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

2.1 Definición del proyecto

2.1.1 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.

2.1.2 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.

2.1.3 Producto final

El producto final se compone de varios elementos que con

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, ofrecen una API en la que poder consultar los datos que la base de datos aloja, siempre con cierta privacidad.

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.

2.2 Descripción de realización

2.2.1 Método de desarrollo

2.2.2 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 las tecnologías a usar
- Documento de seguimiento de las tareas realizadas
- Varios prototipos de la página web
- Varios prototipos del widget
- 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.

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/');
```

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
```

- **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);
```

■ 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'])
```

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
});
```

■ 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).

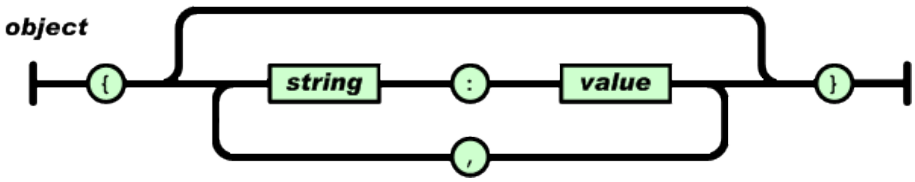

Figura 4.1: Diagrama de la gramática de un objeto JSON

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

Figura 4.2: Ejemplo de un JSON

4.4 Sass (CSS3)

La definición de Sass y CSS3 integrados es la siguiente:

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 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

codigo scss

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

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

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

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

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

Figura 4.3: Scss y css compilado respectivamente

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.

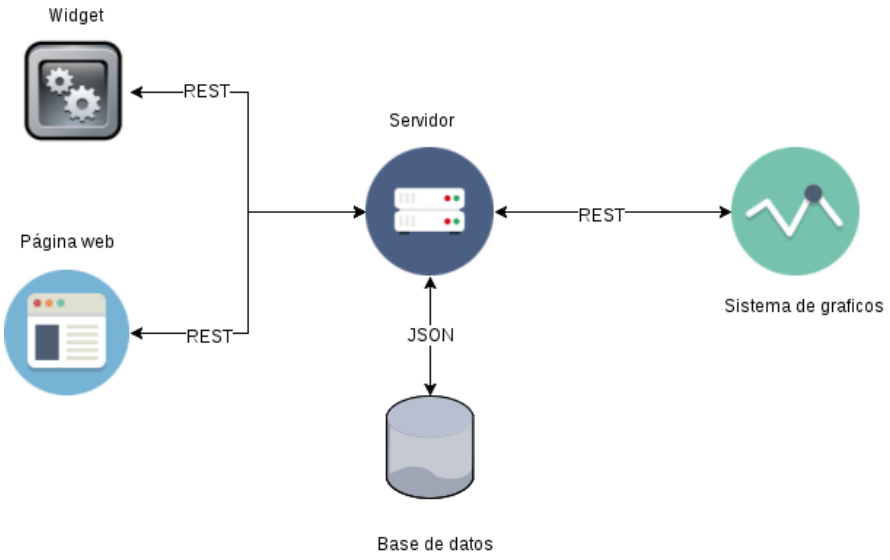

Figura 5.1: Arquitectura del proyecto

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 9.2) 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 9.3) 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.

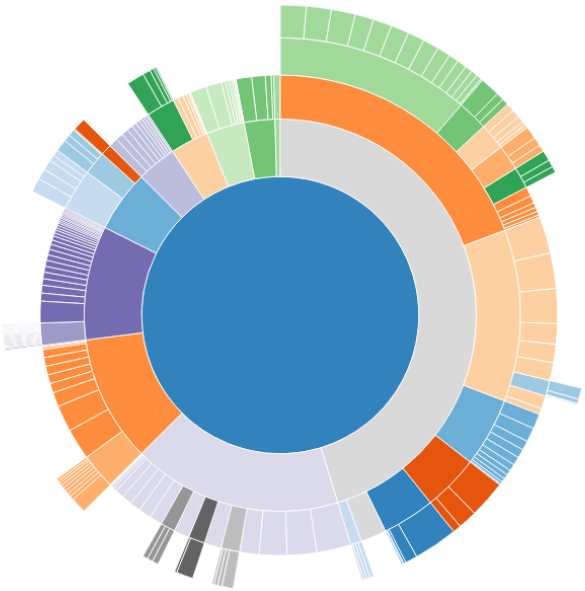

Figura 5.2: Diagrama circular

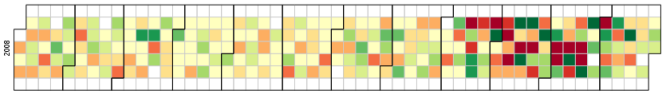

Figura 5.3: Diagrama de fechas

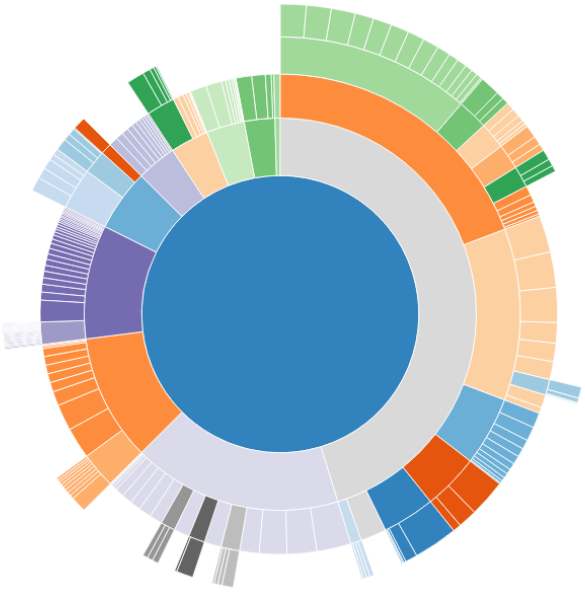

Figura 5.4: Diagrama circular

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

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.

La página web esta dividida por plantillas que permiten

6.5 Implementación del widget

6.6 Implementación de la base de datos

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

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

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

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

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

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

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

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