

Hito 3 Proyecto E-work



Universidad Andrés Bello®

Nombres: Fabio Urrea
Felipe Villalobos

Rut:18299907-5

índice de tablas

Tablas	3
Resumen.....	4
Distribución de incidentes según tipo	5
Distribución mensual de Tickets según tipo de incidente reportado	5
Tipo de incidente: operaciones de ciberseguridad de CSIRT de gobierno	6
Tipo de incidente de código malicioso.....	6
Tipo de incidente: seguridad de contenido de la informacion	7
Tipo incidente de fraude	7
Tipo de incidente: recopilación de información	8
Tipo de incidente: Intrusión	9
Tipo de incidente disponibilidad	9
Tipo de incidente: Intentos de intrusión.....	10
Tipo de incidente: contenido abusivo.....	11
Tipo de incidente: otros.	11
Tickets emitidos a instituciones públicas y privados	12

índice de ilustraciones

Resumen anual ticket y tipos de incidentes	4
Distribución de incidentes según tipo	5
Distribución mensual de tickets según tipo de incidente reportado.....	5
Tipo de incidente: operaciones de ciberseguridad de CSIRT de gobierno	6
Tipo de incidente de código malicioso.....	7
Tipo de incidente: seguridad de contenido de la información	7
Tipo incidente de fraude	8
Tipo de incidente: recopilación de información	8
Tipo de incidente: intrusión	9
Tipo de incidente disponibilidad	10
Tipo de incidente: intentos de intrusión.....	10
Tipo de incidente: contenido abusivo.....	11
Tipo de incidente:otros.	12
Tickets emitidos a instituciones públicas y privados	12
Testing Driven Development	17

índice de contenido

introducción	13
--------------------	----

Proyecto	14
¿Quiénes son nuestros potenciales usuarios y que requieren?	13
Planificación	14
Visión del producto	17
Visión del producto en ambiente operativo	17
¿Cómo construimos este producto?	17
¿Cómo aceleramos nuestro desarrollo?	23
Mitigación de riesgo.....	23
Resultado de la mitigación de riesgo	24
Conclusiones	32
Referencias.....	32

Tablas

A. Resumen anual de ticket y tipos de incidentes

- Podemos deducir que durante el año 2021 los mayores ataques fueron de vulnerabilidad del sistema con 13371 casos en el 2021, además la tabla nos muestra que el índice menor de casos lo muestra el contenido abusivo con 131 caos en el año. Esta tabla nos muestra que la mayor cantidad de ataques de diferentes tipos se dio en el mes de mayo con 2429 casos.

N°	Tipos de Ticket	Código	2020	2021
1	Vulnerabilidad	9V00	3.353	13.371
2	Disponibilidad	6D00	624	2.979
3	Seguridad de los contenidos de información	7S00	1.231	1.486
4	Recopilación de Información	3R00	5.504	1.160
5	Fraude	8F00	2.211	1.153
6	Operaciones Ciberseguridad CSIRT	19OC	662	646
7	Otros	11O00	NA	586
8	Código Malicioso	2C00	1.468	476
9	Intentos de Intrusión	4I00	50	320
10	Intrusión	5I00	111	165
11	Contenido Abusivo	1A00	107	131
Total			15.321	22.473

Incidentes por mes durante 2021													
Código	Ene	Feb	Mar	Abr	May	Jun	Jul	Ago	Sep	Oct	Nov	Dic	Total
9V00	239	921	1.402	886	1.326	870	1.717	1.154	753	1.158	1.288	1.657	13.371
6D00	3	44	5	0	330	369	376	447	391	340	337	337	2.979
7S00	176	124	77	153	88	125	111	146	149	131	167	39	1.486
3R00	372	324	132	225	85	8	0	3	1	6	3	1	1.160
8F00	82	65	40	84	99	136	81	151	105	78	118	114	1.153
19OC	91	68	173	110	204	0	0	0	0	0	0	0	646
11O00	0	0	0	0	0	59	53	76	86	115	106	91	586
2C00	125	106	79	43	24	24	3	5	15	21	16	15	476
4I00	6	2	7	13	238	0	0	3	1	2	0	48	320
5I00	21	21	23	23	15	0	0	5	1	1	0	55	165
1A00	22	16	23	35	20	2	0	6	2	1	1	3	131
Total	1.137	1.691	1.961	1.572	2.429	1.593	2.341	1.996	1.504	1.853	2.036	2.360	22.473

B. Distribución de incidentes según tipo:

- El incidente de vulnerabilidad es el más recurrente representando un 59% del total de los ataques cibernéticos.

Debido a que las políticas de seguridad son tan deficientes están representan un 3% como operaciones de ciber seguridad (CSIRT)

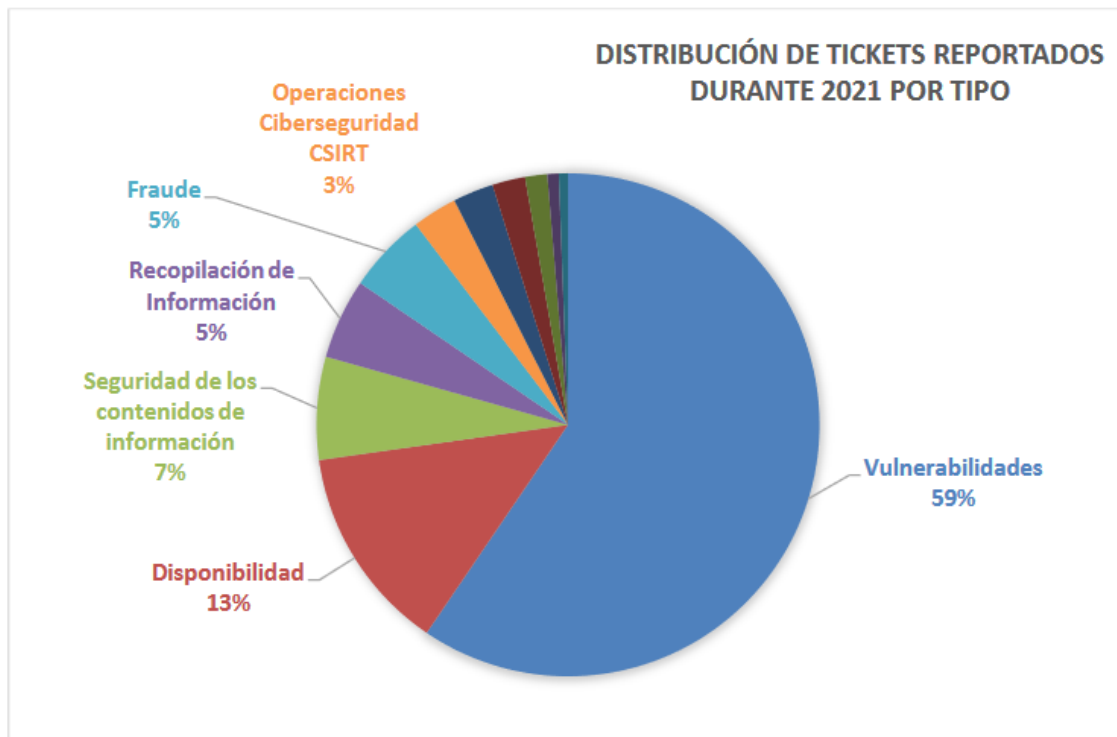


Imagen 1.- Distribución de tickets reportados durante 2021 por tipo.

C. Distribución mensual de Tickets según tipo de incidente reportado

- El mayor incidente reportado es el de vulnerabilidad con 1717 casos en septiembre y la misma cantidad de casos se produce en enero.

3.1.1 Distribución mensual de tickets según tipo de incidente reportado

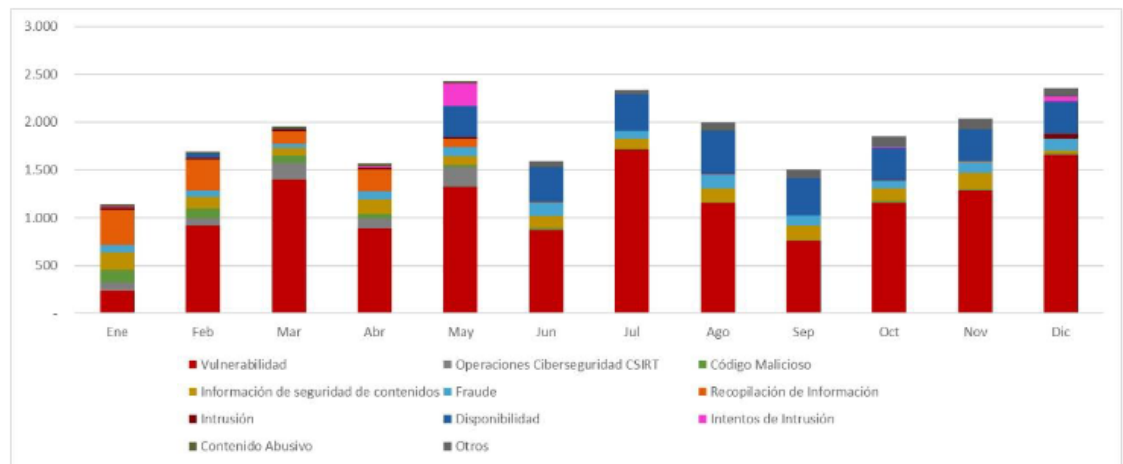


Imagen 2.- Distribución mensual de tickets por tipo.

D. Tipo de incidente: operaciones de ciberseguridad de CSIRT de gobierno

- En mayo se produce un peak de ataques a la red de conectividad del estado (RCE) especialmente al bloqueo de IP.
-
-

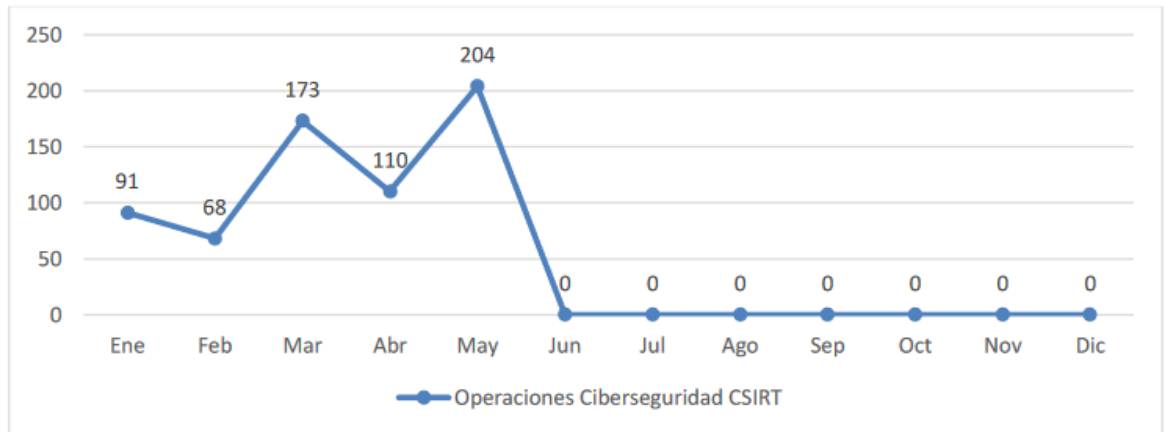


Imagen 4.- Distribución mensual de incidentes del tipo Operaciones Ciberseguridad CSIRT.

E. Tipo de incidente de código malicioso:

- Este tipo de ataque conocido como software o malware que fue disminuyendo notablemente desde enero que tenía 125 casos hasta llegar a diciembre con 15 casos solamente.

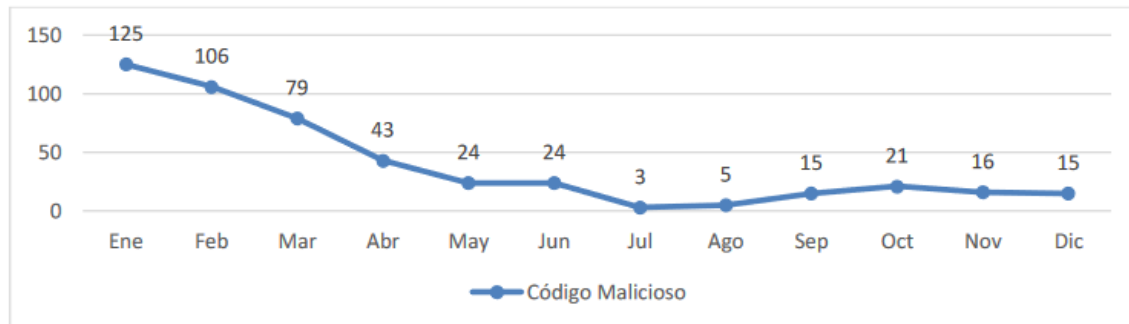


Imagen 5.- Distribución mensual de incidentes del tipo Código Malicioso.

F. Tipo de incidente: Seguridad de contenido de la información

- Este tipo de incidente tiene su mayor alza en enero con 176 casos, donde algún sistema fue vulnerado y modificada su información, hubo fuerte caída de este ataque en diciembre con 39 casos.

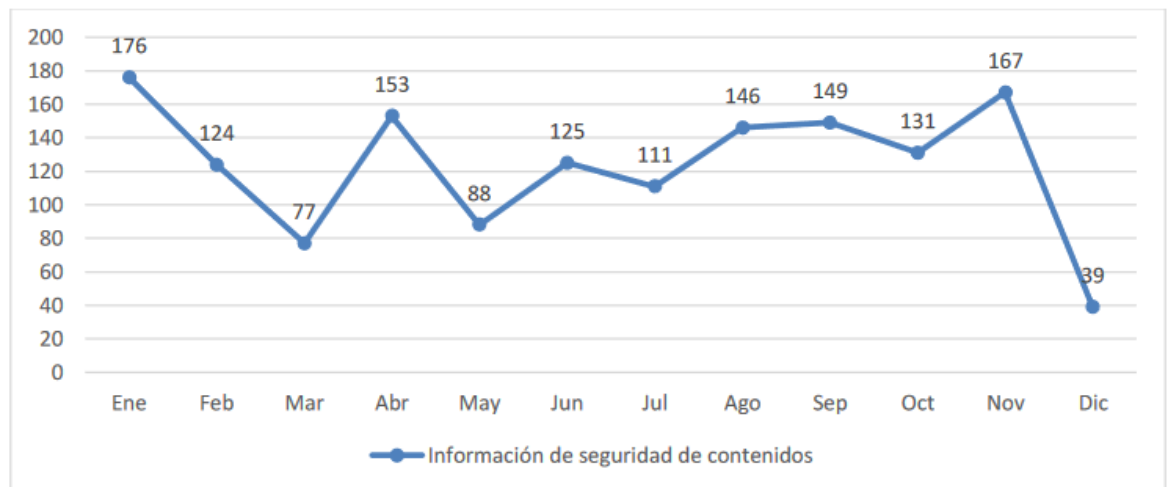


Imagen 6.- Distribución mensual de incidentes del tipo Seguridad del Contenido de Información.

G. Tipo incidente de fraude

- Este tipo de incidente conocido como phishing o spear phishing permite a los cyber delincuentes penetrar a los sistemas aprovechándose de la falta de conocimiento de los usuarios, por eso se han hecho muchas campañas para crear conciencia sobre la seguridad digital de los empleados en las compañías, la mayor cantidad de fraude se produjo en agosto llegando a 151 casos.

-

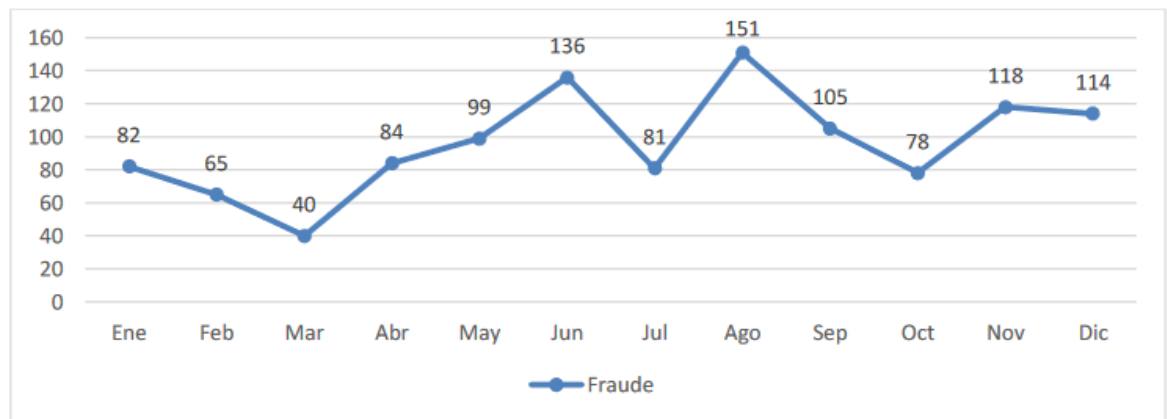


Imagen 7.- Distribución mensual de incidentes tipo Fraude.

H. Tipo de incidente: recopilación de información

- La recopilación de la información es el envío por parte de los delincuentes de solicitudes a un sistema para descubrir los puntos débiles de este. En enero se produjo la mayor cantidad de casos llegando a 372 y después llegó a diciembre con una gran baja de 1 caso.

-

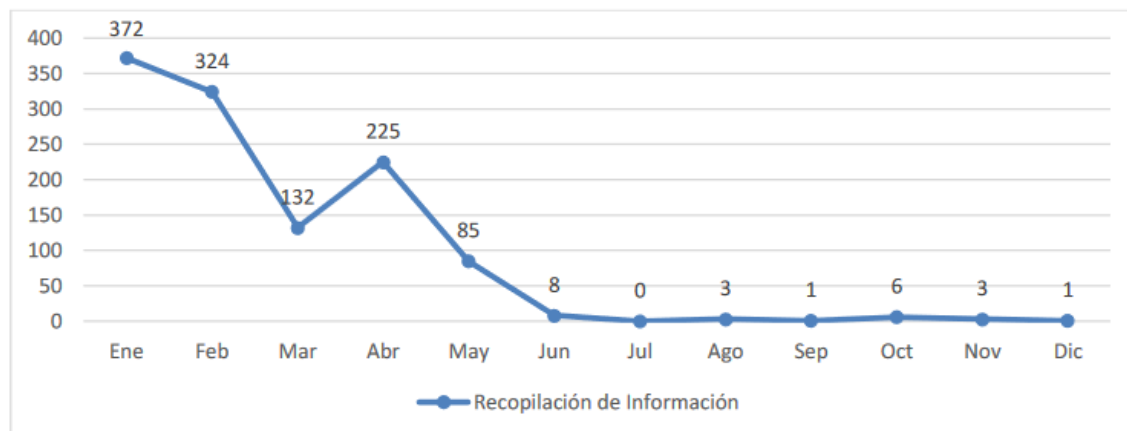


Imagen 8.- Distribución mensual de incidentes del tipo Recopilación de Información

I. Tipo de incidente: intrusión

- Este ataque principalmente se produce por el acceso no autorizado a través del uso de credenciales robados. La mayor cantidad de casos denunciados fue en abril con 56 casos y disminuyó a 3 casos en diciembre.

•

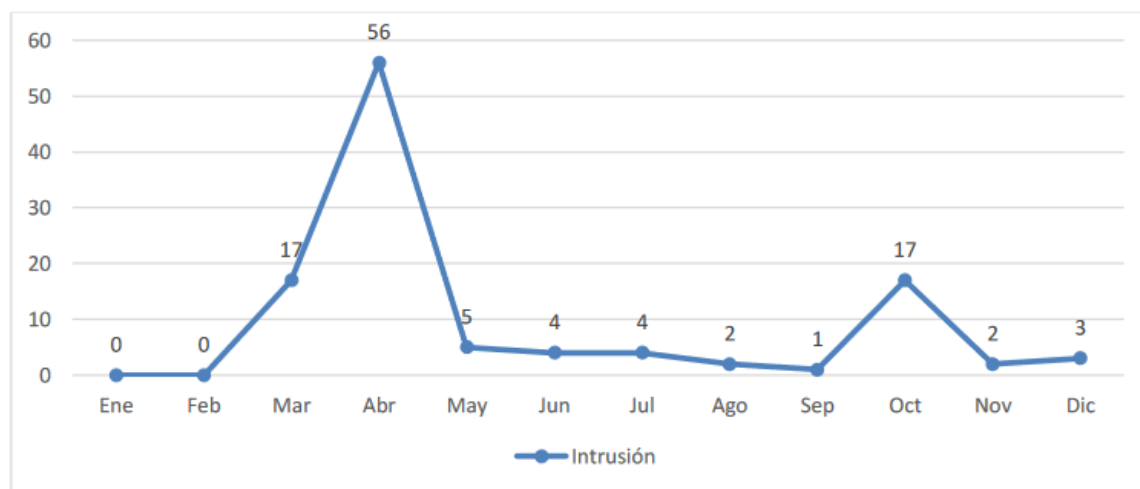


Imagen 9.- Distribución mensual de Intrusión

J. Tipo de incidente disponibilidad

- Asociados a los ataques de denegación de servicio o Dos o DDos, este año se observó un descenso de los registros,

potencialmente debido a una baja de la actividad Hacktivista contra instituciones de gobierno, lo que explicaría que los incidentes aumenten en octubre.

-

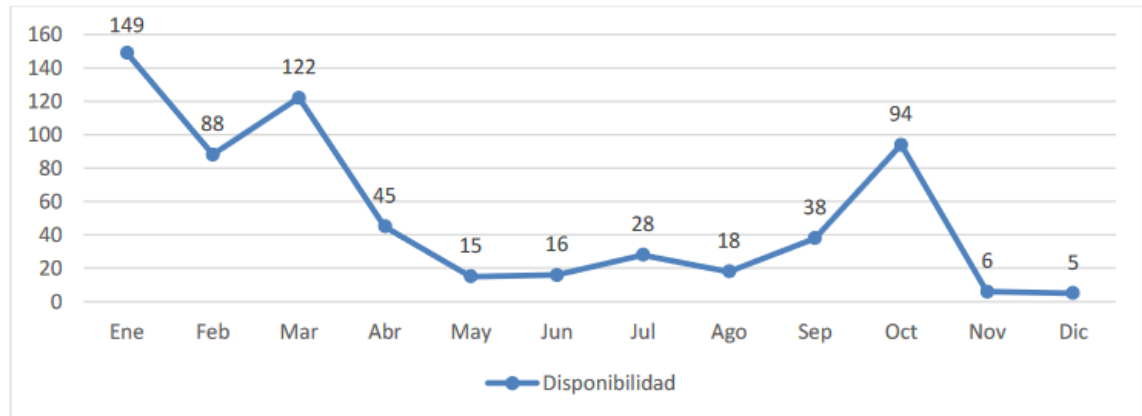


Imagen 10.- Distribución mensual de incidentes tipo Disponibilidad.

K. Tipo de incidente: Intentos de intrusión

- Se trata de intentos fallidos de comprometer un sistema o interrumpir cualquier servicio explotando vulnerabilidades conocidas. En este grafico se muestra que un febrero hubo la mayor cantidad con 26 casos, pero no se tiene claridad si disminuyo notablemente un diciembre porque el CSIRT del gobierno cambio su clasificación.

-

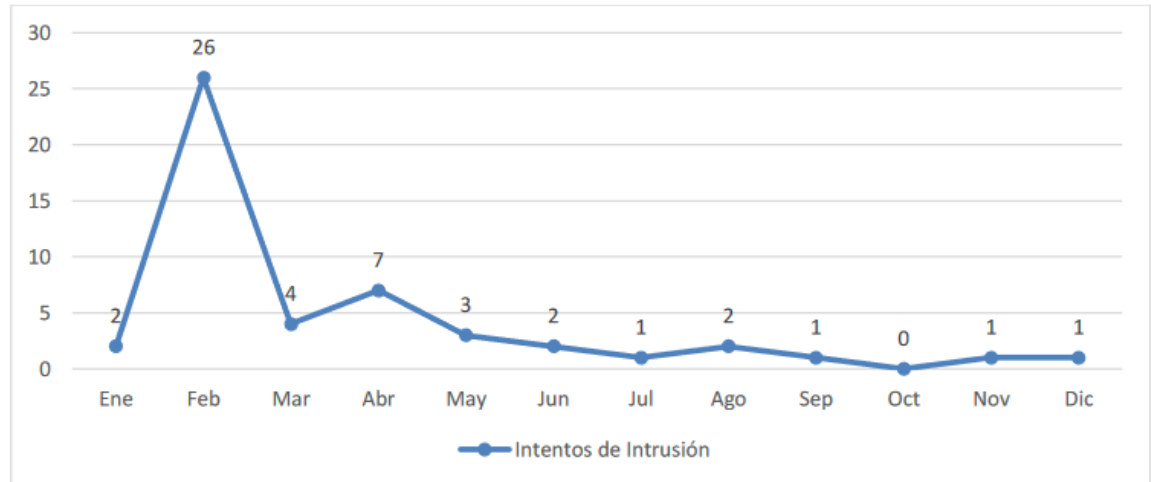


Imagen 11.- Distribución mensual de Intentos de Intrusión

L. Tipo de incidente: contenido abusivo

- En el caso de contenidos abusivos vemos que el grafico muestra un alza de casos llegando a aumentar a 48 casos en diciembre, este aumento se debe a que muchas instituciones no están tomando las debidas precauciones para evitar los ataques de agentes maliciosos que alteran las paginas web de sus victimas y redirigen el trafico hacia contenido abusivo como pornografía, violencia u otros.

-
-

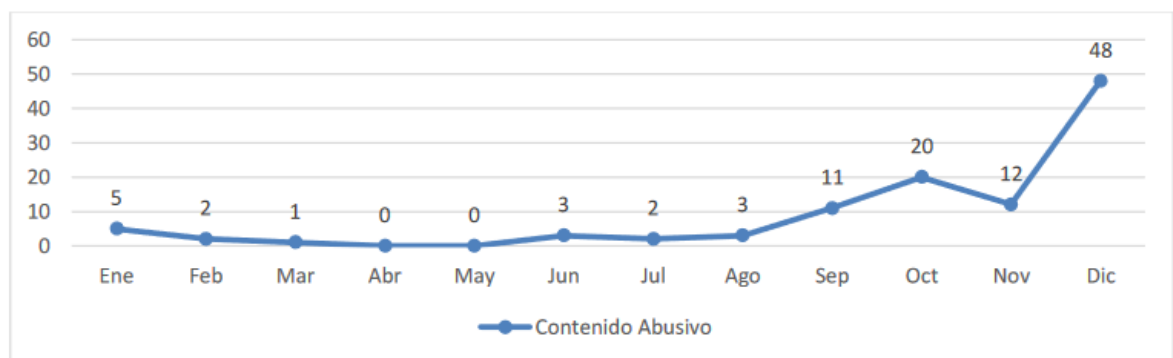
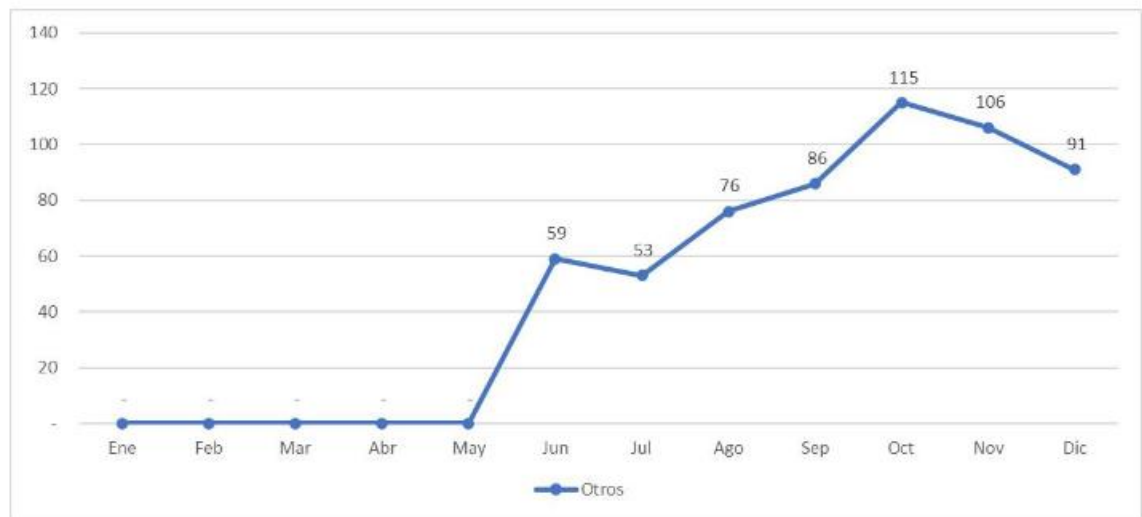


Imagen 12.- Distribución mensual de Contenido Abusivo

M. Tipo de incidente: otros.

- Estos están clasificados como otros, ya que no entran en la categoría de clasificación dado por Enisa (agencia de la unión europea para la ciber seguridad), estos no son resultados para analizar potenciales amenazas. El mayor numero de casos de estos se produjo en octubre con 115 casos y fue disminuyendo hasta llegar a diciembre con 91 casos.

•



N. Tickets emitidos a instituciones públicas y privados

- Desde la creación del CSIRT del gobierno este se fue vinculando con el sector privado y al unir fuerzas han logrado mantener el cyber espacio más seguro y así proteger la información de todos los chilenos. Es así como de un universo del 100% de tickets el 18% (4061 casos) hayan sido aportados por el sector privado.

•

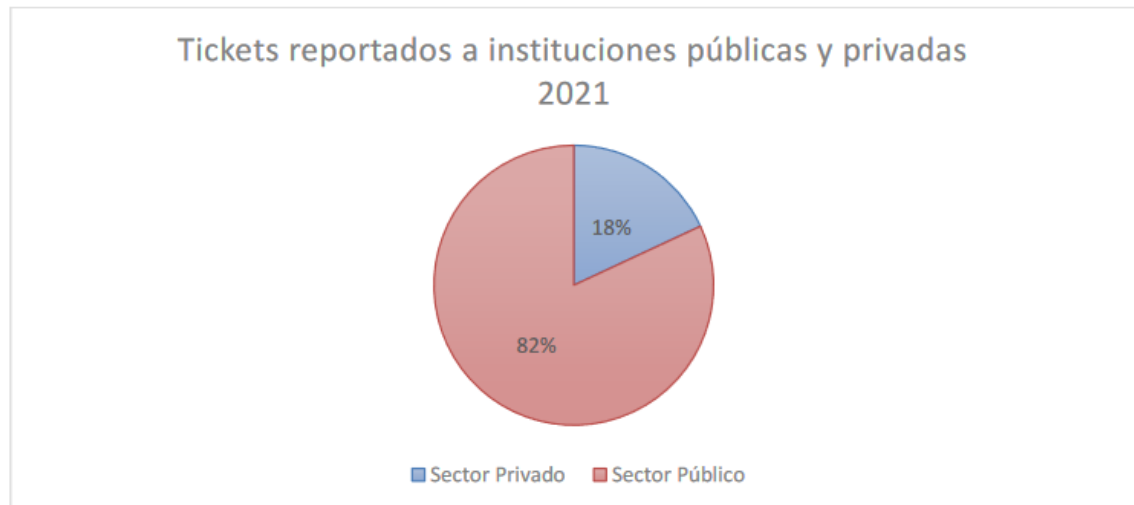


Imagen 3.- Distribución de tickets reportados a instituciones públicas y privadas

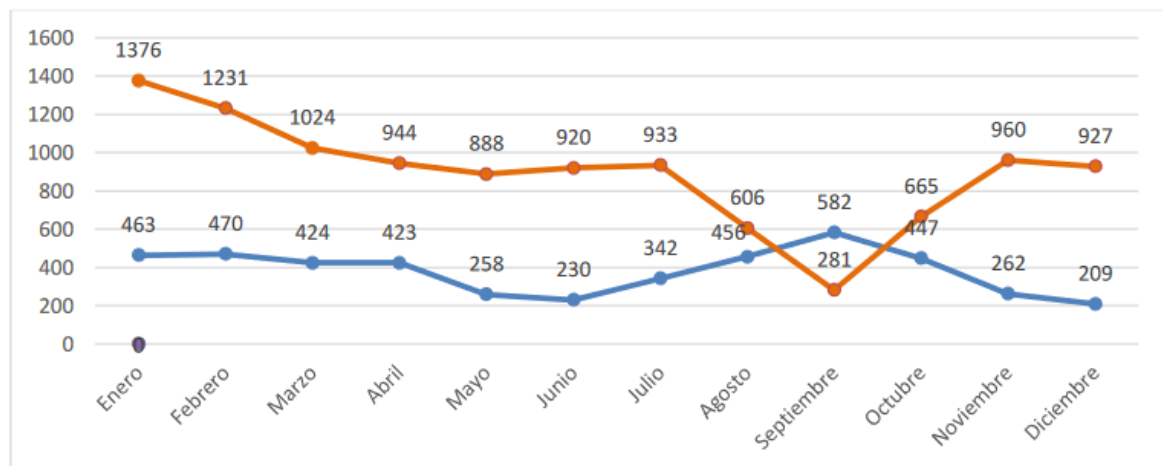


Imagen 13.- Reporte de tickets a instituciones públicas y privadas por mes.

Introducción

Cuéntenos acerca de su producto – las principales características – el segmento del mercado – la competencia – cuando saldrá a mercado.

Las empresas enfrentan un cambio con los trabajadores. Las generaciones más jóvenes esperan cosas distintas de los empleos. Estamos entrando a una nueva era de autodeterminación laboral. Las empresas deberán centrarse en adaptar el trabajo a la vida de las personas y sus necesidades. Por esta razones se crea E-work como una forma de acercar o generar trabajos con características especiales para este nuevo grupo emergente de trabajadores. A pesar de ser numerosas las empresas o aplicaciones web como competencia, en E-work queremos especializarnos en trabajos por empleados independientes, trabajos sin grado académicos y trabajos en pymes para integrar a la comunidad.

Proyecto

¿Quiénes son nuestros potenciales usuarios y que requieren?

Describir nuestros usuarios – indicar sus viajes de usuario, identificar y concluir sus necesidades como problemas y describa los features que debe tener el proyecto, por tanto, listando para ello los requerimientos y los diferentes escenarios que soportará el producto (ya sea mediante historias de usuarios o diagrama de casos de uso). Use imágenes y esquemas monísticos.

Nuestros potenciales usuarios son trabajadores que quieran usar la aplicación como método para publicitarse y expandir su alcance para poder laborar, pymes que tengan dificultad para encontrar empleados estables, empleados que busquen trabajos con flexibilidad horaria y se adecuen a sus necesidades. La particularidad de E-work es poder ofrecer a los usuarios la capacidad de elegir sus horarios de trabajo o encontrar uno que se adecue a las necesidades y preferencias. Además, promover trabajos que no requieran de un grado académico para poder inscribirse.

Planificación

La metodología de trabajo que se utilizó en el proyecto es scrum debido a su adaptabilidad, flexibilidad, orden y organización al desarrollar un proyecto.



El proyecto tiene 3 sprint de 1 semana, comenzando el día 10/11/22 hasta el día 1 / 12 /22. Se trabajará 5 días a la semana, dos horas diarias, completando un total de 10 horas a la semana y 30 horas de primera entrega de avance proyecto.

Hoja de ruta

Categoría de esta... ▼

Epic ▼

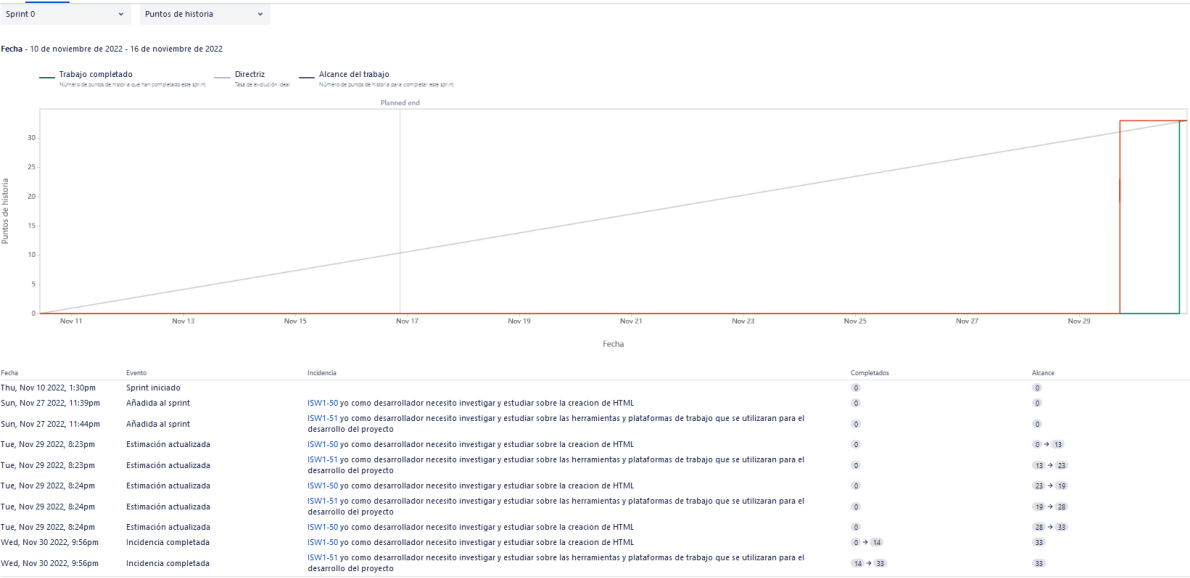
Etiqueta ▼

Ver configuración

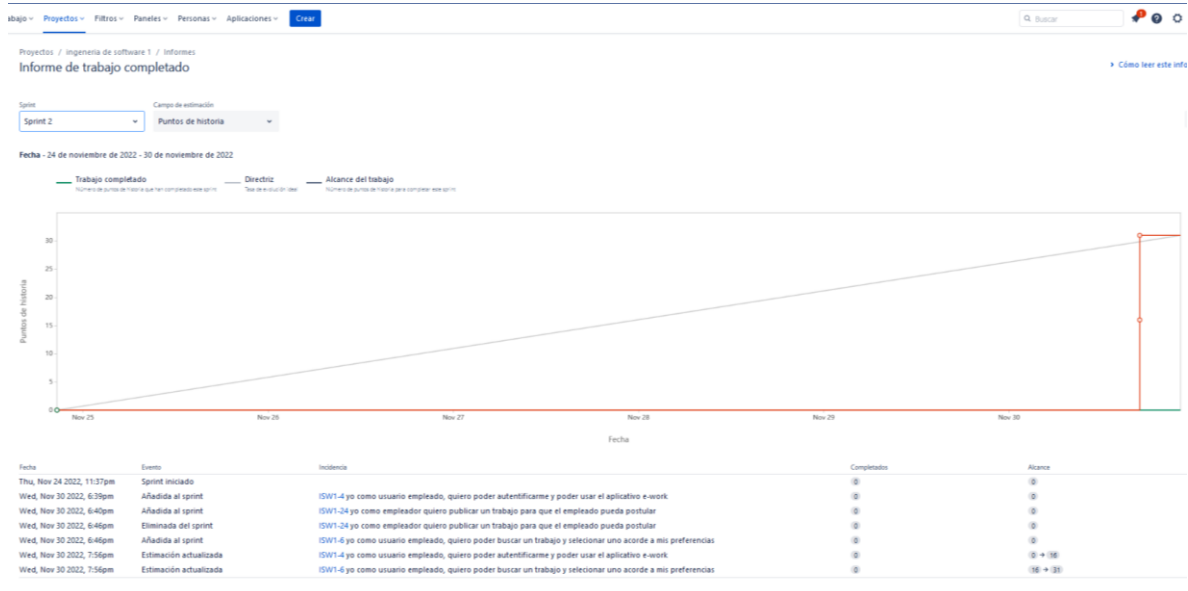
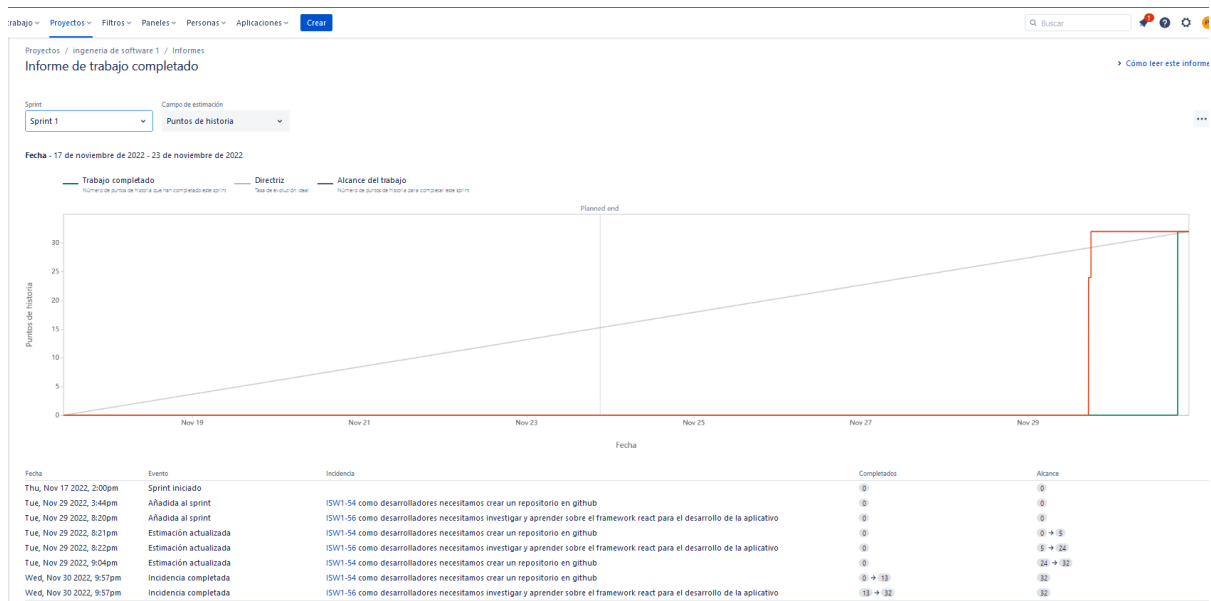
	OCT - DIC	ENE - MAR '23	ABR - JUN '23	JUL - SEP '23	OCT - DIC '23
Sprints					
> ISW1-7 Perfil					
> ISW1-58 empleador					
> ISW1-59 empleado					
+ Crear Epic					

Estimamos que el proyecto podría tardar hasta 1 año en desarrollar una demo completa con todas sus features.

Sprint 0



Sprint 1



Se puede apreciar que como equipo seleccionamos las historias de usuario contenidas en cada sprint para crear el primer demo del proyecto y las dividimos en tareas que cumplimos durante el desarrollo del sprint. Los gráficos de burndown chart indican que como grupo no cumplimos con la metodología de trabajo ya que no hicimos un buen uso de la herramienta jira, pero como equipo desarrollador cumplimos con la fecha de entrega de la demo.

Cuál es el roadmap del producto – en cuantos años veremos todos estos features (use el roadmap de jira) – cual es la metodología de gestión y la de desarrollo (sugerencia scrum + cascada) + muestre el producto backlog priorizado y estimado + hable de cuantos puntos de historia tiene + cuanto durarán los sprints (4 semanas) y cuáles son las historias de usuario – como se subdividen las tareas en tareas y como funciona y lo apoya jira en esto – considere el burndown chart.

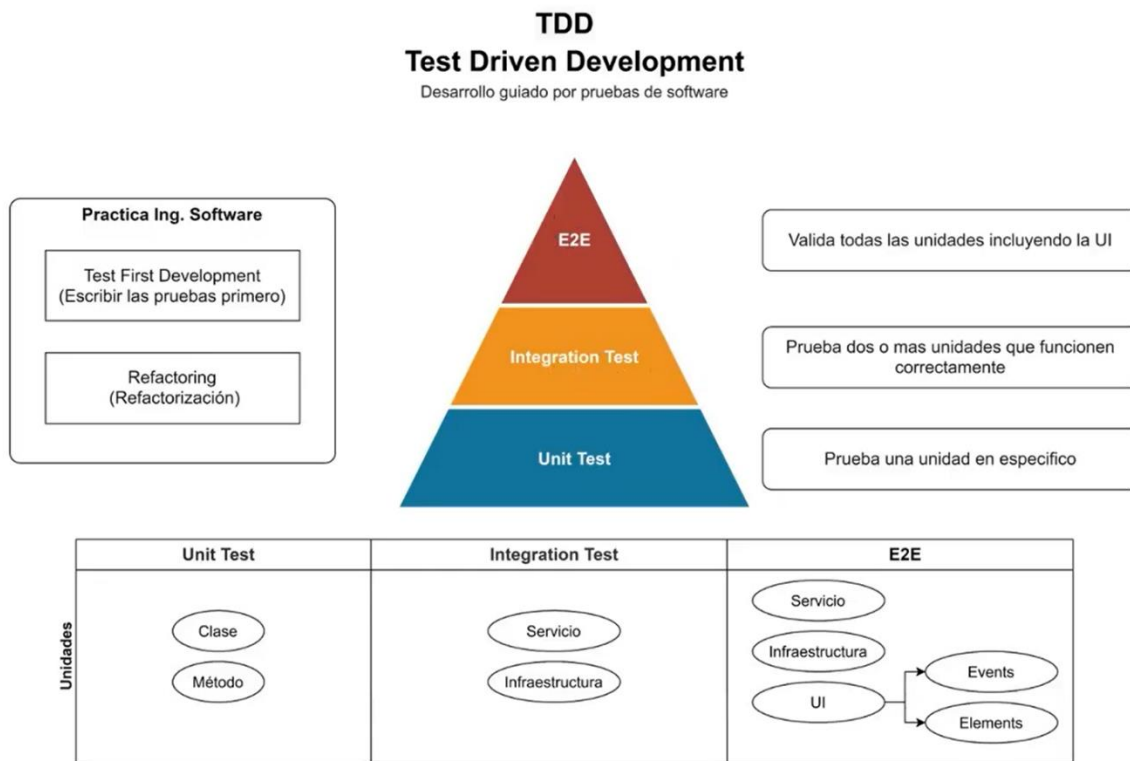
Planificación – se evalúa	Las tareas son trazables a historias de usuario/requisitos Las tareas tienen estimación de esfuerzo/complejidad
----------------------------------	--

Visión del producto

Visión del producto en ambiente operativo

¿Cómo construimos este producto?

Empleando la metodología DevOps usada en el desarrollo de software se hará uso de diferentes herramientas con la finalidad de automatizar procesos de testing.



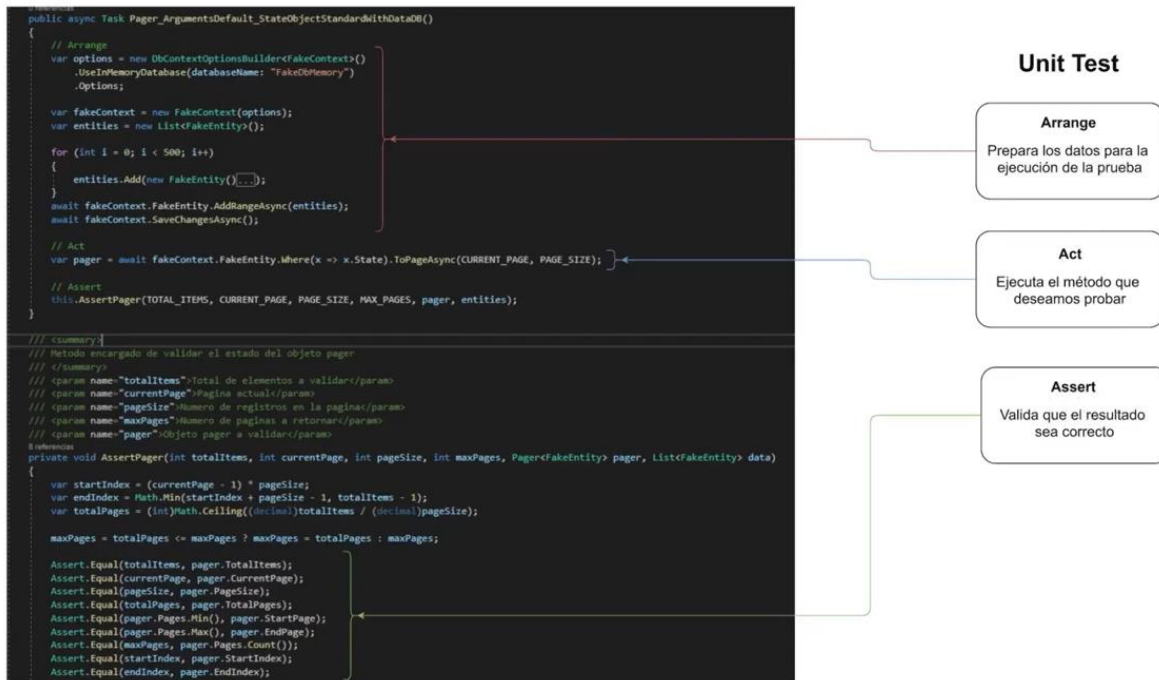
Los testeos que se buscara automatizar corresponden a pruebas unitarias, pruebas de integración y pruebas de aceptación.

- Las pruebas unitarias: testearan lo que corresponde a las unidades que contienen, ya sea un método o una clase del programa, en este caso se va a verificar que lo que codificamos se haga y se haga bien.
- Las pruebas de integración: tienen la capacidad de probar dos o más unidades que funcionan correctamente, así como infraestructuras y servicios como lo son por ejemplo una clase con integración donde se debe verificar que la comunicación entre dos puntos sea
- Las pruebas de aceptación: pruebas que verifican de manera formal la conformidad del sistema con los requisitos y la puesta en producción. Esta prueba mide si se cumplen los requisitos contra actuales que pidió el cliente.

Las Pruebas tanto de integración como las pruebas unitarias es posible automatizarlas usando los framework "XUnit" y ".Net Core".

Se empleará la metodología DevOps para mantener la comunicación entre la parte de Developers y la parte de operaciones, la principal característica del operador es mantener el código de los developers en funcionamiento y la de los developers es crear guías para que este entienda que podría salir mal, como se puede arreglar, que saldrá bien y como se puede observar, en el fondo cómo funciona el servicio. Con DevOps se obtendrán menos errores, los deployment serán más seguros, más rápidos y más confiables. Se pueden automatizar diferentes tipos de prácticas entre ellas los test que se están aplicando en este informe. Es posible medir la productividad como por ejemplo cuanto tarda un fix en salir o cuanto se tarda un operador en arreglar un error y por último esta metodología permite mejorar las herramientas internas compartidas por el o los equipos.

Como bien se había dado una pequeña descripción de lo que era una prueba unitaria, también es necesario decir que su estructura esta compuesta por tres partes las cuales son:



Arrange (Preparar): En esta etapa se preparan los datos necesarios para la prueba unitaria

Act (Ejecutar): Ejecuta el método que vamos a probar

Assert(Validar): En el assert validamos que la prueba de este método sea el esperado según lo que necesitamos.

Docker

- Si bien para la etapa que corresponde a Arrange y Act es recomendado usar los frameworks como Xunit y .Net Core para su automatización, la parte de Assert es posible automatizarla usando SonarQube con SonarScanner.
- Para el uso de estas dos últimas herramientas se usará Docker que automatiza el uso de varias aplicaciones generando un gasto menor de los recursos disponibles en una máquina, mediante la creación de contenedores que a diferencia de una máquina virtual en la que se debe designar un espacio estático, los contenedores usan los mismo recursos y espacio por lo que cada instalación es más liviana que instalar un sistema operativo nuevo por cada aplicación que se necesite usar.

Instalaciones de contenedores y su propósito

- Los contenedores son creados con Docker corriendo en tiempo real y con la utilización de comandos por CMD

Contenedor de SonarQube:

- herramienta que ayuda a hacer análisis estáticos y sirve para mejorar la calidad del código.

Por medio del siguiente comando se crea el contener de SonarQube:

<pre>docker run -d --name sonarqube -p 9000:9000 -p 9092:9092 sonarqube</pre>

- Para desplegar la interfaz de usuario se usa localhost:9000
- Al crear un nuevo proyecto en SonarQube se crea un nombre de proyecto, un token, se especifica el lenguaje en el cual está escrito el código y el sistema operativo en el que se está llevando a cabo el Scan y este generara un código para el SonarScanner.

Métricas SonarQube:

- SonarQube nos muestra las siguientes métricas las cuales serán explicadas brevemente para su entendimiento, para información extra sobre métricas de las métricas se puede consultar en referencias del documento.

Evidencias/Issues

Fragmentos de código de un proyecto que incumplen reglas establecidas para cada lenguaje en su respectivo plan de calidad, las evidencias se subdividen a su vez en categorías de evidencias que solo serán mencionadas:

- Nuevas evidencias
- Nuevas evidencias seguridad
- Falsos positivos
- Evidencias abiertas
- Evidencias reabiertas

Duplicados

Ayuda a mantener una programación limpia encontrando duplicidades en el código con constantes y métodos aplicados. Las métricas entregadas por los duplicados serán mencionadas:

- Archivos duplicados
- Líneas duplicadas
- Porcentaje de líneas duplicadas (%)

Complejidad

Se mide la dificultad del mantenimiento y escalabilidad del proyecto evaluado en SonarQube, esta métrica busca que el proyecto sea un código lo más sencillo posible.

Se mide la complejidad del proyecto por:

- Complejidad ciclomática
- Complejidad cognitiva

Tamaño

Aquí se mide la magnitud del proyecto, no indica nada respecto a la calidad del proyecto.

Pruebas o cobertura

Indica cuantas líneas de código son cubiertas por pruebas unitarias. Es importante destacar que SonarQube no ejecuta las pruebas ni genera informes. Solo importa informes generados previamente. Podemos encontrar información sobre los ficheros afectados, resultados sobre pruebas unitarias ya ejecutadas y medir por medio de un reporte histórico la evolución de la productividad.

Severidades de los fallos

Se califican las severidades dentro de los siguientes puntos:

- **Vulnerabilidades:** fallas en la seguridad del programa que deben corregirse de inmediato
- **Bug:** errores encontrados en el código.
- **Security Hspot:** es un tipo de vulnerabilidad que puede o no ser corregida debido a que representa un bajo riesgo a la seguridad y queda sujeto a juicio del desarrollador si este es corregido o no.
- **Code smell:** código que dificulta el mantenimiento del proyecto.

Los siguientes términos serán usados más adelante por lo cual es importante destacar su significado.

- **Blocker:** Error con una alta probabilidad de afectar el comportamiento de la aplicación en producción.
- **Crítico:** Urge su corrección, aunque no suponga una afectación al comportamiento de la aplicación en producción o porque es una falla de seguridad de impacto alto.
- **Major:** Defecto de calidad que puede afectar enormemente a la productividad del desarrollador.
- **Minor:** Defecto de calidad que puede afectar ligeramente la productividad del desarrollador.

Fiabilidad

capacidad de un sistema para funcionar libre de fallos dentro de sus funciones especificadas. Mide el coste que tendría arreglar esos fallos si se presentaran y para ello se entrega una escala:

- Sin Bugs
- Bug Minor
- Bug Major
- Bug Critico
- Bug Blocker

Seguridad

Nivel de protección otorgado a los datos para que estos no sean leídos o modificados por terceros, se mide la capacidad del sistema para dar una respuesta frente ataques informáticos y reducir el número de estos, midiéndolos según la siguiente escala:

Vulnerabilidades de tipo:

- Minor
- Major
- Critico
- Blocker

Mantenibilidad

Capacidad del producto software para ser modificado efectiva y eficientemente en respuesta a necesidades evolutivas, correctivas o perfectivas.

Contenedor SonarScanner:

Con él se prueban algunos códigos de PHP y Python que al ser analizados muestran errores que deben corregirse a nivel de código, pero no se muestran vulnerabilidades de seguridad en los dos escaneos realizados.

Por medio del siguiente comando se crea el contener de SonarScanner:
--

<code>docker run -it --rm sonarsource/sonar-scanner-cli:4.5 -version</code>

El SonarScanner se puede descargar de:
--

https://docs.sonarqube.org/latest/analysis/scan/sonarscanner/

Se debe descomprimir en una carpeta y en esta crear el .txt "sonar-project.properties" para la configuración del Scan que se realizara al código que se busca analizar. El .txt debe llevar el siguiente contenido:

```
# must be unique in a given SonarQube instance
sonar.projectKey=my:project
```

```
# --- optional properties ---
```

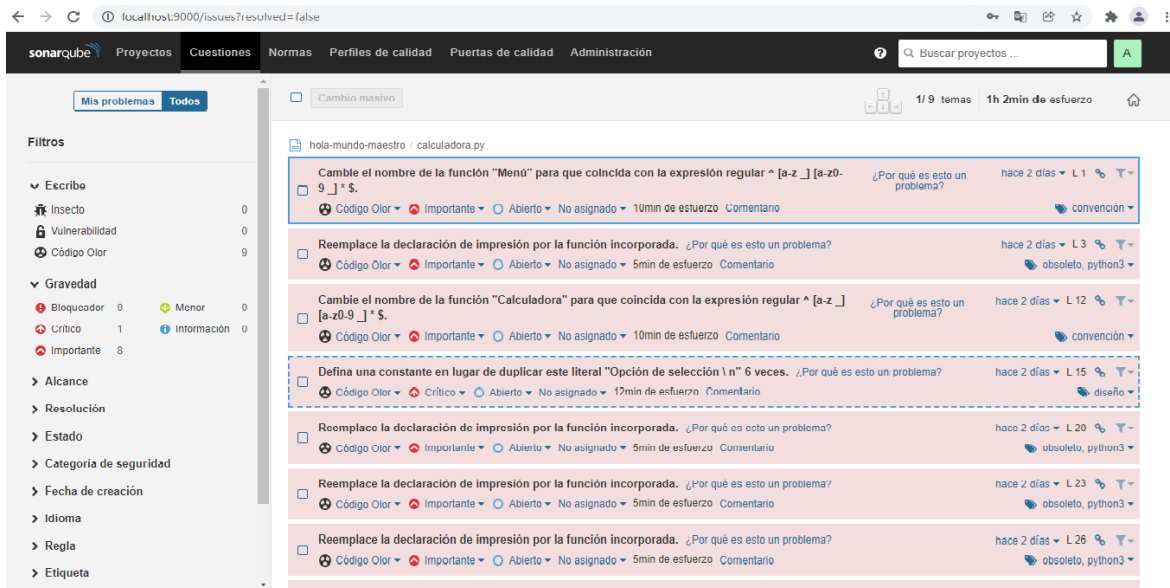
```
# defaults to project key
#sonar.projectName=My project
# defaults to 'not provided'
#sonar.projectVersion=1.0

# Path is relative to the sonar-project.properties file. Defaults to .
#sonar.sources=.

# Encoding of the source code. Default is default system encoding
#sonar.sourceEncoding=UTF-8
```

Donde sonar.projectKey debe llevar el nombre del código a analizar.

Escaneo de calculadora.py por SonarScanner, vulnerabilidades encontradas 0, errores en el código 9.



¿Cómo aceleramos nuestro desarrollo?

Mitigación de riesgo

El inicio en los trabajos demoraba al principio mucho tiempo, el equipo encargado del proyecto se reunía y pensaban muchas semanas organizando las actividades antes de entregar el trabajo encomendado por el cliente.

El inicio de Thoughtworks fue desarrollado principalmente por Luke Barret el 2004, el cual describió y desarrollo aún más las técnicas de inicio de un trabajo.

Esto permitió que, aunque un proyecto varía del otro generalmente se estandariza entre el negocio (no importa cuál sea su rubro) y los técnicos encargados del proyecto, lo que permite crear una lista ordenada de historias de usuario con estimación junto con un plan de lanzamiento.

The lean Inception vino a modificar lo esbelto en el 2006.

El nuevo estilo de inicio es resuelto por 2 razones:

- A) La duración del inicio es más corta eliminando todo lo que no era sobre el producto en sí. Ej: arquitectura, proyecto, otros. Haciéndolo más delgado.
- B) El resultado final del inicio es la comprensión de MVP un concepto principal del movimiento LEAN START UP.

¿Cómo llega a realizar la MVP?

Un inicio esbelto es útil cuando el equipo necesita desarrollar o repetir varias veces un proceso como MVP.

El MVP lo construimos para saber si vale la pena continuar construyendo un producto, por lo cual se eligen funciones basadas en nuestras suposiciones, pensando en lo que es más valioso para el usuario. Hay dos tipos de proyectos los grandes y los pequeños.

- A) grandes proyectos: encuentran que es importante un inicio esbelto para comenzar rápidamente, lo que significa que construyen iteraciones tempranas para descubrir y probar que características son realmente importantes para el usuario.

Los proyectos pequeños toman ideas de otros grandes proyectos que han sido probados por algunos MVP previamente al software y las convierten en un producto software.

El MVP, no sustituye las demás sesiones de ideación de un proyecto, la investigación del cliente, la revisión arquitectónica o el análisis competitivo. El MVP es un análisis específico que forma parte de la comprensión que se necesita para crear un producto exitoso y así ganar tiempo para dirigir el esfuerzo en lograr el producto adecuado a las necesidades del cliente.

Resultado de la mitigación de riesgo

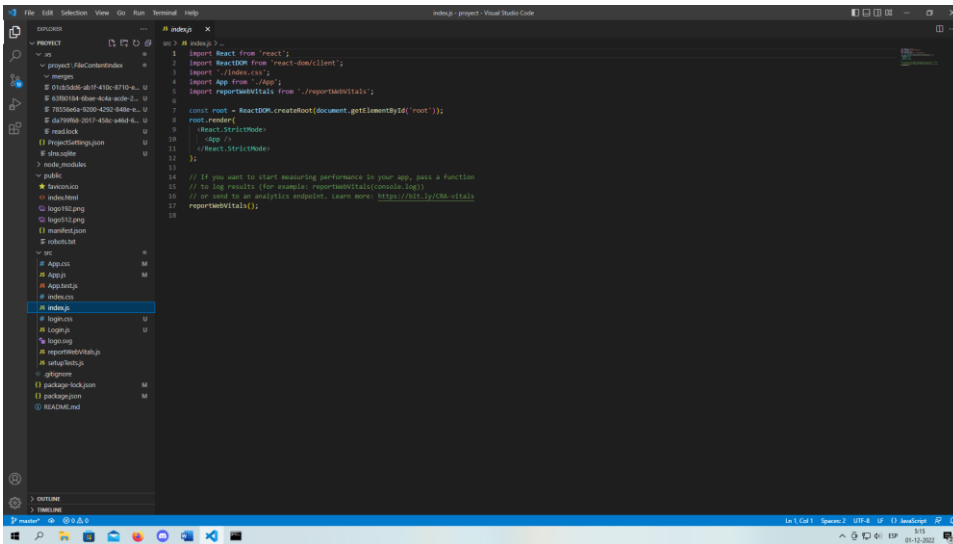
Estas dos imágenes corresponden al código dentro del archivo index.html . Este archivo está dentro de los directorios de los servidores de los sitios webs que se carga siempre que se solicita un dominio y no se especifica el nombre de un archivo específico. Es la página principal o el home de nuestra web sin ella no existe el sitio web, de ella salen las demás ramificaciones que en este caso serían las vistas que el usuario va explorando conforme va usando el sitio web de E-work.


```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8" />
5   <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
6   <meta name="viewport" content="width=device-width, initial-scale=1" />
7   <meta name="theme-color" content="#000000" />
8   <meta
9     name="description"
10    content="Web site created using create-react-app"
11  />
12  <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
13  <!--
14    manifest.json provides metadata used when your web app is installed on a
15    user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/
16  -->
17  <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
18  <!--
19    Notice the use of %PUBLIC_URL% in the tags above.
20    It will be replaced with the URL of the 'public' folder during the build.
21    Only files inside the 'public' folder can be referenced from the HTML.
22
23    Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
24    work correctly both with client-side routing and a non-root public URL.
25    Learn how to configure a non-root public URL by running `npm run build`.
26  -->
27  <title>React App</title>
28  </head>
29  <body>
30    <noscript>You need to enable JavaScript to run this app.</noscript>
31    <div id="root"></div>
32  <!--
33    This HTML file is a template.
34    If you open it directly in the browser, you will see an empty page.
35
36    You can add webfonts, meta tags, or analytics to this file.
37    The build step will place the bundled scripts into the <body> tag.
38
39    To begin the development, run `npm start` or `yarn start`.
40    To create a production bundle, use `npm run build` or `yarn build`.
41  -->
42  </body>
43 </html>
```

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8" />
5   <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
6   <meta name="viewport" content="width=device-width, initial-scale=1" />
7   <meta name="theme-color" content="#000000" />
8   <meta
9     name="description"
10    content="Web site created using create-react-app"
11  />
12  <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
13  <!--
14    manifest.json provides metadata used when your web app is installed on a
15    user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/
16  -->
17  <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
18  <!--
19    Notice the use of %PUBLIC_URL% in the tags above.
20    It will be replaced with the URL of the 'public' folder during the build.
21    Only files inside the 'public' folder can be referenced from the HTML.
22
23    Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
24    work correctly both with client-side routing and a non-root public URL.
25    Learn how to configure a non-root public URL by running `npm run build`.
26  -->
27  <title>React App</title>
28  </head>
29  <body>
30    <noscript>You need to enable JavaScript to run this app.</noscript>
31    <div id="root"></div>
32  <!--
33    This HTML file is a template.
34    If you open it directly in the browser, you will see an empty page.
35
36    You can add webfonts, meta tags, or analytics to this file.
37    The build step will place the bundled scripts into the <body> tag.
38
39    To begin the development, run `npm start` or `yarn start`.
40    To create a production bundle, use `npm run build` or `yarn build`.
41  -->
42  </body>
43 </html>
```

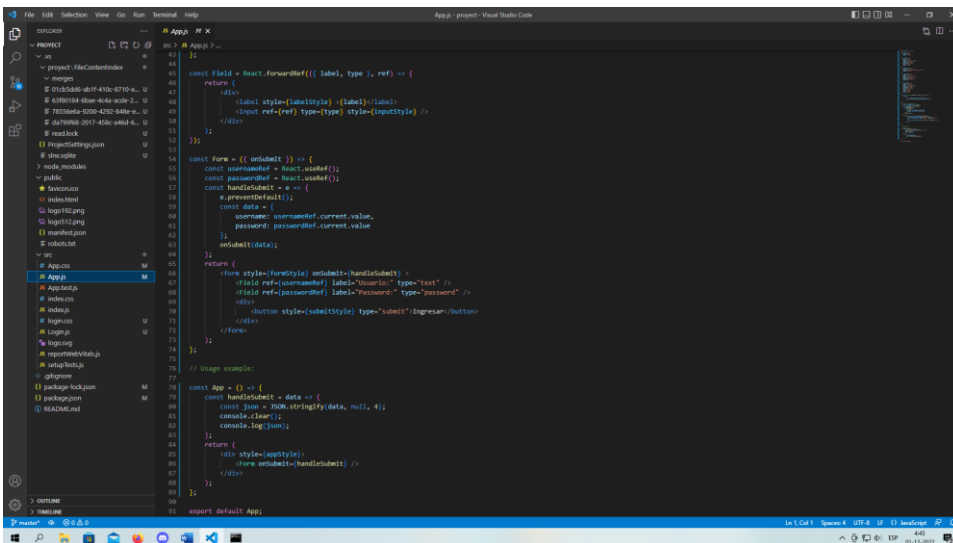
Index.js de este archivo el componente App.js está siendo utilizado. El archivo comienza importando todos los módulos JS y otros activos que necesita para ejecutarse.

Index.js

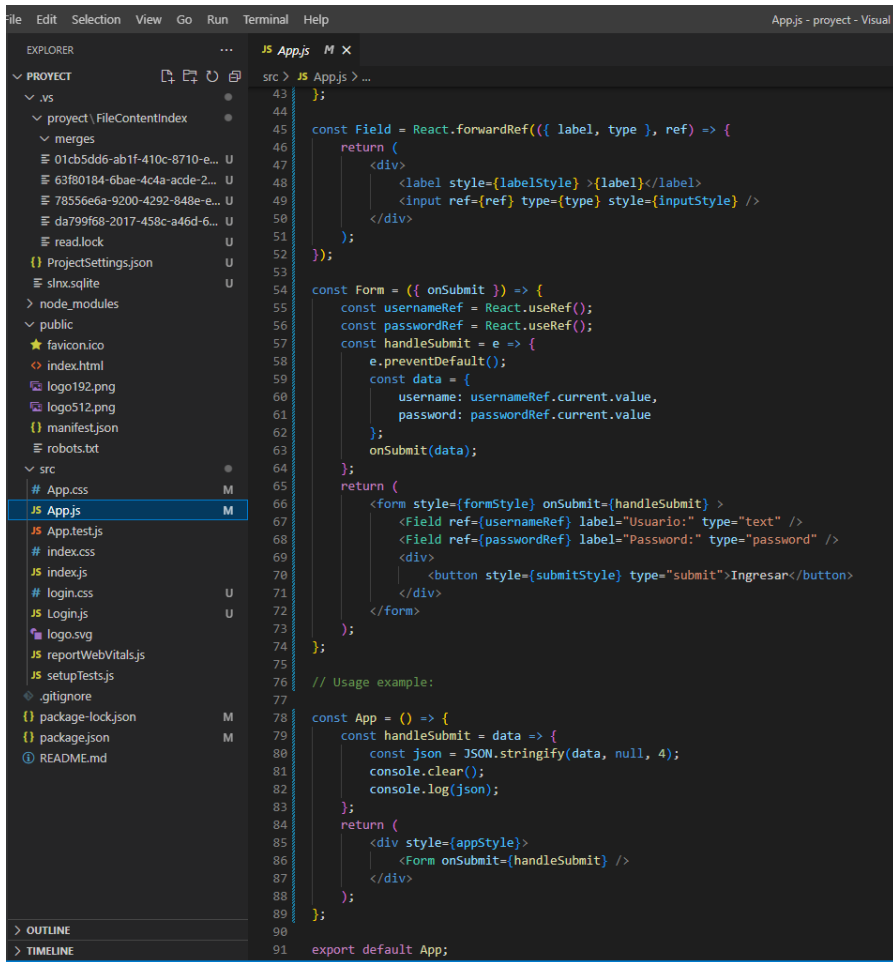


```
1 import React from 'react';
2 import ReactDOM from 'react-dom/client';
3 import './index.css';
4 import App from './App';
5 import reportWebVitals from './reportWebVitals';
6
7 const root = ReactDOM.createRoot(document.getElementById('root'));
8 root.render(
9   <React.StrictMode>
10     <App />
11   </React.StrictMode>
12 );
13
14 // If you want to start measuring performance in your app, pass a function
15 // to log results (for example: reportWebVitals(console.log))
16 // or send to an analytics endpoint. Learn more: https://bit.ly/CMS-vitals
17 reportWebVitals();
```

App.js

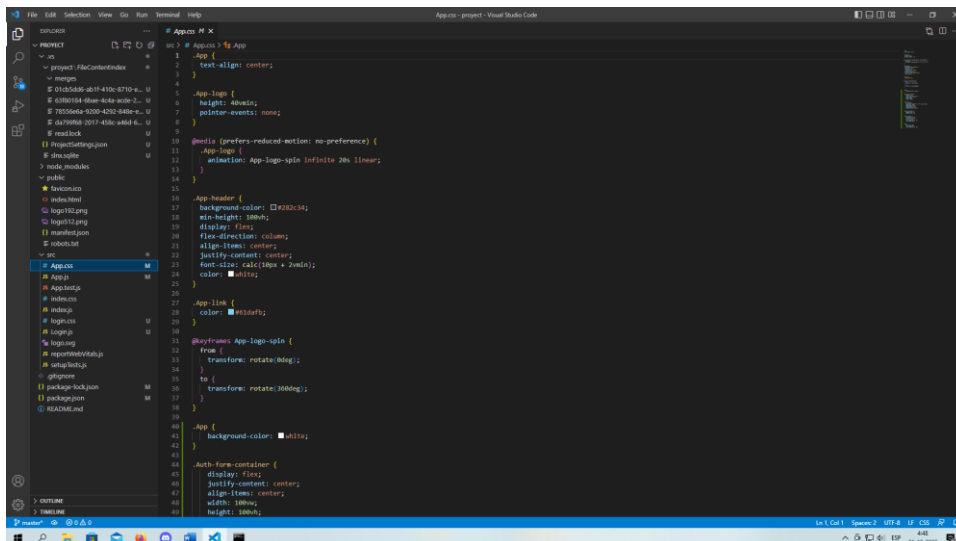


```
1 // ForwardRef prop type
2 const field = React.forwardRef((props, ref) => {
3   return (
4     <div>
5       <input style={inputStyle} type={type} style={inputStyle} ref={ref} />
6     </div>
7   );
8 });
9
10 const form = ({ onSubmit }) => {
11   const handleSubmit = React.useRef();
12   const passwordRef = React.useRef();
13   const data = {
14     username: usernameRef.current.value,
15     password: passwordRef.current.value,
16   };
17   onSubmit(data);
18 };
19
20 const App = () => {
21   return (
22     <div style={formStyle} onSubmit={handleSubmit}>
23       <div ref={usernameRef} label="Username" type="text" />
24       <div ref={passwordRef} label="Password" type="password" />
25       <button style={buttonStyle} type="submit">Ingresar</button>
26     </div>
27   );
28 };
29
30 // Usage example
31 const App = () => {
32   const handleSubmit = data => {
33     console.log(data);
34     console.log('data', null, 4);
35   };
36   return (
37     <div style={appStyle}>
38       <div onSubmit={handleSubmit}>
39         <div>
40           <div ref={usernameRef} label="Username" type="text" />
41           <div ref={passwordRef} label="Password" type="password" />
42           <button style={buttonStyle} type="submit">Ingresar</button>
43         </div>
44       </div>
45     </div>
46   );
47 };
48
49 export default App;
```

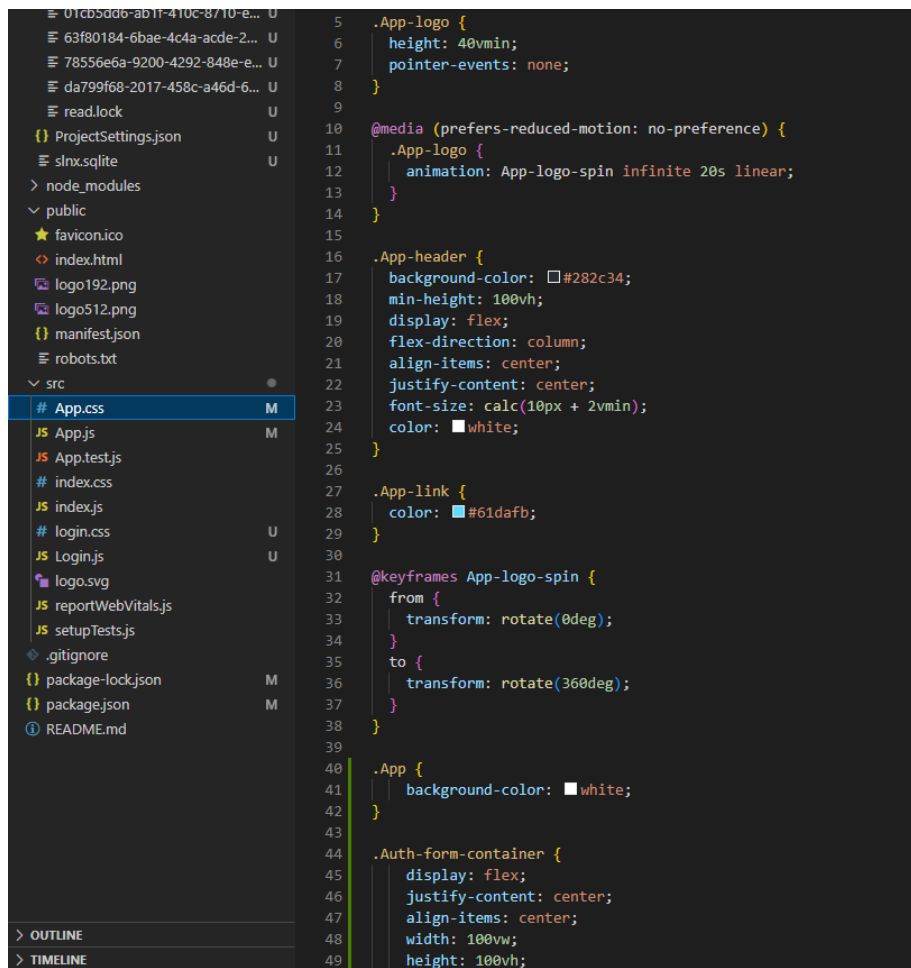


```
43 };
44
45 const Field = React.forwardRef((( label, type ), ref) => {
46   return (
47     <div>
48       <label style={labelStyle} >{label}</label>
49       <input ref={ref} type={type} style={inputStyle} />
50     </div>
51   );
52 });
53
54 const Form = ({ onSubmit }) => {
55   const usernameRef = React.useRef();
56   const passwordRef = React.useRef();
57   const handleSubmit = e => {
58     e.preventDefault();
59     const data = {
60       username: usernameRef.current.value,
61       password: passwordRef.current.value
62     };
63     onSubmit(data);
64   };
65   return (
66     <form style={formStyle} onSubmit={handleSubmit} >
67       <Field ref={usernameRef} label="Usuario:" type="text" />
68       <Field ref={passwordRef} label="Password:" type="password" />
69       <div>
70         <button style={submitStyle} type="submit">Ingresar</button>
71       </div>
72     </form>
73   );
74 };
75
76 // Usage example:
77
78 const App = () => {
79   const handleSubmit = data => {
80     const json = JSON.stringify(data, null, 4);
81     console.clear();
82     console.log(json);
83   };
84   return (
85     <div style={appStyle}>
86       <Form onSubmit={handleSubmit} />
87     </div>
88   );
89 };
90
91 export default App;
```

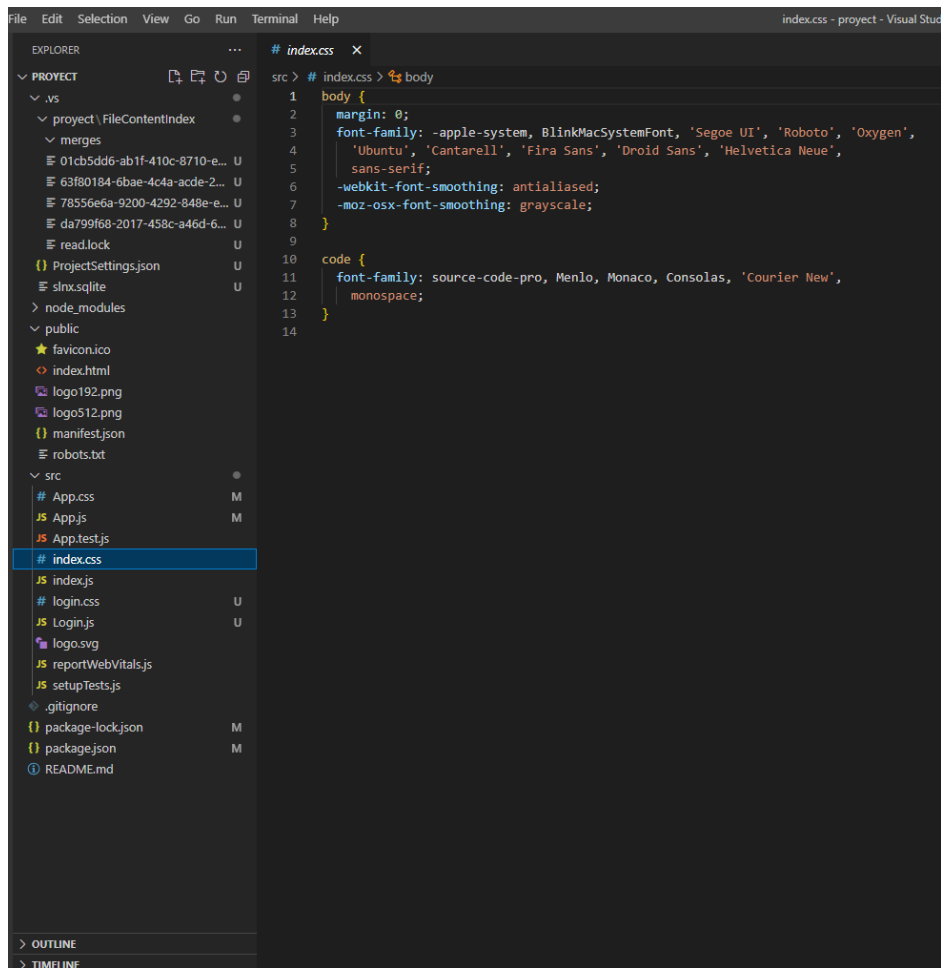
App.css contine los estilos que se le aplicaron al archivo app.js



```
1 .App {
2   text-align: center;
3 }
4
5 .App-logo {
6   height: 40vmin;
7   pointer-events: none;
8 }
9
10 @media (prefers-reduced-motion: no-preference) {
11   .App-logo {
12     animation: App-logo-spin infinite 20s linear;
13   }
14 }
15
16 .App-header {
17   background-color: #282c34;
18   min-height: 100vh;
19   display: flex;
20   flex-direction: column;
21   align-items: center;
22   justify-content: center;
23   font-size: calc(1em + 2vmin);
24   color: white;
25 }
26
27 .App-link {
28   color: white;
29 }
30
31 @keyframes App-logo-spin {
32   from {
33     transform: rotate(0deg);
34   }
35   to {
36     transform: rotate(360deg);
37   }
38 }
39
40 .App {
41   background-color: white;
42 }
43
44 .Auth-form-container {
45   display: flex;
46   justify-content: center;
47   align-items: center;
48   width: 100%;
49   height: 100%;
50 }
```



Index.css archivo que contiene el código que le brindara el estilo por defecto a la pagina



Login.js el código que se encuentra aquí recibe los parámetros que el usuario entrega al completar las casillas de input y conecta con la base de datos.

```
File Edit Selection View Go Run Terminal Help
Loginjs - project - Visual Studio Code

EXPLORER
PROJECT
  .vs
  project/FileContentIndex
  merges
  01cb5dd6-ab1f-410c-8710-e... U
  63f80184-6bae-4c4a-acde-2... U
  78556e6a-9200-4292-848e-e... U
  da799f68-2017-458c-a46d-6... U
  read.lock
  {} ProjectSettings.json
  {} slnx.sqlite
  > node_modules
  > public
    favicon.ico
    index.html
    logo192.png
    logo512.png
    {} manifest.json
    {} robots.txt
  > src
    # App.css
    # App.js
    # App.test.js
    # index.css
    # index.js
    # login.css
    # Loginjs
    logo.svg
    reportWebVitals.js
    setupTests.js
    .gitignore
    {} package-lock.json
    {} package.json
    {} README.md

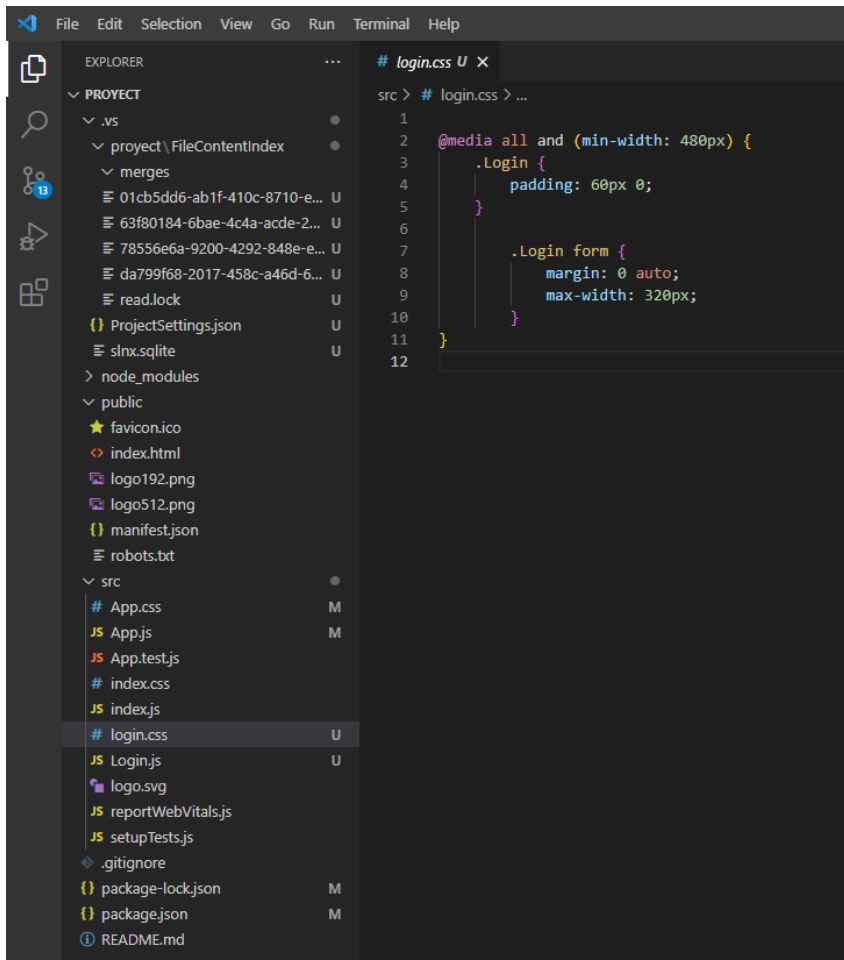
src > JS Loginjs > Login
1 import React, { useState } from "react";
2 import Form from "react-bootstrap/Form";
3 import Button from "react-bootstrap/Button";
4 import "../Login.css";
5
6 export default function Login() {
7   const [email, setEmail] = useState("");
8   const [password, setPassword] = useState("");
9
10  function validateForm() {
11    return email.length > 0 && password.length > 0;
12  }
13
14  function handleSubmit(event) {
15    event.preventDefault();
16  }
17
18  return (
19    <div className="login">
20      <Form onSubmit={handleSubmit}>
21        <Form.Group size="lg" controlId="email">
22          <Form.Label>Email</Form.Label>
23          <Form.Control
24            autoFocus
25            type="email"
26            value={email}
27            onChange={e => setEmail(e.target.value)}
28          />
29        </Form.Group>
30        <Form.Group size="lg" controlId="password">
31          <Form.Label>Password</Form.Label>
32          <Form.Control
33            type="password"
34            value={password}
35            onChange={e => setPassword(e.target.value)}
36          />
37        </Form.Group>
38        <Button block="true" size="lg" type="submit" disabled={!validateForm()}>
39          Login
40        </Button>
41      </Form>
42    </div>
43  );
44 }
```

```
File Edit Selection View Go Run Terminal Help
Loginjs - project - Visual Studio Code

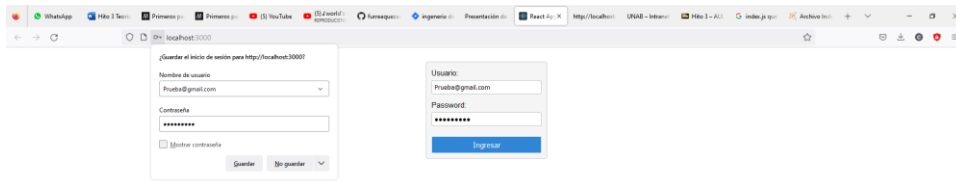
EXPLORER
PROJECT
  .vs
  project/FileContentIndex
  merges
  01cb5dd6-ab1f-410c-8710-e... U
  63f80184-6bae-4c4a-acde-2... U
  78556e6a-9200-4292-848e-e... U
  da799f68-2017-458c-a46d-6... U
  read.lock
  {} ProjectSettings.json
  {} slnx.sqlite
  > node_modules
  > public
    favicon.ico
    index.html
    logo192.png
    logo512.png
    {} manifest.json
    {} robots.txt
  > src
    # App.css
    # App.js
    # App.test.js
    # index.css
    # index.js
    # login.css
    # Loginjs
    logo.svg
    reportWebVitals.js
    setupTests.js
    .gitignore
    {} package-lock.json
    {} package.json
    {} README.md

src > JS Loginjs > Login
1 import React, { useState } from "react";
2 import Form from "react-bootstrap/Form";
3 import Button from "react-bootstrap/Button";
4 import "../Login.css";
5
6 export default function Login() {
7   const [email, setEmail] = useState("");
8   const [password, setPassword] = useState("");
9
10  function validateForm() {
11    return email.length > 0 && password.length > 0;
12  }
13
14  function handleSubmit(event) {
15    event.preventDefault();
16  }
17
18  return (
19    <div className="Login">
20      <Form onSubmit={handleSubmit}>
21        <Form.Group size="lg" controlId="email">
22          <Form.Label>Email</Form.Label>
23          <Form.Control
24            autoFocus
25            type="email"
26            value={email}
27            onChange={e => setEmail(e.target.value)}
28          />
29        </Form.Group>
30        <Form.Group size="lg" controlId="password">
31          <Form.Label>Password</Form.Label>
32          <Form.Control
33            type="password"
34            value={password}
35            onChange={e => setPassword(e.target.value)}
36          />
37        </Form.Group>
38        <Button block="true" size="lg" type="submit" disabled={!validateForm()}>
39          Login
40        </Button>
41      </Form>
42    </div>
43  );
44 }
```

Login.css es el archivo que contiene los estilos del login.js solo se le pusieron códigos para centralizar la información.



Vista del login



Conclusiones

Referencias

GitHub Repositorio E-work

- <https://github.com/furreaquezada/ProyectoE-work>

Jira Project

- <https://searchwork.atlassian.net/jira/software/projects/ISW1/boards/1/backlog>

Pruebas de testing

- <https://programacionymas.com/blog/tipos-de-testing-en-desarrollo-de-software>

CSIRT

- <https://www.csirt.gob.cl/media/2022/01/INFORME-ANUAL-2021-CSIRT-de-Gobierno.pdf>

Lean Inception

- <https://martinfowler.com/articles/lean-inception/>

Docker

-Requisitos Docker

- <https://babel.es/es/Media/Blog/Abril-2017/Docker-en-Windows-10>

SonarQube

-Métricas SonarQube

- <https://oscarmoreno.com/metricas-de-sonarqube-i/>
- <https://enmilocalfunciona.io/las-tripas-de-sonarqube-metricas-y-como-calcula-el-rating-parte-2/>

-Instalación SonarQube

- <https://testeandosoftware.com/sonarqube-instalacion-basica/>
- <https://community.sonarsource.com/t/installing-sonar-scanner-in-alpine-linux-docker/7010/5>
- <https://www.youtube.com/watch?v=vyZYMD8naB8>

-Uso SonarQube y Sonar Scanner

- https://www.youtube.com/watch?v=cOQHc_-CjA4

-Funcionamiento de Sonar Scanner

- <https://docs.sonarqube.org/latest/analysis/scan/sonarscanner/>