Universidad de Valladolid

ESCUELA DE INGENIERÍA INFORMÁTICA

TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA INFORMÁTICA
MENCIÓN EN SOFTWARE

DESARROLLO MULTIPLATAFORMA EN DISPOSITIVOS MÓVILES PARA PREVENCIÓN DE RIESGOS LABORALES

Alumno/a: Javier Rubio Honrado

Tutor/es/as: Blas Torregrosa

A mi familia y a mi novia que siempre creyeron en mi

Agradecimientos

A mi familia, que siempre ha creído en mi y me ha apoyado cuando lo he necesitado.

A mi novia Marta que ha sido un pilar fundamental y quien me ha ayudado en mis peores momentos.

A todos los compañeros de carrera que me han hecho crecer personal y profesionalmente.

A mi tutor Blas por darme la oportunidad de realizar este TFG y hacerme conocer Flutter.

Resumen

El objetivo de este proyecto es desarrollar un aplicación móvil multiplataforma capaz de ayudar a los inspectores de riesgos laborales a la creación de inspecciones, evaluaciones y prevención de riesgos laborales. De forma que sea más fácil, intuitiva y amena la creación de informes. Evitando así mismo el tener que llevar papeles en mano a lo largo de la inspección. Además esta aplicación les permitirá tener en un solo lugar toda la información de todas las inspecciones que hayan realizado, ayudándoles a la búsqueda, consulta y modificación de las mismas.

El proyecto se ha desarrollado usando el framework Flutter y el lenguaje Dart usando la metodología ágil Scrum.

Abstract

The goal of this project is to develop a multi-plataform mobile application capable of helping inspectors of laboral risks in creation of inspections, evaluations and prevention of laboral risks. In an easier, intuitve and enjoyable way of creating informs. Avoiding in itself having to bring papers along the inspection. Furthermore, this aplication will allow them to have one and only place in which all the information of all of their inspections is alocated, helping them in searching, consulting and editing those.

The project has been developed using the framework Flutter and the language Dart using the agil methodology Scrum.

Índice general

Agradecimientos **III**

Resumen **V**

Abstract **VII**

Lista de figuras **XIII**

Lista de tablas **XV**

1. Introducción **1**

1.1. Introducción 1

1.2. Objetivos 2

1.2.1. Alcance del Proyecto 2

2. Prevención de riesgos laborales **3**

2.1. Definiciones 3

2.2. Principios y funcionamiento 3

2.3. Evaluación de los riesgos 5

2.3.1. ¿Qué y cómo se debe evaluar? 5

2.3.2. ¿Cuándo se debe efectuar la evaluación de los riesgos? 7

3. Planificación **9**

3.1. SCRUM 9

3.1.1. Principios de SCRUM 9

3.1.2. Equipo SCRUM 10

3.1.3. Eventos SCRUM 10

3.1.4. Atefactos SCRUM 12

3.1.5. Definición de “Terminado” (Definition of “Done”) 13

3.1.6. Cómo funciona 13

3.2. SCRUM en este proyecto 13

3.2.1. Sprints 14

3.2.2. Funcionalidades 15

4. Tecnologías **17**

4.1. Trello [14] 17

4.2. Git [15] 17

4.3. Visual Studio Code [16] 17

4.4. Flutter [17] 18

4.5. SQFLite [18] 18

5. Tecnologías móviles **19**

5.1. Posibles tecnologías de desarrollo 19

5.1.1. Aplicaciones Nativas 21

5.1.2. Aplicaciones Web 22

5.1.3. Aplicaciones Híbridas 23

5.2. Razonamiento detrás de la decisión tomada 24

6. Seguimiento **25**

6.0.1. Sprint 1 . . . 25

6.0.2. Sprint 2 . . . 26

6.0.3. Sprint 3 . . . 27

6.0.4. Sprint 4 . . . 27

6.0.5. Sprint 5 . . . 28

7. Diseño **29**

7.1. Diseño base de datos . . . 29

7.2. Inherited Widget . . . 31

7.3. Patrón Bloc(Buisness Logic Component) . . . 31

7.4. Patrón Singelton . . . 33

7.5. Estructura de la aplicación . . . 33

7.6. Diseño interfaz de usuario . . . 35

7.7. Implementación . . . 38

7.7.1. Estructura del código . . . 38

7.7.2. Decisiones y cambios . . . 39

7.8. Testing . . . 40

7.8.1. Test Unitarios . . . 41

7.8.2. Test Integración . . . 41

8. Lineas Futuras y Conclusiones **43**

8.1. Lineas Futuras . . . 43

8.1.1. COVID . . . 43

8.1.2. Firebase . . . 43

8.1.3. Plantillas 43

8.2. Conclusiones 44

A. Manual de usuario **45**

A.1. Creación de APK 45

A.1.1. Android 45

A.1.2. IOs 46

A.2. Funcionamiento de la aplicación 46

B. Enlaces adicionales **57**

Bibliografía **59**

Índice de figuras

2.1. Riesgos Laborales 5

3.1. Ciclo de vida SCRUM [10] 14

5.1. Top países por número de descargas en una App Store [4] 19

5.2. Tecnologías para desarrollar aplicaciones móviles [5] 20

7.1. Diagrama UML de la Base de Datos 30

7.2. Inherited Widget 31

7.3. Patrón Bloc 32

7.4. Streams 32

7.5. Estructura de datos 34

7.6. Demo 1 35

7.7. Demo 2 36

7.8. Demo 3 37

7.9. Estructura del código 38

7.10. Piramide de Test de Flutter 40

7.11. Test Unitarios 41

7.12. Test Integración 42

A.1. Captura de la creación del APK 46
A.2. Captura de la creación del APK para IOs 46
A.3. Login 47
A.4. Registro 47
A.5. Creación satisfactoria de un usuario 48
A.6. Página con listado de Inspecciones 48
A.7. Modal de creación de Inspección 49
A.8. Lista Inspecciones con una inspección 50
A.9. Lista Factores de Riesgo de primer nivel 50
A.10.Lista Factores de Riesgo de segundo nivel 51
A.11.Lista Factores de Riesgo a Evaluar 52
A.12.Formulario evaluar 52
A.13.Formulario evaluar 53
A.14.Formulario con una fotografía 54
A.15.Formulario con dos fotografías 54
A.16.Creación de Informe 55
A.17.Archivo Excel 56

Índice de cuadros

3.1. Planificación inicial Sprints 15

3.2. Historias de usuario 15

6.1. Tareas del sprint 1 26

6.2. Tareas del sprint 2 26

6.3. Tareas del sprint 3 27

6.4. Tareas del sprint 4 28

6.5. Tareas del sprint 5 28

Capítulo 1

Introducción

1.1. Introducción

Hoy en día solemos pensar que una oficina es un lugar seguro de trabajo, por lo menos lo suficiente como para no tener un plan de prevención de riesgos. Si comparamos cualquier otro entorno de trabajo como son las obras, la calle, o una fabrica, es normal que pensemos de esa forma. Aunque la realidad es otra; en una oficina los trabajadores pueden caerse, cortarse o sufrir otros tipos de accidente. Para esto existen los planes de prevención de riesgos laborales.

En la actualidad esto nos parece de lo más normal, pero no fue hasta 1971 que se creo El Plan Nacional de Higiene y Seguridad en el Trabajo [1]; la cual se puede considerar como el antecedende del INSHT(Instituto Nacional de Seguridad e Higiene en el Trabajo) 1978. El cual sería, con los años, un cambio de régimen de por medio y sus consiguientes cambios en la legislación, el precursor de la creación de la actual Ley de Prevención de Riesgos Laborales en 1995.

Una de las maneras de asegurarse del cumplimiento de esta ley son las inspecciones de prevención de riesgos laborales, más concretamente trataremos con los pertinentes al ámbito de las oficinas.

1.2. Objetivos

El objetivo de este proyecto es desarrollar una herramienta software para la inspección de prevención de riesgos laborales para dispositivos Android e IOS usando Flutter, un lenguaje multiplataforma móvil. La aplicación permitirá la realización de informes de prevención de riesgos laborales en oficinas de una forma más rápida e intuitiva, haciendo más fácil la formación y el seguimiento de los reglamentos estipulados. Esta aplicación se constituirá de los siguientes objetivos secundarios:

- Creación de una interfaz intuitiva y práctica para el uso de los profesionales en este ámbito, usando iconos descriptivos y formularios acorde a la incidencia.
- Creación de una base de datos local, para el correcto funcionamiento de la aplicación
- Además se espera adquirir ciertas competencias en el uso de este lenguaje

1.2.1. Alcance del Proyecto

En la actualidad, el inspector de riesgos laborales debe adquirir los datos necesarios de cada oficina para más tarde realizar el informe.

Queremos que el inspector de riesgos laborales sea capaz de realizar sus informes de forma rápida, cómoda y eficiente. Permitiéndole solo depender de su teléfono móvil para trabajar. añadir funcionalidades especificas aquí

Capítulo 2

Prevención de riesgos laborales

La prevención de riesgos laborales[2] se refiere al deber del empresario de proteger a a sus trabajadores frente al riesgo laboral, garantizando su salud y seguridad en todos los aspectos relacionados con su trabajo, mediante la integración de la actividad preventiva y adopción de medidas.

2.1. Definiciones

Para entender un poco mejor el proyecto y su ámbito añadimos unas definiciones[3]:

Prevención: Conjunto de actividades o medidas adoptadas o previstas en todas las fases de actividad de la empresa con el fin de evitar o disminuir los riesgos derivados del trabajo.

Riesgo laboral: La posibilidad de que un trabajador sufra un determinado daño derivado del trabajo. Para calificar un riesgo desde el punto de vista de su gravedad se valorarán conjuntamente la probabilidad de que se produzca el daño y la severidad del mismo.

2.2. Principios y funcionamiento

El empresario deberá aplicar las medidas de prevención según los siguientes principios generales[2]:

- Evitar los riesgos: Con planes de prevención.

- Evaluar los riesgos que no se puede evitar: probabilidad y severidad de la ocurrencia del mismo.

- Combatir los riesgos en su origen.

- Adoptar el trabajo a la persona: en particular atenuar el trabajo monótono y repetitivo y así reducir los efectos del mismo en la salud.

- Tener en cuenta la evolución de la técnica.

- Sustituir lo peligroso por lo que entrañe poco o ningún peligro.

- Planificar la prevención; buscando un conjunto coherente que integre en ella la técnica, la organización del trabajo, las condiciones de trabajo, las relaciones sociales y la influencia de los factores ambientales en el trabajo.

- Adoptar medidas que antepongan la protección colectiva a la individual.

- Dar las debidas instrucciones a los trabajadores.

La prevención de riesgos laborales deberá integrarse en el sistema general de gestión de la empresa, tanto en el conjunto de sus actividades como en todos los niveles jerárquicos de esta, a través de la implantación y aplicación de un plan de prevención de riesgos laborales[2].

Los instrumentos esenciales para la gestión y aplicación del plan de prevención de riesgos, que podrán ser llevados a cabo por fases de forma programada, son la evaluación de riesgos laborales y la planificación de la actividad preventiva[2].

Los riesgos laborales los estructuraremos en varios tipos; accidentes, enfermedades, fatiga e insatisfacción, tal y como aparece en la Figura 2.1.

Figura 2.1: Riesgos Laborales

2.3. Evaluación de los riesgos

Aquellos riesgos que no se han podido evitar se deben estimar su magnitud, esto se realiza en la evaluación de riesgos, obteniendo información para que el empresario pueda tomar una decisión apropiada sobre la necesidad de adoptar medidas preventivas y sobre el tipo de las mismas[2].

2.3.1. ¿Qué y cómo se debe evaluar?

Se deben evaluar los riesgos en cada puesto de trabajo. Para ello se tendrá en cuenta por un lado las condiciones de trabajo existentes o previstas y, por otro, el trabajador que ocupa el puesto[2].

La evaluación de riesgos tomará en consideración los siguientes aspectos[2]:

- Las características de los locales.
- Las instalaciones.

- Los equipos de trabajo existentes.
- Los agentes químicos, físicos y biológicos o empleados en el trabajo.
- La propia organización y ordenación del trabajo en la medida en que influyan en al magnitud de los riesgos.
- Deberá tenerse en cuenta la posibilidad de que el trabajador que ocupe ese puesto de trabajo sea especialmente sensible, por sus características personales o estado biológico conocido, incluidos aquellos que tengan reconocida la situación de discapacidad física, psíquica o sensorial, a alguna de dichas condiciones.

La evaluación deberá servir para identificar los elementos peligrosos, los trabajadores expuestos y la magnitud de los riesgos.

En caso de que existiera una normativa especifica de aplicación, el procedimiento de evaluación deberá ajustarse a las condiciones concretas establecidas en la misma.
Cuando la normativa no indique o concrete los métodos que deben emplearse o cuando los criterios de evaluación dependan de otros criterios de carácter técnico, se podrán utilizar, si existen, los métodos o criterios recogidos en[2]:

- Normas UNE
- Guías del Instituto Nacional de Seguridad y Salud en el Trabajo del Instituto Nacional de Silicosis y protocolos y guías del Ministerio de Sanidad, Consumo y Bienestar Social, así como de Instituciones competentes de las comunidades autónomas.
- Normas internacionales.
- En ausencia de los anteriores, guías de otras entidades de reconocido prestigio en la materia y otros métodos o criterios profesionales descritos documentalmente que proporcionen un nivel de confianza equivalente.

Al final del proceso, deberá documentarse la evaluación de los riesgos, incluido cuando el resultado de la evaluación los hiciera necesarios, el resultado de los controles periódicos de las condiciones de trabajo y de la actividad de los trabajadores[2].

2.3.2. ¿Cuándo se debe efectuar la evaluación de los riesgos?

Deberá realizarse o revisarse:

- Al inicio de la actividad.
- Cuando se empleen nuevos equipos de trabajo, tecnologías, preparados o sustancias o se modifique el acondicionamiento de los lugares de trabajo.
- Cuando se cambien las condiciones de trabajo.
- Cuando se incorpore un trabajador especialmente sensible.
- Cuando se hayan detectado daños a la salud de los trabajadores.
- Cuando se haya apreciado a través de los controles periódicos, incluidos los relativos a la vigilancia de la salud, que las actividades de prevención pueden ser inadecuadas o insuficientes.
- Cuando así lo establezca una disposición específica, convenio colectivo o acuerdo entre empresario y representantes de los trabajadores, teniendo en cuenta, en particular, el deterioro por el transcurso del tiempo de los elementos que integran el proceso productivo.

Estas evaluaciones han de realizarse por personal técnico competente que forme parte de la organización de recursos para las actividades preventivas y que cuente con la capacidad y aptitud necesaria para desarrollar las funciones, según correspondan, de nivel básico, intermedio o superior definidas en el RSP.

Capítulo 3

Planificación

Hemos hecho uso de la plataforma Trello para realizar una planificación inicial y poder seguir la metodología SCRUM de una forma más sencilla e intuitiva.

3.1. SCRUM

Scrum[9] es un marco de trabajo de procesos que ha sido usado para gestionar el desarrollo de productos complejos desde principios de los años 90. Dentro del cual se pueden emplear varios procesos y técnicas.

El marco de trabajo Scrum consiste en los Equipos Scrum y sus roles, eventos, artefactos y reglas asociadas. Cada componente dentro del marco de trabajo sirve a un propósito específico y es esencial para el éxito de Scrum y para su uso.

3.1.1. Principios de SCRUM

Tres pilares soportan toda la implementación del control de procesos empírico[9]: transparencia, inspección y adaptación.

- **Transparencia:**
  Los aspectos significativos del proceso deben ser visible y estar definidos por un estándar común para aquellos que son responsables del resultado.

- **Inspección:**
  Los usuarios de Scrum deben inspeccionar los artefactos de Scrum y el progreso hacia el objetivo para detectar variaciones indeseadas. Las inspecciones deben ser hechas de forma diligente y no tan frecuentemente como para que interfiera en el trabajo.

- **Adaptación:**
  Si en una inspección se detecta uno o más aspectos que se desvían de los límites aceptables, posiblemente afectando al resultado haciendolo inaceptable, el proceso debe ajustarse. Este ajuste debe realizarse cuanto antes para evitar mayores desviaciones.

3.1.2. Equipo SCRUM

El Equipo Scrum consiste en un Dueño de Producto (Product Owner), el Equipo de Desarrollo (Development Team) y un Scrum Master[9].

- **Dueño de Producto (Product Owner):**
  Es el responsable de maximizar el valor del producto y el trabajo del Equipo de Desarrollo. Es la única persona responsable de gestionar la Lista del Producto(Product Backlog).

- **Equipo de Desarrollo (Development Team):**
  Son los profesionales encargados de realizar el trabajo de entregar un incremento "Terminado"que potencialmente se pueda poner en producción al final de cada Sprint. Solo el Equipo de Desarrollo participa en la creación del incremento.

- **Scrum Master:**
  Es el responsable de asegurar que Scrum se entienda y se adopte. Se hace esto asegurándose de que el Equipo Scrum trabaja ajustándose a la teoría, prácticas y reglas de Scrum.

3.1.3. Eventos SCRUM

En Scrum[9] existen eventos predefinidos con el fin de crear regularidad y minimizar la necesidad de reuniones no definidas en Scrum. Todos los eventos tienen un tiempo prefijado.

El Sprint, es un contenedor del resto de eventos. Contienen y consisten en la Planificación del Sprint (Sprint Planning),el Obejtivo del Sprint(Sprint Goal), los Scrums Diarios (Daily Scrums), el trabajo de desarrollo, la Revisión del Sprint (Sprint Review), y la Retrospectiva del Sprint (Sprint Retrospective).

- **Sprint**
  Es el corazón de Scrum, es un bloque de tiempo(time-box) de un mes o menos en el cual se crea un incremento de producto "Terminadoütilizable y potencialmente desplegable. Cada nuevo sprint empieza cuando finaliza el anterior.

  Los Sprints contienen y consisten en la Planificación del Sprint(Sprint Planning), los Scrums Diarios(Daily Scrums), el trabajo de desarrollo, la Revisión del Sprint(Sprint Review), y la Retrospectiva del Sprint(Sprint Retrospective).

- **Planificación del Sprint (Sprint Planning)**
  Durante la Planificación del Sprint se planifica el trabajo a realizar durante el Sprint. Este plan se crea mediante el trabajo colaborativo del Equipo Scrum completo.

- **Obejtivo del Sprint(Sprint Goal)**
  Es la meta establecida para el Sprint que puede lograrse mediante la implementación de la Lista de Producto. Proporciona una guía al Equipo de Desarrollo acerca de por qué está construyendo el incremento. Se crea en la Planificación del Sprint.

- **Scrums Diarios (Daily Scrums)**
  Es una reunión con una duración de 15 minutos para que el Equipo de Desarrollo sincronice sus actividades y cree un plan para las siguientes 24 horas. Se inspecciona el trabajo realizado desde el último Scrum Diario y se proyecta el trabajo que se podría hacer antes del siguiente. Se realiza en el mismo lugar y hora todos los días.

- **Revisión del Sprint (Sprint Review)**
  Se lleva a cabo al final del Sprint para inspeccionar el incremento y adaptar la Lista de producto si fuese necesario. Durante esta revisión, el Equipo Scrum y los interesados colaboran acerca de lo que se hizo durante el Sprint. Basándonos en esto y en los cambios a la Lista de Producto durante el Sprint, los asistentes determinan las siguientes cosas que podrían hacerse para optimizar el valor. Es una reunión informal con carácter informativo y colaborativo.

  El resultado es una Lista de Producto revisada que define los elementos posibles para el siguiente Sprint.

- **Retrospectiva del Sprint (Sprint Retrospective)**
  Es una oportunidad para el Equipo Scrum de inspeccionarse a sí mismo y de crear un plan de mejoras que sean abordadas durante el siguiente Sprint. El propósito de la Retrospectiva de Sprint es:

  - Inspeccionar cómo fue el último Sprint en cuanto a personas, relaciones, procesos y herramientas.
  - Identificar y ordenar los elementos más importantes que salieron bien y las posibles mejoras.

- Crear un plan para implementar mejoras a la forma en la que el Equipo Scrum desempeña su trabajo.

3.1.4. Atefactos SCRUM

Los artefactos de Scrum[9] representan trabajo o valor en diversas formas que son útiles para proporcionar transparencia y oportunidades para la inspección y adaptación. Lo cual es necesario para asegurar que todos tengan el mismo entendimiento del artefacto.

- **Lista de Producto (Product Backlog)**
  Lista ordenada de todo lo que podría ser necesario en el producto, es la única fuente de requisitos para cualquier cambia a realizarse en el producto. El Dueño de Producto(Product Owner) es el responsable de la Lista de Producto, incluyendo su contenido, disponibilidad y ordenación.

  Nunca está completa. La lista evoluciona a medida que el producto y el entorno en el que utilizará lo hacen.

  La lista de Producto enumera todas las características, funcionalidades, requisitos, mejoras y correcciones que constituyen cambios a realizarse sobre el producto para entregas futuras.

  Los elementos de esta tienen como atributos: descripción, orden, estimación y valor.

  Lo elementos de la Lista de Producto que pueden ser "Terminados"por el Equipo de Desarrollo en un Sprint son considerados "Preparados.º .ªccionables"para ser seleccionados en una reunión de Planificación de Sprint.

- **Lista de Pendientes del Sprint (Sprint Backlog)**
  Es el conjunto de elementos de la Lista de Producto seleccionados para el Sprint, más un plan para entregar el incremento de producto y conseguir el Objetivo del Sprint. Es una predicción hecha por el Equipo de Desarrollo acerca de qué funcionalidad formará parte del próximo incremento y del trabajo necesario para entregar esa funcionalidad en un incremento "Terminado".

- **Seguimiento del Progreso del Sprint**
  En cualquier momento durante un Sprint es posible sumar el trabajo restante total en los elementos de la Lista de Pendientes del Sprint. El Equipo de Desarrollo hace seguimiento de este trabajo restante total al menos en cada Scrum Diario para proyectar la posibilidad de conseguir el Objetivo del Sprint.

- **Incremento**
  Es la suma de todos los elementos de la Lista de Producto completados durante un

Sprint y el valor de los incrementos de todos los Sprints anteriores. Al final de un Sprint el nuevo incremento debe estar "Terminado", lo cual significa que está en condiciones de ser utilizado y que cumple la Definición de "Terminado"del Equipo Scrum.

3.1.5. Definición de "Terminado" (Definition of "Done")

Cuando un elemento de la Lista de Producto o un incremento se marca como "Terminado", todo el mundo debe entender lo que significa. Esto varía para cada Equipo de Scrum, pero los miembros del equipo deben tener el mismo entendimiento de "Terminado". Se utiliza para evaluar cuando se ha completado el trabajo sobre el incremento de producto.

3.1.6. Cómo funciona

Como hemos explicado anteriormente todo empieza con el **Dueño de Producto** siendo el responsable de la **Lista de Producto**, en la cual están los requisitos del producto a desarrollar. Cada cierto tiempo(fijado) se realiza una **Planificación de Sprint** en la cual el **Equipo de Desarrollo** decide que funcionalidades de la **Lista de Producto** podrán realizarse en este **Sprint**. Posteriormente durante la duración del Sprint el **Equipo de Desarrollo** junto con el **Scrum Master** realizan las llamadas **Scrum Diarias** para ponerse al día e informar de lo realizado el día anterior y lo que se espera realizar en las siguientes 24 horas. Durante el Sprint puede cambiar algunos aspectos de la **Lista de Pendientes** para adecuarse a las necesidades del proyecto y del Equipo de Desarrollo. Al finalizar el Sprint se realiza una **Revisión del Sprint**, en la que se inspecciona el incremento y se definen los posibles elementos para el siguiente Sprint de modo que maximice el valor del mismo. Después se realizará la **Retrospectiva del Sprint** en el cual se crea un plan de mejoras para el siguiente Sprint y así mejorar el desempeño. Y finalmente volvemos al punto de partida en donde se hace una Planificación del siguiente Sprint.

3.2. SCRUM en este proyecto

Debido a la falta de personal para este proyecto el alumno se hará cargo de los roles de Dueño de Producto y Equipo de Desarrollo, siendo el encargado de la Lista de Producto y Lista de Pendientes. El tutor hará el rol de Scrum Master de modo que se encargará de la correcta implementación de Scrum.

No se realizarán Scrum Diarias por disponibilidad de horarios. Se realizará una Planificación de Sprint cada dos semanas, que será la duración de los Sprints. Las Revisiones de

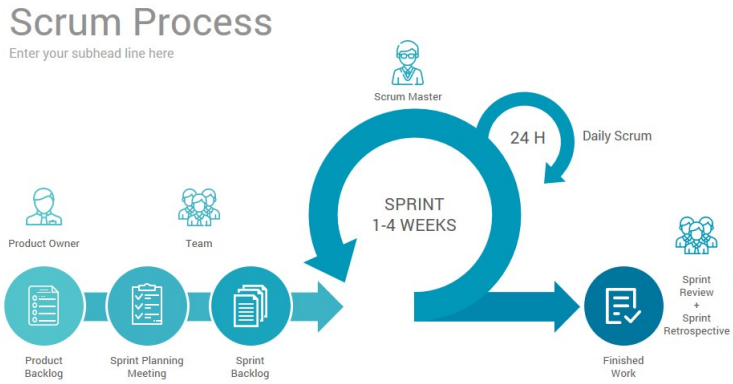


Figura 3.1: Ciclo de vida SCRUM [10]

Sprint y Retrospectiva de Sprint se realizarán si se considera necesario justo antes de la Planificación de Sprint.

El proyecto se realizará en distintas fases bien distinguidas, de modo que la primera fase sea de investigación y documentación, la segunda de investigación y aprendizaje de tecnologías y lenguajes a usar, y la última fase la de implementación y testing.

3.2.1. Sprints

Los Sprints serán de dos semanas como ya se ha comentado anteriormente, y se distribuirán de la siguiente manera:

Las tareas de los sprints los vamos a diferenciar por tipos:

- Chore[11]: una tarea que tiene valor para el equipo, pero no para el Stackeholder, un objeto cuyo valor no se ve reflejado como valor de negocio. Como por ejemplo documentación.
- HU[12]: unidad de trabajo suficientemente pequeña como para ser aceptada en un Sprint, relacionada con las historias descritas en la tabla 3.2.
- Fix: tarea relacionada con algún bug o problema que haya que solucionar de algo ya realizado.
- Refactor: tarea en la que se encarga de realizar una refactorización de alguna parte del proyecto, debido a cambios en los requisitos o por recomendación de algún Stackeholder.

Sprint	Fecha Inicio	Fecha Fin
Sprint 1	12/05/2020	26/05/2020
Sprint 2	26/05/2020	09/06/2020
Sprint 3	09/06/2020	23/06/2020
Sprint 4	23/06/2020	07/07/2020
Sprint 5	07/07/2020	14/07/2020
Sprint 6		
Sprint 7		
Sprint 8		
Sprint 9		
Sprint 10		

Tabla 3.1: Planificación inicial Sprints

3.2.2. Funcionalidades

En este apartado se muestran las tareas a implementar, se trata de 'Epics'[13] ; contenedores de historias o tareas que más tarde se añadirán a los Sprints.

Tarea	Titulo	Descripción
1	Iniciar inspección	Como usuario quiero poder iniciar una inspección
2	Finalizar inspección	Como usuario quiero poder finalizar una inspección
3	Eliminar inspección	Como usuario quiero poder eliminar una inspección
4	Crear riesgo	Como usuario quiero poder crear una deficiencia
5	Evaluar riesgo	Como usuario quiero poder hacer una evaluación de las deficiencias creadas
6	Eliminar riesgo	Como usuario quiero poder eliminar una deficiencia creada
7	Generar informe	Como usuario quiero poder generar un informe de la inspección

Tabla 3.2: Historias de usuario

Capítulo 4

Tecnologías

Las tecnologías que van a ser utilizadas en este proyecto son las siguientes:

4.1. Trello [14]

Es una herramienta para gestión de proyectos online, de forma que puedes crear tus propios Backlogs y poder hacer un seguimiento de Scrum más organizado y limpio.

4.2. Git [15]

Es una herramienta de gestión de versiones, se usará en este proyecto para guardar la implementación y para el control de versiones.

4.3. Visual Studio Code [16]

Editor de código muy potente y no muy pesado. Es el editor de código con el que estoy acostumbrado a programar, de modo que será el elegido para este proyecto.

4.4. Flutter [17]

Kit de herramientas hecho por Google para la realización de aplicaciones compiladas nativamente. Será el framework que usaremos para la realización del proyecto.

4.5. SQFLite [18]

Complemento para el uso de SQLite en aplicaciones de Flutter. Lo usaremos para la creación de la base de datos local en el dispositivo móvil.

Capítulo 5

Tecnologías móviles

5.1. Posibles tecnologías de desarrollo

El sector de aplicaciones móviles ha crecido exponencialmente en los últimos años, así como ha crecido su demanda. En los últimos 3 años(2016-2019) las descargas de aplicaciones móviles han crecido un 45 % [4] y un 6 % en el último año(2019)

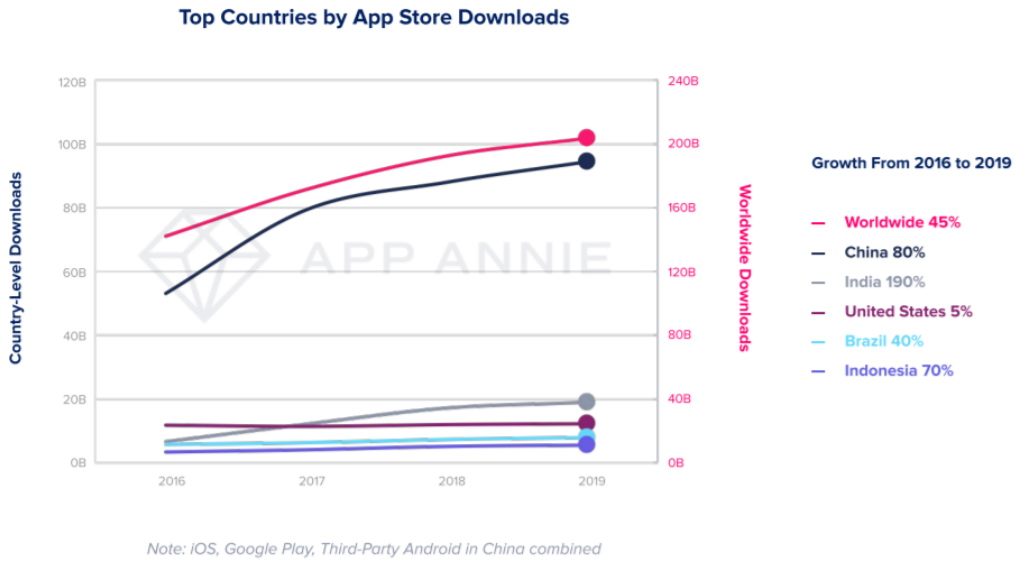


Figura 5.1: Top países por número de descargas en una App Store [4]

Hay muchas tecnologías de aplicaciones móviles que se usan para una plataforma especifica o para varias plataformas a la vez, cuatro de las más usadas son las siguientes[5]:

- **Swift**
  Usada para el desarrollo de productos específicos para Apple. Tiene características avanzadas con una codificación mínima que se puede mantener fácilmente.

- **C++**
  Forma la base simplista para la mayoría de los lenguajes de programación, se usa para la creación de aplicaciones dinámicas. El compilador al ser simple y efectivo hace que sea una herramienta versátil que se puede usar para varias plataformas. Anteriormente su hermana, Objective-C, se usaba para el desarrollo de aplicaciones Apple.

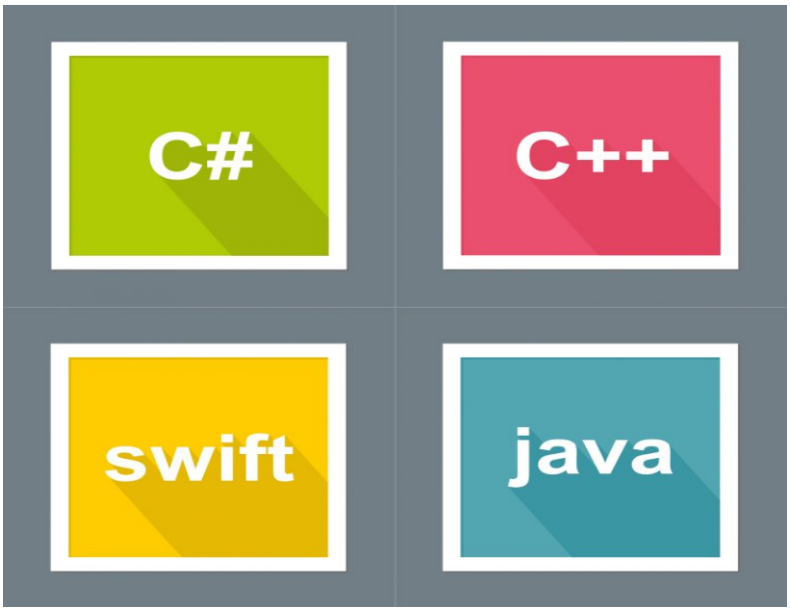


Figura 5.2: Tecnologías para desarrollar aplicaciones móviles [5]

- **Java**
  Este lenguaje orientado a objetos es el lenguaje de desarrollo oficial de Android. Es un lenguaje fácil de manejar con muchas librerías de código abierto disponibles para los usuarios.

- **HTML5**
  Es la tecnología usada para el desarrollo de aplicaciones web-frontend para dispositivos móviles.

Además hay dos frameworks que son muy prometedores de cara al futuro del desarrollo de aplicaciones móviles:

- **Flutter**
  Es un framework de código abierto entre plataformas SDk de Google, contiene una amplia variedad de widgets de Google y permite desarrollar aplicaciones móviles para Android y Apple iOS. Está en auge en cuanto al desarrollo de aplicaciones móviles entre plataformas. Usa DART como lenguaje de programación en vez de Javascript, lo que facilita un análisis, diseño de interfaces y correcciones de errores de forma rápida y efectiva.

- **React Native**
  Es un framework de código abiertoque usa Javascript. Se ha vuelto el preferido para muchos desarrolladores de aplicaciones móviles. Ofrece un amplio soporte de IDEs y otras herramientas para el desarrollo y permite la creación de aplicaciones nativas para iOs y Android.

Las aplicaciones móviles pueden ser desarrolladas en varias plataformas y con varios lenguajes diferentes, para ello tenemos ciertas herramientas o frameworks que nos ayudan en este proceso. Las aplicaciones pueden ser de tres tipos: Nativas, Web o Híbridas. Ahora entraremos en más detalle sobre estos tipos de aplicaciones, sus usos, ventajas y desventajas.

5.1.1. Aplicaciones Nativas

Una aplicación Nativa se refiere a una aplicación que ha sido escrita en el lenguaje de desarrollo nativo y usando las herramientas especificas de esa plataforma. Por ejemplo: una aplicación nativa iOS estaría escrita en Swift o en Objective-C y compilado usando Xcode, mientras que una aplicación nativa de Android estaría desarrollada usando Kotlin o Java y compilada usando Android Studio [6].

- Ventajas[7]

  - Presenta una mayor rendimiento dentro de los sistemas operativos, ya que tiene acceso a todos los recursos de teléfono móvil.
  - Permite notificaciones push.
  - Resulta más sencillo seguir una linea de diseño si nos basamos en una plataforma específica.

- Brinda una mejor experiencia de usuario, ya que dispones de acceso a las interfaces y controles nativos del dispositivo.
- Se puede desarrollar actualizaciones constantes para beneficio de las personas.
- No requieren de una conexión a internet para funcionar.

- Desventajas

- No todas las plataformas pueden gozar de las mismas funciones.
- Su costo de inversión es más elevado.
- El código desarrollado solo sirve para una sola plataforma. Si deseamos que nuestra aplicación esté disponible para otro sistema, tendremos que diseñarla de nuevo utilizando otro lenguaje.

Desarrollar un proyecto como una aplicación nativa es la mejor alternativa si vamos a crear una aplicación con gran alcance, el costo va a ser elevado y queremos aprovechar al máximo las prestaciones de los dispositivos [7].

5.1.2. Aplicaciones Web

Cuando creas una aplicación móvil dependes de que sea aceptada en la tienda de la plataforma, y aún siendo aceptada, más tarde puede ser eliminada de la misma. Por esto muchos desarrolladores han optado por el enfoque web, lo que les da libertad de existencia fuera de esas tiendas de plataforma y así poder ofrecer sus servicios a otros usuarios de dispositivos móviles o de escritorio. Suelen ser desarrolladas usando HTML, CSS y Javascript. Para aprovechar su potencial se hace uso de los frmaworks y librerias como Angular, React o Vue [6].

- Ventajas[8]

- Puedes crear una aplicación desde un mismo código para que sea válida para varias plataformas.
- Buena experiencia de usuario. Ya que es fácil su adaptación a varias plataformas y tamaños de dispositivos.
- Acceso flexible; puedes entrar desde cualquier dispositivo.
- Siempre actualizada, ya que todos están accediendo a la misma versión a través de una URL.

- Almacenamiento más amplio, gracias a la posibilidad de la nube.

- Desventajas

  - La aplicación móvil está restringida a las capacidades del navegador del dispositivo móvil del usuario.
  - Dependen de la conexión a internet.
  - Muchos negocios creen que los datos están menos seguros en la nube.
  - Velocidad reducida.

El enfoque web es una buena opción cuando quieres realizar una aplicación con un acceso flexible, para varias plataformas y además vaya a necesitar de un almacenamiento más grande de lo normal.

5.1.3. Aplicaciones Híbridas

Esta solución es una mezcla, de ahí el nombre, entre las aplicaciones nativas y las web. La aplicación será escrita usando tecnologías web (HTML, CSS, Javascript), y posteriormente será encapsulada en una aplicación nativa. A través del uso de plugins, estas aplicaciones tienen acceso a las características del dispositivo, ventajas propias del enfoque nativo.

Aunque la aplicación sea escrita con tecnología web, no se muestra en un navegador, sino que esta desplegada en una aplicación nativa y su propio navegador, el cual es invisible al usuario. Por ejemplo en Android se usará WebView para desplegar la aplicación.

El código sera embebido en un envoltorio. Esta solución crea una aplicación nativa que es solo la vista de la plataforma en la cual será cargada la aplicación web. Esto nos da la habilidad de crear aplicaciones nativas que podrán ser presentadas a cualquier plataforma para su venta [6].

- Ventajas[7]

  - Se pueden visualizar en cualquier teléfono móvil.
  - Permite la reutilización de código ahorrando bastante tiempo a los desarrolladores.
  - Su costo de inversión es más bajo que las nativas.
  - Las funciones serán las mismas sin importar la plataforma en la que se use.

  - Tiene buen rendimiento.
  - Mantenimiento menos complicado que las nativas.

- Desventajas

  - Sus funcionalidades son limitadas ya que no tienen acceso a los recursos del Smartphone.
  - Generalmente requieren una conexión a internet.
  - Visualmente no son tan atractivas como las nativas.
  - Su rendimiento es menos que las nativas.

Esta solución será óptima en el caso de querer desarrollar una aplicación para distintas plataformas, además de cuando se quiera hacer una aplicación que no dependa esencialmente del rendimiento, y se tenga un presupuesto más bajo.

5.2. Razonamiento detrás de la decisión tomada

En nuestro caso queremos desarrollar una aplicación que se pueda utilizar tanto en Android como en iOS, que tenga un almacenamiento no muy amplio y sin mucha carga en el rendimiento del teléfono.

Con estas características nos decantaremos por una solución híbrida, podríamos haber optado por la solución web, pero no queríamos que dependiese de un navegador y las variaciones que esto conlleva. Además de las ventajas que nos brinda una aplicación con rasgos nativos.

Capítulo 6

Seguimiento

6.0.1. Sprint 1

Este Sprint es el asignado para preparación de documentación y aprendizaje de las tecnologías.

Se añadió la tarea 6 y 7 por petición y consejo del tutor respectivamene.

En este sprint hemos completado todas las tareas asignadas, excepto la tarea 5 .ªprendizaje de Flutter"que todavía se precisa de más tiempo para el aprendizaje de las competencias necesarias para la realización de este proyecto, y la tarea 6 "Diseño de base de datos"la cual se necesita más información y feedback referente a su elaboración, por lo cual estas dos pasarán al siguiente sprint.

La distribución y uso de horas de este Sprint se puede ver reflejado en la Tabla 6.1

Tarea	Tipo	Descripción	Horas	Estado
001	Chore	Documentación: Estructura básica del documento, Introducción y Objetivos	4h	Completada
002	Chore	Documentación: Tecnologías móviles	6h	Completada
003	Chore	Documentación: Scrum	6h	Completada
004	Chore	Documentación: Tecnologías	0.5h	Completada
005	Chore	Aprendizaje Flutter	35h	Incompleta
006	Story	Diseño base de datos	2h	Incompleta
007	Chore	Documentación: Riesgos Laborales	1.5h	Completada
Total			**55h**	

Tabla 6.1: Tareas del sprint 1

6.0.2. Sprint 2

En este Sprint se realizarán las tareas pendientes del anterior sprint, además de las referentes a la maquetación de la aplicación y su navegavilidad.

En este Sprint se han completado todas las tareas excepto la tarea 006 "Diseño base de datos", debido a la indefinición de ciertos aspectos de la funcionalidad. Se han realizado un total de 50 horas.

Tarea	Tipo	Descripción	Horas	Estado
005	Chore	Aprendizaje Flutter	25h	Completa
006	Chore	Diseño base de datos	1h	Incompleta
008	HU-4	Maquetación y funcionalidad: Selección de categoría riesgo laboral	5'5h	Completa
009	HU-4	Maquetación y funcionalidad: Selección de subcategoría riesgo laboral	7'5h	Completa
010	HU-5	Maquetación y funcionalidad: Lista riesgos laborales a evaluar	4h	Completa
011	HU-5	Maquetación y funcionalidad: Formulario de evaluación de riesgo laboral	7h	Completa
Total			**50h**	

Tabla 6.2: Tareas del sprint 2

6.0.3. Sprint 3

En este Sprint se realizarán las tareas pendientes del anterior sprint, además de las referentes a las historias de usuario 1 a 3 y la documentación referente al diseño.

Se ha añadido la tarea 19 a causa de la redefinición de los campos de las interfaces. Se ha añadido la tarea 20 a causa de los requisitos. Y la tarea 21 se ha añadido por preferencia del desarrollador.

No se ha realizado la tarea 14 a causa de una falta de información. Y la 18 por falta de tiempo.

Posponemos la finalización de las tareas 14, 18 y 21 para el siguiente Sprint.

Tarea	Tipo	Descripción	Horas	Estado
006	Chore	Diseño base de datos	8h	Completa
012	HU-1	Maquetación y funcionalidad: Listado inspecciones	3h	Completa
013	HU-1	Funcionalidad: creación inspección	2h	Completa
014	HU-2	Funcionalidad: finalizar inspección	0h	Incompleta
015	HU-3	Funcionalidad: eliminar inspección	0'5h	Completa
016	Chore	Estructura manejo de datos	16h	Completa
017	Chore	Documentación: Redefinición requisitos	1'5h	Completa
018	Chore	Documentación: Base de datos y estructura proyecto	0h	Incompleta
019	Chore	Redefinición Interfaces	9h	Completa
020	Chore	Maquetación y funcionalidad: login para inspectores	7'5h	Completa
021	HU-7	Creación Informe csv	5'5h	Incompleta
Total			**53h**	

Tabla 6.3: Tareas del sprint 3

6.0.4. Sprint 4

En este Sprint se realizarán las tareas pendientes del anterior sprint, y las referentes a la documentación y finalización del código.

Se ha añadido la tarea 23 para reflejar el tiempo usado poblando la base de datos de todos los factores de riesgos. La 24 para avanzar en la documentación al mismo ritmo que avanza

la aplicación, y la 25 para la creación de test.

En este Sprint se han realizado las tareas esperadas y empezado otras debido al tiempo disponible.

Finalmente se ha decidido eliminar la tarea 14 a causa de la decisión de no implementar esa funcionalidad en la aplicación.

Tarea	Tipo	Descripción	Horas	Estado
014	HU-2	Funcionalidad: finalizar inspección	0h	Cancelada
018	Chore	Documentación: Base de datos y estructura proyecto	4h	Completa
021	HU-7	Creación Informe csv	6'5h	Completa
022	Chore	Refactorización del código	5h	Completa
023	Chore	Población base de datos	8'5h	Completa
024	Chore	Documentación: Diseño e Implementación	4h	Incompleta
025	Chore	Test: testing	22h	Completa
Total			**48h**	

Tabla 6.4: Tareas del sprint 4

6.0.5. Sprint 5

En este Sprint se realizarán las tareas pendientes del anterior sprint, y las referentes a la documentación final.

En este Sprint se ha terminado la documentación restante del proyecto.

Tarea	Tipo	Descripción	Horas	Estado
024	Chore	Documentación: Diseño e Implementación	1'5h	Completa
026	Chore	Documentación: Manual de usuario	7'5h	Completa
027	Chore	Documentación: Conclusiones y lineas futuras	1h	Completa
Total			**9'5h**	

Tabla 6.5: Tareas del sprint 5

Capítulo 7

Diseño

En este apartado mostraremos lo referente al diseño de la aplicación, desde la base de datos, a la interfaz de la aplicación y su estructura de datos siguiendo el patrón Bloc.

7.1. Diseño base de datos

Para el desarrollo de este proyecto hemos optado por una base de datos relacional hecha en SQFLite, la versión de SQLite de Flutter. Hemos elegido esta aproximación ya que por presupuesto no se podía costear un almacenamiento en la nube, por lo que hemos optado por el almacenamiento local en el dispositivo.

A lo largo de su diseño ha habido varios cambios a medida que el desarrollador y el cliente redefinían los requisitos de la aplicación.

La aplicación consta de unos *Inspectores*, los cuales van realizar una serie de *Inspecciones*, las cuales van a darse en una serie de *Deficiencias* correspondientes a un *Factor de Riesgo*, a su vez estas deficiencias se podrán *Evaluar*, añadiendo unas *Coordenadas* y unas *Fotos*, para así poder reflejar mejor el riesgo potencial o existente. Tal y como se muestra en la figura 7.1.

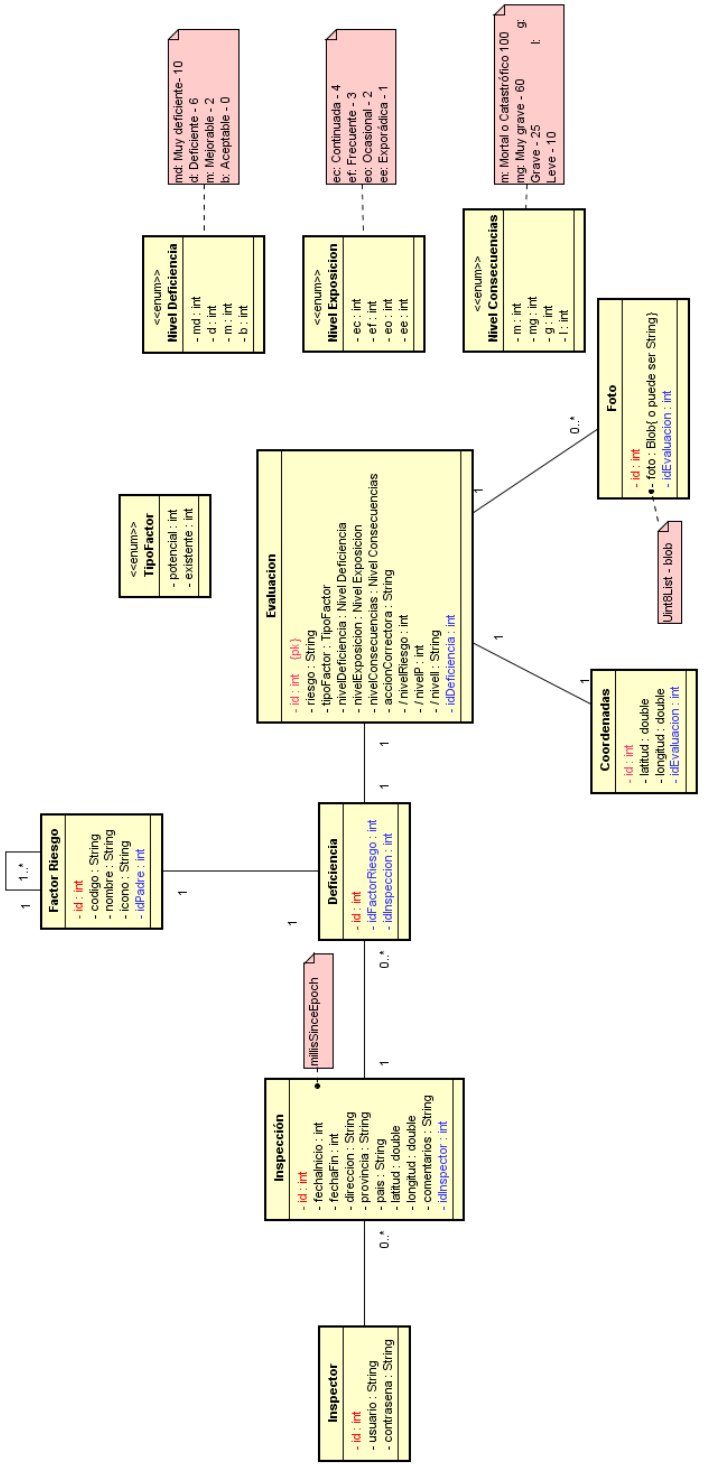


Figura 7.1: Diagrama UML de la Base de Datos

7.2. Inherited Widget

Un Inherited Widget es una clase base de los widgets, la cual propaga información a lo largo del árbol de widgets tal como se muestra en la figura 7.2.

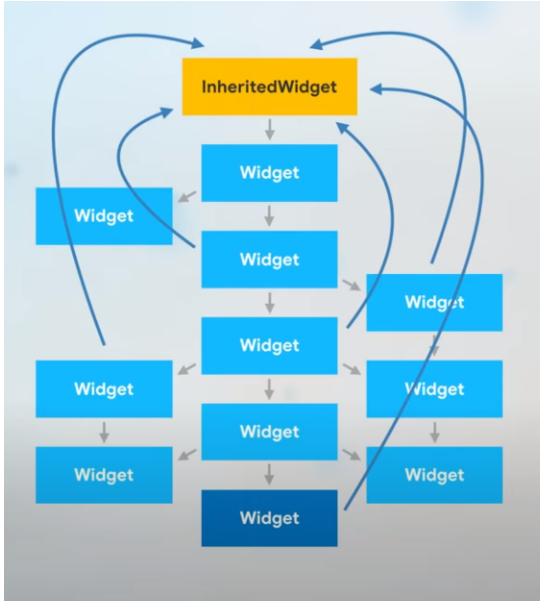


Figura 7.2: Inherited Widget
[19]

Esto es posible gracias al context, el cual guarda el árbol de widgets. Todo lo que tiene que hacer un widget que necesita la información del Inherited Widget es buscar en su árbol el ancestro.

En nuestra aplicación se usará para mantener un acceso a los datos centralizado.

7.3. Patrón Bloc(Buisness Logic Component)

Es una forma para manejar el estado de la data de nuestra aplicación de forma que este centralizada esa información. Ahora bien, ¿cuál es la diferencia con el Inherited Widget?.

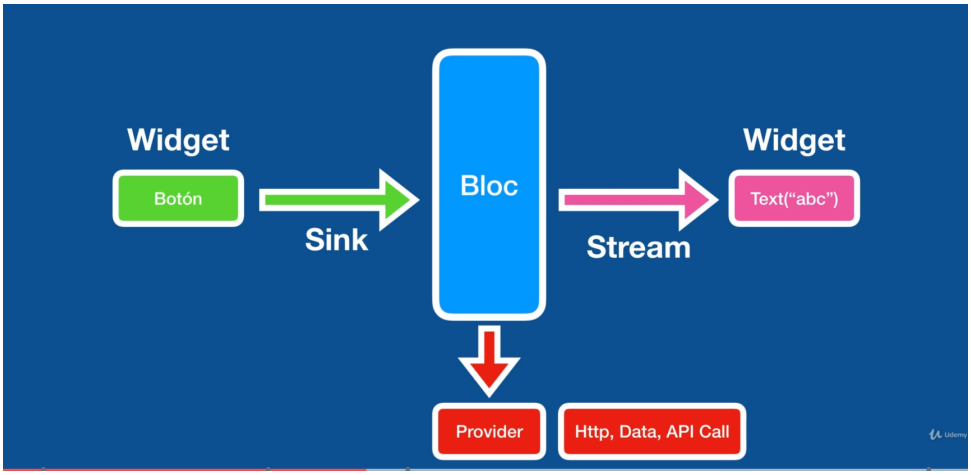


Figura 7.3: Patrón Bloc
[20]

La diferencia principal es que un Bloc está diseñado de forma que tenga un *Stream* de salida, uno de entrada y posiblemente un *StreamController* para la transformación de los datos, de forma que puede entrar y salir tipos de datos totalmente diferentes. Para entender esto tenemos la figura 7.4

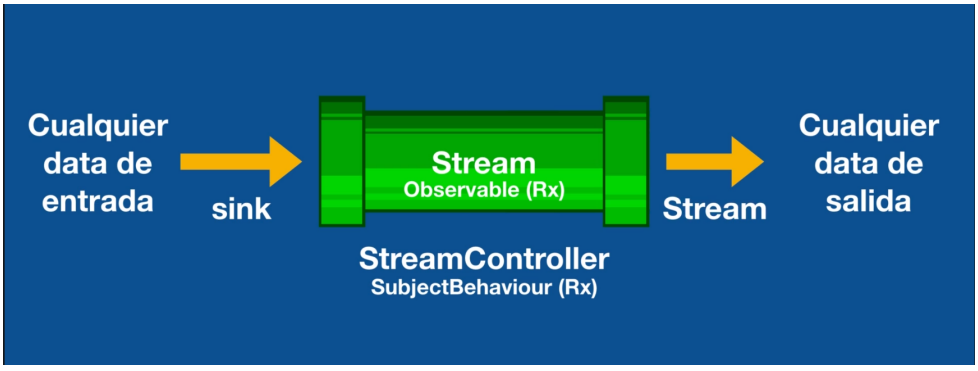


Figura 7.4: Streams
[20]

Hay dos tipos de *StreamControllers*:

- SingleSubscription: solo lo puede escuchar un Widget a la vez.
- Broadcast: para cuando la información se del interes de más de un Widget a la vez.

Nosotros usaremos el broadcast, debido a que habrá varios Widgets que necesiten la data de los Bloc.

En nuestra aplicación usaremos el patrón Bloc para manejar los datos de Inspecciones, Deficiencias, Evaluaciones; de forma que estén separados y haya una gestión de datos más limpia y no tan sobrecargada.

Estos Blocs estarán dentro del Inherited Widget para poder tener un acceso único a los datos y se pueda acceder a ellos desde cualquier Widget.

7.4. Patrón Singelton

Es un patrón de diseño que permite restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. [21]

Esto quiere decir que es un patrón que permite garantizar tener una sola instancia de un objeto, clase...(etc). De esta forma no se crean varias instancias de una misma clase, lo cual permitiría un mejor manejo de los datos.

En nuestra aplicación lo usaremos para la creación de los respectivos Blocs, para así asegurarnos de la coherencia de los datos a lo largo del funcionamiento de la aplicación.

7.5. Estructura de la aplicación

Como he ido comentando en las secciones anteriores los datos se guardarán en una *base de datos*, a la cual los *Blocs* accederán a través de un provider de la misma, estando estos Blocs englobados por un *Inherited Widget*. Consiguiendo así una estructura de datos centralizada y bien estructurada.

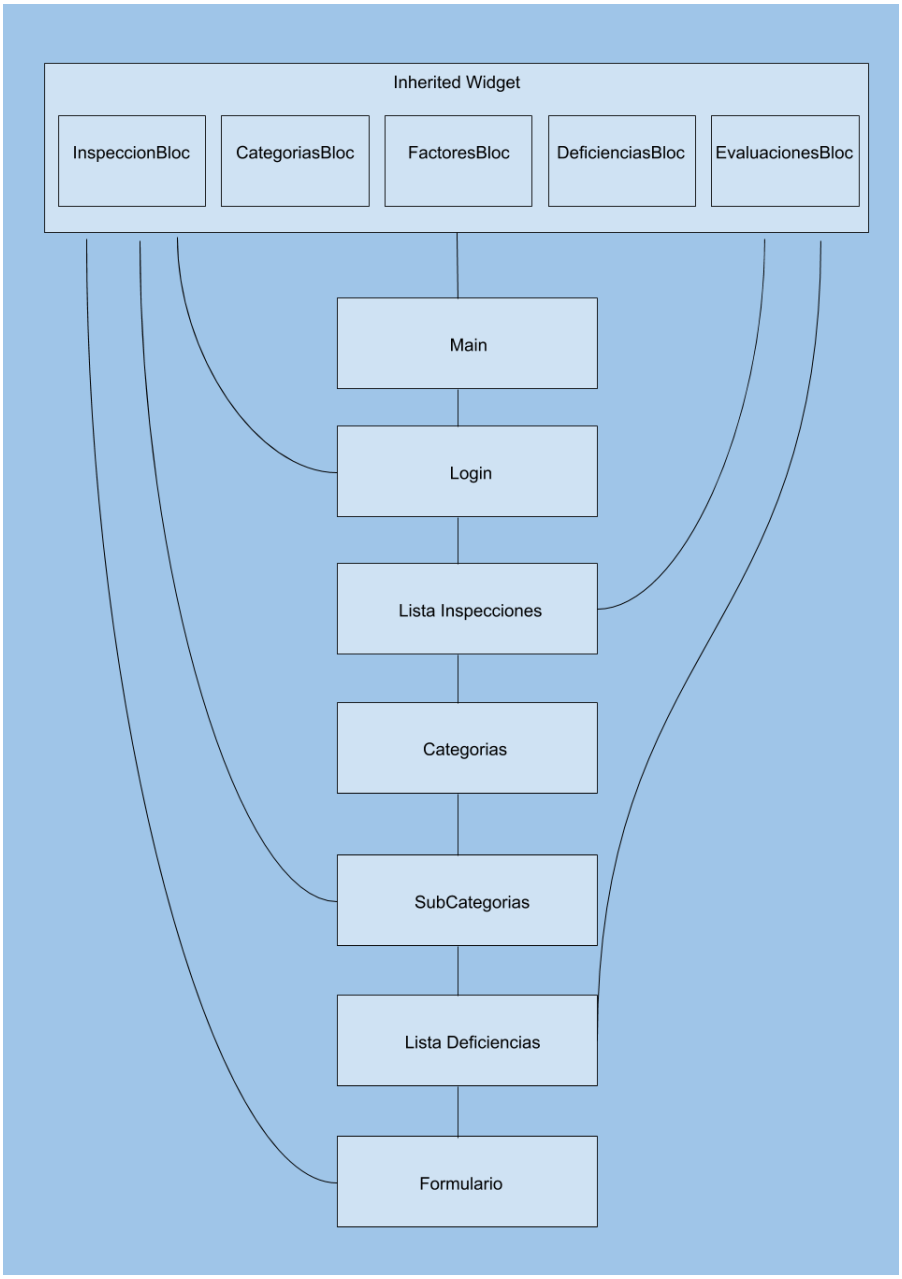


Figura 7.5: Estructura de datos

7.6. Diseño interfaz de usuario

El diseño de la interfaz de usuario se ha ido acordando a lo largo del desarrollo de esta misma con el usuario, primero se le mostró una demo, y más tarde se le hizo un refinamiento de acuerdo a los requisitos funcionales presentados por el mismo, además del punto de vista del desarrollador y de algunas pruebas de UX(Experiencia de Usuario) que se han ido haciendo periódicamente.

En las Figuras 7.6, 7.7 y 7.8 se puede ver la demo mencionada anteriormente.

Figura 7.6: Demo 1

Figura 7.7: Demo 2

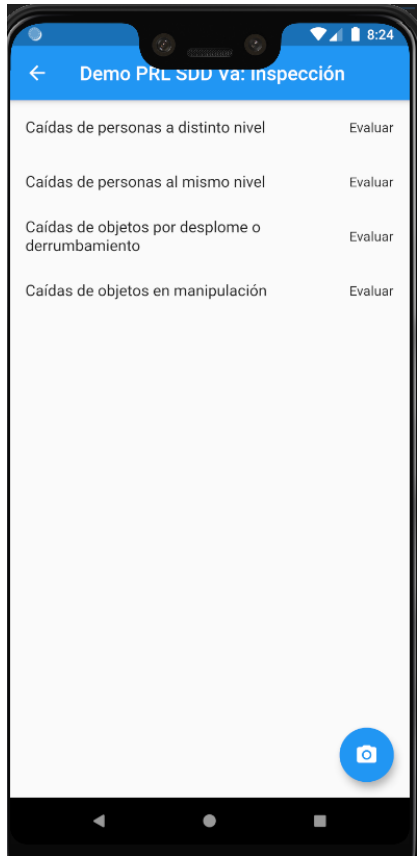


Figura 7.8: Demo 3

En estas figuras se puede ver un listado de los riesgos (Figura 7.6), un listado de las posibles deficiencias (Figura 7.7), junto con un contador de deficiencias en la esquina derecha superior, el cual te lleva a la tercera figura, una lista de evaluaciones (Figura 7.8), además de un botón para añadir fotografías.

El diseño final se ha buscado que sea comprensible, intuitivo, fácil de entender y cómodo de usar, ya que será una aplicación que se usará a diario. Es una aplicación que esta destinada a personal con conocimientos en riesgos laborales y el desarrollo de sus informes, por lo cual no se espera un uso y aprendizaje fluido en personas que no tengan ese conocimiento previo.

7.7. Implementación

En esta sección se habla de como se ha estructurado el código de la aplicación, así como cambios y decisiones sobre la aplicación durante su desarrollo.

7.7.1. Estructura del código

El desarrollo y la estructura del código sigue los estándares de Flutter.

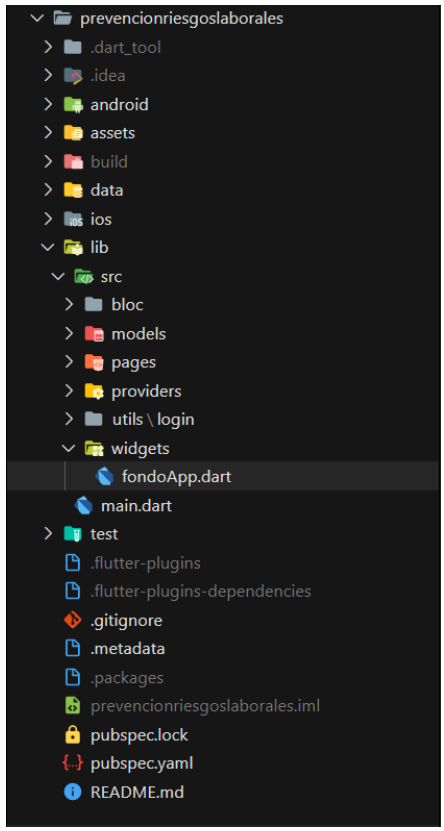


Figura 7.9: Estructura del código

En la Figura 7.9 se pueden observar las distintas partes del código, organizadas en carpetas:

- **assets:** En esta carpeta están guardados los iconos e imágenes base de la aplicación.

- **data:** Contiene unos archivos JSON que se usan para la población de la base de datos la primera vez que se abre la aplicación en un dispositivo.
- **src:** Es la carpeta que engloba el código fuente de la aplicación.
- **bloc:** En esta carpeta están los Blocs y el Inherited Widget comentados en los apartados 7.3 y 7.2 respectivamente.
- **models:** En esta carpeta están los modelos de la estructura de datos y los responsables de la transformación de los datos de la base de datos a la aplicación.
- **pages:** En esta carpeta están las páginas de la aplicación.
- **providers:** En esta carpeta están los providers responsables de la base de datos y su población.
- **utils:** En esta carpeta está una utilidad del login.
- **widgets:** En esta carpeta están los widgets reutilizados en varias páginas.

7.7.2. Decisiones y cambios

A lo largo del desarrollo de la aplicación se han ido produciendo cambios en su interfaz o funcionamiento, ya sea por petición del cliente, por mejora de su funcionalidad o la experiencia del usuario.

En principio la aplicación constaba de 4 apartados, 3 de ellos son los que aparecieron en la demo inicial ya comentada en el apartado 7.6. Los cuales constaban de una lista de factores de riesgo, de una lista de los sub-factores de cada uno de ellos y de una lista de evaluaciones. Además de estos se esperaba la construcción de un formulario para la realización de dicha evaluación.

Más tarde aparecieron los cambios, hubo que crear una página con la lista de inspecciones, que pasaría a ser la página inicial.

Después de un tiempo se vio la necesidad de crear un login, además de añadir aspectos a la creación de inspecciones como la geo-localización, la cual también se añadió al formulario, junto con la petición de poder guardar varias fotografías en cada evaluación.

7.8. Testing

Testing es una fase muy importante en el ciclo de vida del desarrollo de una aplicación[22]. Da seguridad en la calidad de la aplicación, permite saber si todo esta funcionando como se espera, incluso después de cambios que puedan comprometer el correcto funcionamiento de esta, y sirve como criterio de aceptación para el cliente.

En Flutter disponemos de tres tipos de testing automático:

- **Tests Unitarios**
  Es el método más fácil de testear nuestra aplicación. Testea una función o un módulo, pero no refleja la totalidad de la aplicación. [22]
- **Tests de Widgets**
  Se encargan de la correcta creación y renderización de los widgets. [22]
- **Tests de Integración**
  Incluye tanto test unitarios como de widgets, junto con componentes externos como las bases de datos, servicios web, etc. Simula el funcionamiento real de la aplicación.

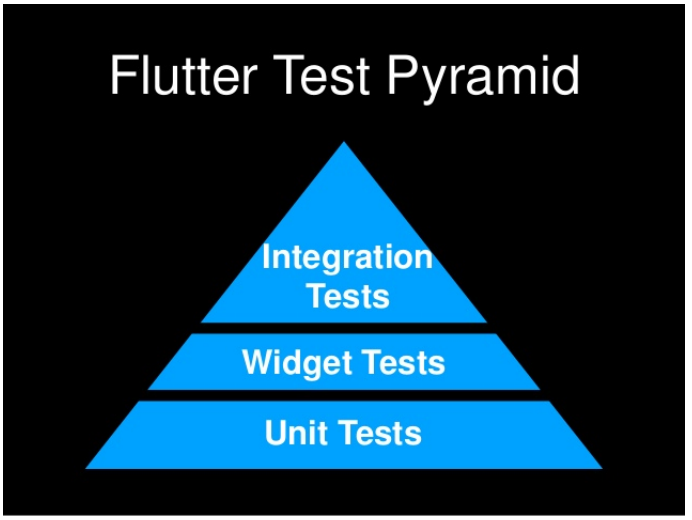


Figura 7.10: Piramide de Test de Flutter

Para nuestra aplicación se realizarán test unitarios y test de integración. No se han realizado test de widgets ya que su correcta creación ya es testeada en los test de integración.

7.8.1. Test Unitarios

Hemos realizado los test unitarios necesarios para probar el correcto funcionamiento de los modelos de datos y las funciones y módulos pertinentes a la lógica de negocio del proyecto.

Gracias a estos test podemos asegurar el correcto tratamiento de datos entre la *Base de Datos* y la *Interfaz gráfica* de la aplicación. Así como el comportamiento esperado de las funciones y módulos testeados.

Para ejecutarlos basta con usar VSCode y presionar el botón de *Start Debugging*.

Cuando se realicen los tests aparecerá la barra lateral con los resultados de los test, tal como aparece en la Figura 7.11

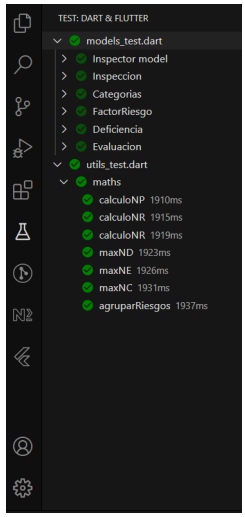


Figura 7.11: Test Unitarios

7.8.2. Test Integración

En los test de integración hemos testeado el correcto flujo de acciones de cada una de las pantallas de nuestra aplicación. Asegurando así su correcto comportamiento en el momento de su uso por parte de un usuario final.

Para la creación de un test de integración primero hay que abrir la pantalla que se quiere testear y después realizar las pruebas automáticas. Por esta razón se han dejado comentados todos los test menos uno, y deben ser ejecutados uno por uno. Para ello se debe comentar el test anterior y dejar solamente el test a testear y su proceso de creación de la interfaz gráfica.

Habrá que seguir los siguientes pasos:

- Primero debemos dirigirnos a la carpeta */test_driver*.
- Una vez allí abrimos el archivo *app.dart*.
- Buscamos la pantalla que queramos testear y se des-comenta, comentando todas las demás.
- Después abrimos el archivo *app_test.dart* y des-comentamos el grupo de test perteneciente a esa pantalla y comentamos los demás.
- Y finalmente se ejecuta en la terminal el siguiente comando:
  flutter drive –target=test_driver/app.dart

Al finalizar cada test habrá una respuesta en el terminar como la de la Figura 7.12

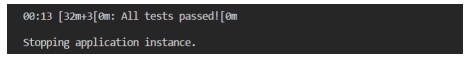


Figura 7.12: Test Integración

Capítulo 8

Lineas Futuras y Conclusiones

8.1. Lineas Futuras

En este apartado hablaré de las funcionalidades y posibles cambios que se puede implantar en la aplicación para su mejora y uso cotidiano.

8.1.1. COVID

En el futuro se puede añadir una serie de **Factores de Riesgo** relacionados con el COVID y sus correspondientes riesgos y su efecto en el día a día.

8.1.2. Firebase

En este momento la aplicación usa el almacenamiento del dispositivo móvil usando **SQ-FLite**, de modo que si se desinstala la aplicación o se pierde el dispositivo móvil no se podrían recuperar los datos. Por lo que en un futuro se recomienda que se migre la base de datos a **Firebase**.

8.1.3. Plantillas

Además se espera la adición de una serie de plantillas para que así no aparezca todos los **Factores de Riesgo**, además de la posibilidad de cambios en los formularios de las

Evaluaciones de estos riesgos según sea de un ámbito o de otro, dependiendo de la plantilla seleccionada al crear la **inspección**.

Esto permitiría un uso más fluido y menos tedioso por parte del usuario. Además de una categorización más clara de las **Inspecciones**.

8.2. Conclusiones

Al finalizar el desarrollo del proyecto podemos considerar que se ha realizado satisfactoriamente y alcanzado los objetivos esperados de la aplicación.

- Se ha creado una interfaz intuitiva y práctica para el uso de profesionales en el ámbito de la evaluación y prevención de riesgos laborales.
- Se ha creado una base de datos local al dispositivo móvil para el correcto funcionamiento de la aplicación.

En cuanto a los objetivos personales considero que los he superado con creces ya que:

- He adquirido un uso fluido en el framework Flutter.
- Así como la creación de test unitarios y de integración.
- Y el uso de muchas librerías necesarias para el uso del dispositivo móvil, como son la ubicación, la cámara, la galería y el almacenamiento.

No esperaba aprender tanto como he aprendido y sobretodo he descubierto lo mucho que me gusta el desarrollo en este framework, de modo que lo usaré en un ámbito personal y si se presenta la ocasión en un ámbito profesional.

Apéndice A

Manual de usuario

En este apartado se explicará el proceso de generación del APK tanto para Android como para IOs, además del funcionamiento de la aplicación para así ofrecer una forma de aprender rápidamente el uso de sus funcionalidades.

A.1. Creación de APK

Antes de realizar la APK de Android o IOs si se quiere subir al *Play Store* o a la *Apple AppStore* debemos firmar la aplicación con un id único, además de realizar los pagos pertinentes a las distintas plataformas.

A.1.1. Android

Para generar la APK de Android se usa el comando:

```
flutter build apk --release
```

El proceso puede tardar unos minutos dependiendo de la capacidad del ordenador. Al finalizar, aparecerá el PATH donde se acaba de generar el APK:

Figura A.1: Captura de la creación del APK

A.1.2. IOs

Para generar la APK de IOs se usa el comando:

```
flutter build ios --release
```

Igual que en la de Android tardará un poco en generarse.

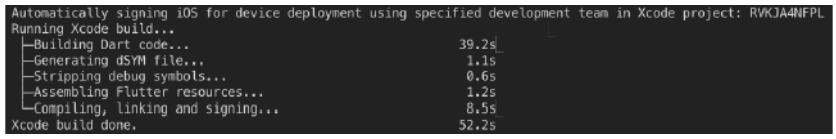


Figura A.2: Captura de la creación del APK para IOs

A.2. Funcionamiento de la aplicación

La aplicación permite la documentación de las inspecciones para la evaluación y Prevención de Riesgos Laborales que se realicen en distintos lugares, así como la generación de un informe resumen de la evaluación.

Cuando entras en la aplicación te encuentras con un Login, tal y como se muestra en la Figura A.3. Además se puede hacer visible lo escrito en el campo *Contraseña* para así asegurarse de su correcta escritura.

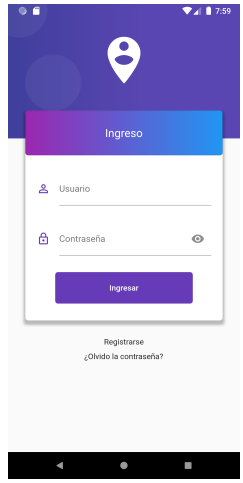


Figura A.3: Login

Para poder **Ingresar** en la aplicación antes hay que **Registrarse**. Clickando el botón *Registrarse* saldrá el model que se muestra en la figura A.4.

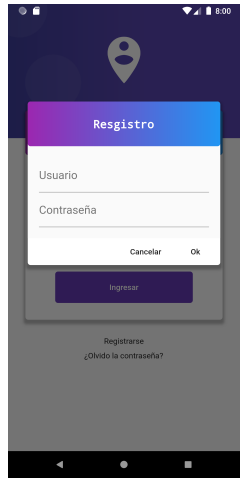


Figura A.4: Registro

Se escribe el usuario y la contraseña que desea tener, se pulsa el botón *Ok* y si el usuario no es repetido se registrará satisfactoriamente y aparecerá el mensaje o *Snackbar* como aparece en la Figura A.5.

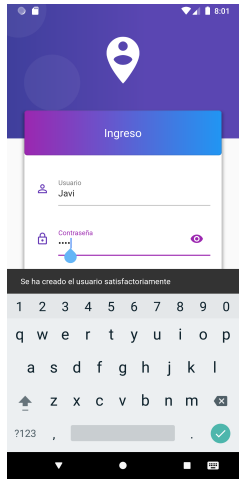


Figura A.5: Creación satisfactoria de un usuario

Después de *Ingresar* nos encontraremos en la pantalla en la que se muestra una lista de las inspecciones pertenecientes a ese usuario. De modo que si es la primera vez que entramos en la aplicación la pantalla que aparecerá será la correspondiente a la Figura A.6

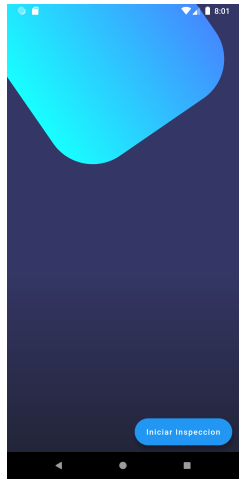


Figura A.6: Página con listado de Inspecciones

Clickando en el botón flotante *Iniciar Inspección* aparecerá el modal de creación de la inspección para rellenar los datos necesarios (Figura A.7).

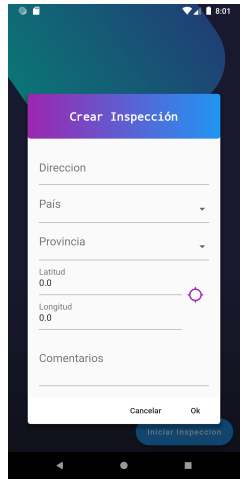


Figura A.7: Modal de creación de Inspección

Para rellenar los datos vamos a necesitar:

- **Dirección**
  Un texto escrito.
- **País**
  Seleccionar una opción.
- **Provincia**
  Seleccionar una opción.
- **Latitud** y **Longitud**
  Clickar en el botón al lado de *Latitud* y *Longitud*, aceptar dar los permisos pertinentes a la ubicación y esperar a que se active la ubicación de tu dispositivo móvil para así obtener los datos pertinentes a la misma.
- **Comentario**
  Un texto escrito (opcional).

Al terminar de rellenar los datos clickamos el botón *Ok*, se creará la inspección, cerrará el modal y aparecerá en la lista de inspecciones la **Inspección** recién creada, tal y como se muestra en la Figura A.8 , mostrando la *Dirección* y los botones pertinentes a empezar la inspección y a la creación del informe resumen.

Figura A.8: Lista Inspecciones con una inspección

Si clickamos en el botón *Inspeccionar* pasaremos a una pantalla en la que aparecerá una lista de **Factores de Riesgo** de primer nivel (Figura A.9).

Figura A.9: Lista Factores de Riesgo de primer nivel

Ahora seleccionamos el **Factor de Riesgo** que se adecue más al riesgo que se quiere

evaluar, lo que nos llevará a la pantalla con **Factores de Riesgo** de segundo nivel (Figura A.10).

Figura A.10: Lista Factores de Riesgo de segundo nivel

Una vez en esta pantalla podemos seleccionar los riesgos que queremos evaluar y se van a ir añadiendo como podemos ver en circulo rojo del botón *Evaluar riesgos*. Dicho esto podemos volver a la pantalla anterior, seleccionar otro **Factor de Riesgo** de primer nivel, que nos llevará a esta misma pantalla con los correspondientes **Factores de Riesgo** de segundo nivel, y seleccionar los que se necesiten para añadirlos a la lista. Más tarde cuando se quiera evaluar cualquiera de estos riesgos se selecciona el botón flotante *Evaluar riesgos*, lo que nos llevará a la pantalla de la Figura A.11.

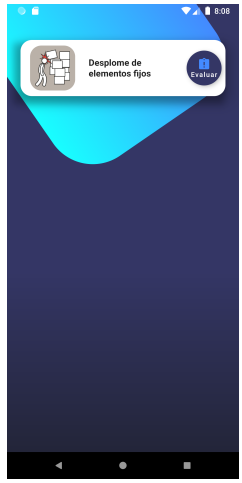


Figura A.11: Lista Factores de Riesgo a Evaluar

Si seleccionamos el botón *Evaluar* o clickamos la tarjeta que lo contiene nos llevará a la pantalla de las Figuras A.12, A.13 la cual es un formulario para rellenar con los datos necesarios para **Evaluar** el riesgo.

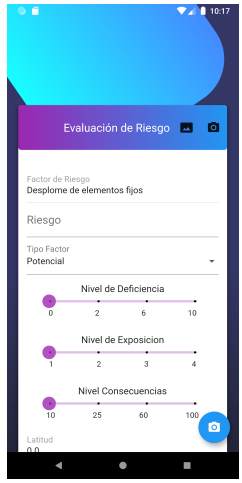


Figura A.12: Formulario evaluar

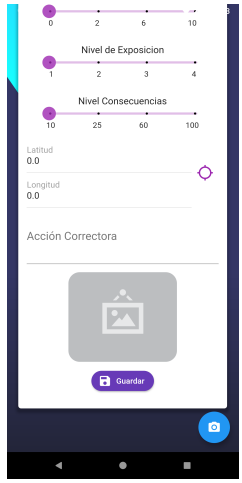


Figura A.13: Formulario evaluar

Los campos del formulario son los siguientes:

- **Factor de Riesgo**
  El Factor de Riesgo de segundo nivel seleccionado.
- **Riesgo**
  Una explicación más específica del riesgo.
- **Tipo Factor**
  Puede ser *Potencial* o *Existente*.
- **Nivel de Deficiencia**
  Slider con los valores referentes a este campo, seleccionar según normas o criterio propio.
- **Nivel de Exposición**
  Slider con los valores referentes a este campo, seleccionar según normas o criterio propio.
- **Nivel Consecuencias**
  Slider con los valores referentes a este campo, seleccionar según normas o criterio propio.
- **Latitud y Longitud**
  Clickar en el botón al lado de *Latitud* y *Longitud*, aceptar dar los permisos pertinentes a la ubicación y esperar a que se active la ubicación de tu dispositivo móvil para así obtener los datos pertinentes a la misma.
- **Acción Correctora**
  Acción propuesta para solventar el riesgo o prevenirlo.

- **Fotografías**
  Se puede seleccionar la imagen vacía o el botón flotante con el icono de la cámara para abrir la cámara del dispositivo y tomar una fotografía. Además los iconos que se ven en la cabecera del formulario en la Figura A.12, la cámara abrirá la cámara y la imagen abrirá la galería para seleccionar una fotografía.

Podemos añadir una fotografía como se ve en la Figura A.14 o varias como se ve en la Figura A.15. Sin importar si proviene de la galería o de la cámara del dispositivo.

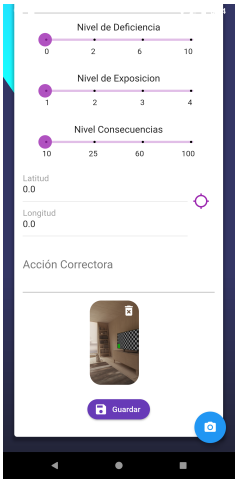


Figura A.14: Formulario con una fotografía

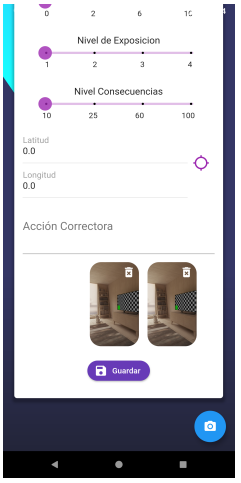


Figura A.15: Formulario con dos fotografías

Una vez clickemos en el botón *Guardar*, se guardará la evaluación en la Base de Datos y se volverá a la pantalla anterior.

Cuando se haya hecho todo esto podemos volver a la pantalla de la Figura A.6 y seleccionar el botón *Informe*, lo cual creará el **informe** resumen de la **Inspección** en el directorio *Download* de nuestro dispositivo (Figura A.16).

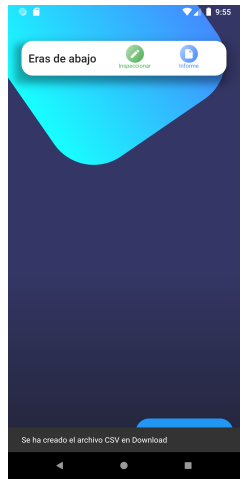
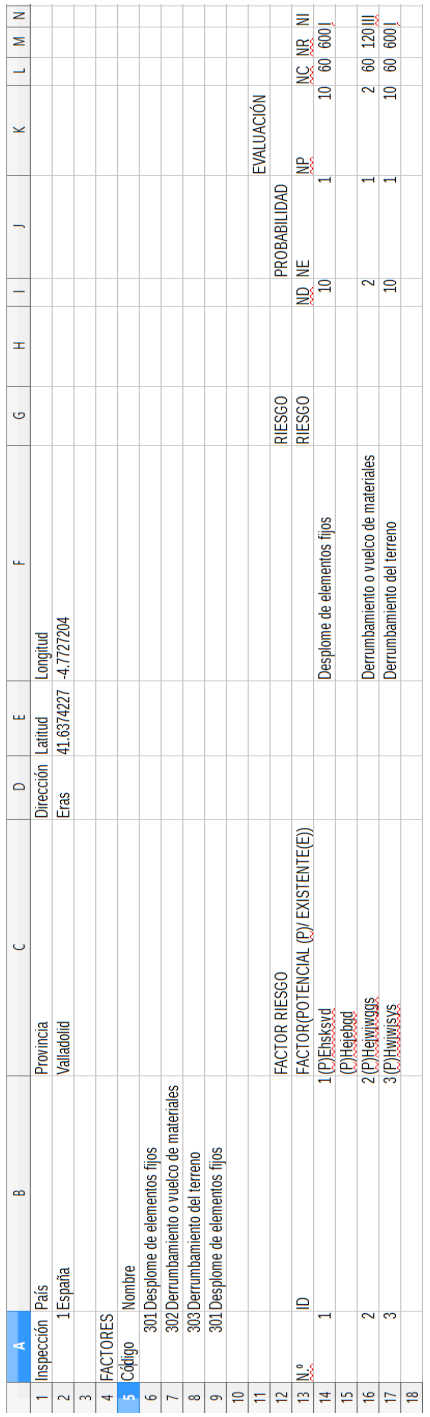


Figura A.16: Creación de Informe

El **Informe** resumen será un archivo Excel en el cual aparecerá la información de la **Inspección**, una lista de los **Factores de Riesgo** que se han seleccionado durante la misma, y una tabla resumen de los datos de las evaluaciones. (Figura A.17).



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Inspección	País	Provincia	Dirección	Latitud	Longitud								
2	1	España	Valladolid	Eras	41.6374227	-4.7727204								
3														
4	FACTORES													
5	Código	Nombre												
6	301	Desplome de elementos fijos												
7	302	Derrumbamiento o vuelco de materiales												
8	303	Derrumbamiento del terreno												
9	301	Desplome de elementos fijos												
10														
11											EVALUACIÓN			
12			FACTOR RIESGO				RIESGO			PROBABILIDAD				
13	N.º	ID	FACTOR(POTENCIAL (P)/ EXISTENTE(E))				RIESGO		ND	NE	NP	NC	NR	NI
14	1		1 (P)Elsksyd			Desplome de elementos fijos			10	1	10	60	600	I
15			(P)Hejebgd											
16	2		2 (P)Heiwiygqs			Derrumbamiento o vuelco de materiales			2	1	2	60	120	III
17	3		3 (P)Hwiwisys			Derrumbamiento del terreno			10	1	10	60	600	I
18														

Figura A.17: Archivo Excel

Apéndice B

Enlaces adicionales

Algunos de los enlaces de interés de este TFG son:

- `https://github.com/Sakrence/TFG.git` Repositorio de GitHub donde se encuentra el código de la aplicación.

- `https://trello.com/b/5eT3itXn` Tablero de Trello que se ha usado a lo largo del desarrollo.

Bibliografía

[1] Facultad de Filosofía y Letras, Universidad de Málaga *Prevención de la PRL en España*, Recuperado de:
https://prevencion-riesgoslaborales.com/historia-prl-espana/

[2] Ministerio de trabajo y economía social *Guía Laboral - La prevención de riesgos laborales*, Recuperado de:
http://www.mitramiss.gob.es/es/Guia/texto/guia_10/contenidos/guia_10_22_1.htm

[3] Agencia Estatal Boletín Oficial del Estado *Ley 31/1995, de 8 de noviembre, de prevención de Riesgos Laborales*,
Recuperado de:
https://www.boe.es/eli/es/l/1995/11/08/31/con

[4] The App Analytics and App Data Industry Standard *State of Mobile-2020*, Recuperado de:
https://www.appannie.com/en/go/state-of-mobile-2020/

[5] *Top Technologies Used to Develop Mobile App*, Recuperado de:
https://www.fingent.com/blog/top-technologies-used-to-develop-mobile-app/

[6] *What is Hybrid App Development?*, Recuperado de:
https://ionicframework.com/resources/articles/what-is-hybrid-app-development

[7] *Apps Nativas VS Híbridas: Ventajas y Desventajas*, Recuperado de:
https://www.vexsoluciones.com/apps-moviles/apps-nativas-vs-hibridas/

[8] *Advantages And Disadvantages – Web Apps*, Recuperado de:
https://objectiveit.com/blog/the-advantages-and-disadvantages-of-web-apps/

[9] Ken Schwaber y Jeff Sutherland *La Guía Definitiva de Scrum: Las Reglas del Juego* , Recuperado de:

https://www.scrumguides.org/docs/scrumguide/v2016/
2016-Scrum-Guide-Spanish.pdf

[10] Alejandro Frechina *Metodología Scrum ¿Que es?*, Recuperado de:
https://winred.es/management/metodologia-scrum-que-es/
gmx-niv116-con24594.htm

[11] ScrumDictionary.com, Recuperado de:
https://scrumdictionary.com/term/chore/

[12] ScrumDictionary.com, Recuperado de:
https://scrumdictionary.com/term/story/

[13] ScrumDictionary.com, Recuperado de:
https://scrumdictionary.com/term/epic/

[14] Atlassian, Recuperado de:
https://trello.com/es

[15] GitHub, Recuperado de:
https://github.com/

[16] Visual Studio Code, Recuperado de:
https://code.visualstudio.com/

[17] Google, Recuperado de:
https://flutter-es.io/

[18] Flutter, Recuperado de:
https://flutter-es.io/docs/cookbook/persistence/sqlite

[19] API Flutter, Recuperado de:
https://api.flutter.dev/flutter/widgets/InheritedWidget-class.html

[20] UDEMY, Recuperado de:
https://www.udemy.com/course/flutter-ios-android-fernando-herrera/learn/
lecture/14732968#overview

[21] Wikipedia, Recuperado de:
https://es.wikipedia.org/wiki/Singleton

[22] Flutter - Testing, Recuperado de:
https://www.tutorialspoint.com/flutter/flutter_testing.htm