upna
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Departamento de Estadística, Informática y Matemáticas

Trabajo de fin de grado

Plataforma de adquisición de datos pluviométricos para la predicción y aviso de inundaciones

Autor:
Arkaitz Oderiz Garin

Supervisor:
Unai Pérez-Goya

Pamplona, 2022

Contenidos

Lista de Figuras

Lista de Tablas

Lista de Códigos

Lista de Teoremas

Lista de Definiciones

Lista de Lemas

1 Introducción

1.1 Esto es una sección

Aquí tenemos una imagen referenciada 1.1. Una dirección Mi dirección[1].

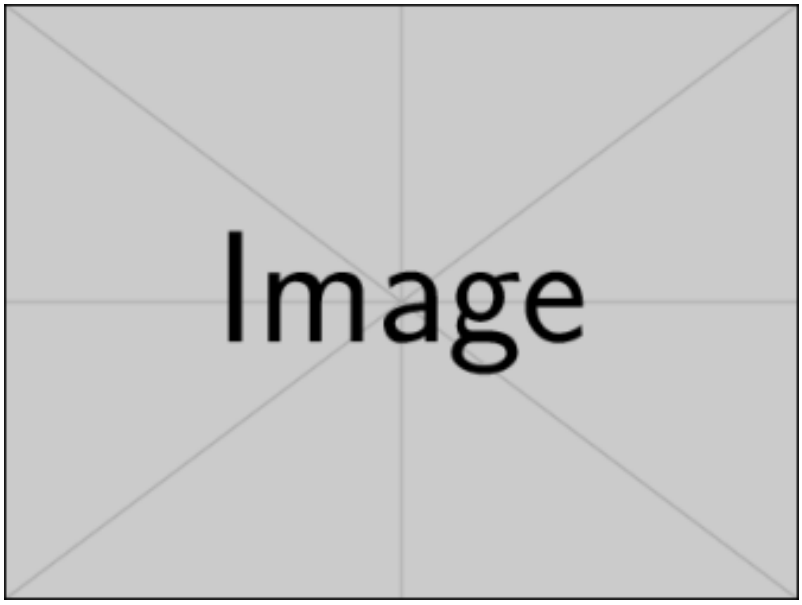

Figura 1.1: Real caption

Para generar entradas en el índice de palabras. SATA.

1.1.1 Esto es una subsección

Una lista de parámetros:

[1] https://www.copernicus.eu/es

- uno.
- dos.
- tres.

Una lista enumerada

1. uno.
2. dos.

Vamos a citar . . . [1]

Tabla 1.1: Mi tabla de ejemplo

Nombre	Medida	Otra cosa
10	10	4

2 Tecnologías

2.1 Lenguaje de Programación Python

2.1.1 Qué es Python?

Siendo su primera aparición en el año 1991 por manos de Guido van Rossum, Python es un lenguaje de programación de alto nivel interpretado que prima la legibilidad del código, siendo este a veces nominado como "seudocódigo ejecutable".

Python a su vez es un lenguaje multiplataforma, fuertemente tipado, dinámico y multiparadigma, pues soporta programación orientada a objetos, imperativa y funcional.

2.1.2 Por qué Python?

La elección de Python sobre otros lenguajes se basa en su sintaxis sencilla y clara que facilita la programación, junto a la gran cantidad de librerías y frameworks potentes de los que dispone para satisfacer las necesidades que presenta este proyecto, como pueden ser la extracción de datos meteorológicos mediante web scraping como la creación de una API.

2.2 Sistema Operativo Debian

2.2.1 Historia

Conforme la computación fue tomando terreno tanto en el ámbito comercial como en el escolar, no tener una forma de instalar y configurar los sistemas de una forma rápida sin tener que partir de cero y sin tener que compilar el software necesario manualmente se convirtió en un problema patente entre los usuarios.

En 1993, tras varios intento fallidos por distintas empresas de solucionar el problema, Ian Murdock, por aquel entonces estudiante de la Universidad Purdue, encontró la solución al problema basándose en el reciente proyecto de Linus Torvalds, el kernel Linux. tras el anuncio de Ian para crear un sistema operativo de forma descentralizada en paralelo como es el caso del kernel Linux, docenas de usuarios se unieron para formar el proyecto Debian Linux con la intención de crear un sistema operativo de gran calidad y mantenimiento, publicando en enero de 1994 la primera versión de Debian 0.91.

2.2.2 Filosofía

Debian no intenta seguir ni competir con los lideres del sector, por el contrario, desde sus inicios el proyecto se ha basado en una filosofía centrada en la robustez y estabilidad del sistema guiada por unos estrictos estándares de calidad, actualizándose conforme las necesidades de sus usuarios a la vez que promueve el software gratuito, lo que le ha ayudado a obtener fama entre los usuarios.

A su vez, debido al apoyo del software libre, hace uso de múltiples licencias de software como la Licencia Pública General GNU (GPL), licencias artísticas o del tipo BSD, lo cual ha llevado al desarrollo de las Directrices de Software Libre de Debian (DFSG) con el fin de definir la construcción del software libre.

Definido por estas licencias, el software libre debe cumplir al menos las que están consideradas la cuatro libertades esenciales:

- Ejecutar el programa como se desee, con cualquier propósito.
- Estudiar cómo funciona el programa, y cambiarlo para que haga lo que se desee.
- Redistribuir copias para ayudar a otros.
- Distribuir copias de sus versiones modificadas a terceros permitiendo ofrecer a toda la comunidad la oportunidad de beneficiarse de las modificaciones.

2.2.3 Modelo de negocio

Todo el modelo del proyecto Debian se basa en su Contrato Social respecto a la comunidad de software libre creado en 1997, sobre el cual se estipulan las directrices a seguir para crear y distribuir software libre.

No hay que confundir el termino libre, definido dentro de DFSG, con gratuito, pues un programa de software puede ser libre aunque sea de pago. Por el contrario, Debian es un proyecto libre y gratuito.

Debian se mantiene gracias a su comunidad, disponiendo más de mil desarrolladores y contribuidores por todo el mundo aportando al proyecto con su tiempo y conocimiento de forma gratuita. Esto, junto a la asistencia de múltiples empresas que dan soporte a Debian dentro del proyecto de socios y el sistema de donaciones usado para hardware, dominios, certificados criptográficos, conferencias, etc han ayudado en el éxito y crecimiento del proyecto dentro de su filosofía.

2.2.4 Por qué Debian?

El mercado está repleto de distintas posibles alternativas a Debian, ya sean de pago, Windows Server OS o Red Hat Enterprise Linux (RHEL), gratuitos, Ubuntu Server y Fedora Server, o incluso en la nube, Amazon Web Services (AWS), Google's Cloud Platform.

Descartando todo sistema de pago, aunque las posibilidades se reducen aun hay múltiples opciones sobre las que elegir, pero son pocas las que ofrecen la misma usabilidad que Debian con sus más de 59000 paquetes en su versión estable o incluso puedes usar las versiones «en pruebas» o «inestable» en caso de querer probar las nuevas funciones antes de su lanzamiento oficial.

Aunque todos los sistemas basados en Linux disponen de las mismas características, siendo software libre y gratuito con soporte multi-usuario, multi-proceso y uso en tiempo real, Debian siendo uno de los sistemas más longevos del mercado a tomado fama entre la competencia por su seguridad y estabilidad. Siendo la base para muchas de las distribuciones más populares contra las que compite, como Ubuntu, Knoppix, PureOS o Tails.

Cabe mencionar, que parte de esta seguridad y estabilidad puede llegar a ser un limitante a la hora de elegir Debian como sistema operativo dependiendo de tus necesidades a nivel de uso pues el software compatible igual no es la versión más reciente.

Finalmente, una característica distintiva de Debian frente a la competencia gratuita es su compatibilidad con un uso 24/7, pues otras alternativas gratuitas o no dan soporte a esta característica, Fedora Server o, necesitas disponer de una licencia de pago como es el caso de Ubuntu Server mediante Ubuntu Pro.

Una vez contado todo esto, puesto que no me afecta negativamente el no disponer de las versiones más recientes de software y necesito de un sistema 24/7 gratuito, parece lógica la elección de Debian como sistema operativo para usar en el servidor.

2.3 Framework

2.3.1 Qué es un framework?

Un framework es software que provee una infraestructura básica sobre la que desarrollar tus proyectos, aportando las funcionalidades y estructuras básicas necesarias para este sin la necesidad de programar todo desde cero, ahorrando tiempo de desarrollo y aportando robustez al proyecto.

Cada framework aportará su propia colección de módulos y paquetes específicos para ayudar en el desarrollo, es por esto que generalmente se clasifican en tres clases distintas según funcionalidades.

Tipos de frameworks

Full Stack
Un framework full-stack es apto tanto para deserrollo back-end como front-end, aportando todas las herramientas posibles que ayuden con el desarrollo gráfico de la interfaz de usuario (UI), gestión de bases de datos, protocolos de seguridad y lógica de negocio entre tantos. Siendo Django un ejemplo de framework full-stack.

Micro
Los framework micro son ligeros por definición, siendo en cierta medida lo contrario de un framework full-stack, pues aunque los componentes que aportan como puede ser la gestión de bases de datos son los mismos, estos no vienen incluidos de forma nativa. Esto se debe a que buscan aportar flexibilidad y libertad a los desarrolladores para que incluyan unicamente aquellas herramientas que necesiten.

Como se explica en la documentación de Flask, uno de los framework tipo micro más relevante, el 'micro' de microframework significa que el núcleo del framework es simple pero extensible.

Asíncrono
Estos framework están dirigidos por eventos. en vez de hacer un manejo operacional linea a linea de las funciones en la que se van ejecutando una detrás de otra, el código asíncrono no es bloqueante por lo que no se espera que un evento termine para ejecutar el siguiente, ejecutándolo de forma simultanea. Debido a esto un framework asíncrono puede llegar a conseguir un gran rendimiento si se usa en un servidor que lo permita.

2.3.2 Librería vs Framework

Aunque ambas ofrecen funcionalidades operacionales, su mayor diferencia radica en la especificidad y complejidad de estas.

Las librerías están compuestas por múltiples métodos para un uso especifico sin aportar mucha complejidad, realizando una tarea por función.

Por el contrario, como los framework tienen en cuenta las posibles necesidades de tu proyecto, pudiéndose permitir ser aún más específicos, ofreciendo la arquitectura y comportamiento básico de la aplicación, dejando la flexibilidad de desarrollar las funcionalidades necesarias para su funcionamiento, aportando herramientas sobre las que trabajar.

2.3.3 Django

Qué es Django?

Djanfo es un framework web de tipo full-stack gratuito y de código abierto para Python. Sigue el principio DRY "Don't Repeat Yourself" por lo que se enfoca en el menor uso de codigo, el desarrollo rápido y la reutilización de componentes.

Hace uso de su auto denominado patrón Modelo-Vista-Template (MVT) 2.1 una variante del conocido Modelo-Vista-Controlador (MVC).

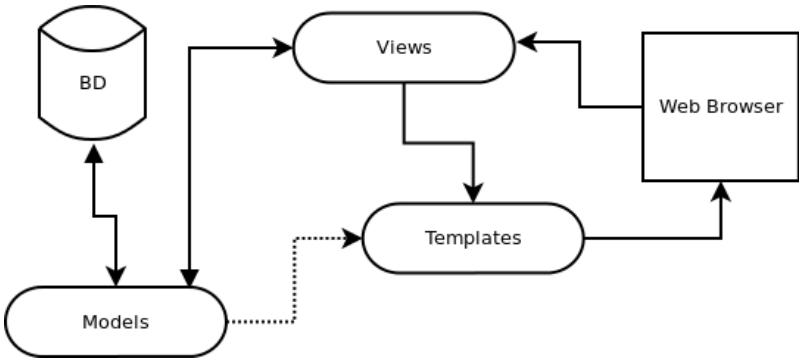

Figura 2.1: Diagrama patrón MVT

Para poder trabajar con bases de datos relacionales tales como, Oracle, MySQL y PostgreSQL, Django usa un Mapeador Relacional de Objetos (ORM) permitiendo interactuar con ellas mediante SQL.

[0]https://docs.hektorprofe.net/django/web-personal/patron-mvt-modelo-vista-template/

Por qué Django?

Partiendo de que el lenguaje de programación a usar es Python, Django es uno de los frameworks más reputados dirigidos a este lenguaje, teniendo en cuanta su naturaleza gratuita y de código abierto. Siendo usado tanto por la comunidad como por empresas tales como Instagram, Mozilla y Facebook. Aunque no por ello significa que no disponga de poca competencia, siendo TurboGears, web2py, Bottle y Flask de las más notables.

El que sea un framework full-stack no va a aportar la flexibilidad, libertad y ligereza que da el uso de un microframework, sobre todo a la hora de agregar únicamente aquellas herramientas que considere necesarias, pero si que atenuará la carga de trabajo que puede suponer un microframework si no se dispone de la experiencia necesaria para usarlo.

Otra característica por la que me he decantado por Django, aunque no sea única de él, es su compatibilidad con bases de datos relacionales de forma nativa, cosa que facilitara el uso de estas en el proyecto.

2.3.4 Scrapy

Qué es Scrapy?

Por qué Scrapy?

3 Algoritmo

Código en R del ejemplo 3.1:

```r
# Establecer las credenciales de la API [1]

library(rsat)
set_credentials("rsat.package", "UpnaSSG.2021")

#Definir la region

library(raster)
dir.create( "./RSATprueba/countries" ,recursive = TRUE)
spain<-getData('GADM', country= 'Spain', path=
   "./RSATprueba/countries", level=2)
```

Código 3.1: pie de código

Teorema 1 (Sum). $1 + 1 = 2$

Definición 1 (Nice numbers). *A number is* nice *if it looks beautiful.*

Teorema 2 (About $C^1(0, 1)$). *The set* $C^1(0, 1)$ *is interesting.*

Teoremaren erreferentzia 2

Proof. To prove it by contradiction try and assume that the statemenet is false, proceed from there and at some point you will arrive to a contradiction. □

Lema 1. *To prove it by contradiction try and assume that the statemenet is false, proceed from there and at some point you will arrive to a contradiction.*

Lemaren erreferentzia 1

$$1 + e^{i\pi} = 0. \tag{3.1}$$

Formularen erreferentzia 3.1

4

Evaluación

5

Conclusiones y Trabajo Futuro

Referencias

[1] L. Lorenzi, F. Melgani, and G. Mercier, "Inpainting strategies for reconstruction of missing data in vhr images," *IEEE Geoscience and remote sensing letters*, vol. 8, no. 5, pp. 914–918, 2011.

Glosario

Anexos

```
1     public void mostrarTodasNotas(){
2         int a = 1;
3         System.out.println("Estas son todas las notas que hay
              guardadas.");
4         for(int j=0; j<lista_notas.size(); j++){
5             System.out.print(a+"-");
6             System.out.println(lista_notas.get(j).toString());
7             a++;
8         }
9     }
```

Código A1: Mostrar notas.