Deusto

Facultad de Ingeniería
Universidad de Deusto

Ingeniaritza Fakultatea
Deustuko Unibertsitatea

Grado en Ingeniería Informática
Informatikako Ingeniaritzako Gradua

Proyecto fin de grado
Gradu amaierako proiektua

Colmena: diseño e implementación de un widget web para micro donaciones, con página web de soporte, almacenamiento de datos y su posterior visualización

Rubén Sánchez Corcuera

Director: Pablo García Bringas

Bilbao, mayo de 2017

Resumen

El proyecto consta de varios módulos. La base de todo es un servidor implementado en node.js. En él se aloja la página web, los widgets de las diferentes empresas que han decidido implantarlo en sus páginas web y una restful API que administra el enrutamiento de la web y permite varias funciones sobre la base de datos.

La página web sirve de apoyo al widget, es donde se muestra la información general del widget, los diferentes proyectos a los que apoyar, un wizard con el que poder crear tu propio widget y la plataforma para poder conseguir el certificado de donación.

La base de datos noSQL aloja los datos de las diferentes donaciones para posteriormente poder crear el certificado de donación y eventualmente generar estadísticas sobre las cantidades, fechas y lugares en los que más donaciones se están obteniendo.

Por último, el widget. Está pensado para que las empresas que tengan portales de compra online lo incrusten durante algunas de sus fases compra del cliente con el fin de que este haga una pequeña donación a la causa que las empresas han "apadrinado". El widget es personalizable por las empresas que lo quieran implantar en su web, así conseguir una mayor integración con ellas. Entre sus opciones de personalización destaca la posibilidad de elegir entre una cantidad fija o variable de donación.

Descriptores

solidaridad, widget, RESTful, NodeJS, MongoDB

Índice general

Índice de figuras

Índice de tablas

Índice de listados

1. INTRODUCCIÓN

1.1 Presentación del documento

El presente documento recoge proyecto desarrollado por el alumno Rubén Sánchez para la ONG ALBOAN. En el proyecto consta de todo el sistema necesario para crear un sistema de micro donaciones mediante un widget en cualquier tienda online. El sistema consta de: una página web, una base de datos, el widget y un servidor que da soporte a todo. Además, el sistema cuenta con una funcionalidad añadida por la que se puede crear gráficas de visualización.

El proyecto se desarrolla para la ONG Alboan la cual al ver el cambio que se da en la sociedad en materia de solidaridad y que las grandes donaciones han sufrido una gran caída decide que la manera de recoger las donaciones tiene que cambiar. Alboan, al explorar las oportunidades y ver que la única opción disponible cobra a las ONGs un porcentaje de las donaciones decide crear un sistema de micro donaciones gratuito que cualquier empresa pueda añadir a su tienda online y así fomentar las donaciones a pequeña escala, consiguiendo así que el 100 % del dinero vaya al destino final.

Los principales capítulos del documento son los siguientes:

- **Definición del proyecto**

  Establecimiento del objetivo fundamental del proyecto, especificando su alcance.

- **Producto final**

  Especificación de los elementos que componen el proyecto Colmena.

- **Organización**

  Definición del equipo de trabajo que desarrollará el proyecto y los perfiles profesionales que formaran parte de este. También incluye la estructura organizativa y el sistema utilizado para gestionar el proyecto.

- **Condiciones de ejecución**

  Definición del entorno de trabajo y del hardware y software a utilizar y de la metodología que se utilizará para hacer las modificaciones o mejoras que alteren el planteamiento inicial en el proyecto.

- **Planificación**

  Estimación de la duración de las tareas durante el transcurso del proyecto, así como su planificación en el tiempo.

- **Valoración económica**

  Determinación del valor correspondiente a este proyecto, las horas de desarrollo y las herramientas y elementos utilizados.

-

1.2 Motivación

Alboan ingreso el año 2015 nueve millones de euros. Este dinero proviene de donaciones tanto privadas como públicas. Las donaciones o ayudas públicas dependen del gobierno del año en la que las recibe, por lo que estas no tienen una solución posible ya que no dependen de nadie más que del gobierno. En cambio, las donaciones privadas, provenientes de personas o instituciones no públicas. Gracias a todas estas donaciones Alboan tiene 200 proyectos activos en 18 países diferentes, esto hace que mantener el dinero que la ONG invierte en cada uno de ellos sea crucial año tras año.

En el ejercicio de 2014 la financiación privada crece gracias a los legados solidarios[1] que la gente deja a la organización, también son muy importantes las cuotas de los socios que representan un 28 % de las aportaciones privadas. Estas donaciones son importantes para que Alboan cada vez pueda llegar a más gente y pueda crear más proyectos necesarios.

Alboan descubre entonces la necesidad de crear un nuevo canal por el que recibir donaciones y hacer que sus proyectos sean mas visibles de cara a la gente que no conoce tanto a Alboan. Entonces es cuando comienzan a gestarse los objetivos y requerimientos que la Colmena deberá cumplir.

EVOLUCIÓN de las APORTACIONES

	2015	2014	2013
PRIVADAS	5.001.937	5.142.652	4.941.067
PUBLICAS	3.567.609	2.965.197	2.566.145
Otros Ingresos	432.918	415.851	263.465
TOTAL Ingresos	9.002.464	8.523.700	7.770.677

Figura 1.1: Ejercicio de cuentas de Alboan

[1] Son las herencias que ciertas personas dejan a entidades solidarias con el fin de construir un mejor mundo.

2. OBJETIVOS DEL PROYECTO

En este capitulo se hablará de como se plantea el proyecto a la hora de ser desarrollado. Es capítulo en el que se marcan los objetivos y la visión que se tiene del proyecto antes de comenzar a desarrollarlo.

2.1 Demandante del proyecto

En esta sección se hablará el demandante del proyecto, en este caso es la ONG Alboan.

2.1.1 Alboan

La actividad de Alboan comenzó en 1994, asumiendo las iniciativas de voluntariado internacional que ya estaban en marcha, pero fue en 1996 cuando se configuró jurídicamente bajo la figura de Fundación. Es la ONG promovida por la Compañía de Jesús en la Provincia de Loyola (País Vasco y Navarra). El nombre, en euskera, Alboan, quiere reflejar el arraigo a la cultura de la tierra vasca y el espíritu de la entidad: estar al lado de las personas más excluidas, junto a organizaciones y centros educativos.

Su logo (Figura 2.1) visibiliza el rol de bisagra y puente que quería jugar la institución para poner en relación dos mundos que en realidad son uno solo. Su misión: ser plataforma de encuentro de personas y organizaciones de aquí y allá que quisieran comprometerse en la construcción de un mundo mejor.

Hoy Alboan cuenta con 7000 personas entre contratados, voluntarios y entidades que le permiten apoyar y acompañar a más de 200 proyectos llevados a cabo por 105 organizaciones aliadas que impactan directamente en las vidas de más de 600.000 personas en todo el mundo.

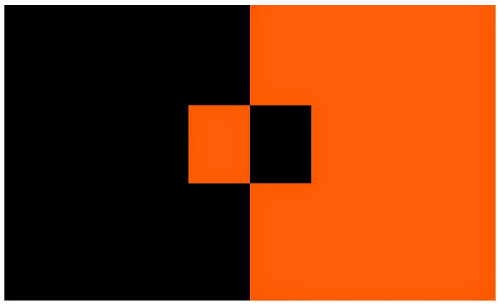

Figura 2.1: Logo de Alboan

Trabaja por la construcción de una ciudadanía global que denuncie las injusticias que provocan desigualdad en el mundo, construya una cultura que promueva el bien común y transforme las estructuras generadoras de pobreza a nivel local y global. Para lograrlo, se une en red con personas y grupos de todo el mundo.

La colaboración de Alboan se centra en 4 temáticas principales en las que enmarca todos sus proyectos y labores de ayuda: Educación de calidad, Desarrollo económico-productivo sostenible y equitativo, Acción humanitaria en crisis recurrentes y Democracia a favor de las personas excluidas.

Todas las actividades que realiza la ONG incluyen 3 ejes transversales que otorgan a los proyectos el carisma que Alboan quiere dar:

- La espiritualidad como dimensión en el horizonte de desarrollo humano.
- El reconocimiento de las desigualdades entre mujeres y hombres y el compromiso con la equidad de género.
- La participación ciudadana para la incidencia social y política.

2.2 Definición del proyecto

En esta sección se entrará a definir lo que va a ser el proyecto y los limites en los que se va a desarrollar.

2.2.1 Objetivos

El principal objetivo de este proyecto es crear un widget solidario que se pueda implantar en cualquier tienda online y crear la página web en la que se va a apoyar y se dará a conocer este widget. La página web también será el portal en el que la gente que ha realizado alguna donación pueda recoger su certificado de donación con el fin de presentarlo en la declaración de la renta.

Otro objetivo sería el de crear el servidor con la base de datos para alojar los datos de las donaciones realizadas mediante el widget. Este servidor seria accesible para las personas de Alboan que quisieran consultar los datos de las donaciones o ver las gráficas relacionadas con esto.

2.2.2 Alcance del proyecto

Teniendo en cuenta los objetivos citado anteriormente la Colmena deberá cumplir con las siguientes funcionalidades:

- Una página web responsiva que informe sobre todo del proyecto Colmena pero que también tenga información sobre la ONG y que tenga algún enlace para redirigir a esta.
- Un widget que permita hacer micro donaciones a los proyectos de Alboan y que informe sobre el proyecto al que va destinado el dinero.
- Una base de datos en la que almacenar las donaciones y los datos de los donantes.
- Una herramienta con la que poder dispensar certificados de donación a las personas que han aportado con los proyectos.
- Una herramienta para poder personalizar el widget y generarlo automáticamente.

- Un servidor en el que centralizar todas las funcionalidades y rutas de la solución. Además el servidor será capaz de ofrecer una API para aplicaciones futuras que quieran añadirse a la solución.
- Un sistema de visualización interactivo en el que poder analizar los datos obtenidos desde la aplicación y sacar conclusiones de ellos.

El proyecto no se encargará de hacer llegar el dinero de las donaciones, desde los donantes hasta la ONG, sino que hará de puente contabilizando y manteniendo un registro de estas para que finalmente el comercio online pueda hacer llegar este dinero a la entidad.

2.2.3 Producto final

El producto final se compone de varios elementos listados a continuación:

El elemento visual del proyecto lo formarían **la página web y el widget**. Estos dos elementos permitirán a los usuarios del sistema informarse de los proyectos que la ONG realiza, contactar con el soporte del proyecto o hacer uso de las funcionalidades que ofrecen, como por ejemplo recibir un certificado de donación o hacer una donación a uno de los proyectos. El widget puede estar alojado en cualquier comercio online. Estos dos elementos están estrictamente ligados a el servidor del proyecto que es el que les permite implementar todas sus funcionalidades.

El elemento central del proyecto esta formado por **el servidor y la base de datos**. Estos dos elementos permiten desarrollar toda la funcionalidad del sistema ya que ofrecen un sistema de enrutado para la parte visual y toda la funcionalidad que deban implementar sus elementos. Por otra parte, este elemento ofrecerá una API en la que se ofrecen diferentes funcionalidades como consultas a la base de datos, siempre con cierta privacidad hacia los datos sensibles.

El elemento final del proyecto consta del **sistema de visualización** de los datos. Gracias a este sistema se podrán visualizar los datos de manera interactiva y permitir a los empleados de Alboan hacer reflexiones sobre sus proyectos o los diferentes sectores de la población que les apoya en diferente medida. Por otra parte, también permitirá a los donantes ver como es la sociedad que les rodea y hacer un análisis de ella.

2.3 Descripción de realización

2.3.1 Método de desarrollo

El proyecto se ha desarrollado utilizando la metodología ágil Scrum. Esta metodología encaja con el proyecto por el desconocimiento de algunas de las herramientas a utilizar y porque algunos de los requisitos pueden ser cambiantes.

Scrum es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos.

En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, Scrum está especialmente indicado

para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

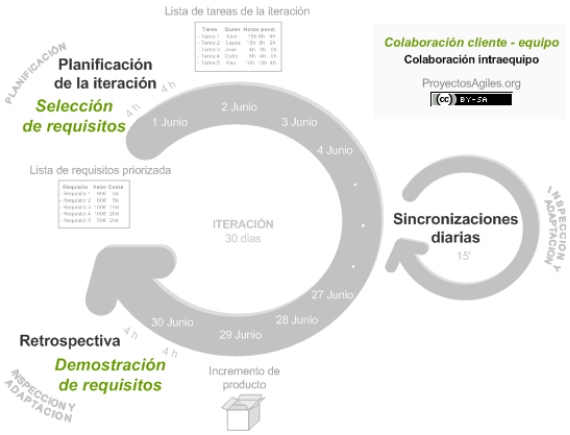

Figura 2.2: Proceso scrum

Las iteraciones o sprints será de 3 semanas cada uno. Al comienzo de los sprints se hará una reunión para pensar cuales deben ser las tareas que deben ir dentro de dicho sprint y al final del mismo se realizará la reunión de revisión en la que se revisará el trabajo realizado y las tareas pendientes. Estas reuniones tendrán unos productos intermedios, los cuales serán explicados más adelante.

Para apoyar la metodología elegida se utiliza Trello. Trello es una aplicación online que sirve para organizar tareas online en diferentes tableros. Estos tableros estan compuestos por diferentes columnas por las que ir moviendo las tareas dependiendo de su estado.

2.3.2 Tareas principales

A continuación se listarán las principales tareas del proyecto, estas posteriormente se desglosaran en tareas mas pequeñas que el equipo de desarrollo pueda desarrollarlas mas agilmente. Estas tareas serán valoradas posteriormente y asignadas con un indicador de prioridad.

- **Organización**
  Consiste en la planificación y preparación del proyecto. Al comienzo del proyecto habrá que definir los requisitos e historias de usuario y posteriormente en cada reunión al final del sprint y al principio del siguiente habrá que realizar de nuevo esta labor.
- **Seguimiento**
  Seguimiento y control del desarrollo del proyecto con la finalidad de detectar los posibles errores tanto en la lógica como en la parte visual del sistema. El principal seguimiento viene dado por las reuniones que se realizan y por el cumplimiento de las historias de usuario.
- **Análisis de tecnologías a usar**
  Esta tarea consiste en analizar las posibles tecnologias que puedan cumplir con las

funciones que se requieren en el proyecto. En esta tarea habrá que decidir cuales seran las tecnologias a utilizar para los diferentes elementos ya que el proyecto consta de varios y diferentes entre si. La decision tiene que estar basada en unos argumentos fundamentados. En principio no existe ningun requerimiento sobre las tecnologias por parte del demandante del proyecto.

- **Arquitectura del proyecto**

  Esta tarea consiste en definir la arquitectura del proyecto. Esto implica el diseño del proyecto en su totalidad y como va a ser la comunicacion entre las diferentes partes.

- **Creación de la parte servidora y la base de datos**

  Esta tarea consiste en crear la parte servidora del proyecto. En ella estará implementada toda la lógica del sistema y habrá que definir las conexiones con los diferentes elementos del sistema. En esta tarea, tambien, habra crear la base de datos y conectarla con el servidor para permitir la insercion y busqueda de los datos en ella.

- **Creación de la página web y el widget**

  Esta tarea consiste en la creacion y el diseño de la página web y el widget. En esta tarea principalmente se creará el diseño visual de ambos elementos y posteriormente se haran las conexiones con el servidor para permitirles implementar la parte lógica.

- **Creación del sistema de visualización de datos**

  Esta tarea consiste en la creacion de un sistema en el que poder hacer una visualizacion interactiva de los datos. Tambien habra que realizar el formateo de la informacion que llegue desde la base de datos, para que la visualizacion este correcta.

- **Cierre del proyecto**

  Esta tarea consiste en entregar a la ONG el sistema completo y prestarles apoyo en la implementacion del mismo. En el momento en el que Alboan haya implementado el sistema Colmena en su servidor, el proyecto estará terminado.

Finalmente se procede a listar las diferentes tareas, más especificas, que se van a llevar a cabo en los sprints marcados. Esta información es orientativa, ya que sabemos, que en la metodologia scrum los sprints pueden ser cambiantes.

- T1 - Sprint 1
  - T1.1 - Investigación sobre las tecnologías a utilizar
  - T1.2 - Redactar el documento sobre las tecnologías a utilizar
  - T1.3 - Instalar las herramientas de desarrollo
  - T1.4 - Formación en las diferentes tecnologías a utilizar
  - T1.5 - Diseñar la arquitectura del proyecto
- T2 - Sprint 2
  - T2.1 - Crear el primer diseño de la página web
  - T2.2 - Crear el primer diseño del widget
  - T2.3 - Desarrollar la lógica del servidor

T2.4 - Pruebas servidor

T2.5 - Configurar la base de datos

T2.6 - Pruebas base de datos

- T3 - Sprint 3

  T3.1 - Desarrollar la lógica de la página web

  T3.2 - Desarrollar la lógica del widget

  T3.3 - Consolidar el diseño de la página web

  T3.4 - Pruebas de la página web

  T3.5 - Consolidar el diseño del widget

  T3.6 - Pruebas del widget

  T3.7 - Crear manual de usuario para los técnicos de mantenimiento

- T4 - Sprint 4

  T4.1 - Investigar los frameworks de visualización de datos

  T4.2 - Configurar el frameworks de visualización de datos

  T4.3 - Pruebas sobre el frameworks de visualización de datos

  T4.4 - Analizar los datos

  T4.5 - Extraer las estadísticas de los datos

2.3.3 Productos intermedios

Gracias a la metodología que se va a utilizar es muy fácil generar productos intermedios. Durante el proceso de desarrollo del proyecto se crearán varios prototipos/mockups del widget y de la página web. Al final de cada sprint se hará una revisión del trabajo que se ha realizado, esto será posible gracias al tablón Trello en el que se podrán ver las tareas realizadas. Para definir los productos intermedios que se van a generar se procede a listarlos a continuación:

- Documento sobre el estudio de las tecnologías a utilizar y conclusiones.
- Documento de analisis sobre la arquitectura y especificaciones del proyecto.
- Varios prototipos de la página web
- Varios prototipos del widget
- Documento de seguimiento de las tareas realizadas
- Manual de usuario para los técnicos que mantendrán el proyecto
- Documento de conclusiones sobre las donaciones realizadas

Gracias a estos productos intermedios se podrá mantener un seguimiento del proyecto y tener una mejor estimación de las tareas y los tiempos necesarios para estas.

2.4 Organización y equipo

2.4.1 Esquema organizativo

- **Product Owner:** Este rol será representado por una persona de Alboan. Sus principales funciones son las siguientes:
  - Ser el representante de todas las personas interesadas en los resultados del proyecto
  - Definir los objetivos del producto o proyecto.
  - Colaborar con el equipo para planificar, revisar y dar detalle a los objetivos de cada iteración.
- **Scrum master:** No se debe confundir con el jefe de proyecto. Sus principales tareas son:
  - Facilitar las reuniones de Scrum
  - Proteger y aislar al equipo de interrupciones externas
  - Quitar los impedimentos que el equipo tiene en su camino
- **Equipo de desarrollo:** Compuesto por los profesionales que desarrollarán el proyecto. Su principal función es desarrollar el proyecto y estas son otras de sus funciones:
  - Equipo auto organizado.
  - Equipo multidisciplinar, capacidad de desarrollar diferentes tareas.
  - Equipo estable durante el proyecto y establecidos en la misma localización física.

En la figura 2.3 se puede ver cómo será la interlocución entre los miembros del equipo. El Scrum master, representado por Rubén Sánchez, será el encargado de interaccionar con el Product Owner, que estará representado por una persona de Alboan. Por último, el equipo de desarrollo incluirá a las personas que desarrollen el proyecto, las cuales podremos ver más adelante.

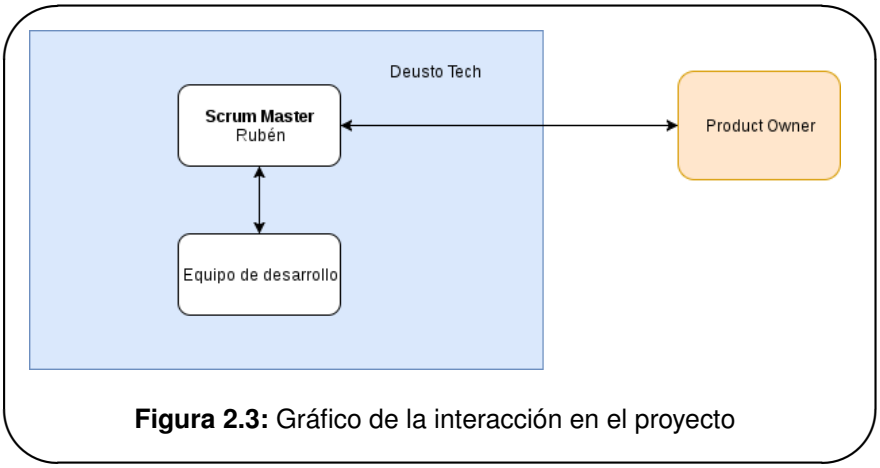

Figura 2.3: Gráfico de la interacción en el proyecto

2.4.2 Plan de recursos humanos

En este apartado se describirá al equipo de proyecto que será el encargado de desarrollar el producto y los diferentes perfiles necesarios para ello.

- **Jefe de proyecto:** Es el encargado de controlar que el proyecto se desarrolle correctamente y de garantizar los tiempos de desarrollo y la calidad del mismo. También se encargará de los aspectos de gestión del proyecto.

- **Diseñador:** Es el encargado de diseñar la interfaz de la página web y del widget.

- **Analista de datos:** El encargado de analizar el sistema en materia de datos, para otorgar una solución consistente y correcta. Por otra parte es el encargado de analizar los datos que se extraen del proyecto y de crear informes interactivos y conclusiones de ellos.

- **Arquitecto web:** Es el encargado de diseñar y desarrollar la arquitectura principal del proyecto. También se encarga de configurar las herramientas que vayan a utilizar los demás para asegurar su correcto funcionamiento.

- **Programador:** Es el encargado de desarrollar los métodos que el arquitecto haya establecido.

En la figura 2.4 se ve la organización en el equipo de proyecto. Alboan está como cliente externo a Deusto Tech. Dentro de Deusto Tech tenemos a Pablo García, el cual será el director del proyecto desde la empresa, y al equipo de la Colmena. Dentro del equipo de la Colmena se han separado los roles del Jefe de proyecto y del equipo de proyecto. Esto se ha hecho para ver la diferenciación en cuanto al plan de recursos humanos y para ver quien tiene la comunicación con el cliente, aunque en el proyecto los haya representado la misma persona.

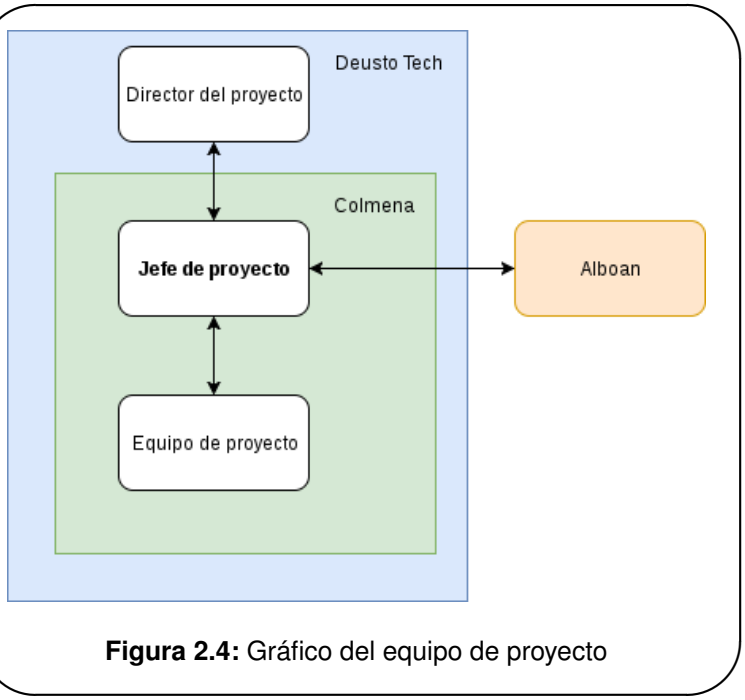

Figura 2.4: Gráfico del equipo de proyecto

2.5 Condiciones de ejecución

2.5.1 Entorno de trabajo

El desarrollo tendrá lugar en DeustoTech, sin un departamento de referencia.

El horario de trabajo será de 4 horas diarias.

DeustoTech y Alboan ofrecerán todos los medios necesarios para el buen desarrollo del proyecto. En caso de contar con un ordenador persona, se utilizará este.

■ Hardware

Este es el hardware que se utilizará para desarrollar el proyecto:

- Lenovo ThinkPad X220
- Monitor Acer AL1714 (Monitor secundario)

■ Software

Todo el software utilizado en el proyecto es gratuito. Este es el software utilizado para desarrollar el proyecto:

- Atom
- Git
- Brackets

2.5.2 Control de cambios

El seguimiento del proyecto se hará desde el tablón de Trello. En el tablón se registran los cambios que se generan y se generan notificaciones para las personas involucradas en la tarea que ha sufrido cambios. Finalmente esos cambios se tendrán en cuenta en las reuniones de final de sprint y se podrán comentar las diferentes opciones al cambio.

En cuanto a reuniones de seguimiento del proyecto, tendremos una reunión cada viernes en la que el equipo de proyecto hablará con el supervisor de Deusto Tech. En esa reunión se podrán ver los avances y problemas que hay cada semana y se pensarán diferentes soluciones para ellos. Estas reuniones también servirían por si se necesitará algo de Deusto Tech.

2.5.3 Recepción de productos

2.6 Planificación

2.6.1 Diagrama de precedencias

2.6.2 Plan de trabajo

2.6.3 Diagrama de Gantt

2.6.4 Estimación de cargas de trabajo

2.7 Presupuesto

3. ESPECIFICACIÓN DE REQUISITOS

3.1 Visión general

En este capitulo se especifican los requisitos que el proyecto debe satisfacer y que definen el funcionamiento de todo el software que compone este proyecto. Para una mejor compresión de los mismos, se dividen en los siguientes bloques:

- *Especificación de requisitos del servidor Node.js:* en esta sección se recogen los requisitos que debe de satisfacer el servidor, este es el encargado de soportar todo el sistema.
- *Especificación de requisitos de la página web:* en esta sección se recogen los requisitos que debe satisfacer la página web del proyecto.
- *Especificación de requisitos del widget:* en esta sección se recogen los requisitos que debe satisfacer el widget del proyecto, el elemento que estará incrustado en las tiendas online.
- *Especificación de requisitos de la base de datos:* en esta sección se recogen los requisitos que debe satisfacer la base de datos, esta albergará los datos de las donaciones y donantes.
- *Especificación de requisitos del sistema de visualización:* en esta sección se recogen los requisitos que debe satisfacer el sistema de visualización, el encargado de crear gráficas interactivas para la visualización y estudio de los datos.

3.2 Especificación de requisitos del servidor Node.js

Los requisitos funcionales del servidor Node.js son:

- **RF.0.1** El servidor debe ser capaz de albergar la página web.
- **RF.0.2** El servidor debe ser capaz de albergar las rutas de la página web y ofrecer el enrutamiento a cada una de estas.
- **RF.0.3** El servidor debe ser capaz de conectarse con la base de datos.
- **RF.0.4** El servidor debe ser capaz de enviar mails.
- **RF.0.5** El servidor debe ser capaz de alterar un PDF.
- **RF.0.6** El servidor debe ser capaz de crear un script y almacenarlo.
- **RF.0.7** El servidor debe ser capaz de ofrecer un script a páginas de terceros.
- **RF.0.8** El servidor debe ser capaz de ofrecer datos a una tercera aplicación.

Los requisitos no funcionales son:

- **RNF.0.1** Mantenibilidad: el sistema tiene que tener un mantenimiento sencillo ya que tiene conexiones con paginas de terceros lo que obliga a que el mantenimiento sea sencillo y rápido.
- **RNF.0.2** Escalabilidad: el servidor tiene que ser escalable ya que la creación de nuevos widgets o la oferta de datos a terceras aplicaciones puede ser grande.

3.3 Especificación de requisitos de la página web

Los requisitos funcionales de la página web son:

- **RF.1.1** La página web debe ofrecer un wizard para la creación de nuevos. widgets
- **RF.1.2** La página web debe ofrecer un formulario para la obtención de un certificado de donación.
- **RF.1.3** La página web debe ofrecer un sistema para comunicarse con el soporte del proyecto.

Los requisitos no funcionales de la página web son:

- **RNF.1.1** La página web debe informar sobre el proyecto.
- **RNF.1.2** La página web debe informar sobre los proyectos de la ONG.
- **RNF.1.3** Accesibilidad: La página web tiene que ser responsiva para ser adaptable en diferentes dispositivos.
- **RNF.1.4** Interfaz: la página web debe tener un diseño simple para facilitar la navegación.
- **RNF.1.5** Escalabilidad: la página tiene que ser escalable ya que se tienen que poder añadir nuevos proyectos a ella.

3.4 Especificación de requisitos del widget

Los requisitos funcionales del widget son:

- **RF.2.1** El widget debe permitir la donación fija o variable de una cantidad de dinero.
- **RF.2.2** El widget debe ser de fácil implantación por parte de la tienda online.

Los requisitos no funcionales del widget son:

- **RNF.2.1** El widget debe informar sobre el proyecto al que va destinado.
- **RNF.2.2** Accesibilidad: el widget debe ser responsivo para que pueda añadirse en cualquier tienda.
- **RNF.2.3** Interfaz: el widget debe ser 100 % personalizable.
- **RNF.2.4** Disponibilidad: el widget debe estar disponible en todo momento para su uso por parte de los comercios online.

3.5 Especificación de requisitos de la base de datos

Los requisitos funcionales de la base de datos son:

- **RF.3.1** La base de datos debe almacenar los datos de las donaciones.
- **RF.3.2** La base de datos debe almacenar los dados de los donantes.
- **RF.3.3** La base de datos debe proporcionar los datos que se le pidan.

Los requisitos no funcionales de la base de datos son:

- **RNF.3.1** Rendimiento: la base de datos debe almacenar y proporcionar los datos en un tiempo razonable.
- **RNF.3.2** Seguridad: la base de datos debe ser segura para no poner en peligro los datos de los donantes.
- **RNF.3.3** Disponibilidad: La base de datos debe estar disponible para almacenar las donaciones e información de donantes.

3.6 Especificación de requisitos del sistema de visualización

Los requisitos funcionales del sistema de visualización son:

- **RF.4.1** El sistema de visualización debe ofrecer gráficos interactivos.
- **RF.4.2** El sistema de visualización debe conectarse con el servidor para adquirir los datos.

Los requisitos no funcionales del sistema de visualización son:

- **RNF.4.1** Escalabilidad: el sistema de visualización de datos debe ser escalable ya que los datos pueden crecer y la manera de mostrarlos cambiar.
- **RNF.4.2** Interfaz: el sistema de visualización de datos debe tener una interfaz intuitiva que permita una buena navegación por los gráficos.

4. TECNOLOGÍAS UTILIZADAS

En esta sección se presentan las diferentes tecnologías que se han usado para el desarrollo del proyecto. Las principales tecnologías utilizadas son las siguientes:

4.1 Node.js

Esta es la autodefinición que se hace Node.js en su propia página web:

> Node.js[5] es un entorno de ejecución para JavaScript construido con el motor de JavaScript V8 de Chrome. Node.js usa un modelo de operaciones E/S sin bloqueo y orientado a eventos, que lo hace liviano y eficiente. El ecosistema de paquetes de Node.js, npm, es el ecosistema mas grande de librerías de código abierto en el mundo.

Node.js es una solución con un solo hilo de ejecución que permite que las peticiones a esta no bloqueen peticiones futuras ni exige grandes pools de hilos para tener conexiones concurrentes. Esto permite que los comercios onlines conecten con el servidor y le hagan una petición para recibir el widget a lo que el servidor responderá con el envió y el cierre de la conexión, evitando así el cuello de botella que se puede generar con conexiones masivas.

Node.js cuenta con un gestor de paquetes llamado npm del que se pueden descargar diferentes modulos para ampliar la funcionalidad de esta tecnología. Gracias a este gestor de paquetes node.js se convierte en una solución integral para la parte servidora habilitándole con todo lo necesario para cumplir las funciones del backend de una solución web.
Entre las ventajas que ofrece Node.js se encuentran las siguientes:

- **Gran documentación:** tiene una gran documentación y 8 años de experiencia por lo que la mayoría de los casos y posibilidades están testadas haciendo su desarrollo mas sencillo.
- **Gran comunidad:** gracias al gestor de paquetes publico y a los años de experiencia Nodejs cuenta con una gran comunidad con la que poder consultar las dudas y usos de los diferentes paquetes.
- **npm:** su gestor de paquetes publico permite reutilizar código y no perder tiempo implementando código que ya ha sido desarrollado y probado anteriormente.
- **Multiplataforma y open-source:** esta desarrollado para ser utilizado en cualquier sistema operativo y cuenta con una licencia MIT, lo cual lo hace gratuito y permite arreglar e incluso mejorar el propio código de la herramienta por los usuarios de esta.

4.1.1 Funcionamiento

En Node.js es muy sencillo crear aplicaciones nuevas que actuen en la parte servidora de una aplicación. Gracias a npm, Node.js es muy polivalente, pero a continuación se mostrará un ejemplo de un servidor HTTP:

```
http.createServer(function (request, response) {
    response.writeHead(200, {'Content-Type': 'text/plain'});
    response.end('Hello World');
}).listen(8081);
console.log('Server running at http://127.0.0.1:8081/');
```

Listado 4.1: Ejemplo de un servidor HTTP en Node.js

En este ejemplo se crea un servidor HTTP en el puerto 8081 de la maquina local. Una vez se entra al puerto 8081 de la maquina local, el servidor creara una respuesta en texto plano y la enviará al navegador, que mostrará por pantalla el mensaje.

Node.js funciona de una manera muy diferente dependiendo de los paquetes utilizados para el desarrollo de la aplicación por lo que a continuación explicare los paquetes utilizados para el desarrollo de este proyecto y como es el funcionamiento de cada uno de estos:

4.1.2 Express

Express es un framework web minimalista y flexible para el desarrollo de soluciones web en node.js. Express además aplica una fina capa con las características fundamentales de las aplicaciones web sobre la base de node.js permitiendo así mantener todas las funcionalidades que ofrece Node.js. Finalmente, Express ofrece una amplia y robusta API para hacer uso de todas las funcionalidades y características que ofrece.

■ Funcionamiento

Express no tiene una estructura de proyecto definida por los desarrolladores o comunidad, en cambio, tanto en la documentación como en numerosos sitios ofrecen un sistema de carpetas para tener el proyecto ordenado y las rutas definidas.

```
project/
|-- node_modules/
|-- public/
    |-- images/
    |-- css/
    |-- javascript/
|-- routes/
|-- views/
|-- app.js
|-- package.json
```

Listado 4.2: Estructura de paquetes tipo de un proyecto express

- **Node_modules:** en esta carpeta irán incluidas todas las dependencias del proyecto una vez descargadas automáticamente por npm.
- **Public:** en esta carpeta estarán los elementos de uso publico por parte del proyecto, imágenes, scripts de JavaScript y hojas de estilo.
- **Routes:** aquí se encuentran las rutas a las diferentes entidades en archivos diferentes.
- **Views:** en esta carpeta estarán incluidas las vistas del proyecto, es decir, las diferentes pantallas de la página web.

- **App.js:** es el archivo principal del proyecto.
- **Package.json:** es un archivo de configuración general del proyecto, en el irán las dependencias y los metadatos del proyecto.

■ Razón de uso

Se ha elegido este framework ante otros disponibles para Node.js por varias razones:

- El primer commit de esta herramienta fue 2 meses después de la creación de Node.js y la primera versión 1 año después por lo que tiene un gran recorrido y madurez.
- Su sencillez para manejar las rutas y las vistas en una solución web.
- Gran comunidad de desarrolladores en la que apoyarse.

4.1.3 Express-ejs-layouts

4.1.4 Nodemailer

Nodemailer es un módulo para Node.js que permite enviar emails. Este módulo esta securizado de manera que los mensajes que envía los hace de manera segura. Permite adjuntar elementos a los mensajes y no depende de ningún otro paquete, lo que le convierte en un paquete muy completo.

■ Funcionamiento

Nodemailer es muy sencillo de usar, solo hay que definir el mensaje con los campos y archivos adjuntos que quieras añadir y también habrá que definir el transporter que sera el encargado de enviar el mensaje. En este caso no esta añadido la configuración del transporter.

```
var message = {
    from: 'sender@server.com',
    to: 'receiver@sender.com',
    subject: 'Message title',
    text: 'Plaintext version of the message',
    html: '<p>HTML version of the message</p>'
};
transporter.sendMail(data);
```

Listado 4.3: Estructura de un mensaje en Nodemailer

■ Razón de uso

Se ha utilizado este paquete para enviar los mails por su sencillez, el gran recorrido que tiene en esta materia, ya que fue creado en 2010 y por su completa documentación.

4.1.5 Mongojs

Mongojs es un paquete para Node.js que permite la conexión con bases de datos MongoDB. Este paquete intenta emular completamente la comunicación directa con la base de datos por lo que su API se asemeja mucho a la de MongoDB.

■ Funcionamiento

Mongojs es muy sencillo de usar, con la siguiente linea de código, conectaremos con una base de datos MongoDB que este en un servidor remoto y si añadimos la variable mycollection, conectaremos directamente con la colección dentro de la base de datos elegida.

```
var db = mongojs('example.com/mydb', ['mycollection'])
```

Listado 4.4: Ejemplo de conexion con una base de datos MongoDB

A continuación se muestra un ejemplo en el que se busca en la base de datos todos los documentos en los que name=Jhon. Esta función devuelve los documentos que cumplan la condición y posteriormente se pueden tratar dentro de la función.

```
 db.mycollection.find(\{ 'Name':'Jhon'\}function (err, docs) {
    // Insert code here
});
```

Listado 4.5: Ejemplo de una busqueda con mongojs

■ Razón de uso

Se ha utilizado este paquete para conectar con la base de datos por la facilidad con la que conecta con la base de datos y porque emula lo máximo posible la API de mongoDB haciendo su uso muy fácil si ya has utilizado mongoDB con anterioridad.

4.1.6 Pdffiller

Pdffiller es un paquete para Node.js que permite rellenar los formularios de los PDF's con datos. Su uso se basa en recibir PDF's con formularios sin rellenar, unos datos y combinarlos de manera que la salida sea un PDF completo.

■ Razón de uso

Se ha utilizado este paquete por ser el mas utilizado por los usuarios entre las posibilidades.

4.2 MongoDB

MongoDB es la base de datos NoSQL líder y permite a las empresas ser más ágiles y escalables. Organizaciones de todos los tamaños están usando MongoDB para crear nuevos tipos de aplicaciones, mejorar la experiencia del cliente, acelerar el tiempo de comercialización y reducir costes.

Es una base de datos muy ágil por lo que permite cambiar los esquemas al cambiar los requisitos y a la vez proporciona las mismas funcionalidades que se esperan de una base de datos tradicional, manteniendo la velocidad en las búsquedas y siendo consistente.

Gracias a ser una base de datos orientada a documentos no hay que ajustarse a un esquema estándar ni obligar a todos los registros a tener la misma información. Esto nos da mucha flexibilidad a la hora de querer introducir nuevos datos o alterar los que ya tenemos.

La base de datos del proyecto ha sido poblada mediante JSON, un formato de texto para el intercambio de datos del que hablaremos a continuación.

4.2.1 Razón de uso

Se ha utilizado esta base de datos para el proyecto por las siguientes razones:

- Gracias a su flexibilidad permite alojar diferentes datos en cada documento, permitiendo así ir modificándolos a medida que se van transformando mientras se mantienen unificados.
- Gran soporte para proyectos realizados con Node.js y sobre todo con express, lo que hace su integración muy sencilla.
- El formato de intercambio de datos que se va a utilizar a lo largo de todo el proyecto será el JSON.
- Los limites que oferta la base de datos en cuanto a los documentos encajan con los requisitos del proyecto.
- Gratuita y open-source.

4.3 JSON

Json (JavaScript Object Notation - Notación de Objetos de JavaScript) es un formato ligero de intercambio de datos. Leerlo y escribirlo es simple para humanos, mientras que para las máquinas es simple interpretarlo y generarlo.

4.3.1 Funcionamiento

Un objeto es un conjunto desordenado de pares nombre/valor. Los objetos comienzan con la llave (llave apertura) y terminan con la llave (llave de cierre). Cada nombre es seguido por : (dos puntos) y los pares nombre/valor están separados por , (coma).

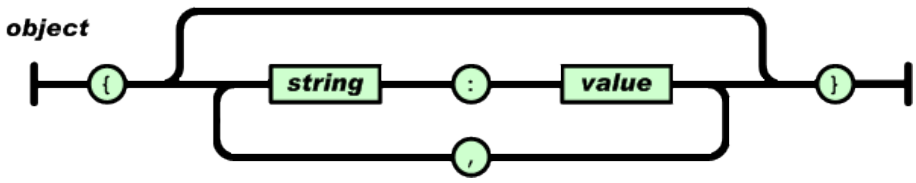

Figura 4.1: Diagrama de la gramática de un objeto JSON

En este ejemplo podemos ver como existe un 2 pares nombre/valor normales y uno que tiene un array de valores dentro.

4.4 Sass (CSS3)

La definición de Sass y CSS3 integrados es la siguiente:

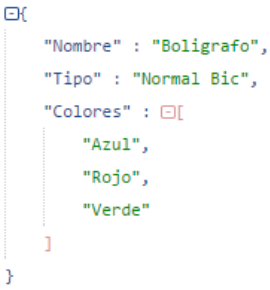

Figura 4.2: Ejemplo de un JSON

Sass es un metalenguaje de Hojas de Estilo en Cascada (CSS). Es un lenguaje de script que es traducido a CSS. Sass consiste en dos sintaxis. La sintaxis más reciente, SCSS, usa el formato de bloques como CSS. Éste usa llaves para denotar bloques de código y punto y coma (;) para separar las líneas dentro de un bloque.

CSS3 consiste en una serie de selectores y pseudo-selectores que agrupan las reglas que son aplicadas. Sass (en el amplio contexto de ambas sintaxis) extiende de CSS proveyendo de varios mecanismos que están presentes en los lenguajes de programación tradicionales, particularmente lenguajes orientados a objetos, pero éste no está disponible para CSS3 como tal. Cuando SassScript se interpreta, éste crea bloques de reglas CSS para varios selectores que están definidos en el fichero SASS. El intérprete de SASS traduce SassScript en CSS. Alternativamente, Sass puede monitorear los ficheros .sass o .scss y convertirlos en un fichero .css de salida cada vez que el fichero .sass o .scss es guardado. Sass es simplemente azúcar sintáctica para escribir CSS.

Las ventajas que ofrece este metalenguaje son las siguientes:

- **Mixins:** Ofrece la posibilidad de crear bloques de estilo o mixins que se apliquen a mas de una etiqueta o clase, asi, al compilar el código, el estilo se replica a todas las etiquetas y clases aligerando el trabajo.
- **Argumentos:** permite utilizar argumentos para unificar la definición de valores.
- **Herencia:** permite definir una herencia para que los hijos implementen el estilo de los padres.

4.4.1 Funcionamiento

En los siguientes listados podemos ver las diferencias y el uso de SCSS. Lo primero que podemos ver son las variables que declara, posteriormente, en la primera clase, vemos que no tiene corchetes ni las definiciones acaban con punto y coma. Tambien podemos ver como utiliza el metodo *darken* que oscurece el color. Por ultimo podemos ver como divide el margen definido entre 2 para que se reduzca en la segunda clase.

```
$blue: #3bbfce
$margin: 16px

.content-navigation
    border-color: $blue
    color: darken($bue, 9%)

.border
    padding: $margin/2
    margin: $margin/2
    border-color: $blue
```

Listado 4.6: Código SCSS

```
.content-navigation {
    border-color: #3bbfce;
    color: #2b9eab;
}

.border {
    padding: 8px;
    margin: 8px;
    border-color: #3bbfce;
}
```

Listado 4.7: Código CSS compilado

4.4.2 Razón de uso

Se ha utilizado este metalenguaje en vez de su competidor más claro, less, por las siguientes razones:

- Sass genera un código más optimo en algunos de las características que ambos metalenguajes comparten.
- Sass tiene varios años más que Less por lo que le hace más robusto y con una mayor comunidad de desarrolladores.

Y se usa este metalenguaje en vez de usar CSS directamente por las siguientes razones:

- Permite un desarrollo más ágil de las hojas de estilo del proyecto.
- Innovación: permite aprender otra herramienta que agilice el desarrollo del CSS.

4.5 D3.js

D3.js (o simplemente D3 por las siglas de Data-Driven Documents) es una librería de JavaScript para producir, a partir de datos, infogramas dinámicos e interactivos en navegadores web. En contraste con muchas otras librerías, D3.js permite tener control completo sobre el resultado visual final.

Gracias a esta librería se pueden crear gráficos interactivos en los que se unifiquen datos que anteriormente se mostrarían en varios gráficos diferentes, creando así una experiencia

más inmersiva en cuanto a la visualización de los datos.

Las ventajas principales que ofrece esta librería son:

- **No añade carga a la página:** utiliza lenguajes ya existentes y que no suponen una carga mas para la página web en la que se aloja.
- **Amplias funcionalidades:** permite seleccionar diferentes elementos en una página web y alterarlos. También incluye transiciones para generar cambios visuales, muy similar a JQuery.
- **Asociación de datos:** los datos pueden dirigir la creación de los elementos permitiendo asi a los datos gobernar la visualización y crear diferentes gráficos dependiendo del dataset introducido.

4.5.1 Razón de uso

Se ha utilizado esta librería de visualización frente a otras por las siguientes razones:

- Innovación: existe un creciente interés de desarrolladores por la herramienta y a su vez falta de programadores con conocimientos sobre ella.
- Gran integración con JSON y las herramientas utilizadas en el proyecto.
- Open-source y gratuita.

5. ESPECIFICACIÓN DEL DISEÑO

5.1 Visión general

Este capitulo tiene como objetivo describir la labor de diseño realizada para desarrollar el proyecto, así como las herramientas que se han utilizado para realizar estos diseños. En el siguiente listado se describen los diferentes diseños que se van a explicar en este capitulo:

- Diseño de la arquitectura: descripción de la arquitectura elegida para el proyecto.
- Diseño del servidor: descripción del diseño del servidor.
- Diseño de la página web: descripción del diseño de la página web, tanto lógica como visual.
- Diseño del widget: descripción del diseño del widget.
- Diseño de la base de datos: descripción del diseño de la base de datos y de su estructura.
- Diseño del sistema de visualización: descripción del diseño del sistema de visualización y de los gráficos incluidos en este.

5.2 Herramientas utilizadas

Las herramientas que se han utilizado a la hora de diseñar el proyecto y sus elementos son las siguientes:

5.2.1 Draw.io

Draw.io es una herramienta web que se utiliza para el desarrollo de diagramas de cualquier tipo, lo cual la hace una herramienta muy potente. Ofrece una gran cantidad de iconos y personalizacion de los mismos para realizar los diagramas lo mas atractivos posibles. Entre sus caracteristicas destaca la integracion con sistemas de almacenamiento online como Google Drive o Dropbox y Github.

En lo respectivo a este proyecto, solo se ha utilizado para diseñar los diagramas: AÑADIR DIAGRAMAS.

5.2.2 gomockingbird

Aplicación para mockups.

5.3 Diseño de la arquitectura

La arquitectura del proyecto (ver figura 9.1) se basa en un modelo en tres capas que gira en torno a el servidor Node.js. Estas tres capas permiten separar la parte visual, la lógica y la arquitectura de datos. Gracias a este modelo se puede centralizar la lógica del proyecto en la capa intermedia y abstraerla de los elementos externos a los que ofrece soporte.

En la capa de presentación se encuentran la página web y el widget. Estos dos elementos dan soporte a la parte visual del proyecto y están conectados con el servidor para que este les proporcione funcionalidad. En esta capa también se encuentra el sistema de visualización de datos del proyecto, este recibe los datos de la base de datos por medio del servidor que se los provee formateados.

En la capa intermedia, la de proceso, se encuentra el servidor que implementa la funcionalidad de toda la solución. Este alberga los widgets que los comercios online van a implementar en sus páginas web y les da la funcionalidad para permitir las donaciones. A la página web le provee del enrutado necesario para implementar sus funcionalidades principales, como la de obtener un certificado de donación o el contacto con el soporte del proyecto. En cuanto al sistema de visualización le otorga los datos, con el formato que necesita, para generar los gráficos deseados. Por ultimo, esta conectado también con la capa de datos, en la que se encuentra la base de datos, con la cual conecta para enviar y pedir datos.

Finalmente, en la tercera capa, la de datos, se encuentra el sistema de base de datos. La base de datos, conectada con el servidor, almacena los datos que le llegan desde este y realiza las búsquedas y peticiones que el servidor le pide. Todo esto lo hace mediante el lenguaje de intercambio de datos JSON del que hemos hablado anteriormente.

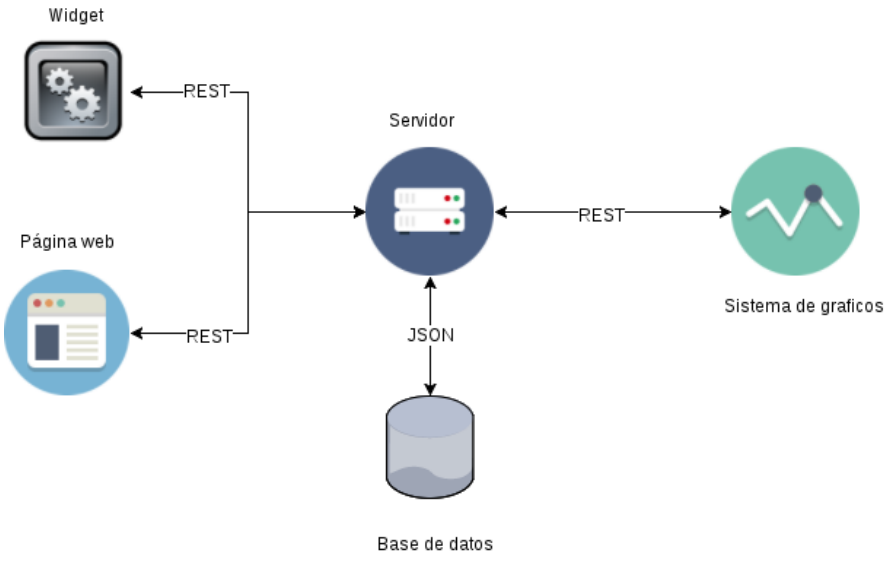

Figura 5.1: Arquitectura del proyecto

5.4 Diseño del servidor

El diseño del servidor se ha hecho siguiendo el modelo en el que el servidor Node.js con el apoyo de Express ofrece un API a las diferentes aplicaciones que están conectadas con el. En esta API el servidor ofrece toda la funcionalidad que estas aplicaciones tienen que desarrollar, por demanda de estas, y ademas puede ofrecer mas funcionalidades que el proyecto pueda ofrecer a aplicaciones o elementos que puedan incorporarse en el futuro del proyecto.

Gracias a este modelo es muy sencillo añadir nuevas rutas y funcionalidades que las aplicaciones, nuevas o ya existentes, demanden del servidor. Además esta API mantendrá una conexión con la base de datos lo que permitirá el envio, consulta y almacenamiento de información de una manera muy veloz.

El servidor estará organizado de manera que en la carpeta *public* estarán alojados todos los archivos comunes incluyendo los scripts de JavaScript que sean necesarios para el desarrollo de las aplicaciones y las librerías correspondientes entre otras cosas

Por último cabe destacar que en el diseño creado para el servidor todo gira alrededor del archivo en el que esta desarrollada la API, aunque la creación del servidor en el puerto especificado y la configuración de red se encuentre en el archivo original del servidor.

HACER DIAGRAMA DEL SERVIDOR

5.5 Diseño de la página web

El diseño de la página web se ha realizado basándose en las convenciones de Bootstrap para que la web sea totalmente responsiva y visualizable desde la gran mayoría de los navegadores. Gracias al diseño con este framework se podrán añadir los componentes especificos que ofrece e implementar utilidades que sin este framework serian mas complejas de desarrollar.

La página web se ha diseñado de manera que toda la información y las funcionalidades disponibles estén a la vista. Por eso se ha planteado una sola vista en la que por medio de algunos botones se desplieguen nuevas ventanas en las que poder añadir alguna funcionalidad o datos que de otra manera alargarían la página de manera negativa. Por otra parte se ha diseñado un menú superior que acompañará al usuario en todo momento para ofrecerle accesos directos a las diferentes secciones de la página web sin tener que ir hasta ella.

La página se ha planteado muy minimalista en cuanto a contenido, siguiendo siempre el estilo y colores de Alboan. Se plantea de esta manera para hacer que esta no eclipse a la página principal de Alboan, sino que sea algo complementario que informe mas sobre el proyecto Colmena que los proyectos de la ONG. En cuanto al diseño estético se plantean secciones acotadas en las que se desarrollará una única funcionalidad o se expondrá un tema informativo.

En cuanto al diseño de la lógica de la página web, este esta preparado para que la página web no tenga que incluir ningún código JavaScript que no sea estrictamente estético. Toda la lógica de la página web ira integrada en la API que ofrece el servidor.

En conclusión, en cuanto a diseño estético la página es altamente escalable ya que solo haría falta añadir una nueva sección. La lógica tampoco supone ningún problema ya que el servicio que la página web quiera dar deberá estar implementado en la API por lo que en la página web solo se deberá desarrollar la llamada a este.

AÑADIR DISEÑO PAGINA WEB

5.6 Diseño del widget

El diseño del widget se ha hecho de la manera mas simple posible, utilizando solo tecnologías que todos los navegadores puedan soportar, estas tecnologías son HTML, CSS y JavaScript.

Dentro de la lógica del widget se encuentran solo las funciones básicas que el widget tiene que realizar. En este caso se ha pensado en desarrollar solo 3 funciones, las dos primeras se encargarán de hacer el cambio dinámico del dinero en el widget, por lo que no son funcionales, solo visuales. La tercera función si que será la encargada de implementar toda la logica del widget. Esta sera la encargada de enviar la donación al servidor con todos los datos que se necesitan.

En cuanto a la estética del widget este se ha diseñado para que sea completamente personalizable, de esta manera, los comercios online puedan diseñar sus propios widgets y los hagan aptos para sus páginas. Para poder mantener esta política de diseño y combinarla con la tecnologías básicas, se ha planteado que el diseño del widget sea guiado mediante un wizard.

5.7 Diseño de la base de datos

La base de datos se ha diseñado partiendo de la premisa de que esta tenia que ser implementada en mongoDB. Una vez analizado los datos que iba a tener que almacenar y viendo los requisitos y necesidades que el proyecto presentaba se ha decidido crear una única base de datos que almacene los documentos en una sola colección.

En cuanto a los documentos que la base de datos almacena, estos solo tendrán dos formatos, el primero será el que tienen cuando llegan del widget de donaciones y posteriormente se les añaden los datos de la persona que dona. Por este motivo se diseña la base de datos con una sola colección en la que se alojan los datos de dos formas diferentes.

Gracias a este diseño podemos simplificar las conexiones con la base de datos desde el servidor ya que solo tendrá que hacerse a una colección. Por otra parte al solo tener una colección podremos unificar las búsquedas que se hagan en la base de datos. Finalmente al no tener documentos con excesivo tamaño podemos almacenar todos los datos en una sola colección y no corremos el riesgo de que esta se desborde ya que no tiene limite de documentos.

5.7.1 Colección

La colección se llamará 'Donacionesén la que se albergaran los documentos en los dos formatos que se han descrito anteriormente. Estos documentos tienen los siguientes datos y formatos:

- **Fecha:** la fecha de la donación con dia, mes y año. Esta dividida en un array de tres entero.
- **Usada:** un booleano de si la donacion ha sido usada o no.
- **IdDonación:** el id de la donación en un entero.
- **Importe:** el importe de la donacion en un entero.

Una vez la donación se usa para obtener el certificado de donación, se le añaden al documento en la base de datos los siguientes datos:

- **DNI:** el DNI o CIF de la persona juridica o fisica en un entero.
- **Nombre:** el nombre de la persona juridica o fisica en un String.
- **Razón social:** la razon social de la persona juridica o fisica en un String.
- **Dirección:** la direccion de la persona juridica o fisica en un String.
- **CodigoPostal:** el código postal de la persona juridica o fisica en un entero.
- **Población:** la poblacion de la persona juridica o fisica en un entero.
- **Provincia:** la provincia de la persona juridica o fisica en un entero.

5.8 Diseño del sistema de visualización

En el diseño del sistema de visualización se ha tenido en cuenta los datos que se podían conseguir y de que manera representarlos. Teniendo en cuenta el diseño de la base de datos y los datos que se van a almacenar en esta se ha decidido implementar varios gráficos en los que se puede mostrar la mayoría de los datos alojados en la base de datos.

Los diseños elegidos para mostrar estos datos serian los siguientes:

- **Sunburst o diagrama circular:** en este gráfico (ver figura 5.2) se muestra un diagrama circular en el que se van clusterizando los datos a medida que vas adentrándote en el. Gracias a este gráfico se pueden analizar sectores mas pequeños de las donaciones y ver la agrupación de diferentes sectores de donantes.
- **Fechas:** en este gráfico (ver figura 5.3) se muestra un calendario en el que se marcan los días con diferentes colores. Gracias a este gráfico se pueden ver los periodos en los que mas y menos se dona.
- **Mapa:** en este gráfico se muestra un mapa de España en el que se marcan las zonas en las que ha habido donaciones. Gracias a este gráfico es muy fácil visualizar como es cada zona y que las personas vean en que tipo de comunidad viven.

El sistema de visualización de datos se ha planteado para que ejerza la minima carga al sistema, por lo que se plantea con una librería de visualización para la creación de infogramas interactivos y archivos JSON para poblarla, de manera que la tarea de formatear los datos para integrarlos en la librería será mínima.

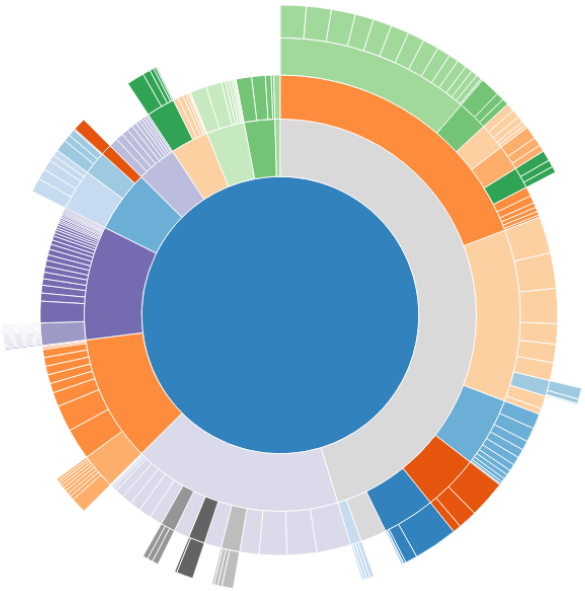

Figura 5.2: Diagrama circular

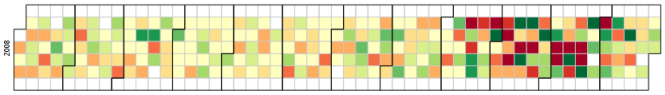

Figura 5.3: Diagrama de fechas

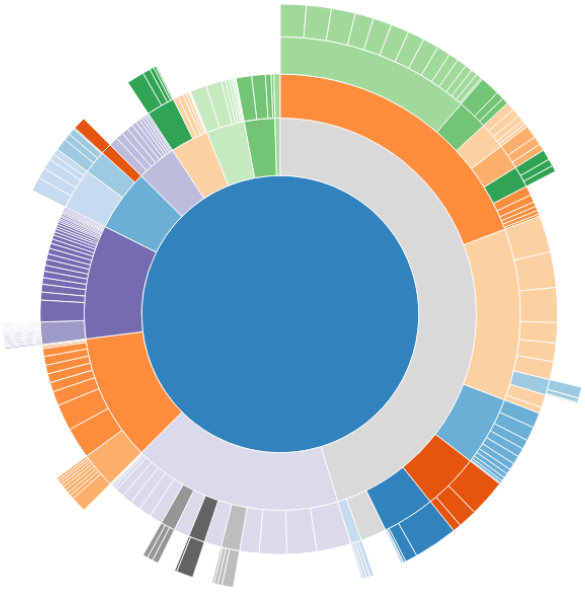

Figura 5.4: Diagrama circular

6. CONSIDERACIONES SOBRE LA IMPLEMENTACIÓN

6.1 Visión general

Este capítulo contiene los aspectos más significativos sobre la implementación del sistema y sus funcionalidades.El contenido central de este capitulo se centra en explicar que pasos se han dado a la hora de desarrollar el sistema y como se ha organizado este, simplificando la labor de comprensión del sistema para terceros. Por último, al principio del capitulo se explicarán las herramientas que se han utilizado a la hora de desarrollar el sistema.

6.2 Entorno de desarrollo

Las herramientas que se han utilizado para facilitar el desarrollo y la arquitectura del proyecto han sido las siguientes.

6.2.1 Atom

Atom se define a si mismo como "el editor de textos hackable del siglo XXI". Dentro de este editor de texto o entorno de desarrollo integrado (IDE), ya que cumple todos los requisitos de uno de estos, se encuentran una serie de funcionalidades y características que lo hacen una de las herramientas mas potentes del mercado. Dentro de las funcionalidades que ofrece esta herramienta se encuentran los paquetes que la comunidad crea y desarrolla, gracias a estos paquetes el programa a conseguido implementar la corrección y sintaxis de muchos lenguajes. Por otra parte, este editor de textos es totalmente personalizable ya que esta desarrollado con tecnologias web y ofrece una IU totalmente modificable mediante CSS/Less. Por ultimo, y no menos importante, Atom esta desarrollado por GitHub, por lo que es open-source y ofrece la posibilidad de ayudar en el desarrollo del código de la aplicación.

6.2.2 Brackets

Brackets es un editor de texto moderno enfocado en el diseño de la parte visual de las aplicaciones web. La funcionalidad mas destacada de este software es la vista previa que permite al desarrollador ver en directo como quedarán los cambios que realice en el código, directamente en la página web, esto ahorra tiempo al no tener que recargar la web constantemente.Brackets también ofrece un editor interno que permite navegar por todos los archivos CSS que una etiqueta implemente, lo que aligera la carga de estar cambiando entre las diferentes hojas de estilo. Por ultimo, entre sus funcionalidades tambien destaca el soporte a los preprocesadores, brackets permite hacer cambios directamente en los archivos de las hojas de estilo de los preprocesadores y ver esos cambios directamente, sin tener que compilarlos.

6.2.3 Git

Git es un sistema de control de versiones distribuido gratuito y de código abierto que está diseñado para manejar desde pequeños proyectos particulares a proyectos de grandes organizaciones con una gran velocidad y eficiencia. Además es muy fácil de aprender, tiene una excelente documentación y al ser usado por muchísimos desarrolladores tiene una gran comunidad de usuarios dispuestos a resolver cualquier problema.

6.3 Implementación del servidor

El servidor esta implementado en Node.js con la ayuda de varios paquetes creados por la comunidad. El servidor esta diferenciado en dos archivos diferentes. El archivo *server.js* contiene la configuracion de los paquetes que va a utilizar la aplicacion y el puerto en el que se va a desplegar. El archivo que contiene toda la funcionalidad del sistema es *api.js*, en este archivo estan definidas las rutas y las funciones del proyecto.

Lo primero en lo que fijarse al entrar en el archivo *server.js* es en los paquetes que tiene como requerimientos y los que utiliza. Entre los paquetes y usos mas destacables esta el requerimiento del otro archivo mas importante del proyecto, *api.js*, el uso del framework express y la carga de EJS como el motor visual del proyecto. Por otra parte, cabe destacar el puntero (Listado) que se crea para indicar al sistema donde van a estar alojados todos los recursos públicos del sistema.

```
app.use(express.static(path.join(__dirname, 'public')));
```

Listado 6.1: Puntero a la carpeta public

En el archivo *api.js* se encuentra toda la funcionalidad del servidor separada por las rutas que este ofrece a la página web. En esta seccion se hablará de como se han implementado las diferentes funcionalidades.

6.3.1 Como añadir una nueva ruta

Para crear una nueva ruta dentro del servidor solo habria que añadir una nueva respuesta a una solicitud de la parte del cliente (Listado). En el direccionamiento existen 4 tipos de solicitudes desde la parte servidora, estas serian: GET, POST, PUT y DELETE. Posteriormente habria que indicar a que ruta harian esta solicitud, en el codigo marcado con /RUTA. Finalmente se entraria en la funcion en la que se desarrollaria todo el codigo que el programador haya especificado.

```
router.get/post/put/delete('/RUTA', function(req, res) {
    ...
});
```

Listado 6.2: Ejemplo de ruta con Express

6.3.2 Conexión con la base de datos

Lo primero a destacar es la conexión con la base de datos, esta esta implementada con mediante el paquete mongojs. La conexión es muy sencilla (Listado) y al solo tener una

colección nos conectamos directamente a ella. La base de datos al ser MongoDB se despliega en el puerto 27017, en caso de estar desplegada solo habria que cambiar ese dato. El nombre de la base de datos es colmena y la coleccion donaciones.

```
const db = mongojs('mongodb://IPSERVIDOR:27017/colmena', ['donaciones']);
```

Listado 6.3: Conexión a la base de datos mediante mongojs

6.3.3 Configuracion de Nodemailer

Nodemailer[?], es el paquete que nos permite enviar los mails a las personas que donan. La configuración de este paquete es muy sencilla y nos permite elegir servicio con el que vamos a enviar los mails, en este caso seria Gmail y solo habria que añadir el mail y la contraseña de la cuenta que queramos utilizar.

```
var transporter = nodemailer.createTransport({
    service: 'Gmail',
    auth: {
        user: 'colmenadeusto@gmail.com',
        pass: '****'
    }
});
```

Listado 6.4: Configuracion de Nodemailer

Las funcionalidades que ofrece el servidor son REST, es decir, esperan a que una peticion entrante para reaccionar y cuando acaban cortan la comunicacion.

6.3.4 Codificacion de la función de recepción de donaciones

El sistema sistema de recepcion de donaciones esta codificado de manera que al recibir una solicitud de donacion desde el widget, la funcion, mediante un algoritmo, calcula un id de donacion unico para cada usuario. Posteriormente empaqueta este numero y lo envia a la base de datos junto con los datos de la donación. Por ultimo genera un email con el id de donacion y se lo envia a la persona que ha realizado la compra.

El algoritmo que genera los numeros de donación se ha creado utilizando un metodo en el que es practicamente imposible generar un numero igual. El algortimo comienza capturando el tiempo en el que llega la peticion, por lo que este numero solo puede ser igual si la peticion llega en el mismo momento exacto. El numero que se genera tiene 13 cifras. Posteriormente se genera un numero aleatorio entre 0 y 100 y se suma al número obtenido anteriormente. Las posibilidades de generar un número igual solo serian significativas, un 1 %, si la peticion llegase exactamente en el mismo momento.

```
  var id = new Date().getTime() + Math.floor(Math.random() * 101);
```

Listado 6.5: Algoritmo de generacion de numeros de donacion

6.3.5 Codificacion de la creacion de certificados

El sistema de creacion de certificados esta apoyado en el paquete pdffiller que permite al sistema completar el formulario de un pdf modelo de un certificado de donacion oficial

de la ONG Alboan.

En primer lugar el sistema comprueba que la donación no este utiliza consultando en la base de datos por el atributo *usada* del documento. Una vez comprobada que esto no es asi redirecciona al usuario al formulario en el que debe añadir los datos con los que quiere que se genere el formulario.

Una vez enviado el formulario el servidor se encarga de rellenar el PDF, enviar el mail con este al usuario y de actualizar la base de datos para que la donacion se marque como usada y se añadan los datos del donante. Todas estas acciones se realizan en conjunto y en caso de haber algun error se paralizará la funcion.

6.3.6 Codificacion del sistema de guardado de widgets

El sistema de guardado de widgets es muy sencillo ya que desde la página web se generar con un formato fijo. Esta funcion recibe desde la página web el script con los datos que el usuario a establecido y le añade unas funciones fijas que comparten todos los widgets. Una vez creado el widget minimizado, es decir, en una sola linea para ahorrar espacio, se intenta guardar en la carpeta donde estarán alojados todos los widgets. En caso de que el nombre del widgets ya exista, se comunicará al usuario.

El nombre que se le aplicará al widget seguirá el siguiente formato: Nombre de la empresa + _colmena.js.

```
router.post('/autosave', function(req, res) {
    var title = req.body.company + '_colmena.js';
    var scriptMin = CODIGO MINIMIZADO DEL SCRIPT
    fs.exists("public/routes/" + title, (exists) => {
        if (exists) {
            res.status(400);
            res.send({
                success: false,
            });
        } else {
            fs.writeFile("public/routes/" + title, scriptMin, function(err) {
                if (err) {
                   res.status(500).send('Archivo no guardado, reintente y si el
                       error persiste pongase en contacto con el soporte');
                }
            });
            res.status(200);
        }
    });
});
```

Listado 6.6: Codigo del sistema de guardado de widgets

6.3.7 Como utilizar la API para consultar datos

El servidor tambien ofrece una API para consultar diferentes datos y hacer nuestras propias estadisticas sin tener que depender de la aplicacion del proyecto. Esta API ofrece los datos en JSON sin formatear y directos de la base de datos. La API estará restringida

para los datos sensibles.

En el siguiente ejemplo se nos ofrece la posibilidad de entrar en la siguiente ruta: IPSERVIDOR/date/:year/:month. Cambiando *year* y *mont* por el año y el mes que queramos respectivamente, se nos ofrecera un json con todas las donaciones que ha habido ese mes.

```
router.get('/date/:year/:month', function(req, res) {
    ...
});
```

Listado 6.7: Ejemplo de ruta de la API

6.4 Implementación de la página web

La página web esta implementada sobre una plantilla de Bootstrap lo que ha permitido que el desarrollo se haga mas agil. En cambio, la plantilla ha sido alterada para que el diseño encaje con el de la ONG, por lo que de la plantilla original solo queda la estructura. Dentro del código de la pagina web tambien se han incluido varias librerias que permiten una experiencia mas inmersiva y con mas funcionalidades.

La página web esta dividida en plantillas *ejs* que permiten modularizar la estructura de la página y hacer la inclusión de nuevos modulos o pantallas mas sencilla. La plantilla principal de la página sera la que incluya todas las demas y las vaya alterando según donde se encuentre el usuario. En la plantilla principal o *layout* se añadiran los diferentes paquetes o librerias que todas las plantillas deban implementar. Es por esto que la primera vez que se entre a la página web se cargaran todos los paquetes y posteriormente la navegación sera mas fluida.

La página web cuenta en su parte superior con un *scrollspy*, es decir, la barra superior es un acceso directo a las diferentes secciones que tiene la página web que va siguiendo al usuario a medida que se va moviendo por la página web, de manera que siempre esta disponible. Esta barra esta diseñada de manera que sea responsiva, cuando el dispositivo desde el que se navegue sea demasiado pequeño, esta se unificará y se convertira en un menu desplegable.

Las diferentes secciones de la página web se han desarrollado mediante la etiqueta *section* (Listado 6.1). Esta etiqueta fue implementada en HTML5, la última version de esta tecnologia web. Estas secciones estan referenciadas desde la barra superior que hemos citado anteriormente, por lo que los usuarios podran moverse mas rapidamente por la página web.

```
<section id="services">
    <div class="container">
    ...
    </div>
</section>
```

Listado 6.8: Uso de section en la página web

Dentro de la página web hay varios botones que al pulsarlos despliegan diferentes ventanas las que hay desde información sobre los proyectos a los que se apoya con el proyecto hasta un asistente para crear tu propio widget. Estas ventanas, denominadas modales,

estan ocultas en todo momento hasta que se les llama desde con algun hipervinculo en un boton. Estas ventanas (Listado 6.2) implementan varias clases del framework Bootstrap, necesarios para conseguir que no esten visibles y hagan un efecto de entrada. El atributo *tabindex* designa cual sera el orden del foco a la hora de pulsar el boton tabulacion. El atributo *role=dialog* se utiliza para marcar que el elemento esta separado de la pagina principal y es un dialogo o ventana. Por ultimo, el atributo *aria-hidden* indica que el elemento no sera visible por el usuario hasta que se llame.

```
<div class="portfolio-modal modal fade" id="portfolioModal3" tabindex="-1" role="
    dialog" aria-hidden="true">
    ...
</div>
```

Listado 6.9: Código de una ventana Modal

6.4.1 Codificación del sistema de expedición de certificados de donación

El sistema de expedición de certificados de donación se basa en un campo en el que introducir la donación. Una vez introducida el numero de donación, unico para cada una y creado mediante un algoritmo explicado en la seccion anterior. El número de donación se consulta en la base de datos mediante un proceso explicado en la seccion anterior.

Una vez pasado la introduccion del codigo de donacion nos aparece un formulario que ha sido implementado mediante clases del framework Bootstrap. Lo mas destacable se encuentra dentro del script *form.js* en el que se implementan un par de funciones que le otorgan logica al formulario.

Dentro de las dos funciones que alberga este script se encuentra la que valida el formulario, comprobando que todos los campos esten completos y sobre todo que el mail este correcto, ya que si el mail esta mal el certificado no llegará al usuario. Para esta comprobacion se ha consultado en internet una expresion regular y se ha comprobado su eficacia en un testeador de expersiones regulares para emails online.

```
^((([a-z]|\d|[!#\$%&'\*\+\-\/=\?\^_`{\|}~]|[\u00A0-\uD7FF\uF900-\uFDCF\uFDF0-\
    uFFEF])+(\.([a-z]|\d|[!#\$%&'\*\+\-\/=\?\^_`{\|}~]|[\u00A0-\uD7FF\uF900-\
    uFDCF\uFDF0-\uFFEF])+)*)|((\x22)((((\x20|\x09)*(\x0d\x0a))?(\x20|\x09)+)?(([\
    x01-\x08\x0b\x0c\x0e-\x1f\x7f]|\x21|[\x23-\x5b]|[\x5d-\x7e]|[\u00A0-\uD7FF\
    uF900-\uFDCF\uFDF0-\uFFEF])|(\\([\x01-\x09\x0b\x0c\x0d-\x7f]|[\u00A0-\uD7FF\
    uF900-\uFDCF\uFDF0-\uFFEF])))*(((\x20|\x09)*(\x0d\x0a))?(\x20|\x09)+)?(\x22)
    ))@((([a-z]|\d|[\u00A0-\uD7FF\uF900-\uFDCF\uFDF0-\uFFEF])|(([a-z]|\d|[\u00A0
    -\uD7FF\uF900-\uFDCF\uFDF0-\uFFEF])([a-z]|\d|-|\.|_|~|[\u00A0-\uD7FF\uF900-\
    uFDCF\uFDF0-\uFFEF])*([a-z]|\d|[\u00A0-\uD7FF\uF900-\uFDCF\uFDF0-\uFFEF])))
    \.)+(([a-z]|[\u00A0-\uD7FF\uF900-\uFDCF\uFDF0-\uFFEF])|(([a-z]|[\u00A0-\uD7FF
    \uF900-\uFDCF\uFDF0-\uFFEF])([a-z]|\d|-|\.|_|~|[\u00A0-\uD7FF\uF900-\uFDCF\
    uFDF0-\uFFEF])*([a-z]|[\u00A0-\uD7FF\uF900-\uFDCF\uFDF0-\uFFEF])))\.?$
```

Listado 6.10: Expresión regular que comprueba los emails

Una vez se valida todo el formulario, se empaquetan todos los datos en JSON y se envian al servidor para que este los guarde en la base de datos. En esta transaccion se utiliza la tecnologia AJAX[6] que permite la comunicacion entre el servidor y la pagina web.

```
var cer = document.getElementById("formCertificateCol");

  var json = {};
  json.id = cer.elements[0].value;
  json.DNI = cer.elements[1].value;
  ...

  var jsonReady = JSON.stringify(json);
  $.ajax({
    type: "POST",
    url: "http://IPSERVIDOR/form2",
    data: jsonReady,
    dataType: "json",
    contentType: "application/json",
    success: function() {
      window.location.href = '/';
    }
  });
```

Listado 6.11: Código del envio de información del certificado al servidor

6.4.2 Codificación del asistente de creación de widgets

El sistema de creacion de widgets es similar al de expedición de certificados en la parte visual. Al clicar en un boton se desplegará una ventana modal y hay estará el asistente para la creacion de widgets. En este asistente no se creará el widget real, es una demo, para acceder al asistente real habrá que contactar con el soporte del proyecto.

El asistente consta de un formulario y un widget interactivo que va cambiando dependiendo de las modificaciones que se realicen en el formulario. Esto es posible gracias al script *widget.js* en el que esta desarrollada toda la logica de este asistente. Para que este asistente funcione tambien se han incluido otras librerias como la de seleccion de color[7] ofrecida por un desarrollador de la comunidad.

Dentro del script nombrado en el anterior parrafo se encuentran una serie de funciones que alteran esteticamente el widget, en este ejemplo (Listado 6.5) se muestra la funcion para cambiar el color del widget. En este caso se utiliza tambien la libreria nombrada anteriormente que ofrece una paleta de colores al usuario. En esta función podemos comprobar como al ir cambiando el color en el selector de color el atributo CSS de color de fondo del widget irá cambiando.

```
function cambiarColor() {
    $('#cp8Fondo').colorpicker().on('changeColor', function(e) {
        $('#colmenaDiv').css("background", e.color.toString('rgba'));
    });
}
```

Listado 6.12: Funcion para cambiar el color del widget

Una vez diseñado el widget a nuestra manera otras 2 funciones seran las encargadas de recoger toda la informacion del diseño del widget y de enviarselo al servidor para que este lo guarde y este disponible automaticamente. La informacion del widget se envia en JSON

y se envía al servidor de la misma manera que se enviaba la informacion del certificado, mediante la tecnologia AJAX (Listado 6.6).

```
    var json = {};
    json.data = script;
    json.company = empresa;
    json.mail = email;
    var jsonReady = JSON.stringify(json);
        $.ajax({
        type: "POST",
        url: "http://IPSERVIDOR/autosave",
        data: jsonReady ,
        success: function( success ){
          if(!success) alert("Ese nombre ya existe");
        },
        dataType: "json",
        contentType: "application/json"
    });
```

Listado 6.13: Un fragmento de la función encargada de enviar el widget al servidor

6.5 Implementación del widget

El widget esta implementado de la manera mas sencilla y con las tecnologias menos pesadas posibles para no suponer una carga para la tienda que lo implemente. El widget estará alojado en el servidor y la tienda implementará una llamada a este para que este se integre en la tienda cada vez que un comprador entra a ella.

El widget consta de dos funciones que permiten que el importe de donacion varie segun las opciones que elija el usuario y otra que es la encargada de enviar los datos al servidor para que este recoja las donaciones. Finalmente el widget tiene el diseño creado por la empresa.

Como implementacion a destacar se encuentra la manera de codificar el widget. Se ha desarrollado de manera que no necesite una hoja de estilos anexa para funcionar, ya que él mismo implementa el estilo dentro del HTML. Todo el widget esta escrito en un String para que este luego sea inyectado en el comercio online. En la muestra de codigo que se puede ver a continuacion (Listado 6.14) no se ha añadido todo el codigo ya que la extension y el formato de este, al estar escritos sobre un String, hacia muy dificil su lectura.

```
var main_container = document.getElementById('colmenaWidget');
main_container.innerHTML = '' +
    '<div>' +
    '<input type="hidden" name="colmena_check" id="colmena_check" value="0"/>' +
    '<input type="hidden" name="colmena_amount" id="colmena_amount" value="1"/>'
        +
    '<div id="colmenaDiv"> <div id="colmenaText"> Apoya al proyecto <strong>
        Tecnologia libre de conflicto</strong> [...] </div>' +
    '<style> #colmenaDiv { background: url(https://s22.postimg.org/6nlsttl5d/
        woman_898760_1920.jpg) no-repeat center; background-size: cover;[...].cbx
        :disabled~label:after { background: #FFA726;}.hidden { display: none;}</
        style>'
```

Listado 6.14: Codigo de un widget

6.6 **Implementación del sistema de visualización**

7. PLAN DE PRUEBAS

8. MANUAL DE USUARIO

9. CONCLUSIONES Y LINEAS FUTURAS

BIBLIOGRAFÍA

[1] Atom(Wikipedia) [online]. mayo 2017. URL: https://es.wikipedia.org/wiki/Atom_(editor_de_textos).

[2] Equipo (Team) [online]. mayo 2017. URL: https://proyectosagiles.org/equipo-team/.

[3] Facilitador (Scrum Master) [online]. mayo 2017. URL: https://proyectosagiles.org/facilitador-scrum-master/.

[4] Alboan [online]. abril 2017. URL: https://www.alboan.org/es.

[5] Node.js [online]. mayo 2017. URL: https://nodejs.org/es/.

[6] Ajax [online]. mayo 2017. URL: https://www.tutorialspoint.com/ajax/ajax_technology.htm.

[7] Bootstrap Color Picker 2.5.1 [online]. mayo 2017. URL: https://itsjavi.com/bootstrap-colorpicker/.

[8] Atom [online]. mayo 2017. URL: https://atom.io/.

[9] MongoDB(Wikipedia) [online]. mayo 2017. URL: https://es.wikipedia.org/wiki/MongoDB.

[10] MongoDB [online]. mayo 2017. URL: https://www.mongodb.com/es.

[11] Git(Wikipedia) [online]. mayo 2017. URL: https://es.wikipedia.org/wiki/Git.

[12] Git [online]. mayo 2017. URL: https://git-scm.com/.

[13] Brackets(Wikipedia) [online]. mayo 2017. URL: https://en.wikipedia.org/wiki/Brackets_(text_editor).

[14] Brackets [online]. mayo 2017. URL: http://brackets.io/.

[15] Cliente (Product Owner) [online]. mayo 2017. URL: https://proyectosagiles.org/cliente-product-owner/.