

Instituto Tecnológico de Costa Rica

Carrera de Ingeniería en Computación,
Campus Tecnológico Local San Carlos

“Implementación de módulo Expense Reports
en aplicación People App de GAP”

Práctica Profesional para optar por el título de
Ingeniero en Computación con el grado
académico de Bachiller Universitario

Mario Alberto Jiménez Barrantes

Costa Rica, 2022

1. Resumen ejecutivo

Con este proyecto se busca aplicar los conocimientos y buenas prácticas del desarrollo de software para ofrecer una solución de software de alta calidad a un problema real. La empresa Growth Acceleration Partners (GAP) ha utilizado en el pasado soluciones de software de terceros para la gestión de viáticos a lo interno, pero dado el crecimiento que ha tenido la empresa en los últimos años, dicha solución para la gestión de viáticos se ha vuelto económicamente insostenible. Por ello, la empresa busca crear una solución de software propia que apoye, agilice y facilite la gestión de viáticos a lo interno de la empresa. Dicha solución se presenta como una aplicación web de una única página, con el uso de la herramienta ReactJS para el desarrollo de frontend, y un backend con una arquitectura orientada a microservicios con la utilización de AWS Lambda. Al concluir el proyecto se tiene una solución de software que le permite a la empresa agilizar el proceso de pago de viáticos, se adapta a las necesidades específicas de la misma, que cuenta con los beneficios de una arquitectura de microservicios y que representa una modernización en los sistemas actuales de la compañía, siendo esta una solución más estable, eficiente, escalable y más fácil de mantener.

Palabras clave: Software, Desarrollo de Software, ReactJS, AWS, AWS Lambda, Microservicios.

2. Executive summary

The goal of this project is to put in practice the knowledge and good practices of software development to offer a high quality software solution to a real problem. In the past, the company Growth Acceleration Partners (GAP) has used third-party software solutions to manage travel expenses internally, but given the growth that the company has seen in recent years, said solution for managing travel expenses has become economically unsustainable. For this reason, the company seeks to create its own software solution that supports, speeds up and facilitates the management of travel expenses within the company. This solution is presented as a single page web application, with the use of the ReactJS tool for frontend development, and a backend with a microservices architecture using AWS Lambda. At the end of this project, there is a software application that allows the company to speed up the travel payment process, it is adapted to the specific needs of the company, has the benefits of a microservices architecture and represents a modernization in the company's current systems, with a more stable, efficient, scalable and easier to maintain solution.

Keywords: : Software, Software Development, ReactJS, AWS, AWS Lambda, Microservices.

3. Tabla de contenidos

Resumen ejecutivo	1
Executive summary	2
Tabla de contenidos	3
Índice de figuras	6
Índice de tablas	7
Lista de abreviaturas	8
Lista de acrónimos	9
Capítulo I	10
Introducción	10
Descripción de empresa	10
Contexto del proyecto	11
Problema	12
Objetivos	12
Objetivo general	12
Objetivos específicos	12
Justificación	13
Capítulo II	14
Revisión de literatura	14
Marco teórico	14
Trabajos relacionados	16
Capítulo III	19
Solución planteada	19
Propuesta	19
Involucrados	19
Procedimiento metodológico	20
Análisis de los riesgos	23
Cronograma de trabajo	27
Capítulo IV	28
Requerimientos y Diseño	28
Arquitectura de la aplicación	28
Casos de Uso	29
Historias de Usuario	29
Diagrama de Base de Datos	50
Capítulo V	51
Plataforma de software	51
Aplicación	51
Evaluación	53
Capítulo VI	57
Conclusiones	57

Recomendaciones	58
Bibliografía	59

4. Índice de figuras

Figura 1: Cronograma de Gantt	28
Figura 2: Arquitectura de la Aplicación	29
Figura 3: Casos de Uso	30
Figura 4: User Story 1	31
Figura 5: User Story 2	32
Figura 6: User Story 2	33
Figura 7: User Story 3 First Step	35
Figura 8: User Story 3 Second Step	35
Figura 9: User Story 3 Third Step	35
Figura 10: User Story 3 Fourth Step	36
Figura 11: User Story 3 Fifth Step	36
Figura 12: User Story 3 Sixth Step	37
Figura 13: User Story 4 First Step	39
Figura 14: User Story 4 Second Step	39
Figura 15: User Story 4 Third Step	39
Figura 16: User Story 4 Fourth Step	40
Figura 17: User Story 4 Fifth Step	40
Figura 18: User Story 4 Sixth Step	41
Figura 19: User Story 5	42
Figura 20: User Story 6 First Step	43
Figura 21: User Story 6 Second Step	43
Figura 22: User Story 6 Third Step	44
Figura 23: User Story 6 Fourth Step	44
Figura 24: User Story 6 Fifth Step	44
Figura 25: User Story 6 Sixth Step	45
Figura 26: User Story 7 Approval Options	46
Figura 27: User Story 7 Approval Form	47
Figura 28: User Story 7 Rejection Form	47
Figura 29: User Story 7 Email Notification	47
Figura 30: User Story 8	48
Figura 31: User Story 9	49
Figura 32: User Story 10	50
Figura 33: Diagrama de Base de Datos	51

5. Índice de tablas

Tabla 1: Trabajo Relacionado 1	17
Tabla 2: Trabajo Relacionado 2	18
Tabla 3: Trabajo Relacionado 3	19
Tabla 4: Personal Involucrado	20
Tabla 5: Matriz para el Procedimiento Metodológico	21
Tabla 6: Análisis de Riesgo 1	24
Tabla 7: Análisis de Riesgo 2	25
Tabla 8: Análisis de Riesgo 3	25
Tabla 9: Análisis de Riesgo 4	26
Tabla 10: Análisis de Riesgo 5	27
Tabla 11: User Story 1	31
Tabla 12: User Story 2	32
Tabla 13: User Story 3	33
Tabla 14: User Story 4	37
Tabla 15: User Story 5	41
Tabla 16: User Story 6	42
Tabla 17: User Story 7	45
Tabla 18: User Story 8	47
Tabla 19: User Story 9	48
Tabla 20: User Story 10	49
Tabla 21: Historias de Usuario - Funciones Lambda	52
Tabla 22: Resultados del Desarrollo	54
Tabla 23: Resultados de la Evaluación	56

6. Lista de abreviaturas

Desc: Descendente.

7. Lista de acrónimos

AWS: Amazon Web Services.

CO: Colombia.

COP: Peso colombiano.

CR: Costa Rica.

CRC: Colón costarricense.

CTA: Llamado a la acción (Call to Action).

DevOps: Desarrollo y operaciones (Development and Operations).

DM: Gerente de entregas (Delivery Manager).

DOM: Modelo de objeto de documento (Document Object Model).

FaaS: Función como servicio (Function as a Service).

GAP: Growth Acceleration Partners.

GAPster: Colaborador de GAP.

ITCR: Instituto Tecnológico de Costa Rica.

JSX: JavaScript XML.

MVC: Modelo Vista Controlador (Model-View-Controller).

PDF: Formato de documento portátil (Portable Document Format).

PET: People Tool.

PM: Director de proyecto (Project Manager).

PO: Propietario del producto (Product Owner).

QA: Aseguramiento de la calidad (Quality Assurance).

S3: Servicio de almacenamiento de objetos (Simple Storage Service).

USD: Dólar estadounidense.

VM: Máquina virtual (Virtual Machine).

Capítulo I

1. Introducción

En este documento se detalla el proyecto de práctica profesional, realizado como requisito para obtener el título de Ingeniero en Computación. Se describe el problema existente, los objetivos y el alcance del proyecto, la solución planteada, las tareas y el procedimiento a seguir para completar el proyecto de manera exitosa.

El desarrollo de este proyecto se llevará a cabo en la empresa Growth Acceleration Partners ubicada en Costa Rica, la cual también posee sedes en Estados Unidos y Colombia.

El proyecto a desarrollar es un módulo nuevo, Expense Reports, para la gestión de viáticos en la empresa, que se integrará a la aplicación People App. People App es una aplicación web para uso interno, que ayuda en la centralización y manejo de la información general de la compañía, de los proyectos en los que se trabaja y de los colaboradores que forman parte de Growth Acceleration Partners.

Además, el desarrollo de la aplicación People App y sus distintos módulos trae consigo beneficios al desarrollo tecnológico de la compañía, ya que esta aplicación representa una modernización en los sistemas actuales de la compañía con la aplicación de los estándares de desarrollo de la empresa, lo que implica un sistema más estable, eficiente, escalable y con mayor facilidad de mantenimiento, y que fomenta el uso de tecnologías modernas como ReactJS y Cloud.

1.1. Descripción de empresa

Growth Acceleration Partners (GAP) es una empresa que nació en Austin, Texas de los Estados Unidos de América y tiene sedes en Costa Rica, Colombia y Estados Unidos.

GAP es una empresa de desarrollo de software que brinda servicios empresariales en distintas áreas. En la nube, por ejemplo, se efectúa tanto el manejo de los ambientes de producción de los clientes, como el desarrollo de aplicaciones directamente en la nube (cloud development con Lambda o similar). Además de ello se ofrecen servicios de migración de aplicaciones a la nube y la implementación de procesos de automatización de creación de servicios/deployments por medio de herramientas DevOps. También se brindan

servicios de análisis de datos, desarrollo de aplicaciones para dispositivos móviles y automatización de control de calidad.

Cabe destacar que la misión de la empresa se enfoca en crear asociaciones estratégicas con sus clientes para ayudarles a escalar de manera más inteligente y económica, transformando sus objetivos en soluciones de datos y software que impulsan los resultados comerciales.

1.2. Contexto del proyecto

El sistema People App es una aplicación web propia de GAP creada desde cero, que busca reemplazar una serie de productos preexistentes en la compañía con la idea de modernizarlos y crear un sistema más estable, eficiente, escalable y más fácil de mantener, desarrollado con tecnologías modernas orientadas a la nube y una arquitectura orientada a microservicios.

People App está compuesta por varios módulos que incluyen un sistema de control del estado de los proyectos en los que trabaja la empresa (DM Dashboard), un sistema de registro de empleados y la asignación de los líderes de los mismos (PM Dashboard), un sistema de seguimiento de la felicidad de los colaboradores en GAP (Happiness Dashboard), un sistema para el seguimiento de la carrera profesional de cada uno de los GAPsters (Portfolio & Career Development), uno para el análisis de las tecnologías que más se utilizan en la organización según proyecto y empleado (Technology Dashboard), un sistema que administra la asignación de los recursos a los diferentes proyectos (Resources Allocation), un módulo de gestión de pago de viáticos (Expense Reports) y la visualización de los datos personales y profesionales de cada uno de los GAPsters en la compañía (My Dashboard).

Los módulos que ya se encuentran completados son My Dashboard, PM Dashboard, Happiness Dashboard y Technology Dashboard. El sistema People App aún se encuentra en desarrollo y no está en operación.

Como proyecto de Práctica Profesional, se va a desarrollar un nuevo módulo, Expense Reports, para la gestión de viáticos de la empresa, que se integrará a la aplicación People App.

En el pasado para la gestión de viáticos a lo interno de GAP, se hizo uso de una aplicación externa de pago, Travel Bank, pero dado el crecimiento que ha tenido la empresa, dicha aplicación se ha vuelto económicamente insostenible.

1.3. Problema

En el pasado, se ha intentado la creación de distintos sistemas de software a la medida para apoyar la gestión de distintas áreas de la compañía, como gestión de proyectos, gestión de personal, gestión de recursos, entre otros. Sin embargo, estos sistemas no han podido ser culminados satisfactoriamente, su mantenimiento ha generado problemas dada la arquitectura de software existente y no hay un diseño visual consistente entre los distintos sistemas. Además, estos proyectos internos son desarrollados por empleados que no tienen asignación de proyectos con clientes, y al utilizar sistemas y tecnologías no tan modernas han dejado de ser un reto interesante para los GAPsters, por lo que los recursos disponibles para continuar con las mejoras necesarias a los sistemas de software son muy limitados.

Para la gestión de viáticos GAP ha hecho uso de Travel Bank, una aplicación externa de pago, y que con una cantidad reducida de personal la empresa podía costear, pero dado el crecimiento que ha tenido GAP esta solución se ha vuelto económicamente insostenible.

1.4. Objetivos

1.4.1. Objetivo general

Desarrollar el módulo Expense Reports para la plataforma web People App, para la agilización y disminución de costos en la gestión de viáticos a lo interno de la empresa, usando una arquitectura orientada a servicios en la nube a través de Amazon Web Services.

1.4.2. Objetivos específicos

1. Definir los requerimientos e historias de usuario del módulo Expense Reports de la aplicación People App de la empresa GAP.
2. Diseñar diagrama de arquitectura del módulo Expense Reports de la aplicación People App de la empresa GAP.
3. Implementar el módulo Expense Reports de la aplicación People App de la empresa GAP.

4. Evaluar el funcionamiento del módulo Expense Reports de la aplicación People App de la empresa GAP.

1.5. Justificación

El desarrollo de la plataforma People App ayuda en la centralización y manejo de la información general de la compañía, la información de los proyectos en los que se trabaja y de los colaboradores. La aplicación reemplaza una serie de productos preexistentes en la compañía y los unifica en un solo lugar, en una sola herramienta a nivel visual que permite llevar un registro más automatizado y organizado de las diferentes secciones de la compañía, y con lo que se logra estandarizar el uso de una única aplicación para gestionar las distintas necesidades específicas de la empresa.

Esta plataforma implica una modernización en los sistemas actuales de la compañía con la aplicación de los estándares de desarrollo de la empresa, lo que implica un sistema más estable, eficiente, escalable y con mayor facilidad de mantenimiento. Además, el proyecto está orientado al desarrollo en la nube y servicios serverless, lo que representa el uso de tecnologías modernas en los sistemas internos de la compañía.

El módulo Expense Reports, único módulo que se desarrollará para este proyecto, es una solución hecha a la medida para las necesidades específicas de GAP y que ayuda en la gestión de viáticos en la empresa. Dado que reemplaza a Travel Bank, una aplicación externa de pago, representa una reducción de gastos considerable para la empresa. Este módulo permite a los colaboradores de la empresa ver y reportar sus gastos de viáticos, y a los administradores de finanzas procesar las solicitudes presentadas por los colaboradores, todo dentro de la misma plataforma People App, lo que facilita y agiliza la gestión de viáticos.

1. Revisión de literatura

1.1. Marco teórico

AWS Lambda

Una de las corrientes modernas del desarrollo de software es la de separar las aplicaciones monolíticas, aquellas en las que todo se concentra en una sola base de código, en microservicios, en donde cada componente de la aplicación tiene una base de código por aparte. Esto permite que cada funcionalidad de la aplicación tenga un ciclo de vida separado de los demás, permite trabajar con lenguajes de programación y tecnologías distintas en cada componente buscando lo que mejor funcione para cada caso, cada uno puede ser mantenido por un equipo de trabajo distinto, cada microservicio se puede escalar por separado dependiendo de la demanda específica generada para cada uno de estos, entre otras ventajas, que permiten agilizar el desarrollo y la puesta en producción de nuevas funcionalidades o mejoras a la aplicación, a tal grado de que esto podría suceder varias veces al día.

... services like AWS Lambda allow implementing microservice architectures without the need of managing servers. Thus, it facilitates the creation of functions (i.e. microservices) that can be easily deployed and automatically scaled, and it also helps reduce infrastructure and operation costs. (Villamizar et al., 2016)

AWS Lambda es un servicio serverless (sin servidor) en la nube tipo FaaS (Function as a Service) de la empresa Amazon, que permite el desarrollo de aplicaciones con una arquitectura de microservicios, en un grado tan alto de separación de componentes que cada función dentro de la aplicación es un microservicio. Para aclarar el significado de esto, podemos pensar en un microservicio que se encargue del manejo de información de autos disponibles en una empresa de alquiler de autos, este microservicio tendría las funciones de leer, crear, editar y eliminar la información de los autos, el microservicio cumple con todas estas funcionalidades, pero en el caso del servicio AWS Lambda, cada una de estas funciones, leer, crear, editar y eliminar, se convierten en un microservicio en sí mismas, no sería un solo microservicio que administre todas estas funciones, sino que tendríamos 4 microservicios, uno para cada función.

... The cloud provider will be scaling microservices automatically as needed. The developers should not worry about servers and their configuration. System owner should not buy or provision any hardware. Users save money because they are paying only for compute time used. They do not pay for idle server time saving up to ninety percent over a cloud VM. (Kaplunovich, 2019)

Una ventaja importante del servicio AWS Lambda, además de las ya mencionadas anteriormente, es la reducción de costos. Mientras que en una aplicación monolítica o en una aplicación con arquitectura de microservicios se necesita tener los servidores activos en todo momento y eso implica un costo, con AWS Lambda el costo se calcula por cada llamada a una función y el tiempo que ésta tarde en ejecución, y en caso de que la función no sea solicitada por un tiempo prolongado no se genera ningún costo.

ReactJS

ReactJS es una biblioteca desarrollada y respaldada por Facebook para el lenguaje de programación JavaScript. Con ReactJS se crean aplicaciones de una única página, que da como resultado una experiencia más rápida, más ágil y más fluida para el usuario, en contraposición a aplicaciones web tradicionales en las que es necesario abrir y cargar nuevas páginas para acceder a las distintas funcionalidades de la aplicación. ReactJS se usa en la implementación de las vistas o interfaces de usuarios de aplicaciones web, lo que se conoce como *frontend*, se podría decir que se utiliza como el componente View en el patrón de diseño Model-View-Controller (MVC), y el *backend* de la aplicación, lo que sería la lógica de negocio, se trabaja por aparte.

React basically enables development of large and complex web based applications which can change its data without subsequent page refreshes. It is used as the View (V) in the Model-View-Controller(MVC). React abstracts the Document Object Model (DOM), thus offering a simple, performing and robust application development experience. (Aggarwal, 2018)

ReactJS trabaja con un DOM virtual y no interactúa directamente con el DOM generado por el navegador. Este DOM virtual lo maneja en memoria y solo refleja los cambios en el navegador de aquellos elementos en los que encuentra diferencias, lo que da como resultado un alto rendimiento.

En una aplicación de ReactJS todo se divide en pequeños componentes que pueden ser reutilizados, se pueden crear botones, campos de texto, tablas, etc., todos son pequeños componentes personalizados y parametrizados, y es la unión de estos componentes lo que genera las vistas, esto permite un desarrollo más ágil y rápido.

ReactJS y su lenguaje JSX es uno de los factores diferenciadores de esta biblioteca. JSX permite que la interacción con el DOM se haga de manera muy sencilla, con este se vuelve muy fácil manipular y actualizar los componentes que forman la vista, a diferencia de trabajar con Javascript puro en donde la interacción con el DOM es un proceso más engorroso.

ReactJS es una de las herramientas más populares para el desarrollo de aplicaciones de una única página y ofrece un gran rendimiento. Además tiene el respaldo de una empresa tan grande como Facebook y una gran comunidad, lo que lo vuelve una opción muy atractiva para el desarrollo de aplicaciones web.

1.2. Trabajos relacionados

Tabla 1: Trabajo Relacionado 1

Autor (es)	Villamizar, M., Garcés, O., Ochoa, L., Castro, H., Salamanca, L., Verano, M., Casallas, R., Gil, S., Valencia, R., Zambrano, A., Lang, M.
Año	2016.
Tipo de investigación	Investigación aplicada tecnológica.
¿Qué se hizo? Objetivos	Medir el costo económico de ejecutar una aplicación software en la nube bajo una arquitectura monolítica, una de microservicios, y otra utilizando AWS Lambda.
¿Cómo se hizo? Metodología	Se ejecutaron las 3 aplicaciones, una monolítica, una de microservicios y una que trabaja con AWS Lambda, utilizando los servicios que ofrece AWS, y se realizó el cálculo de los costos económicos para cada escenario.
¿Qué se obtuvo?	La aplicación que trabaja con AWS Lambda presenta la mayor reducción en los costos.

Conclusiones	
¿Qué aporta esta investigación a mi trabajo? Valoración crítica	Es importante conocer y adaptar el desarrollo de software a los avances que se han dado en el campo. La arquitectura de microservicios y el conjunto de tecnologías que la apoyan va impulsada por equipos más potentes, el poder de la nube que nos da acceso a una gran cantidad de recursos de manera instantánea y la demanda del mercado a cambios rápidos, algo que facilita el trabajo con microservicios. Un trabajo como este demuestra que, aunque la arquitectura de microservicios presenta nuevos y grandes retos, el factor económico no es un impedimento para adoptarla y sacar provecho de todos sus beneficios.

Fuente: elaboración propia

Tabla 2: Trabajo Relacionado 2

Autor (es)	Aggarwal, S.
Año	2018.
Tipo de investigación	Investigación descriptiva.
¿Qué se hizo? Objetivos	Se estudian las características y el funcionamiento de ReactJS.
¿Cómo se hizo? Metodología	No aplica.
¿Qué se obtuvo? Conclusiones	ReactJS tiene la potencia y las características requeridas para desarrollar aplicaciones web modernas.
¿Qué aporta esta investigación a mi trabajo? Valoración	Este trabajo permite comprender las posibilidades que ofrece ReactJS para el desarrollo de aplicaciones Web de una única página. ReactJS es una potente herramienta para el desarrollo de páginas web, cuenta con gran popularidad y una gran comunidad que la respalda, lo que la hace una buena elección.

crítica	
---------	--

Fuente: elaboración propia

Tabla 3: Trabajo Relacionado 3

Autor (es)	Novac, C. M., Novac, O. C., Sferle, R. M., Gordan, M. I., BUJDOSÓ, G., & Dindelegan, C. M.
Año	2021.
Tipo de investigación	Investigación aplicada tecnológica.
¿Qué se hizo? Objetivos	Se realiza un estudio comparativo entre dos herramientas muy populares para el desarrollo de páginas web en el lenguaje de programación Javascript: ReactJS y Vue.
¿Cómo se hizo? Metodología	Se desarrolló una aplicación sencilla tanto en ReactJS como en Vue, una calculadora para cada caso, y se midieron los tiempos de carga utilizando la herramienta de medición de rendimiento de Google Chrome.
¿Qué se obtuvo? Conclusiones	<ul style="list-style-type: none"> - Vue es más fácil de usar y aprender para principiantes, mientras que ReactJS requiere tener mayor conocimiento de la herramienta. - ReactJS presenta un mayor rendimiento en comparación a Vue.
¿Qué aporta esta investigación a mi trabajo? Valoración crítica	Dada la simplicidad de las aplicaciones desarrolladas es difícil comparar los resultados con una aplicación real y de mayor complejidad. Se desconoce si para una aplicación grande, que haga uso de más funcionalidades ofrecidas por cada herramienta, la diferencia de rendimiento será más corta o mayor, o si ReactJS mantendrá la ventaja sobre Vue.

Fuente: elaboración propia

1. Solución planteada

1.1. Propuesta

La solución propuesta para el proyecto se plantea como una aplicación web de una única página basada en una arquitectura orientada a microservicios con la utilización de Lambdas en AWS desarrollado con Python con un enfoque de servidor backend, una base de datos relacional con el motor de base de datos PostgreSQL, distintas tecnologías de AWS para el manejo y control de infraestructura y almacenamiento, y un frontend desarrollado en ReactJS que permite la visualización de la información y está basado en el lenguaje de programación Javascript.

1.2. Involucrados

El personal involucrado en este proyecto se detalla en la siguiente tabla con su nombre, el rol o responsabilidad que conlleva en el desarrollo del proyecto y una pequeña descripción de este. Además, cabe destacar que, el criterio de éxito de todos los involucrados, es obtener una funcionalidad adecuada en un 100% del módulo Expense Reports de la aplicación People App.

Tabla 4: Personal Involucrado

Nombre	Rol	Descripción
Andrés Hernández Monge	Cliente Final	Es quien solicita el proyecto.
David Ramírez Pérez	Scrum Master	Es quien lidera el proyecto a través de todas sus etapas desde una perspectiva de distribución de trabajo en el proceso de la metodología Scrum.
Rodrigo Rodríguez Ramírez	Líder Técnico	Es quien lidera el proyecto a través de todas sus etapas

		desde una perspectiva técnica.
Mario Jiménez Barrantes	Desarrollador	Es quien trabaja en el desarrollo del proyecto durante todas sus etapas.
Juan Esteban Alzate Correa	Ingenieros QA	Son quienes trabajan en el proceso de aseguramiento de la calidad del proyecto durante todas sus etapas.
Gaudy Esquivel Vega	Profesora asesora	Es quien asesora al practicante como estudiante del ITCR y evalúa los informes de la práctica profesional.

Fuente: elaboración propia

1.3. Procedimiento metodológico

Tabla 5: Matriz para el Procedimiento Metodológico

Objetivo Específico	Tarea	Meta	Indicador
Definir los requerimientos e historias de usuario del módulo Expense Reports de la aplicación People App.	<p>A. Entrevistar al cliente para conocer qué espera del proyecto y sus funcionalidades.</p> <p>B. Definir una reunión con el líder técnico para conocer sobre los requerimientos no funcionales que se</p>	<p>A. Diagramar los casos de uso del módulo Expense Reports de la aplicación People App.</p> <p>B. Documentar un listado de historias de usuario.</p> <p>C. Documentar un</p>	<p>A. Diagrama de casos de uso aprobado por el líder técnico del proyecto.</p> <p>B. Documento de historias de usuario aprobado por el líder técnico del proyecto.</p>

	<p>deben de satisfacer.</p> <p>C. Desarrollar los casos de uso, las historias de usuario, y requerimientos no funcionales.</p>	<p>listado de requerimientos no funcionales.</p>	<p>C. Documento de requerimientos no funcionales aprobado por el líder técnico del proyecto.</p>
<p>Definir diagrama de arquitectura y esquema de base de datos del módulo Expense Reports de la aplicación People App.</p>	<p>A. Examinar la arquitectura de la plataforma People App.</p> <p>B. Examinar el prototipo de la plataforma People App.</p> <p>C. Diseñar un diagrama de la arquitectura que conlleva el desarrollo de este módulo.</p> <p>D. Diseñar un diagrama del esquema de la base de datos para este módulo.</p>	<p>A. Documentar diagrama de la arquitectura que conlleva el desarrollo de este módulo.</p> <p>B. Documentar diagrama del esquema de la base de datos para este módulo.</p>	<p>A. Documento con el diagrama de arquitectura aprobado por el líder técnico del proyecto.</p> <p>B. Documento con el diagrama del esquema de la base de datos aprobado por el líder técnico del proyecto.</p>
<p>Implementar el módulo Expense Reports de la aplicación People App.</p>	<p>A. Capacitarse en las tecnologías de ReactJS, Python Lambdas en AWS y PostgreSQL.</p>	<p>A. Completar la implementación del frontend del módulo solicitado.</p>	<p>A. Que el módulo sea 100% funcional dentro de la aplicación People App y que sea</p>

	<p>B. Programar la estructura del frontend que corresponde según el diseño dado y la arquitectura planteada.</p> <p>C. Programar la estructura del backend e integrarlo con el frontend según la arquitectura planteada.</p>	<p>B. Completar la implementación e integración del backend del módulo solicitado.</p> <p>C. Entregar el código fuente de la aplicación.</p>	<p>aprobado por el líder técnico del proyecto.</p>
<p>Evaluar el funcionamiento del módulo Expense Reports de la aplicación People App.</p>	<p>A. Realizar la tarea de Code Review junto con el líder técnico del proyecto por cada tarea completada.</p> <p>B. Hacer un software build del código después de su Code Review correspondiente y subirlo al ambiente de pruebas para que sea revisado por el departamento de QA.</p> <p>C. Reajustar</p>	<p>A. Entregar demos de cada tarea completada y aprobada.</p> <p>B. Comprobar la funcionalidad y el diseño implementado del módulo para que sea colocado en la aplicación People App en producción.</p>	<p>A. Documento con demos y listado de ajustes necesarios para completar la funcionalidad de cada tarea aprobado por el líder técnico del proyecto.</p> <p>B. Que el módulo sea 100% funcional dentro de la aplicación People App en producción y cumplan con el diseño que corresponde y que sea aprobado por el</p>

	<p>cualquier falla reportada por el departamento de QA.</p> <p>D. Hacer un demo al final de cada sprint de la tarea completada y aprobada, además comentar los ajustes que fueron requeridos para obtener la funcionalidad para demostrar el avance al cliente final.</p>		líder técnico del proyecto.
--	--	--	-----------------------------

Fuente: elaboración propia

1.4. Análisis de los riesgos

Tabla 6: Análisis de Riesgo 1

Referencia	Objetivo 3
Código	R-1
Nombre	Falta de conocimiento de los desarrolladores en las tecnologías que usa la aplicación.
Categoría	Personas.
Causa	En el proyecto se trabaja con tecnologías modernas y poco conocidas por los desarrolladores.
Impacto	Se dan atrasos en el desarrollo y no se cumple con el cronograma planificado.
Estrategia de	No existe.

evasión	
Estrategia de mitigación	Se ofrece un período de capacitación a los desarrolladores al inicio del proyecto con cursos en línea y guías del líder técnico.
Estrategia de contingencia	Se trabaja con metodologías ágiles que permiten ver el avance diario de las tareas y se cuenta con el apoyo del líder técnico y de los demás desarrolladores del equipo con más experiencia.

Fuente: elaboración propia

Tabla 7: Análisis de Riesgo 2

Referencia	Objetivo 3 y 4
Código	R-2
Nombre	Atrasos con el cronograma.
Categoría	Personas.
Causa	Planificación errónea.
Impacto	No se cumple con los entregables a tiempo.
Estrategia de evasión	No existe.
Estrategia de mitigación	<ul style="list-style-type: none"> - Aprobación del cronograma por expertos. - Se trabaja con metodologías ágiles que permiten ver el avance diario de las tareas y solicitar ayuda a expertos cuando se encuentran bloqueos.
Estrategia de contingencia	<ul style="list-style-type: none"> - Realizar ajustes en los tiempos y prioridades de las tareas. - Solicitar ayuda a otros miembros del equipo para completar las tareas.

Fuente: elaboración propia

Tabla 8: Análisis de Riesgo 3

Referencia	Objetivo 3
Código	R-3

Nombre	Fallos en la conexión a Internet.
Categoría	Tecnológico.
Causa	El proveedor del servicio de Internet presenta fallas en su infraestructura.
Impacto	Pérdida de comunicación con el equipo y horas de trabajo perdidas.
Estrategia de evasión	No existe.
Estrategia de mitigación	No existe.
Estrategia de contingencia	<ul style="list-style-type: none"> - Se cuenta con todo lo necesario para continuar el desarrollo en ambiente local. - Se tiene al menos otro lugar al que se pueda asistir de manera rápida que cuenta con un proveedor de Internet distinto.

Fuente: elaboración propia

Tabla 9: Análisis de Riesgo 4

Referencia	Objetivo 3
Código	R-4
Nombre	Fallos de conexión con los servicios de AWS con los que trabaja la aplicación.
Categoría	Tecnológico.
Causa	Los servicios de AWS presentan fallas.
Impacto	Horas de trabajo perdidas.
Estrategia de evasión	No existe.
Estrategia de mitigación	No existe.
Estrategia de	Se cuenta con todo lo necesario para continuar trabajando en ambiente

contingencia	local.
--------------	--------

Fuente: elaboración propia

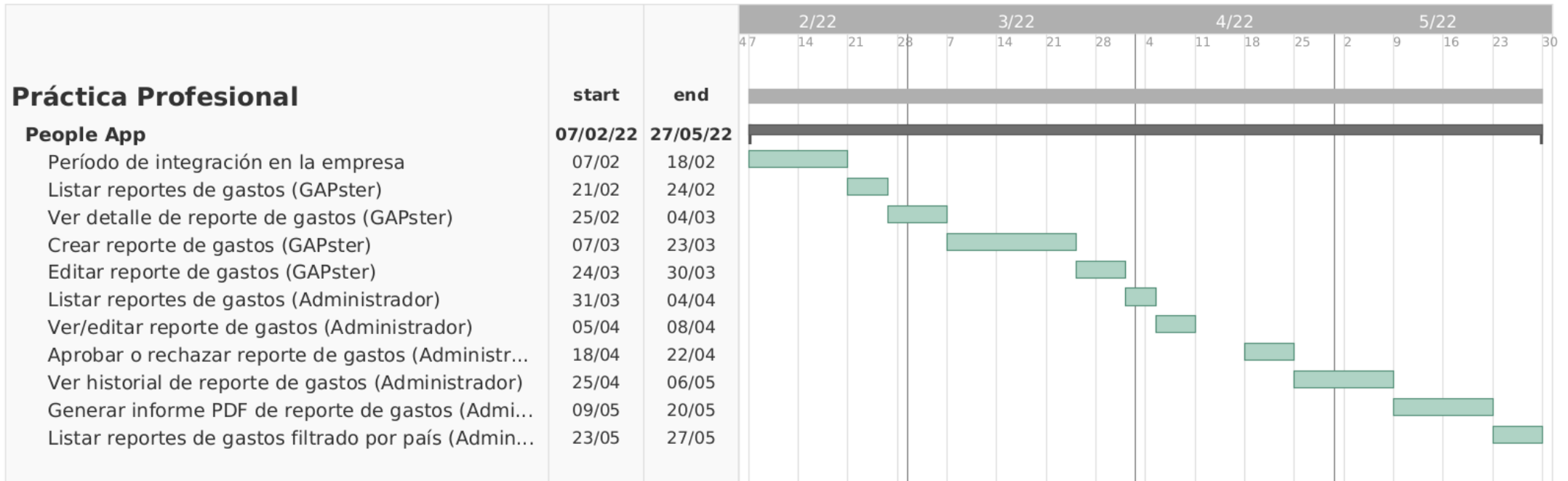
Tabla 10: Análisis de Riesgo 5

Referencia	Objetivos 3 y 4
Código	R-5
Nombre	Mal desarrollo de las historias de usuario.
Categoría	Personas.
Causa	Quedan dudas con las definiciones de las historias de usuario.
Impacto	Los entregables requieren cambios y retrabajo.
Estrategia de evasión	No existe.
Estrategia de mitigación	<ul style="list-style-type: none"> - Se trabaja con metodologías ágiles que permiten ver el avance diario de las tareas y aclarar las dudas de los desarrolladores. - Se pueden realizar reuniones con el PO con facilidad para aclarar dudas en las historias de usuario.
Estrategia de contingencia	Se realiza una validación con el PO de cada entregable e historia de usuario de manera temprana, esto da tiempo para realizar los cambios necesarios.

Fuente: elaboración propia

1.5. Cronograma de trabajo

Figura 1: Cronograma de Gantt



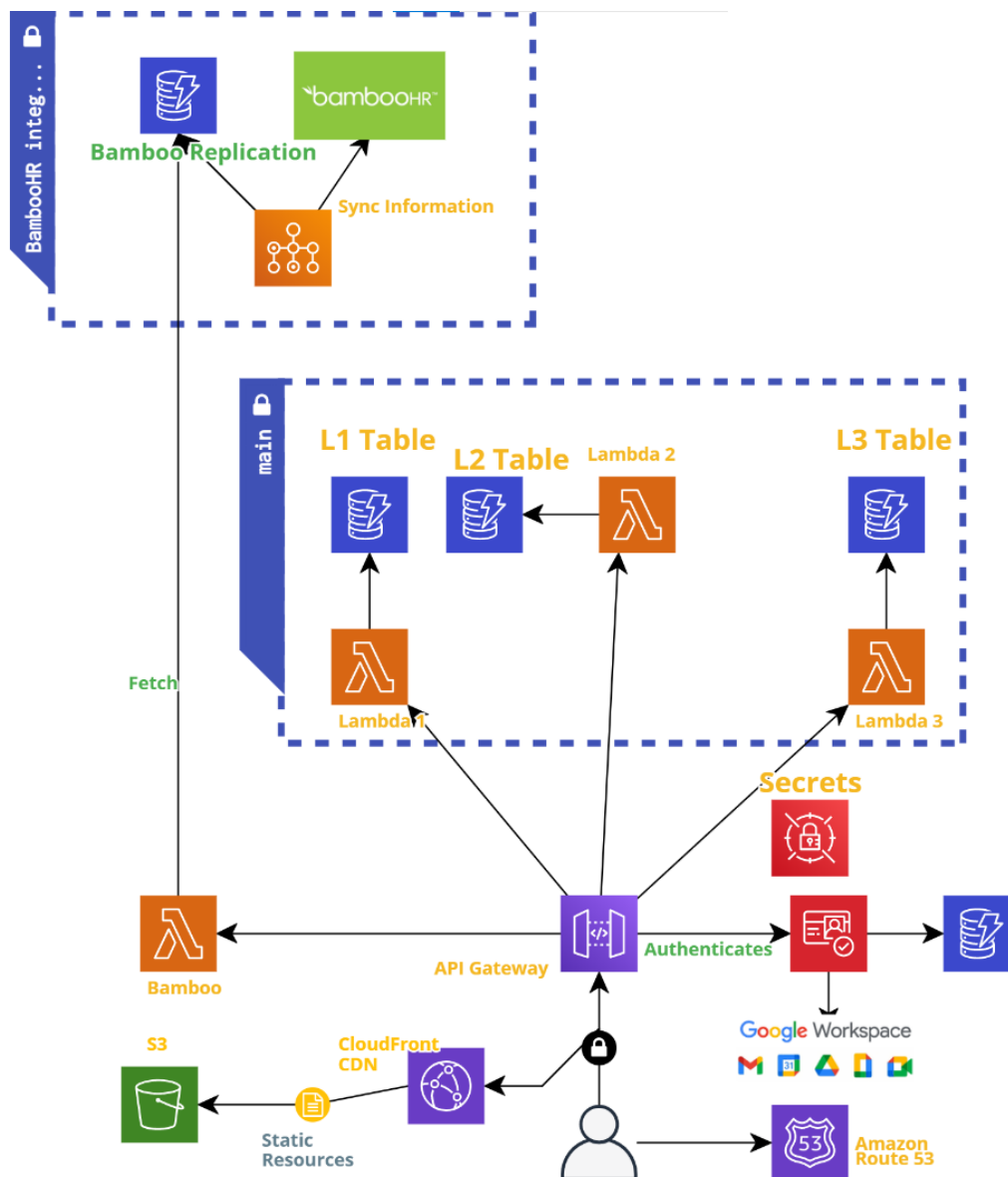
Fuente: elaboración propia

1. Requerimientos y Diseño

1.1. Arquitectura de la aplicación

En el siguiente diagrama se presenta la arquitectura de la aplicación que trabaja con distintos servicios de AWS y en especial AWS Lambda, servicio con el que se desarrolla el backend de la plataforma People App.

Figura 2: Arquitectura de la Aplicación

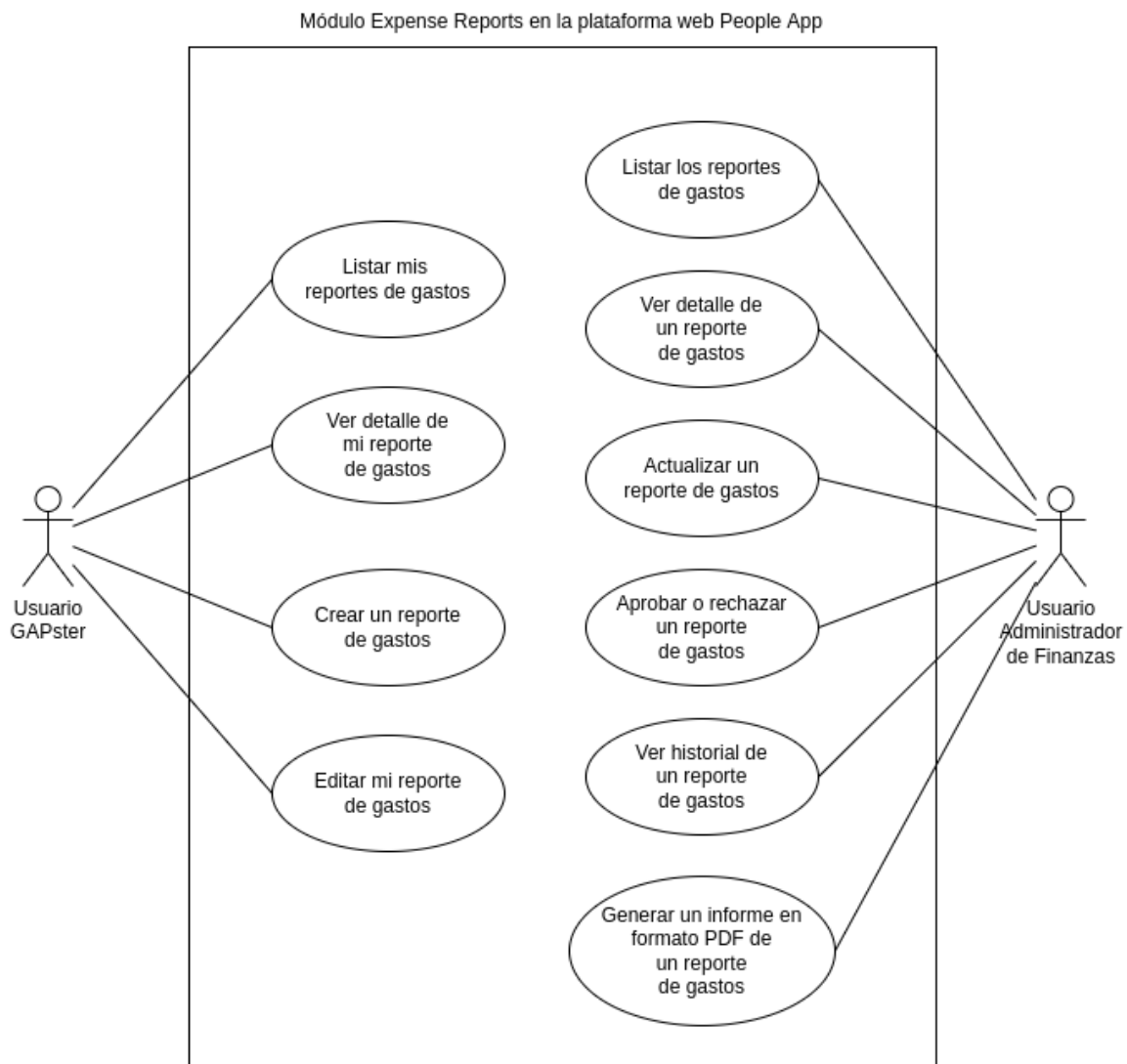


Fuente: elaboración propia

1.2. Casos de Uso

Se presenta el diagrama de casos de uso para el módulo Expense Reports de la plataforma People App.

Figura 3: Casos de Uso



Fuente: elaboración propia

1.3. Historias de Usuario

En esta sección se presentan las historias de usuario y los criterios de aceptación para cada una de estas, además del resultado final del desarrollo aprobado por el PO.

Tabla 11: User Story 1

Title	As a GAPster, I can list all my expense reports
Code	PET-146
User Story	As a GAPster, I want to list all my expense reports so that I can know when to expect them to be paid.
Acceptance Criteria	<ol style="list-style-type: none"> 1. There should be a new link in the main dashboard that redirects the user to this new page (use a similar format as the existing 3 buttons at the top). 2. The new page will display a list of all the existing items created for the logged-in employee (provide the employee id as one path parameter to the API). 3. The list should include the following fields: “merchant”, “amount” (in USD, CRC, COP), “date”, “Customer Name”, “Status”, “description” (optional, we may need to display it just if we have room for it). 4. The list of items should be sorted by default by “date” field (Desc). 5. The user can sort by any column displayed in the UI.

Fuente: elaboración propia

Figura 4: User Story 1

The screenshot shows a web interface for 'Expense Reports'. At the top, there is a green header with a dollar sign icon and the text 'Expense Reports'. Below this is a white card titled 'MY REPORTS' with an 'Add' button. The main content is a table with the following data:

Date	Amount	Reason	Status	Merchant	Actions
05/17/22	CRC 35,000	Employee Loan	PAYED	ExtremeTech	[Icon]
05/17/22	CRC 350,000	Employee Loan	PENDING APPROVAL	Amazon	[Icon] [Icon]
05/12/22	USD 7,800	Travels (Meals, lodging, transportation)	APPROVED	Hotel Luna	[Icon]
05/03/22	USD 1,000	Education Bonus	REJECTED	Platzi	[Icon] [Icon]

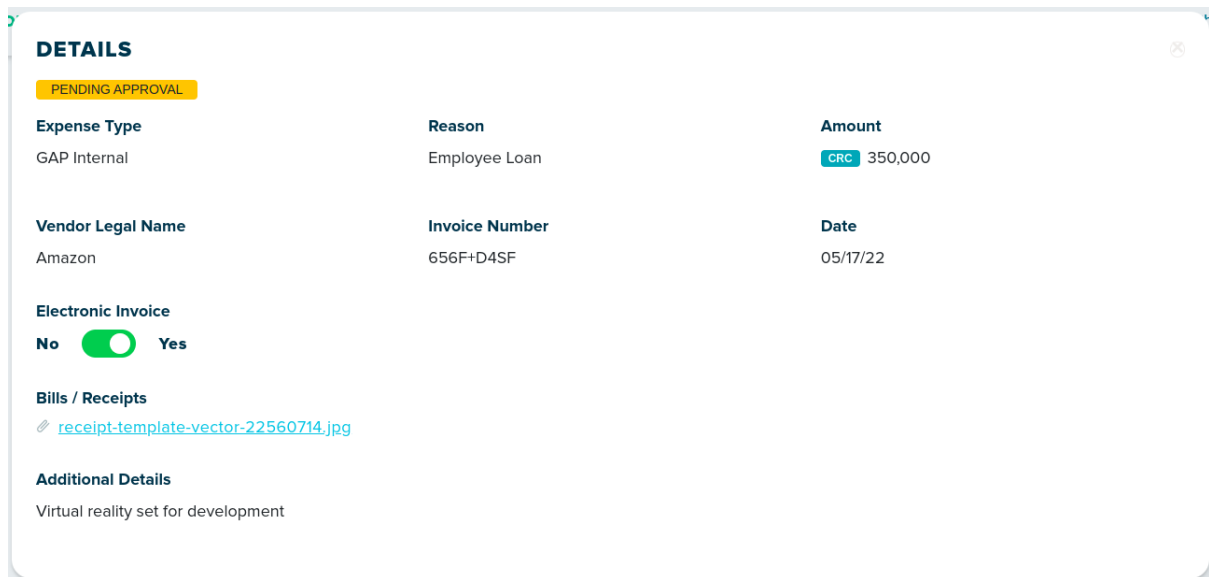
Fuente: elaboración propia

Tabla 12: User Story 2

Title	As a GAPster, I can see the detail of one of my expense report
Code	PET-147
User Story	As a GAPster, I want to view the details of one of my expense reports so that I can know detailed information about my request.
Acceptance Criteria	<ol style="list-style-type: none"> 1. The details should be displayed in a modal box. 2. The page should include all the attributes that we have as part of the expense report. 3. The user cannot perform any action in the modal (only close the modal).

Fuente: elaboración propia

Figura 5: User Story 2



Fuente: elaboración propia

Figura 6: User Story 2

DETAILS

APPROVED

Expense Type Client Billable	Client Project Name People Tool	Reason Travels (Meals, lodging, transportation)
Amount USD 7,800	Vendor Legal Name Hotel Luna	Invoice Number GYGJ+6452UGS
Date 05/12/22	Electronic Invoice No <input checked="" type="checkbox"/> Yes	

Bills / Receipts
[GAPlogo.svg](#)

Additional Details
I manage to get an offer

Fuente: elaboración propia

Tabla 13: User Story 3

Title	As a GAPster, I can create a new expense report
Code	PET-148
User Story	As a GAPster, I want to submit an expense report so that I can get reimbursed quickly.
Acceptance Criteria	<ol style="list-style-type: none"> 1. Wizard-like modal box to submit an expense report. <ol style="list-style-type: none"> a. First step: <ol style="list-style-type: none"> i. Type of Expense (GAP Internal / Client Billable) - Button or CTA for each option. b. Second step: Client Info (only visible if type of expense is Client Billable). <ol style="list-style-type: none"> i. Client Project Name: Auto-complete field. Nice to have: Only options available should be the clients they have permissions. ii. Reason: Auto-complete field. Reasons for client approved expense. See the link referenced below for additional details.. iii. Back button. iv. Next button. c. Third step: GAP Internal Info (only visible if type of

expense is GAP Internal).

- i. Reason: Auto-complete field. Reasons for gap internal approved expense. See the link referenced below for additional details.
 - ii. Back button.
 - iii. Next button.
- d. Fourth step: Amount.
- i. Amount.
 - ii. Currency: Auto-complete field. If only one available option, select it by default.
 1. If Client Billable, only USD available.
 2. CRC, USD and COP available for GAP Internal.
 - iii. Back button.
 - iv. Next button.
- e. Fifth step: Upload.
- i. Bills / Receipts: Upload file from the computer (ideally one per receipt). Mandatory to upload a file. For now, we need to limit the number of files to just one.
 - ii. Back button.
 - iii. Next button.
- f. Sixth step: Other Details.
- i. Additional Details: Comment box. 3 lines.
 - ii. Electronic Invoice: Yes/No Radio button. No selected by default.
 - iii. Vendor Legal Name: Open regular text box.
 - iv. Invoice Number: Open regular text box. Numbers, characters and special characters allowed.
 - v. Back button.
 - vi. Submit button.

2. Upon submission, the expense report record should be created in Pending Approval status, and assigned to the appropriate country responsible (CR, CO and Others).

Fuente: elaboración propia

Figura 7: User Story 3 First Step

NEW EXPENSE REPORT

Expense Type

GAP Internal Client Billable

Back 1 Next

Fuente: elaboración propia

Figura 8: User Story 3 Second Step

NEW EXPENSE REPORT

Client Project Name Reason

People Tool Team Building / Social Activity

Back 2 Next

Fuente: elaboración propia

Figura 9: User Story 3 Third Step

NEW EXPENSE REPORT

Reason

Cafeterian Plan

Back 2 Next

Fuente: elaboración propia

Figura 10: User Story 3 Fourth Step

NEW EXPENSE REPORT

Amount
2500

Currency
USD

Back 3 Next

Fuente: elaboración propia

Figura 11: User Story 3 Fifth Step

NEW EXPENSE REPORT

display_currency.jpg Browse

Back 4 Next

Fuente: elaboración propia

Figura 12: User Story 3 Sixth Step

Fuente: elaboración propia

Tabla 14: User Story 4

Title	As a GAPster, I can edit one of my expense reports
Code	PET-149
User Story	As a GAPster, I want to edit an expense report so that I can get reimbursed quickly.
Acceptance Criteria	<ol style="list-style-type: none"> 1. List of existing reports. Only the ones in Pending Approval have the option to edit. When editing, the same wizard-like modal for the creation is opened. 2. Wizard-like modal box to submit an expense report. <ol style="list-style-type: none"> a. First step: <ol style="list-style-type: none"> i. Type of Expense (GAP Internal / Client Billable) - Button or CTA for each option. b. Second step: Client Info (only visible if type of expense is Client Billable). <ol style="list-style-type: none"> i. Client Project Name: Auto-complete field. Nice to have: Only options available should be the clients they have permissions. ii. Reason: Auto-complete field. Reasons for client approved expense. See the link referenced below for additional details. iii. Back button. iv. Next button.

	<p>c. Third step: GAP Internal Info (only visible if type of expense is GAP Internal).</p> <ul style="list-style-type: none"> i. Reason: Auto-complete field. Reasons for gap internal approved expense. See the link referenced below for additional details. ii. Back button. iii. Next button. <p>d. Fourth step: Amount.</p> <ul style="list-style-type: none"> i. Amount. ii. Currency: Auto-complete field. If only one available option, select it by default. <ul style="list-style-type: none"> 1. If Client Billable, only USD available. 2. CRC, USD and COP available for GAP Internal. iii. Back button. iv. Next button. <p>e. Fifth step: Upload.</p> <ul style="list-style-type: none"> i. Bills / Receipts: Upload file from the computer (ideally one per receipt). Mandatory to upload a file. For now, we need to limit the number of files to just one. ii. Back button. iii. Next button. <p>f. Sixth step: Other Details.</p> <ul style="list-style-type: none"> i. Additional Details: Comment box. 3 lines. ii. Electronic Invoice: Yes/No Radio button. No selected by default. iii. Vendor Legal Name: Open regular text box. iv. Invoice Number: Open regular text box. Numbers, characters and special characters allowed. v. Back button. vi. Submit button. <p>3. Upon Saving, the expense report record is saved, continues to be in Pending Approval status, and assigned to the appropriate country responsible (CR, CO and Others).</p>
--	---

Fuente: elaboración propia

Figura 13: User Story 4 First Step

EDITING EXPENSE REPORT

Expense Type

GAP Internal Client Billable

Back 1 Next

Fuente: elaboración propia

Figura 14: User Story 4 Second Step

EDITING EXPENSE REPORT

Client Project Name Reason

People Tool Client Bonus

Back 2 Next

Fuente: elaboración propia

Figura 15: User Story 4 Third Step

EDITING EXPENSE REPORT

Reason

Employee Loan

Back 2 Next

Fuente: elaboración propia

Figura 16: User Story 4 Fourth Step

EDITING EXPENSE REPORT

Amount
2000.00

Currency
USD

Back 3 Next

Fuente: elaboración propia

Figura 17: User Story 4 Fifth Step

EDITING EXPENSE REPORT

expense_reports_list.png Browse

Back 4 Next

Fuente: elaboración propia

Figura 18: User Story 4 Sixth Step

EDITING EXPENSE REPORT

Vendor Legal Name: SPA

Invoice Number: EFDF46SD6F

Additional Details: Good Work!

Electronic Invoice: No Yes

Back 5 Save

Fuente: elaboración propia

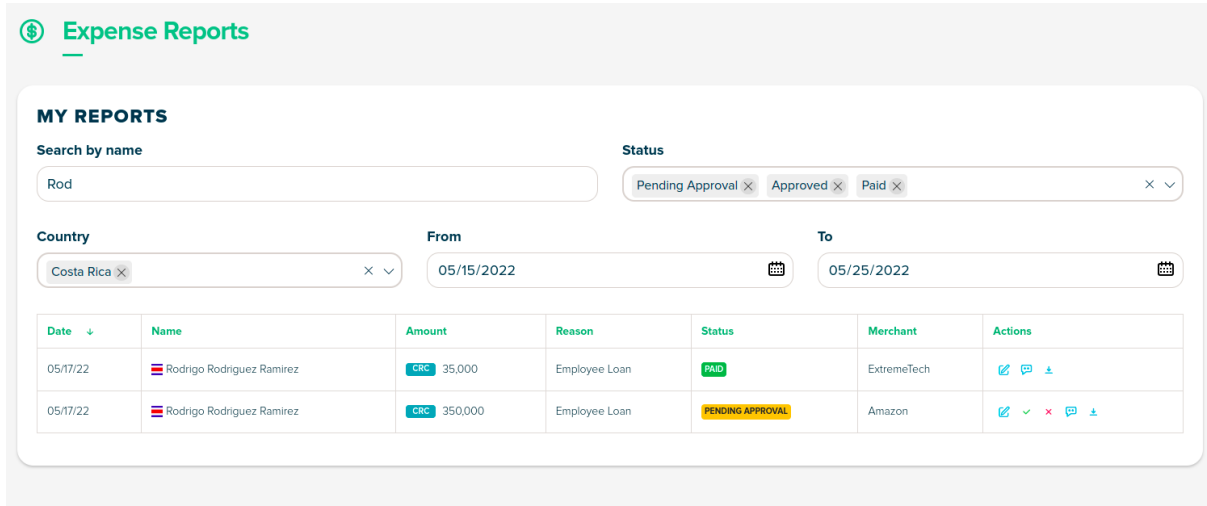
Tabla 15: User Story 5

Title	As a Finance Manager, I can see a list of all the expense reports submitted by GAPsters
Code	PET-150
User Story	As a Finance Manager, I want to see all pending expense reports that are assigned to me so that I can reimburse them quickly.
Acceptance Criteria	<ol style="list-style-type: none"> 1. New module on the left navigation module called Expense Reports. This module should be available to users with a Finance Manager role only. Use the same icon as the title from Expense Reports used in the My Dashboard section. 2. Upon loading the new module, a list is displayed with all existing expense reports prefiltered by my country. The list should be also prefiltered with status Pending Approval. <ol style="list-style-type: none"> a. Date (of expense report registration). b. Name (GAPster) - Add a Flag representing the country of the GAPster. c. Amount. d. Reason. e. Status. f. Merchant. 3. If no expense report is available, a zero state is displayed. Message: No available expense reports.

	<p>4. Ability to filter by status, date range, and name. User shouldn't be able to view expense reports from other countries.</p> <p>5. Ability to order by any field.</p>
--	--

Fuente: elaboración propia

Figura 19: User Story 5



Fuente: elaboración propia

Tabla 16: User Story 6

Title	As a Finance Manager, I can view the detail of an expense report submitted by a GAPster
Code	PET-151
User Story	As a Finance Manager, I want to view details of any given expense reports that are assigned to me so that I can reimburse them quickly.
Acceptance Criteria	<ol style="list-style-type: none"> From the list of reports, each of the expense reports listed will have options to: <ol style="list-style-type: none"> View and update expense reports assigned to me. Approve (PET-152). Reject (PET-152). View History (PET-185). Generate pdf (PET-186). View and update expense report: By clicking on each row in the table, the prepopulated modal with the expense report fields is displayed. Fields can be edited by the user.

3. Attached files should be downloadable.

Fuente: elaboración propia

Figura 20: User Story 6 First Step

GAPSTER EXPENSE REPORT

Expense Type

GAP Internal Client Billable

Back 1 Next

Fuente: elaboración propia

Figura 21: User Story 6 Second Step

GAPSTER EXPENSE REPORT

Client Project Name Reason

People Tool Welcome Lunch

Back 2 Next

Fuente: elaboración propia

Figura 22: User Story 6 Third Step

GAPSTER EXPENSE REPORT

Reason

Leadership Bonus

Back 2 Next

Fuente: elaboración propia

Figura 23: User Story 6 Fourth Step

GAPSTER EXPENSE REPORT

Amount 1000.00 **Currency** USD

Back 3 Next

Fuente: elaboración propia

Figura 24: User Story 6 Fifth Step

GAPSTER EXPENSE REPORT

Bills / Receipts

[display_currency.jpg](#)

Back 4 Next

Fuente: elaboración propia

Figura 25: User Story 6 Sixth Step

GAPSTER EXPENSE REPORT

Vendor Legal Name: El Rey

Invoice Number: G56SD4F6SF

Additional Details

Electronic Invoice: Yes

Back 5 Save

Fuente: elaboración propia

Tabla 17: User Story 7

Title	As a Finance Manager, I can approve or reject one of the expense reports
Code	PET-152
User Story	As a Finance Manager, I want to approve all pending expense reports from my country so that I can reimburse them quickly.
Acceptance Criteria	<ol style="list-style-type: none">1. From the list of reports, each of the expense reports listed will have options to:<ol style="list-style-type: none">a. View and update expense reports assigned to me (PET-151).b. Approve.c. Reject.d. View History (PET-185).e. Generate pdf (PET-186).2. Option to reject:<ol style="list-style-type: none">a. Modal:<ol style="list-style-type: none">i. Comments. Comment box. 3 lines.ii. Cancel button. Secondary action.iii. Reject button. Main action.b. Upon submitting, the expense report changes to Rejected

	<p>status. GAPster is notified via email.</p> <p>c. Create expense report history record.</p> <p>d. Rejected expense reports could be adjusted by the GAPster and changed back to Pending Approval later.</p> <p>3. Option to approve:</p> <p>a. Modal:</p> <ol style="list-style-type: none"> i. Status. Autocomplete field. 3 options: Pending Approval; Approved; Paid. ii. Comments. Comment box. 3 lines. iii. Cancel button. Secondary action. iv. Confirm button. Main action. <p>b. Upon submitting, the expense report changes the specified status. GAPster is notified via email.</p> <p>c. Create expense report history record.</p> <p>NOTE:</p> <ul style="list-style-type: none"> • Format for the notification email should follow the structure below: <ul style="list-style-type: none"> ○ Subject: Expense Report in approved/rejected status. ○ Body: Your Expense Report ([Type] - [\$amount]) created on [createdate] has just been approved/rejected by [approvername] with the following comments: “[comments]”.
--	---

Fuente: elaboración propia

Figura 26: User Story 7 Approval Options

The screenshot shows a web form titled "APPROVAL FORM". At the top left, the word "Status" is written in bold. Below it is a dropdown menu with a white background and a downward arrow on the right. The menu is open, showing four options: "Please Select" (highlighted in grey), "Pending Approval", "Approved", and "Paid". Below the dropdown is a large, empty text input field with rounded corners. At the bottom right of the form, there are two buttons: "Cancel" (with a blue border) and "Submit" (with a grey border).

Fuente: elaboración propia

Figura 27: User Story 7 Approval Form

APPROVAL FORM

Status
Paid

Comments
The expenses were paid successfully

Cancel Submit

Fuente: elaboración propia

Figura 28: User Story 7 Rejection Form

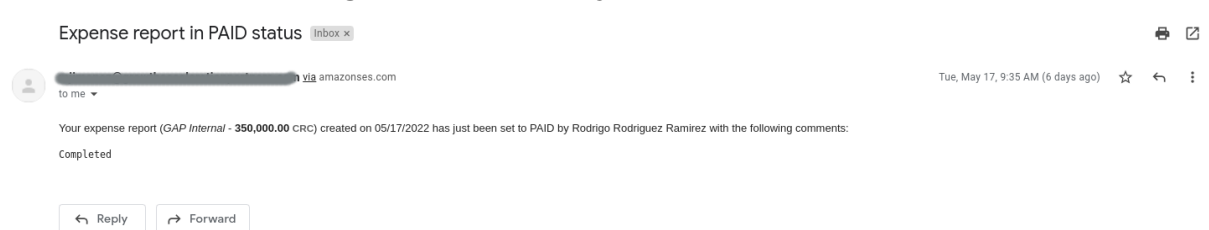
REJECTION FORM

Comments
The submitted receipts include invalid items

Cancel Submit

Fuente: elaboración propia

Figura 29: User Story 7 Email Notification



Fuente: elaboración propia

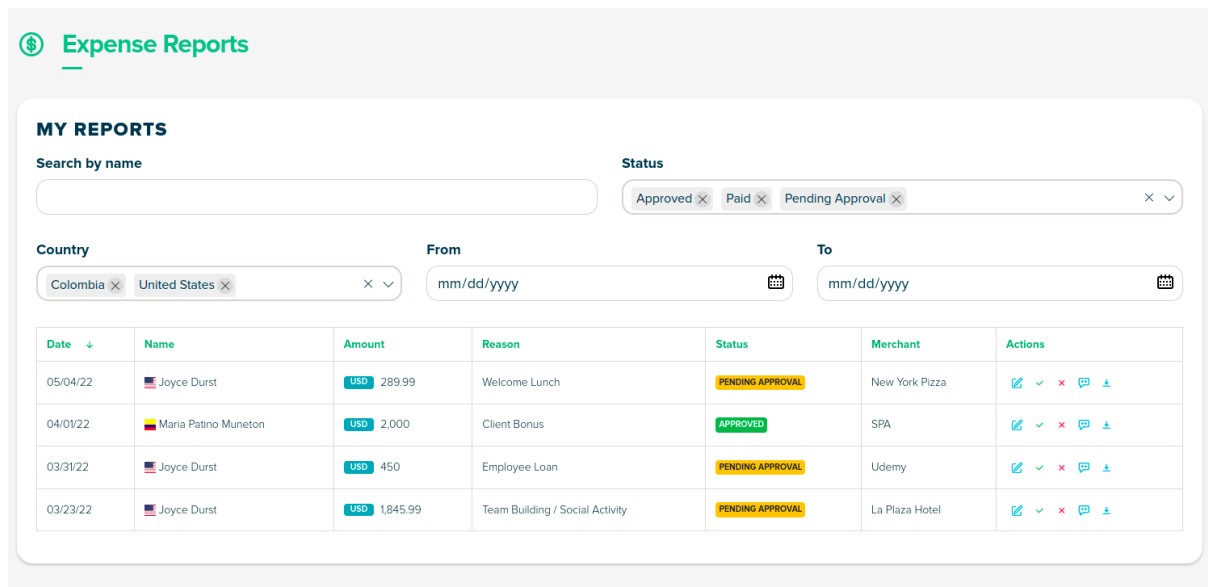
Tabla 18: User Story 8

Title	As a Finance manager, I can filter all expense reports by Country
Code	PET-153
User Story	As a Finance Manager, I want to filter all expense reports by Country so that I can reimburse them quickly.

Acceptance Criteria	<ol style="list-style-type: none"> Expense Reports module can be filtered by country (CR, CO and Others). The user can select more than one country simultaneously. By default, Expense reports should be filtered by the user's country.
---------------------	---

Fuente: elaboración propia

Figura 30: User Story 8



Fuente: elaboración propia

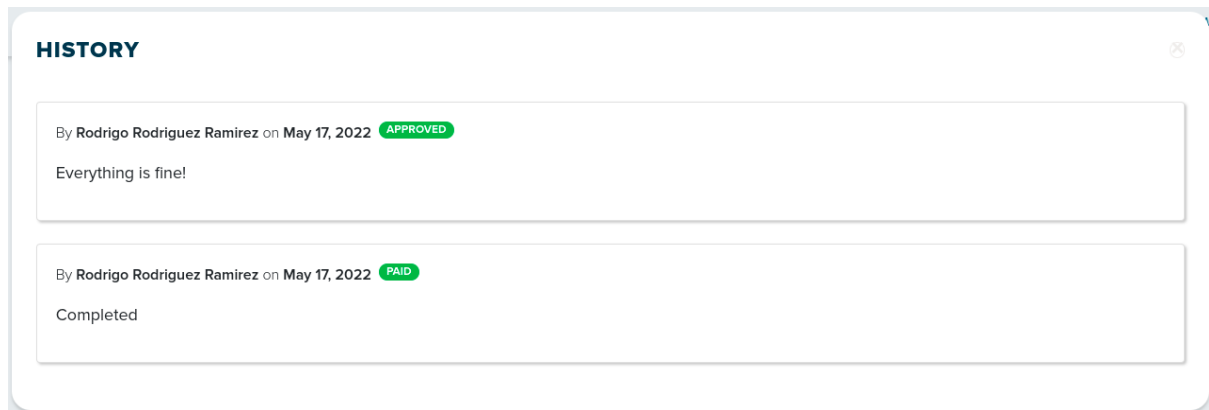
Tabla 19: User Story 9

Title	As a Finance manager, I can view the history of a expense report submitted by a GAPster
Code	PET-185
User Story	As a Finance Manager, I want to view the history of any given expense reports that are assigned to me so that I can reimburse them quickly.
Acceptance Criteria	<ol style="list-style-type: none"> From the list of reports, each of the expense reports listed will have options to: <ol style="list-style-type: none"> View and update expense reports assigned to me (PET-151). Approve (PET-152). Reject (PET-152).

	<ul style="list-style-type: none"> d. View History. e. Generate pdf (PET-186). <p>2. Option to view history:</p> <ul style="list-style-type: none"> a. Modal. <ul style="list-style-type: none"> i. Date. ii. Status. iii. Responsible. iv. Comments.
--	---

Fuente: elaboración propia

Figura 31: User Story 9



Fuente: elaboración propia


Tabla 20: User Story 10

Title	As a Finance manager, I can generate a PDF report of an expense report submitted by a GAPster
Code	PET-186
User Story	As a Finance Manager, I want to generate a PDF report of any given expense reports that are assigned to me so that I can reimburse them quickly.
Acceptance Criteria	<ol style="list-style-type: none"> 1. From the list of reports, each of the expense reports listed will have options to: <ul style="list-style-type: none"> a. View and update expense reports assigned to me (PET-151). b. Approve (PET-152). c. Reject (PET-152).

- d. View History (PET-185).
 - e. Generate pdf.
2. Generate pdf. By clicking on this option, the printed pdf must include the image of the attached invoices and the status of the expense report.
 3. Add a restriction to the Expense Report create or update screens to limit the type of file that can be attached. Only pdf or image files are allowed. This applies for the GAPster module and the Expense Report module used by the Finance Manager.

Fuente: elaboración propia

Figura 32: User Story 10



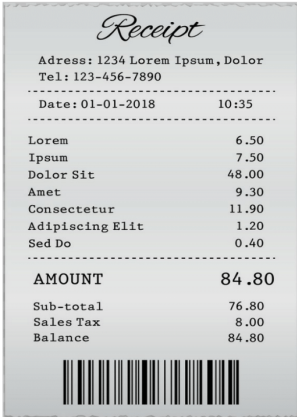
Expense Report 05/23/2022
Rodrigo Rodriguez Ramirez

Amazon

350,000.00 CRC

Virtual reality set for development 05/17/2022

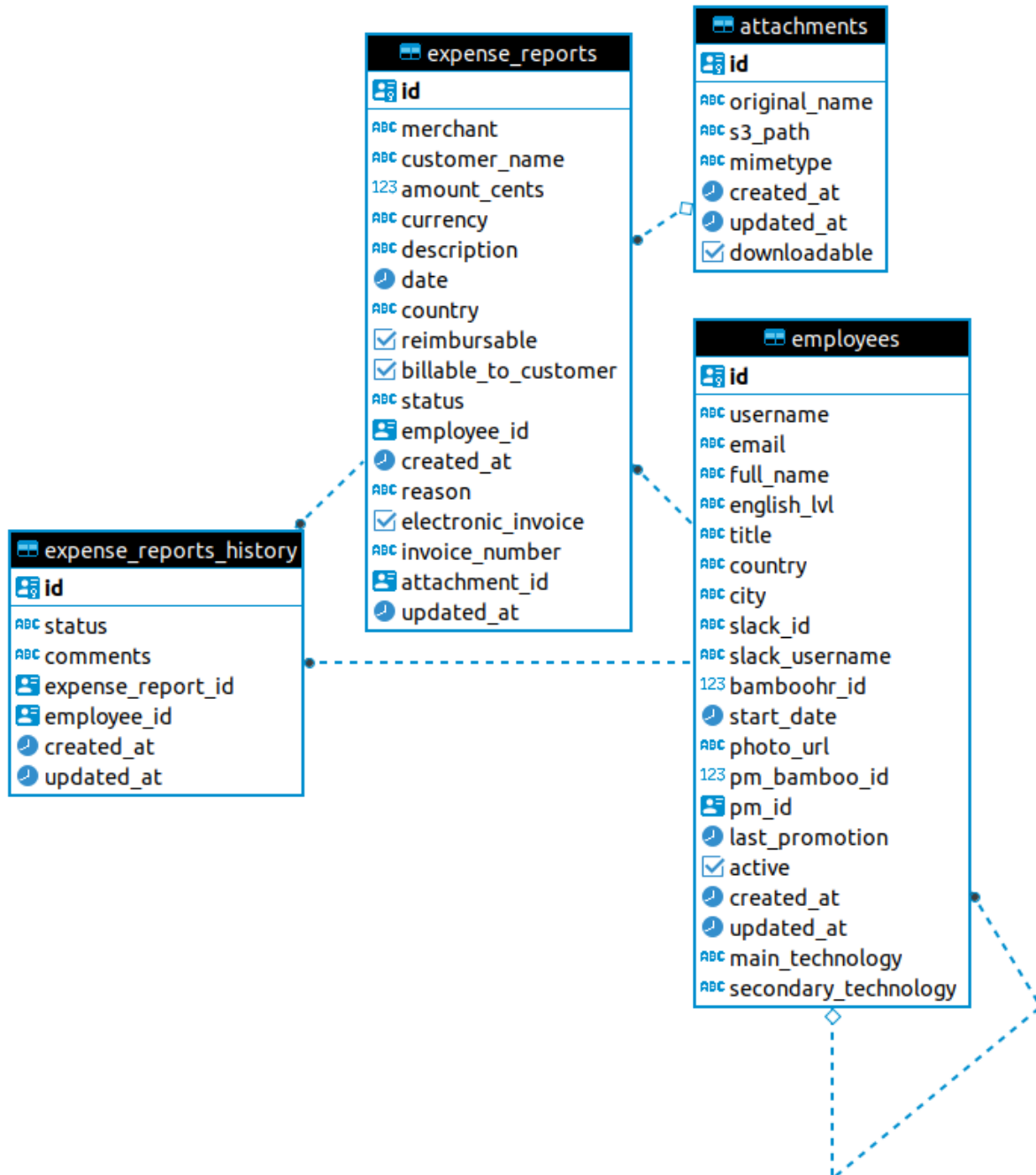
Type	Gapster	Reason	Country	Status
GAP Internal	Rodrigo Rodriguez Ramirez	Employee Loan	CRI	PAID



Fuente: elaboración propia

1.4. Diagrama de Base de Datos

Figura 33: Diagrama de Base de Datos



Fuente: elaboración propia

1. Plataforma de software

1.1. Aplicación

Para el desarrollo de este proyecto el trabajo como desarrollador es full stack, lo que significa que para cada historia de usuario se debe hacer el desarrollo tanto de backend, la aplicación del lado del servidor, como de frontend, la aplicación del lado del cliente.

En el backend se trabaja con el servicio Lambda de AWS en el lenguaje de programación Python, y para el almacenamiento de datos se utiliza el motor de base de datos PostgreSQL y AWS S3 para el almacenamiento de imágenes o archivos en formato PDF. La mayoría de las historias de usuario implican el desarrollo de una nueva función Lambda.

En el frontend se tiene una aplicación desarrollada en el lenguaje de programación JavaScript. Se usa ReactJS, una biblioteca para la construcción de interfaces de usuario, y Redux, una biblioteca para el manejo del estado de la aplicación.

En la siguiente tabla se muestra la relación entre las historias de usuario, las funciones Lambda desarrolladas y las tablas de la base de datos con las que interactúa cada función.

Tabla 21: Historias de Usuario - Funciones Lambda

Historia de Usuario	Función Lambda (API)	Descripción	Tablas Afectadas
PET-146	GET /api/1/employees/{id}/expense-reports	Listar los reportes de gastos presentados por un GAPster.	expense_reports
PET-147 PET-151	GET /api/1/employees/{id}/expense-reports/	Obtener un reporte de gastos presentado por un GAPster.	expense_reports, attachments

	{report_id}		
PET-148	POST /api/1/employees/{id}/expense-reports	Crear un reporte de gastos presentado por un GAPster.	expense_reports, attachments
PET-149	PUT /api/1/employees/{id}/expense-reports/{report_id}	Actualizar un reporte de gastos presentado por un GAPster.	expense_reports, attachments
PET-150 PET-153	GET /api/1/finance-managers/{employee_id}/expense-reports	Listar los reportes de gastos para un administrador de finanzas.	expense_reports, employees
PET-152	POST /api/1/finance-managers/{employee_id}/expense-reports/{report_id}/reviews	Aprobar o rechazar un reporte de gastos.	expense_reports, expense_reports_history
PET-185	GET /api/1/finance-managers/{employee_id}/expense-reports/{report_id}/reviews	Obtener el historial de un reporte de gastos.	expense_reports_history
PET-186	GET /api/1/finance-managers/{employee_id}/expense-reports/{report_id}/reports/pdf	Generar un informe en formato PDF para un reporte de gastos.	expense_reports, employees, attachments

Fuente: elaboración propia

La base de código para ambas partes, tanto backend como frontend, se maneja en un repositorio de la plataforma GitLab. Cada historia de usuario se desarrolla en una nueva rama del repositorio, cuando se completa el desarrollo se pasa por un proceso de revisión y QA (esto se explicará en el siguiente apartado), y cuando finalmente se aprueba el desarrollo, este se unifica con la rama principal.

En la siguiente tabla se presentan los resultados obtenidos del desarrollo de las historias de usuario.

Tabla 22: Resultados del Desarrollo

Historia de Usuario	Resultado
PET-146	Completo
PET-147	Completo
PET-148	Completo
PET-149	Completo
PET-150	Completo
PET-151	Completo
PET-152	Completo
PET-153	Completo
PET-185	Completo
PET-186	Completo

Fuente: elaboración propia

1.2. Evaluación

Para el proceso de evaluación de las historias de usuario desarrolladas se realizan 3 tareas: Code Review (revisión de código), QA (pruebas de aseguramiento de la calidad) y PO Review (revisión con el dueño del producto).

Code Review

En esta etapa se hace una revisión de código entre pares. También se cuenta con el apoyo de GAPsters que no son parte del proyecto People App, pero cuentan con más experiencia y ayudan con la tarea de Code Review. El desarrollador puede recibir solicitudes de mejoras en el código, este realiza los cambios solicitados y el ciclo de evaluación comienza de nuevo en esta etapa.

Como desarrollador fue una experiencia de aprendizaje y crecimiento recibir solicitudes de cambios en mi código, mejorar la calidad del mismo y hacer las cosas de una mejor manera. También realicé la tarea de Code Review para el desarrollo de otros colaboradores una vez que obtuve un poco de experiencia, hice algunas solicitudes de cambios en algunos casos y también aprendí viendo el código de los demás.

Para continuar con la siguiente etapa es requisito que la solución planteada y las mejoras realizadas sean aprobadas por los miembros del equipo.

Los cambios solicitados en esta etapa estaban relacionados con mejoras en el código, utilizar métodos más eficientes para la solución, hacer uso de nuevas funcionalidades que ofrecen los lenguajes de programación, mejorar la estructura y la lectura del código, o para seguir algún estándar del proyecto.

QA

Durante el período de la práctica profesional el proceso de QA fue distinto entre la primera y la segunda mitad del período:

- ★ Durante la primera mitad, no se contaba con un equipo de QA asignado al proyecto People App, y durante este tiempo la etapa de QA implicó la reunión con pares que verificaban que el desarrollo cumplía con el criterio de aceptación de las historias de usuario.
- ★ Al proyecto People App se le asignó un ingeniero de QA en la semana diez, y este se encargó de usar y probar la aplicación desde el punto de vista de un usuario final, aunque a más detalle, de manera más minuciosa. También verificó que el desarrollo cumplía con el criterio de aceptación de las historias de usuario. Cuando el ingeniero de QA encontraba algo erróneo, este lo reportaba como un bug (pulga) y el desarrollador debía corregirlo.

El proyecto People App aún no cuenta con pruebas de código unitarias.

Para continuar con la siguiente etapa es requisito que la solución planteada cumpla con todos los criterios de aceptación de la historia de usuario y que pase todas las pruebas definidas por el ingeniero de QA.

Para esta etapa la correcciones solicitadas estaban relacionadas a cumplir con los criterios de aceptación o con los casos de prueba.

PO Review

Luego de darse la aprobación de QA, prosigue una reunión con el PO en la que se le muestra el desarrollo realizado. El PO puede dar el visto bueno y dar finalización a la historia de usuario, o solicitar cambios. En caso de que el PO solicite cambios, el proceso de desarrollo y evaluación comienza de nuevo.

Los cambios solicitados por el PO fueron principalmente a nivel visual, como cambiar un botón de posición o para mantener la consistencia de estilos en toda la aplicación. También podría ser el caso de que el PO solicitara un cambio por algo que no estaba explícitamente definido en la historia de usuario, pero que al momento de la revisión sí le parecía necesario y que estaba directamente relacionado con la historia de usuario.

Tabla 23: Resultados de la Evaluación

Historia de Usuario	Code Review	QA	PO Review
PET-146	Aprobado	Aprobado	Aprobado
PET-147	Aprobado	Aprobado	Aprobado
PET-148	Aprobado	Aprobado	Aprobado
PET-149	Aprobado	Aprobado	Aprobado
PET-150	Aprobado	Aprobado	Aprobado
PET-151	Aprobado	Aprobado	Aprobado
PET-152	Aprobado	Aprobado	Aprobado

PET-153	Aprobado	Aprobado	Aprobado
PET-185	Aprobado	Aprobado	Aprobado
PET-186	Aprobado	Aprobado	Aprobado

Fuente: elaboración propia

Todo este proceso permite llevar un buen control y orden en el desarrollo de las historias de usuario y mantener un historial de los cambios y las nuevas funcionalidades que se van integrando a la plataforma People App. También asegura que el producto final tenga un alto nivel de calidad al pasar por las distintas etapas de aseguramiento de la calidad. Además, inculca el trabajo en equipo, es un proceso que requiere la participación de todos, se comparte conocimiento, el equipo mejora y crece constantemente, se siente el respaldo y el aporte de todos, con el objetivo común de entregar una solución de software de alta calidad.

1. Conclusiones

- ★ Con el uso de AWS Lambda se obtienen algunos de los beneficios que se ven en el desarrollo de sistemas con una arquitectura de microservicios, aunque ciertamente se podría argumentar que las funciones Lambda son microservicios. Cada función tiene un ciclo de vida independiente, su desarrollo, su proceso de QA y despliegue se trabajan de forma aislada y sin afectar al resto del sistema. Cada desarrollador puede implementar y modificar el código de una función sin alterar el trabajo de los demás. Permite una rápida puesta en producción de las nuevas funcionalidades, ya que cada nueva función Lambda implica una nueva funcionalidad y se integra rápidamente al sistema como un todo. Con esta tecnología se puede iterar más rápidamente sobre la solución, el PO ve la evolución del producto constantemente y se obtiene una retroalimentación oportuna que sirva al equipo, siendo esto algo que va en acorde con las prácticas de desarrollo ágil.

- ★ El desarrollo del módulo Expense Reports dentro de la plataforma People App le permite a la empresa agilizar el proceso de pago de viáticos. Al ser una solución a la medida, se adapta a las necesidades específicas de la empresa, y no es la empresa la que tiene que adaptarse a lo que pueda ofrecer una solución de terceros. A futuro se pueden hacer cambios y mejoras con el objetivo de continuar agilizando y facilitando dicho proceso, algo que no se puede hacer con una solución de terceros.

- ★ Contar con procesos que apoyen el aseguramiento de la calidad es importante y necesario para desarrollar y entregar productos de alta calidad, pero también es importante considerar los recursos con los que se cuenta en las distintas etapas del desarrollo y cómo esto afecta el avance del proyecto. Las etapas de Code Review, QA y PO review, que formaron parte del proceso de aseguramiento de la calidad de este proyecto, pueden ralentizar el progreso y avances en el proyecto si no se tiene disponibilidad de recursos suficientes, los desarrolladores se enfocan en el desarrollo de las funcionalidades y dejan de último los demás procesos del aseguramiento de la calidad. Además, cada etapa del desarrollo puede ser más demandante y consumir más tiempo cuando se tienen miembros con poca experiencia en el equipo, porque podrían requerir más ayuda, más acompañamiento,

tienen que adaptarse al proyecto, todo esto también consume tiempo. Si el equipo siente la presión de completar sus tareas de desarrollo en tiempos muy cortos, se reduce la calidad de su trabajo y se dejan de último las tareas correspondientes al proceso de QA.

1.1. Recomendaciones

- ★ Teniendo la agilidad que ofrece el desarrollo con la tecnología AWS Lambda, también se puede automatizar el proceso de integración y despliegue del sistema, lo que se conoce como CI/CD, Continuous Integration y Continuous Delivery/Continuous Deployment. Con esto, se pueden integrar constantemente y rápidamente los cambios y nuevos desarrollos del equipo, y poner en producción las nuevas funcionalidades del producto en tiempos muy cortos, todo de manera automatizada, lo que libera recursos valiosos que se pueden utilizar en tareas que den más valor al producto.
- ★ Al módulo Expense Reports se le puede agregar la opción de realizar los pagos desde la propia aplicación, esto agilizará y facilitará aún más el pago de los viáticos. Actualmente el pago de viáticos se realiza un solo día a la semana y de forma manual, por lo que si se agrega esta nueva funcionalidad, se podría automatizar este proceso para que el pago se realice todos los días para aquellos reportes que hayan sido aprobados, aunque esto va a depender de las opciones que ofrecen las entidades bancarias para interactuar con su sistema, y de esto último también dependen las futuras mejoras que se puedan hacer al sistema.
- ★ Es importante considerar, al momento de calcular el esfuerzo y tiempo para el desarrollo de cada historia de usuario, no solamente el esfuerzo que le tomará a los desarrolladores implementar las historias de usuario, sino también el esfuerzo y tiempo que se le dedica a las demás etapas del desarrollo, como los procesos de QA, para que el equipo pueda cumplir en cada entrega con un trabajo de gran calidad.

8. Bibliografía

Villamizar, M., Garcés, O., Ochoa, L., Castro, H., Salamanca, L., Verano, M., ... Lang, M. (2016). Infrastructure Cost Comparison of Running Web Applications in the Cloud Using AWS Lambda and Monolithic and Microservice Architectures. *Proceedings of the 16th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing*, 179–182. doi:10.1109/CCGrid.2016.37

Kaplunovich, A. (2019). ToLambda: Automatic Path to Serverless Architectures. *Proceedings of the 3rd International Workshop on Refactoring*, 1–8. doi:10.1109/IWoR.2019.00008

Aggarwal, S. (2018). Modern Web-Development using ReactJS. *International Journal of Recent Research Aspects*, 5(1), 133–137.

Novac, C. M., Novac, O. C., Sferle, R. M., Gordan, M. I., BUJDOSÓ, G., & Dindelegan, C. M. (2021). Comparative study of some applications made in the Vue.js and React.js frameworks. *2021 16th International Conference on Engineering of Modern Electric Systems (EMES)*, 1–4. doi:10.1109/EMES52337.2021.9484149