Hotel del Glaciar

Grupo Nº 14: Mate Amargo

June 2024

INTEGRANTES:

NOMBRES Y APELLIDOS	LEGAJO	E-MAIL
Martín Gabriel Riveiro	106027	mriveiro@fi.uba.ar
Joel Pita	101666	jpita@fi.uba.ar
Bryan Jhoel Ushiña Paillacho	108731	bushina@fi.uba.ar
Emanuel Pérez Martinez	111855	eperezm@fi.uba.ar
Cristian Axel Valdez Romero	112054	cvaldez@fi.uba.ar
Carla Mendoza Coronado	107011	cmendozac@fi.uba.ar
Pablo Martinez	108973	smartinez@fi.uba.ar
Matias Woiciechowski	104837	mwoiciechowski@fi.uba.ar

1 Introducción

El presente documento tiene como estructura:

- Introducción, donde se otorga un breve resumen de la presente documentación y la problemática a resolver.
- Solución propuesta que expone en breves detalles la solución a la problemática presentada anteriormente en la introducción del documento.
- Pruebas - Validaciones define e informa todas las pruebas llevadas a cabo para la continua mejora en la calidad del proyecto.
- Plan de actividades muestra un vistazo a cómo se afrontó la construcción del proyecto, desde la gestión en la metodología de trabajo hasta la implementación del trabajo.
- Hipótesis y supuestos que llegaron a presentarse en el alcance del proyecto, la discusión y el acuerdo de aspectos como la funcionalidad del proyecto que no limiten la ejecución de la misma.
- Anexos, apartado con información a detalle de aspectos técnicos.

2 Solución Propuesta

Con el objetivo de desarrollar el sitio de alojamiento, utilizamos la metodología SCRUM para garantizar un proceso ágil y organizado. En consecuencia, nuestro equipo empleó varias herramientas y técnicas para gestionar las tareas, estimar el esfuerzo requerido y asegurar un flujo de trabajo eficiente.

2.1 Metodología SCRUM

1. **Backlog en Trello:** Creamos un Backlog en Trello donde definimos todas las actividades necesarias para completar el proyecto. La descripción de las tareas se compuso de Historias de usuarios y Criterios de aceptación.

   Las actividades definidas que marcaron la estructura de nuestro proyecto fueron:

   - Crear y configurar aplicación de frontend.
   - Crear y configurar aplicación de backend (API)

   Así, nuestro proyecto se dividió en 2 partes: **API** (que contempló el manejo de la base de datos) y **Frontend**.

   Por otro lado, el contenido del sitio quedó definido por principalmente por las siguientes tareas:

   - Crear módulo de consulta del clima en tiempo real
   - Crear página de habitaciones.
   - Crear página/módulo de contacto.
   - Desarrollar módulo de gestión de precios de habitaciones.
   - Desarrollar módulo de reservas.
   - Desarrollar módulo de reseñas.
   - Crear página ”Sobre nosotros”.
   - Crear página de manejo de errores
   - Crear la página principal
   - Crear portal de administración del sistema
   - Crear página de servicios

2. **Planning Poker:** Usamos Planning Poker para la estimación del esfuerzo para cada tarea.

3. **Calendario tentativo:** Creamos un calendario tentativo de fechas en el que definimos el orden de las tareas.

2.2 Herramientas para estructurar el proyecto

1. Arquitectura: se diseñó con Flask y Jinja2, tanto para el backend como para el frontend. Se modularizo con Blueprints en ambos casos.
2. Frontend responsive: utilizamos HTML, Css y Javascript. Para el diseño responsivo de la interfaz de usuario optamos por manejar Bootstrap.
3. Contenedores: Utilizamos Docker.
4. Base de datos: MySQL

2.3 Flujo del programa

- Inicio: El usuario accede al sitio web y es recibido por la página de inicio, que ofrece enlaces a las diferentes secciones del sitio (habitaciones, contacto, sobre nosotros, etc.) en una barra de opciones fija que además facilita el acceso a la temperatura actual en el área del hotel. A la vez a lo largo de esa pantalla se muestra una porción de cada sección destacada del sitio, con botones que llaman a entrar en ellas.
- Listado de habitaciones: Al acceder a la página de habitaciones, se muestra un listado de las habitaciones disponibles, obtenidas desde la base de datos. Cada habitación tiene un enlace que lleva a su página de detalles.
- Detalles de la habitación: Al seleccionar una habitación específica, se muestra una página con los detalles completos de la misma, incluyendo una descripción detallada, el precio (que puede actualizarse a través del panel de administrador) y la disponibilidad.
- Reservas: Los usuarios pueden realizar reservas a través de un formulario en la sección de reservas. El sistema valida los datos ingresados y guarda la reserva en la base de datos para que el administrador, desde su panel, pueda aprobarlas o rechazarlas a su criterio.
- Contacto: La página de contacto permite a los usuarios enviar mensajes al hotel desde un formulario, ademas de desplegar un listado de informacion que permiten conocer los medios de comunicación del sitio.
- Reseñas: Los usuarios pueden dejar reseñas sobre su estadía, las cuales se muestran tanto en el inicio como en una sección especial para ellas. El administrador desde el panel de reseñas tiene la opción de borrarlas, ocultarlas, mostrarlas y ademas marcarlas como favoritas (lo que hara que tengan prioridad sobre otras al mostrarse). Se muestra un numero limitado de reseñas elegidas, de manera aleatoria.

- Confirmación de reserva: Al completar una reserva, el usuario recibe un correo de confirmación con los detalles de la misma.
- Actualización de precios: Desde el panel de administrador, se puede mantener actualizados los precios de las habitaciones.

2.4 Dificultades encontradas

- Configuración de Docker: Configurar Docker y docker-compose para que funcionen correctamente con Flask y MySQL requirió ajustes continuos.
- Modularización con Blueprints: Surgió de la necesidad de mantener limpia la estructura y requirió una breve adaptación.
- Integración de módulos: Algunos módulos como el de Reseñas y Reservas interactuan entre sí, por ejemplo, es necesario que finalice el dia del Check Out para que un usuario puede dejar su opinión, ademas que para confirmar que es un usuario al enviar su reseña tiene que ingresar un código que le llega en la confirmacion de su reserva. Por lo que ambos módulos debieron estar bien coordinados.
- Uso de Git: Algunos de nosotros éramos nuevos en esto y tuvimos dificultades, especialmente en la parte de mergear nuestras ramas al main, mas aun cuando esta ya tenía algunos commits adicionales.

3 Pruebas — Validaciones

Para asegurar la calidad del proyecto, se realizaron las siguientes pruebas:

1. Se uso Azure Data Studio para ir revisando que las tablas de la base de datos fueron exitosamente creadas, tambien se uso para revisar que los elementos se guardaban de forma correcta en dicha base.
2. Pruebas de instalación correcta de todas las dependencias y inicialización de Docker según lo esperado.
3. Verificación de la accesibilidad de la página del hotel.
4. Pruebas de visualización de las habitaciones con sus respectivas imágenes y precios.
5. Pruebas de capacidad para realizar una reserva.
6. Pruebas de funcionalidad para dejar una reseña.
7. Pruebas de funcionalidades del administrador para editar precios, ordenar reseñas y gestionar reservas en la página.
8. Pruebas de comunicación entre el cliente y el administrador del hotel a través de la página de contacto, incluyendo el envío de correos electrónicos.
9. Pruebas de presentación clara, estética y bien organizada de la información en las vistas.
10. Se uso Postman para ir generarndo usuarios con el rol de admin para ver si funcionaba el acceso al panel de administracion.

4 Plan de Actividades

Nuestras actividades estuvieron determinadas por cuatro aspectos fundamentales

- La creación de un Backlog del Producto
- La estimación del esfuerzo con Planning Poker
- La confección de un cronograma
- El reparto de las tareas

4.1 Backlog del Producto

Tras la primer reunión, definimos cómo queríamos que fuera la página del hotel, para ello jugó un papel determinante la construcción de historias de usuario que permitian definir los requerimientos de nuestro sistema.

4.1.1 Historias de usuario del actor cliente

Nos dimos cuenta que los clientes podrían estar interesados en muchas más cosas que reservar o consultar por una habitación, por lo que pensando y discutiendo, llegamos a determinar una buena cantidad de funcionalidades desde la vista del un cliente.

Funcionalidad (Cliente)	Descripción
Consultar una habitación	• COMO cliente, • QUIERO consultar los detalles de las habitaciones, • PARA evaluar sus comodidades y características.
Consultar un servicio	• COMO cliente, • QUIERO consultar los servicios disponibles, • PARA evaluar si quiero hacer una reserva.
Realizar una reseña	• COMO cliente, • QUIERO dejar una reseña, • PARA compartir mi experiencia del alojamiento.
Realizar una reserva	• COMO cliente, • QUIERO realizar una reserva, • PARA alojarme en el hotel en un futuro.
Consultar ubicación en el mapa	• COMO cliente, • QUIERO consultar la ubicación en un mapa, • PARA obtener la dirección del hotel y los sitios de interés cercanos.
Consultar contacto	• COMO cliente, • QUIERO consultar contacto, • PARA comunicarme con representantes del hotel.
Consultar información del hotel	• COMO cliente, • QUIERO consultar información del hotel, • PARA saber la historia y los valores del mismo.
Consultar pronóstico del clima	• COMO cliente, • QUIERO ver el pronóstico actual en el alojamiento, • PARA informarme sobre el clima del lugar.

Table 1: Funcionalidades del proyecto de alojamiento (Cliente)

4.1.2 Historias de usuario del actor administrador

También consideramos la perspectiva de un administrador, cuyos intereses hacia el sistema podrían agregar otro tipo de funciones.

Funcionalidad (Administrador)	Descripción
Gestionar reservas	• COMO administrador, • QUIERO gestionar las reservas solicitadas, • PARA decidir si aceptarlas o rechazarlas.
Gestionar habitaciones y precios	• COMO administrador, • QUIERO gestionar el precio de las habitaciones, • PARA tener actualizada la información a los clientes y futuras reservas.
Gestionar reseñas	• COMO administrador, • QUIERO gestionar las reseñas, • PARA decidir qué comentarios mostrar o no en la página principal.

Table 2: Funcionalidades del proyecto de alojamiento (Administrador)

Basándonos en esto, cada uno de nosotros creó tareas en Trello que ayudaran a materializar esa visión inicial.

4.2 Planning Poker

Utilizamos Planning Poker para la estimación de tareas. Cada miembro del equipo asignaba puntos de esfuerzo a las tareas, y llegábamos a un consenso sobre la complejidad de cada una discutiendo vía chat.

4.3 Confección del cronograma

Trello ofrece la funcionalidad de aplicar fechas estimativas a las tareas, esto nos permitió diseñar un cronograma que nos orientó sobre qué tareas eran las más necesarias para un avance seguro.

4.4 Reparto de tareas

Cada miembro del equipo eligió una tarea acorde a su experiencia y comenzamos a trabajar en un repositorio de GitHub, donde fuimos subiendo nuestros avances en ramas particulares, asociadas al

título e ID de la tarjeta del Backlog.

5 Hipótesis y Supuestos

5.1 Ubicación y cumplimiento legal:

- El sistema está diseñado para cumplir con las normativas legales y éticas de la industria hotelera en la región específica del hotel, en este caso la República Argentina, en la capital de Tierra del Fuego.

5.2 Registro de Transacciones:

- Todas las transacciones relacionadas con las reservas y pagos se registran de manera precisa por coordinación entre cliente y facturación del hotel, por fuera de nuestro sistema.

5.3 Sobre Reservas y Cancelaciones:

- El sistema permitirá reservas de habitaciones disponibles, las cuales deben ser verificadas por un encargado del hotel. Las reservas pueden ser rechazadas o aceptadas.
- El hotel no admite cancelaciones, una vez aceptadas las reservas deben ser pagadas se use o no los servicios.

Abstract

Este documento informa el trabajo realizado por el grupo Nº 14 - Mate Amargo, cuyo tema elegido para desarrollar fue la implementación de una página web que ofrece servicio de *hospedaje*.

Como equipo se ha realizado una reunión con el objetivo de asignar las tareas que llevará a cabo cada integrante del grupo siguiendo la estructura de *Scrum* y *Kanban*, todas las actividades se dividieron con éxito mediante el uso de *Trello*, herramienta virtual útil para la gestión de proyectos y tareas.

El proyecto utiliza Docker para la contenerización, Python para el desarrollo backend, MySQL para la persistencia de datos, Flask como el framework web, JavaScript para la interactividad, y HTML para la estructura.

Abstract

This document apprises about a work carried out by Group Nº 14 - Mate Amargo, whose choosen topic for development was the implementation of a web page that provides *hosting services*.

As a team, we have done a meetting with the aim of assigning tasks that each member of the group will carry out following the *Scrum* and *Kanban* framework, all the tasks were successfully divided through *Trello*, a virtual tool useful for project and task management.

The project utilizes Docker for containerization, Python for backend development, MySQL for data persistence, Flask for web application framework, JavaScript for interactivity, and HTML for structure.

Palabras clave: Proyecto, Programación, Docker, Base, Datos, MySQL, Python, HTML, JS, javascript, Flask, Web, Desarrollo, Hotel, Hospedaje, PythonAnywhere, SCRUM, KANBAN, Trello, Gestión.

Keywords: Project, Programming, Docker, DataBase, MySQL, Python, HTML, JS, javascript, Flask, Web, Development, Hotel, Hosting, PythonAnywhere, SCRUM, KANBAN, Trello, Management.

6 Referencias

1. Bootstrap. (2024). *Bootstrap Documentation. Getbootstrap.com*.
   https://getbootstrap.com/docs/4.3/getting-started/introduction/

2. Docker Inc. (2024). *Docker Documentation.* Docker.com.
   https://docs.docker.com/

3. MySQL Documentation. (2024). *MySQL 8.0 Reference Manual.* Dev.mysql.com.
   https://dev.mysql.com/doc/refman/8.0/en/