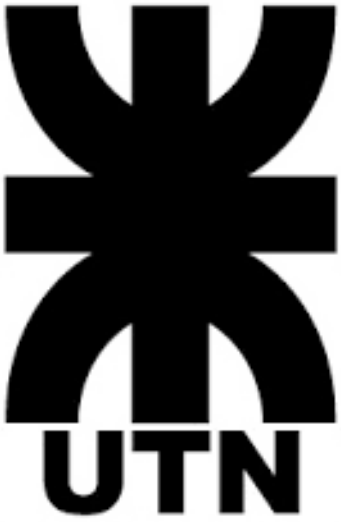
Universidad Tecnológica Nacional

Cátedra Ing. y Calidad de Software

Trabajo Práctico N°8



Tema: SCRUM - Planificación de Release y de Sprint

Fecha de entrega: 11/10/2024

Curso: 4K3

Docentes:

- Ing. Laura Covaro
- Ing. Cecilia Massano
- Ing. Georgina González

Integrantes:

- Garetto, Camila - 93569
- Barrera, Alejo Exequiel - 85380
- Montenegro, Facundo - 90329
- Toledo, Antonio Ramón - 81865
- Zalaya, Ignacio - 85641
- Morales, Emiliano Javier - 54112
- Luna, Facundo Nicolás - 89422

- Reyna, Teodoro - 89891

MVP (Minimum Viable Product)

Objetivo:

- Permitir que los clientes (pasajeros) puedan trasladarse al lugar que desean solicitando un taxi cercano.
- Permitir a los taxistas ver la ubicación del pasajero que solicita un viaje para llevarlo.
- Permitir a los taxistas manejar el estado del taxi.

Justificación:

Se va a desarrollar una aplicación mobile, incluimos las siguientes user stories:

- Loguear taxista: debido a que el taxista debe estar logueado para poder recibir, visualizar los pedidos y poder seleccionar alguno.
- Ocupar taxista: debido a que es necesario que el taxista informe su estado de ocupado para no recibir más pedidos.
- Liberar taxi: debido a que es necesario que el taxista informe su estado de libre para recibir más pedidos.
- Ver ubicación del pasajero: debido a que es absolutamente necesario para que el taxista pueda ver en dónde está el pasajero y poder iniciar su viaje para buscarlo.
- Buscar taxis cercanos: posibilita a los pasajeros poder ver los taxis más cercanos para poder pedir un taxi a su conveniencia. Además nos permite resolver la obtención de datos de geoposicionamiento y tiempos.
- Pedir taxi: permite efectivamente pedir un taxi para el pasajero e iniciar la transacción
- Notificar al taxista solicitud de taxi: permite enviar una notificación que el pasajero ha solicitado un viaje, tanto para que el taxista inicie el viaje, además nos permite resolver el envío de mensaje de tipo push.

No se incluyen los aspectos de loguear pasajero ya que los mismos van a ser cargados por base de datos, ni tampoco ver mapas taxi debido a que es demasiado complejo para agregarlo en nuestro mvp y no nos aporta gran valor de negocio,tampoco se realizará la implementación con la interfaz de facebook para loguear taxista, debido a que no pertenece al mvp por su complejidad y será resuelto a futuro por la historia de usuario ‘Loguear Pasajero’. No se incluirá todo lo relacionado a la central de taxis debido a que no compete a este mvp porque no nos sirve para validar la hipótesis propuesta

Criterio:

Las user stories incluidas permiten validar la idea de negocio en el mercado, focalizando en la funcionalidad que pone en contacto a pasajeros con taxistas. Si bien los taxistas deben poder iniciar sesión para vincularlos con los datos de su vehículo, el análogo para el pasajero no es requisito para poder realizar un pedido. La funcionalidad relacionada con la gestión de taxis por parte de la central no aporta el valor significativo para la validación del mercado por lo que se dejará para futuras iteraciones.

Frases verbales de las historias de usuario incluidas en el MVP (junto con Story Points): Rol Taxista:

- Loguear taxista -> 2 SP.
- Ocupar taxi -> 2 SP.
- Liberar taxi -> 2 SP.
- Ver ubicación del pasajero -> 5 SP.

Rol Pasajero:

- Buscar taxis cercanos -> 3 SP.
- Pedir taxi -> 5 SP.
- Notificar al taxista solicitud de taxi -> 3 SP.

Plan de Release.

Para poder definir el plan de Release, primero debemos entender cuál es la capacidad del equipo y su contexto.

Contexto del Equipo

El proyecto iniciará el 14-10-2024 y como tenemos estimados 2 sprints (4 semanas en total), se finalizará el 08-11-2024 (no hay ningún feriado en este periodo).

Con respecto a la capacidad, nos basamos en los principios de la filosofía ágil para el trabajo continuo y de calidad. Somos un equipo auto-organizado que busca la excelencia técnica a través de:

- Comunicación directa y efectiva.
- Responsabilidad.
- Trabajo en equipo de calidad.
- Ambiente de confianza

Cada sprint tendrá una duración de 2 semanas, con 10 días hábiles para el trabajo (no se cuentan los fines de semana).

Miembro	Días Disponibles	Horas/Días	Horas de esfuerzo disponibles.
Garetto, Camila	10	4	40
Barrera, Alejo Exequiel	10	4	40
Montenegro, Facundo	10	3	30
Toledo, Antonio Ramón	10	3	30
Zalaya, Ignacio	10	5	50
Morales, Emiliano Javier	10	3	30
Luna, Facundo Nicolás	10	4	

Reyna, Teodoro	10	3	30
		Total	290

Las horas disponibles de trabajo de cada integrante fue calculada dependiendo sus horarios de cursado, de actividades particulares y trabajo en cada caso.

Capacidad del Equipo en Horas Ideales: 290 hs.

Ceremonias de SCRUM

- **Sprint Planning**: normalmente, en un sprint de duración de un mes, el Sprint Planning tiene una duración de 8 horas. Como nuestros Sprint tendrán una duración de dos semanas, podemos asumir que el Sprint Planning debería durar como máximo **4 horas** de cada miembro por cada Sprint.
- **Daily meetings**: las “daily” por lo general tienen una duración de 15 minutos por día. Además, se realizan a lo largo de todo el Sprint. En un Sprint de dos semanas tenemos 10 días hábiles (sin tener en cuenta excepciones como feriados). Por ende, calculando, 15 minutos * 10 días hábiles = 150 minutos. Estos 150 minutos equivalen a **2,5 horas** de cada miembro dedicadas a las daily meetings por Sprint.
- **Sprint Review:** por lo general, se dedican 4 horas reloj al Sprint Review en un Sprint de un mes. Como nuestros Sprint tienen una duración de 2 semanas, asumimos que el Sprint Review tendrá una duración de **2 horas** de cada miembro por Sprint.
- **Sprint Retrospective:** se utilizan 3 horas dedicadas para un Sprint Retrospective en un Sprint de un mes. Como nuestros Sprint tienen una duración de dos semanas, se asume que se necesitarán **1,5 horas** de cada miembro para realizar el Sprint Retrospective en nuestro Sprint Planning.

Tiempo total insumido en ceremonias por sprint por integrante del equipo: **10 hs**.

Teniendo en cuenta este tiempo, lo que queda en Horas para el equipo son **210 Hs por sprint**.

El release contará con 2 Sprints de 2 semanas cada uno

Sprint 1				
Prioridad	US	SP	Horas Ideales	Fecha Fin
1	Loguear Taxista	2	38	25-10-2024
2	Ocupar Taxi	2	32	
3	Liberar Taxi	2	36	
4	Pedir Taxi	5	94	
Total		11	200	

Sprint 2				
Prioridad	US	SP	Horas Ideales	Fecha Fin
5	Buscar Taxis Cercanos	3	54	08-11-2024

6	Ver Ubicación del pasajero	5	96
7	Notificar a taxista solicitud de taxi	3	51
Total		11	201

Minuta para Sprint Planning

Sprint Nro.: 1
Duración del Sprint en días: 10
Objetivo del Sprint: Desarrollar e implementar las funcionalidades del MVP que permitan a los pasajeros solicitar taxis cercanos. Incluimos el logueo del taxista como base para poder trabajar con este rol en el próximo Sprint y establecer así la base funcional del sistema. Este Sprint también servirá para medir la velocidad del equipo y ajustar la planificación para los siguientes Sprints en función de los resultados obtenidos.

Equipo Scrum:

- Barrera, Alejo Exequiel - Scrum Master.
- Garetto, Camila - BackEnd Developer.
- Luna, Facundo Nicolás - Tester QA.
- Montenegro, Facundo - Diseñador UX/UI.
- Morales, Emiliano Javier - FrontEnd Developer.
- Reyna, Teodoro - FullStack Developer.
- Toledo, Antonio Ramón - FullStack Developer.
- Zalaya, Ignacio - BackEnd Developer.

Capacidad del Equipo en Horas Ideales:

Definition Of Done

☐**Diseño Revisado**

- ☐ Se ha revisado la interfaz y se verificó que cumple con los criterios de paletas de colores, formas de íconos, etc.
- ☐ Se ha revisado la interfaz y se verificó que es intuitiva y fácil de usar, con la curva de aprendizaje lo más suave posible para cumplir con la simplicidad requerida por el pasajero.
- ☐ Se ha revisado la interfaz y el diseño cumple con la menor cantidad de interacciones por parte del taxista.
- ☐ Se aplicaron las sugerencias de mejora.

☐ **Código Completo**

- ☐ Código refactorizado: se ha revisado el código y se refactoriza de acuerdo con las necesidades de las User Stories.
- ☐ Código con formato estándar: Código llevado a formato estándar de acuerdo con el documento de estilos de código del equipo.
- ☐ Código comentado: se realizaron comentarios en cada clase/método que requiera una explicación para su entendimiento.
- ☐ Código en el repositorio: Se aplica la práctica de integración continua para cada User Story.
- ☐ Código inspeccionado: El código se inspeccionó por más de un miembro del equipo y cumple con todos los lineamientos de código explicitados en el documento de estilos de código.

☐ Documentación de Usuario Actualizada: La documentación de usuario se encuentra actualizada con toda la información asociada a la usabilidad del software, seguridad, recomendaciones, etc.
☐ Documentación técnica Actualizada: La documentación técnica está actualizada para reflejar todos los nuevos cambios introducidos que es de interés documentar (documentación de arquitectura, de arquitectura de Base de Datos, etc).
☐ Pruebas
- ☐ Pruebas de unidad Aceptadas:
  - ☐ Verificar que se envíen notificaciones al taxista y éste las reciba.
  - ☐ Verificar que el taxista pueda rechazar o aceptar solicitudes correctamente.
  - ☐ Verificar que el servicio de búsqueda de taxis cercanos devuelva resultados cercanos a la ubicación del pasajero.
  - ☐ Verificar que el inicio de sesión se realice correctamente.
- ☐ Pruebas de regresión aceptadas:
  - ☐ Verificar que las credenciales antiguas de los taxistas sigan siendo válidas luego de cambios en la aplicación.
  - ☐ Comprobar que el proceso de pedido de taxi siga siendo rápido luego de cambios en la interfaz de usuario.

☐ Cero defectos conocidos
☐ Pruebas de Aceptación realizadas