RESUMEN – TRABAJO DE GRADO

AUTORES	**ROBERTO JOSÉ GUERRERRO HERNÁNDEZ**
FACULTAD	**INGENIERIAS**
PLAN DE ESTUDIOS	**INGENIERÍA DE SISTEMAS**
DIRECTOR	**JUAN CAMILO JAIMES FERNÁNDEZ**
TÍTULO DE LA TESIS	**FORTALECIMIENTO DE LA PLATAFORMA WEB POLARIS CORE DE LA EMPRESA WORLD POS SOLUTIONS S.A.S (WPOSS) A TRAVÉS DE LA REESTRUCTURACIÓN Y DESARROLLO DE UN MÓDULO PARA LA GESTIÓN COMERCIAL.**

RESUMEN

(70 palabras aproximadamente)

ESTE PROYECTO DE PASANTÍA REALIZADO EN LA EMPRESA WORLD POS SOLUTIONS S.A.S (WPOSS). EN LA CIUDAD DE OCAÑA, NORTE DE SANTANDER, SE REALIZA CON EL FIN DE SUPLIR UNA NECESIDAD A UNO DE NUESTROS CLIENTES, DONDE ES REQUERIDA UNA PLATAFORMA WEB PARA LA GESTIÓN DE DATAFONOS (POS) Y COMERCIOS Y TÉCNICOS ASOCIADOS, DESARROLLADA EN CON TECNOLOGÍAS JAVASCRIPT PARA LA ADMINISTRACIÓN DE REDES TRANSACCIONALES.

CARACTERÍSTICAS

PÁGINAS: 56	PLANOS:	ILUSTRACIONES: 41	CD-ROM:

FORTALECIMIENTO DE LA PLATAFORMA WEB POLARIS CORE DE LA EMPRESA WORLD POS SOLUTIONS S.A.S (WPOSS) A TRAVÉS DE LA REESTRUCTURACIÓN Y DESARROLLO DE UN MÓDULO PARA LA GESTIÓN COMERCIAL.

AUTOR

ROBERTO JOSE GUERRERO HERNANDEZ (191200)

Trabajo de Grado bajo la modalidad de Pasantía para optar al título de Ingeniero de Sistemas

Director

Ing. Juan Camilo Jaimes Fernández

Especialista en Auditoría de sistemas

UNIVERSIDAD FRANCISCO DE PAULA SANTANDER OCAÑA

FACULTAD DE INGENIERÍAS

INGENIERÍA DE SISTEMAS

Ocaña, Colombia mayo, 2021

Dedicatoria

Primeramente, dedico este logro a Dios por las bendiciones que me ha dado, por darme la fuerza y el valor necesario para seguir adelante.

Con todo amor y cariño a mis padres, por su sacrificio y esfuerzo, por ser mi más grande fuente de motivación e inspiración para seguir adelante, aunque hemos pasado dificultades siempre han estado ahí brindándome su amor y compañía.

A mi familia, quienes están siempre acompañándome en cada etapa de mi vida, por todo su apoyo y buenos deseos, han sido parte fundamental para mi formación como persona.

A los docentes que me acompañaron en el aprendizaje, quienes compartieron sus conocimientos, alegrías y tristezas, y todas aquellas personas que durante estos 5 años estuvieron a mi lado apoyándome.

Agradecimientos

Agradezco al ingeniero Ezequiel y a su equipo de trabajo por todo el apoyo y la paciencia brindada, por darme la oportunidad de desarrollar mi proceso de pasantías en la empresa World Pos Solutions(WPOSS) fue una experiencia muy gratificante.

A la profesora Olga Reyes y toda la oficina de Subdirección académica, quienes siempre estuvieron atentos a cualquier duda presentada, por el apoyo constante durante el desarrollo del proyecto.

A mis jurados y profesores, por todo su conocimiento compartido. Son muchas las personas que hicieron parte de este proceso fueron grandes momentos y experiencias que me llevo de cada uno de ellos.

Índice

Lista de figuras

Lista de Tablas

Resumen

Este proyecto de pasantía realizado en la empresa WORLD POS SOLUTIONS S.A.S (WPOSS). en la ciudad de Ocaña, Norte de Santander, se realiza con el fin de suplir una necesidad a uno de nuestros clientes, donde es requerida una plataforma web para la gestión de Datafonos (POS) y comercios y técnicos asociados, desarrollada en con tecnologías JavaScript para la administración de redes transaccionales.

En este documento se encontrarán todas las actividades que se realizaron con el fin de lograr el objetivo planteado, teniendo en cuenta optar por la mejor solución que se adaptara a las necesidades del cliente, para su posterior implementación, uso continuo y mejoras o ajuste necesarios.

Introducción

Hoy en día los procesos empresariales como los financieros cada vez se vuelven más complejos debido a los cambios que pasan en el mundo como la revolución industrial. Dentro de estos procesos empresariales los procesos del área operativa, comercial, servicios, soporte, ventas entre otros, estos procesos suelen ser difíciles de abordar de manera. Sin embargo, existes soluciones digitales como software de administración que facilitan la ejecución, pero estos al no contemplar las nuevas necesidades de la actualidad pasan a ser ineficientes y en una última instancia obsoletos. El uso de redes transaccionales en Colombia y latino América atreves de sistemas y dispositivos (POS) en general es cada vez mayor donde cada día existe la apertura de un nuevo establecimiento comercial. El uso común del datafono es más que un servicio para pagos con tarjetas ya sean crédito o débito, gracias a su portabilidad y elementos esenciales el cual se compone de una impresora, una pantalla y el teclado. Con esto es suficiente para muchas aplicaciones. Fácilmente en un datafono se pueden integrar diferentes aplicaciones generando todo un sistema completo en donde se tiene acceso a muchos servicios.

Por otro lado, el tema del área operativa o administrativa correspondiente al manejo de información de clientes, comercios, técnicos, seguimiento de actividades, personal, bodegas, insumos y equipos electrónicos que mantienen a una red transaccional toda esa información es manejable para una empresa cuando los datos almacenados tienen una magnitud que se traduce en cientos y miles de registros en base de datos, pero cuando la red transnacional crese y llega a la extensión de un país o a varios países esto se convierte en un gran volumen de información y su procesamiento se vuelve complejo para sistemas tradicionales y este gran volumen necesita, además de ser almacenado también de un análisis para describir comportamientos y hacer toma

de decisiones que pueden llegar a ser vitales. Por eso es importante el uso de nuevas tecnologías como bases de datos NoSQL, procedimientos de análisis de información como la minería de datos y nuevas soluciones de desarrollo de software, con el fin de garantizar que la información cumpla con los objetivos logísticos de la actualidad.

Capítulo 1. Fortalecimiento de la plataforma web polaris core de la empresa World pos solutions s.a.s (WPOSS) a través de la reestructuración y desarrollo de un módulo para la gestión comercial.

1.1. Descripción breve de la empresa

WPOSS es una empresa Multilatina Colombiana, líder en el mercado de soluciones de medios de pago de forma electrónica, que ofrece servicios innovadores transaccionales de forma rápida y automática, con enfoque en la satisfacción de las necesidades del cliente. Está apoyado por un equipo de trabajo dinámico, experto y comprometido con más de 25 años de experiencia en el desarrollo de Software y ventas de dispositivos electrónicos, y que está respaldado con una infraestructura tecnológica de calidad (Fig. 1).

Figura 1. Logo oficial de la empresa. Fuente: World POS Solutions SAS, 2020.

1.1.1. Misión.

Ser aliado estratégico de nuestros clientes en la innovación y evolución de modelos de negocios de soluciones integrales, que incluyen paquetes completos de productos tecnológicos y servicios asociados, cumpliendo con los estándares de seguridad, altos niveles de servicio y calidad.

1.1.2. Visión.

Ser la primera opción como socio estratégico y tecnológico de nuestros clientes en el mercado de América Latina.

1.1.3. Objetivos de la empresa.

- Obtener ventas de 15.000 millones de pesos.
- Tener un centro de desarrollo con altos estándares enfocado a brindar soluciones tecnológicas y de pago.
- Implementar un sistema de gestión de calidad.

1.2. Diagnóstico inicial de la dependencia asignada

Tabla 1. Matriz DOFA

	FORTALEZAS (F)	DEBILIDADES (D)
MATRIZ DOFA	- El área de IP cuenta con personal capacitado para el desarrollo de aplicaciones web. - Se hace uso de la metodología SCRUM para el trabajo colaborativo entre las áreas de desarrollo. - Se cuenta con instalaciones amplias y sistemas de cómputo de última generación.	- Se cuenta con documentación deficiente y/o desactualizada de software que ha sido desarrollado en el pasado y aún requiere mantenimiento. - El tiempo de desarrollo se ve ralentizado debido a que el personal dispuesto para el desarrollo debe rotar entre proyectos nuevos y antiguos.
OPORTUNIDADES(O)	ESTRATEGIA(FO)	ESTRATEGIA(DO)

- Interés por parte de los clientes en adquirir nuevos productos y servicios de la empresa. - Capacitaciones disponibles para el personal contratado en las distintas áreas de desarrollo.	Desarrollar productos tecnológicos que generen un valor agregado a los modelos de negocio de los clientes y que a su vez aumente la competitividad de la empresa.	Contratar a una cantidad mayor de desarrolladores con los cuales poder priorizar nuevos proyectos sin abandonar los que ya existen, a la vez que se mantiene la documentación actualizada.
AMENAZAS(A)	ESTRATEGIA(FA)	ESTRATEGIA(DA)
- Cancelación del desarrollo del producto. - Cambios inesperados por parte del cliente. - Cambios en las tecnologías y lenguajes de programación usados.	Implementar una arquitectura de software basada en microservicios que permita responder a cambios y nuevas funcionalidades, así como a la puesta en marcha de nuevas tecnologías sin modificar las existentes.	Actualizar la documentación de los proyectos de desarrollo antiguos de la empresa, de manera que estén a la par con los nuevos desarrollos, disminuyendo tiempos de transición entre proyectos.

1.2.1. Planteamiento del problema.

En la actualidad, la empresa WPOSS ofrece una gama de productos y servicios enfocados a brindar soluciones de pago para sus clientes en Latinoamérica. Sin embargo, la necesidad de desarrollar productos y ofrecer servicios novedosos que le permitan aumentar sus ingresos y a la larga tener una mayor competitividad, le hace requerir un equipo de desarrollo capacitado, organizado y suficiente, así como también adquirir la habilidad de apropiarse de las nuevas tecnologías tan pronto como demuestran representar un valor en el mercado. Es así como WPOSS se ve en la necesidad de aumentar su equipo de desarrollo de software que le permita llevar a cabo un nuevo proyecto (Polaris Core) que haga uso de lenguajes de programación potentes como JavaScript, Java y tecnologías como Node.js y Angular, el cual pueda ser incorporado dentro de los demás servicios y aplicaciones de software ofertados por la empresa. Este nuevo proyecto (Plataforma web) es requerido por uno de nuestros clientes debido a la

necesidad operacional y logística del manejo de personal, clientes, técnicos, dispositivos, quipos de cómputo, seguimiento de actividades que comprenden una red transaccional que comprenden una gran cantidad de comercios vinculados. Por consiguiente, la implementación de esta plataforma, el cliente podrá tener mayor dominio en su red transaccional y además explorar resultados estadísticos para la toma de decisiones con el fin de fidelizar a sus respectivos clientes (comercios) de una manera intuitiva y controlada por medio de una plataforma web.

1.2.2. Descripción de la estructura organizacional.

Con el fin de cumplir con los objetivos planteados por la empresa, la estructura organizacional de WPOSS está dada de la siguiente manera (Fig. 2):

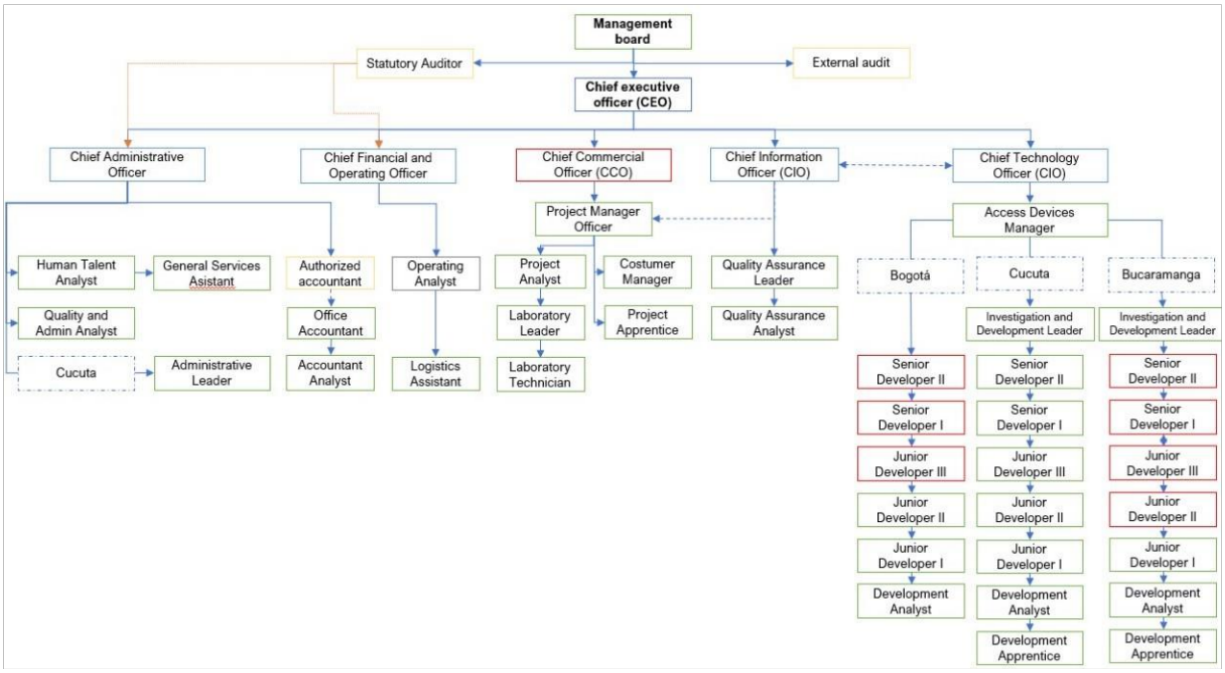

Figura 2. Estructura organizacional Fuente: World Pos Solutions SAS, 2020

1.2.3. Descripción de la dependencia y/o proyecto al que fue asignado.

El área de Desarrollo se encarga de llevar a cabo los proyectos software de la empresa los cuales serán uno de los productos de valor que se les dará a los clientes. Esta área está compuesta

por subáreas las cuales son QA (Quality Assurance), UX (User Experience), POS (Dispositivos Móviles), PMO (Project Management Office) y por último el área de IP (Infrastructure and Platform). Es en la subárea IP la que se me fue asignado, en ella se desarrollan plataformas para la administración y uso de los dispositivos de pago, actualmente se están llevando a cabo varios proyectos de desarrollo entre los cuales se encuentra Polaris Core, el cual consiste en el desarrollo de una plataforma de gestión web de Datafonos Android y comercios. El proyecto surge como una necesidad de los clientes de WPOSS de generar un mayor valor a sus modelos de negocio, de igual forma beneficiar a la propia empresa, adquiriendo mayor competitividad en su nicho de mercado.

1.3. Objetivos de la pasantía

1.3.1. General.

Fortalecer la plataforma web POLARIS CORE de la empresa WORLD POS SOLUTIONS S.A.S (WPOSS) a través de la reestructuración y desarrollo de un módulo para la gestión comercial.

1.3.2. Específicos.

- Estudiar el entorno de desarrollo de las tecnologías y herramientas de la empresa.
- Desarrollar las actividades de programación en la plataforma Polaris Core.
- Realizar las pruebas necesarias que verifique el correcto funcionamiento de la aplicación.

1.4. Descripción de las actividades a desarrollar en la misma

Tabla 2.
Descripción de actividades

Objetivo General	Objetivos Específicos	Actividades a desarrollar
Fortalecer la plataforma web POLARIS CORE de la empresa WORLD POS SOLUTIONS S.A.S (WPOSS) a través de la reestructuración y desarrollo de un módulo para la gestión comercial.	Estudiar el entorno de desarrollo de las tecnologías y herramientas de la empresa.	• Revisión de los requerimientos funcionales. • Estudio del framework Angular, Tecnologia Node JS y Framework Express JS. • Estudio del sistema de almacenamiento Cassandra • Capacitación por parte de la empresa en el uso Git y herramienta Sourcetree para manejo de repositorios y código versionado. • Capacitación por parte de la empresa en la herramienta de gestión de tareas y proyectos Open Project usando la metodología Scrum.
	Desarrollar las actividades de programación en la plataforma Polaris Core.	• Revisión de la arquitectura y ficheros del prototipo de Polaris Core desarrollado con las tecnologías de Angular, Node JS, Express JS, lenguaje Java Script y base de datos NoSQL Cassandra. • Desarrollo de actividades de mejora en diseño y funcionamiento. • Implementación y desarrollo del prototipo Polaris Core al cliente BCP.
	Realizar las pruebas necesarias que verifique el correcto funcionamiento de la aplicación.	• Realización de pruebas funcionales, no funcionales e integración. • Corrección de posibles errores o inconsistencias. • Documentar los servicios desarrollados.

Capítulo 2. Enfoques referenciales

2.1. Enfoque conceptual

- **Metodología ágil**

Según Texier & Bermúdez, IWeb demanda un proceso de software incremental y evolutivo. Pressman también señala que el modelo en las primeras versiones puede ser un modelo en papel o un prototipo, y durante las últimas iteraciones se producen versiones cada vez más completas del sistema diseñado.

- **Diseño y arquitectura**

(Estruga, F, 2 de octubre de 2013) escribe, "Determinar cómo funcionará de forma general sin entrar en detalles incorporando consideraciones de la implementación tecnológica, como el hardware, la red, etc. Consiste en el diseño de los componentes del sistema que dan respuesta a las funcionalidades descritas en la segunda etapa también conocidas como las entidades de negocio. Generalmente se realiza en base a diagramas que permitan describir las interacciones entre las entidades y su secuenciado".

- **Arquitectura cliente servidor**

Es un modelo en el que las tareas son distribuidas entre el cliente y el servidor, en donde el cliente genera peticiones al servidor el cual proporciona la repuesta. Según Marini (2012) "El modelo Cliente/Servidor permite diversificar el trabajo que realiza cada aplicación, de forma que los Clientes no se sobrecarguen, cosa que ocurriría si ellos mismos desempeñan las funciones que le son proporcionadas de forma directa y transparente".

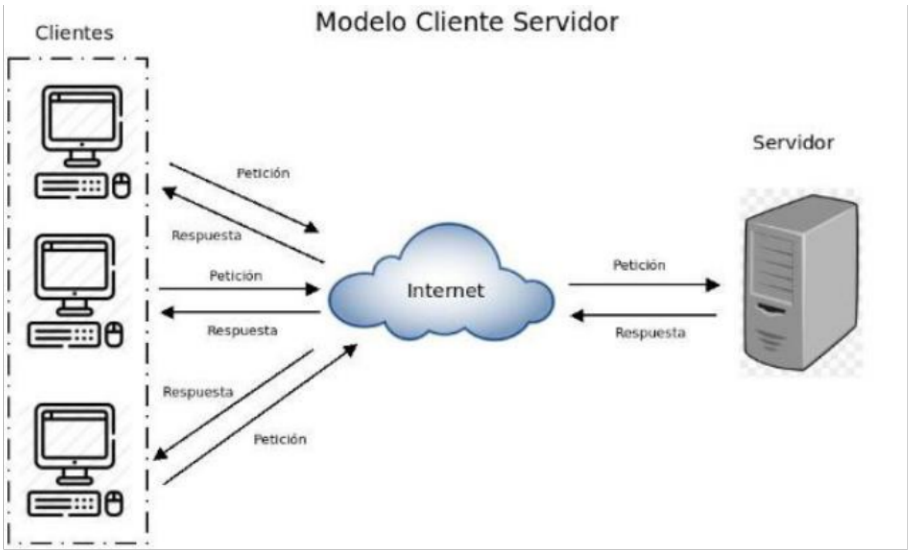

Figura 3. Funcionamiento de la arquitectura cliente servidor. Recuperado de http://redespomactividad.weebly.com/modelo-cliente-servidor.html

- **Modelo vista controlador**

Es un estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

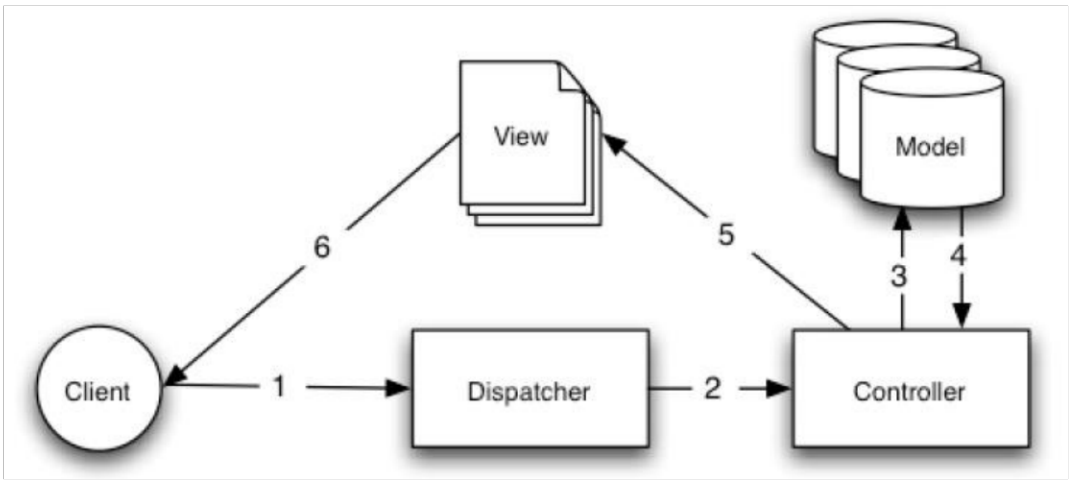

Figura 4. Funcionamiento de la modelo vista/controlador. Recuperado de https: book.cakephp.org/2.0/cake/es/cakephp-overview/understanding-model-view-controller.html

- **Visual Studio Code**

Es un editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código. (Visual Studio code, 2020).

- **Bitbucked**

Es una herramienta para el control de versiones de uno o más proyectos de software creados en base a la colaboración entre miembros del equipo que utilizan Git. Bitbucked es una de las muchas alternativas de control de versiones entre otras herramientas del mercado, como github, gitlab (Vázquez, Gómez, & Serrano, 2019).

- **Scrum**

Es una metodología que las personas o equipos de trabajo puede implementar para resolver problemas complejos. Altamente adaptable mientras entrega productos con el mayor valor de producción. Una de las ventajas o bondades de Scrum es que es ligero y fácil de entender (Schwaber & Jeff, 2013).

- **Angular**

Angular es un marco de diseño de aplicaciones y una plataforma de desarrollo para crear aplicaciones de una sola página eficientes y sofisticadas. (Angular.io, 2020).

- **Node.js**

Ideado como un entorno de ejecución de JavaScript orientado a eventos asíncronos, Node.js está diseñado para crear aplicaciones network escalables. (Node.js, 2020).

- **Cassandra**

Apache Cassandra NoSQL distribuida y basada en un modelo de almacenamiento de «clave-valor», de código abierto que está escrita en Java. Permite grandes volúmenes de datos en forma distribuida. Por ejemplo, lo usa Twitter para su plataforma. Su objetivo principal es la escalabilidad lineal y la disponibilidad. (Apache Software, 2020).

- **Git**

Git es un sistema de control de versiones distribuido gratuito y de código abierto diseñado para manejar todo, desde proyectos pequeños hasta muy grandes, con velocidad y eficiencia. (Git, 2020).

- **Express**

Express es una infraestructura de aplicaciones web Node.js mínima y flexible que proporciona un conjunto sólido de características para las aplicaciones web y móviles. (Express.js, 2020).

- **TypeScript**

TypeScript es un lenguaje de código abierto que se basa en JavaScript, una de las herramientas más utilizadas del mundo, al agregar definiciones de tipos estáticos. (typescriptlang, 2020).

- **JavasScript**

JavaScript es un lenguaje de programación o de secuencias de comandos que te permite implementar funciones complejas en páginas web, cada vez que una página web hace algo más que sentarse allí y mostrar información estática para que la veas, muestra oportunas actualizaciones de contenido, mapas interactivos, animación de Gráficos 2D/3D, desplazamiento

de máquinas reproductoras de vídeo, etc., puedes apostar que probablemente JavaScript está involucrado. (Mozilla Developer, 2020).

- **NoSQL**

Las bases de datos NoSQL están diseñadas específicamente para modelos de datos específicos y tienen esquemas flexibles para crear aplicaciones modernas. Las bases de datos NoSQL son ampliamente reconocidas porque son fáciles de desarrollar, por su funcionalidad y el rendimiento a escala. Esta página incluye recursos que lo ayudan a comprender mejor las bases de datos NoSQL y comenzar a usarlas (Amazon aws, 2020).

- **API REST**

Una API de transferencia de estado representacional (REST), o API de RESTful, es una interfaz de programación de aplicaciones que se ajusta a los límites de la arquitectura REST: Una API o interfaz de programación de aplicaciones es un conjunto de definiciones y protocolos que se usa para diseñar e integrar el software de aplicaciones. Suele considerarse como el contrato entre un proveedor de información y un usuario, donde se establece el contenido que se requiere del consumidor (la llamada) y el que necesita el productor (la respuesta). Por ejemplo, una API de servicio meteorológico podría solicitar que el usuario escribiera un código postal y que el productor diera una respuesta en dos partes: la primera sería la temperatura máxima y la segunda, la mínima. (redhat, 2020).

2.2. Enfoque legal.

2.2.1. Ley 23 de 1992 sobre los derechos de autor.

En esta ley se instituye que "los autores de obras literarias, científicas y artísticas gozarán de protección para sus obras en la forma prescrita por la presente ley y, en cuanto fuere compatible con ella, por el derecho común. También protege esta ley a los intérpretes o ejecutantes, a los productores de fonogramas y a los organismos de radiodifusión, en sus derechos conexos a los del autor". Además, "los derechos de autor recaen sobre las obras científicas, literarias y artísticas las cuales se comprenden todas las creaciones del espíritu en el campo científico, literario y artístico, cualquiera que sea el modo o forma de expresión y cualquiera que sea su destinación, tales como: los libros, folletos y otros escritos". En resumen, esta ley establece las disposiciones generales y especiales que regulan la protección del derecho de autor en Colombia.

2.2.2. Ley de protección de información y de los datos.

El congreso de Colombia decreta un capítulo de los atentados contra la confidencialidad, la integridad y la disponibilidad de los datos y de los sistemas informáticos, donde se establece "Acceso abusivo a un sistema informático. El que, sin autorización o por fuera de lo acordado, acceda en todo o en parte a un sistema informático protegido" El uso de software malicioso "El que, sin estar facultado para ello, produzca, trafique, adquiera, distribuya, venda, envíe, introduzca o extraiga del territorio nacional software malicioso u otros programas de computación de efectos dañinos." Y el artículo 269F donde establece "Violación de datos personales. El que, sin estar facultado para ello, con provecho propio o de un tercero, obtenga, compile, sustraiga, ofrezca, venda, intercambie, envíe, compre, intercepte, divulgue, modifique o

emplee códigos personales, datos personales contenidos en ficheros, archivos, bases de datos o medios semejantes."

Mientras que en la ley 44 se regula todo en cuanto a los derechos de autor, este recae sobre el acceso abusivo o manipulación inadecuada de la información en sistema informático.

2.2.3. Reglamento trabajo de grado modalidad pasantía.

2.2.3.1. Requisitos para realizar trabajo de grado en la modalidad de pasantías.

El estudiante que elija desarrollar su trabajo de grado en la modalidad Pasantías, deberá cumplir con el requisito de aprobación en todas las asignaturas correspondientes al pensum académico del Plan de Estudios. Dentro de esta normatividad se establece "es requisito sine qua non, que el estudiante haya terminado todas las asignaturas del pensum, aunque el acuerdo 065/96 establece que el trabajo de grado se puede empezar habiendo aprobado el 60% de los créditos, para el caso de la Pasantía, los estudiantes deben tener dedicación de tiempo completo. Por lo tanto, no es posible hacer la pasantía y cursar simultáneamente otras asignaturas o cursos, sólo en casos excepcionales previa autorización del Comité Curricular respectivo" además de ellos "Deberá tener un director de trabajo de grado, que acredite profesión titulada a fin con la formación del estudiante pasante, con tarjeta profesional en caso que se requiera, teniendo en cuenta que si es externo deberá anexar la hoja de vida con sus respectivos soportes al Plan de Trabajo, de modo que se pueda evidenciar que se tiene profesión y experiencia afín a las actividades que desarrollará el pasante".

En resumen, este acuerdo establece los requerimientos que debe cumplir un estudiante que opta por la modalidad de pasantías.

Capítulo 3. Informe de cumplimiento de actividades

3.1. Socialización de reglas y lógica de negocio

En esta parte se nos socializa las reglas y lógica de negocios enfocadas a una red transaccional. Por mi parte se entra en una revisión y análisis de cada uno de los aspectos y puntos relevantes que se me expuso. Además, Esta socialización tiene el objetivo de dar un mapa general de la funcionalidad del sistema todo eso con el fin de abordar actividades y desarrollarlas satisfactoriamente.

3.2. Estudio del framework Angular, Tecnologia Node JS y Framework Express JS

En el proyecto de desarrollo antes de realizar las actividades se me indicó las tecnologías a estudiar, para ello se necesitó de dos (2) a tres (3) semanas de estudio con el fin de conseguir las bases teóricas y prácticas.

En el proceso de aprendizaje de las tecnologías se me brindo fuentes y documentaciones en las cuales fueron fundamentales para mi curva de aprendizaje y realización de actividades de desarrollo, estas fuentes son las siguientes:

- Documentación de angular (Angular.io, 2020), Apartado documentación.
- Documentación Node.js (Node.js, 2020), Apartado documentación.
- Documentación express.js (Express.js, 2020), Apartado documentación.

Además de las fuentes documentales se nos brindó fuentes a través de la plataforma Udemy (Udemy, 2021), en esta plataforma se tomaron los siguientes cursos:

- Angular: De cero a experto impartido por el instructor Fernando Herrera.

- Angular Avanzado: Lleva tus bases al siguiente nivel – MEAN, impartido por el instructor Fernando Herrera.
- Node: De cero a experto. Rest, despliegues, Heroku, Mongo, Git, GitHub, Sockets, archivos, JWT y mucho más para ser un experto en Node, impartido por el instructor Fernando Herrera
- Introducción al desarrollo backend con Node.js y Express impartido por el instructor Darwin morocho.

3.3. Estudio del sistema de almacenamiento Cassandra

En el apartado de base de datos se hizo un estudio más profundo debido a la utilización de base de datos NoSQL y además también por el uso de minería de datos, el motor de base de datos que se utilizo fue Cassandra que trata de un software NoSQL distribuido y basado en un modelo de almacenamiento de «clave-valor», (Apache Software, 2020).

En el proceso de aprendizaje de Cassandra se me dieron las indicaciones para estudiar la documentación oficial impartida por Apache foundation (Cassandra, 2021).

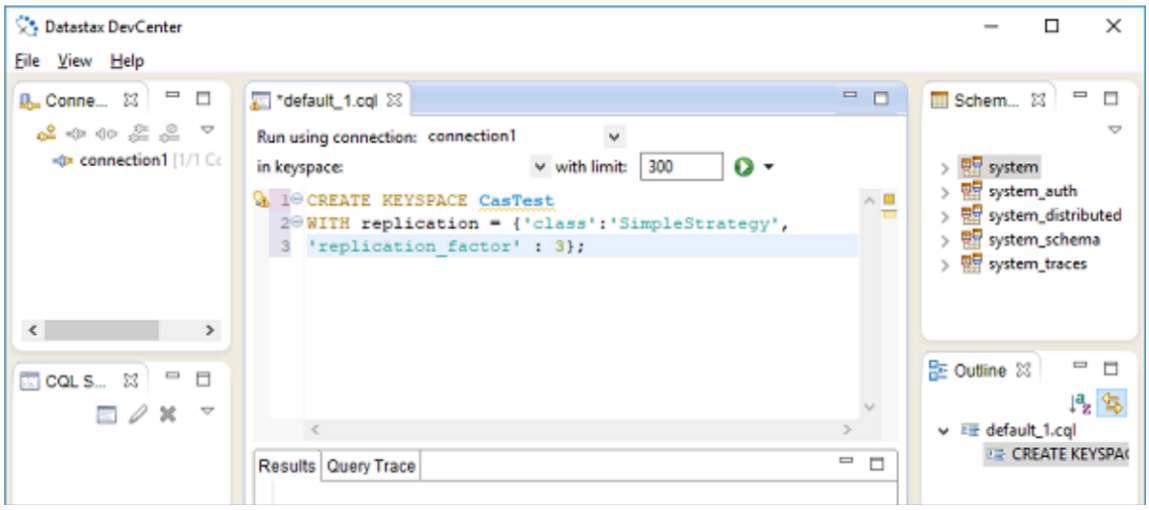

Figura 5. DataStax, Interfax de la herramienta o software DataStax que tiene como intención manejar la administración de base de datos NoSQL Cassandra. Fuente: Roberto Guerrero.

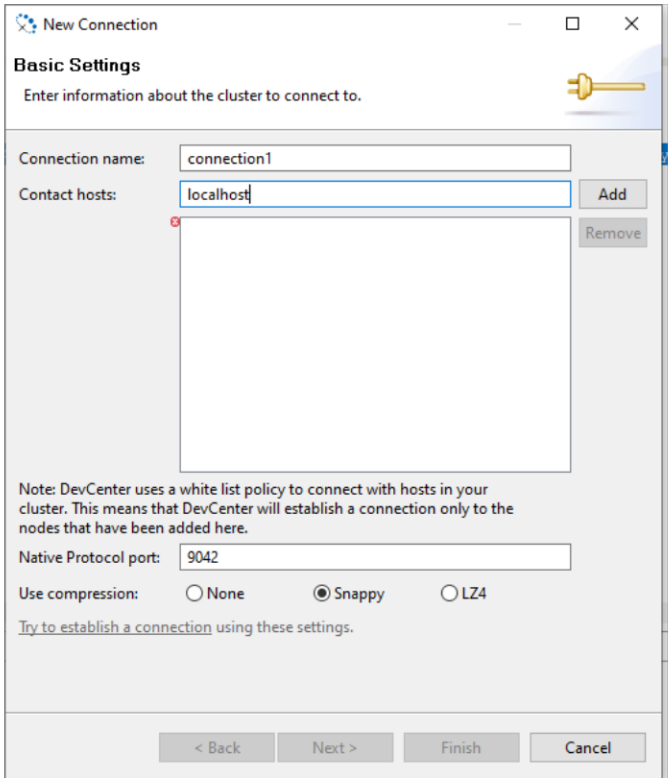

Figura 6. DataStax, Interfaz de configuración del software DataStax, apartado que permite definir los parámetros de conexión a la base de datos cassandra y su manipulación para la administración de información. Fuente: Roberto Guerrero.

3.4. Capacitación por parte de la empresa en la herramienta de gestión de tareas y proyectos Open Project usando la metodología Scrum

En el campo de gestión de tareas y proyectos la empresa brinda una capacitación en la metodología ágil scrum que es muy práctico para socializar y hacer los respectivos seguimientos con las actividades asignadas a cada uno de los miembros del equipo de desarrollo. Por otro lado, una capacitación para el manejo de la herramienta OpenProject que soporta la gestión de tareas, seguimiento de errores, gestión de requisitos, Scrum, planificación de productos, diagramas Gantt, wikis, gestión de reuniones, seguimiento de tiempo y creación de informes de costos, presupuestos y mucho más.

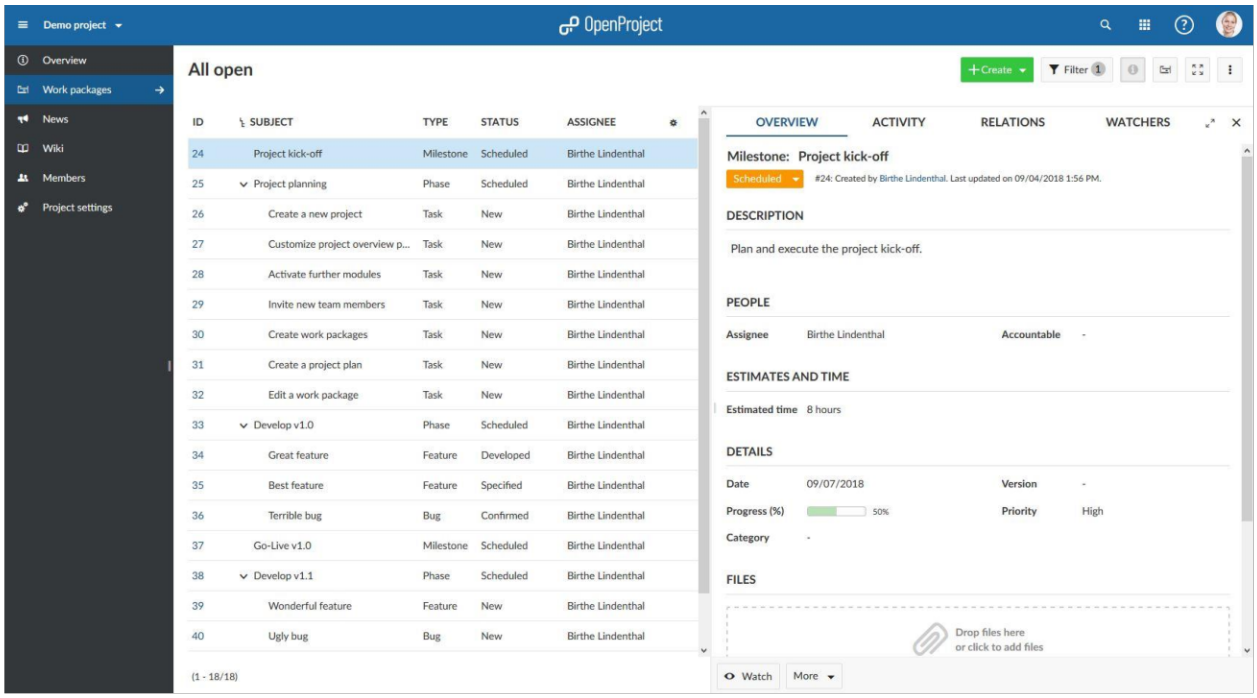

Figura 7. OpenProject, Software de gestión de tareas. Fuente: Roberto Guerrero.

3.5. Capacitación por parte de la empresa en el uso Git y herramienta Sourcetree para manejo de repositorios y código versionado

El control de versiones es una de las tareas fundamentales para la administración de un proyecto de desarrollo de software en general. Surge de la necesidad de mantener y llevar control del código que vamos programando, conservando sus distintos estados. Es absolutamente necesario para el trabajo en equipo. Por tal motivo WPOSS brinda una capacitación como requisito indispensable para la administración de proyectos de desarrollo.

Git es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente. Su propósito es llevar registro de los cambios en archivos de computadora y coordinar el trabajo que varias personas realizan sobre archivos compartidos. (WPOSS, 2018)

La metodología de trabajo está basada en la división de las distintas etapas de producción de software en distintas ramas del repositorio, descritas a continuación

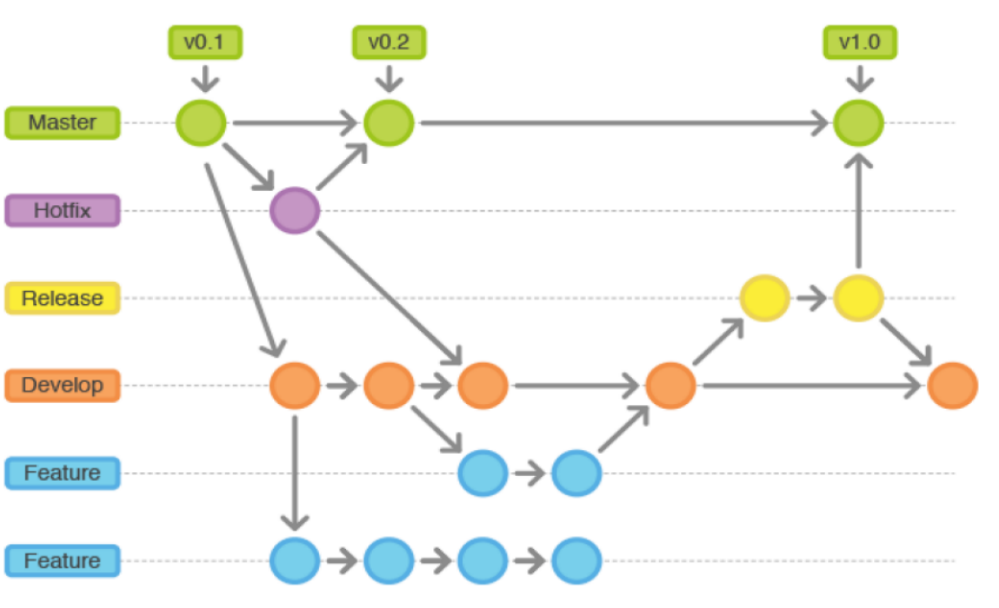

Figura 8. Organización de ramas en un proyecto de desarrollo implementado Git. Fuente: WPOSS.

- **Master.**

Es la rama principal. Contiene el repositorio que se encuentra publicado en producción, por lo que debe estar siempre estable. La rama master es administrada por el líder del proyecto.

- **Feauture.**

Las ramas de feature se usarán para desarrollar nuevas funcionalidades, se crearán a partir de la rama develop y al terminar con esta funcionalidad nueva, se tiene que fusionar otra vez con develop. Cada nueva funcionalidad se debe realizar en una rama nueva.

- **Realese.**

Las ramas release se usarán para lanzar una nueva versión de nuestro proyecto. Se usarán solo para los últimos retoques antes de liberar la nueva versión, como cambiar el número de esta,

y crearán a partir de la rama develop y se fusionarán tanto con master (para poder ser desplegadas en producción) como con develop.

- **Develop.**

La rama surge de la última release de master, es la rama que contiene las características (features) en desarrollo en una iteración, esta rama será posteriormente parte de Release.

- **Hotflix.**

Las ramas "hotfix" se usarán para esos cambios rápidos que queremos realizar como arreglar un bug sencillo en producción mientras que al mismo tiempo estamos desarrollando una nueva funcionalidad y queremos desplegar este arreglo lo antes posible. Se crean a partir de la rama master y se fusionan con master y develop.

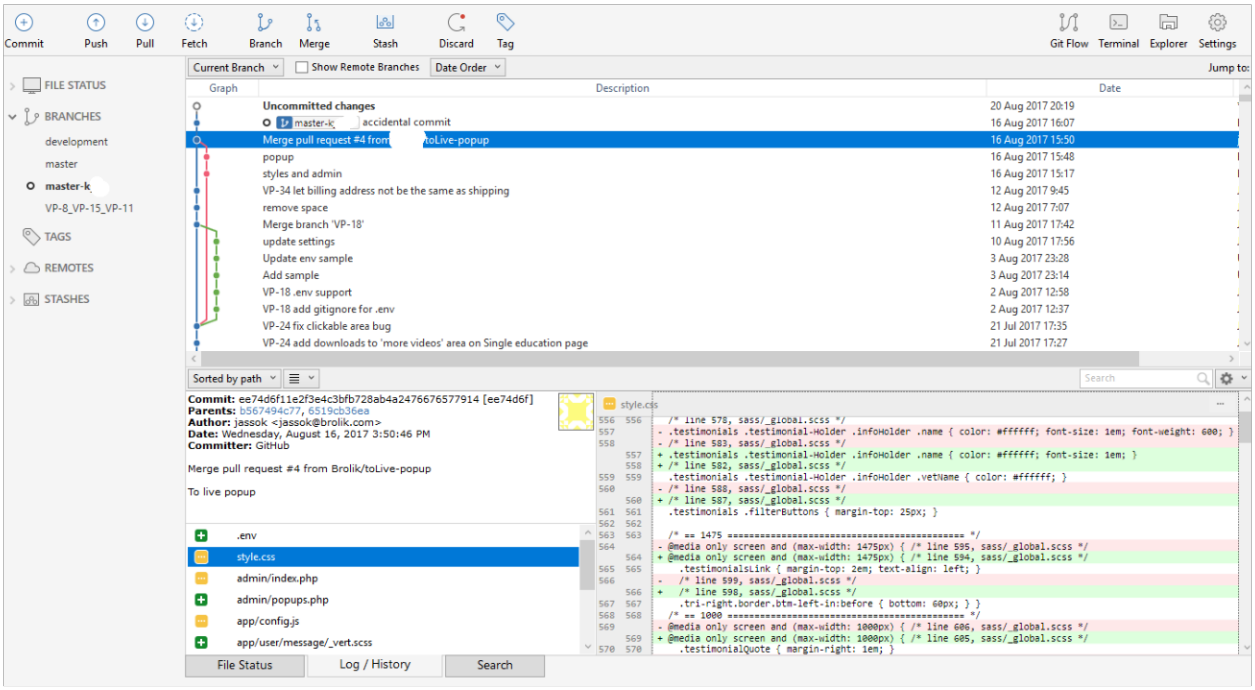

Figura 9. Sourcetree, software grafico para el manejo de Git. Fuente: Roberto Guerrero.

3.6. Revisión de la arquitectura del prototipo de Polaris Core desarrollado con las tecnologías de Angular, Node JS, Express JS, lenguaje Java Script y base de datos NoSQL Cassandra

El propósito de mi labor en el proyecto de Polaris Core es ejecutar actividades de mantenimiento y soporte, creación de nuevas funcionalidades y módulos, a partir de ahí se trabajó sobre un proyecto base funcional ya desarrollado el cual ya tiene denominación como prototipo funcional y posterior hará parte de la adquisición de los clientes modificado a los requerimientos y reglas de negocio que el cliente fije.

En el desarrollo de actividades antes de iniciar se nos dio una introducción en el cual se dio a exponer la arquitectura del frontend y backend.

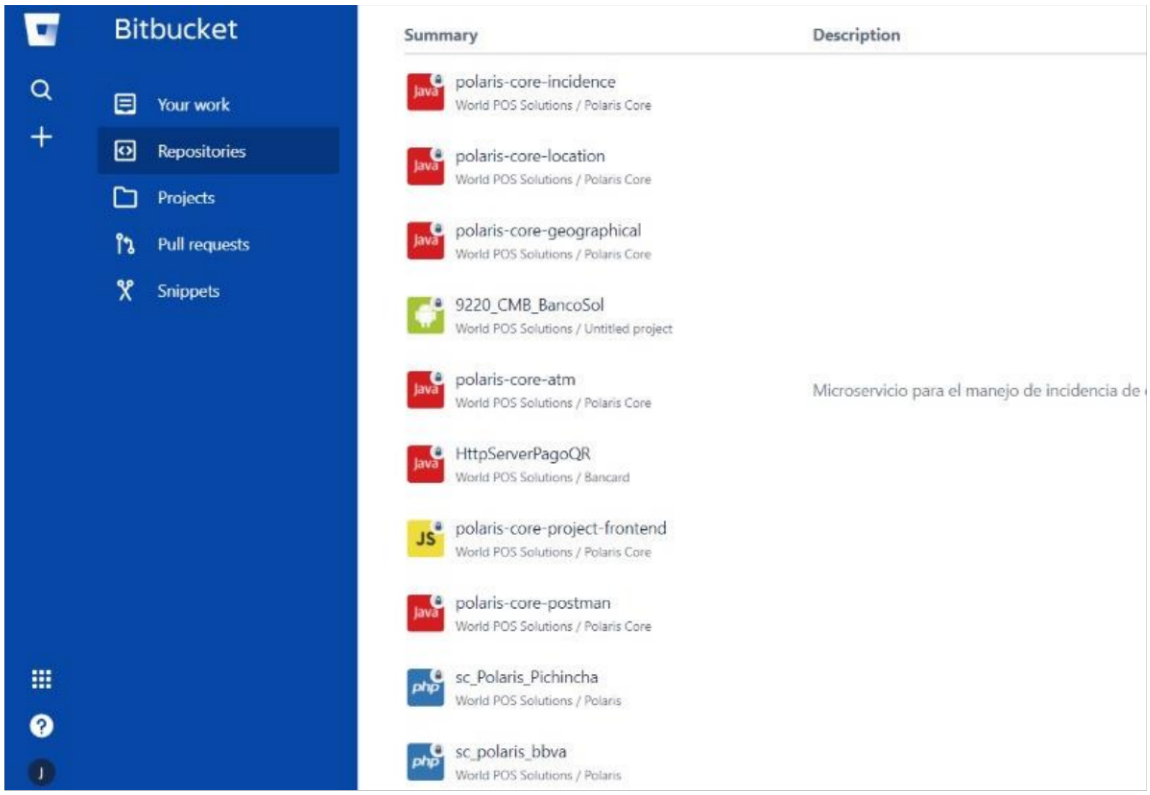

Figura 10. Repositorios Bitbucket, Servicio de alojamiento para el manejo de versiones del proyecto este servicio es proporcionado y administrado por la organización. Fuente: Roberto Guerrero.

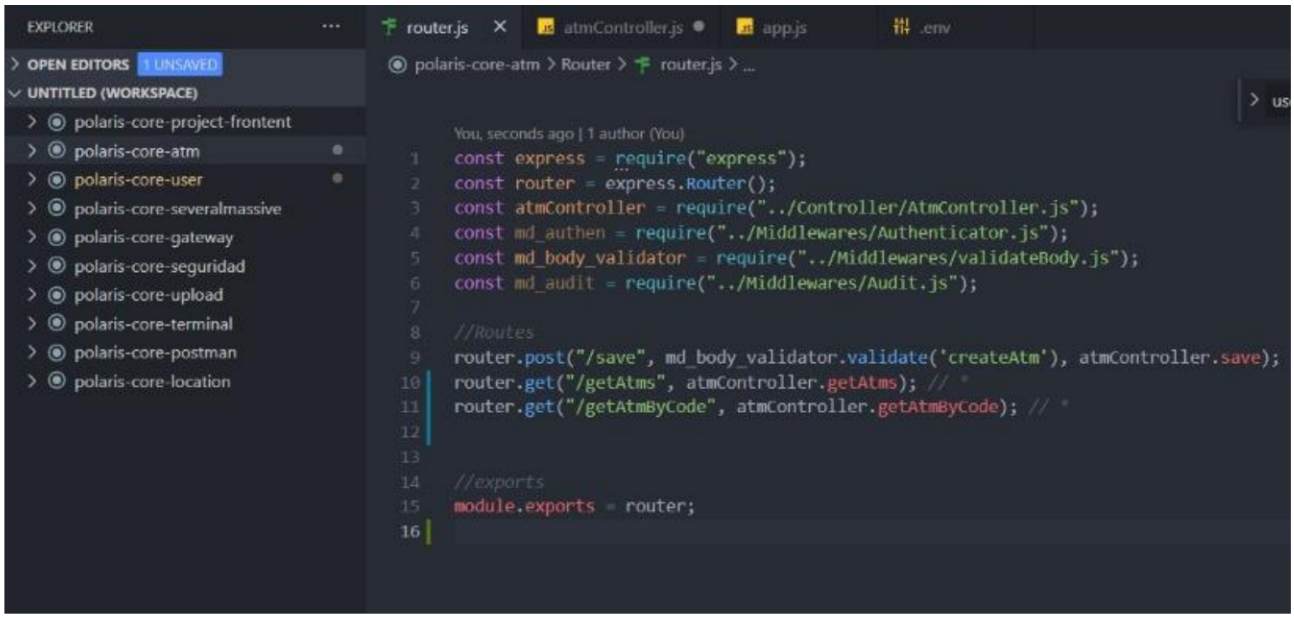

Figura 11. Microservicios en Visual Studio Code. Pantallazo del proyecto en desarrollo donde se evidencia la manipulación de los microservicios. Fuente: Roberto Guerrero

El proyecto de Polaris Core se basa en una arquitectura de microservicios que son los siguientes:

- **Polaris-core-user**.

Este microservicio su funcionalidad es administrar el CRUD de usuarios. Además, incorpora la parte de la autenticación del sistema.

- **Polaris-core-location.**

Este microservicio contrala la información geográfica y el CRUD de información de los comercios.

- **Polaris-core-terminal.**

Este microservicio contrala y administra la información relacionada con los terminales (Dispositivos hardware o datafonos).

- **Polaris-core-upload.**

  Este microservicio es el encargado de controlar y administrar las subidas de archivos.

- **Polaris-core-gateway.**

Este microservicio es un sistema intermediario que proporciona una interfaz API REST o WebSocket para hacer de enrutador desde un único punto de entrada, el API Gateway, hacia un grupo de microservicios y/o API de terceros definidos. Interactúa como puerta de enlace "Gateway".

- **Polaris-core-postman.**

Este microservicio se encarga de la emisión y gestión de correos y notificaciones que se generan de incidentes de la aplicación.

- **Polaris-core-incidence.**

Este microservicio se encarga de la gestión de incidencias que se generan a través de los dispositivos, datafonos y técnicos.

- **Polaris-core-seguridad.**

Este microservicio es un componente que se encarga de la seguridad de la aplicación, pero a través de procesos de terceros como la integración Microsoft Azure y keycloack. En este microservicio se contrala la autenticación, token de seguridad, manejo de sesiones, control y restricción de usuarios. El microservicio es un componente opcional que suele ser un requerimiento pedido por cliente con el fin de que la seguridad tenga más fiabilidad.

3.7. Desarrollo de actividades de mejora en diseño y funcionamiento

Posterior a la introducción de las tecnologías y arquitecturas de proyecto de desarrollo, se me asignaron actividades de mejora en diseño y funcionamiento tales como:

- **Validaciones de formularios.**

En los formularios se encontraba que las validaciones no eran lo suficientemente optimas debido a que la lógica se programaba con condicionales en typeScript aunque su aplicación es correcta y no carece de errores, pero al utilizar angular el desarrollador tiene muchas opciones de validación y el framework trae clases incorporadas para tratar con validaciones de inputs y mucho elementos que incorporan un formulario.

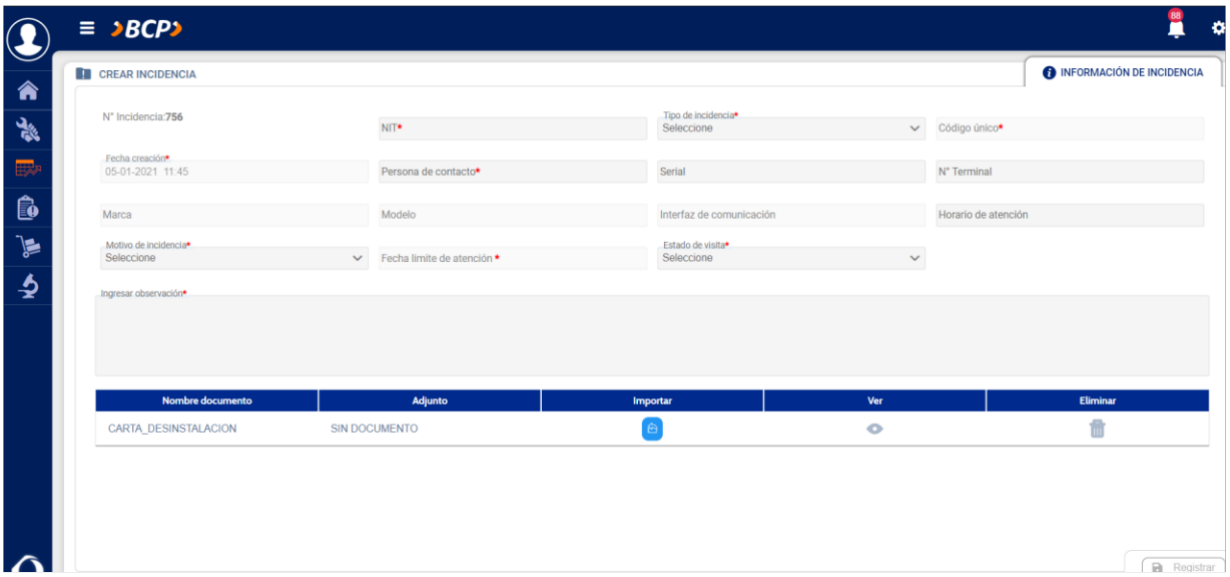

Figura 12. formulario Polaris Core.

- ***Creaciones de nuevas vistas.***

En esta actividad se forman varias actividades y nace de las reglas de negocio impuestas por el cliente y en el requerimiento se pedía la creación de un nuevo módulo denominado gestión comercial este módulo gestionaba información que solo podía ser vista por usuarios comerciales.

- ***Creación de filtros checkbox.***

En un principio la gran mayoría de módulos que listaban en tablas incidencias no tenían filtros para obtener resultados específicos, para poder realizar esta actividad se necesitó del componente multiselect de la librería primeng basada en angular.

```
<p-multiSelect [options]="cities1" [(ngModel)]="selectedCities1"></p-multiSelect>

<p-multiSelect [options]="cities2" [(ngModel)]="selectedCities2" optionLabel="name"></p-multiSelect>
```

Figura 13, Documentación primeNg, sección de código para la implementación de funcionalidad multi selección. Fuente*:* primeng.

```
import {SelectItem} from 'primeng/api';

interface City {
    name: string,
    code: string
}

export class MyModel {

    cities1: SelectItem[];

    cities2: City[];

    selectedCities1: City[];

    selectedCities2: City[];

    constructor() {
        //SelectItem API with label-value pairs
        this.cities1 = [
            {label:'New York', value:{id:1, name: 'New York', code: 'NY'}},
            {label:'Rome', value:{id:2, name: 'Rome', code: 'RM'}},
            {label:'London', value:{id:3, name: 'London', code: 'LDN'}},
            {label:'Istanbul', value:{id:4, name: 'Istanbul', code: 'IST'}},
            {label:'Paris', value:{id:5, name: 'Paris', code: 'PRS'}}
        ];
```

Figura 14. Documentación primeng, Ilustración de cómo se manipula la información es una estructura de multi selección a través objetos JSON. Fuente: primeng.

Figura 15. Tabla gestión de incidencias, En esta imagen se ve la implementación del objeto multiselect para la manipulación de filtros en tablas. Polaris Core.

- **Rediseño de vistas**.

Las actividades que tenían que ver con el rediseño eran cambios que se tenían como mejoras y no eran relevantes en los requerimientos y entregas a producción. En estas actividades se cambiaron botones, inputs con algunas animaciones de validación, aspectos en tarjetas modales, ventanas emergentes modales, coherencia en los colores de gráficos estadísticos y cambios en imágenes que dieran mejor experiencia de usuario.

- **Cambio de versión de la librería primeNg, con el fin de optimizar el rendimiento y corregir errores de los componentes del frontend.**

En esta actividad se llevaron a cabo cambios de mejora en rendimiento debido a que el proyecto utilizaba librerías que estaban desactualizadas y en espacial los componentes de la librería primeng que se basaban en Angular 6.0 y no eran compatibles con la librería de angular 10.0 que se instaló para el proyecto.

- **Cambios de las propiedades de las tablas y elementos de formularios debido al cambio de librería primeng.**

Al cambiar la versión de la librería de primeng muchos elementos como formularios no encajaban en la versión de la librería y el problema radicaba en la definición de atributos que se encontraba en las etiquetas HTML, para poder solucionar esa incompatibilidad se llevó a cabo una tarea de investigación en las documentaciones de la librería primeng.

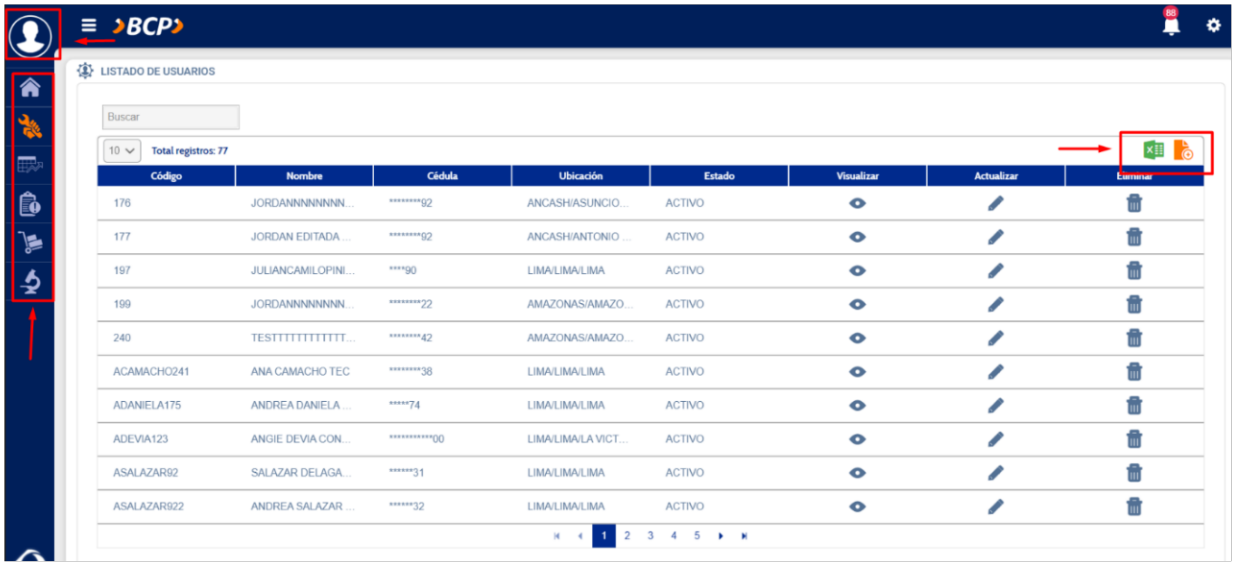

Figura 16. Gestión de usuarios, tablas de registros y aspectos visuales que se mejoraron. Fuente: WPOSS.

3.8. Corrección de errores e inconsistencias en la plataforma

En esta fase podemos tener correcciones en dos momentos, bien sea cuando enviamos versión a nuestra área de QA y ellos nos reportan los errores o ajustes que tenemos que hacer o bien desde nuestro cliente, ellos también cuentan con un área de QA donde evalúan y realizan una serie de pruebas más profundas y detalladas antes de poner en producción la aplicación. Una vez realizada las pruebas por parte de nuestro cliente, nos envían un reporte de los errores o ajustes a corregir y solicitan bien sea de parte de ellos o de nuestro equipo de desarrollo una reunión para aclarar en detalle los puntos a corregir.

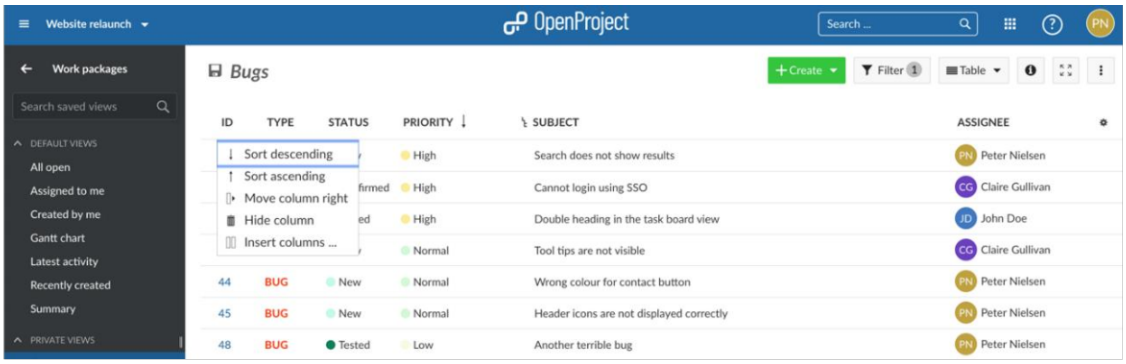

Figura 17. OpenProject errores, Herramienta donde publican errores de los resultados de la pruebas y testeos. Fuente: Roberto Guerrero.

3.9. Implementación y desarrollo del prototipo Polaris Core al cliente BCP

El producto Polaris Core, es una plataforma muy dinámica a los cambios, porque se pensó con el fin de venderse a varios clientes. Ahora, en esta fase con el equipo de desarrollo se planifica y se socializa la incorporación del cliente BCP (Banco crédito del Perú), esta incorporación trae una gran serie de cambios tanto en funcionamiento y diseño debido a los requerimientos del cliente, haciendo que el tiempo de desarrollo sea largo y complejo.

- Creación de tablas en base de datos de la información de la distribución geográfica.
- Creación de servicios REST en el backend para el consumo de la metadata de distribución geográfica.

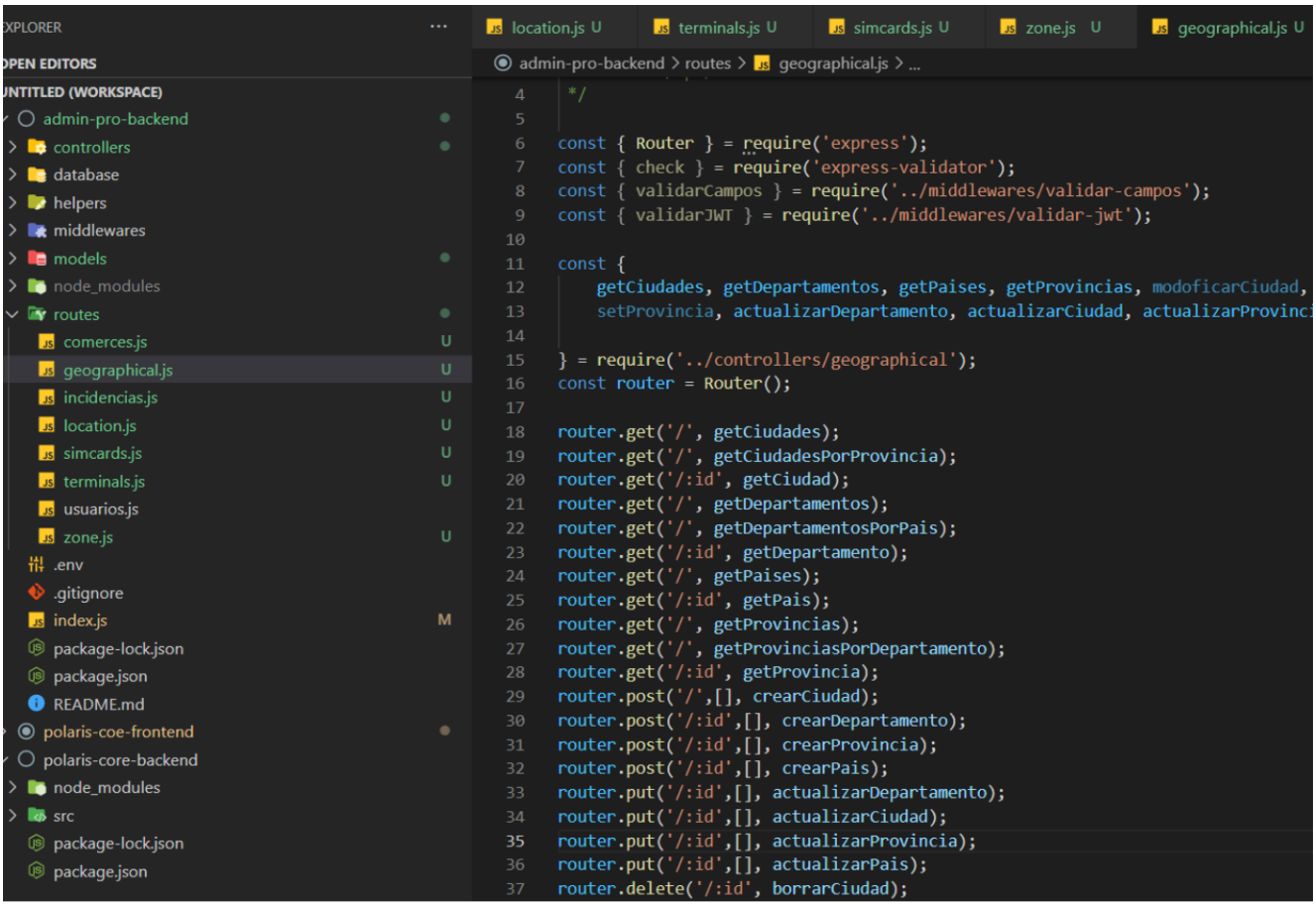

Figura 18. Servicio distribución geográfica, Archivo de rutas donde se manipulan los endpoints para la distribución geográfica de servicios. Fuente: Roberto Guerrero.

- Creación de servicios en angular para consumir el endpoint creado a través de los servicios de distribución geográfica.

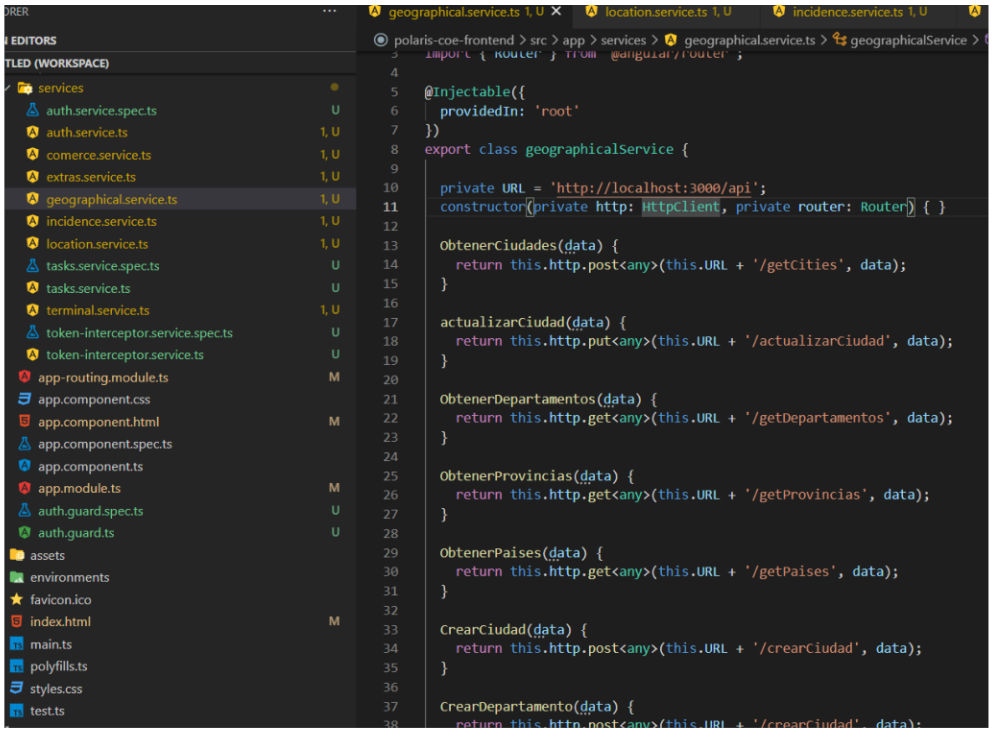

Figura 19. Servicio distribución geográfica, manipulación de endpoints del lado de frontend para la recepción de datos. Fuente: Roberto Guerrero.

- Documentación de los servicios de distribución geográfica para el posterior uso en otras plataformas o aplicaciones móviles.

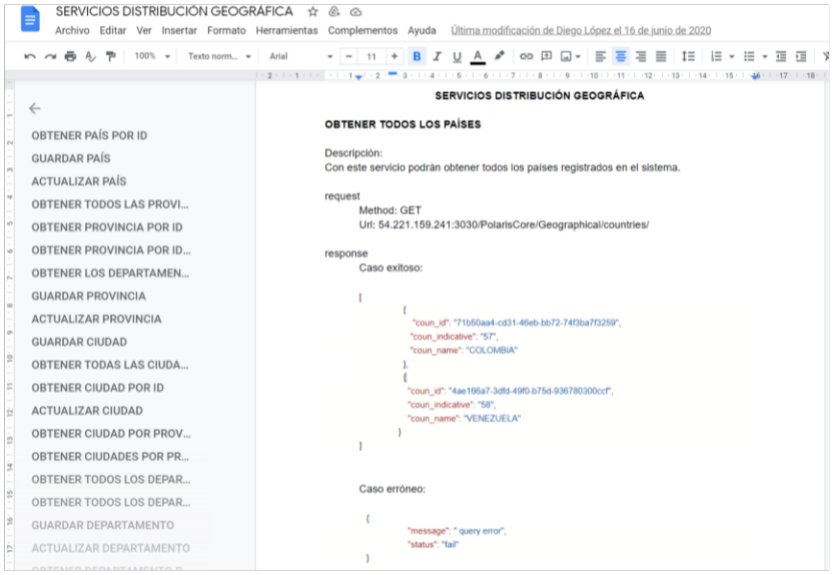

Figura 20. Documento de servicios de distribución geográfica, En este documento se redacta una guía breve de como consumir los endpoints. Fuente: Roberto Guerrero.

- Cambios en los filtros de distribución geográfica de las tablas y formularios del módulo de gestión de incidencias.

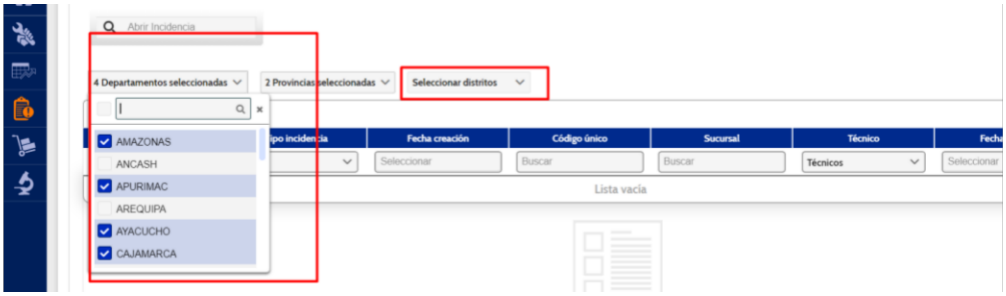

Figura 21. Filtros multiselect para la selección de la distribución geográfica. Fuente: Roberto Guerrero.

- Creación del servicio listar todos los cuadrantes en el backend para el posterior consumo en el frontend de Polaris Core.

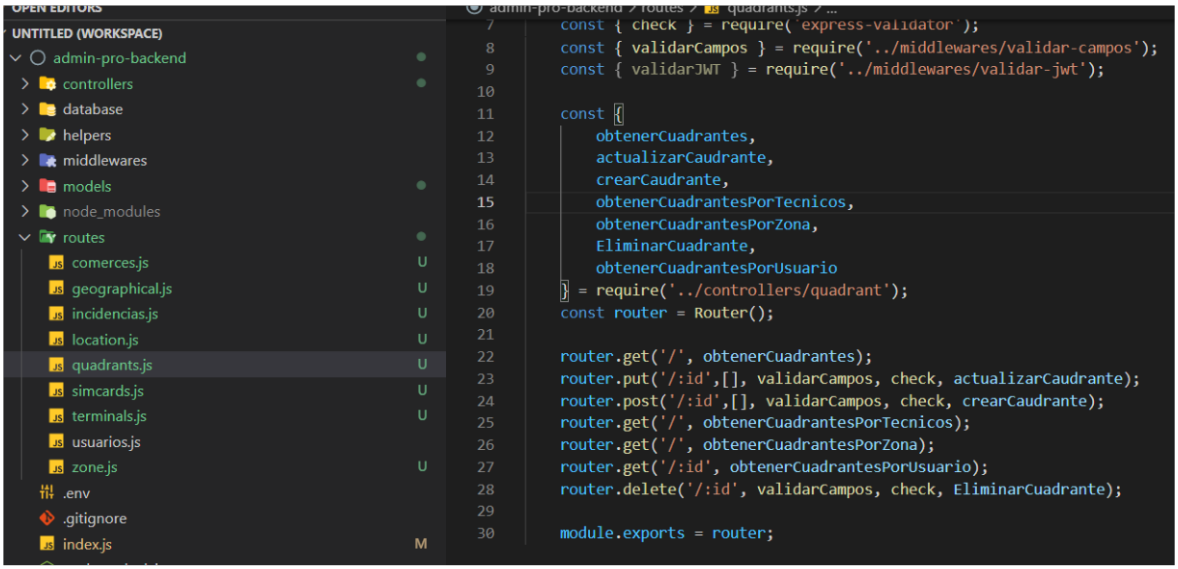

Figura 22. Servicio gestión de cuadrantes, En este archivo se manipulan las rutas definidas como endpoints para gestión de información relacionada con cuadrantes. Fuente: Roberto Guerrero.

- Creación del servicio de recepción para el consumo REST del servicio listar todos los cuadrantes.

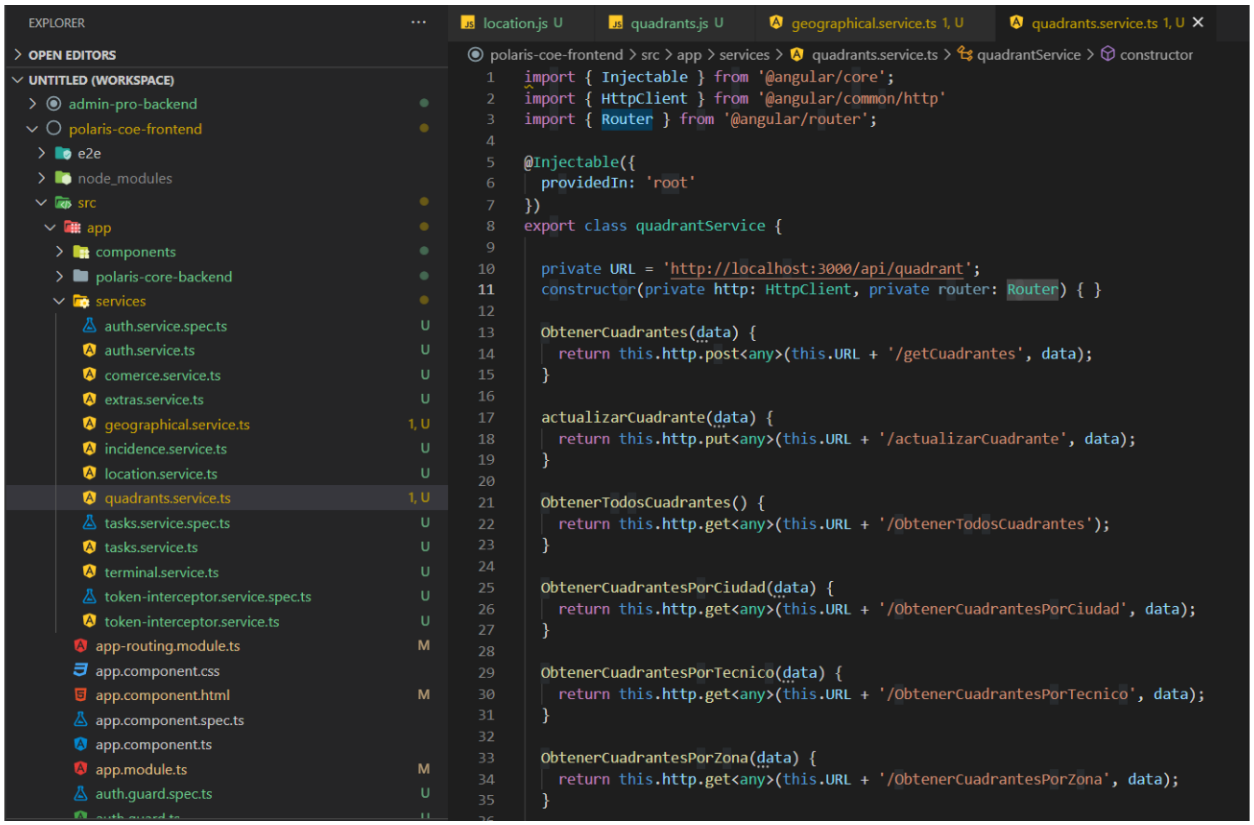

Figura 23. Servicio de recepción de cuadrantes. En este servicio se definen las funciones donde se consumen los endpoints de gestión de cuadrantes, Fuente: Roberto Guerrero.

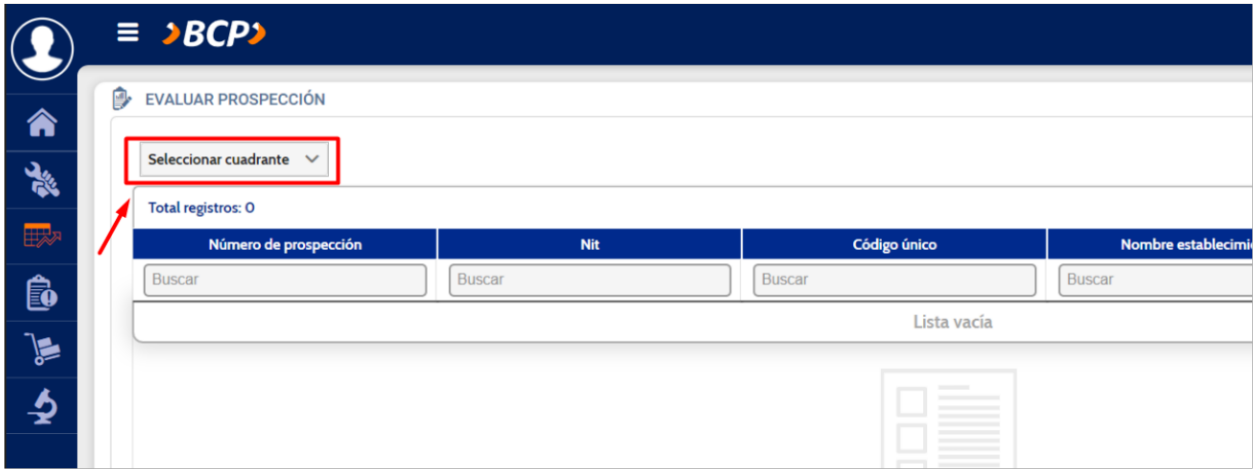

Figura 24, checkbox cuadrantes. En este objeto de listado de opciones se listan los cuadrantes. Fuente: Roberto Guerrero.

- Trabajo en colaboración para el desarrollo del módulo de gestión comercial.

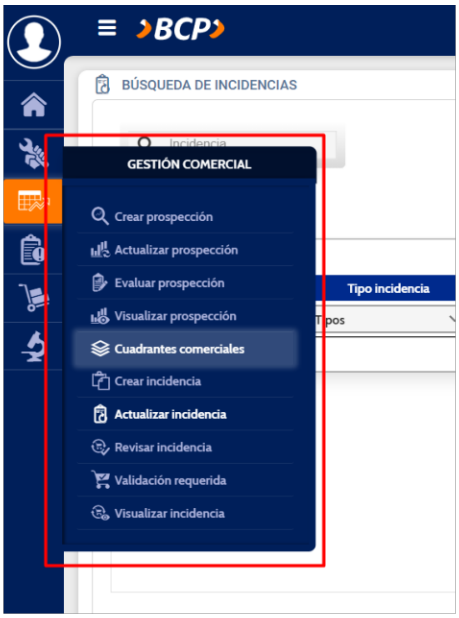

Figura 25. Modulo de gestión comercial. Fuente: Roberto Guerrero.

- Desarrollo de la vista visualizar incidencia, validación requerida, revisar y crear incidencia para el módulo de gestión comercial.

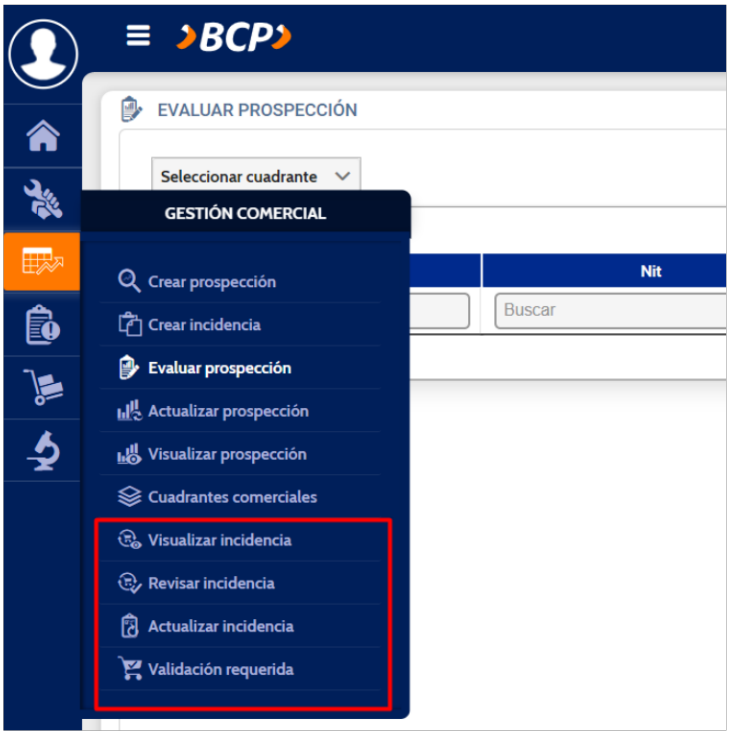

Figura 26. Módulo de gestión comercial. En esta se imagen se ilustra las opciones de menú para las vistas de incidencias. Fuente: Roberto Guerrero.

- Creación de filtros por cuadrantes para las vistas visualizar, revisar incidencia y validación requerida en el módulo de gestión comercial.

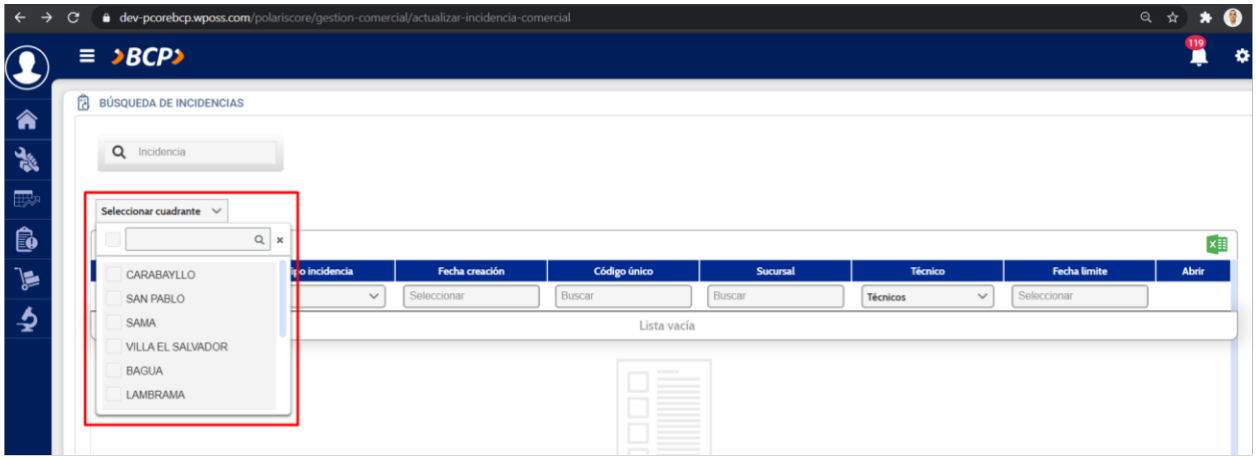

Figura 27. Filtros por cuadrantes. Filtros de registros de incidencias por cuadrantes asignados al usuario. Fuente: Roberto Guerrero.

- Reunión con el equipo de desarrollo para la realización y coordinación de la nueva implementación autenticación requerida por cliente BCP.

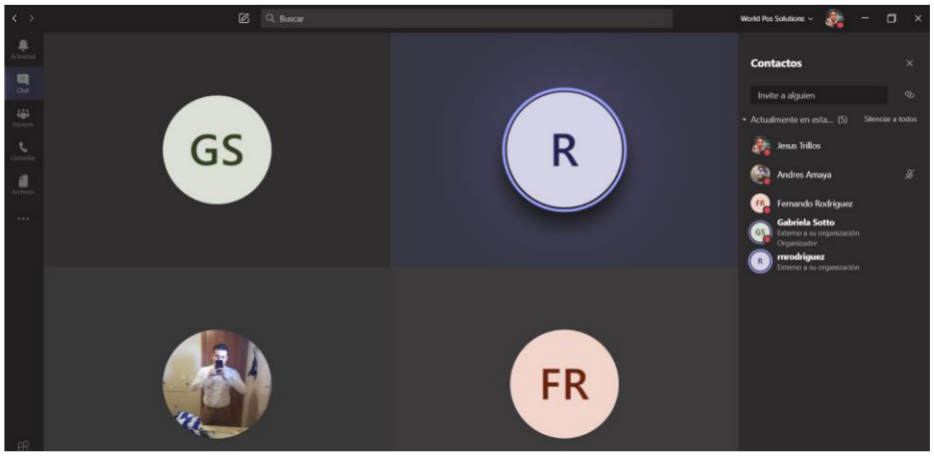

Figura 28. Reunión de capacitación, capture ilustra la reunión entre miembros del equipo de desarrollo. Fuente: Roberto Guerrero.

- Capacitación de los elementos y lógica de autenticación para la integración de Keycloak.
- Conexión con keycloak para la autenticación y validación de usuarios.

- Creación de servicios rest para el consumo de meta data de autenticación.

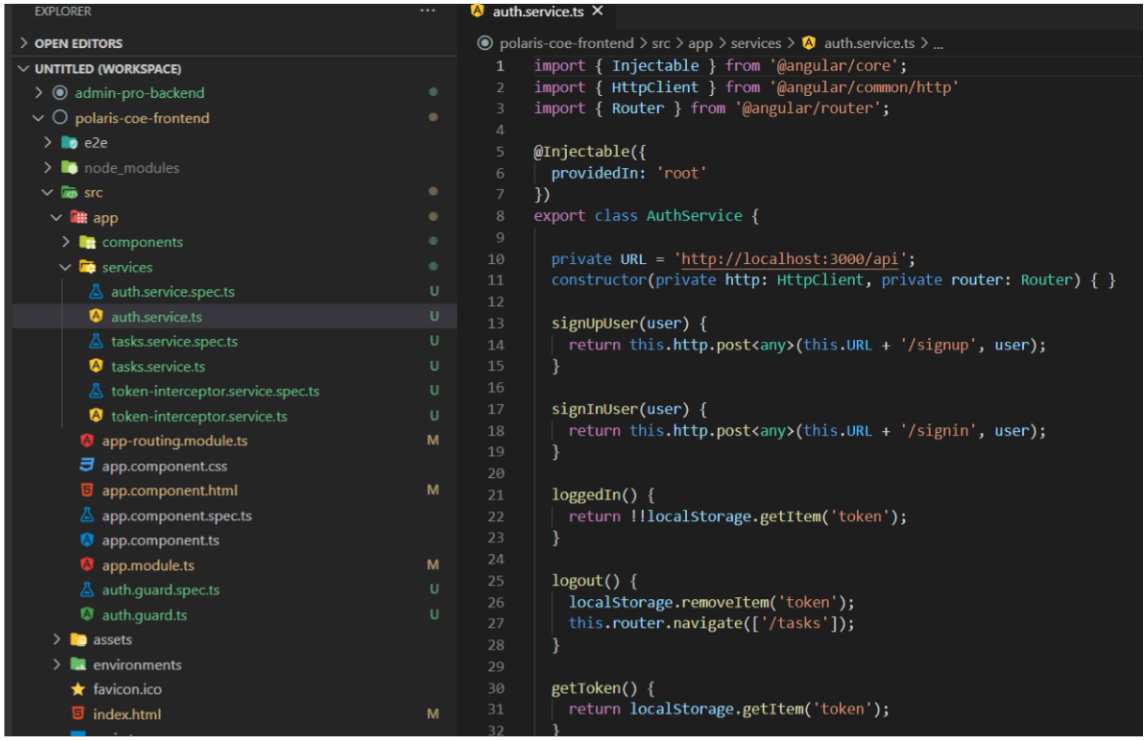

Figura 29. Servicio de autenticación. Endpoints relacionados con la autenticación del lado backend. Fuente: Roberto Guerrero.

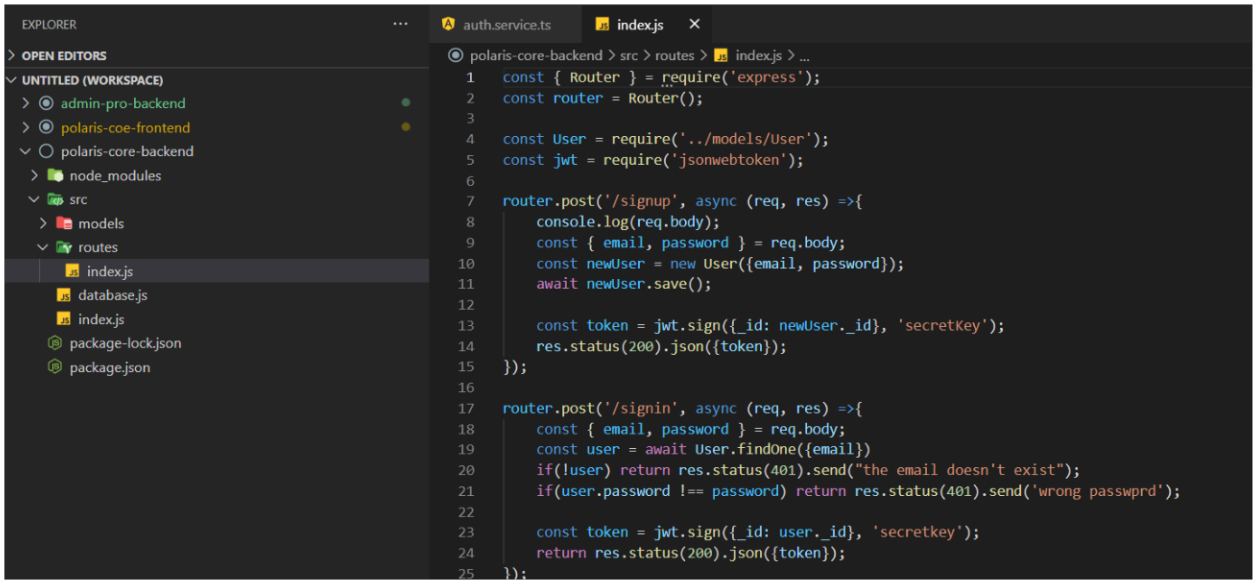

Figura 30. Servicio de autenticación. Endpoints relacionados con autenticación del lado del frontend. Fuente: Roberto Guerrero.

- Creación de funciones para la codificación y decodificación de token.

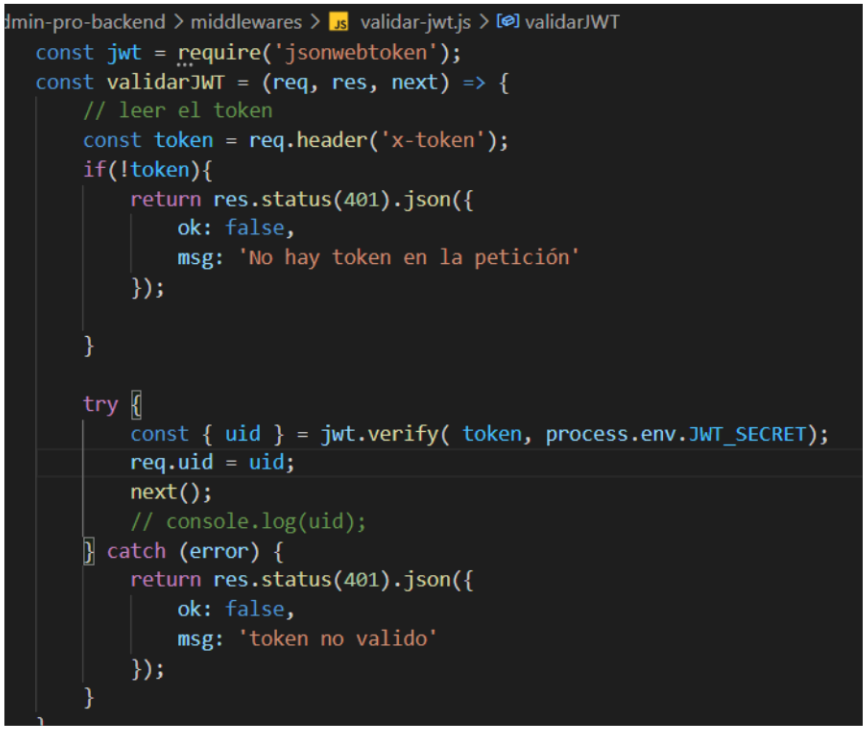

Figura 31. Validación de token. Fuente: Roberto Guerrero.

Figura 32. Generación de token. Código para la generación de JSON web token. Fuente: Roberto Guerrero.

- Modificación de todos microservicios del backend para la implementación de funciones y elementos de la nueva autenticación.
- Modificación de la vista login en el frontend para la nueva implementación.

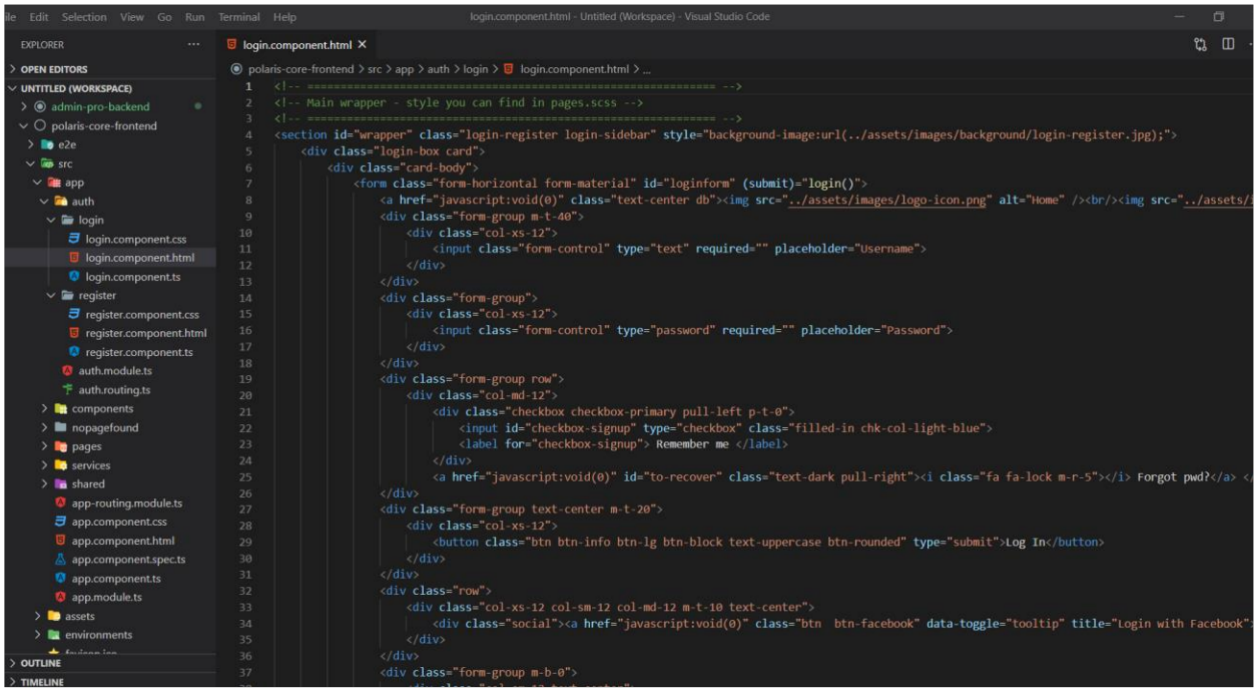

Figura 33. Estructura HTML login. Código HTML que expone la estructura que se apoya bajo la lógica angular. Fuente: Roberto Guerrero.

Figura 34. Formulario login. Fuente: Roberto Guerrero.

Figura 35. Login externo. Interfaz para la autenticación externa mediante Azure. Fuente: Roberto Guerrero.

- Actividades de ofuscación de información requeridas por el cliente BCP.

Figura 36. Tabla listada de usuarios. en esta imagen se visualizan algunos campos de registros enmascarados. Fuente: Roberto Guerrero.

3.10. Realización de pruebas funcionales, no funcionales e integración

Dentro de nuestro equipo de desarrollo se realizan una serie de pruebas de cierta manera naturales que están inmersas dentro del flujo del desarrollo hasta dar con la solución esperada.

- **Pruebas funcionales.**

Estas pruebas se hicieron con el fin de garantizar las características y funcionalidades del software que se comportan según lo esperado sin ningún problema. Valida principalmente toda la aplicación con respecto a las especificaciones y reglas de negocio expuestas por el cliente.

- **Pruebas de integración.**

Estas pruebas analizaban el comportamiento de módulos y submódulos de la aplicación con el fin identificar los errores y problemas relacionados con ellos.

- **Pruebas no funcionales.**

En esta categoría solo se realizaron pruebas de cargue masivo de información para probar el rendimiento del software. En el caso de Polaris Core se cargaba la información por medio de una interfaz y luego ese gran volumen se hacían pruebas de consulta a base de datos.

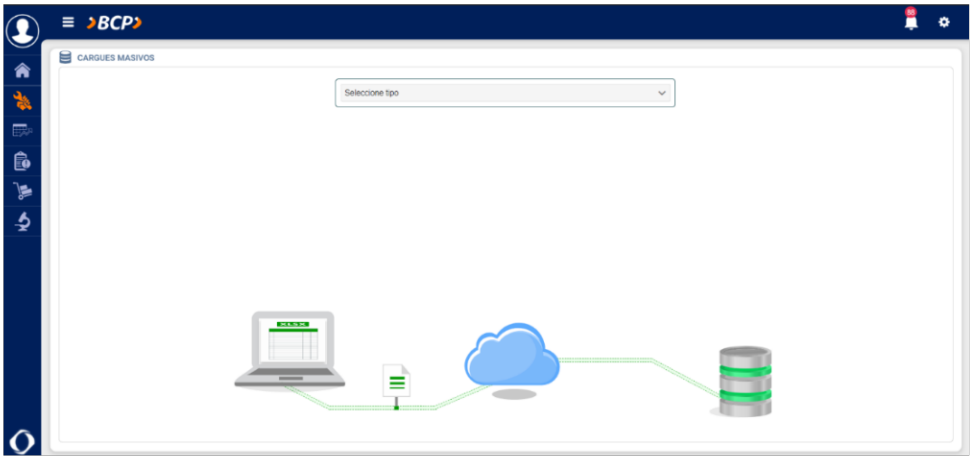

Figura 37. Interfaz de cargue masivo de información. Fuente: Roberto Guerrero.

En las realizaciones de las pruebas se trabajó en coordinación con el equipo de desarrollo y el equipo de QA el cual realizaban pruebas y testeos diarios.

Capítulo 4. Diagnóstico final

Finalizado el proceso de pasantías en la empresa WORLD POS SOLUTIONS S.A.S (WPOSS), y al ser parte del área de IP, se cuenta con una metodología ágil como lo es Scrum que facilita o ayuda la manera en que se afronta el desarrollo de proyectos dentro de la empresa.

Cada proyecto es asignado a un líder desarrollador el cual tiene las capacidades de dirigir y hacerse cargo del proyecto asignado, el líder cuenta con su equipo de trabajo conformado por otros dos desarrolladores que juntos como equipo de trabajo sacan el proyecto adelante. Para el caso particular del proyecto, era el tercero encargado del proyecto, esto quiere decir que hacía parte del respaldo del líder, tanto en funciones como responsabilidades.

Capítulo 5. Conclusiones

En esta pasantía aprendí nuevas experiencias como estudiante, además de conocer como es el entorno de trabajo alrededor del desarrollo de software, siguiendo a cabalidad las peticiones de un cliente, y aportando sugerencias para un equipo de desarrollo como ingenieros.

En la culminación de este proyecto, se logró cumplir con todas las actividades y tareas asignadas durante el periodo establecido por el jefe encargado del área de IP y líder del grupo de desarrollo.

El proyecto propuesto cumplió con los requerimientos que el cliente dispuso, mejorando así la agilidad en los procesos en una red transaccional y el manejo estadístico.

Por último, la realización de pruebas garantizo el correcto funcionamiento y despliegue a producción de la aplicación debido a la rigurosidad del plan pruebas realizados por el área de QA y el equipo de desarrollo.

Capítulo 6. Recomendaciones

los proyectos desarrollados por los diferentes equipos encabezados por el líder deben documentar detalladamente su programación o designen un área encargada de esta documentación, esto con el fin de que los nuevos miembros del equipo o del proyecto puedan adaptarse rápidamente.

Realizar respaldos de información periódicamente de manera que en caso de presentarse incoherencia en los datos estos pueden ser recuperados, garantizando de esta manera que los mismos estén siempre disponibles.

Por otro lado, realizar una constante capacitación en temas claves tantos para miembros nuevos como antiguos, esta capacitación constante hará más fuerte la empresa para afrontar proyectos y clientes potenciales.

7. Referencias

Amazon aws. (2020, 12 22). *aws.amazon.com*. From https://aws.amazon.com/es/nosql/

Android Developers. (2020, 05 01). *Introducción a Android Studio*. From https://developer.android.com/studio/intro

Angular.io. (2020, 12 21). *Angular.io*. From https://angular.io/docs

Apache Software, F. (2020, 12 22). *Apache Software Foundation*. From https://cassandra.apache.org/doc/latest/

Bermúdez, J. D. (2008). *Un Sistema de Escritura de Traductores de Escritura Vía Web.* From Un Sistema de Escritura de Traductores de Escritura Vía Web: http://www.saber.ula.ve/bitstream/handle/123456789/33198/tesis_texier.pdf?sequence=1&isAllowed=y

Cassandra. (2021, 04 7). *cassandra.apache.* From https://cassandra.apache.org/doc/latest/

Estruga. (2013). *Las cinco etapas de la ingeniería de software*. From Las cinco etapas de la ingeniería de software: http://proyectosguerrilla.com/blog/2013/02/las-cinco-etapas-en-la-ingenieria-delsoftware/

Express.js. (2020, 12 22). *expressjs.com*. From https://expressjs.com/es/

Galeano, J. (2014). *Seguridad en transacciones con tarjetas EMV.*

Gil, L. (2014). *Marco de Desarrollo Estándar Basado en el Protocolo ISO-8583 para Terminales de Venta.*

Git. (2020, 12 22). *git-scm.com*. From https://git-scm.com/

Marini, E. (2012). *El modelo Cliente - Servidor.* From https://www.linuxito.com/docs/el-modelo-cliente-servidor.pdf

Mozilla Developer. (2020, 12 22). *developer.mozilla.org*. From https://developer.mozilla.org/es/docs/Learn/JavaScript/First_steps/Qu%C3%A9_es_JavaScript

Node.js. (2020, 12 2020). *nodejs.org*. From https://nodejs.org/es/about/

redhat. (2020, 12 22). *redhat*. From https://www.redhat.com/es/topics/api/what-is-a-rest-api

Schwaber, K., & Jeff, S. (2013). *La Guía Definitiva de Scrum: Las reglas del juego.*

typescriptlang. (2020, 12 22). *typescriptlang*. From https://www.typescriptlang.org/

Udemy. (2021, 04 7). *Udemy.* From https://www.udemy.com/

Vázquez, Á., Gómez, J., & Serrano, R. (2019). *Android: del diseño de la arquitectura al despliegue profesional.* México: Alfaomega Grupo Editor, S.A. de C.V.

Visual Studio code, M. (2020, 12 21). *Visual Studio Code*. From https://code.visualstudio.com/docs

WPOSS. (2018, 02 18). BUENAS PRACTICAS EN EL USO DE REPOSITORIOS GIT WPOSS. Ocaña.