21}

1. **PARTE I**

| **1. Antecedentes Personales** |
| --- |
| A continuación, se presenta una tabla en la que debes completar la información solicitada. |

| Nombre estudiante | **Cristhofer Rossel, Sebastian Michell, Victor Muñoz** |
| --- | --- |
| Rut | **21.509.075-2 , 21.390.679-8 , 21.052.147-K** |
| Carrera | **Ing. En Informática** |
| Sede | **Antonio Varas** |

| **2. Descripción Proyecto APT** |
| --- |
| En la descripción debes señalar brevemente el nombre de tu proyecto APT y las competencias del perfil de egreso que vas a poner en práctica. Si en tu carrera están definidas las áreas de desempeño, también menciona a qué áreas de desempeño está vinculado el proyecto. |

| Nombre del proyecto | DuocGO |
| --- | --- |
| Área (s) de desempeño(s) | Desarrollo de software y aplicaciones móviles.  Gestión de bases de datos y sistemas de información.  Innovación tecnológica aplicada a servicios. |
| Competencias | * Ofrecer propuestas de solución informática, analizando los procesos de compra y consumo en la universidad. * Desarrollar soluciones de software aplicando metodologías sistemáticas y buenas prácticas de codificación. * Construir modelos de datos para soportar la gestión del stock de productos en tiempo real. * Programar consultas y rutinas que manipulen información en la base de datos para reflejar la disponibilidad de productos. * Construir el modelo arquitectónico de una aplicación móvil escalable e integrada con medios de pago. * Implementar soluciones sistémicas integrales para automatizar procesos de compra y retiro de productos. * Gestionar proyectos informáticos con metodologías ágiles en un equipo reducido de desarrollo. * Elaborar un proyecto innovador que agregue valor al contexto universitario, respondiendo a necesidades reales de estudiantes y profesores con tiempos acotados entre clases. * Planificar y organizar tareas en sprints, asignando responsabilidades y estableciendo prioridades para cumplir con los objetivos del proyecto. * Aplicar técnicas de gestión de riesgos para anticipar, identificar y mitigar posibles retrasos o problemas en la ejecución del proyecto. * Monitorear y evaluar el progreso del proyecto mediante métricas de rendimiento y reportes de avance. * Fomentar la comunicación efectiva y la colaboración entre los miembros del equipo de desarrollo y los stakeholders. * Liderar reuniones de planificación, revisión y retrospectiva para garantizar la mejora continua del proceso de trabajo. * Gestionar la documentación del proyecto asegurando trazabilidad, claridad y disponibilidad de la información relevante. * Negociar y coordinar con usuarios finales y áreas interesadas para alinear las funcionalidades del sistema con los requerimientos institucionales. |

| 3. Fundamentación Proyecto APT |
| --- |
| A continuación, se presentan distintos campos que debes completar con la información solicitada. Esta sección busca que describas en detalle tu proyecto y justifiques su relevancia y pertinencia. |

| Relevancia del proyecto APT | El proyecto DuocGO busca responder a la problemática de los tiempos reducidos entre clases que enfrentan estudiantes y profesores dentro de la universidad. Muchas veces, las personas disponen de apenas 10 minutos para trasladarse de una sala a otra, lo que vuelve inviable comprar alimentos en los locales debido a las filas y la saturación en horarios peak.  La aplicación permitirá realizar pedidos con anticipación y programar el retiro, de manera que los usuarios puedan optimizar ese tiempo limitado y acceder igualmente a productos de los locales universitarios.  La relevancia para la carrera de Ingeniería en Informática está en que este proyecto representa una solución tecnológica aplicada a un problema real del entorno académico, integrando bases de datos, sistemas en tiempo real, pagos en línea y metodologías ágiles, competencias altamente demandadas en el campo laboral. |
| --- | --- |
| Descripción del Proyecto APT | El objetivo principal del proyecto es facilitar la compra de alimentos dentro de la universidad a estudiantes y profesores que disponen de tiempos reducidos entre clases, permitiéndoles realizar pedidos anticipados y retirarlos en el momento justo sin necesidad de hacer filas.  En la primera fase, la aplicación se enfocará en productos fríos, con la condición de que los pedidos se realicen al menos 10 minutos antes del retiro. La app integrará bases de datos para mantener el stock actualizado en tiempo real y permitirá pagos en línea, incluyendo el uso de Beca JUNAEB mediante la validación con RUT, clave dinámica y confirmación por correo electrónico. |
| Pertinencia del proyecto con el perfil de egreso | El proyecto se vincula directamente con el perfil de egreso de Ingeniería en Informática, ya que implica:  Levantamiento y análisis de requerimientos para identificar la necesidad de optimizar tiempos de compra.  Desarrollo e integración de sistemas (app + base de datos + pasarela de pago).  Construcción de modelos de datos escalables y consultas para el manejo del stock en tiempo real.  Implementación de soluciones tecnológicas innovadoras aplicadas a un problema real del entorno académico.  Gestión de un proyecto informático en equipo, aplicando metodologías ágiles (Scrum adaptado).  De esta manera, el proyecto no solo responde a una necesidad real, sino que también exige aplicar de forma práctica las competencias definidas en el perfil de egreso. |
| Relación con los intereses profesionales | Mis intereses profesionales se centran en el desarrollo de software, especialmente en el área de backend, bases de datos e integración de sistemas.  El proyecto DuocGO refleja directamente estos intereses, ya que requiere diseñar la lógica del sistema, manejar la persistencia de datos y aplicar conceptos de arquitectura escalable. Además, me permitirá adquirir experiencia en la creación de un producto mínimo viable (MVP) con impacto real, fortaleciendo mis habilidades en desarrollo, innovación y trabajo en equipo. |
| Factibilidad de desarrollo del Proyecto APT | El proyecto es factible porque:  El semestre y las horas asignadas permiten desarrollar una primera fase funcional enfocada en productos fríos.  Los materiales requeridos son principalmente software de desarrollo, frameworks y servicios gratuitos o de bajo costo.  Los factores que facilitan su desarrollo incluyen el conocimiento adquirido en asignaturas previas y la buena coordinación de un equipo de tres personas.  Los factores que dificultan podrían ser la limitación de tiempo y la carga académica paralela, pero se mitigará mediante la planificación en sprints cortos con Scrum adaptado. |

1. PARTE II

| 4. Objetivos |
| --- |
| En este apartado debes definir objetivos generales y específicos del Proyecto APT. Es importante aclarar que los objetivos se deben plantear en forma clara, concisa y sin dar mayores explicaciones, es decir, deben entenderse por sí solos. Se sugiere redactarlos utilizando un verbo en infinitivo, pues ello obliga a precisar acciones concretas. |

| Objetivo general | Desarrollar una aplicación móvil que permita a estudiantes y profesores de la universidad realizar pedidos anticipados de alimentos con stock actualizado en tiempo real, facilitando su compra en tiempos reducidos entre clases. |
| --- | --- |
| Objetivos específicos | Diseñar la arquitectura del sistema y el modelo de datos que soporte la aplicación.  Implementar un catálogo de productos enfocado en alimentos fríos, con actualización automática de stock.  Desarrollar la funcionalidad de pedidos anticipados con un mínimo de 10 minutos de anticipación para el retiro.  Integrar métodos de pago en línea, incluyendo la validación con Beca JUNAEB.  Configurar notificaciones y confirmaciones de pedido por correo electrónico.  Aplicar una metodología de trabajo Scrum de desarrollo que permita gestionar el proyecto en un equipo de tres integrantes. |

| 5. Metodología |
| --- |
| En el siguiente apartado deberás describir la metodología, propia de tu disciplina, que utilizarás para resolver el proyecto APT antes descrito, incluyendo las etapas y métodos de trabajo. |

| Descripción de la Metodología |
| --- |
| **Para el desarrollo del proyecto se utilizará una metodología ágil, específicamente Scrum, ya que permite organizar el trabajo en iteraciones cortas, priorizar funcionalidades según las necesidades del usuario y adaptarse a cambios de manera flexible durante el proceso.**  **La metodología se aplicará de la siguiente manera:**   1. **Planificación del proyecto**     * **Definición de objetivos generales.**    * **Identificación de requerimientos iniciales.**    * **Creación del Product Backlog con todas las historias de usuario.** 2. **Planificación de Sprints**     * **Selección de las historias de usuario más prioritarias.**    * **Estimación del esfuerzo mediante Planning Poker usando la secuencia de Fibonacci.**    * **Definición del objetivo del sprint y asignación de tareas.** 3. **Ejecución del Sprint**     * **Desarrollo de funcionalidades en un lapso de 2 a 3 semanas.**    * **Reuniones diarias (Daily Scrum) para coordinar avances, detectar impedimentos y ajustar el plan de trabajo.** 4. **Revisión del Sprint**     * **Presentación de las funcionalidades terminadas al final de cada sprint.**    * **Retroalimentación inmediata de los interesados.** 5. **Retrospectiva**     * **Identificación de mejoras en el proceso de trabajo.**    * **Ajustes en la dinámica del equipo para optimizar la eficiencia en los siguientes sprints.** 6. **Validación y pruebas**     * **Ejecución de pruebas funcionales y de usabilidad.**    * **Ajustes según resultados para garantizar calidad y cumplimiento de requisitos.**   **Con esta metodología se busca asegurar un desarrollo iterativo e incremental, manteniendo una comunicación constante con los usuarios finales y garantizando que el producto final se ajuste a las necesidades reales de la comunidad universitaria.**  **Product Owner (PO (Sebastian Michell)): responsable de priorizar funcionalidades y asegurar que el producto responda a la necesidad de estudiantes y profesores.**  **Scrum Master (Victor Muñoz): encargado de coordinar el trabajo, organizar las reuniones y facilitar la resolución de bloqueos.**  **Desarrolladores: todos los integrantes comparten la responsabilidad de codificar, diseñar y probar las funcionalidades, aunque con focos distintos:**  **Victor Muñoz: modelado de datos y lógica de backend.**  **Cristhofer Rossel: desarrollo de la interfaz y experiencia de usuario.**  **Sebastian Michell: integración de pagos, notificaciones y pruebas de calidad.** |

| **6. Evidencias** |
| --- |
| A continuación, describe qué evidencias serán evaluadas en el informe de avance y en el informe final de tu proyecto APT. Estas evidencias deben ser acordadas con tu docente. Se entenderá por evidencia los productos que se desarrollen durante el proyecto y cuyo propósito sea visibilizar o documentar cómo se ha implementado el trabajo. |

| **Tipo de evidencia** | **Nombre de la evidencia** | **Descripción** | **Justificación** |
| --- | --- | --- | --- |
| Avance | Documento de requerimientos y backlog | Contiene las historias de usuario, priorización y criterios de aceptación definidos. | Permite evidenciar que se levantaron correctamente las necesidades del usuario y que el equipo cuenta con una guía clara de desarrollo. |
| Avance | Prototipo de interfaz (wireframes/mockups) | Diseño visual inicial de la aplicación móvil con sus pantallas principales. | Sirve para validar con los usuarios la estructura y flujo de la aplicación antes del desarrollo. |
| Avance | Planificación de Sprints | Cronograma de los sprints con la asignación de historias de usuario y estimaciones de esfuerzo. | Muestra la organización del equipo y la aplicación de la metodología ágil. |
| Final | Código fuente de la aplicación | Desarrollo completo de la aplicación en su versión funcional. | Es el producto central del proyecto y representa la implementación técnica de los requerimientos. |
| Final | Manual de usuario | Documento que explica cómo utilizar la aplicación y sus principales funcionalidades. | Facilita la adopción por parte de los estudiantes y usuarios finales. |
| Final | Informe final del proyecto | Documento con la descripción del proceso seguido, metodología aplicada, resultados y conclusiones. | Permite evaluar de manera integral la gestión del proyecto y su impacto esperado. |
| Final | Video demostrativo | Grabación mostrando la navegación y funcionalidades de la aplicación. | Permite evidenciar de manera clara y práctica el funcionamiento del sistema. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

| **7. Plan de Trabajo** |
| --- |
| En la siguiente tabla define la planificación de tu Proyecto APT de acuerdo a lo requerido. |

| **Plan de Trabajo Proyecto APT** | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| **Competencia o unidad de competencia** | **Nombre de actividades/tareas** | **Descripción actividades/tareas** | **Recursos** | **Duración de la actividad** | **Responsable** | **Observaciones** |
| Levantamiento de requerimientos | Reunión inicial con estudiantes y profesores | Entrevistas y encuestas para identificar necesidades y problemas a resolver. | Computador, cuestionarios, sala de reunión | 5 días | **PO: Sebastian Michell** | Validar con docente y usuarios clave. |
| Marco de trabajo Scrum | Elaboración del Product Backlog | Redacción y priorización de historias de usuario. | Trello/Jira, PC, conexión a internet | 3 días | **PO: Sebastian Michell + Scrum Master: Victor Muñoz** | Backlog debe estar aprobado por el docente. |
| Diseño de interfaz | Prototipado en Figma | Creación de mockups y flujo de pantallas. | Figma, PC, conexión a internet | 7 días | **Cristhofer Rossel** | Se revisará con usuarios antes de pasar a desarrollo. |
| Desarrollo backend | Implementación de API y modelado de datos | Programación de los servicios para gestión de pedidos, usuarios y productos. | PC, IDE (VSCode/IntelliJ), servidor local | 15 días | **Victor Muñoz** | Priorizar seguridad y escalabilidad. |
| Desarrollo frontend | Implementación de aplicación móvil | Programación de la app móvil conectada al backend. | PC, Android Studio/React Native, emuladores | 15 días | **Cristhofer Rossel** | Debe cumplir con los prototipos validados. |
| Integraciones clave | Pagos y notificaciones | Desarrollo de la integración de métodos de pago y sistema de notificaciones. | SDKs de pago, librerías de notificaciones | 10 días | **Sebastian Michell** | Asegurar cumplimiento de seguridad en pagos. |
| Pruebas y validación | Testing funcional y de usabilidad | Realizar pruebas unitarias, integración y pruebas con estudiantes. | Herramientas de testing, usuarios de prueba | 10 días | **Sebastian Michell + todo el equipo** | Documentar incidencias y mejoras. |
| Documentación | Manuales y guía técnica | Preparación de manual de usuario y documentación técnica del sistema. | Word, PDF, PC | 5 días | **Sebastian Michell (líder)** | Entregar junto al informe final. |
| Entrega del proyecto | Presentación y demo final | Exposición de resultados, funcionamiento de la app y aprendizajes. | Proyector, PC, app instalada | 2 días | **Todo el equipo** | Evaluación final del docente. |

| **8. Product Backlog** |
| --- |
|  |

# Introducción

En el contexto universitario, uno de los principales problemas que enfrentan estudiantes y profesores es la limitación de tiempo entre clases. Con pausas que muchas veces no superan los diez minutos, la complejidad de trasladarse entre salas y, al mismo tiempo, realizar compras en los locales de alimentación del campus. Esto genera largas filas en horarios de mucha gente, insatisfacción de los usuarios y, en algunos casos, la imposibilidad de comprar un producto a tiempo.

Frente a esta problemática, surge DuocGO, una aplicación móvil cuyo propósito es optimizar la experiencia de compra dentro de la universidad mediante un sistema de pedidos anticipados. La app permitirá a los usuarios revisar el catálogo de productos disponibles (inicialmente enfocados en alimentos fríos), seleccionar lo que desean comprar y programar el retiro con un mínimo de diez minutos de anticipación. Además, contará con pagos en línea, incluyendo la integración con la Beca JUNAEB, lo que ofrece un método de compra rápido, seguro y adaptado a la realidad de los estudiantes.

Para garantizar la confianza y transparencia en el proceso, la aplicación integrará una base de datos con stock actualizado en tiempo real, evitando errores en la disponibilidad de productos, y notificará a los usuarios a través de correos electrónicos y códigos QR que facilitarán el retiro en los locales.

El proyecto será desarrollado aplicando la metodología ágil Scrum, dividiendo el trabajo en sprints cortos que permitan validar avances y recibir retroalimentación constante de los usuarios. Los roles dentro del equipo están claramente definidos (Product Owner, Scrum Master y Desarrolladores), asegurando la organización del trabajo y la correcta priorización de las funcionalidades críticas.

En definitiva, DuocGO no solo busca resolver una necesidad concreta en la vida universitaria, sino que también representa una oportunidad para aplicar de manera práctica los conocimientos adquiridos en el área de desarrollo de software, gestión de datos, metodologías ágiles y arquitectura de aplicaciones móviles, aportando una solución innovadora que agrega valor al entorno académico.

# Definición de Roles y Responsabilidades del Equipo

El equipo de trabajo del proyecto DuocGo se organiza bajo la metodología ágil Scrum, con roles claramente definidos que aseguran una correcta priorización, coordinación y desarrollo del sistema.

Product Owner & Developer – Sebastián Michell

* Responsable de definir y priorizar el Product Backlog.
* Asegurar que el producto responda a las necesidades de estudiantes y profesores.
* Establecer las funcionalidades críticas y validar que los entregables generen valor real.
* Participar como desarrollador en la implementación de pagos, notificaciones y pruebas de calidad.

Scrum Master & Developer – Victor Muñoz

* Encargado de coordinar el trabajo del equipo y organizar las ceremonias Scrum (Sprint Planning, Daily, Review, Retrospectiva).
* Facilitar la resolución de impedimentos y asegurar el correcto uso de marco de trabajo Scrum
* Promover la comunicación efectiva y la mejora continua dentro del equipo.
* Como desarrollador, se centra en el modelado de datos y la lógica de backend.

Developer (Frontend/UI) – Cristhofer Rossel

* Responsable del desarrollo de la interfaz móvil.
* Diseño de la experiencia de usuario (UX), asegurando usabilidad y accesibilidad.
* Validación del flujo de compra y retiro de productos.
* Colaboración en pruebas y retroalimentación continua de la experiencia de usuario.

Stakeholder – Aldo Martínez

* Representa a los usuarios finales (estudiantes y profesores).
* Valida los objetivos del proyecto y sus entregables.
* Proporciona retroalimentación constante sobre funcionalidades y usabilidad.
* Asegura que el producto final cumpla con las necesidades reales de la comunidad estudiantil.

# 

# Matriz RACI

| **Tarea** | **Product Owner (Sebastián Michell)** | **Scrum Master (Victor Muñoz)** | **Developer (Cristhofer Rossel)** | **Stakeholder (Aldo Martinez)** |
| --- | --- | --- | --- | --- |
| Priorización del Product Backlog | **A/R** | **C** | **I** | **C** |
| Estimación de esfuerzo | **C** | **R** | **A** | **I** |
| Planificación de Sprint | **C** | **R** | **A** | **I** |
| Moderación de reuniones diarias | **I** | **R** | **A** | **I** |
| Remoción de impedimentos | **I** | **R** | **C** | **I** |
| Revisión del Sprint | **A** | **R** | **R** | **C** |
| Retrospectiva del Sprint | **C** | **R** | **A** | **I** |
| Validación de entregables | **A** | **C** | **R** | **C/R** |

# Definición de epicas e historias de usuario

Para definir la visión estratégica y las grandes áreas de enfoque que guiarán el desarrollo de este proyecto, hemos consolidado los requerimientos de alto nivel en las siguientes Épicas. Estas representan los principales bloques de valor que se entregarán:

| **Código epica** | **Nombre epica** | **Historia de usuario involucrado** |
| --- | --- | --- |
| **EP1** | **Gestión de usuarios y autenticación** | **US1,US2,US3,US4** |
| **EP2** | **Visualización de menú y productos** | **US5,US6,US7,US8** |
| **EP3** | **Sistema de pedidos con reserva anticipada** | **US9,US10,US11** |
| **EP4** | **Gestión de stock en tiempo real** | **US12,US13,US14** |
| **EP5** | **Integración de pagos (incluyendo JUNAEB)** | **US15,US16,US17** |
| **EP6** | **Notificaciones y confirmaciones** | **US18,US19,US20** |
| **EP7** | **Administración para locales de comida** | **US21,US22,US23** |

Una vez definido nuestras épicas,se realizó la división o desfragmentación de cada una de estas para poder reducir en tareas más específicas que son las siguientes historias de usuario:

# Historias de Usuario derivadas de las épicas

**Épica 1: Gestión de usuarios y autenticación**

US1: Como estudiante quiero registrarme con mi RUT y correo institucional para poder acceder a la aplicación.

US2: Como profesor quiero iniciar sesión con mi correo institucional para poder comprar en la app.

US3: Como local de comida quiero tener un acceso diferenciado para poder administrar mis productos y pedidos.

US4: Como usuario quiero que mi contraseña esté protegida mediante hash para asegurar la confidencialidad de mis datos.

**Épica 2: Visualización de menú y productos**

US5: Como usuario quiero ver un listado de productos disponibles para decidir qué comprar.

US6: Como usuario quiero filtrar productos por categoría (ej: bebidas, snacks, promociones) para encontrar lo que necesito más rápido.

US7: Como usuario quiero buscar un producto específico por nombre para ahorrar tiempo.

US8: Como usuario quiero ver detalles de cada producto (precio, stock, descripción) para tomar una mejor decisión de compra.

**Épica 3: Sistema de pedidos con reserva anticipada**

US9: Como usuario quiero agregar productos a un carrito de compras para realizar mi pedido de forma más cómoda.

US10: Como usuario quiero definir la hora de retiro (mínimo 20 minutos antes) para poder organizarme entre clases.

US11: Como usuario quiero guardar pedidos frecuentes para repetirlos sin tener que seleccionarlos uno a uno.

**Épica 4: Gestión de stock en tiempo real**

US12: Como usuario quiero visualizar en la app si un producto está agotado para no perder tiempo en seleccionarlo.

US13: Como local quiero que se actualice el stock automáticamente cuando se concrete una venta para evitar errores en los pedidos.

US14: Como local quiero recibir notificaciones de bajo stock para reponer productos a tiempo.

**Épica 5: Integración de pagos (incluyendo JUNAEB)**

US15: Como estudiante quiero pagar con mi tarjeta JUNAEB usando RUT y clave dinámica para comprar sin efectivo.

US16: Como usuario quiero guardar mi método de pago preferido para no ingresarlo cada vez que haga un pedido.

US17: Como usuario quiero tener la opción de pagar con tarjeta/débito como método adicional en fases futuras.

**Épica 6: Notificaciones y confirmaciones**

US18: Como usuario quiero recibir un correo de confirmación de compra para tener respaldo de mi pedido.

US19: Como usuario quiero recibir un código QR en mi correo para retirar el pedido de forma rápida.

US20: Como usuario quiero recibir notificaciones en la app sobre el estado de mi pedido (pendiente, listo, retirado).

**Épica 7: Administración para locales de comida**

US21: Como local quiero ver en un panel todos los pedidos recibidos para gestionarlos en orden.

US22: Como local quiero modificar el stock disponible desde mi panel para reflejar cambios en el inventario.

US23: Como local quiero acceder a reportes básicos de ventas para conocer el rendimiento diario.

# 

# Priorización de historias de usuario

Una vez definido cada tarea a realizar, se va a proceder a priorizar las historias de usuario, con la finalidad de centrarse en las funcionalidades principales que requiere nuestro stakeholders, permitiendo entregar el mayor valor posible como a su vez permite tener un control de aquellas historias que pueden ser más complejas y puedan generar un riesgo a futuro.

Por lo que, se va utilizar el método MoSCoW para definir qué tareas deben realizarse, cuales funcionalidades debería tener, pueden tener o que posiblemente no se realice por el poco valor que entrega.

| **Historia de Usuario** | **Tarea Asociada** | **Priorización** |
| --- | --- | --- |
| US1,US2 | Registro con RUT/correo institucional | **Must Have** |
| US4 | Codificación/Hash de contraseñas | **Must Have** |
| US3 | Validación de acceso según rol | **Must Have** |
| US5 | Consulta y visualización de productos | **Must Have** |
| US9,US10 | Selección de productos y hora de retiro | **Must Have** |
| US12,US13 | Panel de gestión de pedidos | **Must Have** |
| US15 | Formulario de pago JUNAEB | **Must Have** |
| US17. | Integración con WebPay u otro medio | **Must Have** |
| US18,US19 | Generar QR + confirmación por correo | **Must Have** |
| US21,US22 | Panel de administración local | **Must Have** |
| US6 | Categorías y filtros | **Should Have** |
| US7 | Buscador de productos | **Should Have** |
| US14 | Notificación de bajo stock | **Should Have** |
| US16 | Métodos de pago guardados | **Should Have** |
| US20 | Notificaciones en la app | **Should Have** |
| US8 | Carrusel de destacados | **Could Have** |
| US11 | Implementación de favoritos | **Could Have** |
| US23 | Reportes de ventas | **Could Have** |

# Criterios de aceptación

### **Épica 1: Gestión de usuarios y autenticación**

**US1: Registro con RUT y correo institucional**

* El sistema debe revisar que el RUT es válido.
* Solo correos institucionales (@duocuc.cl) son aceptados.
* Si los datos son correctos, el usuario queda registrado.
* Si algún dato es inválido, debe mostrarse un mensaje de error claro.

**US2: Inicio de sesión profesor**

* El login solo permite correos institucionales válidos.
* La contraseña debe coincidir con la registrada.
* El profesor debe poder acceder a la app con su rol específico.

**US3: Acceso diferenciado local**

* Los locales deben acceder con credenciales especiales.
* Una vez autenticado, se muestra un panel distinto al de estudiantes/profesores.
* Si intenta ingresar con el rol incorrecto, debe mostrar error.

**US4: Contraseña protegida mediante hash**

* Todas las contraseñas se almacenan encriptadas (no en texto plano).
* El sistema nunca debe mostrar la contraseña del usuario si no es requerido.
* La autenticación debe funcionar correctamente con contraseñas hasheadas.

### **Épica 2: Visualización de menú y productos**

**US5: Listado de productos disponibles**

* La lista debe mostrar nombre, precio, descripción e imagen de cada producto.
* Deben aparecer sólo productos activos/en stock.

**US6: Filtros por categoría**

* Deben existir filtros predefinidos (ej: bebidas, snacks, promociones).
* Al seleccionar un filtro, la lista muestra sólo los productos de esa categoría.

**US7: Buscador de productos por nombre**

* El usuario puede escribir parte del nombre y obtener resultados.
* Si no hay coincidencias, debe mostrarse un mensaje claro (“sin resultados”).

**US8: Detalles de producto**

* Cada producto debe mostrar precio, stock disponible, imagen y descripción.
* Si el producto está agotado, debe indicarlo a la vista.

### **Épica 3: Sistema de pedidos con reserva anticipada**

**US9: Carrito de compras**

* El usuario puede agregar múltiples productos al carrito.
* El carrito debe mostrar el total actualizado.
* El usuario puede eliminar productos antes de confirmar.

**US10: Definir hora de retiro**

* El sistema solo permite seleccionar horas al menos 20 minutos posteriores al pedido.
* No debe aceptar horas fuera del horario del local.

**US11: Guardar pedidos frecuentes**

* El usuario puede guardar un pedido completo como “Favorito”.
* En compras futuras, puede repetirlo con un clic.

### 

### **Épica 4: Gestión de stock en tiempo real**

**US12: Visualizar si un producto está agotado**

* Si un producto no tiene stock, debe marcarse como “Agotado”.
* Productos agotados no pueden añadirse al carrito.

**US13: Actualización automática de stock tras compra**

* Al confirmar un pedido, el stock disminuye automáticamente.
* Si el stock llega a 0, el producto aparece como agotado.

**US14: Notificación de bajo stock**

* El sistema envía alerta cuando un producto baja del umbral definido (ej: 5 unidades).
* El local recibe notificación en su panel.

### **Épica 5: Integración de pagos**

**US 15: Pago con tarjeta JUNAEB**

* El sistema debe pedir RUT y clave dinámica.
* El pago se valida contra la pasarela correspondiente.
* Si el pago es exitoso, se confirma el pedido.

**US16: Guardar método de pago preferido**

* El usuario puede registrar tarjeta u opción de pago.
* En pedidos futuros, la app debe sugerir el último método guardado.

**US17: Pago con tarjeta/débito**

* Debe existir la opción en la interfaz, aunque inicialmente esté deshabilitada.
* El sistema debe permitir su activación en futuras fases.

### **Épica 6: Notificaciones y confirmaciones**

**US18: Correo de confirmación de compra**

* Tras confirmar el pago, el usuario recibe un correo con el detalle del pedido.
* El correo debe enviarse en menos de 1 minuto.

**US19: Código QR para retiro**

* El correo de confirmación incluye un QR único.
* El QR debe ser escaneado por el local para validar el retiro.

**US 20: Notificaciones en la app sobre estado del pedido**

* El usuario debe recibir notificaciones push en cada cambio de estado (pendiente, listo, retirado).
* El estado debe actualizarse en tiempo real.

### 

### **Épica 7: Administración para locales**

**US21: Panel de pedidos recibidos**

* El local debe ver los pedidos ordenados cronológicamente.
* Cada pedido debe mostrar productos, cantidad y hora de retiro.

**US22: Modificar stock desde panel**

* El local puede modificar manualmente el stock disponible.
* Los cambios se reflejan de inmediato en la app del usuario.

**US23: Reportes básicos de ventas**

* El sistema debe mostrar ventas diarias con monto total.
* El local puede exportar o ver estadísticas en su panel.

# 

# Planning Poker

Durante las sesiones de Planning Poker de este proyecto, el equipo estimó cada historia de usuario según el esfuerzo requerido para implementarla, considerando:

* Nivel de complejidad técnica.
* Dependencias con otras historias.
* Tiempo estimado de desarrollo.
* Necesidad de validaciones o pruebas.

Se asignaron valores en puntos de historia según la secuencia de Fibonacci (1, 2, 3, 5, 8, 13), lo que permite reflejar el esfuerzo relativo de cada funcionalidad.

En este caso, la mayoría de las historias fueron estimadas con **3, 5 u 8 puntos**, lo que representa un equilibrio entre tareas de complejidad baja, media y alta.

| **US** | **Historia de Usuario** | **Puntos (Fibonacci)** | **Justificación** |
| --- | --- | --- | --- |
| **US1** | Registro con RUT | 3 | Baja complejidad técnica, requiere validación de datos básicos. |
| **US2** | Inicio de sesión (correo institucional) | 5 | Necesita integración con sistema de autenticación institucional. |
| **US3** | Validación de acceso según rol | 5 | Implica gestión de permisos y middleware de seguridad. |
| **US4** | Hash de contraseñas | 3 | Implementación estándar de librerías de seguridad. |
| **US5** | Visualización de productos | 3 | Listado simple, dependiente del stock cargado. |
| **US6** | Categorías y filtros | 5 | Mayor complejidad por lógica de filtrado y búsqueda dinámica. |
| **US7** | Buscador de productos | 5 | Requiere indexación y lógica de coincidencias. |
| **US9** | Selección de productos (carrito) | 5 | Manejo de estados y persistencia temporal de datos. |
| **US10** | Selección de hora de retiro | 3 | Calendario sencillo con validación de disponibilidad. |
| **US12** | Panel de gestión de pedidos | 8 | Vista completa para locales, manejo de múltiples estados. |
| **US13** | Actualización automática de stock | 5 | Implica validaciones y sincronización con base de datos. |
| **US14** | Notificación de bajo stock | 3 | Lógica de alerta sencilla en backend. |
| **US15** | Pago con JUNAEB (RUT + clave dinámica) | 8 | Alta complejidad por integración externa y seguridad. |
| **US16** | Métodos de pago guardados | 5 | Manejo seguro de tokens de pago y encriptación. |
| **US17** | Integración con WebPay u otro medio | 8 | Alta complejidad técnica por integración con pasarela de pago. |
| **US18** | Correo de confirmación | 3 | Funcionalidad estándar con librerías SMTP/servicios externos. |
| **US19** | Generación de QR | 3 | Lógica sencilla con librería de QR. |
| **US20** | Notificaciones en la app (estado pedido) | 5 | Requiere push notifications e integración en frontend/backend. |
| **US21** | Panel de administración local | 8 | Alta complejidad: CRUD completo, roles y métricas. |
| **US22** | Modificar stock desde panel | 5 | CRUD medio con validación de integridad de datos. |
| **US23** | Reportes básicos de ventas | 5 | Complejidad media al tener que realizar el análisis de ventas |

# Sprints del proyecto

En cuanto al progreso de cada historia de usuario, en esta tabla se muestran todas las historias de usuarios desarrolladas, desde su fecha de inicio y su fecha de finalización.

## Sprint 1 · 05/09/2025 – 17/09/2025 (Autenticación y seguridad)

| **UH** | **Actividad** | **Duración** | **Inicio** | **Fin** |
| --- | --- | --- | --- | --- |
| UH1 | Registro con RUT | 2 días | 05/09 | 06/09 |
| UH2 | Inicio de sesión (correo institucional) | 2 días | 07/09 | 08/09 |
| UH4 | Hash de contraseñas | 3 días | 09/09 | 11/09 |
| UH3 | Validación de acceso según rol | 3 días | 12/09 | 14/09 |
| — | QA / pruebas internas | 3 días | 15/09 | 17/09 |

## 

## Sprint 2 · 20/09/2025 – 03/10/2025 (Catálogo y compra básica)

| **UH** | **Actividad** | **Duración** | **Inicio** | **Fin** |
| --- | --- | --- | --- | --- |
| UH5 | Visualización de productos | 3 días | 20/09 | 22/09 |
| UH9 | Selección de productos (carrito) | 3 días | 23/09 | 25/09 |
| UH10 | Selección de hora de retiro | 2 días | 26/09 | 27/09 |
| — | QA / ajustes | 6 días | 28/09 | 03/10 |

## Sprint 3 · 03/10/2025 – 16/10/2025 (Pagos + gestión de pedidos)

| **UH** | **Actividad** | **Duración** | **Inicio** | **Fin** |
| --- | --- | --- | --- | --- |
| UH12 | Panel de gestión de pedidos (vista local) | 3 días | 03/10 | 05/10 |
| UH13 | Actualización automática de stock | 2 días | 06/10 | 07/10 |
| UH15 | Pago con JUNAEB (RUT + clave dinámica) | 3 días | 08/10 | 10/10 |
| UH17 | Integración con WebPay u otro medio | 3 días | 11/10 | 13/10 |
| — | QA / pruebas integradas | 2 días | 14/10 | 15/10 |

## 

## 

## Sprint 4 · 17/10/2025 – 30/10/2025 (Cierre de ciclo: confirmación + admin)

| **UH** | **Actividad** | **Duración** | **Inicio** | **Fin** |
| --- | --- | --- | --- | --- |
| UH18 | Correo de confirmación | 2 días | 17/10 | 18/10 |
| UH19 | Generación de QR | 1 día | 19/10 | 19/10 |
| UH21 | Panel de administración local | 3 días | 20/10 | 22/10 |
| UH22 | Modificar stock desde panel | 2 días | 23/10 | 24/10 |
| — | QA / UAT con locales | 5 días | 25/10 | 29/10 |

## Sprint 5 · 31/10/2025 – 13/11/2025 (Mejoras: Should/Could)

| **UH** | **Actividad** | **Duración** | **Inicio** | **Fin** |
| --- | --- | --- | --- | --- |
| UH6 | Categorías y filtros | 3 días | 31/10 | 02/11 |
| UH7 | Buscador de productos | 3 días | 03/11 | 05/11 |
| UH14 | Notificación de bajo stock | 2 días | 06/11 | 07/11 |
| UH16 | Métodos de pago guardados | 2 días | 08/11 | 09/11 |
| UH20 | Notificaciones en la app (estado pedido) | 3 días | 10/11 | 12/11 |
| — | QA / cierre de sprint | 1 día | 13/11 | 13/11 |