Documentación Prácticas Ingeniría del Software

uNIVERSIDAD ANTONIO DE NEBRIJA

Searchill

MIGUEL CARBALLO, MARCOS GAGO, David Martínez, , ALberto meneses, ESTEBAN VIDIELLA

2020

**Practica Ingeniería de Software**

**Ejercicio 1**

Nuestra aplicación está desarrollada en html, utilizará una la app de Medline (una base de datos de enfermedades).

Nuestra aplicación consiste en un motor de búsqueda de enfermedades por síntomas. El usuario introduce los síntomas que tiene, y Searchill le devuelve todas las enfermedades que tengan esos síntomas; una vez se presente la lista de enfermedades cuyos síntomas tiene en común con los introducidos por el usuario, éste podrá acceder a aquella que mas le convenza para ver mas información sobre la aflicción en cuestión y en caso de que la enfermedad escogida le resuelva las dudas sobre lo que padece, el usuario puede votar por esta enfermedad para que en caso de que se busque esta enfermedad aparezca con mayor prioridad.

El usuario modelo puede ser cualquiera debido a la facilidad de la aplicación, pero sobre todo aquellas personas que no quieran esperar cola yendo al centro sanitario del vecindario. Advertencia: en ningún momento el grupo de Searchill se pone en contra del diagnóstico de un médico certificado, los resultados de la búsqueda son completamente orientativos, el grupo de Searchill no se hace responsable de las complicaciones que se pueda tener por no ir al médico en vez de confiar en nuestra aplicación.

**Ejercicio 2**

Miembros del grupo

|  |  |
| --- | --- |
| **Nombre** | **Apellidos** |
| Miguel | Carballo |
| Marcos | Gago |
| David | Martínez |
| Alberto | Meneses |
| Esteban | Vidiella |

Versión

|  |  |  |
| --- | --- | --- |
| **Versión** | **Fecha** | **Comentarios** |
| 0.0 | 21/02/2020 | Dia de planificación, no hay demo a la vista |
| 0.1 | 10/05/2020 | Dia de revisión con la demo realizada |

Especificación de Requisitos de Software

Formato IEEE 830

[**Introducción**](#_npbjqvklurii)3

[1.1 Propósito](#_vgwhlvydyflq) 3

[1.2 Alcance](#_3iupoz3hz3yz) 3

[1.3 Definiciones, siglas y abreviaturas](#_pn85vcrayoqt) 3

[1.4 Referencias](#_dyc67je6wpfy) 3

[**Descripción global**](#_ni14snwmfxl)4

[2.1 Perspectiva del producto](#_5i1gydnccy2y) 4

[2.2 Funciones del producto](#_b59y1d9x0isf) 4

[2.3 Características del usuario](#_xmsbjdpnxcyh) 4

[2.4 Restricciones](#_t86wm43dboq8) 4

[2.5 Suposiciones y Dependencias](#_sks1bcxzgtfo) 5

[2.6 Requisitos Futuros](#_iwj7gw6nqcco) 5

[**Requisitos Específicos**](#_9sxyzcqgclpt)5

[3.1 Interfaces externas](#_1zstwsnnohx3) 5

[3.2 Funciones](#_o477g2sdhruf) 5

[3.3 Requisitos de rendimiento](#_77ox7212z8qg) 6

[3.4 Restricciones de diseño](#_wgqfqie9iqr) 6

[3.5 Atributos del sistema](#_ubfl91nn8z5t) 6

[3.6 Otros requisitos](#_mp2if0w0k4sl) 6

[**Apéndices**](#_5sd0lcjml53)6

## Introducción

### 1.1 Propósito

Nuestra aplicación provee a usuarios información de la enfermedad que ellos mismos han buscado; aunque también ofrece la posibilidad de encontrar la enfermedad más posible teniendo en cuenta los síntomas que nos introduce el usuario. Los usuarios son principalmente personas hipocondríacas, o que prefieran auto diagnosticarse.

### 1.2 Alcance

Habrá dos prototipos respecto a Searchill: una aplicación java con una base de datos externa y una página HTML con de nuevo, una base de datos externa. El usuario podrá elegir el modo de búsqueda (mediante síntomas encontrar enfermedad o simplemente buscar la enfermedad). Al encontrar enfermedad, si se está satisfecho con el diagnóstico el usuario podrá darle una review positiva, de manera que a la hora de calcular la heurística se tendrá en cuenta la review positiva; a parte al encontrar la enfermedad, el usuario puede pedir más información sobre ella, de manera que podrá abrirse una página web como las de la OMS para obtenerla.

### 1.3 Definiciones, siglas y abreviaturas

***Enfermedad***: referencia a la clase de enfermedad que vamos a crear compuesta de síntomas

***Síntoma***: sensaciones que percibe el usuario, procedentes de una o varias enfermedades

***Popularidad***: número otorgado a la enfermedad en función de lo popular que sean los síntomas de la enfermedad

***Prioridad***: forma en la que aparecerán las enfermedades en función de parámetros que encajen

***Estaciones***: referente a verano, otoño, invierno y primavera

***Base***: forma de acortar base de datos

HTML:

### 1.4 Referencias

*En esta subsección se mostrará una lista completa de todos los documentos referenciados en la ERS.*

|  |  |
| --- | --- |
| **Título del documento** | **Referencia** |
| Especificación software del buscador de enfermedades | ERS1-REF1 |

## Descripción global

### 2.1 Perspectiva del producto

La aplicación cuya interfaz de cara al usuario es HTML actúa juntamente con el código de Java y las bases de datos que este utiliza

### 2.2 Funciones del producto

* El usuario podrá buscar enfermedades por su nombre.
* El usuario podrá introducir síntomas que padece y el sistema le dará ciertas enfermedades que encajan con sus datos y síntomas.
* El sistema podrá dar una breve descripción de la enfermedad on-code.
* Si el usuario quiere el sistema puede abrir una url contiendo más información sobre la enfermedad.
* El usuario puede introducir datos de sí mismo para hacer la búsqueda más precisa.

### 2.3 Características del usuario

|  |  |
| --- | --- |
| **Tipo de usuario** | Usuario estándar |
| **Criterio** | Todos aquellos que posean la aplicación deben tener acceso a la base de datos. |

### 2.4 Restricciones

*Esta subsección describe aquellas limitaciones que se imponen sobre los desarrolladores del producto:*

* *Limitaciones del hardware*
* *Interfaces con otras aplicaciones*
* *Fechas de auditoría (en vuestro caso del profesor)*
* *Lenguaje(s) de programación*
* Java mediante Eclipse IDE y HTML.
* *Protocolos de comunicación*
* El equipo de desarrollo está siempre intercomunicado entre sí, pero se podrá contactar con él vía correo electrónico.
* *Consideraciones acerca de la seguridad*
* Cumplir con la Ley Orgánica de protección de datos.

### 2.5 Suposiciones y Dependencias

Asumimos que el back-end (Java y la base de datos) se comunicará mediante apis con el front-end (HTML); restringimos el uso al sistema operativo Windows por el momento.

### 2.6 Requisitos Futuros

Se prevé una forma de almacenar los datos de los usuarios en una estructura valida e incluir una gran cantidad de enfermedades y síntomas más específicos para hacer la búsqueda más sencilla.

## Requisitos Específicos

### 3.1 Interfaces externas

Comunicación con el usuario:

* Pantalla para mostrar la página web.

### Funciones

**3.2.1** Mostar información de la enfermedad buscada:

* Actor: Página web. Usuario.
* Descripción: Los usuarios podrán ver en la página web información de la enfermedad buscada.
* Precondiciones: El usuario debe tener conexión a internet.
* Dependencias: Que la API de MedlinePlus esté funcionando correctamente.
* Escenario:
* Paso 1 – Página web: Muestra la página principal.
* Paso 2 – Usuario: Introduce en el buscador el nombre de la enfermedad y pulsa botón Go.
* Paso 3 – Página web: Muestra un enlace a la página de MedlinePlus y una pequeña descripción de la enfermedad.
* Excepciones:
* Excepción 1
* Paso 2 – Usuario: El usuario no escribe el nombre de la enfermedad correctamente.
* Paso 3 – Página web: No mostrará la información hasta que se escriba bien.

**3.2.2** Filtrar la información de la búsqueda:

* Actor: Página web. Usuario.
* Descripción: Los usuarios podrán ver la información filtrada según lo que escriban en el recuadro de búsqueda.
* Precondiciones: El usuario debe a ver realizado una búsqueda al menos una búsqueda.
* Dependencias: Mostar información de la enfermedad buscada
* Escenario:
* Paso 1 – Página web: Muestra información de las búsquedas.
* Paso 2 – Usuario: Introduce en el recuadro de búsqueda cualquier término.
* Paso 3 – Página web: Si el texto que va introduciendo el usuario en el recuadro forma parte del nombre de la enfermedad, solo se mostrará la información de esta.

### 3.3 Requisitos de rendimiento

**3.3.1** NLM requiere que no se envíen más de 100 peticiones por minuto por dirección IP.

### 3.4 Restricciones de diseño

**3.4.1** Las solicitudes de información a MedlinePlus están sujetas al sistema ICD-9-CM de clasificación de enfermedades.

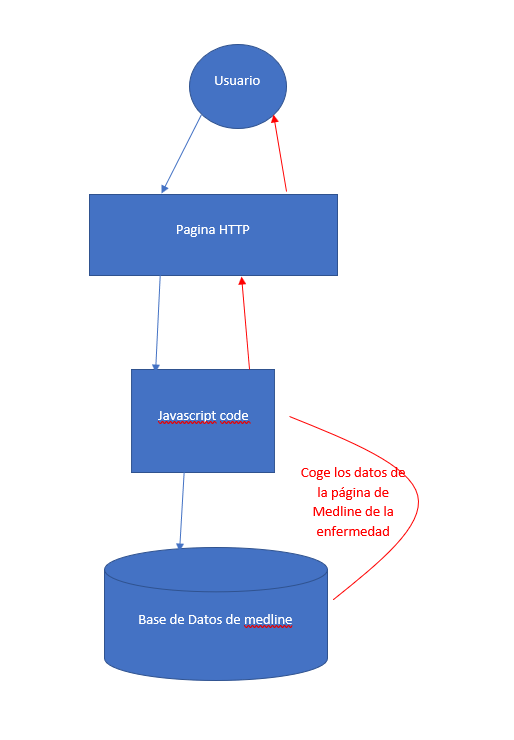
### 3.5 Atributos del sistema

**3.5.1** Disponibilidad: la disponibilidad debe de ser continua mientras que el acceso a internet y las bases de datos sea continuo.

**Ejercicio 3**

El inventario de Riegos está detallado en el ejercicio 6

**Ejercicio 4**



**Ejercicio 5**

El inventario de pruebas se detalla en el ejercicio 6

**Ejercicio 6**

**Plantilla del IEEE** cumplimentada en ejercicio 2

**Riesgos**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Descripción Riesgo** | **Prioridad** | **Prob** | **Plan de contingencia** | **Responsable** | **Estado** | **Categoría** |
| Externalizar la base de datos. Dependemos de una base externa la cual puede no funcionar de acuerdo con nuestra heurística y complicar las consultas a la misma | Alta | Media-alta | Minimizar el impacto de la base de datos con un volumen alto de datos | Todos | En curso | Producto |
| No conseguir integrar el código Java, las bases de datos y el código html de la Web | Alta | Alta | Reducir los puntos de integración | Todos | Cerrado | integración |
| El alcance inicial del proyecto demasiado ambicioso | Alta | Media | Realizar un Producto Mínimo Viable para la demo | Todos | Cerrado | Alcance |
| El COVID 19 ha impedido realizar el trabajo de forma presencial entre los componentes del equipo lo que ha dificultado su realización | Alta | Alta | Reuniones en remoto | Todos | En curso | Modo de trabajar |
| Carecemos del conocimiento de la API para poder incorporarla a nuestro proyecto | Alta | Alta | Utilizar la base de datos sin la API | Todo el equipo | Cerrado | Negocio |
| Baja precisión de la base de datos | Media | Media | Coger los datos más comunes | Todo el equipo | Cerrado | Proyecto |

**Diagrama**.

El diagrama se ha detallado en el ejercicio 4.

**Inventario de Pruebas**

Debido a que hemos realizado el proyecto con archivos json no se pueden hacer test unitarios como en java o test como c++, por lo tanto, hemos utilizado un add-on de vscode llamado go live para probar que las páginas web se abren de manera correcta. El resultado de las pruebas ha sido favorable en todos los casos

Hemos realizado una pequeña prueba de usuario, realizando la demo cada uno de nosotros con miembros de nuestras familias y recibiendo los comentarios sobre los resultados que hemos utilizado para mejorar la experiencia de usuario

**Ejercicio 8**

Enlace al repositorio: <https://github.com/dmartinezc4/Searchill>

|  |  |
| --- | --- |
| Participante | Trabajo realizado |
| Miguel Carballo | Código en html, pruebas en html/javascript, css de la página web y conexión de la API con html. Documentación diseño técnico, y funcional. |
| Marcos Gago | Pruebas en html de la API, código en java y heurística. Inclusión términos médicos para búsqueda. |
| David Martínez | Pruebas en html de la API, documentación y heurística. Inclusión términos médicos para búsqueda. Documentación diseño técnico, y funcional. |
| Alberto Meneses | Documentación de la API, pruebas de la API y base de datos y consultas sobre la misma. Inclusión términos médicos para búsqueda |
| Esteban Vidiella | Documentación de la API, pruebas de la API y base de datos y consultas sobre la misma. Inclusión términos médicos para búsqueda. Documentación diseño técnico, y funcional. |