ProjectForce: Salesforce Integrated Project Management Tool

Alex Navarro Dapia, 1564645

Resumen — Salesforce es una plataforma de *software* en la nube muy potente, cada vez más utilizada por grandes y pequeñas empresas, para gestionar y mejorar sus relaciones con los clientes[1]. Entre las muchas ventajas que ofrece tenemos: administrar contactos y potenciales clientes, optimizar procesos de ventas y marketing, o automatizar tareas repetitivas para aumentar la eficiencia de negocio.
Pese a la enorme cantidad de posibilidades que ofrece este *software*, una actividad tan sencilla como la gestión de proyectos y tareas, o la administración del tiempo, no viene incluida por defecto en el diseño base de la plataforma, y las empresas acaban recurriendo a otros programas externos, como son Jira o Asana para realizar esta tarea. ProjectForce es una aplicación construida sobre la base de Salesforce que permitirá a las empresas que usan Salesforce, poder llevar un control de sus proyectos y tareas activos, gestionar el tiempo, administrar recursos, e incluso generar *dashboards* y reportes, todo desde un único lugar, sin tener que recurrir a licencias de *software* externos.

Palabras clave — Salesforce, Gestión de proyectos, Lighting Web Components, Apex, Visualforce, SOQL, Proyectos, Tareas, Tableros, Informes, Gestión de recursos.

Resum — Salesforce és una plataforma de *software* al núvol molt potent, cada vegada més utilitzada per grans i petites empreses, per gestionar i millorar les seves relacions amb els clients. Entre les molts avantatges que ofereix tenim: administrar contactes i potencials clients, optimitzar processos de vendes i màrqueting, o automatitzar tasques repetitives per augmentar l'eficiència de negoci.
Malgrat l'enorme quantitat de possibilitats que ofereix aquest *software*, una activitat tan senzilla com la gestió de projectes i tasques, o l'administració del temps, no ve inclosa per defecte en el disseny base de la plataforma, i les empreses acaben recorrent a altres programes externs com ara Jira o Asana per dur a terme aquesta tasca. ProjectForce és una aplicació construïda sobre la base de Salesforce que permetrà a les empreses que fan servir Salesforce, poder portar un control dels seus projectes i tasques actius, gestionar el temps, administrar recursos, i fins i tot generar *dashboards* i informes, tot des d'un únic lloc, sense haver de recórrer a llicències de *software* externs.

Paraules clau — Salesforce, Gestió de projectes, Lighting Web Components, Apex, Visualforce, SOQL, Projectes, Tasques, Taulers, Informes, Gestió de recursos.

Abstract — Salesforce is a powerful cloud-based software platform increasingly used by large and small companies to manage and improve their customer relationships. Among the many advantages it offers are managing contacts and potential customers, optimizing sales and marketing processes, or automating repetitive tasks to increase business efficiency.
Despite the high number of possibilities offered by this software, an activity as simple as project and task management, or time management is not included by default in the base design of the platform, and companies end up resorting to other external software such as Jira or Asana to perform this task. ProjectForce is an application built on top of Salesforce that will allow companies using Salesforce to keep track of their active projects and tasks, manage time, manage resources, and even generate dashboards and reports all from a single place without having to resort to external software licenses.

Index Terms — Salesforce, Project Managing, Lighting Web Components, Apex, Visualforce, SOQL, Projects, Tasks, Dashboards, Reports, Resource Management.

1 Introducción - Contexto del trabajo

Salesforce es una plataforma de software para ayudar a otras organizaciones mediante un potente CRM, con una completa gama de productos diseñados para diferentes equipos dentro de las mismas, ventas, atención al cliente, marketing o IT, entre otras.
Cada vez más, debido a las altas exigencias del mercado actual, grandes y pequeñas empresas están recurriendo a los servicios de Salesforce con el fin de ser competitivos, pero pese a ser un SaaS muy potente en el mercado, a veces cubrir todos los aspectos de gestión de una organización es complicado.
Es por ese motivo que Salesforce admite un gran nivel de customización y personalización de su plataforma, con el objetivo de dar libertad a las empresas de adaptar el servicio a sus necesidades. La idea de este proyecto es desarrollar una aplicación sobre la base de Salesforce para cubrir un gap en el servicio ofrecido, como es la gestión de proyectos dentro de

- *E-mail de contacte: 1564645@uab.cat*
- *Menció realitzada: Enginyeria del Software*
- *Treball tutoritzat per: Yolanda Benítez Fernández*
- *Curs 2023/24*

una organización, desde la cual se pueda llevar un control de proyectos y tareas activas, llevar un control del tiempo y de los recursos empleados.

2 OBJETIVOS

Con el desarrollo de este proyecto, el objetivo es obtener una aplicación completamente integrada y accesible desde la plataforma de Salesforce.com que permita a las empresas poder hacer las tareas esenciales de Project Managing, sin tener que recurrir a un software externo, lo que implicaría la contratación de licencias extras y tener que trabajar en múltiples plataformas. Vamos a buscar simplificar y centralizar.

Objetivos de alta prioridad del proyecto:

- La aplicación tiene que cubrir las necesidades básicas de una herramienta de este estilo:
  - Permitir a los usuarios crear, editar, eliminar y visualizar proyectos.
  - Asociar tareas a las fases de un proyecto y gestionar su estado y detalles.
  - Crear, editar y eliminar fases y asignarlas a un proyecto.
  - Crear, editar y eliminar Sprints que tendrán proyectos vinculados.
  - Mostrar una lista de proyectos y tareas en la interfaz de usuario.
- Gestión de Recursos:
  - Permitir crear, editar, eliminar y visualizar recursos.
  - Asignar recursos a tareas y proyectos según su disponibilidad y habilidades.

Objetivos de prioridad media del proyecto:

- Colaboración interna en tiempo real.
- Gestión de documentos:
  - Permitir a los usuarios subir, descargar y ver documentos relacionados con las tareas y proyectos.
  - Mantener una biblioteca centralizada de documentos relevantes para el proyecto.

Objetivos de baja prioridad del proyecto:

- Informes y análisis:
  - Proporcionar herramientas de informes y análisis para evaluar el progreso del proyecto, la utilización de recursos y la eficiencia del equipo.
  - Aplicación accesible desde múltiples tipos de dispositivos.

El desarrollo de este trabajo también incorpora objetivos personales, que van más allá de ofrecer una herramienta que cubra un conjunto de necesidades, como, por ejemplo:

- Adquirir y potenciar mi conocimiento de la plataforma de Salesforce a nivel de configuración, seguridad, personalización, etc.
- Aprender a trabajar con nuevos lenguajes de programación, como Apex (El lenguaje nativo de Salesforce) y LWC (Lightning Web Components) y Visualforce (Programación Web de Salesforce), y combinarlos con lenguajes ya trabajados como JavaScript y CSS.
- Aprender a utilizar herramientas ofrecidas por la plataforma de Salesforce, como SLDS[2] (Salesforce Lightning Design System) y Salesforce Flow Builder.

3 ESTADO DEL ARTE

Actualmente, y como ya se ha comentado con anterioridad, no existe ninguna solución oficial en la plataforma de Salesforce para este tema, aun así dentro de la AppExchange de Salesforce.com existen algunas soluciones propuestas por algunas empresas, alguna completamente Free como Gantt Chart[3] by Salesforce LABS, pero que son aplicaciones muy simples que están muy limitadas en cuanto a las funcionalidades que ofrecen; otras de pago, pero que dan acceso a parte de la app de forma gratuita Cloud Coach Starter[4] by Cloud Coach que son claramente mejores que las anteriores, pero como es evidente, la versión gratuita es muy limitada y prácticamente nos obliga a recurrir a su versión de pago, y otras completamente de pago Mission Control[5] by Aprika Business Solutions que son soluciones muy completas, pero por las que hay que pagar un precio por licencia de forma mensual que suele ser elevado.
También tenemos reconocidos software, pero que son externos a Salesforce, que nos ofrecen servicios similares como Jira[6] o Asana[7] que son aplicaciones referentes en este sector, pero que requieren a las empresas contratar licencias externas a Salesforce, lo cual supone un aumento de costes y la descentralización de los procesos.

4 METODOLOGÍA

Para el desarrollo de este proyecto se han utilizado metodologías Agile, más en concreto la metodología Kanban.
La metodología Kanban se basa en una filosofía centrada en la mejora continua, en ella se muestran las tareas mediante tableros divididos en columnas, los cuales identifican el estado de nuestras tareas según la etapa del trabajo en la que se encuentran.
TO DO – IN PROGRESS – DONE
Para este proyecto, las columnas han sido adaptadas, enfocándolas más concretamente a un proyecto de desarrollo de una App.
BACKLOG – PLANNED – DEVELOPING – TESTING – BLOCKED – DONE

La metodología se ha aplicado usando la herramienta kanbanchi[8]. El procedimiento seguido ha sido ir aña-

diendo al backlog todas las tareas que iban surgiendo durante cada una de las fases del proyecto y se han ido planificando y priorizando según prioridad para cumplir con las fechas de entrega estipuladas.

5 PLANIFICACIÓN

Para llevar a cabo este proyecto, la planificación que se ha seguido recoge las siguientes 6 fases:

Fase 1: Planificación y Definición del Alcance (4 semanas 19/02/24 – 17/03/24)
En esta fase es donde se concreta la idea del proyecto, se marcan objetivos, se establece una metodología de trabajo a seguir y se definen requisitos funcionales y no funcionales que tendrá la aplicación (Tablas disponibles en el anexo A1.1 y A1.2 de este documento).
También se han definido una serie de milestones a alcanzar con fechas relevantes para el proyecto y documentación a entregar en cada uno de ellos.
Fase 2: Investigación y Diseño (2 semanas, 18/03/24 – 31/03/24)
En esta crucial fase para el proyecto, se reunirán todas las herramientas necesarias para llevar a cabo el proyecto, se preparará el entorno de trabajo, se diseñará la arquitectura de la aplicación y la estructura de la DDBB y se definirán los modelos de datos y relaciones entre los objetos de Salesforce.
Fase 3: Desarrollo Backend (4 semanas, 01/04/24 – 28/04/24)
Aquí se desarrollarán las clases Apex para la lógica de la aplicación y el acceso a datos, y se implementará la integración con Salesforce.
Fase 4: Desarrollo Frontend (4 semanas, 29/04/24 – 26/05/24)
Para el desarrollo frontend me centraré en la creación de las páginas de Visualforce y los LWC y Aura Components para la interfaz de usuario, así como de la navegación por la App.
Fase 5: Integración y Pruebas (2 semanas, 27/05/24 – 9/06/24)
Para la fase de integración y pruebas se comprobará que todos los componentes desarrollados se integran correctamente y se realizarán test para identificar y solucionar posibles problemas de funcionamiento en la aplicación, así como para evaluar el rendimiento de la misma y corregirlo si es necesario.
Fase 6: Documentación (1 semana 10/06/24 – 16/06/24)
Documentar el proceso de desarrollo, incluyendo la arquitectura, el diseño y las decisiones técnicas, así como el funcionamiento de la versión final de la aplicación, para dar por completado el informe final del proyecto.

5.1 Herramientas y tecnologías utilizadas

Tanto para el desarrollo del backend como del frontend, no se ha necesitado ninguna herramienta externa más que el IDE Visual Studio Code con el Salesforce Extension Pack, que nos permite hacer la conexión con el entorno de Salesforce[9], reconocer la sintaxis del código en Apex y la capacidad de debuggarlo, además de permitir crear directamente Aura components y trabajar con los metadatos de la aplicación.
La DDBB se diseña y va incorporada en la propia plataforma de Salesforce mediante la creación de objetos.
Tenemos como soporte también la Developer Console de Salesforce, que es una consola Online integrada en la misma plataforma de Salesforce que nos permite ejecutar código Apex in situ, ya sea para realizar pruebas, consultas SOQL u otras pequeñas tareas. Pese a ser una herramienta muy completa, el desarrollo grande de la aplicación estará en el Visual Studio Code, por su mayor facilidad de conectar con GitHub para poder llevar control de versiones durante el desarrollo.

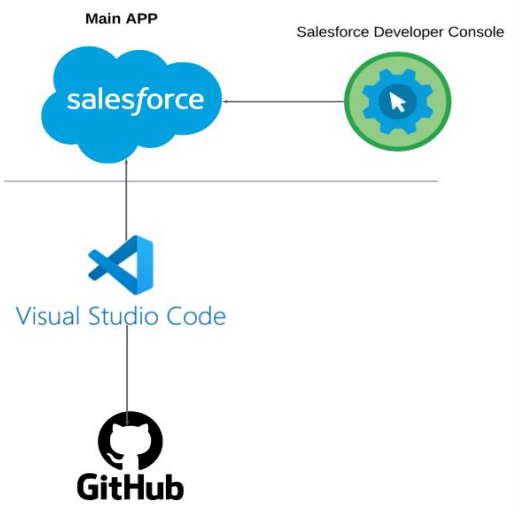


Fig 1: Esquema del entorno utilizado

6 DISEÑO Y DESARROLLO DEL PROYECTO

En esta sección se detalla el proceso que se ha seguido para llevar a cabo todos los desarrollos en torno a la aplicación ProjectForce.

6.1 Diseño

En esta fase se ha tratado de dar forma real a la propuesta inicial del proyecto. Para ello, se trabajó en marcar una serie de objetivos que debería cumplir el proyecto una vez finalizado y también en la redacción de los requisitos que iban a marcar las funcionalidades del sistema, los actores principales de la aplicación y también aquellas herramientas y tecnologías que se iban a necesitar para todo el desarrollo.
En esta fase también se crearon los diagramas que ayudan a entender la estructura y el funcionamiento de la aplicación:

- Diagrama WBS: Que nos muestra la planificación de las principales tareas a trabajar durante todo el proyecto, separadas por fases en un esquema en forma de árbol. (Disponible en Anexo A.2.1 del documento)
- Diagrama de casos de uso: Nos enseña cómo interactúan los actores de la aplicación con las funcionalidades principales de esta. (Disponible en anexo A2.2. del documento)
- Diagrama de la BD: Es un diagrama extraído de la propia plataforma de Salesforce que nos muestra los objetos que hay en nuestra base de datos y cómo estos están relacionados los unos con los otros. (Disponible en anexo A.2.3. del documento)

6.2 Estructura del proyecto

La estructura del proyecto Salesforce ya nos la construye y nos la marca por defecto.
A través del comando *sfdx force:project:create --projectname "Name"*, y gracias a la extensión pack de Salesforce para VS Code, se nos construye de forma predeterminada una estructura de proyecto, la cual va directamente conectada a la Developer Org (Entorno de Sandbox de Salesforce) en la que estemos trabajando los desarrollos.
En esta estructura vemos muy bien organizados los repositorios de nuestro proyecto que contendrán todos los archivos necesarios para el funcionamiento de la app, como son: repositorio de objetos de la base de datos, uno para las clases Apex de nuestra app, los Lightning y Aura Components que son los componentes individuales que vamos construyendo, y que luego irán todos unidos en app final a través de la herramienta lightning app builder, etc.

6.3 Diseño arquitectónico de la aplicación

El diseño arquitectónico de ProjectForce ha seguido uno de los modelos más trabajados durante todo el grado, el Modelo Vista Controlador (MVC).
El Patrón Modelo-Vista-Controlador (MVC)[10] es un patrón arquitectónico utilizado en el diseño de aplicaciones de software, que separa la representación de la información (Modelo), la interfaz de usuario (Vista) y la lógica de control (Controlador). Esta separación permite una organización más modular y escalable del código, facilitando el mantenimiento y la evolución del sistema a lo largo del tiempo.

Aplicación del Patrón MVC en una app de Salesforce:
Modelo: En una aplicación de Salesforce, el Modelo está representado por los objetos de Salesforce y la lógica de negocio implementada en Apex.
Vista: La Vista está representada principalmente por las Lightning Pages construidas a partir de la combinación de Componentes Aura y Componentes Standard.
Controlador: El Controlador estaría representado por las clases Apex, que actúan como controladores de las Lightning Pages, o los controladores de eventos en los componentes Aura, procesando las acciones del usuario y actualizando el Modelo y la Vista según sea necesario.

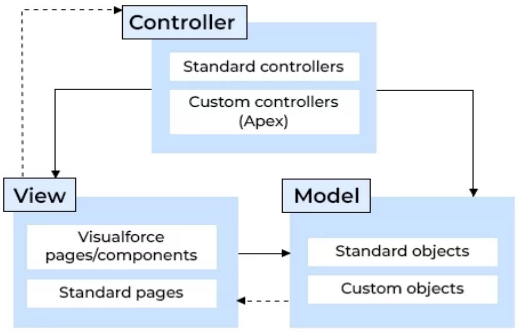


Fig.2 Esquema MVC en Salesforce[11]

6.4 Diseño de la base de datos

Otra de las partes importantes del desarrollo del proyecto ha sido la elaboración de una base de datos que cumpliera con los objetivos, requisitos y casos de uso del proyecto.

La construcción de esta se hace dentro del mismo entorno de Salesforce, a través de la creación de los que se le llaman Custom Objects, que son objetos que nada tienen que ver con los objetos estándar de Salesforce que vienen por defecto, sino que Salesforce aquí nos da libertad de crear y configurar objetos a nuestro antojo, y según nuestras necesidades, con el fin de tener almacenados todos aquellos datos que necesitaremos.

La interfaz de creación de Custom objects es muy sencilla de entender y de utilizar, pero además Salesforce nos ofrece la posibilidad de trabajar estos objetos a través de archivos XML formados por la configuración de cada objeto, permitiéndonos hacer cambios desde VS Code y subiéndolos a la Developer Org cuando queramos.
Para bajar estos archivos de metadatos a nuestro proyecto en local se ha usado el comando *sfdx force:source:retrieve -m CustomObject*.

6.5 Clases Apex

Las clases Apex[12] son las unidades de código que contienen los métodos variables y constantes para poder implementar la lógica de nuestro backend.
Dentro del desarrollo en Salesforce hay una práctica muy extendida en cuanto al desarrollo de clases y es el añadir un sufijo al nombre de la clase, dando ya una previa pista de cuál es la función de la misma.
Los que yo he utilizado para mi aplicación ProjectForce han sido:

- ***Ctrl o *Controller:** Este nos estaría indicando que nuestra clase está asociada a alguno de nuestros componentes Aura y en ella aparecerán los métodos y las consultas SOQL para atacar a la BD y extraer datos que nos ayudarán con la interacción entre la interfaz del usuario y el modelo de nuestra aplicación, como por ejemplo en la

clase ProjectSummaryCtrl.cls la cual está vinculada al componente aura que nos muestra el resumen del objeto proyecto y contiene los métodos para extraer sus fases y tareas vinculadas.

- ***Helper:** Estas clases suelen ser utilizadas en nuestros componentes aura para compartir funcionalidades de utilidad, así que son una manera de agrupar funciones reutilizables por varios objetos en una sola clase, como en nuestra clase PF_ObjectManagerHelper, donde tenemos métodos para el manejo de tareas y fases de un proyecto.
- ***Test o *Tester:** Por último, este sufijo de clase Apex, como su nombre indica, es una clase en la que se realizan pruebas unitarias para probar y verificar la funcionalidad de una clase, como en nuestra clase PF_ObjectTester, donde tenemos métodos para validar CRUD de varios de nuestros objetos.

6.6 Componentes Aura

Los componentes aura son las unidades funcionales que encapsulan secciones modulares y reutilizables de la interfaz de usuario de nuestra aplicación.
Estos componentes están formados por un paquete de archivos, cada uno encargado de una parte del componente, entre los que se encuentran los elementos visuales, funciones que se invocan desde la vista y las conexiones con la lógica de las clases, los estilos personalizados de nuestros elementos, etc.
Para mostrar el funcionamiento de estos componentes y su estructura de archivos, lo haré tomando de ejemplo algunos de los componentes que he desarrollado para la construcción de mi aplicación.

ProjectWrapper.cmp: Es un archivo de tipo XML que contiene la estructura de elementos visuales del componente que nos muestra, en este caso la pantalla de resumen de un proyecto creado en nuestra aplicación. Este componente, por ejemplo, está definido mediante etiquetas <aura:, <force:, y <lightning:

- Las etiquetas <aura se han utilizado principalmente para definir atributos disponibles dentro de la interfaz de nuestro componente.
- Las etiqueta <force:recordData> para escribir datos que provienen desde nuestra BD en el componente.
- Las etiquetas <:lightning: por último, forman parte del marco de componentes lightning y se han utilizado para crear botones, campos de entrada de texto o títulos acorde con el Lightning Design System.

ProjectWrapperController.js es un archivo de tipo JavaScript y contiene métodos que manejan eventos y lógica específica en la interfaz de usuario, por ejemplo, en el caso de mi aplicación, responder a acciones del usuario típicas como pulsar un botón, cambiar el valor de un campo, etc.
ProjectWrapper.css es el archivo donde nos encontramos algo de personalización, para dar apariencia a algunos de los elementos de nuestro componente, pero dado que los elementos definidos en las etiquetas <lightning: del componente ProjectWrapper.cmp ya traen un estilo por defecto acorde con los estándares de Salesforce, en este archivo he incidido en poca cosa más que en tipo y color de la fuente, márgenes y el tamaño de alguno de los botones.

6.7 Lightning App Builder

Lightning App Builder es una interfaz de construcción de aplicaciones, que nos proporciona Salesforce, para montar nuestra aplicación utilizando todos esos componentes que hemos ido desarrollando. En esta interfaz se ha montado la vista para las diferentes partes de la aplicación y una de las grandes ventajas que nos ofrece es la facilidad que ofrece para la reutilización de módulos, la posibilidad de combinar módulos personalizados con otros estándares de Salesforce como son las record views (Vistas amigables y con filtros configurables para visualizar objetos) o módulos como el Chatter, también incluido en los componentes estándar de Salesforce, que es un módulo de chat que se ha aprovechado e incluido en varias de las vistas.

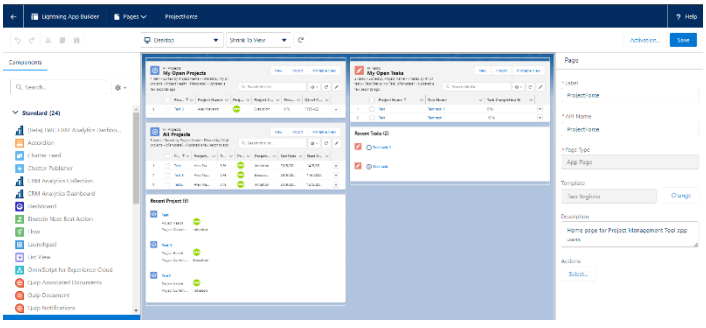


Fig.3 Interfaz Lightning App Builder

6.8 Despliegue de la app al entorno Salesforce

El despliegue de la app en Salesforce se ha trabajado de forma continua, ya que cada vez que se han desarrollado cambios se han trasladado a la plataforma de Salesforce.
El procedimiento a seguir ha sido:

1. Ejecutar primeramente el comando sf project deploy preview --target-org ProjecForce que nos da una preview de los componentes que se van a desplegar en nuestra developer org y sus potenciales conflictos (si los hay).
2. Seguidamente, usar el comando *sf project deploy start* para hacer el despliegue definitivo a la plataforma.

También tenemos otro flujo que contempla las modificaciones que se han hecho directamente desde la plataforma de Salesforce por comodidad, utilizando la Salesforce Developer Console y es que, en este caso, para luego traer los cambios a nuestro proyecto en VSCode, se ha usado el comando *sf project retrieve start*.

6.9 Jerarquía de roles y niveles de Seguridad dentro de la aplicación

Para conseguir la jerarquía de roles y los niveles de seguridad deseados en la aplicación, se han trabajado dos cosas principalmente, la creación de permission sets y la creación de una jerarquía de roles. Ambas cosas se crean en la misma plataforma de Salesforce.

Un Permission Set es un conjunto de configuraciones y permisos que podemos ofrecer a los usuarios de Salesforce, sin la necesidad de que tengan que acceder a la plataforma con un perfil distinto, y permitiéndonos así otorgar acceso a funciones específicas de la plataforma a un grupo reducido de usuarios.

- En ProjectForce se han creado los siguientes Permission Sets:
- PF Global Admin: Tiene permisos para acceder a la configuración de la aplicación y a los metadatos de la misma.
- PF Resource Planner: Encargado de planificar los recursos disponibles por años.
- PF User: Resto de usuarios con acceso a toda la app menos a la posibilidad de hacer modificaciones en la estructura de la app, ni acceso a metadatos ni a la planificación de recursos.

La jerarquía de roles, por otro lado, también juega un importante papel en cuanto a la seguridad de la plataforma y determina los niveles de acceso a datos que tiene cada usuario. Un usuario solo podrá acceder a los datos de aquellos usuarios que estén por debajo de él.

Para mi Developer ORG, que como se puede ver en la imagen Fig.4 se llama UAB y es donde se ha construido la app ProjectForce, se han definido los siguientes roles:

- CEO y App Manager vienen por defecto en Salesforce y no pueden ser eliminados, pero corresponden a los dos usuarios dentro de la organización con nivel completo de acceso en cuanto a seguridad y datos se refiere.
- App Administrator: Son los usuarios con el set de permisos de global admin y que pueden acceder a la aplicación por la "backdoor" y hacer modificaciones y acceder a los metadatos de la app.
- Project Admin: Son todos los usuarios que sean owners de uno, o más proyectos, dentro de la app y solo tienen permiso de modificación en sus proyectos.
- Project Collaborator: Son todos los usuarios que forman parte de un proyecto como colaboradores y que solo tendrán acceso de modificación a tareas de las cuales sean owners, pero no podrán hacer modificaciones en tareas ni proyectos de otros usuarios.

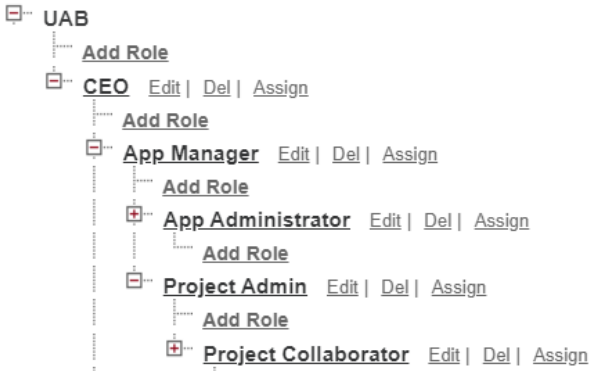


Fig.4 Jerarquía de Roles

6.10 Approval process para el objeto tarea

Con tal de acercarme más a una solución realista, otra de las funcionalidades que se han incluido en la App es un approval process para el objeto tarea.

Un Approval process es un conjunto de reglas que se definen en relación a un objeto para que estos tengan que ser enviados para aprobación y aprobados o rechazados por alguien superior en la jerarquía de roles.

La creación de un Approval process genera automáticamente dos elementos dentro de nuestra app, en primer lugar, un botón mediante el cual el usuario enviará el nuevo record creado para su aprobación y, por otro lado, genera una pantalla extra y ya prediseñada a la app solo accesible por los managers, donde reciben la lista de tareas que los usuarios les envían para ser aprobadas.

El approval process interactúa con el objeto al cual va vinculado, bloqueándolo mientras no sea aprobado y actualizando el campo correspondiente dentro del objeto para indicar en qué estado se encuentra:

- Submitted (Enviado por el usuario para aprobación)
- Approved (Aprobado por el N+1)
- Rejected (Rechazado por el N+1)

7 TEST

En este apartado se revisarán los diferentes test a los que se ha sometido Projectforce para validar su funcionamiento.

7.1 Test Unitarios

Los test unitarios en Salesforce, como su nombre indican, son pruebas aisladas que ayudan a verificar que el código Apex funciona correctamente para cada uno de sus métodos individuales, buscando conseguir con esto una calidad del código adecuada, facilitar la detección de errores en el código, y la refactorización segura a la hora de realizar cambios.

Es importante destacar que Salesforce requiere que al menos el 75% del código Apex esté cubierto por test unitarios antes de permitir su despliegue a un entorno de producción, lo cual fomenta que los desarrolladores hagamos pruebas de todo aquello que programamos.

Las clases de test, como ya se ha mencionado en el apar-

tado 6.5 de este documento, las diferenciamos por la terminación *Test o *Tester en el nombre de la clase y la cabecera @isTest.
En mi aplicación tenemos clases como PF_ObjectTester que es una clase que engloba métodos que testean funcionalidades de varios de los objetos de mi aplicación, comienza con el método initializeData para crear los datos de prueba necesarios y luego prueba varios métodos de la capa de controladores relacionados con la gestión de proyectos y recursos.
Los resultados obtenidos pueden observarse en la Figura 5.

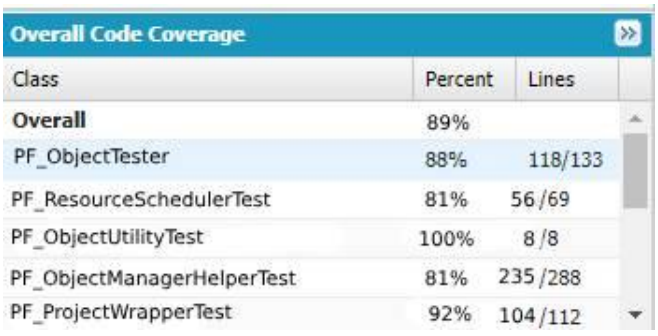


Fig.5 Resultados de tests y code coverage

7.2 Pruebas de rendimiento

Para los test de rendimiento se ha utilizado una herramienta de analítica dentro de la interfaz lightning app builder llamada Page Analysis, que hace una predicción del tiempo de carga de cada una de las páginas/pantallas de nuestra app y nos da una calificación según el tiempo estimado. Page Analysis nos permite hacer las pruebas para las pantallas según el dispositivo en el que van a ser ejecutadas.
En las siguientes dos figuras (Fig.6 y Fig.7) podemos ver los resultados obtenidos para el análisis de la pantalla de Fases de Projectforce en su versión Desktop y Phone:

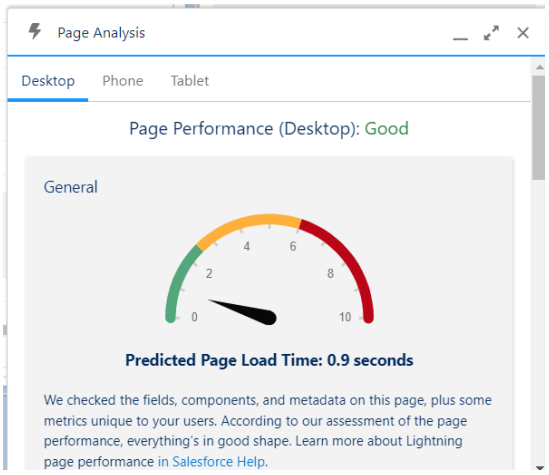


Fig.6 Phase page analysis Desktop

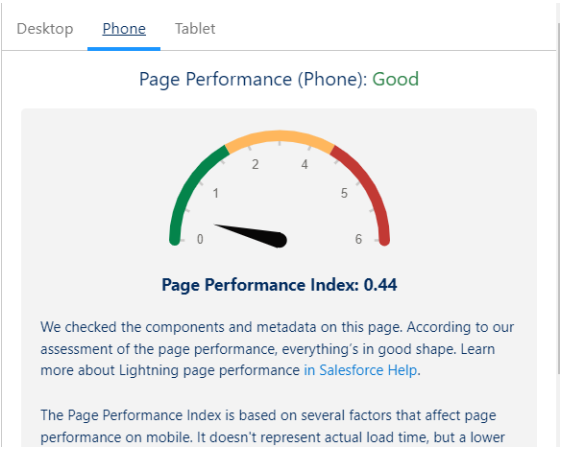


Fig.7 Phase page analysis phone

7.3 Error Handling

Para el manejo de errores de la aplicación, se ha usado una función muy útil de Salesforce llamada Validation Rules, esta función nos permite definir reglas dentro de nuestros Custom Objects para lanzar errores por pantalla al usuario cuando esté incumpliendo la regla que nosotros hayamos definido.
Como ejemplo, yo pondré la regla que he creado en mi objeto proyecto llamada InitialSrateDate_should_be_less, en la que a través de la Error Condition Formula:

NOT(ISNULL(Initial_Start_Date__c)) && NOT(ISNULL(Initial_End_Date__c)) && Initial_End_Date__c< Initial_Start_Date__c

Si los valores son diferentes de NULL y la fecha final del proyecto es más pequeña que la inicial, lanzará al usuario por pantalla un error con el siguiente mensaje (Figura 8):

Initial End date must be after Initial Start date.

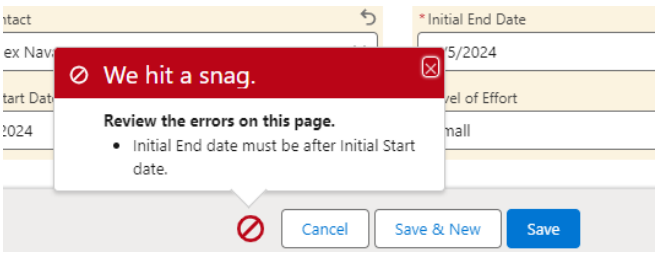


Fig.8 Ejemplo de Pop Up de error

8 RESULTADOS

Finalmente, se ha obtenido una aplicación de gestión de proyectos completamente integrada en Salesforce y funcional que cubre las necesidades básicas que podemos pedir a una aplicación para este propósito.
Aun así, a medida que se ha ido desarrollando la aplicación han ido apareciendo complicaciones y nuevos requerimientos que han hecho que tanto el listado de requisitos, como los casos de uso del proyecto hayan tenido que ser modificados.

En los siguientes apartados se comentarán los objetivos alcanzados y no alcanzados, cómo la aplicación ha cubierto los requerimientos del proyecto, y las vistas obtenidas en su versión final.

8.1 Objetivos alcanzados

En la tabla que hay a continuación se hace una relación de qué funcionalidad se ha desarrollado para cubrir cada uno de los objetivos marcados en el proyecto.

Fig.9 Tabla de funcionalidades - Objetivos cumplidos

Funcionalidad Requerida	¿Ha sido desarrollada?	Objetivo Cumplido
Pantalla para la gestión de proyectos	Sí	1.1.a
Pantalla para la gestión de tareas	Sí	1.1.b
Pantalla para la gestión de fases	Sí	1.1.c
Pantalla para la gestión de Sprints	Sí	1.1.d
Colaboración interna en tiempo real mediante chatter	Sí	2.1
Funcionalidad de subida de archivos tanto en proyectos como tareas	Sí	2.a y 2.b
Incorporación de la herramienta de reporting de Salesforce a la app	Sí	3.1.a
Configuración de lightning pages disponibles para múltiples dispositivos	Sí	3.2
Vista en formato lista y calendario para la visualización de tareas	Solo se ha desarrollado la vista en formato lista, no la de calendario	3.3

8.2 Objetivos NO alcanzados

Como podemos observar en el apartado anterior, la gran mayoría de los objetivos del proyecto han sido cubiertos, pero también tenemos alguno que ha tenido que ser desestimado y no ha podido cubrirse.

(Tabla disponible en el anexo A.3)

En el caso del objetivo del diagrama de Gantt, no ha podido ser cubierto, ya que el volumen de trabajo que requería el desarrollo de una funcionalidad de este estilo, tanto por su enorme complejidad en el manejo de las dependencias entre los diferentes objetos y la necesidad de crear una compleja representación visual de la información mediante componentes Lightning, he preferido desestimar y poder centrar todos los esfuerzos que se habrían invertido en este desarrollo en poder mejorar otras partes de la aplicación, evitando así una posible falta de tiempo que podría haber causado el incumplimiento de otros objetivos prioritarios.

En cuanto a la Vista en formato calendario, al ser uno de los objetivos de más baja prioridad que realmente no afecta a la funcionalidad de la app, ya que está cubierta con la vista tradicional en forma de lista, al final no se ha desarrollado.

8.3 Vistas obtenidas

En la homepage de la app ProjectForce podemos ver un resumen totalmente interactivo de algunas de las funciones de la app.

En el lado izquierdo tenemos 3 módulos relacionados con los proyectos, uno que muestra proyectos abiertos de los que somos owners, otro módulo que nos muestra una lista de todos los proyectos abiertos y otro módulo que nos muestra los últimos proyectos que hemos visitado.
En el lado derecho de la pantalla tenemos dos módulos relacionados con las tareas, uno para ver mis tareas abiertas y otro para ver las tareas recientes visitadas.
Cabe recalcar que estos módulos están configurados para solo mostrar 5 registros, con tal de no saturar ni romper la armonía de la página principal.

También tenemos la opción de acceder directamente a cualquier registro de la pantalla haciendo click sobre él y tenemos botones en cada módulo para acceder de forma rápida a la edición de registros o a los formularios de creación, todo ello sin salir de nuestra pantalla principal.

Desde la barra superior podemos acceder a cada uno de los módulos de la aplicación. Aparte de la homepage tenemos una pantalla para proyectos, otra para fases, tareas,

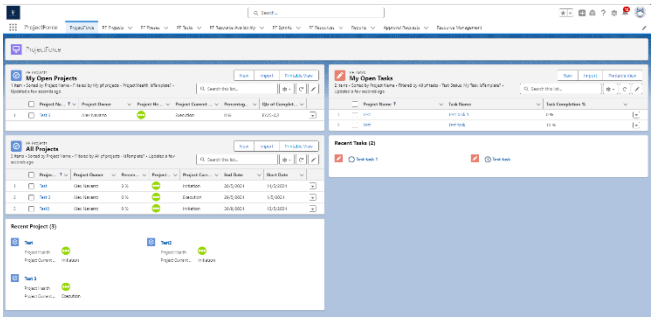


Fig.10 Vista de la Home Page

sprints, recursos y asignación de recursos, reports y approval process (esta última solo para Admins).
Al hacer click en el módulo de Projects se nos abre una lista con todos los proyectos creados sobre la cual podemos aplicar filtros, o buscar por nombre un proyecto en concreto.

Si hacemos click sobre cualquiera de los registros, se nos abre la pantalla de proyecto y podremos ver un resumen con información de nuestro proyecto seleccionado, que nos muestra el project health con un icono en verde si es buena, o en rojo si es mala, el current stage del proyecto y un pequeño resumen de algunos de sus datos relevantes.
La vista también incluye una pestaña llamada chatter & Activity que es un componente estándar de Salesforce, que se ha incluido al módulo de proyecto para que los colaboradores puedan interactuar entre ellos.
Otra de las pestañas es la de Details, donde podemos ver en detalle todos los datos del proyecto, descripción, fechas de inicio y vencimiento, owner, etc.
También tenemos una pestaña para visualizar todas las tareas asignadas a alguna de las fases de este proyecto y otra para visualizar los recursos asignados.
Y finalmente, una última pestaña llamada Files, en la cual los colaboradores pueden subir archivos desde su computador y adjuntarlos al proyecto para almacenarlos en una biblioteca.
Para cambiar entre todas las pestañas mencionadas, la vista tiene una barra de navegación en la parte superior.

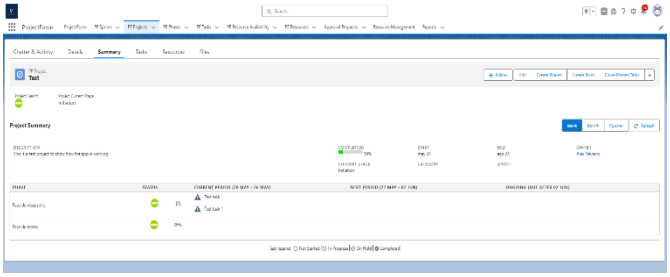


Fig.11 Vista pantalla de proyecto

Si seguimos navegando por las pantallas de nuestra app, podemos llegar a la pantalla de fases.
De la misma forma que con el objeto proyecto, al hacer click en las fases desde la barra de navegación, se nos abre una lista con todos nuestros registros sobre los que podemos buscar o filtrar.
Esta vista está constituida por un resumen de los datos de la fase, como por ejemplo la phase health, su nombre, el porcentaje de terminación, el proyecto al cual está relacionada, las fechas de inicio y fin, etc.

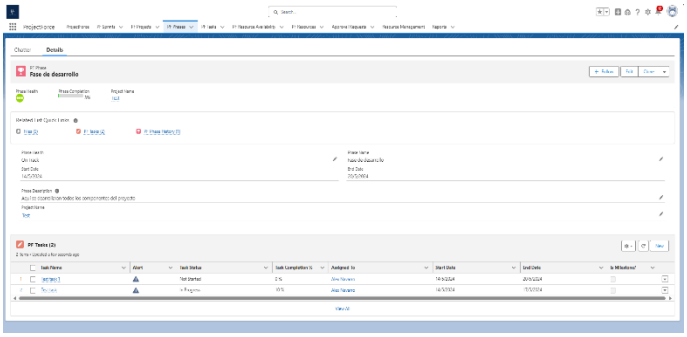


Fig.12 Vista pantalla de fases

Además, también tenemos un resumen de las tareas asignadas a la fase en cuestión y se le ha incluido la pestaña de chatter, de igual forma que en los proyectos, para permitir a los usuarios interactuar mediante mensajes dentro de una fase.
Desde esta vista también tenemos disponibles botones para poder crear entradas nuevas o para editar los datos del registro que tenemos abierto.
Para la ventana de tareas el funcionamiento es el mismo que con las anteriores. Tenemos un listado de tareas con algo de información como el nombre, la fase y proyecto a la que pertenecen y haciendo click sobre cualquiera de ellas abrimos la pantalla resumen de tarea, una sencilla que nos mostrará detalles de los campos de nuestro objeto, como son el nombre de la tarea, las fechas de inicio y fin, el porcentaje de terminación, descripción, el estado de aprobación, o a qué fase de qué proyecto está vinculada.
También se ha incluido la pestaña de chatter, igual que en los proyectos y fases.

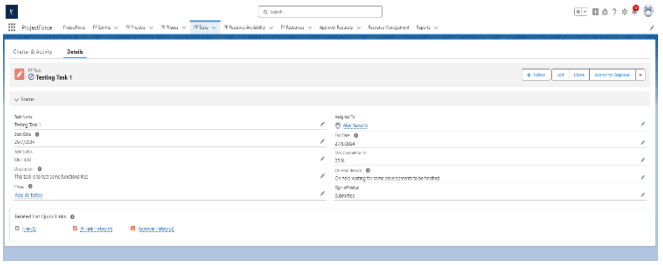


Fig.13 Vista pantalla de tareas

Otra funcionalidad importante que destacar de esta vista es el botón de submit for approval en la esquina superior derecha, mediante el cual el usuario creador de la tarea puede enviar su tarea a través de un approval process previamente configurado, para que la tarea le llegue al project manager para ser aprobada.
En este approval process también entra en juego otra de las pantallas de ProjectForce y es la pantalla de aprobación de tareas, a la cual solo tienen acceso los usuarios con rol de Project manager.

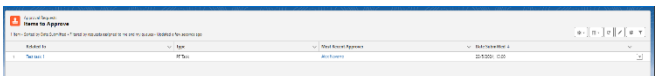


Fig.14 Listado de tareas a aprobar

Es una vista que se crea por defecto cuando definimos un proceso de aprobación para alguno de nuestros objetos de Salesforce.
En esta pantalla, los project managers recibirán un listado de las tareas que los colaboradores han ido creando y que requieren de una aprobación antes de ser procesadas.

Otra de las vistas principales de nuestra aplicación son las dos pantallas que le servirán al Resource Manager a interactuar con los recursos de la app, darlos de alta y asignarlos a los proyectos y tareas correspondientes.
Usando el formulario de alta de recursos, el Resource Manager podrá crear nuevos recursos siempre vinculados a un usuario de la plataforma. Para ello, tendrá que introducir también el año en el cual este nuevo recurso estará disponible y el porcentaje de disponibilidad por meses de este nuevo recurso.
Luego nos vamos directamente hacia la otra vista, la de

recurso, en la que el Resource Manager podrá coger alguno de los recursos previamente dados de alta a través del formulario y asignarlos a un proyecto en concreto. El funcionamiento sería a través del formulario de Allocation, en el cual el manager selecciona el recurso que va a asignar, el rol y el proyecto al cual va a vincularlo.
También tiene que introducir el porcentaje que le va a dedicar al proyecto asignado por cada uno de los meses del año; este porcentaje será restado del porcentaje de disponibilidad total del recurso cuando se dio de alta originalmente.
La vista cuando hacemos click a un recurso asignado es muy simple, ya que solo nos da la información que el Resource manager ha rellenado en el formulario de asignación, eso sí con la posibilidad de que la vuelva a editar cuando lo necesite.

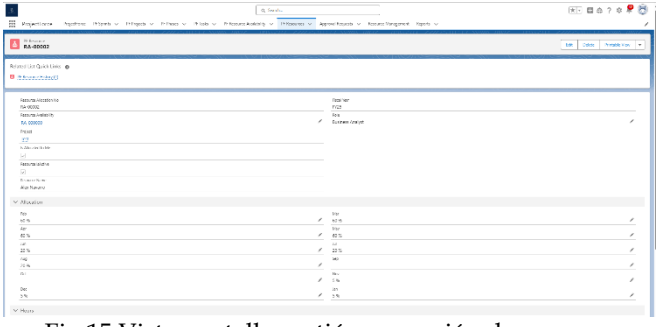


Fig.15 Vista pantalla gestión y creación de recursos

Finalmente, cuando hacemos click en alguno de los recursos originalmente dados de alta a través del formulario de altas en la vista de Resource Availability, nos encontramos con algo más de información, ya que podremos ver en detalle un resumen de la asignación para ese recurso, donde tenemos la capacidad con la que se le dio de alta originalmente, qué porcentaje de este recurso hemos gastado ya, cuánto nos queda y a qué proyectos está asignado.

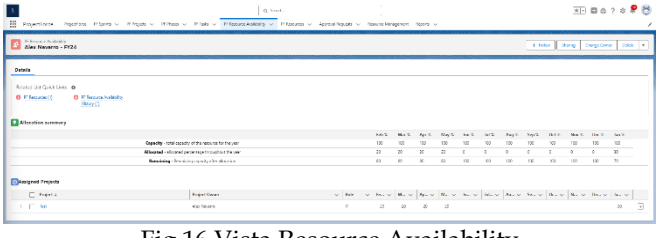


Fig.16 Vista Resource Availability

(En el anexo A.4 se incluyen capturas de las mismas pantallas, pero en dispositivo móvil.)

9 Conclusiones

El objetivo marcado en las fases iniciales de mi proyecto ProjectForce era ofrecer una herramienta gratuita integrada en la plataforma de Salesforce, para que los usuarios de empresas que utilizan Salesforce en sus compañías tuvieran a su disposición una herramienta sencilla que les permitiera gestionar los proyectos que se van trabajando a lo largo del año.
No ha sido para nada un camino sencillo el tener que construir una solución completamente nueva desde 0, en una plataforma sobre la que, si bien tenía algo de conocimiento básico, me faltaban muchas nociones de cómo hacer ciertas cosas.
Pero pese a las dificultades encontradas, y gracias a una buena planificación, ProjectForce ha podido salir adelante y ha sido muy gratificante llegar a la versión actual de la app y ver que cumple las expectativas iniciales que me marqué.

10 Líneas abiertas

Después de todo el trabajo empleado en sacar Projectforce adelante, he decidido seguir con el proyecto rumbo a conseguir una herramienta 100% completa y competitiva en el mercado. Para ello, se va a seguir trabajando en pulir y mejorar las funcionalidades de la app ya existentes con tal de mejorar la experiencia de usuario. Además, trabajaré en traer nuevas funcionalidades a la app que le permitan ser una opción atractiva entre la actual competencia, como por ejemplo la finalización del módulo de diagrama de Gantt que no ha podido ser incorporada en la actual versión integración o integración con la API de Outlook Calendar o Microsoft Teams para poder programar reuniones desde la aplicación.

Agradecimientos

Agradecer a mi tutora Yolanda Benítez por interesarse por el tema de mi trabajo y por las productivas sesiones de seguimiento, aportando ese punto más de experiencia a mi trabajo. También agradecer a mis compañeros de trabajo de Danone Iberia por haber hecho de usuarios finales de mi aplicación, aportando sus opiniones y sugerencias.
Y finalmente agradecer a mi familia, amigos y compañeros, y sobre todo a mi pareja, por haberme ayudado a mantener la motivación y la constancia necesaria para sacar el proyecto.

Bibliografía

[1] Salesforce? (s.f.). https://www.salesforce.com/
[2] LDS. (s.f.). https://www.lightningdesignsystem.com/
[3] Gantt Chart. (s.f.).https://appexchange.salesforce.com/appxListingDetail?listingId=a0N3A00000FMaanUAD&tab=e
[4] Cloud Coach. (s.f.). https://cloudcoach.com/
[5] Aprika. (s.f.). https://aprika.com/
[6] Atlassian. (s.f.). Jira. https://www.atlassian.com/es/software/jira
[7] Asana. (s.f.). https://asana.com/
[8] Kanbanchi Ltd. (s.f.). https://www.kanbanchi.com/
[9] Chaudhary. How to Setup Visual Studio Code for Salesforce. Apex Hours.https://www.apexhours.com/how-to-setup-visual-studio-code-for-salesforce/
[10] PlantUML. (s. f.) https://plantuml.com/es/use-case-diagram
[11] Soni, N. MVC Architecture in Salesforce. https://s2-labs.com/admin-tutorials/mvc-architecture/
[12] Vallejos, G. ¿Cómo escribir clases de Apex en Salesforce? Snake on Code.https://snakeoncode.com/como-escribir-clases-apex-salesforce/

APÉNDICE

A.1 Requerimientos del sistema

A.1.1 Requerimientos funcionales

ID	Descripción
RF_1	El sistema debe permitir al usuario poder crear, eliminar y modificar proyectos
RF_1.1	El sistema debe convertir al usuario en administrador del proyecto una vez lo ha creado
RF_1.2	Si el usuario es administrador del proyecto debe poder añadir y eliminar colaboradores al proyecto
RF_1.3	El sistema debe permitir al administrador del proyecto modificar el estado de sus proyectos
RF_1.4	Si el usuario es administrador del proyecto debe poder asignar y modificar la prioridad del proyecto
RF_1.5	Si el usuario es administrador del proyecto debe poder asignar una fecha de inicio y de fin al proyecto
RF_1.6	El sistema debe llevar un registro de las fechas de inicio y fin de un proyecto
RF_2	El sistema debe convertir en colaborador al usuario que pasa a formar parte de un proyecto
RF_2.1	El sistema debe permitir al colaborador crear, eliminar y modificar tareas
RF_2.2	El sistema debe permitir asociar tareas a la fase de un proyecto
RF_2.3	Si el usuario es colaborador debe poder asignar una fecha de inicio y de fin a una tarea
RF_2.4	El sistema deberá llevar registro de las fechas de inicio y fin de una tarea
RF_2.5	El sistema debe permitir al colaborador visualizar las tareas en forma de listado
RF_2.6	El sistema debe permitir al colaborador enviar las tareas para que sean aprobadas por el administrador del proyecto
RF_2.7	Si el usuario es colaborador debe poder asignar prioridad a una tarea
RF_2.8	El sistema debe permitir a cualquier colaborador asignar a otro integrante como responsable de la tarea
RF_2.9	Cualquier colaborador debe poder dejar un comentario en cualquier tarea
RF_2.10	El sistema tiene que relacionar las fases de un proyecto y sus tareas asignadas a un proyecto en cuestión
RF_2.11	El sistema debe permitir al administrador aprobar las tareas que le envíen sus colaboradores
RF_3	El sistema debe permitir al administrador de recursos crear, eliminar o modificar recursos
RF_3.1	El sistema debe permitir al administrador de recursos asignar recursos a un proyecto
RF_4	Cualquier colaborador debe poder generar reportes que reflejen el rendimiento del proyecto en el que participa
RF_4.1	Cualquier colaborador debe poder visualizar los reportes creados
RF_5	Cualquier colaborador puede adjuntar documentos a una tarea
RF_5.1	Cualquier colaborador puede visualizar documentos adjuntos a una tarea
RF_6	El sistema deber permitir a cualquier usuario cambiar el idioma de la aplicación cuando lo desee
RF_7	El sistema tendrá una jerarquía de roles interna. Solo el creador del proyecto será administrador y dispondrá de los permisos correspondientes. Cualquier usuario que forme parte de un proyecto será colaborador y dispondrá de los permisos correspondientes. Los usuarios que no hayan creado ningún proyecto ni formen parte de uno también tendrán un rol contemplado y solo tendrán habilitado el permiso de crear un proyecto nuevo.
RF_8	El sistema debe permitir al administrador del proyecto crear sprints y asociarlos a un proyecto
RF_9	El sistema deber permitir al administrador del proyecto crear fases y asignarlas a su proyecto
RF_10	El sistema deber permitir al administrador de la app hacer modificaciones en los metadatos de la misma

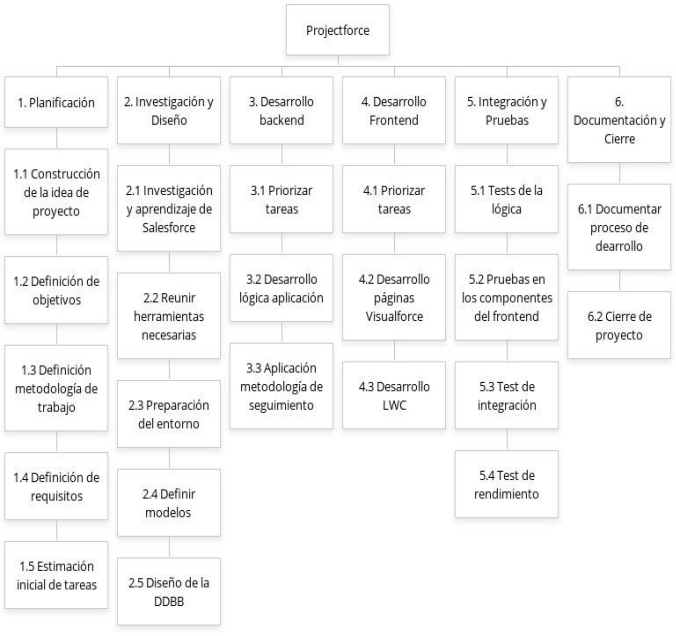
A.1.2 Requerimientos NO Funcionales

ID	Descripción	Categoría
RNF_2	El sistema debe funcionar sin errores en cualquier parte del flujo	Rendimiento
RNF_3	El sistema debe ser accesible desde cualquier navegador y desde la aplicación de Salesforce para dispositivos móviles	Usabilidad
RNF_4	El sistema deberá tener una interfaz amigable e intuitiva que siga los estándares de diseños de Salesforce	Portabilidad
RNF_5	Los tiempos de respuesta ofrecidos por el sistema para cualquiera de sus funcionalidades deben ser aceptables en todo momento	Rendimiento

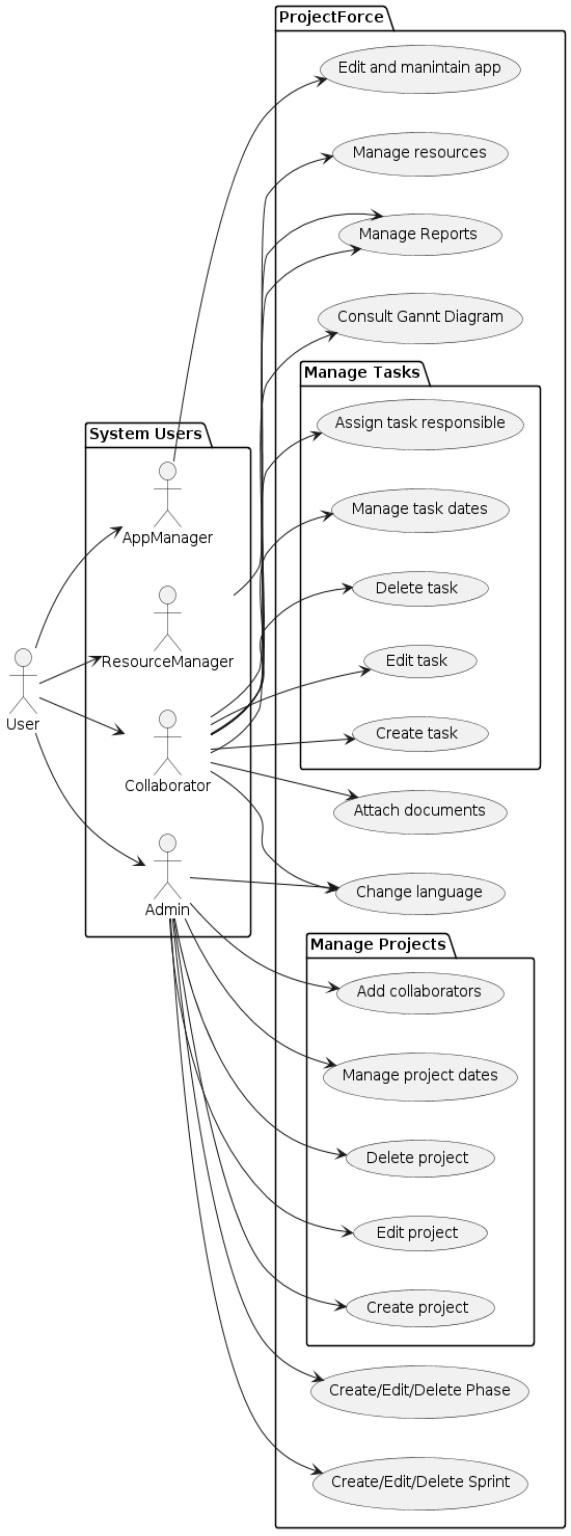
A.2 Diagramas del proyecto

Imágenes de los diferentes diagramas elaborados:

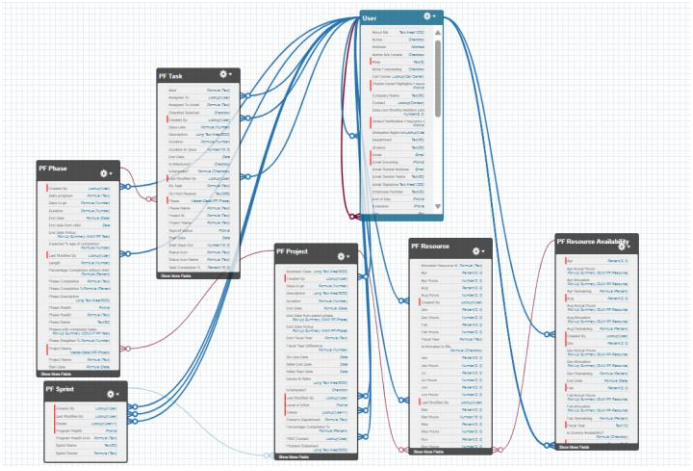
A.2.1 Diagrama WBS



A.2.2 Diagrama Casos de uso



A.2.3 Diagrama de conexiones de la BD



A.3 Tabla de objetivos NO alcanzados

Funcionalidad Requerida	¿Ha sido desarrollada?	Objetivo No Cumplido
Diagrama de Gantt	No	1.3
Vista en formato lista y calendario para la visualización de tareas	Solo se ha desarrollado la vista en formato lista no la de calendario	3.3

A.4 Vistas obtenidas para versión en dispositivo móvil

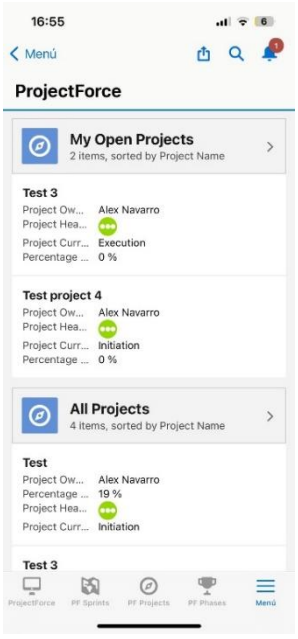


Fig.17 Home page phone version

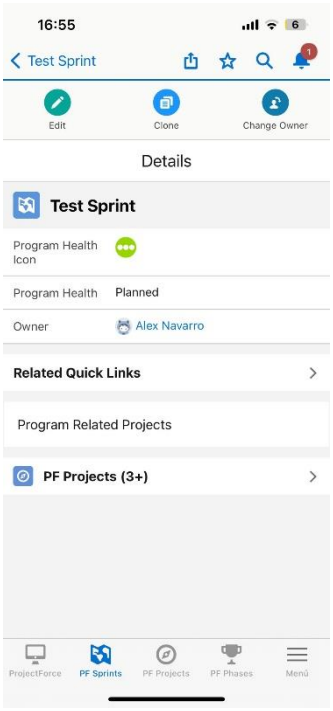


Fig.18 Sprint page phone version

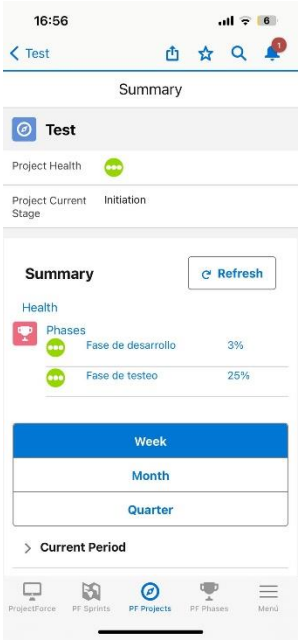


Fig.19 Project page phone version

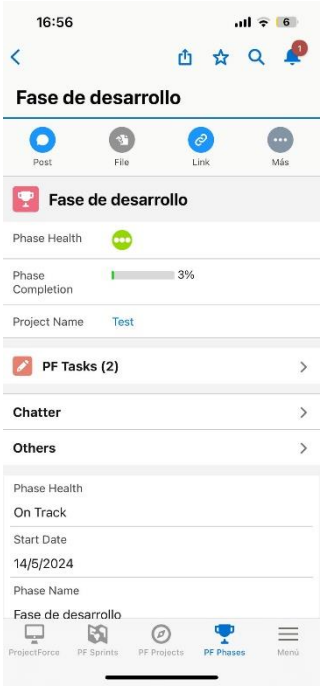


Fig.20 Fase page phone version

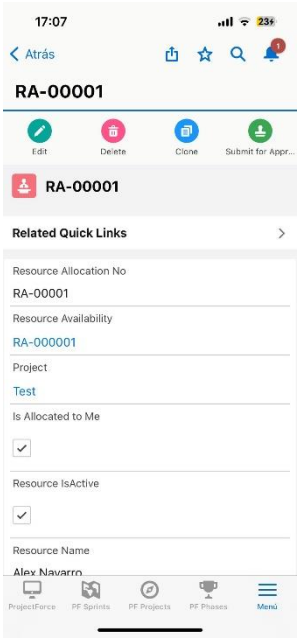


Fig.22 Resource page phone version

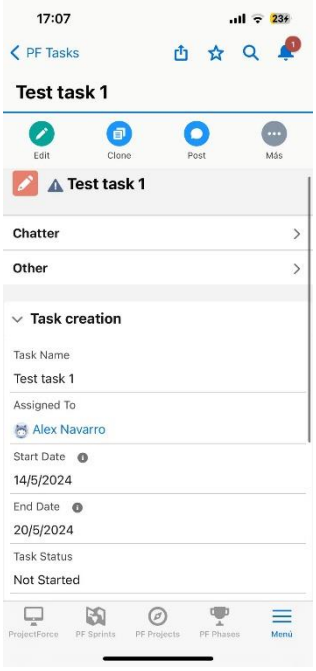


Fig.21 Task page phone version