DESARROLLO DE APLICACIÓN WEB PARA EL SEGUIMIENTO Y REPORTE DE ACTIVIDADES LABORALES EN SOFTTEK.

Jhon Jairo Ballen Agudelo

Código: 1810010725

Estudiante ingeniería de sistemas

Tutor/a:

Isabel Mahecha Nieto

Institución Universitaria Politécnico Grancolombiano

Facultad de ingeniería, diseño e innovación

Bogotá D.C, Colombia

2022

Tabla de contenido

Tabla de figuras **Página**

RESUMEN

En la realización de la practica empresarial se identificó un problema que involucra a cada empleado, diariamente se deben reportar las horas trabajadas y las actividades en las que se empelaron dichas horas. Semanalmente llega un correo electrónico reportando el retraso de los empleados en el cargue de las horas mencionadas anteriormente y solicitándoles que se realice este proceso.

En base en lo mencionado anteriormente, se plantea como una posible solución el desarrollo de una aplicación web, pero para poder realizar este desarrollo primero se realiza un proceso de investigación de los temas que involucran el mismo, como lo son todos los conceptos relacionados a una aplicación web, pasando por los diferentes campos del desarrollo como lo son; bases de datos, back-end, front-end, metodologías de desarrollo y documentación.

El objetivo principal del proyecto es desarrollar una aplicación web que le permita a los empleados llevar un seguimiento de las actividades laborales y generar reportes. Para poder cumplir con este objetivo se seguirá el modelo de ciclo de vida del software, por ende, primero se planearán los requerimientos en base a las necesidades del software, después se clasificarán para así organizar la estructura del software. Para poder iniciar el desarrollo donde se codificarán los primeros componentes funcionales.

1. PLANTEAMINETO DEL PROBLEMA:

En las primeras semanas de iniciar la practica empresarial se identificó un problema el cual se repetía cada semana que pasaba. La empresa debe llevar un registro diario por horas de las actividades que realizan cada uno de sus equipos de trabajo y sus empleados. Continuamente se ve que al iniciar semana llega un correo electrónico solicitando a los empleados realizar el cargue de sus actividades, en las cuales se encuentran atrasados y los números son alarmantes. Por confidencialidad de estos datos no se pueden mostrar dichos números, pero en estos se podían observar personas que no habían realizado el cargue de ningún día a lo largo de todo un mes.

2. OBJETIVOS

2.1 OBJETIVO GENERAL

Desarrollar una aplicación web que permita llevar un seguimiento de las actividades realizadas por los empleados cada día, la aplicación tendrá un sistema de notificaciones vía correo electrónico.

2.2 OBJETIVOS ESPECIFICOS

- Desarrollar una aplicación Back-end que se encargara del almacenamiento de datos.

- Desarrollar una aplicación Front-end que consultará la información almacenada en la base de datos y tendrá las interfaces para la navegación de los usuarios.

- Implementar una metodología de desarrollo ágil en todo el proceso de desarrollo del proyecto.

- Implementar un sistema de notificaciones que funcione por correo electrónico y sea gestionado por el Front-end.

3. CARGO Y FUNCIONES

3.1 CARGO: Practicante en el área de robotización y procesos.

3.2 FUNCIONES:

- Procesos internos.
- Plataformas del ecosistema de la operación.
- Procesos de RPA.
- Productos de RPA.
- Desarrollo en lenguajes JAVA y Groovy.
- Manejo de bases de datos.
- Implementación de herramientas de la operación y complementos.
- Acompañamiento en proyectos RPA.

4. ALCANCE:

El proyecto planteado busca desarrollar un sistema de información piloto donde se registren y se haga seguimiento de las actividades diarias de los empleados. Basándose en la problemática planteada que semanalmente se reportan retrasos en el cargue de las mismas. Para poder cumplir ese objetivo se plantea desarrollar una aplicación web que le permita a sus usuarios registrar sus actividades para generar reportes mensuales o anuales.

Adicionalmente se implementará un sistema de notificaciones que enviará un correo electrónico a los usuarios. Todo esto se realizará por medio de la metodología ágil de desarrollo Scrum, para generar documentación y reportes de todo el proceso.

5. JUSTIFICACIÓN:

En el área de trabajo se tuvo la oportunidad de conversar con varios compañeros y preguntar sobre este reporte diario de horas que solicita la empresa, a lo que ellos comunicaron que encontraban atrasos en sus reportes, dado que durante su día laboral olvidaban realizarlo y a medida que pasaban los días no sacaban el tiempo para hacerlo, hasta las horas se acumulaban a fin de mes.

Teniendo presente lo mencionado anteriormente, el impacto que se espera del desarrollo descrito para este proyecto es que los empleados puedan sacar un espacio diario de su tiempo laboral para poder registrar las actividades, que suelen repetirse, y así dejar en la aplicación el seguimiento de ellas. Adicionalmente la aplicación generará notificaciones a lo largo del día para recordarle a los empleados que realicen el registro descrito anteriormente.

Desde los registros diarios que haga cada empleado en la aplicación también se tendrá la posibilidad de generar reportes, a los que podrán acceder para observar las horas trabajadas y en que actividades las implemento. De este modo tener un control del trabajo realizado por cada uno de ellos.

La herramienta que se utiliza actualmente para este proceso tiene funciones muy limitadas, porque solo le permiten al usuario realizar cargue de horas de cuatro semanas atrás. No les permite almacenar los códigos, los proyectos ni el tipo de actividades que realizaron. Además, todas sus interfaces usan los componentes nativos de HTML.

6. MARCO TEÓRICO:

6.1 Aplicación web:

Son aquellas aplicaciones codificadas que no requieren ser instaladas en un equipo de cómputo para poder funcionar, esto se debe a que estas se ejecutan a través de internet en los principales navegadores comerciales y su funcionamiento está basado en peticiones de información siguiendo el protocolo de hipertexto HTTP.

Este concepto se encuentra directamente relacionado al almacenamiento de datos en la nube. Esto sucede porque la información se almacena en grandes servidores de terceros y permiten así la ejecución de estas aplicaciones web con tan solo tener internet y un navegador.

6.2 Protocolo HTTP:

Este protocolo abarca varios temas, pero los de mayor importancia al momento de realizar la aplicación web son los métodos HTTP que maneja este protocolo, el cual nos permitirá consumir desde el back-end codificado, por medio de peticiones información y poder administrar los datos almacenados en la base de datos desde nuestra aplicación web front-end.

Los principales métodos de HTTP que maneja este protocolo son:

GET: Para obtener recursos.
POST: Enviar datos junto a la petición.
PUT: Actualizar información por medio de una petición.
DELETE: Para eliminar información por medio de una petición.

Existen otros métodos, pero estos son los que se usarán principalmente en el funcionamiento de la aplicación web.

Forma de una petición HTTP:

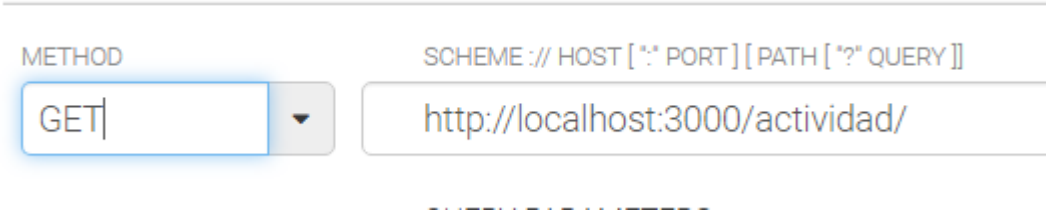


Figura 1. Forma de realizar una petición de información por medio de hipertexto siguiendo el protocoló HTTP, en la imagen se puede observar, el método, el host del servicio, el puerto y el path al que se le realizara la petición. Tomada de la aplicación Talend API Tester.

6.3 Aplicación web:

"JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web capaces de interactuar con el usuario" (Mohedano & Saiz, 2012).

Como se menciona en la anterior cita, JavaScript es otro de los lenguajes de programación que podemos encontrar en todo el repertorio que existen en la actualidad, pero este destaca debido a que su implementación permite a los usuarios de una página web interactuar con esta, pasando de tener paginas estáticas que solo usan HTML, al inicio de páginas dinámicas gracias a este lenguaje de programación.

6.4 HTML (HyperText Markup Language):

Como nos lo describen sus siglas, HTML es un lenguaje de marcas de hipertexto. Una persona sin mucho conocimiento en el área de ingeniería de sistemas, al mencionarle sobre una página web lo primero que podría pensar es en HTML, esto se debe a que la estructura y los componentes más básicos de una página web se ven involucrados con HTML. En la actualidad es muy común ver que se implementen otras herramientas para mejorar y agregar funciones a HTML.

HTML nos brinda la estructura base de una página web, por ello en el podemos encontrar botones, entradas de texto, diferentes etiquetas e incluso apartados para agregar imágenes o archivos de formato multimedia.

6.5 Angular:

"AngularJS es un potente framework de JavaScript creado para el desarrollo de aplicaciones web dinámicas. Nos permite extender la sintaxis de HTML por medio de atributos propios del framework, para expresar componentes de nuestra aplicación de manera dinámica" (Arizmendi, 2018).

Este framework nos permite trabajar sitios web dinámicos, porque a través de JavaScript se pueden realizar diferentes importaciones de funcionalidades únicas del mismo, para el desarrollo del proyecto este fue fundamental en cada paso del desarrollo, debido a que la interacción del usuario con la aplicación web está completamente determinada por las interfaces codificadas en AngularJS.

Al instalar AngularJS en nuestro computador tenemos un proyecto base, ahí es donde empezaremos a utilizar cada una de sus funcionalidades, usando diferentes comandos y generando así los componentes que serán parte fundamental del Front-end de la aplicación web. En conclusión, AngularJS será la estructura principal que reunirá todas las interfaces graficas del sitio web y se comunicará con la aplicación Back-end por medio del protocolo HTTP para el manejo de información y almacenamiento de datos.

6.6 Base de datos no relacional:

Las bases de datos no relacionales surgen debido a la necesidad de algunas aplicaciones que no podían solucionar las bases de datos relacionales, además que las bases de datos no relacionales fueron creadas con el objetivo de poder almacenar grandes conjuntos de datos.

Las bases de datos no relacionales se destacan porque almacenan columnas de una forma diferente, esto se debe a que sus columnas funcionan de forma dinámica y le permiten así generar cambios sobre dichas columnas sin generar complicaciones en la agrupación de la información.

Este tipo de bases de datos tienen que suplir de alguna manera las relaciones entre tablas que se ven en las bases de datos relacionales, por lo que tienen esta funcionalidad por medio de referencias que le permiten ser mucho más flexible y así manejar mayor cantidad de datos.

6.7 MongoDB:

Es una de las principales herramientas para el manejo de bases de datos no relacionales, en el podemos encontrar diferentes funcionalidades y productos, cada uno de ellos con un propósito especifico, en nuestro caso implementaremos Mongo Atlas, que nos permite conectar nuestras aplicaciones con esta herramienta y desplegar así en la nube la base de datos y allí podemos observar las colecciones existentes junto a gráficos sobre la transferencia de los datos allí almacenados.

6.8 Node.js:

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

Este entorno de ejecución permite importar en él diferentes aplicaciones y/o complementos que nos permitirán realizar la aplicación Back-end supliendo las necesidades del desarrollo, en nuestro caso le agregamos varios complementos que repasaremos más adelante, pero es parte fundamental del desarrollo, porque Node.js será la estructura principal de dicha aplicación.

6.9 Express.js:

Es una infraestructura para el desarrollo de aplicaciones web implementando Node.js, el cual nos brinda un conjunto solido de funcionalidades para el desarrollo de este tipo de aplicaciones, además en el podemos implementar los diferentes métodos que conforman el protocolo HTTP, abriéndonos la posibilidad de crear API's.

6.10 Definición de UML (Lenguaje Unificado de Modelado):

Lo podemos definir como un lenguaje para modelar objetos de manera visual, para así poderlo implementar al desarrollo de software como una herramienta para definir los planos y el diseño de una aplicación, no es considerado un lenguaje de programación, pero su aplicación nos ayuda a visualizar lo que se desea desarrollar.

Su función principal pasa a ser documentación del diseño y el modelado de las diferentes partes que componen un sistema de software, es por ello que UML define los siguientes diagramas para desarrollar una aplicación:

Diagrama de clases: Este diagrama es la base principal porque se enfoca en una solución orientada a objetos, teniendo presentes los atributos y operaciones de las entidades que conforman un sistema de información.

Diagrama de componentes: Permite evidenciar la relación estructural entre los elementos de un sistema de software.

Diagrama de despliegue: En este diagrama se pueden ilustrar las partes tanto de hardware como de software que serán necesarios para la implementación del software desarrollado.

Diagrama de actividades: Representa gráficamente algún componente del sistema junto a su funcionamiento.

Diagrama de secuencia: Este diagrama nos permite observar la interacción entre los objetos que conforman el software y pueden evidenciar el funcionamiento de un escenario especifico.

Diagrama de estados: Demuestran el comportamiento de diferentes objetos, teniendo presente el estado en el que estos se encuentran.

Diagrama de casos de uso: Permite observar la funcionalidad del software, donde se verán reflejadas las entidades involucradas en el mismo.

6.11 Metodología ágil Scrum:

Es una metodología de desarrollo que se usa bastante en la actualidad, debido a su flexibilidad sobre el ciclo de vida del software, esta metodología está enfocada principalmente en grupos de trabajo grandes. A continuación, revisaremos las principales características que la componen:

Product Backlog: Esta conformado por un listado que contiene las necesidades que define el cliente.

Sprint: Un intervalo de tiempo definido por el equipo, donde se dividen las tareas que se van a trabajar.

Sprint Backlog: Esta conformado por la lista de tareas que se realizan en un Sprint especifico.

Product Owner: Es el encargado de crear las historias de usuario y de gestionar las tareas junto a la asignación de ellas.
Team: Equipo de trabajo en cargado del desarrollo y cumplimiento de cada una de las tareas.

Scrum Master: Es el encargado de llevar el seguimiento y control de las practicas fundamentales que conforman la metodología Scrum.

Estas serán las principales características que se manejaran en este proyecto, en consecuencia, de que no es un grupo de trabajo el que desarrollara el software, si no una sola persona, la cual se encargara de planear los Sprints, definir el Product Backlog y el Sprint Backlog. Como evidencia de la implementación de la metodología ágil, se obtendrán los siguientes requerimientos:

- Requerimientos funcionales.
- Requerimientos no funcionales.
- Diagramas UML.

6.12 Nodemailer:

Esta tecnología fue la que permitió realizar el envió de notificaciones vía correo electrónico de la aplicación, para implementar esta se realizó su instalación en el back-end y por medio de una petición de tipo POST consultada por la aplicación front-end se envía una notificación al correo electrónico almacenado en la base de datos.

7. METODOLOGÍA:

7.1 Analizar la página actual para el cargue de horas

El primer paso antes de empezar la realización de la aplicación web, fue analizar tanto el funcionamiento como la apariencia de la página actual, como se pudo observar en la justificación. Su funcionamiento es sencillo, en el se encontro un apartado para registrar las horas laborales, donde se debe elegir la semana del mes y allí determinar las horas diarias, junto a la actividad en la que se trabajó.

En la aplicación actual de la empresa solo se puede navegar entre las semanas del mes actual y no tiene un apartado para guardar el código del proyecto en el que se trabajó y en las opciones para cada actividad, solo se observa el código y no se sabe desde el formulario de que se trata cada actividad.

En la siguiente sección se encuentra un pequeño análisis de la herramienta actual para el cargue de horas en la empresa.

7.2 Elegir una metodología ágil de desarrollo:

Al momento de elegir la metodología de desarrollo se realizó esta elección en base a las metodologías de desarrollo que más se usan actualmente, una de ellas es la metodología ágil Scrum.

Metodología ágil Scrum:

Es una metodología que se basa en intervalos de tiempo, en los que se planea y se distribuye el trabajo, es principalmente utilizada en grupos de trabajo, en el desarrollo de este proyecto, se tuvieron que realizar ciertos cambios sobre la metodología, esto a consecuencia de que solo trabajo una persona en el desarrollo.

En el marco teórico se explica mejor cómo funciona esta metodología y cada una de sus principales características.

Al elegir esta metodología ágil para realizar el desarrollo obtendremos los siguientes resultados:

- Product Backlog
- Sprint Backlog.
- Hoja de ruta.
- Informes de trabajo realizado.
- Informes de trabajo pendiente.

7.3 Levantamiento de requerimientos

Después del análisis realizado en el paso anterior, el paso a seguir en el desarrollo de la aplicación fue determinar las necesidades que deberá satisfacer la aplicación web que desarrollemos, tanto para la empresa como para los empleados que le darían uso diario a esta. Para ver reflejado esto se obtendrá a través de la siguiente documentación.

- Requerimientos funcionales.
- Requerimientos no funcionales.
- Diagramas UML.

7.4 Selección de tecnologías para el desarrollo

Por todo lo obtenido en el punto anterior, se realiza un análisis de cuáles serán las tecnologías que se utilizaran para el desarrollo de la aplicación, teniendo presente todo el levantamiento de requerimientos y de este proceso se obtendrá:

- El listado de las tecnologías que se usaran.
- Descripción de la implementación de cada una de las herramientas.

7.5 Diseño

En esta etapa se creará toda la aplicación que se encargara del Back-end, entonces es donde se creara la base de datos que almacenara toda la información, se creara el proyecto en Express para codificar cada una de las peticiones bajo las cuales funcionara la aplicación. Con este proceso obtendremos:

- Aplicación Back-end.
- Servicios rest que consumirá el front.
- Colecciones de datos en MongoDB Atlas.

Muck-Ups:

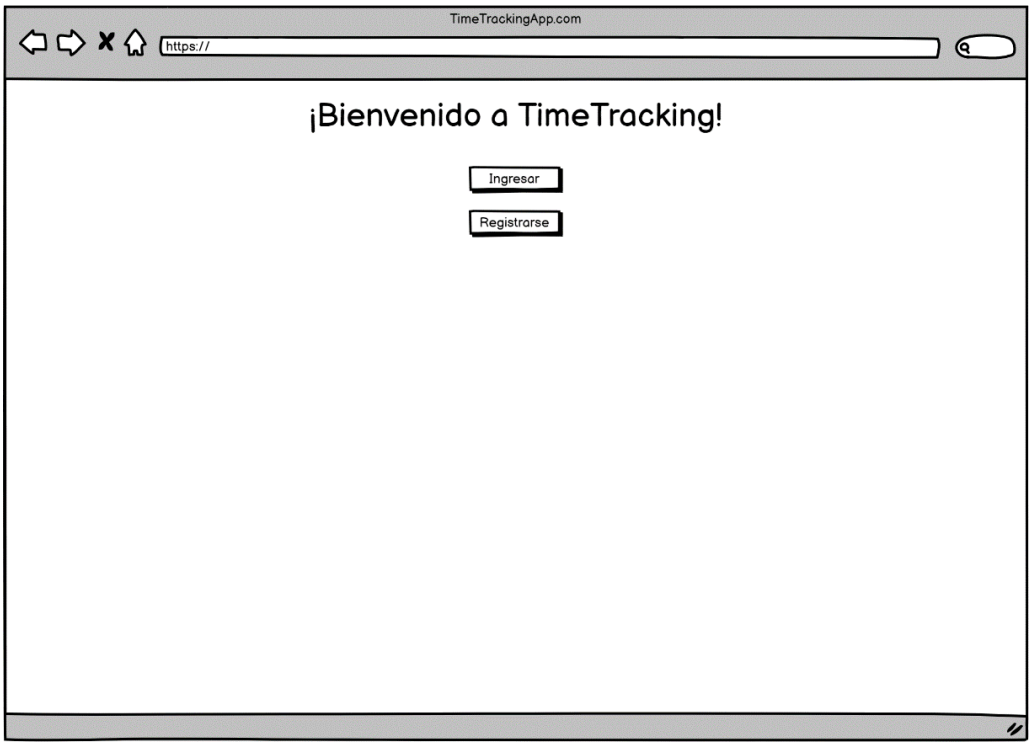


Figura 2. Diseño para la página principal de la aplicación web TimeTracking. Generado en Balsamiq.

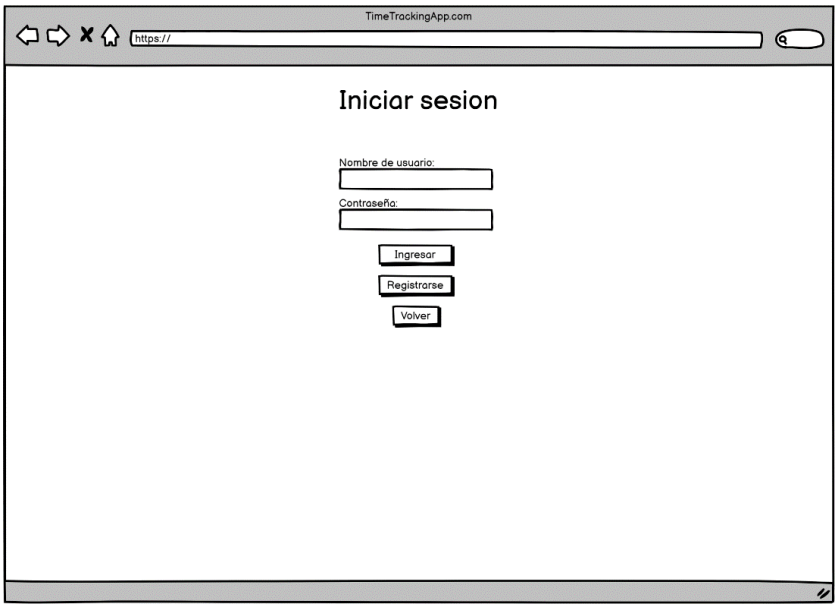


Figura 3. Diseño para la página de autenticación de la aplicación web TimeTracking. Generado en Balsamiq.

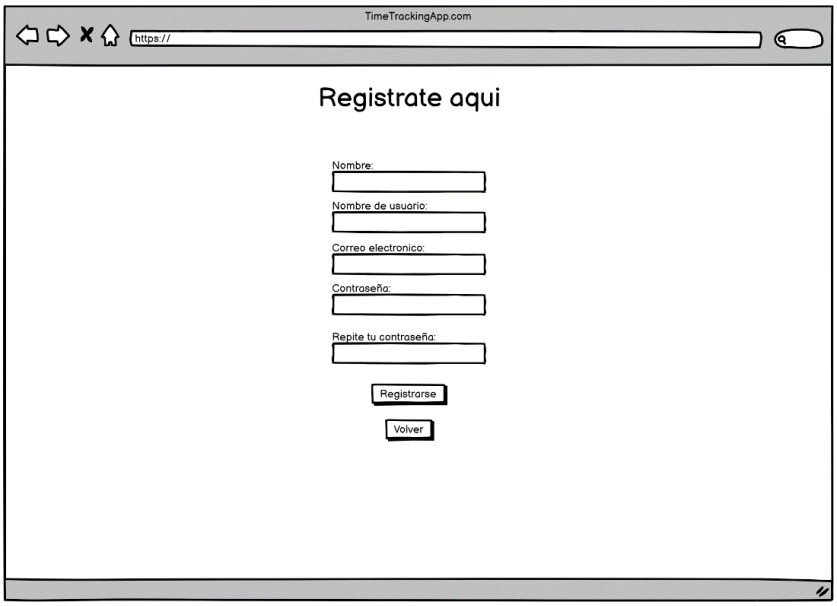


Figura 4. Diseño para la página de registro para nuevos usuarios de la aplicación TimeTracking. Generado en Balsamiq.

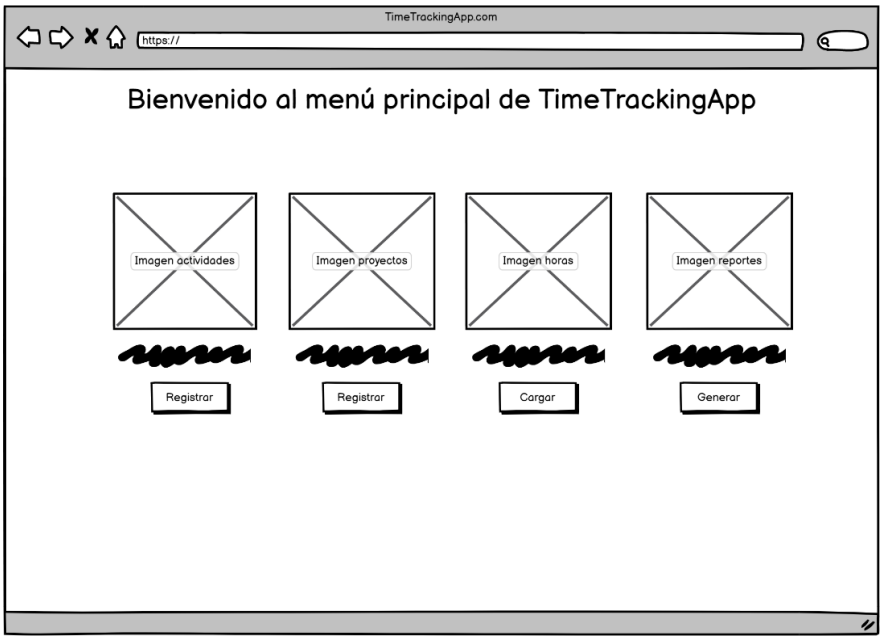


Figura 5. Diseño para la página del menú principal de la aplicación web TimeTracking cuando el usuario se autentica con éxito. Generado en Balsamiq.

Figura 6. Diseño para la página de registro para una nueva actividad del usuario de la aplicación web TimeTracking. Generado en Balsamiq.

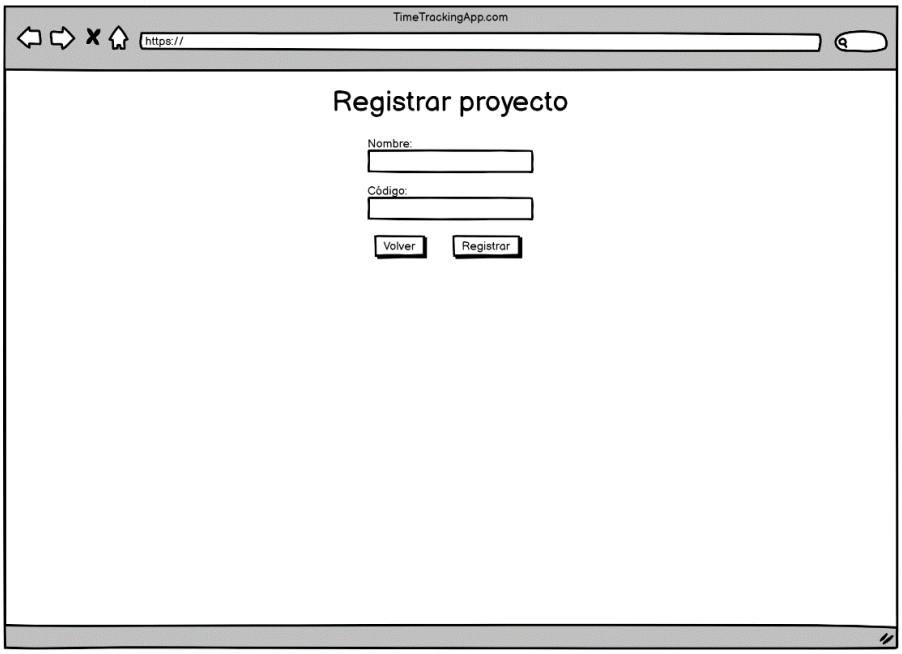


Figura 7. Diseño para la página de registro para un nuevo proyecto del usuario que se encuentra autenticado en la aplicación web TimeTracking. Generado en Balsamiq.

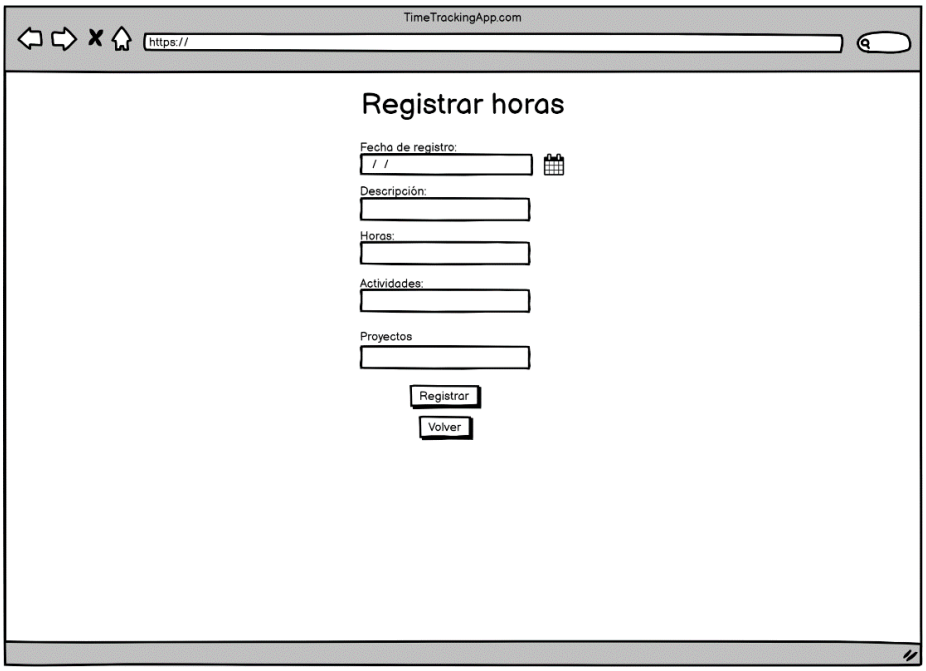


Figura 8. Diseño para la página de registro de horas del usuario que se encuentra autenticado, aquí es donde el realizara el cargue correspondiente a sus horas y podrá ingresar el proyecto y la actividad a la que le corresponda. Generado en Balsamiq.

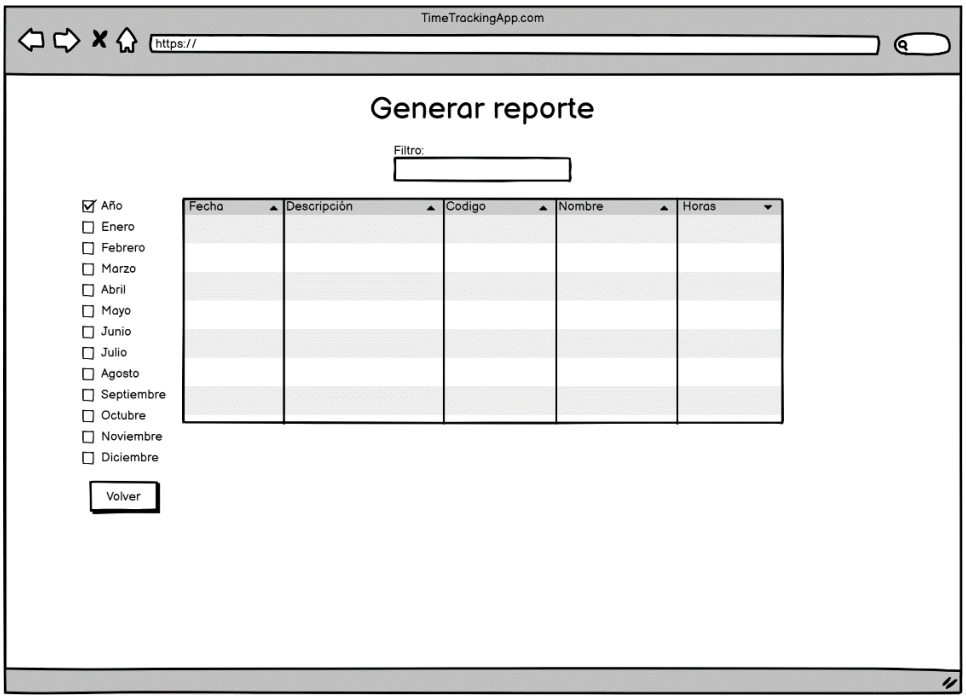


Figura 9. Diseño para la página de generación de reportes, aquí el usuario podrá realizar la búsqueda por año actual o cada uno de los doce meses del año. Generado en Balsamiq.

Arquitectura de software: Patrón Modelo, Vista y Controlador (MVC).

Es muy común la implementación de este patrón para el desarrollo de software cuando se trata de alguna aplicación web, esto se debe a que funciona muy bien en aquellas aplicaciones en donde se deben manejar interfaces de usuario y navegar entre las mismas. Para esto este patrón los separa de la siguiente forma, por un lado, van los datos y los modelos, por el otro nos encontramos con el controlador el cual es el encargado de relacionar estos dos.

En la aplicación es evidente que se utilizó este patrón, porque podemos observar que por un lado manejamos los datos y los modelos, para después poderlos relacionar por medio de controladores y en este apartado poderles brindar comportamientos y así obtener su correspondiente funcionamiento.

7.6 Implementación

En esta etapa es donde se realizará todo el proceso de codificación de la aplicación web, se programarán las interfaces en AngularJS, se consumirá el API rest obtenida en el punto anterior, se resaltarán las funcionalidades de la aplicación final, con la página web de la empresa donde se cargan actualmente las horas.

- Producto piloto final de la aplicación web.
- Comparativa entre el piloto final con la página web actual de la empresa.

8. RESULTADOS ESPERADOS:

- Análisis de la página actual para el cargue de horas.
- Product Backlog
- Sprint Backlog.
- Hoja de ruta.
- Informes de trabajo realizado.
- Informes de trabajo pendiente.
- Actas de trabajo.
- Requerimientos funcionales.
- Requerimientos no funcionales.
- Diagramas UML.
- El listado de las tecnologías que se usaran.
- Restricciones en base a las tecnologías elegidas.
- Descripción de la implementación de cada una de las herramientas.
- Aplicación Back-end.
- Servicios rest que consumirá el front.
- Colecciones de datos en MongoDB Atlas.
- Producto piloto final de la aplicación web.
- Comparativa entre el piloto final con la página web actual de la empresa.

9. RESULTADOS OBTENIDOS:

En esta parte del documento se verán reflejados los resultados obtenidos de todo el proceso descrito anteriormente, en cada uno de ellos se elaborará un corto análisis del resultado obtenido.

9.1 Análisis de la página actual de la empresa para el cargue de horas.

El primer paso antes de realizar todo el desarrollo de la aplicación fue realizar un análisis de la aplicación actual que se encarga de llevar el seguimiento de las horas en la empresa.

Al navegar en la aplicación se encontró que los componentes HTML utilizados son los de apariencia nativa, por lo que no se usa ningún estilo CSS para mejorar su apariencia.

En cuanto al funcionamiento solo permite navegar entre las semanas del mes actual, no tiene un apartado para ver las horas que se han registrado anteriormente.

Además, no cuenta con un apartado para que los usuarios puedan almacenar los códigos o nombres de los proyectos en los que se encuentran trabajando. No tiene en cuenta información de las actividades más allá del código y una breve descripción.

Por último, claramente no maneja un sistema de notificaciones automático y el correo que se mencionó al inicio del documento es enviado por un empleado que se encarga de realizar los reportes y llevar el seguimiento de las horas trabajadas.

9.2 Resultados obtenidos de la metodología Scrum

Product Backlog: Para la implementación de la metodología ágil de desarrollo Scrum, se utilizó Jira Software y en la siguiente imagen se puede observar el Product Backlog en la mitad del primer Sprint.

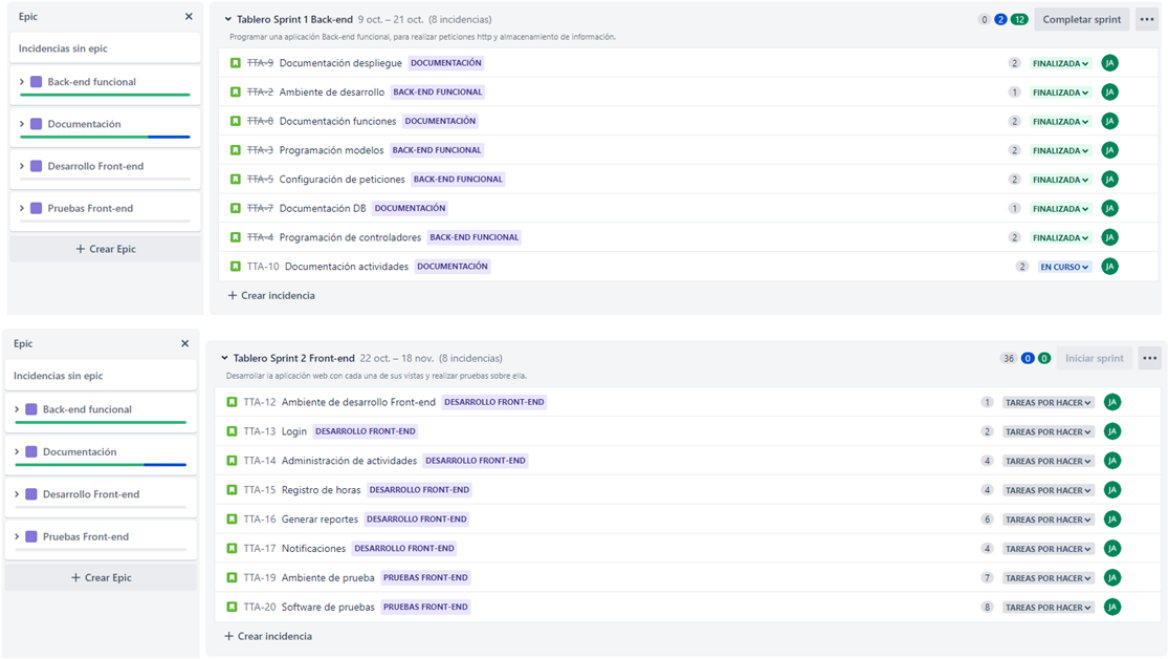


Figura 10. Product Backlog generado desde Jira Software. Generado en Jira Software.

Sprint Backlog: Se utilizó la herramienta de Jira Software para generar los Sprints Backlog, en el proyecto se generaron dos Sprint, a continuación, se podrá observar el resultado:

Sprint 1:

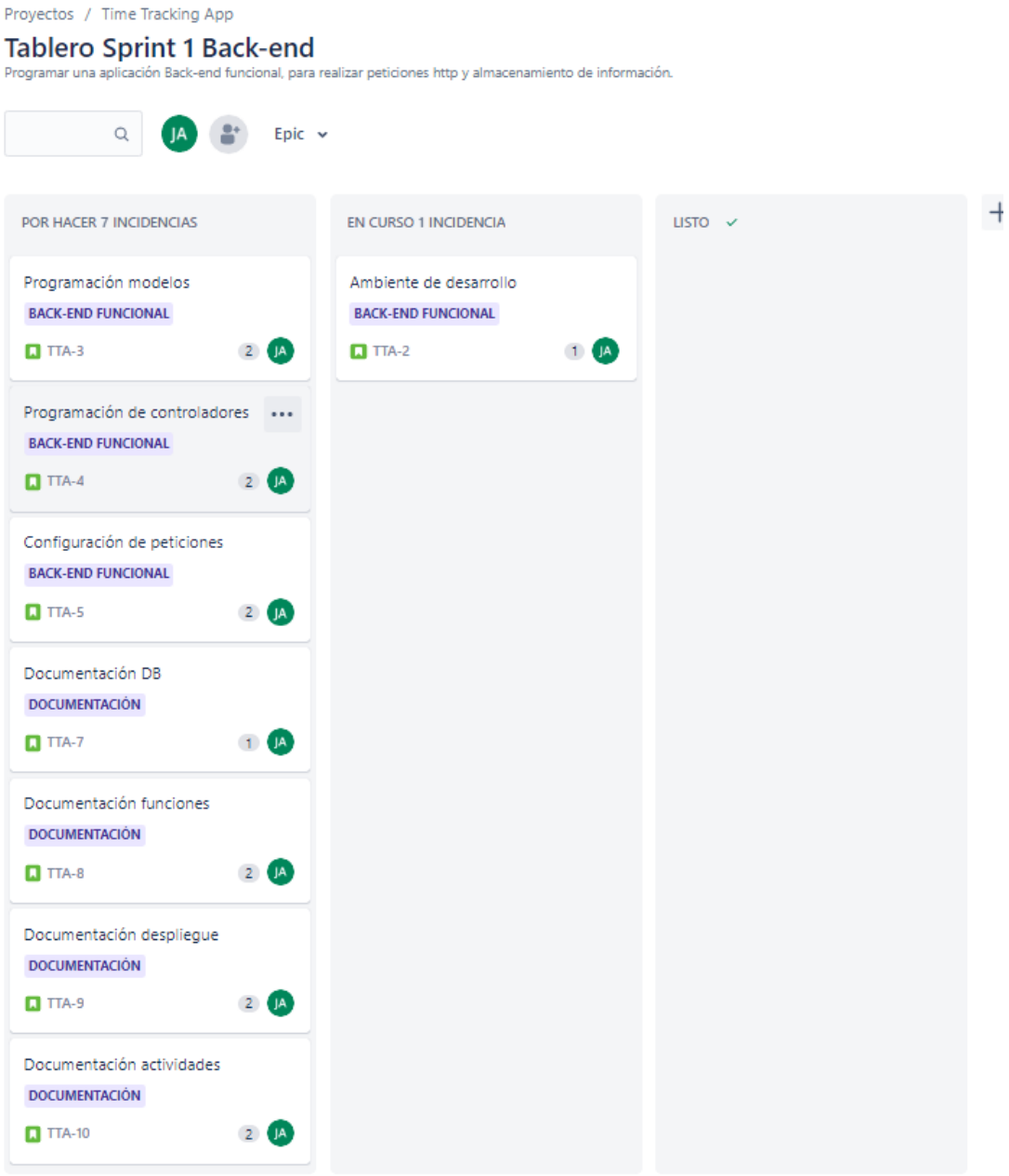


Figura 11. Sprint Backlog para el primer Sprint, donde se puede observar las tareas generadas con su respectiva estimación y estado. Generado en Jira Software.

Sprint 2:

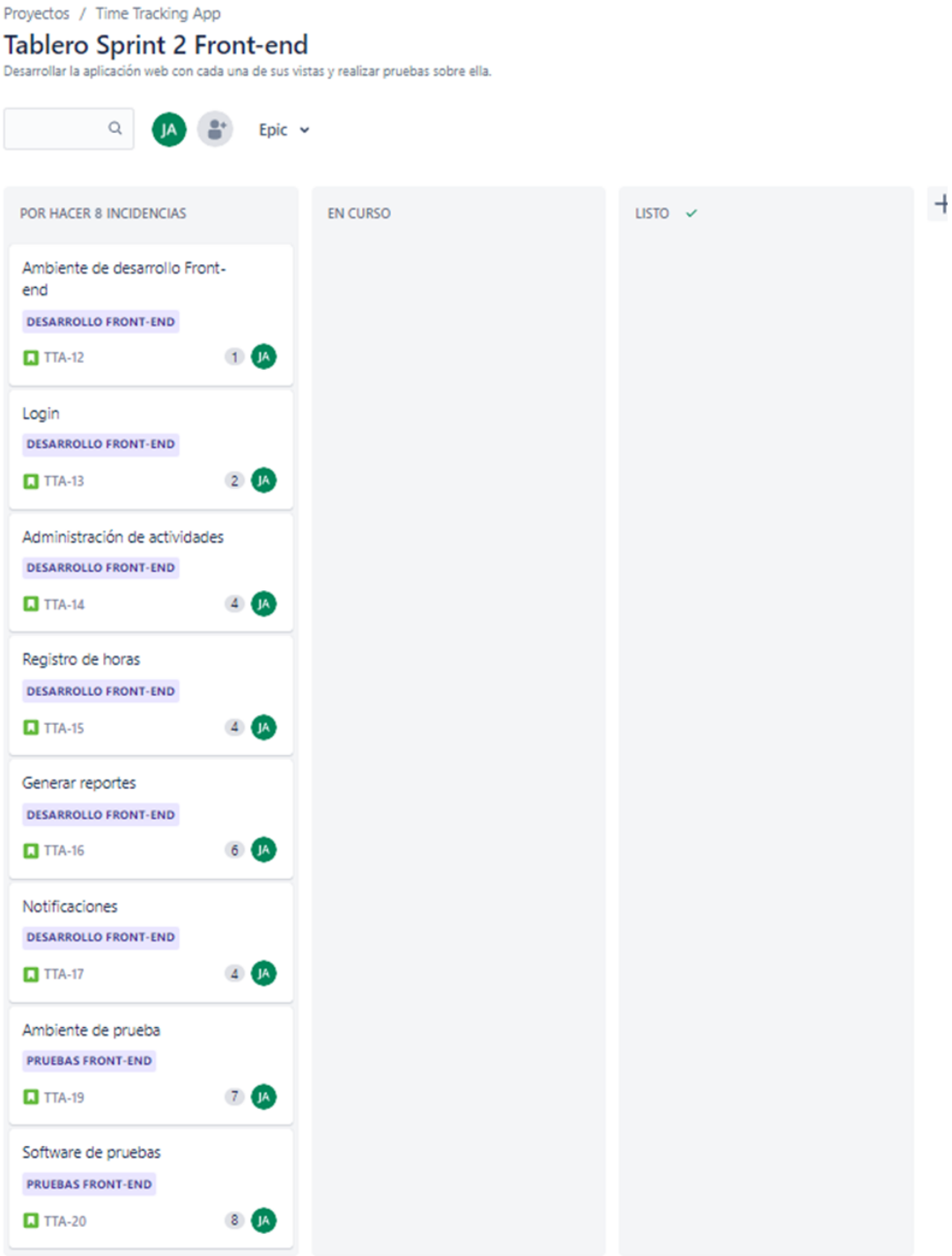


Figura 12. Sprint Backlog para el segundo Sprint, donde se puede observar las tareas generadas con su respectiva estimación y estado. Generado en Jira Software.

Hoja de ruta: En la hoja de ruta se ven reflejadas las tareas generadas para cada uno de los Sprints generados anteriormente teniendo en cuenta el inicio y el final de cada uno de ellos, como se observa en la siguiente imagen:

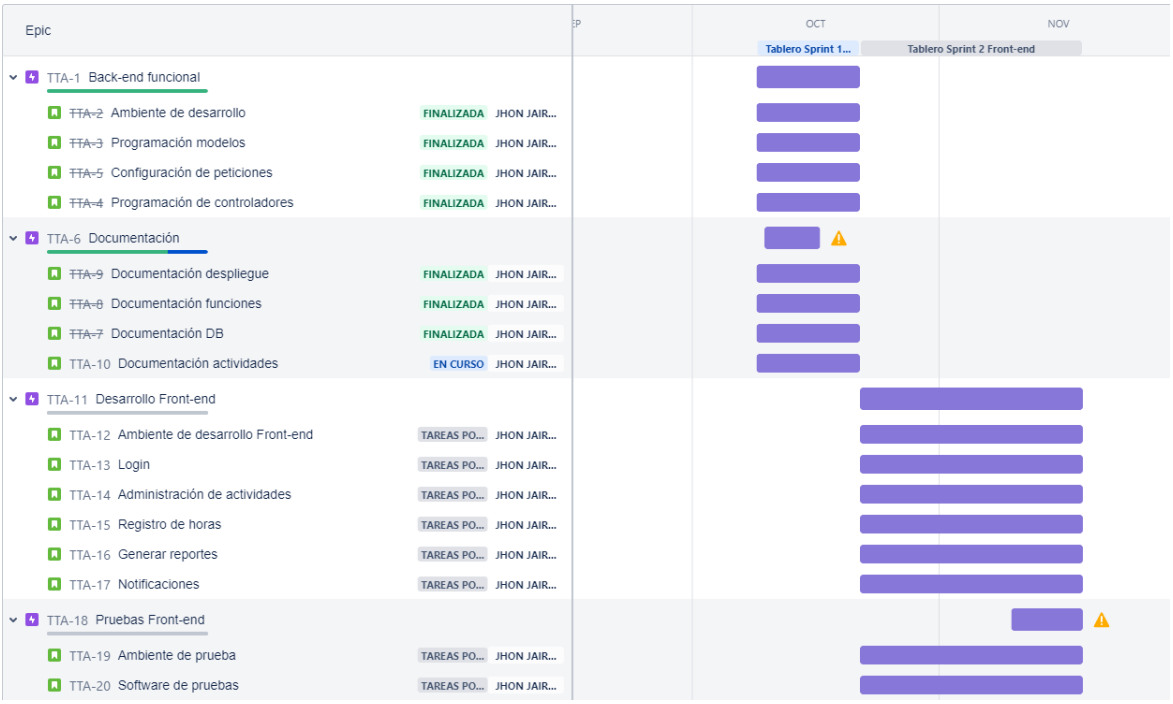


Figura 13. Este es un gráfico de hoja de ruta que brinda la herramienta de Jira Software en donde se ve reflejado las tareas creadas en cada épica teniendo presente las fechas y el estado de cada tarea. Generado en Jira Software.

Informes de trabajo realizado: En este gráfico que facilita la herramienta de Jira Software se puede observar cómo se ve reflejado la realización de las tareas definidas en el Sprint, contrastadas con el tiempo que se definió para el mismo.

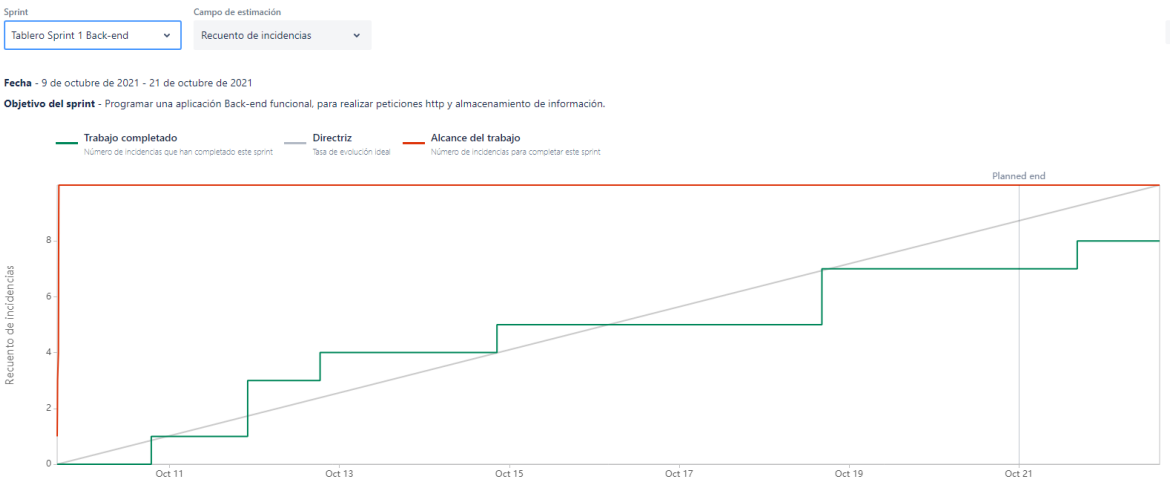


Figura 14. Gráfico de trabajo realizado del primer Sprint generado por Jira Software.

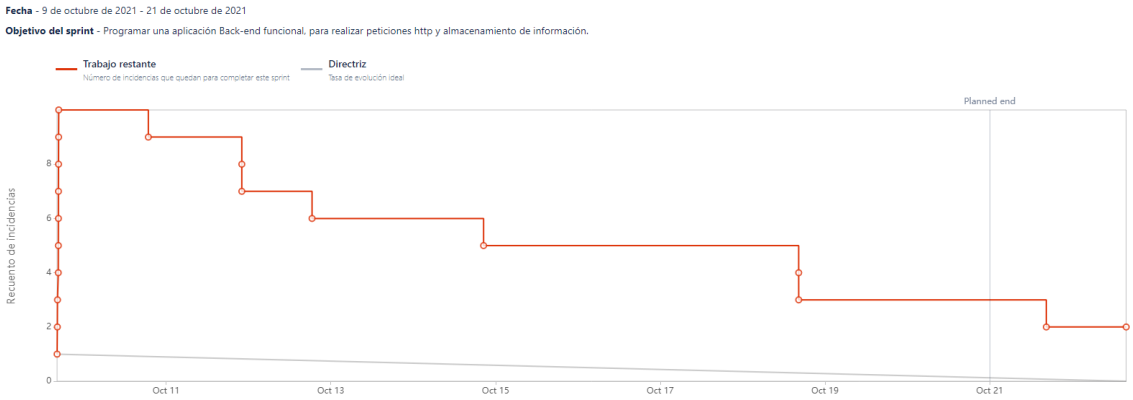


Figura 15. Gráfico de trabajo realizado del segundo Sprint generado por Jira Software.

Informes de trabajo pendiente: En estos gráficos se puede observar cómo se va disminuyendo la carga de tareas de un Sprint, reflejado a la fecha en la que se da por terminada una tarea.

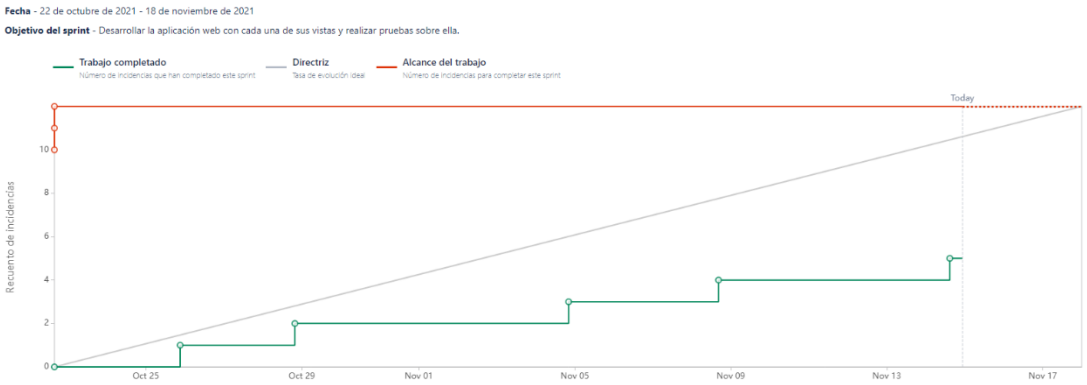


Figura 16. Gráfico de trabajo pendiente del primer Sprint generado por Jira Software.

Figura 17. Gráfico de trabajo pendiente del segundo Sprint generado por Jira Software.

9.3 Levantamiento de requerimientos

Requerimientos funcionales: En estos requerimientos se definieron las principales necesidades que deberá satisfacer la aplicación para cumplir con el objetivo de ser una aplicación web piloto para el cargue de horas en la empresa. Debería permitir registrar la información de los usuarios, tener un espacio para registrar actividades, registrar proyectos,

realizar el cargue de horas y un apartado completo para generar reportes de las horas cargadas por mes.

Tabla 1

Requerimiento funcional	
Número: 1	**Nombre:** Registro de usuario.
Descripción:	Yo como usuario quiero una interfaz para registrarme en la aplicación y almacenar mis datos personales junto a mis credenciales para el ingreso.
Observaciones:	Los datos ingresados por el usuario serán almacenados en la base de datos por medio del Back-end.

Nota: Requerimiento funcional 1 aplicación TimeTracking.

Requerimiento funcional	
Número: 2	**Nombre:** Autenticación de usuario.
Descripción:	Yo como usuario quiero una interfaz donde pueda ingresar mis credenciales y autenticarme en la aplicación para tener acceso a sus funcionalidades.
Observaciones:	Se realizará una consulta a la base de datos para comprobar que las credenciales ingresadas son correctas.

Nota: Requerimiento funcional 2 aplicación TimeTracking.

Requerimiento funcional	
Número: 3	**Nombre:** Vista de bienvenida.
Descripción:	Yo como usuario quiero una interfaz principal en la que encuentre la información relevante de la aplicación, junto a un apartado para registrarme o autenticarme en la aplicación.
Observaciones:	Esta será la página de bienvenida y contendrá todo el contenido que se considere relevante para la aplicación.

Nota: Requerimiento funcional 3 aplicación TimeTracking.

Requerimiento funcional	
Número: 4	**Nombre:** Registro de actividades.
Descripción:	Yo como usuario quiero una interfaz en donde pueda registrar mis actividades recurrentes y se almacenen con nombre, código y una descripción.
Observaciones:	El usuario tendrá libertad total, para asignar el código, el nombre y la descripción.

Nota: Requerimiento funcional 4 aplicación TimeTracking.

Requerimiento funcional	
Número: 5	**Nombre:** Registro de proyectos.

Descripción:	Yo como usuario quiero una interfaz en donde pueda registrar mis proyectos en los que me encuentro trabajando actualmente en donde pueda almacenar el nombre y el código de este.
Observaciones:	El usuario tendrá libertad total, para asignar el código y el nombre.

Nota: Requerimiento funcional 5 aplicación TimeTracking.

Requerimiento funcional	
Número: 6	**Nombre:** Registro de horas laboradas.
Descripción:	Yo como usuario quiero una interfaz en donde pueda registrar mis horas trabajadas, en donde pueda consultar la actividad y el proyecto en el que se invirtieron estas.
Observaciones:	Se realizará una consulta de los proyectos y actividades correspondientes al usuario autenticado y al ser seleccionadas se le compartirá la información al registro.

Nota: Requerimiento funcional 6 aplicación TimeTracking.

Requerimiento funcional	
Número: 7	**Nombre:** Consultar registros de horas.
Descripción:	Yo como usuario quiero una interfaz en donde pueda consultar mis horas registradas, para poder evidenciar, fecha, nombre, actividad y proyecto.

Observaciones:	Para esta aplicación piloto será una consulta plana de los datos y se implementaran filtros de búsqueda.

Nota: Requerimiento funcional 7 aplicación TimeTracking.

Requerimiento funcional	
Número: 8	**Nombre:** Filtrar por meses los registros de horas.
Descripción:	Yo como usuario quiero una interfaz en donde pueda consultar mis horas registradas u filtrarlas por cada mes trabajado.
Observaciones:	Para esta aplicación piloto será una consulta plana de los datos y se implementaran filtros de búsqueda por mes.

Nota: Requerimiento funcional 8 aplicación TimeTracking.

Requerimiento funcional	
Número: 9	**Nombre:** Sistema de notificaciones.
Descripción:	Yo como usuario quiero recibir notificaciones para recordarme registrar mis horas laborales de ese día.
Observaciones:	Se enviará una notificación al correo electrónico registrado para recordarle al usuario cargas sus horas trabajadas.

Nota: Requerimiento funcional 9 aplicación TimeTracking.

Requerimientos no funcionales:

Requerimiento NO funcional	
Descripción:	Los datos beberán ser almacenados en la base de datos de manera correcta como son ingresados en la interfaz por el usuario.
Observaciones:	Es importante que la información almacenada por el usuario sea la misma que consulte para generar registros y reportes de horas.

Nota: Requerimiento NO funcional 1 aplicación TimeTracking.

Requerimiento NO funcional	
Descripción:	La codificación de la aplicación deberá soportar cambios en su Frontend y en su Backend.
Observaciones:	Al ser un desarrollo piloto, deberá quedar totalmente abierto para futuras modificaciones.

Nota: Requerimiento NO funcional 2 aplicación TimeTracking.

Requerimiento NO funcional	
Descripción:	La aplicación web será soportada por los principales navegadores web existentes.

Observaciones:	La aplicación podrá ser ejecutada en los navegadores web que soporten aplicaciones desarrolladas con AngularJS.

Nota: Requerimiento NO funcional 3 aplicación TimeTracking.

Requerimiento NO funcional	
Descripción:	La aplicación web será soportada por los principales navegadores web existentes.
Observaciones:	La aplicación podrá ser ejecutada en los navegadores web que soporten aplicaciones desarrolladas con AngularJS.

Nota: Requerimiento NO funcional 4 aplicación TimeTracking.

Requerimiento NO funcional	
Descripción:	La aplicación web no requerirá ser instalada en un equipo de cómputo.
Observaciones:	Al ser una aplicación web, lo ideal es que se pueda ejecutar en el navegador web desde que se acceda a ella.

Nota: Requerimiento NO funcional 5 aplicación TimeTracking.

Requerimiento NO funcional	
Descripción:	La aplicación web deberá ser responsive para adaptarse a vistas desde PC y dispositivos móviles.

Observaciones:	Sus vistas deberán ser dinámicas y no arruinarse al modificarse el tamaño del navegador.

Nota: Requerimiento NO funcional 5 aplicación TimeTracking.

Diagramas UML:

Diagrama de clases:

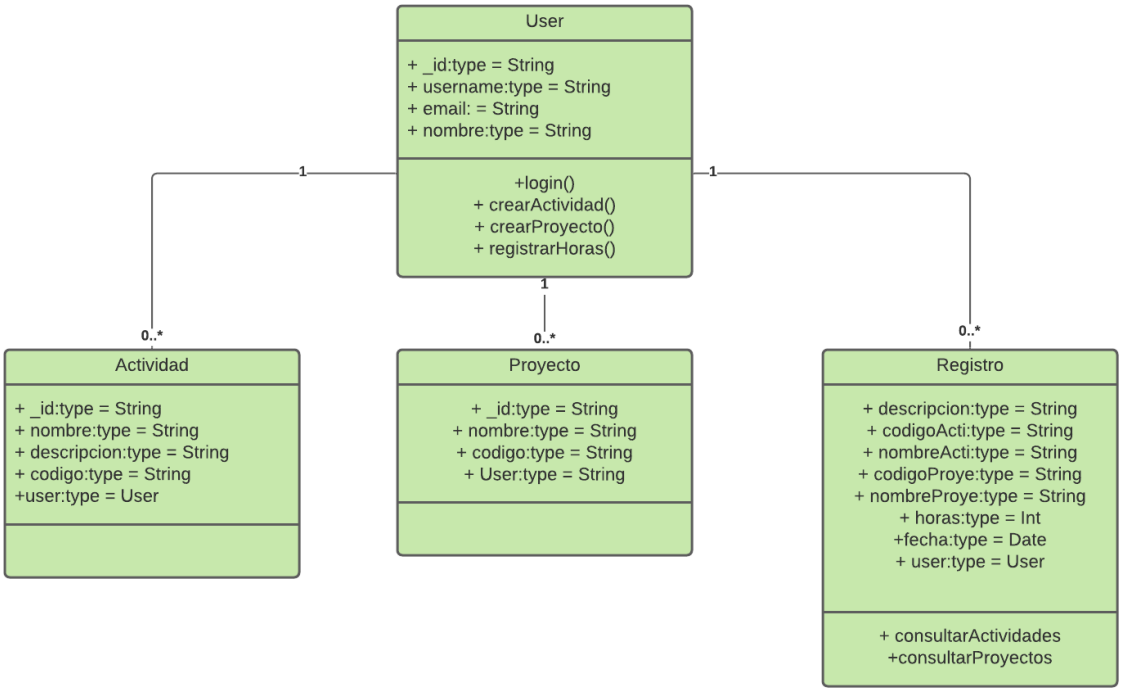


Figura 18. Diagrama de clases TimeTracking. Generado en Lucidchart.

Diagrama de componentes:

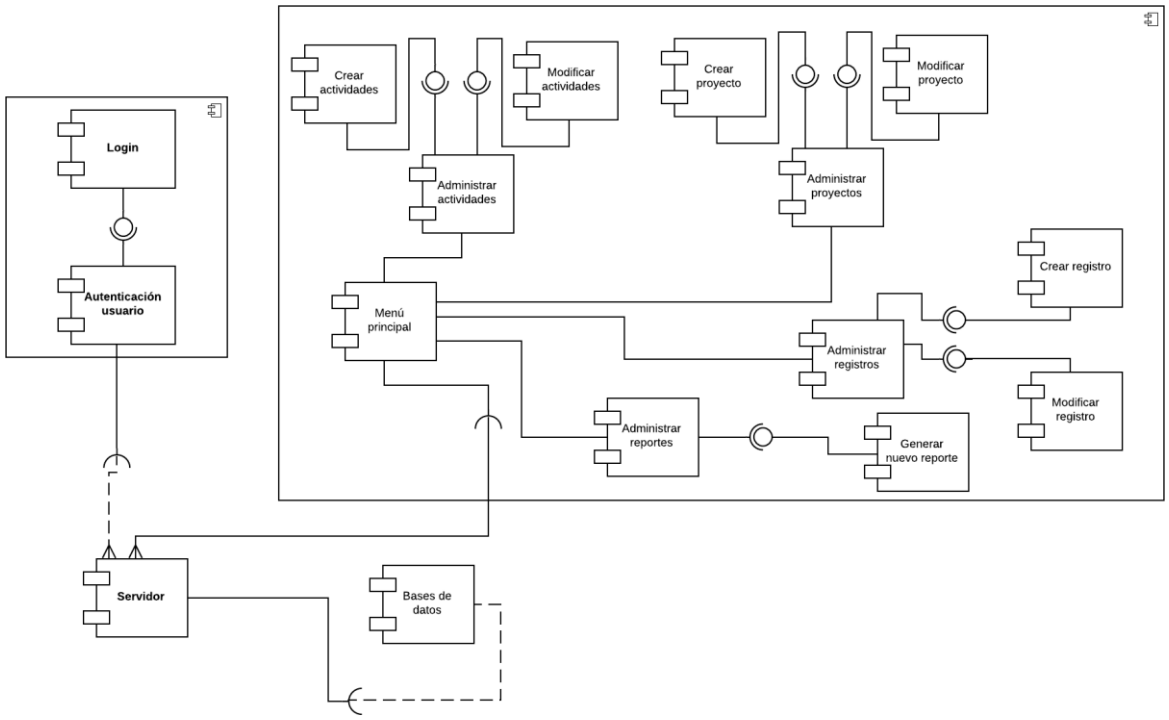


Figura 19. Diagrama de componentes TimeTracking. Generado en Lucidchart.

Diagrama de despliegue:

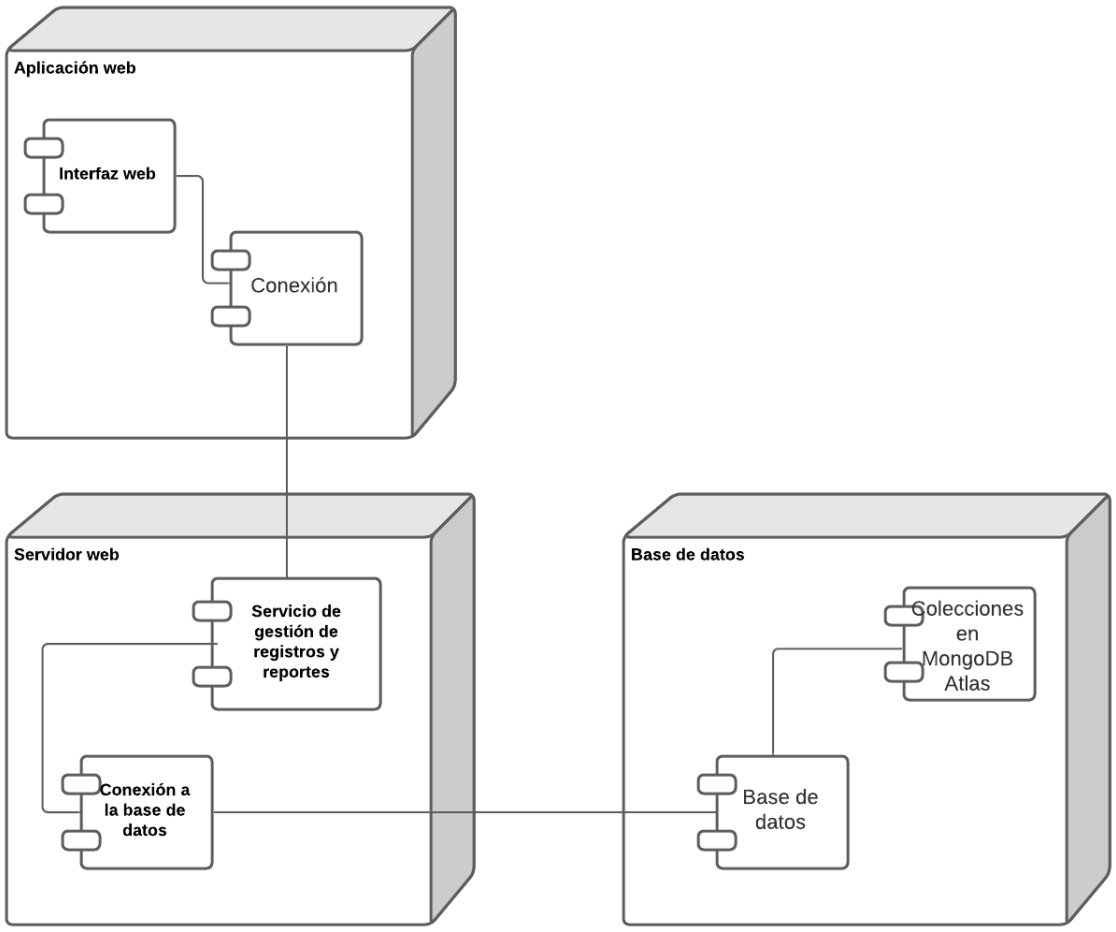


Figura 20. Diagrama de despliegue Time Tracking. Generado en Lucidchart.

Diagrama de actividades:

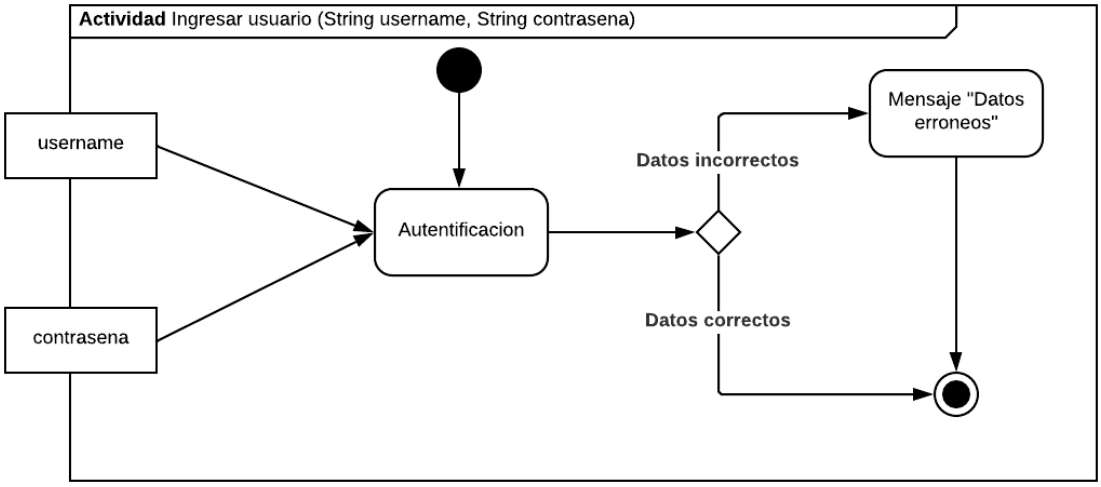


Figura 21. Diagrama de actividades “Ingresar usuario” para TimeTracking. Generado en Lucidchart.

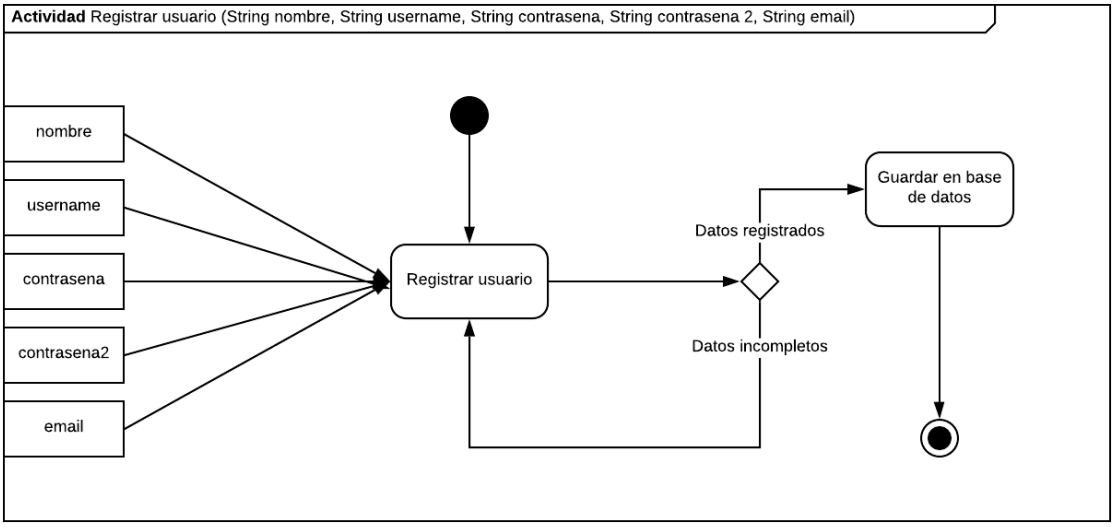


Figura 22. Diagrama de actividades “Registrar usuario” para TimeTracking. Generado en Lucidchart.

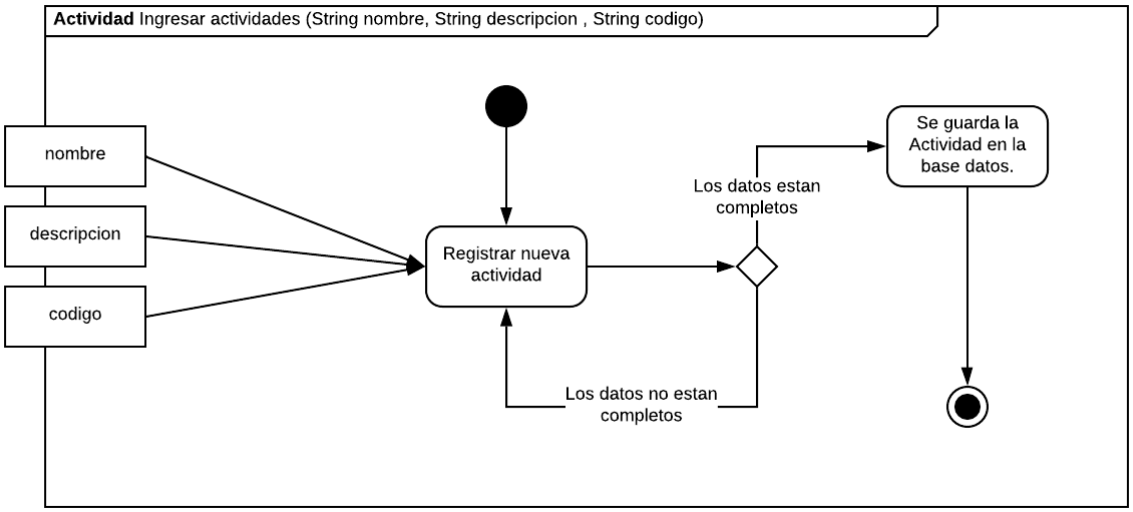


Figura 23. Diagrama de actividades “Ingresar actividad” para TimeTracking. Generado en Lucidchart.

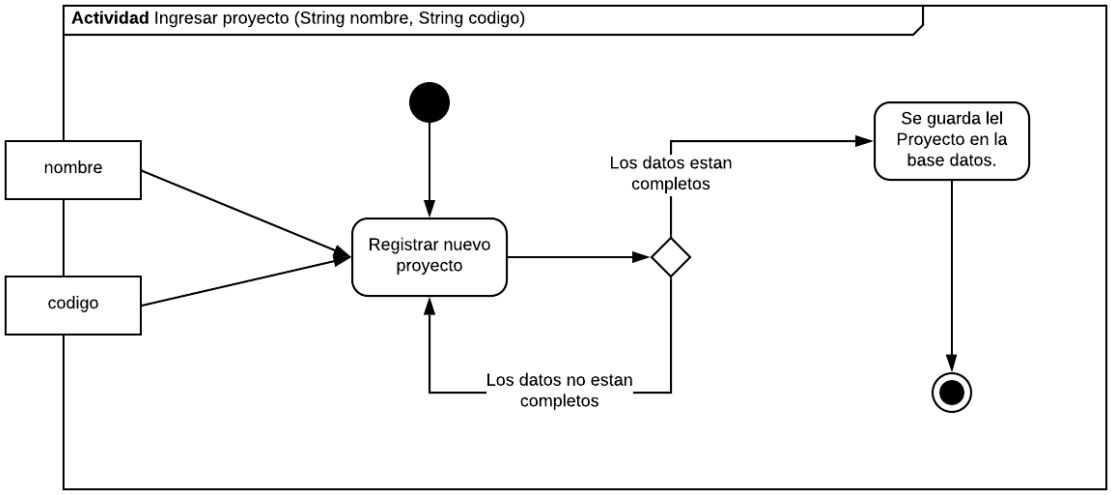


Figura 24. Diagrama de actividades “Ingresar proyecto” para TimeTracking. Generado en Lucidchart.

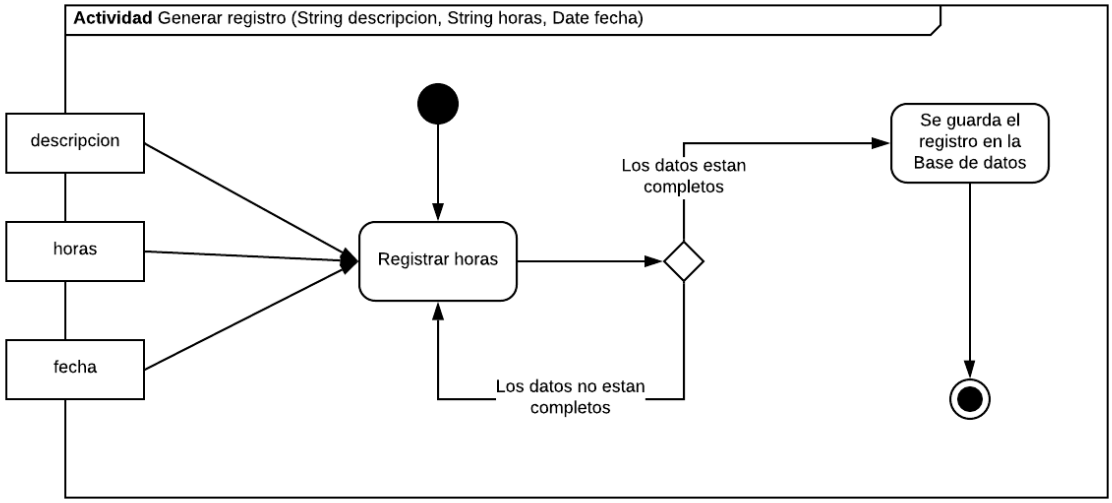


Figura 25. Diagrama de actividades "Generar registro" para TimeTracking. Generado en Lucidchart.

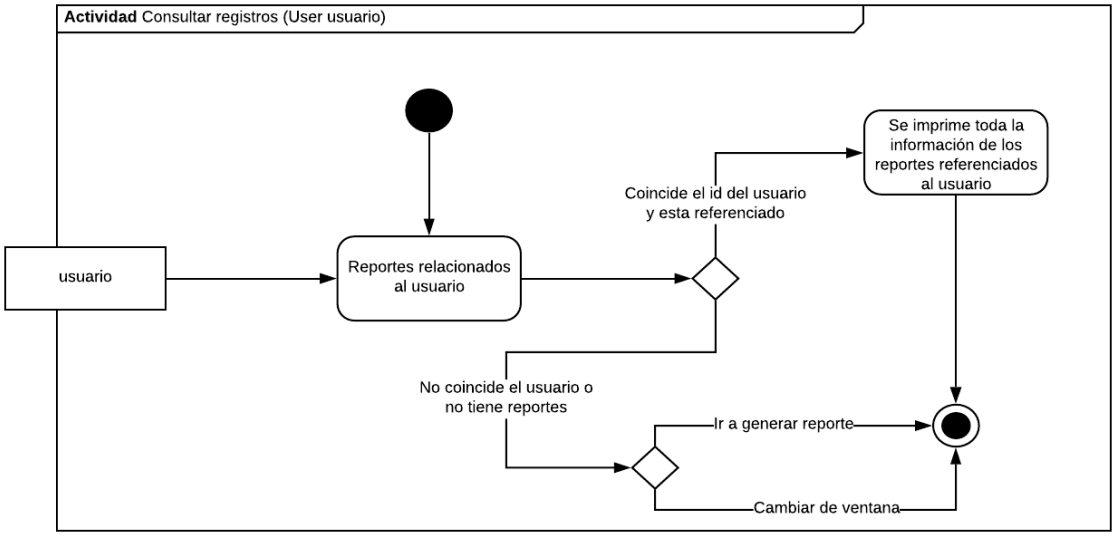


Figura 26. Diagrama de actividades "Consultar registro" para TimeTracking. Generado en Lucidchart.

Diagrama de secuencia:

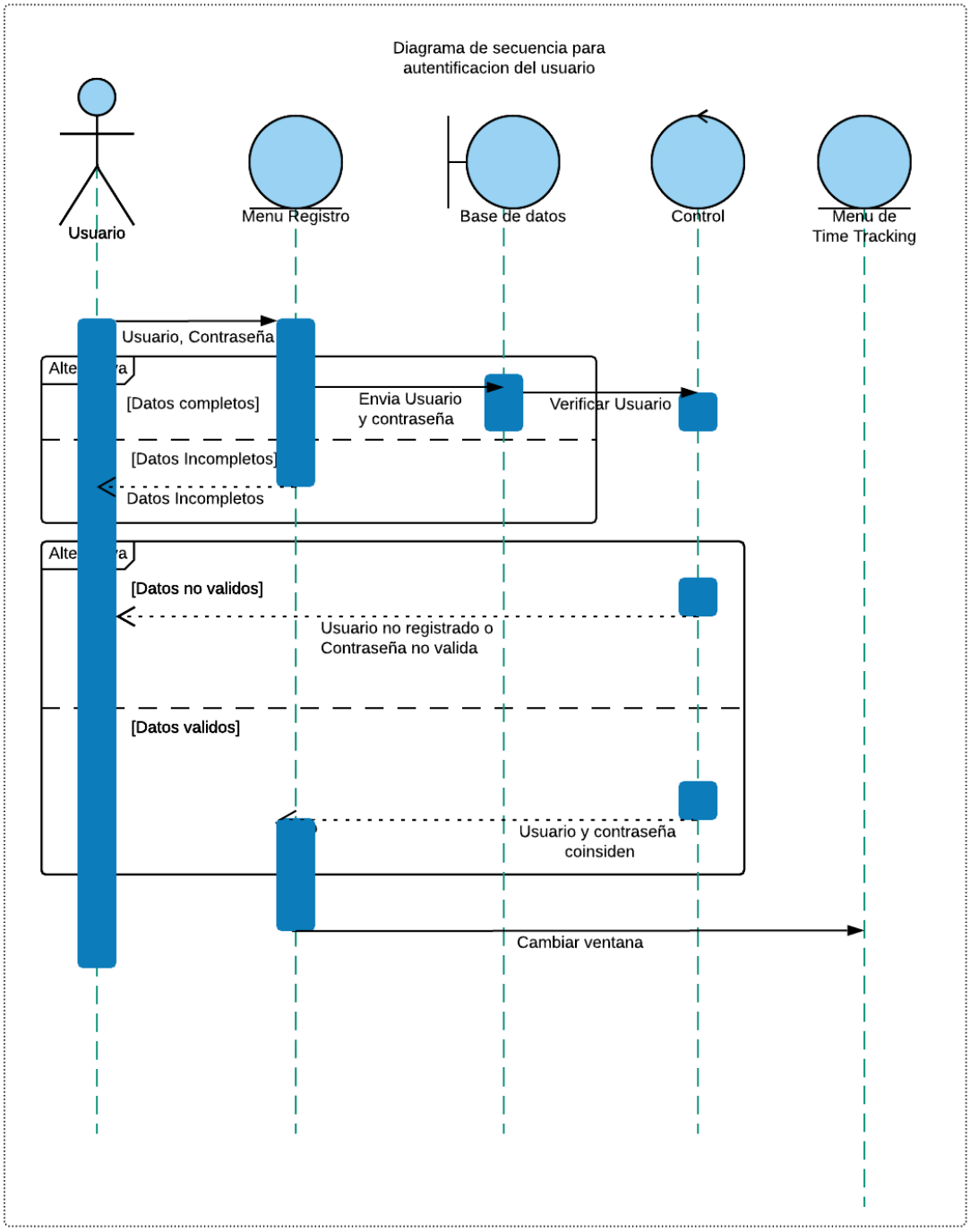


Figura 27. Diagrama de secuencia “Autenticación de usuario” para TimeTracking. Generado en Lucidchart.

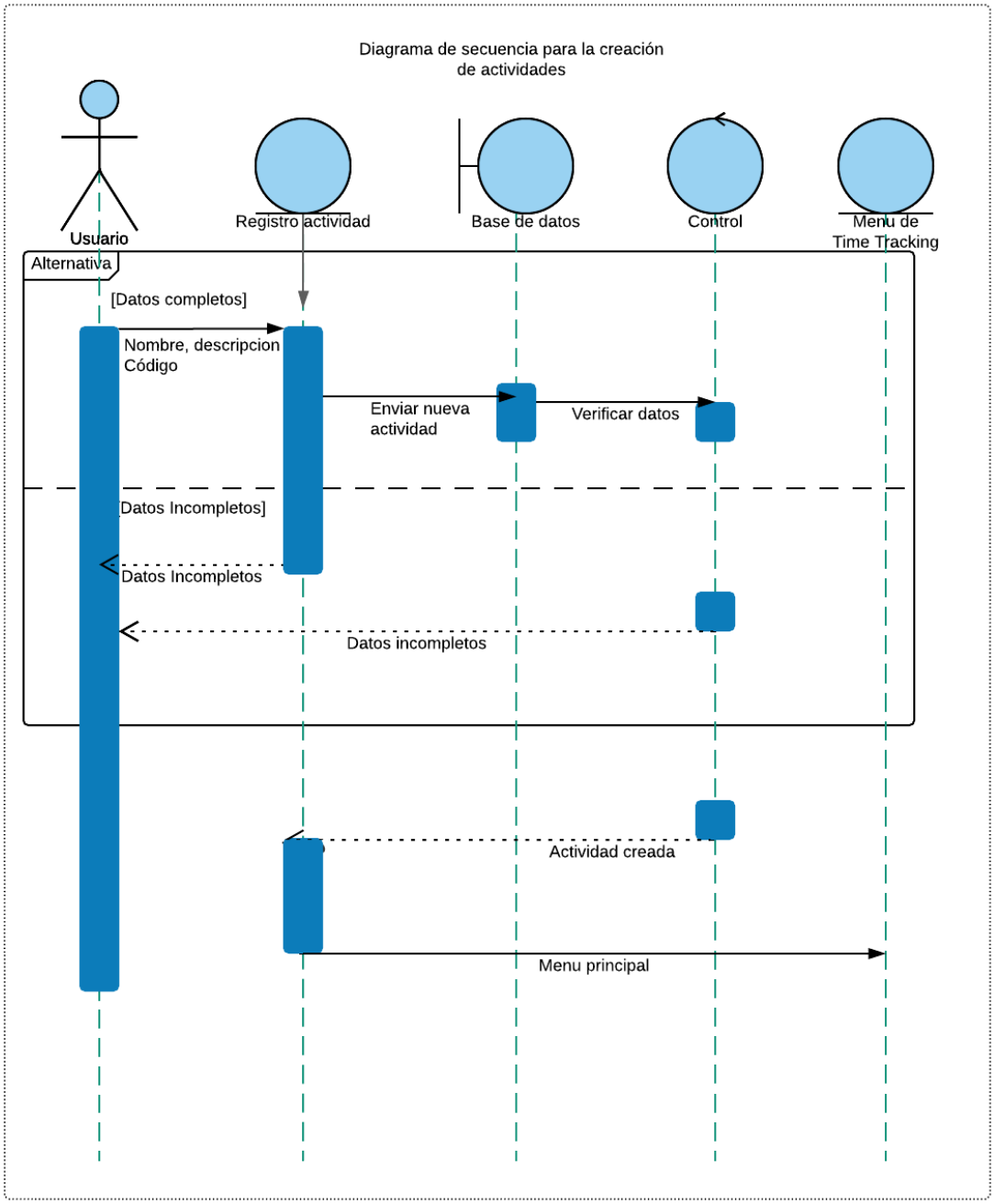


Figura 28. Diagrama de secuencia “Creación de actividad” para TimeTracking. Generado en Lucidchart.

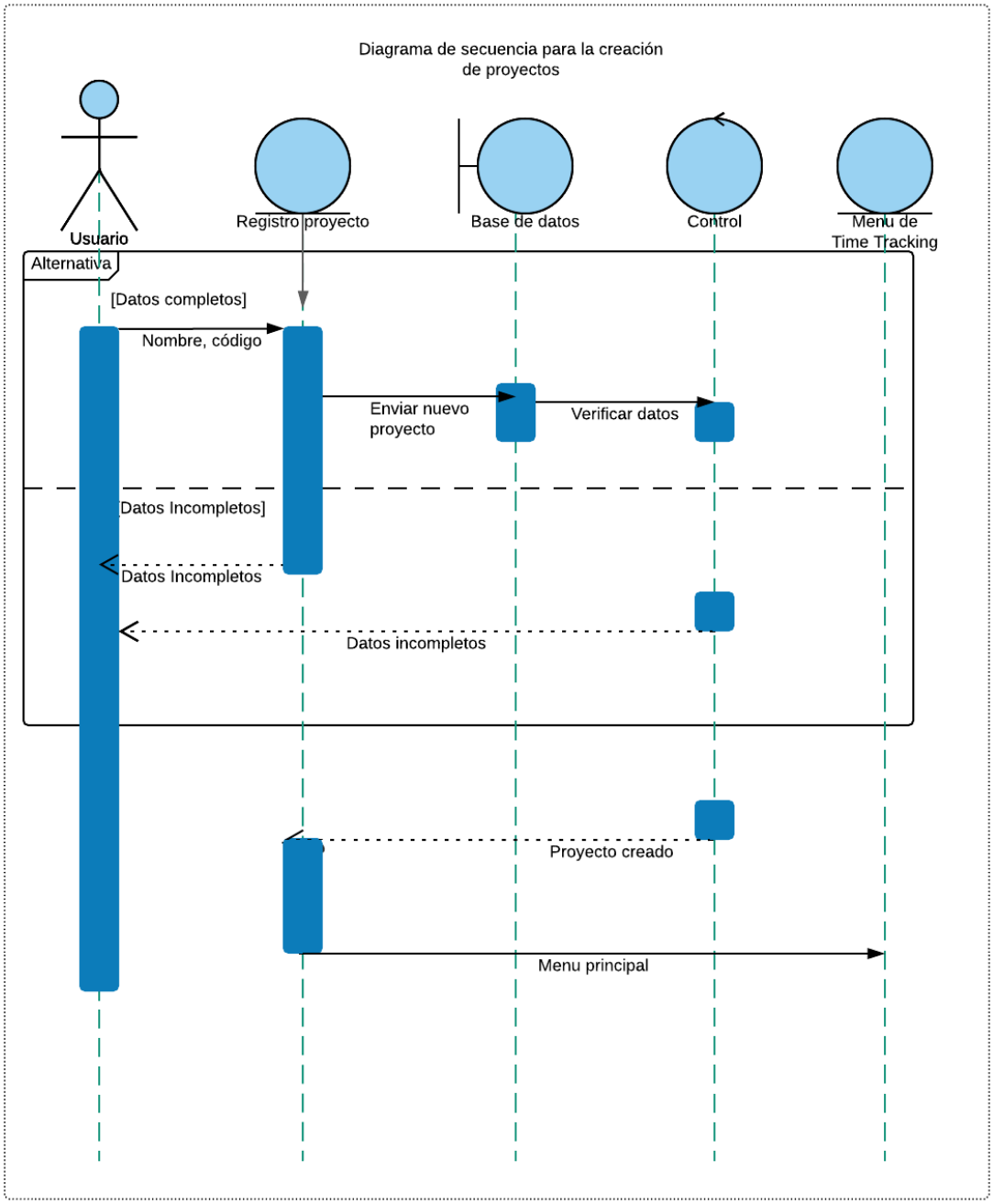


Figura 29. Diagrama de secuencia “Creación de proyecto” para TimeTracking. Generado en Lucidchart.

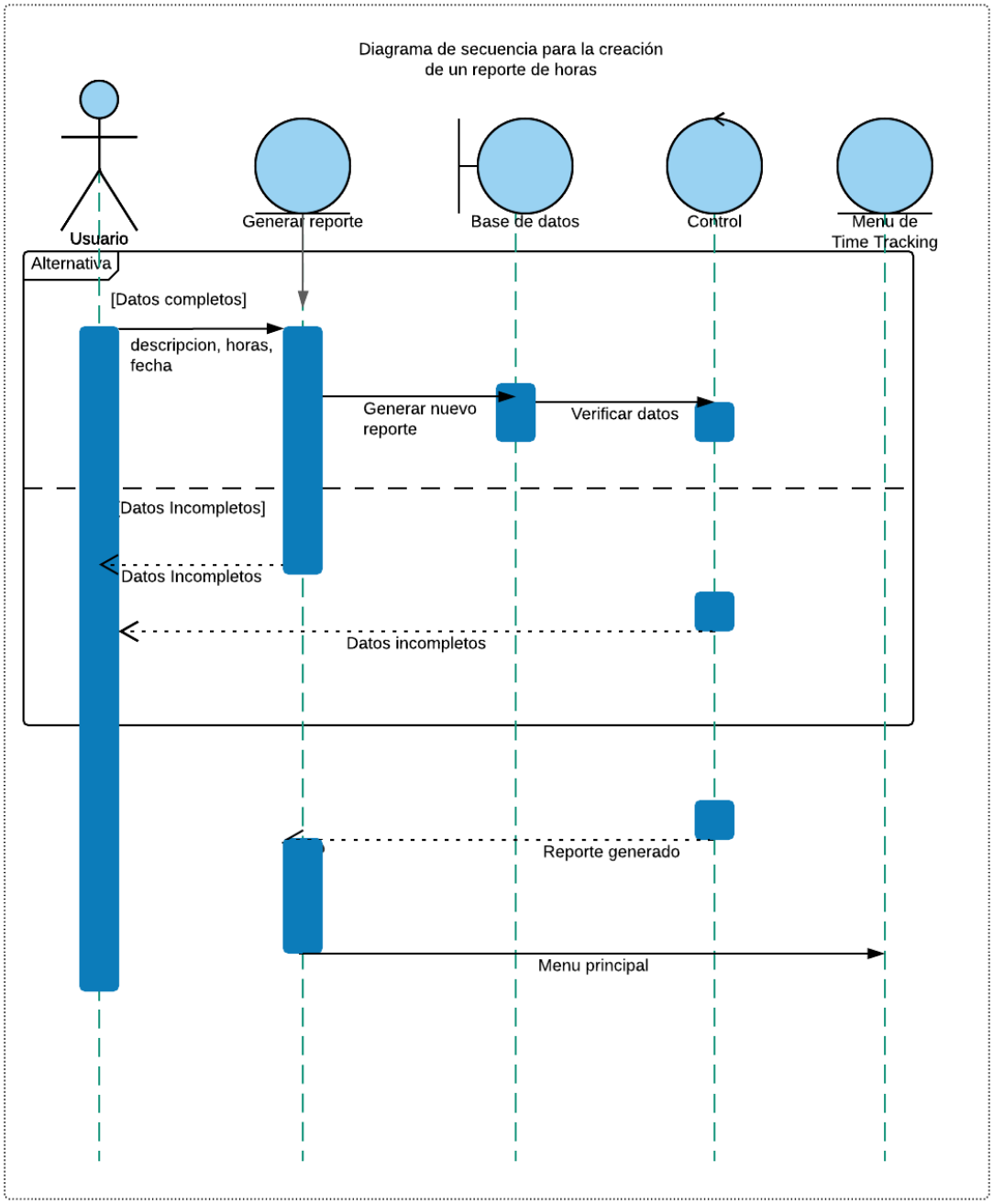


Figura 30. Diagrama de secuencia “Generar un reporte” para TimeTracking. Generado en Lucidchart.

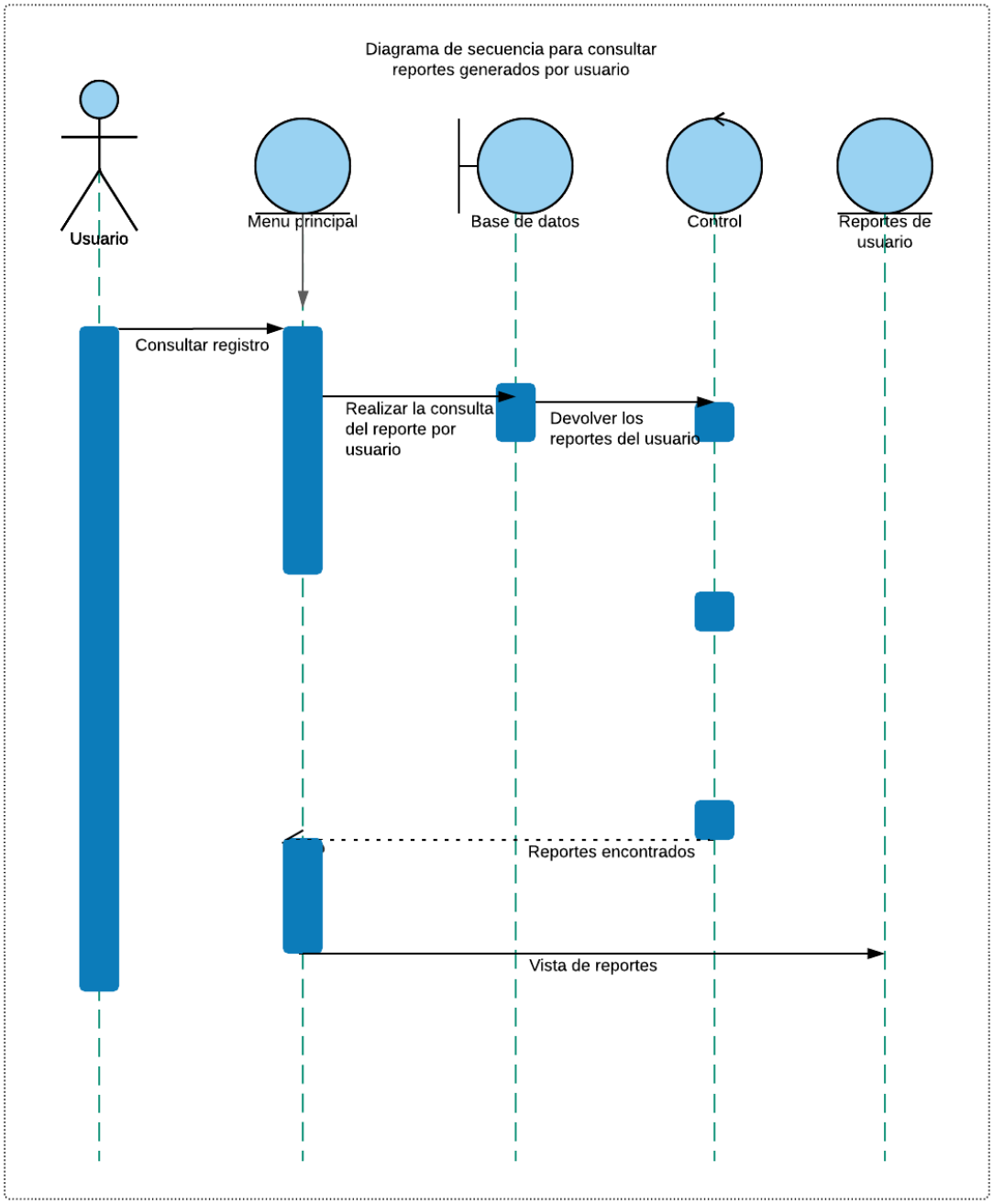


Figura 31. Diagrama de secuencia “Consultar un reporte” para TimeTracking. Generado en Lucidchart.

Diagrama de estados:

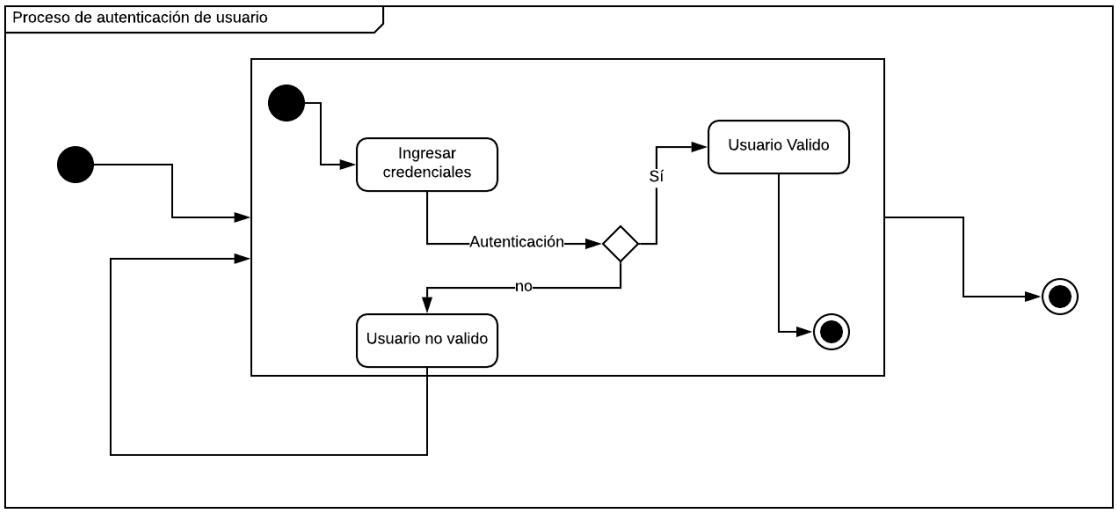


Figura 32. Diagrama de estados “Autenticación de usuario” para TimeTracking. Generado en Lucidchart.

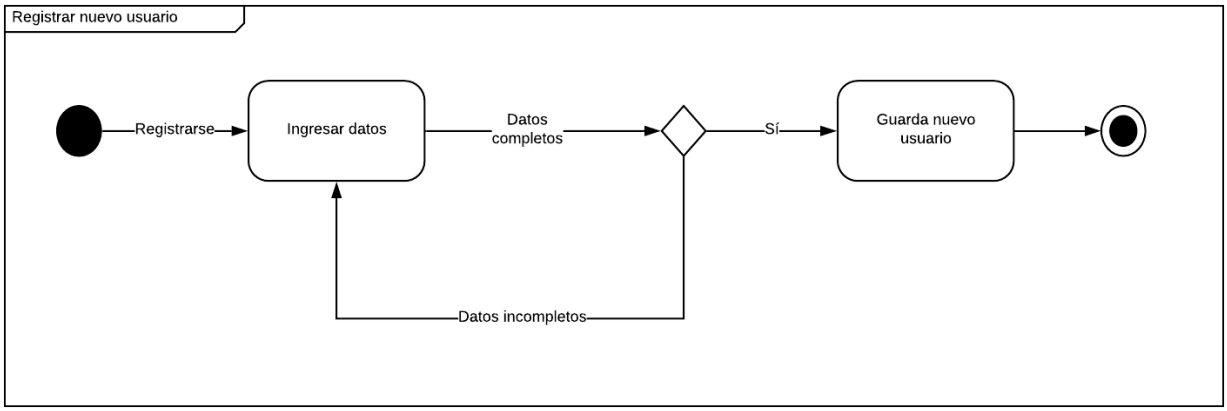


Figura 33. Diagrama de estados “Registrar nuevo usuario” para TimeTracking. Generado en Lucidchart.

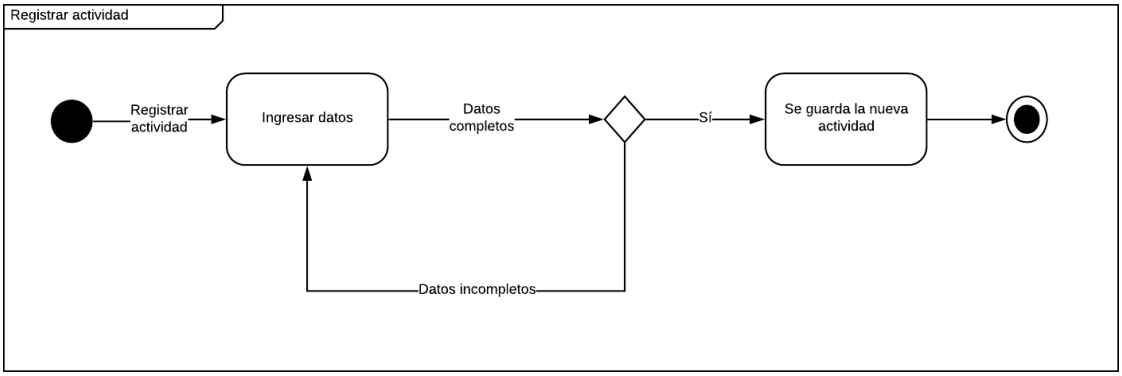


Figura 34. Diagrama de estados “Registrar nueva actividad” para TimeTracking. Generado en Lucidchart.

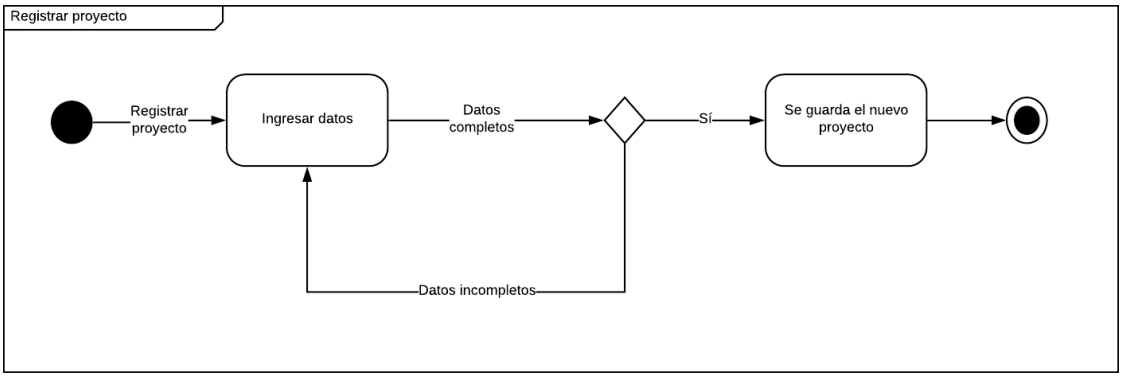


Figura 35. Diagrama de estados “Registrar nuevo proyecto” para TimeTracking. Generado en Lucidchart.

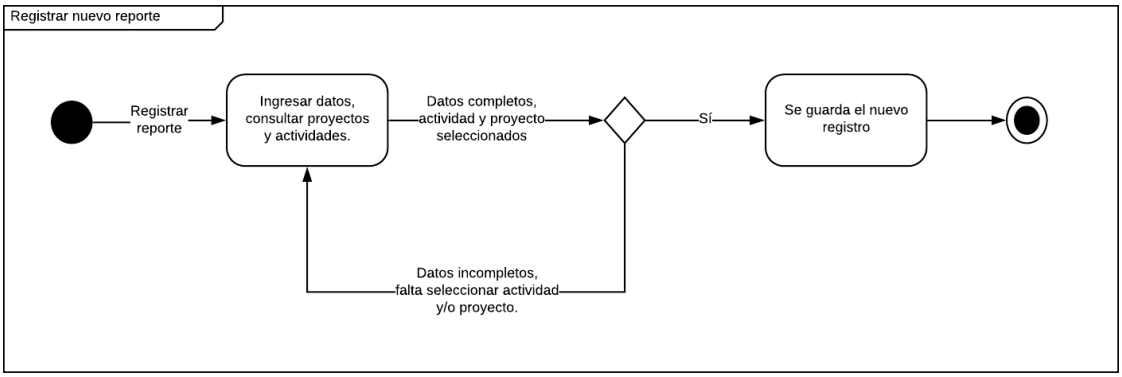


Figura 36. Diagrama de estados “Registrar nuevo reporte” para TimeTracking. Generado en Lucidchart.

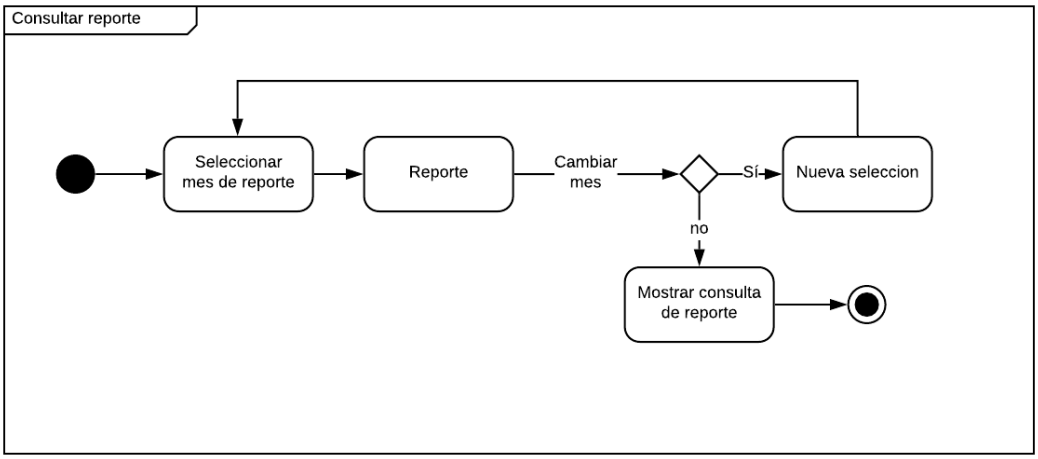


Figura 37. Diagrama de estados "consultar reporte" para TimeTracking. Generado en Lucidchart.

Diagrama de casos de uso:

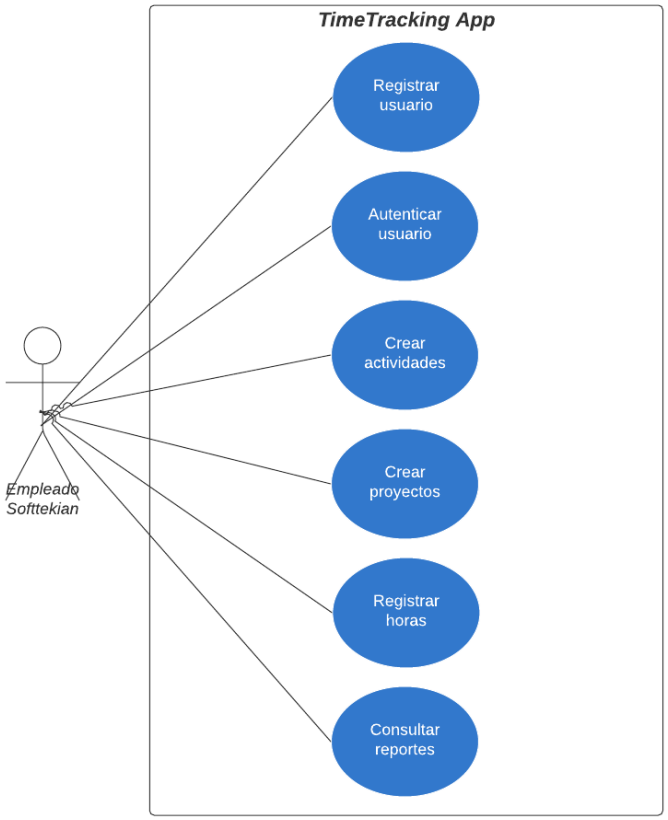


Figura 38. Diagrama de casos de uso para TimeTracking. Generado en Lucidchart.

9.4 Selección de tecnologías para el desarrollo

Listado de tecnologías que se usaron:

- Node.js
- Express.js
- Mongoose (ORM Para el Back-end)
- Mongo DB Atlas
- AngularJS
- Angular Material
- PrimeNG
- Talent API Tester
- Visual Studio Code
- Google Chrome
- Cors
- Npm
- Nodemailer

Descripción de la implementación de cada una de las herramientas:

Node.js, Express.js, Monfoose, Mongo DB Atlas, cors y npm: Las sensaciones al usar estas herramientas en general fueron bastante buenas, con ellas se desarrolló la aplicación de Back-end en formato de API rest y el resultado fue totalmente funcional. Una pequeña observación es que aparte de la documentación no se encuentra mucha información relevante en internet. Además, Mongo DB Atlas ofrece un ambiente cloud para almacenar las colecciones de la base de datos con tiempos de respuesta relativamente rápidos y lo más importante funcionales.

AngularJS, Angular Material, y PrimeNG: Para satisfacer las necesidades planteadas en la aplicación de Front-end, cumplieron exitosamente el propósito estas

tecnologías, permitieron usar sus complementos y sus funcionalidades para cada una de las vistas obtenidas.

9.5 Diseño

Aplicación Back-end: Como evidencia de la aplicación Back-end en la siguiente imagen se pueden evidenciar los Script.js que se codificaron para el funcionamiento de esta, allí podemos observar los modelos de la base de datos donde se encuentran los atributos de cada uno de ellos, los controladores en donde se encuentra la lógica implementada sobre cada modelo. También se puede observar las rutas definidas para las peticiones HTTP. Por último, un Script con la configuración de la base de datos, en donde se codifica la conexión con la base de datos en mongo Atlas.

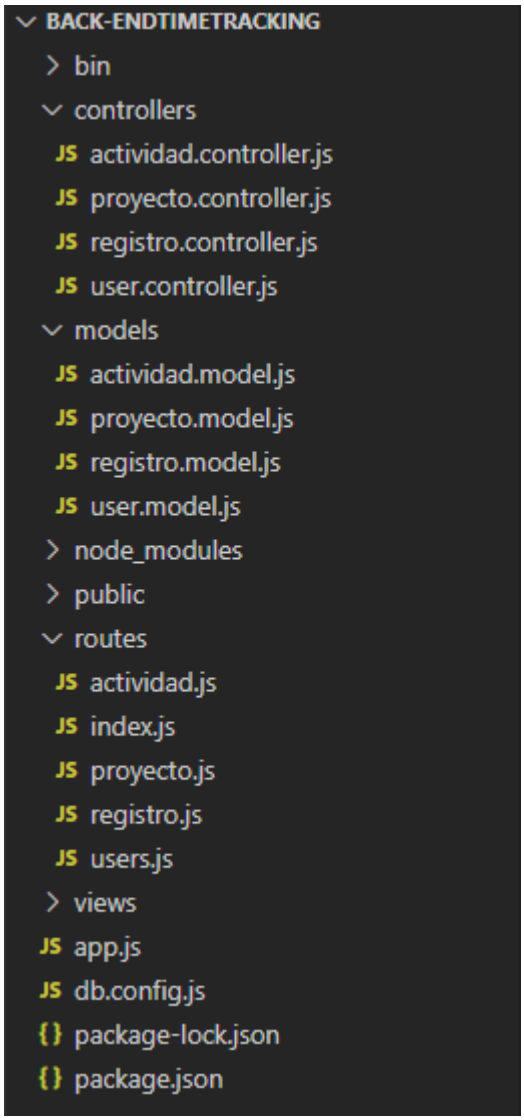


Figura 39. Vista de los Scripts codificados en la aplicación Back-end. Generado en Visual Studio Code.

De la aplicación Back-end también se puede mostrar las dependencias de Node.js, en la imagen se puede observar el nombre de la aplicación, el cors, express y moongose.

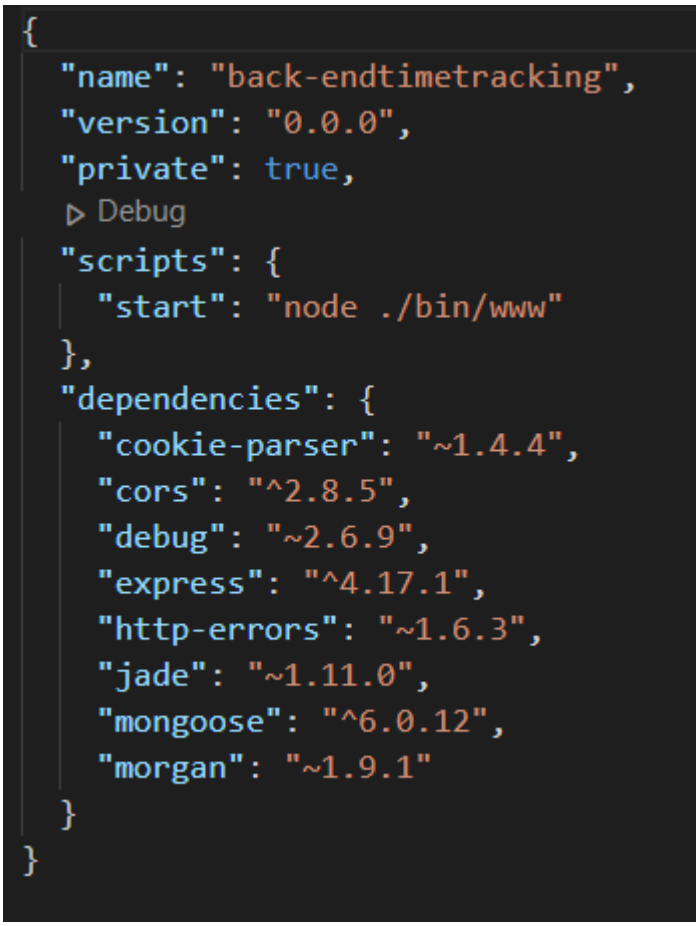


Figura 40. Vista de las dependencias utilizadas en Node.js. Generado en Visual Studio Code.

Servicios REST que consumirá el front: Ahora se puede observar cada una de las peticiones REST que se configuración para ser utilizadas por el Front-end, para ello se utilizó la herramienta tecnología Talent API Tester, para visualizar la respuesta del back el enviar una petición y el contenido de la petición en caso de ser una de tipo GET.

Peticiones para el modelo “User”:

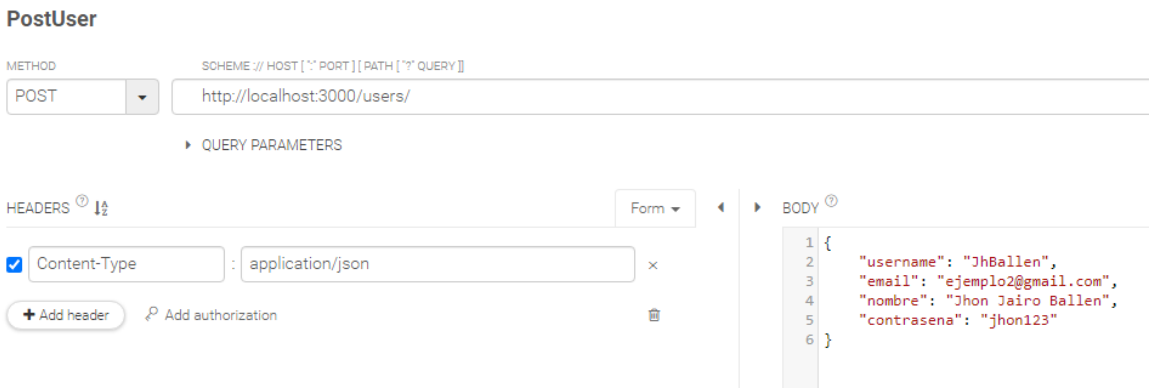


Figura 41. Petición HTTP tipo POST, para la creación de un nuevo usuario. Generado en Talend API Tester.

Figura 42. Respuesta del Back-end al realizar la petición HTTP. Generado en Talend API Tester.

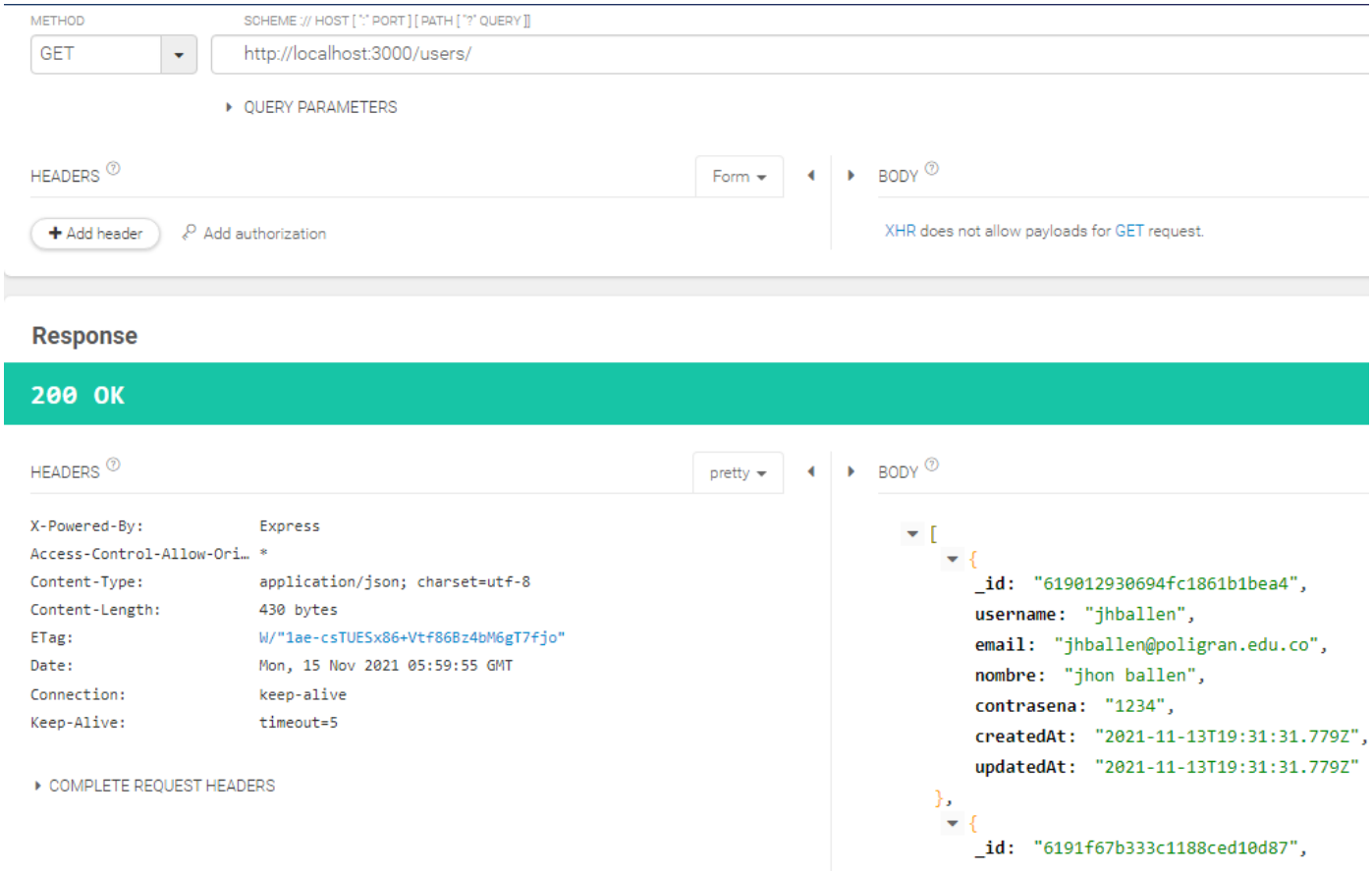


Figura 43. Petición HTTP tipo GET, para consultar todos los usuarios registrados en la aplicación, junto a la respuesta del Back-end. Generado en Talend API Tester.

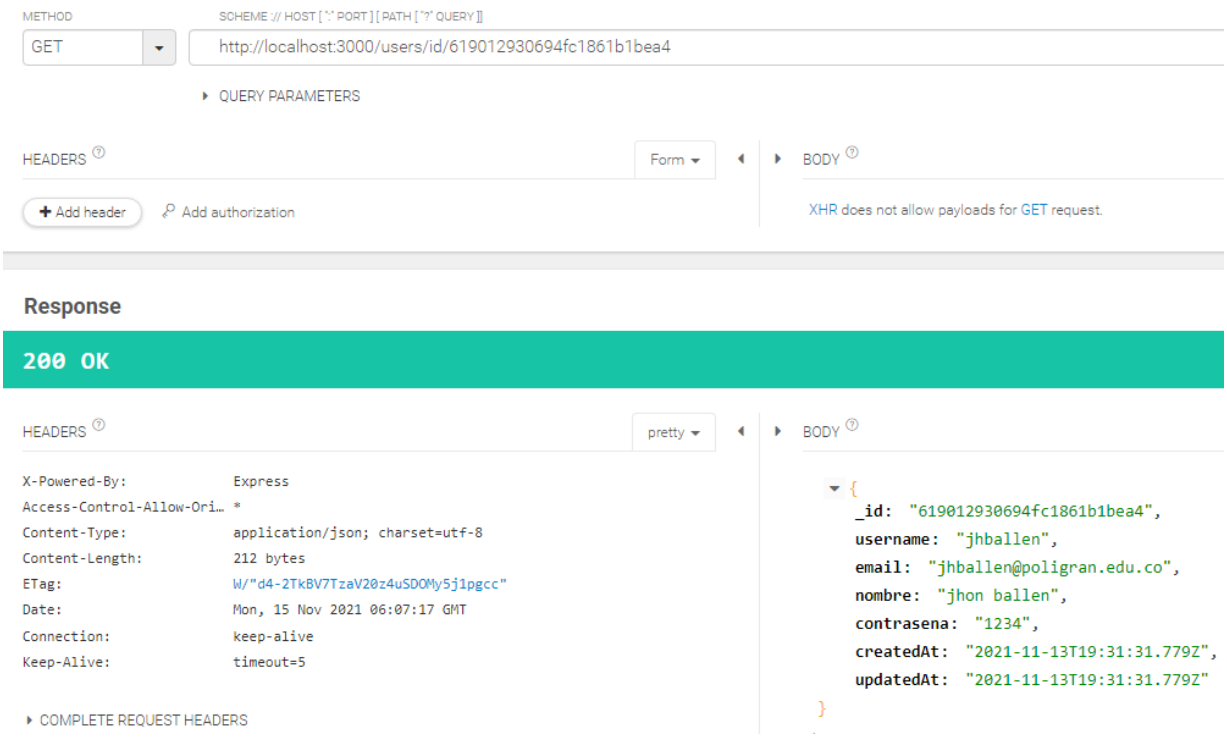


Figura 44. Petición HTTP tipo GET, para consultar un usuario registrado en la aplicación enviando su id por la URL, junto a la respuesta del Back-end. Generado en Talend API Tester.

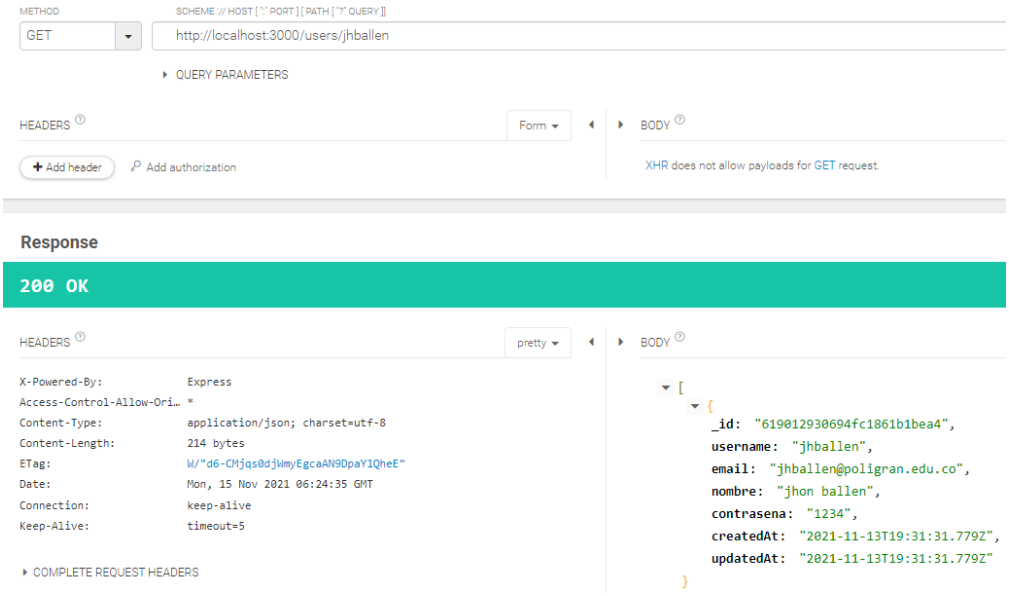


Figura 45. Petición HTTP tipo GET, para consultar un usuario registrado en la aplicación enviando su username por la URL, junto a la respuesta del Back-end. Generado en Talend API Tester.

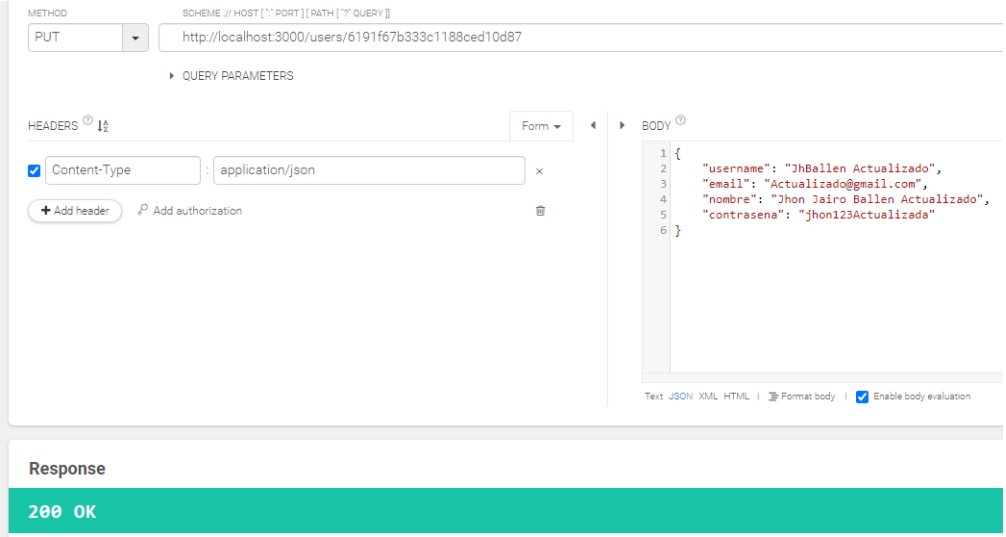
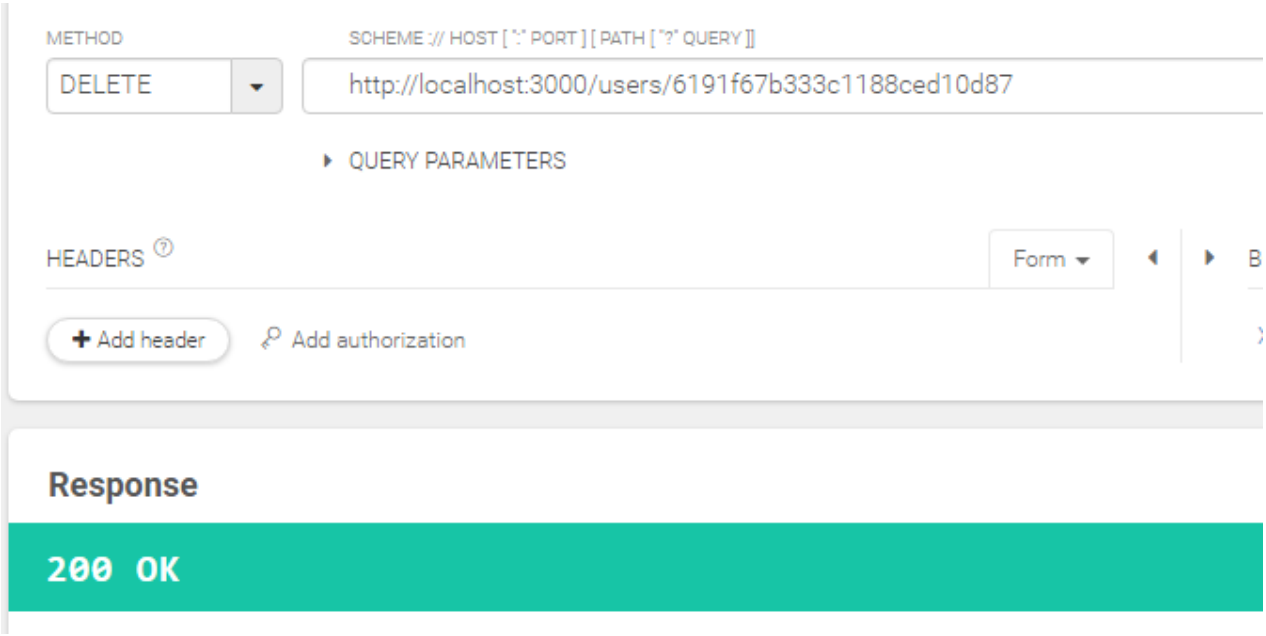


Figura 46. Petición HTTP tipo PUT, para actualizar un usuario registrado en la aplicación enviando su id por la URL, junto a la respuesta del Back-end.

Figura 47. Petición HTTP tipo DELETE, para eliminar un usuario registrado en la aplicación enviando su id por la URL, junto a la respuesta del Back-end. Generado en Talend API Tester.

Colecciones de datos en MongoDB Atlas: Desde la herramienta de MongoDB Atlas se puede observar las colecciones actuales en la base de datos, como se observa



en las siguientes imágenes estarán las cuatro entidades programadas desde el Back-end con los datos registrados en la base de datos.

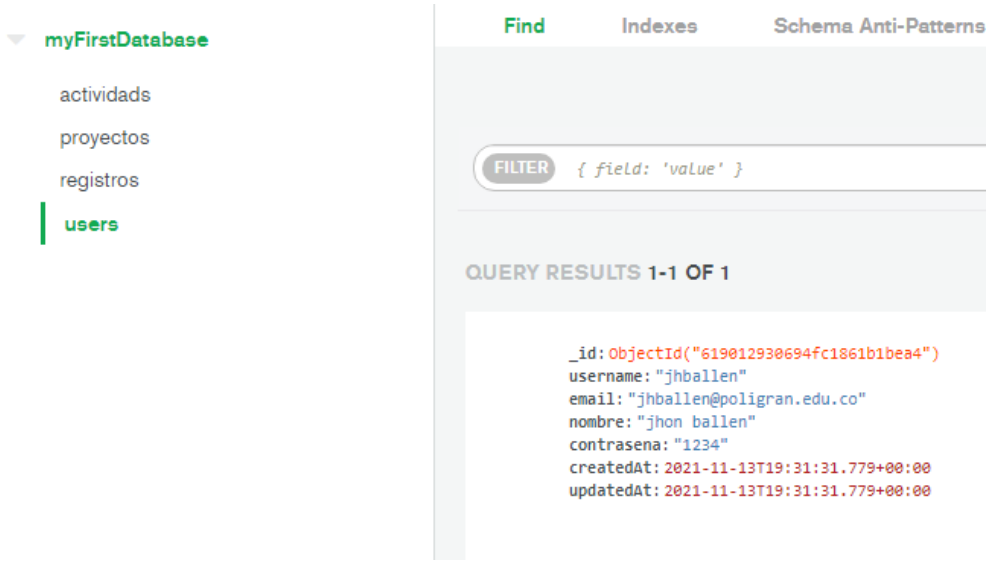


Figura 48. Colección de Users en MongoDB Atlas. Generado en MongoDB Atlas.

Figura 49. Colección de Actividades en MongoDB Atlas. Generado en MongoDB Atlas.

Figura 50. Colección de Registra en MongoDB Atlas. Generado en MongoDB Atlas.

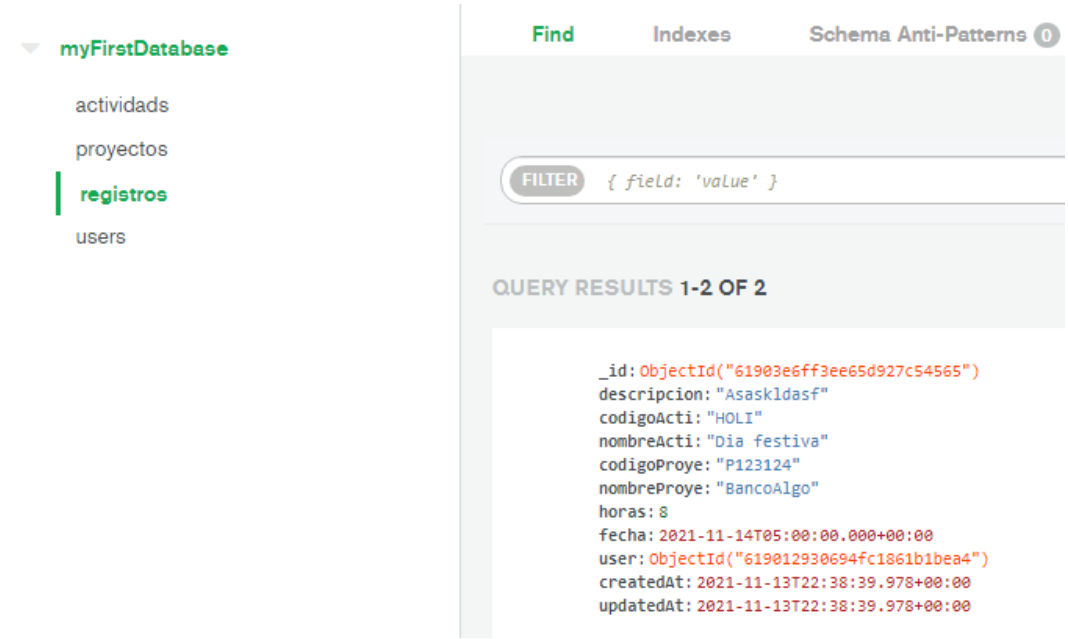


Figura 51. Colección de Registro en MongoDB Atlas. Generado en MongoDB Atlas.

9.6 Implementación:

- Producto piloto final de la aplicación web:

En este apartado se mostrarán por imágenes cada una de las interfaces finales de la aplicación y se explicara el funcionamiento de cada una de ellas.

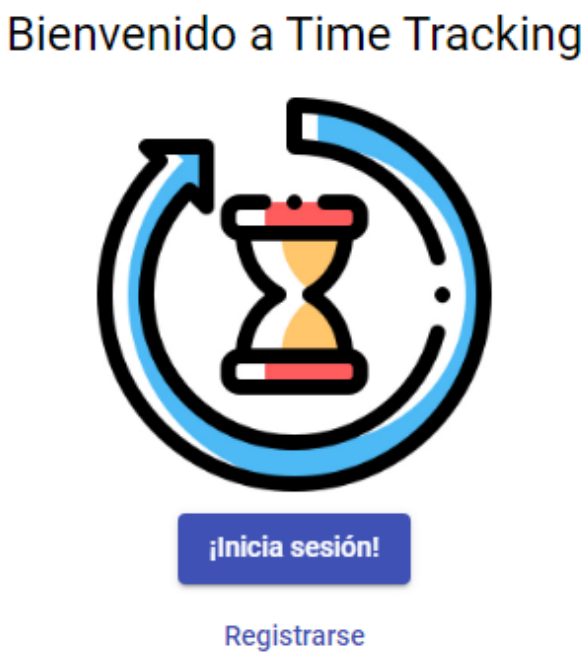


Figura 52. Página principal de la aplicación web TimeTracking. Elaboración propia.

En esta primera interfaz se puede observar el nombre de la aplicación y un par de botones, uno permite al usuario dirigirse a la vista encargada de la autenticación de la aplicación y el otro en caso de no estar registrado aun en la aplicación, le permite registrar sus datos para acceder a la misma.

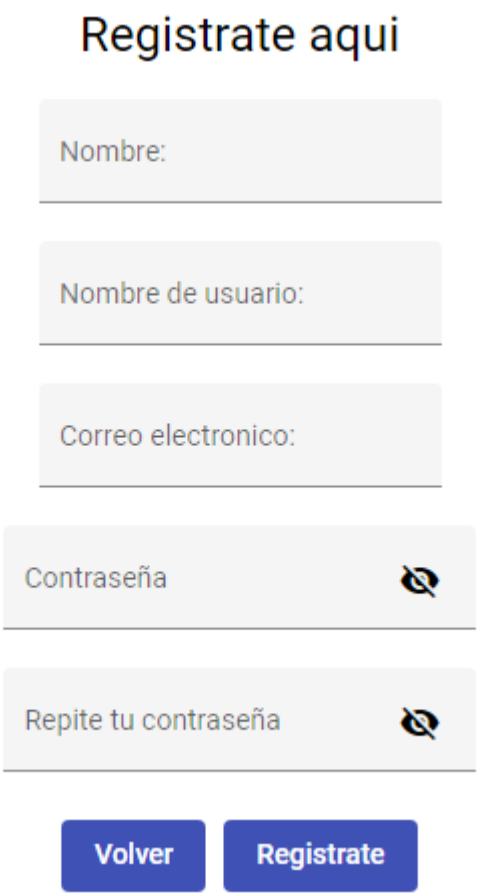


Figura 53. Interfaz para el registro de nuevos usuarios en la aplicación TimeTracking. Elaboración propia.

En esta interfaz es donde se le solicitan los datos principales al usuario que terminaran almacenados en la base de datos y se enviaran por una petición POST directa al API, es importante resaltar que en caso de que no coincidan las contraseñas no se podrá registrar y que es importante el nombre de usuario, porque con este es con el que se realizara la autenticación.

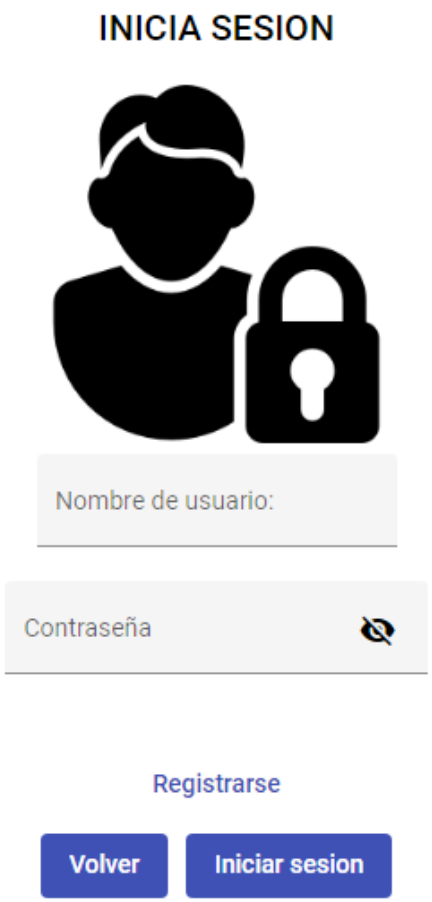


Figura 54. Interfaz para la autenticación en la aplicación TimeTracking. Elaboración propia.

El usuario puede ingresar los datos de autenticación por medio de esta interfaz, en donde solo se solicita el nombre de usuario registrado en la aplicación y la contraseña. Para realizar la validación la aplicación web hace una consulta de tipo GET por medio del nombre de usuario al API Rest, si la contraseña relacionada a ese usuario coincide con la que se ingresa en dicha interfaz, se dará por exitoso el ingreso.

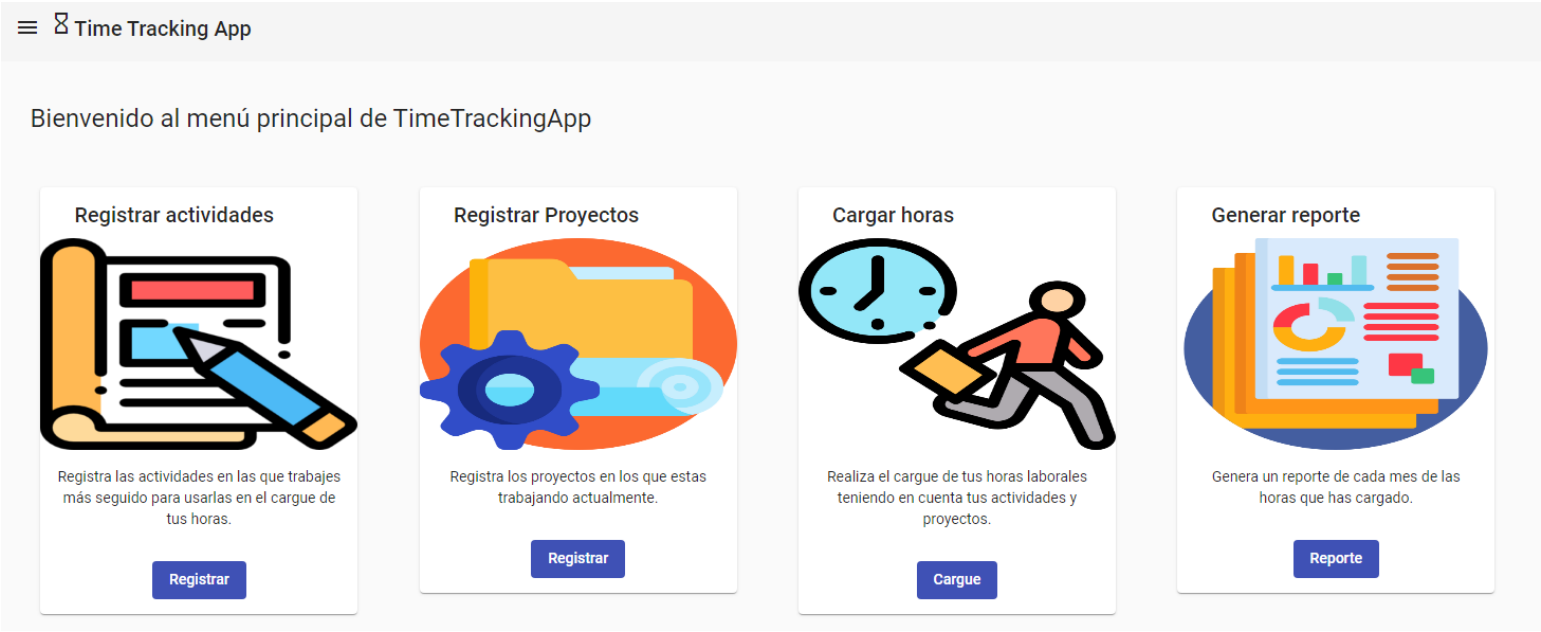


Figura 55. Menú principal de la aplicación TimeTracking. Elaboración propia.

En esta imagen se puede observar el menú al que es redireccionado el usuario tan pronto realiza la autenticación en la aplicación, allí el encuentra cuatro funcionalidades que repasaremos a continuación.

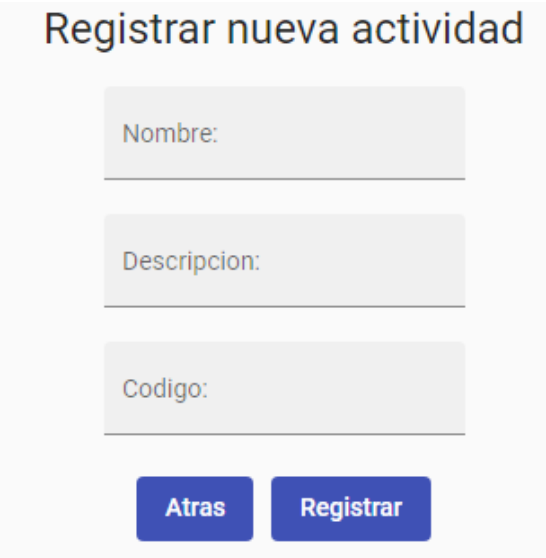


Figura 56. Interfaz para almacenar la información correspondiente a las actividades laborales del usuario. Elaboración propia.

En la empresa es común que los trabajadores realicen actividades del mismo tipo con bastante frecuencia, es por ello que esta aplicación web les permite almacenar los datos relevantes de dichas actividades, como lo son el nombre, una breve descripción y el código. Toda esta información es de bastante valor para los trabajadores y para la empresa, pero el código resalta entre estas debido a que él es el que define que tipo de actividad estuvo realizando el empleado. Pueden ser actividades como documentación, codificación, días festivos, etc.

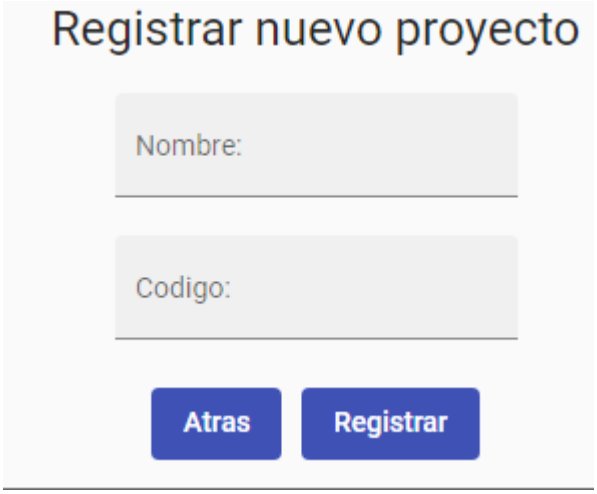


Figura 57. Interfaz para almacenar la información correspondiente a los proyectos en los cuales está trabajando el usuario. Elaboración propia.

Cuando un usuario desea realizar el reporte de sus horas trabajadas, es de suma importancia que estas se encuentren relacionadas a algún proyecto activo en la empresa, estos códigos son bastante extensos y difíciles de memorizar. Es por ello que esta aplicación web permite que el usuario almacene cuantos proyectos el necesite con la información de importancia para el cargue de horas que es el nombre y el código correspondiente a ese proyecto.

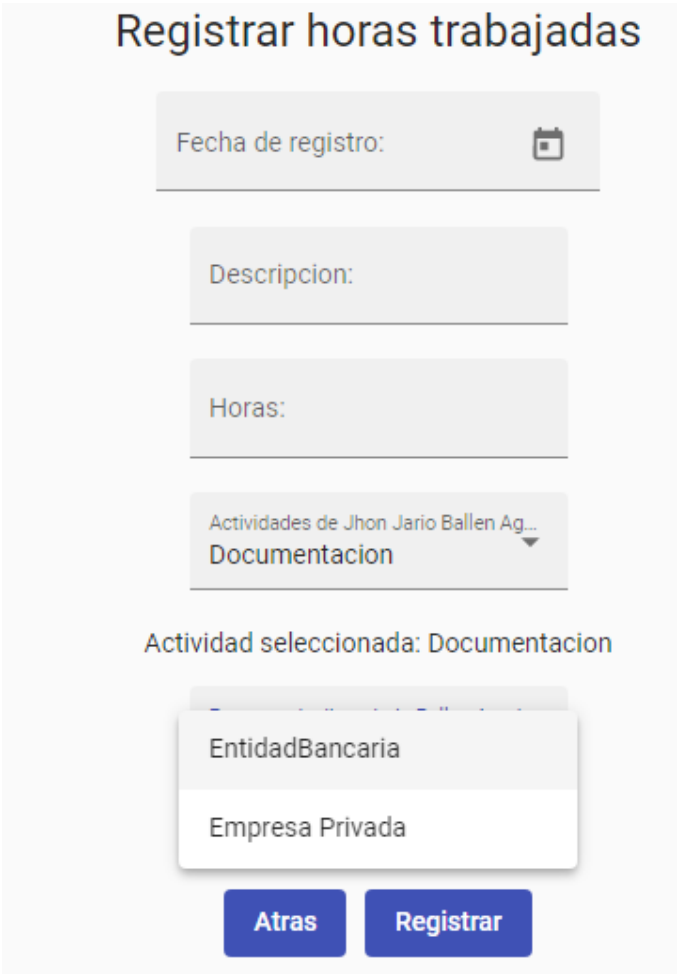


Figura 58. Interfaz para realizar el cargue de horas asociado a una fecha y con la información relevante para la empresa al realizar el cargue correspondiente. Elaboración propia.

Es aquí donde el usuario realizara el cargue diario de sus horas laboradas, como se observa en la imagen se solicita una fecha a la cual se relacionara el registro de horas creado, una corta descripción, el número de horas trabajado en dicha actividad en esa fecha correspondiente al proyecto seleccionado.

Las dos listas de selección que se pueden observar en la imagen traen desde el back-end las actividades y proyectos relacionadas al usuario que se encuentra autenticado en la aplicación, permitiéndole así al usuario relacionarlas en el cargue de sus horas y en el reporte que se generara más adelante.

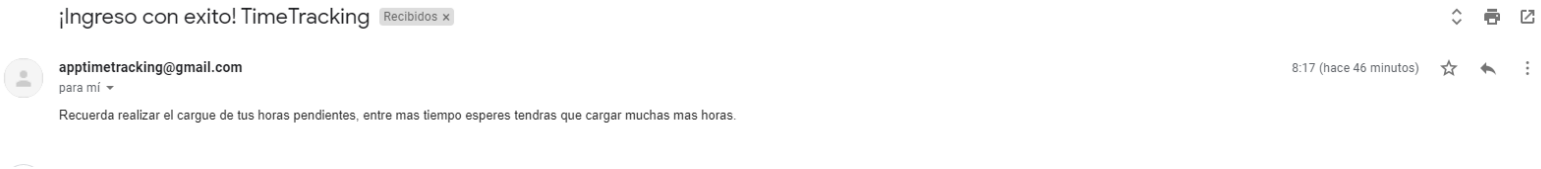
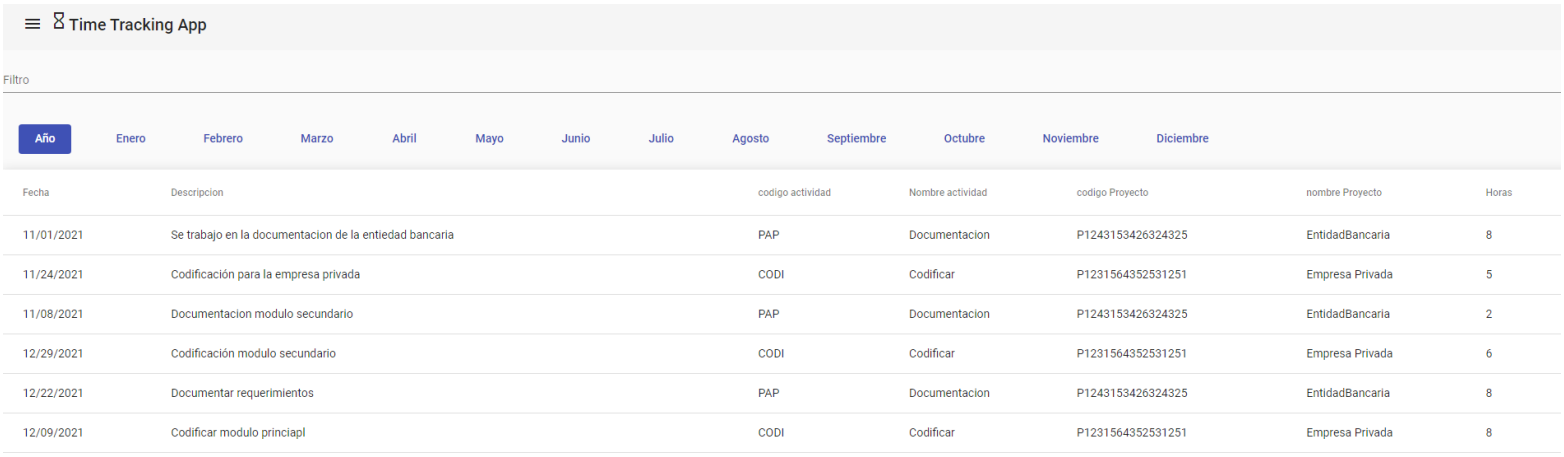


Figura 59. Sistemas de notificaciones de la aplicación TimeTracking. Elaboración propia.

El sistema de notificaciones se implementó desde la aplicación Back-end, permitiendo así que en cualquier momento que se quiera notificar desde la aplicación web, sea tan sencillo como realizar una petición al API rest con la información relevante.

En este ejemplo que se observa en la imagen, cuando la aplicación web realiza la consulta envía la notificación al usuario que se encuentra autenticado en ese momento en la aplicación y con la información que se le asigne para el mensaje, en donde se puede ingresar texto para el asunto y el cuerpo del correo electrónico.

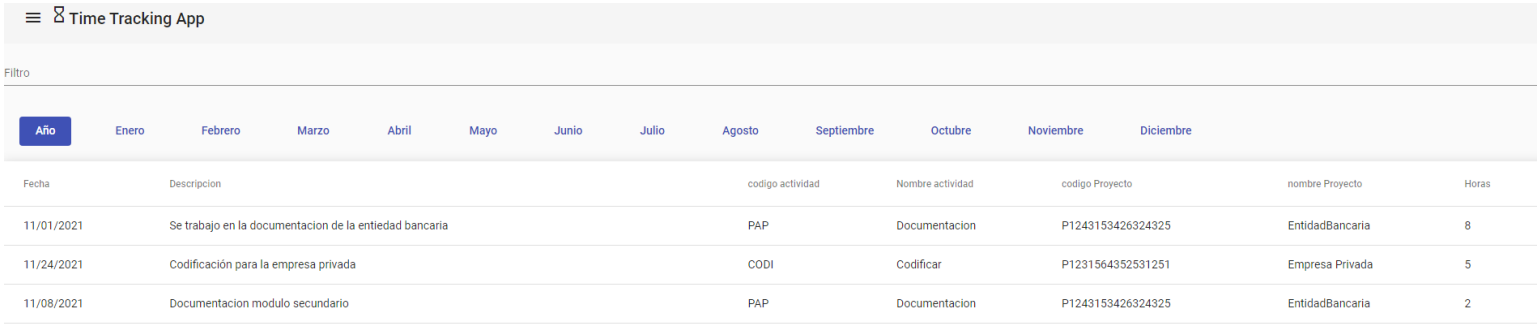


Fecha	Descripcion	codigo actividad	Nombre actividad	codigo Proyecto	nombre Proyecto	Horas
11/01/2021	Se trabajo en la documentacion de la entiedad bancaria	PAP	Documentacion	P1243153426324325	EntidadBancaria	8
11/24/2021	Codificación para la empresa privada	CODI	Codificar	P1231564352531251	Empresa Privada	5
11/08/2021	Documentacion modulo secundario	PAP	Documentacion	P1243153426324325	EntidadBancaria	2
12/29/2021	Codificación modulo secundario	CODI	Codificar	P1231564352531251	Empresa Privada	6
12/22/2021	Documentar requerimientos	PAP	Documentacion	P1243153426324325	EntidadBancaria	8
12/09/2021	Codificar modulo principl	CODI	Codificar	P1231564352531251	Empresa Privada	8

Figura 60. Generación de reportes de las actividades registradas por los empleados en la aplicación TimeTracking. Elaboración propia.

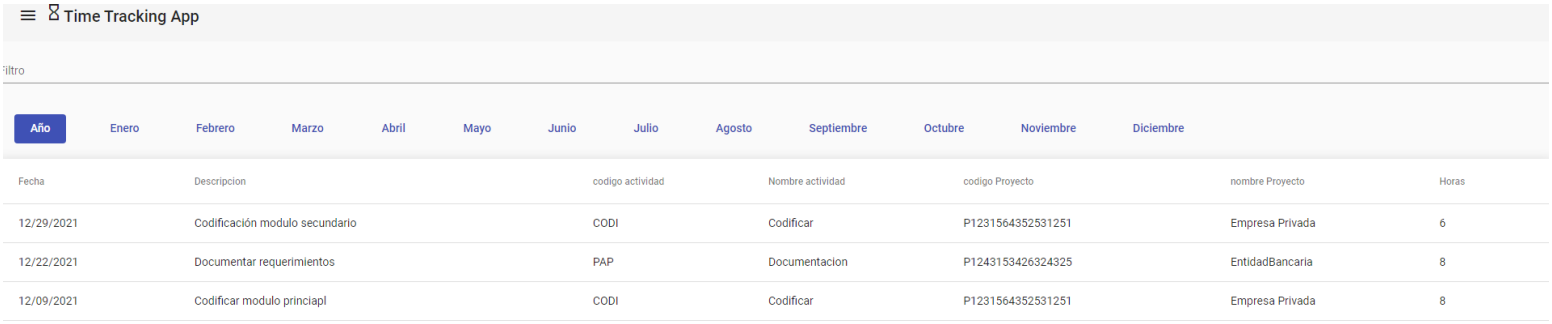
En esta imagen se puede observar el reporte de las horas registradas por el usuario en el año, se puede ver la fecha, la descripción del cargue de horas, el código

al que corresponde la actividad, el nombre de la actividad, el nombre del proyecto, el código del proyecto y claramente el número de horas cargadas.



Fecha	Descripcion	codigo actividad	Nombre actividad	codigo Proyecto	nombre Proyecto	Horas
11/01/2021	Se trabajo en la documentacion de la entiedad bancaria	PAP	Documentacion	P1243153426324325	EntidadBancaria	8
11/24/2021	Codificación para la empresa privada	CODI	Codificar	P1231564352531251	Empresa Privada	5
11/08/2021	Documentacion modulo secundario	PAP	Documentacion	P1243153426324325	EntidadBancaria	2

Figura 61. Generación de reportes de las actividades registradas por los empleados en la aplicación TimeTracking correspondiente al mes de noviembre. Elaboración propia.



Fecha	Descripcion	codigo actividad	Nombre actividad	codigo Proyecto	nombre Proyecto	Horas
12/29/2021	Codificación modulo secundario	CODI	Codificar	P1231564352531251	Empresa Privada	6
12/22/2021	Documentar requerimientos	PAP	Documentacion	P1243153426324325	EntidadBancaria	8
12/09/2021	Codificar modulo princiapl	CODI	Codificar	P1231564352531251	Empresa Privada	8

Figura 62. Generación de reportes de las actividades registradas por los empleados en la aplicación TimeTracking correspondiente al mes de diciembre. Elaboración propia.

En estados dos imágenes se puede observar la generación de los reportes del usuario, pero en este caso se realiza un filtro de cada mes y como en el reporte anterior, se puede observar la información relevante de cada reporte. En la parte superior se puede observar un filtro, el cual permite filtrar registros de horas en cualquiera de las columnas dependiendo lo que se ingrese en dicho filtro.

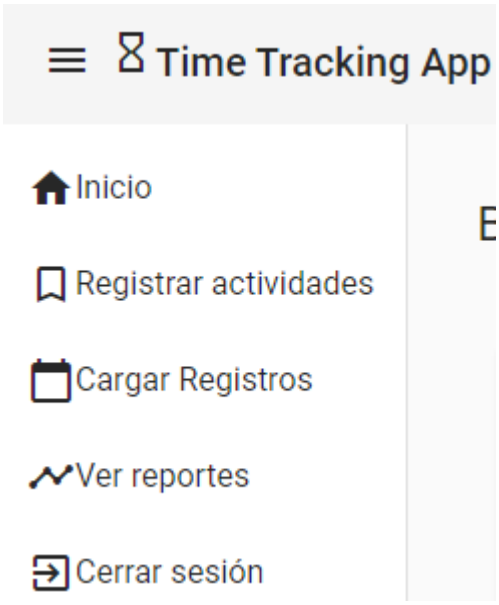


Figura 63. La aplicación también cuenta con una barra de navegación que se mantiene presente en todas las vistas cuando el usuario esta autenticado. Elaboración propia.

Este menú de navegación le permite al usuario dirigirse a cada una de las funcionalidades que forman parte de la aplicación y también le permiten cerrar la sesión cuando termine de usar la aplicación.

- Comparativa entre el piloto final con la página web actual de la empresa.

En esta corta comparativa se mencionará principalmente las funcionalidades adicionales con las que cuenta el producto final de aplicación web obtenida en el lapso de la práctica.

La aplicación web les permite a los usuarios almacenar información relevante para ellos para tener acceso a esta información al momento de realizar su cargue de horas

obligatorio. La aplicación es un espacio personal para cada usuario y pueden consultar todos los registros que hayan generado a lo largo del año por cada mes.

Las notificaciones sirven de recordatorio constante para los usuarios y así no se les olvide realizar el cargue de las horas, es un sistema de notificaciones automático, por lo que no es necesario dedicar recursos en este proceso y por último toda la aplicación cuenta con estilos y animación CSS que hacen la aplicación más vistosa.

10. CRONOGRAMA:

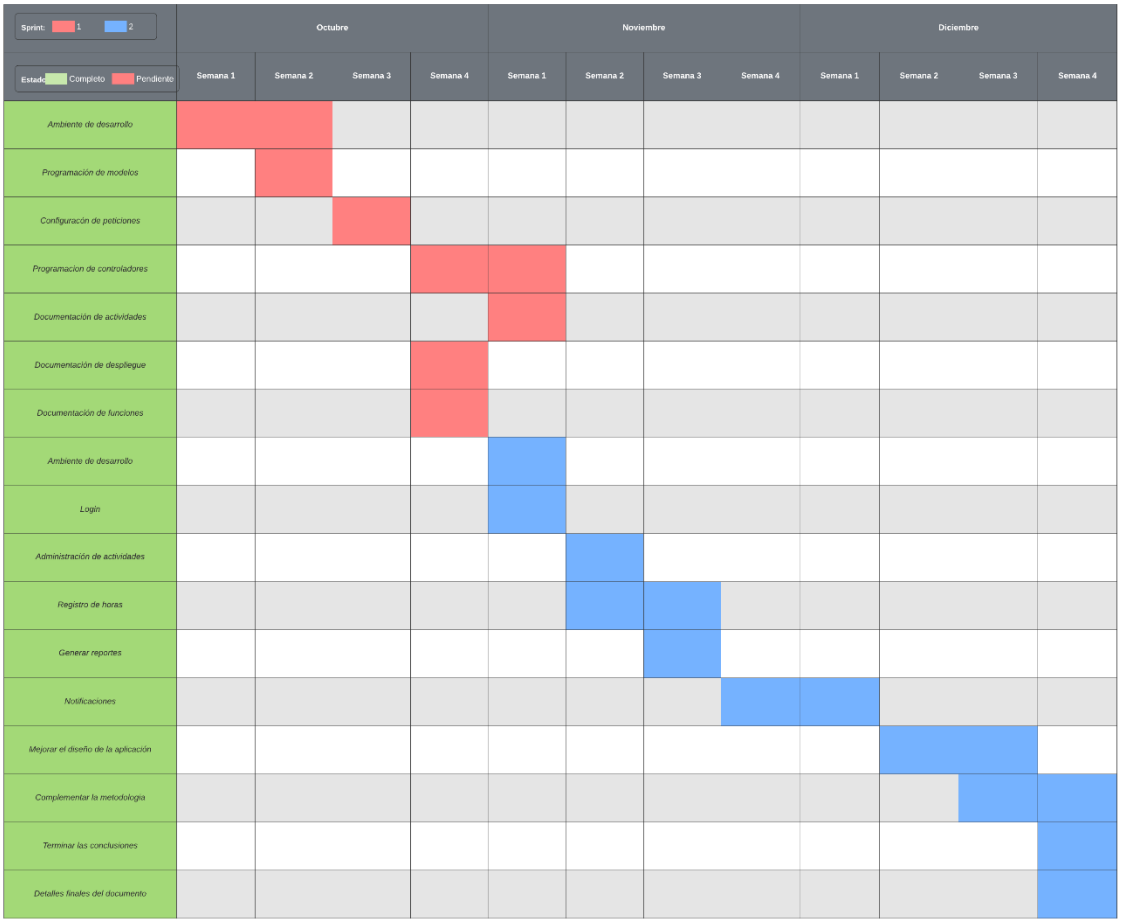


Figura 64. Diagrama de Gantt con el cronograma que se siguió para el desarrollo de las actividades. Generado en Lucidchart.

11. CONCLUSIONES:

Después de realizar un análisis completo a cada uno de los resultados obtenidos, se llegó a varias conclusiones. La primera, desde un comienzo el objetivo principal fue conseguir una aplicación web piloto, la cual debía cumplir con las principales funciones propuestas para mejorar el sistema actual.

Se implementó con éxito la metodología Scrum a pesar de ser solo una persona la que trabajo principalmente en el desarrollo del proyecto y acompañado a ello se obtuvo un listado de documentos y evidencias de este proceso. Para la propuesta inicial del proyecto piloto, junto al análisis de la viabilidad, se recibió una buena retroalimentación del jefe directo y a los empleados que se les presento la aplicación les pareció una excelente idea para mejorar la herramienta que tienen actualmente.

Por último, se concluyó que el proyecto aún se encuentra en un estado piloto, para futuras contribuciones. Es por ello, que para futuras mejoras se puede buscar la forma de desplegar la aplicación en un servidor privado de la empresa. De esta forma se podría buscar una implementación con el sistema de usuarios que maneja actualmente, una interfaz completa para que los jefes de área tengan acceso a revisar las horas que registro cada usuario en su grupo de trabajo. Finalmente, con el equipo de robotización y procesos se podría analizar la viabilidad de implementar un bot para automatizar procesos.

12. BIBLIOGRAFÍA:

Arizmendi, P. (2018). AngularJS: Conviértete en el profesional que las compañías de software necesitan (1.a ed., Vol. 3). Paiminix. https://books.google.com.co/books/about/AngularJS_Convi%C3%A9rtete_en_el_profesional.html?id=q1FjDwAAQBAJ&redir_esc=y

Camarena, S., Gamaliel, J., & Espinosa, T. (2012, noviembre). redalyc. Automatización de la codificación del patrón modelo vista controlador (mvc) en proyectos orientados a la Web. Recuperado 20 de diciembre de 2021, de https://www.redalyc.org/pdf/104/10423895005.pdf

Córdova, R. F., & Cuzco, B. E. (2013). Análisis comparativo entre bases de datos relacionales con bases de datos no relacionales. (N.o 1). https://dspace.ups.edu.ec/bitstream/123456789/6977/1/UPS-CT003639.pdf

Express. (s. f.). Express - Infraestructura de aplicaciones web Node.js. Expressjs. Recuperado 2 de noviembre de 2021, de https://expressjs.com/es/

Fustik, V. (2017). Scrum Methodology Compared with Other Methodologies Applied in the Software Development Projects. Proceedings of the International Conference on Information Technologies, 7–16.

Gaete, J., Villarroel, R., Figueroa, I., Cornide-Reyes, H., & Muñoz, R. (2021). Enfoque de aplicación ágil con Scrum, Lean y Kanban. INGENIARE - Revista Chilena de Ingeniería, 29(1), 141–157.

Generalidades del protocolo HTTP - HTTP | MDN. (2021, 12 noviembre). NDM Web Docs. Recuperado 14 de noviembre de 2021, de https://developer.mozilla.org/es/docs/Web/HTTP/Overview

Gil Vera, V. D., Da Silva, C. R. G., Gil Vera, J. C., & Teutsch, J. (2018). Frameworks para el desarrollo de prototipos WEB: Un caso de aplicación. Lámpsakos, 20, 40–53. https://doi.org/10.21501/21454086.2065

HTML: Lenguaje de etiquetas de hipertexto | MDN. (s. f.). NDM Docs. Recuperado 14 de noviembre de 2021, de https://developer.mozilla.org/es/docs/Web/HTML

Informática Básica: ¿Qué son las aplicaciones web? (s. f.). GCFGlobal.org. Recuperado 20 de noviembre de 2021, de https://edu.gcfglobal.org/es/informatica-basica/que-son-las-aplicaciones-web/1/

Londoño-Rojas, L. F., Tabares-Morales, V., Rosecler-Bez, M., & Duque-Mendez, N. D. (2018). Análisis Comparativo De Guías Para El Desarrollo Web Accesible / Comparative Analysis of Guides for Accessible Web Development. Ciencia e Ingeniería Neogranadina, 28(1), 101–115. https://doi.org/10.18359/rcin.2683

Lucidchat. (s. f.). Qué es el lenguaje unificado de modelado (UML). Lucidchart. Recuperado 15 de noviembre de 2021, de https://www.lucidchart.com/pages/es/que-es-el-lenguaje-unificado-de-modelado-uml/#section_0

Mohedano, J., & Saiz, J. M. (2012). Iniciación a JavaScript (1.a ed., Vol. 1). Ministerio de educación de España.

MongoDB. (s. f.). The most popular database for modern apps. Recuperado 1 de noviembre de 2021, de https://www.mongodb.com/

Node.js. (s. f.). Node .Js. Recuperado 1 de noviembre de 2021, de https://nodejs.org/es/about/

Pantaleo, G. (2015). *Ingeniería de software* (1.ª ed.). Alfaomega Ediciones.

Reinman, A. (s. f.). Nodemailer :: Nodemailer. NODEMAILER. Recuperado 8 de diciembre de 2021, de https://nodemailer.com/about/

Trigas Gallego, M. (2012). Gestión de proyectos informáticos- metodología Scrum (N.o 1). http://openaccess.uoc.edu/webapps/o2/bitstream/10609/17885/1/mtrigasTFC0612memoria.pdf