

Interacción Hombre-Computadora

Trabajo Práctico

Gestor de Listas de Compras

Tercera Entrega: Implementación de Interfaz Móvil



Compartir

Grupo 1

Keoni Lucas Dubovitsky - 62815

Franco Ferrari - 63094

Nicolas Domingo Mazzitelli - 62334

Mateo Pirola Paulovich - 62810



Índice

1. Introducción.....	3
2. Requisitos funcionales y no funcionales implementados.....	3
2.2. Requisitos no funcionales.....	6
3. Capturas de pantalla de las vistas de la aplicación móvil.....	7
3.1. Capturas de pantallas de las vistas y versión final de los prototipos (cuando estos difieran) para poder reflejar la evolución que sufrieron estos últimos a lo largo de la etapa de implementación.....	7
3.2. Capturas de pantalla de las vistas que pudieran verse afectadas por el factor de forma (teléfonos y tabletas) del dispositivo.....	29
3.3. Capturas de pantalla de las vistas que pudieran verse afectadas por la orientación (vertical y horizontal) del dispositivo.....	33
4. Decisiones de usabilidad tomadas durante la etapa de implementación.....	35
4.1. Justificación de las diferencias que pudieran presentar las vistas respecto de la versión final de los prototipos.....	35
4.2. Justificación de los aspectos relacionados con el diseño gráfico (colores, tipografías, imágenes, etc.).....	37
4.3. Justificación haciendo referencia a los lineamientos de diseño de la plataforma Android.....	37
4.4. Justificación haciendo referencia a los temas vistos en la materia demostrando la aplicación práctica de los mismos.....	38
4.5. Justificación haciendo referencia a los modelos de Personas.....	38
4.6. Justificación haciendo referencia a las sugerencias y/o correcciones realizadas por los docentes. repetitivo con decisiones de usabilidad?.....	39
5. Instructivo de instalación.....	40
1. Backend (API):.....	40
2. App móvil (Android):.....	40
6. Conclusión.....	40

1. Introducción

Este informe corresponde a la tercera entrega del trabajo práctico. Presentamos la implementación de la interfaz móvil del gestor de Listas de Compras "Compartir", desarrollada a partir de los prototipos de la primera entrega y de los modelos de personas definidos previamente. El objetivo fue convertir dichos prototipos en una interfaz útil y usable, que brinde una experiencia de usuario satisfactoria, aplicando los principios de HCI y usabilidad vistos en clase y priorizando patrones de interacción consistentes y prevención de errores.

Para la construcción utilizamos Kotlin y Jetpack Compose como framework de interfaz, junto con Material Design 3 y Navigation Compose para la gestión del flujo entre pantallas. La aplicación sigue una arquitectura por capas basada en ViewModels, repositorios y fuentes de datos, lo que permite separar claramente la lógica de presentación y el acceso a datos. La implementación está orientada a Android 10 o superior y fue probada en distintos dispositivos y emuladores, contemplando variaciones de tamaño de pantalla y orientación.

A lo largo del informe se documentan los requisitos funcionales y no funcionales alcanzados, capturas de las principales pantallas y las decisiones de diseño y usabilidad adoptadas, justificadas en relación con la teoría de la materia y con nuestros modelos de personas.

2. Requisitos funcionales y no funcionales implementados

2.1. Requisitos funcionales

2.1.1. RF1 - Registrar cuenta.

La funcionalidad de **registro de cuenta** se considera implementada porque el flujo completo permite crear un usuario nuevo con validaciones inmediatas y mensajes en contexto. En la pantalla de Register se indican los campos obligatorios y, ante un envío incompleto, cada campo se marca en rojo y aparece el texto "Campo requerido". Este comportamiento valida entradas, orienta al usuario y evita reintentos innecesarios.

2.1.2. RF2 - Verificar cuenta.

La **verificación de cuenta** está implementada porque, tras el registro, el usuario ingresa su código y el sistema informa si el código es correcto o inválido. Cuando el código es válido la cuenta queda verificada y habilitada para el ingreso. Cuando es inválido se mantiene el bloqueo y se comunica el motivo junto al campo de verificación. Se preserva la coherencia visual y textual con el registro, lo que garantiza comprensión y continuidad del proceso.

2.1.3. RF3 - Modificar Contraseña.

El **restablecimiento y la modificación de contraseña** están implementados porque el formulario informa requisitos mínimos y refleja en pantalla si la nueva clave los cumple. Una contraseña que no satisface las reglas muestra un error inmediato. Una contraseña válida confirma el

cambio y habilita el inicio de sesión con la nueva credencial. El usuario obtiene visibilidad del estado en cada paso.

2.1.4. **RF4** - Iniciar y Cerrar Sesión.

El **inicio y cierre de sesión** están implementados porque las credenciales válidas dirigen a la home y las inválidas generan un mensaje de error sin recargar por completo. El cierre de sesión limpia el estado y retorna a la pantalla de acceso. Este comportamiento confirma que la autenticación y la terminación de sesión funciona por completo.

2.1.5. **RF5** - Gestionar Perfil.

La **gestión de perfil** está implementada porque los datos personales se editan en un único flujo centralizado, con etiquetas claras, validación y confirmación de guardado. Las acciones de seguridad se ubican en un bloque independiente para evitar confusión con los campos de perfil. El resultado de cada cambio se refleja en pantalla.

2.1.6. **RF6** - Gestionar Productos.

La **gestión de productos** está implementada porque el sistema permite crear, editar, eliminar y buscar productos, y ese estado se refleja en listados y resúmenes. Un producto creado aparece de inmediato en las vistas que lo consumen. Una edición actualiza la tarjeta correspondiente. Una eliminación lo retira de los listados y ajusta las métricas vinculadas.

2.1.7. **RF7** - Gestionar Listas de Compras.

La **gestión de Listas de Compras** está implementada porque desde la vista de Listas es posible crear nuevas listas con nombre obligatorio, editar atributos y eliminar registros. La creación agrega la nueva lista a las tarjetas visibles y el panel de resumen ajusta totales. La edición modifica nombre o descripción y la eliminación retira la tarjeta y actualiza los conteos. Todo sucede con actualización parcial y feedback inmediato.

2.1.8. **RF8** - Gestionar Productos en Listas de Compras.

La **gestión de productos dentro de las listas** está implementada porque en la vista de Lista específica se pueden agregar, marcar como comprados, reordenar y eliminar ítems. Cada acción se refleja al instante en el progreso de la lista y en los contadores asociados. El usuario permanece en la misma vista y no pierde el foco de interacción, lo que confirma la correcta integración entre la lista y sus elementos.

2.1.9. **RF9** - Compartir Listas de Compras.

La **función de compartir listas** está implementada porque el usuario puede invitar colaboradores por correo y recibir confirmación visible del resultado. Las listas compartidas se reflejan en la sección correspondiente de la cuenta del destinatario y la actividad asociada se centraliza en notificaciones. Los errores de invitación, como un correo con formato inválido, se informan en el mismo lugar de la acción, lo que valida el manejo de casos de éxito y de error.

2.1.10. **RF10** - Categorizar Productos.

La **categorización de productos** está implementada porque se pueden asignar categorías y luego filtrar o buscar por ellas. Al aplicar una categoría, los listados responden de acuerdo con el filtro y los resúmenes se mantienen consistentes. Al cambiar o quitar la categoría, la interfaz actualiza de inmediato la vista. Esto demuestra que la taxonomía es operativa y útil para la navegación.

2.1.11. **RF11** - Marcar Productos como Adquiridos.

La funcionalidad de **marcado de productos como adquiridos** está implementada porque, en la vista de Lista específica, cada ítem incluye un control táctil (checkbox/botón) que permite alternar entre “pendiente” y “comprado” sin abandonar la pantalla. Al marcar un producto como adquirido, la tarjeta cambia de estado visual (estilo atenuado y/o sección diferenciada), y se actualizan en tiempo real los contadores de la lista y el progreso general. Este comportamiento confirma que el sistema registra y refleja de manera clara qué productos ya fueron adquiridos y cuáles siguen pendientes, facilitando el control de la lista durante el recorrido de compra.

2.1.12. **RF12** - Recuperar Contraseña.

La **recuperación de contraseña** está implementada porque la vista solicita un correo válido, muestra validación en tiempo real y confirma el envío de instrucciones. Un correo mal formado o inexistente genera un mensaje claro sin ejecutar acciones. El usuario recibe feedback visible del resultado y recupera el control del flujo sin ambigüedades.

2.1.13. **RF13** - Consultar el Historial de Listas de Compras.

La **consulta del historial de listas** está implementada porque las listas finalizadas se listan con filtros por fecha o descripción y con acciones directas como Ver detalles y Restaurar. Las tarjetas usan color verde, para tener un diferenciador de las listas que ya están completas. Restaurar genera una nueva lista basada en una finalizada y la incorpora a “Tus listas recientes”. Este comportamiento confirma la reutilización efectiva de listas pasadas y reduce el tiempo de preparación de compras repetidas.

2.1.14. **RF14** - Marcar Listas de Compras como Recurrentes.

El marcado de **listas recurrentes** está implementado porque el modal de creación permite definir la recurrencia y esa condición queda visible en los listados. Las listas recurrentes se priorizan en los flujos de reutilización, lo que responde al patrón de compras semanales detectado y aporta previsibilidad al usuario.

2.1.15. **RF15** - Gestionar productos en la Despensa

La **gestión de productos** en despensa está implementada porque la vista de Despensas permite crear y administrar Despensas por ubicación y la vista de Despensa específica admite agregar productos con cantidad y unidad, editar valores y eliminarlos. Las tarjetas muestran el progreso y el

panel de resumen consolida el estado. Al compartir una despensa por correo se replica el patrón ya probado en listas. La actualización es inmediata y la interfaz confirma cada paso.

2.2. Requisitos no funcionales

Obligatorios

- 2.2.1. **RNF1** - Utilizar el idioma establecido en la configuración regional del dispositivo y estar localizada por lo menos a los idiomas español e inglés.

Se considera implementado porque la aplicación define todos los textos en archivos de recursos “values/strings.xml” y “values-es/strings.xml”, y selecciona automáticamente el idioma según la configuración regional del dispositivo. Además, desde la sección de Perfil/Configuración se puede forzar manualmente el idioma entre español e inglés, guardando la preferencia en DataStore para que se mantenga entre sesiones.

- 2.2.2. **RNF2** - Utilizar la barra de aplicación para mostrar un título contextual y/o accesos directos que permitan interactuar con la información contextual mostrada en la pantalla.

Se considera implementado porque todas las pantallas se estructuran alrededor de un Top App Bar que muestra el título contextual (por ejemplo, “Listas”, “Despensas”, “Perfil”) y acciones relevantes como buscar, filtrar, compartir o volver atrás. Las acciones secundarias se ubican en el menú de overflow de la barra, respetando el rol de la App Bar como contenedor de comandos contextuales.

- 2.2.3. **RNF3** - Permitir la personalización de aspectos relacionados con el funcionamiento de la aplicación.

Se considera implementado porque la sección de Configuración permite elegir el tema visual de la aplicación (claro u oscuro), guardando esta preferencia en DataStore. Al cambiar el tema, toda la UI se recomponen usando los mismos tokens de color de la marca (primario, fondo, superficie, error) en su versión light o dark, sin cerrar la sesión del usuario. La elección se mantiene al volver a abrir la app, ofreciendo una experiencia personalizada y consistente.

- 2.2.4. **RNF4** - Brindar una experiencia de usuario distinta dependiendo del factor de forma (teléfonos y tabletas) del dispositivo.

Se considera implementado porque la app utiliza layouts adaptativos: en teléfonos la navegación se organiza principalmente con Bottom Navigation, mientras que en tabletas se aprovecha el mayor ancho de pantalla para mostrar esquemas tipo Master/Detail (lista a la izquierda y detalle a la derecha) y/o Navigation Rail. De esta forma, la estructura se ajusta al factor de forma sin cambiar los flujos principales.

- 2.2.5. **RNF5** - Brindar una experiencia de usuario distinta dependiendo de la orientación (vertical y horizontal) del dispositivo.

Se considera implementado porque ciertas pantallas reorganizan su contenido al rotar el dispositivo. En orientación vertical se prioriza un flujo más lineal y apilado, mientras que en horizontal se redistribuyen paneles (por ejemplo, lista y detalle) para aprovechar mejor el ancho disponible. La UI se adapta automáticamente al cambio de orientación manteniendo la coherencia visual y funcional.

2.2.6. **RNF6** - Funcionar en dispositivos Android 10.0 - API Level 29 (o superior).

Se considera implementado porque el módulo app define un minSdkVersion compatible con Android 10 (API 29) y se probaron builds en emuladores y/o dispositivos físicos con esa versión o superior. Además, las dependencias utilizadas (Compose, librerías de soporte) están alineadas con ese nivel mínimo de API.

No obligatorios

2.2.7. **RNF7** - Escanear Códigos de Productos.

No fue implementado.

2.2.8. **RNF8** - Interactuar Usando Comandos de Voz.

No fue implementado.

2.2.9. **RNF9** - Tomar Fotografías de los Productos.

No fue implementado.

3. Capturas de pantalla de las vistas de la aplicación móvil

3.1. Capturas de pantallas de las vistas y versión final de los prototipos (cuando estos difieran) para poder reflejar la evolución que sufrieron estos últimos a lo largo de la etapa de implementación.

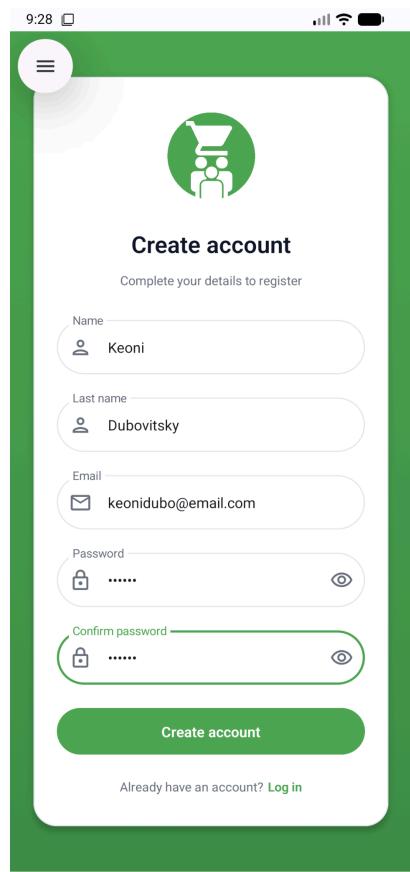


Figura 1.1 Register (versión final)

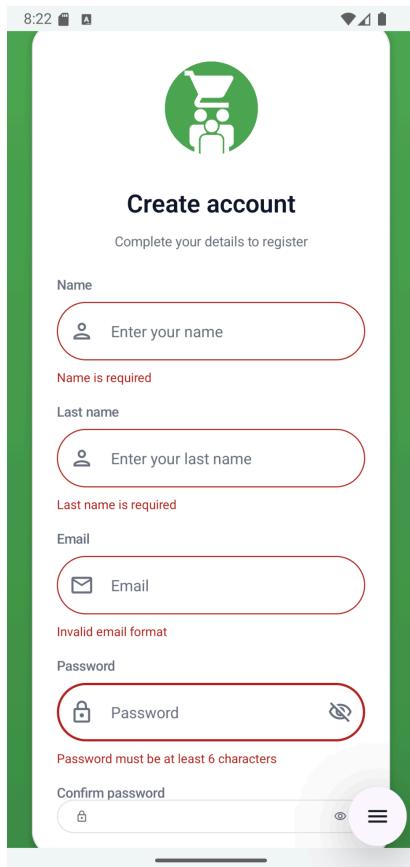


Figura 1.2 Register: mensaje en rojo cuando no se introducen los datos necesarios (versión final)

Aquí se muestra el estado de validación del formulario al crear una cuenta. Cuando el usuario intenta avanzar sin completar los campos obligatorios, cada input que no esté completo se resalta en **rojo** y aparece la leyenda “Campo requerido” debajo del campo correspondiente.

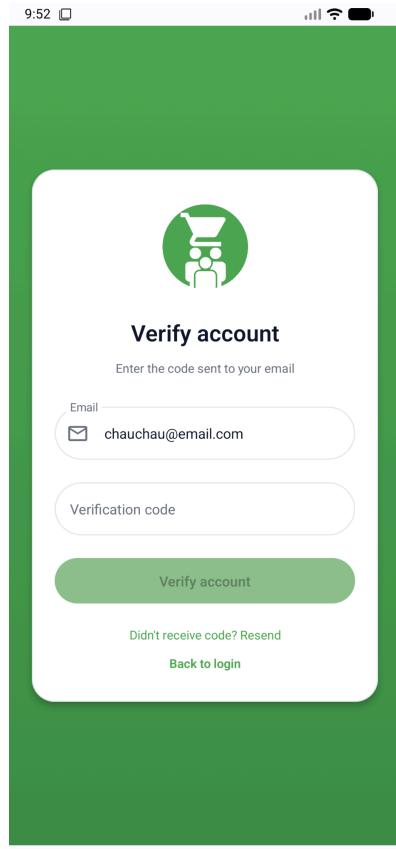


Figura 2.1 Verificar cuenta (versión final)

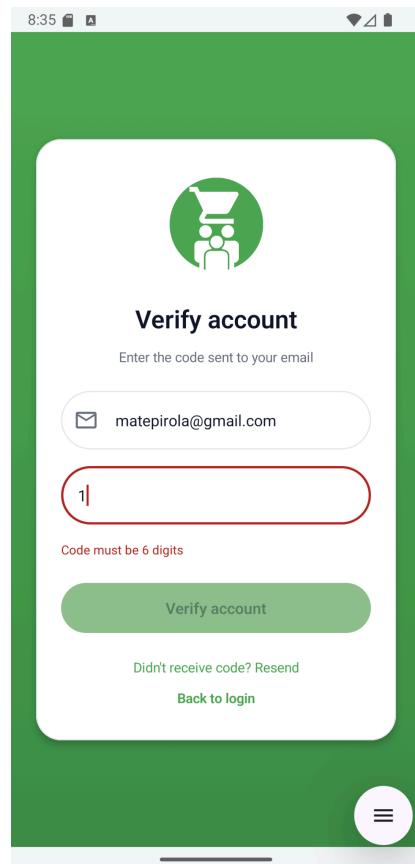


Figura 2.2 Verificar cuenta: mensaje en rojo cuando no se introducen los datos de forma correcta (versión final)

Acá como en cualquier otro sitio web que requiere autenticación, disponemos de vistas para que el usuario pueda registrarse, verificarla, e iniciar sesión en la plataforma. En caso de que el código de verificación sea correcto la cuenta queda verificada, de lo contrario muestra mensaje en rojo.

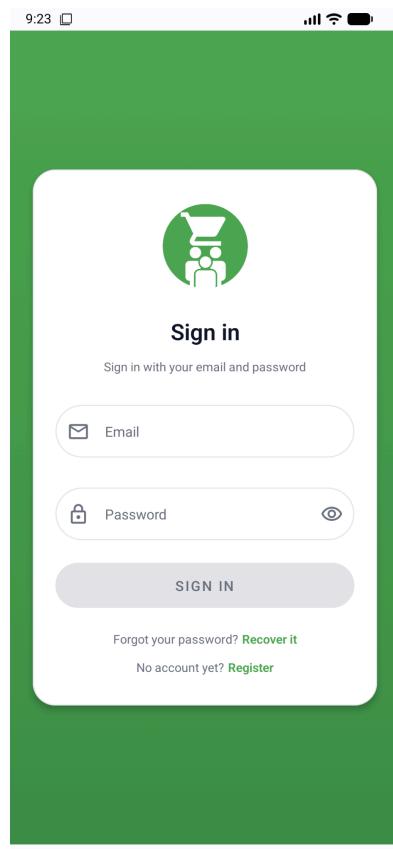


Figura 3.1 login (versión final)

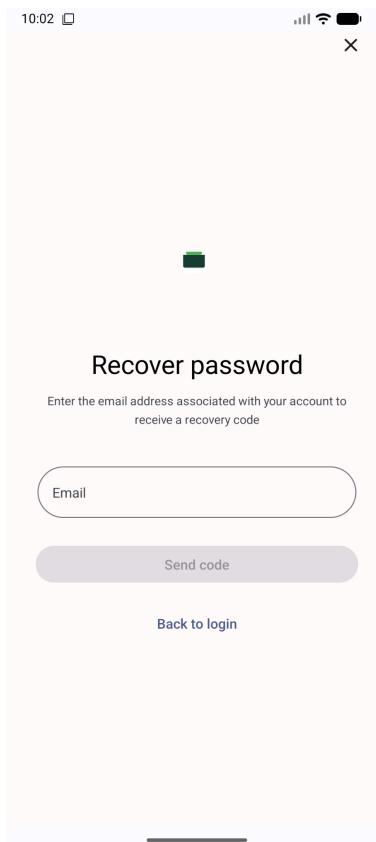


Figura 4.1 Recover password (versión final)

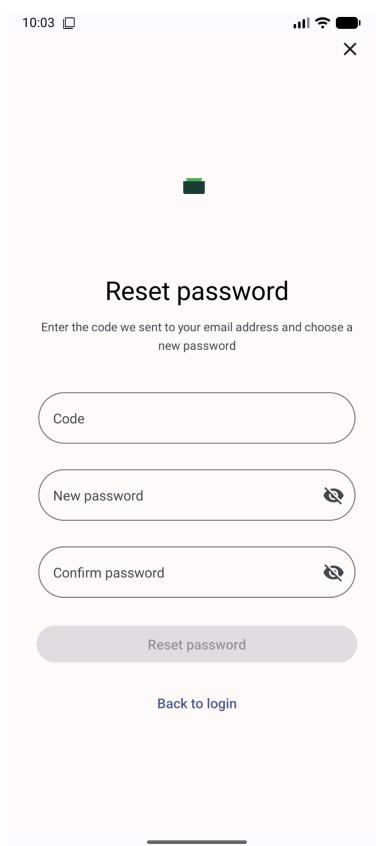


Figura 5.1 Reset password (versión final)

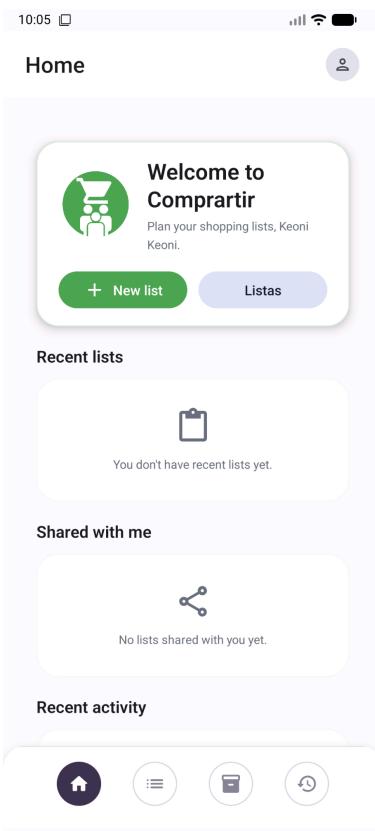


Figura 6.1 home (versión final)



Figura 6.2 home (versión final)

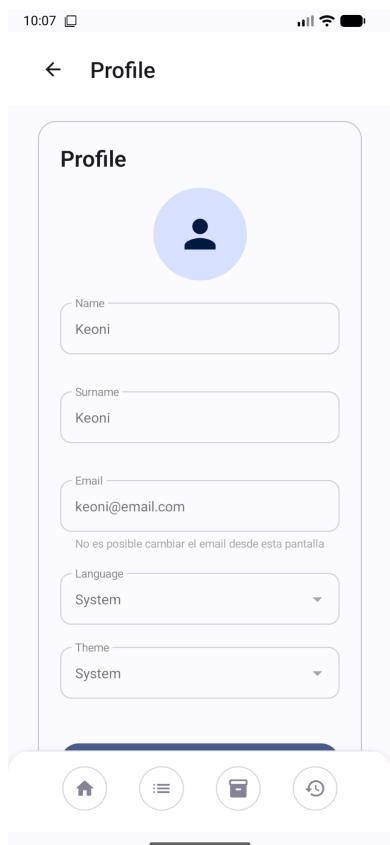


Figura 7.1 perfil (versión anterior)

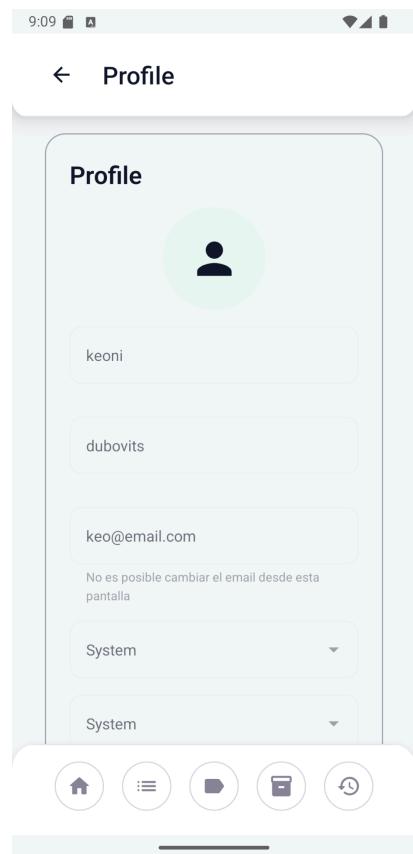


Figura 7.2 perfil (versión final)

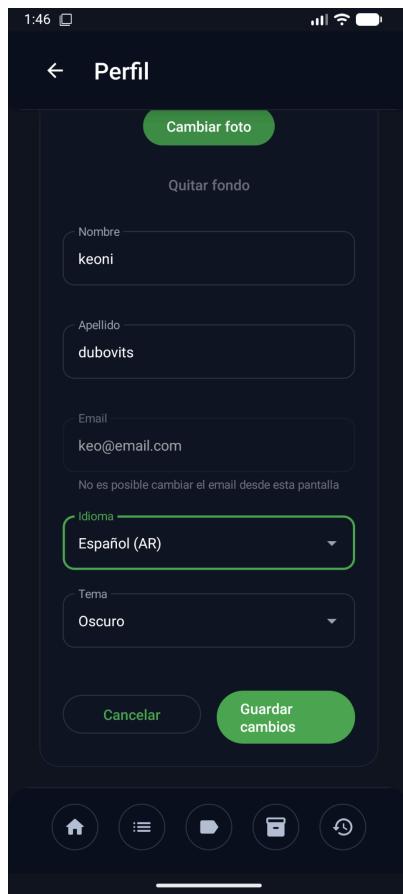


Figura 7.3 perfil (versión final oscuro)

En la parte de mi perfil se optó por tomar la decisión de hacer ciertos cambios de estética y colores. Priorizamos colores que queden mejor con el color general de la app (el verde). Esto fue para brindarle al usuario una interfaz más linda y que mantenga la misma paleta de colores que las demás partes de la app.

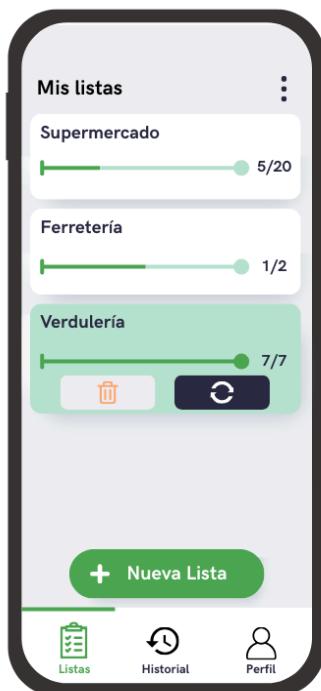


Figura 8.1 Listas (versión prototípico)

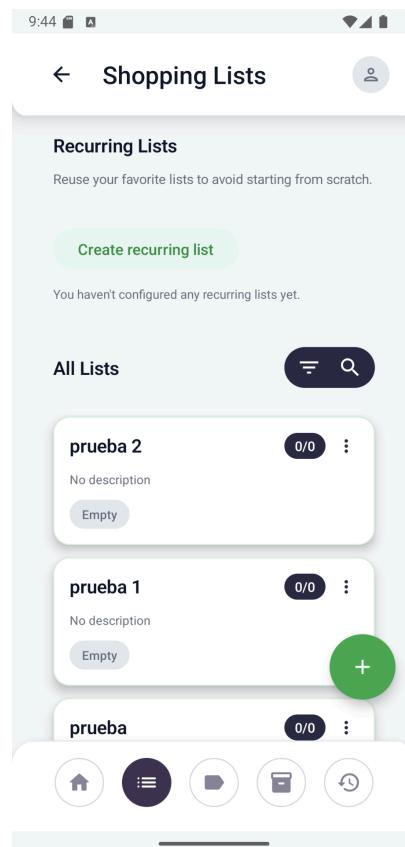


Figura 8.2 Listas (versión final)



Figura 8.2 Listas (versión oscura)

En la parte de listas sufrió algunos cambios tanto en la parte visual como en la parte funcional. En la parte visual se modificó el botón de agregar lista, y también los carteles de las listas ya creadas. A su vez se le hicieron pequeñas modificaciones estéticas en general. En la parte funcional lo que más se destaca es el agregado de listas concurrentes y un buscador y filtrado de listas. Todos estos cambios se llevaron a cabo con la intención de que el usuario pierda menos tiempo al momento de crear una lista o de buscar una de estas.

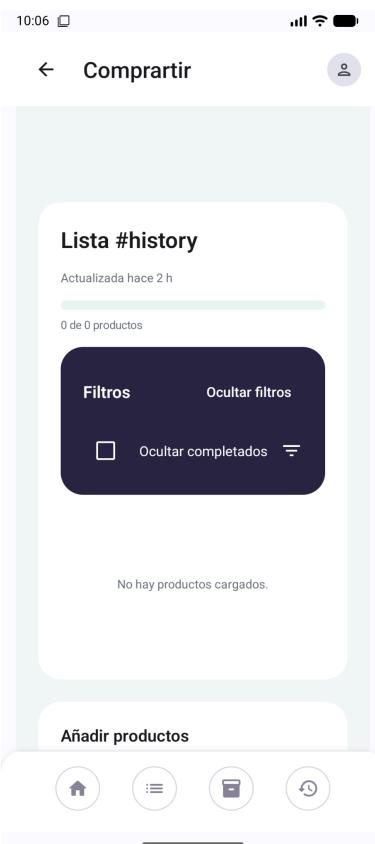


Figura 8.3 Lista específica (versión final)

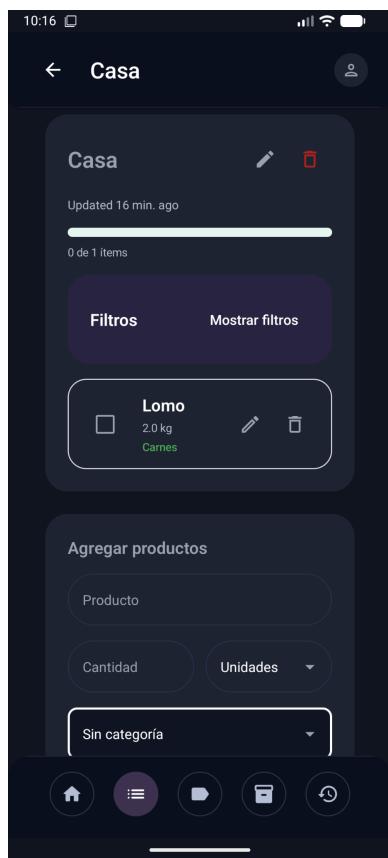


Figura 8.3 Lista específica (versión oscuro)

Pantalla Añadir Item

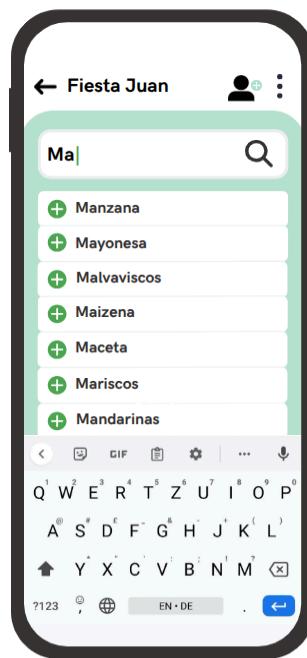


Figura 8.4 Añadir ítem(versión prototipo)

Pantalla Compartir Lista

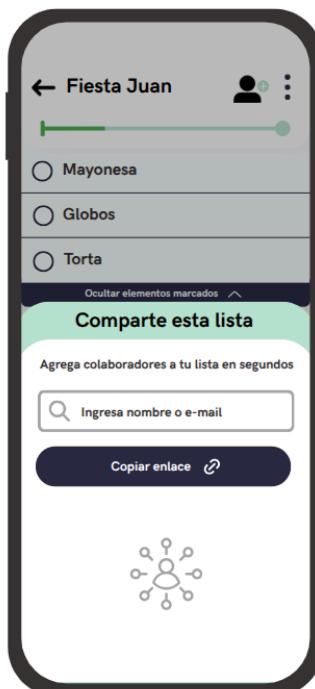


Figura 8.5 Compartir lista (versión prototipo)

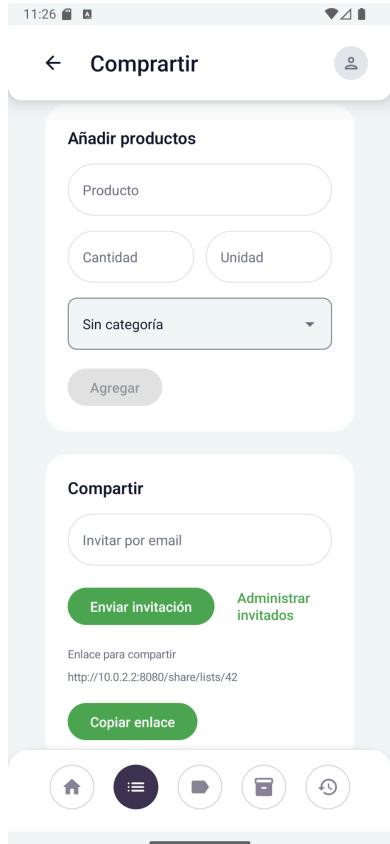


Figura 8.6 lista específica: añadir ítem y compartir(versión final)

En la parte de lista específica tomamos la decisión de hacer cambios priorizando la claridad y la rapidez. Se unificó la función compartir y añadir ítems dentro de una lista específica, además de que nosotros mismos como usuario creamos el ítem que queremos agregar a la lista. Esto se hizo con la intención de simplificar y facilitar el uso de funciones para el usuario y tengan un más rápido acceso a estas.

Pantalla Crear Lista

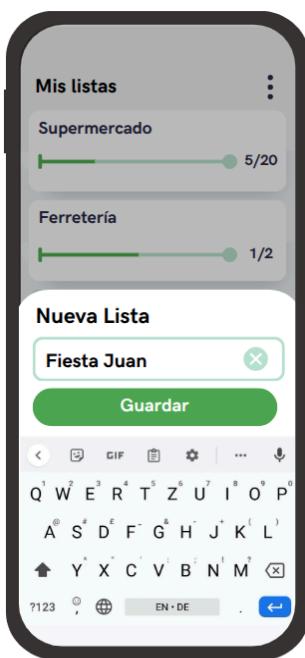


Figura 8.7 crear lista (versión prototipo)

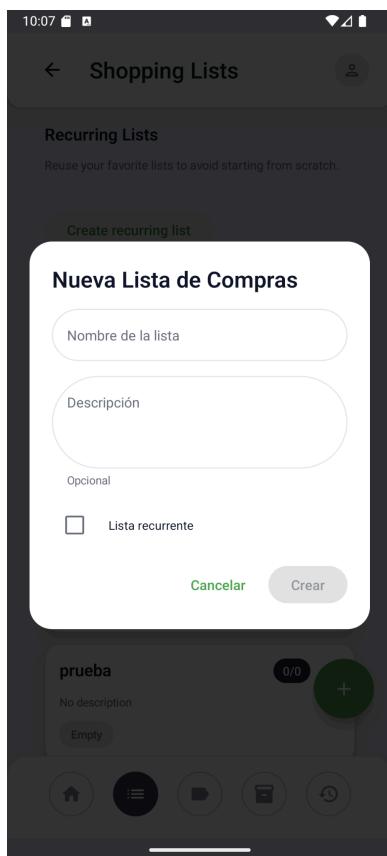


Figura 8.9 crear lista (versión final)

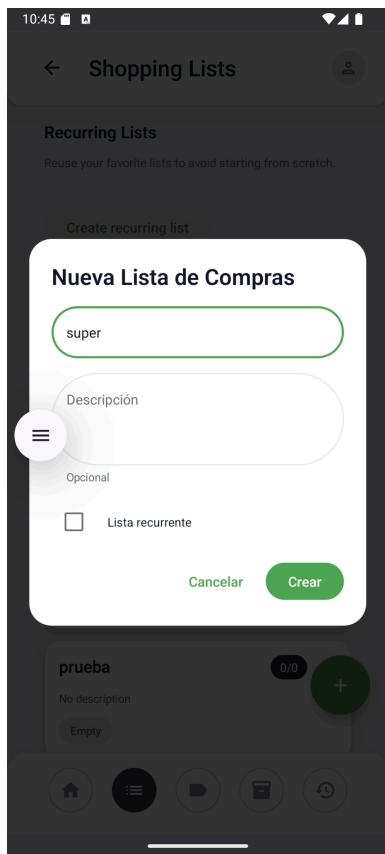


Figura 8.10 crear lista: Asignó nombre (versión final)

El modal de “crear lista” se abre desde listas con los campos requeridos de nombre (obligatorio) y descripción (opcional), y además la opción de lista recurrente. Notemos que si el usuario intenta confirmar sin nombre, no permite apretar el botón, y al completar el nombre el botón **Crear** queda habilitado para generar la lista (con posibilidad de marcarla como recurrente).

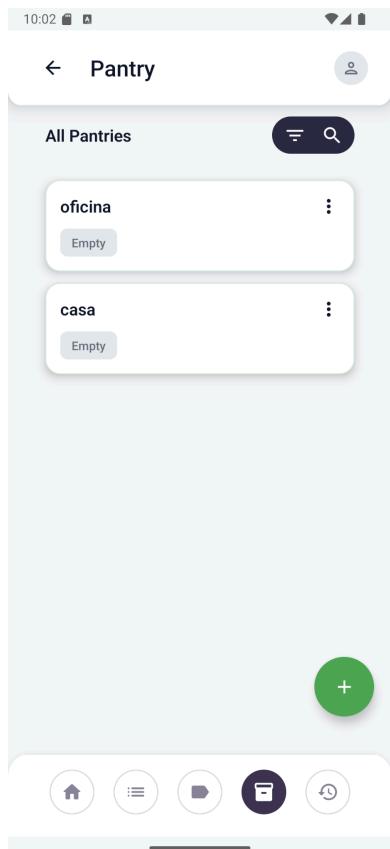


Figura 9.1 Despensas (versión final)

Despensas nos permite crear y gestionar Despensas con buscador, filtros y orden. En la parte derecha implementamos resumen, que le da una vista al usuario del total de las Despensas. A su vez cada tarjeta indica progreso de productos y acciones rápidas (Abrir/Editar/Eliminar), ideal para organizar stock por lugar.

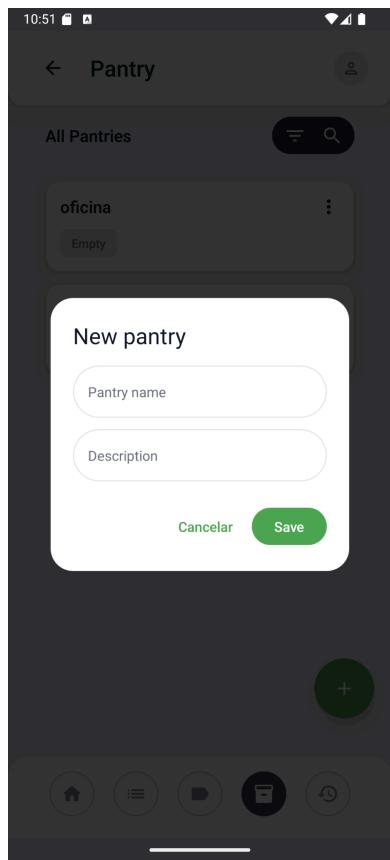


Figura 9.2 nueva despensa(versión final)

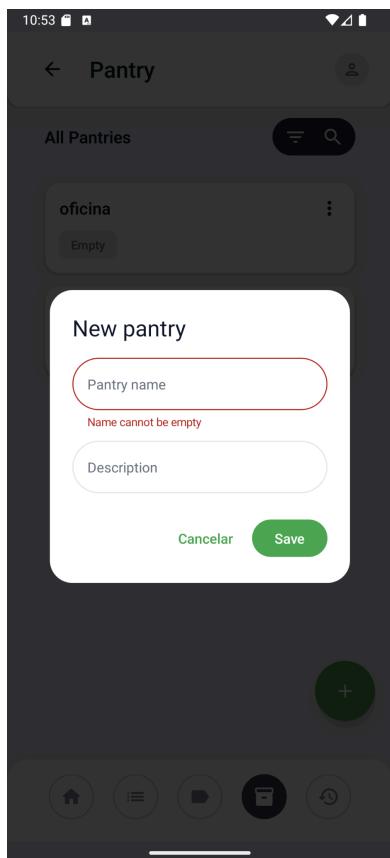


Figura 9.3 nueva despensa: mensaje de error campo obligatorio (versión final)

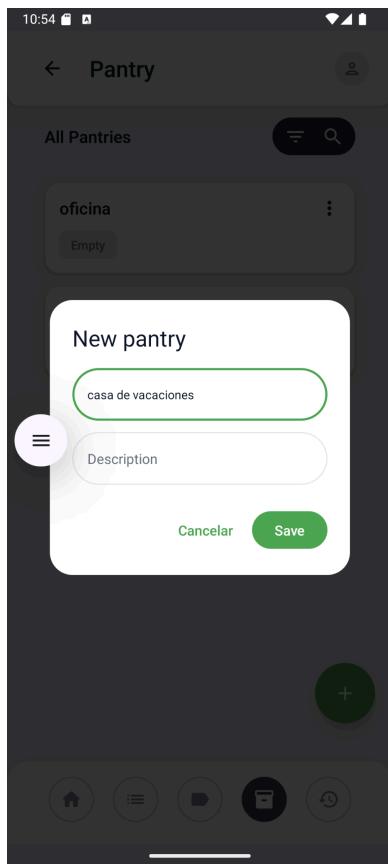


Figura 9.4 nueva despensa: Asignó nombre(versión final)



Figura 10.1 historial

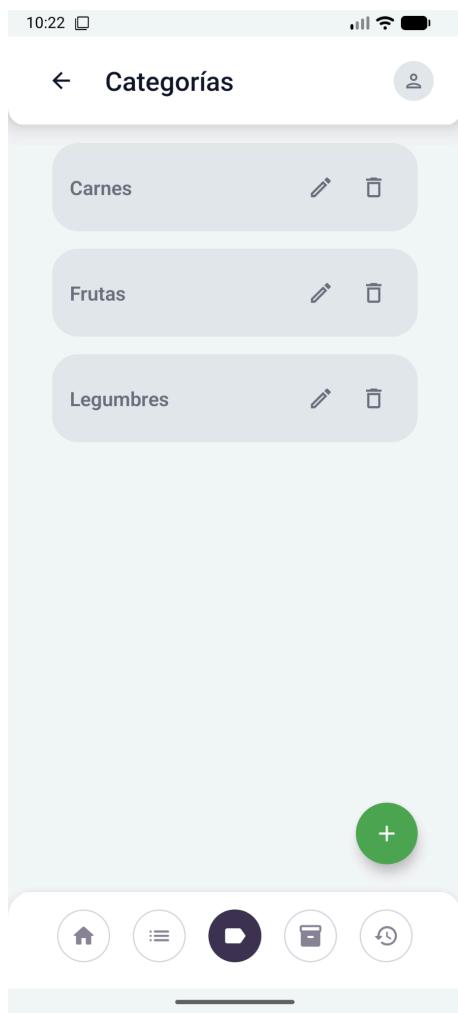


Figura 11.1 Categorías(version final)

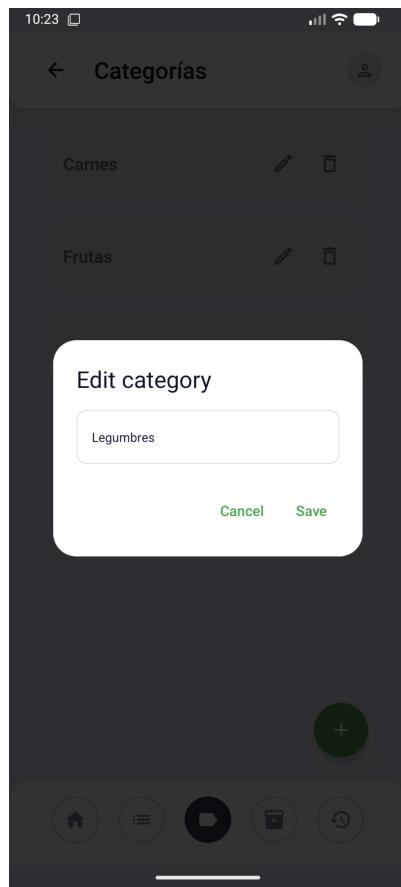


Figura 11.1 Categoría específica

El modal “nueva despensa” se abre desde Despensas con el campo Nombre, que si se intenta crear sin completarlo, el input se marca en rojo con el mensaje “name cannot be empty”. Al ingresar un nombre válido, el botón Crear queda habilitado y se confirma la creación de la despensa. (p.ej. “casa de vacaciones”).

3.2. Capturas de pantalla de las vistas que pudieran verse afectadas por el factor de forma (teléfonos y tabletas) del dispositivo.

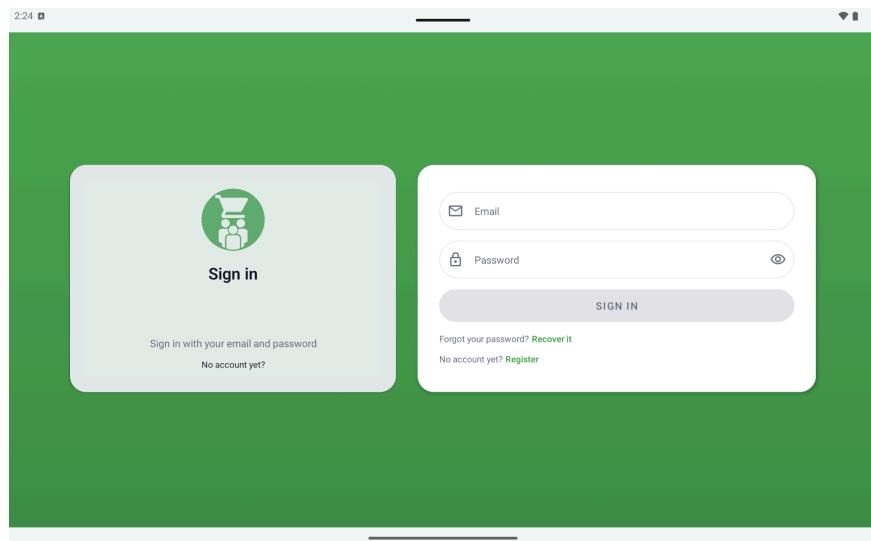


Figura 12.1 login (versión tablet)

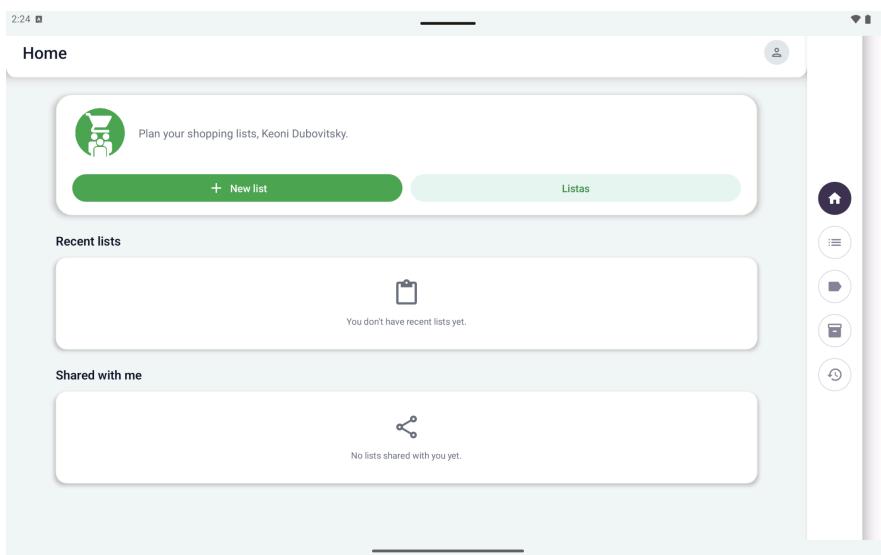


Figura 13.1 home (versión tablet)

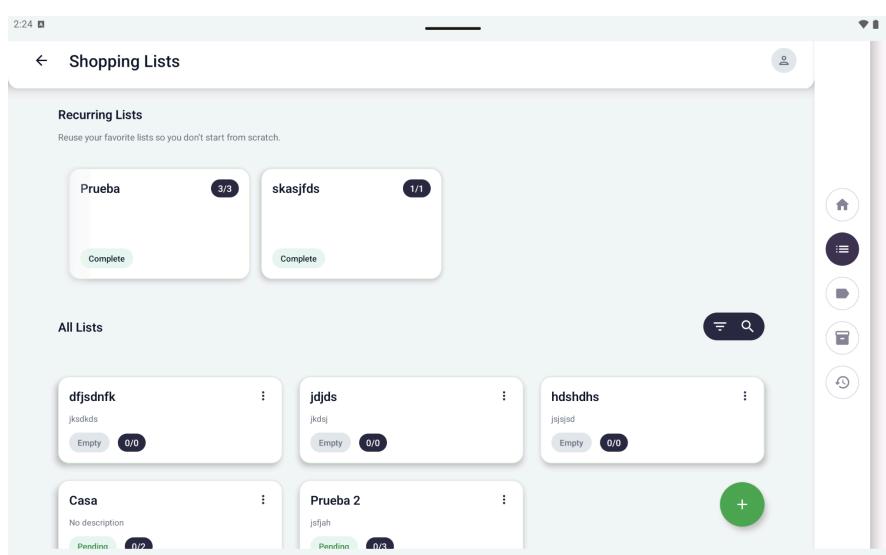


Figura 14.1 Listas (versión tablet)

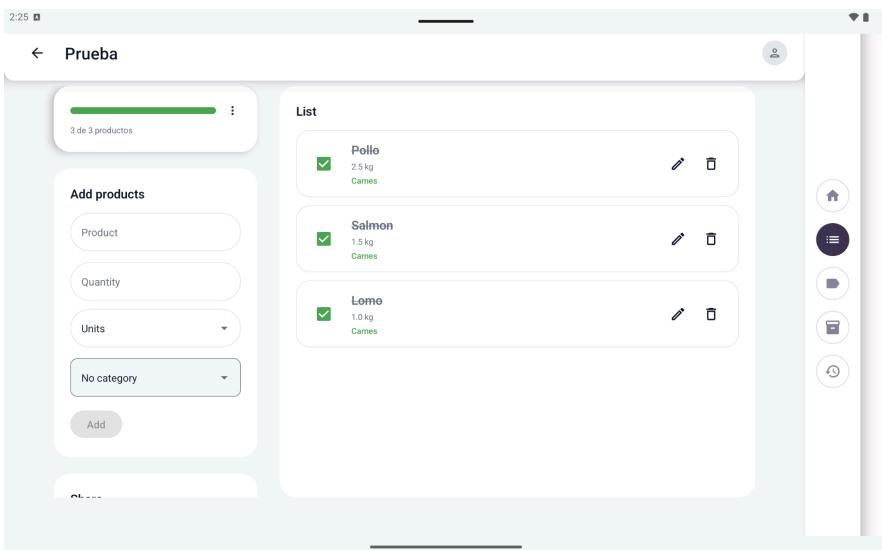


Figura 15.1 Lista específica (versión tablet)

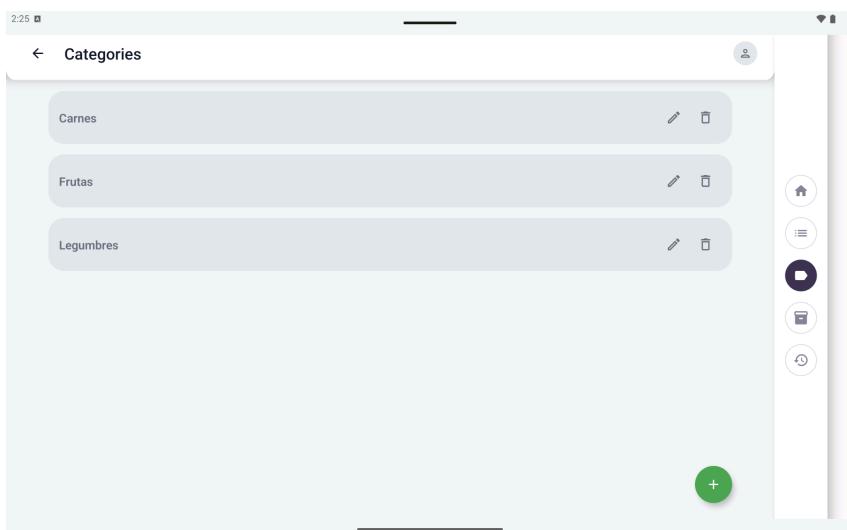


Figura 16.1 Categorías (versión tablet)

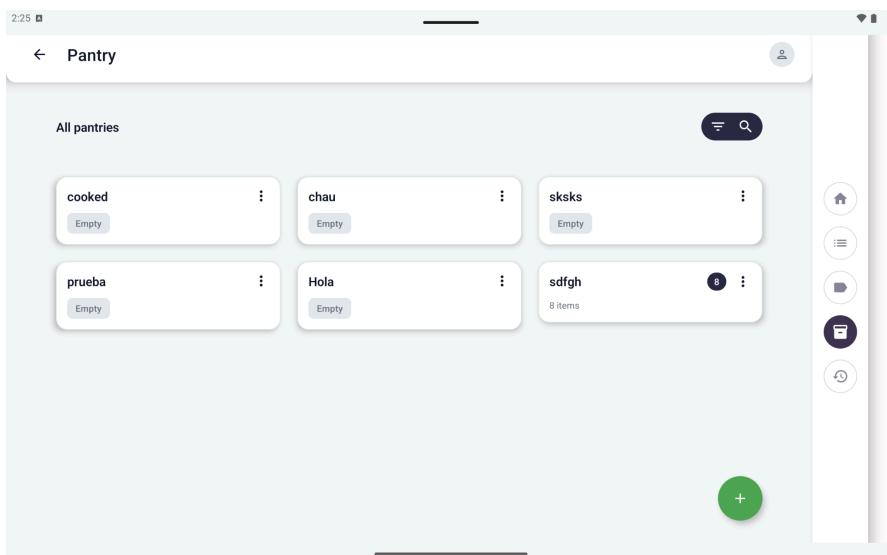


Figura 17.1 Pantry (versión tablet)

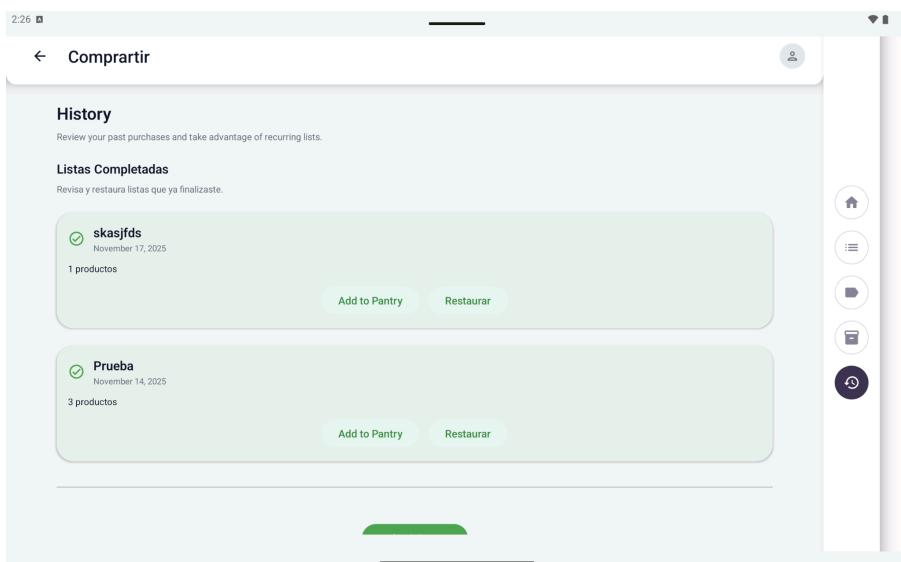


Figura 18.1 Historial (versión tablet)

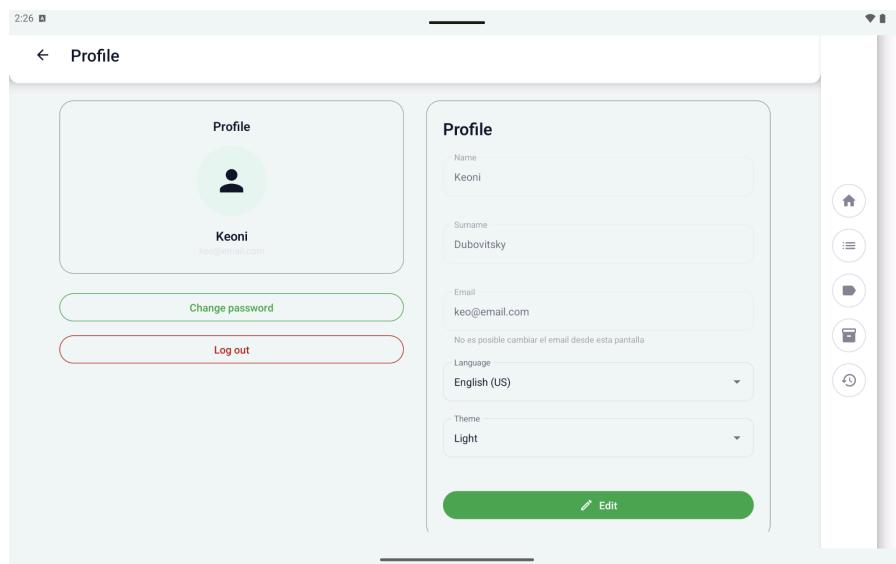


Figura 19.1 Perfil (versión tablet)

3.3. Capturas de pantalla de las vistas que pudieran verse afectadas por la orientación (vertical y horizontal) del dispositivo.



Figura 20.1 login (versión mobile)

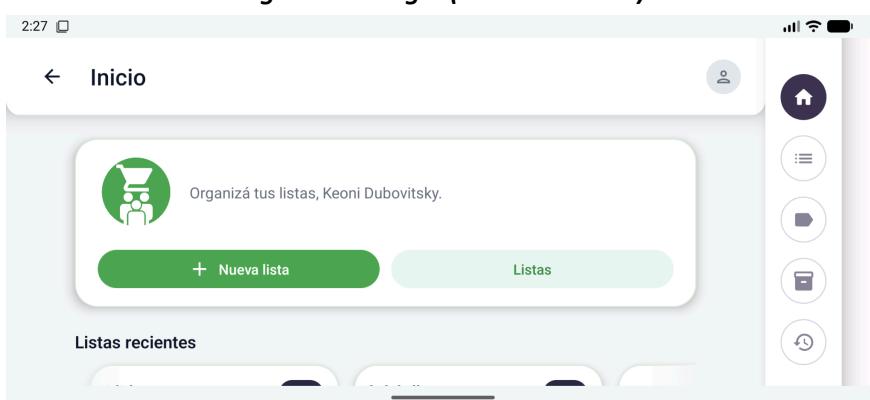


Figura 21.1 home (versión mobile)

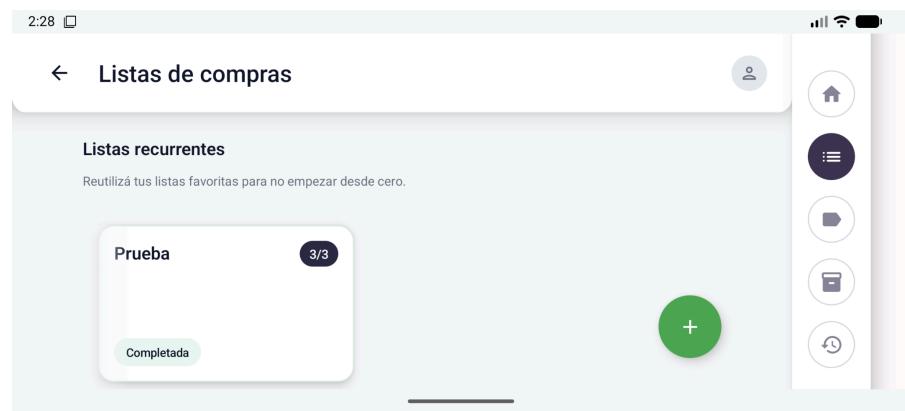


Figura 22.1 Listas (versión mobile)

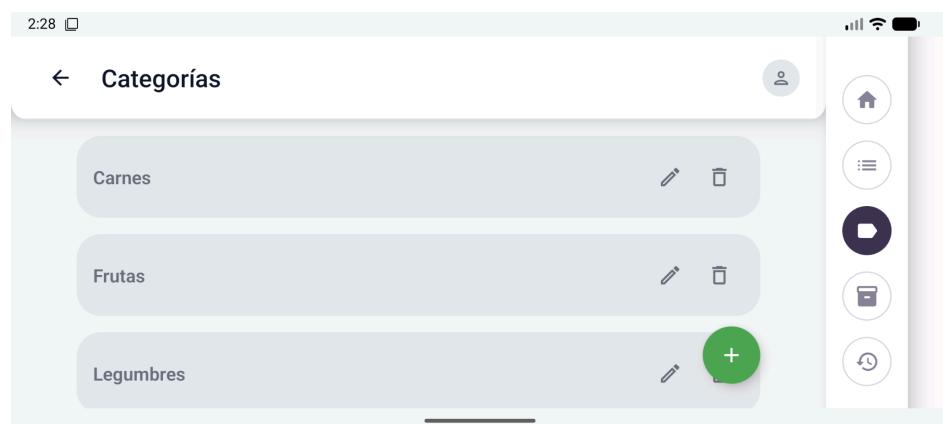


Figura 23.1 Categorías (versión mobile)

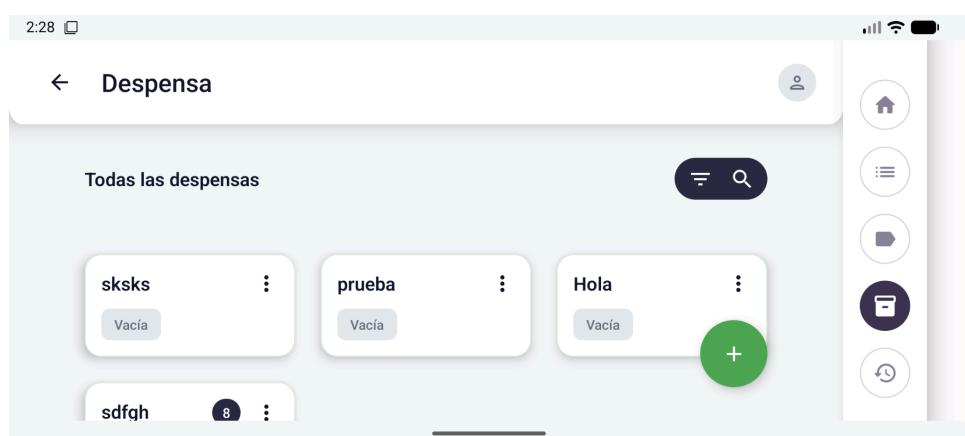


Figura 24.1 Despensa (versión mobile)

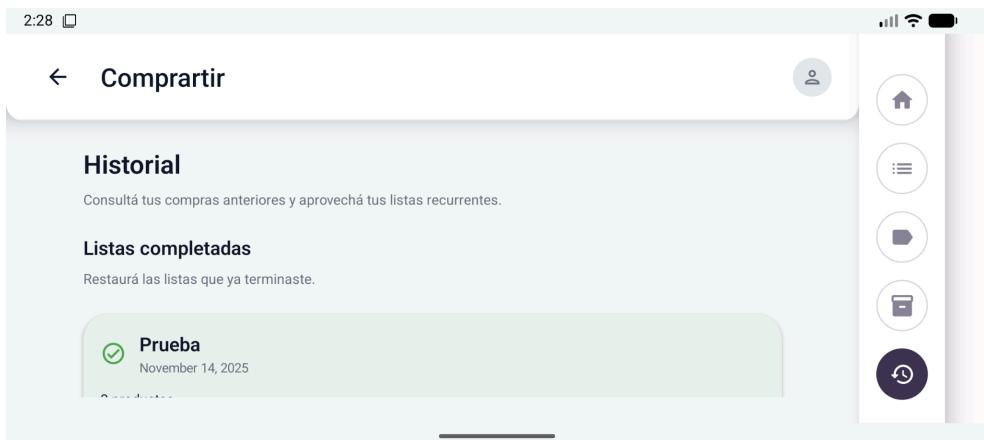


Figura 25.1 Historial (versión mobile)

4. Decisiones de usabilidad tomadas durante la etapa de implementación

Durante la implementación de la aplicación móvil de Compartir se tomaron decisiones de usabilidad fundamentadas en las devoluciones del primer y segundo trabajo práctico (tanto web como móvil), las recomendaciones docentes y los hallazgos de la investigación de usuarios. El objetivo fue adaptar esos aprendizajes al contexto de pantalla pequeña y uso táctil, priorizando las tareas más frecuentes: crear y abrir listas, marcar productos como comprados, gestionar la despensa y compartir listas.

Varios comentarios realizados sobre los prototipos web también se consideraron para la versión móvil, especialmente los relacionados con consistencia de rótulos, uso de colores y jerarquía visual. Además, las correcciones específicas para los prototipos móviles marcaron puntos clave a resolver: evitar adjetivos posesivos como “Mi/Mis” en los títulos, revisar el uso de Bottom Sheets con campos de texto, mejorar el tamaño de ciertos controles táctiles, incorporar una CAB (Contextual Action Bar) y adaptar la navegación a dispositivos tipo tablet.

En cuanto al feedback y manejo de errores, se priorizó la retroalimentación inmediata en contexto, aprovechando patrones móviles como snackbars, cambios de estado visual en botones y mensajes inline debajo de los campos de formulario. Esto permite que el usuario corrija errores sin perder de vista la tarea principal ni cambiar de pantalla.

4.1. Justificación de las diferencias que pudieran presentar las vistas respecto de la versión final de los prototipos.

A diferencia del caso web, donde sí había comentarios de pantallas “abarrotadas”, en los prototipos móviles las observaciones se centraron en **detalles de interacción y ergonomía** más que en la cantidad de información. Las principales diferencias entre los prototipos móviles del primer trabajo práctico y la implementación final responden directamente a esas correcciones:

- **Bottom Sheets vs. teclado virtual**

En los prototipos se utilizaban Bottom Sheets para ingreso de texto (por ejemplo, al agregar productos) que podían entrar en conflicto con el teclado virtual, ocultando campos o botones de confirmación. En la implementación se migraron estos casos a **pantallas completas o diálogos** más integrados al flujo, asegurando que el teclado no tape elementos críticos y que el usuario mantenga el foco en una sola tarea.

- **Controles pequeños para ocultar comprados y ajustar cantidades**

La devolución indicaba que el switch de “ocultar/mostrar” productos comprados y los botones para incrementar/decrementar cantidades podían ser difíciles de tocar. En la versión final se **amplió el tamaño y el padding** de estos controles, se aumentó el contraste y se les dio más separación visual, cumpliendo las recomendaciones de tamaños mínimos para objetivos táctiles.

- **Navegación adaptada a tablet (diseño Master / Detail)**

En la devolución se marcó como “mal” que el sistema de navegación no se hubiera adaptado para los prototipos de tablet en orientación horizontal. En la versión final se implementó un **layout Master/Detail** específico para tablet, donde las listas se muestran en una columna lateral y el detalle de la lista seleccionada en un panel principal. Esto aprovecha mejor el espacio disponible y corrige la falta de adaptación de los prototipos.

- **Acceso al perfil desde la barra superior**

Siguiendo la sugerencia de “desplazar la opción que permite acceder al perfil a la barra superior”, el acceso al perfil se reubicó en el Top App Bar mediante un ícono de cuenta, alineado con el patrón de aplicaciones de Google. Esto libera espacio en la navegación inferior y hace más predecible el acceso a la sección Perfil.

En síntesis, las diferencias respecto de los prototipos móviles no buscan cambiar el concepto general, sino **resolver los problemas puntuales señalados en la devolución** y alinearse mejor con los patrones recomendados para Android.

4.2. Justificación de los aspectos relacionados con el diseño gráfico (colores, tipografías, imágenes, etc.).

El diseño gráfico de la aplicación móvil mantiene la identidad visual definida en Compartir, adaptada al lenguaje de Material Design para Android. Se consolidó un color primario en la gama de verdes de la marca para destacar acciones positivas y elementos interactivos (botones principales, FAB, indicadores activos), reservando el rojo exclusivamente para errores o acciones destructivas. Esta codificación cromática (verde = avanzar/confirmar, rojo = advertencia/riesgo) facilita interpretar el estado del sistema sin leer todo el texto.

La tipografía sigue las recomendaciones de Material Design, utilizando una familia sans-serif con jerarquías claras: títulos y encabezados con mayor tamaño y peso, textos secundarios y notas con tamaños menores y colores más suaves. Esto mejora la legibilidad en pantallas pequeñas y ayuda

a que el usuario identifique rápidamente la información principal (por ejemplo, nombre de la lista o del producto) frente a datos secundarios (unidad, comentarios).

En cuanto a iconografía, se emplean íconos familiares del ecosistema Android (lista, agregar, compartir, perfil, historial) para reducir la curva de aprendizaje. Los estilos de botones, tarjetas y contenedores se mantienen consistentes entre pantallas: un botón primario se ve igual en toda la app, y los estados de error/success comparten los mismos colores. Esta coherencia gráfica refuerza la percepción de sistema unificado y contribuye a la facilidad de uso.

4.3. Justificación haciendo referencia a los lineamientos de diseño de la plataforma Android.

La interfaz móvil se diseñó alineada con los **lineamientos de Material Design para Android**, adaptando patrones de navegación, jerarquía y comportamiento a las expectativas del ecosistema.

Para la navegación principal en dispositivos teléfono se optó por una **Bottom Navigation Bar**, que concentra las secciones más frecuentes (listas, despensa, productos, actividad) y facilita el acceso con el pulgar. En pantallas de mayor tamaño u orientación horizontal, esta barra puede transformarse en un **Navigation Rail** o combinarse con un layout Master/Detail, siguiendo las recomendaciones de diseño adaptativo para diferentes tamaños de pantalla.

Las acciones flotantes (por ejemplo, “Agregar lista” o “Agregar despensa”) se implementan mediante un **Floating Action Button (FAB)** cuando la creación es la acción más frecuente y se quiere resaltar su importancia. Acciones menos frecuentes se ubican en menús de overflow (tres puntos), aplicando el principio de “progressive disclosure” y respetando el rol del App Bar como un indicador contextual..

También se tuvieron en cuenta los **gestos táctiles estándar** de la plataforma: pulsación simple para abrir, activar CAB en listas, y uso de patrones de scroll naturales dentro de listas extensas. Se evitó redefinir gestos de forma inesperada para no entrar en conflicto con la memoria muscular de los usuarios de Android. En general, se busca que la app “se sienta Android” y no una interfaz ajena injertada en el sistema operativo.

4.4. Justificación haciendo referencia a los temas vistos en la materia demostrando la aplicación práctica de los mismos.

Las decisiones implementadas en la app móvil reflejan la aplicación práctica de varios conceptos vistos en la materia:

- **Reducción de carga cognitiva / Ley de Hick**

Al limitar la cantidad de opciones visibles y agrupar acciones avanzadas en menús contextuales, se reduce el tiempo de decisión. Por ejemplo, en el detalle de lista la acción principal (marcar comprados / agregar ítems) está siempre disponible, mientras que mover, compartir o eliminar se accede desde menús secundarios.

- **Ley de Fitts y objetivos táctiles**

El aumento del tamaño y espaciado de controles como los botones +/- de cantidad y el switch de ocultar comprados responde directamente a las recomendaciones de Fitts. Se

minimizan errores de toque y se mejora la precisión, algo especialmente relevante en pantallas de teléfono.

- **Heurísticas de Nielsen**

La app refuerza la visibilidad del estado del sistema (snackbars tras acciones, cambios de color al marcar un ítem como comprado), ofrece control y libertad (confirmaciones antes de acciones destructivas, posibilidad de deshacer en algunos flujos) y previene errores mediante validación temprana de formularios y restricciones sobre inputs inválidos. La consistencia de rótulos e iconos también responde a la heurística de estándares y coherencia.

- **Diseño adaptable**

La diferenciación entre interfaces para teléfono y tablet (Master/Detail en pantallas grandes, layouts más lineales en teléfonos) ejemplifica el diseño adaptativo, donde no solo se escala la UI sino que se reorganiza la jerarquía de información para cada contexto de uso.

4.5. Justificación haciendo referencia a los modelos de Personas.

Las decisiones de usabilidad también se apoyan en los modelos de Personas definidos en etapas anteriores:

- **Santiago (estudiante que comparte compras con compañeros)**

Se priorizó un flujo de **compartir listas simple y visible**, accesible desde el detalle de lista mediante un botón claramente identificado. La posibilidad de seleccionar varios ítems con pulsación larga y aplicar acciones en lote le permite organizar la lista rápida y eficientemente antes de salir a comprar. Las notificaciones y estados de “última actividad” le ayudan a ver qué agregaron sus compañeros sin saturar la pantalla.

- **Celeste (madre que prioriza claridad y rapidez)**

La interfaz móvil presenta botones grandes, listas bien espaciadas y un mínimo de ruido visual, facilitando el uso con una sola mano mientras realiza otras tareas. El flujo para agregar productos está optimizado con campos predefinidos y sugerencias, de modo que puede registrar lo que falta con pocos toques. El toggle para ocultar elementos ya comprados simplifica la vista cuando está efectivamente en el supermercado, enfocándose solo en lo pendiente.

- **Raúl (adulto con menor familiaridad digital)**

Se reforzaron las **confirmaciones claras** y el lenguaje sencillo en todos los mensajes de error y éxito, para que entienda qué está pasando en cada paso. Los iconos se acompañan siempre de texto y se evitó el uso de jerga técnica. Al marcar un producto como comprado o eliminar una lista, el sistema muestra feedback explícito y, cuando es posible, opciones de deshacer, reduciendo la ansiedad a equivocarse.

- **Lucía (usuario que alterna móvil y web)**

La continuidad en la paleta de colores, tipografías y estructura general entre la versión web y la móvil permite que Lucía transfiera lo aprendido en un dispositivo al otro sin reaprender flujos completos. La sincronización de listas y el uso de patrones similares (por ejemplo, cómo se agregan ítems o cómo se comparte una lista) hacen que pueda empezar la tarea en la computadora y terminarla en el celular en el supermercado sin sorpresas.

4.6. Justificación haciendo referencia a las sugerencias y/o correcciones realizadas por los docentes. repetitivo con decisiones de usabilidad?

Por último, se explicita cómo se incorporaron las correcciones docentes a los prototipos de aplicación móvil:

- Se evitó el uso sistemático de adjetivos posesivos como “Mi/Mis” en títulos, reemplazandolos por nombres neutros (“Listas”, “Perfil”, “Despensas”), tal como sugerido.
- Se revisaron los Bottom Sheets con campos de texto y se migraron a pantallas completas o diálogos donde el teclado virtual no tapa campos ni botones, atendiendo la preocupación de conflicto entre Bottom Sheet y teclado.
- Se incrementó el tamaño y espaciado de controles pequeños, especialmente el switch de ocultar/mostrar productos comprados y los botones de cantidad, para mejorar la ergonomía táctil y responder a la observación de que eran difíciles de usar.
- Se implementó un diseño Master/Detail para tablet, adaptando la navegación a dispositivos de mayor tamaño y orientación horizontal, corrigiendo el “mal” indicado en la devolución por reutilizar el layout de teléfono en esos contextos.
- La opción de acceder al perfil se desplazó a la barra superior (Top App Bar / menú de cuenta), en línea con el comportamiento de aplicaciones Android de Google y con la recomendación explícita de la cátedra.

En conjunto, estos cambios muestran que la implementación móvil no se limita a replicar los prototipos iniciales, sino que integra de forma sistemática las correcciones docentes y los principios de usabilidad trabajados durante la materia.

5. Instructivo de instalación

1. Backend (API):

Descargar y descomprimir la carpeta `api`.

Abrir una terminal y moverse a esa carpeta:

```
cd api
```

Ejecutar:

```
npm install
```

Luego levantar la API con:

```
npm run api
```

Dejar esa terminal abierta mientras se usa la app.

2. App móvil (Android):

Descargar y descomprimir la carpeta `compartir-mobile`.

Abrir Android Studio y elegir Open → seleccionar la carpeta `compartir-mobile`.

Esperar a que termine de sincronizar Gradle.

Elegir un emulador Android (o un dispositivo físico conectado) en la barra de herramientas.

Ejecutar la app con el botón Run ▶

6. Conclusión

En esta tercera entrega pasamos de los prototipos del primer trabajo práctico a una interfaz móvil funcional y coherente con los principios de Interacción Hombre-Computadora. Las personas y escenarios de uso definidos en el primer trabajo práctico guiaron las decisiones de diseño en los flujos críticos, priorizando claridad, prevención de errores y feedback inmediato. A partir de esas referencias simplificamos la jerarquía visual, unificamos barras de acción y mejoramos los estados de validación, lo que reduce pasos.

Desde la perspectiva de requisitos, implementamos todos los RFs y cumplimos los RNFs obligatorios. La arquitectura por capas con ViewModels, repositorios y una UI declarativa en Jetpack Compose aporta una base mantenible y extensible para futuras iteraciones. También avanzamos en accesibilidad y consistencia visual de componentes reutilizables.

En síntesis, entregamos una base sólida, minimalista y fácil de usar que materializa los aprendizajes del primer y segundo trabajo práctico y deja un camino claro de mejora incremental hacia la versión final del proyecto.