



**UNIVERSIDAD DE CASTILLA-LA MANCHA
ESCUELA SUPERIOR DE INFORMÁTICA**

GRADO EN INGENIERÍA INFORMÁTICA

Ingeniería del Software

TRABAJO FIN DE GRADO

**HERA iOS: Herramienta Móvil de Soporte al Proceso
de Triaje durante Traslados Sanitarios**

Alejandro Sánchez Arcos

julio, 2024



**UNIVERSIDAD DE CASTILLA-LA MANCHA
ESCUELA SUPERIOR DE INFORMÁTICA**

Departamento de Tecnologías y Sistemas de Información

Ingeniería del Software

TRABAJO FIN DE GRADO

**HERA iOS: Herramienta Móvil de Soporte al Proceso
de Triaje durante Traslados Sanitarios**

Autor: Alejandro Sánchez Arcos

Tutor(a): José Antonio Cruz Lemus

Co-tutor(a): Fernando de Cózar Ruano

julio, 2024

HERA iOS

© Alejandro Sánchez Arcos, 2024

Este documento se distribuye con licencia CC BY-NC-SA 4.0. El texto completo de la licencia puede obtenerse en <https://creativecommons.org/licenses/by-nc-sa/4.0/>.

La copia y distribución de esta obra está permitida en todo el mundo, sin regalías y por cualquier medio, siempre que esta nota sea preservada. Se concede permiso para copiar y distribuir traducciones de este libro desde el español original a otro idioma, siempre que la traducción sea aprobada por el autor del libro y tanto el aviso de copyright como esta nota de permiso, sean preservados en todas las copias.

Este texto ha sido preparado con la plantilla L^AT_EX para Trabajo Fin de Estudios en Ingeniería Informática para la UCLM publicada por [Jesús Salido](#) en el repositorio público Zenodo, DOI: [10.5281/zenodo.4561708](https://doi.org/10.5281/zenodo.4561708), como parte del curso «[L^AT_EX esencial para preparación de TFG, Tesis y otros documentos académicos](#)» impartido en la Escuela Superior de Informática de la Universidad de Castilla-La Mancha.



TRIBUNAL:

Presidente: _____

Vocal: _____

Secretario(a): _____

FECHA DE DEFENSA: _____

CALIFICACIÓN: _____

PRESIDENTE

VOCAL

SECRETARIO(A)

Fdo.:

Fdo.:

Fdo.:

*A mis abuelos,
por creer en mí desde pequeño.*

HERA iOS

Alejandro Sánchez Arcos
Ciudad Real, julio 2024

Resumen

En el dinámico y crítico entorno de las emergencias médicas, cada segundo cuenta. Por ello, el proceso de triaje en las ambulancias es fundamental para priorizar la atención en función de la gravedad de los pacientes. Sin embargo, los métodos tradicionales de triaje enfrentan desafíos que pueden suponer un aumento en el riesgo de la salud de los pacientes. Brindar a los sanitarios de una herramienta que optimice su tiempo durante una emergencia puede suponer la diferencia entre salvar una vida o no.

Desafortunadamente, en la actualidad, en España todavía hay comunidades autónomas que no cuentan con este tipo de herramientas. *Inetum* ha tenido la oportunidad de desarrollar HERA iOS, una solución cuyo objetivo es modernizar el proceso de triaje y mejorar la atención al paciente en todas aquellas comunidades autónomas que deseen optimizar su sector sanitario.

Este Trabajo de Fin de Grado (TFG) se ha desarrollado en el contexto del convenio FORTE, firmado entre la Escuela Superior de Informática (ESI) e *Inetum*. En él se abordará el desarrollo de HERA iOS, utilizando una arquitectura VIPER y tecnología nativa de Apple (Swift), específicamente diseñada para dispositivos iOS (iPad). Se implementarán funcionalidades para la creación de informes, permitiendo completar diferentes tipos como informes de exploración primaria, física o de anamnesis, con detalles como alergias, enfermedades, y evaluación de los pulmones, entre otros. Los informes podrán ser enviados al hospital para proporcionar información relevante sobre la emergencia. Además, se podrá acceder a la información básica de los afectados en un incidente y asignarlos a unidades de emergencia. La plataforma estará disponible tanto *online* como *offline*, para asegurar que los sanitarios puedan llenar los informes en caso de falta de conexión a Internet, entre otras funcionalidades.

Gracias al desarrollo de esta herramienta, los sanitarios tendrán la oportunidad de optimizar su tiempo, ya que este tipo de herramientas permiten una atención médica más precisa y eficaz. Esto les permitirá enfocarse en lo realmente importante: la atención directa al paciente.

HERA iOS

Alejandro Sánchez Arcos
Ciudad Real, July 2024

Abstract

In the dynamic and critical environment of medical emergencies, every second counts. Therefore, the triage process in ambulances is essential to prioritize care based on the severity of patients. However, traditional triage methods face challenges that may increase the risk to patients' health. Providing healthcare professionals with a tool that optimizes their time during an emergency can make the difference between saving a life or not.

Unfortunately, currently in Spain, there are still autonomous communities that do not have this type of tools. *Inetum* has had the opportunity to develop HERA iOS, a solution aimed at modernizing the triage process and improving patient care in all those autonomous communities that wish to optimize their healthcare sector.

This Bachelor dissertation has been developed in the context of the FORTE agreement, signed between the School of Computer Science and *Inetum*. It will address the development of HERA iOS, using a VIPER architecture and Apple's native technology (Swift), specifically designed for iOS devices (iPad). Functionalities will be implemented for report creation, allowing the completion of different types such as primary examination reports, physical reports, or medical history reports, with details such as allergies, diseases, and lung evaluations, among others. Reports can be sent to the hospital to provide relevant information about the emergency. Additionally, basic information about the affected individuals in an incident can be accessed and assigned to emergency units. The platform will be available both online and offline, to ensure that healthcare professionals can fill out reports in case of lack of Internet connection, among other functionalities.

Thanks to the development of this tool, healthcare professionals will have the opportunity to optimize their time, as this type of tool allows for more precise and effective medical care. This will enable them to focus on what really matters: direct patient care.

Agradecimientos

Un viejo refrán dice que cada final es el comienzo de una nueva aventura. Con estas palabras, inicio el agradecimiento de mi Trabajo de Fin de Grado, que no solo marca el cierre de una etapa fundamental y especial en mi vida, sino que también abre la puerta a un umbral de caminos por descubrir. Es el momento de expresar mi gratitud a todos aquellos que han contribuido a hacer esto posible. Sin embargo, siento que las palabras por sí solas no pueden expresar plenamente el profundo agradecimiento que siento por la ayuda y confianza depositadas en mí.

A mis padres, Francisco e Inma, por ser los pilares fundamentales en mi vida ya que sin vuestro esfuerzo, sacrificio y cariño no habría llegado hasta aquí. A mi hermana, Laura, por su apoyo en todas mis decisiones y por ser el incordio más maravilloso de mi vida. Al resto de mi familia, por estar siempre ahí y por confiar siempre en mí, incluso cuando yo no lo hacía.

A mis amigos que conocí en la carrera, Lucía, Gonzalo A. y Gonzalo G., por esas llamadas interminables hasta las cinco de la mañana para entregar un trabajo; por las risas en las cervezas; por esos momentos de desconexión durante los descansos; y, sobre todo, por compartir este camino juntos, lleno de alegrías y sufrimiento, que me han llevado a convertirme, en gran parte, en lo que soy.

A la Escuela Superior de Informática y a la empresa Inetum, por brindarme la oportunidad de participar en el programa FORTE.

A todo el equipo de Movilidad de Inetum, por la gran acogida y por convertirme en uno más desde el primer día; por vuestra ayuda y orientación; y sobre todo, por haberme facilitado el camino durante mi primera experiencia laboral.

Y por último, a mis tutores, José Antonio por parte de la ESI y Fernando de Inetum Agradezco vuestra orientación constante, paciencia infinita al responder preguntas que podrían considerarse poco relevantes, y sobre todo, por dedicarme todo el tiempo necesario para poder hacer esto posible.

A todos, simplemente gracias.

Alejandro Sánchez Arcos
Ciudad Real, 2024

Lista de acrónimos

API	:	Application Programming Interface
CMMI	:	Capability Maturity Model Integration
DNI	:	Documento Nacional de Identidad
DOD	:	Definition Of Done
ECDE	:	Estació Clínica d'Emergències
ESI	:	Escuela Superior de Informática
FORTE	:	FORTalecimiento de la Empleabilidad
HERA	:	Health Record from Ambulance
HTTP	:	Hypertext Transfer Protocol
JSON	:	JavaScript Object Notation
JWT	:	JSON Web Token
MVP	:	Minimum Viable Product
OSHCAR	:	Registro Español de Parada Cardiaca Extrahospitalaria
PBI	:	Product Backlog Item
SACYL	:	Salud de Castilla y León
SEM	:	Sistema de Emergencias Médicas
SITREM	:	Sistema Integral de Tratamiento de Emergencias
SMS	:	Sistema Murciano de Salud
SVB	:	Soporte Vital Básico
SVA	:	Soporte Vital Avanzado
TDD	:	Test-Driven Development
TFG	:	Trabajo Fin de Grado
URL	:	Uniform Resource Locator

Índice general

Resumen	V
Abstract	VII
Agradecimientos	IX
Lista de acrónimos	XI
Índice de figuras	xv
Índice de tablas	xix
1. Introducción	1
1.1. Motivación y Contexto	1
1.2. FORTE en Inetum	1
1.3. Estructura del documento	2
2. Objetivo	5
2.1. Objetivo Principal	5
2.2. Objetivos Parciales	5
2.3. Objetivos Docentes	6
3. Estado del Arte	7
3.1. Historia del proyecto HERA	7
3.2. Patrón de Arquitectura VIPER	13
4. Metodología	17
4.1. Metodología de Trabajo en Inetum	17
4.2. <i>Scrum</i>	19
4.3. Adaptaciones de <i>Scrum</i> para el desarrollo de este TFG	24
4.4. Tecnologías Software y Hardware Utilizadas	24
5. Resultados	29
5.1. <i>Inception</i> de HERA iOS	29
5.2. Estructura del Desarrollo de los Sprints	38
5.3. Sprint 1	38
5.4. Sprint 2	44
5.5. Sprint 3	52
5.6. Sprint 4	60
5.7. Sprint 5	66
5.8. Diagrama de Gantt	72
6. Conclusiones	73
6.1. Consecución de objetivos parciales	73
6.2. Consecución de objetivos docentes	75
6.3. Consecución del objetivo general	76
6.4. Trabajo Presente y Futuro	76

6.5. Valoración crítica	78
A. Prototipos de interfaces	81
A.1. Prototipo interfaz Login Offline	81
A.2. Prototipo interfaz Selección de unidad	81
A.3. Prototipo interfaz Menú Principal	82
A.4. Prototipo interfaz Bandeja de incidentes	82
A.5. Prototipo interfaz Pestaña Cabecera	83
A.6. Prototipo interfaz Pestaña Tipo y Finalización de servicio	83
A.7. Prototipo interfaz Pestaña Anamnesis	84
A.8. Prototipo interfaz Pestaña Exploración Primaria	84
A.9. Prototipo interfaz Pestaña Exploración Física 1	85
A.10. Prototipo interfaz Pestaña Exploración Física 2	85
A.11. Prototipo interfaz Pestaña Exploración Física 3	86

Índice de figuras

1.1. Relación ESI- <i>Inetum</i> en la realización de TFG dentro del programa FORTE	2
3.1. Captura de login en proyecto SMS.	9
3.2. Captura de asignación de unidad en proyecto SMS.	10
3.3. Captura de menú principal en proyecto SMS.	10
3.4. Captura de la pestaña Cabecera del módulo de creación y edición de informes en proyecto SMS.	10
3.5. Captura de la pestaña Exploración física III del módulo de creación y edición de informes en proyecto SMS.	11
3.6. Captura de bandeja de incidentes en proyecto SMS.	11
3.7. Captura de login en proyecto Sacyl.	12
3.8. Captura de menú principal en proyecto Sacyl.	12
3.9. Captura de la pestaña Datos administrativos del módulo de creación y edición de informes en proyecto Sacyl.	12
3.10. Captura de la pestaña Valoración secundaria III del módulo de creación y edición de informes en proyecto Sacyl.	13
3.11. Captura de bandeja de incidentes en proyecto Sacyl.	13
3.12. Arquitectura de VIPER.	14
3.13. Flujos de comunicación entre los componentes. Fuente: traducción de <i>VIPER. Design pattern's modules interaction description</i>	15
3.14. Diagrama de clases Módulo VIPER.	15
4.1. Fases del ciclo de vida de un proyecto en <i>Inetum</i>	18
4.2. Pilares y Valores de <i>Scrum</i> . Fuente: Scrum.org	19
4.3. Roles de <i>Scrum</i>	20
4.4. Artefactos de <i>Scrum</i>	22
4.5. Eventos de <i>Scrum</i>	22
4.6. Procedimiento <i>Planning Poker</i>	23
4.7. Marco tecnológico HERA iOS	24
4.8. <i>Agile Board</i> con incidencia en la tarea SURE-1250.	25
4.9. Incidencia abierta de la tarea SURE-1250.	25
4.10. Interfaz de ramas en <i>Sourcetree</i> de HERA iOS	26
4.11. <i>Podfile</i> de HERA iOS	27
5.1. Resultado Participantes <i>Elevator Pitch</i> de HERA iOS	30
5.2. Resultado final <i>Elevator Pitch</i> de HERA iOS en Miro.	31
5.3. Comunidad del Proyecto de HERA iOS en Miro.	32
5.4. Restricciones de HERA iOS en Miro.	34
5.5. Riesgos de HERA iOS en Miro.	35
5.6. Story Map de HERA iOS en Miro.	37
5.7. <i>Login online</i> HERA iOS	40
5.8. Selección de unidad de HERA iOS	41
5.9. <i>Home</i> o menú principal de HERA iOS	42
5.10. Fragmento de código de acceso a memoria <i>sandbox</i> para crear la carpeta donde se van a guardar los informes en la <i>tablet</i>	43
5.11. Fragmento de código para crear un nuevo informe en memoria <i>sandbox</i>	43
5.12. Listado de informes en interfaz de <i>Home</i> de HERA iOS	46

5.13. Eliminación informe en interfaz de <i>Home</i> de HERA iOS	46
5.14. Filtrado de informes por número en interfaz de <i>Home</i> de HERA iOS	47
5.15. Interfaz Bandeja de Incidentes de HERA iOS	48
5.16. Interfaz Bandeja de Incidentes a la hora de asignar un afectado de HERA iOS	49
5.17. Interfaz Bandeja de Incidentes a la hora de desasignar un afectado de HERA iOS	49
5.18. Inicio de nuevo informe desde Bandeja de incidentes de HERA iOS	50
5.19. Interfaz cabecera de módulo de edición y creación de informes de HERA iOS	51
5.20. Gráfico <i>Burndown</i> proporcionado por Jira del <i>sprint 2</i> de HERA iOS	52
5.21. Interfaz de pestaña “Cabecera” del módulo de edición y creación de informes de HERA iOS	54
5.22. Definición del protocolo para la funcionalidad del guardado en el módulo de edición y creación de informes de HERA iOS	55
5.23. Definición del protocolo para la funcionalidad del guardado en pestaña “Cabecera” de HERA iOS	55
5.24. Autoguardado exitoso en pestaña “Cabecera” del módulo de edición y creación de informes de HERA iOS	56
5.25. Autoguardado fallido en pestaña “Cabecera” del módulo de edición y creación de informes de HERA iOS	56
5.26. Menú superior donde se encuentra la funcionalidad de guardado en HERA iOS	57
5.27. Guardado al salir de un informe en HERA iOS	57
5.28. Componente <i>MultipleSelectionView</i> en pestaña “Tipo y Finalización de servicio” del módulo de edición y creación de informes de HERA iOS	58
5.29. Interfaz de pestaña “Tipo y Finalización de servicio” del módulo de edición y creación de informes de HERA iOS	59
5.30. Gráfico <i>Burndown</i> proporcionado por Jira del <i>sprint 3</i> de HERA iOS	60
5.31. Funcionalidades de contenedor “Tipo de servicio” de pestaña “Tipo y finalización de servicio” de HERA iOS	62
5.32. Funcionalidad de contenedor “Fin de paciente” de pestaña “Tipo y finalización de servicio” de HERA iOS	62
5.33. Componente <i>FillFieldsView</i> en pestaña “Anamnesis” del módulo de edición y creación de informes de HERA iOS	63
5.34. Interfaz de pestaña “Anamnesis” del módulo de edición y creación de informes de HERA iOS	64
5.35. Funcionalidad botón “Estado general” de pestaña “Exploración Primaria” del módulo de edición y creación de informes de HERA iOS	65
5.36. Funcionalidad del contenedor “Pupilas” de pestaña “Exploración Primaria” del módulo de edición y creación de informes de HERA iOS	65
5.37. Gráfico <i>Burndown</i> proporcionado por Jira del <i>sprint 4</i> de HERA iOS	66
5.38. Interfaz de pestaña “Exploración Física 1” del módulo de edición y creación de informes de HERA iOS	68
5.39. Interfaz de pestaña “Exploración Física 2” del módulo de edición y creación de informes de HERA iOS	69
5.40. Interfaz de pestaña “Exploración Física 3” del módulo de edición y creación de informes de HERA iOS	70
5.41. Interfaz de inicio de sesión <i>offline</i> de HERA iOS	71
5.42. Gráfico <i>Burndown</i> proporcionado por Jira del <i>sprint 5</i> de HERA iOS	71
5.43. Diagrama de Gantt durante el proyecto de HERA iOS	72
 6.1. Control de error si se intenta guardar un informe sin número de afectado en HERA iOS	76
6.2. Control de error si se intenta guardar un informe con un formato erróneo en el DNI del paciente en HERA iOS	76
6.3. Cambios en interfaz <i>home</i> para la integración de dispositivos <i>Bluetooth</i> de HERA iOS	77
6.4. Interfaz integración dispositivos <i>Bluetooth</i> de HERA iOS	78
 A.1. Prototipo interfaz login offline	81
A.2. Prototipo interfaz selección unidad.	81
A.3. Prototipo interfaz menú principal.	82
A.4. Prototipo interfaz bandeja de incidentes.	82
A.5. Prototipo interfaz pestaña Cabecera.	83
A.6. Prototipo interfaz pestaña Tipo y Finalización de servicio.	83

A.7. Prototipo interfaz pestaña Anamnesis.	84
A.8. Prototipo interfaz pestaña Exploración Primaria.	84
A.9. Prototipo interfaz pestaña Exploración Física 1.	85
A.10. Prototipo interfaz pestaña Exploración Física 2.	85
A.11. Prototipo interfaz pestaña Exploración Física 3.	86

Índice de tablas

4.1. Responsabilidades de cada rol dentro de <i>Scrum</i>	21
4.2. Especificaciones hardware.	28
5.1. Resultado final <i>Elevator Pitch</i> de HERA iOS	30
5.2. Comunidad del Proyecto de HERA iOS	32
5.3. Restricciones de Fechas y Entregables.	33
5.4. Acciones de mitigación y contingencia para riesgos de HERA iOS	36
5.5. <i>Product Backlog</i> inicial de HERA iOS con prioridades y épicas.	37
5.6. Planificación del <i>Sprint 1</i>	38
5.7. Planificación del <i>Sprint 1</i> en tareas/horas.	39
5.8. Estimación tareas de la historia de usuario H1: Login online	39
5.9. Estimación tareas de la historia de usuario H2: Selección de unidad	40
5.10. Estimación tareas de la historia de usuario H3: Pantalla Home	41
5.11. Estimación tareas de la historia de usuario H4: Gestión de informes guardados	42
5.12. Estimación tareas de la historia de usuario H5: Nuevo informe offline	43
5.13. Planificación del <i>Sprint 2</i>	44
5.14. Planificación del <i>Sprint 2</i> en tareas/horas.	45
5.15. Estimación tareas de la historia de usuario H6: Pantalla con afectados	47
5.16. Estimación tareas de la historia de usuario H7: Inicio de informes desde bandeja de entrada	50
5.17. Estimación tareas de la historia de usuario H8: Pantalla de edición de formularios	51
5.18. Planificación del <i>Sprint 3</i>	52
5.19. Planificación del <i>Sprint 3</i> en tareas/horas.	53
5.20. Estimación tareas de la historia de usuario H9: Pestaña de cabecera	53
5.21. Estimación tareas de la historia de usuario H10: Autoguardado	54
5.22. Estimación tareas de la historia de usuario H11: Opción de guardar informe y guardar al salir	57
5.23. Estimación tareas de la historia de usuario H12: Pestaña tipo y finalización de servicio	58
5.24. Estimación tareas de la historia de usuario H13: Subida de informe	59
5.25. Planificación del <i>Sprint 4</i>	61
5.26. Planificación del <i>Sprint 4</i> en tareas/horas.	61
5.27. Estimación tareas de la historia de usuario H14: Pestaña Anamnesis	63
5.28. Estimación tareas de la historia de usuario H15: Pestaña Exploración Primaria	64
5.29. Planificación del <i>Sprint 5</i>	66
5.30. Planificación del <i>Sprint 5</i> en tareas/horas.	67
5.31. Estimación tareas de la historia de usuario H16: Pestaña Exploración física 1	67
5.32. Estimación tareas de la historia de usuario H17: Pestaña Exploración física 2	67
5.33. Estimación tareas de la historia de usuario H18: Pestaña Exploración física 3	68
5.34. Estimación tareas de la historia de usuario H19: Pruebas WS	69
5.35. Estimación tareas de la historia de usuario H20: Acceso offline	70
6.1. Resumen de satisfacción de objetivos parciales.	75
6.2. Planificación del <i>Sprint 6</i>	77
6.3. Estimación tareas de la historia de usuario H21: Integración con dispositivos bluetooth . .	77

CAPÍTULO 1

Introducción

En el presente capítulo se abordará, en primer lugar, la contextualización y motivación que dieron origen a este Trabajo de Fin de Grado (TFG), seguido de una síntesis del contenido de cada uno de los capítulos que lo componen.

1.1. MOTIVACIÓN Y CONTEXTO

El sector sanitario se encuentra actualmente inmerso en un periodo de notable transformación, impulsado por la aparición de novedosas herramientas tecnológicas que están reemplazando radicalmente la forma en que se brinda la atención médica.

A pesar de los avances considerables en este sector, especialmente en el diagnóstico, tratamiento e investigación de nuevas patologías, el progreso no se ha extendido uniformemente hacia algunos ámbitos de la salud del paciente.

Uno de los casos más evidentes, es el proceso de triaje durante el traslado de pacientes en ambulancias, donde la escasa tecnología utilizada y protocolos empleados son rudimentarios.

Las ambulancias desempeñan un rol crucial en la atención médica del paciente, llegando a ser un factor determinante en la supervivencia del mismo en situaciones extremas. Brindar al personal sanitario herramientas tecnológicas de soporte rápidas y eficientes es fundamental para optimizar su trabajo durante el traslado del paciente.

Estas herramientas permiten una atención médica más precisa y oportuna, lo que en algunas ocasiones puede suponer la diferencia entre salvar la vida de un paciente o no.

En muchas de las comunidades autónomas de España se siguen cumpliendo los informes de triaje de ambulancias a mano, un proceso tedioso, lento y propenso a errores que un *software* podría hacer de una manera más eficaz y en menos tiempo.

En consecuencia, se presenta a la empresa *Inetum*¹ la oportunidad de desarrollar **HERA iOS** (HEalth Record from Ambulance), una herramienta de soporte para dispositivos iPad (iOS²), destinada a optimizar el proceso de triaje durante el traslado de pacientes en ambulancias.

HERA iOS se presenta como una solución eficaz para modernizar el proceso de triaje mejorando la atención del paciente para todas aquella comunidades autónomas que quieran optimizar su sector sanitario.

Inetum ofrece el desarrollo de esta herramienta como proyecto del convenio **FORTE**³ o Fortalecimiento de la Empleabilidad de la Escuela Superior de Informática. El objetivo principal de este convenio es el acercamiento del alumno a la empresa en un proyecto y en un ambiente de trabajo real participando con un equipo experimentado, realizando prácticas y TFG de forma conjunta.

1.2. FORTE EN INETUM

En relación al entorno del presente Trabajo de Fin de Grado (TFG), como se ha mencionado en la sección 1.1 Motivación y Contexto, se ha realizado durante el periodo de prácticas en la empresa *Inetum* como proyecto del programa **FORTE** (FORTalecimiento de la Empleabilidad), abarcando desde el mes de febrero hasta julio de 2024.

¹<https://www.inetum.com/es>

²<https://www.apple.com/es/ios>

³<https://esi.uclm.es/index.php/programas-singulares/programa-forte/>

Dicho programa surge de la iniciativa de acercar al alumno al mundo empresarial antes de concluir el grado, teniendo la oportunidad de realizar tanto las prácticas como el TFG de forma conjunta en un proyecto real en una empresa. En consecuencia, el alumno adquirirá experiencia y competencias al ser partícipe de un equipo experimentado de la empresa, encargado de alcanzar con el proyecto asignado unos objetivos empresariales más amplios.

Inetum ha formado parte de esta iniciativa propuesta por la ESI (Escuela Superior de Informática) desde casi su origen, permitiéndole ganar una amplia experiencia instruyendo a alumnos que están acabando el grado, debido a su atención e implicación. Gracias a su participación en más de trece ediciones del programa FORTE, *Inetum* se encuentra en una posición óptima para la atracción de talento, dado que facilita una realización sobresaliente del TFG por parte del alumno. Este éxito, en la mayoría de las circunstancias, resulta en la posibilidad de garantizar la retención del estudiante en la empresa, lo cual representa una de las motivaciones por las cuáles las organizaciones realizan este programa.

Durante el transcurso del desarrollo del Trabajo de Fin de Grado (TFG), diversos miembros del equipo de desarrollo de la empresa, particularmente el co-tutor académico, junto con el tutor académico colaboran activamente con el alumno, proporcionándole orientación en la organización y diseño del resultado final. Incluso, en algunas circunstancias, el alumno puede recibir el apoyo de compañeros de la empresa que realizaron su TFG dentro del programa FORTE en la misma empresa y que fueron contratados, lo que resulta fundamental para fomentar el entendimiento y la comprensión entre ambas partes.

En la Figura 1.1, se presentan algunos Trabajos de Fin de Grado realizados a lo largo de los años en el marco del programa FORTE, en colaboración entre la ESI e *Inetum*.

RELACIÓN FORTE ESI-INETUM			
Trabajo de Fin de Grado	Autor	Año	Estado
MULPER: Sistema para la gestión de cuentas de personal de una empresa	Carmen María Palmero Yébenes	2015	Completado
METRICAS: Herramienta para la Generación de Informes en base a Indicadores	Alberto Crespo Sánchez	2017	Completado
OMNIA: Sistema de seguridad app móviles empresa	Silvia Muñoz Atienza	2017	Completado
VECINET: Extranet para la gestión de fichas de hoteles de V.C.Inglés	Enrique Forner Díaz	2018	Completado
CyCredApp: Aplicación de pólizas de crédito	Fernando de Cozár Ruano	2018	Completado
Social Media Dashboard: Herr. Medio social empresa	Gonzalo Pérez Fernández	2018	Completado
ONCOSUP: Sistema de Gestión de Datos de Pac. Super.	María López Carrasco	2018	Completado
Sistema de Gestión de procedimientos Administrativos del Ayunt. Barcelona	Christian Rivera Balseras	2018	Completado
SPA: Sistema de Producciones Ajenas de Radio Televisión Española	Amanda Sánchez García	2019	Completado
GECO: Sistema de Gestión de Expedientes	Maria de los Ángeles Moreno Jiménez	2019	Completado
TPRE: Trámites de Procedimientos de Reintegro	Arturo Carretero Simón	2020	Completado
Evolutivo Portfolio Plugin para JIRA	Mario Donaire Becerra	2022	Completado
ECIPOFRC: Ent. De Colaboración Inter. Pensiones - Procesos Batch	Daniel Gijón Robas	2020	Completado
Smart mallorca: App Turismo	Óscar Domínguez Ocaña	2020	Completado
EULEN RH 4.0	Ramón Díaz Rodrigo	2022	Completado
Cartera Proyectos Inetum	Guilomar Pareja de Diego	2023	Completado
Herramienta P. Plan Seg. Económico Proyectos	Guillermo Santos Molero	2024	En proceso
Aplicat. I.N. Estadística	Raúl Moya Campillo	2024	En proceso
Arq. Web. Híbrida con OpenAI	Carlos Almagro Rubio	2024	En proceso
Agile Journey Platform	Alejandro Quintana Rodríguez	2024	En proceso
Plataforma RTVE Open AI	Alonso Hurtado Lillo	2024	En proceso
...			

Figura 1.1: Relación ESI-*Inetum* en la realización de TFG dentro del programa FORTE.

1.3. ESTRUCTURA DEL DOCUMENTO

Este documento se estructura, a partir del capítulo 1 de introducción de la siguiente forma:

- **Capítulo 2 - Objetivos del TFG:** En el presente capítulo se describirá el objetivo principal del TFG, así como los objetivos parciales y docentes que permitirán alcanzarlo.
- **Capítulo 3 - Estado del Arte:** En este capítulo se detallará una investigación sobre la historia del proyecto y su aplicación base en la que está fundamentada el TFG. Se expondrá en qué consiste, así como sus módulos funcionales y adaptaciones que sirven de modelo para la realización de la nueva aplicación. Además se procederá a explicar y justificar el uso del patrón de arquitectura para aplicaciones móviles VIPER, que se va a utilizar en el desarrollo de este proyecto.
- **Capítulo 4 - Metodología:** En este capítulo se introducirá la metodología seleccionada por *Inetum*. Se presentará el marco de trabajo *Scrum* explicando sus características fundamentales así como los roles que lo forman, eventos que se realizan, artefactos producidos y la técnica de estimación de historias de usuario *Planning Poker*. Además, se describirán las adaptaciones de *Scrum* que se han realizado. Por último, se detallará el marco tecnológico utilizado para el desarrollo de este TFG.
- **Capítulo 5 - Resultados:** En el presente capítulo se describirá con detalle el desarrollo del proyecto, donde se aplicará la metodología anteriormente mencionada en el capítulo 4. Cada sección de este

capítulo contendrá el desarrollo de un *sprint*, conservando una estructura uniforme que incluirá: el objetivo general del *sprint*, la planificación del *sprint*, el desarrollo del mismo por historias de usuario y la reunión de revisión incluyendo una conclusión final del *sprint*. Además, se incluirá una sección que detallará la reunión de *Inception*, durante la cual se estableció la visión global del proyecto y su alcance. Por último, se mostrará el diagrama de Gantt con el objetivo de tener una visión general del tiempo empleado en el proyecto.

- **Capítulo 6 - Conclusiones:** En este capítulo, se presentarán las conclusiones alcanzadas por el autor tras la realización del TFG, indicando si se han cumplido o no los objetivos planteados en el capítulo 2. Además, se describirá el trabajo actual que se está realizando en este proyecto, así como los próximos avances necesarios para completarlo. Por último, se expondrá una valoración crítica del autor de este TFG.

Al final del documento se incluye un anexo, denominado Anexo A: Prototipos de interfaces. En este anexo se muestran los prototipos utilizados para el desarrollo de las interfaces en **HERA iOS**.

Cabe destacar que no se ha incluido una sección específica para la bibliografía en el documento. Las referencias bibliográficas, junto con cualquier otro contenido citado, se han añadido como notas al pie en distintas páginas (*ad hoc*), con el fin de mejorar la legibilidad y aprovechar de manera más eficiente la navegación de recursos *online*.

CAPÍTULO 2

Objetivo

En este capítulo se presenta el objetivo principal del TFG, así como los objetivos parciales y docentes que permitirán alcanzarlo.

2.1. OBJETIVO PRINCIPAL

El objetivo general del TFG es el desarrollo de una **aplicación móvil iOS**, siguiendo un patrón de arquitectura VIPER y desarrollado con tecnología nativa (Swift¹). La finalidad de esta aplicación es facilitar al sector sanitario una herramienta de **soporte que les ayude a optimizar el proceso de triaje durante el traslado de pacientes en ambulancias**. Esta herramienta les facilitaría la tarea de cumplimentar informes a mano, un proceso tedioso y repetitivo, ahorrándole tiempo y esfuerzo, pudiendo dedicar toda su atención al paciente. Además, habría una gestión del informe del paciente en tiempo real, pudiéndose comunicar con el hospital para poder analizar el triaje del paciente antes de la llegada.

2.2. OBJETIVOS PARCIALES

Para poder alcanzar el objetivo general, se ha dividido en una serie de objetivos parciales, los cuáles realizarán los procesos de estimación, planificación, desarrollo y pruebas de los siguientes módulos:

- **Módulo de acceso a la plataforma y selección de unidad.**

El módulo de acceso a la plataforma se define como dos interfaces distintas: una diseñada para facilitar el acceso en línea a la plataforma y otra concebida para permitir el acceso sin conexión a internet.

En el caso de acceso *online*, se empleará el protocolo de autenticación abierto *Apereo CAS*, el cual proporciona una interfaz estandarizada para iniciar sesión. Para el acceso sin conexión, se implementará una interfaz alternativa que permitirá la identificación del usuario mediante su nombre de usuario y contraseña.

El módulo de selección de unidad consistirá en una interfaz, donde el usuario podrá introducir o seleccionar la unidad de asistencia deseada para acceder al menú principal de la plataforma.

- **Módulo de gestión de la edición y creación de informes, tanto *online* como *offline*.**

El módulo actual se compone de dos interfaces que se encuentran disponibles para su selección dentro del menú principal de la plataforma: una destinada a la creación de informes y otra para la edición de informes previamente creados, con la capacidad de funcionar tanto en línea como sin conexión.

La interfaz destinada a la creación de un nuevo informe SVA (Soporte Vital Avanzado) proporcionará al usuario la capacidad de introducir información detallada sobre un paciente mediante diversas pestañas, dependiendo del tipo de información.

La interfaz destinada a la visualización de informes ofrecerá al usuario un listado de los informes almacenados, permitiéndole seleccionarlos para visualizar, modificar o eliminar la información de cada una de las pestañas.

En ambas interfaces, el usuario tendrá la posibilidad de guardar el informe del paciente en la *tablet* en cualquier momento, así como de enviarlo al hospital.

¹<https://www.apple.com/lae/swift/>

■ Módulo para la asignación de afectados.

Este módulo está compuesto por una interfaz llamada bandeja de incidentes, la cual se podrá seleccionar a través de la interfaz de menú principal de la plataforma.

A través de esta interfaz, el usuario puede verificar si la unidad de asistencia seleccionada ha experimentado algún incidente y tienen la capacidad de asignar o desasignar los afectados a dicha unidad. Además, una vez que se hayan asignado, el usuario podrá generar un nuevo informe a partir de los detalles del afectado.

■ Módulo para la integración con distintas fuentes de datos (anamnesis, desfibrilador, etc.).

Este módulo se utilizará para proporcionar información a las distintas pestañas del informe, cuando se necesite utilizar algún servicio de anamnesis, desfibrilador etc.

2.3. OBJETIVOS DOCENTES

En la presente sección, se describen una serie de objetivos relativos a la docencia recibida en el Grado en Ingeniería Informática, que serán considerados a lo largo del desarrollo del TFG:

- Empleo de una metodología de desarrollo en el ciclo de vida del software.
- Empleo de una adaptación del marco de trabajo *Scrum* para el desarrollo ágil de software.
- Empleo de técnicas de estimación, como *Planning Poker*, en la gestión de un proyecto software.
- Empleo de principios de usabilidad para el diseño de interfaces de aplicaciones.

CAPÍTULO 3

Estado del Arte

En este capítulo se detallará una investigación sobre la historia del proyecto y su aplicación base en la que está fundamentada el TFG. Se expondrá en qué consiste, así como sus módulos funcionales y adaptaciones que sirven de modelo para la realización de la nueva aplicación. Además se procederá a explicar y justificar el uso del patrón de arquitectura para aplicaciones móviles VIPER, que se va a utilizar en el desarrollo de este proyecto.

3.1. HISTORIA DEL PROYECTO HERA

El proyecto **HERA**, anteriormente conocido como *SURE* o *SUREM*, abarca un conjunto de aplicaciones desarrolladas para diversos clientes de *Inetum*, pero con un objetivo común: agilizar el triaje mediante una herramienta que optimice el proceso. Tradicionalmente, el triaje se lleva a cabo mediante un formulario en papel que los enfermeros de la ambulancia entregan al hospital. Por lo tanto, una herramienta que automatice este proceso ofrece a los profesionales sanitarios la oportunidad de optimizar su tiempo de manera más eficaz y eficiente, permitiéndoles dedicar más atención directa al paciente.

3.1.1. ECDE (Estació Clínica d'Emergències)

ECDE (Estació Clínica d'Emergències) fue la primera herramienta que *IECISA* (Informática el Corte Inglés S.A), posteriormente *Inetum*, desarrolló en el año 2017 para el Sistema de Emergencias Médicas (SEM) de Cataluña, con el objetivo de proporcionar una solución que agilizara el proceso de triaje en las ambulancias. Esta herramienta está diseñada específicamente para dispositivos *Android*¹ y, en la actualidad, *Inetum* continúa manteniéndola y desarrollándola.

3.1.1.1. Funcionalidades Principales

ECDE ha experimentado una evolución significativa desde su primera versión, adquiriendo diversas funcionalidades adicionales. Sin embargo, en su versión inicial, la herramienta contemplaba las siguientes características:

- **Acceso a la plataforma:** el usuario puede autenticarse en la plataforma tanto de forma online como offline. El proceso de inicio de sesión se lleva a cabo mediante el ingreso de un nombre de usuario y una clave de acceso, y se espera que esta tarea sea realizada por el usuario de mayor rango dentro de la unidad.
- **Asignación de unidad:** una vez autenticado, el usuario tendrá la capacidad de seleccionar una unidad específica a la cual se le asignará la tablet. Además, será necesario que el usuario ingrese la información de los profesionales asignados a dicha unidad durante cada turno, incluyendo médicos, enfermeros, entre otros.
- **Menú Principal:** una vez seleccionada la unidad, el usuario podrá gestionar los informes creados por el, podrá crear nuevos informes de SVA (Soporte Vital Avanzado) y SVB (Soporte Vital Básico) y también podrá acceder a la bandeja SITREM.
- **Bandeja SITREM (Sistema Integral de Tratamiento de Emergencias):** en esta funcionalidad el usuario podrá cargar desde SITREM los incidentes asignados a la unidad de la tablet permitiéndole asignar afectados a su unidad.

¹<https://www.android.com>

- **Generación de informes:** en esta pantalla, el usuario tendrá la capacidad de generar y modificar informes, permitiéndole introducir información en diversas pestañas:
 - **Pestaña de Cabecera:** en esta pestaña el usuario podrá introducir tanto información descriptiva del incidente y del equipo médico asignado como los datos identificativos del afectado. Asimismo, se permitirá introducir los datos de contacto de una persona de referencia relacionada con el paciente.
 - **Pestaña de Cargo a Terceros:** en la pestaña actual el usuario podrá introducir información acerca de la compañía aseguradora, número de póliza, adjuntar imágenes etc.
 - **Pestaña de Anamnesis:** en esta pestaña el usuario podrá introducir información sobre las alergias, antecedentes patológicos, hábitos tóxicos o información sobre su enfermedad actual. También podrá consultar el plan de medicación actual del paciente.
 - **Pestaña de Exploración Primaria:** dentro de esta pestaña el usuario podrá detallar información de una exploración primaria como puede ser el estado general del paciente, estado de la vía área y neurológico, ventilación, circulación o detallar información sobre las pupilas.
 - **Pestaña de Exploración Complementaria:** en esta sección el usuario podrá introducir información complementaria en el caso de que fuera necesario añadiendo un comentario o adjuntando imágenes.
 - **Pestaña de Exploración Exploración Física 1:** en esta pestaña el usuario podrá introducir información más detallada sobre el estado del aparato respiratorio y cardiovascular del paciente.
 - **Pestaña de Exploración Física 2:** dentro de esta pestaña el usuario podrá incluir información detallada sobre la neurología y el estado del abdomen del paciente.
 - **Pestaña de Exploración Física 3:** en la pestaña actual el usuario podrá introducir información sobre el estado del aparato locomotor, oído o boca del paciente.
 - **Pestaña de Monitorización Constantes:** en esta sección el usuario podrá introducir información tanto de las constantes vitales del paciente como de la monitorización cardiaca.
 - **Pestaña de Monitorización Constantes 2:** dentro de esta pestaña el usuario podrá añadir información adicional sobre las constantes del paciente.
 - **Pestaña de Escalas de Valor:** en esta sección el usuario podrá introducir observaciones que según unas escalas de valor de evaluación determinará el estado del paciente.
 - **Pestaña de Códigos de Pre-Activación:** en la pestaña actual el usuario podrá, según un tipo de código de emergencia como por ejemplo ICTUS, introducir información sobre esa condición médica grave.
 - **Pestaña de Procedimiento de Tratamiento:** dentro de esta pestaña el usuario podrá introducir información sobre el tratamiento del paciente para la ventilación, circulación, fluidoterapia etc. Además podrá detallar información sobre movilizaciones o evolución.
 - **Pestaña de Orientación Diagnóstica y Tratamiento:** en esta sección el usuario podrá introducir un plan terapéutico para el paciente así como una orientación del diagnóstico medicado.
 - **Pestaña de Valoración Enfermería:** en esta pestaña se detallará información de la valoración de enfermería al paciente como la oxigenación tisular, movilidad, homeostasis, nutrición etc.
 - **Pestaña de Diagnóstico Enfermería:** dentro de esta pestaña se podrá añadir información sobre el diagnóstico, resultado e intervención de la valoración de enfermería.
 - **Pestaña de Registro Ohscar:** en la pestaña actual el usuario podrá introducir información sobre el informe OHSCAR (Registro Español de Parada Cardiaca Extrahospitalaria) recogiendo datos de la parada cardiorrespiratoria y fluidoterapia del paciente.
 - **Pestaña de Tipos y Finalización de Servicio:** en esta sección el usuario podrá introducir información sobre el tipo de servicio brindado al paciente, finalización y efectos personales del paciente.
- **Subida de informes:** el usuario podrá enviar al SITREM tanto los datos de un informe completo, como parcial.

Es importante señalar que la mayoría de los datos en las figuras de las adaptaciones y del propio desarrollo han sido pixelados debido a la sensibilidad de la información y a la Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales².

²<https://www.boe.es/buscar/act.php?id=BOE-A-2018-16673>

3.1.2. Adaptaciones de ECDE

Debido al éxito que experimentó la herramienta ECDE, *Inetum* expandió su desarrollo para satisfacer las necesidades de más clientes. Desde ese momento, han surgido tres aplicaciones que se fundamentan en las funcionalidades principales establecidas por el proyecto ECDE:

- **SMS (Sistema Murciano de Salud):** este proyecto fue desarrollado en el año 2021 para el sistema sanitario de la Región de Murcia y está diseñado específicamente para la plataforma Android. Esta inspirada fielmente en las funcionalidades ofrecidas por ECDE debido a que fue construida para la misma plataforma. Entre sus funcionalidades principales destacan:
 - **Módulo de Acceso a la plataforma:** en la Figura 3.1, se muestra el módulo de acceso a la plataforma del proyecto SMS.
 - **Módulo Asignación unidad:** en la Figura 3.2, se muestra el módulo de asignación de unidad del proyecto SMS.
 - **Módulo de Menú Principal:** en la Figura 3.3, se muestra el módulo de menú principal del proyecto SMS.
 - **Módulo de Edición y Creación de informes:** en las Figuras 3.4 y 3.5, se muestran algunas pestañas del módulo de edición y creación de informes del proyecto SMS.
 - **Módulo de Bandeja de Incidentes:** en la Figura 3.6, se muestra el módulo de bandeja de incidentes del proyecto SMS.

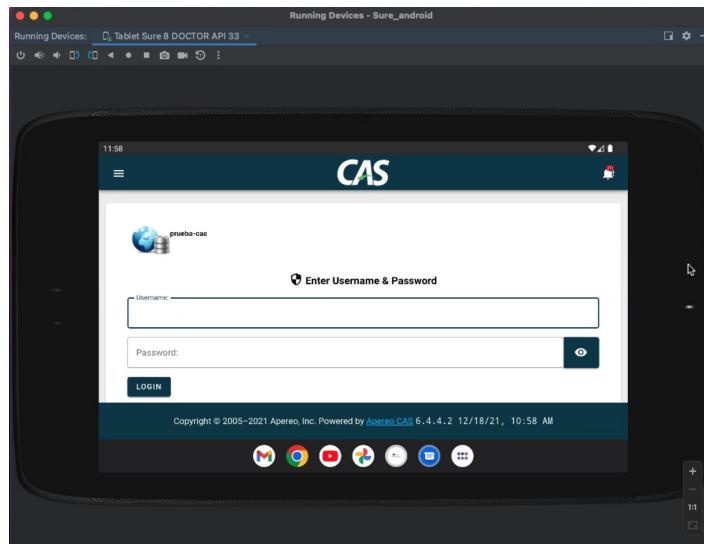


Figura 3.1: Captura de login en proyecto SMS.

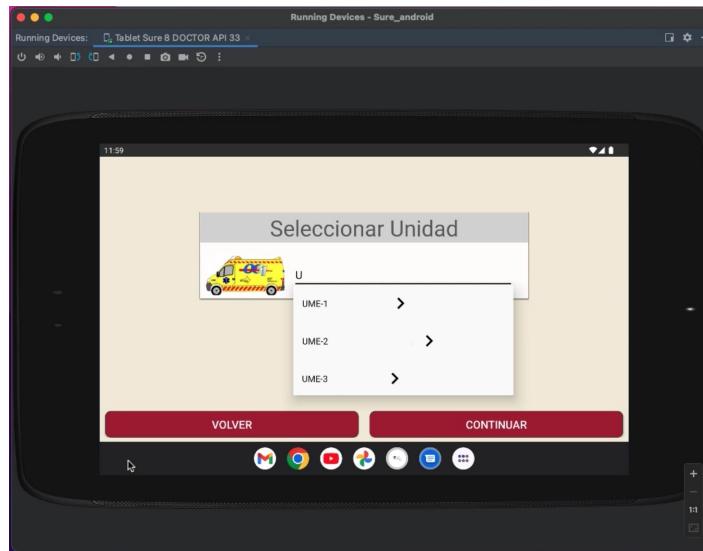


Figura 3.2: Captura de asignación de unidad en proyecto SMS.

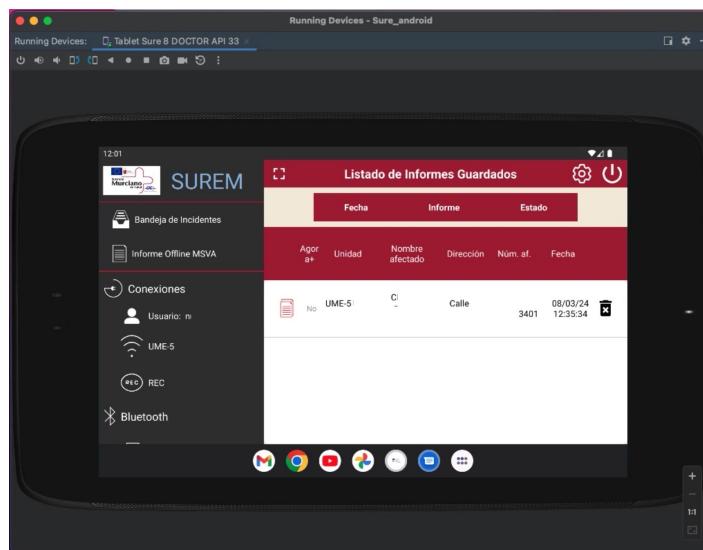


Figura 3.3: Captura de menú principal en proyecto SMS.

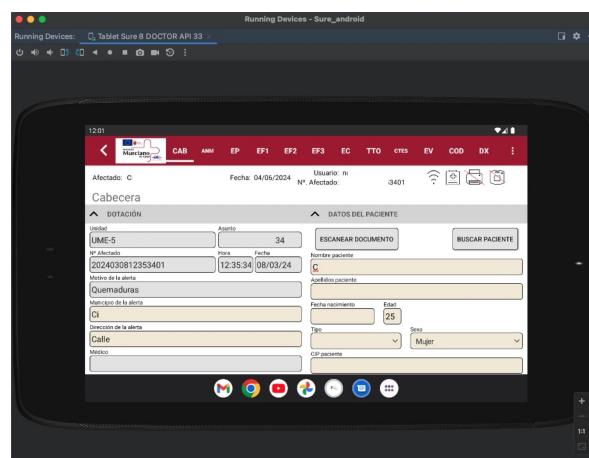


Figura 3.4: Captura de la pestaña Cabecera del módulo de creación y edición de informes en proyecto SMS.

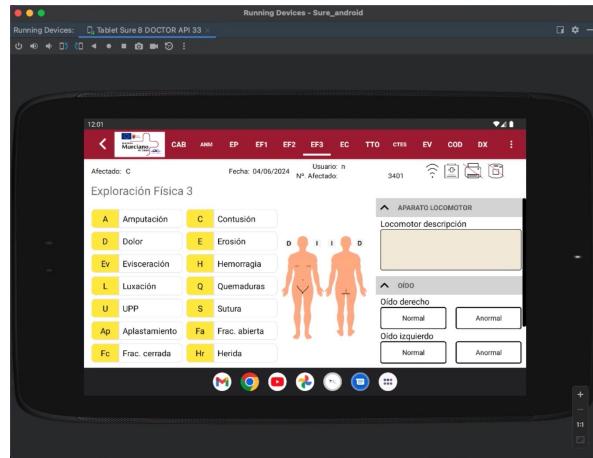


Figura 3.5: Captura de la pestaña Exploración física III del módulo de creación y edición de informes en proyecto SMS.

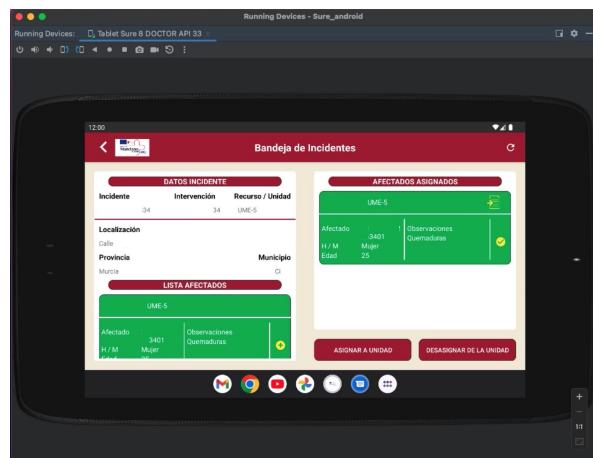


Figura 3.6: Captura de bandeja de incidentes en proyecto SMS.

- **Sacyl (Salud de Castilla y León):** este proyecto fue desarrollado en el año 2022 para la sanidad de Castilla y León. Aunque se inspira en las funcionalidades de ECDE, constituyó un proyecto completamente nuevo, ya que fue diseñado específicamente en IONIC³ para Windows⁴. Entre sus funcionalidades principales destacan:

- **Módulo de Acceso a la plataforma:** en la Figura 3.7, se muestra el módulo de acceso a la plataforma del proyecto Sacyl.
- **Módulo de Menú Principal:** en la Figura 3.8, se muestra el módulo de acceso a la plataforma del proyecto Sacyl.
- **Módulo de Edición y Creación de informes:** en las Figuras 3.9 y 3.10, se muestran algunas pestañas del módulo de edición y creación de informes del proyecto SMS.
- **Módulo de Bandeja de Incidentes:** en la Figura 3.11, se muestra el módulo de acceso a la plataforma del proyecto Sacyl.

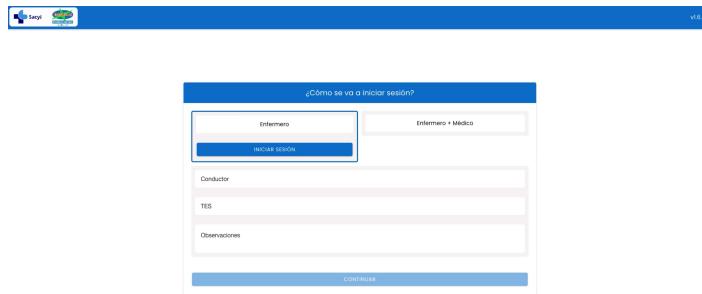


Figura 3.7: Captura de login en proyecto Sacyl.

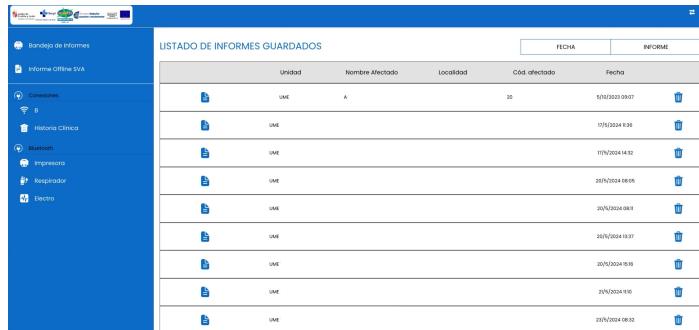


Figura 3.8: Captura de menú principal en proyecto Sacyl.

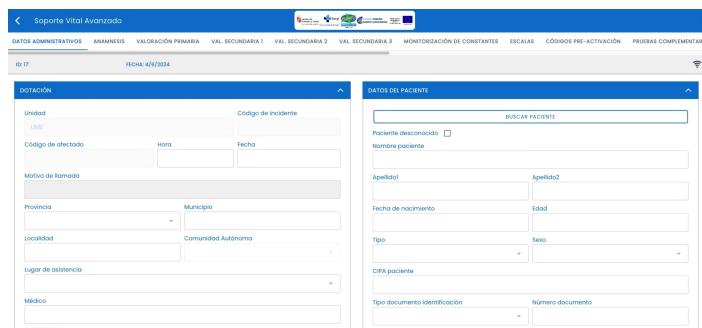


Figura 3.9: Captura de la pestaña Datos administrativos del módulo de creación y edición de informes en proyecto Sacyl.

³<https://ionicframework.com/>

⁴<https://www.microsoft.com/es-es/windows?r=1>

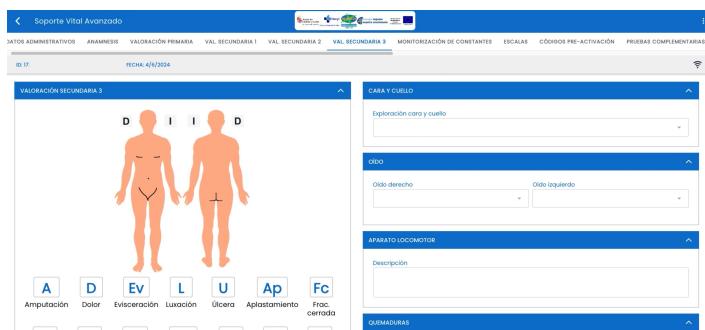


Figura 3.10: Captura de la pestaña Valoración secundaria III del módulo de creación y edición de informes en proyecto Sacyl.

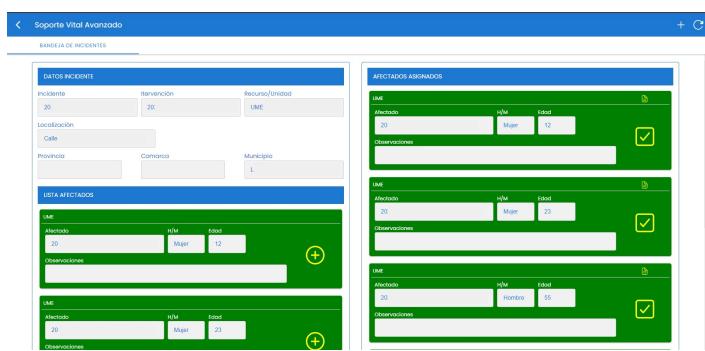


Figura 3.11: Captura de bandeja de incidentes en proyecto Sacyl.

- **HERA iOS:** nuevo proyecto que constituye este TFG. Surge de la necesidad de realizar una aplicación de carácter general desarrollada para la plataforma iOS. Esta herramienta se inspira en el proyecto SMS y su propósito principal es ser comercializada entre aquellas comunidades autónomas interesadas en mejorar la eficiencia de sus procesos de triaje.

3.2. PATRÓN DE ARQUITECTURA VIPER

En el contexto actual del desarrollo *software*, caracterizado por la creciente demanda en aplicaciones móviles altamente funcionales y escalables, el uso de un patrón arquitectónico se vuelve un factor clave debido a la complejidad inherente de este tipo de aplicaciones.

Estos patrones proporcionan un marco estructurado y validado de directrices que simplifican la modularización del código, lo que resulta en una gestión de la complejidad más eficiente. Siguiendo estas directrices, los desarrolladores pueden organizar su código de una forma más estructurada, lo que contribuye a la reducción de errores y a un mantenimiento sostenible en la evolución del proyecto a lo largo del tiempo.

VIPER⁵, cuyo nombre deriva del acrónimo *View*, *Interactor*, *Presenter*, *Entity* y *Router*, es un patrón de arquitectura ampliamente utilizado para el desarrollo de aplicaciones en entornos con arquitecturas de código limpio (*Clean Architecture*⁶), especialmente en plataformas como iOS. Se fundamenta en los principios de diseño **SOLID**, los cuales constituyen un conjunto de directrices reconocidas en el ámbito del desarrollo *software*. Estos principios están diseñados para facilitar al programador la creación de *software* más evolucionable y mantenable. Entre estos principios se encuentran:

- **Principio de Responsabilidad Única (*Single Responsibility Principle*):** cada clase o componente del sistema debe de ser diseñado para tener una única responsabilidad y en consecuencia, solo tendrá que haber una única razón en caso de cambio. En VIPER, esta premisa nos indica que cada componente como la *View*, el *Presenter*, entre otros, debe de tener una única responsabilidad bien definida y clara.
- **Principio Abierto-Cerrado (*Open-Closed Principle*):** toda entidad *software* debe de estar abierta a extensión y cerrada a modificación. Dentro del contexto de VIPER, este principio se manifiesta en la

⁵<https://www.objc.io/issues/13-architecture/viper/>

⁶Martin, R. C. (2017). Clean architecture.

capacidad de extender o añadir nuevas funcionalidades en cada componente del patrón sin tener que alterar código en otras capas existentes.

- **Principio Sustitución de Liskov (*Liskov Substitution Principle*):** establece que las clases derivadas deben de ser intercambiables por sus clases bases sin ocasionar alteraciones en el comportamiento del programa. En *VIPER*, se refleja en la capacidad de intercambiar implementaciones particulares en cada componente sin alterar el flujo general.
- **Principio de Segregación de interfaz (*Interface Segregation Principle*):** establece que ninguna clase o módulo debería ser obligado a depender de métodos que no utiliza. Dentro del contexto de *VIPER*, este principio se manifiesta en que cada componente tenga su propia interfaz específica y clara, diseñada para satisfacer las necesidades específicas del mismo.
- **Principio de Inversión de dependencia (*Dependency Inversion Principle*):** las dependencias deben depender de abstracciones, no de implementaciones concretas. En *VIPER*, este principio se consigue a través de interfaces específicas entre los componentes. Estas interfaces sirven como abstracciones que especifican la forma en que los componentes interactúan entre sí, en lugar de establecer una dependencia directa de las implementaciones concretas de dichos componentes.

Es crucial resaltar que el pilar esencial sobre el cual *VIPER* se fundamenta es la clara separación de responsabilidades, en la cual cada componente desempeña un rol específico y bien definido dentro del sistema. Esta organización garantiza que cada parte del sistema esté dedicada a una tarea particular, lo que facilita la comprensión y el mantenimiento del código a lo largo del ciclo de vida del proyecto. La Figura 3.12 muestra la arquitectura de *VIPER*.

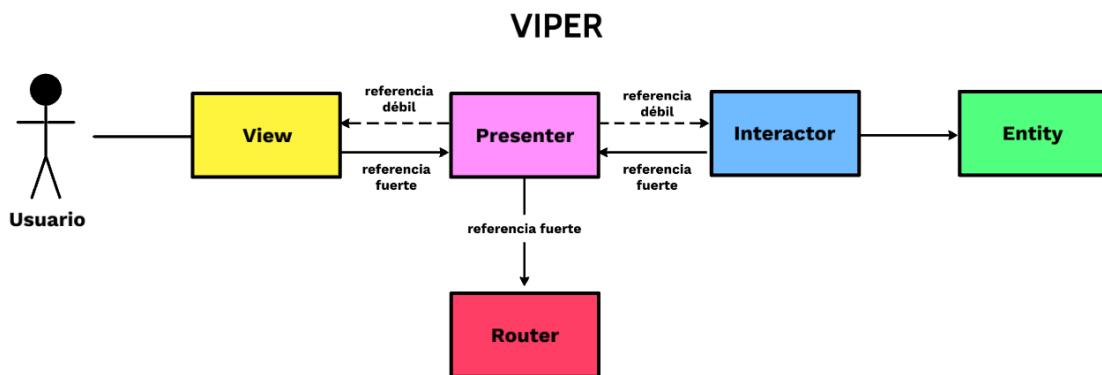


Figura 3.12: Arquitectura de *VIPER*.

3.2.1. Componentes

En *VIPER*, se emplean los siguientes componentes:

- **View:** constituye la capa responsable de la presentación de la interfaz de usuario. Su función principal es recibir la información del *Presenter* para mostrársela al usuario. Además, será capaz de responder a cada interacción del usuario con la interfaz. En caso de que se requiera algún tipo de procesamiento, delegará esta acción al *Presenter* y esperará una respuesta que indique qué información debe mostrar al usuario. En consecuencia, podemos inferir que el papel de la vista es pasivo, ya que no inicia ninguna acción por sí sola, sino que responde a las interacciones del usuario y las indicaciones proporcionadas por el *Presenter*.
- **Interactor:** gestiona toda la lógica de negocio relacionada con las entidades de manera independiente a la interfaz de usuario. Su función principal es proporcionar la información solicitada por el *Presenter*. Para ello, interactúa con *Entities* que representan modelos de datos provenientes de diversas fuentes, como bases de datos locales, servicios web u otras fuentes de datos. Una vez obtiene estos datos, los procesa según la lógica de negocio establecida y los comunica al *Presenter* para su presentación en la interfaz de usuario.
- **Presenter:** esta capa contiene la lógica de la vista. Su principal función es recoger los datos suministrados por el *Interactor* y prepararlos para su visualización en la vista. Además, administra las interacciones del usuario y efectúa las actualizaciones necesarias en la *View*.

- **Entity:** representa los modelos de datos de la aplicación que serán manejados por el *Interactor*.
- **Router:** esta capa gestiona la navegación entre las pantallas de la aplicación. Su función principal es la gestión del flujo y transmisión de datos entre módulos. Es responsable tanto de la creación como del inicio del módulo de la vista al que se desea navegar.

La Figura 3.13 muestra los flujos de comunicación entre los componentes.

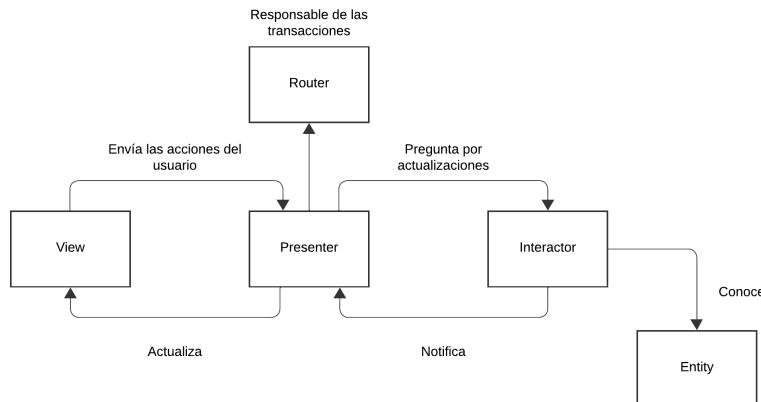


Figura 3.13: Flujos de comunicación entre los componentes.

Fuente: traducción de [VIPER. Design pattern's modules interaction description](#).

Cada uno de los componentes está orientado a protocolos, por lo que para comunicarse entre sí deben de seguir un conjunto estricto de reglas. Por lo tanto, el diagrama de clases⁷ de un módulo VIPER podría definirse de la siguiente manera (véase Figura 3.14):

- El componente *View* tendría una relación de *uno a uno* con el *Presenter*. Así logramos mantener una única referencia del *Presenter* para poder comunicarle los acciones del usuario.
- El componente *Presenter* tendría tres referencias de relación *uno a uno*. La primera sería hacia la *View*, con el fin de poder actualizarla. La segunda sería hacia el *Interactor*, de manera que, en caso de necesitar actualizar o solicitar información para la *View*, se pudiera realizar esta solicitud. La tercera referencia sería hacia el *Router*, para los casos en los que se requiriera navegar a otra ventana de un módulo diferente.
- El componente *Interactor* tendría una relación *uno a uno* con el *Presenter*, permitiéndole enviar la información solicitada. Además, este componente tendría una relación de *muchos a muchos* con los *Entity* debido a que un *Interactor* puede conocer varios *Entities* y un *Entity* puede ser conocido en varios módulos de VIPER.
- El componente *Router*, además de la relación *uno a uno* con el *Presenter* mencionada anteriormente, mantiene una relación de *ninguno a muchos* con otros componentes *Router* de distintos módulos, con el propósito de poder realizar la navegación.

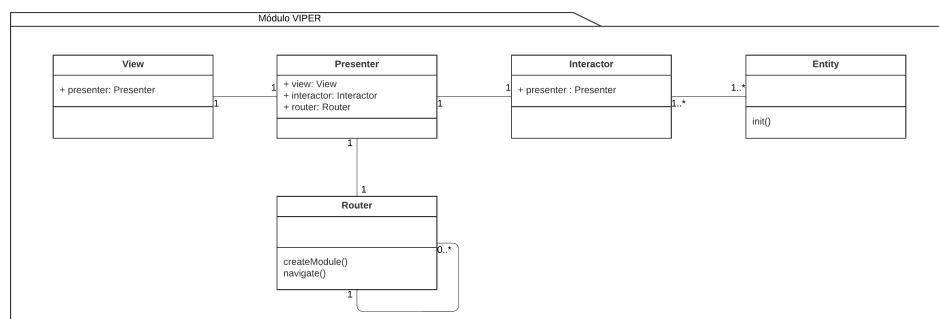


Figura 3.14: Diagrama de clases Módulo VIPER.

⁷Rumbaugh, J., Jacobson, I., & Booch, G. (1996). The unified modeling language. University Video Communications.

3.2.2. Ventajas e Inconvenientes

Algunas de las ventajas de utilizar el patrón arquitectónico *VIPER* en un proyecto software son:

- **Simplificación de proyectos complejos:** los módulos *VIPER* independientes permiten una gestión eficiente de grandes equipos de desarrollo.
- **Incremento en la escalabilidad:** facilita el trabajo simultáneo de desarrolladores sin conflictos.
- **Desacoplamiento del código:** permite la reutilización y pruebas más efectivas del código.
- **Separación clara de responsabilidades:** divide los componentes según su rol, facilitando la adición de nuevas características y mejorando la organización.
- **Facilidad para pruebas automatizadas:** la separación de la lógica de interfaz de usuario y de negocio simplifica la implementación de tests automáticos y TDD (Test Driven Development).
- **Interfaces claras y bien definidas:** al ser independientes de otros módulos facilitan cambios y mejoras.
- **Código más limpio y reutilizable:** hace el código más claro, compacto y consistente al aplicar los principios SOLID.

El principal inconveniente de este patrón arquitectónico es la sobrecarga asociada a la creación de cinco componentes y protocolos para cada módulo. Esto puede resultar en una considerable cantidad de código repetitivo y estándar que puede hacer que la base de código sea más extensa y menos legible sin agregar valor significativo.

CAPÍTULO 4

Metodología

En este capítulo se introducirá la metodología seleccionada por *Inetum*. Se presentará el marco de trabajo *Scrum* explicando sus características fundamentales así como los roles que lo forman, eventos que se realizan, artefactos producidos y la técnica de estimación de historias de usuario *Planning Poker*. Además, se describirán las adaptaciones de *Scrum* que se han realizado. Por último, se detallará el marco tecnológico utilizado para el desarrollo de este TFG.

4.1. METODOLOGÍA DE TRABAJO EN INETUM

La empresa *Inetum* está caracterizada por contar con diversos espacios de desarrollo de software, conocidos como factorías. Una factoría, es un centro de software en el cual los requisitos nos los da el cliente, ahorrando así dicho proceso de obtención de requisitos, negociación con el cliente, etc. Estas factorías están certificadas con el nivel 5 de madurez del CMMI¹ (Capability Maturity Model Integration), lo cual garantiza la aplicación de las mejores prácticas de ingeniería del software en el desarrollo de sus proyectos. Esta certificación asegura que se siguen procesos bien estructurados, documentados y medidos, permitiendo así su mejora continua a lo largo del tiempo.

Este Trabajo de Fin de Grado ha sido realizado en la factoría Espacio Calatrava de Miguelturra (Ciudad Real) por el equipo de Movilidad. El proyecto fue desarrollado por un único miembro, bajo la supervisión de un tutor y la dirección de un líder técnico, todos pertenecientes al equipo de Movilidad.

Para el desarrollo del mismo se seleccionó como metodología de desarrollo ágil de software una **metodología iterativa e incremental**. Esta metodología de desarrollo, junto con el marco de trabajo *Scrum* para la gestión de proyectos y el modelo de madurez CMMI, garantizan entorno de desarrollo robusto y adaptable que maximiza la eficiencia, la calidad y la satisfacción del cliente en el desarrollo de software.

En el ciclo de vida ágil de los proyectos llevados a cabo en las factorías en *Inetum* se pueden diferenciar varias fases:

1. **Fase de Preventa:** periodo que abarca desde que se identifica una oportunidad en el cliente hasta la presentación de la oferta.
2. **Fase de Inception:** esta fase abarca desde la confirmación de la adjudicación del proyecto hasta el comienzo de la primera iteración de desarrollo. Durante esta fase, se toman en cuenta los requisitos funcionales y no funcionales, así como las especificaciones técnicas y funcionales del proyecto. A partir de ellos, se elabora el Plan de Ejecución, el Plan de Calidad y la Especificación Funcional final del proyecto.
3. **Fase de Ejecución:** periodo que abarca desde el arranque de la primera iteración de desarrollo hasta la aceptación del proyecto por parte del cliente. Durante esta fase se realizan varias etapas:
 - **Análisis:** el propósito de esta etapa es comprender las historias de usuario del *Product Backlog* priorizado para desarrollar un MVP (Producto Mínimo Viable) y aclarar cualquier duda. Al inicio de cada *sprint*, se realizará una reunión de *Sprint Planning* para cumplir con los requisitos de esta etapa.
 - **Desarrollo:** dentro de esta etapa se identifican tres fases:

¹<https://cmmiinstitute.com/>

- Diseño: en esta fase, primero se realizará la preparación del entorno de desarrollo en la Iteración 0, considerando los estándares pertinentes. Posteriormente, se revisará el diseño de alto nivel basado en la solución técnica propuesta en la fase de *Inception*. A continuación, se llevará a cabo el diseño de interfaces según las historias de usuario priorizadas en el *Backlog*, utilizando el documento de diseño técnico disponible. El diseño de bajo nivel se realizará únicamente si se especifica en el Acuerdo de Ejecución del proyecto.
 - Codificación: en esta fase, se desarrollan los componentes del sistema o interfaces, basándose en el diseño del sistema. A continuación, se llevan a cabo las pruebas unitarias, incluyendo la codificación de las pruebas aplicables y la definición de umbrales de cobertura. Se realiza el análisis del código desarrollado, corrigiendo los defectos detectados. Posteriormente, se determina la secuencia de integración entre componentes, teniendo en cuenta el desarrollo previo de los mismos. Se elabora la documentación del usuario final, seguida de una revisión de la misma. Finalmente, se establece la trazabilidad de casos/interfaces con componentes, asegurando la relación entre los casos de uso en Jira y los componentes desarrollados.
 - Pruebas: el propósito de esta fase es realizar pruebas exploratorias por el equipo, con el apoyo de testers integrados. En consecuencia, se inicia con la ejecución de pruebas exploratorias, donde se realizan las pruebas y se corrigen los defectos encontrados, seguido de la preparación del entorno de pruebas funcionales para integrar componentes y ejecutar pruebas. Posteriormente, se diseña y ejecuta las pruebas funcionales, corrigiendo los defectos detectados y preparando la entrega del sistema al cliente. Finalmente, se realizan correcciones de defectos asociados a la entrega, asegurando su registro en Jira para un seguimiento adecuado.
 - **Seguimiento:** en esta etapa se inicia con la ejecución de tareas planificadas, donde se completan las tareas asignadas según lo previsto, registrando el esfuerzo en Jira. Además de la reunión diaria (*Daily Scrum*), se llevan a cabo reuniones adicionales: *Sprint Planning* al inicio de cada *sprint*, *Sprint Review* al finalizar cada *sprint*, y *Sprint Retrospective* al concluir cada *sprint*, en conjunto con la *Sprint Review*.
 - **Cambios:** el proceso de gestión de cambio se sigue de forma estándar. Si el cambio es aceptado, se planificará como un paquete de cambio de alcance adicional en el *Backlog*, que será priorizado por el *Product Owner*.
 - **Mejora Continua:** en esta etapa se sigue un proceso para la recopilación de lecciones aprendidas y la comunicación de mejoras en el proyecto. Una vez informadas, se destaca cómo se resolvieron satisfactoriamente cuestiones técnicas que podrían repetirse, y se publican artículos al respecto en Confluence.
 - **Incidencias:** la resolución de una incidencia sigue la siguiente secuencia: primero, se realiza todo lo necesario para corregirla, sin descomponerla en subtareas en Jira. Posteriormente, se controlan los cambios y versiones de los elementos modificados asociados a la incidencia. Las tareas pueden abarcar construcción, pruebas, documentación y generación de la nueva construcción. Finalmente, una vez cerrada la incidencia, se informa en Jira del esfuerzo real empleado en la corrección.
4. **Fase de Mantenimiento:** esta fase abarca desde la aceptación del cliente y periodo de garantía acordado en el contrato con el cliente para el mantenimiento correctivo. Opcionalmente esta fase podrá ampliarse mediante un servicio de mantenimiento correctivo, evolutivo y de soporte adicional.

En la Figura 4.1, se muestran las fases del ciclo de vida de un proyecto en *Inetum*.

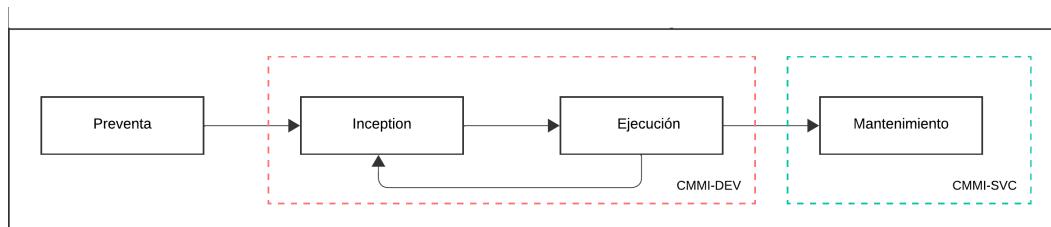


Figura 4.1: Fases del ciclo de vida de un proyecto en *Inetum*.

4.2. SCRUM

Scrum² es un marco de trabajo (*framework*) para la gestión de proyectos ágiles que facilita a los equipos la organización de su trabajo a través de un conjunto de buenas prácticas, principios y valores.

Este *framework* es ampliamente utilizado debido a su simplicidad y, especialmente, por su capacidad para promover la colaboración, autoorganización, la mejora continua y la adaptación en los equipos. Estas características se logran gracias a su enfoque iterativo e incremental, que enfatiza la entrega de valor de forma rápida y constante.

4.2.1. Características fundamentales

De acuerdo con la **Guía oficial de Scrum**³, este marco de trabajo se fundamenta en el empirismo, donde las decisiones se basan en la observación, experiencia y experimentación. Para ello se apoya en tres pilares fundamentales:

- **Transparencia:** cualquier aspecto relevante en el proceso debe de ser visible para los responsables del resultado.
- **Inspección:** se deben de realizar frecuentemente inspecciones de los artefactos de *Scrum* y del progreso hacia el objetivo.
- **Adaptación:** en el momento en el que se detecten desviaciones, el equipo se tiene que adaptar para realizar ajustes en el proceso.

Como se puede observar en la Figura 4.2, la base de *Scrum* es la confianza. En un equipo *Scrum*, la ausencia de confianza puede generar un ambiente de tensión que puede resultar en un entorno de trabajo menos productivo y colaborativo. Para impedir que suceda esto, *Scrum* está guiado por un conjunto de valores, los cuales son:

- **Coraje:** para enfrentar desafíos y tomar decisiones difíciles.
- **Enfoque:** priorizar el trabajo del Sprint y los objetivos del equipo.
- **Compromiso:** al éxito del equipo y del proyecto.
- **Respeto:** hacia todos los miembros del equipo.
- **Apertura:** al trabajo y a los retos que se planteen durante su desarrollo.

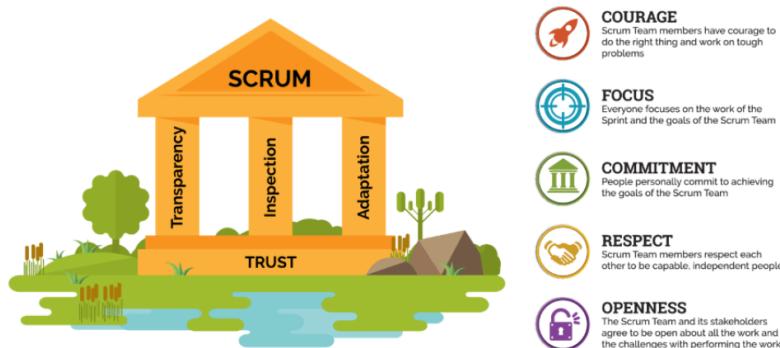


Figura 4.2: Pilares y Valores de Scrum. Fuente: [Scrum.org](https://scrum.org).

²<https://www.scrum.org/>

³<https://scrumguides.org/download.html>

4.2.2. Roles

Scrum establece roles específicos que son clave para el éxito del proyecto. Este conjunto de roles constituye el **equipo Scrum**, el cual está diseñado para cumplir una serie de características fundamentales:

1. **Auto-organizados:** capacidad inherente que debe de tener el equipo *Scrum* para tomar decisiones sobre cómo llevar a cabo su trabajo de manera autónoma y efectiva. Esta autonomía promueve un entorno colaborativo, donde cada miembro del equipo puede usar sus habilidades al máximo y adaptarse rápidamente a los desafíos del proyecto.
2. **Multifuncionales:** cada uno de los miembros del equipo *Scrum* debe de poseer una variedad de habilidades necesarias para completar todas las tareas requeridas para entregar un incremento de producto. La multifuncionalidad promueve la flexibilidad dentro del equipo, lo que posibilita una respuesta a cambios más ágil.
3. **Colaboración:** característica fundamental para el éxito de *Scrum*, ya que promueve la transparencia, la comunicación abierta y el intercambio de conocimiento entre los miembros sin importar su rol o puesto dentro de la organización.

Un equipo *Scrum* debe de ser lo suficientemente compacto para mantener su agilidad, pero al mismo tiempo capaz de abordar un trabajo significativo en un *Sprint*, generalmente con diez personas o menos. Un equipo de estas dimensiones será más productivo y facilitará una mejor comunicación. Su objetivo fundamental es crear un incremento útil y valioso en cada sprint.

Está conformado por un *Scrum Master*, un *Product Owner* y desarrolladores, todos ellos centrados en un único objetivo: el *Product goal*. En consecuencia, dentro de un equipo *Scrum* se identifican tres roles (ver Figura 4.3):

Equipo Scrum

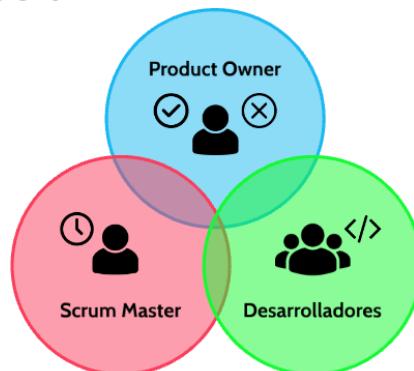


Figura 4.3: Roles de *Scrum*.

- **Propietario del producto (*Product Owner*):** persona encargada de identificar, medir y maximizar el valor del producto resultante del trabajo del equipo *Scrum* a lo largo de todo su ciclo de vida. Además, se encarga de gestionar la Pila del producto o *Product Backlog*.
- **Desarrolladores:** conjunto de personas encargadas de desarrollar el producto, es decir, generar un incremento terminado (*Done*) en cada *Sprint* a partir de los elementos definidos del *Product Backlog*.
- **Scrum Master:** persona encargada de la gestión de *Scrum*, proporcionando orientación tanto teórica como práctica del *framework*, tanto dentro del equipo *Scrum* como en la organización.

En la Tabla 4.1, se muestra un resumen de las responsabilidades llevadas a cabo por cada rol.

Rol	Responsabilidades
Product Owner	<ul style="list-style-type: none"> - Desarrollar y comunicar explícitamente el objetivo del producto. - Establecer, gestionar y ordenar los elementos de la Pila del producto. - Asegurar que el equipo de desarrollo entiende los elementos de la pila del producto. - Comunicar y ser intermediario entre las necesidades del cliente y el equipo <i>Scrum</i>. - Máximo responsable del valor entregado.
Desarrolladores	<ul style="list-style-type: none"> - Planear el sprint mediante el desarrollo de la Pila del sprint (<i>Sprint Backlog</i>). - Adaptar su plan de cada día al objetivo del sprint. - Compromiso con los estándares de calidad acordados en la definición de terminado (<i>Definition of Done</i>).
Scrum Master	<ul style="list-style-type: none"> - Facilitar y fomentar la adopción y práctica de los principios y valores ágiles en la cultura de la organización mediante la introducción de <i>Scrum</i>. - Formar y ayudar al equipo <i>Scrum</i> promoviendo la colaboración, la transparencia y el enfoque en la entrega de valor. - Garantizar el cumplimiento de roles y formas de modelo de <i>Scrum</i>. - Facilitar los eventos de <i>Scrum</i> y eliminar cualquier impedimento que afecte al valor del producto.

Tabla 4.1: Responsabilidades de cada rol dentro de *Scrum*.

4.2.3. Artefactos

Son elementos tangibles que se crean, mantienen o utilizan durante el proceso de desarrollo del producto que representan trabajo o valor. Además, proporcionan transparencia y oportunidades de inspección y adaptación, lo que ayuda al equipo *Scrum* a comprender el progreso del trabajo hacia los objetivos del proyecto.

En consecuencia, cada artefacto se compromete a proporcionar información que mejore la transparencia y el enfoque, siendo estos los puntos de referencia mediante los cuales se puede medir el progreso:

- Para la Pila del producto (*Product Backlog*), es el objetivo del producto (*Product Goal*).
- Para la Pila del sprint (*Sprint Backlog*), es el objetivo del sprint.
- Para el Incremento, es la Definición de Hecho (*Definition of Done*).

A continuación, se detalla la función y el propósito de cada uno de estos artefactos (ver Figura 4.4):

- **Pila del Producto (*Product Backlog*)**: es una lista dinámica y priorizada de requisitos (historias de usuario) que a partir de una visión inicial crece y evoluciona durante el desarrollo. Este artefacto se inicia con las aportaciones del cliente, reflejando sus necesidades y requisitos iniciales para el producto. Posteriormente, el *Product Owner* asume la responsabilidad de gestionarla priorizando y refinando los elementos en función del valor que aportan al producto y las necesidades del cliente.
- **Pila del Sprint (*Sprint Backlog*)**: es un subconjunto del artefacto anterior, es decir, es una lista para el equipo de desarrollo con un conjunto de tareas del Product Backlog para desarrollarlas en el siguiente incremento del producto.
- **Incremento**: es el resultado final que aporta valor al producto tras la realización de los elementos del *Product Backlog* completados en el *Sprint* actual, sumado al valor de los incrementos de todos los *Sprints* anteriores.

En consecuencia, al finalizar un *Sprint*, un nuevo incremento debe estar completado conforme al *Definition of Done* establecida por el equipo *Scrum*. La Definición de Hecho (*Definition of Done*) es un



Figura 4.4: Artefactos de *Scrum*.

documento establecido por el equipo *Scrum* que detalla los criterios necesarios para considerar que un elemento está terminado.

4.2.4. Eventos

Son períodos de tiempo con una duración máxima, cuyo objetivo es garantizar transparencia y regularidad en el proceso. Los eventos se emplean tanto para la inspección como para la adaptación de los artefactos, evitando la convocatoria de reuniones innecesarias que podrían carecer de utilidad. *Scrum* presenta los siguientes eventos (ver Figura 4.5):

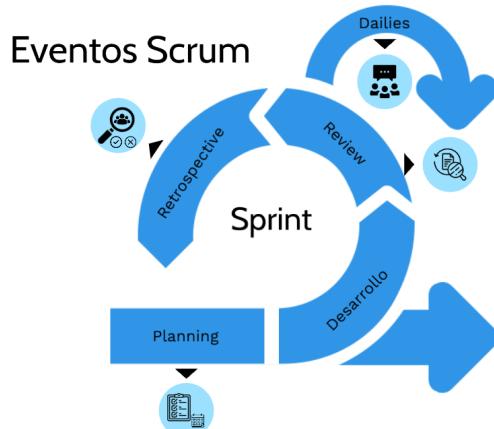


Figura 4.5: Eventos de *Scrum*.

- **Sprint:** es una iteración, generalmente comprendiendo un periodo de tiempo de entre una y cuatro semanas, dedicada al desarrollo de un Incremento del producto. Este Incremento se compone de un conjunto de elementos seleccionados por el equipo Scrum del *Product Backlog*, conformando el *Sprint Backlog*. El objetivo principal será generar una versión entregable del producto que incremente su valor y que sea funcional.

En consecuencia, todas las tareas necesarias para alcanzar el objetivo del producto, incluidas la Planificación del Sprint (*Sprint Planning*), reuniones diarias (*Dailies*), la Revisión del Sprint (*Sprint Review*) y la Retrospectiva del Sprint (*Sprint Retrospective*), se llevan a cabo durante un *Sprint*.

- **Planificación del Sprint (*Sprint Planning*):** es una reunión que se lleva a cabo al comienzo de cada *sprint*, cuyo propósito es definir su objetivo y qué elementos del *Product Backlog* se van a abordar. Su duración como máximo son ocho horas y participa todo el equipo *Scrum*. Comprende tres fases:

1. **¿Por qué?**: durante esta fase se define el objetivo del sprint, es decir, cuál es el valor que se le entrega al cliente. Además, el Product Owner propone cómo incrementar el valor del producto durante el sprint actual.
2. **¿Qué?**: se determinan qué elementos del *Product Backlog* se van a abordar durante el sprint.

3. **¿Cómo?:** en esta fase se planifica el trabajo necesario para generar a un incremento según la DoD (*Definition of Done*). Para ello, se descomponen los elementos seleccionados del *Product Backlog* en tareas más pequeñas dando lugar al *Sprint Backlog*.
- **Reunión diaria (*Daily Scrum*):** es una reunión diaria de quince minutos en la que participan el *Scrum Master* y el equipo de desarrollo. Su objetivo principal es evaluar el progreso hacia el objetivo del *sprint*. Los miembros pueden exponer los avances del día anterior, sus tareas para el día hoy y los impedimentos encontrados.
 - **Revisión del Sprint (*Sprint Review*):** es una reunión en la que participa todo el equipo *Scrum* y se lleva a cabo al final del *sprint*. Su propósito es analizar y revisar el incremento realizado. Basándose en los elementos completados y los pendientes, el *Product Owner* ajusta el *Product Backlog* para la siguiente iteración.
 - **Retrospectiva del Sprint (*Sprint Retrospective*):** es una reunión realizada al final del *sprint*, donde los miembros del equipo *Scrum* reflejan sus impresiones sobre el *sprint* que acaba de terminar. Su objetivo principal es promover la mejora continua del equipo y del producto.

4.2.5. Estimaciones - *Planning Poker*

Un aspecto crucial durante la planificación del *sprint* es la estimación del esfuerzo necesario para completar las historias de usuario. Una estimación precisa ayuda al equipo a gestionar adecuadamente el desarrollo, garantizando una entrega puntual.

Con este propósito, existen diversas técnicas de estimación, entre las cuales destaca popularmente la técnica conocida como *Planning Poker*.

Planning Poker es una técnica de estimación de desarrollo software empleada en metodologías ágiles, la cual se fundamenta en la estimación por consenso del esfuerzo o tamaño relativo de las tareas a desarrollar. Esta técnica es ampliamente utilizada debido a su bajo coste, sencillez y a la combinación de la opinión experta, analogía y desviación del esfuerzo.

Cada estimador va a disponer de un mazo de cartas numeradas (normalmente utilizando la serie de Fibonacci) para estimar las historias de usuario según su juicio. El procedimiento consta de cuatro fases (ver Figura 4.6):

1. **Presentación de requisitos:** en esta etapa se presentan las historias de usuario que se van a estimar, proporcionando una descripción detallada de cada una de ellas.
2. **Elección de la carta (estimación):** en esta fase cada estimador escoge una carta del mazo numerado que representa su estimación del trabajo para desarrollar la historia de usuario en cuestión.
3. **Publicación:** en esta etapa los estimadores muestran a la vez su carta seleccionada. La publicación de la carta de los integrantes del equipo debe de suceder al mismo tiempo para evitar plagios o influencias en la estimación.
4. **Consenso:** en esta fase se realiza un acuerdo entre todas las estimaciones de los integrantes del equipo. Si existe una gran dispersión entre ellas se vuelve a repetir el proceso.

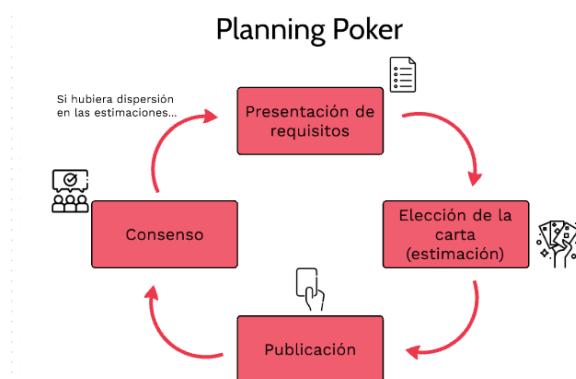


Figura 4.6: Procedimiento *Planning Poker*.

4.3. ADAPTACIONES DE SCRUM PARA EL DESARROLLO DE ESTE TFG

Para el desarrollo de este TFG, como se ha mencionado en la sección 4.1, se ha utilizado una metodología iterativa-incremental. Gracias a la adaptación del marco de trabajo *Scrum*, este enfoque ha resultado muy práctico para llevar a cabo el desarrollo completo del proyecto. Se han llevado a cabo dos adaptaciones:

- En primer lugar, hay una adaptación en cuanto al tamaño del equipo. En este caso, el equipo de desarrollo ha estado formado por un único integrante, un *Scrum Master* y un *Product Owner*. El rol de *Product Owner* ha sido asumido por el líder técnico del equipo de Movilidad, mientras que el tutor, por parte de la empresa, del alumno ha desempeñado el papel de *Scrum Master*. Esto constituye una adaptación, ya que *Scrum* establece que el número de desarrolladores debe estar entre tres y nueve personas.
- En segundo lugar, se ha realizado una adaptación en cuanto a los eventos. Debido a limitaciones de tiempo de los integrantes del equipo, se decidió omitir la *Sprint Review* y, en su lugar, combinar sus objetivos con los de la reunión de *Sprint Retrospective* añadiendo una conclusión. Esto constituye una adaptación, ya que *Scrum* establece como eventos: *Sprint*, *Sprint Planning*, *Daily*, *Sprint Review* y *Sprint Retrospective*.

4.4. TECNOLOGÍAS SOFTWARE Y HARDWARE UTILIZADAS

En esta sección se detallará el marco tecnológico que se ha utilizado para la elaboración de **HERA iOS**, organizado en diversas categorías según su función (véase Figura 4.7). Además, se proporcionará una descripción de las especificaciones del hardware utilizado para su desarrollo.



Figura 4.7: Marco tecnológico HERA iOS.

4.4.1. Herramientas de planificación y documentación del proyecto

En este punto se presentan las herramientas utilizadas para garantizar una planificación efectiva y una documentación detallada del proyecto **HERA iOS**.

4.4.1.1. Jira

Es una plataforma de gestión de proyectos desarrollada por Atlassian. Proporciona una extensa variedad de herramientas diseñadas para la planificación y el seguimiento de historias de usuario, defectos, errores y otros elementos relevantes, lo que permite una gestión eficaz de proyectos. **Jira**⁴ ha permitido realizar la planificación y gestión de incidencias para cada uno de los *sprints* del proyecto mediante la utilización de un *Agile Board* o Tablero *Scrum*. Este tablero se divide en tres secciones: "To Do" (por hacer), "In Progress" (en progreso) y "Done" (hecho), reflejando las etapas del ciclo de trabajo en un sprint. En las Figuras 4.8 y 4.9, se

⁴<https://www.atlassian.com/es/software/jira>

ilustra un ejemplo del *Agile Board*, donde una incidencia ha surgido en una historia de usuario durante el transcurso del proyecto.

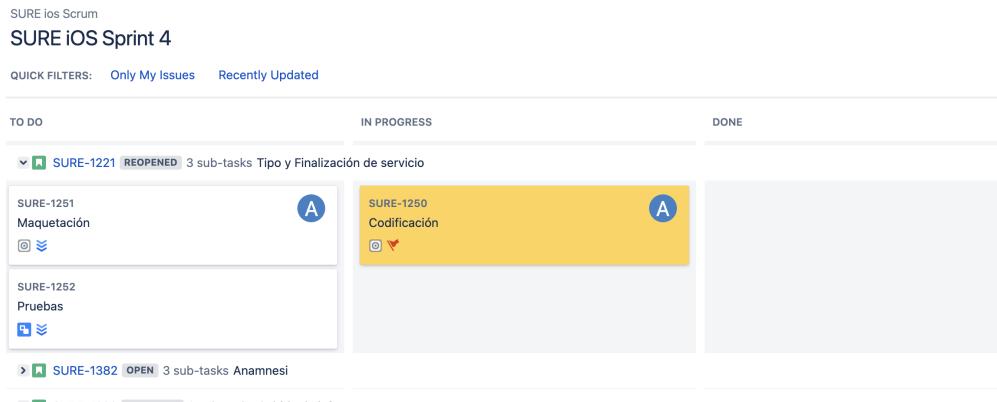


Figura 4.8: Agile Board con incidencia en la tarea SURE-1250.

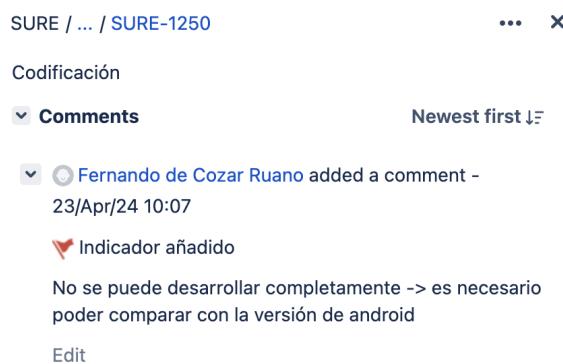


Figura 4.9: Incidencia abierta de la tarea SURE-1250.

4.4.1.2. Confluence

Es una plataforma, desarrollada por Atlassian, que permite centralizar el conocimiento y la información de una organización. Proporciona un conjunto de herramientas de documentación que permite a los usuarios escribir y editar contenido como en una *wiki*, facilitando la colaboración entre equipos y la reutilización de conocimiento. **Confluence**⁵ ha facilitado la realización y reutilización de documentación del proyecto, sirviendo como una guía de referencia para resolver cualquier duda que pudiera surgir.

4.4.1.3. Miro

Es una plataforma de colaboración que permite a los equipos realizar diagramas, compartir ideas, organizar información y participar en tiempo real en un espacio digital compartido. Proporciona un conjunto de herramientas para la creación de diagramas, mapas mentales, tableros ágiles, todo de una forma visual y atractiva, que favorecen a la productividad de los equipos y les permite visualizar conceptos complejos. **Miro**⁶ se ha empleado durante el transcurso de este proyecto para la organización de la reunión de *Inception*, ofreciendo herramientas que han facilitado una representación visual más clara y comprensible.

4.4.1.4. Overleaf

Es una plataforma *online* que permite la creación, edición y colaboración en documentos científico-técnicos en *LATEX*. **Overleaf**⁷ se ha utilizado para redactar este trabajo de fin de Grado.

⁵<https://www.atlassian.com/software/confluence>

⁶<https://miro.com/es/>

⁷<https://es.overleaf.com/>

4.4.2. Herramientas de comunicación

En este apartado se enumeran las herramientas que han facilitado la comunicación entre los miembros del equipo durante el desarrollo tanto de **HERA iOS** como del TFG, dado que este proyecto se ha llevado a cabo de forma remota.

4.4.2.1. Microsoft Teams

Es una aplicación de colaboración creada para el trabajo híbrido que ofrece chat, videoconferencias, almacenamiento de archivos y herramientas de productividad para que el equipo esté informado, organizado y conectado en todo momento, en un solo lugar. **Microsoft Teams**⁸ ha facilitado la comunicación y colaboración de los miembros del equipo durante todo el desarrollo del proyecto, desempeñando un papel fundamental durante las *Dailies* (reuniones diarias).

4.4.2.2. Outlook

Es una aplicación de correo electrónico y gestión de información personal desarrollada por *Microsoft*. **Outlook**⁹ ha permitido la comunicación entre los tutores y el alumno para coordinar reuniones o resolver dudas.

4.4.3. Herramientas de control de versiones

En este punto se presentan las herramientas empleadas en el proyecto **HERA iOS** para llevar a cabo un control de versiones del código fuente.

4.4.3.1. Git

Es un sistema de control de versiones distribuido, gratuito y de código abierto, diseñado para gestionar con rapidez y eficacia cualquier tipo de proyecto. **Git**¹⁰ permite a los desarrolladores mantener un historial completo de revisiones del código, lo que ha facilitado realizar un seguimiento y control de los cambios en el código fuente del proyecto.

4.4.3.2. GitLab

Es una plataforma para la gestión de repositorios Git, como se describe en el apartado anterior 4.4.3.1, de código fuente y colaboración en el desarrollo de software que ofrece un conjunto de herramientas para el ciclo de vida del desarrollo de aplicaciones. **GitLab**¹¹ ha facilitado la creación y clonación de repositorios, garantizando un branching y merging seguros, junto con un seguimiento exhaustivo de los cambios realizados en el proyecto.

4.4.3.3. Sourcetree

Es un cliente gratuito, disponible para los sistemas operativos Windows y macOS, que proporciona una interfaz intuitiva para simplificar el manejo de sistemas de control de versiones como Git y Mercurial. **Sourcetree**¹² ha simplificado la gestión del repositorio del proyecto ya que permite realizar seguimiento de cambios, crear y fusionar ramas de manera eficiente. Además, proporciona herramientas visuales para la revisión del código, resolución de conflictos y visualización del historial de cambios. En la Figura 4.10, podemos observar la interfaz de ramas en Sourcetree.



Figura 4.10: Interfaz de ramas en Sourcetree de HERA iOS.

⁸<https://www.microsoft.com/es-es/microsoft-teams/log-in>

⁹<https://www.microsoft.com/es-es/microsoft-365/outlook/email-and-calendar-software-microsoft-outlook>

¹⁰<https://git-scm.com/>

¹¹<https://about.gitlab.com/>

¹²<https://www.sourcetreeapp.com/>

4.4.4. Herramientas de desarrollo

En este apartado se enumeran las herramientas que han permitido que el desarrollo del proyecto HERA iOS fuera posible.

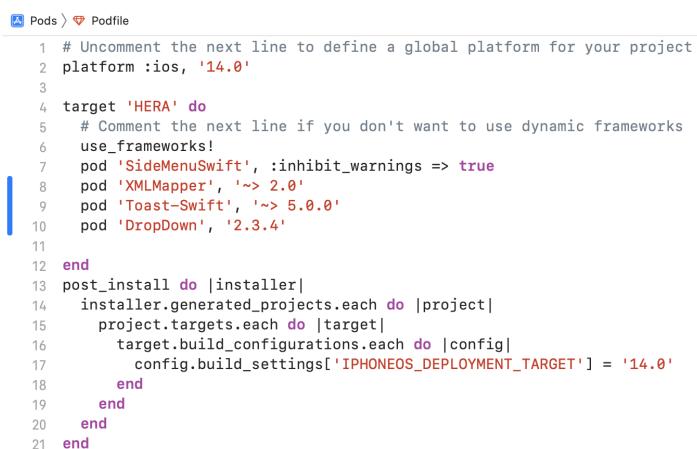
4.4.4.1. XCode 14.2

Es un entorno de desarrollo integrado (IDE) diseñado por Apple para la creación de software para sus plataformas, incluyendo macOS, iOS, watchOS, tvOS. **XCode**¹³ ofrece un conjunto completo de herramientas que incluyen un editor de código, compilador y un depurador, entre otros recursos para el desarrollo. Además, facilita la creación de interfaces de manera intuitiva mediante la herramienta gráfica *Interface Builder*, permitiendo probarlas en un entorno simulado cuando no se disponga de un dispositivo físico, gracias a la herramienta *Simulator*. Es compatible con los lenguajes C, C++, Swift y Objective-C.

4.4.4.2. CocoaPods

Es un gestor de dependencias que sirve tanto para proyectos Swift como para proyectos Objective-C. **Cocoa-Pods**¹⁴ facilita la incorporación y gestión de librerías de terceros en aplicaciones, automatizando el proceso de descarga, instalación y configuración de estas dependencias.

El uso de CocoaPods en XCode se caracteriza por su simplicidad, ya que el desarrollador únicamente necesita incorporar un archivo de configuración denominado *Podfile* (véase Figura 4.11) a su proyecto, donde especifica las librerías necesarias junto con las versiones requeridas. Posteriormente, CocoaPods se encarga de gestionar estas dependencias de manera automatizada, facilitando así su instalación en el proyecto.



```

Pods > Podfile
1 # Uncomment the next line to define a global platform for your project
2 platform :ios, '14.0'
3
4 target 'HERA' do
5   # Comment the next line if you don't want to use dynamic frameworks
6   use_frameworks!
7   pod 'SideMenuSwift', :inhibit_warnings => true
8   pod 'XMLMapper', '~> 2.0'
9   pod 'Toast-Swift', '~> 5.0.0'
10  pod 'DropDown', '2.3.4'
11
12 end
13 post_install do |installer|
14   installer.generated_projects.each do |project|
15     project.targets.each do |target|
16       target.build_configurations.each do |config|
17         config.build_settings['IPHONEOS_DEPLOYMENT_TARGET'] = '14.0'
18       end
19     end
20   end
21 end

```

Figura 4.11: *Podfile* de HERA iOS.

4.4.4.3. Postman

Es una plataforma para el desarrollo de APIs que ofrece una amplia gama de herramientas para diseñarlas, probarlas y documentarlas de manera eficiente. **Postman**¹⁵ se ha empleado en este proyecto para enviar solicitudes HTTP a servidores web y API endpoints de HERA, para realizar pruebas de conexión y visualizar la estructura de las respuestas que éstos devuelven.

¹³<https://developer.apple.com/documentation/xcode/>

¹⁴<https://cocoapods.org/>

¹⁵<https://www.postman.com/>

4.4.5. Medios hardware

En cuanto al equipo hardware empleado, es relevante señalar que se ha utilizado exclusivamente el equipo portátil proporcionado por *Inetum*, cuyas especificaciones se encuentran detalladas en la Tabla 4.2.

Especificaciones hardware	
Modelo	MacBook Pro (15 pulgadas, 2016)
Procesador	2,6 GHz Intel Core i7 de 4 núcleos
Memoria	16 GB 2133 MHz LPDDR3
Gráficos	Radeon Pro 450 2 GB Intel HD Graphics 530 1536 MB
Almacenamiento	250 GB
Sistema Operativo	macOS Monterey 12.7.4

Tabla 4.2: Especificaciones hardware.

CAPÍTULO 5

Resultados

En el presente capítulo se describirá con detalle el desarrollo del proyecto, donde se aplicará la metodología anteriormente mencionada en el capítulo 4. Cada sección de este capítulo contendrá el desarrollo de un *sprint*, conservando una estructura uniforme que incluirá: el objetivo general del *sprint*, la planificación del *sprint*, el desarrollo del mismo por historias de usuario y la reunión de revisión incluyendo una conclusión final del *sprint*. Además, se incluirá una sección que detallará la reunión de *Inception*, durante la cual se estableció la visión global del proyecto y su alcance. Por último, se mostrará el diagrama de Gantt con el objetivo de tener una visión general del tiempo empleado en el proyecto.

5.1. INCEPTION DE HERA iOS

En esta sección se abordará la naturaleza y desarrollo de la reunión de *Inception*, así como su ejecución durante la fase inicial del presente proyecto en *Inetum*.

5.1.1. Definición

La reunión de *Inception*, también conocida como *Agile Inception*, constituye un taller colaborativo que reúne a todos los interesados clave del proyecto. Su objetivo principal es establecer una base sólida que garantice el desarrollo y la ejecución exitosa del proyecto.

La duración de la reunión de *Inception* de **HERA iOS** fue de 3 horas y entre los asistentes se encontraban el *Scrum Master*, el jefe del Proyecto, miembros del equipo de desarrollo y el tutor académico del autor de este TFG.

5.1.2. Fases

Para llevar a cabo la reunión de *Inception* de **HERA iOS**, se empleó la plataforma colaborativa Miro, en la cual se abordaron las siguientes fases:

- Visión
- Comunidad
- Definición del Alcance de la Colaboración
- Restricciones y Riesgos
- Historias de usuario

5.1.2.1. Visión

Durante esta fase se pretende alinear la visión y conseguir una idea clara del motivo detrás del proyecto incluyendo los objetivos principales y el valor que se espera conseguir, esto ayudará al equipo a tomar mejores decisiones y encontrar mejores soluciones.

En la etapa de Visión, es fundamental que el equipo responda a la pregunta ¿por qué estamos aquí? A partir de la respuesta a la pregunta en cuestión, es factible llevar a cabo un análisis con el fin de determinar el producto que se quiere conseguir, quiénes son los clientes y por qué se ha decidido construir ese producto. En consecuencia, se obtendrá una valoración del valor que se va a crear con ese producto y su relevancia para la empresa.

Para llevar a cabo esta fase se utilizó la técnica de ***Elevator Pitch***. Este enfoque nos proporciona la capacidad de presentar nuestro producto de manera concisa en un período de tiempo limitado, no excediendo los treinta segundos, similar a una situación hipotética en un ascensor. Es una herramienta ideal para saber si todos los involucrados en el proyecto son capaces de crear un discurso para vender el producto a otros.

Esta técnica se empleó de forma colaborativa, requiriendo la participación de todos los miembros de la reunión. Cada participante tuvo que responder a una serie de cuestiones específicas, con el objetivo de llegar a un consenso colectivo sobre qué es lo que se iba a llevar a cabo. Las cuestiones que se tuvieron que responder son las siguientes:

- **Para:** Grupo de usuarios al que va dirigido el producto.
- **Qué:** Las necesidades que tienen esos individuos.
- **El:** Nombre del producto.
- **Es un:** Tipo de producto.
- **Qué:** Objetivo o funcionalidad principal.
- **A diferencia de:** Productos ya existentes, nuestros o de competidores.
- **Aporta:** Gran ventaja de nuestro producto frente a los que ya existen.

Como podemos observar en la Figura 5.1, el resultado de cada uno de los participantes en la reunión fue el siguiente:

	PARA	QUE	EL	ES UN	QUE	A DIFERENCIA DE	APORTA
Fernando	Concejalías de Salud	Informar el estado de pacientes	App HERA iOS	Aplicación para iPad	Permite a través de relatar diferentes formas de informar el estado de los pacientes	?	considera problemas como poca o nula conexión a internet
Enrique	Futuros clientes del sector Salud	informar en tiempo real del estado de los pacientes de forma remota en una ambulancia	HERA	App para iOS (iPad)	Documentar enfermo atendido por ambulancia	Informe Escrito	Rapidez Atención Sanitaria
Christian	Personal Médico de Una Unidad de Atención	necesitan informar sobre el paciente	HERA	app para iPad	Documento sobre un paciente	Elaboración manual del informe	Automatización y rapidez en la elaboración del informe y envío en el tiempo a Hospital
Jose Antonio	personal médico en ambulancias	quieren apuntar datos básicos sobre un paciente y enviarlos directamente al centro médico	HERA	aplicación iOS	Permite obtener el historial del paciente y seguir su evolución de situación actual	?	soporte multiplataforma y un uso ya contrastado
Alejandro	Personal médico de hospitales	Necesitan consultar e informar sobre un paciente	HERA	aplicación desarrollada en iOS para iPad	consultar el historial de los pacientes y relesar informes	tener que introducir informes a mano	portabilidad a un nuevo sistema y mayor agilidad en el proceso de relesar informes
César	Personal sanitario de diferentes unidades de atención	necesitan conocer información detallada sobre el estado de salud de un cierto afectado	HERA	app para iOS (iPad)	nos ayuda a registrar los datos del afectado	otras aplicaciones de salud	agilidad y rapidez en el proceso de registro de información

Figura 5.1: Resultado Participantes *Elevator Pitch* de HERA iOS.

En la Figura 5.2 se ilustra el resultado del proceso llevado a cabo en la herramienta Miro, mientras que en la Tabla 5.1 se muestra un resumen el resultado final del *Elevator Pitch* de HERA iOS llevado a cabo por consenso de todos los participantes en la reunión.

Resultado final <i>Elevator Pitch</i>	
Para:	Personal médico sanitario
Qué:	Informar sobre el estado del paciente.
El:	<i>HHealth Record from Ambulance</i> (HERA iOS).
Es un:	Aplicación iOS (iPad).
Qué:	Introduce, actualiza y consulta de distintas fuentes los datos del paciente.
A diferencia de:	Informar en papel.
Aporta:	Tanto rapidez en el proceso de llenar informes como en su uso contrastado.

Tabla 5.1: Resultado final *Elevator Pitch* de HERA iOS.

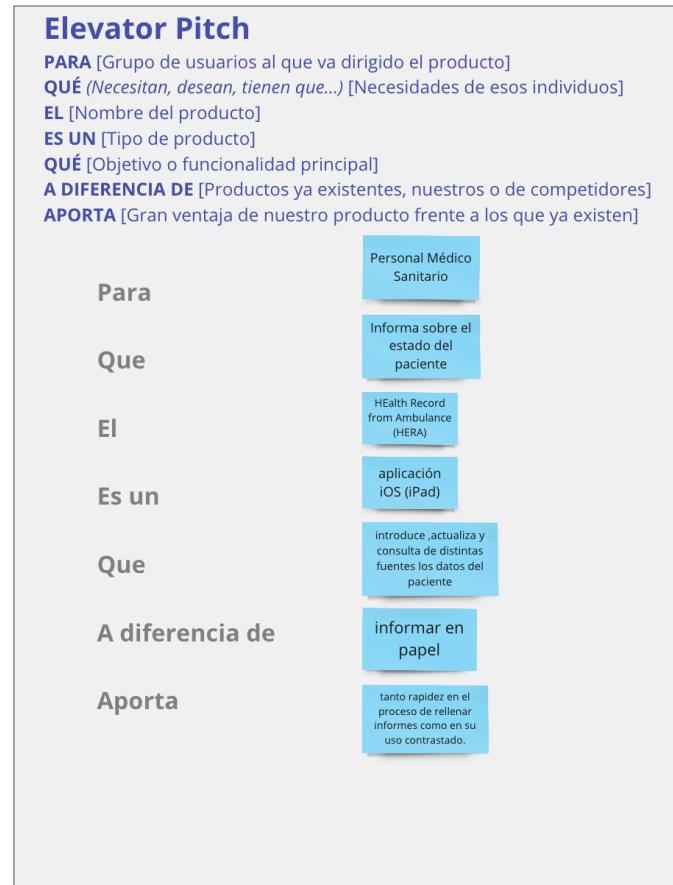


Figura 5.2: Resultado final *Elevator Pitch* de HERA iOS en Miro.

5.1.2.2. Comunidad del Proyecto

Durante esta fase se pretende identificar a todas las personas que participarán durante el desarrollo del proyecto. Una temprana identificación y establecer un ambiente de colaboración, confianza y compromiso entre las partes involucradas es fundamental, para el éxito del proyecto a largo plazo.

En la Figura 5.3 se ilustra el resultado del proceso llevado a cabo en la herramienta Miro, mientras que en la Tabla 5.2 se muestra un resumen de las personas que van a estar involucradas en el desarrollo del proyecto HERA iOS y sus responsabilidades.

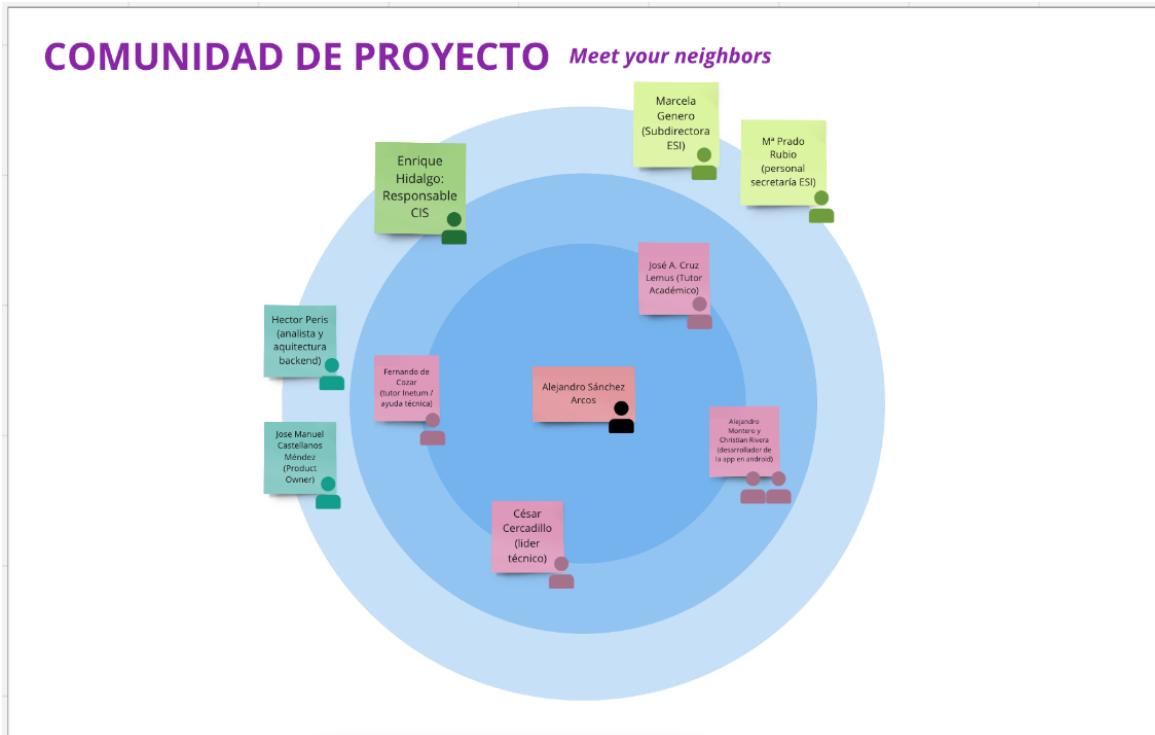


Figura 5.3: Comunidad del Proyecto de HERA iOS en Miro.

Organización	Nombre	Cargo
UCLM	Alejandro Sánchez Arcos	Desarrollador
UCLM	José Antonio Cruz Lemus	Tutor Académico
UCLM	Marcela Genero Bocco	Coordinadora programa FORTE
UCLM	María del Prado Rubio Serrano	Contacto gestiones ESI
INETUM	Fernando de Cózar Ruano	Co-Tutor Académico
INETUM	César Cercadillo López De Medrano	Líder Técnico
INETUM	José Manuel Castellanos Méndez	Product Owner
INETUM	Christian Rivera Balseras	Desarrollador App Android
INETUM	Alejandro Montero González	Desarrollador App Android
INETUM	Hector Peris Gimeno	Analista y Arquitectura Backend
INETUM	Enrique Hidalgo Vera	Responsable CIS Miguelturra

Tabla 5.2: Comunidad del Proyecto de HERA iOS.

5.1.2.3. Definición del Alcance de la Colaboración

En relación con la fase de colaboración, se establecieron las siguientes directrices:

- **Sprints:** se establece que la duración de cada *sprint* será de **tres semanas**.
- **Sprint Planning:** se organiza una reunión con el *Scrum Team* para abordar qué se va a realizar durante el *sprint*.
- **Daily Scrum:** se establece una reunión diaria entre el alumno y el equipo de movilidad de *Inetum* con el propósito de evaluar el progreso del desarrollo.
- **Sprint Review y Retrospective:** al concluir cada *sprint*, se establece la posibilidad de realizar tanto una revisión como una retrospectiva, siempre y cuando se pueda coordinar adecuadamente en cuanto a los horarios.
- **Planning memoria TFG:** cada cierto tiempo acordado entre el tutor académico y el alumno, se organizarán reuniones para llevar a cabo una revisión de la memoria del trabajo de fin de Grado.
- **Reuniones con tutores:** se establecen reuniones periódicas entre el alumno, el tutor y co-tutor académico para llevar a cabo un seguimiento del proyecto.

5.1.2.4. Restricciones y Riesgos

En cuanto a restricciones, se establecieron:

- **Fechas y Entregables.** como se puede observar en la Tabla 5.3 , se establecen las siguientes fechas clave para los entregables:

Entregable	Fecha
Envío Propuesta TFG	22 de Marzo de 2024
Versión Completa Memoria	26-28 de Junio de 2024
Depósito TFG	5 Julio de 2024
Defensa TFG	15-19 de Julio de 2024

Tabla 5.3: Restricciones de Fechas y Entregables.

- **Tecnología y Herramientas.** se estableció que se utilizarían las siguientes herramientas:
 - **XCode 14.2:** entorno de desarrollo integrado (IDE) que se utilizará para llevar a cabo el desarrollo de la aplicación **HERA iOS** para iPad.
 - **Jira:** plataforma para llevar a cabo la gestión del proyecto.
 - **Confluence:** plataforma para consultar y realizar documentación técnica.
 - **GitLab:** plataforma para llevar a cabo la subida del código y hacer control de versiones.
 - **Microsoft Teams:** plataforma que se usará para comunicarse con el equipo.
 - **Postman:** plataforma que se utilizará para hacer las comprobaciones oportunas de las peticiones con el *backend*.
 - **Overleaf:** editor de documentos para redactar la memoria del TFG.
- **Arquitectura e Infraestructura.** se definió lo siguiente:
 - **Endpoints:** para llevar a cabo las peticiones se harán a los *endpoints* del *backend* existente del servicio HERA actual.
 - **Arquitectura Genérica CIS:** se utilizará la arquitectura VIPER, mencionada en la sección 3.2, a través del uso de una plantilla genérica del CIS (Centro de Ingeniería Software) de Miguelturra.
 - **Cuenta Apple Developers:** se utilizará una cuenta *Apple Developer* para hacer las configuraciones correspondientes en la *App Store*.
- **Metodología.** se estableció que se utilizaría:
 - **Agile Framework CIS (SCRUM-CMMI).**
- **Seguridad.** en cuanto a seguridad se debe de tener en cuenta una comunicación segura en cuanto a los datos del paciente.

- **Calidad.** no se consideró ningún aspecto en cuanto a calidad en el momento.

Todo lo mencionado anteriormente se puede observar en la Figura 5.4.

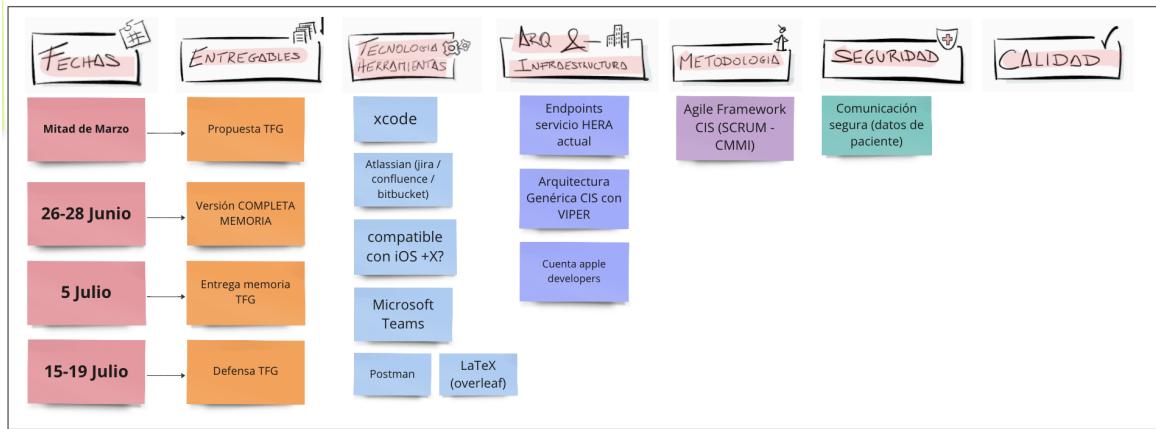


Figura 5.4: Restricciones de HERA iOS en Miro.

Como es evidente, todos los proyectos presentan una serie de riesgos con una probabilidad inherente de ocurrir, y en caso de suceder, dichos riesgos tienen un impacto significativo en alguna dimensión del proyecto.

Durante esta fase, cada integrante del equipo identificó una serie de riesgos potenciales que podrían incidir en el proyecto. Tras un proceso de consenso, se determinó un valor para la probabilidad e impacto de cada uno de ellos.

Se identificaron tres niveles de tipos de riesgos:

- **Nivel Personal:** riesgos relacionados con el autor de este Trabajo de Fin de Grado. Categorizado con el color violeta en Miro.
- **Nivel Equipo:** riesgos relacionados con el equipo de desarrollo. Categorizado con el color azul en Miro.
- **Nivel Desarrollo:** riesgos relacionados durante el desarrollo de la aplicación. Categorizado con el color azul añil en Miro.

Una vez definidos los niveles, se identificaron los siguientes riesgos correspondientes a cada nivel:

- **Nivel Personal.**
 - R1. Suspender asignatura optativa
 - R2. Compromiso y actitud
 - R3. Desconocimiento de las Tecnologías
 - R4. Adaptación nuevas herramientas
- **Nivel Equipo.**
 - R5. Cambios en la estructura del equipo
- **Nivel Desarrollo.**
 - R6. Cambios en el dominio de la aplicación
 - R7. Indisponibilidad de servicios *backend*

Se estableció que los riesgos que fueran más conflictivos (mayor probabilidad e impacto) deberían de tener unas acciones de mitigación y contingencia, mientras que para el resto se establecen de mitigación. Como se puede observar en la Figura 5.5, se identificaron que los riesgos R3 y R4 tienen criticidad media, por consiguiente, se le asignarán acciones de mitigación y contingencia. Por otro lado, los demás riesgos se consideran de criticidad baja, por lo que solo se les asignarán acciones de mitigación.

En la Tabla 5.4, se muestra como quedó la asignación de acciones de mitigación y contingencia.

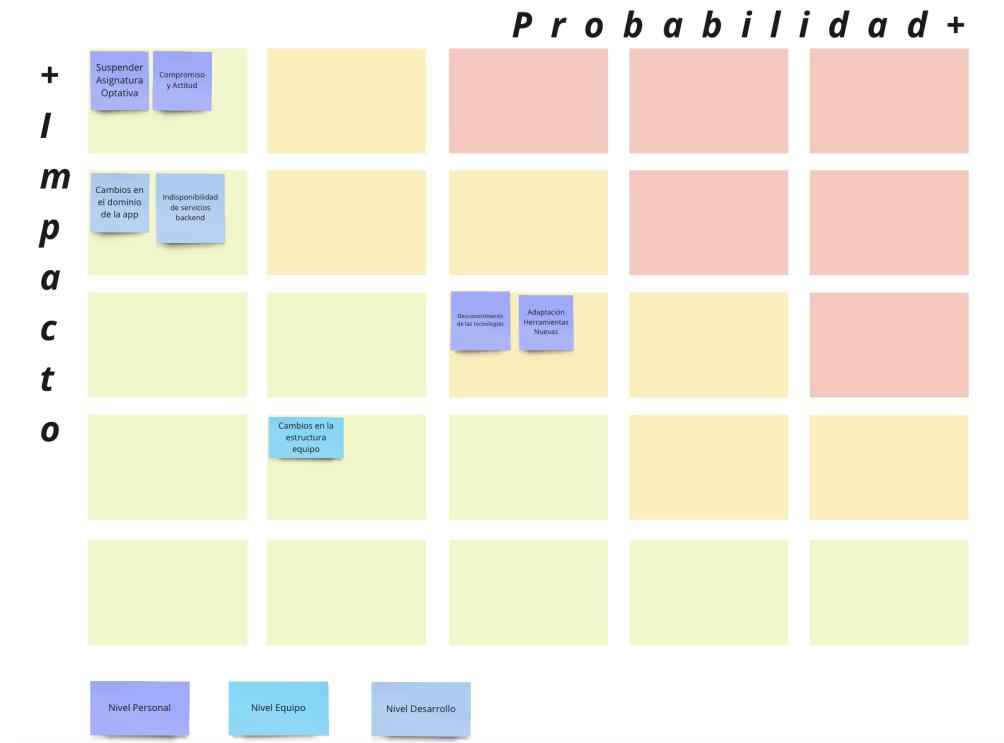


Figura 5.5: Riesgos de HERA iOS en Miro.

5.1.2.5. Historias de Usuario

Durante esta fase, se llevará a cabo la identificación de las historias de usuario, las cuales, serán desglosadas en tareas para dirigir el desarrollo. Estas historias de usuario constituirán el *product backlog* del proyecto.

Story Map

Aquí todos los participantes de la reunión colaboran para identificar las historias de usuario y *epics* que consideran necesarias para realizar el desarrollo del proyecto. Para poder identificar las prioridades de cada historia de usuario se categorizaron por colores siendo:

- **Rojo:** Prioridad alta
 - **Amarillo:** Prioridad media
 - **Verde:** Prioridad baja

En la Figura 5.6 se muestra como quedó la identificación final de todas las historias de usuario categorizadas por su prioridad.

Product Backlog

Una vez finalizada la identificación de las historias de usuario durante el *story map*, se establece el *Product Backlog* inicial de **HERA iOS**, incorporando el consenso de los participantes respecto a las historias de usuario identificadas.

La Tabla 5.5 muestra el Product Backlog inicial de **HERA iOS** con sus prioridades y a la épica a la que pertenece cada historia de usuario.

Riesgo	Acción Mitigación	Acción Contingencia
R1	- Establecer un plan de estudio detallado para la asignatura priorizando el tiempo para su completitud.	-
R2	- Establecer una comunicación abierta y transparente con el desarrollador para comprender sus necesidades, preocupaciones y motivaciones.	-
R3	- Proporcionar capacitación y recursos adecuados para que el desarrollador adquiera el conocimiento necesario sobre las tecnologías utilizadas en el proyecto.	- En el caso donde al desarrollador se le dificulte el aprendizaje, se le ofrecería una estrategia de aprendizaje más personalizada , asignando un mentor para proporcionarle una orientación individualizada.
R4	- Asignar un tutor que pueda brindar apoyo y orientación individualizada al desarrollador durante el proceso de adaptación.	- En el caso de indisponibilidad del tutor, se le designará otro miembro del equipo como tutor temporal para que le pueda brindar apoyo durante el proceso.
R5	- Realizar reuniones regulares para discutir los cambios planificados en la estructura del equipo y abordar cualquier inquietud o pregunta que puedan tener los miembros del equipo.	-
R6	- Mantener una comunicación abierta y regular con los stakeholders del proyecto para identificar y abordar cualquier cambio o evolución en el dominio de la aplicación.	-
R7	- Proporcionar acceso a entornos de desarrollo que simulen los servicios <i>backend</i> en caso de indisponibilidad.	-

Tabla 5.4: Acciones de mitigación y contingencia para riesgos de HERA iOS.

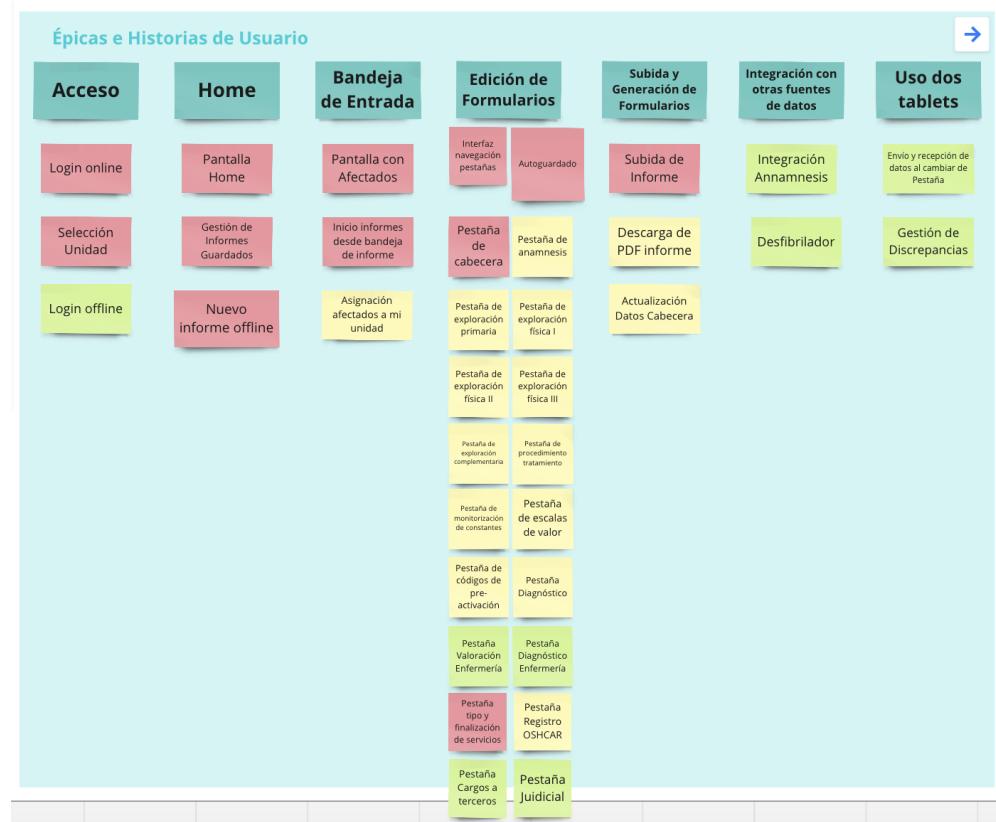


Figura 5.6: Story Map de HERA iOS en Miro.

Id	Épica	Historia de usuario	Prioridad
H1	Acceso	<i>Login online</i>	Alta
H2	Acceso	Selección de unidad	Alta
H3	Home	<i>Pantalla Home</i>	Alta
H4	Home	Gestión de informes guardados	Alta
H5	Home	<i>Nuevo informe offline</i>	Alta
H6	Bandeja de Entrada	Pantalla con afectados	Alta
H7	Bandeja de Entrada	Inicio de informes desde bandeja de entrada	Alta
H8	Edición de Formularios	<i>Pantalla edición de formularios</i>	Alta
H9	Edición de Formularios	Pestaña de cabecera	Alta
H10	Edición de Formularios	Autoguardado	Alta
H11	Edición de Formularios	Opción de guardar informe y guardar al salir	Alta
H12	Edición de Formularios	Pestaña tipo y finalización de servicio	Alta
H13	Subida y gen. de informes	Subida de informe	Alta
H14	Edición de Formularios	Pestaña Anamnesis	Media
H15	Edición de Formularios	Pestaña Exploración primaria	Media
H16	Edición de Formularios	Pestaña Exploración física 1	Media
H17	Edición de Formularios	Pestaña Exploración física 2	Media
H18	Edición de Formularios	Pestaña Exploración física 3	Media

Tabla 5.5: Product Backlog inicial de HERA iOS con prioridades y épicas.

5.2. ESTRUCTURA DEL DESARROLLO DE LOS SPRINTS

Tras completar la fase de *Inception* y concluir los estudios previos acerca de las tecnologías fundamentales para el TFG “**HERA iOS: Herramienta Móvil de Soporte al Proceso de Triaje durante Traslados Sanitarios**”, se procedió a iniciar el desarrollo del mismo.

Inicialmente, se estableció el alcance del proyecto mediante el uso de Jira. Esta herramienta resulta esencial para proyectos que implementan la metodología *Scrum*, véase en el capítulo 4.

Jira facilita la gestión del *product backlog* así como del *sprint backlog*. Cada *Product Backlog Item (PBI)* estará conformado de una historia de usuario o *epics*, las cuales las podemos definir como historias de usuario a más alto nivel. En esta herramienta, cada historia de usuario es un tipo de “*issue*”, la cual se puede dividir en varias tareas o *tasks* que son elementos de trabajo necesarios para implementar esa historia de usuario. A su vez, cada *task* se puede dividir en subtareas o *subtask*, las cuales son piezas aún más específicas de trabajo necesarias para completar una tarea. Con estos elementos podemos establecer el alcance del proyecto mediante la definición de la pila del producto.

En consecuencia, tras la reunión de *Inception*, la definición del *Product Backlog* inicial, con la inclusión de las historias de usuario acordadas (véase Figura 5.6), se encuentra disponible para su revisión en la Tabla 5.5.

Con el objetivo de gestionar de manera sistemática el desarrollo de cada *sprint*, se adoptará la siguiente estructura:

- Objetivo general del *Sprint*
- *Sprint Planning*
- Desarrollo
- *Sprint Review* y Conclusión

5.3. SPRINT 1

El inicio del primer *sprint* comenzó el 20 de febrero de 2024, concluyendo el 12 de marzo de 2024. Por lo tanto, la duración total del *sprint* fue de aproximadamente tres semanas, equivalentes a 15 días laborales.

5.3.1. Objetivo general del Sprint 1

El propósito de este *sprint* era desarrollar el acceso a la plataforma de forma *online* y una pantalla *home*. Para garantizar el acceso a la plataforma, fue necesario implementar la funcionalidad de inicio de sesión *online* y el módulo de selección de unidad.

Además, para obtener una versión preliminar de la pantalla de *home*, se llevó a cabo el desarrollo prescindiendo del módulo de informes guardados, y se inició la implementación de la funcionalidad para la creación de nuevos informes en modo *offline*.

5.3.2. Sprint 1 Planning

La reunión de planificación del primer *sprint* se llevó a cabo el 20 de febrero de 2024, con una duración aproximada de dos horas. En ella, se realizó un análisis de los puntos historia utilizando la técnica *Planning Poker*, véase en el capítulo 4, de las historias de usuario planificadas para el *sprint*.

En consecuencia, la planificación quedaría de la siguiente manera, según se muestra en la Tabla 5.6.

Historia de usuario	Estimación Puntos/Historia	Estimación Horas
H1: Login online	4	32
H2: Selección de unidad	4	32
H3: Pantalla Home	1	8
H4: Gestión de informes guardados	5	40
H5: Nuevo informe offline	2	16

Tabla 5.6: Planificación del *Sprint 1*.

Durante el *Sprint Planning*, las historias de usuario fueron desglosadas en tareas, y se estimó individualmente el tiempo requerido para cada una de ellas en horas, véase Tabla 5.7.

Tareas	Estimación Horas
H1: Maquetación	2
H1: Codificación	25
H1: Pruebas	5
H2: Maquetación	10
H2: Codificación	17
H2: Pruebas	5
H3: Maquetación	5
H3: Codificación	2
H3: Pruebas	1
H4: Maquetación	15
H4: Codificación	17
H4: Pruebas	8
H5: Codificación	13
H5: Pruebas	3

Tabla 5.7: Planificación del *Sprint 1* en tareas/horas.

5.3.3. Desarrollo por historias del Sprint 1

En esta sección se abordará el desarrollo del *sprint*, desglosando la implementación de cada una de las historias de usuario estimadas.

5.3.3.1. H1: Login online

En la Tabla 5.8, se presentan las tareas y su estimación correspondiente para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero iniciar sesión de forma online para acceder a la plataforma.”

Tareas	Estimación Horas
H1: Maquetación	2
H1: Codificación	25
H1: Pruebas	5

Tabla 5.8: Estimación tareas de la historia de usuario H1: *Login online*.

El objetivo de esta historia de usuario es proporcionar un inicio de sesión *online* a la plataforma. Para ello, se acordó que el acceso se tenía que hacer a través de la API *open source Apereo CAS*¹ (*Central Authentication Service*). Esta API, ofrece un servicio *web* de autenticación *Single sign on*, que permite al usuario poder autenticarse de forma única y segura.

Para incorporar una vista *web* que integre contenido de un navegador *web* interactivo en aplicaciones iOS, Swift ofrece el componente *WKWebView*². Este componente puede ser añadido a la interfaz de la aplicación mediante el *Interface Builder* de Xcode. Utilizando *WKWebView*, se puede acceder a la URL de la API Apereo CAS para iniciar sesión y gestionar los eventos que ocurran en el navegador al intentar autenticarnos.

Una vez validada la autenticación, el navegador web proporcionará una respuesta que deberá ser capturada para obtener una URL de sesión con un parámetro especial denominado “*jsessionid*”. Posteriormente, se realizará otra petición a la URL capturada, de la cual se obtendrá una respuesta que contendrá un *JSON Web Token* (JWT). Este token contendrá la información del usuario codificada de manera segura.

En la Figura 5.7, podemos observar en el simulador de XCode el inicio de sesión en **HERA iOS** a través de la interfaz de autenticación de la API *Apereo CAS*.

¹<https://github.com/apereo/cas>

²<https://developer.apple.com/documentation/webkit/wkwebview>

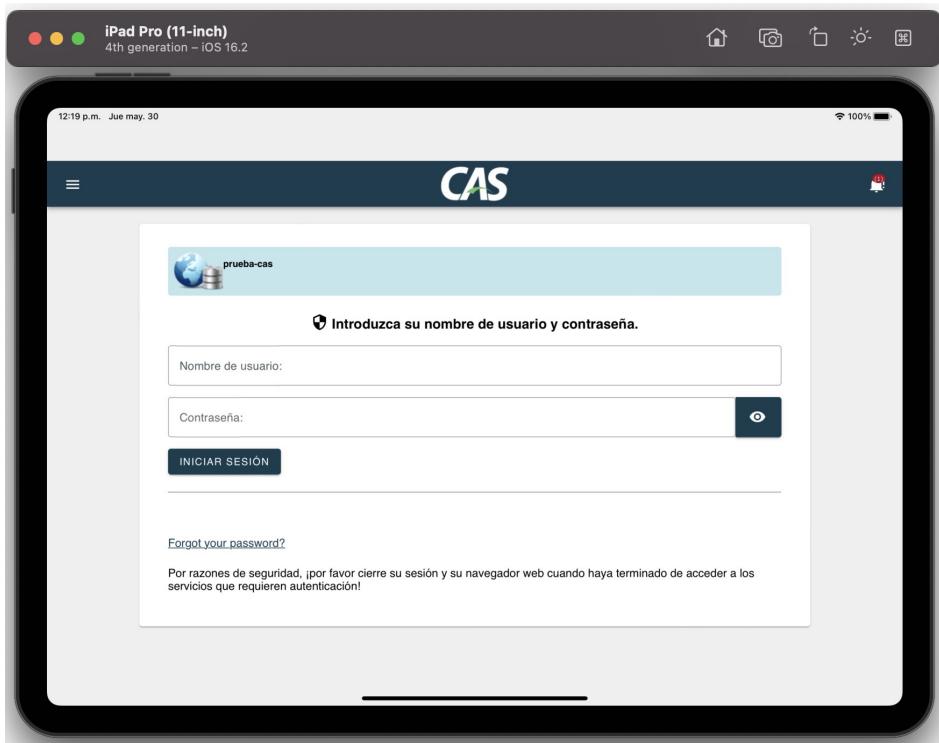


Figura 5.7: Login online HERA iOS.

5.3.3.2. H2: Selección de unidad

En la Tabla 5.9, se presentan las tareas y su estimación correspondiente para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero seleccionar una unidad para asignarla a la tablet.”

Tareas	Estimación Horas
H2: Maquetación	10
H2: Codificación	17
H2: Pruebas	5

Tabla 5.9: Estimación tareas de la historia de usuario H2: Selección de unidad.

El objetivo de esta historia de usuario es realizar una interfaz para que el usuario pueda seleccionar la unidad para asignarla a la *tablet*.

En consecuencia, se deberá crear un nuevo módulo VIPER y diseñar la interfaz mediante el *Interface Builder* de Xcode. En esta interfaz, el usuario tendrá la opción de seleccionar una unidad de entre un conjunto mostrado o ingresarla manualmente a través del campo de texto. Una vez seleccionada, el usuario podrá navegar hacia la pantalla del menú principal o volver a la de inicio de sesión.

Para habilitar al usuario la selección de una unidad entre un conjunto de opciones, se ha empleado el componente *UIPickerView*³ de *Swift*. Este componente permite al usuario desplazarse entre las opciones mostradas y realizar una selección. Una vez elegida la unidad, el usuario puede confirmar su selección y navegar hacia la pantalla de *home* o menú principal, pulsando en el botón “Continuar”.

Asimismo, el usuario podrá volver a la interfaz de inicio de sesión pulsando en el botón “Volver”.

En la Figura 5.8, podemos observar en el simulador de XCode la interfaz de selección de unidad de **HERA iOS**.

³<https://developer.apple.com/documentation/uikit/uipickerview>

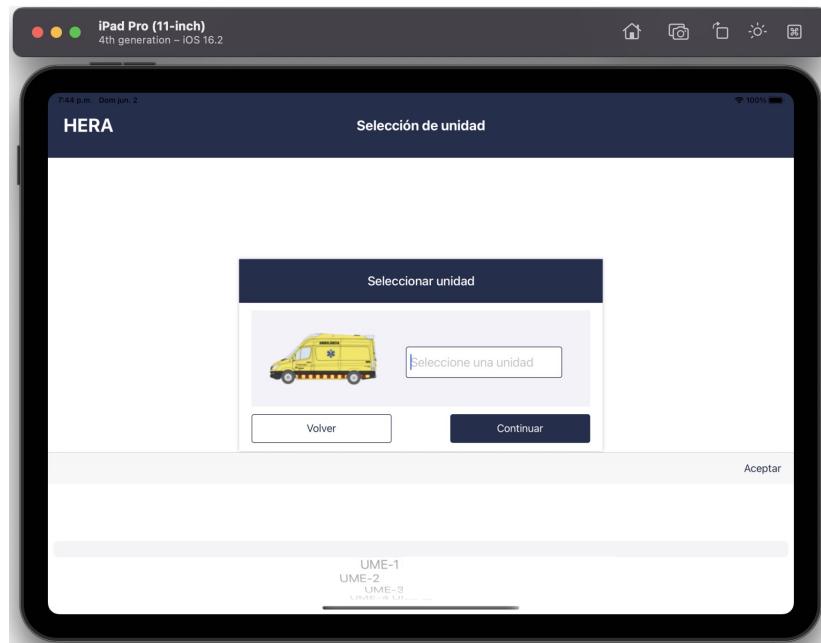


Figura 5.8: Selección de unidad de HERA iOS.

5.3.3.3. H3: Pantalla Home

En la Tabla 5.10, se presentan las tareas y su estimación correspondiente para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero acceder a una pantalla de menú principal para seleccionar fácilmente entre varias opciones y navegar por las funcionalidades de la aplicación.”

Tareas	Estimación Horas
H3: Maquetación	5
H3: Codificación	2
H3: Pruebas	1

Tabla 5.10: Estimación tareas de la historia de usuario H3: Pantalla Home.

El objetivo de esta historia de usuario es realizar una interfaz para que el usuario pueda seleccionar entre las funcionalidades de la aplicación.

En consecuencia, se deberá crear un nuevo módulo VIPER y diseñar la interfaz. En esta interfaz, el usuario podrá visualizar una lista de informes guardados en la *tablet*. Para cada informe, se mostrará información sobre el nombre de la unidad, el afectado, la dirección del incidente, el número de afectados y la fecha del incidente. El usuario tendrá la capacidad de eliminar cualquier informe y podrá filtrar la lista de informes guardados por número de informe o por fecha. Además, la interfaz cuenta con un menú lateral que proporciona información básica de la sesión del usuario, como la unidad asignada a la *tablet* y el nombre de usuario de inicio de sesión. En este menú, el usuario puede seleccionar entre dos opciones: la primera, “Bandeja de incidentes”, que permite la navegación hacia el módulo correspondiente; y la segunda, “Nuevo informe offline”, que redirige al módulo de creación y edición de informes, permitiendo al usuario generar uno nuevo. Asimismo, la interfaz permite al usuario cerrar sesión mediante un botón ubicado en la esquina superior derecha de la cabecera.

En esta historia, se procederá también a implementar una llamada a un servicio de *backend* con el fin de obtener la información completa de los integrantes de la unidad, incluyendo detalles sobre los sanitarios, tales como enfermeros y médicos. Dicha llamada se efectuará a través de un *endpoint* de los servicios actuales de HERA.

Cabe destacar que esta historia de usuario se limita únicamente a la implementación de la interfaz y la llamada al servicio. Las funcionalidades relacionadas con la gestión del listado de informes y los módulos correspondientes a las opciones del menú “Bandeja de incidentes” y “Nuevo informe offline” no serán implementadas en el alcance de esta historia.

En la Figura 5.9, podemos observar en el simulador de XCode la interfaz de *home* o menú principal de HERA iOS.

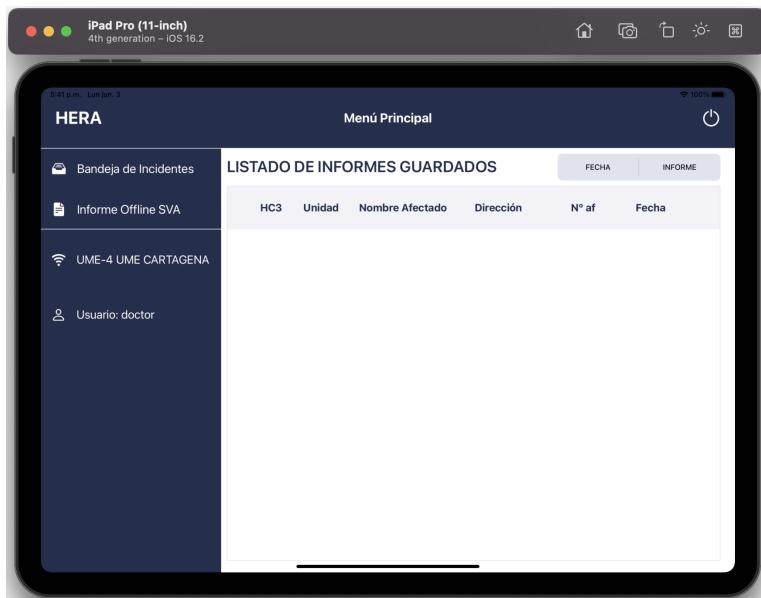


Figura 5.9: *Home* o menú principal de HERA iOS.

5.3.3.4. H4: Gestión de informes guardados

En la Tabla 5.11, se presentan las tareas y su estimación correspondiente para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero visualizar el listado de informes guardados de la tablet para poder gestionarlos y consultar su información básica.”

Tareas	Estimación Horas
H4: Maquetación	15
H4: Codificación	17
H4: Pruebas	8

Tabla 5.11: Estimación tareas de la historia de usuario H4: Gestión de informes guardados.

El objetivo de esta historia de usuario es implementar el almacenado de informes en la *tablet* para poder proporcionárselo a la interfaz de *home*. Además, se implementará las funcionalidades de eliminación de informes seleccionados y el filtrado del listado de informes según el número y la fecha.

Para poder implementar el almacenado de informes en la *tablet* se ha hecho uso de la memoria *sandbox* del dispositivo. La memoria *sandbox*, en el contexto de los dispositivos *Apple*, se refiere a un entorno de ejecución aislado y seguro diseñado para garantizar que las aplicaciones solo puedan acceder y manipular sus propios datos. Dentro de ella, cada aplicación tiene su propio espacio de almacenamiento privado donde puede crear, leer, escribir y eliminar archivos sin interferir con otros datos del sistema o de otras aplicaciones.

En consecuencia, para poder llevar a cabo esta funcionalidad en HERA iOS, se ha hecho uso del patrón *Singleton* para garantizar un acceso controlado y centralizado a la funcionalidad de manejo de datos. Esto se ha logrado mediante la creación de una clase aislada denominada “*DataManager*”, encargada de gestionar cualquier carga o modificación de informes en la memoria *sandbox*.

En la Figura 5.10, se presenta un fragmento de código donde se accede a la memoria *sandbox* para crear una carpeta destinada a almacenar todos los informes generados. Este enfoque asegura que los datos relacionados con los informes estén organizados y protegidos dentro del entorno seguro de la aplicación.

Dado que esta historia se abordó al final del *sprint*, habiéndose priorizado previamente la historia de usuario H5: “Creación de informes *offline*”, las funcionalidades para visualizar el listado, eliminar informes y filtrar el listado de informes se tratarán en el próximo *sprint*. Esta decisión se tomó debido a restricciones de tiempo que impidieron su implementación durante el *sprint* actual.

```

let fileManager = FileManager.default
guard let documentDirectory = try? fileManager.url(for: .documentDirectory, in:
    .userDomainMask, appropriateFor: nil, create: true) else{
    return}

let reportsFile = documentDirectory.appendingPathComponent("reportsFile")

// Verificar si la carpeta de informes no existe antes de crearla
if !fileManager.fileExists(atPath: reportsFile.relativePath, isDirectory: nil) {
    do {
        try fileManager.createDirectory(at: reportsFile, withIntermediateDirectories:
            false)
    } catch {
        print("Error al crear la carpeta de informes: \(error.localizedDescription)")
    }
}

```

Figura 5.10: Fragmento de código de acceso a memoria *sandbox* para crear la carpeta donde se van a guardar los informes en la *tablet*.

5.3.3.5. H5: Nuevo informe offline

En la Tabla 5.12, se presentan las tareas y su estimación correspondiente para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero crear informes offline para poder realizarlos, incluso cuando no haya conexión.”

Tareas	Estimación Horas
H5: Codificación	13
H5: Pruebas	3

Tabla 5.12: Estimación tareas de la historia de usuario **H5: Nuevo informe offline**.

El objetivo de esta historia de usuario es implementar la creación de nuevos informes para poder almacenarlos en la *sandbox* del dispositivo.

Para poder implementar la creación de informes, lo primero que se hizo fue crear una entidad llamada “Report”, para almacenar todos los campos requeridos para el informe. En ese momento, dado que no había ningún otro campo que almacenar, la entidad tenía únicamente un atributo denominado “cod” para su distinción. Esta estructura se definió empleando el protocolo *Codable*, ya que los informes debían almacenarse en formato JSON en la memoria. De este modo, nos simplificará la carga de este tipo de archivo en la estructura correspondiente.

Una vez creada la entidad, será necesario implementar un método en la clase *DataManager* para permitir el almacenamiento en la memoria *sandbox*. En la Figura 5.11, podemos observar el fragmento de código utilizado para almacenar un informe. En primer lugar, buscamos en la sandbox el directorio previamente creado para almacenar los informes (*reportsFile*). Una vez localizado, procedemos a codificar la estructura como JSON, seguido de la creación de la URL de la carpeta y finalmente escribimos en la URL del archivo correspondiente.

```

if let documentDirectoryURL = FileManager.default.urls(for: .documentDirectory, in:
    .userDomainMask).first {

    let reportsDirectoryURL = documentDirectoryURL.appendingPathComponent("reportsFile")
    do {
        let encoder = JSONEncoder()
        encoder.outputFormatting = .prettyPrinted
        let reportJSON = try encoder.encode(report)
        let jsonFileURL =
            reportsDirectoryURL.appendingPathComponent("\(report.COD).json")
        try reportJSON.write(to: jsonFileURL)
        return true
    } catch {
        print("Error al convertir o escribir el archivo JSON:
            \(error.localizedDescription)")
        return false
    }
}

```

Figura 5.11: Fragmento de código para crear un nuevo informe en memoria *sandbox*.

5.3.4. Sprint 1 Review y Conclusión

Se planificaron un total de cinco historias de usuario para este *sprint*. Tras realizar la revisión, se determinó que se habían completado un total de cuatro historias de usuario. En consecuencia, una historia de usuario, en la que se habían realizado avances pero que aún estaba incompleta, queda pendiente para el siguiente *sprint*. La historia de usuario en cuestión es:

- H4. Gestión de informes guardados:
 - Visualización del listado de informes.
 - Eliminar informe seleccionado.
 - Filtrar el listado de informes por número de informe y fecha.

Como conclusión de este *sprint*, el desarrollador ha demostrado productividad y eficiencia al completar casi todas las historias de usuario sin enfrentar problemas significativos. La comunicación dentro del equipo ha sido un aspecto clave, por lo que es importante intentar mantenerla en *sprints* futuros. La única historia de usuario pendiente indica una estimación optimista que puede ser refinada en el futuro, quedando pendiente como una sugerencia de mejora.

5.4. SPRINT 2

El inicio del segundo *sprint* comenzó el 13 de marzo de 2024, concluyendo el 3 de abril de 2024. La planificación del *sprint* se diseñó para que su duración total fuera de aproximadamente tres semanas, equivalentes a 15 días laborales. Sin embargo, no se consideró la festividad de Semana Santa, que abarcó del 24 de marzo al 31 de marzo, lo que resultó en dos días laborales menos, reduciendo el total del *sprint* a 13 días laborales.

5.4.1. Objetivo general del Sprint 2

El propósito de este *sprint* era completar la visualización y gestión de informes en la lista de la pantalla *home*, implementar el módulo de bandeja de incidentes y comenzar con las funcionalidades y pestañas del módulo de edición y creación de informes.

5.4.2. Sprint 2 Planning

La reunión de planificación del segundo *sprint* se llevó a cabo el 13 de marzo de 2024, con una duración aproximada de dos horas. En ella, se realizó un análisis de los puntos historia utilizando la técnica *Planning Poker*, véase en el capítulo 4, para estimar las historias de usuario planificadas para el *sprint*.

En consecuencia, la planificación quedaría de la siguiente manera, según se muestra en la Tabla 5.13.

Historia de usuario	Estim. Ptos/Historia	Estim. Horas
H4: Gestión de informes guardados	5	40
H6: Pantalla con afectados	3	24
H7: Inicio de informes desde bandeja de entrada	2	16
H8: Pantalla edición de formularios	3	24
H9: Pestaña de cabecera	3	24
H10: Autoguardado	1	8
H11: Opción de guardar informe y guardar al salir	1	8
H12: Pestaña tipo y finalización de servicio	6	48

Tabla 5.13: Planificación del *Sprint 2*.

Durante el *Sprint Planning*, las historias de usuario fueron desglosadas en tareas, y se estimó individualmente el tiempo requerido para cada una de ellas en horas, véase Tabla 5.14.

5.4.3. Desarrollo por historias del Sprint 2

En esta sección se abordará el desarrollo del *sprint*, desglosando la implementación de cada una de las historias de usuario estimadas.

Tareas	Estimación Horas
H4: Maquetación	15
H4: Codificación	17
H4: Pruebas	8
H6: Maquetación	10
H6: Codificación	10
H6: Pruebas	4
H7: Maquetación	1
H7: Codificación	13
H7: Pruebas	2
H8: Maquetación	15
H8: Codificación	7
H8: Pruebas	2
H9: Maquetación	15
H9: Codificación	7
H9: Pruebas	2
H10: Maquetación	4
H10: Codificación	3
H10: Pruebas	1
H11: Maquetación	5
H11: Codificación	2
H11: Pruebas	1
H12: Maquetación	22
H12: Codificación	22
H12: Pruebas	4

Tabla 5.14: Planificación del *Sprint 2* en tareas/horas.

5.4.3.1. H4: Gestión de informes guardados

En la Tabla 5.11, se presentan las tareas y su estimación correspondiente del *sprint* anterior para la implementación de esta historia de usuario. A continuación, se proporciona su descripción como recordatorio:

“Como usuario, quiero visualizar el listado de informes guardados de la tablet para poder gestionarlos y consultar su información básica.”

Tras finalizar el *sprint* anterior, quedaba pendiente la implementación de las funcionalidades:

- Visualización del listado de informes.
- Eliminar informe seleccionado.
- Filtrar el listado de informes por número de informe y fecha.

Para implementar la funcionalidad de visualización de informes se añadió a la interfaz de *home* el componente *UITableView*⁴ de *Swift*. Este componente se complementa con otro llamado *UITableViewCell*⁵, el cual permite personalizar cada celda de la tabla. De este modo, el usuario podrá visualizar en cada celda información básica del informe (ver campos Figura 5.9). Además, se añade en cada celda un ícono de “papelera” para implementar la funcionalidad de eliminación de informes.

En la Figura 5.12, se puede visualizar el resultado final de este funcionalidad en la interfaz de *home*.

Para la funcionalidad de eliminación de informes seleccionado, como se ha mencionado anteriormente, se añadió a cada celda de la tabla un ícono de “papelera”. A este ícono se le añadió el componente *UITapGestureRecognizer*⁶ de *Swift*, el cual detecta la interacción del usuario al pulsar sobre él. Al activarse este

⁴<https://developer.apple.com/documentation/uikit/uitableview>

⁵<https://developer.apple.com/documentation/uikit/uitableviewcell>

⁶<https://developer.apple.com/documentation/uikit/uitapgesturerecognizer>

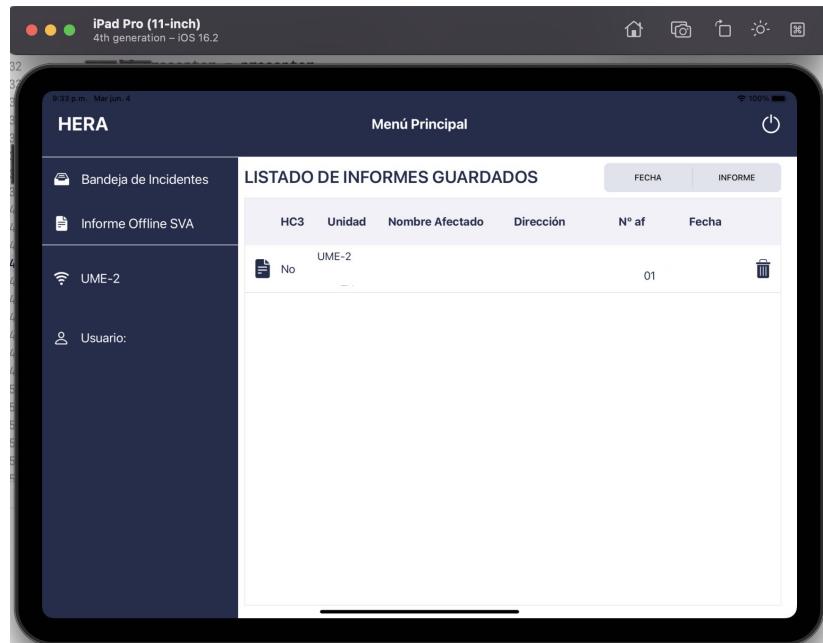


Figura 5.12: Listado de informes en interfaz de *Home* de HERA iOS.

gesto, se presenta un *UIAlertController*⁷ para solicitar al usuario la confirmación de la eliminación del informe correspondiente. Del mismo modo, para poder eliminar un informe de la memoria *sandbox*, se implementa en la clase “*DataManager*” un método el cual buscará el informe por su número para eliminarlo de la memoria del dispositivo.

En la Figura 5.13, se presenta en la interfaz de *home* el componente *UIAlertController* de *Swift* tras haber pulsado en el ícono de “papelera” de un informe para poder eliminarlo.

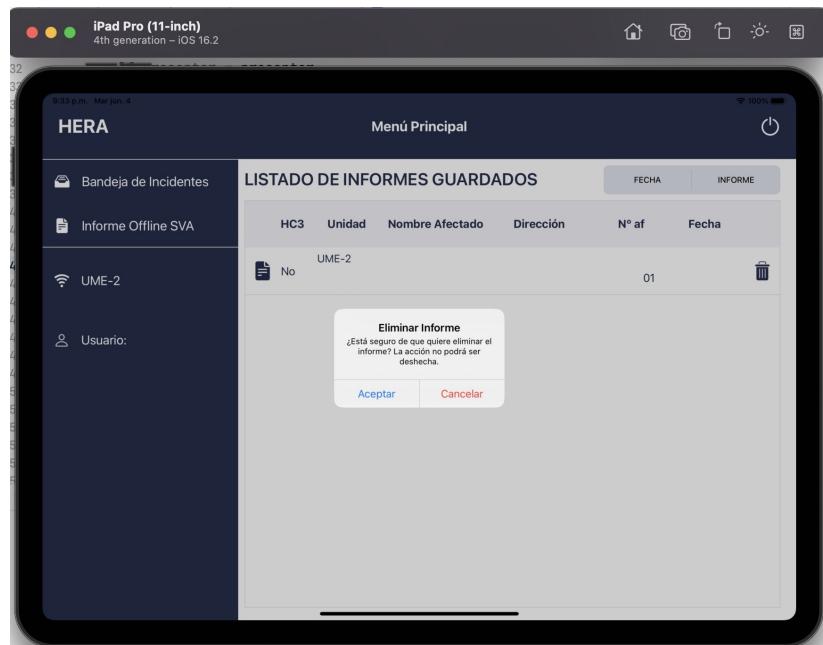


Figura 5.13: Eliminación informe en interfaz de *Home* de HERA iOS.

⁷<https://developer.apple.com/documentation/uikit/uialertcontroller>

La funcionalidad del filtro de listado de informes se ha implementado mediante el componente *UISegmentedControl*⁸ de Swift , el cual proporciona una apariencia similar a la de un filtro. Este componente estará vinculado a la lista de la tabla, de manera que al seleccionar un elemento del filtro, se mostrará la lista ordenada según el filtro. En caso de activarse el filtro “Informe”, la lista se ordenará en función del número de informe seleccionado. Del mismo modo, al activar el filtro “Fecha”, la lista se organizará según ese criterio.

En la Figura 5.14, se muestra el filtrado de informes por su número en la interfaz de *Home*.

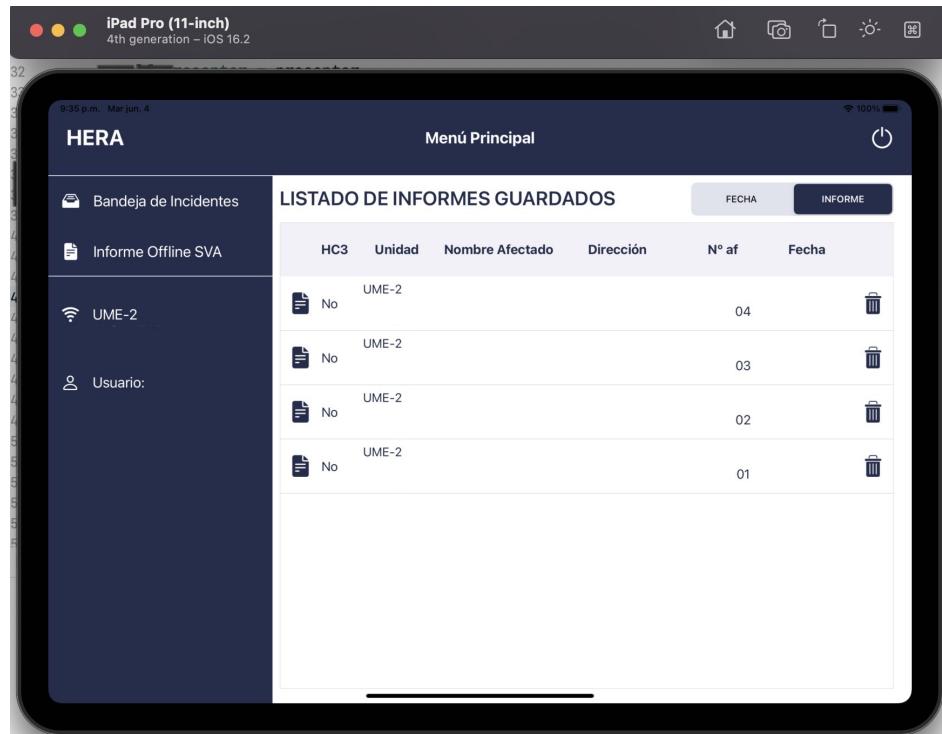


Figura 5.14: Filtrado de informes por número en interfaz de *Home* de HERA iOS.

5.4.3.2. H6: Pantalla con afectados

En la Tabla 5.15, se presentan las tareas y su estimación correspondiente para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero acceder a la bandeja de incidentes para visualizar, asignar y desasignar afectados a la unidad.”

Tareas	Estimación Horas
H6: Maquetación	10
H6: Codificación	10
H6: Pruebas	4

Tabla 5.15: Estimación tareas de la historia de usuario H6: Pantalla con afectados.

El objetivo de esta historia de usuario es implementar la interfaz de bandeja de incidentes donde el usuario pueda realizar las siguientes funcionalidades:

- Visualización del incidente de la unidad, incluyendo los afectados.
- Asignación de los afectados a la unidad correspondiente.
- Desasignación de los afectados en caso de ser necesario.

En consecuencia, se deberá crear un nuevo módulo VIPER y diseñar la interfaz. Para implementar la primera funcionalidad añadiremos el componente *UIView*⁹, que comprende un contenedor que contendrá la

⁸<https://developer.apple.com/documentation/uikit/uisegmentedcontrol>

⁹<https://developer.apple.com/documentation/uikit/uiview>

información básica del incidente como número de incidente y de intervención, unidad, localización, entre otros.

Para poder visualizar los afectados tanto asignados como desasignados se hará uso de dos *UITableView* con celdas personalizadas. Para cada afectado, la celda proporcionará información de la unidad, número de afectado, sexo, entre otros.

Dentro de este módulo, se le facilitará al usuario la opción de volver a la pantalla de *home* a través de un botón situado en la esquina superior izquierda.

En la Figura 5.15, se presenta la interfaz de bandeja de incidentes para la primera funcionalidad.

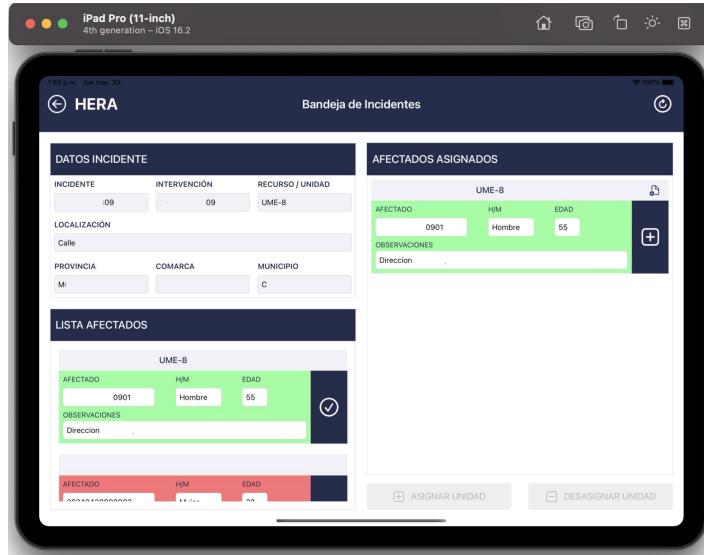


Figura 5.15: Interfaz Bandeja de Incidentes de HERA iOS.

Además, se procederá a implementar una llamada a un servicio de *backend* con el fin de obtener la información completa del incidente, incluyendo todos los detalles mencionados anteriormente. Dicha llamada se efectuará a través de un *endpoint* de los servicios actuales de HERA y su respuesta será parseada en un entity llamada “*Incident*”.

Esta llamada se hará nada mas entrar en el módulo o cuando el usuario pulse sobre el botón de refrescar situado en la parte superior derecha de la interfaz.

Para implementar las funcionalidades de asignar y desasignar afectado, se han añadido dos botones a la interfaz (véase Figura 5.15). Para asignar un afectado, se deberá seleccionar al afectado correspondiente presionando el botón “+” en la celda de la lista de afectados. Esta acción habilitará el botón necesario para completar la asignación a la unidad. De manera similar, para desasignar un afectado, se deberá presionar el botón “+” en la celda de la lista de asignados, lo que habilitará el botón para proceder con la desasignación.

Para distinguir visualmente a los afectados asignados, la celda correspondiente se marcará de color verde, mientras que las celdas de los afectados no asignados se mostrarán en rojo. El afectado seleccionado, ya sea para asignación o desasignación, se resaltará en color amarillo.

En las Figuras 5.16 y 5.17, se pueden observar como cambia la interfaz para las dos funcionalidades de asignar y desasignar afectado en la Bandeja de Incidentes.

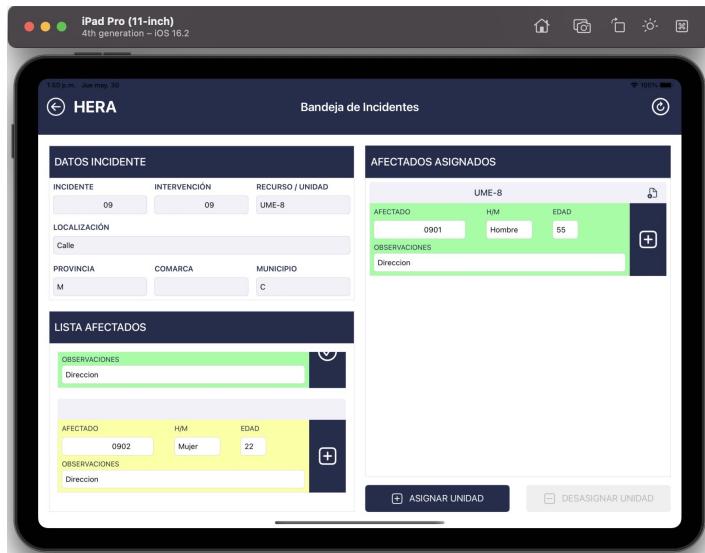


Figura 5.16: Interfaz Bandeja de Incidentes a la hora de asignar un afectado de HERA iOS.

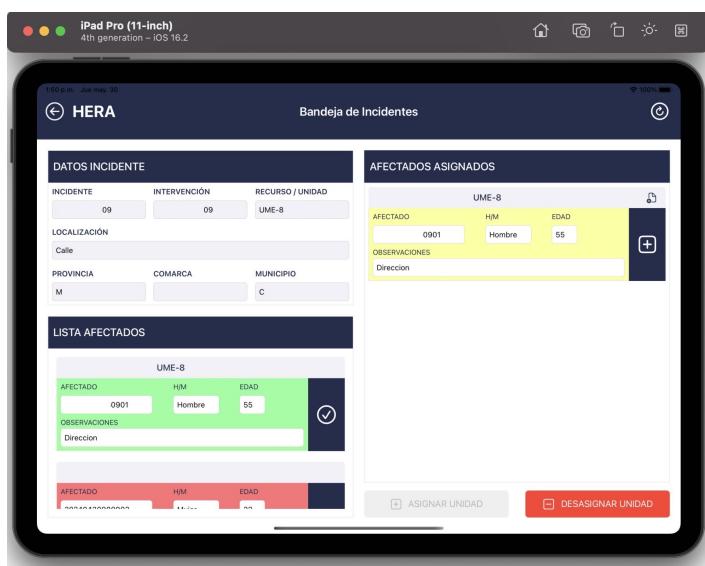


Figura 5.17: Interfaz Bandeja de Incidentes a la hora de desasignar un afectado de HERA iOS.

Para poder llevar a cabo las dos funcionalidades, se implementarán dos llamadas a un servicio de *backend*. Dicha llamada se efectuará a través de un *endpoint* de los servicios actuales de HERA y su respuesta será si la asignación o desasignación han tenido éxito o no. Esta respuesta se le presentará al usuario como un *UIAlertController*.

Cabe destacar que durante el desarrollo de esta historia de usuario se experimentó un bloqueo debido por la falta de disponibilidad de los servicios web de HERA, por lo que su desarrollo se vio retrasado.

5.4.3.3. H7: Inicio de informes desde bandeja de entrada

En la Tabla 5.16, se presentan las tareas y su estimación correspondiente para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero crear un nuevo informe utilizando la información del afectado disponible en la Bandeja de Incidentes, para poder generararlo.”

Tareas	Estimación Horas
H7: Maquetación	1
H7: Codificación	13
H7: Pruebas	2

Tabla 5.16: Estimación tareas de la historia de usuario **H7: Inicio de informes desde bandeja de entrada.**

El objetivo de esta historia de usuario es implementar la funcionalidad de crear un nuevo informe a partir de la información del afectado asignado en la bandeja de incidentes.

Durante la implementación de la interfaz para la Bandeja de Incidentes, se añadió un ícono a los afectados asignados para simbolizar la creación de un informe (ver Figura 5.18). Sin embargo, la implementación de esta funcionalidad no estaba incluida en el alcance de esa historia de usuario.



Figura 5.18: Inicio de nuevo informe desde Bandeja de incidentes de HERA iOS.

Para implementar esta funcionalidad, se añadió el componente *Swift UITapGestureRecognizer* al ícono para detectar su pulsación. Una vez que el usuario lo pulse, se presentará un *UIAlertController* solicitando confirmación para crear un nuevo informe a partir de la información del afectado.

Si el usuario confirma, se verificará en la memoria *sandbox* si dicho informe ya existe. En caso de que no exista, se creará un nuevo informe con la información correspondiente y se navegará hacia el módulo de creación y edición de informes. Si el informe ya existe, se presentará otro *UIAlertController* para que el usuario confirme si desea acceder al módulo de creación y edición de informes utilizando el informe que ya existe.

Es importante destacar que la navegación hacia el módulo de creación y edición de informes no pudo ser probada durante este *sprint*, ya que dicho módulo aún no había sido implementado.

5.4.3.4. H8: Pantalla de edición de formularios

En la Tabla 5.17, se presentan las tareas y su estimación correspondiente para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero navegar entre las pestañas del informe para introducir la información que corresponda.”

Tareas	Estimación Horas
H8: Maquetación	15
H8: Codificación	7
H8: Pruebas	2

Tabla 5.17: Estimación tareas de la historia de usuario H8: Pantalla de edición de formularios.

El objetivo de esta historia de usuario es implementar la interfaz cabecera de módulo de edición y creación de informes, donde el usuario pueda elegir entre las distintas pestañas para introducir información del informe.

En consecuencia, será necesario crear un nuevo módulo VIPER y diseñar la interfaz. Un informe consta de dieciocho pestañas, por lo que se deberá implementar un menú de navegación para seleccionar entre ellas. Este menú de navegación incluirá dieciocho botones, uno para cada pestaña, y el componente *UIScrollView*¹⁰ de *Swift* para permitir el desplazamiento entre los botones.

Además del menú de navegación, será necesario incluir la información básica del informe que se está editando o creando. Los datos correspondientes incluirán el nombre del usuario, número de afectado y nombre, unidad, entre otros (véase campos en Figura 5.19).

Dentro de este módulo, se le facilitará al usuario la opción de volver a la pantalla de *home* a través de un botón situado en la esquina superior izquierda.

En la Figura 5.19, se presenta la interfaz cabecera del módulo de creación y edición de informes.

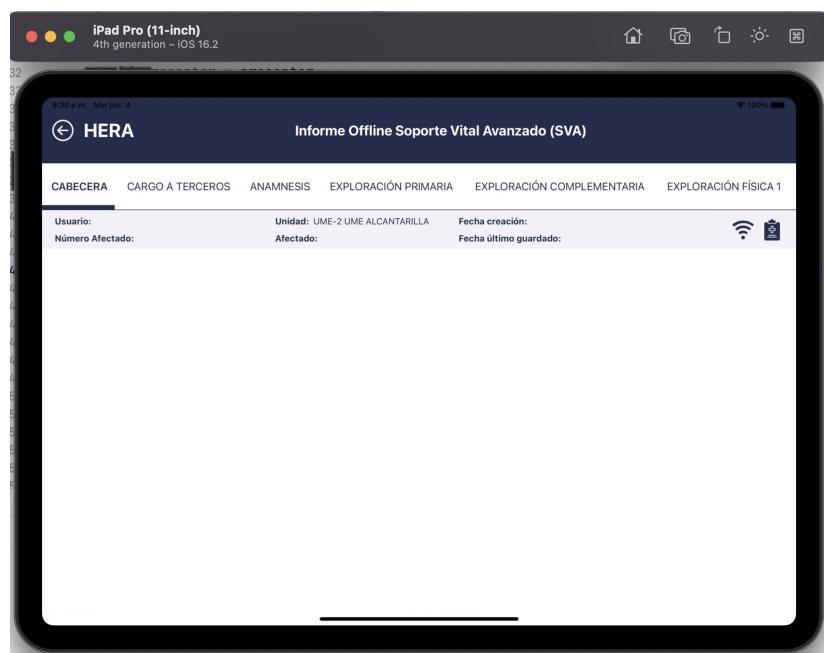


Figura 5.19: Interfaz cabecera de módulo de edición y creación de informes de HERA iOS.

5.4.4. Sprint 2 Review y Conclusión

Se planificaron un total de ocho historias de usuario para este *sprint*. En la Figura 5.20, se muestra el gráfico *Burndown* del *sprint* proporcionado por Jira, donde se destaca que se completaron un total de cuatro historias de usuario. Esto se debió a los bloqueos vistos durante el desarrollo del *sprint*, así como a las limitaciones temporales impuestas por la festividad de Semana Santa, la cual no se consideró durante la estimación y, por lo tanto, afectó a la duración del mismo.

En consecuencia, tras realizar la revisión, se determinó que se habían completado un total de cuatro historias de usuario. Por lo tanto, para el próximo *sprint* quedaría pendiente:

- H9: Pestaña de cabecera

¹⁰<https://developer.apple.com/documentation/uikit/uiscrollview>

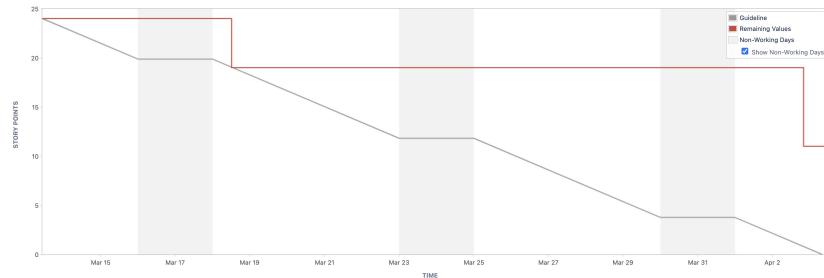


Figura 5.20: Gráfico Burndown proporcionado por Jira del *sprint* 2 de HERA iOS.

- H10: Autoguardado
- H11: Opción de guardar informe y guardar al salir
- H12: Pestaña tipo y finalización de servicio

Como conclusión de este *sprint*, se ha producido un avance significativo en el proyecto, pero no el esperado según la estimación. Durante este *sprint* se ha materializado el riesgo R7, que corresponde a la indisponibilidad de servicios *backend*, ya que el desarrollador sufrió un bloqueo en la historia de usuario H6: Pantalla con afectados. Para mitigar este riesgo en el próximo *sprint*, se mantendrán conversaciones entre el equipo de Movilidad y el equipo de *backend* de HERA con el objetivo de reducir su impacto. Además, como se ha mencionado anteriormente, la duración del *sprint* se vio reducida debido a la festividad de Semana Santa. Para el próximo *sprint*, se considerarán estos casos y se intentará realizar una estimación más conservadora.

5.5. SPRINT 3

El inicio del tercer *sprint* comenzó el 3 de abril de 2024, concluyendo el 24 de abril de 2024. Por lo tanto, la duración total del *sprint* fue de aproximadamente tres semanas, equivalentes a 15 días laborales.

5.5.1. Objetivo general del Sprint 3

El propósito de este *sprint* era iniciar la implementación de la pestaña “Cabeecera” del módulo de edición y creación de informes, desarrollar la funcionalidad de guardado y subida de informes, así como comenzar con el desarrollo de las pestañas “Tipo y finalización de servicio” y “Anamnesis”.

5.5.2. Sprint 3 Planning

La reunión de planificación del tercer *sprint* se llevó a cabo el 3 de abril de 2024, con una duración aproximada de dos horas. En ella, se realizó un análisis de los puntos historia utilizando la técnica *Planning Poker*, véase en el capítulo 4, para estimar las historias de usuario planificadas para el *sprint*.

En consecuencia, la planificación quedaría de la siguiente manera, según se muestra en la Tabla 5.18.

Historia de usuario	Estimación Puntos/Historia	Estimación Horas
H9: Pestaña de cabecera	3	24
H10: Autoguardado	1	8
H11: Opción de guardar informe y guardar al salir	1	8
H12: Pestaña tipo y finalización de servicio	6	48
H13: Subida de informe	1	8
H14: Pestaña Anamnesis	4,5	36

Tabla 5.18: Planificación del *Sprint* 3.

Durante el *Sprint Planning*, las historias de usuario fueron desglosadas en tareas, y se estimó individualmente el tiempo requerido para cada una de ellas en horas, véase Tabla 5.19.

Tareas	Estimación Horas
H9: Maquetación	15
H9: Codificación	7
H9: Pruebas	2
H10: Maquetación	4
H10: Codificación	3
H10: Pruebas	1
H11: Maquetación	5
H11: Codificación	2
H11: Pruebas	1
H12: Maquetación	22
H12: Codificación	22
H12: Pruebas	4
H13: Maquetación	1
H13: Codificación	6
H13: Pruebas	1
H14: Maquetación	20
H14: Codificación	15
H14: Pruebas	1

Tabla 5.19: Planificación del *Sprint* 3 en tareas/horas.

5.5.3. Desarrollo por historias del Sprint 3

En esta sección se abordará el desarrollo del *sprint*, desglosando la implementación de cada una de las historias de usuario estimadas.

5.5.3.1. H9: Pestaña de cabecera

En la Tabla 5.20, se presentan las tareas y su estimación correspondiente del *sprint* anterior para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero crear la pestaña Cabecera en el módulo de creación y edición de informes para introducir su información correspondiente.”

Tareas	Estimación Horas
H9: Maquetación	15
H9: Codificación	7
H9: Pruebas	2

Tabla 5.20: Estimación tareas de la historia de usuario H9: Pestaña de cabecera.

El objetivo de esta historia de usuario es implementar la pestaña “Cabecera” del módulo de edición y creación de informes para que el usuario pueda introducir información básica del incidente, paciente y persona de referencia para el paciente.

Cabe destacar que cada una de las pestañas constituirá un nuevo módulo VIPER que se insertará en la parte inferior de la interfaz del módulo de edición y creación de informes (ver Figura 5.19). En otras palabras, la vista del módulo de edición actuará como vista principal o “padre”, mientras que las vistas de las pestañas funcionarán como vistas secundarias o “hijos”.

En consecuencia, será necesario enlazar los botones creados en la barra de menú de navegación de pestañas con la creación del correspondiente módulo, el cual adaptará su vista a la parte inferior de la interfaz.

Para la implementación de la pestaña “Cabecera” se deberán añadir tres contenedores *UIView*, los cuales albergarán los campos necesarios para introducir la información básica mencionada anteriormente. Además, cada contenedor dispondrá de la funcionalidad para ocultar su contenido y un *UIScrollView* para que el usuario pueda navegar por él.

Durante la edición de algunos campos del formulario, se ha utilizado la librería externa “*DropDown*”¹¹ debido a que *Swift* no proporciona un componente nativo que permita seleccionar entre múltiples opciones. En la Figura 5.21, se puede observar su uso en el campo “Sexo” del paciente.

En la Figura 5.21, se observa la pestaña de “Cabecera” del módulo de edición y creación de formularios.

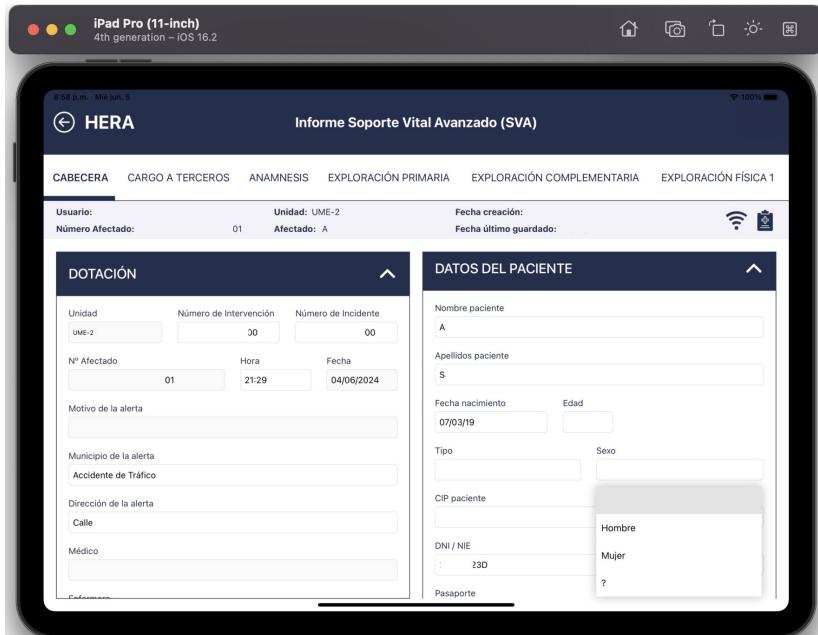


Figura 5.21: Interfaz de pestaña “Cabecera” del módulo de edición y creación de informes de HERA iOS.

5.5.3.2. H10: Autoguardado

En la Tabla 5.21, se presentan las tareas y su estimación correspondiente del *sprint* anterior para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero guardar el informe de manera automática sin tener que pulsar ningún botón.”

Tareas	Estimación Horas
H10: Maquetación	4
H10: Codificación	3
H10: Pruebas	1

Tabla 5.21: Estimación tareas de la historia de usuario H10: Autoguardado.

El objetivo de esta historia de usuario es implementar el guardado del informe de forma automática cada cierto tiempo durante su edición o creación.

Cabe destacar que se ha establecido un canal de comunicación entre la vista padre (módulo de creación y edición de informes) y las vistas hijas (pestañas) mediante el uso de un protocolo. Este protocolo es definido en la vista padre, y especifica los métodos que las vistas hijas deben implementar para poder comunicarse con la vista padre. Cuando la vista padre necesita interactuar con una pestaña, emplea un delegado para invocar los métodos especificados en el protocolo, que las vistas hijas han implementado previamente. Por lo tanto, este protocolo se ha diseñado específicamente para facilitar la transmisión de información entre las dos vistas.

Para implementar la funcionalidad de autoguardado, añadiremos a la definición del protocolo un método para el guardado de datos. Cada pestaña, por su parte, debe implementar este método de guardado de manera que recoja todos los datos ingresados en su formulario. Una vez recopilados, estos datos se envían a la vista padre para su almacenamiento en el informe final. Para garantizar un guardado de los datos cada cierto tiempo, se ha implementado un temporizador en la vista padre. Este temporizador, configurado para activarse cada treinta segundos, invoca mediante un delegado el método de guardado implementado en las pestañas. De esta forma, se asegura la actualización regular de la información almacenada sin que el usuario tenga que pulsar

¹¹<https://cocoapods.org/pods/DropDown>

ningún botón. En las Figuras 5.22 y 5.23, se puede observar tanto la definición del protocolo en la vista padre como su implementación, en este caso, en la pestaña “Cabecera”.

```
protocol ReportGeneralViewControllerDelegate {
    func saveData(autosave : Bool, moveTab : Bool) -> (Bool, Report?)
}
```

Figura 5.22: Definición del protocolo para la funcionalidad del guardado en el módulo de edición y creación de informes de HERA iOS.

```
extension ReportHeaderViewController : ReportGeneralViewControllerDelegate{

    func saveData(autosave : Bool, moveTab : Bool)-> (Bool, Report?) {
        report = getDataReport()
        if isValidDNIorNIE(report.report_json.ID_DNI_NIE_AFFECTED[0]){
            if report.report_json.ID_NUM_AFFECTED[0] != ""{
                self.updateSaveDate()
                presenter?.saveReport(report: report, autosave : autosave, moveTab: moveTab)
                return (true, report)
            }else if report.report_json.ID_NUM_AFFECTED[0] == "" && !autosave{
                if !moveTab{
                    saveReportFailure(error:"Para guardar el informe es necesario introducir un
                        número de afectado.")
                    return (false, nil)
                }else{
                    return (false, nil)
                }
            }else if report.report_json.ID_NUM_AFFECTED[0] == "" && autosave{
                autosaveReportFailure(error: "Informe no guardado. Es necesario introducir un
                    número de afectado.")
                return (false, nil)
            }else{
                return (false, nil)
            }
        }else{
            autosaveReportFailure(error: "Informe no guardado. El número DNI o NIE del afectado
                tiene un formato incorrecto.")
            return (false, nil)
        }
    }
}
```

Figura 5.23: Definición del protocolo para la funcionalidad del guardado en pestaña “Cabecera” de HERA iOS.

Para notificar al usuario sobre el autoguardado, se ha integrado la librería externa “*Toast_Swift*”¹². Esta librería permite implementar mensajes emergentes breves que informen al usuario sobre acciones importantes, como podría ser el guardado automático de datos.

En la Figura 5.24, se muestra un ejemplo de un autoguardado exitoso en la pestaña “Cabecera”. Además, en la Figura 5.25, se presenta un ejemplo de un autoguardado fallido debido a la falta de número de afectado en la misma pestaña.

¹²<https://cocoapods.org/pods/Toast-Swift>

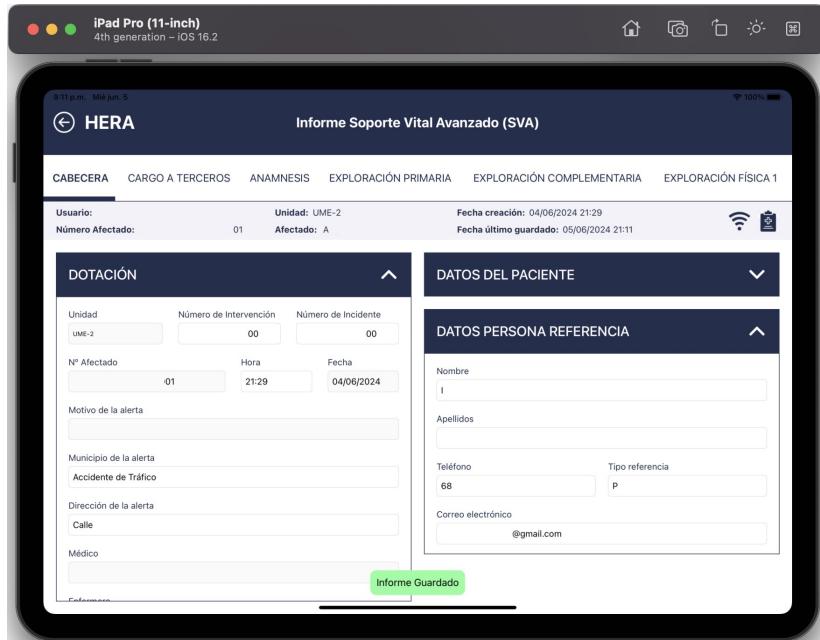


Figura 5.24: Autoguardado exitoso en pestaña “Cabecera” del módulo de edición y creación de informes de HERA iOS.

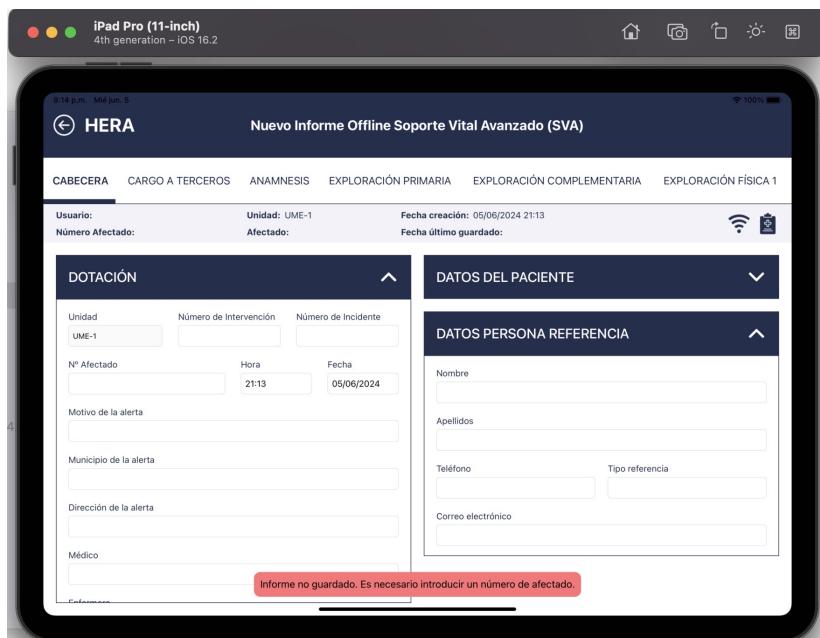


Figura 5.25: Autoguardado fallido en pestaña “Cabecera” del módulo de edición y creación de informes de HERA iOS.

5.5.3.3. H11: Opción de guardar informe y guardar al salir

En la Tabla 5.22, se presentan las tareas y su estimación correspondiente del *sprint* anterior para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero guardar el informe a través de un botón o al salir del mismo.”

Tareas	Estimación Horas
H11: Maquetación	5
H11: Codificación	2
H11: Pruebas	1

Tabla 5.22: Estimación tareas de la historia de usuario **H11: Opción de guardar informe y guardar al salir.**

El objetivo de esta historia de usuario es implementar el guardado del informe a través de un botón y al salir del informe.

Para implementar esta funcionalidad, se empleará el método de guardado definido en el protocolo implementado en la historia anterior. Para proporcionar ambas opciones, se añadirá un menú superior a la interfaz del módulo de creación y edición de informes, el cual facilitará la ejecución de dichas acciones. Una vez que el usuario seleccione alguna de las funcionalidades, se presentará un *UIAlertController* para informarle sobre el éxito o fracaso de la acción.

En la Figura 5.26, se observa menú superior para realizar distintas funcionalidades. Además, en la Figura 5.27 se observa el guardado al salir de un informe.



Figura 5.26: Menú superior donde se encuentra la funcionalidad de guardado en HERA iOS.

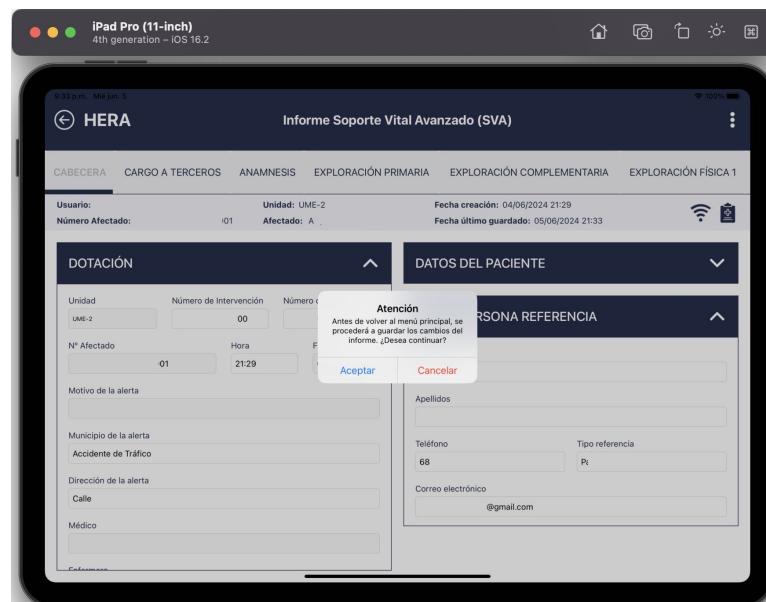


Figura 5.27: Guardado al salir de un informe en HERA iOS.

5.5.3.4. H12: Pestaña tipo y finalización de servicio

En la Tabla 5.23, se presentan las tareas y su estimación correspondiente del *sprint* anterior para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero crear la pestaña Tipo y Finalización de servicio en el módulo de creación y edición de informes para introducir su información correspondiente.”

Tareas	Estimación Horas
H12: Maquetación	22
H12: Codificación	22
H12: Pruebas	4

Tabla 5.23: Estimación tareas de la historia de usuario H12: Pestaña tipo y finalización de servicio.

El objetivo de esta historia de usuario es implementar la pestaña “Tipo y Finalización de servicio” del módulo de edición y creación de informes para que el usuario pueda introducir información básica del tipo de servicio, actuaciones de otros equipos de emergencia, finalización del informe del paciente y unidad, y efectos personales entregados del paciente.

La interfaz de la pestaña “Tipo y Finalización de servicio” está compuesta por cuatro contenedores *UIView*, los cuales albergarán los campos necesarios para introducir la información mencionada anteriormente. Además, cada contenedor dispondrá de la funcionalidad para ocultar su contenido y un *UIScrollView* para que el usuario pueda navegar por él. Es importante destacar que en esta pestaña se empleará la librería previamente mencionada, “*DropDown*”, para determinados campos.

Los campos del contenedor “Otros actuantes” requieren la capacidad de selección múltiple, una funcionalidad que no se encuentra disponible de forma nativa en *Swift*. Por ello, fue necesario desarrollar un componente personalizado para satisfacer esta necesidad. Este componente, diseñado para ser reutilizable, se comunicará con la pestaña correspondiente mediante un protocolo, a través del cual transmitirá los valores seleccionados por el usuario. En la Figura 5.28, se observa el componente reutilizable “*MultipleSelectionView*” al pulsar sobre el campo “Otros actuantes no SUREM”.

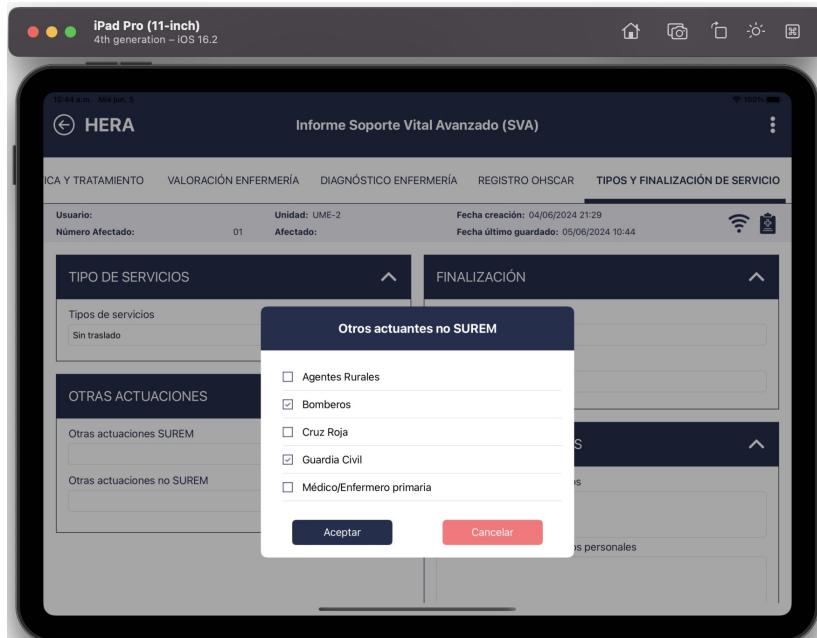


Figura 5.28: Componente *MultipleSelectionView* en pestaña “Tipo y Finalización de servicio” del módulo de edición y creación de informes de HERA iOS.

En la Figura 5.29, podemos observar en el simulador de XCode la pestaña “Tipo y Finalización de servicio” de HERA iOS.

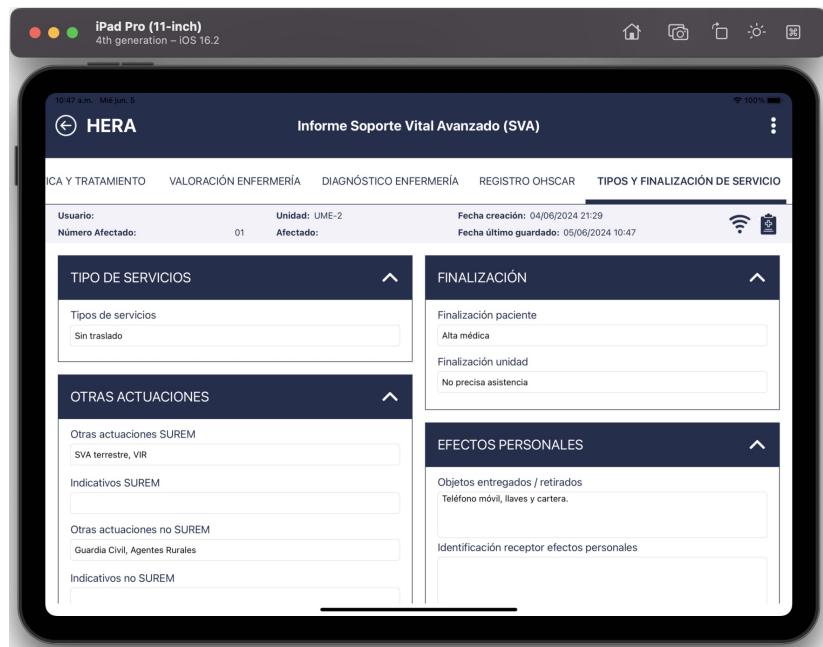


Figura 5.29: Interfaz de pestaña “Tipo y Finalización de servicio” del módulo de edición y creación de informes de **HERA iOS**.

5.5.3.5. H13: Subida de informe

En la Tabla 5.24, se presentan las tareas y su estimación correspondiente del *sprint* anterior para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero enviar el informe completo o parcial al hospital (SITREM).”

Tareas	Estimación Horas
H13: Maquetación	1
H13: Codificación	6
H13: Pruebas	1

Tabla 5.24: Estimación tareas de la historia de usuario **H13: Subida de informe**.

El objetivo de esta historia de usuario es implementar el envío de informe completo o parcial al hospital vía servicio web.

Con el propósito de establecer la comunicación entre la pestaña y el módulo de edición y creación de informes para el envío, se empleará el mismo protocolo utilizado para el guardado. Sin embargo, dado que el envío de informe puede efectuarse en su totalidad o de manera parcial, en función de la opción seleccionada, la pestaña procederá a enviar todos los campos o sólo los propios, a la vista padre (módulo de edición y creación de informes), quien se encargará de llevar a cabo el proceso de envío.

Para poder realizar el envío, se procederá a implementar una petición al servicio de *backend* con el fin de mandarle el informe completo o parcial, dependiendo de la opción seleccionada por el usuario. Dicha petición se efectuará a través de un *endpoint* de los servicios actuales de HERA. Tras enviarla, se recibirá una respuesta que se le presentará al usuario en forma de *UIAlertController*, para informarle de la realización exitosa o no de la acción.

En la Figura 5.26, podemos observar en el simulador de XCode el menú de la interfaz del módulo de edición y creación de informes, donde el usuario podrá elegir entre el envío parcial (Enviar Pre-informe) o el envío en su totalidad (Enviar Informe).

5.5.4. Sprint 3 Review y Conclusión

Se planificaron un total de seis historias de usuario para este *sprint*. En la Figura 5.30, se muestra el gráfico *Burndown* del *sprint* proporcionado por Jira, donde se destaca que se completaron un total de cinco historias de usuario, aunque dos de ellas fueron reabiertas. La historia de usuario H12 fue reabierta debido a la identificación

de defectos ocasionados por la omisión de algunas funcionalidades, las cuales no pudieron ser consideradas debido a la indisponibilidad de la aplicación de Android. La historia de usuario H13 también fue reabierta, ya que no pudo ser probada por la falta de disponibilidad de los servicios *backend* de HERA, lo que ocasionó un bloqueo.

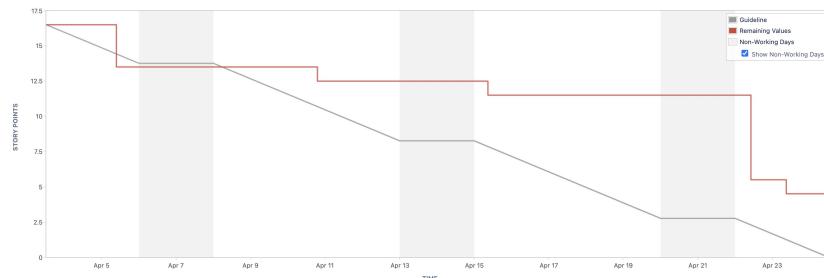


Figura 5.30: Gráfico Burndown proporcionado por Jira del *sprint* 3 de HERA iOS.

En consecuencia, tras realizar la revisión, se determinó que se habían completado un total de tres historias de usuario. Por lo tanto, para el próximo *sprint* quedaría pendiente:

- H12: Pestaña tipo y finalización de servicio
 - Tipo de servicio: mostrar centro emisor y receptor según la opción seleccionada.
 - Tipo de servicio: botón Obtener centro emisor/receptor.
 - Finalización paciente: mostrar según la opción seleccionada campos dni, datos testigos o causa anulación.
- H13: Subida de informe
 - Pruebas.
- H14: Pestaña Anamnesis

Como conclusión de este *sprint*, se siguen produciendo avances significativos, pero aún no se encuentra la estimación perfecta. Por lo tanto, en el próximo *sprint*, será necesario seguir refinando la estimación. La dificultad para refinar la estimación se debe en parte a los bloqueos continuos de los servicios *backend* de HERA que retrasan al desarrollador, como se mencionó anteriormente. Además, se suma la falta de disponibilidad de la aplicación base de Android, en la cual se inspira HERA iOS, para probar las funcionalidades. Para el próximo *sprint*, se mantendrán conversaciones con el equipo *backend* de HERA y el equipo de Android para garantizar su disponibilidad durante el desarrollo.

5.6. SPRINT 4

El inicio del cuarto *sprint* comenzó el 24 de abril de 2024, concluyendo el 14 de mayo de 2024. Por lo tanto, la duración total del *sprint* fue de aproximadamente tres semanas, equivalentes a 15 días laborales.

5.6.1. Objetivo general del Sprint 4

El propósito de este *sprint* era completar la pestaña “Tipo y Finalización de servicio”, probar la subida de informe y comenzar con el desarrollo de las pestañas “Anamnesis” y “Exploración Primaria”.

5.6.2. Sprint 4 Planning

La reunión de planificación del cuarto *sprint* se llevó a cabo el 24 de abril de 2024, con una duración aproximada de dos horas. En ella, se realizó un análisis de los puntos historia utilizando la técnica *Planning Poker*, véase en el capítulo 4, para estimar las historias de usuario planificadas para el *sprint*.

En consecuencia, la planificación quedaría de la siguiente manera, según se muestra en la Tabla 5.25.

Durante el *Sprint Planning*, las historias de usuario fueron desglosadas en tareas, y se estimó individualmente el tiempo requerido para cada una de ellas en horas, véase Tabla 5.26.

5.6.3. Desarrollo por historias del Sprint 4

En esta sección se abordará el desarrollo del *sprint*, desglosando la implementación de cada una de las historias de usuario estimadas.

Historia de usuario	Estim. Puntos/Historia	Estimación Horas
H12: Pestaña tipo y finalización de servicio	6	48
H13: Subida de informe	1	8
H14: Pestaña Anamnesis	4,5	36
H15: Pestaña Exploración Primaria	3	24

Tabla 5.25: Planificación del *Sprint 4*.

Tareas	Estimación Horas
H12: Maquetación	22
H12: Codificación	22
H12: Pruebas	4
H13: Maquetación	1
H13: Codificación	6
H13: Pruebas	1
H14: Maquetación	20
H14: Codificación	15
H14: Pruebas	1
H15: Maquetación	13
H15: Codificación	10
H15: Pruebas	1

Tabla 5.26: Planificación del *Sprint 4* en tareas/horas.

5.6.3.1. H12: Pestaña tipo y finalización de servicio

En la Tabla 5.23, se presentan las tareas y su estimación correspondiente del *sprint* anterior para la implementación de esta historia de usuario. A continuación, se proporciona su descripción como recordatorio:

“Como usuario, quiero crear la pestaña Tipo y Finalización de servicio en el módulo de creación y edición de informes para introducir su información correspondiente.”

Tras finalizar el *sprint* anterior, quedaba pendiente la implementación de las funcionalidades:

- Tipo de servicio: mostrar centro emisor y receptor según la opción seleccionada.
- Tipo de servicio: botón Obtener centro emisor/receptor.
- Finalización paciente: mostrar según la opción seleccionada DNI, datos testigos o causa anulación.

Para añadir la funcionalidad de mostrar centro emisor y receptor según la opción seleccionada en el campo Tipo y Servicio, se añadieron dos contenedores *UIViews*, uno para cada centro, junto con sus respectivos campos. Posteriormente, se debía controlar la visibilidad de estos contenedores según la opción seleccionada en el campo. Si la opción seleccionada es “Primario”, solo se mostrará el contenedor del receptor; si la opción es “Entre centros sanitarios”, se mostrarán ambos contenedores. Otro aspecto que se debía controlar era el filtrado de los nombres de los centros según su municipio y tipo, basado en la opción seleccionada previamente.

Otra funcionalidad que faltaba del contenedor “Tipo de servicio” era la del botón “Obtener centro emisor/receptor”. Al pulsar sobre él, los campos se llenarán automáticamente sin necesidad de introducirlos manualmente. Para lograr esto, se debe implementar una solicitud a los servicios *backend* de HERA. Una vez recibida la respuesta, se debe parsear para obtener la información de los centros.

En la Figura 5.31, se muestra un ejemplo del filtrado donde al pulsar sobre el campo centro sólo aparecerán disponibles los centros públicos de Murcia, y también se muestra el botón de “Obtener centro emisor/receptor”.

La última funcionalidad pendiente en esta pestaña consistía en mostrar, según la opción seleccionada en “Fin de paciente”, los campos correspondientes a DNI, datos de testigos o causa de anulación. Para ello, se añadirían los campos correspondientes en el contenedor “Finalización” y se debía controlar su visibilidad según la opción seleccionada en “Fin de paciente”. Si la opción seleccionada es “Alta voluntaria”, solo se mostrará el

TIPO DE SERVICIOS

Tipos de servicios
Entre centros sanitarios

Centro emisor

Municipio
Murcia

Tipo de centro
Hospital público

Centro

Servicio
UCI

Obtener centro emisor / receptor

Figura 5.31: Funcionalidades de contenedor “Tipo de servicio” de pestaña “Tipo y finalización de servicio” de HERA iOS.

campo DNI; si la opción es “Rechaza Asistencia” o “Rechaza Traslado”, se mostrará DNI y datos; si la opción es “Servicio nulo”, se mostrará causa de anulación.

En la Figura 5.32, se observa en el simulador de XCode como quedaría la interfaz de “Tipo y Finalización de servicio” tras seleccionar, por ejemplo, en el campo “Fin de paciente” la opción “Rechaza Asistencia”.

FINALIZACIÓN

Finalización paciente
Rechaza Asistencia

DNI / NIE paciente

Datos identificativos testigo

Finalización unidad
No precisa asistencia

Figura 5.32: Funcionalidad de contenedor “Fin de paciente” de pestaña “Tipo y finalización de servicio” de HERA iOS.

5.6.3.2. H13: Subida de informe

En la Tabla 5.24, se presentan las tareas y su estimación correspondiente del *sprint* anterior para la implementación de esta historia de usuario. A continuación, se proporciona su descripción como recordatorio:

“Como usuario, quiero enviar el informe completo o parcial al hospital (SITREM).”

Tras finalizar el *sprint* anterior, quedaba pendiente probar la funcionalidad de envío cuando los servicios de HERA estuvieran disponibles. Dado que en ese momento tampoco estaban disponibles, se consideró crear una historia de usuario para probarlo en el futuro.

5.6.3.3. H14: Pestaña Anamnesis

En la Tabla 5.27, se presentan las tareas y su estimación correspondiente del *sprint* anterior para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero crear la pestaña Anamnesis en el módulo de creación y edición de informes para introducir su información correspondiente.”

Tareas	Estimación Horas
H14: Maquetación	20
H14: Codificación	15
H14: Pruebas	1

Tabla 5.27: Estimación tareas de la historia de usuario H14: Pestaña Anamnesis.

El objetivo de esta historia de usuario es implementar la pestaña “Anamnesis” del módulo de edición y creación de informes para que el usuario pueda introducir información básica del paciente sobre alergias, antecedentes patológicos, hábitos tóxicos, enfermedad y medicación actual.

La interfaz de la pestaña “Anamnesis” está compuesta por cinco contenedores *UIView*, los cuales albergarán los campos necesarios para introducir la información básica del paciente mencionada anteriormente. Además, cada contenedor dispondrá de la funcionalidad para ocultar su contenido y un *UIScrollView* para que el usuario pueda navegar por él. Es importante destacar que en esta pestaña se empleará la librería previamente mencionada, “*DropDown*”, y el componente reutilizable “*MultipleSelectionView*”, desarrollado anteriormente, para determinados campos.

El contenedor “Medicación Actual” requiere la capacidad de crear una tabla para almacenar los valores relacionados con la dosis del medicamento que el usuario introduzca. Esta funcionalidad no está disponible de forma nativa en *Swift*, por lo que fue necesario desarrollar un componente personalizado para satisfacer esta necesidad. Este componente se comunicará con la pestaña correspondiente mediante un protocolo, a través del cual transmitirá los valores introducidos por el usuario, añadiendo una fila a la tabla con dichos valores. Además el usuario tendrá la opción de borrar esa medicación mediante el ícono “papelera”. En la Figura 5.33, se observa el componente “*FillFieldsView*” al pulsar sobre el ícono de añadir medicación.

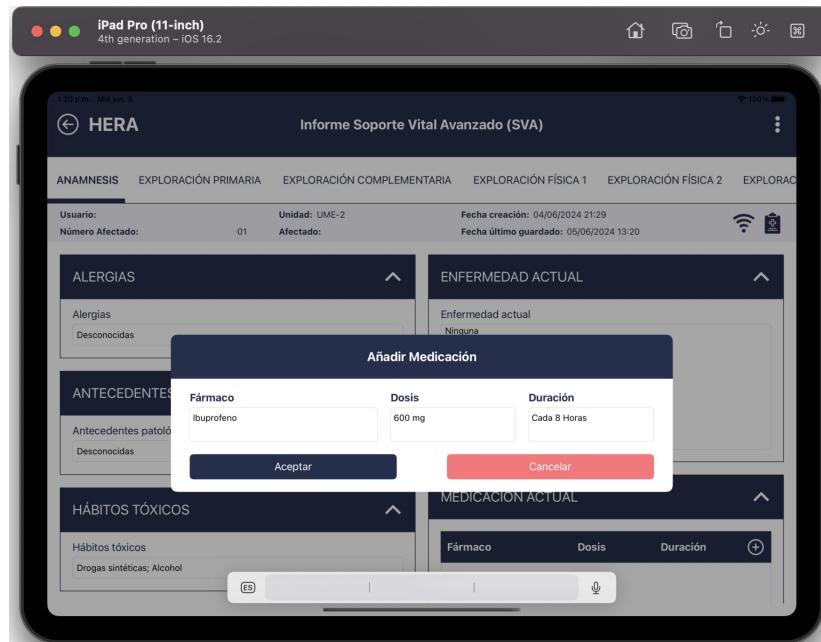


Figura 5.33: Componente *FillFieldsView* en pestaña “Anamnesis” del módulo de edición y creación de informes de **HERA iOS**.

En la Figura 5.34, podemos observar en el simulador de XCode la pestaña “Anamnesis” de **HERA iOS**.

5.6.3.4. H15: Pestaña Exploración Primaria

En la Tabla 5.28, se presentan las tareas y su estimación correspondiente del *sprint* anterior para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero crear la pestaña Exploración Primaria en el módulo de creación y edición de informes para introducir su información correspondiente.”

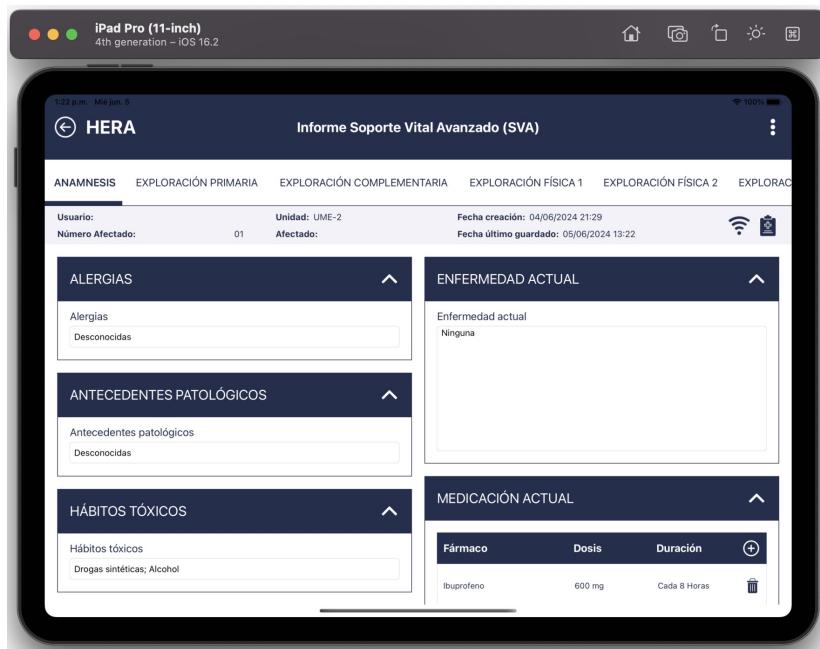


Figura 5.34: Interfaz de pestaña “Anamnesis” del módulo de edición y creación de informes de HERA iOS.

Tareas	Estimación Horas
H15: Maquetación	13
H15: Codificación	10
H15: Pruebas	1

Tabla 5.28: Estimación tareas de la historia de usuario H15: Pestaña Exploración Primaria.

El objetivo de esta historia de usuario es implementar la pestaña “Exploración Primaria” del módulo de edición y creación de informes para que el usuario pueda introducir información básica del paciente sobre su estado general, vías respiratorias, circulación, estado de vías aéreas, estado neurológico y pupilas.

La interfaz de la pestaña “Exploración Primaria” está compuesta por cinco contenedores *UIView*, los cuales albergarán los campos necesarios para introducir la información básica del paciente mencionada anteriormente. Además, cada contenedor dispondrá de la funcionalidad para ocultar su contenido y un *UIScrollView* para que el usuario pueda navegar por él. Es importante destacar que en esta pestaña se empleará la librería previamente mencionada, “*DropDown*”, y el componente reutilizable “*MultipleSelectionView*”, desarrollado anteriormente, para determinados campos.

En el contenedor “Estado general”, el usuario tendrá la opción de pulsar un botón para establecer un estado normal del paciente en toda la pestaña. Esto permitirá llenar automáticamente campos como la ventilación, temperatura, estado neurológico, entre otros. En la Figura 5.35, se muestra la funcionalidad después de que el usuario haya presionado el botón.

En el contenedor “Pupilas”, se implementó una funcionalidad especial para calcular la simetría de las pupilas en función de su estado. El usuario puede cambiar el estado de las pupilas pulsando sobre la imagen correspondiente. Para detectar este evento, se añadió un *UITapGestureRecognizer* a las imágenes de las pupilas. Habrá tres imágenes de pupilas que representan los estados normal, grande (Midriática) y pequeña (Miótica), las cuales cambiarán de forma cíclica en ese orden cada vez que el usuario pulse sobre ellas.

La simetría se calculará para el estado inicial y final automáticamente basándose en las tres imágenes posibles de cada pupila siguiendo este criterio:

- Si pupila izquierda y derecha son iguales: “Isocóricas”.
- Si la derecha es más grande que la izquierda: “Anisocóricas D>I”.
- Si la derecha es más pequeña que la izquierda: “Anisocóricas I>D”.

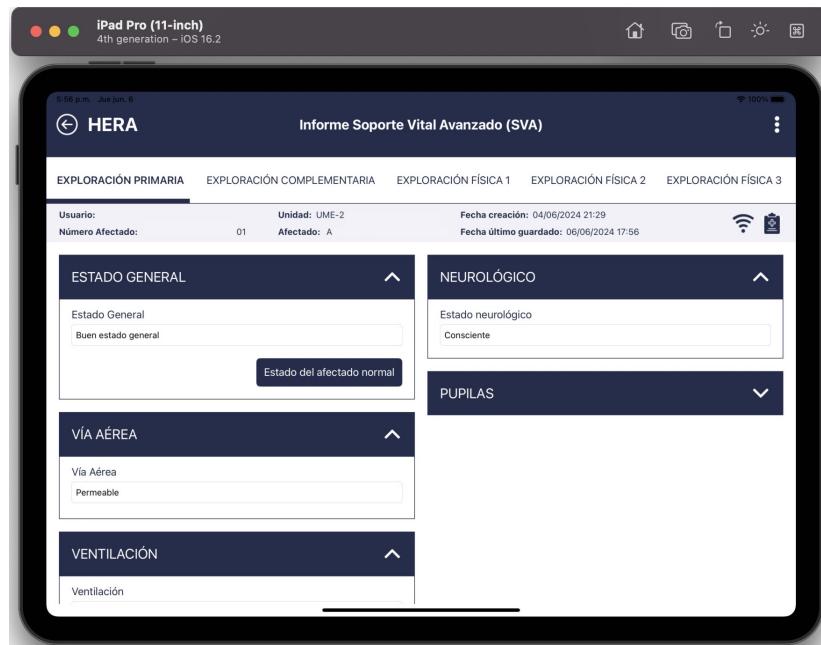


Figura 5.35: Funcionalidad botón “Estado general” de pestaña “Exploración Primaria” del módulo de edición y creación de informes de **HERA iOS**.

En la Figura 5.36, podemos observar en el simulador de XCode la funcionalidad del contenedor “Pupilas” de la pestaña “Exploración Primaria” de **HERA iOS**.

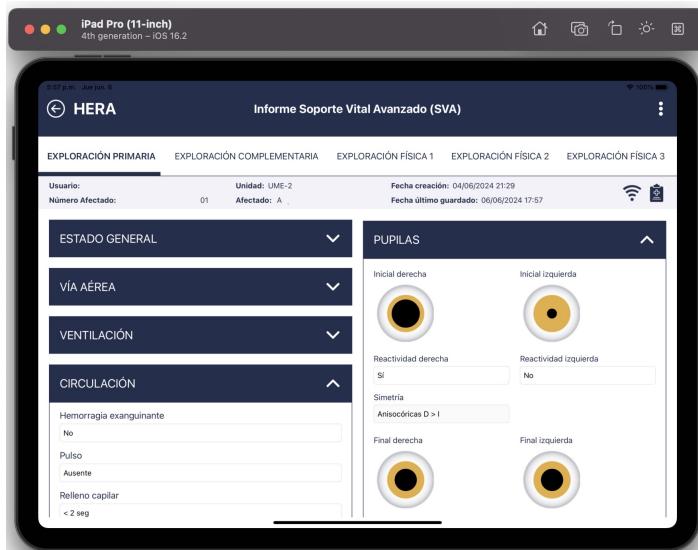


Figura 5.36: Funcionalidad del contenedor “Pupilas” de pestaña “Exploración Primaria” del módulo de edición y creación de informes de **HERA iOS**.

5.6.4. Sprint 4 Review y Conclusión

Se planificaron un total de cuatro historias de usuario para este *sprint*. En la Figura 5.37, se muestra el gráfico *Burndown* del *sprint* proporcionado por Jira, donde se puede observar que, tras realizar la revisión, se determinó que se completaron todas las historias de usuario estimadas para este *sprint*.

Como conclusión de este *sprint*, el equipo logró una estimación perfecta y se pudo completar todo lo planificado. Para el próximo *sprint*, se buscará mantener este nivel de precisión y productividad.

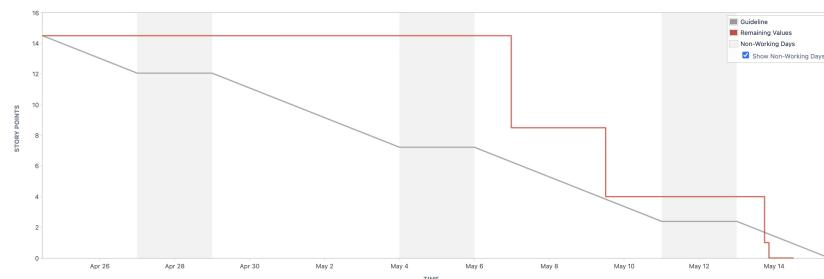


Figura 5.37: Gráfico Burndown proporcionado por Jira del *sprint 4* de HERA iOS.

5.7. SPRINT 5

El inicio del quinto *sprint* comenzó el 14 de mayo de 2024, concluyendo el 4 de junio de 2024. Por lo tanto, la duración total del *sprint* fue de aproximadamente tres semanas, equivalentes a 15 días laborales.

5.7.1. Objetivo general del Sprint 5

El propósito de este *sprint* era implementar las pestañas “Exploración Física 1”, “Exploración Física 2” y “Exploración Física 3”. Además, se deberían probar las peticiones de los servicios web y desarrollar el acceso a la plataforma *offline*.

5.7.2. Sprint 5 Planning

La reunión de planificación del quinto *sprint* se llevó a cabo el 14 de mayo de 2024, con una duración aproximada de dos horas. En ella, se realizó un análisis de los puntos historia utilizando la técnica *Planning Poker*, véase en el capítulo 4, para estimar las historias de usuario planificadas para el *sprint*.

En consecuencia, la planificación quedaría de la siguiente manera, según se muestra en la Tabla 5.29.

Historia de usuario	Estimación Puntos/Historia	Estimación Horas
H16: Pestaña Exploración física 1	3	24
H17: Pestaña Exploración física 2	3	24
H18: Pestaña Exploración física 3	3	24
H19: Pruebas WS	3	24
H20: Acceso offline	2	16

Tabla 5.29: Planificación del *Sprint 5*.

Durante el *Sprint Planning*, las historias de usuario fueron desglosadas en tareas, y se estimó individualmente el tiempo requerido para cada una de ellas en horas, véase Tabla 5.30.

5.7.3. Desarrollo por historias del Sprint 5

En esta sección se abordará el desarrollo del *sprint*, desglosando la implementación de cada una de las historias de usuario estimadas.

5.7.3.1. H16: Pestaña Exploración física 1

En la Tabla 5.31, se presentan las tareas y su estimación correspondiente para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero crear la pestaña Exploración física 1 en el módulo de creación y edición de informes para introducir su información correspondiente.”

El objetivo de esta historia de usuario es implementar la pestaña “Exploración física 1” del módulo de edición y creación de informes para que el usuario pueda introducir información del paciente sobre su aparato respiratorio y posibles síntomas, y su aparato cardiovascular.

La interfaz de la pestaña “Exploración física 1” está compuesta por tres contenedores *UIView*, los cuales albergarán los campos necesarios para introducir la información del paciente mencionada anteriormente y sus síntomas. Además, cada contenedor dispondrá de la funcionalidad para ocultar su contenido y un *UIScrollView* para que el usuario pueda navegar por él. Es importante destacar que en esta pestaña se empleará la librería

Tareas	Estimación Horas
H16: Maquetación	13
H16: Codificación	10
H16: Pruebas	1
H17: Maquetación	13
H17: Codificación	10
H17: Pruebas	1
H18: Maquetación	12
H18: Codificación	10
H18: Pruebas	2
H19: Pruebas WS Módulo Home	8
H19: Pruebas WS Módulo Bandeja Incidentes	8
H19: Pruebas WS Módulo Edición y creación de informes	8
H20: Maquetación	7
H20: Codificación	7
H20: Pruebas	2

Tabla 5.30: Planificación del *Sprint 5* en tareas/horas.

Tareas	Estimación Horas
H16: Maquetación	13
H16: Codificación	10
H16: Pruebas	1

Tabla 5.31: Estimación tareas de la historia de usuario **H16: Pestaña Exploración física 1**.

previamente mencionada, “*DropDown*”, y el componente reutilizable “*MultipleSelectionView*”, desarrollado anteriormente, para determinados campos.

En el contenedor “Exploración Física 1”, se implementó una funcionalidad especial que permite al usuario establecer los síntomas del paciente en el aparato respiratorio. Esto se realiza colocando “pegatinas” de síntomas en una imagen que simboliza los pulmones del paciente, la cual está dividida en regiones. De esta manera, el usuario podrá representar los síntomas respiratorios de manera más visual.

Para implementar esta funcionalidad, se añadirán imágenes para cada tipo de síntoma, a los cuales se les agregará un *UITapGestureRecognizer* para reconocer la pulsación del usuario. Posteriormente, el usuario podrá tocar la imagen de los pulmones, y el síntoma seleccionado se posicionará en las coordenadas del toque.

En la Figura 5.38, podemos observar en el simulador de XCode la pestaña “Exploración Física 1” de **HERA iOS**.

5.7.3.2. H17: Pestaña Exploración física 2

En la Tabla 5.32, se presentan las tareas y su estimación correspondiente para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“*Como usuario, quiero crear la pestaña Exploración física 1 en el módulo de creación y edición de informes para introducir su información correspondiente.*”

Tareas	Estimación Horas
H17: Maquetación	13
H17: Codificación	10
H17: Pruebas	1

Tabla 5.32: Estimación tareas de la historia de usuario **H17: Pestaña Exploración física 2**.

El objetivo de esta historia de usuario es implementar la pestaña “Exploración física 2” del módulo de edición y creación de informes para que el usuario pueda introducir información del paciente sobre su abdomen y posibles síntomas, y una segunda evaluación de su estado neurológico.

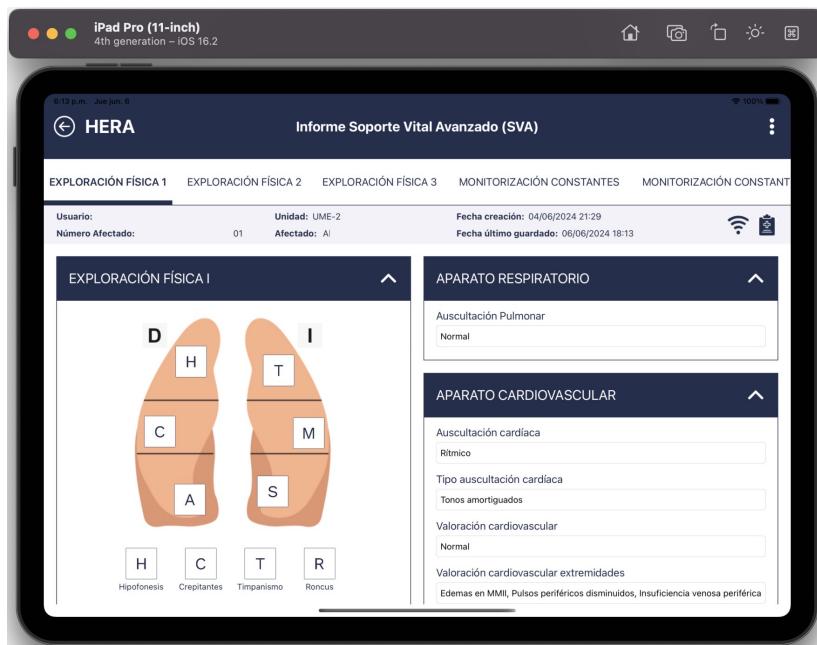


Figura 5.38: Interfaz de pestaña “Exploración Física 1” del módulo de edición y creación de informes de **HERA iOS**.

La interfaz de la pestaña “Exploración física 2” está compuesta por tres contenedores *UIView*, los cuales albergarán los campos necesarios para introducir la información del paciente mencionada anteriormente y sus síntomas. Además, cada contenedor dispondrá de la funcionalidad para ocultar su contenido y un *UIScrollView* para que el usuario pueda navegar por él. Es importante destacar que en esta pestaña se empleará la librería previamente mencionada, “*DropDown*”, y el componente reutilizable “*MultipleSelectionView*”, desarrollado anteriormente, para determinados campos.

En el contenedor “Exploración Física 2”, se ha integrado la funcionalidad desarrollada en la historia de usuario anterior, permitiendo al usuario establecer los síntomas del paciente, pero esta vez focalizado en el abdomen. De igual manera, se logra colocando “pegatinas” de síntomas en una imagen que representa el abdomen del paciente, dividido en regiones. De esta manera, el usuario podrá representar los síntomas del abdomen de manera más visual.

En la Figura 5.39, podemos observar en el simulador de XCode la pestaña “Exploración Física 2” de **HERA iOS**.

5.7.3.3. H18: Pestaña Exploración física 3

En la Tabla 5.33, se presentan las tareas y su estimación correspondiente para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero crear la pestaña Exploración física 1 en el módulo de creación y edición de informes para introducir su información correspondiente.”

Tareas	Estimación Horas
H18: Maquetación	12
H18: Codificación	10
H18: Pruebas	2

Tabla 5.33: Estimación tareas de la historia de usuario **H18: Pestaña Exploración física 3**.

El objetivo de esta historia de usuario es implementar la pestaña “Exploración física 3” del módulo de edición y creación de informes para que el usuario pueda introducir información del paciente sobre su aparato locomotor y posibles síntomas, oído y boca.

La interfaz de la pestaña “Exploración física 3” está compuesta por cuatro contenedores *UIView*, los cuales albergarán los campos necesarios para introducir la información del paciente mencionada anteriormente y sus

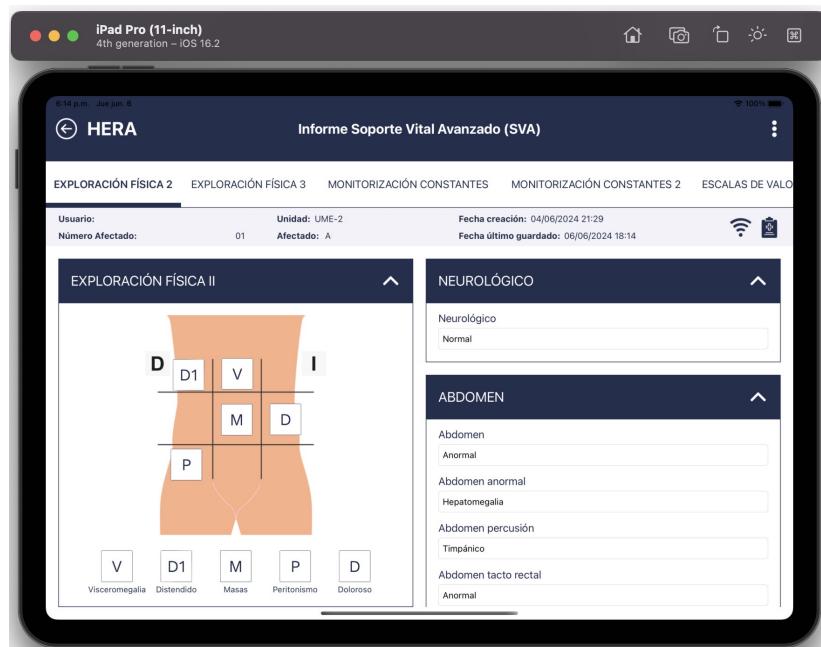


Figura 5.39: Interfaz de pestaña “Exploración Física 2” del módulo de edición y creación de informes de **HERA iOS**.

síntomas. Además, cada contenedor dispondrá de la funcionalidad para ocultar su contenido y un *UIScrollView* para que el usuario pueda navegar por él. Es importante destacar que en esta pestaña se empleará la librería previamente mencionada, “*DropDown*”, y el componente reutilizable “*MultipleSelectionView*”, desarrollado anteriormente, para determinados campos.

En el contenedor “Exploración Física 3”, se ha integrado la funcionalidad desarrollada en la historia de usuario anterior, permitiendo al usuario establecer los síntomas del paciente, pero esta vez focalizado en el aparato locomotor. De igual manera, se logra colocando “pegatinas” de síntomas en una imagen que representa el cuerpo del paciente por delante y por detrás.

La imagen final generada deberá enviarse junto al informe. En consecuencia, será necesario combinar todas las pegatinas y la imagen de fondo en una sola, con el fin de codificarla en base 64 para poder enviarla. Para implementarlo, primero se crea un contexto gráfico utilizando la función *UIGraphicsBeginImageContext* de *Swift*, escalando la imagen de fondo del cuerpo para mantener sus proporciones. Posteriormente, se agregan todas las imágenes de los síntomas en las coordenadas correspondientes de la imagen del cuerpo, para fusionarlas utilizando la función *UIGraphicsGetImageFromCurrentImageContext* de *Swift*. Finalmente, se cierra el contexto gráfico con la función *UIGraphicsEndImageContext* y codificando la imagen en base64 estará lista para ser enviada.

En la Figura 5.40, podemos observar en el simulador de XCode la pestaña “Exploración Física 3” de **HERA iOS**.

5.7.3.4. H19: Pruebas WS

En la Tabla 5.34, se presentan las tareas y su estimación correspondiente para la implementación de esta historia de usuario.

Tareas	Estimación Horas
H19: Pruebas WS Módulo Home	8
H19: Pruebas WS Módulo Bandeja Incidentes	8
H19: Pruebas WS Módulo Edición y creación de informes	8

Tabla 5.34: Estimación tareas de la historia de usuario **H19: Pruebas WS**

El objetivo de esta historia de usuario es proporcionar al desarrollador tiempo adicional para probar las peticiones al servicio y realizar modificaciones en caso de detectar algún defecto. Esto es debido a que la mayor

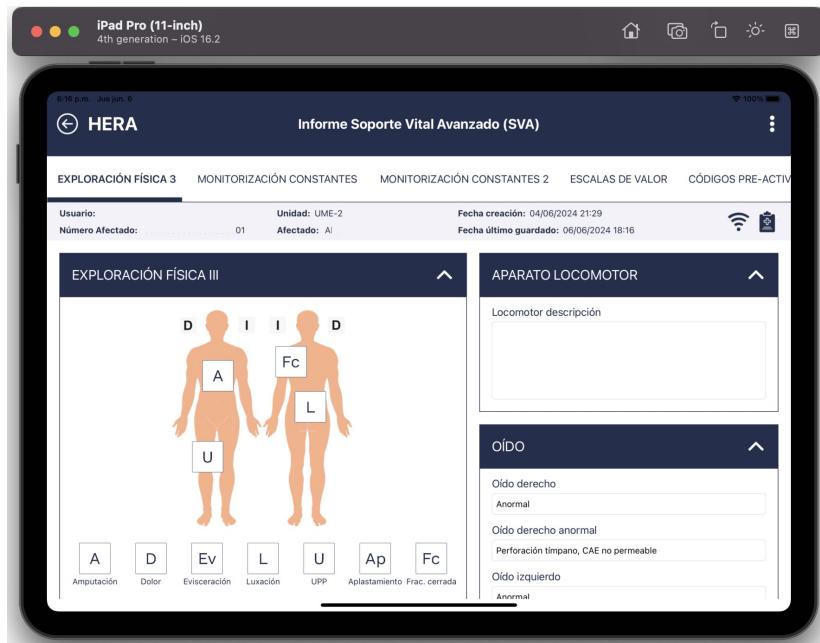


Figura 5.40: Interfaz de pestaña “Exploración Física 3” del módulo de edición y creación de informes de **HERA iOS**.

parte del tiempo los servicios de HERA no están disponibles y no se pudieron probar en el momento de su desarrollo.

5.7.3.5. H20: Acceso offline

En la Tabla 5.35, se presentan las tareas y su estimación correspondiente para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero iniciar sesión de forma offline, para acceder a la plataforma en el caso en el que no se puede acceder de forma online.”

Tareas	Estimación Horas
H20: Maquetación	7
H20: Codificación	7
H20: Pruebas	2

Tabla 5.35: Estimación tareas de la historia de usuario **H20: Acceso offline**.

El objetivo de esta historia de usuario es proporcionar un inicio de sesión *offline* a la plataforma, para el caso que no se pueda acceder de forma *online*.

Para llevar a cabo esta historia de usuario, se diseñó una interfaz que incluye un formulario básico de inicio de sesión. Este formulario solicita al usuario un nombre de usuario y una contraseña, manteniendo la misma estructura que la interfaz utilizada para el inicio de sesión *online*.

Para autenticarse, el usuario deberá introducir su nombre de usuario y contraseña habituales. Se compararán estas credenciales con las del inicio de sesión *online*, y si son correctas, se le permitirá el acceso a la plataforma. Este proceso se realiza de la siguiente manera: una vez que el usuario ha iniciado sesión al menos una vez en línea, es crucial almacenar de forma segura sus credenciales para proteger la información confidencial. Esto se logra cifrándolas con una técnica de *hashing* antes de guardarlas en la caché del dispositivo, que ya cuenta con cierto nivel de seguridad. Esta práctica asegura que incluso si alguien obtiene acceso no autorizado al dispositivo, las credenciales permanezcan protegidas y no sean fácilmente descifrables. Posteriormente, en caso de no tener conexión y que se haya cerrado sesión, se verificarán las nuevas credenciales introducidas comparándolas con las almacenadas mediante sus respectivos *hashes*. Es importante destacar que si el usuario no ha iniciado sesión al menos una vez en el dispositivo, no podrá acceder a la plataforma en el caso de no tener conexión.

En la Figura 5.41, se observa en el simulador de XCode la interfaz de inicio de sesión *offline*.

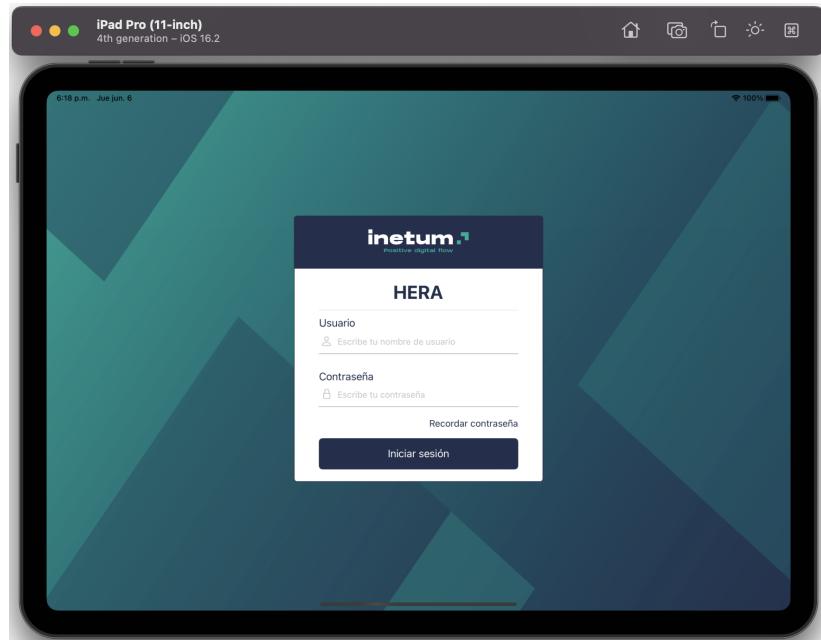


Figura 5.41: Interfaz de inicio de sesión *offline* de HERA iOS.

5.7.4. Sprint 5 Review

Se planificaron un total de cinco historias de usuario para este *sprint*. En la Figura 5.42, se muestra el gráfico *Burndown* del *sprint* proporcionado por Jira, donde se destaca que se completaron un total de cuatro historias de usuario, aunque una de ellas fue reabierta. La historia de usuario H19 también fue reabierta, ya que no pudo ser probada del todo por la falta de disponibilidad de los servicios *backend* de HERA, lo que ocasionó un bloqueo. Además, la historia de usuario H20 no pudo ser cerrada debido a la imposibilidad de probarla, ya que el inicio de sesión *online* no estaba disponible, lo que causó otro bloqueo.

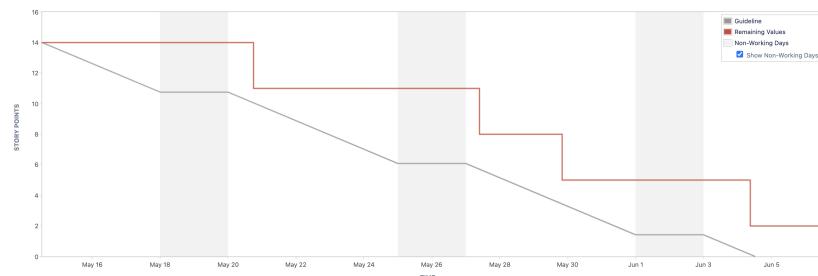


Figura 5.42: Gráfico *Burndown* proporcionado por Jira del *sprint* 5 de HERA iOS.

En consecuencia, tras realizar la revisión, se determinó que se habían completado un total de tres historias de usuario. Por lo tanto, para el próximo *sprint* quedaría pendiente:

- H19. Pruebas WS
 - Pruebas WS Módulo Edición y creación de informes.
- H20. Acceso *offline*
 - Pruebas

Como conclusión de este *sprint*, todas las historias de usuario han sido completadas, pero aún continúan los bloqueos que impiden probarlas. Los continuos bloqueos de los servicios *backend* de HERA han generado retrasos en el desarrollo, por lo que será necesario tomar medidas adicionales. Además, la falta de disponibilidad de los servicios de *login*, mencionado anteriormente, ha complicado la situación. Para el próximo *sprint*, se buscará resolver estos bloqueos y mantener la productividad alcanzada, dado que todas las tareas se han completado.

5.8. DIAGRAMA DE GANTT

En la Figura 5.43, se muestra el diagrama de Gantt del periodo desde que comenzaron las prácticas (05/02/2024) hasta el fin del proyecto. El objetivo de este diagrama es tener una visión general del tiempo empleado en el proyecto.

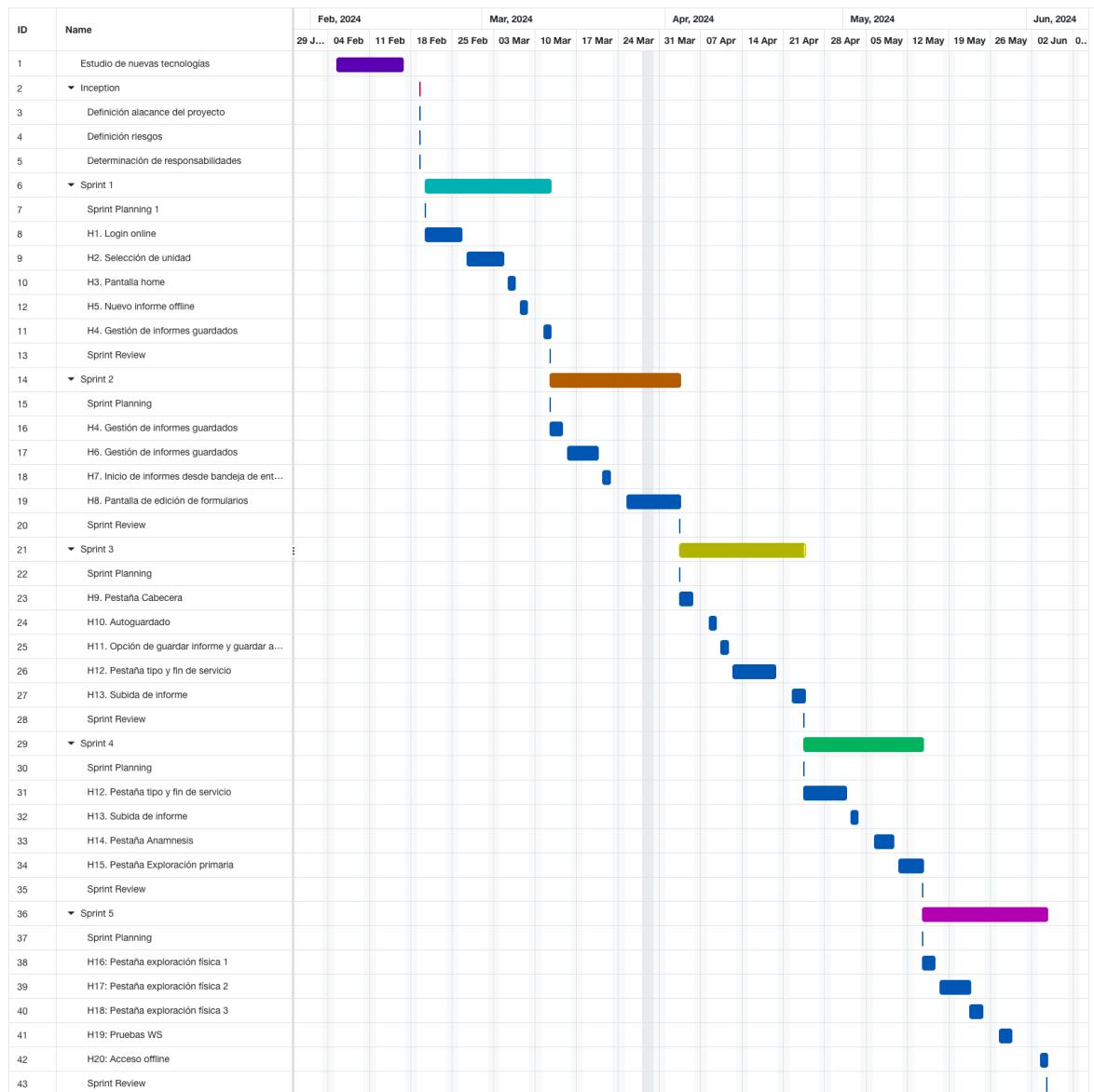


Figura 5.43: Diagrama de Gantt durante el proyecto de HERA iOS.

CAPÍTULO 6

Conclusiones

En este capítulo, se presentarán las conclusiones alcanzadas por el autor tras la realización del TFG, indicando si se han cumplido o no los objetivos planteados en el capítulo 2. Además, se describirá el trabajo actual que se está realizando en este proyecto, así como los próximos avances necesarios para completarlo. Por último, se expondrá una valoración crítica del autor de este TFG.

6.1. CONSECUCIÓN DE OBJETIVOS PARCIALES

En esta sección se analizará el cumplimiento de los objetivos parciales definidos en el capítulo 2, sección 2.2 de este documento, evaluando cuándo y cómo se han alcanzado. En la Tabla 6.1, se muestra un resumen de la satisfacción de los objetivos parciales.

6.1.1. Módulo de acceso a la plataforma y selección de unidad

Cabe destacar que el acceso a la plataforma estaba disponible tanto *online* como *offline*. Por lo tanto, se especificaron dos historias de usuario para completar el módulo de acceso a la plataforma:

- **H1.Login online:** su objetivo era proporcionar el acceso *online* al usuario. Esta historia de usuario se completó durante el *Sprint 1* (sección 5.3), véase en el apartado 5.3.3.1.
- **H20.Acceso offline:** su objetivo era proporcionar el acceso *offline* al usuario. Esta historia de usuario se completó durante el *Sprint 5* (sección 5.7), véase en el apartado 5.7.3.5.

Para completar el módulo de selección de unidad se definió una historia de usuario, denominada **H2: Selección de unidad**. El objetivo de esta historia era proporcionar al usuario la posibilidad de seleccionar una unidad para asignarla al dispositivo. Esta historia de usuario se completó durante el *Sprint 1* (sección 5.3), véase en el apartado 5.3.3.2.

Por lo tanto, se puede inferir que este objetivo parcial fue satisfecho durante el *Sprint 1* (sección 5.3) y *Sprint 5* (sección 5.7).

6.1.2. Módulo de gestión de la edición y creación de informes, tanto online como offline

Cabe destacar que este módulo comprendía la creación y eliminación de informes, edición de los diferentes informes de las distintas pestañas, así como el guardado y la subida de los mismos. Para ello, se definieron las siguientes historias de usuario:

- **H4.Gestión de informes guardados:** su objetivo era proporcionar la visualización y eliminación de informes en el menú principal. Esta historia de usuario se completó durante el *Sprint 1* (sección 5.3), véase en el apartado 5.3.3.4 y *Sprint 2* (sección 5.4), véase en el apartado 5.4.3.1.
- **H5.Nuevo informe offline:** su objetivo era implementar la creación de un informe. Esta historia de usuario se completó durante el *Sprint 1* (sección 5.3), véase en el apartado 5.3.3.5.
- **H8.Pantalla edición de formularios:** su objetivo era implementar la cabecera de navegación del módulo de edición y creación de informes. Esta historia de usuario se completó durante el *Sprint 2* (sección 5.4), véase en el apartado 5.4.3.4.
- **H9.Pestaña de cabecera:** su objetivo era implementarla para que el usuario pudiera introducir información básica del incidente, paciente y persona de referencia para el paciente. Esta historia de usuario se completó durante el *Sprint 3* (sección 5.5), véase en el apartado 5.5.3.1.

- **H10. Autoguardado:** su objetivo era implementar el guardado del informe de forma automática. Esta historia de usuario se completó durante el *Sprint 3* (sección 5.5), véase en el apartado 5.5.3.2.
- **H11. Opción de guardar informe y guardar al salir:** su objetivo era implementar el guardado al salir o manual. Esta historia de usuario se completó durante el *Sprint 3* (sección 5.5), véase en el apartado 5.5.3.3.
- **H12. Pestaña tipo y finalización de servicio:** su objetivo era implementarla para que el usuario pudiera introducir información básica del tipo de servicio, actuaciones de otros equipos de emergencia, finalización del informe del paciente y unidad, y efectos personales entregados del paciente. Esta historia de usuario se completó durante el *Sprint 3* (sección 5.5), véase en el apartado 5.5.3.4 y *Sprint 4* (sección 5.6), véase en el apartado 5.6.3.1.
- **H13. Subida de informe:** su objetivo era implementar la subida del informe. Esta historia de usuario se completó durante el *Sprint 3* (sección 5.5), véase en el apartado 5.5.3.5 y *Sprint 4* (sección 5.6), véase en el apartado 5.6.3.2.
- **H14. Pestaña Anamnesis:** su objetivo era implementarla para que el usuario pudiera introducir información básica del paciente sobre alergias, antecedentes patológicos, hábitos tóxicos, enfermedad y medicación actual. Esta historia de usuario se completó durante el *Sprint 4* (sección 5.6), véase en el apartado 5.6.3.3.
- **H15. Pestaña Exploración primaria:** su objetivo era implementarla para que el usuario pudiera introducir información básica del paciente sobre su estado general, vías respiratorias, circulación, estado de vías aéreas, estado neurológico y pupilas. Esta historia de usuario se completó durante el *Sprint 4* (sección 5.6), véase en el apartado 5.6.3.4.
- **H16. Pestaña Exploración física 1:** su objetivo era implementarla para que el usuario pudiera introducir información básica del paciente sobre su aparato respiratorio y posibles síntomas, y su aparato cardiovascular. Esta historia de usuario se completó durante el *Sprint 5* (sección 5.7), véase en el apartado 5.7.3.1.
- **H17. Pestaña Exploración física 2:** su objetivo era implementarla para que el usuario pudiera introducir información del paciente sobre su abdomen y posibles síntomas, y una segunda evaluación de su estado neurológico. Esta historia de usuario se completó durante el *Sprint 5* (sección 5.7), véase en el apartado 5.7.3.2.
- **H18. Pestaña Exploración física 3:** su objetivo era implementarla para que el usuario pudiera introducir información básica del paciente sobre su aparato locomotor y posibles síntomas, oído y boca. Esta historia de usuario se completó durante el *Sprint 5* (sección 5.7), véase en el apartado 5.7.3.3.

Por lo tanto, se puede inferir que este objetivo parcial fue satisfecho durante todos los *sprints* que duró el desarrollo.

6.1.3. Módulo para la asignación de afectados

Cabe destacar que este módulo comprendía la visualización de los afectados de un incidente, asignación y desasignación de afectados a la unidad, así como la creación de un informe a partir de la información de un afectado. Para ello, se definieron las siguientes historias de usuario:

- **H6. Pantalla de afectados:** su objetivo era implementar la funcionalidad de visualización de afectados de un incidente, así como su asignación y desasignación a la unidad. Esta historia de usuario se completó durante el *Sprint 2* (sección 5.4), véase en el apartado 5.4.3.2.
- **H7. Inicio de informes desde bandeja de entrada:** su objetivo era implementar la creación de un informe a partir de la información del afectado asignado a la unidad. Esta historia de usuario se completó durante el *Sprint 2* (sección 5.4), véase en el apartado 5.4.3.3.

Por lo tanto, se puede inferir que este objetivo parcial fue satisfecho durante el *Sprint 2* (sección 5.4).

6.1.4. Módulo para la integración con distintas fuentes de datos (anamnesis, desfibrilador, etc.)

Cabe destacar que este módulo comprendía la integración con fuentes de datos externas, vía bluetooth, para proporcionar información a las pestañas del informe. Debido a restricciones de tiempo, en el momento de redactar este TFG se está abordando durante el *Sprint 6*. Para completarlo se ha definido, de momento, una historia de usuario, denominada **H21: Integración con dispositivos bluetooth**. Su desarrollo se incluirá en la sección de Trabajo Presente y Futuro (ver [6.4](#)).

Por lo tanto, se puede inferir que este objetivo ha sido satisfecho parcialmente durante el *Sprint 6* (sección [6.4](#)).

Objetivo	Completado	Evidencia
Módulo de acceso a la plataforma y selección de unidad		<i>Sprint 1</i> (sección 5.3) y <i>Sprint 5</i> (sección 5.7). Historias de usuario (sección 6.1.1)
Módulo de gestión de la edición y creación de informes, tanto online como offline		<i>Sprint 1</i> (sección 5.3), <i>Sprint 2</i> (sección 5.4), <i>Sprint 3</i> (sección 5.5), <i>Sprint 4</i> (sección 5.6) y <i>Sprint 5</i> (sección 5.7). Historias de usuario (sección 6.1.2)
Módulo para la asignación de afectados		<i>Sprint 2</i> (sección 5.4). Historias de usuario (sección 6.1.3)
Módulo para la integración con distintas fuentes de datos	Parcialmente	<i>Sprint 6</i> (sección 6.4). Historias de usuario (sección 6.1.4)

Tabla 6.1: Resumen de satisfacción de objetivos parciales.

6.2. CONSECUCIÓN DE OBJETIVOS DOCENTES

En esta sección se proporcionará una descripción detallada de las asignaturas del grado de Ingeniería Informática donde se adquirieron los conocimientos necesarios y cómo se han aplicado en este proyecto. De esta manera, se establecerá una relación entre la formación académica y el trabajo realizado en la empresa.

6.2.1. Empleo de una metodología de desarrollo en el ciclo de vida del software

Este objetivo se ha completado al haber empleado una metodología iterativa-incremental, véase capítulo [4](#). Estos conocimientos se adquirieron y comprendieron a lo largo del grado como en las asignaturas de Ingeniería de Software I y II. En el capítulo [5](#), se puede observar como tras terminar cada iteración se produce un incremento del producto.

6.2.2. Empleo de una adaptación del marco de trabajo Scrum para el desarrollo ágil de software

Este objetivo se ha completado tras usar una adaptación de *Scrum* para el desarrollo del proyecto. Estos conocimientos se adquirieron durante el segundo año de la especialización en Ingeniería de Software, particularmente en la asignatura de Gestión de Proyectos Software. En el capítulo [5](#), se puede apreciar cómo se emplean los artefactos, roles y se llevan a cabo todos los eventos de *Scrum*, excepto la *Sprint Retrospective*. Como se detalla en la sección [4.3](#), se ha optado por una conclusión conjunta con la *Sprint Review*.

6.2.3. Empleo de técnicas de estimación, como *Planning Poker*, en la gestión de un proyecto software

Este objetivo se ha completado tras usar la técnica de estimación *Planning Poker* durante el *Sprint Planning*. Estos conocimientos se adquirieron en la asignatura de Gestión de Proyectos Software. En el capítulo 5, se puede apreciar cómo se emplea a la hora de realizar el *Sprint Planning*.

6.2.4. Empleo de principios de usabilidad para el diseño de interfaces de aplicaciones

Este objetivo se ha completado al aplicar los principios de usabilidad estudiados en la asignatura de Interacción Persona-Ordenador. Por ejemplo, se puede observar la correspondencia entre el sistema y el mundo real en las imágenes, donde los botones como “Borrar informe” muestran una papelera (ver Figura 5.13), y en el menú desplegable del módulo de edición de informes se utiliza una puerta para la funcionalidad “Salir del informe” (ver Figura 5.26). Además, el diseño se caracteriza por su alta fiabilidad, con un control de errores que alerta sobre datos que faltan o incorrectos de manera clara y comprensible. En las Figuras 6.1 y 6.2, se pueden ver ejemplos de estos controles de errores.

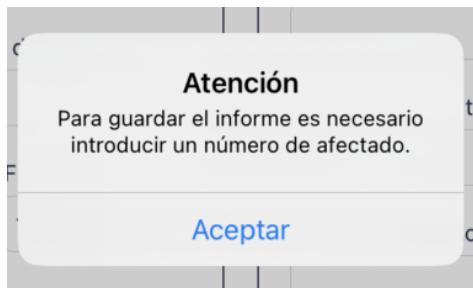


Figura 6.1: Control de error si se intenta guardar un informe sin número de afectado en HERA iOS.



Figura 6.2: Control de error si se intenta guardar un informe con un formato erróneo en el DNI del paciente en HERA iOS.

6.3. CONSECUCIÓN DEL OBJETIVO GENERAL

Tras la finalización completa del desarrollo de este TFG, y como se ha evidenciado en las secciones anteriores de objetivos parciales (ver sección 6.1) y docentes (ver sección 6.2), se puede inferir que se ha logrado cumplir con el objetivo principal establecido en la sección 2.1. Dicho objetivo consistía en desarrollar una aplicación móvil iOS utilizando el patrón de arquitectura VIPER y tecnología nativa (Swift), con el propósito de proporcionar al sector sanitario una herramienta de soporte que optimice el proceso de triaje durante el traslado de pacientes en ambulancias.

6.4. TRABAJO PRESENTE Y FUTURO

En esta sección se detallará parte del *Sprint 6* que es necesaria para la consecución parcial del objetivo 6.1.4. Este sprint se abordará en esta sección, en lugar de en el capítulo 5 de Resultados, ya que a fecha de redacción de este TFG, el *sprint* aún no se ha completado. Además, se describirán los próximos pasos a seguir en el proyecto **HERA iOS** y posibles mejoras.

6.4.1. Sprint 6

El inicio del sexto *sprint* comenzó el 6 de junio de 2024 y, a fecha de redacción de este TFG, sigue en desarrollo.

6.4.1.1. Objetivo general del Sprint 6

El propósito de este *sprint* era completar las pruebas de las historias del anterior sprint, implementar la integración con dispositivos *bluetooth* y comenzar la pestaña “Monitorización de constantes”.

6.4.1.2. Sprint 6 Planning

La reunión de planificación del sexto *sprint* se llevó a cabo el 6 de junio de 2024, con una duración aproximada de dos horas. En ella, se realizó un análisis de los puntos historia utilizando la técnica *Planning Poker*, véase en el capítulo 4, para estimar las historias de usuario planificadas para el *sprint*.

En consecuencia, la planificación quedaría de la siguiente manera, según se muestra en la Tabla 6.2.

Historia de usuario	Estim. Puntos/Historia	Estimación Horas
H19: Pruebas WS	3	24
H20: Acceso offline	2	16
H21: Integración con dispositivos bluetooth	4	32
H22: Descarga JSON informe debug	1	8
H23: Pestaña Monitorización de constantes	9	72

Tabla 6.2: Planificación del *Sprint* 6.

6.4.1.3. Desarrollo por historias del Sprint 6

H21: Integración con dispositivos *bluetooth*

En la Tabla 6.3, se presentan las tareas y su estimación correspondiente para la implementación de esta historia de usuario, cuya descripción se proporciona a continuación:

“Como usuario, quiero poder conectarme con otros dispositivos, vía Bluetooth, para integrar su información al informe.”

Tareas	Estimación Horas
H21: Maquetación	10
H21: Codificación	20
H21: Pruebas	2

Tabla 6.3: Estimación tareas de la historia de usuario H21: Integración con dispositivos *bluetooth*.

El objetivo de esta historia de usuario es proporcionar una interfaz donde el usuario pueda visualizar un listado de los dispositivos disponibles vía *Bluetooth*, para poder seleccionarlo y vincularse con él.

Para llevar a cabo esta historia de usuario, se diseñó una interfaz que incluye un listado de los dispositivos disponibles y una barra de búsqueda para poder buscar el dispositivo según su nombre. El nombre del dispositivo seleccionado y su unidad aparecerá en un contenedor.

Para poder implementar la búsqueda de dispositivos *Bluetooth* se ha utilizado la biblioteca *Core Bluetooth*¹ de *Swift*. Esta biblioteca proporciona las clases y delegados necesarios para buscar dispositivos vía *Bluetooth* y establecer conexiones con ellos.

Para facilitar al usuario el acceso a esta funcionalidad, se han realizado cambios en la interfaz de *home*, incluyendo la opción de “Zoll” para buscar dispositivos.

En la Figura 6.3, se muestra el cambio producido en la interfaz de menú principal para habilitar esta funcionalidad. En la Figura 6.4, se muestra la interfaz de esta historia de usuario.

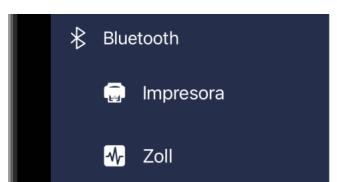


Figura 6.3: Cambios en interfaz *home* para la integración de dispositivos *Bluetooth* de HERA iOS.

¹<https://developer.apple.com/documentation/corebluetooth>

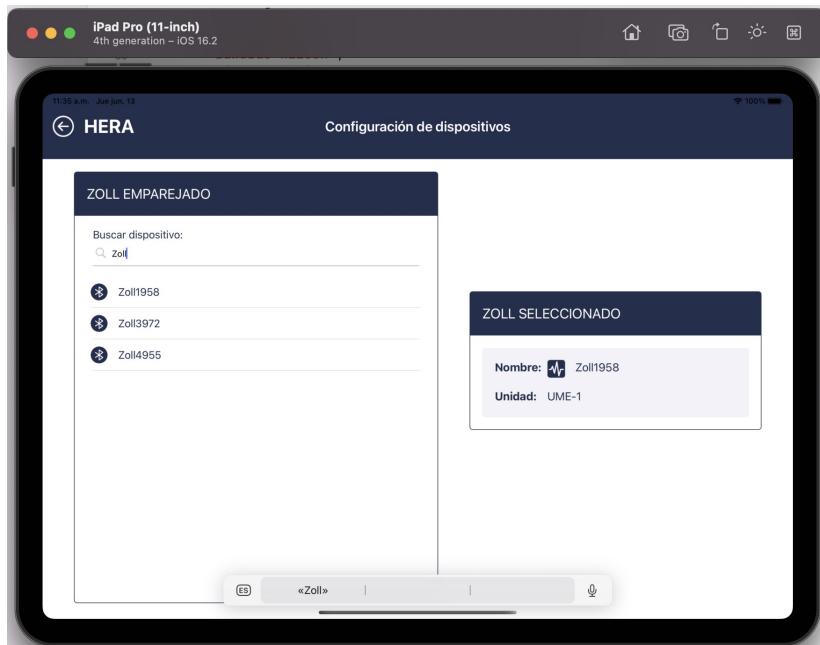


Figura 6.4: Interfaz integración dispositivos Bluetooth de **HERA iOS**.

6.4.2. Próximos pasos

Una vez completado el TFG y desarrollado un MVP de **HERA iOS**, se propone continuar con el desarrollo de las funcionalidades restantes de las adaptaciones actuales de HERA. Este avance podría conducir a futuras versiones de la aplicación y a posibles colaboraciones con la ESI mediante la definición de un nuevo FORTE. Además de las funcionalidades pendientes, se sugieren las siguientes mejoras que podrían resultar interesantes de implementar:

- **Funcionalidad de grabación de ambulancia y monitorización del paciente en tiempo real.** Esta característica permitiría al personal del hospital observar en directo los acontecimientos dentro de la ambulancia, lo que les permitiría ofrecer asistencia inmediata en caso de emergencia. Además, proporcionaría un registro visual y auditivo de las condiciones del paciente y las acciones realizadas durante el traslado, mejorando la coordinación y la calidad del cuidado médico.
- **Integración con un asistente mediante inteligencia artificial.** Esta funcionalidad permitiría al personal médico dictar los síntomas del paciente, que junto con los datos recopilados de dispositivos médicos integrados, se utilizarían para generar un informe detallado. Este informe se elaboraría mediante el uso de plantillas inteligentes, asegurando precisión y eficiencia en la documentación médica.

6.5. VALORACIÓN CRÍTICA

Una vez finalizado mi TFG, me gustaría expresar mi agradecimiento al programa FORTE y a *Inetum* por brindarme la oportunidad de adentrarme en el mundo laboral por primera vez. Esta valiosa experiencia no solo me ha permitido aplicar los conocimientos adquiridos a lo largo del grado, sino que también ha potenciado habilidades esenciales que no se trabajan tanto en el ámbito académico. Además, he adquirido un amplio conocimiento en tecnologías con las que no había tenido experiencia previa, gracias al apoyo y la orientación de mis compañeros y tutores.

Considero que lo más valioso que he aprendido es que, en el mundo real, la teoría puede diferir significativamente de la práctica. Esta experiencia me ha enseñado la importancia de saber adaptarse a cualquier circunstancia.

A nivel personal, considero que ha representado un gran desafío, dado que nunca antes había afrontado un proyecto de tal magnitud y complejidad. Esta experiencia te enseña el verdadero significado de asumir una responsabilidad real. A pesar de la magnitud y la complejidad de los problemas enfrentados, con esfuerzo y determinación es posible encontrar una solución a cualquier problema.

Siento que el esfuerzo invertido ha valido la pena, tanto para mí como para la empresa, y el resultado obtenido refleja claramente el trabajo y el compromiso dedicados al proyecto.

ANEXOS

ANEXO A

Prototipos de interfaces

En el presente anexo, se muestran los prototipos de interfaces proporcionadas por *Inetum* para realizar el desarrollo de **HERA iOS**.

A.1. PROTOTIPO INTERFAZ LOGIN OFFLINE

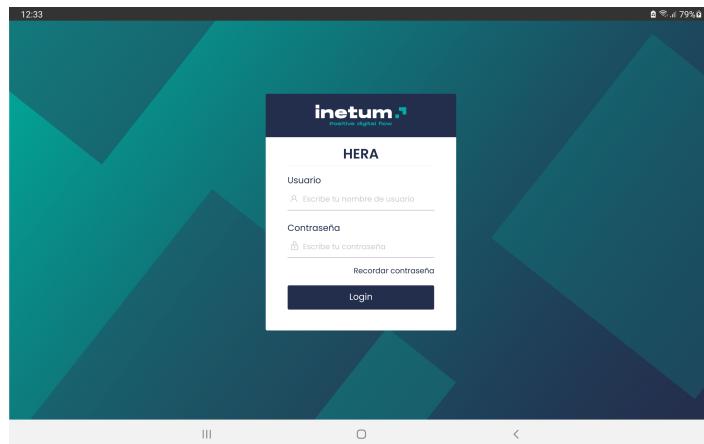


Figura A.1: Prototipo interfaz login offline.

A.2. PROTOTIPO INTERFAZ SELECCIÓN DE UNIDAD

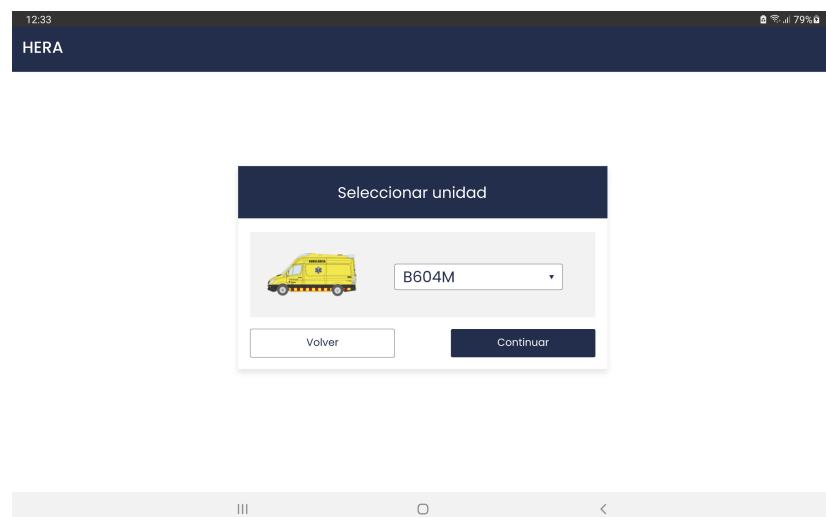


Figura A.2: Prototipo interfaz selección unidad.

A.3. PROTOTIPO INTERFAZ MENÚ PRINCIPAL

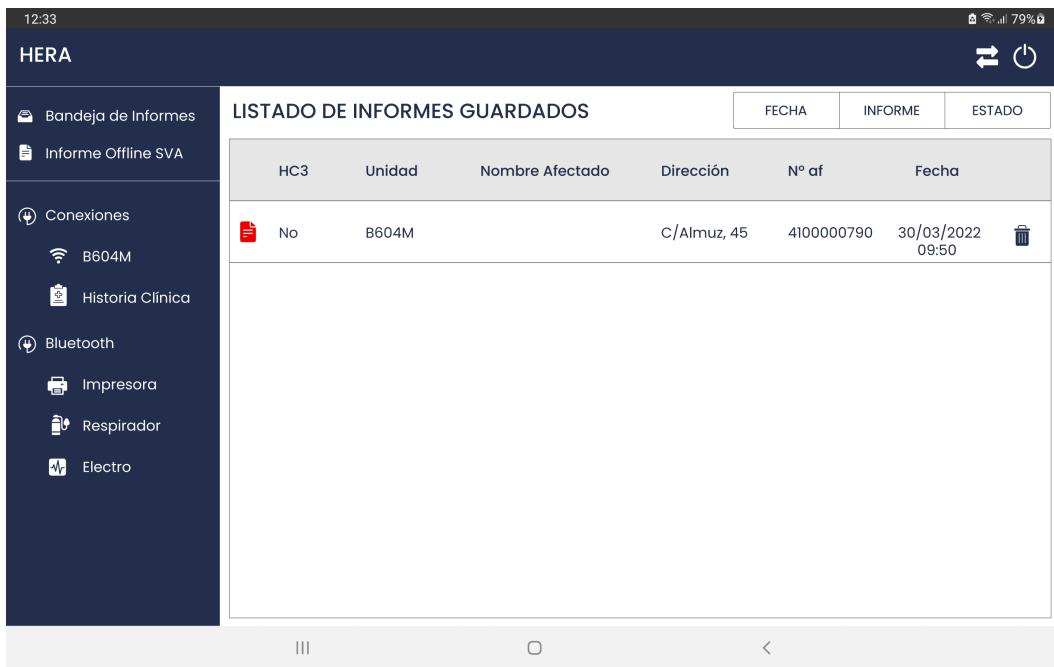


Figura A.3: Prototipo interfaz menú principal.

A.4. PROTOTIPO INTERFAZ BANDEJA DE INCIDENTES

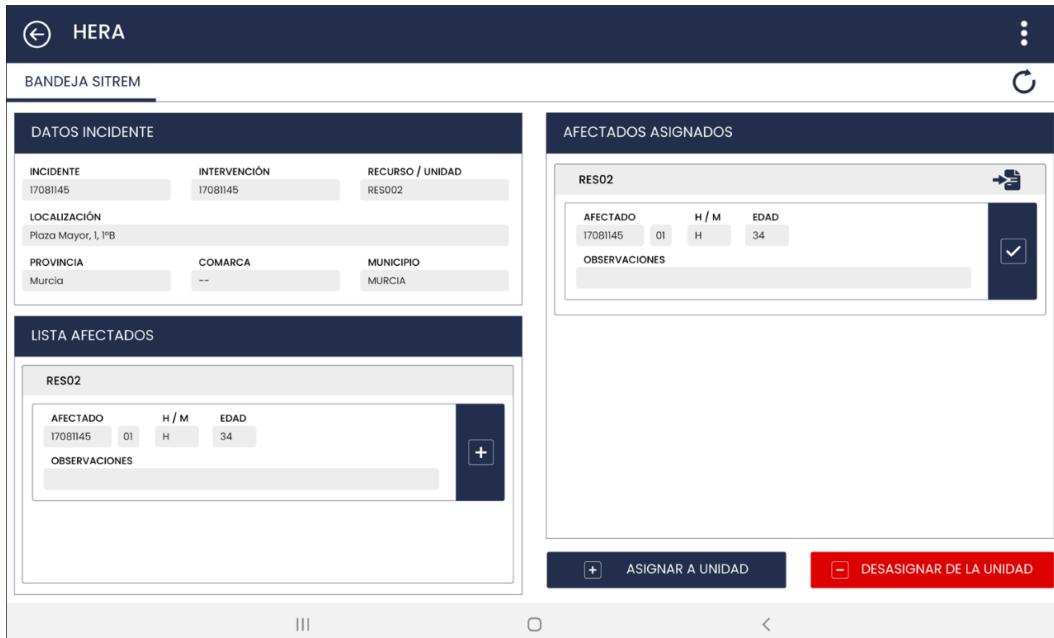


Figura A.4: Prototipo interfaz bandeja de incidentes.

A.5. PROTOTIPO INTERFAZ PESTAÑA CABECERA

12:33 HERA

CABECERA CARGO A TERCEROS ANAMNESIS EXPLORACIÓN PRIMARIA EXPLORACIÓN COMPLEMENTARIA EXPLORACIÓN FÍSICA 1 >

ID: 1625987412553654 Fecha: 30/03/2022

DOTACIÓN

- Unidad: B604M
- Número de intervención:
- Número de incidente:
- Nº Afectados:
- Hora:
- Fecha:
- Motivo de la alerta:
- Municipio:
- Dirección:
- Médico:
- Enfermero:
- TTS / TES / Piloto:

DATOS DEL PACIENTE

- Nombre paciente:
- Apellidos paciente:
- Fecha nacimiento: Edad:
- Tipo: Sexo:
- CIP paciente:
- Tipo documento identificación: Número documento: Valida
- Dirección paciente:

Figura A.5: Prototipo interfaz pestaña Cabecera.

A.6. PROTOTIPO INTERFAZ PESTAÑA TIPO Y FINALIZACIÓN DE SERVICIO

12:33 HERA

< VALORACIÓN ENFERMERÍA DIAGNÓSTICO ENFERMERÍA REGISTRO OHSCAR TIPOS Y FINALIZACIÓN SERVICIO

ID: 1625987412553654 Fecha: 30/03/2022

TIPOS DE SERVICIOS

Tipos de servicios:

OTRAS ACTUACIONES

Otras actuaciones SEM:
Otras actuaciones NO SEM:

FINALIZACIÓN

Finalización paciente:
Finalización unidad:

EFFECTOS PERSONALES

Objetos entregados / retirados:
Identificación receptor efectos personales:

Figura A.6: Prototipo interfaz pestaña Tipo y Finalización de servicio.

A.7. PROTOTIPO INTERFAZ PESTAÑA ANAMNESIS

12:33 79% HERA

CABECERA CARGO A TERCEROS ANAMNESIS EXPLORACIÓN PRIMARIA EXPLORACIÓN COMPLEMENTARIA EXPLORACIÓN FÍSICA 1 >

ID: 1625987412553654 Fecha: 30/03/2022

ALERGIAS

Alergias

ANTECEDENTES PATOLÓGICOS

Antecedentes patológicos

HÁBITOS TÓXICOS

Hábitos tóxicos

ENFERMEDAD ACTUAL

Enfermedad actual

PLAN DE MEDICACIÓN (RECETA ELECTRÓNICA)

CONSULTAR RECETA ELECTRÓNICA

MEDICACIÓN ACTUAL

Fármaco Dosis Duración

Figura A.7: Prototipo interfaz pestaña Anamnesis.

A.8. PROTOTIPO INTERFAZ PESTAÑA EXPLORACIÓN PRIMARIA

12:33 79% HERA

CABECERA CARGO A TERCEROS ANAMNESIS EXPLORACIÓN PRIMARIA EXPLORACIÓN COMPLEMENTARIA EXPLORACIÓN FÍSICA 1 >

ID: 1625987412553654 Fecha: 30/03/2022

ESTADO GENERAL

Estado general

ESTADO DEL AFECTADO NORMAL

VÍA AÉREA

Vía aérea

VENTILACIÓN

Ventilación

NEUROLOGÍA

Estado neurológico

PUPILAS

Inicial derecha Inicial izquierda

Reactividad derecha Reactividad izquierda

Simetría Isocóricas

Final derecha Final izquierda

Figura A.8: Prototipo interfaz pestaña Exploración Primaria.

A.9. PROTOTIPO INTERFAZ PESTAÑA EXPLORACIÓN FÍSICA 1

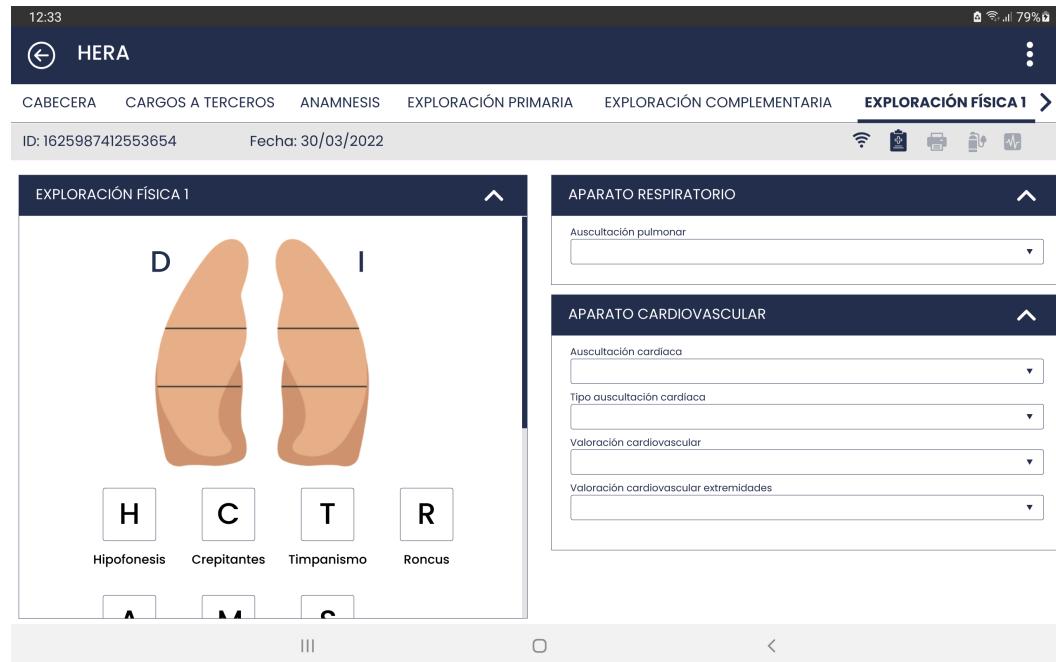


Figura A.9: Prototipo interfaz pestaña Exploración Física 1.

A.10. PROTOTIPO INTERFAZ PESTAÑA EXPLORACIÓN FÍSICA 2

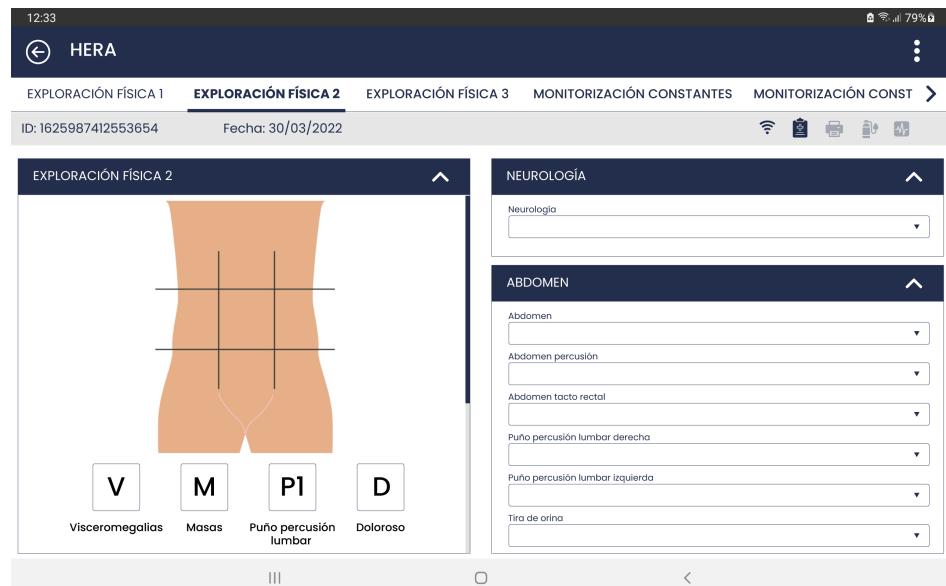


Figura A.10: Prototipo interfaz pestaña Exploración Física 2.

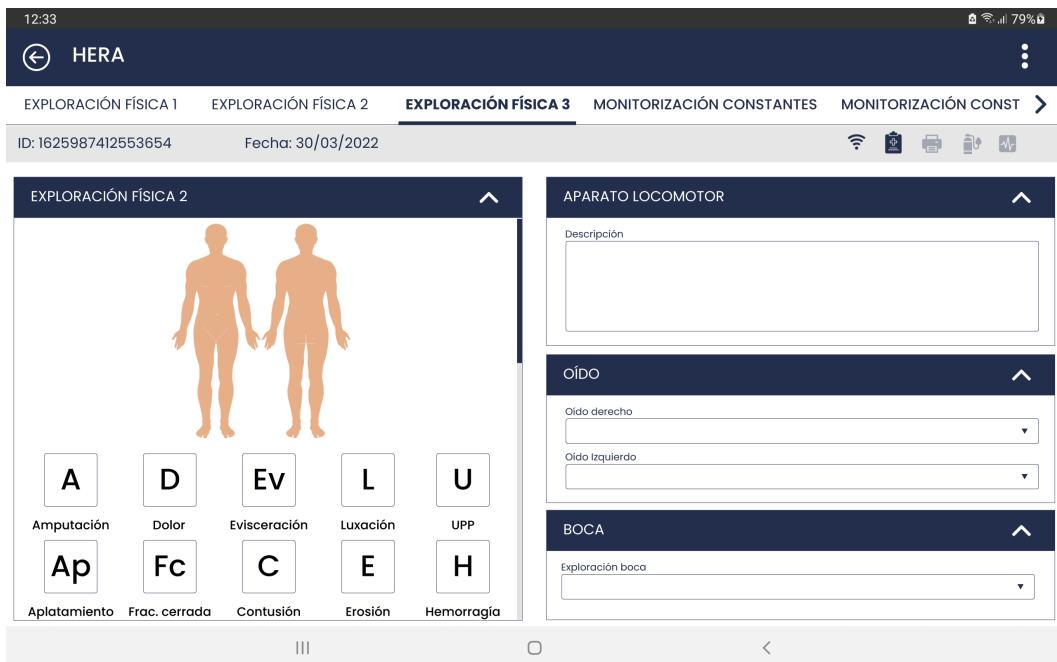
A.11. PROTOTIPO INTERFAZ PESTAÑA EXPLORACIÓN FÍSICA 3

Figura A.11: Prototipo interfaz pestaña Exploración Física 3.