UNIVERSIDAD
REY JUAN CARLOS

INGENIERÍA EN SISTEMAS AUDIOVISUALES Y MULTIMEDIA

Curso Académico 2016/2017

Trabajo Fin de Grado

CREACIÓN DE LA PLATAFORMA DR.PENCILCODE

Autor : Sara Blázquez Calderay

Tutor : Dr. Gregorio Robles

Proyecto Fin de Carrera

CREACIÓN DE LA PLATAFORMA DR.PENCILCODE

Autor : Sara Blázquez Calderay

Tutor : Dr. Gregorio Robles Martínez

La defensa del presente Proyecto Fin de Carrera se realizó el día de de 2017, siendo calificada por el siguiente tribunal:

Presidente:

Secretario:

Vocal:

y habiendo obtenido la siguiente calificación:

Calificación:

Fuenlabrada, a de de 2017

Dedicado a mis hermanos,

Alejandro y Nuria, os quiero.

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].

1.1. Marco General

1.2. Pencilcode

Pencilcode[2] es una aplicación web creada por David Bau, miembro del equipo educativo de Google. Enfocada en su totalidad a la enseñanza, Pencilcode trata de introducir a sus usuario al mundo de la programación de una manera más fácil y sencilla.

Su objetivo principal es facilitar el aprendizaje orientado a lenguajes de programación de .

- Soporta CoffeScript, html, css y Javascript.

- A diferencia de Scratch, Pencilcode ofrece la posibilidad de porgramar en bloques o codigo. Se explica las categorias y los bloques.

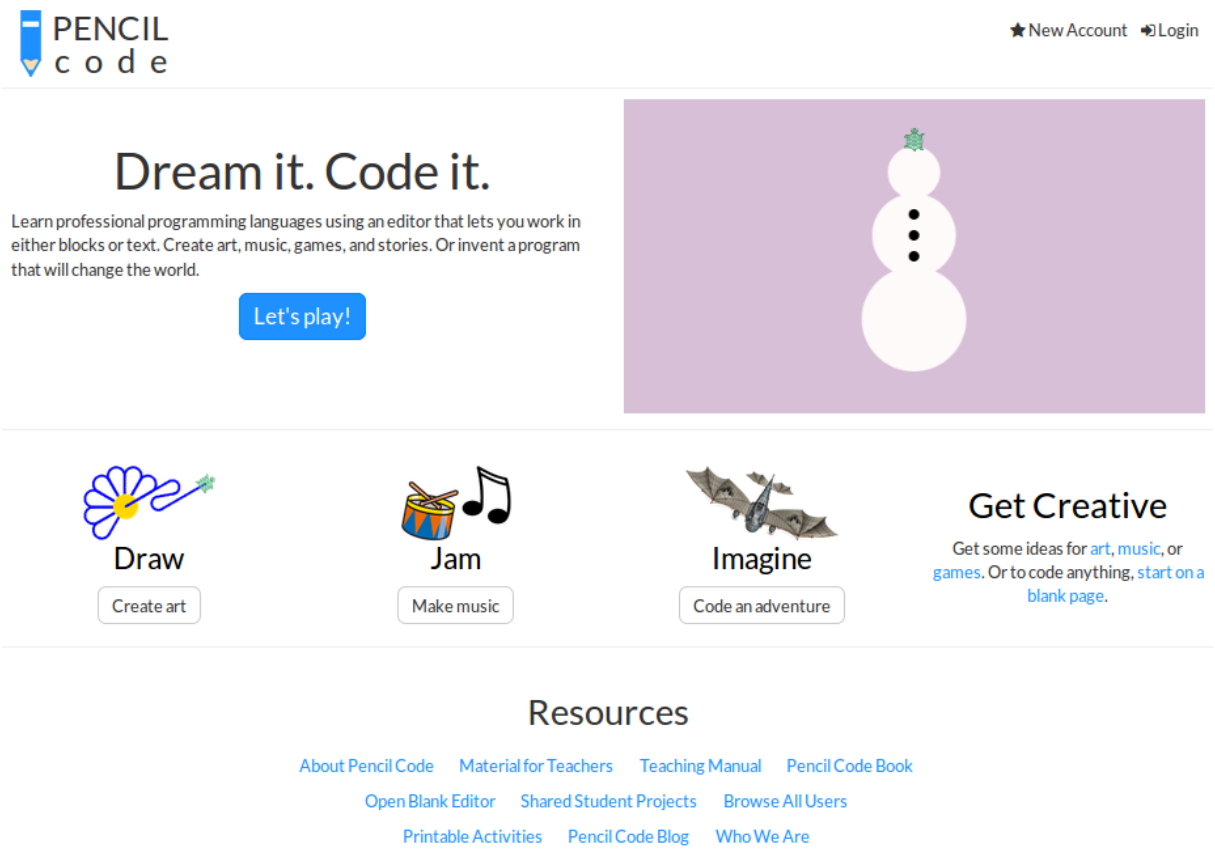
Pencil Code was created by Googler David Bau, and provides a step up from the block-based coding of Scratch. With Pencil Code, students can code art, music, and more in block-based form to begin, and then transition to directly to Javascript, HTML, and CSS when they?re ready, merely with the click of a ?Gear? button.

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

[2] `https://pencilcode.net/`

Learn professional programming languages using an editor that lets you work in either blocks or text.

Imagenes de la pagina principal del logo, decir que está basado en CoffeeScript Poner los ejemplos de bloques explicar un poco las secciones. y explicar en sí lo que hace pencilcode.



- Move:
- **Art:**
- **Text:**
- **Sound:**
- **Control:**
- **Operators:**

[fotos ejemplos de bloques de pencilcode]

1.3. Marco de Referencia

1.4. Estructura de la memoria

En esta sección se debería introducir la estructura de la memoria. Así:

- **Resumen:**
- **Introducción:**
- **Objetivos:**
- **Estado del arte:** En esta sección se enumeran las tecnologías utilizadas en el desarrollo de este proyecto, tanto en el back-end como en el fron-end, realizando una breve descripción de cada una de ellas.
- **Diseño e implementación:**
- **Resultados:**
- **Conclusiones:**

Capítulo 2

Objetivos

2.1. Objetivo general

labelsec:objetivo-general

2.2. Objetivos específicos

labelsec:objetivos-especificos

2.3. Planificación temporal

labelsec:planificacion-temporal

Capítulo 3

Estado del arte

Cliente - servidor. Hablar un poco de esto.

Puedes citar libros, como el de Bonabeau et al. sobre procesos estigmérgicos [1].

3.1. Frontend

El front-end es la capa de presentación de cualquier página o aplicación web. Situada en el lado del cliente, ésta se centra en el diseño de la interfaz con el propósito de que el usuario sea capaz de interactuar con ella de una forma más sencilla, cómoda e intuitiva.

Las tecnologías más utilizadas para desarrollar el front-end de este proyecto han sido las enumeradas a continuación:

Figura 3.1: Tecnologías utilizadas en el front-end

3.1.1. HTML

HTML (en inglés, HyperText Markup Language) es un lenguaje de etiquetas, también denominado lenguaje de marcado, con el que se pueden crear páginas y aplicaciones web. A través

de las marcas o etiquetas, HTML permite dar estructura al texto dividiéndolo en secciones y párrafos, además de incorporar contenido multimedia en sitios web. Actualmente, se trata del estandar más útilizado por todos los navegadores.

En general, por cada elemento, es necesario añadir una etiqueta de apertura junto con su respectiva etiqueta de cierre, ésta última no siempre necesaria. El contenido mostrado en la página será el escrito entre dichas etiquetas. Las etiquetas de apertura pueden incluir atributos, que proporcionan información adicional sobre el contenido del elemento.

Las páginas HTML se suelen dividir en dos secciones principales: la cabecera y el cuerpo, ambas partes delimitadas por sus correspondientes etiquetas <head></head> y <body></body>. En la cabecera se incluye información relevante, no visible para el usuario, sobre la página web, como metadatos, enlaces a documentos externos o información acerca del estilo de la página, mientras que en el cuerpo se incorpora todo el contenido que el usuario puede ver en pantalla como texto y multimedia.

A continuación se muestra como sería la estructura básica de un archivo HTML:

```
<!DOCTYPE html>
<html>
    <head>

        <title> Titulo de la página web </title>
        <meta name="author" content="URJC Libresoft FECyT">
        <link rel="stylesheet" href="static/app/content/style.css">

    </head>
    <body>

        <p> Información que queremos mostrar al usuario </p>

    </body>
</html>
```

Figura 3.2: Documento básico de HTML

3.1.2. CSS

Las Hojas de estilo en cascada (cuyas siglas en inglés significan Cascading StyleSheets) describen como los elementos de un documento escrito en lenguaje de marcado, como puede ser HTML, deberían de ser mostrados o presentados en pantalla. Se utiliza principalmente para el diseño visual de la interfaz de usuario de páginas web, mejorando, así, su apariencia.

A pesar de que en los documentos HTML se puede definir el estilo de sus contenidos no es nada recomendable debido al costoso mantenimiento que esto acarrea. CSS tiene como objetivo marcar la separación del contenido del documento y la forma de presentación o aspecto de éste.

CSS consiste en un conjunto de reglas o normas. Para indicar que norma de estilo debe seguir un elemento HTML, dicho elemento debe identificarse con un atributo de tipo id o class que también será indicado en la regla que se le quiera aplicar. Otra opción sería indicar la etiqueta deseada seguida de las reglas de estilo que se quieran incluir.

Algunas ventajas de utilizar CSS en nuestra web son:

- Compatibilidad con distintos dispositivos: mayor flexibilidad de adaptación a las diferentes plataformas, e.g, dispositivos móviles, impresoras, ordenadores, etc.

- Reducción de la complejidad del documento HTML, evitando la repetición de código fuente.

- Consistencia y ahorro de tiempo: Es aconsejable escribir el código CSS en un documento separado del código HTML. CSS ofrece la posibilidad de aplicar el mismo estilo, escrito en un documento externo .css, a diferentes archivos HTML y múltiples páginas web.

Independientemente de como se quieran aplicar las normas de estilo, es necesario incluir en el HTML la etiqueta <link> para hacer referencia al documento css que se va a utilizar.

```html
<!DOCTYPE html>
<html>
    <head>
        <title> Aplicando un estilo </title>
        <link rel="stylesheet" href="estilo.css">
    </head>

    <body>
        <h1> Mi pagina web </h1>
        <p class='parrafo'> Dr Pencilcode </p>
    </body>

</html>
```

(a) html

```css
h1 {
    font-family: Sans-serif;
    font-size: 16px;
    line-height: 28px;
    font-weight: 300;
    color: #313131;
    text-align: center;
}

.parrafo {
    font-family: Sans-serif;
    color: blue;
    font-size: 12px;
    font-style: italic;
    text-align: center;
}
```

(b) css

Figura 3.3: Ejemplo de aplicación de estilo CSS

3.1.3. JavaScript

JavaScript, conocido comunmente como JS, es un lenguaje de programación interpretado orientado a objetos que se utiliza principalmente para añadir dinamísmo a las páginas web. A pesar de su similitud con el nombre del lenguaje de programación Java, no debe confundirse con éste ya que ambos tienen finalidades muy diferentes.

Proporciona al usuario un mayor control sobre el navegador, además de añadir mejoras en la interfaz, dotando a la web de una mayor usabilidad, interactividad y navegabilidad.

Usado generalmente en el lado del cliente, JavaScript también es utilizado, junto con otras tecnologías o herramientas, para enviar y recibir información del servidor.

3.1.4. jQuery

jQuery es una biblioteca de JavaScript de código abierto y software libre que permite simplificar y facilitar el uso de este lenguage en nuestra página web.

Entre las numerosas funcionalidades que jQuery ofrece, una de las más importantes es la manipulación del arbol DOM (Document Object Model). Mediante el uso de las funciones **jQuery**() o **$**() es posible modificar el contenido de la web sin tener que recargarla.

El DOM es una representación de todos los elementos que conforman una página web. Sigue una estructura en forma de árbol donde los elementos de XHTML, denominados nodos, están interconectados, especificando la relación que existe entre cada uno de ellos. jQuery simplifica la sintaxis para encontrar, seleccionar y manipular dichos elemetos DOM.

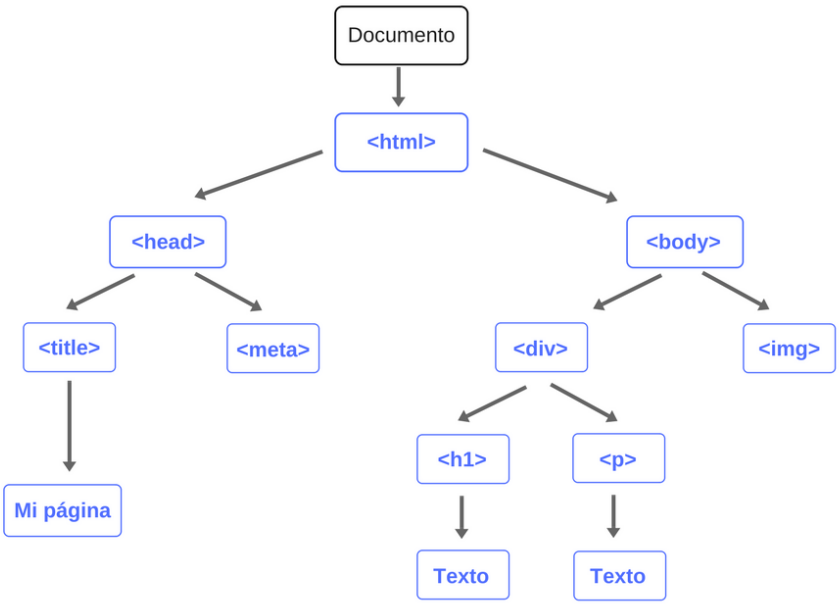


Figura 3.4: Document Object Model

Otras características de esta biblioteca multiplataforma son:

- Eventos HTML
- Manipulación de CSS.
- Animaciones y efectos.
- AJAX.

3.1.5. Bootstrap

Bootstrap[1] es uno de los frameworks más conocidos para la creación de páginas web. Desarrollado por Twitter, Bootstrap contiene plantillas basadas en HTML y CSS que facilitan la

[1] http://getbootstrap.com/

implementación del diseño web, así como extensiones adicionales de JavaScript. Su principal finalidad es convertir el desarrollo del front-end en una tarea más sencilla, rápida y eficaz.

Algunas de las características que hacen de Bootstrap una herramienta tan popular entre los desarrolladores web son las mencionadas a continuación:

- Se trata de una herramienta de código abierto, disponible en GitHub[2], por lo que se puede usar de forma gratuita.
- Tiene compatibilidad con la mayoría de los navegadores web.
- Desde la version 2.0, soporta diseño web responsive. Esta funcionalidad favorece la adaptación dinámica de la interfaz al tamaño del dispositivo (ya sean tablets, teléfonos móviles, pantallas de ordenador, etc) que se esté utilizando para acceder al sitio web.
- Incluye Grid system, útil para diseñar y maquetar por columnas. Este sistema permite añadir hasta un total de 12 columnas en la página.

3.2. Backend

El back-end de una página web, también denominado capa de acceso a datos, se ejecuta en el lado del servidor. Éste es el encargado de analizar los datos de entrada del front-end y de realizar acciones tales como cálculos, procesamiento de la información, gestión de ficheros, interacciones de bases de datos, rendimiento, etc.

Las tecnologías elegidas en esta capa en el desarrollo del proyecto han sido:

Python como lenguaje de programación. **Django** como framework para la creación del sitio web. **Sqlite3** como base de datos. **CoffeLint** como analizador de código CoffeScript. **Servidor HTTP Apache**

3.2.1. Django

Escrito en el lenguaje de programación Python, **Django**, es un framework de software libre y código abierto enfocado al desarrollo web que tiene como objetivo ayudar al programador a crear aplicaciones web complejas de una manera más rápida y clara.

[2] `https://github.com/twbs/bootstrap`

Sigue el patrón de arquitectura de software llamado **Modelo-Vista-Controlador** o MVC, éste divide la aplicación en tres partes interconectadas entre sí con el fin de separar la representación de la información y la forma en la que la información es presentada y aceptada por el usuario (interación con el usuario).

1. **Modelo**: es el componente principal del patrón. Gestiona el acceso y procesamiento de datos, las peticiones para acceder a la base de datos se realizan a través del Controlador.

2. **Vista**: se encarga de presentar al usuario la información disponible en un formato adecuado (generalmente como interfaz de usuario, página HTML) para que éste pueda interactuar con ella. Dicha información se obtiene a traves del Modelo.

3. **Controlador**: esta capa se encarga de gestionar y responder las solicitudes realizadas por el usuario apoyándose tanto en el Modelo como en la Vista.

En el caso de Django, el patrón de diseño sigue la estructura mostrada a continuación:

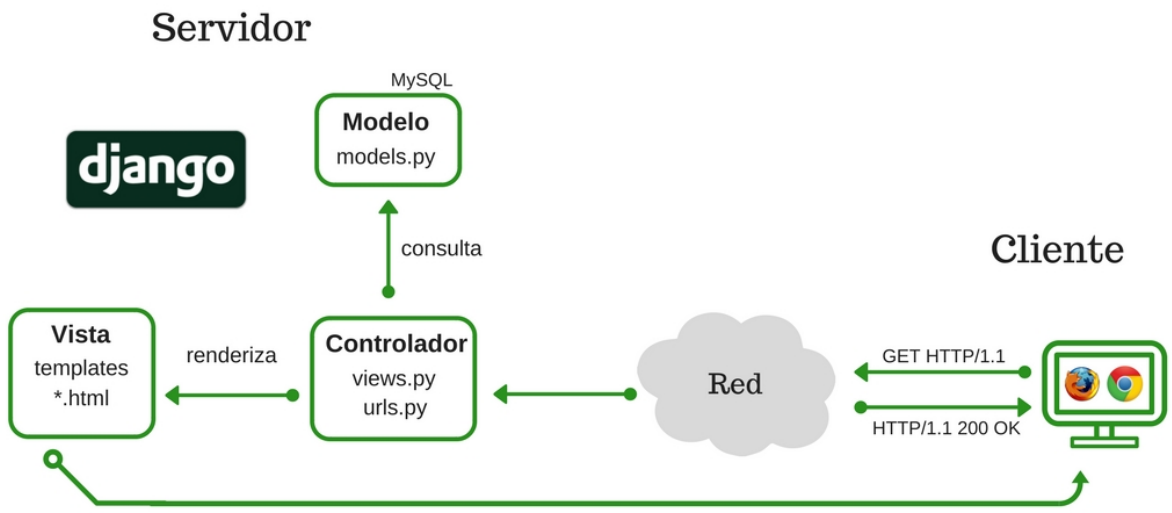


Figura 3.5: Patrón de diseño MVC de Django

Algunas de las características más destacadas de Django son:

-
-
-
-

3.2.2. Python

Desarrollado por Guido van Rossum a finales de los ocheta y con el objetivo de mejorar y ampliar las funcionalidades del *lenguaje de programación ABC*, **Python**, es un lenguaje de programación de alto nivel, interpretado y orientado a objetos que se caracteriza por su concisión, legibilidad y transparencia. Debe su nombre a los humoristas británicos *Monthy Python*.

Python ofrece un amplio abanico de posibilidades, desde la pogramación web tanto en el lado del cliente como del servidor(Django), hasta la programación con bases de datos. Es considerado como uno de los lenguajes más populares en lo que al mundo de Open Source se refiere.

Se conocen tres versiones, 1.X, 2.X y 3.X, la primera de ellas obsoleta, cuyos últimos lanzamientos han sido las versiones 1.6, 2.7 y 3.4 respectivamente. Desde la versión 2.1, Python posee una licencia compatible con *GNU General Public License*, denominada **Python Software Foundation License**, además, desde entonces, Python Software Foundation es la fundación sin ánimo de lucro responsable del código del lenguaje y de procesos como la administración de los derechos de autor y la obtención de fondos para la comunidad Python.

Las caraterísticas más significativas de este lenguaje son:

- Orientado a objetos:
- Multiplataforma: se puede implementar en diferentes plataformas informáticas.
- Open Source:
- Interpretado: el código se ejecuta directamente mediante un interprete, siendo innecesaria la compilación previa del programa a código máquina.
- Multiparadigma: soporta el uso de dos o más paradigmas de programación. Python es orientado a objetos, imperativo, reflexivo y funcional.
- Fuertemente tipado: si lo tipos no son exactamente iguales, no es posible pasar una variable como parámetro de procedimiento. Se debe realizar una conversión con anterioridad para que coincidan.

- Librerias: Hay cantidad de librerias disponibles para ampliar la funcionalidad de Python.

3.2.3. Apache

3.2.4. Sqlite Mysql

3.2.5. CoffeeLint

CoffeLint es un analizador estático de programas escritos en CoffeScript, se trata de un software de código abierto[3] bajo la licencia MIT (Massachusetts Institute of Technology) que puede instalarse vía línea de comandos. Éste se encarga de comprobar que el programa sea consistente, limpio y esté correctamente escrito.

By default, CoffeeLint will help ensure you are writing idiomatic CoffeeScript, but every rule is optional and configurable so it can be tuned to fit your preferred coding style

helps enforcing a coding standard

[3] `https://github.com/clutchski/coffeelint`

Capítulo 4

Diseño e implementación

4.1. Arquitectura general

figura 4.1.

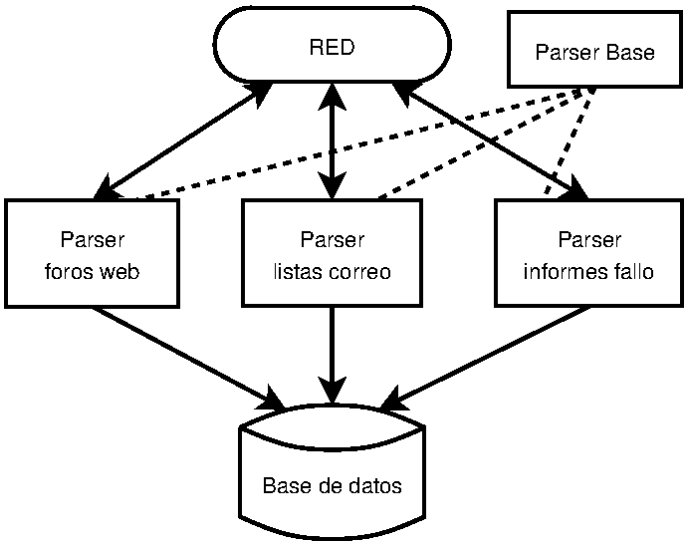


Figura 4.1: Estructura del parser básico

Capítulo 5

Resultados

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.

6.2. Aplicación de lo aprendido

A medida que he ido trabajando y avanzando en el proyecto, he sido testigo del valor de los conocimientos adquiridos en las asignaturas cursadas a lo largo de la carrera. Este proyecto ha sido un claro ejemplo de ello ya que, durante su desarollo, he aplicado gran parte de lo aprendido en asignaturas como:

1. **Informatica I:** me permitió un primer contacto con el mundo de la programación, gracias a ella, aprendí, desde cero, las técnicas y conceptos básicos de la programación de computadores. Qué era una función, qué era un procedimiento, un bucle, un array, una condición, etc. Términos que han sido necesarios para poder llevar a cabo este proyecto.

2. **Informatica II:** Al igual que Informática I, esta asignatura continuó con mi formación como programadora, aunque el lenguaje con el que trabajé fue Ada y no el utilizado en este

proyecto (Python), fue muy útil para seguir mejorando mis habilidades y conocimientos.

3. **Arquitectura de Internet:** fue la primera que cursé sobre el campo de las Redes de Ordenadores e Internet. En concreto, la asignatura me enseñó como está estructurada la red, así como las arquitecturas de los principales protocolos de redes de ordenadores.

4. **Sistemas Telematicos para Medios Audiovisuales:** siguiendo con los prótocolos, aquí conocí el funcionamiento del protocolo HTTP, entre otros.

5. **Protocolos para la transmisión de Audio y Video en Internet:** fundamental para la realización de este proyecto ya que fue aquí donde aprendí a programar en Python y gracias a la cual he podido llevar un seguimiento más sencillo en el desarrollo del fichero views.py.

6. **Construcción de Servicios y Aplicaciones Audiovisuales en Internet:** En la cual, adquirí los conocimientos básicos de HTML, CSS y Javascript, todos ellos empleados en el desarrollo de Dr.Pencilcode.

7. **Gráficos y visualización en 3D:** Aunque ésta estaba orientada a la creación de videojuegos tanto en 2D como 3D, los lenguajes utilizados para ello fueron WebGL y Javascript, lo que me permitió sequir puliendo mi pensamiento computacional.

6.3. Lecciones aprendidas

Con el desarrollo de Dr.Pencilcode he conseguido ampliar y mejorar los conocimientos adquiridos durante la carrera además de aprender muchos otros:

1. He aprendido a aprender de forma autodidacta. Django.

2. Nuevas tecnologias: MysQL, squlite3:

3. regex, python

4. json,

5. Latex

6. Servidor Apache, maquinas virtuales

7. He ganado experiencia en el area de Diseño Web, no solo a nivel de diseño gráfico web,(html, css) sino tambien considerando pensando a nivel de usuario, usabilidad, interactividad.facilidad con que las personas pueden utilizar una herramienta Bootstrap, jQuery, JavaScript.

8. pensamiento computacional

9. Rúbricas. pensar forma de evaluar. quizá ha sido lo más complicado.

   En definitiva, coom diferentes tecnologías se unen/se coordinan con un objetivo/fin común.

6.4. Trabajos futuros

Ningún 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.

6.5. Valoración personal

Finalmente (y de manera opcional), hay gente que se anima a dar su punto de vista sobre el proyecto, lo que ha aprendido, lo que le gustaría haber aprendido, las tecnologías utilizadas y demás.

Apéndice A

Manual de usuario

Bibliografía

[1] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Articial Systems*. Oxford University Press, Inc., 1999.