



UXIFY Workshop Software

Made by Uxify Studio

Adrián López Martín-Forero
Zahir Allonso Rivera Chacón
Jason Alejandro varas fabres

Índice

Uxify Workshop Software.....	1
Índice.....	2
INTRODUCCIÓN.....	3
Enunciado:.....	3
Presentación:.....	3
Motivo:.....	3
INICIO DEL PROYECTO.....	4
Planificación:.....	4
Herramientas:.....	4
DESARROLLO.....	6
Diseño:.....	6
Base de Datos:.....	7
Codificación PHP/Laravel:.....	7
Codificación JavaScript/Vue:.....	8
Despliegue con Docker:.....	8
Documentación:.....	8
Presentación:.....	8
Herramientas:.....	8
PROPUESTAS Y FUTURO.....	9
Objetivos Cumplidos:.....	9
Problemas:.....	9
¿Funciona?.....	9
Mejoras:.....	9
BIBLIOGRAFÍA.....	10

INTRODUCCIÓN

Enunciado:

Vuestra empresa ha sido escogida por el Departamento de Mecatrónica de Egibide para desarrollar su nueva aplicación web de Gestión de Mantenimiento por Ordenador (GMAO).

Egibide es un centro educativo de carácter Social dedicado a la enseñanza, con amplia oferta en la Formación Profesional. Necesita un software que gestione el mantenimiento de los equipos de la escuela, y pueda ser utilizado también en la enseñanza en los ciclos de Mantenimiento industrial. Hasta ahora la forma que tiene la empresa de operar es la:

- Los docentes/alumnado que trabajan en la escuela dan un parte de incidencias sobre las máquinas que usan(avería, mantenimiento necesario, etc.) a un operador.
- El responsable del taller anota en un cuaderno los detalles de la incidencia.
- El responsable de taller llama a los técnicos hasta que encuentra a uno disponible que pueda realizar las tareas de reparación o mantenimiento necesarias.
- Al resolver la incidencia, el técnico llama al responsable de taller, y este la marca como resuelta en su cuaderno.
- En ocasiones la incidencia no se puede resolver en el mismo momento (falta de algún material, servicio...) y esta se queda abierta. En estos casos, los técnicos tienen que avisar al responsable de taller, que marca la incidencia como pendiente en su cuaderno.
- Además de estas incidencias creadas por los docentes/alumnado, también existen mantenimientos preventivos que crean incidencias. Estos mantenimientos son periódicos. Hasta ahora los responsables de taller se encargaban de controlar cuando tenían que hacerse y asignarlos a los técnicos de la misma manera que el resto de incidencias.

El crecimiento de ocupación de taller que ha tenido la escuela en los últimos años ha hecho que el servicio de atención y gestión de las incidencias se vuelva una carga difícil de gestionar y que no asegura el correcto mantenimiento de los equipos.

Egibide quiere dar un salto tecnológico con la implantación de una nueva aplicación web que permita a los docentes/alumnado reportar incidencias y a la escuela gestionarlas de una forma más rápida y eficiente.

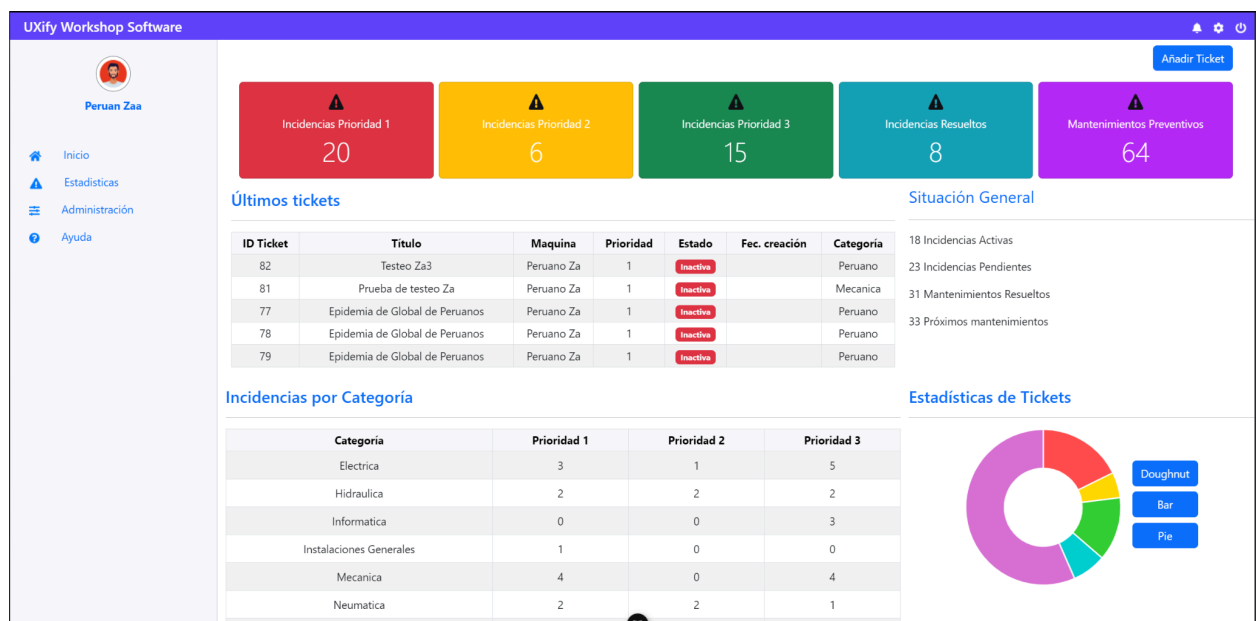
Además, de esta forma, el trabajo del responsable de taller podría reducirse. En lugar de esto necesitaran un administrador que se encargue de crear máquinas, docentes, alumnado, técnicos y mantenimientos preventivos con los que poder funcionar

Presentación:

Somos el Grupo Uxify Studio conformado por Adrián López Martín-Forero, Zahir Allonso Rivera Chacón y Jason Alejandro Varas Fabres. Nos dedicamos a desarrollar, actualizar y mejorar páginas web en las empresas .

Motivo:

Nuestra empresa fue escogida por el Departamento de Mecatrónica de Egibide para desarrollar su nueva aplicación web de Gestión de Mantenimiento por Ordenador (GMAO).



INICIO DEL PROYECTO

Planificación:

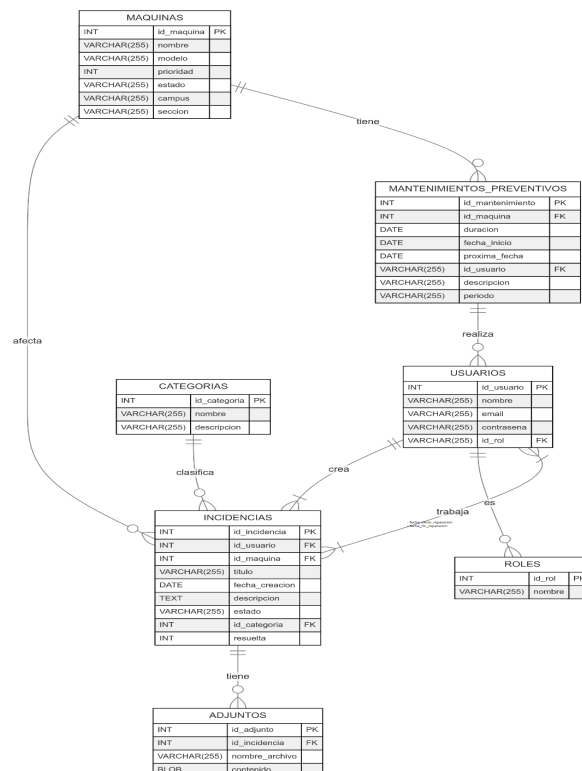
El primer día, al empezar con el proyecto, lo primero que hicimos fue una rápida y eficiente planificación de lo que tendríamos que hacer en las próximas semanas. Para eso nos ayudamos de unas herramientas. Usamos Trello para poder organizar las tareas y funcionalidades que tendríamos que hacer en la página web, también utilizamos diferentes programas para comunicarnos y hacer reuniones como Discord y Whatsapp.

Herramientas:

Las herramientas que Usamos en el transcurso del proyecto::

- **Comunicación:** Whatsapp, Discord
- **Planificación:** Trello
- **Diseño Gráfico:** Figma
- **Diseño Base de Datos:** Mermaid chart
- **Desarrollo de la Aplicación:** PHPStorm, Visual Studio Code, Xampp/Laragon
- **Ayuda:** Chat GPT

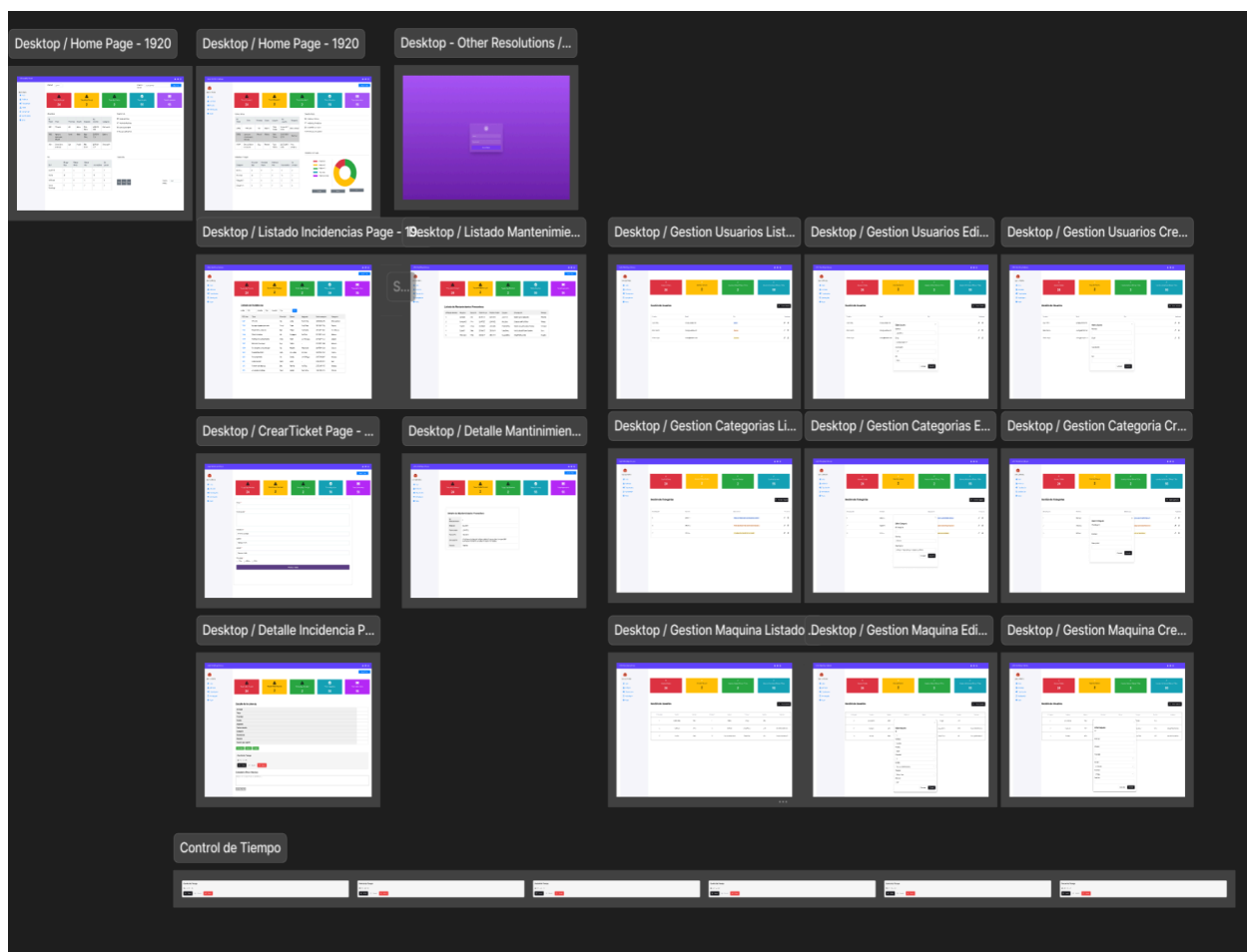
También tenemos el diseño de la base de datos que hicimos entre los 3 miembros del grupo:



DESARROLLO

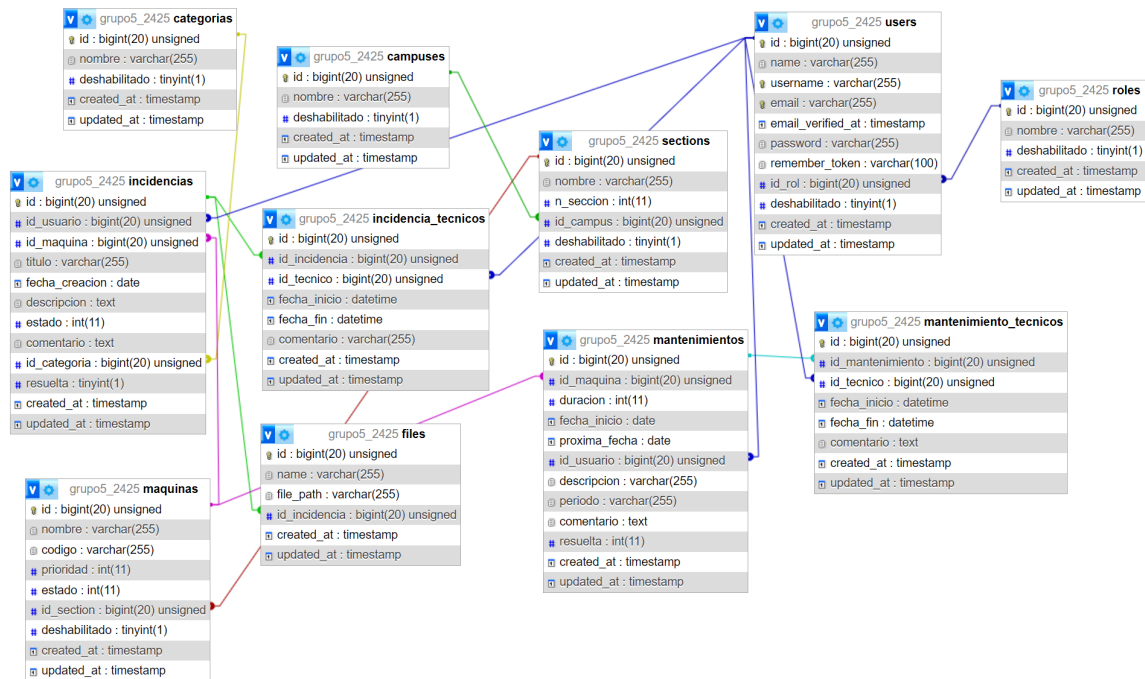
Diseño:

Una vez terminada la planificación inicial del proyecto comenzamos con la parte del diseño que tendría la página web, nosotros optamos por un diseño limpio pero muy completo , a su vez también elegimos una tipografía simple pero que combine bien con el resto de la página y que se ajuste a los gustos del cliente para el cual estábamos haciendo la página.



Bases de Datos:

La base de datos la planificamos y creamos con los 3 miembros del grupo ya que combinamos nuestros puntos de vista para poder tener un visión más amplia sobre cómo tratar el los datos de la página, a su vez también conseguimos coincidir en varios puntos y crear una base de datos bien estructurada y muy simple de entender y procesar.



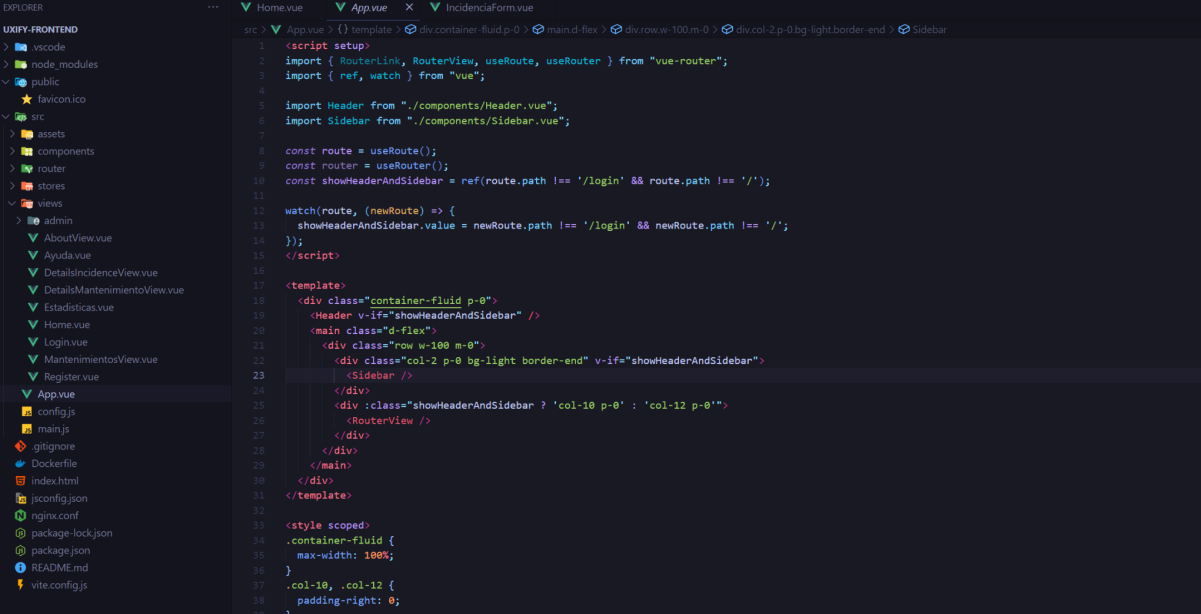
Codificación Laravel/PHP:

La codificación de laravel la comenzamos una vez creado el proyecto con los comandos respectivos y la creación de los migrate para la base de datos, el Laravel se encargará de de la parte del BackEnd donde tendríamos todo las llamadas a la base de datos en lo cuales estaría organizado con los modelos y los controladores. También contamos con una API la cual conectará con el FrontEnd además de la seguridad para la página creada solo para que el administrador/técnico pueda acceder a todas las funciones. El despliegue se llevará a cabo mediante Docker, el backend tiene su dockerfile.

```
71:
72:
73: Route::controller([Illuminate\Http\Controllers::class])>group(function () {
74:     Route::get('/incidencias', 'IncidenciaController@index')->middleware('auth:api');
75:     Route::post('/incidencias', 'IncidenciaController@store')->middleware('auth:api');
76:     Route::get('/incidencias/latest', 'IncidenciaController@getUltimasIncidenciasPorPrioridad')->middleware('auth:api');
77:     Route::get('/incidencias/prioridad', 'IncidenciaController@countIncidenciasPorPrioridad')->middleware('auth:api');
78:     Route::get('/incidencias/campus/{campus}', 'IncidenciaController@getIncidenciasByCampus')->middleware('auth:api');
79:     Route::get('/incidencias/section/{section}', 'IncidenciaController@getIncidenciasBySection')->middleware('auth:api');
80:     Route::get('/incidencias/{id}', 'IncidenciaController@show')->middleware('auth:api');
81:     Route::put('/incidencias/{id}/resuelta', 'IncidenciaController@marcarIncidenciaComoResuelta')->middleware('auth:api');
82:     Route::put('/incidencias/{id}/comentario', 'IncidenciaController@actualizarComentario')->middleware('auth:api');
83:     Route::get('/incidencias/{id}/comentario', 'IncidenciaController@getComentarios')->middleware('auth:api');
84:     Route::get('/incidencias/count', 'IncidenciaController@countIncidenciasStatus')->middleware('auth:api');
85:     Route::get('/incidencias/latest', 'IncidenciaController@getUltimasIncidenciasPorPrioridad');
86: });
87:
88: Route::get('/roles', [RoleController::class, 'index']);
89:
90: Route::controller([Illuminate\Auth\AuthController::class])>prefix('auth')>group(function () {
91:     Route::post('/login', 'AuthController@login');
92:     Route::post('/register', 'AuthController@register')->middleware('auth:api', 'admin');
93:     Route::post('/logout', 'AuthController@logout')->middleware('auth:api');
94:     Route::post('/refresh', 'AuthController@refresh')->middleware('auth:api');
95:     Route::get('/me', 'AuthController@me')->middleware('auth:api');
96: });
97:
98: Route::controller([Illuminate\Http\Controllers::class])>group(function () {
99:     Route::get('/users', 'UserController@index')->middleware('auth:api', 'admin');
100:    Route::get('/users/count', 'UserController@countUsers')->middleware('auth:api', 'admin');
101:    Route::post('/users/store', 'UserController@store')->middleware('auth:api', 'admin');
102:    Route::put('/users/{id}', 'UserController@update')->middleware('auth:api', 'admin');
103:    Route::put('/users/{id}/enable', 'UserController@enable')->middleware('auth:api', 'admin');
104:    Route::put('/users/{id}/disable', 'UserController@disable')->middleware('auth:api', 'admin');
105:});
```

Codificación Vue/JavaScript:

La codificación de Vue es el FrontEnd y es la que se encarga del diseño de la página, la organización de las funcionalidades y el despliegue de esta con ayuda del Docker. Esta se conforma por varias carpetas pero las más usadas son assets en la que se guarda el las imágenes que se usarán en el proyecto, la siguiente sería la carpeta de componentes la cual contiene las partes de código de diferentes páginas y que contienen la mayoría de las funcionalidades, la siguiente sería router la cual solo es una pero contiene todos los enlaces para ir a las diferentes páginas en el proyecto, después seguiría stores que son llamada hechas con Pinea sería otra forma de llamar al backend para usar los datos en las vistas y por último tenemos las views la cuales son las páginas principales que contiene los componentes y que son las bases de las vistas de las páginas.



```
1 <script setup>
2 import { RouterLink, RouterView, useRoute, useRouter } from "vue-router";
3 import { ref, watch } from "vue";
4
5 import Header from "../components/Header.vue";
6 import Sidebar from "../components/Sidebar.vue";
7
8 const route = useRoute();
9 const router = useRouter();
10 const showHeaderAndSidebar = ref(route.path !== '/login' && route.path !== '/');
11
12 watch(route, (newRoute) => {
13   showHeaderAndSidebar.value = newRoute.path !== '/login' && newRoute.path !== '/';
14 });
15 </script>
16
17 <template>
18   <div class="container-fluid p-0">
19     <Header v-if="showHeaderAndSidebar" />
20     <main class="d-flex">
21       <div class="row w-100 m-0">
22         <div class="col-2 p-0 bg-light border-end" v-if="showHeaderAndSidebar">
23           <Sidebar />
24         </div>
25         <div :class="showHeaderAndSidebar ? 'col-10 p-0' : 'col-12 p-0'">
26           <RouterView />
27         </div>
28       </div>
29     </main>
30   </div>
31 </template>
32
33 <style scoped>
34   .container-fluid {
35     max-width: 100%;
36   }
37   .col-10, .col-12 {
38     padding-right: 0;
39   }
40 </style>
```

Despliegue con Docker:

Documentación:

La documentación la comenzamos a mediados del proyecto cuando ya teníamos una gran parte del código con algunas funcionalidades, cuando empezamos la documentación pudimos organizar mejor lo que ya teníamos para tener mejores ideas de manera ordenada además ya que la comenzamos antes del la fase final del proyecto y pudimos abordar esta fase de manera más completa.

Presentación:

La presentación está hecha para tener un diseño que combine con la página y a su vez ser llamativa y entretenida para captar la atención de las personas, y estar dotada de una cantidad moderada de información lo cual es necesario si queremos explicar todo de manera clara y que las personas puedan entender.

Herramientas:

Las herramientas que al final terminamos usando generalmente a lo largo del proyecto:

- **Comunicación:** Whatsapp, discord
- **Planificación:** Trello
- **Diseño Gráfico:** Figma
- **Diseño Base de Datos:** Mermaid chart
- **Desarrollo de la Aplicación:** PHPStorm, Xampp, GitHub
- **Ayuda:** Chat GPT, Copilot, Google, StackOverFlow
- **Documentación y Presentación:** Google Doc, Canva



Uxify Studio





PROPUESTAS Y FUTURO

Objetivos Cumplidos:

Cumplimos con todo los requerimientos básicos que nos podio el cliente:

- Agregar Incidencias y mantenimientos
- Poder crear, eliminar (deshabilitar) y editar máquinas, usuarios, secciones, categorías y campus.
- Tenemos la creación de mantenimientos al final otro mantenimiento
- También tenemos un contador para ver cuánto tiempo los técnicos trabajan en las incidencias.

Problemas:

Tuvimos algunos problemas al desplegar el proyecto en la red pero al final pudimos conseguirlo.

¿Funciona?

Si funciona todo muy bien y completamente con buena optimización y de manera muy práctica e intuitiva para que el cliente pueda manejar la aplicación sin problemas.

Mejoras:

La mejora más notable e interesante es el apartado de estadísticas en el cual encontramos mucha información de manera gráfica y ordenada.

BIBLIOGRAFÍA

Para hacer este proyecto nos estuvimos buscando en diversas fuentes información y ayuda de los profesores para poder cumplir con este reto en su totalidad, al final lo conseguimos.

Por eso en primer lugar quisiera agradecer a todos los profesores Maider, Ines y Nieves por ayudarnos a completar este proyecto con los documentos y la orientación que nos brindaron.

También que gracias a la información que conseguimos encontrar en internet y con diversidad herramientas que usamos en el proyecto una de las más importantes fueron:

- ChatGPT: <https://chatgpt.com/gpts>
- Copilot: <https://copilot.microsoft.com/onboarding>
- StackOverFlow: <https://stackoverflow.com/>
- Vue Docs: <https://vuejs.org/guide/introduction.html>
- Laravel Docs: <https://laravel.com/docs/11.x/readme>

Además de inteligencias artificiales también usamos librerías diversas para algunas funcionalidades como para las gráficas.