



Departamento de Informática
Universidad Técnica Federico Santa María



Entregable III

Proyecto: Gestión de pabellones y sillones
de quimioterapia
segfault

Integrantes:

Nombres y Apellidos	Email	ROL USM
Francisco Abarca Moraga	francisco.abarcamo@sansano.usm.cl	201673552-6
Ian Pérez Santis	ian.perez@sansano.usm.cl	201773549-K
José Miguel Quezada Silva	jose.quezada@sansano.usm.cl	201773528-7

1) Descripción del contexto de negocio

- **Contexto Recuperación:** Corresponde a la información de la sala de recuperación, provenientes de los datos manejados entre sus implementos (camas) y sus usuarios (pacientes), mediante sus respectivos identificadores. Esto a su vez es utilizado por el personal de salud encargado de esta área. Se asume que en este contexto siempre existirá disponibilidad de recursos y, por ende, no requiere de gestión, entendiéndose esta como el uso de métodos para la priorización y elección de pacientes para su asignación de recursos limitados.
 - *En este contexto se encuentran las existencias que se utilizan para llevar a cabo la recuperación de los pacientes luego de terminar su procedimiento.*

2) Definición API

ID	1
Acción/Objetivo	Obtener camas de recuperación en existencia
Verbo HTTP	GET
Argumentos	Recuperación/
Cuerpo de la petición	N.A
Cuerpo de la respuesta	<pre>[{ "id": <long>, "paciente": <long>, "condition": <boolean> }]</pre>

ID	2
Acción/Objetivo	Obtener cama de recuperación por ID
Verbo HTTP	GET
Argumentos	Recuperación/{id} Donde: <ul style="list-style-type: none"> Id: Identificador del registro (long)
Cuerpo de la petición	N.A
Cuerpo de la respuesta	<pre>{ "id": <long>, "paciente": <long>, "condition": <boolean> }</pre>

ID	3
Acción/Objetivo	Obtener camas por condición
Verbo HTTP	GET
Argumentos	Recuperación/condition={condition} Donde: <ul style="list-style-type: none">• condition: estado de ocupación de cama (boolean)
Cuerpo de la petición	N.A
Cuerpo de la respuesta	<pre>[{ "id": <long>, "paciente": <long>, "condition": <boolean> }]</pre>

ID	4
Acción/Objetivo	Obtener cama por paciente
Verbo HTTP	GET
Argumentos	Recuperación/paciente={paciente} Donde: <ul style="list-style-type: none">• Paciente: id de paciente (long)
Cuerpo de la petición	N.A
Cuerpo de la respuesta	<pre>{ "id": <long>, "paciente": <long>, "condition": <boolean> }</pre>

ID	5
Acción/Objetivo	Actualizar cama
Verbo HTTP	PUT
Argumentos	Recuperación/
Cuerpo de la petición	<pre>{ "paciente": <long>, "condition": <boolean> }</pre>
Cuerpo de la respuesta	<pre>{ "id": <long>, "paciente": <long>, "condition": <boolean> }</pre>

ID	6
Acción/Objetivo	Insertar nueva cama con paciente
Verbo HTTP	POST
Argumentos	Recuperación/
Cuerpo de la petición	<pre>{ "paciente": <long>, "condition": <boolean> }</pre>
Cuerpo de la respuesta	<pre>{ "id": <long>, "paciente": <long>, "condition": <boolean> }</pre>

ID	7
Acción/Objetivo	Insertar nueva cama vacía
Verbo HTTP	POST
Argumentos	Recuperación/
Cuerpo de la petición	N.A.
Cuerpo de la respuesta	<pre>{ "id": <long>, "paciente": <long>, "condition": <boolean> }</pre>

ID	8
Acción/Objetivo	Eliminar cama por identificador
Verbo HTTP	DELETE
Argumentos	Recuperación/{id} Donde: <ul style="list-style-type: none">• Id: identificador del registro (long)
Cuerpo de la petición	N.A.
Cuerpo de la respuesta	<pre>{ "success": <boolean> }</pre>

4) Pruebas

A continuación, se presentan los escenarios obtenidos al probar los métodos de la API desarrollada. Para cada uno, en general, se consideran ingresos de datos tanto válidos como inválidos.

1. Obtener camas de recuperación en existencia

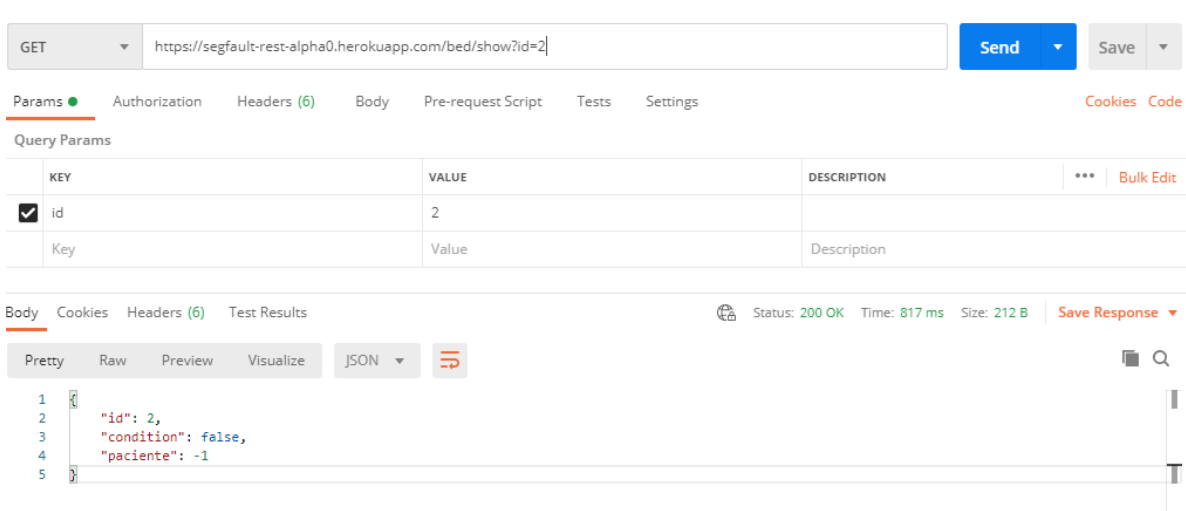
The screenshot shows a REST client interface with a GET request to `https://segfault-rest-alpha0.herokuapp.com/bed/list`. The response status is 200 OK, with a time of 174 ms and a size of 460 B. The response body is displayed in JSON format, showing a list of four beds.

KEY	VALUE	DESCRIPTION
Key	Value	Description

```
1 [
2   {
3     "id": 1,
4     "condition": false,
5     "paciente": -1
6   },
7   {
8     "id": 2,
9     "condition": false,
10    "paciente": -1
11  },
12  {
13    "id": 3,
14    "condition": false,
15    "paciente": -1
16  },
17  {
18    "id": 4,
19    "condition": false,
20    "paciente": -1
21  }
```

Se obtiene con éxito la lista de camas de la sala de recuperación.

2. Obtener cama de recuperación por ID



GET <https://segfault-rest-alpha0.herokuapp.com/bed/show?id=2> Send Save

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies Code

Query Params

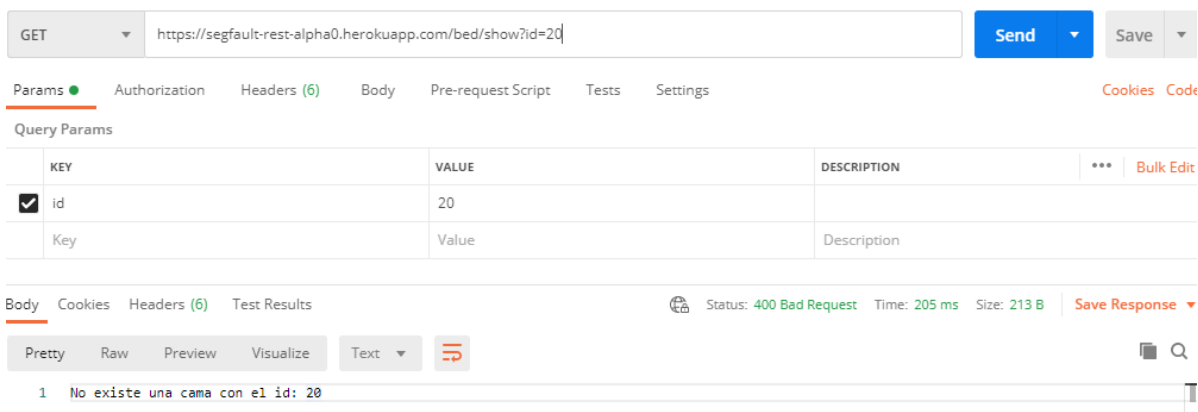
KEY	VALUE	DESCRIPTION
id	2	
Key	Value	Description

Body Cookies Headers (6) Test Results Status: 200 OK Time: 817 ms Size: 212 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 2,
3   "condition": false,
4   "paciente": -1
5 }
```

Se obtiene correctamente una camilla al ingresar su id correspondiente.



GET <https://segfault-rest-alpha0.herokuapp.com/bed/show?id=20> Send Save

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION
id	20	
Key	Value	Description

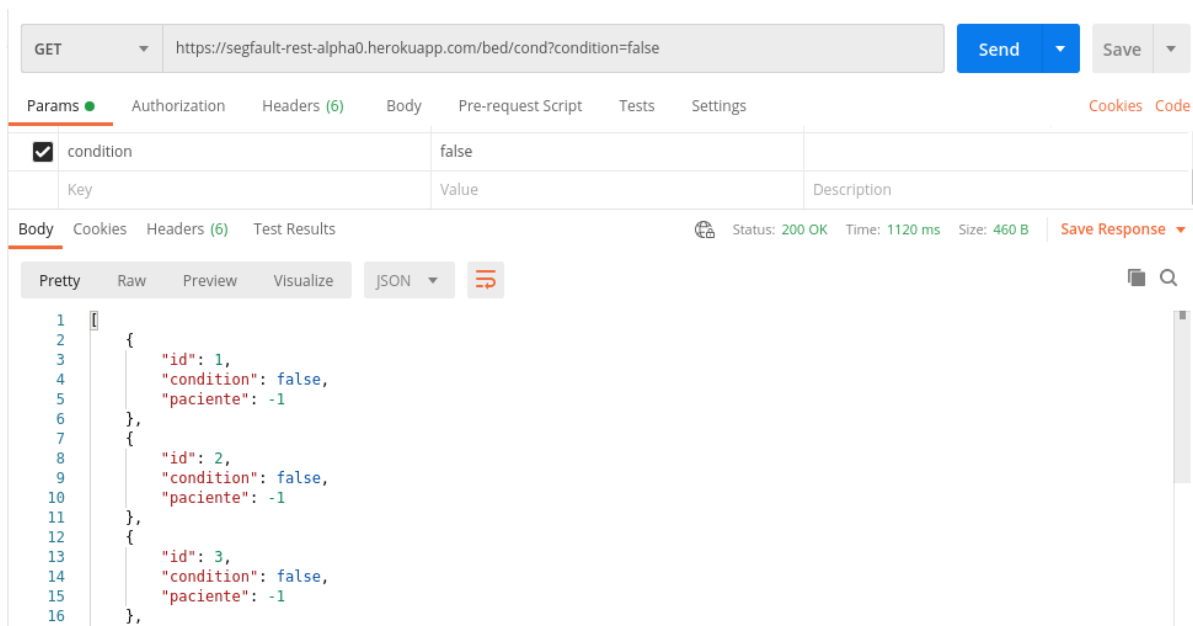
Body Cookies Headers (6) Test Results Status: 400 Bad Request Time: 205 ms Size: 213 B Save Response

Pretty Raw Preview Visualize Text

```
1 No existe una cama con el id: 20
```

Se obtiene un mensaje de error al ingresar un id inválido en la consulta.

3. Obtener camas por condición



GET <https://segfault-rest-alpha0.herokuapp.com/bed/cond?condition=false> Send Save

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies Code

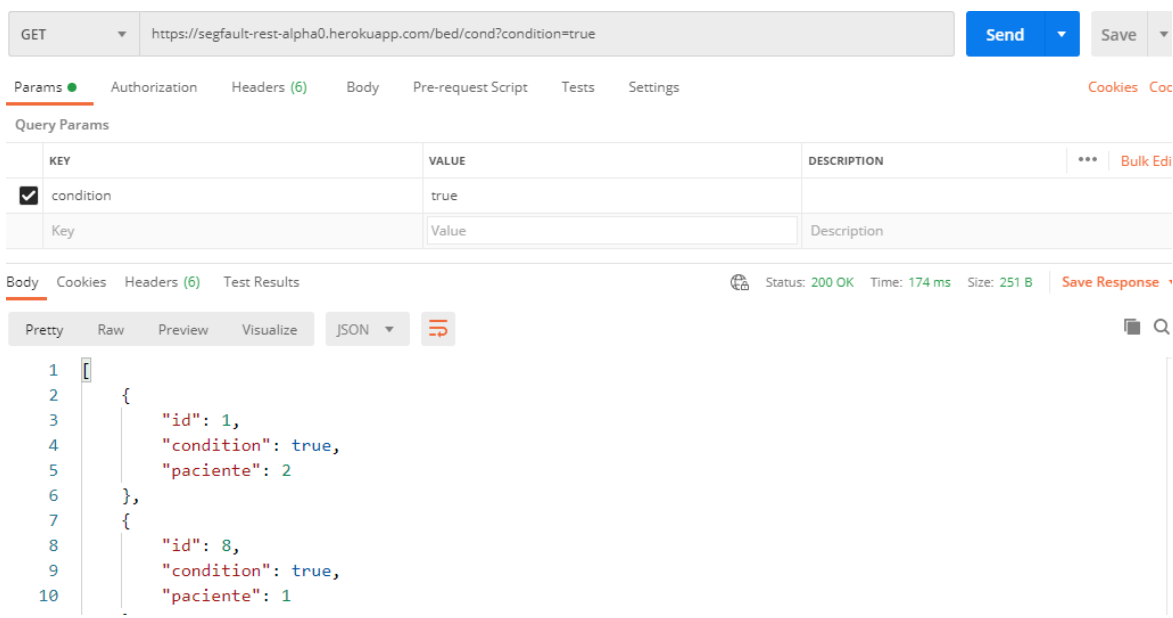
Key	Value	Description
condition	false	

Body Cookies Headers (6) Test Results Status: 200 OK Time: 1120 ms Size: 460 B Save Response

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "id": 1,
4     "condition": false,
5     "paciente": -1
6   },
7   {
8     "id": 2,
9     "condition": false,
10    "paciente": -1
11  },
12  {
13    "id": 3,
14    "condition": false,
15    "paciente": -1
16  },
17 ]
```

Se obtiene la lista de camas vacías y sus datos al ingresar la condición “false”.



GET <https://segfault-rest-alpha0.herokuapp.com/bed/cond?condition=true> Send Save

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies Coc

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
condition	true			

Body Cookies Headers (6) Test Results Status: 200 OK Time: 174 ms Size: 251 B Save Response

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "id": 1,
4     "condition": true,
5     "paciente": 2
6   },
7   {
8     "id": 8,
9     "condition": true,
10    "paciente": 1
11  },
12 ]
```

Se obtiene la lista de camas usadas y sus datos al ingresar la condición “true”.

GET <https://segfault-rest-alpha0.herokuapp.com/bed/cond?condition=vacio> Send Save

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> condition	vacio			
Key	Value	Description		

Body Cookies Headers (6) Test Results Status: 400 Bad Request Time: 172 ms Size: 293 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "timestamp": "2020-07-26T22:30:31.502+00:00",
3   "status": 400,
4   "error": "Bad Request",
5   "message": "",
6   "path": "/bed/cond"
7 }
```

Mensaje impreso al ingresar una condición no existente.

4. Obtener cama por paciente

GET <https://segfault-rest-alpha0.herokuapp.com/bed/paciente?paciente=1> Send Save

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> paciente	1			
Key	Value	Description		

Body Cookies Headers (6) Test Results Status: 200 OK Time: 171 ms Size: 210 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 8,
3   "condition": true,
4   "paciente": 1
5 }
```

Se obtiene exitosamente la cama asociada al paciente 5.

GET <https://segfault-rest-alpha0.herokuapp.com/bed/paciente?paciente=15> Send Save

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> paciente	15			
Key	Value	Description		

Body Cookies Headers (6) Test Results Status: 400 Bad Request Time: 151 ms Size: 219 B Save Response

Pretty Raw Preview Visualize Text

```
1 No existe una cama con el paciente: 15
```

Se obtiene mensaje de fracaso al intentar buscar una cama con un paciente que no se encuentra en ninguna. De todos modos, aún falta implementar el caso en que no exista el paciente buscado, una vez se sincronice con ese contexto.

5. Actualizar cama

The screenshot shows a REST client interface with a PUT request to `https://segfault-rest-alpha0.herokuapp.com/bed/?id=7&paciente=4`. The request body is a JSON object: `{ "id": 7, "condition": true, "paciente": 4 }`. The response status is 200 OK, with a time of 991 ms and a size of 210 B.

KEY	VALUE	DESCRIPTION
paciente	4	

```
1 {
2   "id": 7,
3   "condition": true,
4   "paciente": 4
5 }
```

Se obtienen las variables de la camilla en caso de ingresar un id válido.

The screenshot shows a PUT request to `https://segfault-rest-alpha0.herokuapp.com/bed/?id=24&paciente=3`. The response status is 400 Bad Request, with a time of 611 ms and a size of 213 B. The response body is a text message: "No existe una cama con el id: 24".

KEY	VALUE	DESCRIPTION
id	24	
paciente	3	

```
1 No existe una cama con el id: 24
```

Se muestra un mensaje de error en pantalla al ingresar un id que no corresponde a ninguna camilla.

The screenshot shows a PUT request to `https://segfault-rest-alpha0.herokuapp.com/bed/?id=7&paciente=-8`. The response status is 400 Bad Request, with a time of 974 ms and a size of 231 B. The response body is a text message: "No es posible realizar esa actualización de cama.".

KEY	VALUE	DESCRIPTION
id	7	

```
1 No es posible realizar esa actualización de cama.
```

6. Insertar nueva cama con paciente

POST <https://segfault-rest-alpha0.herokuapp.com/bed/add?condition=true&paciente=1> Send

Params ● Authorization Headers (7) Body Pre-request Script Tests Settings

Query Params

	KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/>	condition	true	
<input checked="" type="checkbox"/>	paciente	1	
	Key	Value	Description

Body Cookies Headers (6) Test Results Status: 200 OK Time: 792 ms Size: 210 B

Pretty Raw Preview Visualize JSON ≡

```
1 {  
2   "id": 8,  
3   "condition": true,  
4   "paciente": 1  
5 }
```

Se retorna la camilla creada, y sus propiedades.

POST <https://segfault-rest-alpha0.herokuapp.com/bed/add?condition=true&paciente=Juan> Send

Params ● Authorization Headers (7) Body Pre-request Script Tests Settings

Query Params

	KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/>	condition	true	
<input checked="" type="checkbox"/>	paciente	Juan	
	Key	Value	Description

Body Cookies Headers (6) Test Results Status: 400 Bad Request Time: 802 ms Size: 292 B

Pretty Raw Preview Visualize JSON ≡

```
1 {  
2   "timestamp": "2020-07-26T22:24:21.203+00:00",  
3   "status": 400,  
4   "error": "Bad Request",  
5   "message": "",  
6   "path": "/bed/add"  
7 }
```

Como no hay errores de condición o paciente no existentes, se prueba con un tipo de variable incorrecto: un String en una variable de tipo entero. El retorno es un error con un estado de tipo 400 (solicitud incorrecta).

7. Insertar nueva cama

The screenshot shows a REST client interface with a POST request to `https://segfault-rest-alpha0.herokuapp.com/bed/vacia`. The response status is 200 OK, with a time of 1285 ms and a size of 212 B. The response body is displayed in JSON format:

```
{
  "id": 1,
  "condition": false,
  "paciente": -1
}
```

Se retorna la nueva cama con sus datos correspondientes, no pudiendo encontrarse casos de error, ya que no se ingresan datos de forma manual.

8. Eliminar cama por identificador

The screenshot shows a REST client interface with a DELETE request to `https://segfault-rest-alpha0.herokuapp.com/bed?id=3`. The response status is 204 No Content, with a time of 183 ms and a size of 139 B. The response body is displayed in Text format:

```
1
```

Se retorna un 1 en el caso exitoso, representando un valor booleano verdadero.

GET <https://segfault-rest-alpha0.herokuapp.com/bed/show?id=3> Send Save

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> id	3	
Key	Value	Description

Body Cookies Headers (6) Test Results Status: 400 Bad Request Time: 156 ms Size: 212 B Save Response

Pretty Raw Preview Visualize Text

```
1 No existe una cama con el id: 3
```

Se comprueba que la cama eliminada no exista mediante una búsqueda por identificador.

DELETE <https://segfault-rest-alpha0.herokuapp.com/bed?id=19> Send Save

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> id	19	
Key	Value	Description

Body Cookies Headers (6) Test Results Status: 500 Internal Server Error Time: 168 ms Size: 309 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "timestamp": "2020-07-26T22:45:40.899+00:00",
3   "status": 500,
4   "error": "Internal Server Error",
5   "message": "",
6   "path": "/bed/"
7 }
```

Al intentar eliminar una cama con un id inválido, se retorna un error de estado 500 (Error interno de servidor), al no existir el objeto solicitado.

5) Planificación

Actividades que desarrollar

En la siguiente tabla se encuentran las tareas a desarrollar y sus prioridades.

Resumen	Clave de incidencia	ID de la incidencia	Tipo de Incidencia	Estado	Prioridad	Resolución	Creada	Resuelta
Generar un sistema que represente el sistema de recuperaciones del modelo de dominio	SEISW-1	12115	Épica	Por hacer	Highest	Por hacer	30-06	
Levantar entorno spring-boot	SEISW-2	12116	Tarea	Listo	High	Listo	30-06	01-07
Instalar JDK y Maven	SEISW-3	12117	Tarea	Listo	High	Listo	01-07	01-07
Configurar y enlazar BD	SEISW-4	12118	Tarea	Listo	Medium	Listo	01-07	04-07
Generar prototipo barebone funcional	SEISW-5	12119	Tarea	Listo	Medium	Listo	07-07	10-07
Desarrollar REST API del modelo de recuperaciones	SEISW-6	12120	Tarea	Listo	Medium	Listo	07-07	18-07
Subir REST API a Heroku	SEISW-7	12121	Tarea	Listo	Low	Listo	07-07	22-07
Añadir Swagger para documentación de la API al proyecto	SEISW-8	12122	Tarea	Listo	Lowest	Listo	18-07	22-07
Diseñar la interfaz web	SEISW-9	12123	Tarea	Por hacer	Low	Por hacer	09-07	
Implementar la interfaz web	SEISW-10	12124	Tarea	Por hacer	High	Por hacer	10-07	
Verificación de interfaz con los usuarios	SEISW-11	12125	Tarea	Por hacer	Low	Por hacer	11-07	
Integrar o exportar el microservicio a una plataforma común	SEISW-12	12126	Tarea	Por hacer	High	Por hacer	12-07	
Obtener feedback de lo desarrollado	SEISW-13	12127	Tarea	Por hacer	Medium	Por hacer	13-07	

Actividades realizadas

Dentro de las actividades realizadas se encuentra el primer sprint, este fue desde el 30 de junio al 20 de julio con el siguiente resultado:

Incidentes terminadas		Ver en el navegador de incidencias		
Clave	Resumen	Tipo de Incidencia	Prioridad	Estado
SEISW-2 *	Levantar entorno springboot	<input checked="" type="checkbox"/> Tarea	High	LISTO
SEISW-3 *	Instalar JDK y Maven	<input checked="" type="checkbox"/> Tarea	High	LISTO
SEISW-4 *	Configurar y enlazar BD	<input checked="" type="checkbox"/> Tarea	Medium	LISTO
SEISW-5 *	Generar prototipo barebone funcional	<input checked="" type="checkbox"/> Tarea	Medium	LISTO
SEISW-6 *	Desarrollar REST API del modelo de recuperaciones	<input checked="" type="checkbox"/> Tarea	Medium	LISTO

Incidentes Sin Completar		Ver en el navegador de incidencias		
Clave	Resumen	Tipo de Incidencia	Prioridad	Estado
SEISW-7 *	Subir REST API a Heroku	<input checked="" type="checkbox"/> Tarea	Low	EN PROGRESO
SEISW-8 *	Añadir Swagger para documentación de la API al proyecto	<input checked="" type="checkbox"/> Tarea	Lowest	EN PROGRESO

Las actividades que quedaron fueron incluidas en el próximo sprint el cual está en el siguiente estado en este momento:

Sprint 2		
Desarrollar el front-end e incorporar el microservicio al sistema final		
FILTROS RÁPIDOS: Sólo Mis Incidencias Recientemente Actualizadas		
POR HACER	EN PROGRESO	HECHO
<div>SEISW-10 Implementar la interfaz web</div> <div><input checked="" type="checkbox"/> High</div>	<div>SEISW-9 Diseñar la interfaz web</div> <div><input checked="" type="checkbox"/> Medium</div>	<div>SEISW-7 Subir REST API a Heroku</div> <div><input checked="" type="checkbox"/> Low</div>
<div>SEISW-11 Verificación de interfaz con los usuarios</div> <div><input checked="" type="checkbox"/> Medium</div>		<div>SEISW-8 Añadir Swagger para documentación de la API al proyecto</div> <div><input checked="" type="checkbox"/> Lowest</div>
<div>SEISW-12 Integrar o exportar el microservicio a una plataforma común</div> <div><input checked="" type="checkbox"/> High</div>		
<div>SEISW-13 Obtener feedback de lo desarrollado</div> <div><input checked="" type="checkbox"/> Medium</div>		

El sprint mostrado finaliza el 10 de agosto, algunas subtareas que salgan de las tareas pendientes se incluirán en Jira al ser alcanzadas durante el desarrollo del proyecto.