Instituto Tecnológico de Costa Rica

Carrera de Ingeniería en Computación,
Campus Tecnológico Local San Carlos

“Implementación de módulos PM Dashboard y Happiness en la aplicación People App de Growth Acceleration Partners”

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

José Fabian Alfaro Campos

Costa Rica, 2021

Resumen ejecutivo

El proyecto por desarrollar es llamado People app, una aplicación web de centralización y manejo de la información general de la compañía, los proyectos que se llevan a cabo y los empleados (GAPsters) que forman parte de Growth Acceleration Partners (GAP). Esta nació con el objetivo de reemplazar y modernizar una serie de productos preexistentes con el mismo enfoque que no han sido culminados satisfactoriamente, ya que su mantenimiento ha generado problemas dada la arquitectura de software existente y no hay un diseño visual consistente entre los distintos sistemas.

El problema que esta propuesta ataca es la dificultad de gestión de empleados y proyectos de GAP, de manera que no se puede llevar una administración adecuada de los proyectos que se están desarrollando, los empleados que forman parte de estos y su estado de ánimo con el trabajo asignado. Este problema surge debido al crecimiento exponencial en cuanto a la cantidad de miembros en los diferentes equipos y áreas de la empresa.

La solución implementada se detalla mediante un apartado de tareas completadas, en el cual se describen los requerimientos, las historias de usuario, los resultados de implementación de cada una de estas y la evaluación a la que fueron sometidas, la cual evidencia cómo cumple satisfactoriamente con las necesidades de calidad del cliente, para que se califique como una solución viable y positiva al problema que se enfrenta.

Palabras clave — Software ERP, Arquitectura de microservicios, React, Amazon Web Services

Executive summary

The project to be developed is called People app, a web application for centralizing and managing the general information of the company, the projects that are carried out, and the employees that are part of Growth Acceleration Partners. This was born with the aim of replacing and modernizing a series of pre-existing products with the same approach that has not been successfully completed, since their maintenance has generated problems given the existing software architecture and that there is no consistent visual design between the different systems.

The problem that this proposal attacks is the difficulty of managing employees and GAP projects so that an adequate administration of the projects that are being developed, the employees that are part of these, and their state of mind with the assigned work. This problem arises due to the exponential growth in the number of members in the different teams and areas of the company.

The implemented solution is detailed through a section of completed tasks, which describes the requirements, user stories, implementation results of each of these, and the evaluation to which they were submitted to qualify as a viable solution. and positive to the problem that is faced, in such a way that it is shown how it satisfactorily meets the quality needs of the client.

Keywords — ERP Software, Microservices Architecture, React, Amazon Web Services

Tabla de contenidos

Índice de figuras

Índice de tablas

Lista de abreviaturas

PM: Personal manager

DM: Delivery Manager

AWS: Amazon Web Services

DNS: Domain Name System

JS: Javascript

SQL: Structured Query Language

RDS: Relational Database Service

Lista de acrónimos

GAP: Growth Acceleration Partners

API: Application Programming Interface

1. Introducción

Este documento presenta la propuesta de proyecto de práctica profesional, realizado como requisito final de graduación de la carrera de Ingeniería en Computación, el cual describe detalles del problema presente, la solución planteada para enfrentarlo y se complementa con otros apartados para mostrar la forma en que el mismo se aborda y el proceso que conlleva.

El proyecto por desarrollar es llamado People app, una aplicación web de centralización y manejo de la información general de GAP, los proyectos que se llevan a cabo y acerca de los GAPsters. Esta nació con el objetivo de reemplazar y modernizar una serie de productos preexistentes con el mismo enfoque.

Asimismo, trae consigo beneficios al desarrollo tecnológico de la compañía, ya que, como se mencionó anteriormente, esta aplicación representa una modernización en los sistemas actuales con los estándares de desarrollo de la institución, lo que implica un sistema más estable, eficiente, escalable y con mayor facilidad de mantenimiento al fomentar la utilización de tecnologías modernas.

1.1. Descripción de empresa

Growth Acceleration Partners es una empresa que nació en Austin, Texas de los Estados Unidos de América, que tiene sede en Costa Rica, Colombia y Estados Unidos. Esta empresa se dedica al desarrollo de software hacia clientes por medio de subcontratación (outsource) y más recientemente en la creación de soluciones integrales que permitan resolver problemas específicos de los clientes o su modernización técnica. La empresa brinda servicios de análisis, nube, dispositivos móviles y automatización de control de calidad. Cabe destacar que, la misión de la empresa se enfoca en crear asociaciones estratégicas con los 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

People app es una aplicación web de centralización y manejo de la información general de la compañía, los proyectos que se llevan a cabo y los empleados que forman parte de GAP. Esta cuenta con distintos módulos que se van desarrollando con el objetivo de reemplazar dos de las aplicaciones preexistentes, antes del final del 2021 y con ello reducir las necesidades de mantenimiento de estas.

Los principales módulos del sistema incluyen un control del estado de los proyectos 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 empleados en GAP (Happiness dashboard); el seguimiento de la carrera profesional de cada uno de los GAPsters (portfolio & career development); el análisis de las tecnologías que más se utilizan en la organización según proyecto y empleado (Technology dashboard); asignación de los recursos a los diferentes proyectos (Resources Allocation); y la visualización de los datos personales y profesionales de cada uno de los GAPsters en la compañía (My Dashboard).

Por otro lado, es importante señalar que el alcance completo del proyecto excede el tiempo disponible para la Práctica Profesional, por lo que, el alcance del proyecto será acotado a dos módulos (PM Dashboard y Happiness Dashboard) y se contempla como un incremento de People App, con el fin de asegurar que la práctica culmine satisfactoriamente y a la vez el proyecto cumpla con el avance esperado.

1.3. Problema

El problema que esta propuesta ataca es la dificultad de gestión de empleados y proyectos de GAP, de manera que no se puede llevar una administración adecuada de los proyectos que se están desarrollando, los empleados que forman parte de estos y su estado de ánimo con el trabajo asignado. Este problema surge debido al crecimiento exponencial en cuanto a la cantidad de miembros en los diferentes equipos y áreas de la empresa. En el pasado, se ha intentado la creación de distintos sistemas de software a la medida, sin embargo,

no han sido culminados satisfactoriamente, ya que su mantenimiento ha generado problemas dada la arquitectura de software existente y no hay un diseño visual consistente entre los distintos sistemas. Además, estos sistemas internos son desarrollados por empleados que no tienen asignación de proyectos con clientes y al utilizar 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 son muy limitados.

1.4. Objetivos

1.4.1. Objetivo general

Desarrollar módulos adicionales en una plataforma web interna de la compañía Growth Acceleration Partners llamada People App, para agilizar la gestión de empleados desde la perspectiva de los encargados de personal (PM Dashboard) y los encargados de felicidad (Happiness Dashboard), utilizando tecnologías de desarrollo web enfocadas en frontend, con una interfaz de usuario intuitiva que respete los lineamientos de diseño de la compañía y consumiendo un backend con una arquitectura basada en microservicios preexistente en la nube a través de Amazon Web Services (AWS).

1.4.2. Objetivos específicos

1. Definir los requerimientos e historias de usuario de los módulos PM Dashboard y Happiness Dashboard de la aplicación People App.
2. Definir diagrama de arquitectura de los módulos PM Dashboard y Happiness Dashboard de la aplicación People App.
3. Implementar los módulos PM Dashboard y Happiness Dashboard de la aplicación People App.
4. Evaluar el funcionamiento de los módulos PM Dashboard y Happiness Dashboard de la aplicación People App.

1.5. Justificación

People App es una herramienta innovadora y viene a reemplazar una serie de productos preexistentes en la compañía con la idea de modernizarlos, lo que representa un sistema más estable, eficiente, escalable y con mayor facilidad de mantenimiento al fomentar la utilización de tecnologías modernas como React JS y Cloud. A nivel general, la aplicación contempla módulos específicos para el establecimiento del control de empleados, control de proyectos, seguimiento de la "felicidad" de los GAPsters con respecto al tiempo, seguimiento de la carrera profesional de cada GAPster y el reporte de información de diversa índole según los datos almacenados. Todo centralizado en una sola herramienta a nivel visual, pero desarrollado con las tecnologías anteriormente mencionadas y una arquitectura basada en microservicios. Por lo tanto, contribuye directamente a todos los colaboradores al ser un sistema de gestión de uso interno que permite llevar un registro más automatizado y organizado de las diferentes secciones de la compañía.

2. Marco teórico

People App al ser una aplicación web conlleva una implementación bajo una tecnología como JavaScript, la cual según Mozilla (2020), “Es un lenguaje de programación de alto nivel y multiparadigma que se utiliza principalmente para añadir características complejas a páginas web”. Por lo tanto, Javascript resulta ser una herramienta indispensable, ya que es el lenguaje de programación a utilizar para el desarrollo del frontend, el cual posee cierta complejidad ya que cuenta con una cantidad de módulos significante, por lo cual, se recurre a la utilización de una librería como React JS, la cual según Facebook Inc (2021), “Es una biblioteca de JavaScript declarativa, eficiente y flexible para crear interfaces de usuario. Permite componer interfaces de usuario complejas a partir de piezas de código pequeñas y aisladas llamadas componentes”. De esta manera, brinda gran potencial al proyecto, ya que brinda herramientas que facilitan el desarrollo y además incrementa el desempeño de la aplicación web, ya que su objetivo principal es la renderización únicamente de los componentes necesarios, de forma que economiza poder computacional.

Como se mencionó anteriormente, React JS está basado en la utilización de componentes, lo cual se refiere a cada uno de los elementos con los que se construye la aplicación, cada uno de estos es independiente y son reutilizables, lo cual es una más de las ventajas que brinda la utilización de una librería como esta, además, cada componente posee estados, lo cual se refiere a los valores que posee en cada momento del flujo de uso de la aplicación y dependiendo de estos valores los componentes se visualizan de una forma u otra, de modo que, es indispensable el correcto manejo de los estados de estos componentes para que el flujo de datos sea el deseado. Para cumplir satisfactoriamente con esto, se recurrió al uso de Redux; “Es una biblioteca de código abierto de JavaScript que permite manejar el estado de una aplicación utilizando el concepto de un único almacenamiento en el lado del cliente de la aplicación.” (Abramov, D, et al ,2021). Es decir, esta biblioteca proporciona una arquitectura para el manejo de datos y el estado del sistema, lo

cual garantiza un mejor control de estos y una mayor escalabilidad para el desarrollo.

Como cualquier aplicación de software, es indispensable la creación de una buena interfaz y experiencia de usuario, para esto se utilizó la tecnología llamada Bootstrap; “Framework front-end para los estilos o diseños, utilizado para desarrollar aplicaciones web y está enfocado a ser mobile-first, esto quiere decir que, la disposición de los elementos de la página tiene la capacidad de adaptarse a la pantalla del dispositivo utilizado por el usuario.” (Bootstrap, 2021). La utilización de este framework brinda las herramientas necesarias para cumplir satisfactoriamente con los requisitos de la empresa según sus estándares en cuanto a interfaz y experiencia de usuario.

Además, como parte de la implementación, se requiere del uso de tecnologías de almacenamiento en la nube, lo que permite un acceso de forma remota, para esto se utiliza Amazon Web Services; “Es la plataforma en la nube más adoptada y completa en el mundo, que ofrece más de 175 servicios integrales de centros de datos a nivel global.” (Amazon Web Services, Inc, 2021). Además, este mismo servicio de AWS brinda un gran catálogo de herramientas para el desarrollo del backend y su arquitectura en general.

Primeramente, se recurrió al uso de Amazon Route 53, el cual se conceptualiza como:

> Un servicio web de sistema de nombres de dominio (DNS) en la nube escalable y de alta disponibilidad. Está diseñado para brindar a los desarrolladores y empresas una forma extremadamente confiable y rentable de enrutar a los usuarios finales a las aplicaciones de Internet traduciendo nombres como www.example.com a las direcciones IP numéricas que las computadoras usan para conectarse entre sí. (Amazon Web Services, Inc, 2021).

Por lo tanto, Amazon Route 53, se encarga totalmente del DNS para conectar de forma eficaz las solicitudes o peticiones de los usuarios con la infraestructura

que se ejecuta en AWS, en este caso a los buckets de Amazon S3, lo cual según la descripción de Amazon Web Services, Inc (2021) “Es un servicio de almacenamiento altamente escalable para Internet. Posee una interfaz de servicios web que permite almacenar y recuperar cualquier cantidad de datos, en cualquier momento y desde cualquier parte de la web”. Por lo tanto, este servicio es utilizado para almacenar los recursos estáticos, en este caso el código de React JS, es decir el frontend, además el uso de esta tecnología se encuentra complementado con Cloudfront en AWS; “Es un servicio rápido de red de entrega de contenido que distribuye datos, vídeos, aplicaciones y API a clientes de todo el mundo de forma segura, con baja latencia, altas velocidades de transferencia y dentro de un entorno intuitivo para desarrolladores.” (Amazon Web Services, Inc, 2021). Este concepto hace referencia a un caché de la información almacenada en en S3 para tener un acceso más rápido al código del frontend.

Como parte de la implementación de People App se emplearon tecnologías de AWS enfocadas al desarrollo del backend, primeramente, cabe mencionar que se recurrió al uso de Amazon API Gateway;

> Es un servicio completamente administrado que facilita a los desarrolladores crear, publicar, mantener, monitorear y asegurar API a cualquier escala. Las API actúan como la "puerta de entrada" para que las aplicaciones accedan a los datos, la lógica empresarial o la funcionalidad de sus servicios de backend. Con API Gateway, puede crear API RESTful y API WebSocket que habilitan aplicaciones de comunicación bidireccional en tiempo real. API Gateway admite cargas de trabajo en contenedores y sin servidor, así como aplicaciones web. (Amazon Web Services, Inc, 2021)

Es decir, API Gateway maneja todas las tareas involucradas en aceptar y procesar todas las llamadas o peticiones API concurrentes, incluida la administración del tráfico, el soporte CORS, autorización y control de acceso, limitaciones, monitoreo y administración de versiones. Cabe destacar que, en este caso las peticiones al backend reciben una respuesta mediante la utilización de

expresiones Lambda; “Es un servicio informático que permite ejecutar código sin aprovisionarse y administrar servidores. Lambda ejecuta su código en una infraestructura informática de alta disponibilidad y realiza toda la administración de los recursos informáticos.” (Amazon Web Services, Inc, 2021). Por lo tanto, Lambda permite ejecutar código para prácticamente cualquier tipo de aplicación o servicio de backend, únicamente se le debe proporcionar el código en alguno de los lenguajes de programación que admite, en este caso se proporcionará código Python, el cual es un lenguaje multiparadigma y brinda un gran potencial para el desarrollo de backend en AWS.

Como parte de la funcionalidad del backend se emplearon tecnologías como Amazon Cognito; “...servicio que permite agregar el registro de usuario, el inicio de sesión y el control de acceso a sus aplicaciones web y móviles de forma rápida y sencilla. Escala a millones de usuarios y admite el inicio de sesión con proveedores de identidad social, como Apple, Facebook, Google, Amazon, entre otros.” (Amazon Web Services, Inc, 2021). Por lo tanto, se aprovecha mediante AWS de un servicio completo de autentificación, de forma que se obtiene un método más eficiente y seguro, lo cual repercute directamente en la calidad del sistema.

Por otra parte, como elemento fundamental de la arquitectura del software debe existir una base de datos, la cual brinde persistencia a los datos, para que estos pueden ser guardados y utilizados en cualquier momento, en este caso se utilizó PostgreSQL, el cual según The PostgreSQL Global Development Group (2021) “Es un poderoso sistema de base de datos relacional de objetos de código abierto que usa y extiende el lenguaje SQL combinado con muchas características que almacenan y escalan de manera segura las cargas de trabajo de datos más complicadas.” Cabe destacar que, esta base de datos es complementada con una aplicación externa que permite transferir la información de la empresa y utiliza tecnologías de AWS que incrementan su capacidad y eficiencia de uso como parte del sistema, es decir, en su interacción con el backend y por consiguiente, con el frontend. Primeramente, se recurrió al uso de Amazon RDS Proxy; “Es un proxy de base de datos de alta disponibilidad y completamente administrado para Amazon Relational Database Service (RDS) que hace que las aplicaciones sean más

escalables, más resistentes a fallas de la base de datos y más seguras.” (Amazon Web Services, Inc, 2021). Es decir, proporciona conexiones más eficientes entre el backend y la base de datos.

Como se mencionó anteriormente, las conexiones entre la base de datos y el backend son necesarias para cumplir con la arquitectura del software, y cabe destacar que para esto se utilizó SQLAlchemy, herramienta que brindó gran facilidad de desarrollo, ya que posibilitó que mediante código Python se realizarán las consultas necesarias a la base de datos, esta herramienta se describe como; “...un conjunto de herramientas Python SQL y Object Relational Mapper que brinda a los desarrolladores de aplicaciones todo el poder y la flexibilidad de SQL.” (SQLAlchemy, 2021).

Por último, es necesario mencionar que, en el proceso de implementación de un sistema de software, la seguridad de los datos es un elemento fundamental, y para garantizar esto, se recurrió al uso de AWS Key Management Service (KMS), el cual,” Es una herramienta que facilita la creación y administración de claves criptográficas y el control de su uso en una amplia gama de servicios de AWS y en sus aplicaciones. AWS KMS es un servicio seguro y resistente para proteger sus claves.” (Amazon Web Services, Inc, 2021). Analizando las tecnologías que se comentaron anteriormente, se puede observar cómo AWS cumple un papel fundamental en toda la arquitectura de la aplicación, velando por principios de alta escalabilidad, disponibilidad, seguridad y velocidad de respuesta.

3. Solución planteada

A continuación, se presenta un análisis de la propuesta de la solución que se va a brindar para el proyecto, la descripción de los involucrados o personas interesadas en este, además, se detalla el procedimiento metodológico que se llevará a cabo para el desarrollo de cada una de las tareas que forman parte de la propuesta y, por último, se muestra el cronograma de trabajo.

3.1. Propuesta

La solución propuesta plantea una aplicación web de una sola página basada en una arquitectura orientada a microservicios con la utilización de expresiones Lambda en AWS desarrollado con Python con un rol de servidor backend, una base de datos relacional de PostgreSQL, distintas tecnologías de AWS para el manejo, control de infraestructura y almacenamiento y un frontend desarrollado en React JS que permite la visualización de la información almacenada en el mismo y está basado en el lenguaje de programación Javascript.

3.2. 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 de este 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% de los módulos mencionados anteriormente

Tabla 1: Involucrados

Nombre	Rol	Descripción
Andrés Hernández	Cliente Final	Es quien solicita el proyecto
Alejandra Parra	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	Líder Técnico	Es quien lidera el proyecto a través de todas sus etapas desde una perspectiva técnica.
Fabian Alfaro Carina Elizondo	Desarrolladores Web	Son quienes trabajan en el desarrollo del proyecto durante todas sus etapas.

Fuente: elaboración propia

3.3. Procedimiento metodológico

Como parte del procedimiento metodológico se emplea el marco de referencia de la empresa el cual se centra en la metodología ágil de Scrum. Por lo tanto, se plantea una reunión (Sprint Planning) cada 2 semanas para definir las tareas que se llevarán a cabo en un periodo de 2 semanas (Sprint). Estas tareas suponen las metas para cumplir con los objetivos del proyecto.

Durante este periodo se realiza una reunión (Daily Scrum) diaria para llevar un seguimiento del trabajo realizado. Cuando las tareas son finalizadas deben enviarse a un proceso llamado Code Review, en este profesionales en la materia revisan el código implementado y determinan si el código es adecuado o no, en caso de que no, se indican las correcciones que se deben de llevar a cabo y se implementa lo necesario hasta que lo consideren acertado, seguidamente se procede a someter el código a un proceso de testeo para asegurar la calidad del código implementado y posteriormente la tarea es revisada por uno de los encargados del equipo Scrum (Product Owner) para confirmar si satisface las necesidades del cliente, por lo tanto, estas aprobaciones mencionadas anteriormente suponen los indicadores de que la meta se ha cumplido satisfactoriamente.

Para finalizar, el último día de este período establecido en el que se debe llevar a cabo cada tarea se realiza una reunión (Sprint Review) para en conjunto con el equipo de trabajo ver los resultados obtenidos, además se complementa con otra reunión (Sprint Retrospective) para determinar asuntos positivos y negativos como equipo de trabajo para el desarrollo de las tareas.

3.4. Cronograma de trabajo

A continuación, se describe el cronograma preliminar del proyecto, incluyendo un listado de tareas generales, con sus respectivas fechas de inicio y finalización, las cuales representan el plazo de tiempo que comprende cada uno de los sprints.

Tabla 2: Cronograma de trabajo

Tarea	Fecha de inicio y finalización
Configuración local y adaptación a los procesos de la compañía y la aplicación.	26/07/2021 - 30/07/2021 **(Sprint 0)**
En la sección de **PM Dashboard** se debe obtener el listado de personas a cargo por el PM al que corresponde la sesión, su respectiva información que corresponda y crear una tabla con estos datos para mostrarlos.	02/08/2021 - 13/08/2021 **(Sprint 1)**
En la sección de **Happiness Dashboard** se debe obtener el listado de todos los colaboradores de la empresa, su respectiva información que corresponda y crear una tabla con estos datos para mostrarlos. En la sección **Happiness Dashboard** obtener, calcular y mostrar datos estadísticos gráficamente sobre el estado de ánimo de los colaboradores de la empresa.	16/08/2021 - 27/08/2021 **(Sprint 2)**
En la sección de **PM Dashboard** el sistema debe permitir en una sección ver la información detallada de cada colaborador, su portafolio, curriculum vitae, certificaciones, habilidades e historial de estado de ánimo. En la sección **Happiness Dashboard** el sistema debe permitir en una sección ver información general y el historial de estado de ánimo de cada	30/08/2021 - 10/09/2021 **(Sprint 3)**

colaborador en un listado y en un gráfico de tipo línea de tiempo según sus reportes en el formulario de felicidad.	
Informe #1 de práctica (3 Sprints)	17/09/2021
En la sección de My Dashboard se debe crear y mostrar en pantalla un formulario (Happiness Form) para evaluar el estado de ánimo de los colaboradores, para obtener los datos necesarios para la implementación de **Happiness Dashboard**.	13/09/2021 - 24/09/2021 **(Sprint 4)**
Se deben obtener, analizar y guardar los resultados del formulario (Happiness Form) cuando este es contestado por un colaborador, para obtener los datos necesarios para su utilización en **Happiness Dashboard**.	27/09/2021 - 08/10/2021 **(Sprint 5)**
Informe #2 de práctica (2 sprints)	15/10/2021
En la sección de **Happiness Dashboard** crear filtros de búsqueda para los resultados de la tabla y los gráficos.	11/10/2021 - 22/09/2021 **(Sprint 6)**
Informe #3 de práctica	12/11/2021

Fuente: elaboración propia

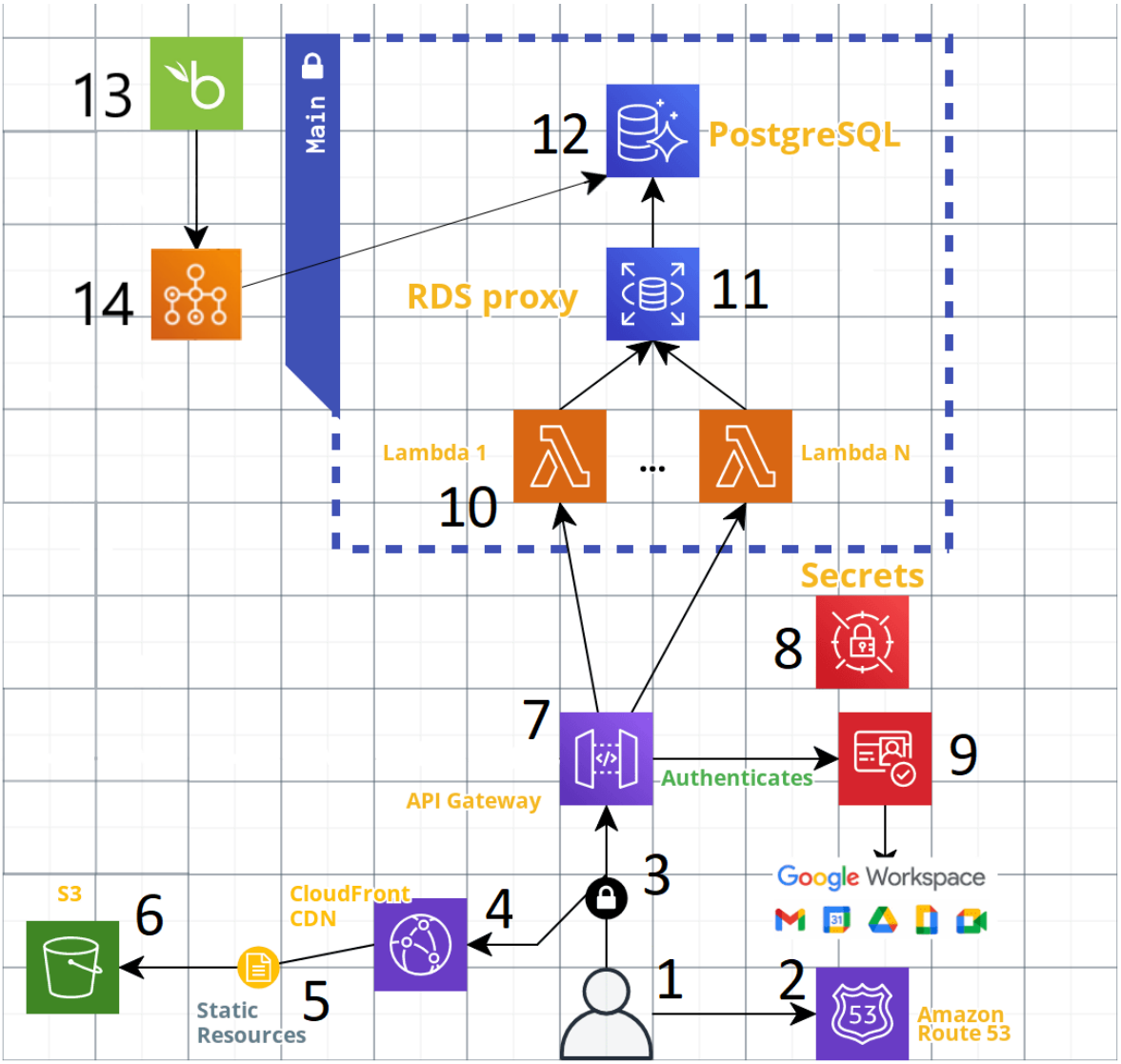
4. Notas de despliegue del entregable

A continuación, se realiza una descripción de la solución planteada tomando en consideración diversos recursos como diagramas de la arquitectura conceptual de la solución enfocada en frontend y backend, la respectiva estructura que posee la base de datos utilizada y, por último, se describe detalladamente cada una de las tareas implementadas para tener una visión más clara de los resultados obtenidos.

4.1. Diseño de la plataforma de software

A continuación, se presenta el diagrama de la arquitectura conceptual de la solución enfocada en frontend y backend, de manera que se detalla el uso de cada una de las tecnologías mediante su logo y nombre, asimismo, se refleja la interacción que hay entre estas para obtener la solución planteada.

Figura 1: Arquitectura conceptual de la solución



Fuente: Elaboración propia

A continuación, se explica la representación y el flujo de interacción que posee cada uno de los objetos del diagrama mostrado anteriormente, se analiza según la numeración que tiene cada nodo de este.

1. **Cliente:** Navegador utilizado por el usuario desde cualquier dispositivo para hacer uso People App. Desde este dispositivo se realizan las peticiones a la aplicación.

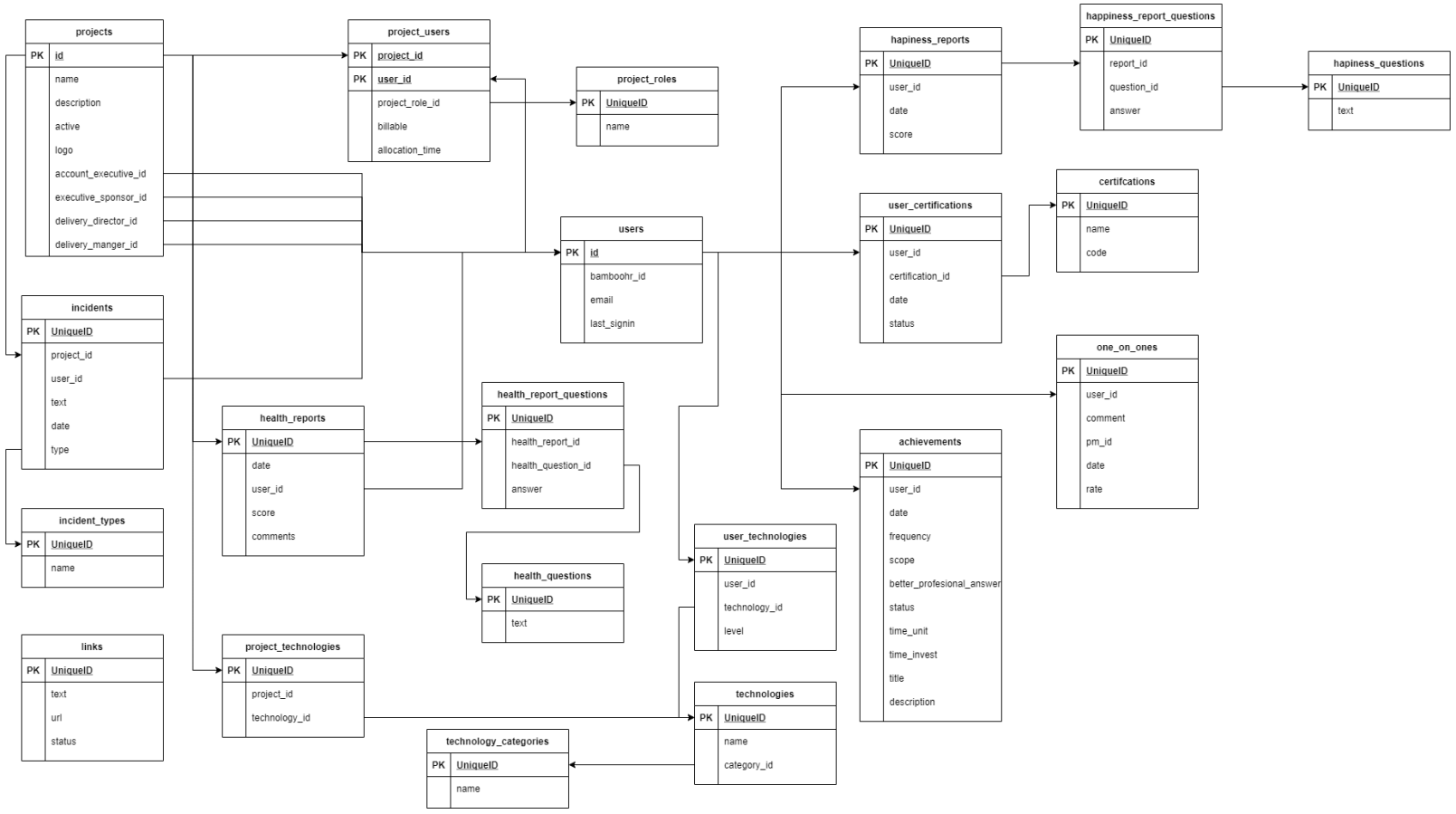
2. **Amazon Route 53:** Se encarga del DNS para la conexión entre el cliente y la aplicación. Es decir, cada una de las peticiones realizadas por el cliente son procesadas por este para que se lleve a cabo el direccionamiento de estas.
3. **Candado:** Representa la necesidad de una autenticación en la aplicación de parte del usuario para que las peticiones a esta se puedan llevar a cabo. Esta autentificación otorga un token al cliente para que pueda realizar peticiones válidas al sistema.
4. **Amazon CloudFront:** Brinda respuestas a las peticiones realizadas por el cliente al frontend. Funciona como caché de la información almacenada en los buckets de Amazon S3 (6) para un acceso más rápido.
5. **Static Resources:** Representan los recursos estáticos, en este caso, el código del frontend que es requerido por el cliente. Estos recursos son almacenados en Amazon S3 (6) y transferidos a Amazon CloudFront (5).
6. **Amazon S3:** Almacena los recursos estáticos y además, se los transfiere a CloudFront para que este se encargue de brindarlos al cliente según las peticiones realizadas por éste.
7. **Amazon API Gateway:** Se encarga de administrar todas las peticiones realizadas al backend de parte del cliente. Es decir, distribuye las peticiones en cada una de las expresiones Lambda (10) según corresponda.
8. **AWS Key Management Service:** Es un administrador de secretos del backend, este se encarga de proteger datos confidenciales o sensibles para la empresa utilizados en el desarrollo de la aplicación, por ejemplo, contraseña de acceso a la base de datos.
9. **Amazon Cognito:** Como parte de las peticiones que puede realizar el cliente al backend se encuentra la autentificación al sistema, Amazon Cognito se enfoca en la administración de todo lo que conlleva este servicio. En este caso, la autenticación se realiza mediante un inicio de sesión único en Google.
10. **Expresiones Lambda:** Cada petición realizada al backend por el cliente es asignada a una expresión lambda, las cuales se encargan de realizar los

procesos necesarios para resolver la petición del cliente. En la mayoría de los casos, realizan una conexión con la base de datos PostgreSQL (11) para interactuar con la información almacenada en esta.

11. **Amazon RDS Proxy:** Funciona como un puente entre la conexión del backend y la base de datos, esto brinda mayor seguridad y velocidad entre el paso de datos de los involucrados. En concreto actúa como intermediario entre las expresiones Lambda (10) y PostgreSQL (12)
12. **PostgreSQL:** Cumple el rol de base de datos, es decir, almacena la información que necesita persistir dentro de ella y se la brinda al backend, en específico a las expresiones Lambda (10) que la soliciten.
13. **BambooHR:** Es una aplicación externa que posee almacenada la información de los colaboradores de la empresa y se sincroniza con la base de datos (12) de People App para transferir esta información.
14. **Syncs information:** Representa el proceso de sincronización entre los datos almacenados en BambooHR (13) y la base de datos de PostgreSQL (12).

A continuación, se presenta el diagrama de la base de datos la cual fue implementada en PostgreSQL mediante la utilización de migraciones con la herramienta de SQLAlchemy y bajo un modelo incremental a lo largo del desarrollo del proyecto según se presentaban las necesidades.

Figura 2: Diseño de base de datos



Fuente: Elaboración propia

4.2. Tareas completadas

Para el desarrollo de este proyecto se utilizará la metodología de trabajo llamada Scrum, la cual se compone principalmente por Sprints, por lo tanto, como método de documentación del trabajo realizado, en este apartado se utilizará el modelo de Release Notes utilizado por la empresa, el cual consiste en describir brevemente las tareas o las historias de usuario que se deben llevar a cabo en cada Sprint, los detalles de implementación que conlleva, el resultado obtenido con un modelo gráfico y por último, un caso de prueba, que describe un camino de usabilidad de la aplicación para probar la funcionalidad de la tarea implementada.

Tarea 1: Como encargado de personal quiero ver los GAPsters a mi cargo.

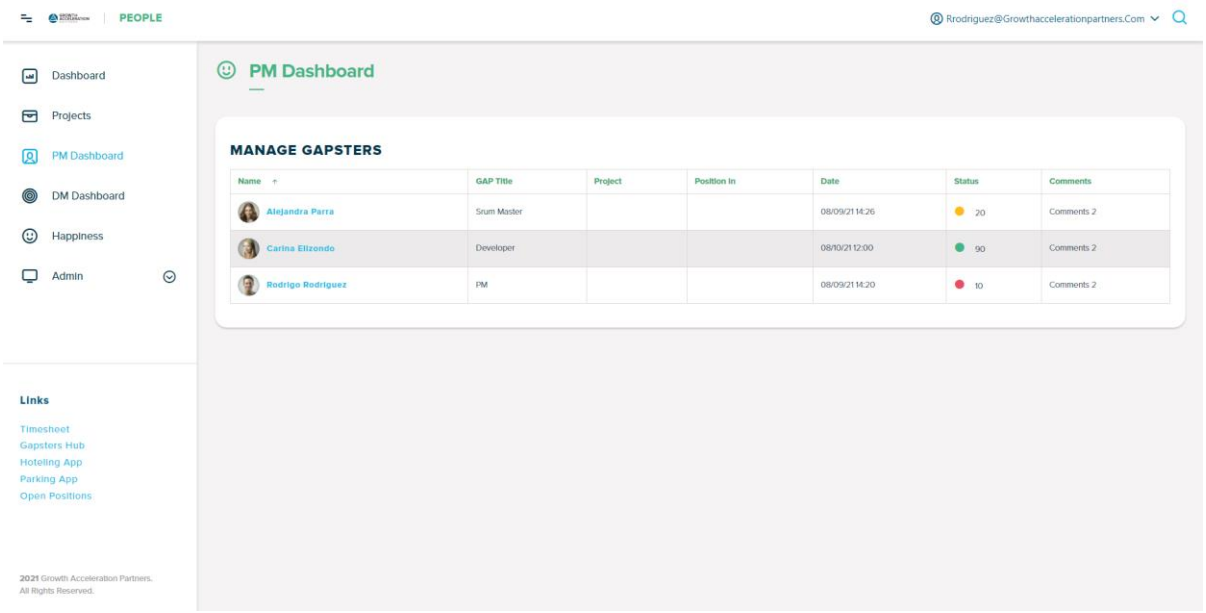
Como encargado de personal, quiero poder iniciar sesión en la herramienta People App y acceder al panel de control llamado PM Dashboard para poder ver un listado de los GAPsters a mi cargo.

Detalles de implementación:

1. Presentar una lista de todos los GAPsters asignados a un encargado de personal, con sus respectivos datos como; nombre, foto, puesto, posición dentro del proyecto asignado, último estado de felicidad, la fecha de su reporte y comentarios adicionales sobre este.
2. Funcionalidad de ordenación de columnas ascendente o descendente.
3. Un encargado de personal sólo tendrá un máximo de 8 GAPsters asignados, por lo que la paginación y el filtrado son innecesarios.
4. El nombre de un GAPster debe reaccionar y comportarse como un enlace, pero no se espera que se vincule a ningún lugar en este momento.

Resultado de implementación:

Figura 3: PM Dashboard



Fuente: People App

Caso de prueba:

1. Loguearme en la aplicación
2. En el menú de la izquierda, dar clic al botón de "PM Dashboard"
3. Luego de cargar, la página muestra una tabla con las siguientes columnas: Name, GAP Title, Project, Position In, Date, Status, Comments
4. Al dar clic en la cabecera de cualquier columna la información se ordena ascendentemente según la columna cliqueada, con un segundo clic sobre la misma se ordena de manera descendente y al tercer clic vuelve al orden predeterminado, el cual es ascendente según la columna Name.

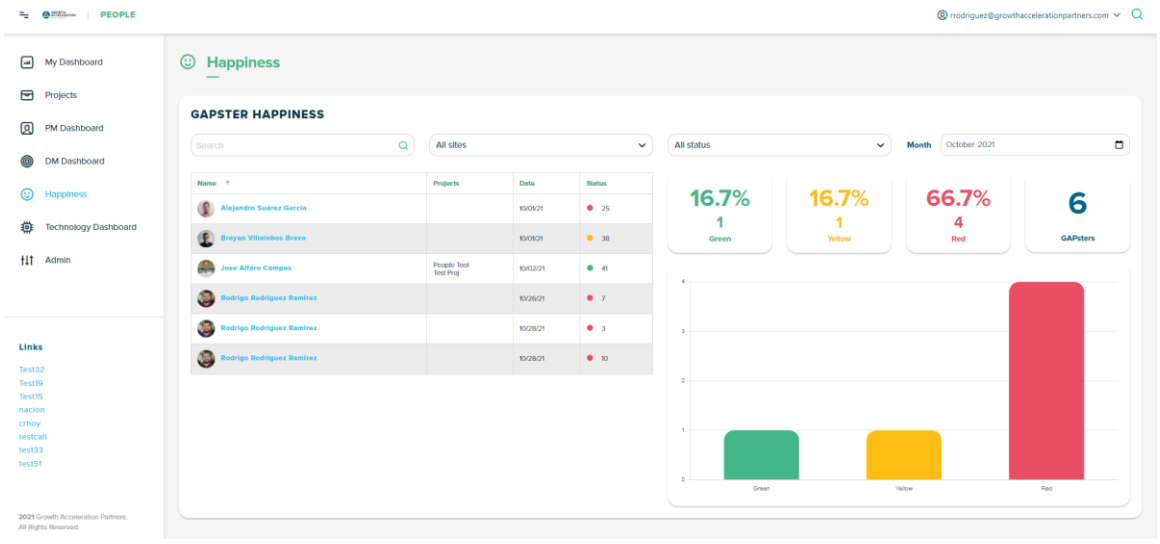
Tarea 2: Como encargado de Felicidad, quiero poder ver los resultados de felicidad de todos los GAPsters activos

Como encargado de felicidad quiero poder iniciar sesión en la herramienta People App y acceder al panel de control llamado Happiness Dashboard para ver los resultados de felicidad para todos los GAPsters activos.

Detalles de implementación:

1. Presentar una lista de todos los GAPsters activos con sus respectivos datos como nombre, foto, proyecto asignado, último estado de felicidad, la fecha de su reporte y comentarios adicionales sobre este.
2. Funcionalidad de ordenación de columnas ascendente o descendente.
3. Presentar opciones de filtrado de resultados como; búsqueda por nombre mediante entrada de texto libre, selector por país sede, selector por estado de felicidad y selector por mes.
4. Presentar indicadores gráficos para representar el porcentaje de GAPsters que posee su estado de felicidad en color verde, amarillo y rojo, así como el número total de GAPsters activos.

Figura 4: Happiness Dashboard



Fuente: People App

Caso de prueba:

1. Loguearme en la aplicación
2. En el menú de la izquierda, dar clic al botón de "Happiness"
3. Luego de cargar, la página muestra una tabla con las siguientes columnas: Name, GAP Title, Project, Date, Status, Comments. Al lado de esta tabla se muestran datos estadísticos con un color según el status que representa, un gráfico de barras y el número total de GAPsters activos y en la sección superior de la pestaña cuatro diferentes tipos de filtro para la información mostrada.
4. Al dar clic en la cabecera de cualquier columna la información se ordena ascendentemente según la columna cliqueada, con un segundo clic sobre la misma se ordena de manera descendente y al tercer clic vuelve al orden predeterminado, el cual es ascendente según la columna Name.

Tarea 3: Como encargado de personal, quiero poder acceder al perfil e historial de felicidad de cualquiera de mis GAPsters asignados

Como encargado de personal, quiero acceder al panel de control llamado PM Dashboard y poder seleccionar a cualquiera de mi lista de GAPsters asignados para acceder a su perfil, en el cual se puede ver una sección con información detallada de cada colaborador, su portafolio, currículum vitae, certificaciones, habilidades e historial de estado de ánimo.

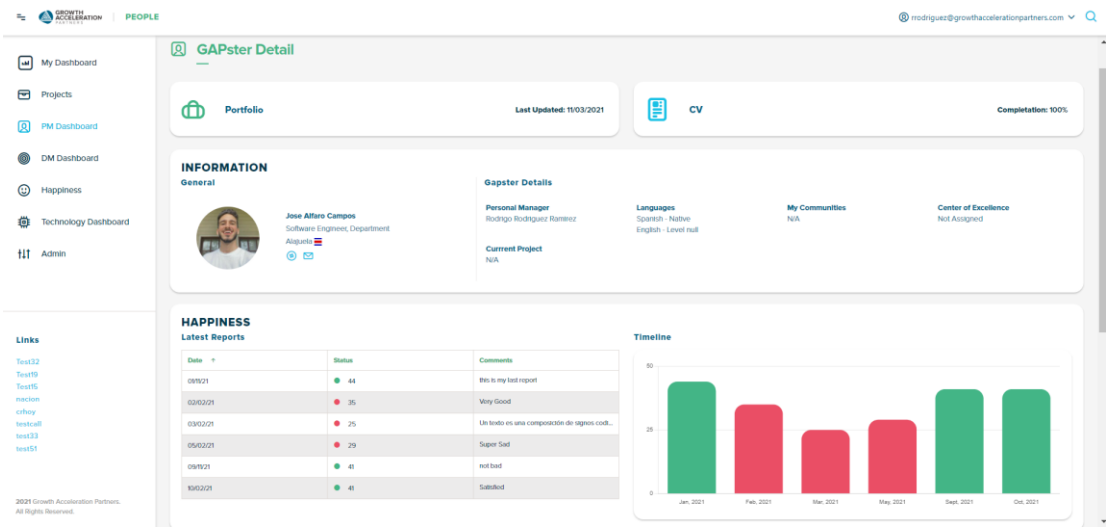
Detalles de implementación:

1. Presentar botones de acceso al portafolio y al currículum vitae del empleado seleccionado.
2. Mostrar recuadro con información general del empleado seleccionado como nombre, foto, puesto, país, ubicación, contacto por correo y slack, pm a cargo, idiomas, comunidades a las que pertenece dentro de la empresa, proyecto actual y premios recibidos.

3. Presentar recuadro con el historial de felicidad del último año del empleado seleccionado, en un listado y de forma gráfica.
4. Designar recuadros con el listado de certificaciones y habilidades que posee el empleado seleccionado.

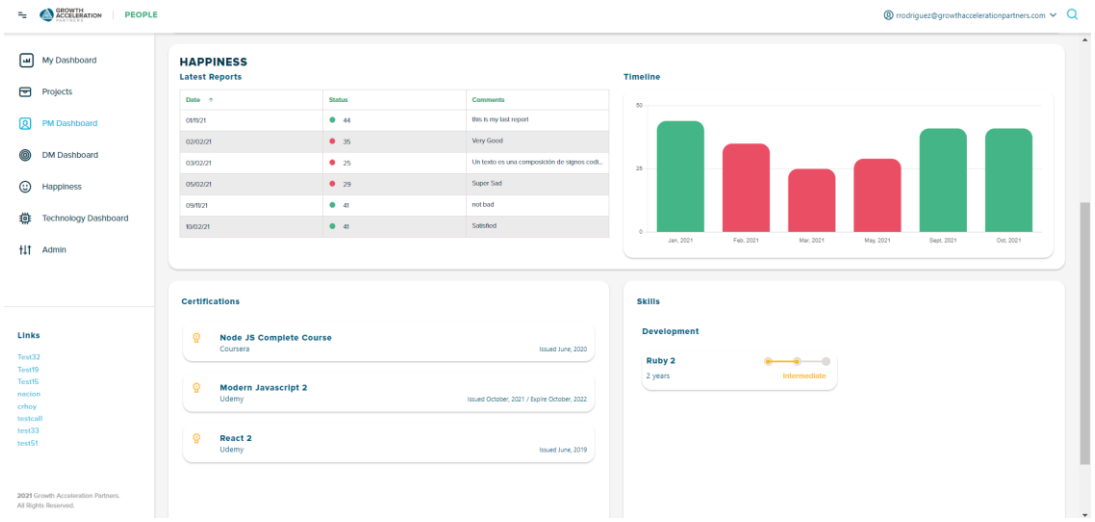
Resultado de implementación:

Figura 5: PM Dashboard GAPster detail (Parte superior)



Fuente: People App

Figura 6: PM Dashboard GAPster detail (Parte inferior)



Fuente: People App

Caso de prueba:

1. Loguearme en la aplicación
2. En el menú de la izquierda, dar clic al botón de "PM Dashboard"
3. Luego de cargar, la página muestra una tabla con las siguientes columnas: Name, GAP Title, Project, Position In, Date, Status, Comments
4. Dar clic sobre cualquiera de los elementos en la columna “Name”, de forma que el nombre cliqueado representa el GAPster seleccionado para ver su perfil.
5. Luego de cargar, la página muestra en la parte superior dos botones de acceso al portafolio y al currículum vitae del empleado seleccionado, cada uno de estos al ser cliqueado lo redirige a la sección que corresponde, en la parte media muestra un recuadro con información general como nombre, foto, puesto, país, ubicación, contacto por correo y slack, nombre de su encargado, idiomas, comunidades a las que pertenece dentro de la empresa, proyecto actual y premios recibidos, además, un recuadro con el historial de felicidad del último año, en un listado con funcionalidad de ordenación y con un gráfico de líneas y en su parte más baja muestra un recuadro al lado izquierdo con el listado de certificaciones y al lado derecho otro con las habilidades que posee el empleado seleccionado.

Tarea 4: Como encargado de felicidad, quiero poder acceder al perfil e historial de felicidad de cualquier GAPster

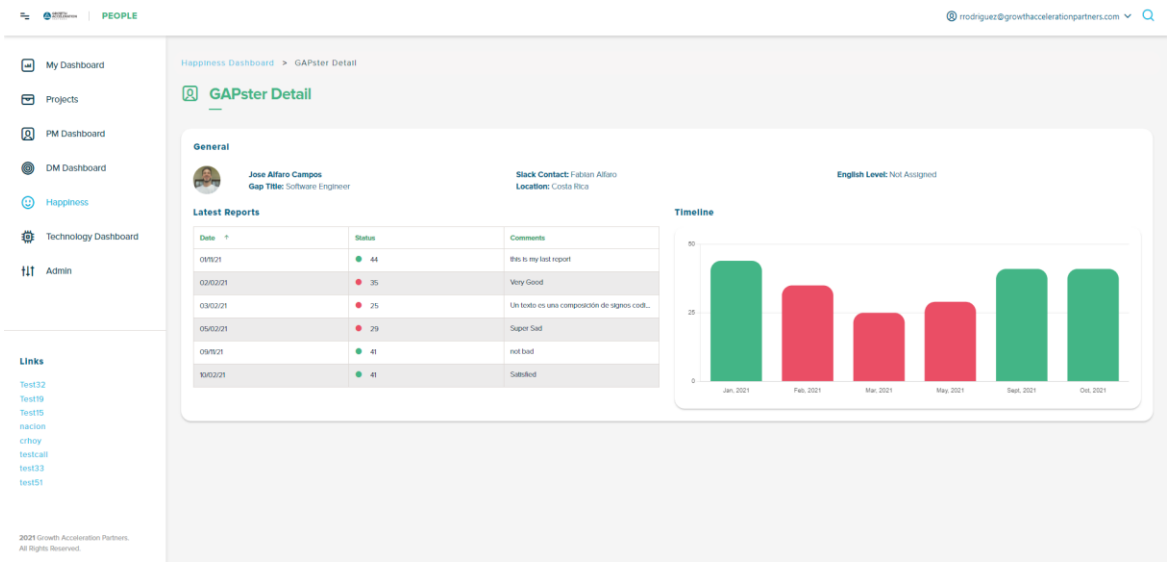
Como encargado de felicidad, quiero acceder al panel de control llamado Happiness Dashboard y poder seleccionar a cualquiera de mi lista de GAPsters asignados para acceder a su perfil, en el cual se puede ver una sección con información más detallada de su historial de estado de ánimo.

Detalles de implementación:

1. Mostrar información general del empleado seleccionado como nombre, foto, puesto, país, contacto de Slack y nivel de inglés.
2. Presentar el historial de felicidad del último año del empleado seleccionado, en un listado y de forma gráfica.

Resultado de implementación:

Figura 7: Happiness Dashboard GAPster detail



Fuente: People App

Caso de prueba:

1. Loguearme en la aplicación
2. En el menú de la izquierda, dar clic al botón de "Happiness Dashboard"
3. Luego de cargar, la página muestra una tabla con las siguientes columnas: Name, GAP Title, Project, Date, Status, Comments
4. Dar clic sobre cualquiera de los elementos en la columna “Name”, de forma que el nombre cliqueado representa el GAPster seleccionado para ver su perfil.
5. Luego de cargar, la página muestra un recuadro el cual en su parte superior posee información general del empleado seleccionado como nombre, foto,

puesto, país, contacto de slack, nivel de inglés, y en la parte inferior muestra el historial de felicidad del último año, al lado izquierdo en un listado y al lado derecho con un gráfico de líneas.

Tarea 5: Como GAPster quiero poder completar y enviar el formulario de felicidad

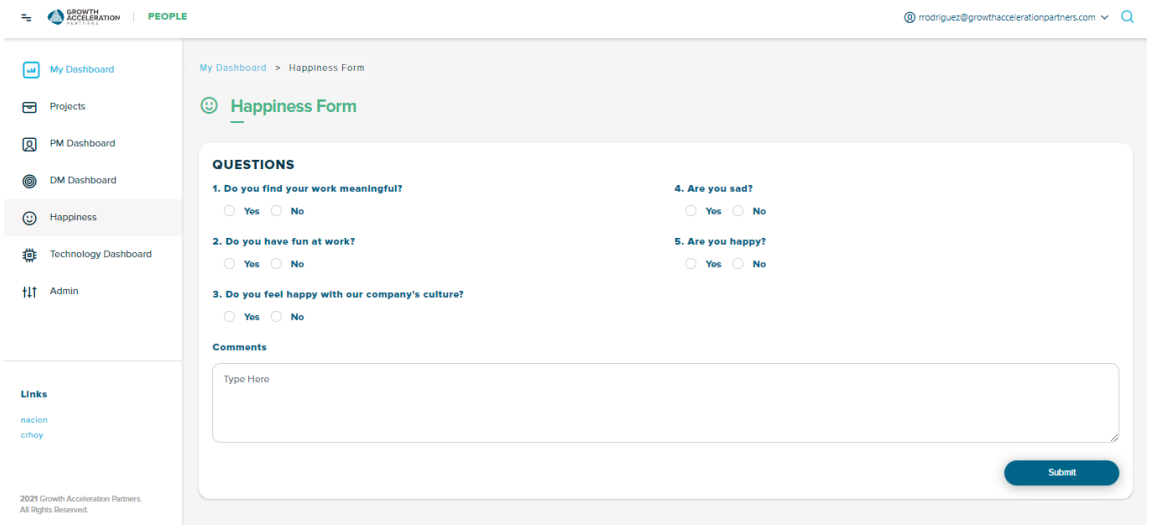
Como GAPster, quiero acceder al formulario de felicidad desde el panel principal (My dashboard) para poder verlo, completarlo y enviarlo cada vez que se requiera.

Detalles de implementación:

1. Al hacer clic en el botón de felicidad, el formulario de felicidad debe cargarse desde el backend y renderizarse
2. El formulario Felicidad debe ser una colección de preguntas de sí o no y 1 pregunta de cierre de texto libre
3. Cuando se guardó con éxito, el sistema debe comunicar que el formulario se guardó con éxito y volver al panel de control principal.
4. Cuando se haya guardado el formulario, el formulario de felicidad debe estar en gris y comunicar que se ha enviado el formulario actual, así como la fecha.
5. El botón solo debe estar activo si hay un formulario disponible y está pendiente de cumplimentar.
6. Si hay un error al guardar o intentar renderizar el formulario, el sistema debe volver al tablero y el botón de felicidad debe permanecer activo
7. Los formularios deben tener soporte activo-inactivo según la ventana de fecha de inicio y finalización.

Resultado de implementación:

Figura 8: Happiness Form



Fuente: People App

Caso de prueba:

1. Loguearme en la aplicación
2. En el menú de la izquierda, dar clic al botón de "My Dashboard"
3. Luego de cargar, la página muestra en la parte superior izquierda un botón de “Happiness Form”, al cual debe dar clic para poder acceder al formulario. Únicamente puede acceder si se encuentra dentro del periodo de tiempo para completarlo y si no lo ha contestado antes, en caso de no cumplir con alguna de estas condiciones el botón va a estar deshabilitado.
4. Luego de cargar, la página muestra un formulario de 2 columnas con una colección de preguntas de si o no y 1 pregunta de cierre de texto libre.
5. Dar clic sobre una de las opciones de cada pregunta del formulario y digitar en el campo de texto algún comentario adicional si así lo desea.
6. Cuando haya completado de contestar todas las preguntas del formulario, dar clic al botón “Submit” para enviar las respuestas de este.
7. El sistema le debe notificar mediante una alerta que el formulario fue enviado satisfactoriamente.

Tarea 6: Como encargado de personal o encargado de felicidad, quiero ver las respuestas del formulario de felicidad para cualquiera de los últimos 6 períodos en un GAPster seleccionado

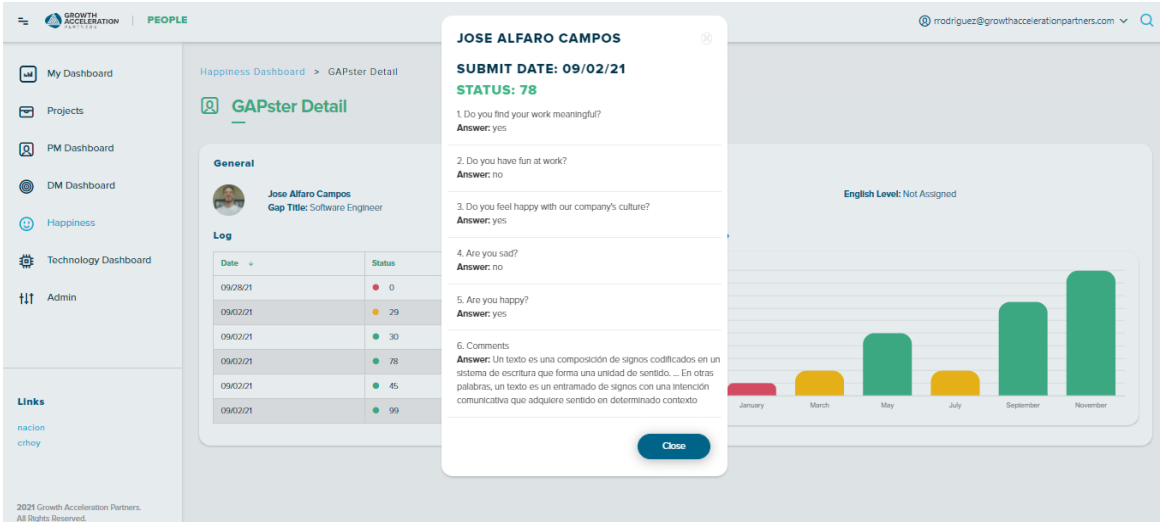
Como encargado de personal o encargado de felicidad, quiero ver las respuestas del formulario de felicidad con sus respectivas preguntas para cualquiera de los últimos 6 períodos reflejados en el historial de felicidad de un GAPster seleccionado.

Detalles de implementación:

1. Para ver las respuestas de alguno de los formularios de felicidad que fue completado en algún periodo se puede hacer desde la sección de GAPster Detail con la tabla correspondiente seleccionando una fila o desde el gráfico haciendo clic en una barra.
2. Se debe mostrar un modal de solo lectura con las preguntas y sus respectivas respuestas, además, de la fecha en la que fue completado y el puntaje obtenido.

Resultado de implementación:

Figura 9: GAPster Details Happiness Form



Fuente: People App

Caso de prueba:

1. Loguearme en la aplicación
2. En el menú de la izquierda, dar clic al botón de "Happiness Dashboard" en el caso de ser un encargado de felicidad o dar clic al botón de “PM Dashboard” en el caso de ser un encargado de personal.
3. Luego de cargar, la página muestra una tabla con las siguientes columnas: Name, GAP Title, Project, Date, Status, Comments
4. Dar clic sobre cualquiera de los elementos en la columna “Name”, de forma que el nombre cliqueado representa el GAPster seleccionado para ver su perfil.
5. Luego de cargar, la página muestra un recuadro que muestra el historial de felicidad del último año del GAPster seleccionado, al lado izquierdo en una tabla con formato de lista y al lado derecho con un gráfico de barras.
6. Al dar clic en cualquiera de las filas de la tabla o sobre las barras del gráfico se despliega un modal que corresponde a las preguntas y respuestas de ese reporte de felicidad el cual fue completado en el periodo en el que se muestre.

4.3. Evaluación

En el apartado anterior se describen las tareas completadas, con sus detalles de implementación, resultados y se expone el caso de prueba necesario para su proceso de evaluación y aseguramiento de la calidad, el cual involucró un proceso ágil adaptado a la metodología de trabajo utilizada y al corto periodo de tiempo que se dispone para el desarrollo de este, esto por decisión de la empresa debido a la priorización de necesidades que tenía el cliente.

En este proceso, cada vez que una tarea es finalizada debe someterse a su revisión, primeramente se debe presentar un demo a los otros desarrolladores del equipo y al Product Owner, el cual consiste en mostrar la funcionalidad de dicha tarea integrada a la aplicación siendo guiado por el caso de prueba establecido, de manera que se evalúa que cumpla con cada uno de los puntos descritos en los detalles de implementación y que visualmente satisfaga con los requisitos de la empresa según sus estándares en cuanto a interfaz y experiencia de usuario, lo cual está establecido en un prototipo diseñado antes de que se iniciara el proceso de desarrollo. En caso de que se mencionen comentarios de mejora, se procede a su correspondiente implementación o corrección para volver a mostrar el demo con sus respectivos cambios.

Cuando la tarea es aceptada en esa etapa se procede a la siguiente, en esta debe subir el código de la tarea al repositorio de GitLab del proyecto, en el cual es revisado y evaluado por un grupo de desarrolladores expertos en las tecnologías utilizadas, de manera que colocan comentarios y consejos para optimizar el código. En el momento que la evaluación es finalizada se deben realizar los cambios que correspondan y se suben al mismo repositorio, con lo cual se da por finalizado el proceso de calidad del software.

5. Conclusiones

1. Se determinó que para la definición de requerimientos e historias de usuario la utilización de Scrum como metodología de trabajo y el modelo de Release Notes para la descripción de las tareas influyó de manera positiva en la implementación del proyecto, ya que permitió que se desarrollarán a lo largo de este como incrementos lo cual por su naturaleza era necesario. Los requerimientos de este no estaban completamente definidos al iniciar y el cliente tampoco los tenía claros, lo cual posibilitó al equipo y al cliente definirlos o modificarlos según su conveniencia basándose en el prototipo ya diseñado con anticipación o lo que el cliente fuera descubriendo que le brindaría mayor utilidad a su producto, esto propició una mayor organización de las tareas según un orden de prioridad.

2. Se planteó un diagrama que incide favorablemente en el desarrollo de la aplicación pues representa la arquitectura de la solución planteada desde la perspectiva de las tecnologías a utilizar, el flujo de datos entre estas y la estructura que conlleva la base de datos. Esta estructuración en las etapas iniciales del proyecto supone ser una base para el correcto desarrollo de este, evitando correcciones innecesarias al final del proceso de desarrollo, lo cual conlleva un mayor gasto de producción. Además, asegura en mayor medida que se satisfagan las necesidades del cliente con respecto a la calidad del producto.

3. Se implementaron los módulos PM Dashboard y Happiness Dashboard satisfactoriamente, tanto su estructura de frontend como de backend, según los requerimientos y las historias de usuario descritas, lo cual garantiza que las tecnologías seleccionadas para el desarrollo son una opción viable. Además, para su diseño visual fue utilizado como guía un prototipo diseñado al inicio del proyecto por otros colaboradores de la empresa, sin embargo,

por las observaciones y comentarios realizados por el cliente durante las reuniones el aspecto visual no es su totalidad una réplica del prototipo, si no que se adaptó a las necesidades que el cliente percibió durante el desarrollo de la aplicación. Cabe destacar que los resultados obtenidos en esta implementación se evidencian visualmente en el apartado de tareas completadas y se expone un caso de prueba para comprobar su funcionalidad.

Además, es importante mencionar que la implementación de estos módulos representa un incremento para la aplicación de People App y supone una fortaleza tecnológica para los procesos internos de la empresa en el manejo de los empleados con respecto a sus proyectos y estado de ánimo dentro de estos.

4. Los resultados obtenidos en el proceso de evaluación y aseguramiento de la calidad de los módulos implementados son positivos, ya que se puede evidenciar un resultado de implementación de cada tarea, lo cual implica haber superado satisfactoriamente todas las pruebas correspondientes a este proceso para que se integrara a la aplicación en producción. Sin embargo, debido a la metodología de trabajo utilizada y al corto periodo de tiempo que se disponía para el desarrollo del proyecto, la evaluación del software se considera limitada, ya que no se implementaron pruebas unitarias o de integración las cuales son pertinentes a un proyecto de esta magnitud, si no que en su mayoría fueron pruebas de optimización de código y visualización.

6. Recomendaciones

1. Plantear una metodología de trabajo ágil para el desarrollo del proyecto brinda ventajas en cuanto a la velocidad de los resultados y la flexibilidad a los cambios, sin embargo, este marco podría optimizarse si se realiza un levantamiento de requerimientos general anticipado con el objetivo de conceptualizar el proyecto y no depender tanto de reuniones con el equipo de trabajo y el cliente, para tener un listado de tareas más robusto y evitar que los desarrolladores se queden con tiempos libres en los cuales no se produce. Además, el hecho de que no se conociera anticipadamente una idea general de la aplicación puede provocar una extensión de tiempo de desarrollo descontrolada.

2. Se recomienda en etapas tempranas de desarrollo plantear diagramas específicos para la estructura del frontend, lo que puede establecer en el caso de la utilización de React con Redux; un modelo de clases, los componentes necesarios, el flujo de datos entre estos y la estructura más adecuada para el modelo de Redux que permita manejar estados globales en un solo lugar de la aplicación. Asimismo, sería favorable la creación de diagramas más detallados para plantear la estructura del backend y el flujo de datos entre este, la base de datos y el frontend.

3. La implementación de los módulos PM Dashboard y Happiness Dashboard fue exitosa, sin embargo, podría mejorarse implementando notificaciones o alertas para que el usuario tenga retroalimentación del porqué ha sucedido algo en el sistema, de manera que se obtenga una experiencia de usuario más satisfactoria y guiada. Por otra parte, la implementación de estilo y diseño que tuvieron estos módulos fue mediante librerías como Bootstrap, por lo tanto, el uso de CSS puro podría otorgar una mayor optimización en el código, una menor dependencia a librerías externas y menor cantidad de errores por actualizaciones en estas mismas.

4. Se considera favorable para la calidad del producto implementar un entorno de testing más robusto, en el cual se ejecuten pruebas unitarias y pruebas de integración, de manera que se asegura en gran medida el correcto funcionamiento de cada componente y de su integración con los demás, lo cual puede evitar la ejecución de errores del sistema en etapa de producción. Por lo tanto, se recomienda un porcentaje mínimo de cobertura de código en este tipo de pruebas de un 80% según lo mencionado por expertos en la materia de la misma empresa.

7. Bibliografía

Abramov, D, et al. (2021). *Redux Essentials, Part 1: Redux Overview and Concepts.* https://redux.js.org/tutorials/essentials/part-1-overview-concepts

Amazon Web Services, Inc. (2021). *Amazon API Gateway.* https://aws.amazon.com/api-gateway/

Amazon Web Services, Inc. (2021). *Amazon CloudFront.* https://aws.amazon.com/es/cloudfront/

Amazon Web Services, Inc. (2021). *Amazon Cognito.* https://aws.amazon.com/cognito/

Amazon Web Services, Inc. (2021). *Amazon RDS Proxy.* https://aws.amazon.com/rds/proxy/

Amazon Web Services, Inc. (2021). *Amazon Route 53.* https://aws.amazon.com/route53/

Amazon Web Services, Inc. (2021). *Amazon S3*. https://aws.amazon.com/es/s3/

Amazon Web Services, Inc. (2021). *AWS Key Management Service (KMS).* https://aws.amazon.com/kms/

Amazon Web Services, Inc. (2021). *What is AWS?* https://aws.amazon.com/what-is-aws/

Amazon Web Services, Inc. (2021). *What is AWS Lambda?* https://docs.aws.amazon.com/lambda/latest/dg/welcome.html

Bootstrap. (2021). *Bootstrap Docs Introduction.* https://getbootstrap.com/docs/5.1/getting-started/introduction/

Facebook Inc. (2021). *Tutorial: Intro to React.* https://reactjs.org/tutorial/tutorial.html

Mozilla. (2020). *What is JavaScript? Mozilla Web Docs.* https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript

SQLAlchemy. (2021). *The Python SQL Toolkit and Object Relational Mapper.* https://www.sqlalchemy.org/

The PostgreSQL Global Development Group. (2021). *What is PostgreSQL?.* https://www.postgresql.org/about/