



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática



**TFG del Grado en Ingeniería
Informática**

SmartBeds

**Aplicación de técnicas de
minería de datos para la
detección de crisis epilépticas**



Presentado por José Luis Garrido Labrador
en Universidad de Burgos – 24 de abril
de 2019

Tutores: Dr. Álgvar Arnáiz González
y Dr. José Francisco Díez Pastor

Índice general

| | |
|---|------------|
| Índice general | I |
| Índice de figuras | III |
| Índice de tablas | IV |
| Apéndice A Plan de Proyecto Software | 1 |
| A.1. Introducción | 1 |
| A.2. Planificación temporal | 1 |
| A.3. Estudio de viabilidad | 5 |
| Apéndice B Especificación de Requisitos | 7 |
| B.1. Introducción | 7 |
| B.2. Objetivos generales | 7 |
| B.3. Catalogo de requisitos | 7 |
| B.4. Especificación de requisitos | 9 |
| Apéndice C Especificación de diseño | 19 |
| C.1. Introducción | 19 |
| C.2. Diseño de datos | 19 |
| C.3. Diseño procedimental | 20 |
| C.4. Diseño arquitectónico | 20 |
| C.5. Diseño de interfaces | 20 |
| Apéndice D Documentación técnica de programación | 23 |
| D.1. Introducción | 23 |
| D.2. Estructura de directorios | 23 |

| | |
|--|-----------|
| D.3. Manual del programador | 23 |
| D.4. Compilación, instalación y ejecución del proyecto | 26 |
| D.5. Pruebas del sistema | 26 |
| Apéndice E Documentación de usuario | 27 |
| E.1. Introducción | 27 |
| E.2. Requisitos de usuarios | 27 |
| E.3. Instalación | 27 |
| E.4. Manual del usuario | 27 |
| Bibliografía | 29 |

Índice de figuras

| | |
|--|----|
| B.1. Diagramas de casos de uso | 10 |
| B.2. Diagramas de casos de uso | 11 |
| C.1. Diagrama entidad-relación | 19 |
| C.2. Diagrama relacional | 20 |
| C.3. Prototipo para escritorio | 21 |
| C.4. Prototipo para móvil | 22 |

Índice de tablas

| | |
|--|----|
| B.1. Caso de uso 1: Loguear | 9 |
| B.2. Caso de uso 2: Visualizar de datos | 10 |
| B.3. Caso de uso 2.1: Elegir cama | 11 |
| B.4. Caso de uso 2.2: Ver datos en tiempo real | 12 |
| B.5. Caso de uso 3: Administrar de usuarios | 12 |
| B.6. Caso de uso 3.1: Añadir usuarios | 13 |
| B.7. Caso de uso 3.2: Modificar contraseña | 14 |
| B.8. Caso de uso 3.3: Borrar usuario | 14 |
| B.9. Caso de uso 4: Administrar de camas | 15 |
| B.10.Caso de uso 4.1: Añadir cama | 16 |
| B.11.Caso de uso 4.2: Modificar cama | 17 |
| B.12.Caso de uso 4.3: Borrar cama | 17 |
| B.13.Caso de uso 4.4: Asignar cama a usuario | 18 |
| D.1. Especificaciones del API | 26 |

Apéndice A

Plan de Proyecto Software

A.1. Introducción

A.2. Planificación temporal

El proyecto se desarrolló siguiendo una metodología ágil, ligeramente basada en *Scrum*. Se dividió el progreso en *Sprints*, cada uno con una serie de tareas y su estimación en esfuerzo. Esta estimación o peso, se ha evaluado según la dificultad que se preveía tener, no como una medida horaria, esto es debido que algunas tareas simples tardan más en desarrollarse por la necesidad de la ejecución automática mientras que otras que requirieron mucho más trabajo se ejecutaron mucho antes.

A continuación se mostrarán las listas de las tareas realizadas con los enlaces a las *issues* del repositorio *GitHub*. Los *Sprints* fueron:

Sprint 1 - 10/11/18 hasta 20/11/18

En este primer *Sprint* solamente hubo dos tareas:

1. **Hacer exploración bibliográfica**
2. **Configurar repositorio de Git**

Sprint 2 - 21/11/18 hasta 05/12/18

Se desarrollaron 4 tareas, todas de investigación, por lo que no hay *commits* asociados, para esto las tareas fueron:

3. Lectura de "Automated Epileptic Seizure Detection Methods: A Review Study"
4. Exploración Bibliográfica - Orientado a caídas
5. Lectura del primer tema de "Minería de Datos"
6. Configurar VPN en Archlinux

Sprint 3 - 06/12/18 hasta 21/12/18

En este *sprint* se comenzó la documentación migrando las plantillas al repositorio y continuó la investigación a un área más técnica, las tareas fueron:

7. Iniciar documentación
8. Tabla de extracción de características
9. Búsqueda de librerías con funciones sofisticadas

Sprint 4 - 22/12/18 hasta 08/01/2019

Este *sprint* fue del aprendizaje de técnicas de minería de datos aplicada. Las tareas realizadas fueron:

10. Instalacion de entorno python
11. Graficar datos mediante PCA
12. Aprender el uso de librerías

Sprint 5 - 09/01/2019 hasta 17/01/2019

En este *sprint* el objetivo fue probar distintas lineas de investigación para ver las características intrínsecas a los datos.

13. Configurar acceso a gamma
14. Leer apuntes de Minería de Datos
15. Filtrado y suavizado de datos
16. Probar otras formas de proyección

Sprint 6 - 18/01/2019 hasta 24/01/2019

Tras descartar algunas proyecciones se exploraron las más prometedoras y se probaron nuevos filtros y detección de anomalías. Las tareas fueron por consiguiente:

17. Mejor preprocesamiento
18. Dibujado alrededor de las crisis
19. Probar formas de filtrado
20. Estudiar puntos clave de las proyecciones
21. Probar detección de anomalías por one-class

Sprint 7 - 25/01/2019 hasta 07/02/2019

Este *sprint* tuvo una duración mayor debido a que durante el periodo señalado se realizó un curso en la universidad donde se estudiaron las series temporales. Por este motivo el servidor donde se han estado realizando las ejecuciones no estaba disponible retrasando la ejecución de los experimentos.

Las tareas se centraron en comprobar las proyecciones más interesantes con datos estadísticos además de profundizar en *One-Class* o detección de anomalías. Las tareas realizadas fueron:

22. Proyección LTSA con valores estadísticos
23. Documentar Sprints del 1 al 6
24. Documentar investigación con Alicia
25. One Class - Mejorar la investigación de este apartado
26. Transformers
27. Trasladar códigos a transformers

Sprint 8 - 8/02/2019 hasta 14/2/2019

Este *sprint* fue dedicado a seguir desarrollando el estudio de One-Class cuyos resultados se pueden ver en el apéndice *Cuaderno de Investigación*. Las tareas fueron:

28. Particionar los datos
29. Estudiar one-class¹
30. One-Class con crisis
31. One-Class sin crisis
32. Preprocesado básico para one-class
 - a) Bruto
 - b) Filtrado *Butter* de 3 y 0.05
 - c) Filtrado *SavGol* de tamaño 15
 - d) Concatenación de estadísticos en ventana 25 sobre bruto
 - e) Concatenación de estadísticos con ventana 25 sobre *Butter* de 3 y 0.05
 - f) Concatenación de estadísticos con ventana 25 sobre *SavGol* de tamaño 15
33. Testear clasificadores con otras crisis

Las tareas de la 30 a la 33 se incluyeron en la tarea épica 29 aunque esta tuvo tareas del siguiente *sprint*

Sprint 9 - 15/2/2019 a 21/2/2019

Este *sprint* trató de completar la linea de investigación de *One-Class* abierta en la tarea épica 29 y documentar los resultados de *sprints* anteriores.

34. Documentar PCA y Proyecciones de 2-Variedad
35. Entrenamiento One-Class con dos crisis
36. Testeo One-Class con tercera crisis

¹Tarea épica

37. Calcular área bajo la curva con entrenamiento de una clase

38. Visualización de constantes vitales

Son las tareas de la 35 a la 37 las que forman parte de la tarea épica 29.

A.3. Estudio de viabilidad

Viabilidad económica

Viabilidad legal

Apéndice *B*

Especificación de Requisitos

B.1. Introducción

B.2. Objetivos generales

B.3. Catalogo de requisitos

En esta sección se presentan los requisitos funcionales y los no funcionales

Requisitos funcionales

- **RF-1 Confidencialidad del sistema:** solamente usuarios permitidos podrán acceder al sistema.
 - **RF-1.1 Identificación de usuario:** los usuarios se identificarán con un *nickname* y una contraseña
 - **RF-1.2 Rol de administración:** existirá un usuario especial que podrá administrar el sistema completamente sin restricciones.
 - **RF-1.3 Visualización de una cama:** los usuarios validados deben poder observar los datos en tiempo real de las camas disponibles.
 - **RF-1.4 Restricción de acceso:** los usuarios solamente podrán tener acceso a los datos de las camas permitidas.
 - **RF-1.5 Acceso completo al administrador:** el administrador debe poder acceder a todas las camas existentes.
- **RF-2 Gestión de las camas:** el administrador ha de gestionar las camas pudiendo añadir, modificar y borrar.

- **RF-2.1 Añadir cama:** el administrador ha de poder añadir una nueva cama al sistema.
- **RF-2.2 Modificar cama:** el administrador ha de poder modificar los datos una cama existente.
- **RF-2.3 Borrar cama:** el administrador ha de poder borrar una cama del sistema.
- **RF-2.4 Asignar camas a usuarios:** el administrador se encarga de decidir que usuario puede acceder a que cama.
- **RF-3 Gestión de los usuarios:** el administrador ha de gestionar los usuarios pudiendo añadir, modificar y borrar.
 - **RF-3.1 Añadir usuario:** el administrador ha de poder añadir un nuevo usuario al sistema.
 - **RF-3.2 Modificar usuario:** el administrador ha de poder modificar los datos un usuario existente.
 - **RF-3.3 Borrar usuario:** el administrador ha de poder borrar un usuario del sistema.
- **RF-4 Visualización de los datos:** los usuarios han de poder ver de las camas disponibles el estado actual del paciente, sus constantes vitales y las presiones.

Requisitos no funcionales

- **RNF-1 Usabilidad:** la aplicación debe cumplir estándares de usabilidad teniendo una curva de aprendizaje baja y un uso de metáforas adecuado.
- **RNF-2 Disponibilidad:** las camas existentes han de ser siempre accesibles por sus usuarios asociados y dar una información correcta de su estado
- **RNF-3 Confidencialidad:** los datos de las camas, al ser en parte constantes vitales de pacientes, solamente han de ser accesibles por los usuarios permitidos.
- **RNF-4 Escalabilidad:** el sistema debe ser escalable para adaptarse mejor a un incremento de carga del sistema.
- **RNF-5 Seguridad:** los usuarios deben poder identificarse sólidamente con el sistema sin que sus datos o sus credenciales (*tokens*) sean accesibles por terceros, incluso el administrador.
- **RNF-6 Exstensibilidad:** la API del sistema debe ser fácilmente extensible a nuevas funcionalidades incorporando de manera eficaz soporte a nuevas peticiones.

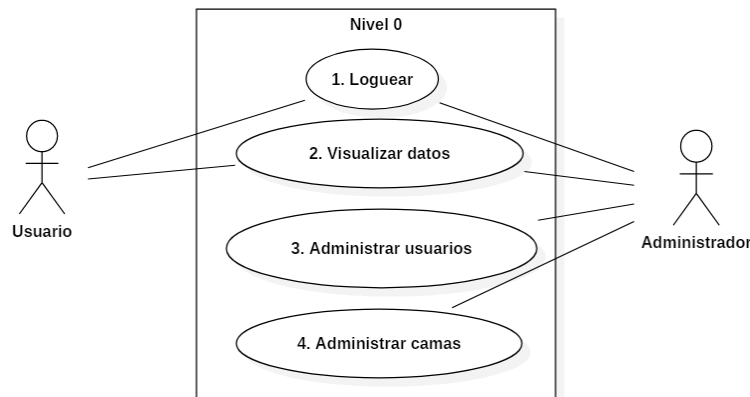
- **RNF-7 Persistencia:** los servicios de procesamiento de las camas activas deben mantenerse funcionando aunque no existan clientes activos para evitar retrasos muy altos ante nuevas conexiones.
- **RNF-8 Fiabilidad:** los datos de la aplicación son correctos y actuales además de garantizar una predicción óptima del estado del paciente.

B.4. Especificación de requisitos

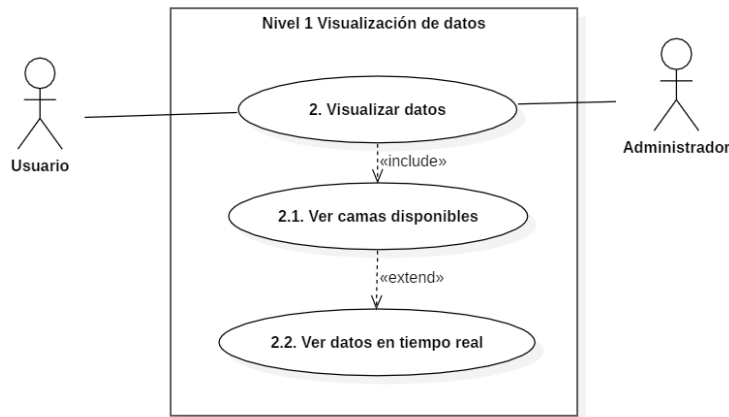
Los requisitos funcionales generan un conjunto de casos de uso que serán la base del desarrollo de la aplicación. La especificación de los mismos se encuentran entre la tabla B.1 y la tabla B.13. La representación gráfica se puede ver en los diagramas de las figuras B.1 y B.2. Existen dos actores, el **administrador** que se encarga de toda la labor de gestión tanto de usuarios como de camas y el **usuario** que únicamente puede gestionarse a si mismo y ver los datos de las camas que tenga permitidas.

| | | |
|------------------|---|--|
| CU-1: Loguear | | |
| Descripción | El usuario se identifica en el sistema | |
| Precondiciones | No existe una sesión activa válida | |
| Requisitos | RF-1, RF-1.1 | |
| Usuario | Anónimo | |
| Secuencia normal | Paso | Acción |
| | 1 | El cliente envía sus credenciales al servidor |
| | 2 | El servidor acepta las credenciales devolviendo el token de sesión |
| Postcondiciones | El usuario tiene una sesión activa válida | |
| Excepciones | Paso | Acción |
| | 2 | Si las credenciales son incorrectas el servidor responde con error |
| Frecuencia | Alta | |
| Importancia | Crítico | |
| Comentarios | Es siempre lo primero que aparecerá | |

Tabla B.1: Caso de uso 1: Loguear



(a) Diagrama casos de uso - Nivel 0

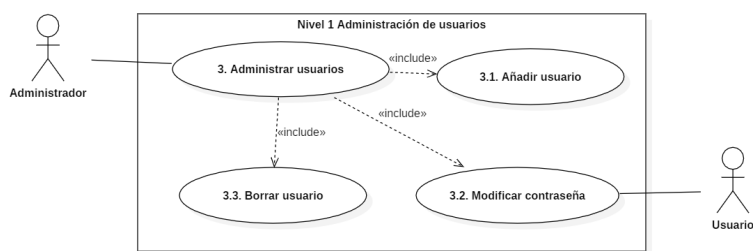


(b) Diagrama CU-2 - Nivel 1

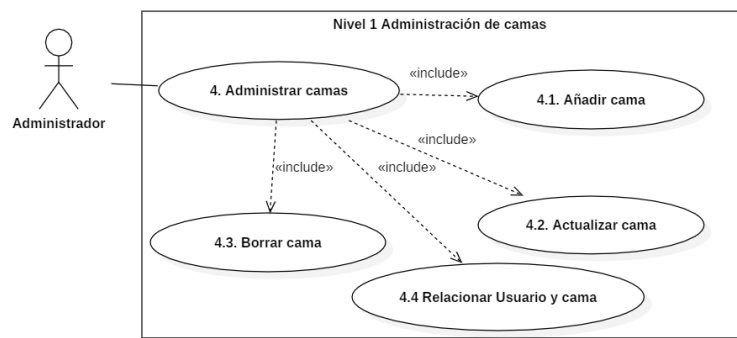
Figura B.1: Diagramas de casos de uso

| | |
|---------------------------|--|
| CU-2: Visualizar de datos | |
| Descripción | Ver lista de las camas disponibles |
| Precondiciones | Sesión activa válida |
| Requisitos | RF-1.3, RF-1.4 |
| Usuario | Administrador y Usuario |
| Secuencia normal | Paso Acción |
| | 1 El cliente solicita ver las camas disponibles |
| | Postcondiciones El cliente está en la pantalla de camas disponibles |
| Frecuencia | Alta |
| Importancia | Alta |

Tabla B.2: Caso de uso 2: Visualizar de datos



(a) Diagrama CU-3 - Nivel 1



(b) Diagrama CU-4 - Nivel 1

Figura B.2: Diagramas de casos de uso

| | | |
|---------------------|--|---|
| CU-2.1: Elegir cama | | |
| Descripción | Elegir cama | |
| Precondiciones | Sesión activa válida | |
| Requisitos | RF-1.3, RF-1.4, RF-4 | |
| Usuario | Logueado | |
| Secuencia normal | Paso | Acción |
| | 1 | El cliente solicita ver las camas disponibles |
| | 2 | El servidor abre conexiones paralelas para actualizar en tiempo real el estado de las camas |
| | 3 | El cliente decide que cama ver |
| Postcondiciones | El cliente entra en la ventana de los datos en tiempo real | |
| Frecuencia | Alta | |
| Importancia | Alta | |

Tabla B.3: Caso de uso 2.1: Elegir cama

| | | |
|----------------------------------|---|--|
| CU-2.2: Ver datos en tiempo real | | |
| Descripción | Ver datos en tiempo real | |
| Precondiciones | Sesión activa válida y cama existente y accesible | |
| Requisitos | RF-1.3, RF-1.4, RF-4 | |
| Usuario | Administrador y usuario | |
| Secuencia normal | Paso | Acción |
| | 1 | El cliente solicita una nueva conexión |
| | 2 | El servidor provee una conexión en tiempo real con los datos |
| Postcondiciones | El usuario tiene una conexión paralela abierta con los datos en tiempo real | |
| Excepciones | Paso | Acción |
| | 2 | Si un paquete faltase o la señal fuera débil se alertaría al usuario |
| Frecuencia | Alta | |
| Importancia | Máxima | |

Tabla B.4: Caso de uso 2.2: Ver datos en tiempo real

| | | |
|-------------------------------|--|---|
| CU-3: Administrar de usuarios | | |
| Descripción | Administración de usuario: alta, baja y modificación | |
| Precondiciones | Sesión de administrador válida | |
| Requisitos | RF-3 | |
| Usuario | Administrador | |
| Secuencia normal | Paso | Acción |
| | 1 | El administrador entra en el menú de administración de usuarios |
| Postcondiciones | El administrador está en el menú de administración de usuarios | |
| Frecuencia | Baja | |
| Importancia | Alta | |

Tabla B.5: Caso de uso 3: Administrar de usuarios

| | | |
|-------------------------|---------------------------------------|--|
| CU-3.1: Añadir usuarios | | |
| Descripción | Añadir usuarios | |
| Precondiciones | Sesión de administración activa | |
| Requisitos | RF-3.1 | |
| Usuario | Administrador | |
| Secuencia normal | Paso | Acción |
| | 1 | El administrador elige añadir un nuevo usuario |
| | 2 | Se introduce un nombre de usuario para identificarlo |
| | 3 | Se introduce una contraseña dos veces |
| | 4 | Se almacenan los datos |
| Postcondiciones | Existe un nuevo usuario en el sistema | |
| Excepciones | Paso | Acción |
| | 2 | Si el nickname existiese |
| | 3 | La contraseña añadida no coincide en las dos ocasiones |
| Frecuencia | Baja | |
| Importancia | Alta | |

Tabla B.6: Caso de uso 3.1: Añadir usuarios

| | | |
|------------------------------|---|--|
| CU-3.2: Modificar contraseña | | |
| Descripción | Cambiar la contraseña de un usuario | |
| Precondiciones | Sesión activa válida, usuario existente | |
| Requisitos | RF-3.2 | |
| Usuario | Administrador y Usuario | |
| Secuencia normal | Paso | Acción |
| | 1 | Si es usuario normal ir a 3 |
| | 2 | Si es administrador elegir a que usuario cambiar la contraseña |
| | 3 | Se introduce una contraseña nueva dos veces |
| | 4 | Se actualizan los datos |
| Postcondiciones | La contraseña ha cambiado | |
| Excepciones | Paso | Acción |
| | 3 | La contraseña añadida no coincide en las dos ocasiones |
| Frecuencia | Baja | |
| Importancia | Alta | |

Tabla B.7: Caso de uso 3.2: Modificar contraseña

| | | |
|------------------------|--|--|
| CU-3.3: Borrar usuario | | |
| Descripción | Elimina un usuario de la base de datos | |
| Precondiciones | Sesión de administración válida, usuario existente | |
| Requisitos | RF-3.3 | |
| Usuario | Administrador | |
| Secuencia normal | Paso | Acción |
| | 1 | Elegir a que usuario (no administrador) eliminar |
| | 2 | Eliminar usuario y todos los datos vinculados |
| Postcondiciones | El usuario ha sido eliminado | |
| Frecuencia | Baja | |
| Importancia | Media | |

Tabla B.8: Caso de uso 3.3: Borrar usuario

| CU-4: Administrar de camas | | |
|----------------------------|---|--|
| Descripción | Administración de camas: alta, baja, modificación y asignación a usuarios | |
| Precondiciones | Sesión de administración válida | |
| Requisitos | RF-2 | |
| Usuario | Administrador | |
| Secuencia normal | Paso | Acción |
| | 1 | El administrador entra en el menú de administración de camas |
| Postcondiciones | El administrador está en el menú de administración de camas | |
| Frecuencia | Baja | |
| Importancia | Media | |

Tabla B.9: Caso de uso 4: Administrar de camas

| | | |
|---------------------|--|--|
| CU-4.1: Añadir cama | | |
| Descripción | Añadir cama | |
| Precondiciones | Sesión de administración válida | |
| Requisitos | RF-2.1 | |
| Usuario | Administrador | |
| Secuencia normal | Paso | Acción |
| | 1 | El administrador elige añadir una nueva cama |
| | 2 | Se introduce el grupo multicast de la cama (IP y Puerto) |
| | 3 | Se introduce el nombre identificador |
| | 4 | Se almacenan los datos |
| Postcondiciones | Existe una nueva cama en el sistema | |
| Excepciones | Paso | Acción |
| | 2 | El grupo multicast pertenece a otra cama |
| | 3 | El nombre identificativo existe para otra cama |
| Frecuencia | Media | |
| Importancia | Crítica | |
| Comentarios | El grupo multicast se configura en la cama y el administrador solamente de conocerlo, no configurar la cama física | |

Tabla B.10: Caso de uso 4.1: Añadir cama

| | | |
|------------------------|---|---|
| CU-2.2: Modificar cama | | |
| Descripción | Modificar los datos de la cama | |
| Precondiciones | Sesión de administración válida, cama existente | |
| Requisitos | RF-2.2 | |
| Usuario | Administrador | |
| Secuencia normal | Paso | Acción |
| | 1 | Se elige que cama modificar |
| | 2 | Se actualizan los datos a conveniencia del administrador según CU-4.1 |
| | 4 | Se actualizan los datos |
| Postcondiciones | Los datos de la cama se modifican | |
| Excepciones | Paso | Acción |
| | 2 | Mismas excepciones que en 4.1 |
| Frecuencia | Baja | |
| Importancia | Alta | |

Tabla B.11: Caso de uso 4.2: Modificar cama

| | | |
|---------------------|--|--|
| CU-4.3: Borrar cama | | |
| Descripción | Elimina una cama de la base de datos | |
| Precondiciones | Sesión de administrador válida, cama existente | |
| Requisitos | RF-2.3 | |
| Usuario | Administrador | |
| Secuencia normal | Paso | Acción |
| | 1 | Elegir a que cama eliminar |
| | 2 | Eliminar cama y todos los datos vinculados |
| Postcondiciones | La cama ya no está en la base de datos | |
| Frecuencia | Baja | |
| Importancia | Media | |

Tabla B.12: Caso de uso 4.3: Borrar cama

| CU-4.4: Asignar cama a usuario | | |
|--------------------------------|---|--|
| Descripción | Permite a un usuario ver los datos de una cama o quitar ese permiso | |
| Precondiciones | Sesión de administración válida, cama y usuario existentes | |
| Requisitos | RF-2.4 | |
| Usuario | Administrador | |
| Secuencia normal | Paso | Acción |
| | 1 | Elegir cama |
| | 2 | Elegir usuario |
| | 3 | Si la relación existe se puede eliminar el permiso |
| | 3 | Si la relación no existe se puede crear el permiso |
| Postcondiciones | El usuario tiene acceso a la cama, o pierde el mismo | |
| Frecuencia | Media | |
| Importancia | Crítica | |

Tabla B.13: Caso de uso 4.4: Asignar cama a usuario

Apéndice C

Especificación de diseño

C.1. Introducción

C.2. Diseño de datos

De los requisitos y casos de usos se deduce el diseño de los datos. Para poder cumplir correctamente con las necesidades del cliente se introducen dos entidades, los **usuario** y las **camas** en la que cada uno almacena la información relevante propia. A su vez se relacionan en el sistema de permisos de que persona puede ver que cama. El diagrama Entidad-Relación se puede observar en la figura [C.1](#)

De este diagrama generamos el diagrama relacional (figura [C.2](#)) en el que se especifican las tablas que se usarán en el producto final.

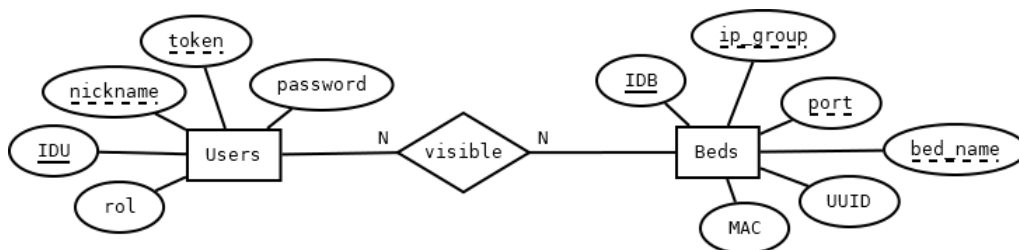


Figura C.1: Diagrama entidad-relación

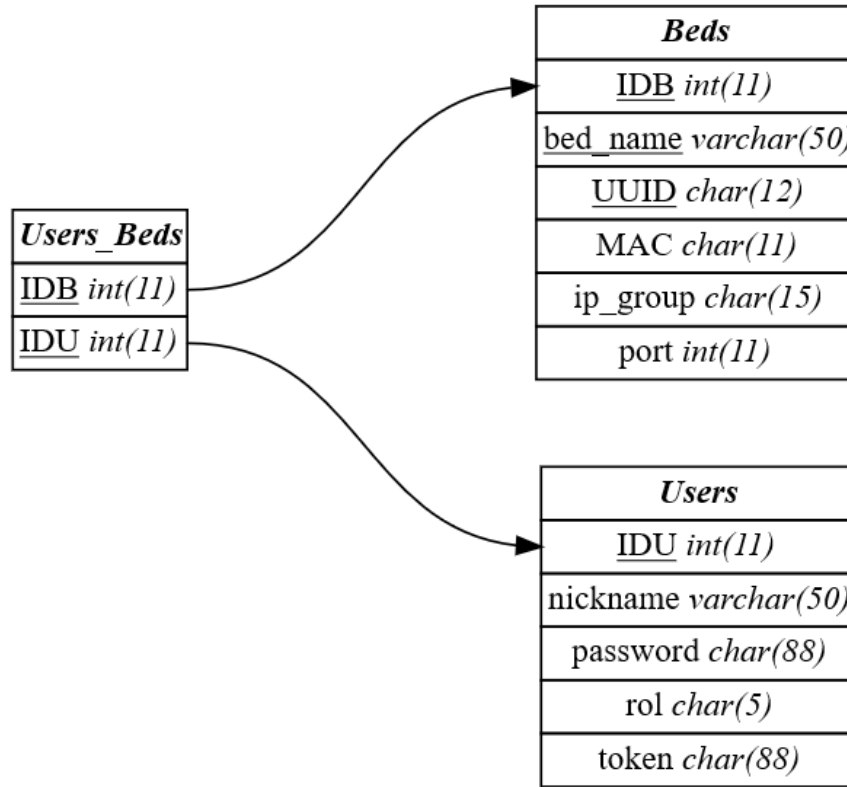


Figura C.2: Diagrama relacional

C.3. Diseño procedimental

C.4. Diseño arquitectónico

C.5. Diseño de interfaces

Las interfaces se han diseñado teniendo en cuenta la usabilidad de la misma así como un diseño simple que permitiese que fuese más intuitiva con una curva de aprendizaje leve. Se ha tenido en cuenta también que la interfaz sea adaptable a multitud de pantallas teniendo en cuenta que, al ser una aplicación web, el uso de la misma puede ser en multitud de dispositivos diferentes.

Los primeros prototipos fueron realizados sobre el caso de uso especificado en la tabla B.4, tanto para escritorio (imagen C.3) como para móvil (imagen C.4).

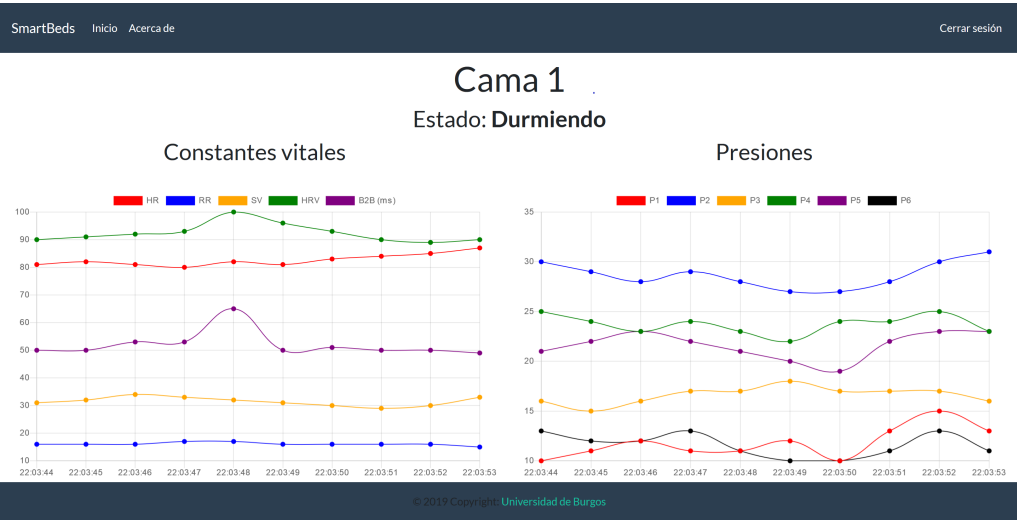


Figura C.3: Prototipo para escritorio

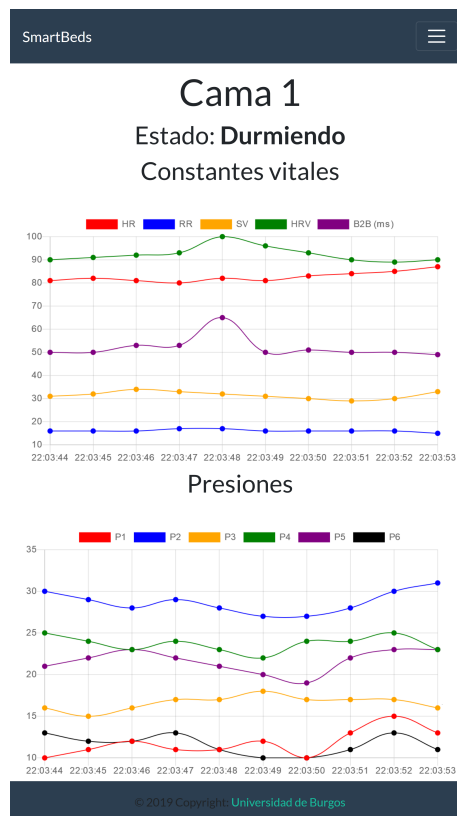


Figura C.4: Prototipo para móvil

Apéndice *D*

Documentación técnica de programación

D.1. Introducción

D.2. Estructura de directorios

D.3. Manual del programador

API

Para proveer los servicios de esta aplicación a nuevos entornos se incorpora una API para utilizar los diferentes servicios del sistema especificados en los Casos de Uso [B.4](#).

El funcionamiento general de la API serán peticiones **POST** ante rutas específicas con los datos requeridos para cada petición. Y según sea el caso el servidor contestará con un fichero **JSON** con la respuesta adecuada.

Todas las respuestas del servidor contendrán los siguientes campos

```
1 {  
2   "status": 200|400|401|403|404|500,  
3   "message": "OK"|"Error description"  
4 }
```

El valor de **status** tendrá un valor según los códigos HTTP definidos en el RFC 7231 y el mensaje será una explicación detallada del error producido.

APÉNDICE D. DOCUMENTACIÓN TÉCNICA DE PROGRAMACIÓN

Las distintas peticiones se especifican en la tabla [D.1](#)

| CU | URI | Petición | Respuesta |
|----------------|---------------|---|---|
| CU-1 | /api/auth | <pre>{ "user": text, "pass": text }</pre> | <pre>{ ..., "token": text }</pre> |
| CU-2.1 CU-4 | /api/beds | <pre>{ "token": text }</pre> | <pre>{ ..., "beds": [text, ...,text] }</pre> |
| CU-2.2 | /api/bed | <pre>{ "token": text, "bedname": text }</pre> | <pre>{ ..., "namespace": text }</pre> |
| CU-2.2 | / (WebSocket) | <pre>{ "token": text, "namespace": text }</pre> | <pre>{ "state": number, "instance": datetime "vital": [HR,RR, ...,B2B], "pressure": [P1, P2,...,P6] }</pre> |
| CU-3 | /api/users | <pre>{ "token": text }</pre> | <pre>{ ..., "users": [text, ...,text] }</pre> |
| CU-3.1 | /api/user/add | <pre>{ "token": text, "username": text, "password": text, "password-re": text }</pre> | <pre>{ ... }</pre> |

continúa en la página siguiente

continúa desde la página anterior

| CU | URI | Petición | Respuesta |
|--------|---------------|--|----------------------|
| CU-3.2 | /api/user/mod | <pre>{ "token": text, "username": text, "password": text, "password-re": text, ["pasword-old": text] }</pre> | <pre>{ ... }</pre> |
| CU-3.3 | /api/user/del | <pre>{ "token": text, "username": text }</pre> | <pre>{ ... }</pre> |
| CU-4.1 | /api/bed/add | <pre>{ "token": text, "bed_name": text, "ip_group": text, "port": text, "MAC": text, "UUID": text }</pre> | <pre>{ ... }</pre> |
| CU-4.2 | /api/bed/mod | <pre>{ "token": text, "bed_name": text, "ip_group": text, "port": text, "MAC": text, "UUID": text }</pre> | <pre>{ ... }</pre> |

continúa en la página siguiente

continúa desde la página anterior

| CU | URI | Petición | Respuesta |
|--------|---------------|--|---|
| CU-4.3 | /api/bed/del | <pre>{ "token": text, "bed_name": text }</pre> | <pre>{ ... }</pre> |
| CU-4.4 | /api/bed/perm | <pre>{ "token": text, "mode": "info" "change", ["bed_name": text, "username": text] }</pre> | <pre>{ ..., "permission": [{ "username": text, "bed_name": text }, ...] }</pre> |

Tabla D.1: Especificaciones del API

- D.4. Compilación, instalación y ejecución del proyecto
- D.5. Pruebas del sistema

Apéndice E

Documentación de usuario

- E.1. Introducción
- E.2. Requisitos de usuarios
- E.3. Instalación
- E.4. Manual del usuario

Bibliografía
