Deusto
Universidad de Deusto

Ingeniería Web

Mikel García, Jon Cañadas y Eneko Fuente

Entrega 4 – Documentación de la aplicación en Django.

29/05/2024

Índice.

Contenido

1. Introducción 3
2. Objetivos del proyecto. 3
   2.1. Resumen para la dirección. 3
   2.2. Tareas principales. 3
   2.3. Planificación temporal. 4
3. Especificación de requisitos del sistema. 4
   3.1. Descripción general. 4
   3.2.1. Catálogo de requisitos. 4
   3.2.2. Modelo lógico de datos. 5
   3.2. Representación de la interfaz del sistema. 5
   3.3. Representación jerárquica de la interfaz. ..... **¡Error! Marcador no definido.**
4. Especificación del diseño. 6
   4.2. Introducción. 6
   4.2.1. Principales funciones del software. 6
   4.2.2. Descripción del entorno de desarrollo. 6
   4.3. Esquema de interacción entre los clientes y el servidor. 7
   4.4. Diseño de la estructura física de los datos. 8
5. Manual de usuario. 8
6. Incidencias del proyecto y conclusiones. 9
7. Bibliografía. 10

1. Introducción

Deustotil Tech S.L. es una empresa de consultoría que gestiona sus operaciones mediante proyectos específicos para sus clientes. La necesidad de mejorar la organización y planificación de estos proyectos ha llevado a la decisión de desarrollar un software que facilite estas tareas.

El objetivo principal de este proyecto es desarrollar un software de gestión de proyectos que permita a los jefes de equipo planificar, asignar y seguir el progreso de las tareas dentro de cada proyecto, optimizando la comunicación y eficiencia en la empresa.

2. Objetivos del proyecto.

2.1. Resumen para la dirección.

La aplicación de gestión de proyectos ha sido desarrollada en Django, un framework de desarrollo de aplicaciones web en Python.

En esta aplicación se podrá navegar entre diferentes páginas, con diferentes funcionalidades como por ejemplo visualizar, añadir, modificar y eliminar los proyectos, tareas, clientes y empleados de la empresa. Además, cuenta con un sistema de identificación y autenticación para que solamente puedan acceder los usuarios correspondientes.

2.2. Tareas principales.

En primer lugar, se ha llevado a cabo el planteamiento de la estructura de datos de la aplicación, para ello ha sido necesario adquirir conocimientos sobre las necesidades de Deustotil Tech para llevar a cabo la solución.

Mas adelante se ha creado un repositorio de control de versiones en GitHub en donde podremos guardar todos los cambios realizados en la aplicación y trabajar así de forma simultánea.

La estructura consta de proyectos solicitados por los clientes, con sus respectivas tareas asignadas a diferentes empleados. En el apartado 3.2.2 se encuentra el diagrama de entidad-relación en donde se ve más específicamente esta estructura.

La siguiente tarea identificada ha sido la configuración del proyecto y la aplicación en Django. Es decir, la construcción de los cimientos necesarios para pasar de la idea planteada a crear los modelos, vistas y todo lo que vendría a ser la primera parte del backend.

Una vez creado el entorno de programación, se ha pasado ha crear las funcionalidades mencionadas anteriormente en el apartado 2.1, además de comenzar con el frontend o parte visual de la aplicación utilizando diferentes plantillas HTML para las funcionalidades CRUD(create, read, update y delete), decorando las mismas con archivos estáticos CSS y JavaScript.

Por último, se han añadido a la base de datos todos los proyectos, clientes y empleados de la empresa para permitir a los usuarios comenzar a utilizar la aplicación desarrollada.

2.3. Planificación temporal.

El reparto de las tareas identificadas ha sido elegido de forma equitativa por parte de los representantes del grupo y cada uno de ellos ha tomado acción en cada una de las tareas, no obstante, se ha elegido un responsable principal para llevar a cabo cada una de ellas.

A continuación, se muestra de una forma más visual la información sobra cada una de las tareas realizadas.

Tabla 1: Reparto de tareas

Tarea	Encargado	Inicio	Fin
Diagrama Entidad-Relación	Eneko Fuente	24/04	27/04
Creación Repositorio GitHub	Jon Cañadas	29/04	29/04
Setup Django	Mikel García	29/04	3/05
Creación de Templates	Jon Cañadas	30/04	17/05
Aplicar estilos con CSS	Eneko Fuente	30/04	27/05
Funcionalidades CRUD	Mikel García	2/05	6/05
Funcionalidades Login + Contacto	Jon Cañadas	23/05	26/05
Adición de datos empresariales	Mikel García	25/05	25/05
Funcionalidades JavaScript	Jon Cañadas	23/05	27/05
Fetch API	Mikel García	26/05	27/05
Documentación adicional (readme, requirements y otros archivos)	Eneko Fuente	27/05	28/05

3. Especificación de requisitos del sistema.

3.1. Descripción general.

3.1.1. Catálogo de requisitos.

Funciones que cumplir:

- Gestionar Proyectos: Crear, modificar, borrar y detallar.
- Gestionar Tareas: Crear, modificar, borrar y detallar.
- Gestionar Notas: Añadir notas a tareas.

Funcionalidades adicionales:

- Inicio de sesión: Autentificación y registro de usuarios para la web.
- Envió de correos: Enviar correos desde la página web.
- Gestionar Clientes: Añadir, modificar, borrar y detallar.
- Gestionar Empleador: Añadir, modificar, borrar y detallar.

3.2. Modelo lógico de datos.

En el siguiente modelo de clases se pueden observar las relaciones entre entidades del ejercicio y los valores utilizados en cada uno de los atributos.

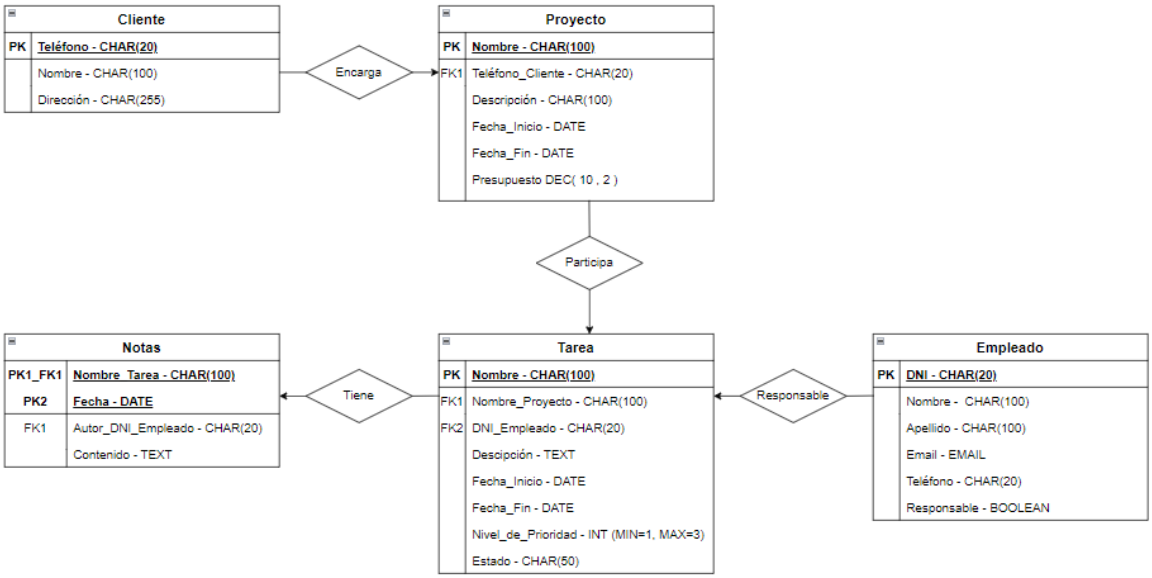

Ilustración 1: Modelo lógico de datos.

3.3. Representación de la interfaz del sistema.

Pantalla de inicio: En esta pantalla se realiza la autentificación de los usuarios para poder acceder a la pantalla principal de la web. En el caso de no haber iniciado sesión, se redirige continuamente a la pantalla de inicio.

Pantalla registro: En esta pantalla se puede crear un usuario en el caso de no tenerlo.

Pantalla principal: Esta pantalla hay una serie de tablas listadas para realizar la gestión de los diferentes elementos.

Pantallas crear/añadir: En esta pantalla hay una de serie de campos a rellenar para la creación de dichos elementos.

Pantallas modificar: En esta pantalla los campos del elemento seleccionado aparecerán completos por su valor actual pudiendo ser modificables a voluntad.

Pantallas detallar: En esta pantalla se pueden observar todos los valores del elemento solicitado ya que en la pantalla principal solo se encuentran algunos valores.

3.3.1. Representación jerárquica de la interfaz.

A continuación, se puede observar la representación de la interfaz jerárquica en la aplicación:

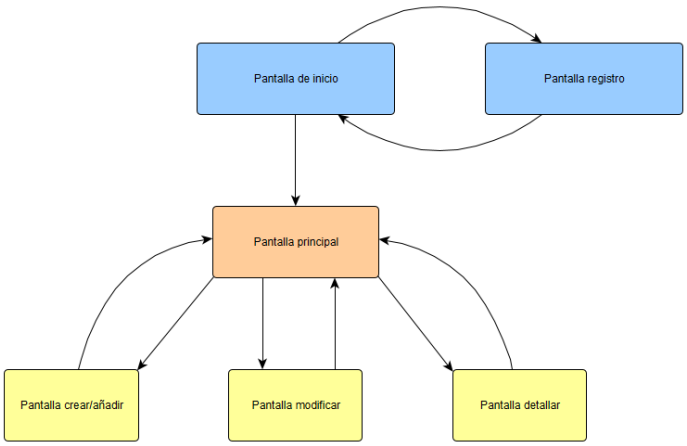

Ilustración 2: Diagrama jerárquico de la interfaz.

4. Especificación del diseño.

4.1. Introducción.

En este apartado, exploraremos las múltiples funcionalidades del sistema web desarrollado utilizando diversos lenguajes de programación, diseñado para realizar determinadas funciones y con determinadas capacidades avanzadas.

4.1.1. Principales funciones del software.

En este apartado, abordaremos las diversas funcionalidades del sistema web desarrollado utilizando lenguajes de programación como Python, HTML, CSS y Java. Este sistema es capaz de gestionar clientes, tareas, proyectos y empleados, incluyendo funciones de adición, modificación y eliminación de estos elementos.

Además, el sistema incorpora funcionalidades avanzadas, como el envío automático de correos electrónicos, un sistema de ampliación de información y un sistema de registro y login de usuarios. También se ha implementado un script en Java para generar contenido HTML dinámico, así como un script para la disminución de tamaño de algunos textos concretos.

4.1.2. Descripción del entorno de desarrollo.

El entorno de desarrollo es un aspecto crucial para el éxito de cualquier trabajo de ingeniería web. A continuación, describiremos el entorno de trabajo que hemos utilizado, detallando las diferentes herramientas, lenguajes y configuraciones empleadas.

En primer lugar, el equipo de desarrollo está formado por tres desarrolladores que han trabajado de manera conjunta y eficiente en computadoras personales para lograr el mejor proyecto posible.

En cuanto a las herramientas y tecnologías empleadas, hemos utilizado una variedad de lenguajes de programación. Python ha sido empleado para la lógica del servidor. Por otro lado, HTML y CSS se han utilizado para estructurar y diseñar la interfaz con la que interactuarán los usuarios. Finalmente, JavaScript ha sido utilizado tanto para la disminución del tamaño de las

fuentes empleadas en los diferentes textos de la página como para la generación dinámica de contenido HTML.

Para el desarrollo del código en los diferentes lenguajes, hemos utilizado la aplicación Visual Studio Code. Además, para el control de versiones y la colaboración en el desarrollo y almacenamiento del código fuente, hemos empleado GitHub, lo que nos ha permitido un seguimiento sencillo y detallado del código.

En cuanto a la base de datos, hemos utilizado db.sqlite3, que funciona como nuestro sistema de gestión de bases de datos. Esta base de datos es predefinida en entornos de desarrollo como Django y Python, y nos permite gestionar la información relacionada con clientes, tareas, proyectos y empleados.

4.2. Esquema de interacción entre los clientes y el servidor.

A continuación, se muestra un diagrama ilustrativo de las interacciones entre cliente y servidor.

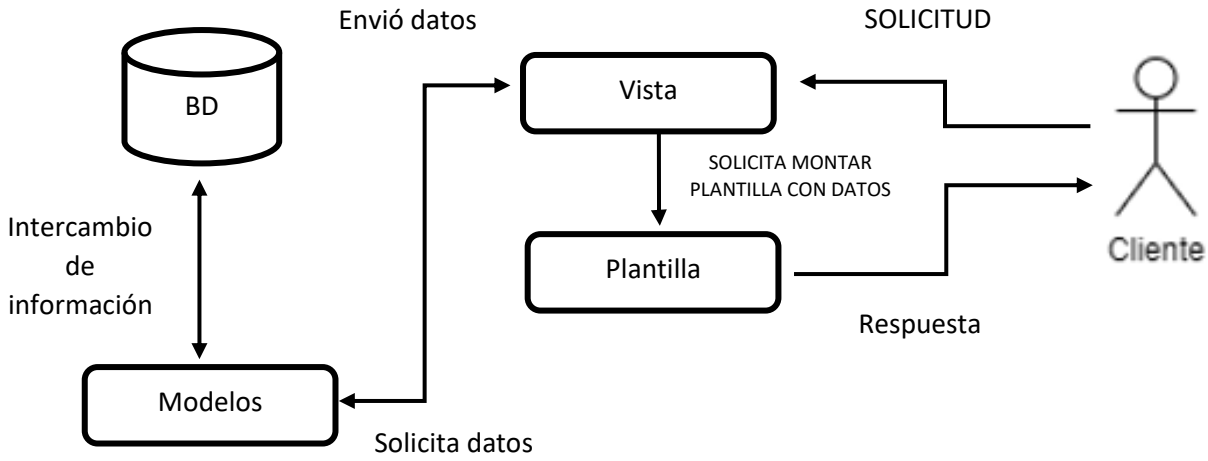

Ilustración 3: Diagrama interacción usuario-servidor

4.3. Diseño de la estructura física de los datos.

A continuación, se muestra un diagrama ilustrativo de las interacciones entre cliente y servidor.

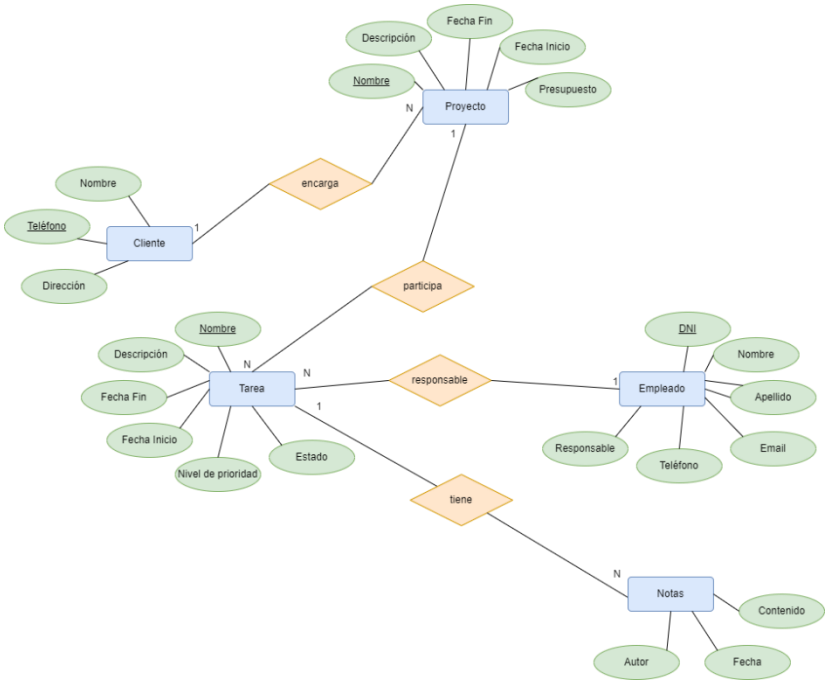

Ilustración 4: Diagrama entidad-relación

5. Manual de usuario.

Lo primero que se aparece es un formulario de autenticación de usuario, en donde se puede registrar o iniciar sesión con el usuario “empresadjango” y contraseña “empresadjango”.

Una vez iniciada la sesión accederá a la página principal que se trata de un botón informativo en donde puede hacer clic para conocer más sobre la empresa, justo debajo nos encontramos con la barra de funcionalidades en donde puede elegir desde volver al propio listado, añadir elementos de la empresa, recibir información de contacto a su correo electrónico y cerrar sesión.

En cada listado podrás apreciar una vista previa de los proyectos, tareas, empleados y clientes. Cada uno de ellos cuenta con las diferentes opciones:

Ver: Visualizar todos los detalles sobre ese elemento

Ver (JS): Visualizar todos los detalles sobre ese elemento en el mismo listado sin tener que recargar la página (no funciona correctamente, toda la explicación está en el Google Forms)

Modificar: Aparecerá un menú en donde se puede modificar cualquier atributo del elemento seleccionado.

Borrar: Al hacer clic en este botón, automáticamente se eliminará de la lista el elemento y a modo de cascada todos sus elementos asociados, es decir, si borras un proyecto se borrarán sus tareas y también las notas asociadas a cada una de esas tareas.

6. Incidencias del proyecto y conclusiones.

Durante el desarrollo del software de gestión de proyectos Deustotil Tech S.L., se encontraron diversas dificultades que fueron solucionadas. A continuación, se detallan las principales incidencias y las soluciones adoptadas, así como las conclusiones derivadas de la experiencia.

Inicialmente, se detectó que el esquema entidad-relación diseñado contenía errores, lo que llevó a la creación de modelos en models.py que no representaban correctamente las relaciones y estructuras necesarias. Esta situación se resolvió revisando y corrigiendo el diagrama entidad-relación, permitiendo ajustar los modelos en Django y definir correctamente las relaciones entre entidades, eliminando campos innecesarios.

En cuanto a la interfaz de usuario, inicialmente, las tareas y proyectos se mostraban en una misma página, lo que dificultaba la navegación y la claridad de la información. La solución fue separar estas entidades en páginas diferentes, mejorando la organización y el acceso a la información. Además, se añadió la funcionalidad de ver las tareas específicas de un proyecto directamente desde la página del proyecto, optimizando así la gestión y el seguimiento de las tareas.

La interfaz también presentaba demasiado desplazamiento vertical, afectando la experiencia del usuario. Para mejorar esto, se volvió a rediseñar la interfaz haciéndola más compacta y eficiente.

Hubo problemas para integrar la funcionalidad de JavaScript destinada a la validación de campos de formularios. Ante esta dificultad, se decidió implementar una funcionalidad alternativa y así cumplir con los requisitos de la entrega.

Por último, los datos de ejemplo iniciales eran limitados, dificultando las pruebas y la validación del sistema. Se añadieron más datos de ejemplo, cubriendo una variedad de escenarios para asegurar un testeo más robusto.

En cuanto a los puntos de mejora, el primero de ellos sería la previa validación de los esquemas de datos desde el inicio del proyecto para evitar problemas mayores durante las etapas de desarrollo. Una correcta definición del diagrama entidad-relación y su implementación precisa en models.py es crucial para asegurar la integridad de los datos y el funcionamiento correcto de la aplicación.

Otro punto de mejora sería asegurarnos de que las funcionalidades y las vistas de la aplicación estén claramente separadas y organizadas, evitando la mezcla de diferentes tipos de información en una misma página. Esto mejora la navegación y la claridad de la información para los usuarios.

Por último, mantener el código limpio y eliminar elementos innecesarios en archivos críticos como models.py es fundamental. Mejorando el rendimiento del sistema.

A modo de conclusión cabe destacar que finalmente el software desarrollado ha alcanzado los objetivos planteados, proporcionando una herramienta eficaz para la gestión de proyectos

dentro de la empresa. Las mejoras identificadas durante el proyecto, como la validación temprana de esquemas o la optimización de la interfaz, serán fundamentales para futuros proyectos.

La experiencia adquirida en este proyecto no solo ha mejorado las habilidades técnicas del equipo, sino que también ha resaltado la importancia de una buena organización y comunicación. De esta forma hemos adquirido lecciones para así optimizar tanto el proceso de desarrollo como la funcionalidad del software en versiones futuras, asegurando así el éxito de futuros proyectos.

7. Bibliografía.

- Lazo, R. (2022, 11 de agosto). CRUD en Django 2022. Dev.to. https://dev.to/rodrigolazo/crud-en-django-2022-1ga6

- Vadillo, J. (2021, 2 de abril). Curso completo de Django en 15 minutos [Video]. YouTube. https://www.youtube.com/watch?v=hKXn7Sub-QU&t=387s

- Ray, R. (2020, 2 de diciembre). Django y Fetch API: Envíos de formularios sin recargar la página. Medium. https://ridwanray.medium.com/django-and-fetch-api-form-submissions-without-page-reloading-dc5106598005

- JSONPlaceholder. (s.f.). JSONPlaceholder - API REST falsa gratuita. https://jsonplaceholder.typicode.com/