Alquifiestas: Sensacional Eventos
Sprint 5

Hugo Eduardo Rivas Fajardo - 22500
José Santiago Pereira Alvarado - 22318
Nancy Gabriela Mazariegos Molina - 22513
Giovanni Alejandro Santos Hernández - 22523
Mauricio Julio Rodrigo Lemus Guzmán- 22461
Alexis Mesías Flores - 22562

Product Backlog

Pila del producto
https://docs.google.com/spreadsheets/d/1pHFa_t3TDiU_gIoxtiY6VaneJB60KE6gEEqGTWLpKHg/edit?gid=1739314590#gid=1739314590

Historias de Usuario

1. Como dueña de Sensacional Eventos, quiero asegurarme de que la aplicación funcione correctamente en todo momento para evitar interrupciones en las operaciones y mantener la satisfacción del cliente.
    - Descripción: Implementación de pruebas unitarias para funciones críticas de la aplicación, garantizando que el sistema sea robusto y confiable.
    - Priorización: Alta

2. Como encargada del inventario, quiero tener la confianza de que las actualizaciones de la aplicación no introducirán errores que afectan mi capacidad para gestionar el inventario, para mantener la eficiencia operativa.
    - Descripción:Desarrollar y ejecutar pruebas unitarias que verifiquen la funcionalidad relacionada con la gestión de inventarios, asegurando que cualquier cambio en el código no rompa las funcionalidades existentes.
    - Priorización: Alta

3. Como transportista, quiero que la información de los pedidos esté siempre precisa y sin errores para poder realizar las entregas de manera eficiente y sin contratiempos.
    - Descripción:Crear pruebas unitarias que validen la precisión de la información de los pedidos y la correcta integración con el sistema de transporte.
    - Priorización: Media

4. Como encargado de pedidos, quiero estar seguro de que la aplicación es fiable y que los errores humanos en la reserva de eventos se minimizan mediante pruebas exhaustivas de las funcionalidades.
    - Descripción: Implementar pruebas unitarias para las funcionalidades de reserva de eventos, asegurando que el sistema sea intuitivo y sin fallos.
    - Priorización: Alta

https://docs.google.com/spreadsheets/d/1pHFa_t3TDiU_gIoxtiY6VaneJB60KE6gEEqGTWLpKHg/edit?gid=0#gid=0

Sprint Backlog
https://docs.google.com/spreadsheets/d/1pHFa_t3TDiU_gIoxtiY6VaneJB60KE6gEEqGTWLpKHg/edit?gid=1476095036#gid=1476095036

Incremento

- Código desarrollado
    - Vínculo al repositorio: https://github.com/Rodlemus03/SensacionalEventos
- Lista de funcionalidades planificadas que se terminaron completamente.
    - Pruebas postman collections clientes
    - Pruebas postman collections productos
    - Núcleo de pruebas automatizadas
    - Prueba de Login
    - Crear cliente
    - Eliminar cliente
    - Editar cliente
    - Buscar cliente
    - Buscar producto
    - Evidencia

Prueba

Master plan de unit test:

1. Configuración Inicial

- Frameworks y Herramientas:
  - Postman para pruebas de API.
  - Selenium para automatización de pruebas de interfaz de usuario.
  - Pytest para organizar y ejecutar las pruebas.
  - Flask-Testing para facilitar las pruebas en Flask.
  - PostgreSQL como base de datos.
  - Docker para ambientes de prueba aislados (opcional pero recomendado).

2. Plan de Pruebas

Clientes (CRUD)

1. Create Cliente:
   - Prueba de éxito:
     - Enviar una solicitud POST con datos válidos para crear un cliente.
     - Verificar que el cliente se creó correctamente en la base de datos.
   - Prueba de fallo:
     - Enviar una solicitud POST con datos inválidos (e.g., campos obligatorios faltantes).
     - Verificar que se recibe una respuesta de error adecuada.

2. Read Cliente:
  - Prueba de éxito:
    - Enviar una solicitud GET para recuperar un cliente existente.
    - Verificar que los datos del cliente sean correctos.
  - Prueba de fallo:
    - Enviar una solicitud GET para recuperar un cliente inexistente.
    - Verificar que se recibe una respuesta de error adecuada.

3. Update Cliente:
  - Prueba de éxito:
    - Enviar una solicitud PUT con datos válidos para actualizar un cliente.
    - Verificar que los datos del cliente se actualizaron correctamente en la base de datos.
  - Prueba de fallo:
    - Enviar una solicitud PUT con datos inválidos.
    - Verificar que se recibe una respuesta de error adecuada.

4. Delete Cliente:
  - Prueba de éxito:
    - Enviar una solicitud DELETE para eliminar un cliente existente.
    - Verificar que el cliente se eliminó correctamente de la base de datos.
  - Prueba de fallo:
    - Enviar una solicitud DELETE para eliminar un cliente inexistente.
    - Verificar que se recibe una respuesta de error adecuada.

Inventario (CRUD)

1. Create Inventario:
  - Prueba de éxito:
    - Enviar una solicitud POST con datos válidos para crear un ítem de inventario.
    - Verificar que el ítem se creó correctamente en la base de datos.
  - **Prueba de fallo:**
    - Enviar una solicitud POST con datos inválidos.
    - Verificar que se recibe una respuesta de error adecuada.

2. Read Inventario:
  - Prueba de éxito:
    - Enviar una solicitud GET para recuperar un ítem de inventario existente.
    - Verificar que los datos del ítem sean correctos.
  - Prueba de fallo:
    - Enviar una solicitud GET para recuperar un ítem inexistente.
    - Verificar que se recibe una respuesta de error adecuada.

3. Update Inventario:
  - Prueba de éxito:

- Enviar una solicitud PUT con datos válidos para actualizar un ítem de inventario.
- Verificar que los datos del ítem se actualizaron correctamente en la base de datos.
- Prueba de fallo:
- Enviar una solicitud PUT con datos inválidos.
- Verificar que se recibe una respuesta de error adecuada.

4. Delete Inventario:
- Prueba de éxito:
- Enviar una solicitud DELETE para eliminar un ítem de inventario existente.
- Verificar que el ítem se eliminó correctamente de la base de datos.
- Prueba de fallo:
- Enviar una solicitud DELETE para eliminar un ítem inexistente.
- Verificar que se recibe una respuesta de error adecuada.

Pedidos

1. Create Pedido:
- Prueba de éxito:
- Enviar una solicitud POST con datos válidos para crear un pedido.
- Verificar que el pedido se creó correctamente en la base de datos.
- Prueba de fallo:
- Enviar una solicitud POST con datos inválidos.
- Verificar que se recibe una respuesta de error adecuada.

2. Read Pedido:
- Prueba de éxito:
- Enviar una solicitud GET para recuperar un pedido existente.
- Verificar que los datos del pedido sean correctos.
- Prueba de fallo:
- Enviar una solicitud GET para recuperar un pedido inexistente.
- Verificar que se recibe una respuesta de error adecuada.

3. Update Pedido:
- Prueba de éxito:
- Enviar una solicitud PUT con datos válidos para actualizar un pedido.
- Verificar que los datos del pedido se actualizaron correctamente en la base de datos.
- Prueba de fallo:
- Enviar una solicitud PUT con datos inválidos.
- Verificar que se recibe una respuesta de error adecuada.

4. Delete Pedido:
- Prueba de éxito:
- Enviar una solicitud DELETE para eliminar un pedido existente.
- Verificar que el pedido se eliminó correctamente de la base de datos.

- Prueba de fallo:
  - Enviar una solicitud DELETE para eliminar un pedido inexistente.
  - Verificar que se recibe una respuesta de error adecuada.

3. Automatización con Selenium

- Login:
  - Verificar que el usuario pueda iniciar sesión correctamente.
  - Verificar que los intentos de inicio de sesión fallidos muestren mensajes de error adecuados.

- Navegación:
  - Verificar que los usuarios puedan navegar entre las diferentes páginas (Clientes, Inventario, Pedidos) correctamente.

- Formularios:
  - Verificar que los formularios de creación, actualización y eliminación funcionen correctamente para cada módulo (Clientes, Inventario, Pedidos).

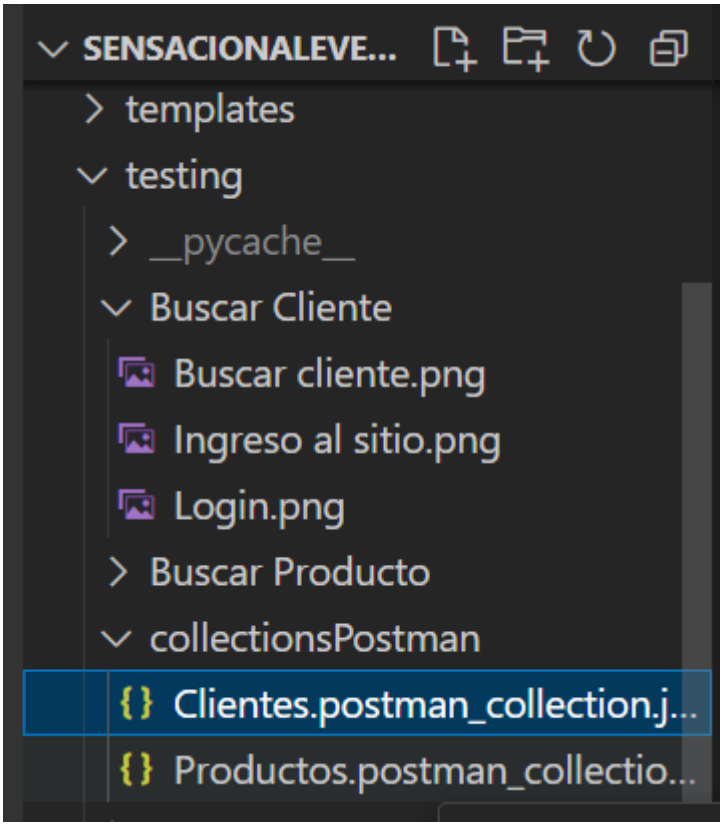
4. Ejecución de Pruebas

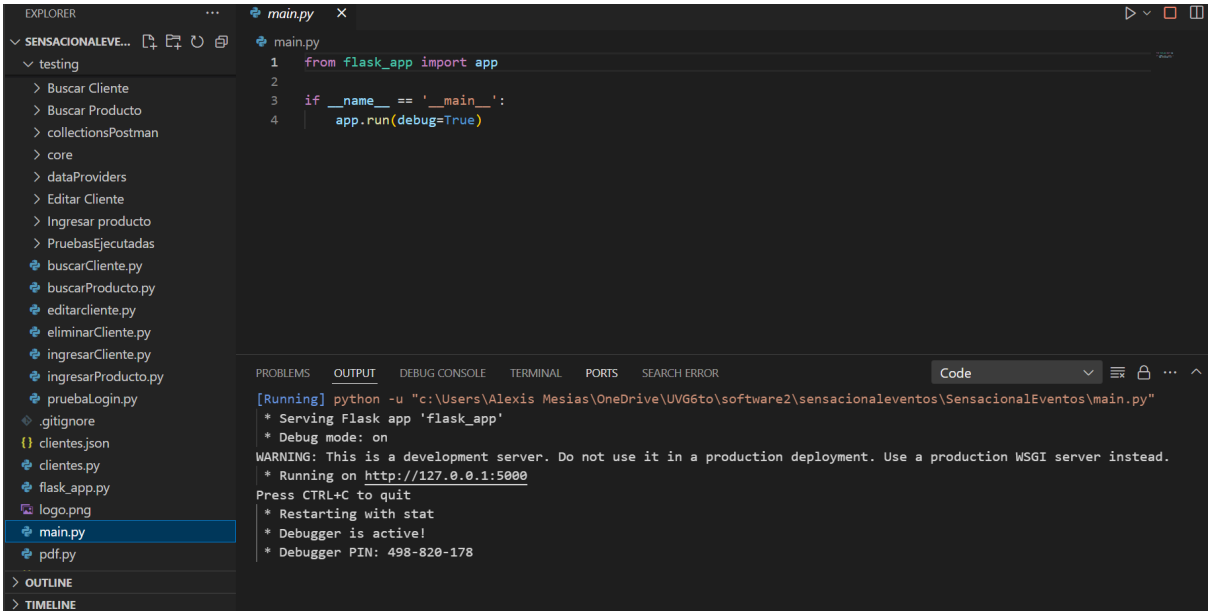
- Crear scripts de Postman para todas las pruebas de API.
- Utilizar Pytest para organizar y ejecutar las pruebas.
- Configurar Selenium para automatizar las pruebas de interfaz de usuario.
- Ejecutar las pruebas en un entorno de CI/CD para garantizar que las pruebas se ejecuten automáticamente con cada cambio en el código.

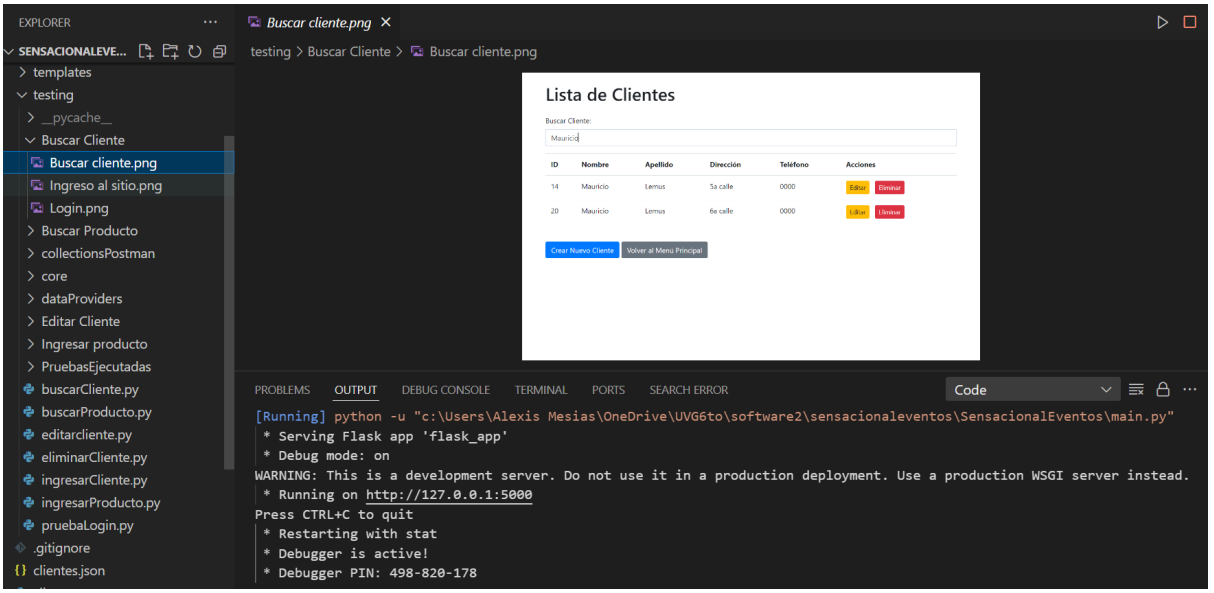
5. Informe de Resultados

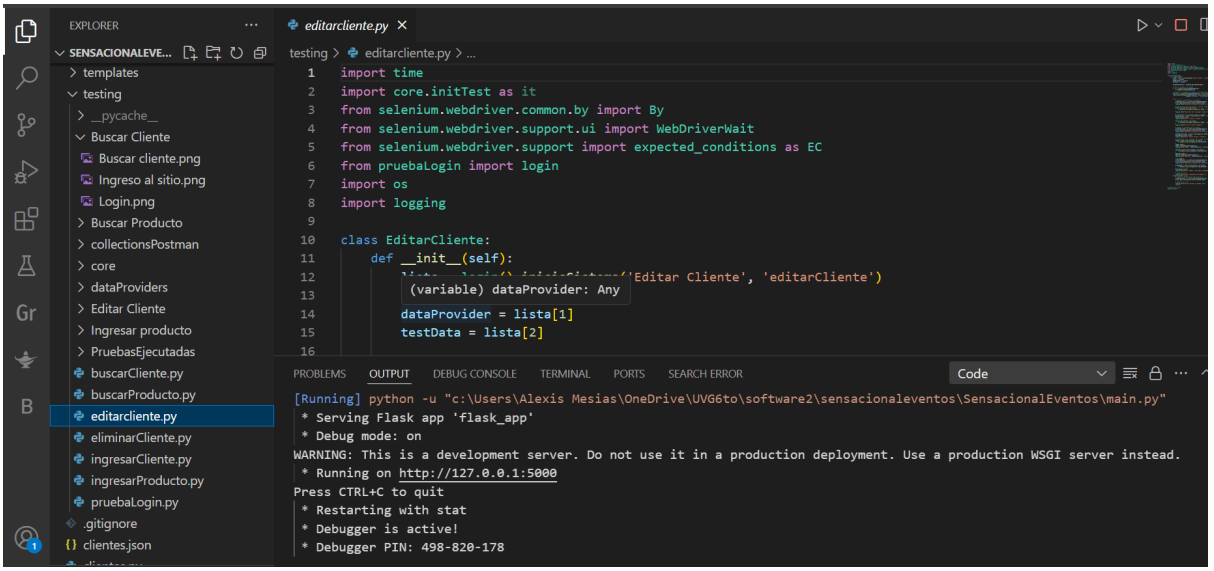
- Generar informes detallados con los resultados de las pruebas.
- Incluir información sobre las pruebas que fallaron, junto con los detalles del error.
- Revisar y corregir los errores identificados durante las pruebas.

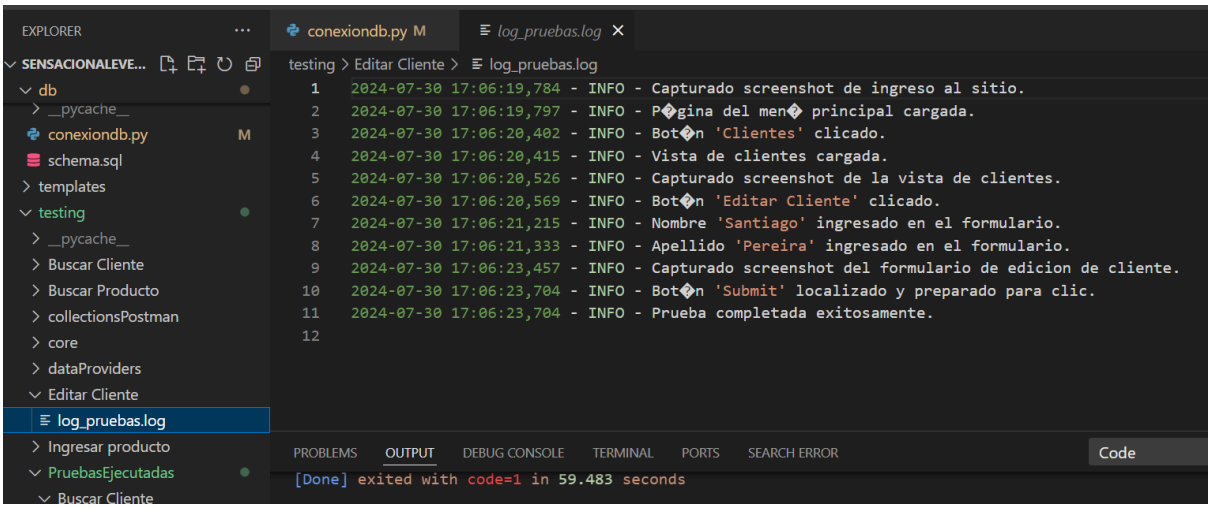
- Evidencias de implementación de pruebas unitarias en la herramienta automatizada y resultados obtenidos al ejecutarlas.





- 

- 

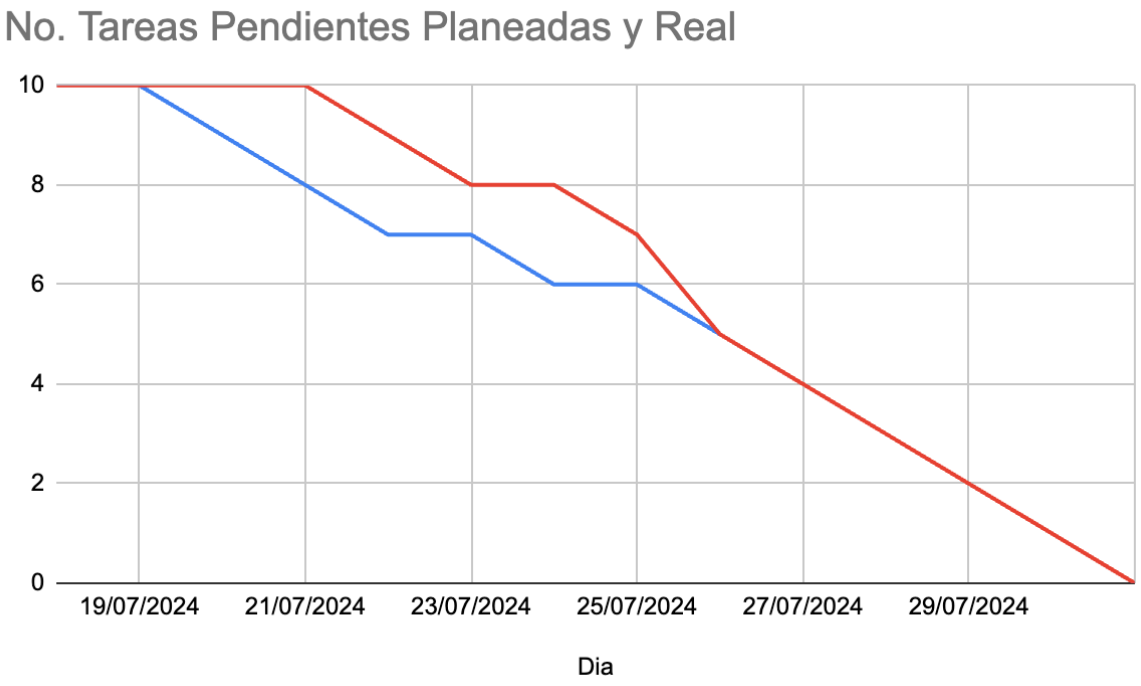
- 

- Se realizaron pruebas unitarias para este sprint utilizando Postman. Luego, se llevaron a cabo pruebas automatizadas mediante Selenium, el cual, al ejecutar la prueba, guarda una serie de capturas de pantalla que se toman automáticamente en cada paso

de la prueba. También se guarda un registro del paso a paso de cada prueba, y si se genera algún tipo de error, se describe en qué paso sucedió.

Resultados

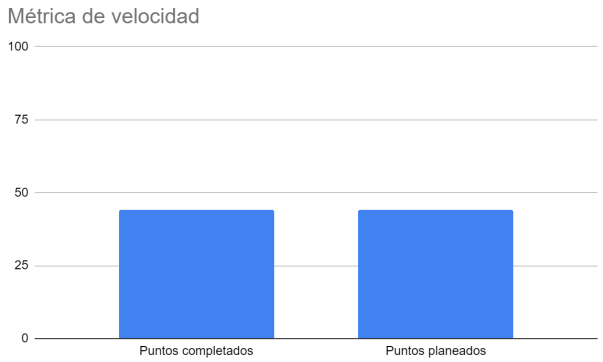
- **Gráfico burndown**



- **Métrica de velocidad**

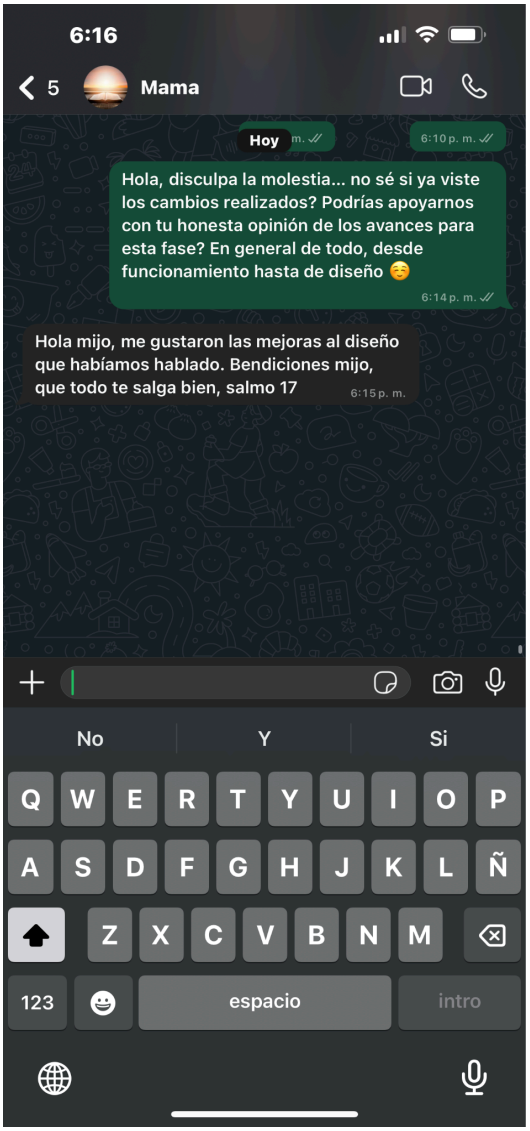
  Puntos de historia planificados: 44

  Puntos de historia terminados: 44



Velocidad del equipo en el sprint $= \frac{44}{44} \; - \; 0 \; = \; 100\%$

- Como se puede observar en los gráficos que se presentaron anteriormente, se puede observar que el equipo pudo concluir con las metas propuestas para este sprint, ya que se tenían diez tareas y se completaron las diez en el tiempo estipulado.
- Evidencias de muestra del incremento desarrollado al product owner.



- Retrospectiva del sprint.
  Para este Sprint, experimentamos un notable aumento en el rendimiento de la organización del equipo de trabajo. Este mejor desempeño se refleja claramente tanto en el progreso del sprint como en el burndown chart, indicadores que nos muestran una mejora significativa en la eficiencia y efectividad de nuestro equipo.

  A diferencia de los sprints anteriores, en este hemos abordado una variedad de tareas diferentes a las previamente asignadas. Este cambio no solo ha diversificado nuestro enfoque, sino que también ha permitido al equipo adquirir nuevas habilidades y enfrentar desafíos de manera más efectiva. Además, hemos logrado cumplir con las

pruebas planificadas dentro del tiempo establecido, lo que demuestra una mejora en nuestra capacidad para gestionar el tiempo y los recursos disponibles.

La distribución del trabajo se optimizó considerablemente, lo que nos permitió anticipar posibles conflictos relacionados con el tiempo y gestionar las cargas de trabajo de manera más equilibrada. Esta proactividad ha sido clave para evitar retrasos y asegurar que cada miembro del equipo pueda contribuir de manera efectiva sin sentirse abrumado. Gracias a estas mejoras, hemos podido mantener un ritmo constante y productivo a lo largo del sprint.

En resumen, la combinación de una mejor organización, una distribución más efectiva del trabajo y una planificación anticipada ha sido fundamental para mejorar el rendimiento del equipo en este sprint. Estamos en una posición sólida para continuar este progreso y asegurar una entrega exitosa al final de este sprint, aprendiendo de las experiencias pasadas y aplicando estas lecciones para futuros proyectos.

- Ejecución del presupuesto y el tiempo planificado.

Ejecución del presupuesto

Descripción	Monto
Server EC2	$ 20.00
Dominio	$ 1.40
Salarios (6 empleados)	$ 2,240.00
Sistema Declara Guate	$ 20.00
Renta de 6 Computadoras	$ 1,200.00
Backup Físico	$ 50.00
Internet	$ 50.00
Total	$ 3,581.40

Gestión del tiempo

Nombre: Mauricio Julio Rodrigo Lemus Guzmán
Carné: 22461

Fecha	Inicio	Fin	Tiempo Interrupción	Delta tiempo	Fase	Comentarios
26-07-24	10:00	12:00	20 minutos	1 hr 40 min	Sprint 5	Empecé la prueba automatizada para eliminar un cliente
29-07-24	2:00	2:30	0 minutos	30 minutos	Sprint 5	Finalización de la prueba automatizada
30-07-24	2:00	3:30	30 minutos	1 hora	Sprint 5	Realización de gráficos del sprint

Nombre: José Santiago Pereira Alvarado
Carné: 22318

Fecha	Inicio	Fin	Tiempo Interrupción	Delta tiempo	Fase	Comentarios
26-07-24	10:00	12:00	20 minutos	1 hr 40 min	Sprint 5	Empecé la prueba automatizada para eliminar un cliente
29-07-24	2:00	2:30	0 minutos	30 minutos	Sprint 5	Finalización de la prueba automatizada
30-07-24	2:00	3:30	30 minutos	1 hora	Sprint 5	Realización de gráficos del sprint

Nombre: Nancy Gabriela Mazariegos Moliba
Carné: 22513

Fecha	Inicio	Fin	Tiempo Interrupción	Delta tiempo	Fase	Comentarios
26/07	21:00	22:45	15 min	2.5 horas	Sprint 5	Prueba de buscar

						producto
30/07	14:30	15:30	0 min	1 hora	Sprint 5	Resultados en el documento del Sprint 5

Nombre: Hugo Eduardo Rivas Fajardo
Carné: 22500

Fecha	**Inicio**	**Fin**	**Tiempo Interrupción**	**Delta tiempo**	**Fase**	**Comentarios**
26/07	22;00	23:00	0	1 hora	Sprint 5	Prueba buscar cliente
30/07	12:00	14:00	30	1 hora y media	Sprint 5	Product Backlog(documento)

Nombre: Giovanni Alejandro Santos Hernández
Carné: 22523

Fecha	**Inicio**	**Fin**	**Tiempo Interrupción**	**Delta tiempo**	**Fase**	**Comentarios**
25/07	22:00	22:30	0	30 minutos	Sprint 5	Prueba agregar producto
28/07	12:00	13:05	5	1 hora	Sprint 5	Terminar prueba agregar producto

Nombre: Alexis Mesías Flores
Carné: 22562

Fecha	**Inicio**	**Fin**	**Tiempo Interrupción**	**Delta tiempo**	**Fase**	**Comentarios**
25/07/24	13:00	14:30	20 min	1h 10m	Sprint 5	Prueba editar cliente inicio (no funcional aun)
26/07/24	8:30	10:00	45min	45min	Sprint 5	Prueba editar cliente funcional pero solo con un cliente en especifico.

29/07/24	11:00	1:00	55 min	1h 5m	Sprint 5	Prueba Editar cliente funcional con cualquier cliente.
29/07/24	1:00	1:45	25min	20 min	Sprint 5	Se realizo la prueba repetidas veces comprobando su funcionamiento con cualquier cliente.

Referencias

- ***Postman Collections: Organize API Development and Testing*. (s. f.). Postman API Platform. https://www.postman.com/collection/**

- **García, B. (s. f.). *Boni García*. https://bonigarcia.dev/**