



**UNIVERSIDAD
DE ANTIOQUIA**

**Desarrollo de una aplicación web y móvil para que sirva
como referencia técnica en proyectos de desarrollo de
software emergentes dentro la compañía Cidenet S.A.S**

Autor(es)

Juliana Muñoz Marín

Universidad de Antioquia

Facultad de Ingeniería, Departamento de Electrónica y
Telecomunicaciones

Medellín, Colombia

2020



Desarrollo de una aplicación web y móvil para que sirva como referencia técnica en proyectos de desarrollo de software emergentes dentro la compañía Cidenet S.A.S

Juliana Muñoz Marín

Informe de práctica para optar al título de:
Ingeniera de telecomunicaciones.

Asesores (a):

Natalia Gaviria Gómez

PhD en Ingeniería Eléctrica

Pablo Muñoz Zapata

Ingeniero de Sistemas e Informática

Universidad de Antioquia

Facultad de Ingeniería, Departamento de Electrónica y Telecomunicaciones.

Medellín, Colombia

2020.

Tabla de contenido

Resumen	4
Introducción	4
Objetivos	5
Objetivo general	5
Objetivos específicos	5
Marco Teórico	5
Gestión de conocimiento	6
SCRUM	7
Servicios	7
Microservicios	7
Metodología	8
Realizar el levantamiento de requisitos de las aplicaciones web y móvil para definición de las historias de usuarios de un software basado en arquitectura de microservicios.	8
Listado de requisitos	8
Requisitos microservicio de seguridad	8
Requisitos microservicio de perfiles	8
Requisitos aplicaciones web y móvil	8
Historias de usuario	9
Implementar las aplicaciones web y móvil para la ejecución de los requerimientos mínimos de una arquitectura de microservicios.	10
Arquitectura de las aplicaciones web y móvil	10
Servidor	11
Cliente	12
Estructura de carpetas	12
Metodología	14
Planeación de Sprint	14

Pruebas unitarias	16
Pruebas unitarias en Angular	18
Pruebas unitarias en Flutter	19
Evaluar la implementación de las aplicaciones web y móvil por medio del contraste con historias de usuario para la verificación del cumplimiento de los requerimientos del software basado en arquitectura de microservicios.	20
Validar el cumplimiento de los requerimientos de las aplicaciones web y móvil con los líderes de desarrollo para que puedan ser usadas como referente técnico en los proyectos emergentes.	25
Formato de caso de prueba	25
Resultados y análisis	26
Conclusiones	31
Referencias Bibliográficas	31

1 Resumen

Este proyecto surge a partir de la necesidad de la empresa Cidenet S.A.S de documentar o conservar el conocimiento, con el fin de evitar futuras redundancias en los diferentes proyectos o para tener un referente técnico en los proyectos emergentes. Para dar solución a este problema, se propuso y desarrolló un proyecto cliente - servidor, en dos tecnologías de plataformas diferentes (web y móvil). El cliente se implementó como una arquitectura de microservicios que permiten separar las funcionalidades en módulos. Por otro lado, se aseguró por medio de pruebas unitarias, con una cobertura del 100% en cada proyecto, la integridad del código implementado.

2 Introducción

Cidenet S.A.S es una empresa de desarrollo de software a la medida fundada en 2012, y que actualmente tiene presencia comercial en Suramérica, Centroamérica y Norteamérica. Dentro de la empresa, se trabaja con tecnologías como Angular, Bootstrap, React Native, Flutter, Oracle Service Bus, Android, Java, Nodejs, entre otras, para ajustarse a las necesidades del cliente y así ayudar a mejorar su productividad y competitividad. La empresa dispone de equipos de trabajo que están especializados en las tecnologías mencionadas y además utilizan las prácticas del marco ágil SCRUM, ejecutando Sprints de máximo 3 semanas, para así entregar al cliente productos funcionales en periodos cortos de tiempo.

Si bien Cidenet S.A.S tiene más de 5 años de trayectoria en el campo de desarrollo de software, aún no cuenta con un proceso claro que permita retener el conocimiento adquirido por los analistas desarrolladores. Así mismo, se ha evidenciado que existen proyectos que deben ser desarrollados desde cero sin una referencia de antemano que permita hacer uso de las buenas prácticas definidas en Cidenet S.A.S. Un claro ejemplo de esta situación se dio recientemente cuando la empresa le presentó un conjunto de servicios a un banco que tiene diferentes sedes en Centroamérica. En un principio, se decidió sólo implementar el desarrollo para una sola sede, pero posteriormente se empezó el desarrollo para las otras. Sin embargo, como no se tenía documentación sobre la primera fase del proyecto, al desarrollar muchos de estos servicios se presentaron problemas que ya se habían solucionado en la primera fase. El resultado en este caso, así como en muchos otros, es que muchos desarrollos son devueltos, redundando en retraso en las entregas de los mismos.

Para el año 2020, la empresa definió un objetivo que apunta a documentar y desarrollar proyectos que sirvan como futuras referencias técnicas. De acuerdo con lo anterior, se desarrollaron una aplicación web y una aplicación móvil, en las tecnologías Angular y Flutter, respectivamente, a partir de una arquitectura de microservicios que exponen servicios REST.

Este informe empieza por exponer los objetivos a alcanzar con el desarrollo del proyecto, haciendo uso de una contextualización con el marco teórico que más tarde será necesaria para entender el desarrollo de la metodología, donde se evidencia el cumplimiento de cada uno de los objetivos. Y finaliza con un apartado de resultados y conclusiones.

3 Objetivos

3.1 Objetivo general

Desarrollar una aplicación web y una móvil con base en una arquitectura de microservicios y siguiendo el marco de trabajo ágil SCRUM, para que sirva como referencia técnica en proyectos de desarrollo de software emergentes dentro de la compañía Cidenet S.A.S

3.2 Objetivos específicos

- Realizar el levantamiento de requisitos de las aplicaciones web y móvil para definición de las historias de usuarios de un software basado en arquitectura de microservicios.
- Implementar las aplicaciones web y móvil para la ejecución de los requerimientos mínimos de una arquitectura de microservicios.
- Evaluar la implementación de las aplicaciones web y móvil por medio del contraste con historias de usuario para la verificación del cumplimiento de los requerimientos del software basado en arquitectura de microservicios.
- Validar el cumplimiento de los requerimientos de las aplicaciones web y móvil con los líderes de desarrollo para que puedan ser usadas como referente técnico en los proyectos emergentes.

4 Marco Teórico

La gestión de conocimiento puede ser un reto para las diferentes empresas a la hora de manejarla, por ello, el uso de estrategias o metodologías para implementar una solución de software son necesarias, como por ejemplo SCRUM, dicha solución puede implementarse usando servicios o microservicios. Dicho esto, es necesario ahondar en estos términos para una comprensión holística del desarrollo del proyecto.

4.1 Gestión de conocimiento

La gestión del conocimiento (Knowledge Management-KM) es un concepto aplicado a las organizaciones que tiene el conocimiento como foco, para así garantizar que el ritmo de aprendizaje se acerque lo máximo posible al ritmo de cambio del mercado[1][2], ya que se pretende siempre enfrentar los cambios en las necesidades de los clientes. Así entonces, KM es una exigencia para que las compañías puedan afrontar la presión de los competidores y el constante cambio en las tecnologías [4]. De modo que en un mundo de negocios donde la ventaja competitiva se basa en la innovación y la demanda de la renovación de productos y servicios es enorme, las organizaciones que son capaces de crear las condiciones para que el ritmo de aprendizaje sea superior al exigido por el mercado tienen la capacidad de innovar, y por tanto, ofrecer una calidad superior[2][3].

KM tiene dos fases, donde la primera es planear, capturar, organizar, interconectar y permitir el acceso al capital intelectual de la organización a través de diferentes tecnologías. La segunda fase consiste en dirigir o supervisar los bienes intelectuales y a aquellos que están involucrados en estos procesos [3].

Una buena gestión del conocimiento permite identificar, capturar, recuperar, compartir y evaluar la información de la organización y así transmitir a través de ésta el conocimiento adquirido para que cualquier empleado pueda acceder a este en el momento correcto. KM ayuda a las organizaciones a manejar correctamente la memoria corporativa y los activos intelectuales para así mejorar su competencia, la cual no se debe basar sólo en productos y servicios [1][3].

Una organización ofrece y vende lo que sabe hacer, y este saber viene de las personas que trabajan de forma coordinada con diferentes tecnologías y adquiriendo experiencia para posteriormente crear una estructura que con el tiempo ayudará a la organización a ganar ventaja competitiva. En otras palabras, la gestión del conocimiento incluye una sobreposición de las personas, la tecnología y los procesos [2][1].

En general, las diferentes definiciones de la gestión del conocimiento contienen los siguientes aspectos [1]:

- Usar conocimiento accesible de recursos externos.
- Unir y almacenar el conocimiento en los procesos de negocios, productos y servicios.
- Representar el conocimiento en bases de datos y documentos.
- Incentivar el desarrollo del conocimiento a través de la cultura de la organización.
- Compartir el conocimiento a través de la organización.

- Evaluar el valor del bien intelectual y su impacto regularmente.

4.2 SCRUM

SCRUM es un marco de trabajo diseñado para manejar el desarrollo de aplicaciones, con el cual se pueden direccionar problemas complejos al tiempo que entregan de forma productiva y creativa productos con el mayor valor posible [5]. Scrum se desarrolla en un número de iteraciones llamadas Sprints y está basado en la noción de equipo y se construye alrededor de tres roles: El dueño del producto, el Scrum Master y el equipo de desarrollo.

El dueño del producto o Product Owner es quien debe maximizar el valor del producto mediante la transmisión de requerimientos o historias de usuario, las cuales son una lista de necesidades que forman lo que se llama Product Backlog, además, el Product Owner es quien debe priorizarlas [6].

El Scrum Master es quien asegura que el marco Scrum se sigue de forma correcta tomando las decisiones necesarias en cuanto a la aplicación del método. Entre sus responsabilidades están: planificar los Sprints, realizar los Scrum Meetings, revisión de los Sprint[6].

Los Sprints están constituidos por historias de usuario, y dependiendo del tiempo que sea asignado al Sprint, el equipo de desarrollo debe entonces desarrollar cierta cantidad de historias de usuario con el objetivo de ofrecer un entregable de calidad.

4.3 Servicios

En la Arquitectura Basada en Servicios (SOA), un servicio es una funcionalidad de negocio que puede ser fácilmente accedida desde cualquier cliente usando un protocolo de acceso estándar. Los servicios SOA proveen el servicio por sí mismos, acceder a ellos varias veces con la misma petición producirá el mismo resultado y además, no se guarda ningún estado en nombre de ningún servidor[7].

4.4 Microservicios

Los microservicios son un acercamiento diferente a la arquitectura de una aplicación, son una colección de servicios llamados módulos pequeños y autónomos que trabajan juntos y que están diseñados para enfocarse y desempeñar una sola tarea, sencilla y bien definida[7][8].

5 Metodología

5.1 Realizar el levantamiento de requisitos de las aplicaciones web y móvil para definición de las historias de usuarios de un software basado en arquitectura de microservicios.

5.1.1 Listado de requisitos

Los líderes de desarrollo manifestaron que algunas tecnologías usadas dentro de la compañía no eran empleadas por todos los desarrolladores y que debido a esto se necesitaban diferentes aplicaciones en estas tecnologías para ser usadas como referente técnico.

Se decidió realizar una aplicación web y una móvil con relación a una red social para héroes, dónde estos podrán modificar y ver su perfil, además de ver los demás héroes que pertenecen a la plataforma.

La definición de los requisitos se realizó entre el líder de desarrollo y analista-desarrollador que resuelve el presente proyecto, respondiendo a la pregunta, ¿Qué se espera que pueda hacer un héroe cuando entre a la aplicación?

A continuación, se presentan los requisitos mínimos de cada componente:

5.1.1.1 Requisitos microservicio de seguridad

- Se debe guardar en la base de datos un nuevo usuario que deberá estar asociado a un héroe.
- Se deben generar tokens para que un usuario pueda acceder a las aplicaciones.

5.1.1.2 Requisitos microservicio de perfiles

- Se debe guardar en la base de datos un nuevo héroe con todos sus datos básicos.
- Se debe modificar la información guardada en la base de datos de un héroe.
- Se debe entregar la información de todos los perfiles que estén registrados, de forma masiva y/o individual.

5.1.1.3 Requisitos aplicaciones web y móvil

Se deben tener las siguientes pantallas:

- Una pantalla que permita al usuario registrarse como héroe en la aplicación.
- Una pantalla que permita a un usuario ya registrado poder acceder a su cuenta.

- Una pantalla que permita a un usuario visualizar todos los perfiles que estén registrados en la aplicación.
- Una pantalla que permita visualizar y editar la información de un usuario autenticado.

5.1.2 Historias de usuario

Las historias de usuario son realizadas a partir de los requisitos que fueron definidos anteriormente. En estas se explica de forma corta y simple cada una de las funcionalidades.

En la tabla 1 se muestra un ejemplo de historia de usuario (HU) para presentar la plantilla que se usó para modelarlas.

En general, tiene 5 secciones:

- **Código:** Identificador único de la HU y se acompaña con un breve título.
- **Cómo:** Rol de la persona que hará uso de la funcionalidad.
- **Requiere:** Se explica qué se espera de la funcionalidad.
- **Para:** Se explica el propósito de la funcionalidad.
- **Requerimientos de aceptación:** Se enuncian los detalles de la funcionalidad, es decir, lo que se debe cumplir para marcar la HU como desarrollada completamente.
- **Material de referencia:** Se agrega un Mock de cómo debería quedar la parte gráfica de la HU.

Tabla 1. Ejemplo de Historia de usuario.

Código (Code)	WAH-02 Profile
Cómo (As)	Héroe
Requiere (Require)	Ver mi perfil
Para (To)	Revisar mi información y habilidades
Requerimientos de aceptación	<ul style="list-style-type: none"> • Mostrar mi información y habilidades • Mostrar mi foto de perfil • Un botón para editar mi perfil
Material de referencia	

Con base a lo anterior, en las tablas 2 y 3 se listan las historias de usuario que sirven como resumen de las funcionalidades a desarrollar. En el anexo 1 se observan con más detalle las historias de usuario.

Tabla 2. Historias de usuario de la aplicación web.

Código	Historia de usuario
WAH-01	Top Heroes
WAH-02	Profile
WAH-03	Edit profile
WAH-08	Hero details
WAH-09	Sign up page
WAH-10	Login page

Tabla 3. Historias de usuario de la aplicación móvil.

Código	Historia de usuario
MFH-01	Login page
MFH-02	Sign up page
MFH-03	Top Heroes
MFH-04	Hero details
MFH-05	Profile
MFH-06	Edit profile

5.2 Implementar las aplicaciones web y móvil para la ejecución de los requerimientos mínimos de una arquitectura de microservicios.

5.2.1 Arquitectura de las aplicaciones web y móvil

Las aplicaciones de software web y móvil son por definición cliente-servidor, por lo cual el cliente para la aplicación web se desarrolló con el framework Angular en su versión 8. Para la aplicación móvil, se usó el framework Flutter y el servidor, que

en este caso son los microservicios de seguridad y perfiles, se desarrollaron con el framework Spring Boot. La comunicación entre éstos es por medio de servicios REST.

Por otro lado, la base de datos que se usará en este caso será h2 que está programada en Java y fue especialmente creada para ambientes de desarrollo y así agilizar el proceso.

En la figura 1 se muestra por medio de un diagrama la arquitectura cliente-servidor que se mencionó anteriormente.

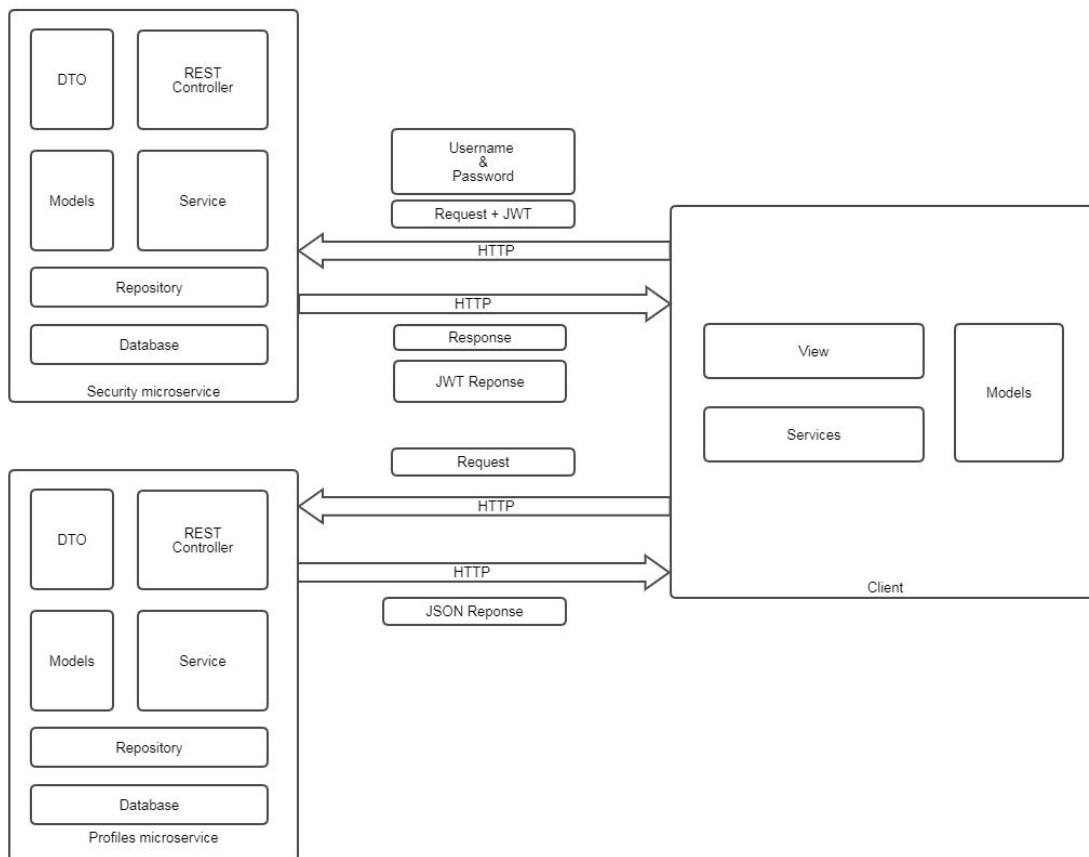


Figura 1. Diagrama de arquitectura de las aplicaciones web y móvil.

5.2.2 Servidor

Como se mencionó anteriormente, el servidor está compuesto por dos microservicios de perfiles y de seguridad, y sus componentes son:

- **REST Controllers:** responsables de exponer los servicios REST.
- **Services:** son los encargados de la lógica de negocio.
- **Repositories:** responsables de dar el acceso a la base de datos.

- **DTO (Data Transfer Object):** responsable de encapsular la información que se expone a través de los servicios REST.
- **Models:** se encargan de modelar contra la base de datos.
- **Database:** responsable de mantener la información.

5.2.3 Cliente

El cliente, tanto en Flutter como en Angular, se implementó con los siguientes componentes en general:

- **View:** contiene la interfaz de usuario.
- **Services:** responsables de consumir los servicios REST expuestos por el servidor.
- **Models:** se encargan de modelar la información para ser enviada a través de los servicios REST.

5.2.4 Estructura de carpetas

Debido a que el proyecto también va orientado a las buenas prácticas que se han definido dentro de Cidenet S.A.S, la estructura de carpetas que se definieron para el proyecto está dada por:

En Angular, cada módulo fue dividido de la siguiente forma:

- Servicios.
- Componentes.
- Modelos.
- Módulo y Routing.

En la figura 2 se muestra la estructura de carpetas para la aplicación en Angular.

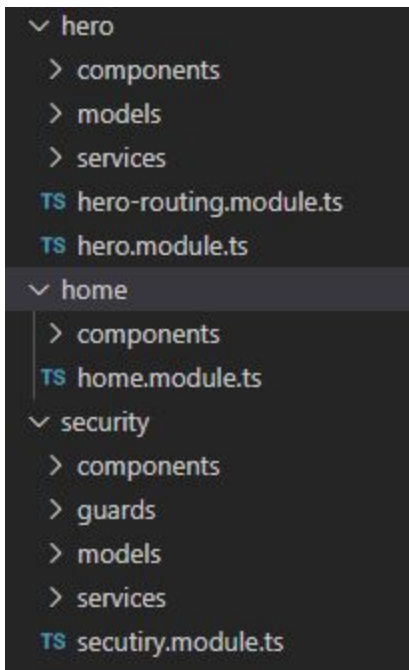


Figura 2. Estructura de carpetas en Angular.

En Flutter, cada módulo fue dividido de la siguiente forma:

- Servicios.
- Pantallas (Screens).
- Modelos.

En la figura 3 se muestra la estructura de carpetas para la aplicación en Flutter.

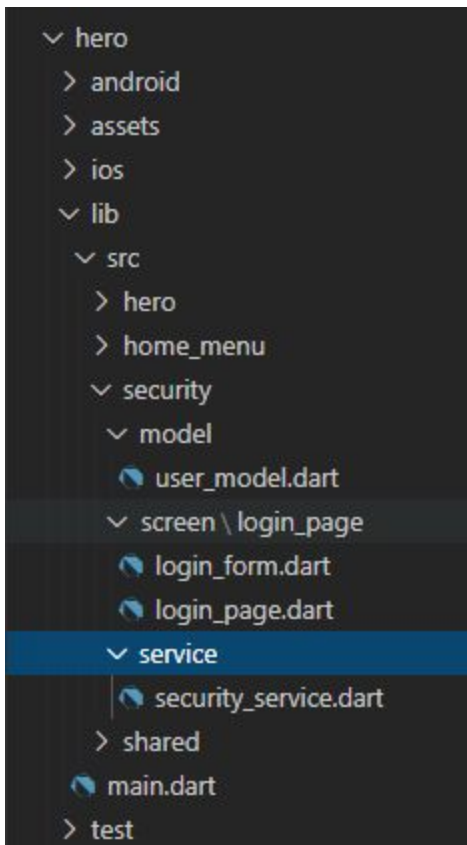


Figura 3. Estructura de carpetas en Flutter.

5.2.5 Metodología

Se decidió usar SCRUM como el marco de trabajo ya que es bastante útil para darle gestión al proyecto de forma ágil. Por esto, se desarrollaron las historias de usuario de acuerdo a este marco.

5.2.5.1 Planeación de Sprint

A continuación, se presenta en las tablas 4 a la 11, la planeación que se realizó por Sprint. La tarea de estimación se hizo de cuenta del desarrollador solamente, con acompañamiento del líder para evitar desfases o atrasos en la implementación.

Sprint 0

En este Sprint se realizó el Backlog de historias de usuario, es decir, se definieron las historias de usuario y su prioridad.

Tabla 4. Historias de usuario desarrolladas en el Sprint 1.

Código	Historia de usuario
H-001	Create the Project basic structure
H-002	Hero management
H-003	Skills management

Tabla 5. Historias de usuario desarrolladas en el Sprint 2.

Código	Historia de usuario
S-001	Create the Project basic structure
S-002	User management

Tabla 6. Historias de usuario desarrolladas en el Sprint 3.

Código	Historia de usuario
WAH-01	Top Heroes
WAH-08	Hero details

Tabla 7. Historias de usuario desarrolladas en el Sprint 4.

Código	Historia de usuario
WAH-02	Profile
WAH-03	Edit profile

Tabla 8. Historias de usuario desarrolladas en el Sprint 5.

Código	Historia de usuario
WAH-09	Sign up page
MFH-03	Top Heroes

Tabla 9. Historias de usuario desarrolladas en el Sprint 6.

Código	Historia de usuario
MFH-04	Hero details
MFH-05	Profile

Tabla 10. Historias de usuario desarrolladas en el Sprint 7.

Código	Historia de usuario
MFH-06	Edit profile

Tabla 11. Historias de usuario desarrolladas en el Sprint 8.

Código	Historia de usuario
WAH-10	Login page
MFH-01	Login page

5.2.6 Pruebas unitarias

Las pruebas unitarias son prácticas para ayudar a verificar o comprobar qué secciones del código funcionan correctamente y de la forma esperada.

Ambas aplicaciones, web y móvil, fueron sometidas a este proceso de pruebas unitarias, las cuales son constantes y cada función fue validada.

Para la implementación de estas pruebas, se trabajó el framework Jest en Angular, y se trabajó con los paquetes flutter_test y mockito en Flutter. Además, con Gitlab CI (Integración continua), que es una herramienta que funciona con pipelines, que sirven para dar ejecución a un Script que ejecuta un conjunto de comandos tales como compilar y correr las pruebas unitarias. En la figura 4 se observa la ejecución de un pipeline y en la figura 5 la ejecución de una de las tareas de éste.

passed Pipeline #156794060 triggered 3 weeks ago by Juan Camilo Peña Vahos

Merge branch 'feature/jumunoz/sign-up-page' into 'develop'

Feature/jumunoz/sign up page
See merge request !9

2 jobs for `develop` in 6 minutes and 11 seconds (queued for 1 second)

latest

8de9aee3

No related merge requests found.

Pipeline DAG Beta Jobs 2 Tests 0

Validate Test

code_quality unit_tests

Figura 4. Ejemplo de Pipeline en Gitlab.

```
101 Test Suites: 14 passed, 14 total
102 Tests:      49 passed, 49 total
103 Snapshots:  0 total
104 Time:       15.896s
105 Ran all test suites.
107 Running after_script
109 Saving cache
110 Creating cache develop...
111 node_modules/: found 43921 matching files
112 Uploading cache.zip to https://storage.googleapis.com/gitlab-com-runners-cache/project/15462312/develop
113 Created cache
115 Uploading artifacts for successful job
117 Job succeeded
```

Figura 5. Ejemplo de un Job en Gitlab.

5.2.6.1 Pruebas unitarias en Angular

Como se mencionó anteriormente, la realización de las pruebas unitarias de la aplicación web en Angular se realizaron con el framework jest, se codificaron en los archivos .spec.ts y en total son 53 pruebas.

En la figura 6 se muestra un ejemplo de una prueba unitaria en Angular y en la figura 7 se muestra el resultado total de las pruebas unitarias.

```
test('should get a hero by id', async () => {
  component.id = RESPONSE_SINGLE_HERO.response.id;
  await component.getHeroById();

  expect(heroService.getHeroById).toHaveBeenCalled();
  expect(heroService.getHeroById).toHaveBeenCalledWith(RESPONSE_SINGLE_HERO.response.id);
  expect(component.hero).toEqual(RESPONSE_SINGLE_HERO.response);
  expect(component.editProfileForm.valid).toBe(true);
  expect(component.selectedSkills).toEqual(RESPONSE_SINGLE_HERO.response.skillsDto);
});
```

Figura 6. Ejemplo de pruebas unitarias en Angular con el framework jest.

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
All files	100	100	100	100	
app	100	100	100	100	
app.component.ts	100	100	100	100	
app/hero/components/edit-profile	100	100	100	100	
edit-profile.component.ts	100	100	100	100	
app/hero/components/hero-details	100	100	100	100	
hero-details.component.ts	100	100	100	100	
app/hero/components/profile	100	100	100	100	
profile.component.ts	100	100	100	100	
app/hero/components/top-heroes	100	100	100	100	
top-heroes.component.ts	100	100	100	100	
app/hero/services/hero	100	100	100	100	
hero.service.ts	100	100	100	100	
app/hero/services/skills	100	100	100	100	
skills.service.ts	100	100	100	100	
app/home/components	100	100	100	100	
home.component.ts	100	100	100	100	
app/security/components/login-page	100	100	100	100	
login-page.component.ts	100	100	100	100	
app/security/components/navbar-login	100	100	100	100	
navbar-login.component.ts	100	100	100	100	
app/security/components/sign-up-page	100	100	100	100	
sign-up-page.component.ts	100	100	100	100	
app/security/services	100	100	100	100	
security.service.ts	100	100	100	100	
app/shared/utils/alert	100	100	100	100	
alert.utils.ts	100	100	100	100	
app/shared/utils/modal	100	100	100	100	
modal.utils.ts	100	100	100	100	
Test Suites: 14 passed, 14 total					
Tests: 53 passed, 53 total					
Snapshots: 0 total					

Figura 7. Resultado de pruebas unitarias en Angular.

5.2.6.2 Pruebas unitarias en Flutter

En flutter, con la ayuda de mokito y flutter_test, se encuentran en la carpeta Test y se realizaron 24 pruebas unitarias en total, que comprueban la integridad de los servicios, funciones y modelos.

En la figura 8 se muestra un ejemplo de una prueba unitaria en Flutter y en la figura 9 se muestra el resultado total de las pruebas unitarias.

```

test('should call the GET and return all the heroes', () async {
  when(mockClient.get(baseUrl)).thenAnswer(
    (_) async => http.Response(HeroMock.RESPONSE_MULTIPLE_HEROES, 200));

  final List<Hero> heroes = await heroService.getHeroes();

  expect(heroes.length, 2);
  expect(heroes[0].id, equals(HeroMock.heroes[0].id));
  expect(heroes[0].name, equals(HeroMock.heroes[0].name));
  expect(heroes[0].photoPath, equals(HeroMock.heroes[0].photoPath));
  expect(heroes[0].description, equals(HeroMock.heroes[0].description));
  expect(heroes[0].score, equals(HeroMock.heroes[0].score));
  expect(heroes[0].skills.length, equals(HeroMock.heroes[0].skills.length));
  expect(heroes[1].id, equals(HeroMock.heroes[1].id));
  expect(heroes[1].name, equals(HeroMock.heroes[1].name));
  expect(heroes[1].photoPath, equals(HeroMock.heroes[1].photoPath));
  expect(heroes[1].description, equals(HeroMock.heroes[1].description));
  expect(heroes[1].score, equals(HeroMock.heroes[1].score));
  expect(heroes[1].skills.length, equals(HeroMock.heroes[1].skills.length));
});

```

Figura 8. Ejemplo de pruebas unitarias en Flutter con flutter_test y mockito.

Directory	Line Coverage ↕	Functions ↕
hero/model	100.0 % 26 / 26	- 0 / 0
hero/service	100.0 % 34 / 34	- 0 / 0
security/model	100.0 % 8 / 8	- 0 / 0
security/service	100.0 % 13 / 13	- 0 / 0
shared/model	100.0 % 17 / 17	- 0 / 0
shared/validators	100.0 % 14 / 14	- 0 / 0

Figura 9. Resultado pruebas unitarias en Flutter.

5.3 Evaluar la implementación de las aplicaciones web y móvil por medio del contraste con historias de usuario para la verificación del cumplimiento de los requerimientos del software basado en arquitectura de microservicios.

Como se expuso en el formato de las historias de usuario, cada una de éstas tiene unos requerimientos de aceptación, los cuales hacen referencia a componentes gráficos que deben estar presentes a la hora de confirmar que la historia de usuario se realizó satisfactoriamente.

Para la validación, se hizo un contraste de estos requerimientos de aceptación por historia de usuario junto con la aplicación que se desarrolló.

En las figuras 10 y 11 se puede observar dos ejemplos de un requerimiento de forma resaltada en la pantalla desarrollada para la aplicación web. Para ver las validaciones de todas las historias de usuario de esta aplicación diríjase al anexo 2.

Acceptance requirements:

- Fields for the basic data:
 - Email
 - Password
- A button to sign in
- A button to sign up

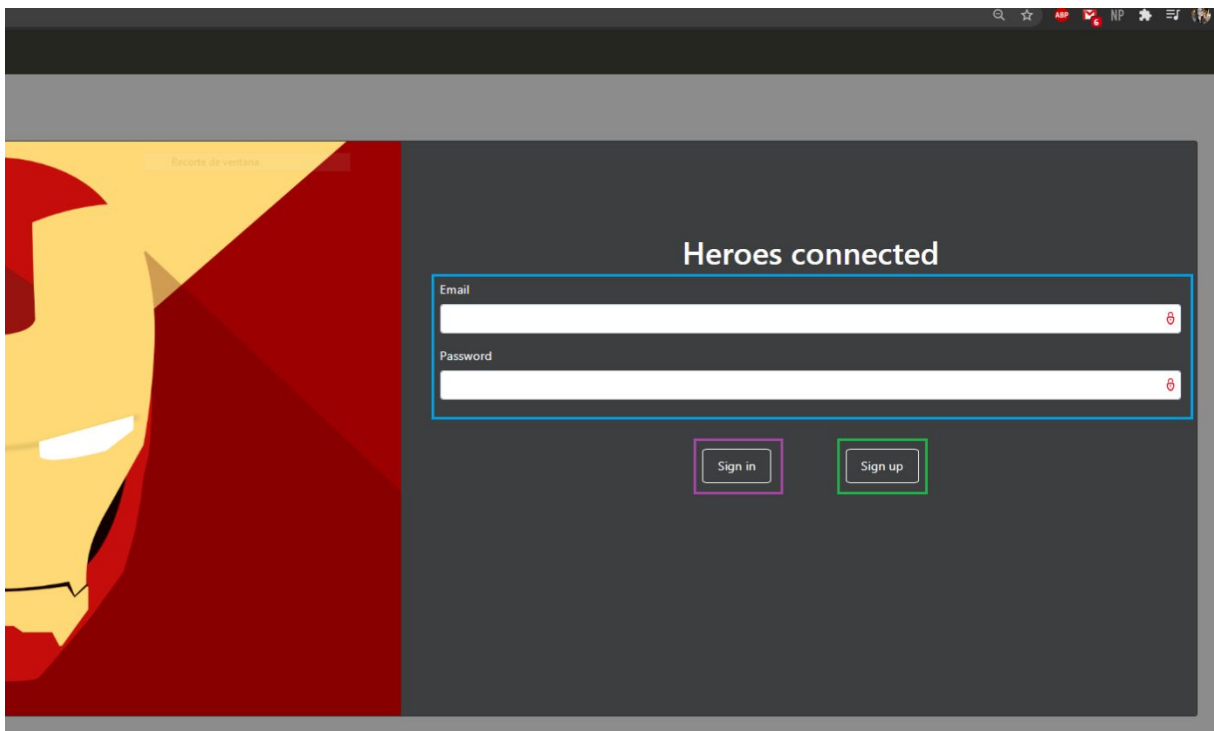


Figura 10. Ejemplo contraste de historia de usuario WAH-10 con la aplicación web.

Acceptance requirements:

- Fields for the basic data:
 - Email
 - Password
 - Name (Confidential)
 - Hero name
 - Description
- A button to save the profile
- A button to cancel the transaction

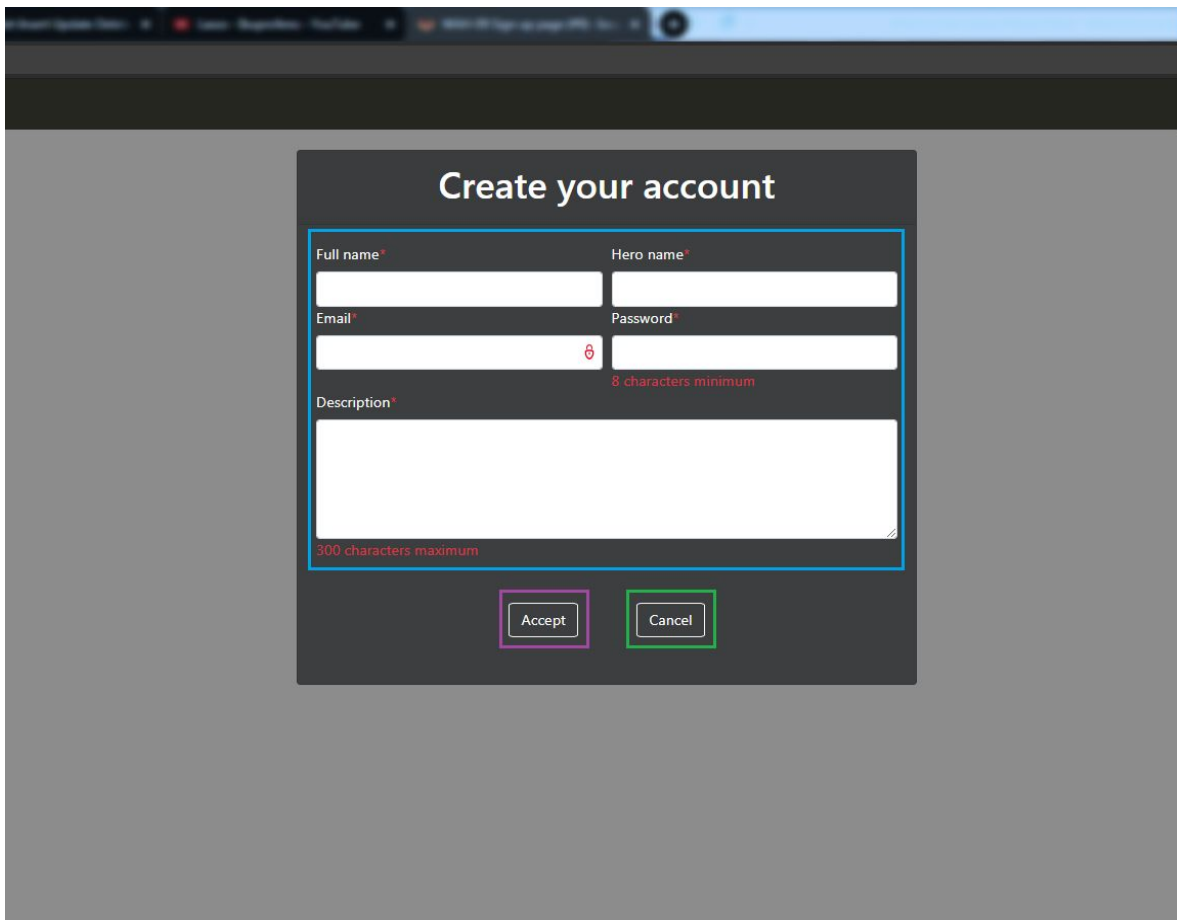


Figura 11. Ejemplo contraste de historia de usuario WAH-09 con la aplicación web.

Ahora, en las figuras 12 y 13 se puede observar dos ejemplos de un requerimiento de forma resaltada en la pantalla desarrollada para la aplicación móvil. Para ver las validaciones de todas las historias de usuario de esta aplicación diríjase al anexo 3.

Acceptance requirements:

- Fields:
 - Email
 - Password
- A button to login.
- A button to create an account.

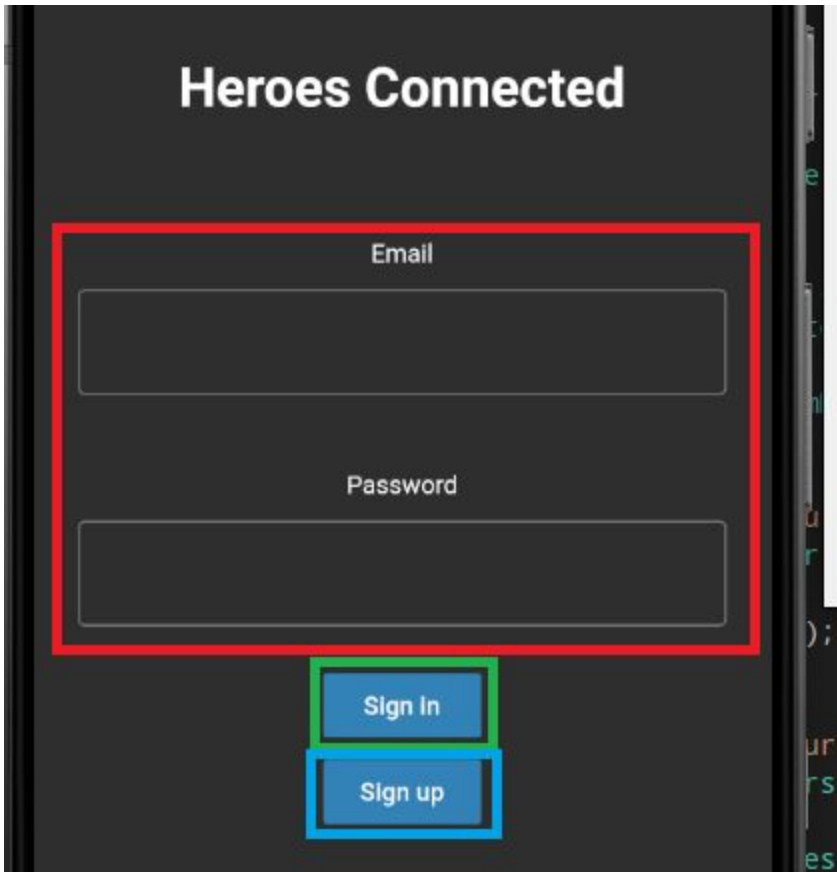


Figura 12. Ejemplo contraste historia de usuario MFH-01 con la aplicación móvil.

Acceptance requirements:

- Fields for the basic data:
 - Email
 - Password
 - Name (Confidential)
 - Hero name
 - Description
- A button to save the profile
- A button to cancel the transaction

Create an account

Full name

Hero name

Email

Password

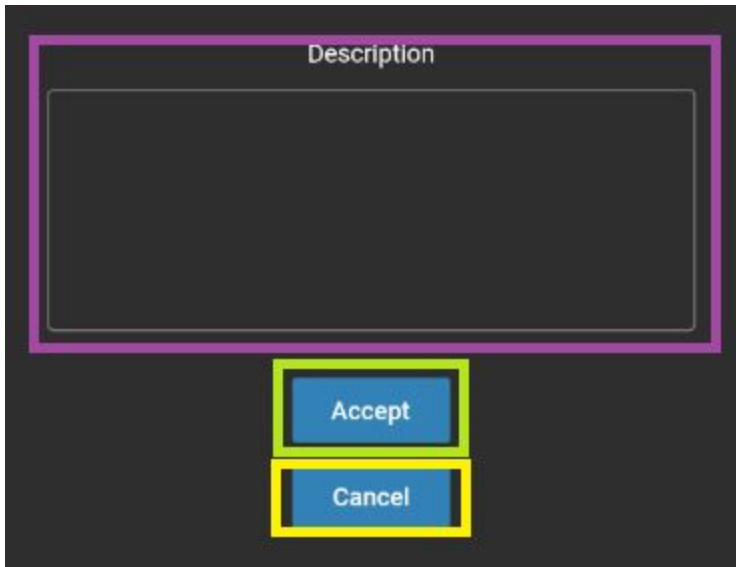


Figura 13. Ejemplo contraste historia de usuario MFH-02 con la aplicación móvil.

5.4 Validar el cumplimiento de los requerimientos de las aplicaciones web y móvil con los líderes de desarrollo para que puedan ser usadas como referente técnico en los proyectos emergentes.

Para la validación, se hicieron un conjunto de casos de prueba que hacen contraste a los requisitos que se definieron en el primer objetivo específico.

5.4.1 Formato de caso de prueba

El formato usado para definir los casos de prueba se presenta a continuación, y para ver la totalidad de éstos, diríjase al anexo 4.

Tabla 12. Ejemplo de caso de prueba.

Código Caso de prueba: CP-01	Funcionalidad: Login
Descripción	Se debe acceder con un usuario ya registrado a la aplicación web/móvil.
Datos relevantes	Se debe ingresar un correo electrónico y una contraseña que se encuentren almacenadas en la base de datos.
Resultado esperado	La pantalla de Top héroes debe cargarse.

6 Resultados y análisis

El producto final consta de una aplicación web y una aplicación móvil que está basada en un conjunto de buenas prácticas definidas por Cidenet S.A.S y sus códigos fuentes se pueden encontrar en los siguientes repositorios en GitLab.

- Aplicación móvil: <https://gitlab.com/cidenetlabs/get-started/mobile/flutter/hero>
- Aplicación web: <https://gitlab.com/cidenetlabs/get-started/web/angular/hero>
- Microservicio de seguridad: <https://gitlab.com/cidenetlabs/get-started/backend/java/spring-boot/security>
- Microservicio de perfiles: <https://gitlab.com/cidenetlabs/get-started/backend/java/spring-boot/hero>

Como ya se mencionó en este informe, la verificación del correcto funcionamiento del código se midió con la implementación de pruebas unitarias, que tuvieron una cobertura del 100% para ambas aplicaciones. Además, con la ayuda de la integración continua se ejecutaron estas pruebas unitarias en cada rama, donde cada una corresponde a una historia de usuario. De igual manera, una vez todas las ramas fueron mezcladas a la rama principal, se obtuvo el mismo resultado de una cobertura del 100%.

En las figuras 14 y 15 se muestran los últimos merge request que se hicieron a las aplicaciones web y móvil respectivamente, donde se puede observar la cobertura de ambas.

Overview 1 Commits 6 Pipelines 6 Changes 29 1 unresolved thread

Request to merge feature/jumunoz/login into develop

Pipeline #176801013 passed for c61662dc on feature/jumunoz/login
Coverage 100.00% (0.00%) from 1 job

Approval is optional
View eligible approvers

Merged by Juan Camilo Peña Vahos 3 weeks ago Revert Cherry-pick
The changes were merged into develop with 53c8fc14
The source branch has been deleted

Pipeline #177092250 passed for 53c8fc14 on develop
Coverage 100.00% (0.00%) from 1 job

Figura 14. Último merge request de la aplicación web, con cobertura del 100%.

Overview 0 Commits 1 Pipelines 1 Changes 7

Request to merge feature/jumunoz/edit-profile into develop

Pipeline #188564323 passed for fd5faf9e on feature/jumunoz/edit-profile
Coverage 100.00% (0.00%) from 1 job

Merge request approved. Approved by

View eligible approvers

Merged by Pablo Muñoz Zapata 3 weeks ago Revert Cherry-pick
The changes were merged into develop with 39b2436f
The source branch has been deleted

Figura 15. Último merge request de la aplicación móvil, con cobertura del 100%

A continuación, se presentan de la figura 16 a la 19, una parte de la aplicación web y móvil desarrollada. Para ver su totalidad diríjase al anexo 5.

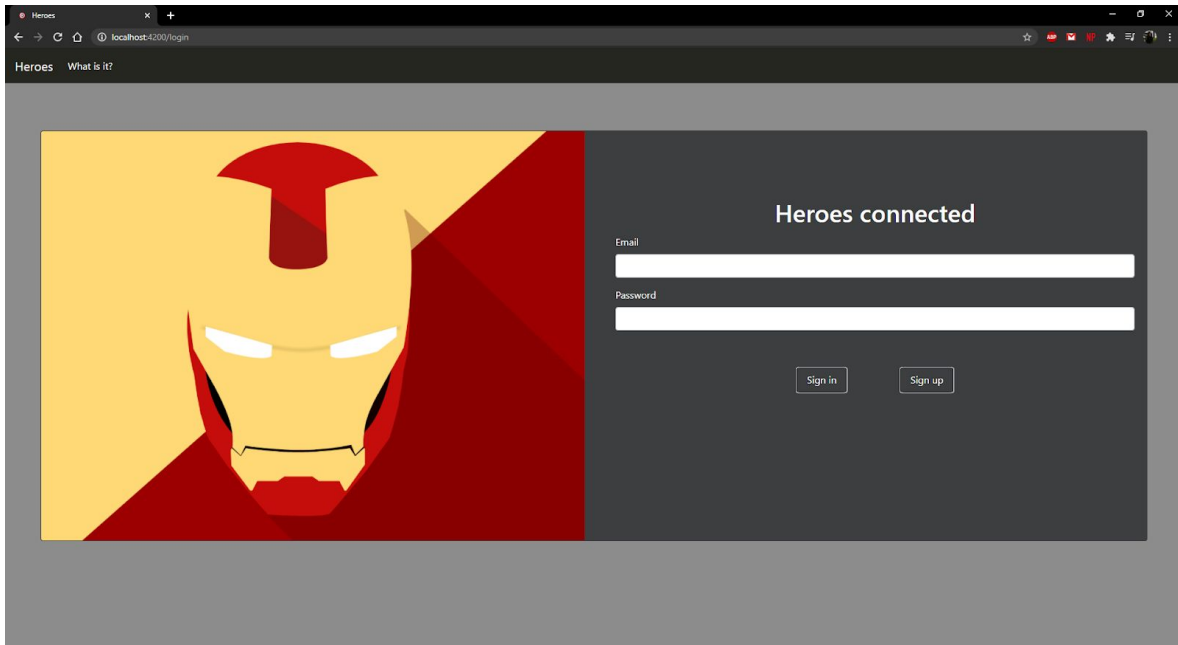


Figura 16. Pantalla de autenticación aplicación web.

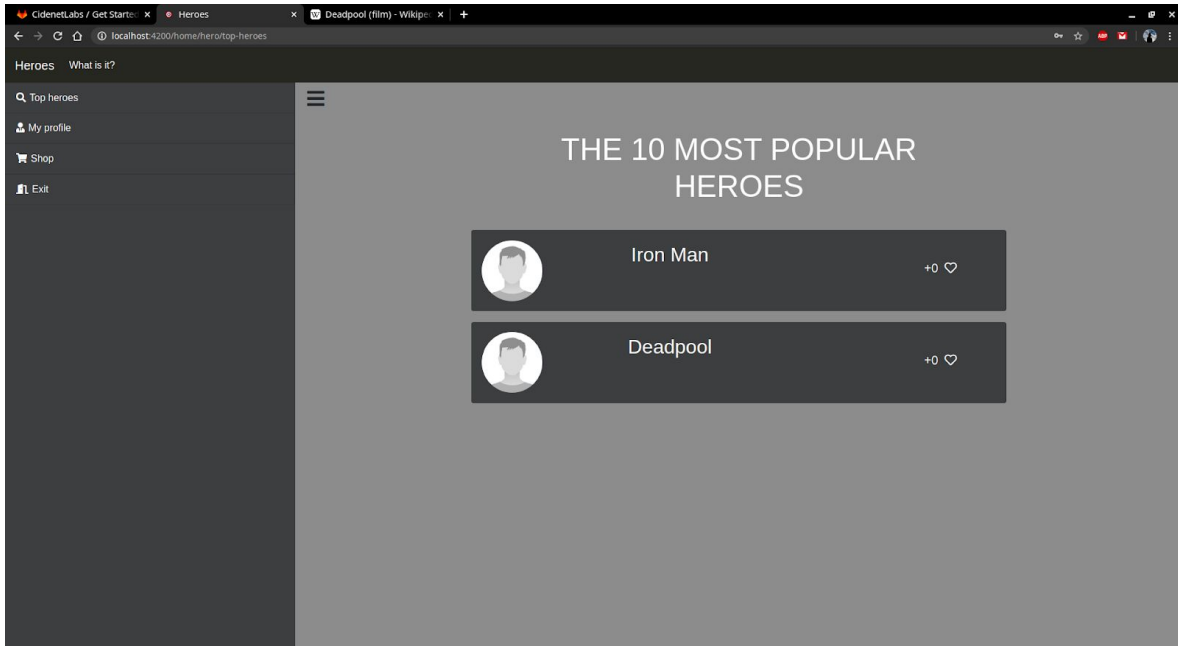


Figura 17. Pantalla de listado de héroes aplicación web.

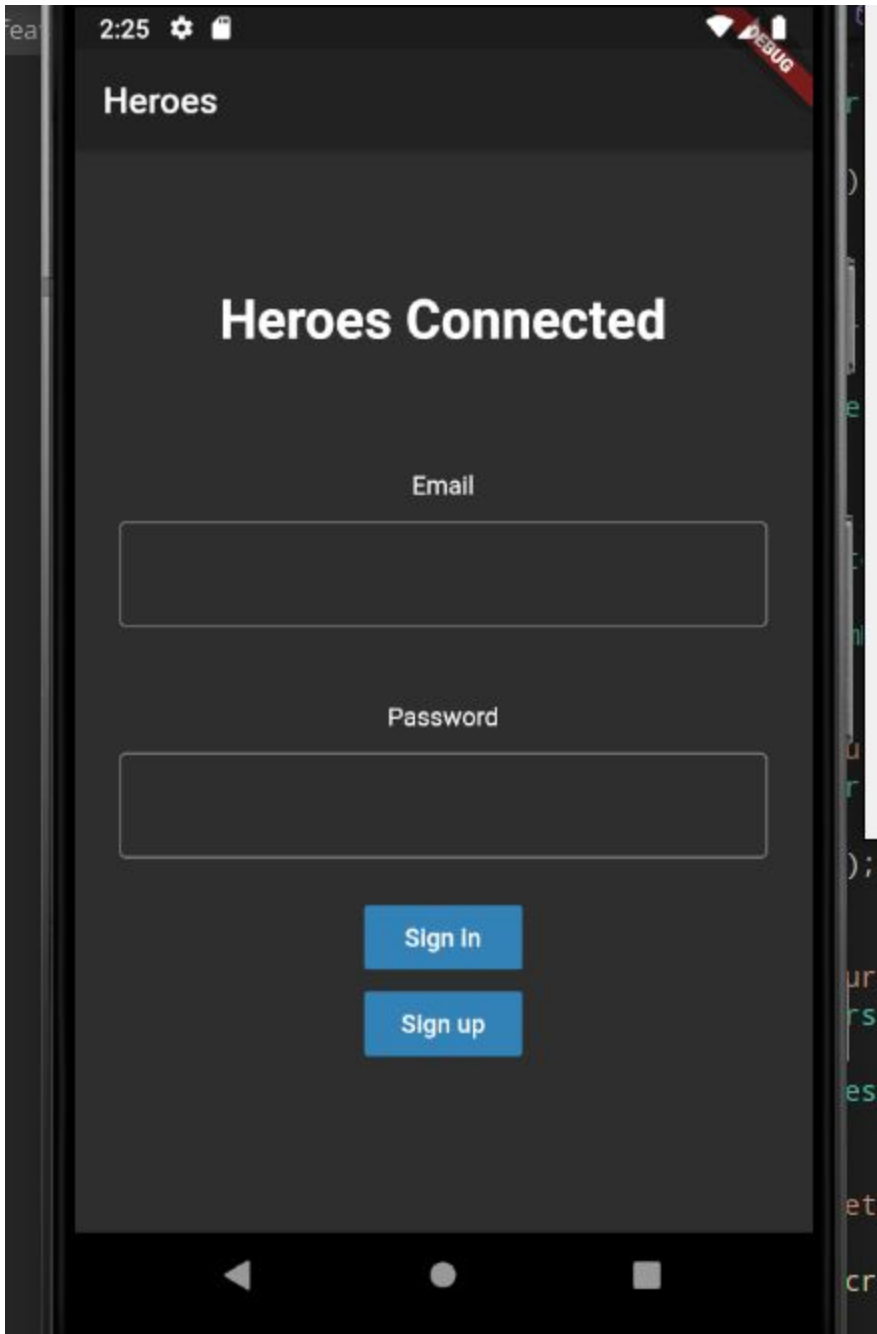


Figura 18. Pantalla de autenticación aplicación móvil.

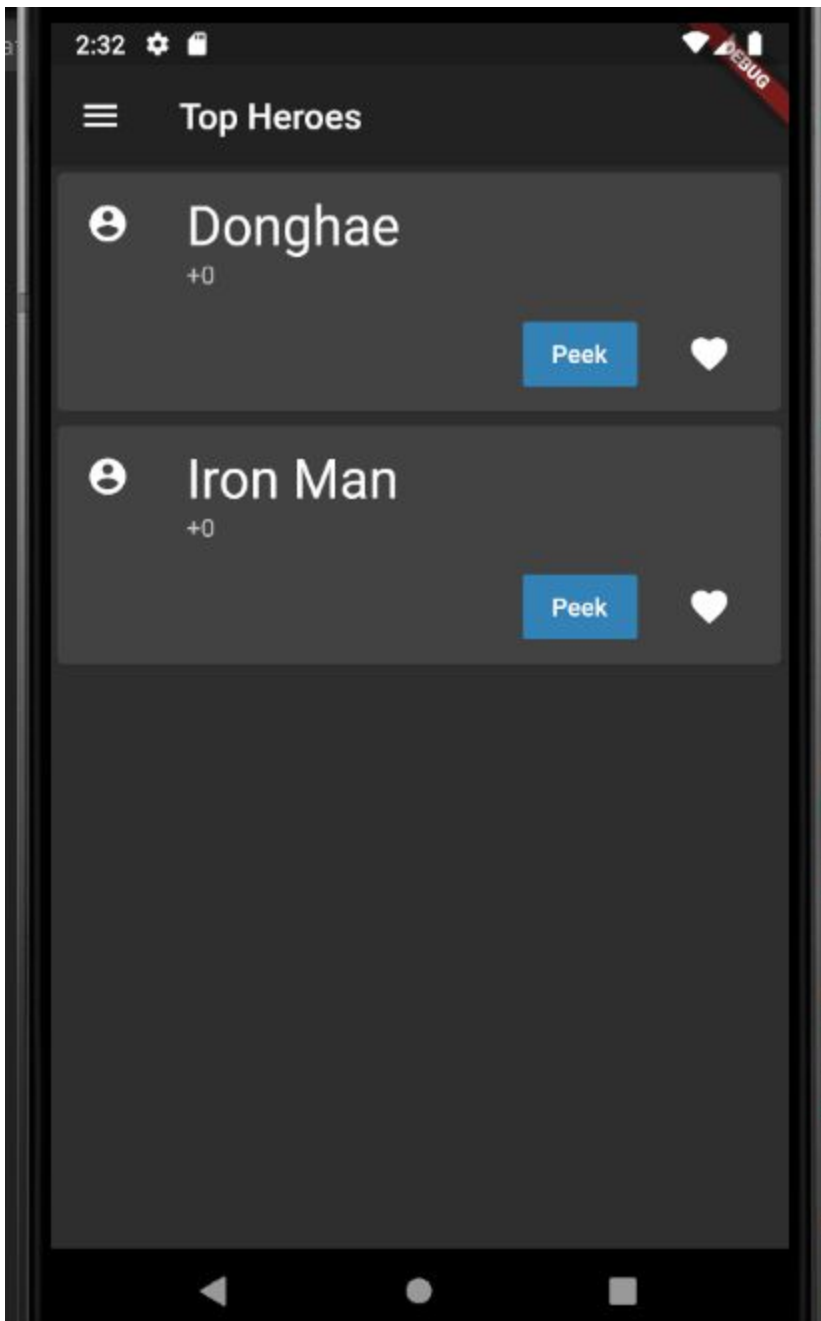


Figura 19. Pantalla de listado de héroes aplicación móvil.

7 Conclusiones

En desarrollo de software, los referentes técnicos de una tecnología u herramienta, son una colección de buenas prácticas adoptados y/o recomendadas por la comunidad, en este caso la empresa Cidenet S.A.S. En este trabajo se obtuvieron resultados concretos, de hecho, se desarrollaron dos aplicaciones en las tecnologías Angular y Flutter, siguiendo el marco de trabajo ágil SCRUM y por ello recopila documentación suficiente para exponer las buenas prácticas de la compañía en:

- Planeación y construcción de historias de usuario.
- Codificación en Angular y Flutter (Estándares y patrones de diseño).
- Prácticas en la gestión y versionamiento de código con tecnologías como Git y Gitlab.
- Testing convencional con casos de prueba y testing con pruebas unitarias.

8 Referencias Bibliográficas

[1] AWAD, Elias M. Knowledge Management. Pearson Education India, 2004. 480 p. ISBN: 8131714039

[2] MOLINA, Josi L y MARSAL. Montserrat. La gestión del conocimiento en las organizaciones. LibrosEnRed, 2001. 188 p. ISBN: 9871022603

[3] GEISLER, Eliezer y WICKRAMASINGHE, Nilmini. Principles of Knowledge Management: Theory, Practice, and Cases. Routledge, 2015. 400 p. ISBN: 1317415167

[4] Entramado. Cali, Colombia: Universidad Libre, enero, 2015, vol. 11, núm. 1. ISSN: 1900-3803

[5] LACEY, Mitch. The Scrum Field Guide: Agile Advice for Your First Year and Beyond. Addison-Wesley Professional, 2015. 99998 p. ISBN: 0133853713

[6] SUBRA, Jean-Paul y VANNIEUWENHUYSE, Aurélien. Scrum: Un método ágil para sus proyectos. Ediciones ENI, 2018. 246 p. ISBN: 2409012922

[7] CHRISTUDAS, Binildas. Practical Microservices Architectural Patterns: Event-Based Java Microservices with Spring Boot and Spring Cloud. Apress, 2019. 902 p. ISBN: 1484245016

Anexos

Anexo 1. Historias de usuario de las aplicaciones web y móvil.

Historias de usuario de la aplicación web

WAH-01 Top Heroes

As: Hero

Require: A top hero board.

To: View a list of the most popular heroes.

Acceptance requirements:

- Display a list of the top ten heroes order by the most popular.
- Display a list of all heroes with the ability to like them, each element of the list must contain:
 - Hero's photo
 - Like button
 - Like count
 - Hero's name

Reference Material:

Figura 20. Historia de usuario WAH-01.

WAH-02 Profile

As: Hero

Require: View my profile.

To: Review my information and skills.

Acceptance requirements:

- Display my information and skills.
- Display a photo of me.
- A button to edit my profile.

Reference Material:

Figura 21. Historia de usuario WAH-02.

WAH-03 Edit profile

As: Hero

Require: Edit my personal information and my skills.

To: Add or delete skills and change my profile photo.

Acceptance requirements:

- Fields with my information to change it.
- Field to change my profile photo.
- Field to search for a new skill.
- A button to cancel the transaction.
- A button to save the new information.

Reference Material:

Figura 22. Historia de usuario WAH-03.

WAH-08 Hero details

As: User

Require: Open a certain hero's profile

To: View hero's information, skills and photo.

Acceptance requirements:

- Display Hero's information, skills and photo.
- A button to go back.

Reference Material:

Figura 23. Historia de usuario WAH-08.

WAH-09 Sign up page

As: Hero

Require: Create an account

To: Access to the application

Acceptance requirements:

- Fields for the basic data:
 - Email
 - Password
 - Name (Confidential)
 - Hero name
 - Description
- A button to save the profile
- A button to cancel the transaction

Reference Material:

Figura 24. Historia de usuario WAH-09.

WAH-10 Login page

As: Hero

Require: Login into the app

To: See others heroes or edit his/her own profile

Acceptance requirements:

- Fields for the basic data:
 - Email
 - Password
- A button to sign in
- A button to sign up

Figura 25. Historia de usuario WAH-10.

Historias de usuario de la aplicación móvil

MFH-01 Login page

As: Hero

Require: A login page

To: Access to the application.

Acceptance requirements:

- Fields:
 - Email
 - Password
- A button to login.
- A button to create an account.

Reference Material:

Figura 26. Historia de usuario MFH-01.

MFH-02 Sign up page

As: Hero

Require: Create an account

To: Access to the application

Acceptance requirements:

- Fields for the basic data:
 - Email
 - Password
 - Name (Confidential)
 - Hero name
 - Description
- A button to save the profile
- A button to cancel the transaction

Reference Material:

Figura 27. Historia de usuario MFH-02.

MFH-03 Top Heroes

As: Hero

Require: A top hero board.

To: View a list of the most popular heroes.

Acceptance requirements:

- Display a list of the top ten heroes order by the most popular.
- Display a list of all heroes with the ability to like them, each element of the list must contain:
 - Hero's photo
 - Like button
 - Like count
 - Hero's name

Reference Material:

Figura 28. Historia de usuario MFH-03.

MFH-04 Hero details

As: User

Require: Open a certain hero's profile

To: View hero's information, skills and photo.

Acceptance requirements:

- Display Hero's information, skills and photo.
- A button to go back.

Reference Material:

Figura 29. Historia de usuario MFH-04.

MFH-05 Profile

As: Hero

Require: View my profile.

To: Review my information and skills.

Acceptance requirements:

- Display my information and skills.
- Display a photo of me.
- A button to edit my profile.

Reference Material:

Figura 30. Historia de usuario MFH-05.

MFH-06 Edit profile

As: Hero

Require: Edit my personal information and my skills.

To: Add or delete skills and change my profile photo and personal information.

Acceptance requirements:

- Fields with my information to change it.
- Field to change my profile photo.
- Field to search for a new skill.
- A button to cancel the transaction.
- A button to save the new information.

Reference Material:

Figura 31. Historia de usuario MFH-06.

Anexo 2. Contraste de historias de usuario y aplicación web.

Acceptance requirements:

- Display a list of the top ten heroes order by the most popular.
- Display a list of all heroes with the ability to like them, each element of the list must contain:
 - Hero's photo
 - Like button
 - Like count
 - Hero's name

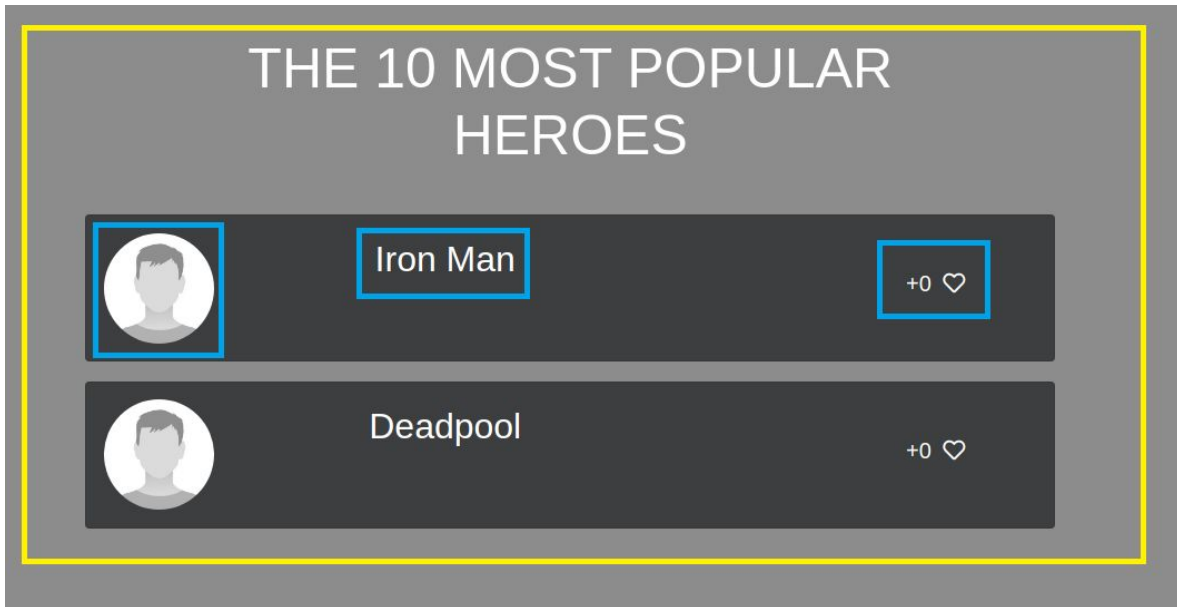


Figura 32. Contraste de historia de usuario WAH-01 con la aplicación web.

Acceptance requirements:

- Display Hero's information, skills and photo.
- A button to go back.

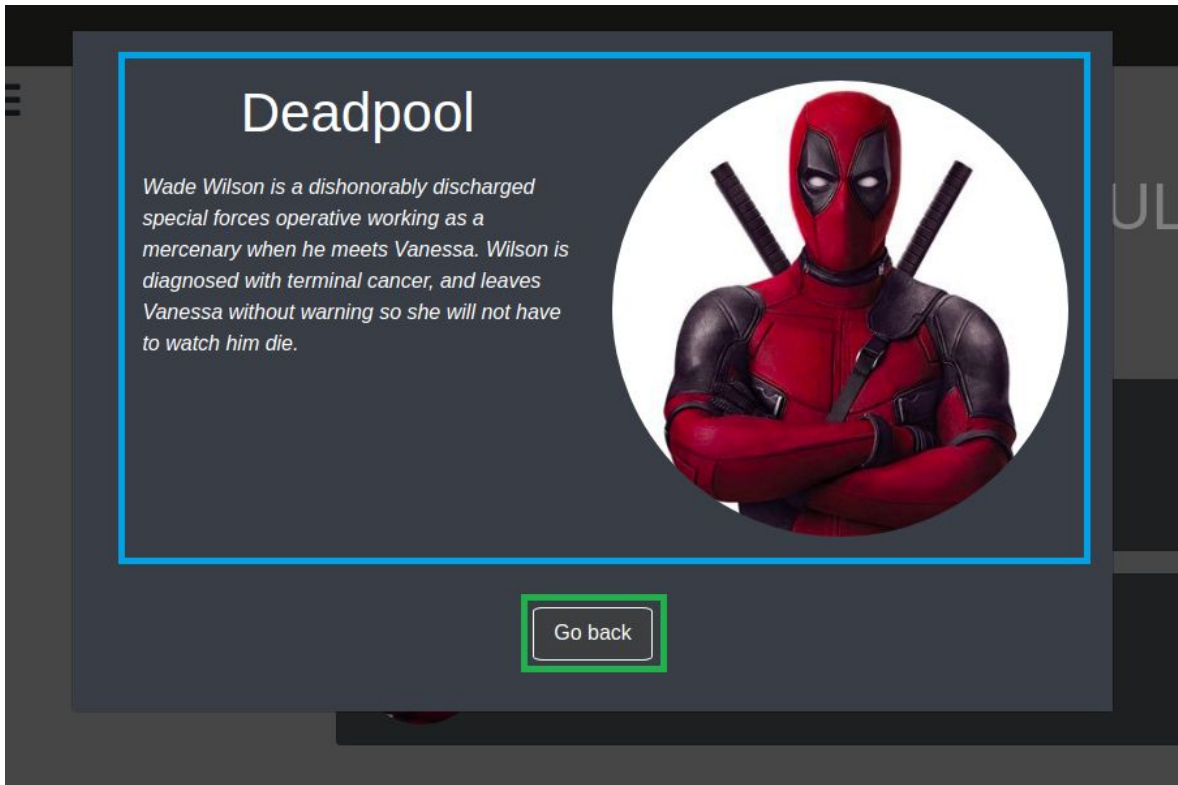
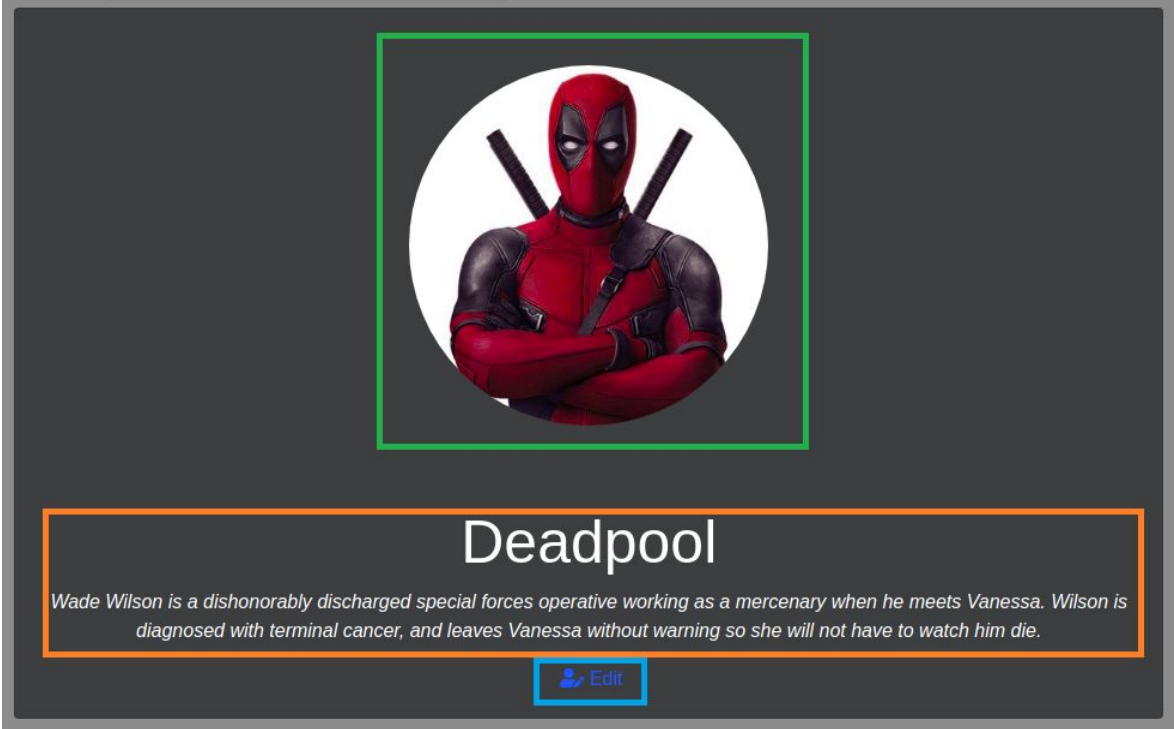


Figura 33. Contraste de historia de usuario WAH-08 con la aplicación web.

Acceptance requirements:

- Display my information and skills.
- Display a photo of me.
- A button to edit my profile.

Reference Material:



The image shows a user profile card for 'Deadpool'. At the top center is a circular profile picture of the character Deadpool, wearing his red and black suit with his arms crossed and two swords on his back. The profile picture is enclosed in a green rectangular border. Below the profile picture, the name 'Deadpool' is written in a large, white, sans-serif font. Underneath the name is a short biography in a smaller, italicized white font: 'Wade Wilson is a dishonorably discharged special forces operative working as a mercenary when he meets Vanessa. Wilson is diagnosed with terminal cancer, and leaves Vanessa without warning so she will not have to watch him die.' At the bottom center of the card is a small blue button with a white pencil icon and the word 'Edit' next to it.

Figura 34. Contraste de historia de usuario WAH-02 con la aplicación web.

Acceptance requirements:

- Fields with my information to change it.
- Field to change my profile photo.
- Field to search for a new skill.
- A button to cancel the transaction.
- A button to save the new information.

The image shows a web application interface for editing a user profile. The form is set against a dark grey background and contains several sections:

- Select a picture:** A white rectangular input field with a green border.
- Name:** A white input field containing the text "Iron Man" with a green border.
- Description:** A white text area containing the text "Tony is a genius engineer and wealthy owner of a technology company. Tony built the Iron Man suit when he was kidnapped and suffered an injury to his heart and gets his superpowers from it." with a yellow border.
- Skills:** A section with a white dropdown menu showing "Knowledge" and a plus icon. Below it are two blue pill-shaped buttons: "Money" and "Knowledge", each with a white 'x' icon.
- Save:** A white rectangular button with a red border.

Figura 35. Contraste de historia de usuario WAH-02 con la aplicación web.

Anexo 3. Contraste de historias de usuario y aplicación móvil.

Acceptance requirements:

- Display a list of the top ten heroes order by the most popular.
- Display a list of all heroes with the ability to like them, each element of the list must contain:
 - Hero's photo
 - Like button
 - Like count
 - Hero's name

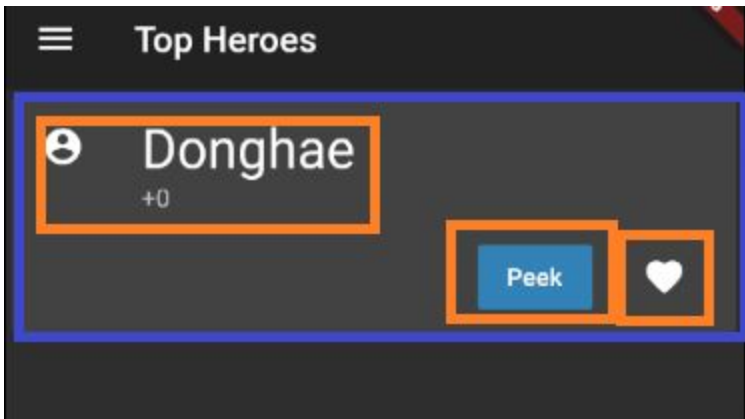


Figura 36. Contraste de historia de usuario MFH-03 con la aplicación móvil.

Acceptance requirements:

- Display Hero's information, skills and photo.
- A button to go back.

Reference Material:

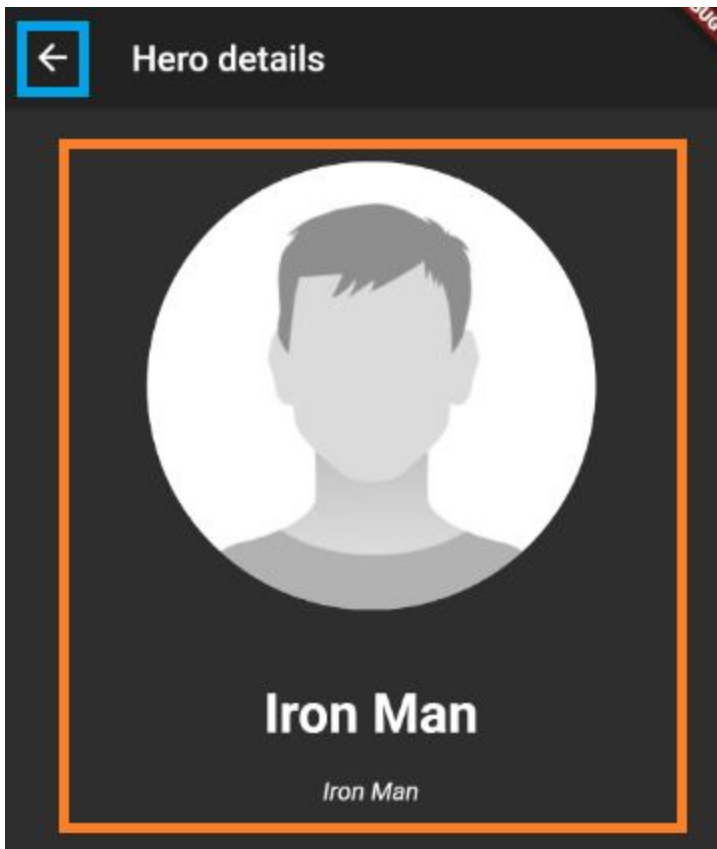


Figura 37. Contraste de historia de usuario MFH-04 con la aplicación móvil.

Acceptance requirements:

- Display my information and skills.
- Display a photo of me.
- A button to edit my profile.

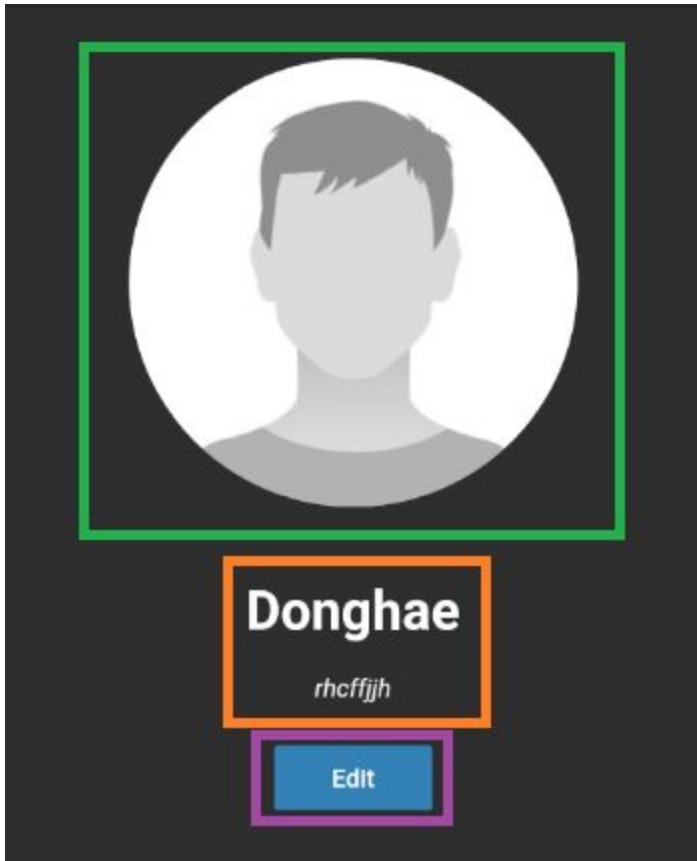


Figura 38. Contraste de historia de usuario MFH-05 con la aplicación móvil.

Acceptance requirements:

- Fields with my information to change it.
- Field to change my profile photo.
- Field to search for a new skill.
- A button to cancel the transaction.
- A button to save the new information.

The image shows a dark-themed user profile form with several highlighted elements. At the top, there is a section labeled "URL Image" with a corresponding empty input field, highlighted with a green border. Below this is a section labeled "Hero name" with an input field containing the text "Deadpool", highlighted with an orange border. Underneath is a "Description" section with a text area containing the text: "A dishonorably discharged special forces operative working as a mercenary when he meets Vanessa, a prostitute. They become romantically involved, and a year later she accepts his marriage proposal. Willson is". At the bottom of the form, there are two buttons: "Save" and "Cancel", both highlighted with yellow borders.

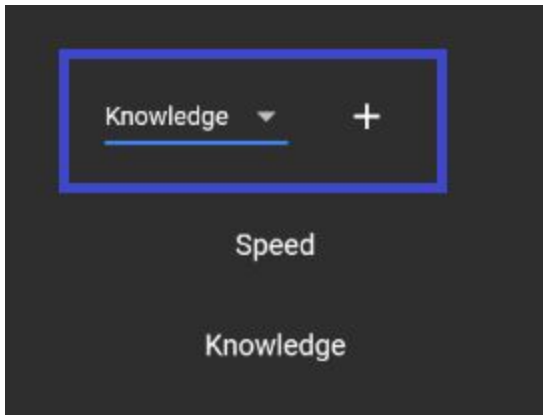


Figura 39. Contraste de historia de usuario MFH-06 con la aplicación móvil.

Anexo 4. Casos de prueba.

Tabla 13. Caso de prueba 02.

Código Caso de prueba: CP-02	Funcionalidad: Top Heroes
Descripción	Se debe garantizar que cualquier usuario pueda visualizar todos los perfiles registrados.
Datos relevantes	
Resultado esperado	Deben cargar los héroes con sus datos, en orden al puntaje que tengan.

Tabla 14. Caso de prueba 03.

Código Caso de prueba: CP-03	Funcionalidad: Heroes details
Descripción	Se debe garantizar que cualquier usuario puede ver en detalle la información de un héroe.
Datos relevantes	
Resultado esperado	Un componente modal/nueva pantalla con toda la información del héroe escogido.

Tabla 15. Caso de prueba 04.

Código Caso de prueba: CP-04	Funcionalidad: Edit profile
Descripción	Se debe garantizar que un usuario pueda modificar su información y sus habilidades.
Datos relevantes	Se deben poder modificar los siguientes datos: <ul style="list-style-type: none">• Nombre de héroe• Descripción• Foto de perfil• Agregar/eliminar habilidades
Resultado esperado	Mensaje de éxito y que los datos actualizados se muestren en el perfil.

Tabla 16. Caso de prueba 05.

Código Caso de prueba: CP-05	Funcionalidad: Profile
Descripción	Se debe garantizar que un usuario pueda visualizar su información.
Datos relevantes	
Resultado esperado	Pantalla con la foto de perfil, información y habilidades.

Tabla 17. Caso de prueba 06.

Código Caso de prueba: CP-06	Funcionalidad: Sign up
Descripción	Se debe garantizar que un nuevo usuario se pueda registrar en la aplicación web/móvil.
Datos relevantes	Se deben guardar los siguientes datos: <ul style="list-style-type: none">• Nombre completo• Nombre de héroe• Descripción• Email• Contraseña
Resultado esperado	Mensaje de éxito en el registro.

Anexo 5. Resultado final aplicaciones web y móvil.

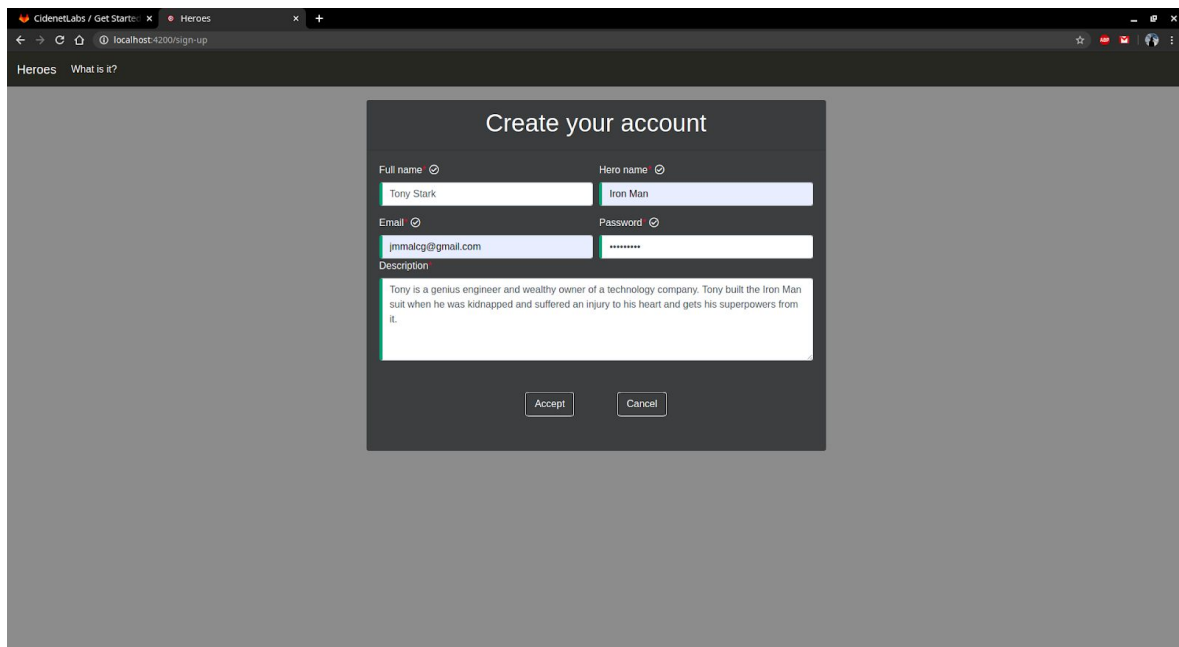


Figura 40. Pantalla de registro aplicación web.

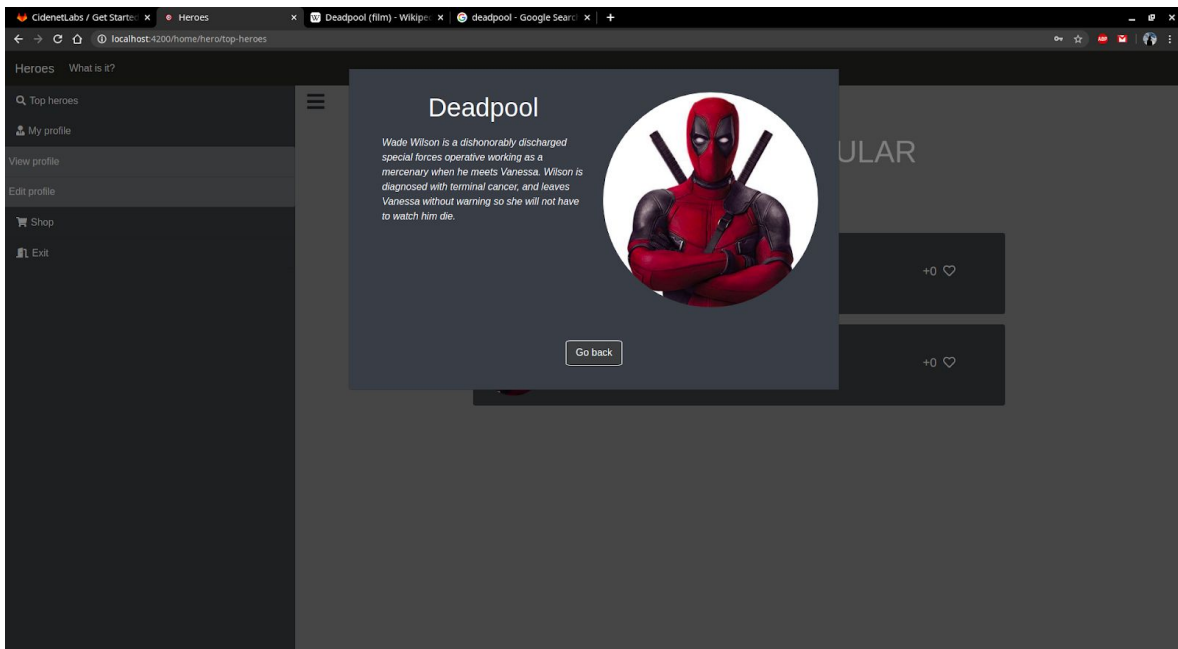


Figura 41. Pantalla de detalles de héroe aplicación web.

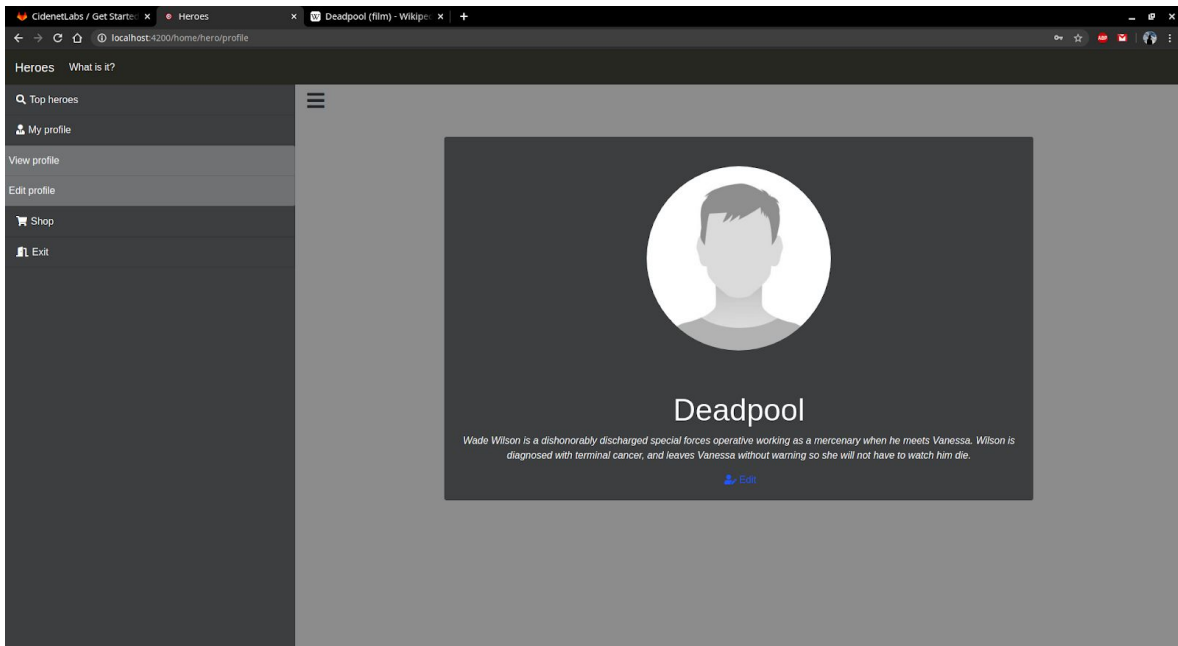


Figura 42. Pantalla de perfil aplicación web.

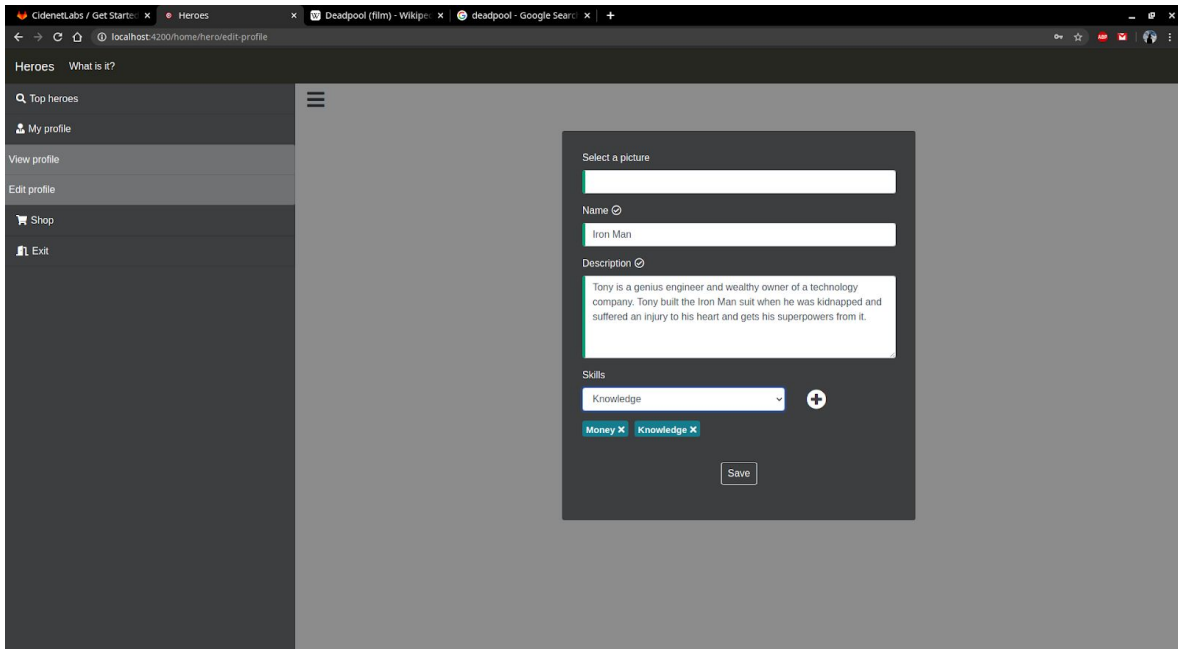


Figura 43. Pantalla de editar perfil aplicación web.

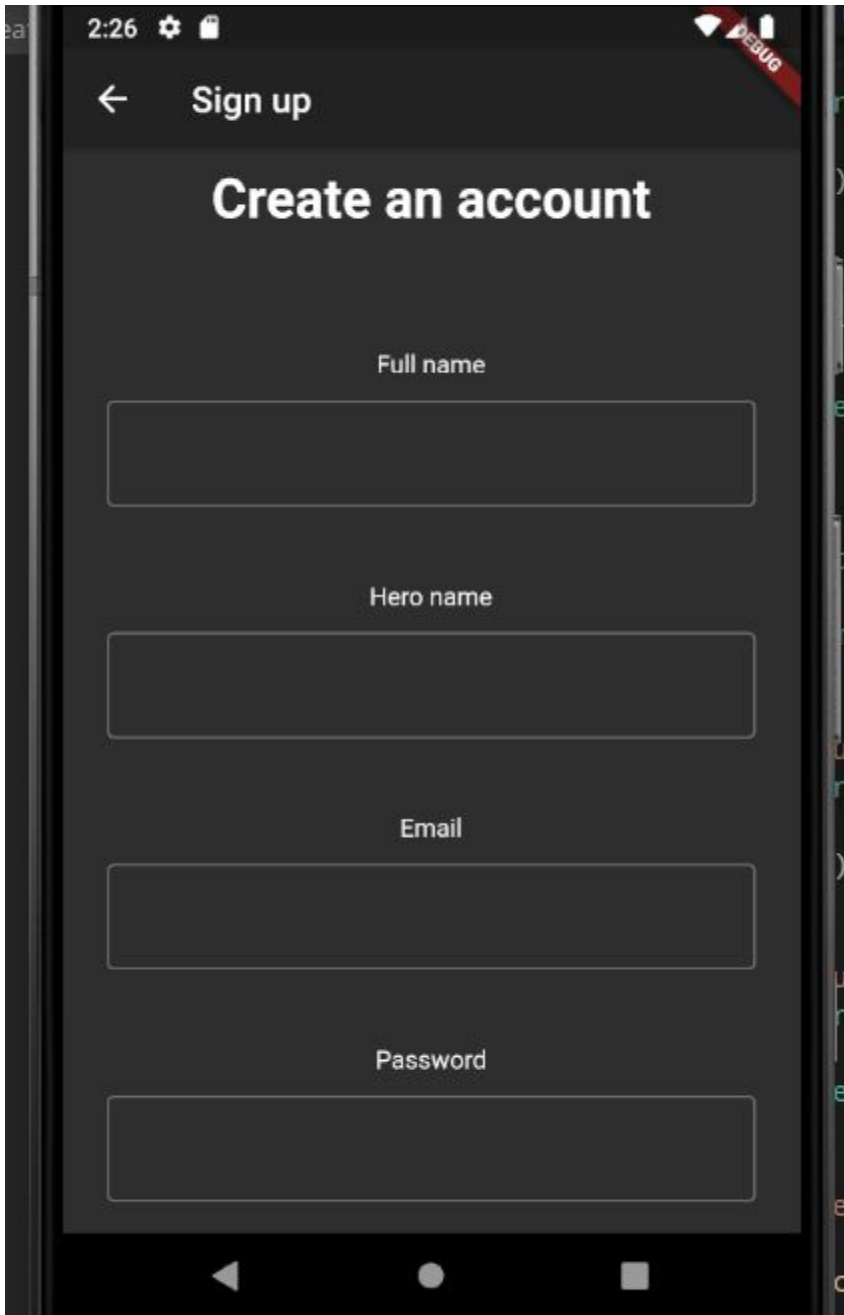


Figura 44. Pantalla de registro aplicación móvil.



Figura 45. Pantalla de detalles de héroe aplicación móvil.

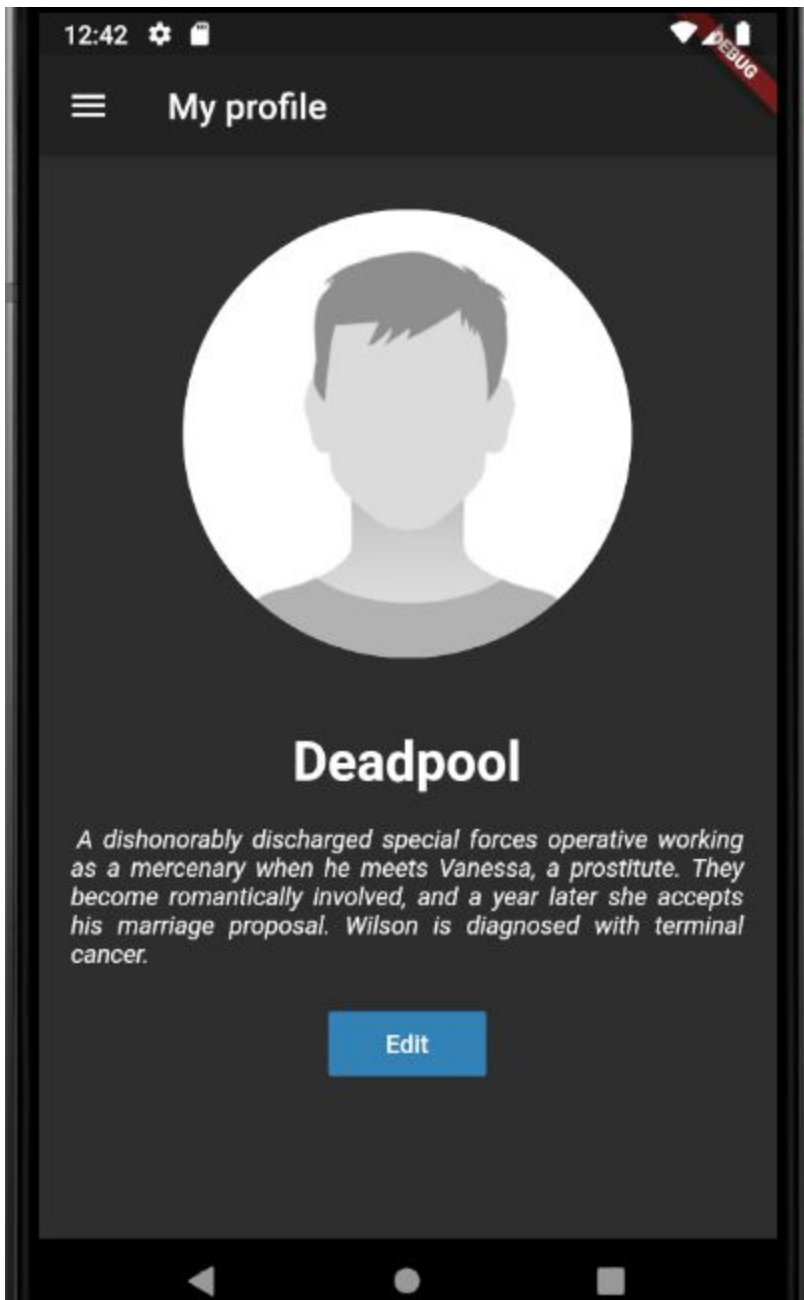


Figura 46. Pantalla de perfil aplicación móvil.

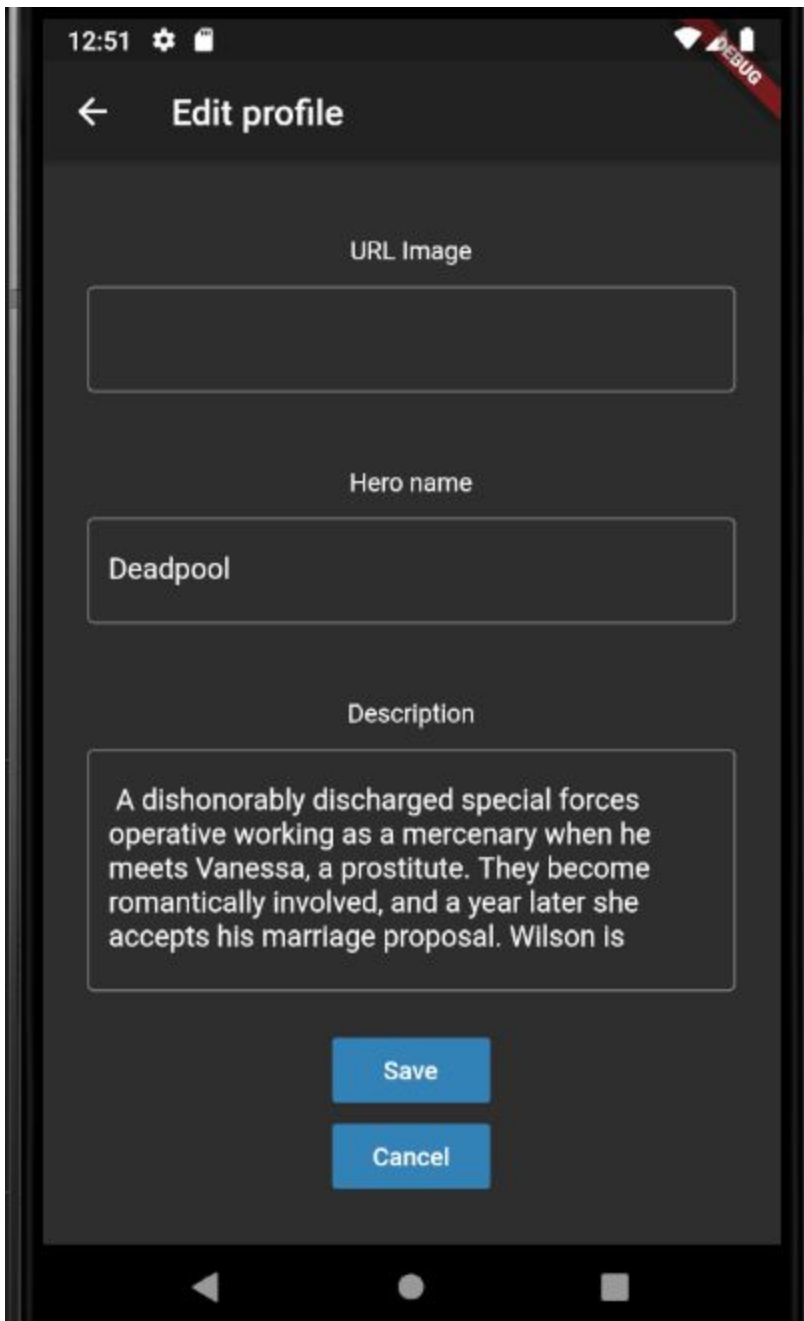


Figura 47. Pantalla de editar perfil aplicación móvil.

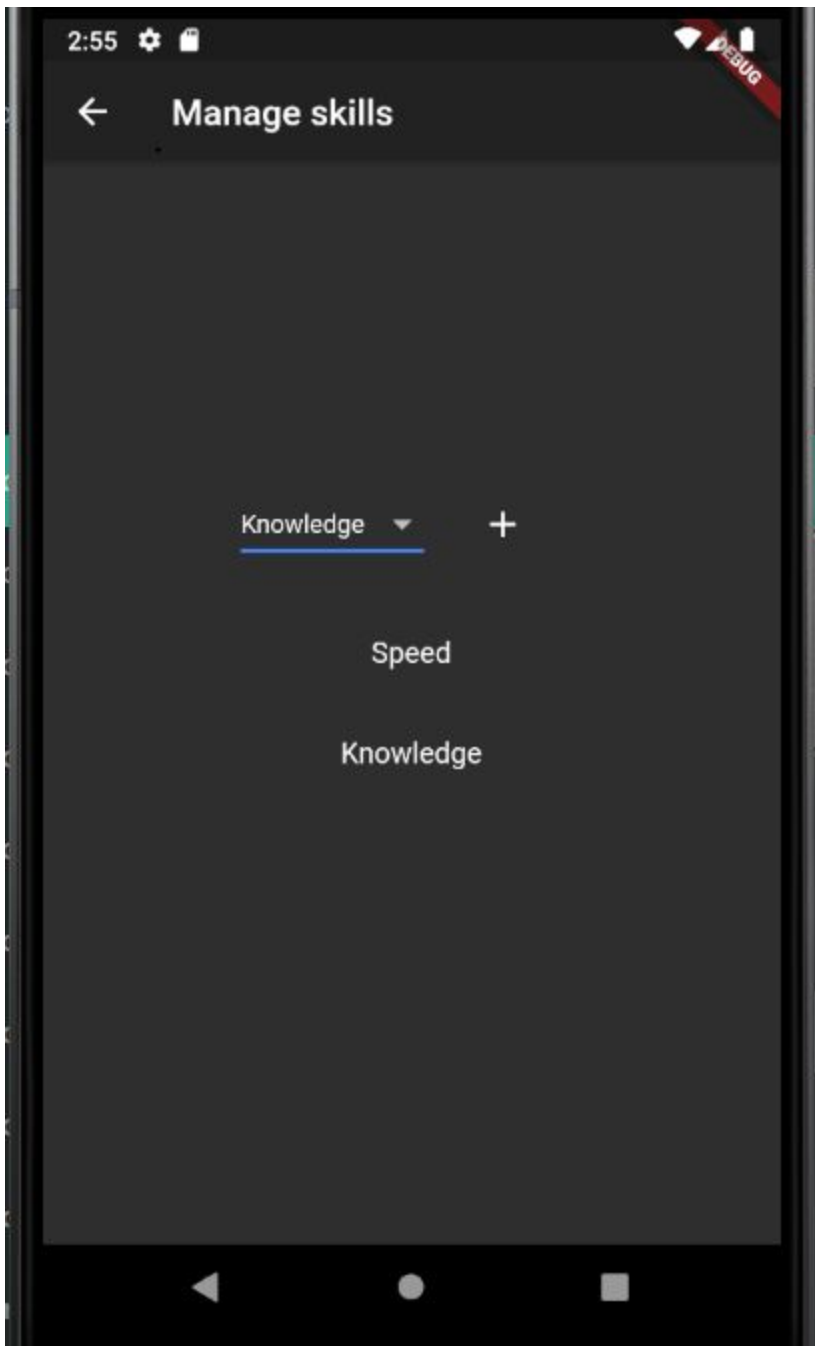


Figura 48. Pantalla de manejar habilidades aplicación móvil.