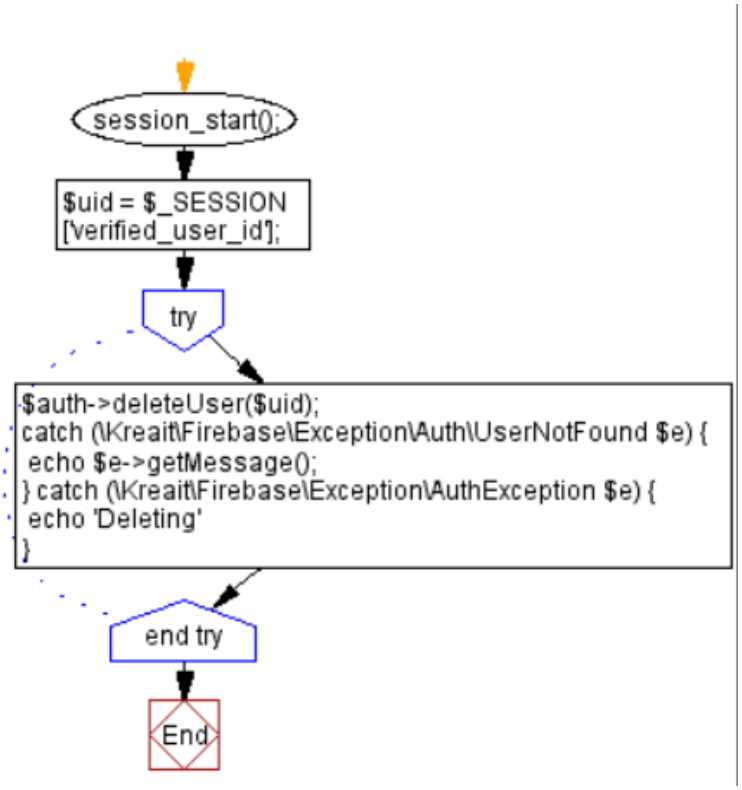
C1: $doc->id() == id_cuestionario && $existe=1

C2: $doc->id() == id_cuestionario && $existe=0

C3: $doc->id() != id_cuestionario

- Borrar datos usuario

-BorrarDatosUsuarios():



Nº Nodos Predicados: 1

V(G) = Nº Nodos Predicados + 1 = 1+1 = 2

2 Rutas: 1-2-3-4-5-6

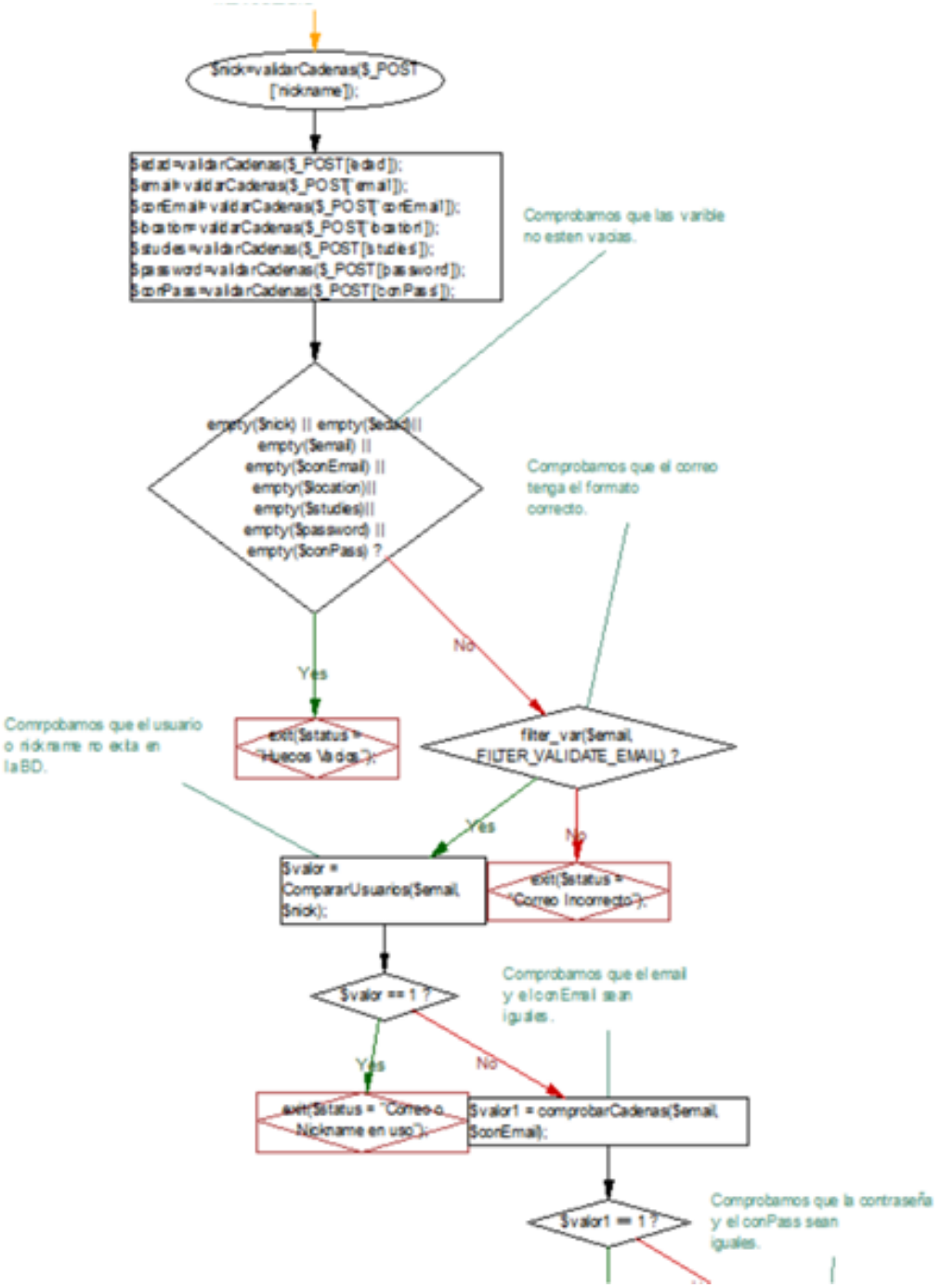
1-2-3-5-6

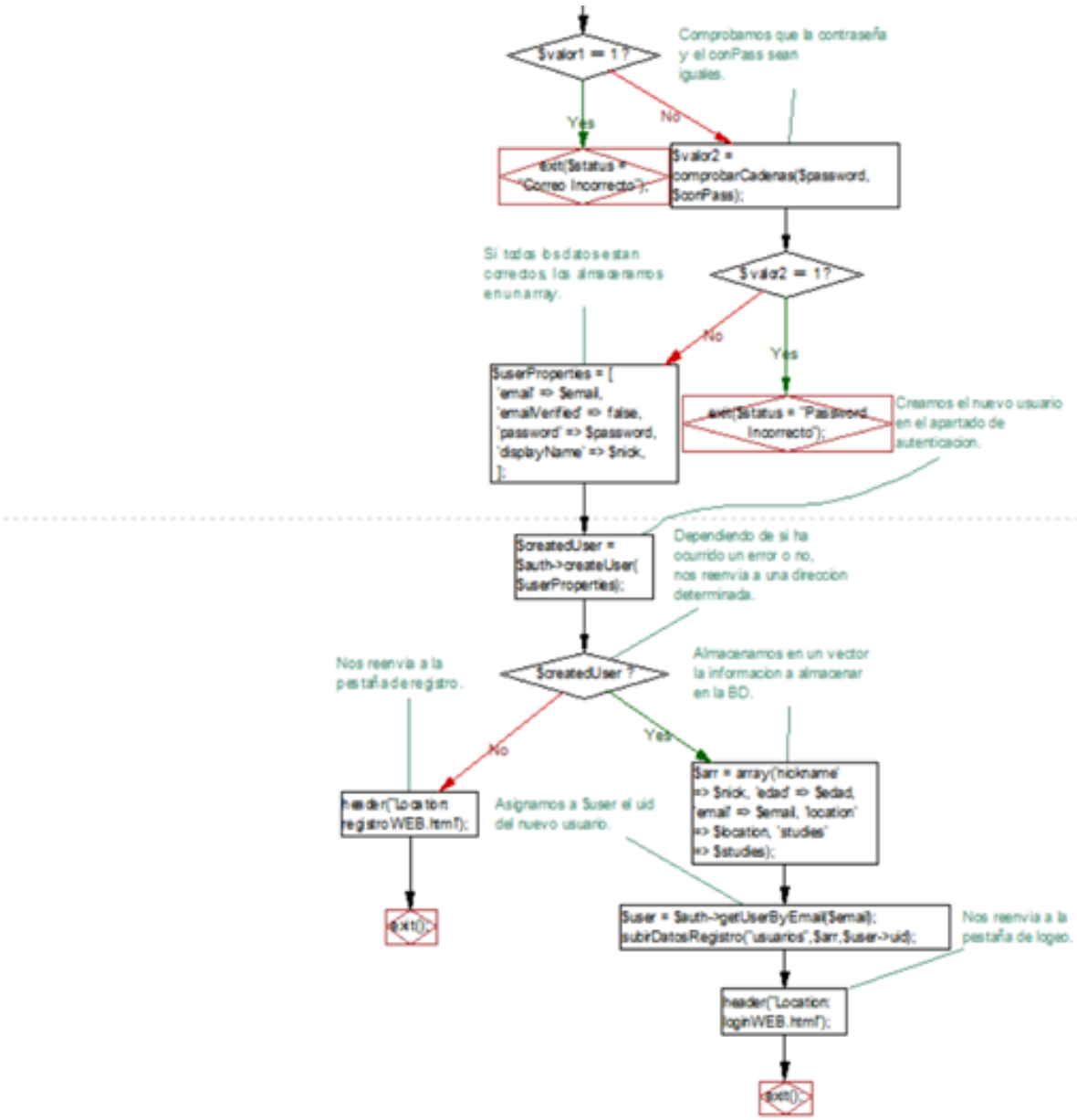
C1: Usuario encontrado y eliminado

C2: Usuario no encontrado (se ejecuta excepción)

- Registrar usuarios y login

-Registro


$nick=validarCadenas($_POST['nickname']);
$edad=validarCadenas($_POST['edad']);
$email=validarCadenas($_POST['email']);
$conEmail=validarCadenas($_POST['conEmail']);
$location=validarCadenas($_POST['location']);
$studies=validarCadenas($_POST['studies']);
$password=validarCadenas($_POST['password']);
$conPass=validarCadenas($_POST['conPass']);
Comprobamos que las varible no esten vacias.
empty($nick) || empty($edad)|| empty($email) || empty($conEmail) || empty($location)|| empty($studies)|| empty($password) || empty($conPass) ?
Yes
No
exit($status = "Huecos Vacios");
Comrpobamos que el usuario o ricknaime no exita en la BD.
Comprobamos que el correo tenga el formato correcto.
filter_var($email, FILTER_VALIDATE_EMAIL) ?
Yes
No
exit($status = "Correo Incorrecto");
$valor = CompararUsuarios($email, $nick);
$valor == 1 ?
Comprobamos que el email y el conEmail sean iguales.
Yes
No
exit($status = "Correo o Nickname en uso");
$valor1 = comprobarCadenas($email, $conEmail);
$valor1 == 1 ?
Comprobamos que la contraseña y el conPass sean iguales.

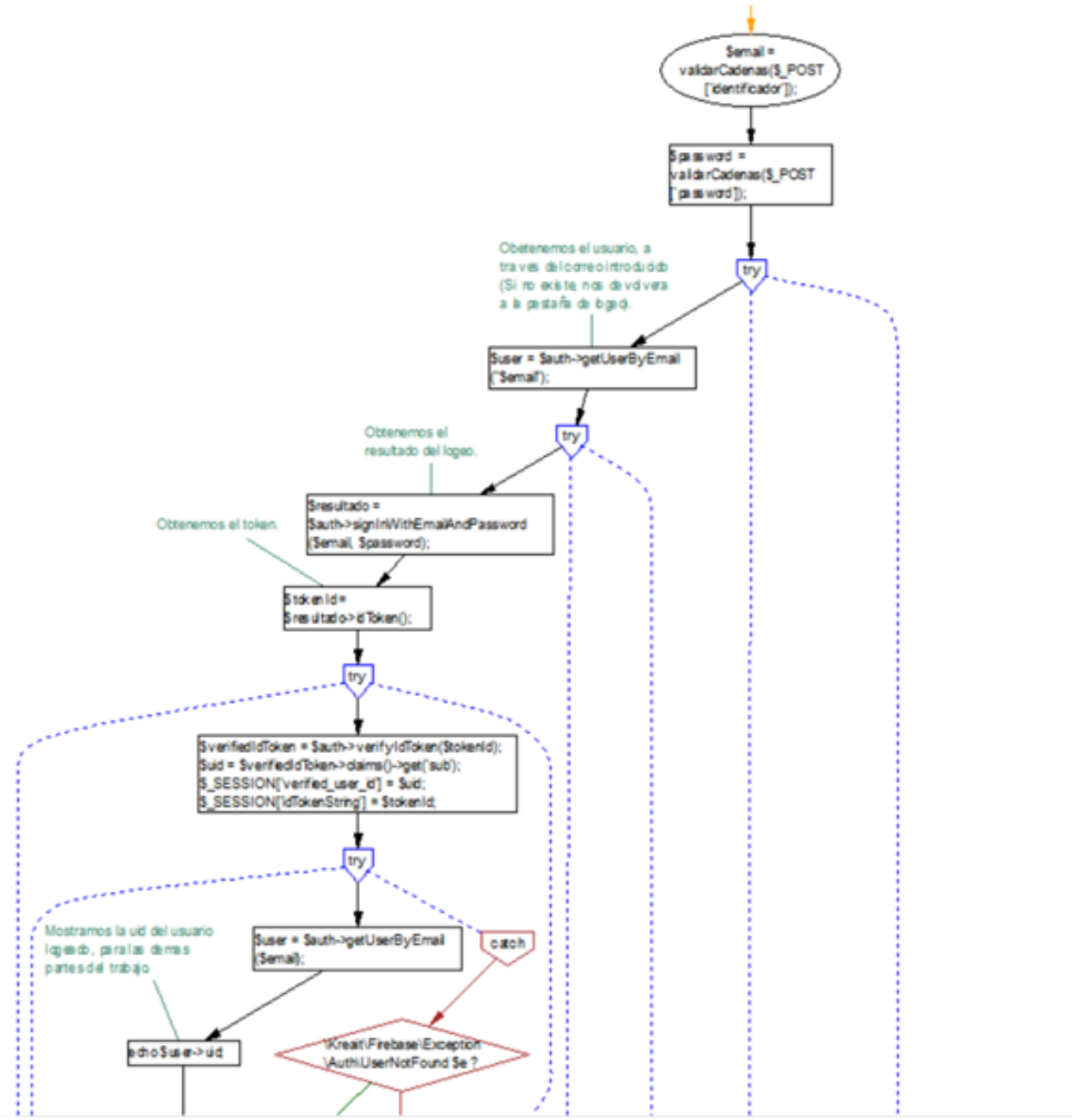


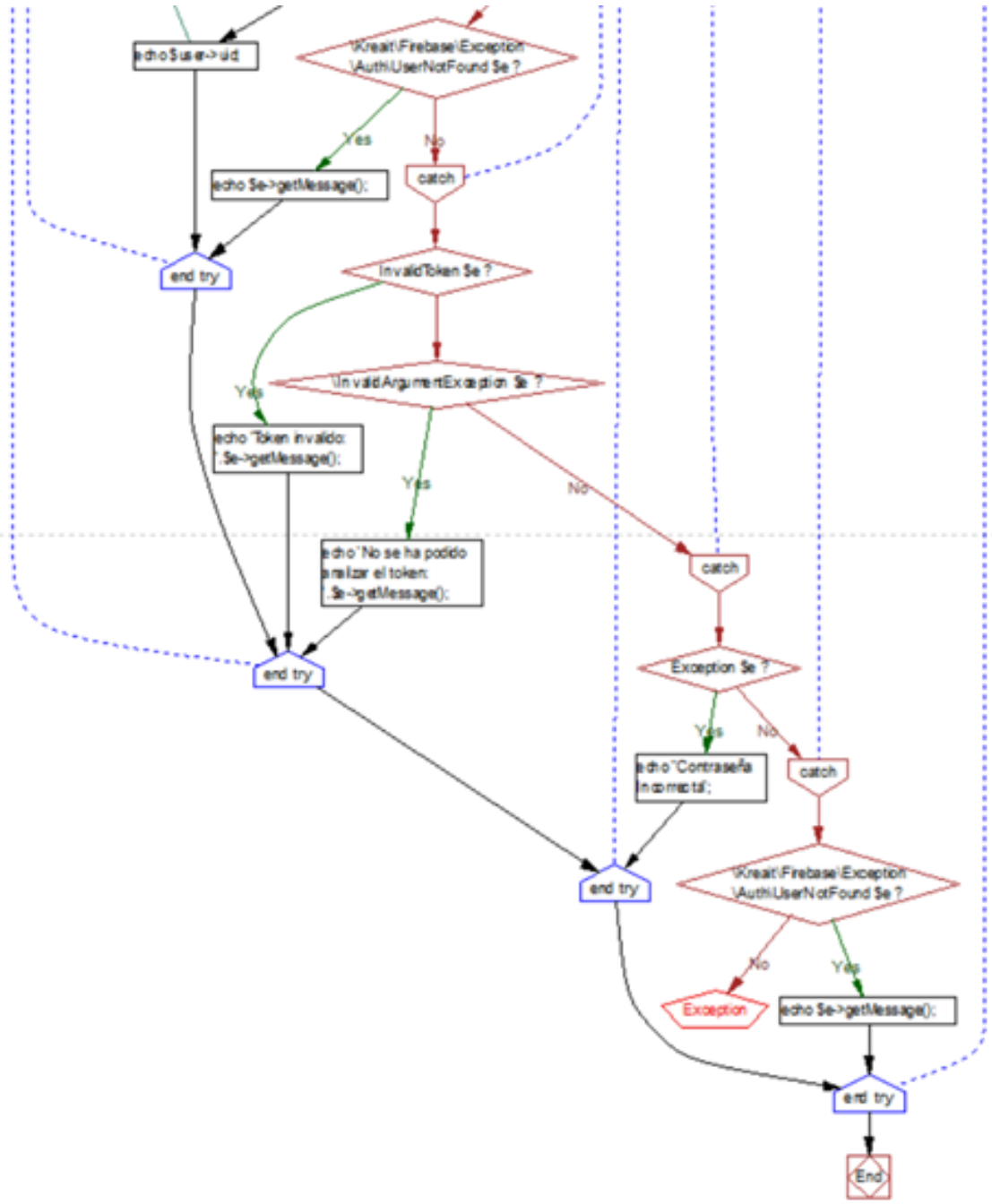
Nº Nodos Predicados: 6
V(G) = Nº Nodos Predicados + 1 = 6+1 = 7

7 Rutas: 1-2-4-6-7-9-10-12-13-15-16-17
1-2-4-6-7-9-10-12-13-15-16-18
1-2-4-6-7-9-10-12-13-14
1-2-3-4-6-7-9-10-11
1-2-4-6-7-8
1-2-4-5
1-2-3

C1: !empty(datos) && filter_var($email, FILTER_VALIDATE_EMAIL)) == True && $valor==$valor1==$valor2==0 && $createdUser==True
C2: !empty(datos) && filter_var($email, FILTER_VALIDATE_EMAIL)) == True && $valor==$valor1==$valor2==0 && $createdUser==False
C3: !empty(datos) && filter_var($email, FILTER_VALIDATE_EMAIL)) == True && $valor==$valor1 == 0 && $valor2==1
C4: !empty(datos) && filter_var($email, FILTER_VALIDATE_EMAIL)) == True && $valor==0 && $valor1== 1
C5: !empty(datos) && filter_var($email, FILTER_VALIDATE_EMAIL)) == True && $valor==1
C6: !empty(datos) && filter_var($email, FILTER_VALIDATE_EMAIL)) == False
C7: empty(datos) (hay algún dato vacio)

-Login


$email = validarCadenas($_POST ['identificador']);
$password = validarCadenas($_POST ['password']);
Obtenemos el usuario, a traves del correo introducido (Si no existe, nos devolvera a la pestaña de logeo).
try
$user = $auth->getUserByEmail ($email);
try
Obtenemos el resultado del logeo.
$resultado = $auth->signInWithEmailAndPassword ($email, $password);
Obtenemos el token.
$tokenId = $resultado->idToken();
try
$verifiedIdToken = $auth->verifyIdToken($tokenId);
$uid = $verifiedIdToken->claims()->get('sub');
$_SESSION['verified_user_id'] = $uid;
$_SESSION['idTokenString'] = $tokenId;
try
Mostramos la uid del usuario logeado, para las demas partes del trabajo
$user = $auth->getUserByEmail ($email);
catch
echo $user->uid;
\Kreait\Firebase\Exception \Auth\UserNotFound $e ?



Nº Nodos Predicados: 5

V(G) = Nº Nodos Predicados + 1 = 5+1 = 6

6 Rutas: 1-2-4-6-7-9-10-12-13-15-16-17
1-2-4-6-7-9-10-12-13-15-16-18
1-2-4-6-7-9-10-12-13-14
1-2-3-4-6-7-9-10-11
1-2-4-6-7-8
1-2-4-5
1-2-3

C1: Se obtiene usuario, se obtienen resultado del login, se obtiene el idtoken, se analiza el token correctamente y es válido, se obtiene la uid de usuario correctamente

C2: Se obtiene usuario, se obtienen resultado del login, se obtiene el idtoken, se analiza el token correctamente y es válido, no se encuentra el identificador del usuario en la base de datos (excepción)

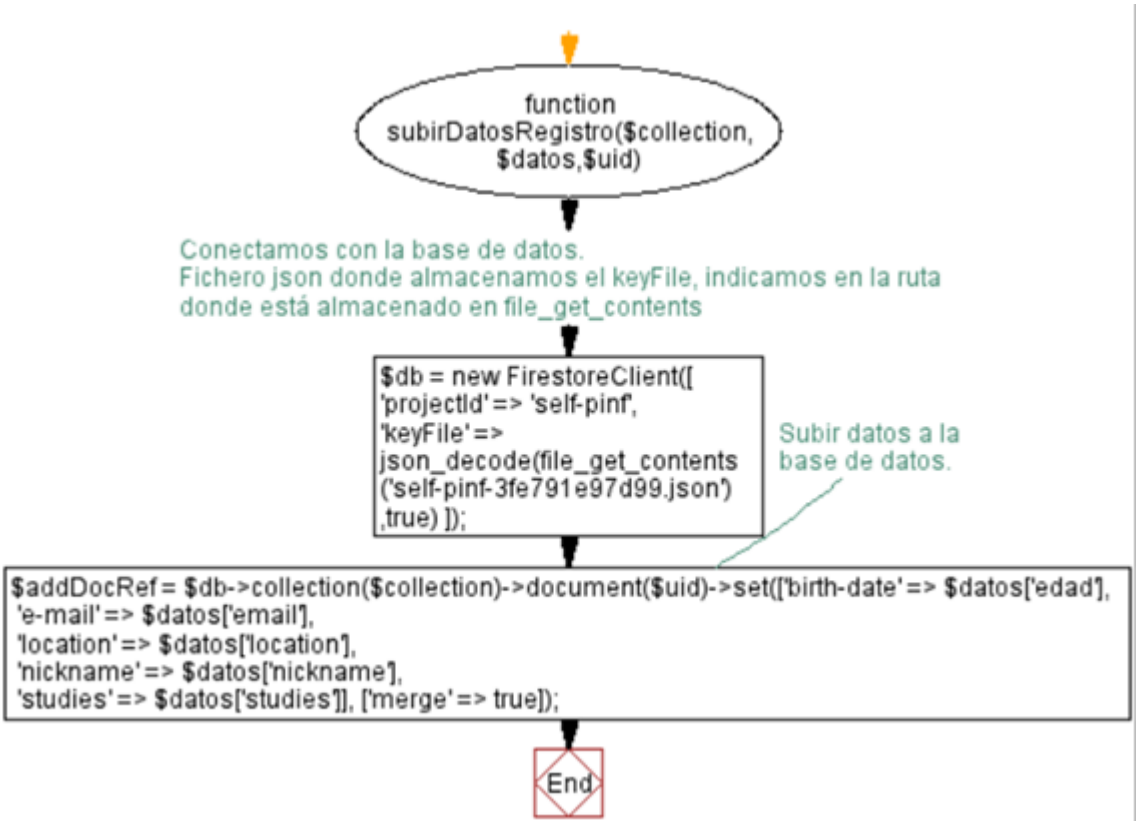
C3: Se obtiene usuario, se obtienen resultado del login, se obtiene el idtoken, se analiza correctamente, pero es inválido

C4: Se obtiene usuario, se obtienen resultado del login, el idtoken no ha podido ser analizado (excepción)

C5: Contraseña incorrecta (excepción)

C6: Usuario no encontrado durante login(excepción)

-subirDatosRegistro($collection, $datos, $uid)



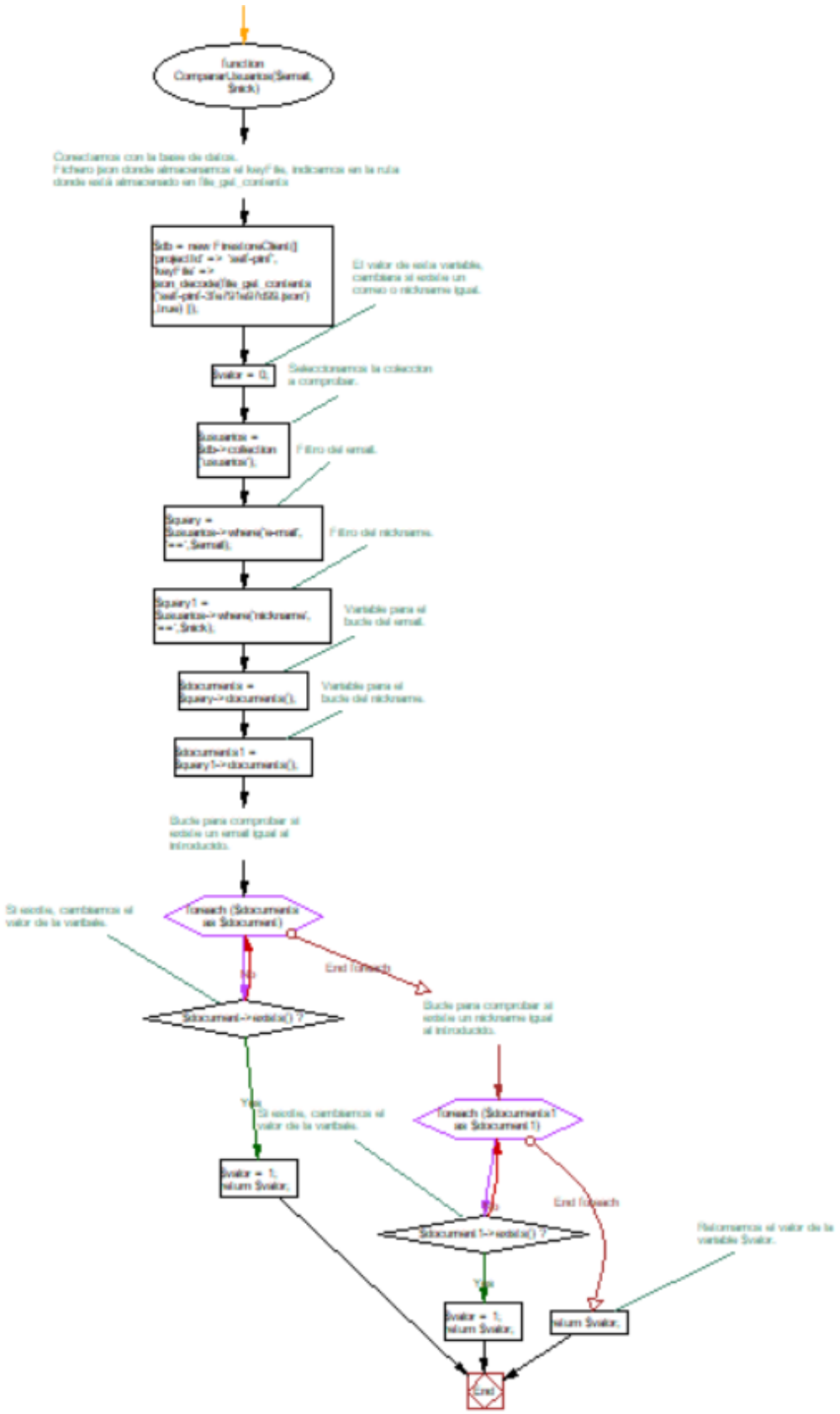
Nº Nodos Predicados: 0

V(G) = Nº Nodos Predicados + 1 = 0+1 = 1

1 Ruta: 1-2-3

C1: Datos no están vacios

-compararUsuarios($email, $nick)



Nº Nodos Predicados: 2

V(G) = Nº Nodos Predicados + 1 = 2+1 = 3

3 Rutas: 1-2-3-4-5-6-7-8-9-10-11-12-13-14

1-2-3-4-5-6-7-8-9-10-11-12-14

1-2-3-4-5-6-7-8-9-14

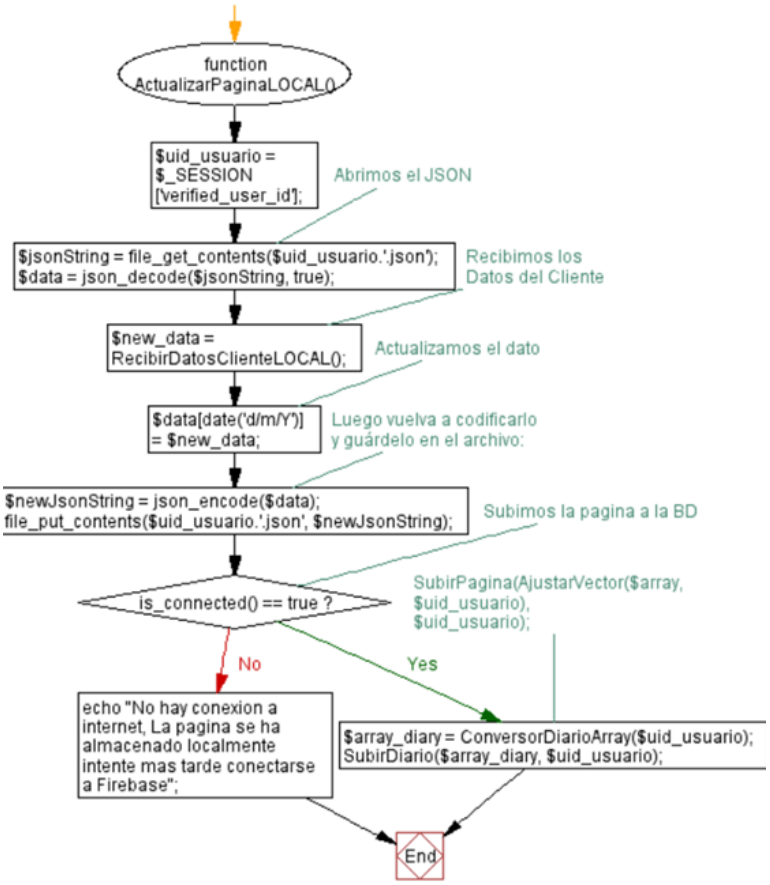
C1: $document->exists() = true && $document1->exists() = true

C2 : $document->exists() = true && $document1->exists() = false

C3 : $document->exists() = false

- Diario

-CrearPaginaLocal()



Nº Nodos Predicados: 2
V(G) = Nº Nodos Predicados + 1 = 2+1 = 3

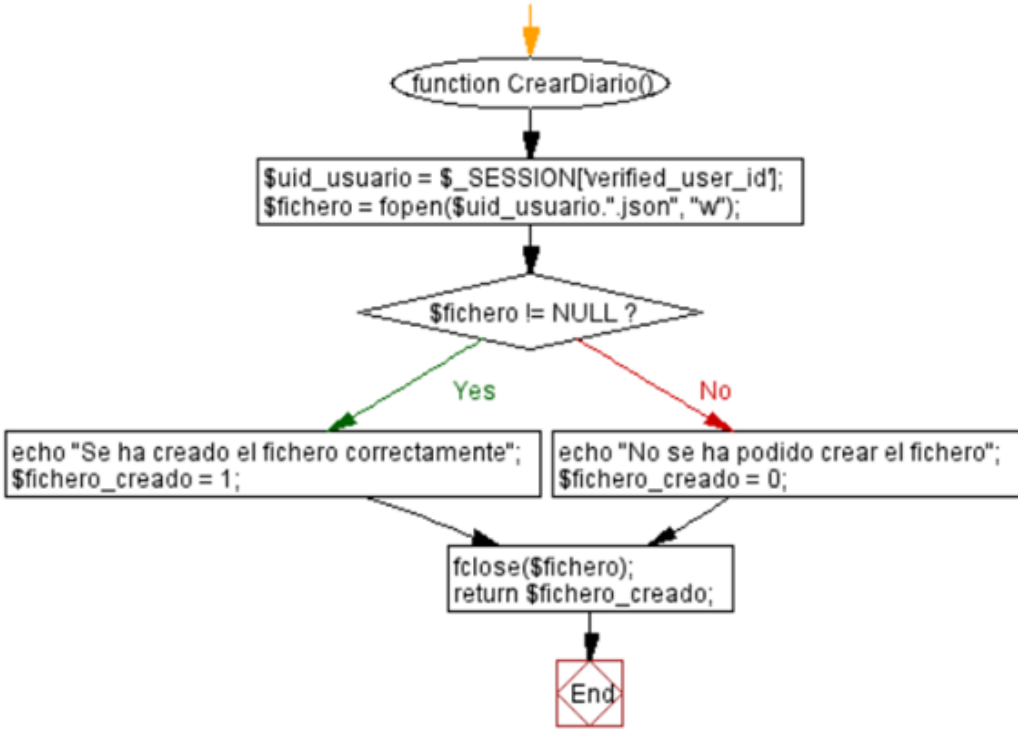
3 Rutas: 1-2-3-4-5-6-7-9-10-11-13
1-2-3-4-5-6-8-9-10-11-13
1-2-3-4-5-6-7-9-10-12-13 (también incluye 1-2-3-4-5-6-8-9-10-12-13)

C1: !empty($data) && is_connected==true

C2: empty($data) && is_connected==true

C3: !empty($data) && is_connected==false

-ActualizarPaginaLocal($fecha)



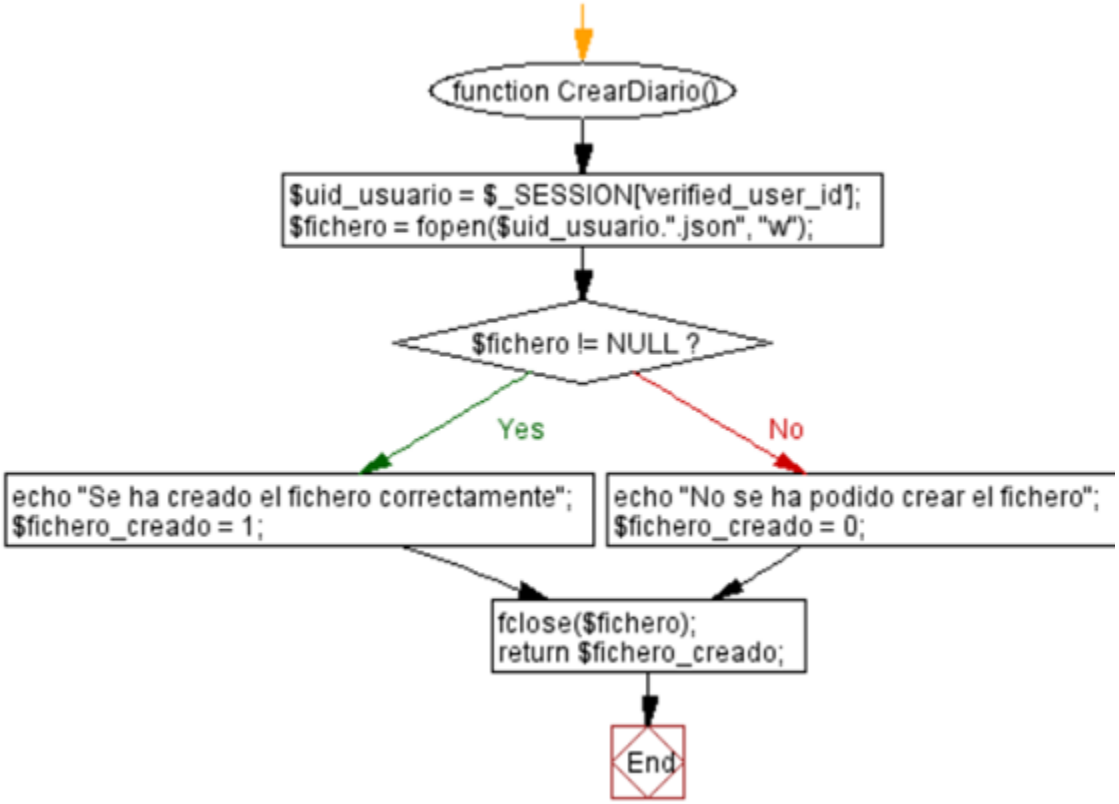
Nº Nodos Predicados: 1
V(G) = Nº Nodos Predicados + 1 = 1+1 = 2

2 Rutas: 1-2-3-4-5-6-7-8-10
1-2-3-4-5-6-7-9-10

C1: is_connected()==true

C2: is_connected()==false

-CrearDiario()



Nº Nodos Predicados: 1
V(G) = Nº Nodos Predicados + 1 = 1+1 = 2

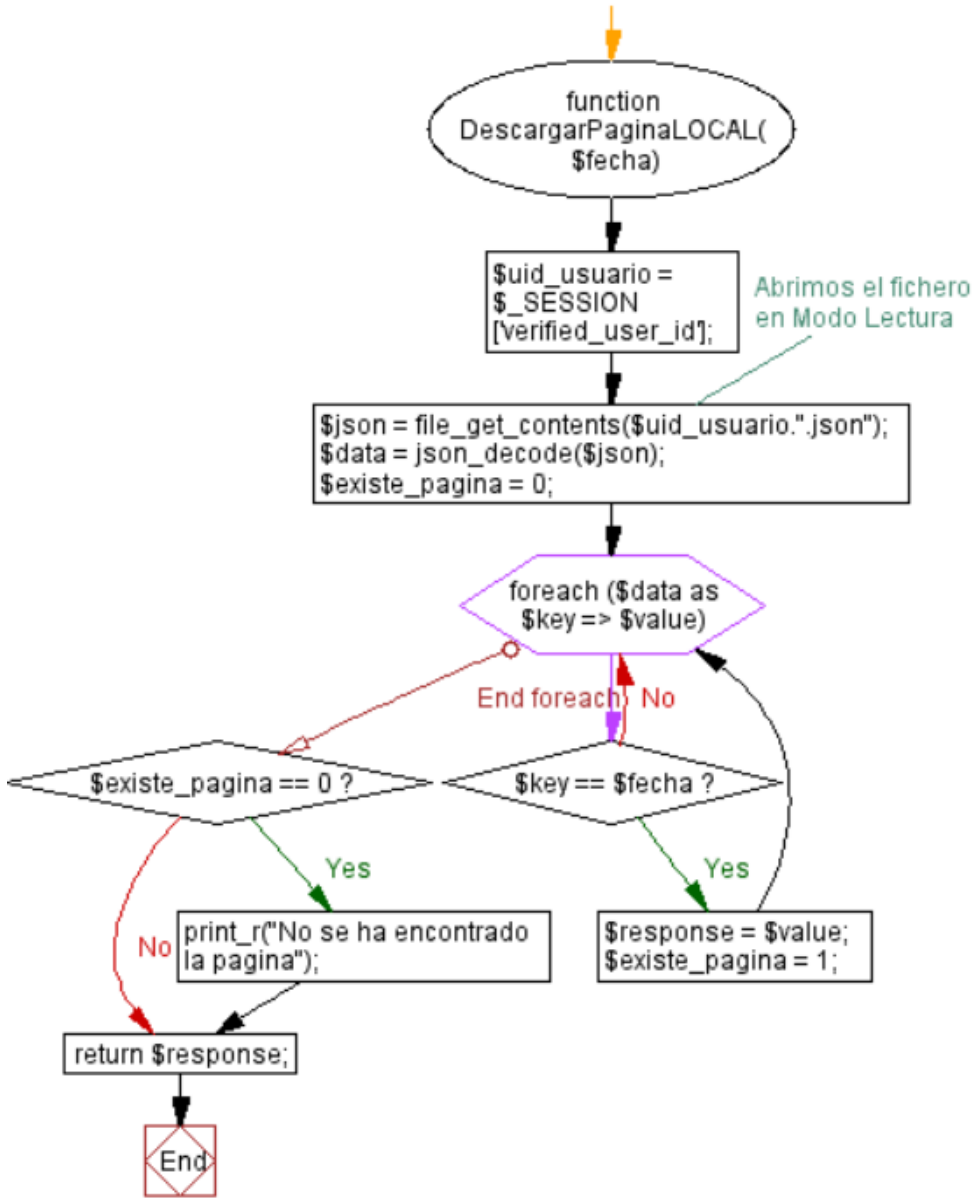
2 Rutas: 1-2-3-4-6-7

1-2-3-5-6-7

1º Ruta: $fichero != NULL

2º Ruta: $fichero == NULL

-DescargarPaginaLocal($fecha)



Nº Nodos Predicados: 3

V(G) = Nº Nodos Predicados + 1 = 3+1 = 4

4 Rutas: 1-2-3-4-5-6-4-7-8-9

1-2-3-4-5-6-4-7-9

1-2-3-4-5-4-7-8-9

1-2-3-4-5-4-7-9

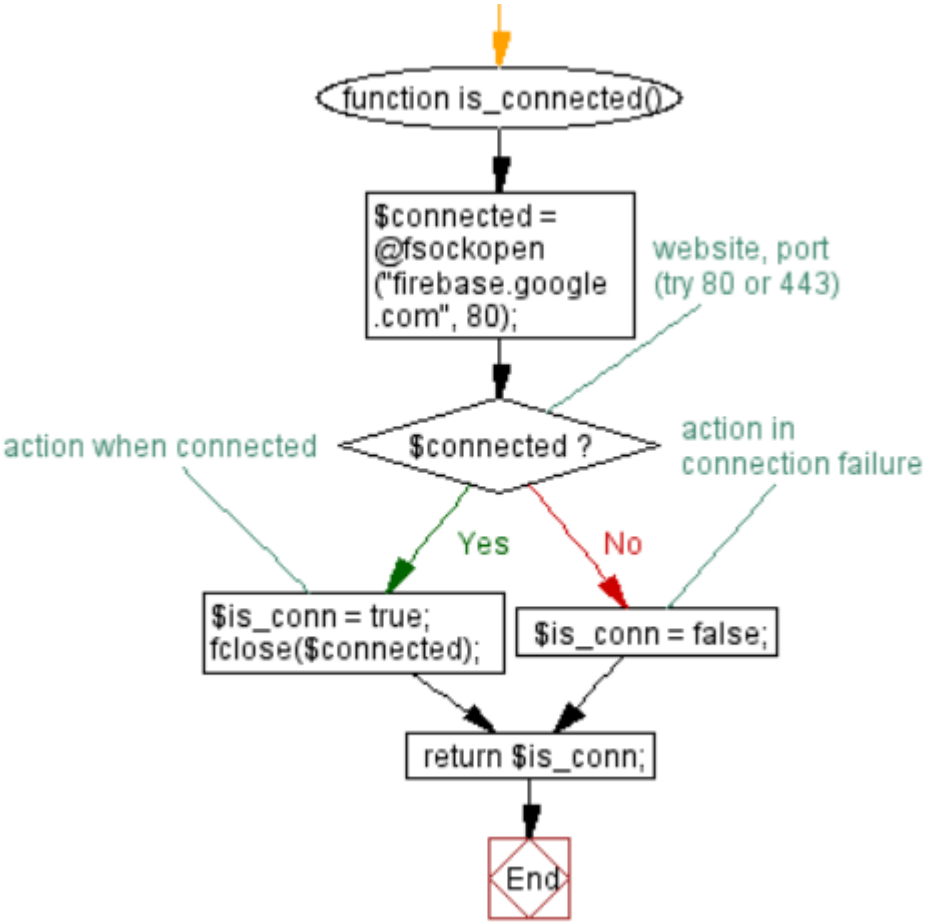
C1: $key==$fecha && $existe_pagina=1

C2: $key==$fecha && $existe_pagina=0

C3: $key!=$fecha && $existe_pagina=0

C4: $key!=$fecha && $existe_pagina=1

Is_connected()



Nº Nodos Predicados: 1

V(G) = Nº Nodos Predicados + 1 = 1+1 = 2

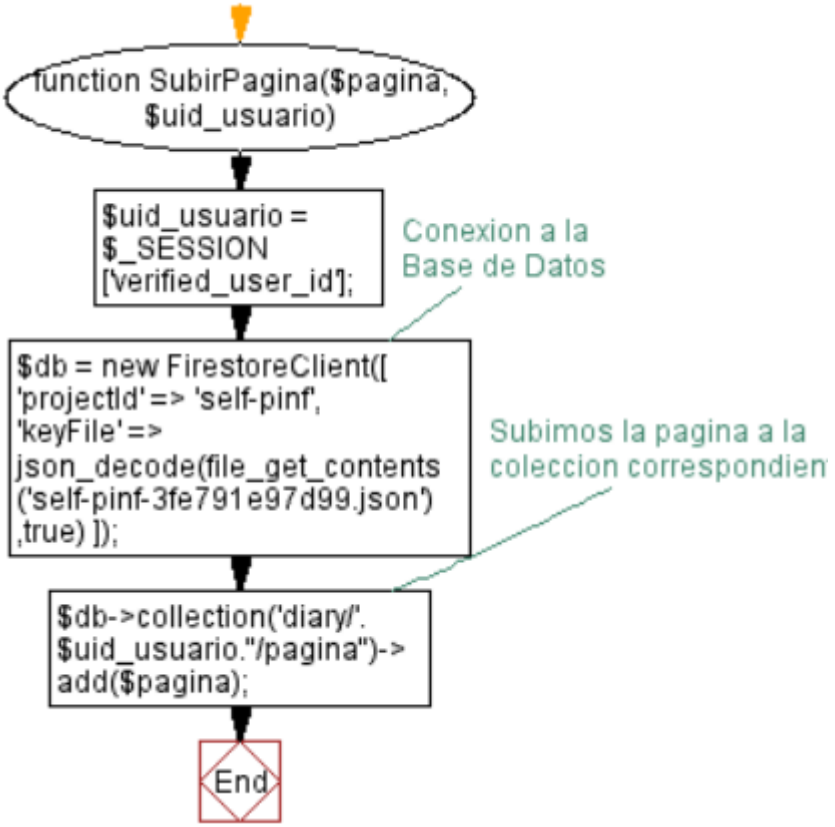
2 Rutas: 1-2-3-5-6

1-2-4-5-6

C1: $connected = True

C2: $connected = False

-SubirPagina($pagina, $uid_usuario)



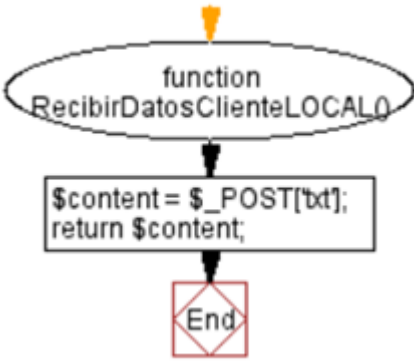
Nº Nodos Predicados: 0

V(G) = Nº Nodos Predicados + 1 = 0+1 = 1

1 Rutas: 1-2-3-4

C1: Conexión realizada correctamente

-RecibirDatosClienteLocal()



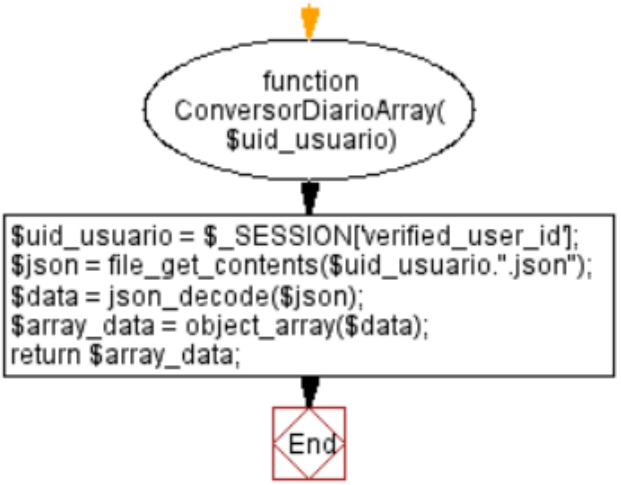
Nº Nodos Predicados: 0

V(G) = Nº Nodos Predicados + 1 = 0+1 = 1

1 Rutas: 1-2

C1: $content != vacio

-ConversorDiarioArray($uid_usuario)

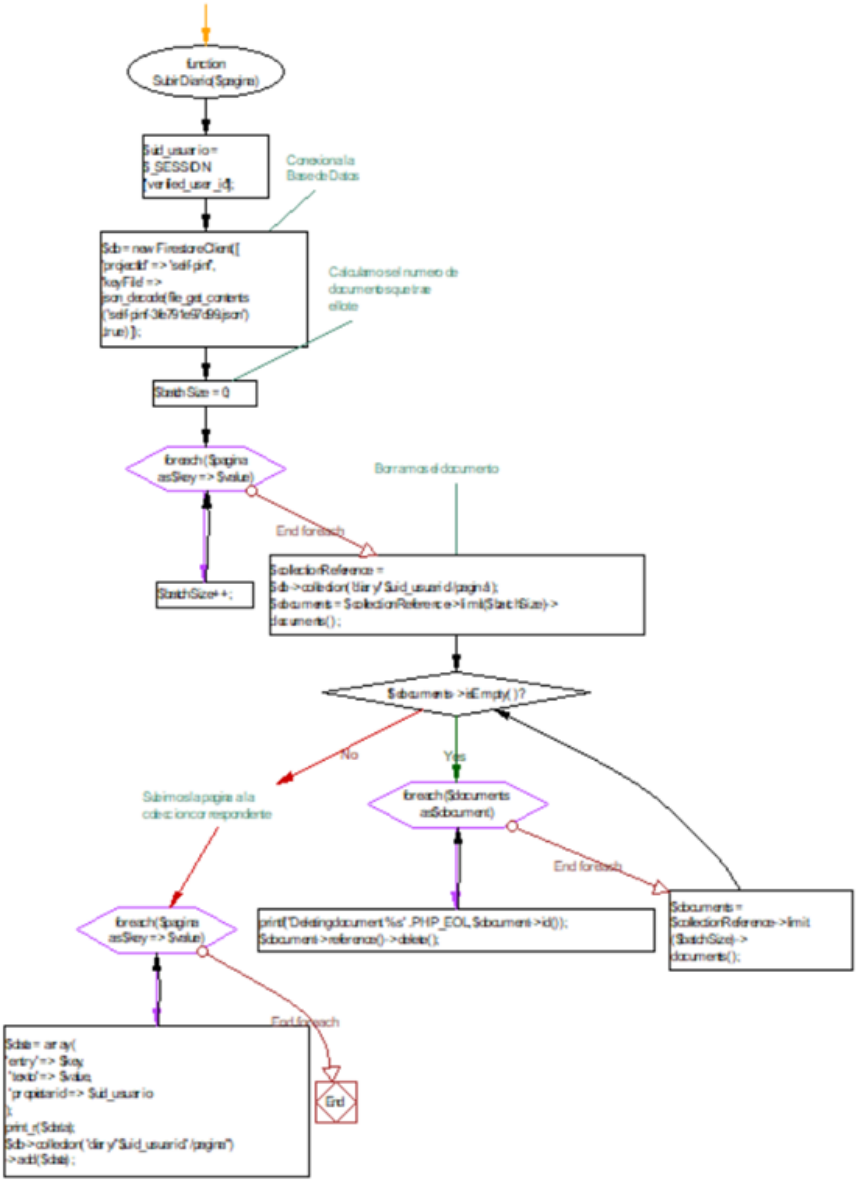


Nº Nodos Predicados: 0

V(G) = Nº Nodos Predicados + 1 = 0+1 = 1

1 Rutas: 1-2

C1: $uid_usuario != vacio && $json != vacio && $array_data != vacio



-SubirDiario($pagina)

Nº Nodos Predicados: 1

V(G) = Nº Nodos Predicados + 1 = 1+1 = 2

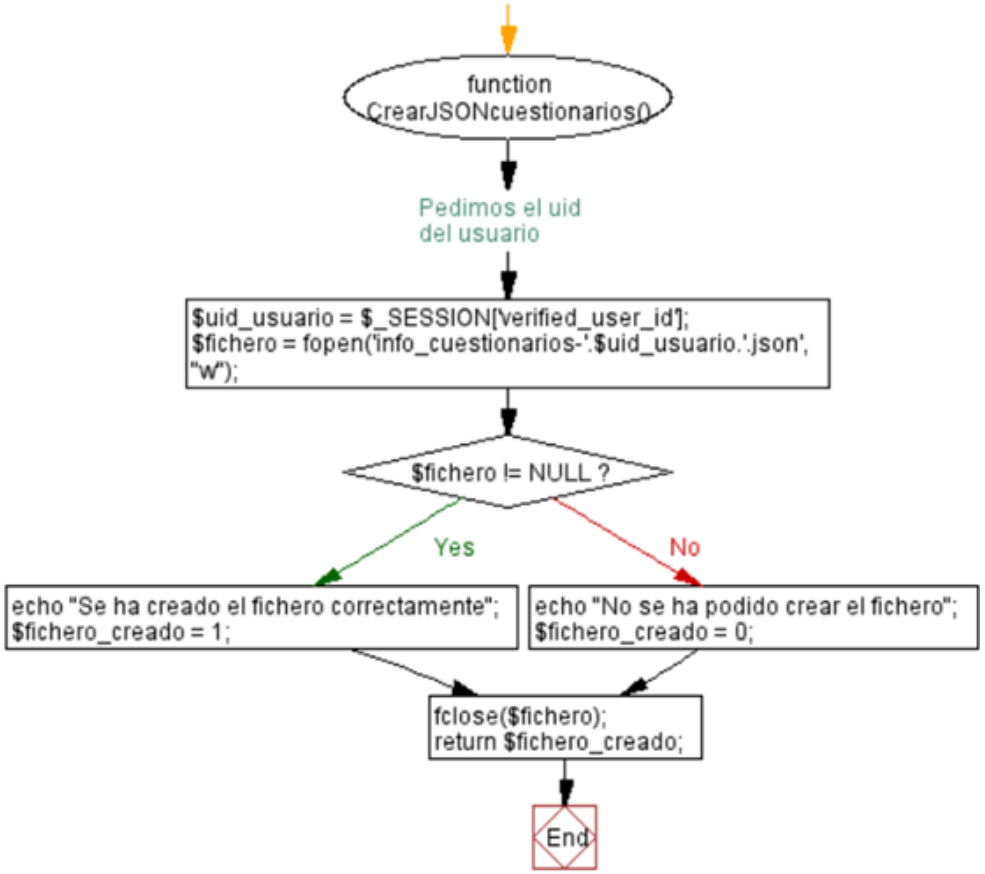
2 Rutas: 1-2-3-4-5-6-7-8-9-8-10-7-11-12-13

1-2-3-4-5-6-7-11-12-13

C1: !$documents->isEmpty = True

C2: !$documents->isEmpty = False

-CrearJsoncuestionarios()



Nº Nodos Predicados: 1

V(G) = Nº Nodos Predicados + 1 = 1+1 = 2

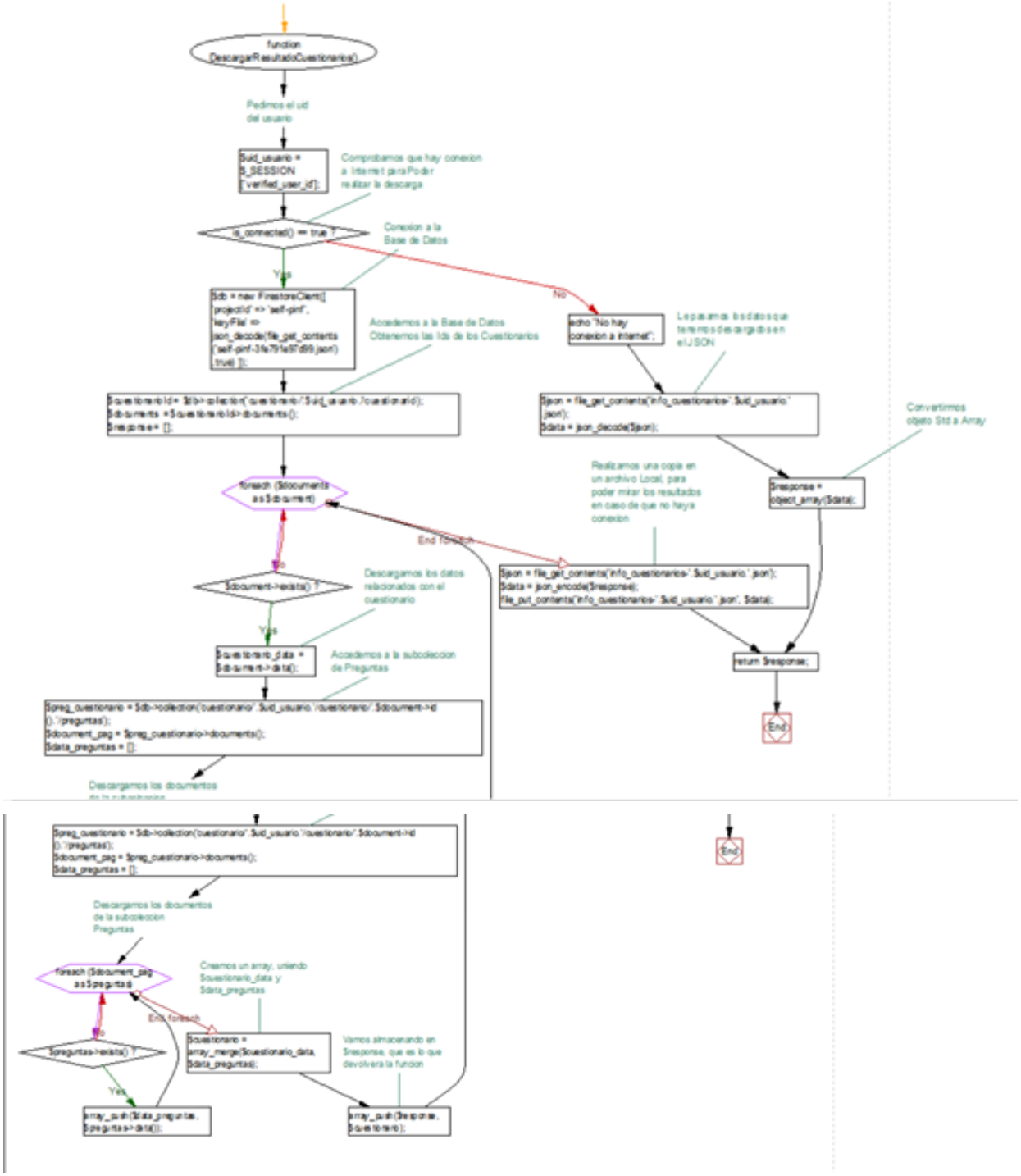
2 Rutas: 1-2-3-5-6

1-2-4-5-6

C1: $fichero != NULL

C2: $fichero == NULL

-DescargarResultadoCuestionarios()



Nº Nodos Predicados: 3

V(G) = Nº Nodos Predicados + 1 = 3+1 = 4

4 Rutas: 1-2-3-5-6-7-8-9-10-11-12-10-13-14-6-15-16-17

1-2-4-17-18-16-17

1-2-3-5-6-15-16-17

1-2-3-5-6-7-8-9-10-13-14-6-15-16-17

C1: is_connected() == True && $document->exists() == True && $preguntas->exists() == True

C2: is_connected() == False

C3: is_connected() == True && $document->exists() == False

C4: is_connected() == true && $document->exists() == True && $preguntas->exists() == False

6.2 Pruebas de adaptación

En esta sección someteremos a nuestro programa a diferentes pruebas de adaptación que verifican el funcionamiento del sistema en diferentes entornos. Por ejemplo se puede probar un sistema en los distintos navegadores o en distintas versiones de máquina virtual o en aplicación en todos aquellos entornos en los que tiene que funcionar de forma correcta.

- **Errores funcionales** :
    - Algunos botones que activen peticiones a la Base de Datos, como podrían ser: Acceder a un página del Diario, o Rellenar un Cuestionario podrían verse afectados en caso de la ausencia de conexión a internet.

- **Errores estéticos**:
    - Si el dispositivo tuviera activada alguna extensión que estuviera relacionada con la apariencia del dispositivo, como por ejemplo: *Dark Rider*, podría afectar a la experiencia del usuario.

7. Trabajo de investigación

7.1 Tratamiento de Información

- Ansiedad: Las preguntas fueron escogidas tomando de referencia tanto el libro de psicología DSM-V del cual seleccionamos las 7 primeras preguntas en base al apartado de la ansiedad generalizada y el resto de preguntas fueron tomadas en base al famoso test/informe STAI el cual permite evaluar tanto la ansiedad riesgo como la ansiedad estado, se define como la condición emocional transitoria caracterizada por sentimientos subjetivos de tensión y aprensión así como por una hiperactividad del sistema nervioso autónomo, la cual era la conveniente para nosotros ya que es lo que nosotros evaluamos con este test de ansiedad.
- Baja autoestima: En el caso de la batería de preguntas escogidas para la baja autoestima se tomó en cuenta tanto el famoso Test de Rosenberg o Escala de Autoestima de Rosenberg, la cual es una de las más utilizadas para la valoración de la autoestima en adolescentes. Esta fué desarrollada por Rosenberg en el año 1965 y traducida al castellano por Echeburua en el año 1995. Por otra parte las demás preguntas fueron creadas en base a distintas páginas la cuales hablaban sobre síntomas de la baja autoestima (las cuales serán referenciadas en la bibliografía) además de consultar sobre la veracidad de dichos síntomas con personas que están cursando la carrera de psicología o que ya la terminaron para asegurar que sea lo más orientativo posible.

- Depresión:
  Siguiendo la guia y consejo de varios psicólogos y terapeutas, nos encaminaron a un libro estadounidense DSM-5 “Guía de consulta de los criterios diagnósticos del dsm5”

  Es un libro que contiene como su nombre indica una guía que permite el diagnóstico y la clasificación de enfermedades mentales. Nosotros solo nos centramos en las enfermedades mentales que vimos acordes al criterio.
  Contiene varias secciones para cada enfermedad mental así como lista sus síntomas, nosotros hemos refinado y adaptado estos síntomas en forma de preguntas para nuestros test. Manteniendo así una credibilidad y estándar muy alta

- Medidas tóxicas: Las preguntas fueron escogidas en base a la investigación en diferentes páginas, las cuales serán todas referenciadas, las cuales trataban los problemas que puede haber en base al uso de redes sociales y ahí fuimos sacando los síntomas que más se adecuaban a lo que necesitábamos lo cual era la toxicidad del uso de las redes sociales y lo que esto puede provocar en las persona, ya sea depresión, ansiedad o baja autoestima.

7.2 Puntuación

- Ansiedad:
  Puntuación (sobre 15 preguntas que es el test específico)
  ansiedad alta: 45-22
  sobre promedio: 21-17
  promedio: 16
  bajo promedio: 15-11
  Bajo: 10-0
  Puntuación (sobre 5 preguntas, test general)
  alta: 15-8
  sobre promedio: 7-6
  bajo promedio: 5-4
  bajo:3-0

  En cuanto al sistema de evaluación del mismo test, tomamos de referencia la evaluación que el test STAI lleva a cabo ya que muchas de las preguntas recogidas en nuestra batería de preguntas son tomadas como referencia del mismo, por ello tanto para el test general como para el específico usamos la evaluación del STAI ajustada al número de preguntas que ambos poseen ya que en ambos es distinto a la cantidad de preguntas (items) que el STAI posee (20 preguntas).

- Baja Autoestima:
  Puntuaciones sobre 15 preguntas
  De 45 a 20 puntos: autoestima baja
  De 19 a 15 puntos: autoestima media
  De 14 a 0 puntos: autoestima alta
- Puntuaciones sobre 5 preguntas
  De 15 a 7 puntos: autoestima baja

De 6 a 4 puntos: autoestima media
De 3 a 0 puntos: autoestima alta

En cuanto a la evaluación de las preguntas se tomó en cuenta el sistema de evaluación que realiza el test de rosenberg ajustado tanto al número de preguntas de ambos test como a la puntuación de nuestros items en comparación a la puntuación del test de Rosenberg, la cual va de 1-4 y la nuestra es realizada de 0-3 puntos.

- Depresión:
  Puntuaciones sobre 15 preguntas
  De 45 a 31 puntos: depresión alta
  De 30 a 19 puntos: depresión media
  De 18 a 0 puntos: depresión baja
  Puntuaciones sobre 5 preguntas
  De 15 a 10 puntos: depresión alta
  De 9 a 5: depresión medio
  De 4 a 0:
  Tomando estos valores de la evaluación parametrizados mediante la guía de “Guía de consulta de los criterios diagnósticos del dsm5”. Uno de los rasgos más clave es el Trastorno de Depresión Mayor, que es más fácil de distinguir en sintomatología.

  Como estandarizamos todos los test a una puntuación de 3 puntos una suma de 0 a 18 pueden significar claramente unos niveles bajos de depresión, muchas personas pueden no encontrarse en una situación óptima pero no por ello padecen una enfermedad mental.

  De 19 a 30 he establecido una depresión media, no ha de significar aún un riesgo grave, pero aún así ya centraría un foco de atención especial. Quizás acercarse a un profesional o acudir a alguna charla sería un consejo idóneo

  Más de 31 Una puntuación alta en este test refleja muchos de los rasgos de depresión mayor. Acudir a un especialista es la opción ideal

- Medidas tóxicas:
  puntuaciones sobre 10 preguntas (test específico):

Alta: 30-17
Medio: 16-10
Bajo: 9-0

Puntuaciones sobre 5 preguntas (test general):

Alta: 15-8
Medio: 7-5
Bajo: 4-0

Este sistema de evaluación fue escogido en base a los demás ya realizados previamente para los demás diagnósticos, ya que sobre este tema no había referencias de test parecidos en ningún lado de los cuales poder sacar un sistema de puntuación de referencia.

7.2.1 Ayuda

- Ansiedad:

  Ansiedad alta: Si sufres ataques fuerte de ansiedad, sensaciones de sofocos. palpitaciones etc acuda a un especialista. Intente realizar ejercicios de respiración para minimizar el episodio. Nunca automedicarse sin la supervisión de especialistas.

  Ansiedad media: Es normal en periodos concretos tener una ansiedad mayor, cargas grandes de trabajos, situaciones familiares, épocas de estudios/oposiciones etc. Intenta aprender técnicas de relajación, pero, si son persistentes o muy fuertes acuda a un especialista.

  Ansiedad baja: No te preocupes, tu ansiedad está dentro de unos límites bastante bajos. Si necesitas un pelin de ayuda extra existen muchos ejercicios de respiraciones.

- Baja Autoestima:

  Autoestima baja: El principal consejo en caso de tener una baja autoestima en especial si tu puntuación es alta sería que acudiera a un especialista que le ayude en persona sobre todo este tema, ya que el sabrá con exactitud cómo hay que tratarle y ayudarle para superar la baja autoestima.

Autoestima media: En este caso hay ciertos consejos más orientativos que pueden llegar a ayudarte a aumentar tu autoestima, como por ejemplo el juntarte con aquellas personas que te traten bien, con las que tu te sientas agusto y sepas que no te juzgarán y no harán sentirte peor. Otro consejo es el que te aceptes como eres, ya que nadie es perfecto y todo el mundo tiene sus defectillos pero tienes que convivir con ello, ya que si tienes personas a tu alrededor que te tratan bien por como eres tú también deberías valorarte a ti mismo como ellos lo hacen ya que para ellos tu lo vales como persona. Y por último otro pequeño consejo es ayudar a los demás y ser generoso con los demás, ya sea ayudando en casa con las tareas, o incluso ayudando a un compañero de clase en alguna asignatura que se te de bien, eso hará que te sientas orgulloso de ti mismo y ayudará a subir tu autoestima.

Autoestima alta: no necesitas ningún tipo de ayuda ya que pareces quererte tal y como eres, por tanto como único consejo, sigue así.

- Depresión:

  Depresión alta. Acude a un especialista de la salud, ya sea psicólogo o psiquiatra. Si te ves con fuerzas intenta escribir en un papel en que cosas eres buen@. Y realizar alguna actividad que mejore tu estado de ánimo

  Depresión media. Puedes compartir tus inquietudes con alguien de tu circulo que no te juzgue y te apoye, recuerda que según el día y el contexto podemos tener baja o alta autoestima, si persiste en el tiempo acudir a un especialista

  Depresión baja. Te encuentras bastante bien, realiza alguna actividad que mejore tu humor

- Medidas Tóxicas:

  En caso de que el test de un nivel alto de toxicidad generado por las redes sociales, lo más recomendable para el usuario será que compruebe aquellas preguntas en las que su puntuación sea mayor e intente remediarlo de la siguiente manera.

- En caso de que su puntuación en las preguntas 1,8,9 y 10 sea una puntuación alta los más aconsejable sería que el usuario o desinstalara toda red social o en caso de esto generar más inconvenientes que convenientes, intentar hacer el menor uso de redes sociales posible ya sea por voluntad propia o mediante la función de los teléfonos de capar el uso de las redes sociales a ciertas horas y también que el usuario busque otros hobbies en esas horas que antes pasaba en redes sociales para ir olvidándose de las mismas o también estudiando en esos ratos libres en caso de que la puntuación de la pregunta 10 sea una puntuación elevada.

- En caso de que la puntuación elevada se encuentre en la pregunta 2, una de las opciones sería evitar el uso de las redes sociales cuando no se encuentre bien el usuario ya que puede generar un mayor malestar en el usuario y otra opción sería el pedir ayuda a terceros o psicólogos para solucionar los problemas del usuario antes de utilizar las redes con frecuencia para no empeorar la situación del mismo.

- En caso de que la puntuación sea elevada en las 3,4,5 y 6 el mejor consejo para darle al usuario sería que es mejor ser como eres y no reflejarse como alguien más o compararte con otras personas ya que tu eres tu mismo y es mejor reflejar cómo eres tú normalmente en las redes sociales para así encontrar personas que de verdad estén a tu lado por quien realmente eres y no por el personaje que has creado en las redes sociales para poder buscar reconocimiento o atención.

En caso de que el test de un nivel medio lo más recomendable para el usuario/paciente será principalmente que aproveche mejor el tiempo en vez de estar en las redes sociales, y que aproveche este tiempo para estar en la familia, para así crear aprovechar todo el tiempo posible con ellos para en un futuro no arrepentirte y también aprovechar ese tiempo para estudiar y llevar los estudios hacia adelante y de buena manera.

Si el paciente/usuario da un nivel bajo es una buena señal, ya que hace un buen uso de las redes sociales sin que esto pueda afectar en gran manera a su vida personal o a su bienestar.

8. Manual del usuario

Aquí comenzaremos con el apartado referido al Manual del Usuario, que es un apartado, el cual se basa en la explicación de cómo se utiliza Self de manera básica e intuitiva.

1. Después de la instalación del archivo con extensión .apk, procederemos a la apertura de la misma pulsando en su icono.
2. Posterior al paso 1 , procederemos al registro dentro de la misma aplicación mediante la acreditación de credenciales que se solicitan en la pantalla de registro
3. A partir de completar el registro, introduciremos nuestras credenciales anteriormente introducidas en la pantalla de registro para iniciar sesión dentro de Self.
4. Se abrirá la pantalla de inicio del perfil asociado, donde se destaca los apartados de diario, cuestionarios, resultados y foro, sin olvidar la configuración el icono con el engranaje y el apartado de ayuda sugerido por el signo de interrogacion “?”.
5. Si seleccionamos los diferentes podremos acceder a funciones diferentes dentro de Self:
   5.1. Diario: En este apartado como su mismo nombre especifica, sirve como un diario para que el usuario de manera local , vaya escribiendo lo que va pensando y su desarrollo dentro de Self.
   5.2. Cuestionarios: En este apartado se realizarán los cuestionarios de evaluación orientativos sobre las patologías que sufran los usuarios de esta app, si es que los tiene.
   5.3. Resultados: En este apartado se muestra el progreso y evolución del usuario dentro de Self a partir de los cuestionarios que ha ido realizando.
   5.4. Foro : Como su mismo nombre define es un medio de comunicación asíncrona donde los usuarios podrán comunicarse dentro de la aplicación aportando experiencias , opiniones e inquietudes con respecto las patologías que le afectan.
   5.5. Ayuda : Apartado que incluye los términos y condiciones de uso, políticas de seguridad y números telefónicos de referencia.
   5.6. Configuración : En este apartado se podrá modificar a nivel gráfico y de datos la aplicación mediante el cambio de opciones.

9. Manual de instalación y explotación

Aquí se hará una breve descripción de como instalar la aplicación para su posterior uso.
La instalación será mediante una apk la cual descargamos y ejecutamos para así poder instalar la aplicación de Self, ya lo que quedaría sería esperar a que esta se termine de instalar para así ya poder usar la aplicación.

10. Conclusiones

Como finalización del proyecto , podríamos decir que hemos aprendido a cómo desarrollar una aplicación híbrida de manera básica , incluyendo el trabajo en equipo entre los diferentes integrantes del grupo.

El descubrimiento de nuevas tecnologías , que en realidad muchos de los integrantes del equipo de trabajo desconocían y las cuales se utilizan en la vida real y laboral.

Informarnos sobre la baja autoestima , depresión , ansiedad y medidas tóxicas de una manera contrastada y útil para poder orientar a las soluciones de esos problemas.

11. Bibliografía

Bibliografía Documentación e información sobre test:
https://www.marketingandweb.es/marketing/peligros-de-las-redes-sociales/
https://www.mayoclinic.org/es-es/diseases-conditions/depression/symptoms-causes/syc-20356007
https://www.esmental.com/problemas-psicologicos-por-el-mal-uso-de-redes-sociales/

http://www.infocop.es/view_article.asp?id=6949
https://digitum.um.es/digitum/bitstream/10201/57088/1/Uso%20de%20las%20redes%20sociales.%20Cuestionario%20para%20adolescentes.pdf
https://www.liberties.eu/es/stories/redes-sociales-toxicas/43494
https://www.psicologia-online.com/caracteristicas-de-personas-con-autoestima-baja-2319.html
https://dialnet.unirioja.es/servlet/articulo?codigo=7032614
http://www.cop.es/colegiados/m-13106/images/Art%C3%ADculoAutoestima.pdf
https://www.researchgate.net/profile/Sebastian-Urquijo/publication/228545028_Rasgos_de_personalidad_y_depresion_en_adolescentes/links/00b7d52a1c66fe4ed8000000/Rasgos-de-personalidad-y-depresion-en-adolescentes.pdf
https://pdfcoffee.com/stai-3-pdf-free.html
https://kidshealth.org/es/teens/self-esteem.html

Bibliografía sobre backend PHP:

https://firebase.google.cn/docs/firestore/query-data/queries?hl=es_419#php_3
https://firebase.google.com/docs/firestore/quickstart?hl=es
https://cloud.google.com/firestore/docs/samples/firestore-data-set-id-random-collection
https://github.com/firebase/functions-samples
https://firebase.google.com/codelabs/firebase-emulator?continue=https%3A%2F%2Ffirebase.google.com%2Flearn%2Fpathways%2Ffirebase-emulators%23codelab-https%3A%2F%2Ffirebase.google.com%2Fcodelabs%2Ffirebase-emulator#0
https://firebase.google.com/docs/emulator-suite/install_and_configure?authuser=1
https://www.youtube.com/watch?v=1xK7q6aL_-4&list=PLr_acfJGVciqwwYjdo5iI7wuNtFgYhYX_&index=5&ab_channel=ArthurMann
https://www.youtube.com/watch?v=GEtUtqtOpXg&list=PLr_acfJGVciqwwYjdo5iI7wuNtFgYhYX_&index=5&ab_channel=ArthurMann
https://www.youtube.com/watch?v=3ACxp56r7ag&list=PLr_acfJGVciqwwYjdo5iI7wuNtFgYhYX_&index=1&ab_channel=ArthurMann
https://firebase.google.com/docs/firestore/quickstart?hl=es
https://firebase.google.cn/docs/firestore/manage-data/add-data?hl=es
https://firebase.google.cn/docs/firestore/query-data/queries?hl=es
https://firebase.google.com/docs/firestore/data-model?hl=es-419
https://www.youtube.com/watch?v=1xK7q6aL_-4

Bibliografía sobre Pruebas de Software

https://www.youtube.com/watch?v=GVegCwwfBZ0&ab_channel=JuanV.Carrillo%28jvprofe%29
https://www.youtube.com/watch?v=O6Cg4ing5bo&ab_channel=UniversitatPolit%C3%A8cnicadeVal%C3%A8ncia-UPV
https://programmerclick.com/article/20251004707/
https://www.infor.uva.es/~jvalvarez/docencia/tema7.pdf
https://oa.upm.es/40012/1/PFC_JOSE_MANUEL_SANCHEZ_PENO_3.pdf