**Especificación de Requerimientos**

**Descripción del Diseño**

**Agenda telefónica digital**

**Para ingreso, modificación, consulta y eliminación de contactos**

|  |  |  |
| --- | --- | --- |
| Apellidos, Nombres | Correo electrónico | Rol |
| Julio Blanco | Ju\_gre@outlook.com | Backend |
| Danny Lasso | Danny.lasso@mail.escuelaing.edu.co | Frontend |

Fecha de presentación: 15/10/2021

Contenido

[1 INTRODUCCIÓN 3](#_Toc77242272)

[1.1 Propósito 3](#_Toc77242273)

[1.2 Alcance o Ámbito del Sistema 3](#_Toc77242274)

[1.3 Definiciones, Acrónimos y Abreviaturas 3](#_Toc77242275)

[1.3.1 Definiciones 3](#_Toc77242276)

[1.3.2 Acrónimos 3](#_Toc77242277)

[1.3.3 Abreviaturas 3](#_Toc77242278)

[1.3.4 Referencias 4](#_Toc77242279)

[1.4 Perspectiva General del Documento 4](#_Toc77242280)

[2 DESCRIPCIÓN GENERAL DE LA APLICACIÓN 4](#_Toc77242281)

[2.1 Perspectiva de la Aplicación 4](#_Toc77242282)

[2.2 Funciones de la Aplicación 4](#_Toc77242283)

[2.3 Características de los Usuarios 5](#_Toc77242284)

[2.4 Restricciones 5](#_Toc77242285)

[2.5 Suposiciones y Dependencias 5](#_Toc77242286)

[2.6 Requerimientos Diferidos 5](#_Toc77242287)

[3 REQUERIMIENTOS ESPECÍFICOS 5](#_Toc77242288)

[3.1 Requerimientos 6](#_Toc77242289)

[3.1.1 Product Backlog 6](#_Toc77242290)

[3.1.2 Ciclo de Sprints del proyecto 6](#_Toc77242291)

[3.1.3 Sprint Backlog 6](#_Toc77242292)

[3.1.4 Historias de usuario (Tareas y Subtareas) 6](#_Toc77242293)

[3.1.5 Mecánica de organización del grupo. (Reuniones, evidencias/artefactos) 6](#_Toc77242294)

[3.2 Modelo de Requerimientos 7](#_Toc77242295)

[3.2.1 Modelo de Casos de Uso 7](#_Toc77242296)

[4 DESCRIPCIÓN DEL DISEÑO 9](#_Toc77242297)

[4.1 Interfaz gráfica (Mockups) 9](#_Toc77242298)

[5 Gestión de la configuración 9](#_Toc77242299)

[6 PRUEBAS 9](#_Toc77242300)

[6.1 Descripción de pruebas unitarias 9](#_Toc77242301)

[6.2 Descripción de pruebas de aceptación 9](#_Toc77242302)

[7 GLOSARIO 10](#_Toc77242303)

[8 ANEXO(S) 10](#_Toc77242304)

# Introducción

En este documento se proporcionará toda la información referente al proceso de desarrollo de una agenda digital comprendiendo los pasos desde la conformación del grupo de trabajo, definición de roles, tareas a realizar y tiempo de dedicación para cada una de ellas. Se discutirá también acerca de los usos que tendrá el software, cual será su alcance, sus limitaciones, el diseño que tendrá la interfaz de usuario, los requerimientos que debe cumplir para ser funcional y que pruebas debe enfrentar para su aprobación final.

Finalmente se discutirá el tipo de configuración que hará parte del diseño y desarrollo del software.

Para el desarrolló de esta aplicación se utilizaron diferentes herramientas como Visual studio code para poder modificar los algoritmos principales, lenguajes como Java Script y html, para realización de front end, se utilizarón archivos con extensiones .css y vuejs y finalmente las conexiones a bases de datos y despliegue del proyecto en web se realizaron con ayuda de MongoDB y Heroku.

## Propósito

El propósito de este documento es el de guiar al lector a través del proceso completo llevado a cabo para la realización de la agenda digital y un análisis detallado de cada una de las fases que se deben tener en cuenta para poder realizar con éxito el proyecto.

Este documento esta dirigido a lectores con experiencia en desarrollo de software y estudiantes de programación básica, ya que para su desarrolló se partió de ejercicios realizados en las clases.

## Alcance o Ámbito del Sistema

Se tiene como objetivo diseñar una **agenda digital** como una herramienta personal para el almacenamiento de listas de contactos con distintos campos relacionados a cada uno de ellos como nombre, correo, y teléfono. Dicha agenda debe permitir agregar, modificar consultar y eliminar dichos campos mediante la selección de botones de agregar, eliminar o modificar que permitirán realizar dichas operaciones que actúan directamente sobre una base de datos la cual se gestionó con ayuda de las aplicaciones MongoDS, MongoCompass y Mongo Atlas.

Las pruebas iniciales de la interacción del sistema en el servidor local se realizaron mediante la API Postman que es una plataforma que permite la construcción y uso de APIs.

El alcance planteado para esta aplicación que puede ser utilizada para organizar contactos y tener a la mano información importante como dirección, correo, teléfono, redes sociales, etc; es la de poderse desplegar en la web y que sirva como un proyecto para poder comprender la interacción entre bases de datos, backend y frontend para el desarrollo futuro de proyectos más avanzados.

Dicha aplicación debe lograr responder solicitudes para poder crear, modificar, eliminar y visualizar información mediante una interfaz de usuario amigable e intuitiva.

## Definiciones

En este apartado se describirán y explicarán diferentes palabras clave que se referirán constantemente a lo largo del presente documento para la claridad del lector.

### Definiciones

* **Back-end**:  es la parte que se conecta con la base de datos y el servidor que utiliza dicho sitio web, por eso decimos que el backend corre del lado del servidor.
* **Nodemon**: es una utilidad que monitorea los cambios en el código fuente que se esta desarrollando y automáticamente re inicia el servidor. Es una herramienta muy útil para desarrollo de aplicaciones en nodojs.
* **Servidor**: Un servidor es un conjunto de computadoras capaz de atender las peticiones de un [cliente](https://es.wikipedia.org/wiki/Cliente_(inform%C3%A1tica)) y devolverle una respuesta en concordancia. Los servidores se pueden ejecutar en cualquier tipo de computadora, incluso en computadoras dedicadas a las cuales se les conoce individualmente como «el servidor».
* **Middleware**: Un middleware es un software que proporciona un enlace entre aplicaciones, sistemas o proyectos independientes. En otras palabras es un elemento que usamos como la vía para conectar dos aplicaciones o dos partes, su función es pasar datos entre ellas o de un lado a otro.

De hecho su significado es “medio” y lo fabricamos e implementamos en cualquier área, un middleware puede ser un hardware, *este termino se usa desde hace varias décadas en informática*.

* **Express**: es un marco de aplicación web de back-end para Node.js, lanzado como software gratuito y de código abierto bajo la licencia MIT. Está diseñado para crear aplicaciones web y API.
* **Morgan**: Morgan es un middleware que nos ayudará a identificar a los clientes que acceden a nuestra aplicación. Básicamente un registrador.
* **HTTP**: HTTP, de sus siglas en inglés: "Hypertext Transfer Protocol", es el nombre de un protocolo el cual nos permite realizar una petición de datos y recursos, como pueden ser documentos [HTML](https://developer.mozilla.org/es/docs/Glossary/HTML). Es la base de cualquier intercambio de datos en la Web, y un protocolo de estructura cliente-servidor, esto quiere decir que una petición de datos es iniciada por el elemento que recibirá los datos (el cliente), normalmente un navegador Web.
* **CORS**: El Intercambio de Recursos de Origen Cruzado ([CORS (en-US)](https://developer.mozilla.org/en-US/docs/Glossary/CORS)) es un mecanismo que utiliza cabeceras [HTTP](https://developer.mozilla.org/es/docs/Glossary/HTTP) adicionales para permitir que un [user agent (en-US)](https://developer.mozilla.org/en-US/docs/Glossary/User_agent" \o "Currently only available in English (US)) obtenga permiso para acceder a recursos seleccionados desde un servidor, en un origen distinto (dominio) al que pertenece. Un agente crea una petición HTTP de origen cruzado cuando solicita un recurso desde un dominio distinto, un protocolo o un puerto diferente al del documento que lo generó.
* **JSON**: JSON son las iniciales de JavaScript Object Notation el cual es un formato que guarda objetos de javascript muy ligeros y fáciles de interpretar, lo cual hace que sea mas aceptado que xml y soportado por otros lenguajes de programación como python, C/C++, entre otras.
* **Objetos**: En el [paradigma de programación](https://es.wikipedia.org/wiki/Paradigma_de_programaci%C3%B3n) [orientada a objetos](https://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos) (POO, o bien OOP en [inglés](https://es.wikipedia.org/wiki/Idioma_ingl%C3%A9s)), un objeto es un ente orientado a objetos ([programa de computadoras](https://es.wikipedia.org/wiki/Programa_(computaci%C3%B3n))) que consta de un estado y de un comportamiento, que a su vez constan respectivamente de datos almacenados y de tareas realizables durante el tiempo de ejecución. Un objeto puede ser creado [instanciando](https://es.wikipedia.org/wiki/Instancia_(programaci%C3%B3n)) una [clase](https://es.wikipedia.org/wiki/Clase_(inform%C3%A1tica)), como ocurre en la programación orientada a objetos, o mediante escritura directa de código y la replicación de otros objetos, como ocurre en la [programación basada en prototipos](https://es.wikipedia.org/wiki/Programaci%C3%B3n_basada_en_prototipos).
* **Vue**: Vue.js es un framework de JavaScript de código abierto para la construcción de interfaces de usuario y aplicaciones de una sola página.
* **Babel**: Babel se define como un transpilador de JavaScript de código abierto, o un compilador de fuente a fuente, como se conocen a nivel técnico. Es una herramienta que lee el código fuente, que está escrito en un lenguaje de programación en concreto y produce el código equivalente en otro lenguaje.
* **JavaScript**: JavaScript es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, ​ basado en prototipos, imperativo, débilmente tipado y dinámico.
* **Mongoose**: Mongoose es una librería para Node.js que nos permite escribir consultas para una base de datos de MongooDB, con características como validaciones, construcción de queries, middlewares, conversión de tipos y algunas otras, que enriquecen la funcionalidad de la base de datos.

La parte central del uso de Mongoose está en la definición de un esquema donde se indica la configuración de los documentos para una colección de MongoDB. Y aunque MongoDB es una base de datos nosql, donde los documentos se almacenan sin un esquema predefinido, el uso de un esquema te permite normalizar tu información, sin sacrificar la flexibilidad. Además, hace que la transición de sql a nosql, sea más sencilla.

* **Front-end**: es la parte de un sitio web que interactúa con los usuarios, por eso decimos que está del lado del cliente.
* **Gitignore**: gitignore , es un archivo de texto que le dice a Git qué archivos o carpetas ignorar en un proyecto. Un archivo local . gitignore generalmente se coloca en el directorio raíz de un proyecto. También puedes crear un archivo global.
* **Bootstrap**: Bootstrap es una biblioteca multiplataforma o conjunto de herramientas de [código abierto](https://es.wikipedia.org/wiki/C%C3%B3digo_abierto) para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en [HTML](https://es.wikipedia.org/wiki/HTML) y [CSS](https://es.wikipedia.org/wiki/Hojas_de_estilo_en_cascada), así como extensiones de [JavaScript](https://es.wikipedia.org/wiki/JavaScript) adicionales. A diferencia de muchos frameworks web, solo se ocupa del desarrollo [front-end](https://es.wikipedia.org/wiki/Front-end" \o "Front-end).

### Referencias

[1] npm. 2021. *nodemon*. [online] Available at: <https://www.npmjs.com/package/nodemon> [Accessed 16 October 2021].

[2] Azure.microsoft.com. 2021. *Qué es middleware: definición y ejemplos | Microsoft Azure*. [online] Available at: <https://azure.microsoft.com/es-es/overview/what-is-middleware/> [Accessed 16 October 2021].

[3] DigitalOcean. 2021. *How To Use morgan in Your Express Project | DigitalOcean*. [online] Available at: <https://www.digitalocean.com/community/tutorials/nodejs-getting-started-morgan> [Accessed 16 October 2021].

[4] Engineering Education (EngEd) Program | Section. 2021. *How to use CORS in Node.js with Express*. [online] Available at: <https://www.section.io/engineering-education/how-to-use-cors-in-nodejs-with-express/> [Accessed 16 October 2021].

[5] Node.js. 2021. *What is the file `package.json`? | Node.js*. [online] Available at: <https://nodejs.org/en/knowledge/getting-started/npm/what-is-the-file-package-json/> [Accessed 16 October 2021].

[6] Es.wikipedia.org. 2021. *Vue.js - Wikipedia, la enciclopedia libre*. [online] Available at: <https://es.wikipedia.org/wiki/Vue.js> [Accessed 16 October 2021].

[7] En.wikipedia.org. 2021. *Babel (transcompiler) - Wikipedia*. [online] Available at: <https://en.wikipedia.org/wiki/Babel\_(transcompiler)> [Accessed 16 October 2021].

[8] Es.wikipedia.org. 2021. *MongoDB - Wikipedia, la enciclopedia libre*. [online] Available at: <https://es.wikipedia.org/wiki/MongoDB> [Accessed 16 October 2021].

[9] CódigoFacilito. 2021. *Qué es mongoose*. [online] Available at: <https://codigofacilito.com/articulos/que-es-mongoose> [Accessed 16 October 2021].

[10] Vortexbird.com. 2021. *Nodemon*. [online] Available at: <https://vortexbird.com/nodemon/#:~:text=Nodemon%20es%20una%20utilidad%20que,desarrollo%20de%20aplicaciones%20en%20nodojs.> [Accessed 16 October 2021].

[11] Es.wikipedia.org. 2021. *Servidor - Wikipedia, la enciclopedia libre*. [online] Available at: <https://es.wikipedia.org/wiki/Servidor> [Accessed 16 October 2021].

[12] 2021. [online] Available at: <https://platzi.com/blog/laravel-middleware/?utm\_source=google&utm\_medium=cpc&utm\_campaign=12915366154&utm\_adgroup=&utm\_content=&gclid=CjwKCAjwzaSLBhBJEiwAJSRokgB875GIzX\_ekbuucpFji5s55dXCfn0xwNehFVNuSuldp1CJKNJbfxoCnzsQAvD\_BwE&gclsrc=aw.ds> [Accessed 16 October 2021].

[13] Developer.mozilla.org. 2021. *Generalidades del protocolo HTTP - HTTP | MDN*. [online] Available at: <https://developer.mozilla.org/es/docs/Web/HTTP/Overview> [Accessed 16 October 2021].

## Perspectiva General del Documento

[Este apartado debe:

a) describir lo que contiene el resto del documento;

b) explicar cómo está organizado el documento.]

# Descripción general de la aplicación

Es importante realizar una descripción general de los factores principales que afectal al desarrollo de la agenda digital así como también de los requerimientos básicos que permitirán que la aplicación funcione correctamente tanto en las etapas de prueba como durante el despliegue final en un servidor web mediante el uso de Heroku.

## Perspectiva de la Aplicación

Para lograr un optimo funcionamiento de nuestra agenda digital fue necesario contar con diferentes herramientas que juntas nos permitieron establecer una estructura para interconectar diferentes sistemas entre los que encontramos la capa back end, front end y la intereacción bases de datos con el usuario final.

Principalmente para poder obtener nuestro producto final tenemos como requerimiento principal contar con dependencias como Mongo, Boostrap, Babel, Heroku y Vue. Tampoco debemos dejar de lado el Middleware que para el desarrolló de esta aplicación fue Morgan desplegado en nuestra aplicación Express.

Para llevar un orden en la explicación de funcionalidades y componentes, empezaremos describiendo el desarrollo y componentes que se encuentran en el desarrollo del Backend.

Nodemon es una herramienta que nos ayudó principalmente a desarrollar aplicaciones basadas en java script mediante el reinicio automático de aplicaciones tipo nodo cuando un intercambio de archivos en el directorió se detecte. Este servidor de escucha nos va a permitir monitorear los cambios y permitirá reiniciar el server automaticamente como se mencionó anteriormente sin tener que parar la aplicación para reiniciar y subir cambios.

Los middleware son una parte del software que se sitúa entre un sistema operativo y las aplicaciones que se ejecutan en el. Este funciona como una capa oculta de traducción que permite la comunicación y administración de datos en ap´licaciones distribuidas. El uso de middleware permite a los usuarios realizar solicitudes como el envio de formularios en un exploirador web o permitir que un servidor web devuelva paginas web dinámicas en función de un perfil de usuario.

Morgan es un middleware Node.js y Express que permite mostrar las peticiones HTTP, los errores y simplifica el proceso. Existen unos prerrequisitos para poder lanzar Morgan, estos son:

* Entendimiento general de Node.js
* Instalación local de Node.js

Lo primero para poder desplegar moran es asegurarse de contar con la ultima versión de Node.js, luego debemos incluir Morgan en nuestro proyecto Express para lo cual es necesario instalarlo como una dependencia.

Para usar Morgan en el servidor, fue necesario invocar una instancia y enviar un argumento en una función del middleware .use() antes que la solicitud HTTP.

Morgan nos permitirá flexibilidad cuando se realicen solicitudes HTTp y actualizaciones en tiempos precisos en formatos que podemos definir según nuestras necesidades.

Node.js es un plataforma runtime usada cuando se ejecuta código JavaScript en un servidor. Uno de sus frameworks mas populares es Express y de este se deriva Cors el cual nos ayuda a acceder a múltiples funcionalidades en el buscador.

Cors nos permite contar con una seguridad en el uso de nuestras APIs. Esto se realiza mediante el Bypassing de las cabeceras Access-Control-Allow-Origin, las cuales especifican que orígenes pueden acceder a la API. En pocas palabras, CORS es una característica de seguridad de un buscador que restringe acceso a solicitudes HTTP con otros servidores y especifica a que dominios se les permitirá el acceso a los recursos.

Todas las dependencias que vamos instalando se van a almacenar en nuestro archivo Package.json el cual se encuentra en la raíz de nuestro proyecto. Este archivo contiene mucha metadata relevante para el proyecto. Este archivo le da información al npm que permite identificar el proyecto así como manejar las dependencias del mismo. Este archivo también puede contener otra metadata como una descripción del proyecto.

Otro de los frameworks de JavaScript utilizados en la construcción de nuestra interfaz de usuario es Vue. Este framework es utilizado para la construcción de interfaces de usuario y aplicaciones de una sola pagina. Vue utiliza una sintaxis de plantillas basadas en HTML que permiten vincular el dominio renderizado a los datos de la instancia subyacente de Vue. Toda las plasntillas de Vue son validas para HTML y pueden ser analizados por navegasdores que cumplan con las especificaciones y por analizadores HTML. Vue compila las plantillas en funciones de representación de un objeto de documento virtual que permite a Vue renderizar componentes en su memoria antes de actualizar el navegador. Como lo he mos descrito, Vue se utilizó en nuestra aplicación para crear las vistas que finalmente aparecerán ante el usuario final.

Antes de describir en detalle la interacción del framework encargado del front end, es importante mencionar que para lograr una compatibilidad universal de nuestro código JavaScript, fue necesario usar un transcompiler JavaScript llamado Babel.

Babel es usado para convertir código EMCAScript 2015+ en versionas anteriores compatibles con JavaScript que pueden ser ejecutadas por servidores mas antiguos.

Los desarrolladores pueden usar las nuevas funciones de versiones mas modernas de JavaScript mediante babel para que estas puedan ejecutarse en servidores con versiones anteriores. En síntesis, Babel se encarga de transformar la sintaxis que no es ampliamente soportada en una versión anterior compatible.

Una vez se han definido todas las características que va a tener nuestro servidor, es hora de realizar la conexión con el servidor con ayuda de Mongo.

Mongo es una base de datos para documentos que soporta diferentes lenguajes y puede ser ejecutada tanto local como alojada en un servidor web. Mongo debe configurarse en nuestro proyecto para permitir realizar consultas, almacenamiento de archivos, agregación de información y todo esto mediante la ejecución de códigos JavaScript que permitan realizar dichas interacciones.

Para poder realizar las consultas a esta base de datos, también fue necesario instalar una librería Node.js llamada Mongoose. Esta librería nos permite escribir consultas para una base de datos de MongooDB, con características como validaciones, construcción de Queries, Middlewares, conversión de tipos y algunas otras, que enriquecen la funcionalidad de la base de datos. En resumen, Mongoose funciona como una capa adicional sobre MongoDC a través de la cual se implementan y automatizan muchas de las tareas habituales de trabajar con una base de datos.

Finalmente tras realizar la conección con la base de datos, tenemos listo el servidor y podemos proceder a crear los esquemas. Toda la esquematización que se va a establecer en el proyecto se llevá a cabo en la carpeta models, y las rutas de cada vista se van a gestionar en el archivo routes.

Es aquí donde empezaremos a exponer las funcionalidades que se desprenden del desarrollo front end del proyecto.

Como mencionamos anteriormente, Vue se utilizó para crear las vistas del cliente.

En App.vue se encuentra la vista principal que se mostrará al usuario cuando acceda a la pagina web. Por otra parte, en router, tenemos un archivo index.js donde se encuentran todas las rutas relacionadas a las diferentes vistas del proyecto.

Lo primero que se realiza para generar als vistas es crear las rutas.

Los diseños de la tabla que se muestra en al ventana principal de contactos se obtuvo de los diseños predefinidos de Bootstrap.

Bootstrap es una biblioteca multiplataforma o conjunto de herramientas de código abierto para el diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografia, formularios, botones, cuadros, menús de navegación y otros elementos de diseño absado en HTML y CSS, así como extensiones de JavaScript adicionales y solo se ocupa del desarrollo front-end.

Todas las rutas del proyecto se definieron según las rutas que se estableceron en la carpeta router del backend.

## Funciones de la Aplicación

A continuación describiremos las funciones con las que cuenta la aplicación que se desarrolló y describió en el apartado anterior.

Esta aplicación funciona como una agenda digital que permite a los usuarios almacenar información sobre sus contactos de forma sencilla, así mismo, permite a los usuarios consultar los contactos, modificarlos o eliminarlos según sus preferencias

Las funciones de la aplicación serían entonces:

* Agregar contacto

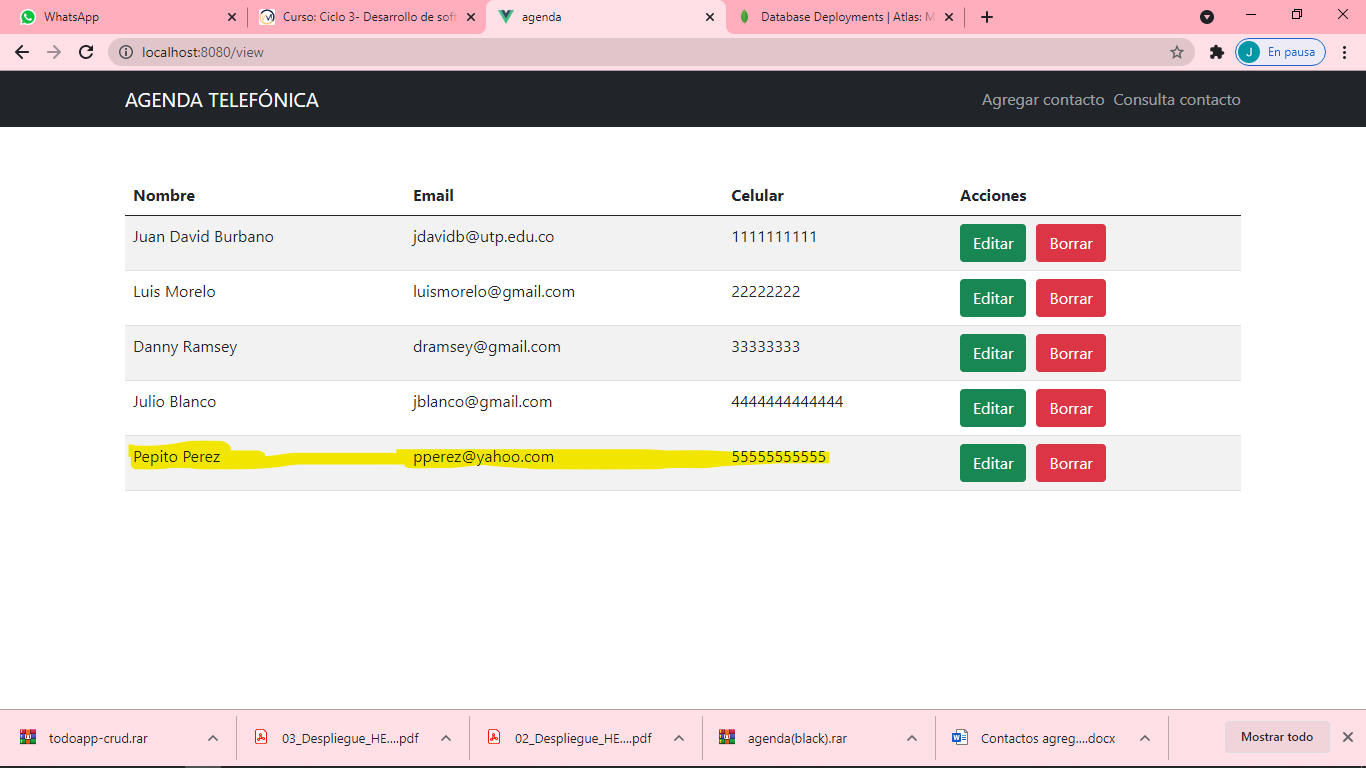
Interfaz de usuario gráfica

Descripción generada automáticamente

Interfaz de usuario gráfica, Sitio web

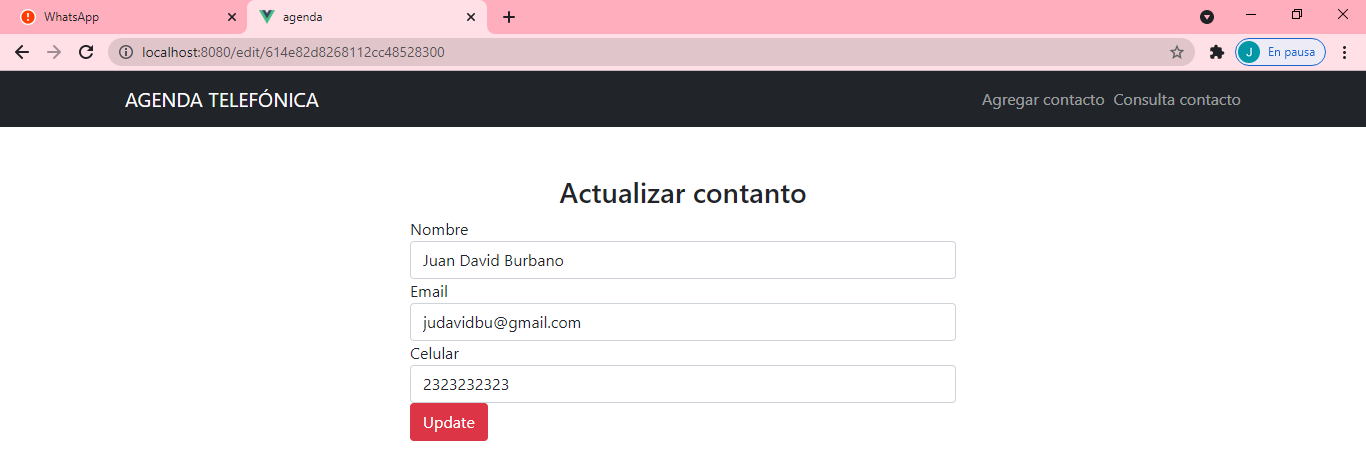
Descripción generada automáticamente

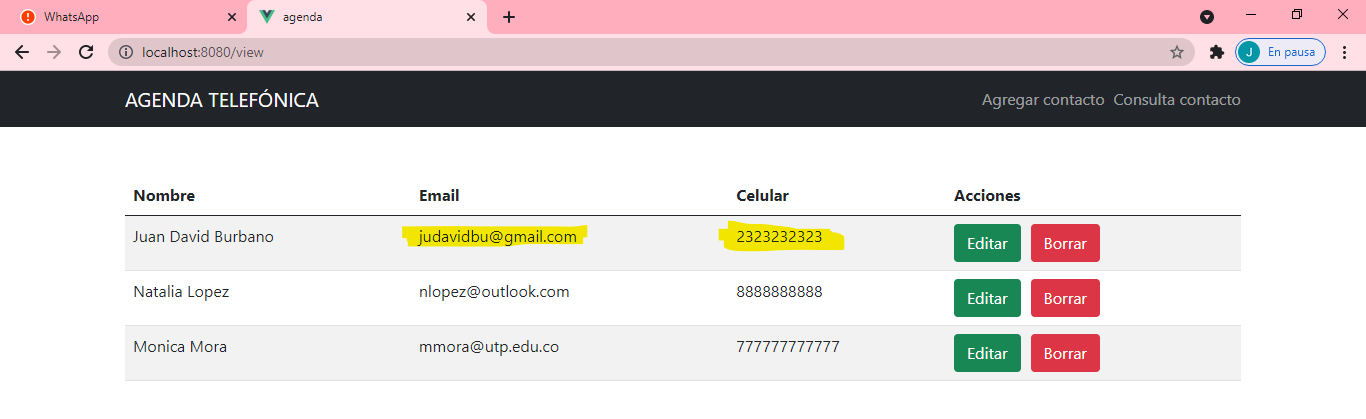
* Consultar contacto



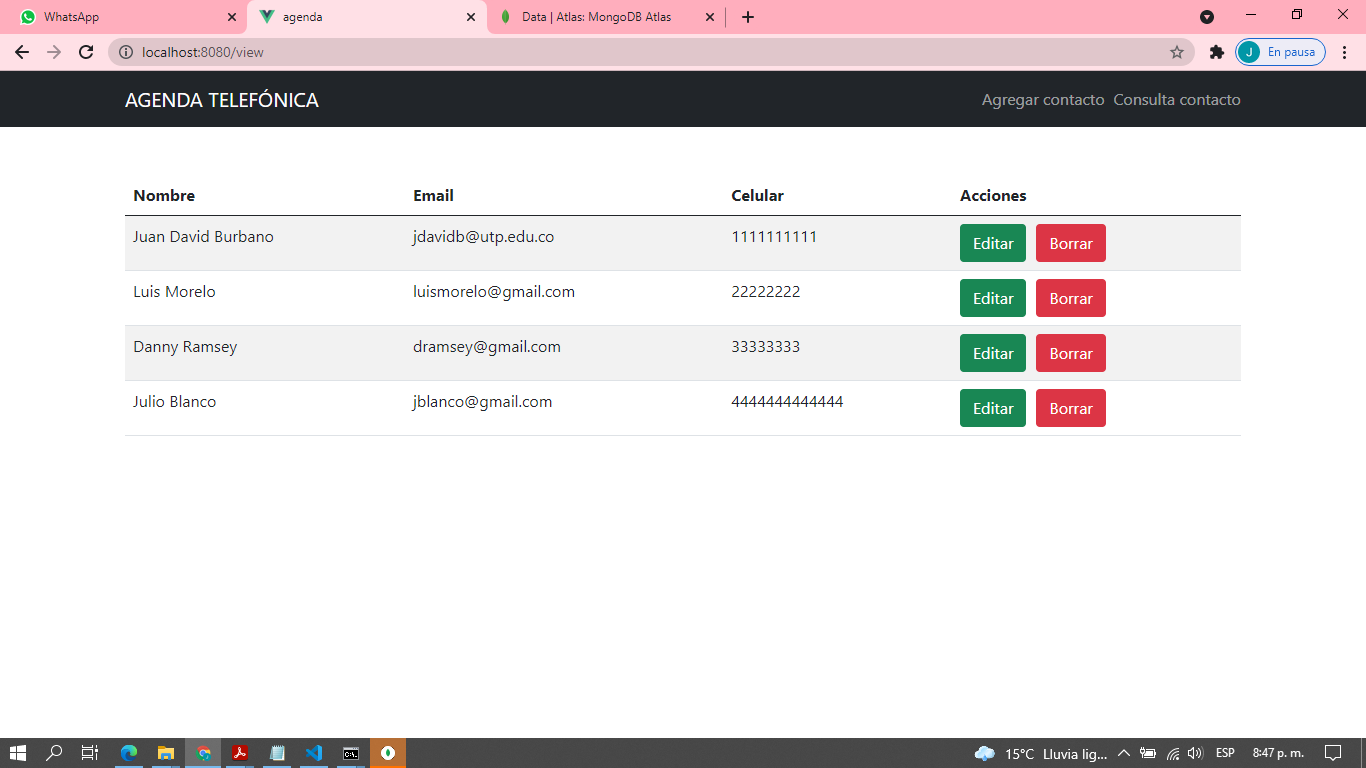
* Actualizar contacto

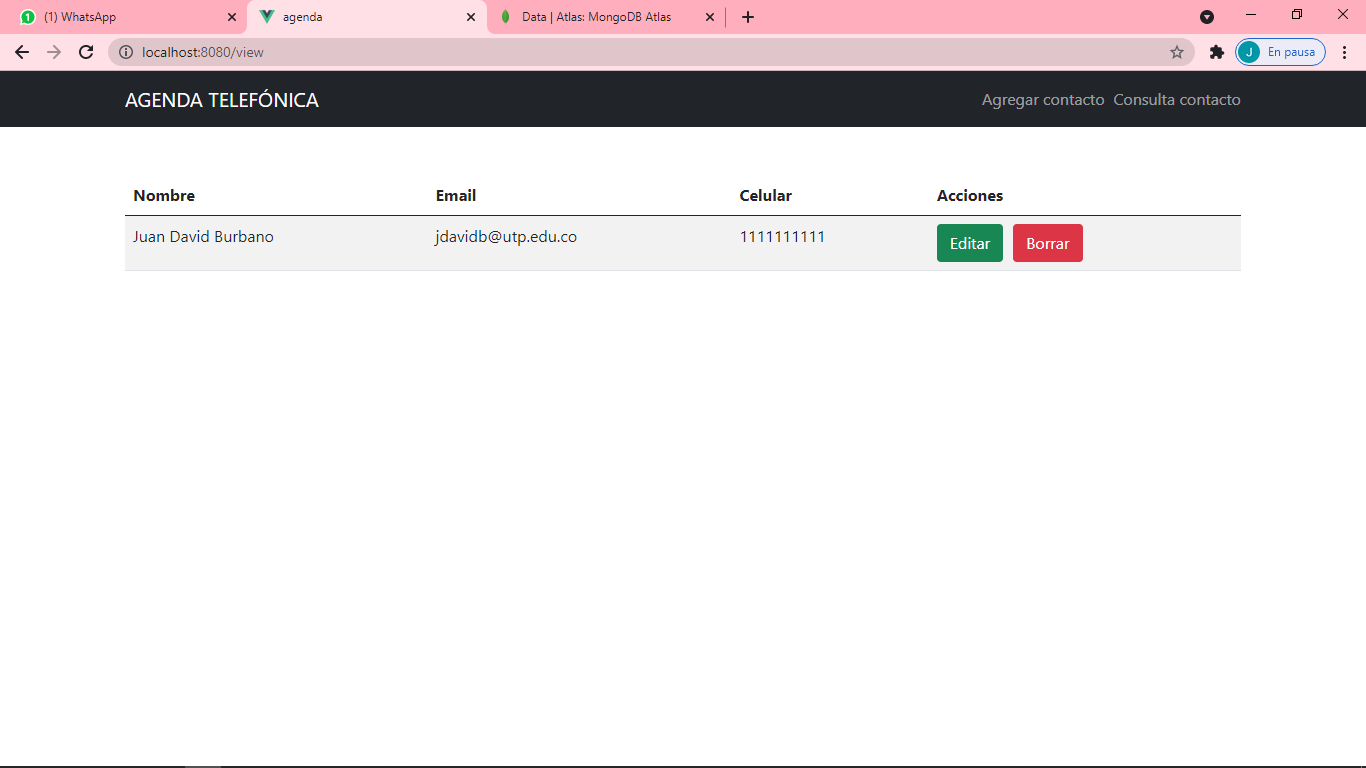






* Eliminar contacto





## Características de los Usuarios

Los usuarios a quienes va dirigida esta aplicación son personas del común que requieran de una agenda alojada en un servidor web para así poder consultar información de sus contactos de manera remota en cualquier lugar del mundo sin la necesidad de depender de sus dispositivos móviles.

## Restricciones

Las limitaciones del proyecto apuntan directamente a los conocimientos que se lograron adquirir durante las 5 semanas que duró el ciclo actual ya que las tematiacas son bastante profundas y ricas en contenidos que se tocaron de manera superficial. Sin embargo, durante este tiempo se logró comprender las herramientas necesarias básicas para comprender de forma general el proceso y los componentes necesarios para poder realizar un correcto despliegue de una aplicación web en un servidor y que esta aplicación sea completamente funcional.

## Suposiciones y Dependencias

Las razónes por las cuales el correcto funcionamiento de la aplicación se podrían ver afectados dependen del uso de un navegador desactualizado o posiboemente ejecución de la aplicación desde dispositivos móviles con navegadores no soportados.

## Requerimientos Diferidos

Los requerimientos básicos para el funcionamiento de la aplicación son sencillamente contar con un navegador actualizado ejecutado desde un ordenador o en versión de escritorio.

# Requerimientos específicos

Los requerimientos específicos para poder ejecutar la aplicación responden directamente a la ejecución de los comandos integrados en el despliegue de la misma en un servidor web por lo que es importante asegurarse de contar con un sistema operativo moderno y actualizado y así mismo utilizar un navegador popular como es el caso de Google Chrome o Mozilla Firefox.

## Requerimientos

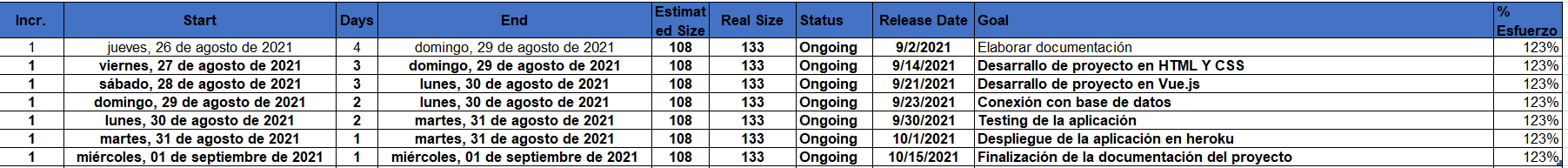
* Mozilla Firefox 92.0
* Google Chrome 94.0
* Windows 10 64bits
* Procesador Intel core
* Memoria ram 500Mb
* Tarjeta de video: No
* Conexión a internet estable de al menos 20 megas

### Product Backlog

Tabla

Descripción generada automáticamente con confianza media

### Ciclo de Sprints del proyecto

****

### Sprint Backlog

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Story ID** | **Story name** | **Status** | **Size** | **Sprint** | **Priority** | **Story Type** | **Comments** |
| 1 | Desarrollar Product Backlook Priorizado | Done | 3 | 0 | 1 | Documentacion | Se Desarrolla Product Backlog Priorizado |
| 2 | Construir Repositorio (Github) donde integraran el Trabajo | Done | 8 | 0 | 1 | Documentacion | Se Construye Repositorio (Github) en Proyecto de Equipo, El Proyecto tiene tres fases (To Do, In Progress, Done) |
| 3 | Construir Archivo Gestion Configuracion | Done | 20 | 0 | 1 | Documentacion | Stack: MEVN Front: Vue Back: JavaScript, Express, Node BD: Mongo |
| 4 | Construir Historias de Usuario a desarrollar en Sprint No.1 | Ongoing | 13 | 0 | 1 | Documentacion | Construccion Historias de Usuario Spring 1 |
| 5 | Creacion y configuracion cuenta Devops (SrumBoard y tareas) | Ongoing | 13 | 0 | 1 | Documentacion | Configuración inicial DevOps |
| 6 | Construccion Incial de MockUps | Planned | 13 | 0 | 1 | Documentacion | Diseño mockups |
| 7 | Construccion UML casos de uso | Planned | 13 | 0 | 1 | Documentacion | Desarrollo inicial UML casos de uso de la aplicación |
| 8 | Documento IEEE 29148 | in progress | 8 | 0 | 2 | Documentacion | Llenado preliminar |

### Historias de usuario (Tareas y Subtareas)

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **ID:** 1 | **Nombre:** Crear interfaz de Consulta de contactos |
| **Prioridad en Negocio:** Alta | **Iteración Asignada: 3** |
| **Rol:** Yo como creador de la interfaz gráfica | |
| **Funcionalidad**: Necesito crear interfaz de consulta | |
| **Criterio de Aceptación:** ● Interfaz intuitiva para el usuario (que no necesite explicación) ● Interfaz facil de usar | |
|  |  |
|  |  |
| **Historia de Usuario** | |
| **ID:** 2 | **Nombre:** Crear link para acceder a consultar los contactos |
| **Prioridad en Negocio:** Alta | **Iteración Asignada: 3** |
| **Rol:** Yo como creador del link | |
| **Funcionalidad**: Necesito crear un link para consultar los contactos | |
| **Criterio de Aceptación:** ● Link en la vista de consulta ● Al hacer click en el link mostrar la pagina de consulta de contactos | |
|  |  |
|  |  |
| **Historia de Usuario** | |
| **ID:** 3 | **Nombre:** Crear link para acceder agregar los contactos |
| **Prioridad en Negocio:** Alta | **Iteración Asignada: 3** |
| **Rol:** Yo como creador del link | |
| **Funcionalidad**: Necesito crear un link para agregar los contactos | |
| **Criterio de Aceptación:** ● Link en la vista de Agregar contacto ● Al hacer click en el link mostrar la pagina de agregar contacto | |
|  |  |
|  |  |
| **Historia de Usuario** | |
| **ID:** 4 | **Nombre:** Crear interfaz de Agregar de contactos |
| **Prioridad en Negocio:** Alta | **Iteración Asignada: 3** |
| **Rol:** Yo como creador de la interfaz gráfica | |
| **Funcionalidad**: Necesito crear interfaz para agregar los contactos | |
| **Criterio de Aceptación:** ● Interfaz intuitiva para el usuario (que no necesite explicación) ● Interfaz facil de usar | |
|  |  |
|  |  |
| **Historia de Usuario** | |
| **ID:** 5 | **Nombre:** Crear casillar de texto para agregar información de contactos (Nombre, teléfono y email) |
| **Prioridad en Negocio:** Alta | **Iteración Asignada: 3** |
| **Rol:** Yo como creador de la interfaz gráfica | |
| **Funcionalidad**: Necesito crear interfaz para agregar los contactos | |
| **Criterio de Aceptación:** ● Casillas de texto para agregar información de contactos | |
|  |  |
|  |  |
| **Historia de Usuario** | |
| **ID:** 6 | **Nombre:** Crear botón para agregar contactos |
| **Prioridad en Negocio:** Alta | **Iteración Asignada: 3** |
| **Rol:** Yo como creador del botón | |
| **Funcionalidad**: Necesito crear un botón para agregar la información del contacto | |
| **Criterio de Aceptación:** ● Contacto agregado | |
|  |  |
|  |  |
| **Historia de Usuario** | |
| **ID:** 7 | **Nombre:** Crear botón para cancelar la operación de agregar contactos |
| **Prioridad en Negocio:** Alta | **Iteración Asignada: 3** |
| **Rol:** Yo como creador del botón | |
| **Funcionalidad**: Necesito crear un botón para cancelar el registro de la información del contacto | |
| **Criterio de Aceptación:** ● Cancelación del registro de la información que se deseaba agregar | |
|  |  |
|  |  |
| **Historia de Usuario** | |
| **ID:** 8 | **Nombre:** Crear botón para actualizar la información de los contactos registrados |
| **Prioridad en Negocio:** Alta | **Iteración Asignada: 3** |
| **Rol:** Yo como creador del botón | |
| **Funcionalidad**: Necesito crear un botón para actualizar la información registrada | |
| **Criterio de Aceptación:** ● Mostrar botón de actualizar en la pagina de Agregar los contactos ● Visualización de la información actualizada | |
|  |  |
|  |  |
|  |  |
| **Historia de Usuario** | |
| **ID:** 9 | **Nombre:** Crear botón para eliminar contactos registrados |
| **Prioridad en Negocio:** Alta | **Iteración Asignada: 3** |
| **Rol:** Yo como creador del botón | |
| **Funcionalidad**: Necesito crear un botón para eliminar la información registrada | |
| **Criterio de Aceptación:** ● Mostrar botón de eliminar en la pagina de Agregar los contactos● Eliminación del contacto de la página | |

### Mecánica de organización del grupo. (Reuniones, evidencias/artefactos)

La forma de trabajar para poder llevar a cabo la realización de nuestro producto final de manera clara y ordenada con nuestros compañeros s ebaso principalmente en el establecimiento de tareas por nuestro grupo de whatsapp y posteriormente al conocimiento de las tareas de cada uno, dichas tareas se veían reflejadas en el DevOps de azure donde se asignaban a cada uno las tareas mediante envio de notificaciones al correo electrónico como se puede ver en las siguientes imágenes:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

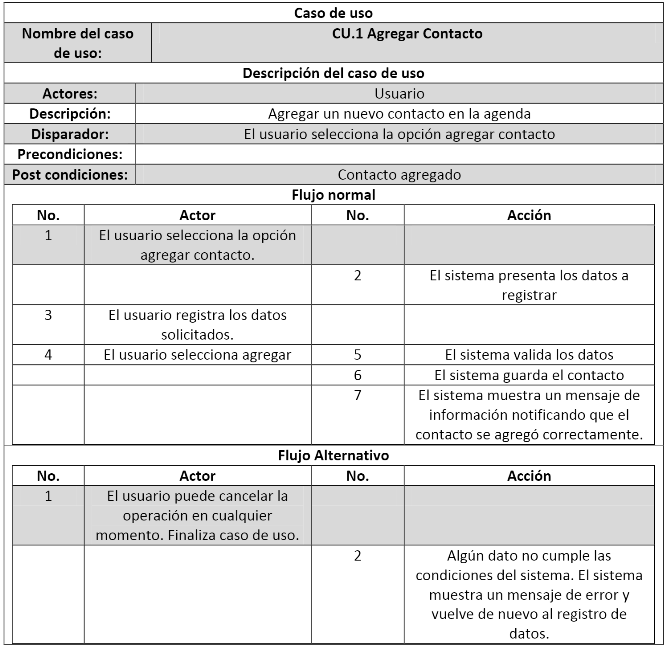
Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

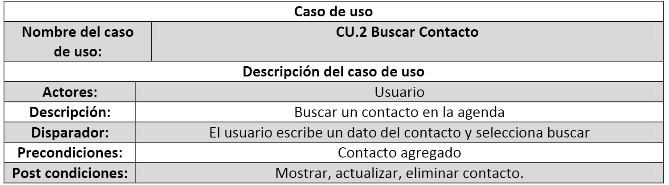
Descripción generada automáticamente

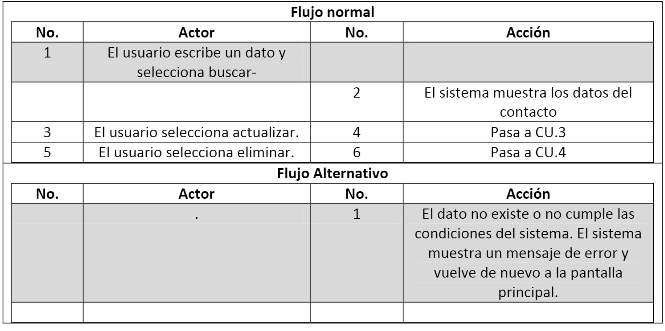
Interfaz de usuario gráfica, Aplicación, Teams

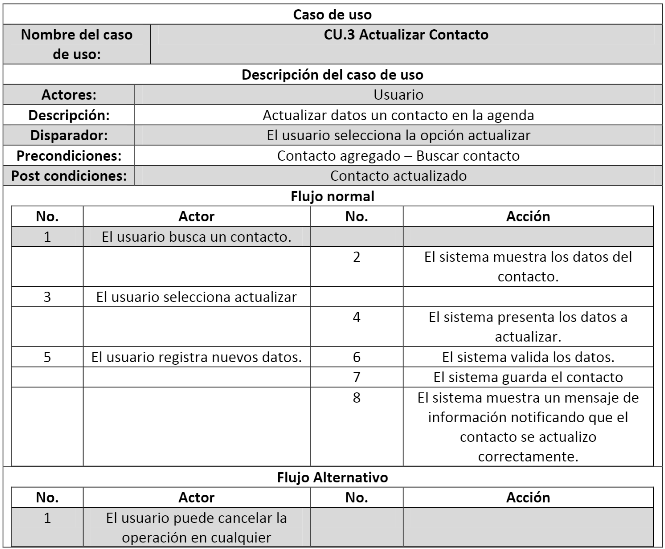
Descripción generada automáticamente

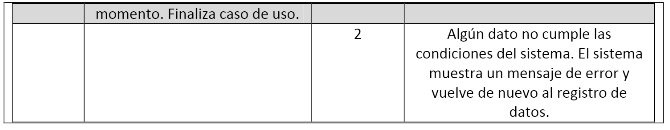
## Modelo de Requerimientos

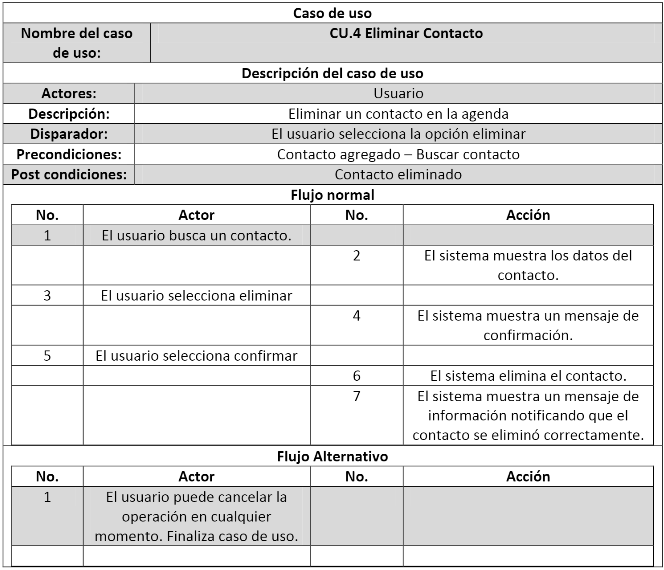










Has

### Modelo de Casos de Uso

Diagrama

Descripción generada automáticamente

# Descripción del diseño

## Interfaz gráfica (Mockups)

Inicialmente cuando se desarrollo la idea del proyecto se planteó crear una interfaz de usuario como se muestra a continuación:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media Interfaz de usuario gráfica

Descripción generada automáticamente

Posteriormente se empezó a realizar un diseño del producto según lo que se trabajó en clase por lo que el diseño final fue el siguiente:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

# Gestión de la configuración

Dicha agenda debe permitir agregar, modificar consultar y eliminar dichos campos mediante la selección de botones de agregar, eliminar o modificar que permitirán realizar dichas operaciones que actúan directamente sobre una base de datos la cual se gestionó con ayuda de las aplicaciones MongoDS, MongoCompass y Mongo Atlas.

Las pruebas iniciales de la interacción del sistema en el servidor local se realizaron mediante la API Postman que es una plataforma que permite la construcción y uso de APIs.

El alcance planteado para esta aplicación que puede ser utilizada para organizar contactos y tener a la mano información importante como dirección, correo, teléfono, redes sociales, etc; es la de poderse desplegar en la web y que sirva como un proyecto para poder comprender la interacción entre bases de datos, backend y frontend para el desarrollo futuro de proyectos más avanzados.

Dicha aplicación debe lograr responder solicitudes para poder crear, modificar, eliminar y visualizar información mediante una interfaz de usuario amigable e intuitiva.

Nuestra aplicación esta alojada en un repositorio Github

Captura de pantalla de un celular

Descripción generada automáticamente

Y también se gestionó mediante el uso de Azure DevOps

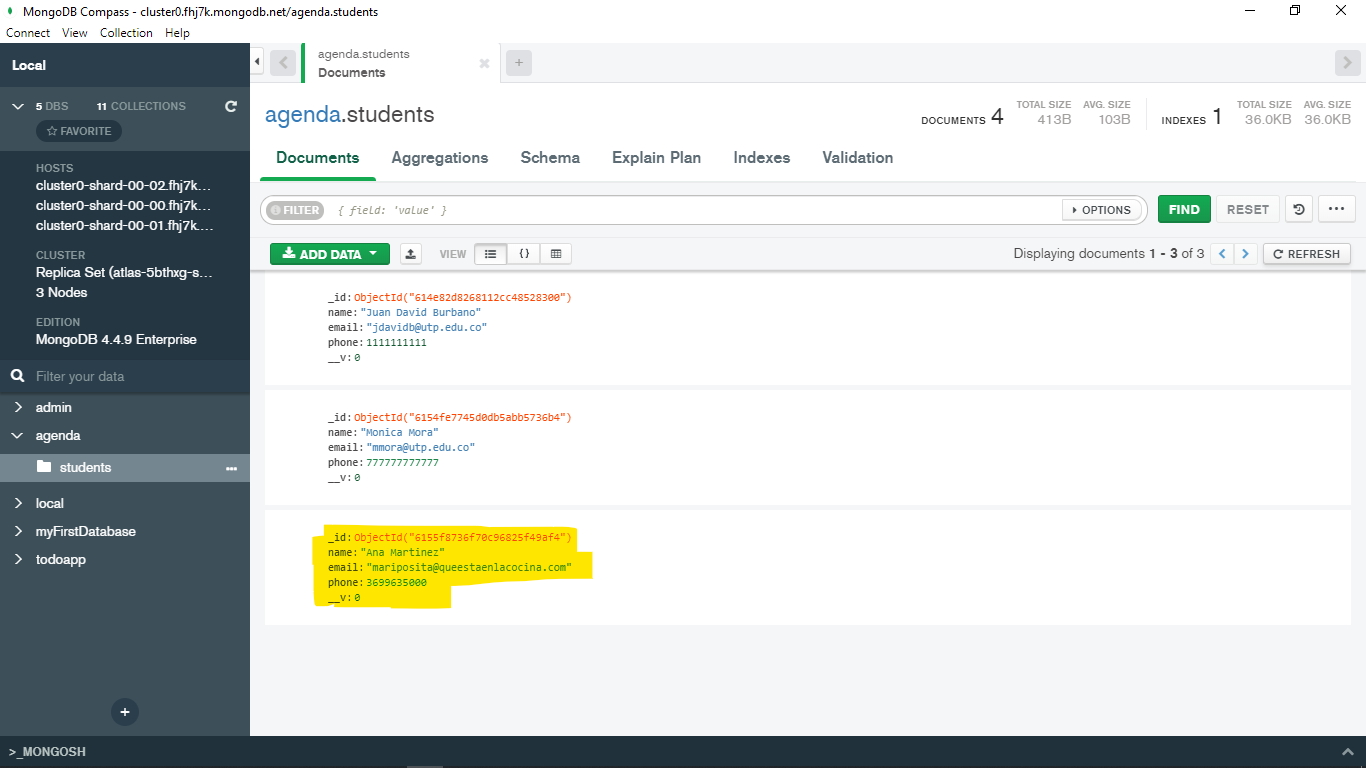
Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

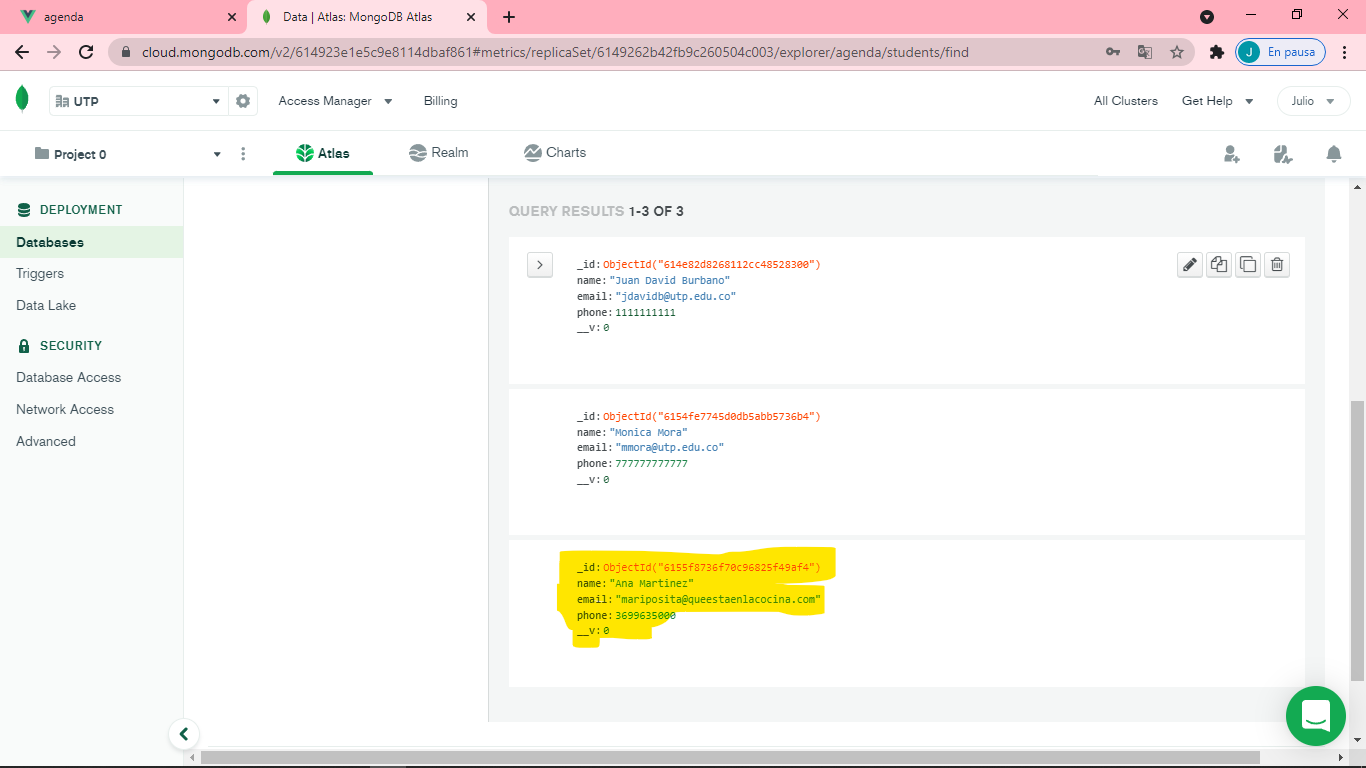
Descripción generada automáticamente

# Pruebas

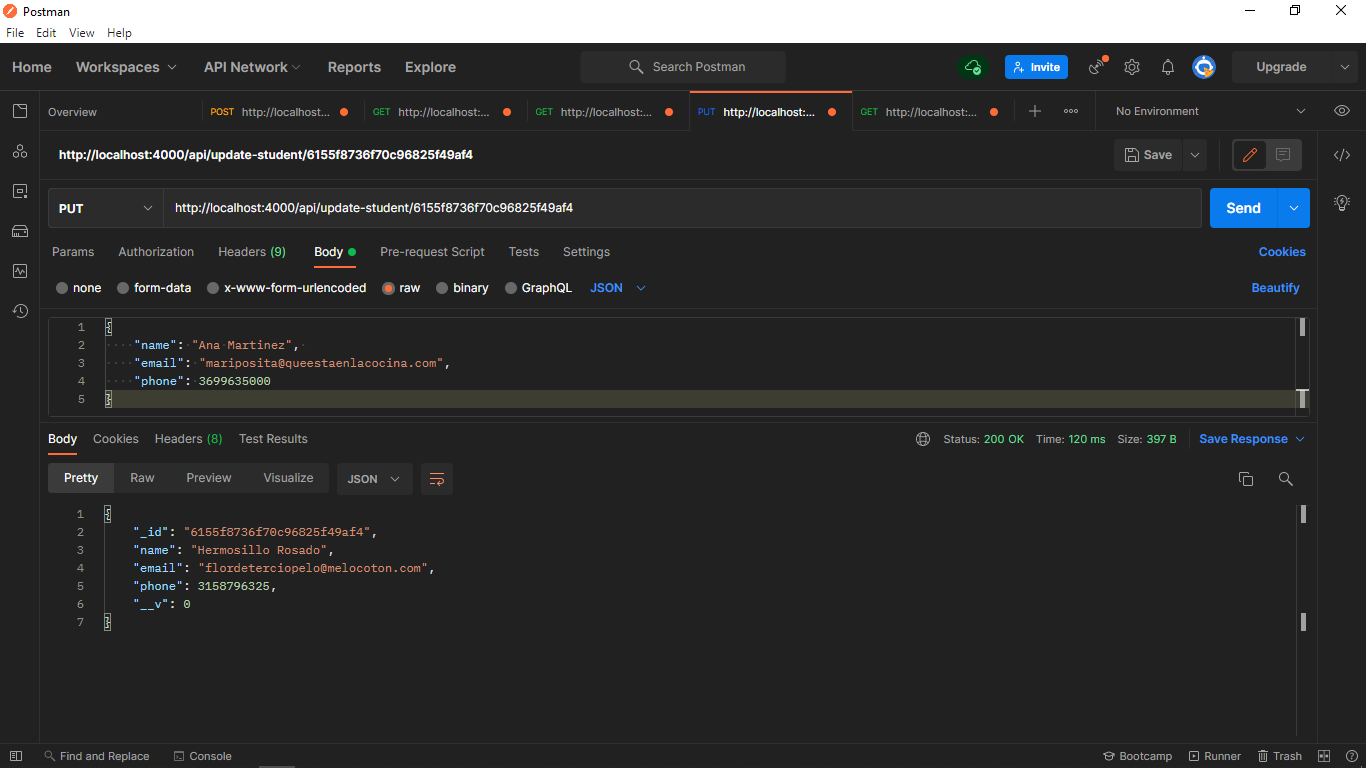
## Descripción de pruebas unitarias

A continuación se muestran las pruebas de conexión a base de datos MongoAtlas y mongoDB.



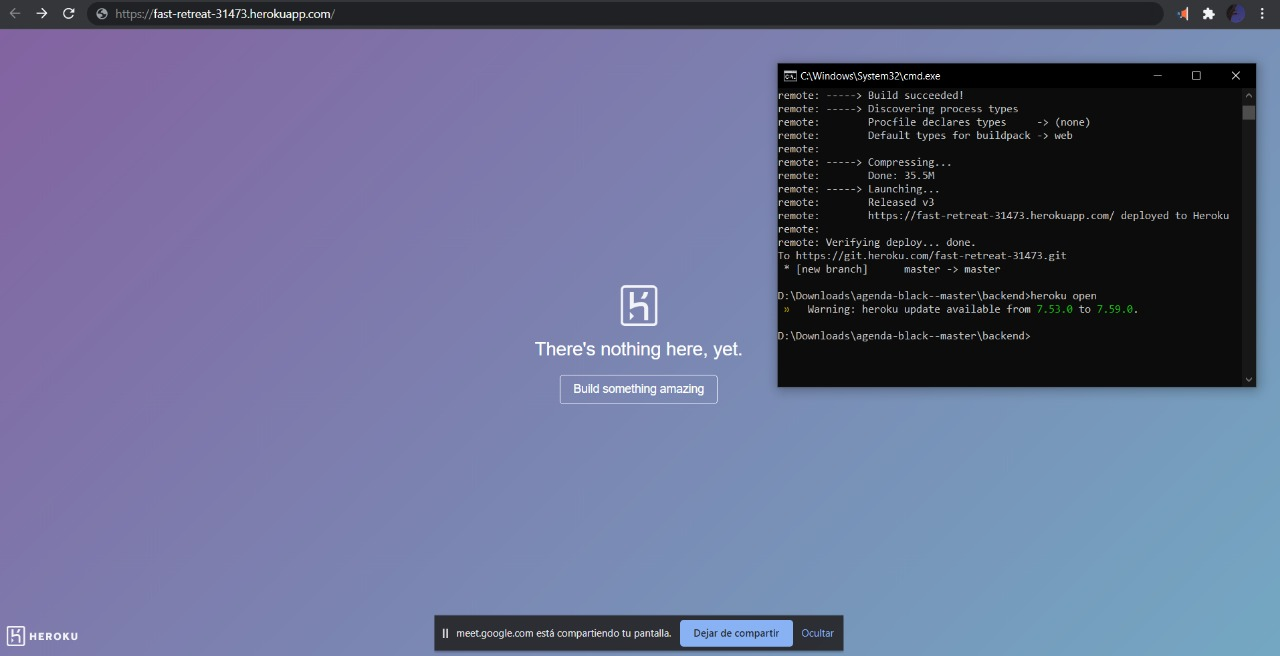
****

Posteriormente podemos ver las pruebas que se realizaron usando Postman para obtener y enviar mensajes.



## Descripción de pruebas de aceptación

En este apartado se muestra como se desplegó la aplicación al servidor web mediante Heroku, sin embargo, hubo problemas a la hora del despliegue y se obtuvo el siguiente error.



# Glosario

Arquitectura de Información

La Arquitectura de Información se encarga de efectuar la planificación estratégica previa a la creación de un website.

ASCII

Conocido como American Standard Code for Information Exchange, el ASCII es un estándar para relacionar un código específico a los caracteres tanto del alfabeto, números y signos que componen la información textual que se visualiza en la Web

Aside

Etiqueta HTML introducida en HTML5, sirve para definir un bloque de contenido generalmente relacionado con el bloque que se encuentra a su lado; se aprovecha para la creación de barras laterales. Representa un fragmento completo de contenido independiente del contenido principal de la página. Por ejemplo, puede usar <aside> para generar una barra lateral con material relacionado o con enlaces junto a un artículo principal. También se puede usar en bloques publicitarios.

ASP

Active Server Pages. Páginas activas de servidor creadas mediante una tecnología de páginas dinámicas que permite alterar su contenido o incluso su forma, sin necesidad de editar o rescribir su código. Se trata de páginas con etiquetas de lenguaje HTML normales a las que se añaden una serie de programas.

Atributo

Es una parte de un elemento que añade información adicional a ese elemento.

Avatar

Personalidad o identidad visual que se atribuyen algunos usuarios de Internet, ya sea en algún chat, juegos, etc. Un avatar es un facsímil gráfico que se puede utilizar en las habitaciones de discusión virtual o chat rooms y que permite jugar un rol e interactuar con gente en línea. El término se hizo popular en la novela Snow Crash de Neal Stephenson.

Base de datos

Datos almacenados en una computadora que pueden ser accesados y manipulados fácilmente; piense en un gran gabinete físico con muchas etiquetas con archivos de información relevante.

Big Data

El Big Data trata trabaja con una gran masa de datos, los ordena y los estudia para obtener ideas que lleven a mejores movimientos comerciales. Su estudio y análisis ayuda a los negocios a aprovechar sus datos para identificar nuevas oportunidades.

Bing

Buscador (motor de búsqueda, search engine) de la empresa Microsoft.

Bit

Dígito binario. Unidad mínima de almacenamiento de la información cuyo valor puede ser 0 ó 1; o bien verdadero o falso.

Bitmap

Son imágenes compuestas por pixeles, lo cual significa que si uno la aumenta de tamaño la computadora trata de "rellenarla" con más pixeles y el resultado es una imagen de pobre calidad.

Buscadores de Internet

Son compañías que utilizan software elaborado para visitar todos los sitios de la Web regularmente, indexar la información que encuentra y utilizando sus propios parámetros, despliegan la relevancia de esas páginas cuando un usuario busca información relevante al gremio que pertenecen. Los tres buscadores más grandes son Google, Yahoo y Bing.

Busqueda orgánica o natural

La mayoría de motores de busqueda ofrecen dos tipos de resultados de busqueda a sus clientes: resultados de pago (normalmente en la parte superior o lateral) y resultados orgánicos o naturales. Mientras que los resultados de pago pueden hacer que tu sitio Web se coloque en una posición alta para una palabra clave específica (porque has pagado por ello), la mayoría de clientes consideran estos resultados poco mejores que la publicidad, y generalmente los evitarán en beneficio de los resultados naturales o orgánicos.

<button> (Botón)

El elemento <button> permite originar un botón sobre el que se puede hacer clic dentro de un formulario, con el contenido que desee dentro de él (por ejemplo, puede colocar una frase o una imagen entre sus etiquetas de inicio y fin). Al igual que con cualquier otro control de formulario, debe proporcionar un nombre único y un valor que el formulario enviará cuando un visitante haga clic en el botón. Su contenido se coloca entre las etiquetas de apertura y cierre:

Cabecera (header)

Encabezamiento de una página o una sección de esta. En el caso de encabezamiento de página, se suele colocar el nombre de esta, un logotipo, menús de navegación que pueden ser desplegables, barras de búsqueda, entre otros. Si bien se pueden generar con bloques comunes, a partir de HTML5 se introduce una etiqueta específica para otorgarle un mayor significado semántico, lo que nos ayuda en cuestión de SEO.

Caché

Es la habilidad de un navegador o de un servidor para guardar copias de páginas Web en un disco duro con la ventaja de que carga más rápido dado que la información no debe de ser "pedida y entregada" por un servidor si no que apunta a un lugar en específico mucho más cercano.

Call to Action (CTA) o llamada a la acción

Un Call to Action o CTA (también llamado llamada a la acción) es un botón o enlace situado en nuestro site que busca atraer clientes potenciales y convertirles en clientes finales, normalmente a través de un formulario en una página de destino o landing page.  
  
Un CTA es el vínculo entre el contenido regular que el usuario consume y una página (landing page) con una oferta más interesante para nuestro usuario, que es lo suficientemente relevante e interesante para persuadirle a completar nuestro formulario.

Case sensitive

Case sensitive (en castellano sensible a las mayúsculas/minúsculas) es una expresión usada en jerga informática que se aplica a los textos en los que tiene alguna relevancia escribir un caracter en mayúsculas o minúsculas (ejemplo: los comandos en sistemas basados en UNIX, o el contenido de los archivos de texto). Lo contrario se denomina case insensitive.

CDATA

Permite introducir caracteres especiales si que sean procesados en documentos XML. <![CDATA[ Texto unicode. ]]>

CDN

Una red de distribución de contenido no es más que un servicio que te permite alojar una versión de tu sitio Web en varios servidores para mejorar la velocidad de carga de tu página Web. De manera sencilla, se almacena la información en estos servidores y, dependido de dónde se haga la consulta, se redirige la petición al servidor más cercano, esto ahorra recursos en el servidor original, porque disminuye el número de solicitudes que se hacen en un mismo periodo de tiempo.

CEO

Siglas del inglés Chief Executive Officer (o director ejecutivo).

CERN

Conseil Europeen pour la Recherche Nucleaire. Laboratorio Europeo de Física de Partículas, en Génova, Suiza. Fue donde nació el world wide web en 1991, basado en la propuesta de Tim Berners-Lee.

Class

Selector CSS que nos permite especificar propiedades de estilo para un grupo de elementos. En HTML se representa como class, en CSS se representa con un símbolo de punto "."

Clear

Propiedad CSS que especifica que no se deben mostrar elementos flotantes del lado especificado con el selector: left (izquierda), right (derecha) o both (ambos).

Dirección IP

Una dirección IP es la identificación numérica de un ordenador conectado a Internet. Estas son escritas a menudo como 4 grupos de 3 números (IPv4). Los nombres de dominio utilizan las direcciones IP como su dirección de forma que los navegadores Web puedan encontrarlos en Internet.

Diseñador web

Un diseñador web es una persona que diseña la parte visible de la página web: concepto, imágenes, estilo, etc. definiendo así parte de los requisitos de uso de ésta. Normalmente hacen uso de editores WYSIWYG más que escribir el código HTML directamente.

DOM

Document Object Model o DOM es el API que proporciona un conjunto estándar de objetos para representar documentos HTML y XML, un modelo estándar sobre cómo pueden combinarse dichos objetos, y una interfaz estándar para acceder a ellos y manipularlos. A través del DOM, los programas pueden acceder y modificar el contenido, estructura y estilo de los documentos HTML y XML, que es para lo que se diseñó principalmente.

Dominio

Un dominio de internet es un nombre de equipo que proporciona nombres más fácilmente recordados en lugar de la IP numérica. Por medio de los dominios podemos encontrar páginas fácilmente. Por ejemplo con sólo escribir google.com entramos a la web de Google. El dominio representa nuestro nombre en internet y es la forma por el cual las personas llegarán a nuestro sitio.

Dominio de Alto Nivel

Caracteres que figuran en último lugar en una dirección de Internet los cuales van precedidos de un punto (.). Por ejemplo, en los dominios de alto nivel son "com" , "es" e "info".

Estándar web

Estos son las especificaciones y reglas para crear y diseñar páginas web, creados y mantenidos por el W3C. Algunas de las más importantes son: HTML: tales como XHTML, HTML 4, HTML 5 CSS: las versiones CSS 1, CSS 2 y CSS 3 XML: como por ejemplo las especificaciones RSS, XSLT, XPath y XML Schemas

Estilo de fuente

Variaciones estilísticas de una fuente, como italicas, subrayado, negrita... En CSS, se refiere únicamente al estado italica de la fuente.

Firewall

(pared a prueba de fuego) Conjunto de programas de protección y dispositivos especiales que ponen barreras al acceso exterior a una determinada red privada. Es utilizado para proteger los recursos de una organización de consultas externas no autorizadas.

Flame

Mensajes de correo insultantes, en los que se ataca más a la persona a la que se intenta responder que a los argumentos que esta ha expuesto. Por lo general son de carácter personal (aunque no son raros los dirigidos a grupos étnicos), a pesar de lo cual son enviados a grupos de noticias o listas de correo.