UNIVERSIDAD
REY JUAN CARLOS

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE LA TELECOMUNICACIÓN

Curso Académico 2019/2020

Trabajo Fin de Grado

TÍTULO DEL TRABAJO EN MAYÚSCULAS

Autor : Cristian Fabián Martínez Rosero

Tutor : Dr. Óscar Barquero Pérez

Trabajo Fin de Grado

Título del Trabajo con Letras Capitales para Sustantivos y Adjetivos

Autor : Cristian Fabián Martínez Rosero

Tutor : Dr. Óscar Barquero Pérez

La defensa del presente Proyecto Fin de Carrera se realizó el día de de 2020, siendo calificada por el siguiente tribunal:

Presidente:

Secretario:

Vocal:

y habiendo obtenido la siguiente calificación:

Calificación:

Fuenlabrada, a de de 2020

Dedicado a

mi familia

II

Agradecimientos

Aquí vienen los agradecimientos... Aunque está bien acordarse de la pareja, no hay que olvidarse de dar las gracias a tu madre, que aunque a veces no lo parezca disfrutará tanto de tus logros como tú... Además, la pareja quizás no sea para siempre, pero tu madre sí.

Resumen

Aquí viene un resumen del proyecto. Ha de constar de tres o cuatro párrafos, donde se presente de manera clara y concisa de qué va el proyecto. Han de quedar respondidas las siguientes preguntas:

- ¿De qué va este proyecto? ¿Cuál es su objetivo principal?
- ¿Cómo se ha realizado? ¿Qué tecnologías están involucradas?
- ¿En qué contexto se ha realizado el proyecto? ¿Es un proyecto dentro de un marco general?

Lo mejor es escribir el resumen al final.

Summary

Here comes a translation of the “Resumen” into English. Please, double check it for correct grammar and spelling. As it is the translation of the “Resumen”, which is supposed to be written at the end, this as well should be filled out just before submitting.

Índice general

Índice de figuras

Capítulo 1

Introducción

En este capítulo se introduce el proyeto. Debería tener información general sobre el mismo, dando la información sobre el contexto en el que se ha desarrollado.

No te olvides de echarle un ojo a la página con los cinco errores de escritura más frecuentes[1].

Aconsejo a todo el mundo que mire y se inspire en memorias pasadas. Las memorias de los proyetos que he llevado yo están (casi) todas almacenadas en mi web del GSyC[2].

1.1. Sección

Esto es una sección, que es una estructura menor que un capítulo.

Por cierto, a veces me comentáis que no os compila por las tildes. Eso es un problema de codificación. Cambiad de "UTF-8" a "ISO-Latin-1" (o vicecersa) y funcionará.

1.1.1. Estilo

Recomiendo leer los consejos prácticos sobre escribir documentos científicos en LaTeX de Diomidis Spinellis[3].

Lee sobre el uso de las comas[4]. Las comas en español no se ponen al tuntún. Y nunca, nunca entre el objeto y el predicado (p.ej. en "Yo, hago el TFG" sobre la coma).

[1] http://www.tallerdeescritores.com/errores-de-escritura-frecuentes

[2] https://gsyc.urjc.es/~grex/pfcs/

[3] https://github.com/dspinellis/latex-advice

[4] http://narrativabreve.com/2015/02/opiniones-de-un-corrector-de-estilo-11-recetas-par
html

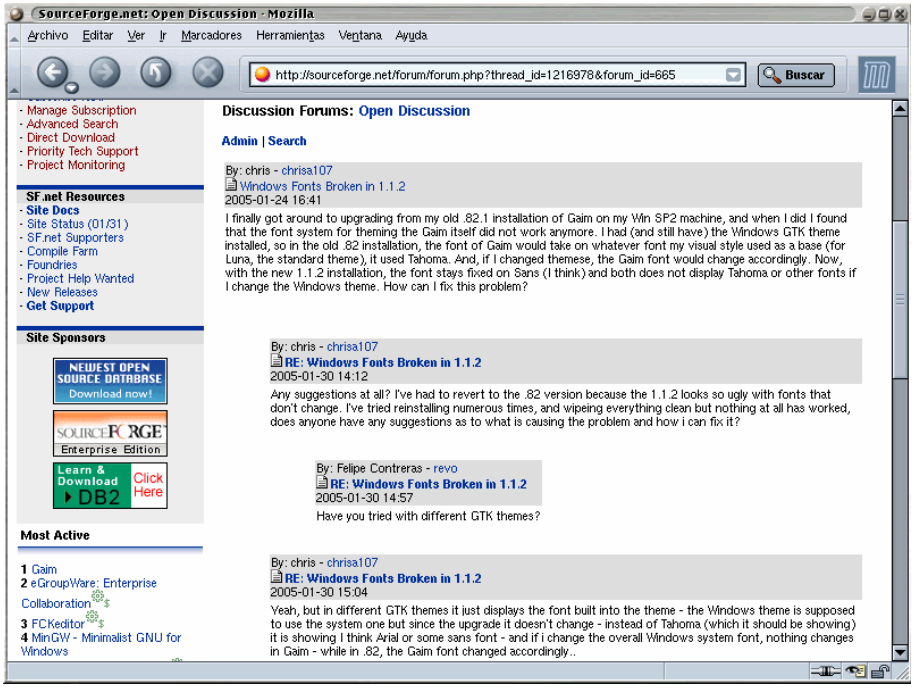

Figura 1.1: Página con enlaces a hilos

A continuación, viene una figura, la Figura 1.1. Observarás que el texto dentro de la referencia es el identificador de la figura (que se corresponden con el “label” dentro de la misma). También habrás tomado nota de cómo se ponen las “comillas dobles” para que se muestren correctamente. Nota que hay unas comillas de inicio (“) y otras de cierre (”), y que son diferentes. Volviendo a las referencias, nota que al compilar, la primera vez se crea un diccionario con las referencias, y en la segunda compilación se “rellenan” estas referencias. Por eso hay que compilar dos veces tu memoria. Si no, no se crearán las referencias.

A continuación un bloque “verbatim”, que se utiliza para mostrar texto tal cual. Se puede utilizar para ofrecer el contenido de correos electrónicos, código, entre otras cosas.

```
From gaurav at gold-solutions.co.uk  Fri Jan 14 14:51:11 2005
From: gaurav at gold-solutions.co.uk (gaurav_gold)
Date: Fri Jan 14 19:25:51 2005
Subject: [Mailman-Users] mailman issues
Message-ID: <003c01c4fa40$1d99b4c0$94592252@gaurav7klgnyif>

Dear Sir/Madam,
How can people reply to the mailing list?  How do i turn off
this feature? How can i also enable a feature where if someone
```

```
replies the newsletter the email gets deleted?
Thanks

From msapiro at value.net  Fri Jan 14 19:48:51 2005
From: msapiro at value.net (Mark Sapiro)
Date: Fri Jan 14 19:49:04 2005
Subject: [Mailman-Users] mailman issues
In-Reply-To: <003c01c4fa40$1d99b4c0$94592252@gaurav7klgnyif>
Message-ID: <PC17302005011410485105 7801b04d55@msapiro>

gaurav_gold wrote:
>How can people reply to the mailing list?  How do i turn off
this feature? How can i also enable a feature where if someone
replies the newsletter the email gets deleted?

See the FAQ
>Mailman FAQ: http://www.python.org/cgi-bin/faqw-mm.py
article 3.11
```

1.2. Estructura de la memoria

En esta sección se debería introducir la esctura de la memoria.

Así:

- En el primer capítulo se hace una intro al proyecto.
- En el capítulo 2 (ojo, otra referencia automática) se muestran los objetivos del proyecto.
- A continuación se presenta el estado del arte en el capítulo 3.
- ...

Capítulo 2

Objetivos

2.1. Objetivo general

Aquí vendría el objetivo general en una frase: Mi trabajo fin de grado consiste en crear de una herramienta de análisis de los comentarios jocosos en repositorios de software libre alojados en la plataforma GitHub.

Recuerda que los objetivos siempre vienen en infinitivo.

2.2. Objetivos específicos

Los objetivos específicos se pueden entender como las tareas en las que se ha desglosado el objetivo general. Y, sí, también vienen en infinitivo.

2.3. Planificación temporal

A mí me gusta que aquí pongáis una descripción de lo que os ha llevado realizar el trabajo. Hay gente que añade un diagrama de GANTT. Lo importante es que quede claro cuánto tiempo llevas (tiempo natural, p.ej., 6 meses) y a qué nivel de esfuerzo (p.ej., principalmente los fines de semana).

Capítulo 3

Estado del arte

Descripción de las tecnologías que utilizas en tu trabajo. Con dos o tres párrafos por cada tecnología, vale. Se supone que aquí viene todo lo que no has hecho tú.

Puedes citar libros, como el de Bonabeau et al., sobre procesos estigmérgicos [**?**]. Me encantan los procesos estigmérgicos. Deberías leer más sobre ellos. Pero quizás no ahora, que tenemos que terminar la memoria para sacarnos por fin el título. Nota que el ~ añade un espacio en blanco, pero no deja que exista un salto de línea. Imprescindible ponerlo para las citas.

Citar es importantísimo en textos científico-técnicos. Porque no partimos de cero. Es más, partir de cero es de tontos; lo suyo es aprovecharse de lo ya existente para construir encima y hacer cosas más sofisticadas. ¿Dónde puedo encontrar textos científicos que referenciar? Un buen sitio es Google Scholar[1]. Por ejemplo, si buscas por "stigmergy libre software" para encontrar trabajo sobre software libre y el concepto de estigmergia (¿te he comentado que me gusta el concepto de estigmergia ya?), encontrarás un artículo que escribí hace tiempo cuyo título es "Self-organized development in libre software: a model based on the stigmergy concept". Si pulsas sobre las comillas dobles (entre la estrella y el "citado por ...", justo debajo del extracto del resumen del artículo, te saldrá una ventana emergente con cómo citar. Abajo a la derecha, aparece un enlace BibTeX. Púlsalo y encontrarás la referencia en formato BibTeX, tal que así:

```
@inproceedings{robles2005self,
  title={Self-organized development in libre software:
         a model based on the stigmergy concept},
  author={Robles, Gregorio and Merelo, Juan Juli\'an
```

[1] `http://scholar.google.com`

Uno	2	3
Cuatro	5	6
Siete	8	9

Cuadro 3.1: Ejemplo de tabla. Aquí viene una pequeña descripción (el *caption*) del contenido de la tabla. Si la tabla no es autoexplicativa, siempre viene bien aclararla aquí.

```
          and Gonz\'alez-Barahona, Jes\'us M.},
  booktitle={ProSim'05},
  year={2005}
}
```

Copia el texto en BibTeX y pégalo en el fichero `memoria.bib`, que es donde están las referencias bibliográficas. Para incluir la referencia en el texto de la memoria, deberás citarlo, como hemos hecho antes con [**?**], lo que pasa es que en vez de el identificador de la cita anterior (bonabeau:_swarm), tendrás que poner el nuevo (robles2005self). Compila el fichero `memoria.tex` (`pdflatex memoria.tex`), añade la bibliografía (`bibtex memoria.aux`) y vuelve a compilar `memoria.tex` (`pdflatex memoria.tex`)... y *voilà* ¡tenemos una nueva cita!

También existe la posibilidad de poner notas al pie de página, por ejemplo, una para indicarte que visite la página de LibreSoft[2].

3.1. Sección 1

Hemos hablado de cómo incluir figuras. Pero no hemos dicho nada de tablas. A mí me gustan las tablas. Mucho. Aquí un ejemplo de tabla, la Tabla 3.1 (siento ser pesado, pero nota cómo he puesto la referencia).

[2] `http://www.libresoft.es`

Capítulo 4

Diseño e implementación

Aquí viene todo lo que has hecho tú (tecnológicamente). Puedes entrar hasta el detalle. Es la parte más importante de la memoria, porque describe lo que has hecho tú. Eso sí, normalmente aconsejo no poner código, sino diagramas.

4.1. Arquitectura general

El proyecto consta de un servidor que envía tanto componentes web como gráficas de Plotly al lado del cliente y que, mediante el uso de callbacks, se comunican entre sí.

Los componentes web que se envían al cliente emplean las herramientas del framework Dash (Html, React, Bootstrap, Dash Core Components, Dash Datatable). Respecto a las gráficas, se utiliza la biblioteca Plotly. En cuanto al lado del servidor, se ha utilizado Flask como servidor de contenidos. Además, el servidor implementa el módulo ecgreader, para la lectura de los ficheros con formato ISHNE y Physionet, y el módulo WDelimitador, el cual es utilizado para detectar los distintos tipos de ondas (QRS, P, T) que hay en una señal de electrocardiograma.

Para la capa de persistencia, se ha utilizado una base de datos no relacional, MongoDB. Tanto el lado del cliente, como para el lado del servidor y la comunicación entre servidor y la base de datos, se ha utilizado sintáxis de Python para codificar sus funcionalidades.

En la figura 4.1 se muestra un esquema a grandes rasgos de los componentes que intervienen en la arquitectura de la aplicación.

Cuando un usuario realiza la petición de la página principal de la aplicación mediante el

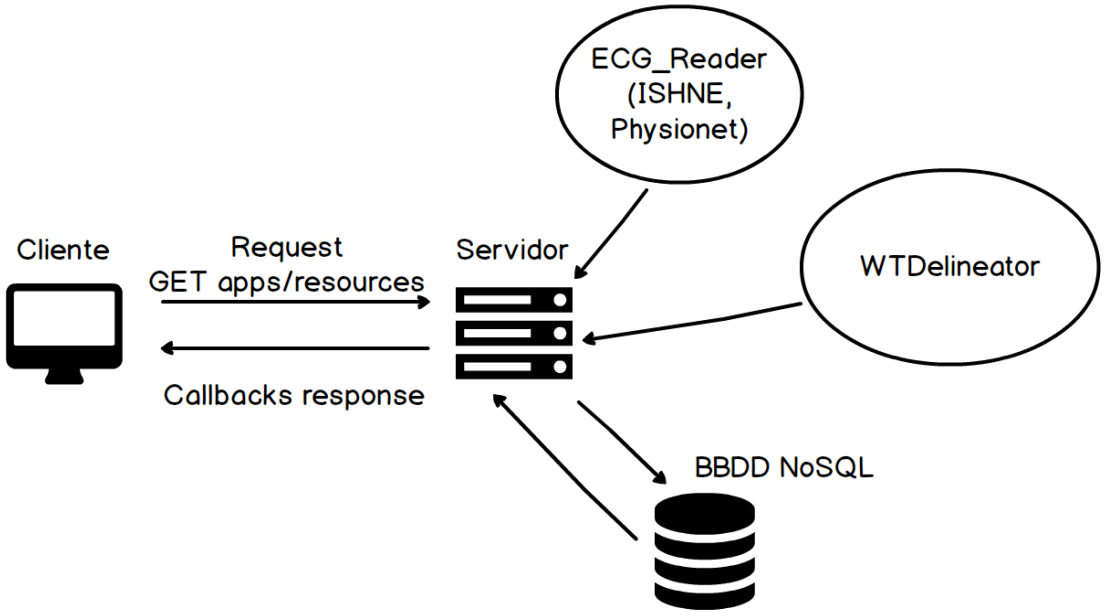

Figura 4.1: Arquitectura de la aplicación.

navegador, éste realizará una petición HTTP al servidor pidiendo dicho recurso. El servidor comprobará si el recurso existe, utilizará las herramientas de Dash para renderizar el código Python en componentes HTML + React, y enviará el resultado en una respuesta HTTP al navegador, el cual interpretará la página devuelta y mostrará los distintos componentes de esta en pantalla.

Cuando el usuario interactúe con los componentes de una página, el navegador hará llamadas HTTP asíncronas al servidor, el cual, mediante el uso de funciones callback gestionará dichas llamadas, teniendo en cuenta los valores de entrada que afecten a cada función. Después de realizar el proceso que se haya implementado en la función, se enviará una respuesta HTTP, donde se envolverán en ella los distintos valores de salida que podrán afectar a uno o a varios componentes de la página.

4.2. Diseño e implementación del servidor

Si hablamos de enrutamiento de recursos, el servidor implementa una parte REST, que es la que se encarga de enviar al cliente las distintas páginas o recursos asociados a la aplicación en general o a módulos de la aplicación.

También se encarga de renderizar los componentes de cada una de las páginas que se quiera servir en cada uno de los módulos de la aplicación mediante Dash; además de permitir la comunicación de estos componentes desde el cliente con él mediante funciones callbacks, de esta forma, tendremos una funcionalidad por cada uno de los componentes. Más adelante se explicará la manera en cómo se declaran las entradas de estas funciones, así como las salidas, y en cómo esto está ligado a distintos aspectos de los componentes que intervengan en la función.

Dentro de los componentes que renderiza el servidor, tenemos el más importante para la aplicación, que es la parte de los gráficos. Para mostrar el gráfico de electrocardiograma, el servidor recibirá como parámetros una serie de ficheros, los cuales se subirán a un directorio del servidor, se realizará una serie de validaciones y finalmente, se leerán utilizando el módulo *ecgreader*. Éste devolverá un objeto de una clase abstracta, así se logra que el objeto devuelto sea independiente del formato del fichero, obteniendo siempre una misma estructura. Debido a esto, para el componente encargado de mostrar la gráfica será transparente si el electrocardiograma a mostrarse es de un formato u otro. Los datos más importantes de este objeto, tales como frecuencia de muestreo, array de la señal y número de derivación, serán usados para hacer una llamada a la función *wtDelinador* del módulo WTDelineator, y de esta forma, obtener los datos de las ondas más representativas del electrocardiograma, como la onda P, onda T y complejo QRS. Para todas estas ondas, se obtendrá delimitadores, que nos dirá el inicio y el fin de cada una de las ondas.

Por último, para ciertas funcionalidades de la aplicación, como la edición de anotaciones de la señal ecg, acceso multiusuario y posiblilidad de multisesión, el servidor se comunicará con una base de datos no relacional, MongoDB, utilizando las bibliotecas *PyMongo* y *Flask-PyMongo*. De esta manera se añade persistencia para ciertos datos, que son de utilidad para implementar las funcionalidades antes descritas.

4.2.1. Modelos de la base de datos

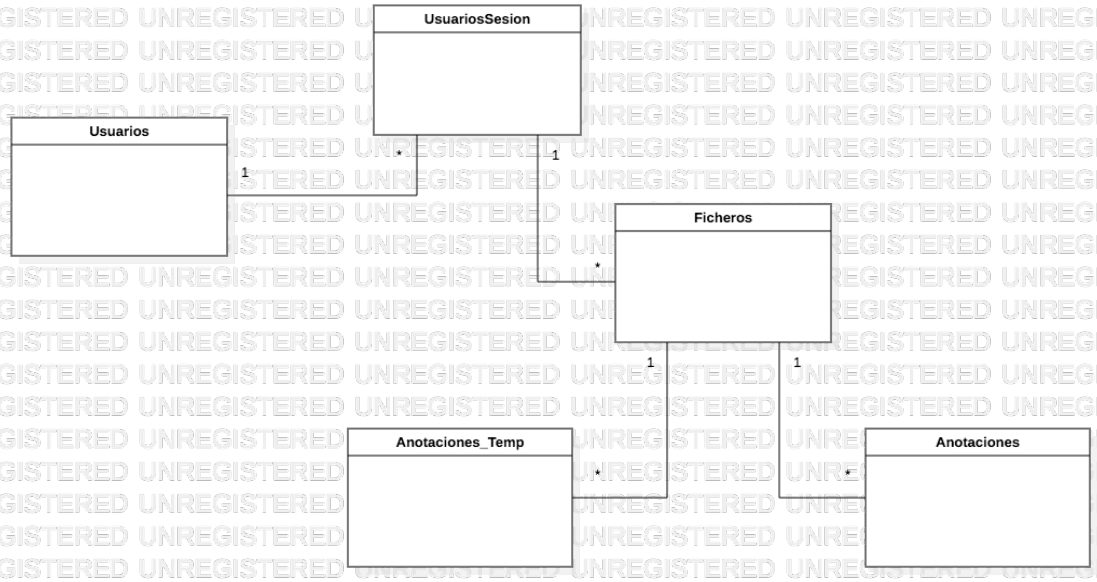

Figura 4.2: Diagrama entidad-relación de la base de datos

Capítulo 5

Resultados

En este capítulo se incluyen los resultados de tu trabajo fin de grado.

Si es una herramienta de análisis lo que has realizado, aquí puedes poner ejemplos de haberla utilizado para que se vea su utilidad.

Capítulo 6

Conclusiones

6.1. Consecución de objetivos

Esta sección es la sección espejo de las dos primeras del capítulo de objetivos, donde se planteaba el objetivo general y se elaboraban los específicos.

Es aquí donde hay que debatir qué se ha conseguido y qué no. Cuando algo no se ha conseguido, se ha de justificar, en términos de qué problemas se han encontrado y qué medidas se han tomado para mitigar esos problemas.

Y si has llegado hasta aquí, siempre es bueno pasarle el corrector ortográfico, que las erratas quedan fatal en la memoria final. Para eso, en Linux tenemos aspell, que se ejecuta de la siguiente manera desde la línea de *shell*:

```
aspell --lang=es_ES -c memoria.tex
```

6.2. Aplicación de lo aprendido

Aquí viene lo que has aprendido durante el Grado/Máster y que has aplicado en el TFG/TFM. Una buena idea es poner las asignaturas más relacionadas y comentar en un párrafo los conocimientos y habilidades puestos en práctica.

1. a

2. b

6.3. Lecciones aprendidas

Aquí viene lo que has aprendido en el Trabajo Fin de Grado/Máster.

1. Aquí viene uno.
2. Aquí viene oto.

6.4. Trabajos futuros

Ningún proyecto ni software se termina, así que aquí vienen ideas y funcionalidades que estaría bien tener implementadas en el futuro.

Es un apartado que sirve para dar ideas de cara a futuros TFGs/TFMs.

Apéndice A

Manual de usuario

Esto es un apéndice. Si has creado una aplicación, siempre viene bien tener un manual de usuario. Pues ponlo aquí.