Whoka: Aplicación Web para la gestión de clientes, empleados y programación de citas

Evelyn Ramírez Vargas

Whoka es una aplicación web desarrollada para el sector servicios, como por ejemplo peluquerías o consultorios de psicología, que permite la gestión de trabajadores,clientes y citas. El proyecto desarrollado en Python, Flask, HTML, CSS y MongoDB ofrece una interfaz sencilla e intuitiva para que los usuarios puedan organizar su horario, sus citas, sus clientes y sus trabajadores de manera eficiente. Además de contar con una funcionalidad que permite recordar la cita a los clientes vía email con un día de antelación de manera automática. En pocas palabras, el proyecto busca optimizar y reducir el tiempo de gestión y comunicación entre un negocio y sus clientes.

Whoka, gestión de citas, notificación vía email, horario semanal, automatización de mensajes, agenda virtual.

Whoka is a web application developed for the service sector, such as hair salons or psychology clinics, that allows appointment management with clients. The project, developed using Python, Flask, HTML, CSS, and MongoDB, offers a simple and intuitive interface for users to efficiently organize their schedules, appointments, clients, and staff. Additionally, it includes a feature that automatically reminds clients of their appointments via email one day in advance. In short, the project aims to optimize and reduce the time spent on management and communication between a business and its clients.

Whoka, appointment management, email notification, weekly schedule, message automation, virtual diary.

1 Introducción - Contexto del trabajo

En el ámbito del sector servicios la organización de citas con los clientes a menudo carece de un registro adecuado por falta de tiempo y herramientas. Además, rara vez se notifica a los clientes para recordarles la cita, lo que provoca muchas ausencias. Whoka pretende solucionar estos problemas ofreciendo una interfaz de usuario intuitiva para dar apoyo a los comercios del sector servicios y ofrecerles un espacio donde puedan gestionar a sus trabajadores, clientes, calendario y citas.
El usuario principal de la plataforma será la persona encargada de agendar las citas entre su comercio y sus clientes y para ello contará con diferentes pantallas y funcionalidades que explicaremos durante el informe.
La motivación para desarrollar esta idea surge de mi necesidad de automatizar la gestión de citas con mi terapeuta, a raíz del olvido de una de las sesiones. Al empezar a trabajar en el proyecto, me pareció buena idea, plantearlo de una forma mucho más general y no limitarlo a mi uso personal con mi psicóloga. Para ello pensé en una web que pueda servir para diferentes usuarios, con el objetivo de facilitar la gestión de citas y automatizar tareas como el recordatorio a los clientes o la visualización de un calendario completo.
El siguiente paso fue ver como de innovadora era esta idea y que debía destacar en mi proyecto para diferenciarlo del resto. Haciendo una búsqueda encontré diferentes softwares parecidos. Los más destacables son los siguientes:

Agendit

Web de pago para negocios, que permite visualizar la agenda del profesional y automatizar el aviso de las citas a sus clientes. Aunque las funcionalidades son parecidas, Agendit es de pago, mientras que Whoka es de uso gratuito. Además, nuestra web permite gestionar más aspectos del negocio como los trabajadores y clientes.

Figura 1. Pantalla de inicio de Agendid.

Nubimed

Software para la gestión de citas en el sector médico. Permite agendar citas online y firmar documentos a través del software. Tiene muchas más funcionalidades relacionadas con el sector de la medicina en concreto. También se trata de un software de pago que además cuenta con funcionalidades como firmar documentos oficiales.

Figura 2. Pantalla de inicio de Nubimed.

Meetfox

Software multiplataforma que permite agendar citas además de videollamadas, sincronizar los calendarios y automatizar los recordatorios. Este software tiene funcionalidades gratis y otras de pago, además de ser multiplataforma, mientras que mi proyecto es para web.

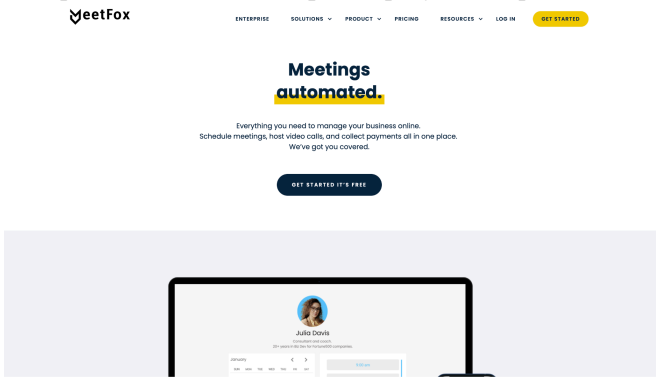

Figura 3. Pantalla de inicio de Meetfox.

Aunque todos estos softwares tienen algo en común con el proyecto, como hemos comentado, todos ellos tienen diferencias significativas. Además quiero destacar que el punto fuerte de este proyecto es que permite gestionar otras características además de las citas y lo pueden usar comercios de diferentes sectores.

1.1 Objetivos

El objetivo principal de este proyecto es desarrollar una aplicación web que habilite al usuario para registrarse, iniciar sesión, visualizar el calendario semanal, así como crear y administrar clientes, trabajadores y citas. Además, el sistema debe enviar notificaciones automáticas a los clientes por correo electrónico recordándoles las citas.
La web debe cumplir con las siguientes funcionalidades:

1. Registrarse y loguearse como usuario.
2. Visualizar tu calendario semanal.
3. Automatizar el recordatorio via email.
4. Gestionar perfiles de trabajadores y clientes.

Estos objetivos son los principales a conseguir durante el desarrollo del trabajo de fin de grado, pero además la web debe ser ampliable para más adelante tener la posibilidad de implementar otras opciones, como una pantalla para los clientes u otras funcionalidades.

Además de los objetivos relacionados con los requisitos funcionales, añadimos otros como són el aprendizaje de las tecnologias y metodologias que utilizaré, la redacción de los informes adecuados para el proyecto, un diseño simplificado y con un estilo minimalista de la parte visual de la web y, por último, la búsqueda de hacer el servicio lo más óptimo posible obteniendo un software de calidad.

1.2 Metodología

Para la organización del proyecto se ha utilizado la metodología ágil, en concreto "Kanban". Se trata de un sistema de gestión visual que cuenta con un tablero dividido en diferentes columnas que representan las etapas del proceso de trabajo. En estas columnas se colocan diferentes tarjetas y cada una de estas tarjetas representa una unidad de trabajo individual o tarea. Las tarjetas se cambian de columna según su progreso.
Los puntos que han favorecido la elección de Kanban versus otras metodologías, es que se trata de una metodología ágil pero que no precisa de definición de roles para su desarrollo, permite optimizar el flujo de trabajo y es factible para un solo desarrollador.

1.3 Herramientas tecnológicas

El desarrollo de la web consta de diferentes lenguajes. El backend está hecho en Python junto con el framework Flask que permite comunicarlo al frontend con HTML y CSS. Por otro lado, la base de datos es no relacional en MongoDB.
Siguiendo el patrón Modelo Vista Controlador nos queda el siguiente diagrama.

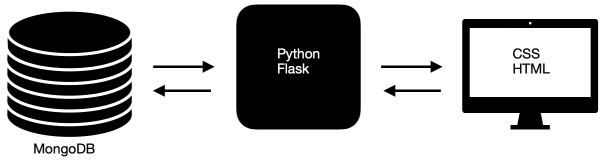

Figura 4. Diagrama del patrón MVC

La decisión de utilizar estas tecnologías se debe a que Python es un lenguaje fácil de usar y cuenta con muchísimas librerías y documentación accesible. Además, Flask es un framework ligero para el desarrollo web. Por último, MongoDB se trata de una base de datos no relacional y por lo tanto permite un esquema flexible, fácil de gestionar y utilizar desde python, además de aportar escalabilidad, cosa que beneficia a un proyecto que pretende seguir en crecimiento.

1.4 Planificación

Durante el desarrollo del proyecto se planea realizar el análisis de requisitos, el prototipo de la web, entrevista a posibles usuarios, el desarrollo del código y el testeo de las funcionalidades. Para ello, se plantea la siguiente planificación:

- → 30 h para el desarrollo de informes.
- → 9 h para reuniones con la tutora.
- → 260 h para el desarrollo del proyecto.

En el anexo se puede encontrar una tabla con las horas dedicadas a las diferentes tareas.

2 Diseño

A continuación definiremos los requisitos analizados, mostraremos el diseño del prototipo y de la base de datos, además de las conclusiones extraídas de la entrevista a posibles usuarios.

2.1 Requisitos funcionales

Perfil de usuario

- RF-1 Crear cuenta: El sistema debe permitir a los usuarios crear una cuenta.
- RF-2 Iniciar sesión: El sistema debe permitir a los usuarios iniciar sesión.
- RF-3: Editar perfil: El sistema debe permitir a los usuarios editar su perfil.
- RF-4 Borrar perfil: El sistema debe permitir a los usuarios borrar su perfil.

Perfil de cliente

- RF-5 Crear cliente: El sistema debe permitir a los usuarios crear un nuevo cliente.
- RF-6 Editar perfil cliente: El sistema debe permitir a los usuarios editar el perfil de un cliente.
- RF-7 Borrar cliente: El sistema debe permitir a los usuarios borrar el perfil de un cliente.
- RF-8 Listar clientes: El sistema debe mostrar una lista con los clientes registrados.
- RF-9 Buscador de clientes: El sistema debe permitir a los usuarios buscar un cliente a través de un buscador.

Perfil de empleado

- RF-10 Crear empleado: El sistema debe permitir a los usuarios crear un nuevo trabajador.
- RF-11 Editar perfil empleado: El sistema debe permitir a los usuarios editar el perfil de un empleado.
- RF-12 Borrar empleado: El sistema debe permitir a los usuarios borrar el perfil de sus empleados.
- RF-13 Listar empleados: El sistema debe mostrar una lista con los trabajadores registrados.
- RF-14 Buscador de empleados: El sistema debe permitir a los usuarios buscar un empleado a través de un buscador.

Agendar citas

- RF-15 Crear cita: El sistema debe permitir a los usuarios crear citas.
- RF-16 Mostrar empleados libres: El sistema debe mostrar al usuario los empleados disponibles para una hora concreta al agendar una cita.
- RF-17 Editar cita: El sistema debe permitir a los usuarios editar una cita previamente creada.
- RF-18 Cancelar cita: El sistema debe permitir a los usuarios cancelar una cita.

Visualizar calendario

- RF-19 Mostrar calendario de citas: El sistema debe permitir a los usuario ver el calendario semanal de citas.
- RF-20 Mostrar calendario por trabajador: El sistema debe mostrar el calendario de citas de cada trabajador de manera individual.

Recordatorio de citas:

- RF-21 Recordatorio de citas: El sistema debe enviar un mensaje recordando la cita vía correo electrónico.

Destacar que el análisis de requisitos está más extendido y detallado en el anexo.

2.2 Requisitos no funcionales

Requisitos de rendimiento

- RNF-1: El sistema debe permitir usar la web a cualquier usuario independientemente de las características técnicas de su dispositivo.
- RNF-2: El sistema debe responder a cualquier petición de una funcionalidad en menos de 4 segundos.

Requisitos de seguridad

- RNF-3: El sistema debe evitar que se pueda acceder a la base de datos desde la interfaz gráfica.

Disponibilidad

- RNF-4: El sistema debe estar disponible 24/7.

Mantenibilidad

- RNF-5: El sistema debe ser fácilmente escalable con el fin de hacer crecer la web en un futuro.
- RNF-6: El sistema debe disponer de una buena documentación que permita realizar operaciones de mantenimiento con el menor esfuerzo posible.

Usabilidad

- RNF-7: El tiempo de aprendizaje de un usuario, de media, debe ser menor a 30 minutos.
- RNF-8: El sistema debe tener un diseño UIX sencillo, claro y que permita operar con fluidez.

2.3 Base de datos

Para hacer la base de datos se ha utilizado la herramienta MongoDB Compass. En esta aplicación de escritorio,

tenemos una base de datos guardada en local llamada "TFG". En ella tenemos 4 colecciones: citas, clientes, trabajadores, users.

Citas

En esta colección tenemos los campos:

- empresa: correo de la empresa que agenda la cita.
- cliente: email del cliente que atenderá a la cita.
- trabajador: email del trabajador que atenderá la cita.
- fecha: Fecha en formato "YYYY-MM-DD".
- hora: Hora en formato "HH:mm".
- labor: Qué se realizará durante la cita?
- mensaje: Mensaje que se enviará al cliente para recordarle la cita.

Clientes

En esta colección tenemos los campos:

- name: nombre del cliente.
- email: correo electrónico del cliente.
- phone: número de teléfono del cliente.
- usuario: email de la empresa que agenda la cita.

Trabajadores

En esta colección tenemos los campos:

- name: nombre del trabajador.
- email: correo electrónico del trabajador.
- phone: número de teléfono del empleado.
- empresa: email de la empresa a la que pertenece.
- horario: Array que contiene el horario del trabajador en formato "indice: dia-franja". Ejemplo: 0: "lunes-mañana".
- labor: labor que realiza el trabajador en la empresa.

Users

En esta colección tenemos los campos:

- name: nombre de la empresa.
- email: correo electrónico de la empresa.
- password: contraseña cifrada de la empresa.

2.4 Prototipo

Por tal de facilitar el desarrollo, se ha creado un prototipo de la web. Este prototipo proporciona una representación inicial de la interfaz de usuario y las funcionalidades básicas del proyecto. Además, sirve como referencia durante el desarrollo. También puede ser utilizado durante la entrevista a posibles usuarios para valorar posibles cambios o virtudes del proyecto. A continuación podemos ver algunas pantallas del prototipo:

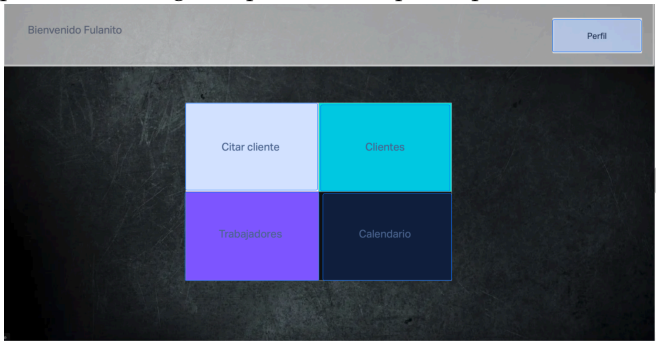

Figura 5. Pantalla principal del prototipo.

Figura 6. Calendario del prototipo.

Figura 7. Listado de clientes del prototipo.

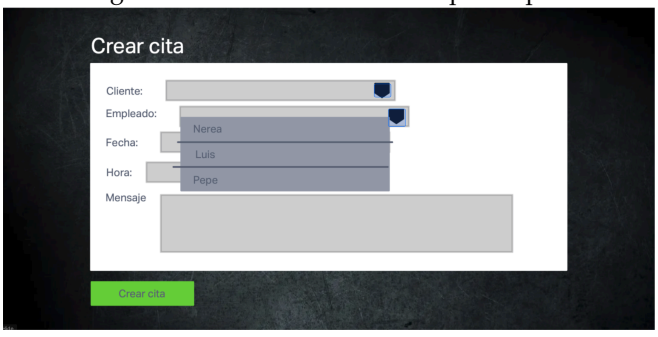

Figura 8. Pantalla de crear cita del prototipo.

El prototipo consta de muchas pantallas y para no sobrecargar este documento con capturas de las mismas, a continuación dejo un enlace donde se puede acceder: https://tfgevelyn.invisionapp.com/overview/TFGEvelyn-clv3zkz5e05pv0191b7iy790k/screens?sortBy=1&sortOrder=1&viewLayout=2

Este enlace te lleva a la web de inVision, la cual he utilizado porque es muy útil para el desarrollo de proyectos, no únicamente para la realización de prototipos, sino también para crear diseños, tableros de tareas etc.

2.5 Entrevista a posibles usuarios

Por tal de obtener información valiosa se ha realizado una entrevista a posibles usuarios. Para ello se ha mostrado el prototipo de la web a la directora de una podología donde gestionan citas de podología y ortopedia. La usuaria ha entendido rápido el funcionamiento de la web, no le ha costado encontrar y desplazarse por las diferentes pantallas ni entender su funcionamiento. Ha comentado que ella tiene una aplicación para la gestión de sus citas y

que comparándolas, Whoka es más adecuada. Por último, la usuaria añadió que le gustaría poder tener la funcionalidad de apuntar comentarios a citas que ya han surgido y que dicha información se guarde en el perfil del cliente, para por ejemplo, controlar qué clientes llegan tarde, entre otras cosas.
En conclusión la entrevista fue beneficiosa para saber qué funcionalidades pueden dar valor a Whoka y si su uso es sencillo para los usuarios reales.

3 Implementación

Como se ha comentado anteriormente, el proyecto sigue el patrón MVC. A continuación explicaremos los diferentes archivos, clases y funciones que lo forman.

3.1 Esqueleto del proyecto

El proyecto se guarda en una carpeta llamada "mi-tfg-proyecto" en cuyo interior hay 2 carpetas más y diferentes archivos. Estas 2 carpetas se llaman "static" y "templates" respectivamente, mientras que los archivos corresponden al enrutador y las diferentes clases. La carpeta "static" contiene los archivos css y otra carpeta con algunas imágenes como el logo o el fondo de la web. Por otro lado, la carpeta "templates" tiene los archivos HTML de las diferentes pantallas de la aplicación.
En conclusión, la carpeta "mi-tfg-proyecto" contiene el modelo y el controlador de la web, mientras que "static" y "templates" contienen los archivos que corresponden a la vista. A continuación podemos ver una foto del esqueleto.

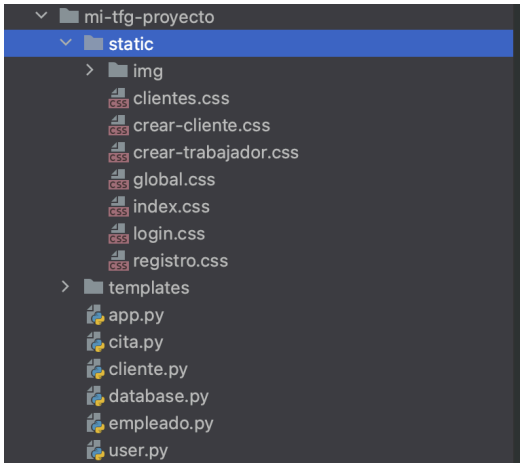

Figura 9. Esqueleto del proyecto

3.2 Archivos Python

App.py

En este archivo podemos encontrar las rutas para acceder a las diferentes pantallas de la aplicación, además de las funciones necesarias por parte del servidor para gestionar las peticiones del usuario. A continuación muestro las funciones que lo componen:

- enviar_correo: Enviar los emails de recordatorio a los clientes.
- revisar_citas_y_enviar_correo: Revisa todas las citas con fecha del dia siguiente en la bd y llama a la función enviar_correo.
- load_user: Cargar un usuario de la BD según su email.
- home: Cargar la vista del loggin.
- addUser: Crear un nuevo usuario.
- delete: Borrar un usuario existente.
- modificar_perfil: Modificar perfil de usuario.
- cambiar_usuario: Si se modifica el correo del usuario en modificar_perfil, cambia el correo de las citas, clientes y trabajadores del usuario.
- citas_semanales: Carga la vista "index.html" pasandole una lista con las citas semanales de la empresa y los diferentes empleados que tiene.
- loggin: Verificar el loggin de un usuario.
- crear_cliente: Crear un nuevo cliente.
- crear_trabajador: Crear un nuevo trabajador.
- buscar_cliente: Devuelve una lista de clientes y en caso de que el usuario busque por nombre, devuelve al cliente buscado.
- buscar_trabajador: Devuelve una lista de trabajadores y en caso de que el usuario busque por nombre, devuelve al trabajador buscado.
- editar_cliente: Modificar perfil de un cliente.
- editar_trabajador: Modificar perfil de un trabajador.
- delete_cliente: Borrar perfil de un cliente.
- delete_trabajador: Borrar perfil de un trabajador.
- registrarse: Cargar la vista para registrarse.
- crear_cita: Crear una nueva cita.
- trabajador_labor: Devuelve un json con los trabajadores que corresponden a la profesión seleccionada.
- obtener_horario: Devuelve el horario de un empleado.
- obtener_dia_semana: Recibe una fecha y devuelve el día de la semana al que pertenece.
- comprobar_disponibilidad: Devuelve true si un trabajador tiene disponible una hora y un día especificados, en caso contrario, devuelve false.
- pantalla_principal: Cargar la vista del índice.
- notFound: Devuelve un mensaje cuando una página no se encuentra.

User.py, empleado.py, cliente.py, cita.py

Estos archivos contienen las clases user, empleado, cliente y cita. Las cuales corresponden a las 4 colecciones de la base de datos y cuyos parámetros hacen referencia a los que se almacenan en las mismas. Las funciones con las que cuentan son las necesarias para crear, guardar o modificar datos de la bd.

Database.py

Este archivo contiene la función dbConnection(), encargada de hacer la conexión con la base de datos.

3.3 Templates

Como hemos comentado, en la carpeta "templates" tenemos los archivos HTML que corresponden a las vistas.

- registrarse.html: Contiene los campos necesarios para crear un nuevo usuario.
- loggin.html: Contiene los campos necesarios para iniciar sesión y un link para acceder a "registrarse.html".

- index.html: Contiene una cabecera con 4 botones para acceder a las diferentes pantallas, el horario semanal que contiene las citas que se realizarán y una columna en el costado para acceder al calendario individual de cada trabajador.
- modificar_perfil.html: Contiene los campos necesarios para editar el perfil de un usuario y otro apartado para borrarlo, el cual pedirá confirmación de la contraseña actual.
- crear_cita.html: Contiene los campos necesarios para crear una cita y código en javascript para que aparezcan los empleados según la labor seleccionada.
- clientes.html: Contiene un buscador, una lista de clientes y un botón para acceder a "crear_cliente.html". Los clientes que aparecen en la lista contienen dos botones, uno para editar el perfil del cliente y otro para crear una cita a ese cliente.
- crear_cliente.html: Contiene los campos necesarios para crear un nuevo cliente.
- editar_cliente.html: Contiene los campos necesarios para editar el perfil de un cliente y un botón para borrarlo.
- crear_empleado.html: Contiene los campos necesarios para crear un nuevo trabajador, entre ellos se encuentra un apartado para el horario laboral.
- trabajadores.html: Contiene un buscador, una lista de trabajadores y un botón para acceder a "crear_empleado.html". Los trabajadores que aparecen en la lista contiene dos botones, uno para editar el perfil del trabajador y otro para observar su calendario semanal.
- editar_trabajador.html: Contiene los campos necesarios para editar el perfil de un trabajador y un botón para borrarlo.

3.4 Test

Para testear Whoka se han realizado diferentes pruebas. Se han hecho test de integración para asegurar que los diferentes módulos del sistema interactúan correctamente entre sí. También se han realizado test funcionales utilizando la web para simular la experiencia del usuario y comprobar que las funcionalidades operan correctamente. Por último se han llevado a cabo test de usabilidad, utilizando usuarios sin conocimientos informáticos y observando su interacción con la web, identificando posibles problemas de usabilidad y asegurándonos que la interfaz de usuario es intuitiva y fácil de usar.

4 Resultados

Como resultado del desarrollo obtenemos la web Whoka con las funcionalidades comentadas en los requisitos y las pantallas que mostramos a continuación.

Figura 10. Pantalla de inicio de sesión

Figura 11. Pantalla de registro

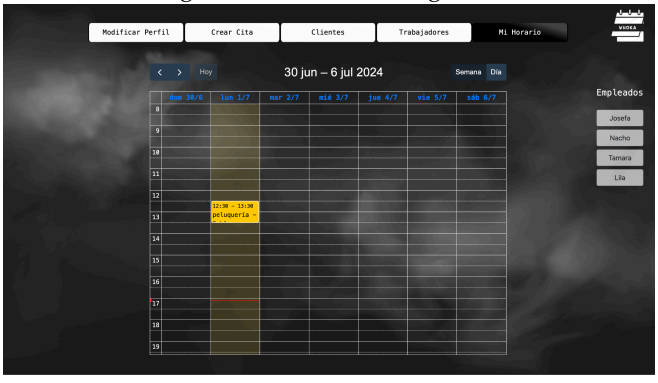

Figura 12. Pantalla principal que contiene el calendario semanal

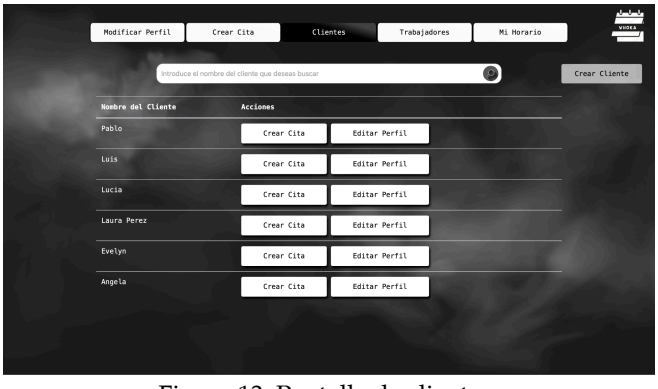

Figura 13. Pantalla de clientes

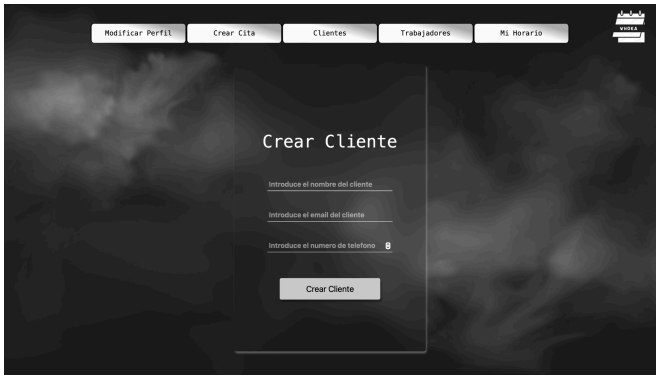

Figura 14. Pantalla de crear cliente

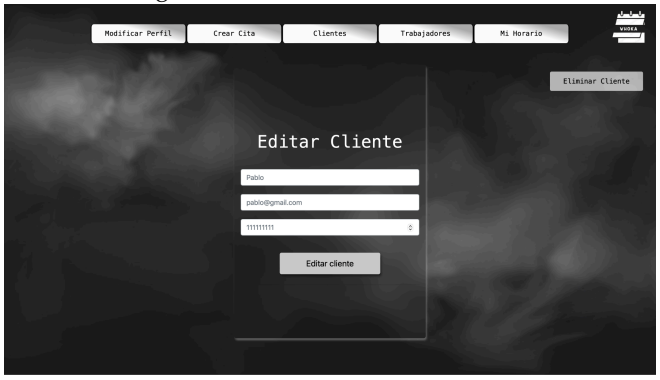

Figura 15. Pantalla de editar perfil del cliente

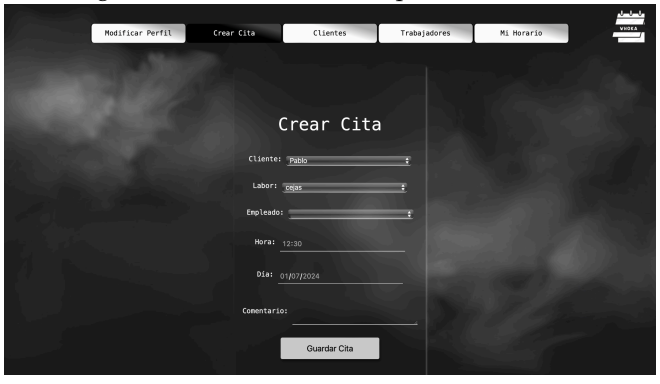

Figura 16. Pantalla de crear cita

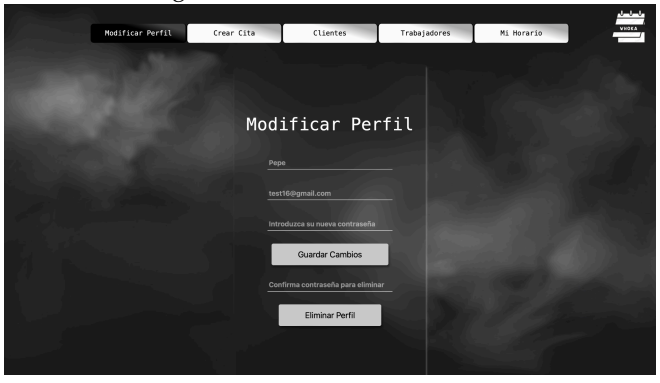

Figura 17. Pantalla de editar perfil de usuario

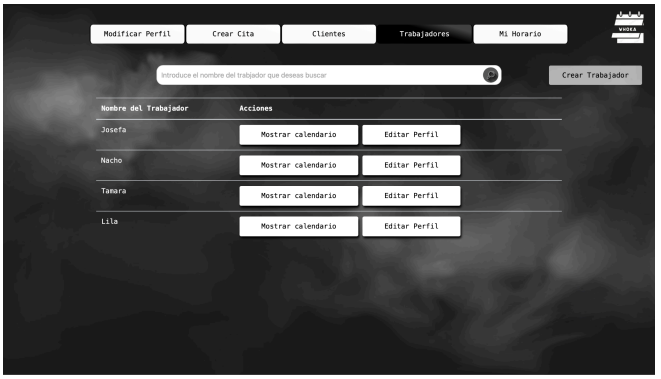

Figura 18. Pantalla de trabajadores

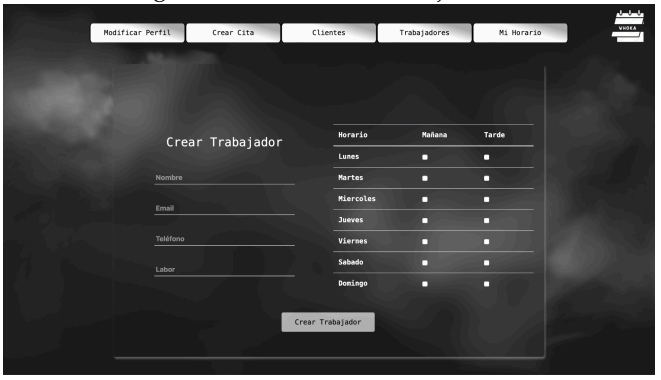

Figura 19. Pantalla de crear trabajador

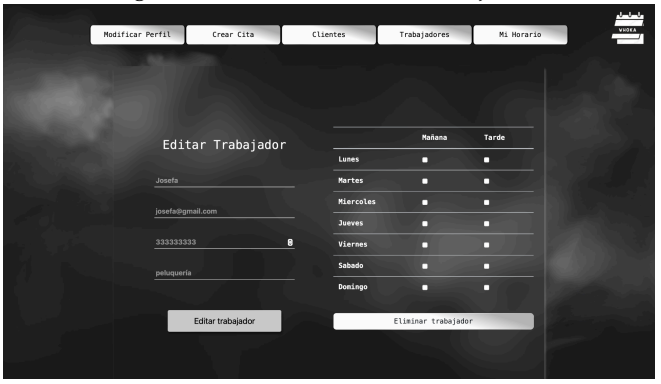

Figura 20. Pantalla de editar trabajador

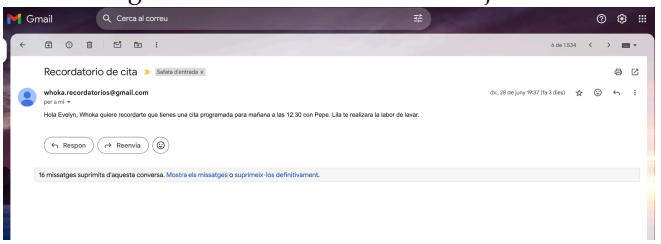

Figura 21. Correo que recibe el cliente al recordarle una cita

En el anexo podemos encontrar un diagrama que define el flujo entre pantallas.

5 CONCLUSIONES

En resumen, este informe refleja el planteamiento, el desarrollo y los resultados de este proyecto. Whoka se encuentra en una primera versión y ofrece una aplicación web funcional que cumple con los objetivos establecidos en un inicio. Aun así, como comentamos, se trata de una

primera versión y el siguiente objetivo es definir las líneas de futuro que va a tomar, por tal de crear un producto que cumpla con las necesidades de gestión de los negocios, como por ejemplo, que el recordatorio a los clientes se haga mediante whatsapp y que el usuario pueda añadir más información a los diferentes perfiles que gestiona la web.

Después de dejar claro el interés en seguir mejorando la aplicación, debo decir, que estoy satisfecha con el trabajo realizado y los resultados obtenidos. Durante el desarrollo han surgido muchos imprevistos propios de la programación que se han resuelto con éxito para dar pie a nuevos imprevistos, hasta que se ha conseguido tener un producto y no únicamente una idea. La elección de las tecnologías y metodologías empleadas han sido adecuadas y nos ofrecen un proyecto escalable, con un código entendible y una base de datos flexible. El diseño es simple, pero adecuado y facilita su uso a los usuarios con el propósito de reducir el tiempo que deben invertir en la gestión de su negocio.

En conclusión, Whoka es un proyecto que ha implicado el uso del conocimiento adquirido durante los estudios, sobre todo durante la mención cursada y además no limita la creatividad únicamente a este proyecto, sino que da pie a un proyecto más grande por el cual siento la motivación de seguir trabajando.

6 Agradecimientos

Me gustaría agradecer a todas las personas que han contribuido en mi educación, tanto en el ámbito educativo, como en el personal. Espero poder ofrecer conocimiento y sabiduría a las personas así como otros lo hacen conmigo. En especial, quiero agradecer a mi abuelo Lalo y mi padre Javi por apostar por mi educación. Y también, a mi tía Mar y mi abuela Merche por creer en mí, ofrecerme un apoyo incondicional siempre que lo he necesitado y sobretodo por ser un referente como mujeres.

7 Bibliografia

[1] Metodologias agiles. (s. f.). Recuperado marzo de 2024, de https://www.salesforce.com/mx/blog/que-son-metodologias-agiles/

[2] Martins, J. (2024, 19 enero). Recuperado marzo de 2024 ¿Qué es la metodología Kanban y cómo funciona? [2024] • Asana. Asana. https://asana.com/es/resources/what-is-kanban

[3] Free Kanban Board |Recuperado marzo de 2024 Kanban Tool. (s. f.). Kanban Tool. https://kanbantool.com/free-kanban-board

[4] J2logo. (2022, 8 abril). Tutorial Flask – Lección 4: Login de usuarios en Flask. (Mayo de 2024) J2LOGO. https://j2logo.com/tutorial-flask-leccion-4-login/

[5] MongoDB. (s. f.).Recuperado marzo de 2024 MongoDB: la plataforma de datos para aplicaciones. https://www.mongodb.com/es

[6] Gómez, P. (2022, 8 agosto). Recuperado marzo de 2024 ¿Qué es Flask en programación web? - DevCamp. DevCamp. https://devcamp.es/que-es-flask/

[7] Agendit. (s. f.). Recuperado marzo de 2024 https://agendit.com.py/#features

[8] Nubimed. (2022, 15 julio). Recuperado marzo de 2024 • Software de Gestión de Clínicas en la Nube | Nubimed. https://www.nubimed.com/?gad_source=1&gclid=CjwKCAiAi6uvBhADEiwAWiyRdhF1h3PuBexIWAl-zY4gj00hcFaAa3a2CxH3ppjw2hp_8DcZ_OVcLRoCWqM

[9] Brevo. (2024, 31 enero). Recuperado marzo de 2024 Programas para agendar citas: las 5 mejores herramientas - Brevo. Brevo. https://www.brevo.com/es/blog/mejores-programas-para-a

[10] Contributors, M. o. J. T. A. B. (s. f.). Bootstrap. (Mayo de 2024) https://getbootstrap.com/

[11] What is MongoDB? - MongoDB Manual v7.0. (s. f.). (Mayo de 2024) https://www.mongodb.com/docs/manual/

[12] QuickStart — Documentación de Flask (3.0.X). (s. f.). (Mayo de 2024) https://flask.palletsprojects.com/es/main/quickstart/

[13] Stack Overflow en español. (s. f.). Stack Overflow En Español. (Mayo de 2024) https://es.stackoverflow.com/

[14] Requisitos funcionales. (s. f.). (Abril de 2024) https://elvex.ugr.es/idbis/db/docs/design/2-requirements.

[15] Frank Andrade. (2022, 6 abril). Como crear una página web muy fácil con Python & Flask [Vídeo]. YouTube.(Abril de 2024) https://www.youtube.com/watch?v=TK6DlBl63aI

[16] Roelcode. (2023, 11 abril). 🔴Crea una lista de tareas con Flask y Python utilizando Bootstrap 5: Tutorial paso a paso [Vídeo]. YouTube.(Abril de 2024) https://www.youtube.com/watch?v=BetOsz7aCbU

[17] Codenautas. (2022, 5 julio). Aplicación CRUD con Python, Flask y MongoDB (Abril de 2024) [Vídeo]. YouTube. https://www.youtube.com/watch?v=A05cBJ_P7Q8

[18] Roelcode. (2023b, abril 11). 🔴Crea una lista de tareas con Flask y Python utilizando Bootstrap 5: Tutorial paso a paso [Vídeo]. YouTube. (Abril de 2024) https://www.youtube.com/watch?v=BetOsz7aCbU

[19] Daniel Andres Rincon. (2023, 27 noviembre). Proyecto Python Flask - MongoDB - Login y CRUD de tareas [Vídeo]. YouTube. (Abril de 2024) https://www.youtube.com/watch?v=DQABmOJErqY

[20] Sajib Hossain. (2021, 1 julio). Event Calendar in Python and Django [Vídeo]. (Abril de 2024) YouTube. https://www.youtube.com/watch?v=nTIMYHJRW1c

[21] InVision. (s. f.). Collaborate better | InVision. (Abril de 2024) https://www.invisionapp.com/

- *E-mail de contacte: 1569037@uab.cat*
- *Menció realitzada: Enginyeria del Software*
- *Treball tutoritzat per: Helena Bolta Torrell (departament)*
- *Curs 2023/24*

Julio de 2024, Escuela de Ingenieria (UAB)

8 Anexo

En el anexo se añadirá información complementaría del trabajo, que aporta valor y muestra información estructurada sobre puntos importantes del proyecto.

8.1 Planificación

En el apartado "1.4 Planificación", desglosamos las horas invertidas en el proyecto en diferentes puntos. A continuación mostramos una tabla que refleja esta planificación inicial de tareas, durante las diferentes semanas que ha durado el proyecto.

Fase	Duración	Tareas	Dedicación
Primera fase	2 semanas	• Buscar información. • Primeras reuniones con la tutora. • Desarrollo del informe inicial.	5h por semana
Segunda fase	6 semanas	• Redacción de requisitos funcionales y no funcionales. • Diseño del prototipo de la web. • Entrevista a posibles usuarios. • Comenzar el desarrollo de las funcionalidades. • Redacción del segundo informe. • Reunión con la tutora.	20h por semana
Tercera fase	4 semanas	• Centrarse en el desarrollo. • Redacción del tercer informe. • Pruebas de funcionalidades. • Reunión con la tutora.	20h por semana
Cuarta fase	3 semanas	• Acabar de pulir el código. • Hacer todas las pruebas finales. • Redactar el informe final. • Reunión con la tutora.	20h por semana
Quinta fase	1,5 semanas	• Preparar la presentación. • Reunión con la tutora. • Mejoras. • Revisar el informe global.	10h por semana
Última fase	3 dias	• Preparar la defensa ante el tribunal. • Presentar ante el tribunal.	2h al día

Figura 22. Tabla de planificación inicial

Como hemos comentado, esta tabla refleja la planificación inicial del proyecto, a lo largo del mismo, ha sufrido modificaciones según los imprevistos surgidos.

8.2 Análisis de requisitos

En el apartado "2 Diseño" nos encontramos con el análisis de requisitos funcionales y no funcionales. En el documento, numeramos los requisitos. Pero, por tal de aportar más información y estructura, hemos hecho las tablas que mostramos a continuación.

Número identificador	Nombre del requisito	Descripción	Bloque al que pertenece

RF-1	Crear cuenta	El sistema debe permitir a los usuarios crear una cuenta.	Perfil de usuario
RF-2	Iniciar sesión	El sistema debe permitir a los usuarios iniciar sesión.	Perfil de usuario
RF-3	Editar perfil	El sistema debe permitir a los usuarios editar su perfil de usuario.	Perfil de usuario
RF-4	Borrar perfil	El sistema debe permitir a los usuarios borrar su perfil.	Perfil de usuario
RF-5	Crear cliente	El sistema debe permitir a los usuarios crear un nuevo cliente.	Perfil de cliente
RF-6	Editar perfil cliente	El sistema debe permitir a los usuarios editar un cliente.	Perfil de cliente
RF-7	Borrar cliente	El sistema debe permitir a los usuarios borrar el perfil de un cliente.	Perfil de cliente
RF-8	Listar clientes	El sistema debe mostrar una lista con los clientes registrados.	Perfil de cliente
RF-9	Buscador de clientes	El sistema debe permitir buscar un cliente, previamente registrado, a través de un buscador.	Perfil de cliente
RF-10	Crear empleado	El sistema debe permitir a los usuarios crear un nuevo trabajador.	Perfil de empleado
RF-11	Editar perfil empleado	El sistema debe permitir a los usuarios editar el perfil de los trabajadores.	Perfil de empleado
RF-12	Borrar empleado	El sistema debe permitir a los usuarios borrar el perfil de sus trabajadores.	Perfil de empleado
RF-13	Listar empleados	El sistema debe mostrar una lista con los trabajadores registrados.	Perfil de empleado
RF-14	Buscador de empleados	El sistema debe permitir buscar un trabajador, previamente registrado, a través de un buscador.	Perfil de empleado
RF-15	Crear cita	El sistema debe permitir a los usuarios crear una cita.	Agendar citas
RF-16	Mostrar empleados libres	Al crear empleados el sistema debe mostrar a los empleados que tengan libre ese horario	Agendar citas

		para asignarles la cita.	
RF-17	Editar cita	El sistema debe permitir a los usuarios editar una cita ya creada.	Agendar citas
RF-18	Cancelar cita	El sistema debe permitir a los usuarios cancelar una cita.	Agendar citas
RF-19	Mostrar calendario de citas	El sistema debe mostrar al usuario un calendario entre las fechas que el usuario indique.	Visualizar calendario
RF-20	Mostrar calendario de trabajador	El sistema debe mostrar al usuario un calendario por cada trabajador, con las citas que estos trabajadores tienen asignadas próximamente.	Visualizar calendario
RF-21	Recordatorio de cita	El sistema debe enviar un mensaje recordando la cita al cliente vía correo electrónico.	Recordatorio de cita

Figura 23. Tabla de análisis de requisitos funcionales

Número identificador	Descripción	Bloque al que pertenece
RNF-1	El sistema debe permitir usar la aplicación a cualquier usuario independientemente de las características técnicas de su dispositivo.	Requisitos de rendimiento
RNF-2	El sistema debe responder a cualquier petición de una funcionalidad en menos de 4 segundos.	Requisitos de rendimiento
RNF-3	El sistema no debe permitir que se pueda acceder a la base de datos desde la interfaz gráfica.	Requisitos de seguridad
RNF-5	El sistema debe estar disponible 24/7.	Disponibilidad
RNF-6	El sistema debe ser fácilmente escalable, con el fin de hacer crecer la web y aplicar en el futuro nuevas funcionalidades.	Mantenibilidad
RNF-7	El sistema debe disponer de una buena documentación que permita realizar operaciones de mantenimiento con el menor esfuerzo posible.	Mantenibilidad
RNF-8	El tiempo de aprendizaje de un usuario, de media, debe ser menor a	Usabilidad

	30 min.	
RNF-9	El sistema debe tener un diseño UIX sencillo, claro y que permita operar con fluidez.	Usabilidad

Figura 24. Tabla de análisis de requisitos no funcionales

8.3 Diagrama de flujo entre pantallas

En el apartado "4 Resultados" mostramos capturas de las diferentes pantallas que forman la web "Whoka". A continuación, añadimos un diagrama que muestra el flujo entre pantallas y demuestra cómo un usuario loggeado puede acceder a cualquier pantalla con un máximo de 2 clicks.

Figura 25. Diagrama de flujo entre pantallas