Módulo 2. Innovación ágil: Lean Startup y prototipado rápido

1. Enfoques ágiles para innovar

2. Herramientas para idear y prototipar soluciones

Referencias

Descarga en PDF

1. Enfoques ágiles para innovar

1. Enfoques ágiles para innovar

En el módulo 1 analizamos el enfoque del *Design Thinking* y comprendimos su utilidad para abordar desafíos desde una perspectiva centrada en las personas. Esta metodología promueve la empatía, la experimentación y la iteración, lo que la convierte en una herramienta flexible y creativa para diseñar soluciones innovadoras. Sin embargo, *Design Thinking* no es el único enfoque ágil que puede aplicarse a proyectos de innovación. Existen otras metodologías que permiten gestionar los procesos de desarrollo de ideas de forma dinámica, adaptativa y colaborativa, con una fuerte orientación a la entrega de valor.

En esta unidad abordaremos tres metodologías ampliamente utilizadas en equipos que impulsan proyectos innovadores: **Scrum, Kanban y Lean Startup**. Si bien todas forman parte del universo ágil, se aplican de manera diferente según las características del proyecto y los objetivos del equipo. Analizaremos sus principios, funcionamiento y ventajas frente a enfoques más estructurados, para comprender cómo ayudan a reducir la

incertidumbre, acortar los ciclos de desarrollo y adaptarse a nuevas necesidades sin perder de vista el valor para las personas usuarias.

Scrum y Kanban: metodologías ágiles para proyectos de innovación

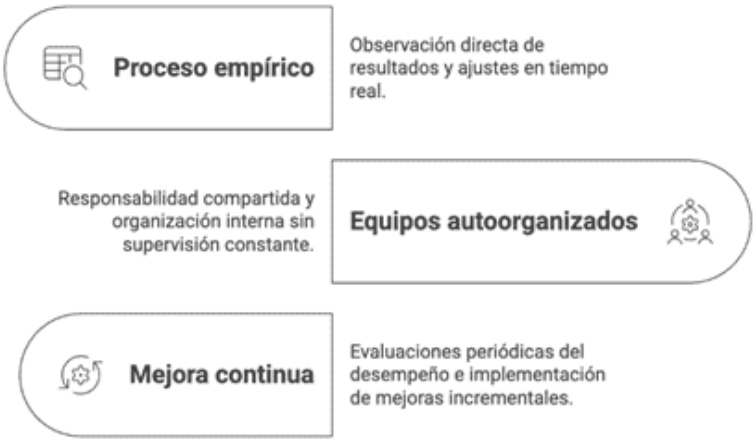
En el trabajo por proyectos, especialmente cuando se busca desarrollar soluciones nuevas, resulta útil contar con metodologías que permitan avanzar sin seguir un plan rígido, incorporar ajustes sobre la marcha y organizar tareas de forma clara. Las metodologías ágiles responden a esta necesidad: son esquemas de trabajo que promueven ciclos cortos, revisión frecuente y mejora continua. Algunas se originaron en el sector tecnológico, otras en la industria, pero hoy se aplican en distintos campos donde se trabaja de forma colaborativa para resolver problemas o diseñar productos y servicios.

A continuación, nos centraremos en dos enfoques ágiles que se utilizan cada vez más para gestionar proyectos de innovación: *Scrum* y *Kanban*. Cada uno propone una lógica distinta para organizar el trabajo, pero comparten una misma intención: **facilitar que los equipos avancen de manera ordenada, identifiquen lo que funciona y lo que no, y tomen decisiones con base en la experiencia acumulada.**

- **Scrum: ciclos cortos para desarrollar soluciones**

En 1995, Jeff Sutherland y Ken Schwaber presentaron por primera vez el marco de trabajo conocido como *Scrum*. Su objetivo era encontrar una manera más eficaz de organizar el desarrollo de software, y para ello se basaron en tres principios simples pero potentes:

Figura 1. Principios de *Scrum*



Fuente: elaboración propia

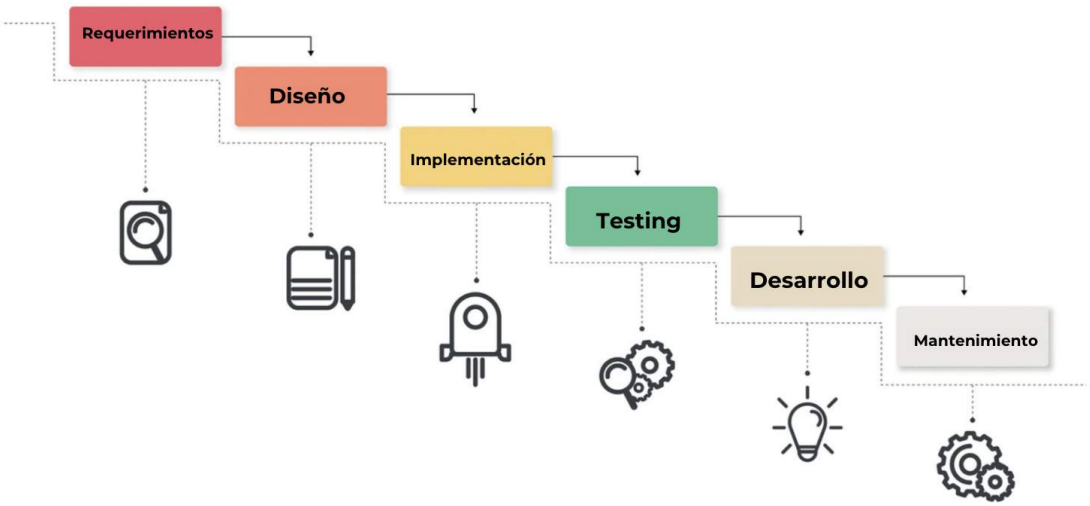
Estas tres ideas dieron forma a un marco de trabajo que, con el tiempo, se convirtió en una referencia para equipos que desarrollan nuevas soluciones, especialmente cuando el proceso requiere probar, ajustar y volver a intentar. Aunque *Scrum* nació en el ámbito del *software*, hoy se utiliza en sectores tan distintos como la industria automotriz, la biotecnología o los servicios públicos. No importa qué se está desarrollando, sino **cómo se organiza el equipo para avanzar con claridad, detectar lo que funciona y modificar lo que no.**

Scrum resulta particularmente útil en proyectos que demandan creatividad e innovación. Los principios que lo sostienen —proceso empírico, equipos autoorganizados y mejora continua— permiten trabajar con ideas que no están completamente definidas desde el inicio. A través de ciclos breves, planificación iterativa y dinámicas colaborativas, se crean condiciones para experimentar, equivocarse rápido y mejorar con base en la experiencia real. Veamos, entonces, cómo estos conceptos se aplican al desarrollo de soluciones innovadoras y transforman la forma en que se organizan los equipos para crear.

Planificación iterativa vs. planificación en cascada

Una de las diferencias más relevantes entre los enfoques tradicionales de planificación y los métodos ágiles está en **cómo se concibe el proceso de trabajo.** En los modelos tradicionales, se suele utilizar la **planificación en cascada,** es decir, cada etapa del proyecto se realiza de forma secuencial. Primero se identifican los requisitos, luego se diseña el sistema, producto o servicio, se implementa, se prueba, se desarrolla y, por último, se realiza el mantenimiento.

Figura 2. Planificación secuencial



Fuente: Coworking FY, 2025, https://goo.su/yg1UI4

Cada paso depende de la finalización del anterior y el proyecto no avanza hasta que se completa por completo la etapa previa. Este tipo de planificación puede ser funcional cuando los objetivos y procedimientos están claramente definidos desde el inicio y se busca cumplir con una secuencia cerrada. Sin embargo, presenta limitaciones cuando se aplica a proyectos donde todavía se están definiendo soluciones o cuando es necesario poner a prueba distintas alternativas antes de avanzar.

Imaginemos un equipo que desarrolla un nuevo sistema de atención ciudadana para una municipalidad. Se diseña un recorrido completo con formularios, respuestas automatizadas y flujos de derivación. Todo se aprueba al inicio, se programa, se lanza… Y luego se detecta que muchas personas abandonan el trámite en el segundo paso. Bajo una lógica de planificación en cascada, el equipo ya tendría implementado todo el

sistema y no tendría margen para ajustar sin rearmar procesos enteros. Esto retrasa la mejora e impide actuar rápidamente sobre los puntos críticos.

Frente a este tipo de situaciones, la planificación iterativa —concepto ya trabajado en el marco del *desing thinking*— resulta más adecuada. En lugar de ejecutar todo el sistema desde el comienzo, se desarrolla una primera versión básica del recorrido. A partir de su uso, el equipo observa cómo interactúan las personas, detecta problemas o dificultades, y ajusta el diseño para el siguiente ciclo. Cada iteración se convierte en una oportunidad para mejorar la solución sin necesidad de detener todo el proyecto. Es una lógica de avance por tramos, donde lo aprendido se incorpora de manera continua.

Figura 3. Planificación iterativa

Fuente: [imagen sin título sobre planificación iterativa], (s.f.), https://goo.su/1Wn3POE

Este modelo representa una forma cíclica de organización del trabajo. Comienza con una planificación inicial que establece un objetivo general y un conjunto reducido de acciones prioritarias. A partir de ahí, el equipo avanza en un ciclo continuo que integra distintas etapas conectadas entre sí: se definen los requerimientos (por ejemplo, qué funcionalidades se necesitan en una primera versión), se analizan y diseñan las soluciones posibles, se implementan las propuestas seleccionadas, se prueban en condiciones reales y se evalúan los resultados obtenidos. Con base en esta evaluación, se ajusta la planificación del próximo ciclo.

- **Dinámica de trabajo colaborativo: células de trabajo**

Pensemos en una organización de trabajo tradicional, generalmente jerárquica. En este tipo de estructura, las funciones están claramente separadas, las decisiones fluyen desde los niveles superiores hacia abajo, y cada área trabaja de forma relativamente aislada del resto. Si lo trasladamos a un equipo de innovación, este modelo podría verse así: existe una dirección que define el enfoque general; luego, un equipo técnico desarrolla propuestas; después, un área de diseño implementa soluciones visuales; estas pasan por validación en otra instancia, muchas veces con revisiones cruzadas entre distintas áreas; y, finalmente, otro sector se encarga de ejecutar. Cada paso requiere aprobaciones,

correcciones y reenvíos. El flujo de trabajo es largo, fragmentado y dependiente de múltiples autorizaciones.

Frente a esta dinámica tradicional, la metodología ágil propone otra forma de organización: el trabajo en células. Se trata de equipos reducidos, multidisciplinarios y autogestionados que tienen la capacidad de planificar, ejecutar y revisar su trabajo de manera continua. En lugar de depender de decisiones externas y secuencias rígidas de validación, estas células concentran en su interior los perfiles necesarios para avanzar de forma colaborativa y flexible.

Retomemos la figura 3 sobre planificación iterativa. Las células ágiles se organizan en torno a esta lógica de trabajo cíclico. En cada ciclo —también llamados *sprint*, se involucran todas las etapas del proceso: planificación, análisis y diseño, implementación, desarrollo, testeo y evaluación. A diferencia de los modelos jerárquicos, estas etapas no están distribuidas entre distintos departamentos que trabajan por separado, sino que son asumidas por un mismo equipo que colabora de principio a fin.

Dentro de esta célula, los perfiles se integran según las necesidades del equipo, y si bien la composición tiende a mantenerse estable, pueden incorporarse perfiles de apoyo para proyectos específicos. En un equipo de innovación, lo más habitual es contar con los siguientes perfiles:

- ***Product owner.*** Define los objetivos del producto, gestiona las prioridades del equipo y representa

la perspectiva de las personas usuarias.

- ***Scrum master*:** facilita el proceso de trabajo, promueve la mejora continua y colabora para que se eliminen los obstáculos que puedan afectar el ritmo del equipo.

- **Equipo de desarrollo:** compuesto por personas con distintos saberes técnicos, de diseño, análisis o comunicación, que llevan adelante la planificación, implementación y evaluación de cada iteración.

Estos perfiles trabajan de manera coordinada, sin compartimentos estancos, y participan en todas las etapas del ciclo: desde la planificación hasta la evaluación. La colaboración es constante y las decisiones se toman en conjunto, con foco en la mejora continua.

Entonces, supongamos que el equipo está desarrollando una aplicación para que estudiantes de escuelas secundarias puedan compartir propuestas de mejoras para sus instituciones, comentarlas entre sí y votar cuáles consideran prioritarias. La idea surge a partir de una necesidad concreta: generar un espacio de participación que no dependa exclusivamente de las instancias formales como los centros de estudiantes o las reuniones escolares.

En la reunión de planificación del ciclo, el *product owner* presenta como objetivo desarrollar una primera versión en la que se puedan cargar

ideas, ver las propuestas de otras personas y seleccionar las que parecen más relevantes. El equipo organiza las tareas: una persona diseña la pantalla principal con las categorías de propuestas, otra se ocupa de programar el botón de votación y otra arma una pequeña guía para que quienes usen la aplicación por primera vez entiendan cómo funciona. Durante la semana, el *scrum master* acompaña los avances, organiza reuniones breves para coordinar y ayuda a resolver obstáculos. Al final del ciclo, el equipo se reúne para revisar lo que funcionó bien, qué dificultades encontraron y qué pueden mejorar para seguir trabajando en la próxima versión.

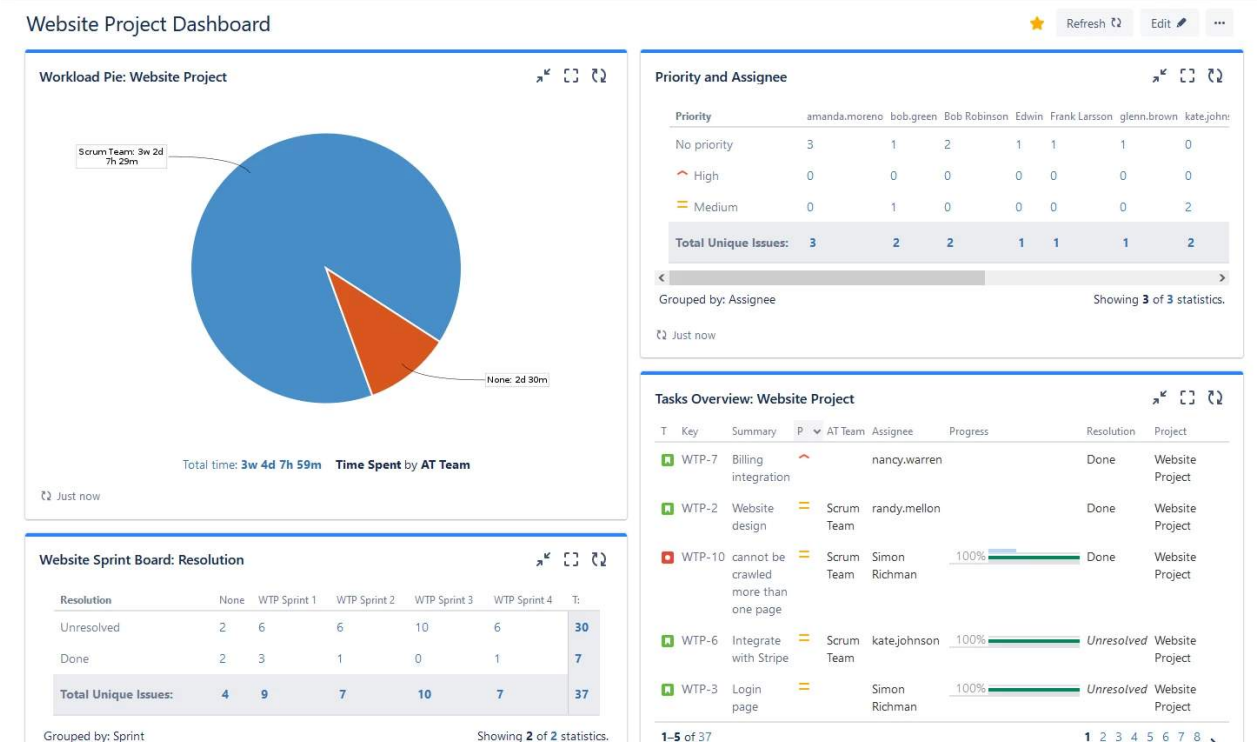
- **Gestión transparente del flujo de tareas**

En una célula ágil organizada bajo la metodología *Scrum*, uno de los principios centrales es la transparencia del trabajo. Esto significa que todas las personas del equipo deben poder visualizar, en tiempo real, qué tareas están planificadas, cuáles están en curso, quién las tiene asignadas y cómo avanza el proyecto. Esta visibilidad compartida evita malentendidos, mejora la coordinación y facilita la toma de decisiones en conjunto.

Para lograrlo, se utilizan *dashboards* o tableros visuales que centralizan toda la información relevante. En lugar de depender de documentos dispersos o intercambios aislados, el equipo accede a una vista unificada del flujo de trabajo. Herramientas como Jira, Trello, ClickUp o Asana permiten configurar tableros personalizados donde se visualizan tareas

organizadas por estado, responsables asignados, prioridades, fechas clave y métricas básicas.

Figura 4. Ejemplo de *dashboard* de seguimiento en Jira



Fuente: Activity Timeline, 2025, https://surl.li/xwtmbc

En el gráfico circular del panel izquierdo, se visualiza el tiempo total invertido en el proyecto. La mayor parte corresponde al *scrum team*, es decir, el equipo completo que incluye al *product owner*, *scrum master* y las personas del equipo de desarrollo. Esto indica que la mayoría de las tareas fueron asumidas colectivamente por el equipo, mientras que una pequeña proporción figura aún sin asignar. A la derecha, un resumen por prioridad y persona asignada permite identificar la carga de trabajo y

detectar rápidamente si hay tareas sin responsables o si hay una distribución desigual. En este caso, muchas tareas aún no tienen prioridad definida. El panel inferior ofrece una visión por *sprint*, con la cantidad de tareas resueltas y no resueltas en cada ciclo. Esta información es clave para analizar la productividad del equipo, identificar cuellos de botella o revisar la planificación de próximos sprints. Finalmente, el listado de tareas muestra de manera clara el estado de avance de cada una: título, persona asignada, nivel de progreso, resolución y sprint correspondiente. Esta vista permite tomar decisiones rápidas —como reasignar una tarea estancada o revisar un entregable urgente— sin necesidad de depender de reuniones o reportes adicionales.

Este tipo de visualización mejora la coordinación, permite distribuir mejor la carga de trabajo y facilita ajustes en función del avance real. Pensemos nuevamente en el ejemplo del equipo que está desarrollando una aplicación para que estudiantes de escuelas secundarias puedan compartir propuestas de mejoras para sus instituciones. Si el panel muestra que casi todo el tiempo se está concentrando en tareas de programación, pero no hay avances en actividades vinculadas a la prueba con usuarios o al diseño de contenidos explicativos, el equipo puede decidir reequilibrar sus esfuerzos en el siguiente sprint. De ese modo, la visualización actúa como una alerta temprana que ayuda a intervenir antes de que un desajuste afecte el desarrollo general del proyecto.

Otra forma de visualizar el avance y el flujo de trabajo en equipos ágiles es a través de tableros Kanban. Este enfoque pone el foco en cómo se mueven las tareas a lo largo del proceso, permitiendo identificar

fácilmente en qué etapa está cada una, cuántas están activas, y dónde podrían surgir cuellos de botella. Se trata de una herramienta muy utilizada en equipos de contenido y comunicación, que veremos en detalle a continuación.

Kanban para flujos de trabajo de innovación

El término Kanban proviene del japonés y significa literalmente «cartel» o «tarjeta visual». Esta denominación refleja con precisión el principio que dio origen al sistema: usar señales visuales para activar tareas dentro de un proceso. Su historia se remonta a fines de los años 40, cuando Taiichi Ohno, ingeniero industrial de Toyota, buscaba mejorar la eficiencia de las plantas automotrices japonesas. Su inspiración provino de un lugar inesperado: un supermercado estadounidense. Observó que los productos eran repuestos solo cuando se vendían, lo que generaba un sistema de abastecimiento basado en la demanda real.

Ohno trasladó esa lógica al entorno industrial. En lugar de producir piezas por adelantado, propuso que cada estación de trabajo «pidiera» lo que necesitaba al paso anterior mediante una tarjeta: el kanban. Así, si se vaciaba un contenedor de piezas, la tarjeta asociada volvía al proceso anterior como señal de que debía reponerse. Esto dio origen a un sistema de producción en flujo continuo, ajustado a la demanda real y sin acumulación innecesaria de inventario. Este modelo, conocido como just-in-time, buscaba minimizar el desperdicio y mantener la producción en equilibrio.

Con el tiempo, esta lógica visual y secuencial se trasladó del mundo industrial a otras áreas donde también se manejan flujos de trabajo complejos, como el desarrollo de software, los servicios y los proyectos de innovación. En estos entornos, Kanban ya no se aplica con tarjetas físicas, sino mediante tableros digitales donde se visualiza cada etapa del proceso y el recorrido de cada tarea. Lo que permanece es el enfoque: visibilizar el trabajo para mejorarlo.

Para implementar esta lógica de forma práctica, existen diversas herramientas digitales que permiten diseñar y gestionar tableros Kanban, adaptadas a las necesidades de equipos de distintos tamaños y sectores. Algunas de las más utilizadas son Trello, Jira, ClickUp, Asana y Monday. Estas plataformas ofrecen interfaces visuales en las que las tareas se representan como tarjetas que se mueven entre columnas, reflejando su estado dentro del flujo de trabajo. Permiten asignar responsables, agregar fechas de entrega, etiquetas, comentarios y archivos adjuntos, lo que facilita la coordinación y el seguimiento colaborativo.

Figura 5. Ejemplo de tablero *Kanban*

Fuente: Atlassian, s.f., https://goo.su/a8E2sr

El objetivo principal de *Kanban* es lograr un flujo de trabajo continuo, flexible y eficiente. En lugar de imponer estructuras rígidas o crear roles nuevos, propone observar cómo se trabaja en el presente y organizarlo de forma visual para facilitar su gestión. Al mostrar el estado real de cada tarea, el equipo puede detectar cuellos de botella, evitar bloqueos y tomar decisiones con base en datos concretos. La mejora es progresiva, no disruptiva, y el foco está puesto en el trabajo, no en el control de las personas.

En general, como observamos en la figura, los tableros Kanban se organizan en columnas que representan los distintos estados por los que pasa una tarea desde que se planifica hasta que se completa. Las más comunes son «Por hacer» («To do»), «En proceso» («Doing») y «Hecho» («Done»). Sin embargo, cada equipo puede adaptar esta estructura según sus necesidades específicas, agregando columnas intermedias como

«Backlog» (pendientes o ideas en espera), «Up Next» (prioridades próximas), «In Progress» (en curso), «On Hold» (en pausa) o incluso secciones auxiliares como «Questions» (consultas o temas abiertos), cuando el proyecto lo requiere. El objetivo es que cada columna represente con claridad una etapa real del flujo de trabajo del equipo.

En los proyectos de innovación, donde las tareas suelen ser interdependientes y los resultados están sujetos a prueba y error, un tablero *Kanban* permite organizar de manera visual todo el proceso. Por ejemplo, si un equipo está desarrollando una serie de mejoras en un sistema de transporte urbano, podría armar un tablero con columnas como «Ideas», «En análisis», «En desarrollo», «En testeo», «Pendiente de revisión» y «Validado». Cada tarjeta representaría una funcionalidad específica, como la incorporación de sensores para el control de velocidad, la integración con una *app* de seguimiento en tiempo real o la implementación de un sistema de alerta para mantenimiento preventivo. A medida que las tareas avanzan, se desplazan de una columna a otra, lo que permite detectar posibles bloqueos. Por ejemplo, si se acumulan muchas tarjetas en la columna de «Testeo», eso podría señalar que hay demoras en conseguir usuarios o contextos adecuados para hacer pruebas.

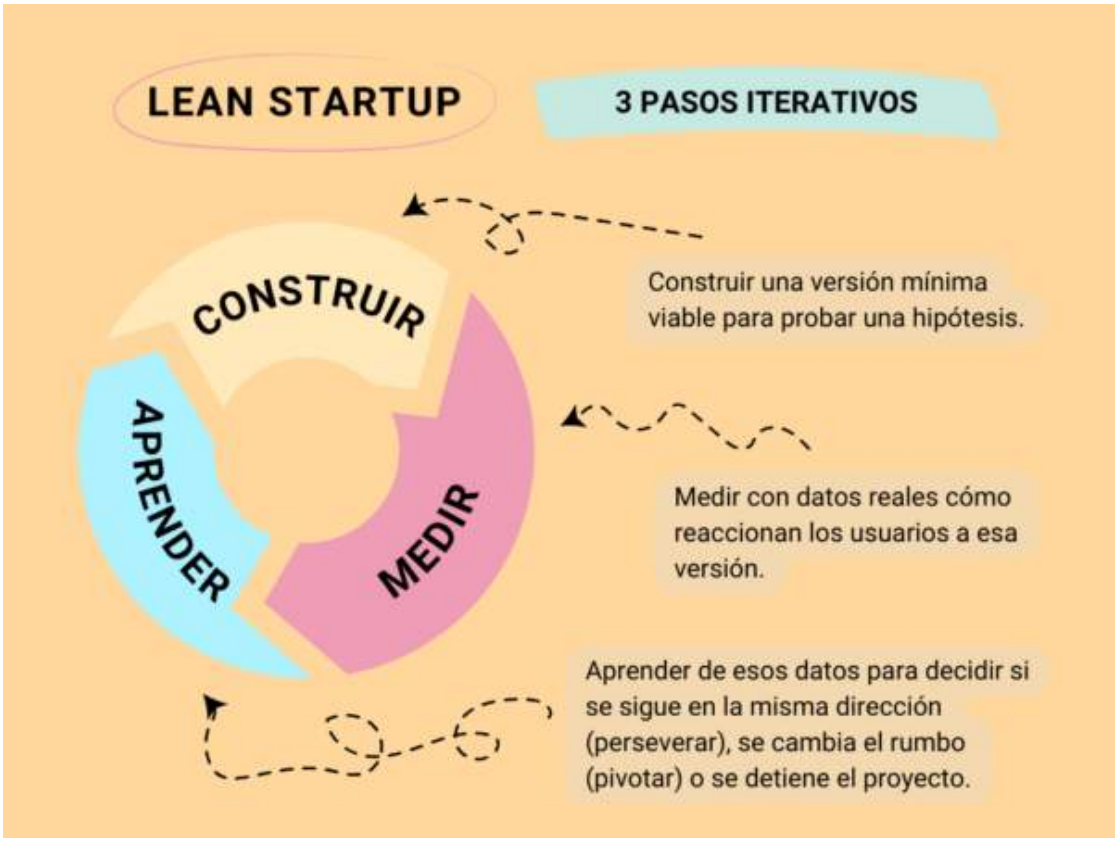
Lean Startup: proceso y principios

El enfoque *Lean Startup* fue desarrollado por Eric Ries en Silicon Valley, a partir de su experiencia como emprendedor en el mundo de las startups tecnológicas. Frustrado por los fracasos que resultaban de invertir meses —o incluso años— en desarrollar productos que después nadie quería,

Ries propuso un modelo completamente distinto: lanzar versiones mínimas del producto, observar cómo reaccionan las personas usuarias, y ajustar el rumbo en función de los aprendizajes reales. Esta lógica se basa en los principios de la *lean manufacturing*, nacida en Toyota, que buscaba eliminar desperdicios y optimizar procesos. En el caso del *Lean Startup*, el «desperdicio» no son las piezas sobrantes, sino el tiempo, el esfuerzo y el dinero mal invertidos en ideas sin validar. Por eso, en lugar de partir de certezas, el proceso comienza con hipótesis que deben ser puestas a prueba. Así, mediante ciclos breves de experimentación, medición y adaptación, se avanza hacia soluciones que respondan realmente a las necesidades de las personas, evitando suposiciones infundadas y reduciendo el riesgo al mínimo.

El proceso de *Lean Startup* se organiza en un ciclo de tres etapas muy concreto e iterativo:

Figura 6. Proceso de *Lean Startup*



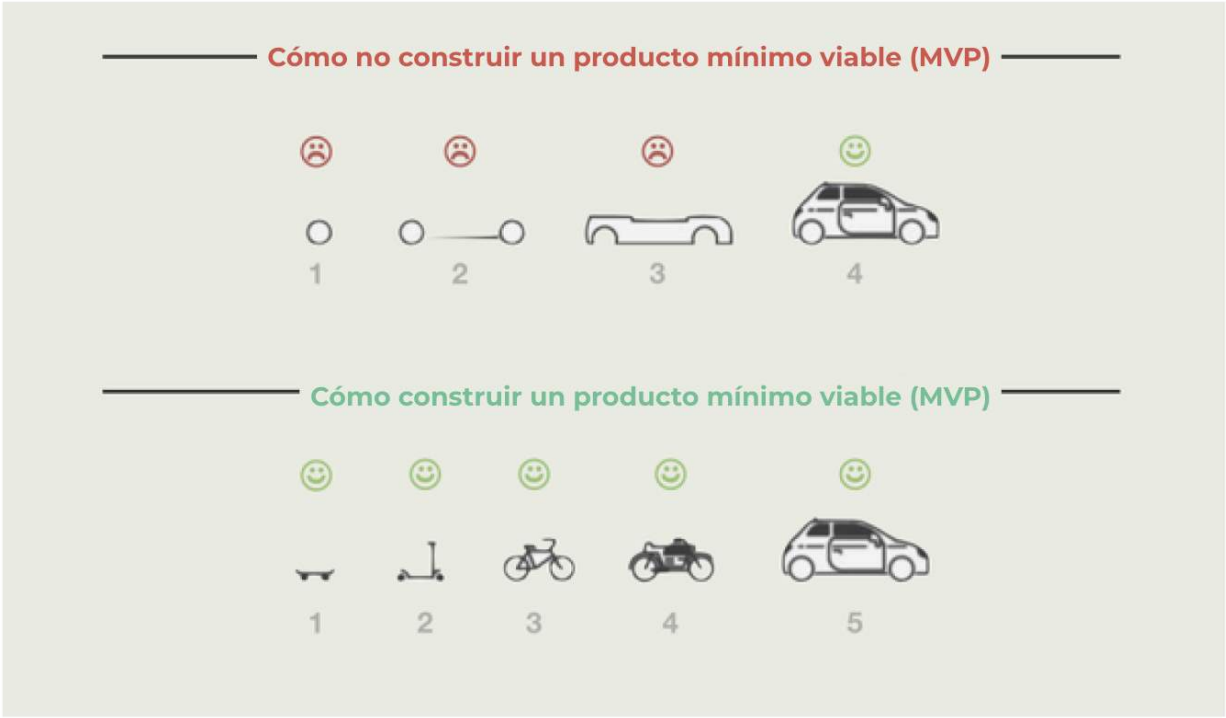
Fuente: Cruz, 2025, https://surl.lu/lumqwq

Este proceso permite validar ideas paso a paso, con foco en el uso real que las personas hacen del producto, y no solo en su desarrollo técnico. Para entenderlo mejor, imaginemos que un equipo está trabajando en una solución para reducir el desperdicio de alimentos en **panaderías, supermercados y restaurantes**, canalizando esos excedentes hacia comedores comunitarios que los necesitan. La idea final es crear una plataforma digital que conecte comercios que tienen alimentos próximos a vencer o sobrantes del día con organizaciones barriales encargadas de repartir comida. Veamos cómo implementar el proceso *Lean Startup*:

Construir

En esta primera etapa, el objetivo no es desarrollar la versión definitiva del producto, sino generar un **producto mínimo viable** —o MVP, por sus siglas en inglés—: una versión simple pero funcional que permita poner a prueba una idea y aprender de su uso real. El MVP no tiene que ser perfecto ni contar con todas las funciones previstas para el producto final, sino ofrecer una primera solución concreta que ya permita resolver el problema de forma parcial. La siguiente imagen ayuda a entender esta lógica:

Figura 7. Producto mínimo viable (MVP)



Fuente: Karimi, 2017, https://surl.li/ootqoh

Como se ve en la parte inferior, la clave está en construir versiones que realmente funcionen desde el comienzo, aunque sean básicas. En lugar de avanzar con partes aisladas (como una rueda o un chasis, que por sí solos no sirven para moverse), se parte de una solución más simple —como un monopatín— que ya resuelve el problema de trasladarse. A partir de ahí, cada nueva versión incorpora mejoras, pero siempre manteniendo la funcionalidad.

Entonces, en el caso del sistema para distribuir excedentes alimentarios, el primer MVP podría ser algo tan sencillo como **un grupo de WhatsApp** en el que algunos comercios informan qué productos tienen disponibles cada día, y los comedores confirman si pueden pasar a retirarlos. Esta solución no requiere desarrollar *software*, pero ya permite validar si hay interés real, si la logística es viable, y qué tipo de productos circulan. Ese aprendizaje inicial es clave para decidir si vale la pena seguir avanzando con el proyecto, y en qué dirección. Luego, con lo aprendido, se podría desarrollar un segundo MVP más sofisticado —como un formulario digital y una planilla compartida— y así continuar el ciclo.

Medir

Una vez que se pone en marcha el primer MVP, el siguiente paso es observar cómo lo usan las personas y recolectar información concreta sobre su funcionamiento. El foco no está en métricas generales, sino en datos que permitan evaluar si la solución realmente está resolviendo el problema para el cual fue creada.

Por ejemplo, si el MVP consiste en un grupo de mensajería donde los comercios pueden avisar cuando tienen alimentos disponibles y alguien del comedor pasa a buscarlos, lo importante es medir cuántos avisos se generan por semana, cuántas entregas se concretan, si hay demoras o faltantes, cómo se coordinan los intercambios y si las personas entienden fácilmente cómo participar.

También se pueden identificar obstáculos: ¿hay horarios difíciles de coordinar?, ¿alguien recibe siempre más donaciones que otros?, ¿hay comercios que ofrecieron una vez y no volvieron a hacerlo? Toda esa información ayuda a detectar qué aspectos funcionan bien, cuáles generan fricción y qué debería cambiarse en la próxima versión.

Aprender

Con los datos recolectados, el equipo analiza qué funcionó y qué no para tomar decisiones informadas. Este es el momento clave del proceso Lean: no se trata solo de seguir con lo planeado, sino de ajustar en base a lo aprendido.

Siguiendo con el ejemplo, imaginemos que muchos comercios usan el grupo de mensajería, pero hay confusión porque los mensajes se pierden o se repiten. A partir de esa observación, el equipo podría decidir crear una lista organizada con horarios de retiro, o un formulario simple donde cada comercio cargue su oferta y el comedor más cercano reciba una notificación. Es decir, la próxima versión no es necesariamente más compleja, pero sí mejor orientada a resolver los problemas reales detectados en el uso.

Este ciclo —construir, medir, aprender— se repite todas las veces que sea necesario, sumando mejoras en cada vuelta. Así se avanza paso a paso, reduciendo el riesgo y diseñando soluciones que realmente se ajustan a las necesidades de quienes las van a usar. En la próxima unidad, vamos a explorar las herramientas que permiten idear y prototipar esas soluciones, para transformar ideas iniciales en versiones concretas que puedan ponerse a prueba.

CONTINUAR

Lección 2 de 4

2. Herramientas para idear y prototipar soluciones

Herramientas para idear y prototipar soluciones

En la unidad anterior vimos cómo metodologías como *Lean Startup* y *Scrum* permiten gestionar proyectos de innovación en contextos inciertos, a través de ciclos breves de construcción, validación y mejora continua. A lo largo del proceso, se parte de una idea general, se la traduce en un producto mínimo viable y se lo pone a prueba para aprender rápidamente y ajustar según la experiencia real de las personas usuarias. Este enfoque reduce el riesgo, optimiza los recursos y asegura que las soluciones se diseñen en función de necesidades concretas.

Ahora bien, para poder avanzar con esas ideas y empezar a testearlas, es fundamental contar con herramientas que nos permitan convertir conceptos en algo tangible. En esta unidad vamos a conocer dos tipos de recursos que nos permiten dar ese paso: los prototipos de baja fidelidad, como el papel, que permiten testear rápidamente, y los prototipos digitales interactivos, que simulan cómo funcionaría una versión más avanzada del producto. Ambas herramientas son esenciales para dar forma a las ideas, recibir devoluciones tempranas y seguir iterando con base en el uso real.

Prototipos de baja fidelidad: del papel a las primeras pruebas

El término «fidelidad» en diseño se refiere al nivel de detalle de un prototipo. Uno de **baja fidelidad** tiene pocos elementos visuales, no simula interacciones reales y suele estar hecho a mano. Su objetivo es enfocarse en la **funcionalidad general** de una propuesta, más que en su aspecto visual o su comportamiento técnico final.

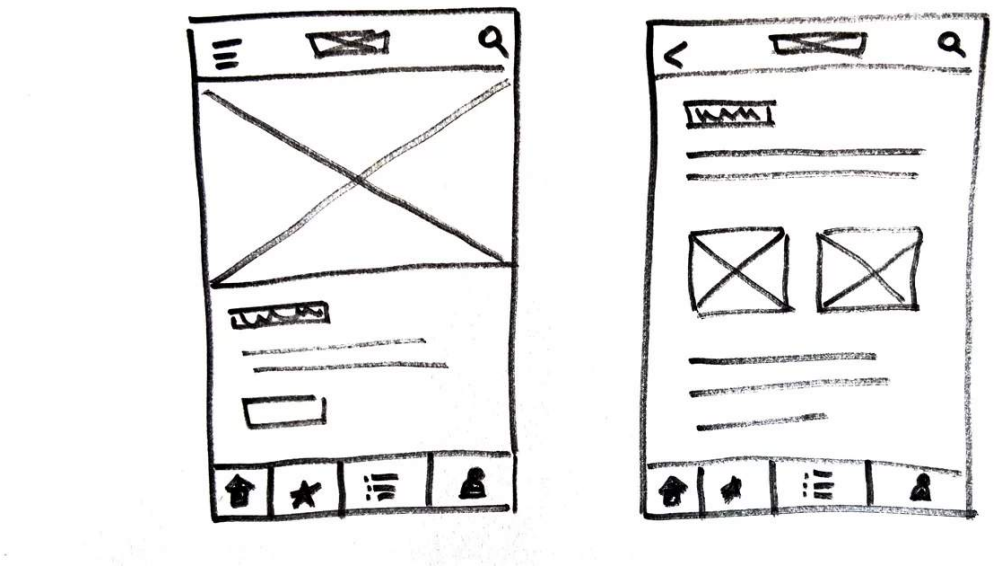
Los prototipos de baja fidelidad permiten **dar forma inicial a una idea** de manera rápida, simple y económica. No están pensados para verse como una versión terminada, sino para ayudar al equipo a **entender cómo funcionaría una solución**, qué componentes debería incluir, en qué orden se usarían y si hay aspectos confusos o que conviene revisar. Gracias a esta sencillez, resultan especialmente útiles para testear conceptos, recoger devoluciones tempranas y hacer ajustes sin invertir demasiado tiempo o recursos.

Este tipo de prototipos no requiere conocimientos técnicos ni herramientas digitales: basta con lápiz, papel y algo de creatividad. Por eso, **facilitan la participación de todo el equipo**, incluso de personas sin formación en diseño. Al permitir representar ideas de forma tangible, ayudan a alinear visiones, compartir propuestas y resolver dudas en etapas tempranas del proceso.

Existen diferentes formas de representar ideas mediante prototipos de baja fidelidad. Algunas de las más comunes incluyen las siguientes:

1. ***Wireframes* dibujados a mano.** Se trata de esquemas simples que representan la estructura de una interfaz (como una página web o una *app*), sin detalles visuales. Ayudan a definir qué elementos debe tener cada pantalla y cómo se organizan. Son útiles para alinear al equipo sobre la lógica general de navegación.

Figura 8. Ejemplo de prototipo de baja fidelidad - *Wireframe* dibujados a mano

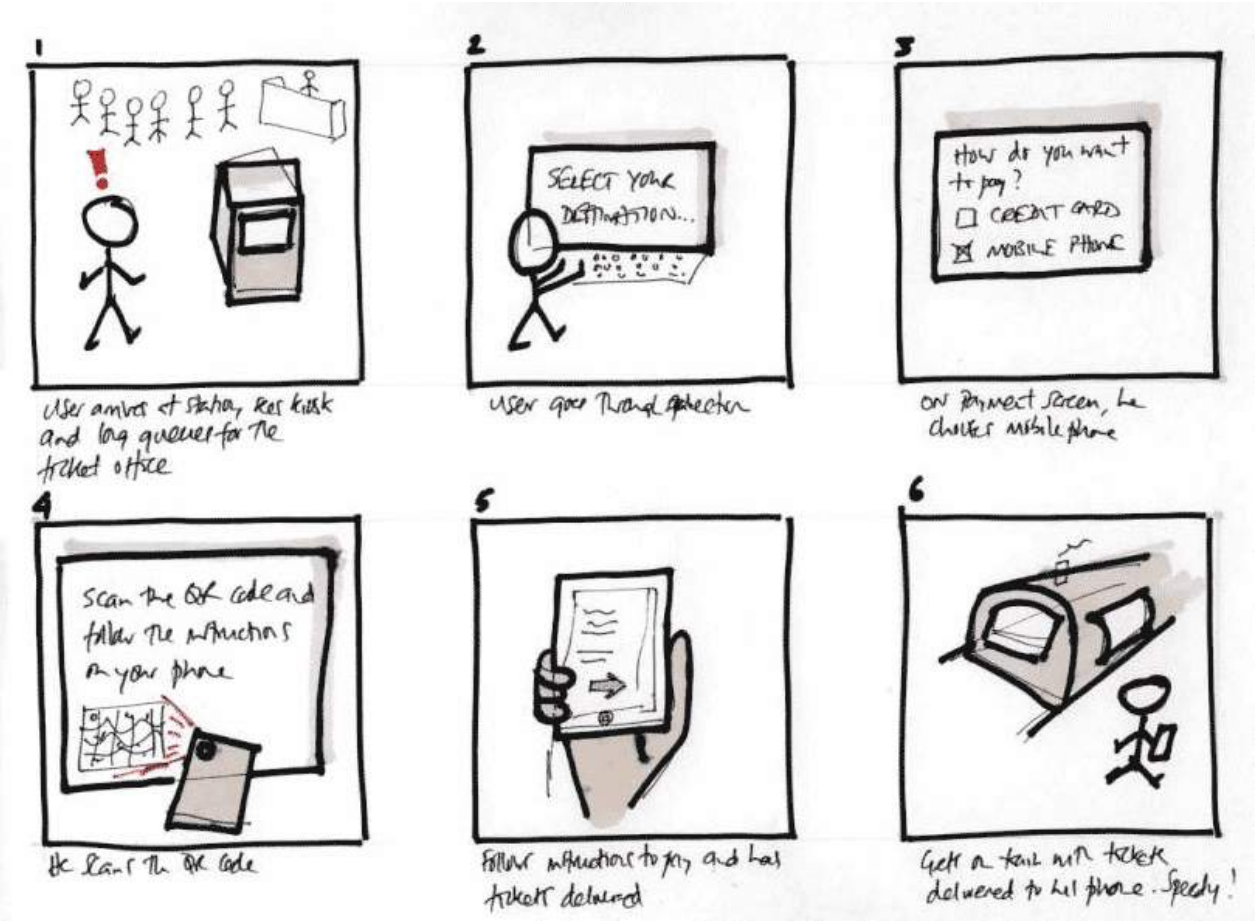


Fuente: Busquet, s.f., https://surl.li/qxietr

2

Storyboards. Son secuencias de viñetas que narran una situación de uso. A través de escenas ilustradas, muestran cómo una persona interactuaría con el producto o servicio en un contexto determinado. Permiten entender el recorrido del usuario y detectar puntos críticos o momentos clave de la experiencia.

Figura 9. Ejemplo de prototipo de baja fidelidad - *Storyboards*



Fuente: User Focus, s.f., https://surl.li/xfbaka

3

Prototipos en papel con elementos móviles. Consisten en dibujos sobre papel que incluyen piezas recortadas o *post-its* que pueden moverse o superponerse para simular interacciones (como botones, pantallas que cambian, formularios que se completan, etc.). Esto permite testear flujos básicos sin recurrir a lo digital.

Figura 10. Ejemplo de prototipo de baja fidelidad - Prototipos en papel con elementos móviles

Fuente: Interaction Desing Foundation, s.f., https://surl.li/iuylke

Estos distintos formatos de prototipos de baja fidelidad no solo permiten materializar ideas de forma ágil, sino que también fomentan la conversación en torno al diseño. Al tener algo concreto sobre la mesa, es más fácil detectar problemas, proponer mejoras y tomar decisiones en conjunto. Lejos de buscar una solución definitiva, este tipo de prototipos ayudan a explorar posibilidades, aprender rápido y avanzar con mayor claridad hacia versiones más desarrolladas. En el siguiente apartado, veremos cómo dar un paso más incorporando prototipos digitales interactivos, que permiten simular con mayor precisión la experiencia de uso.

Prototipos digitales: simulaciones interactivas

A diferencia de los prototipos de baja fidelidad, los prototipos digitales se consideran de alta fidelidad porque se acercan mucho más a cómo funcionaría el producto final. Incluyen mayor nivel de detalle visual —como tipografías, colores y distribución real del contenido—, y permiten simular interacciones, como clics, transiciones o navegación entre pantallas. Esto los convierte en herramientas valiosas para testear la experiencia de uso en condiciones más realistas, validar flujos complejos y anticipar cómo se comportará la solución en manos de las personas usuarias. Aunque su desarrollo requiere más tiempo y herramientas específicas, resultan especialmente útiles en etapas avanzadas del proceso, cuando ya se tiene claridad sobre la propuesta general y se busca afinar detalles.

Para diseñar prototipos digitales interactivos existen múltiples plataformas, pero en esta sección nos enfocaremos en dos

especialmente útiles por su accesibilidad y funcionalidades: Figma y Marvel. Ambas permiten crear versiones navegables de una app o sitio web, simular interacciones y compartir los diseños con otras personas para recibir devoluciones. A continuación, exploraremos cómo funcionan y qué ventajas ofrece cada una a la hora de transformar una idea en una experiencia digital concreta.

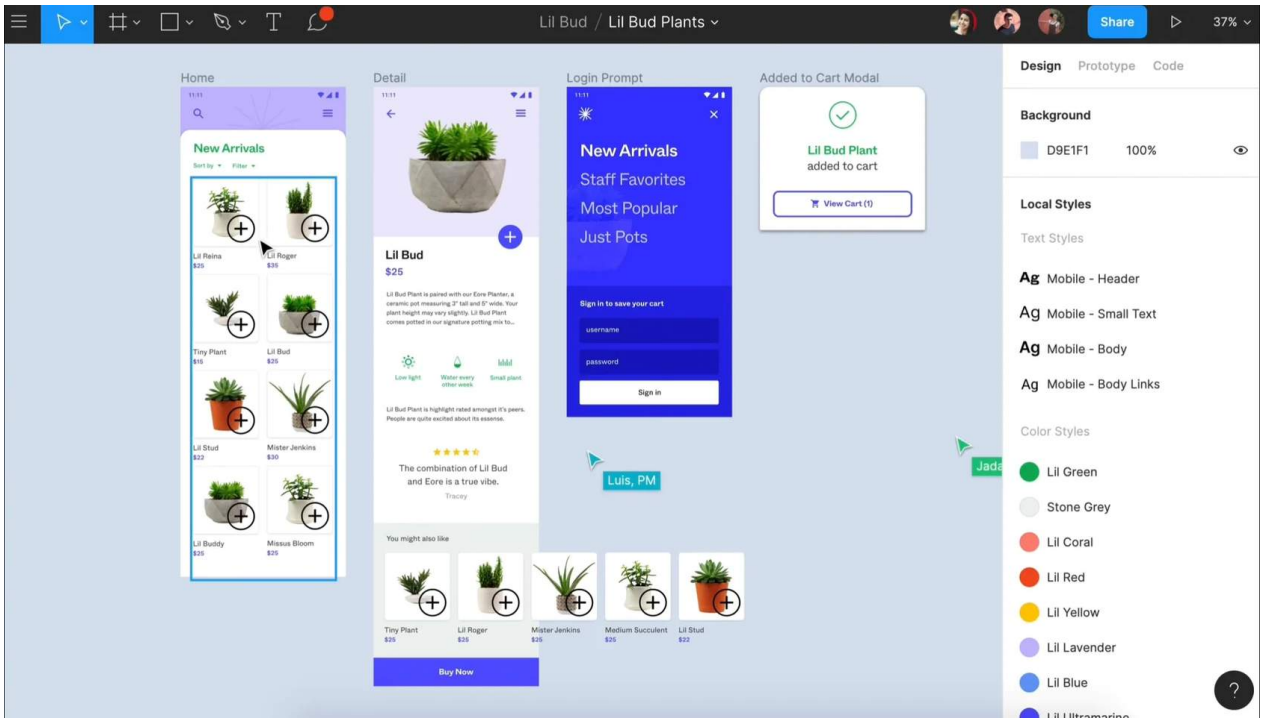
Figma: diseño colaborativo y prototipos interactivos

Figma es una herramienta digital que permite crear interfaces y prototipos directamente desde el navegador. Se utiliza para diseñar productos como aplicaciones móviles, plataformas web o sistemas interactivos, y visualizar cómo funcionarían antes de ser desarrollados. No requiere instalación y permite que varias personas trabajen en simultáneo sobre el mismo archivo.

Uno de sus usos más extendidos es la creación de prototipos de alta fidelidad, es decir, representaciones visuales detalladas y navegables de una solución digital. A diferencia de un dibujo estático, en este tipo de prototipos se puede hacer clic, cambiar de pantalla, completar formularios o simular una compra. Esto permite testear no solo la apariencia del producto, sino también su comportamiento.

En la siguiente imagen se muestra un ejemplo de prototipo creado en Figma: una tienda de plantas en la que el usuario puede navegar por el catálogo, ver el detalle de un producto, iniciar sesión o agregar ítems al carrito. Cada pantalla fue diseñada y vinculada de forma interactiva, como si se tratara de una aplicación real. Este tipo de simulación permite observar cómo sería la experiencia de uso de principio a fin.

Figura 11. Ejemplo de prototipo en Figma



Fuente: Bahaieva, 2020, https://surl.li/muucwk

Figma también permite trabajar de forma colaborativa. En proyectos donde participan diseñadores, programadores y otras personas del equipo, la posibilidad de dejar comentarios o hacer modificaciones en tiempo real ayuda a mantener el flujo de trabajo ordenado y

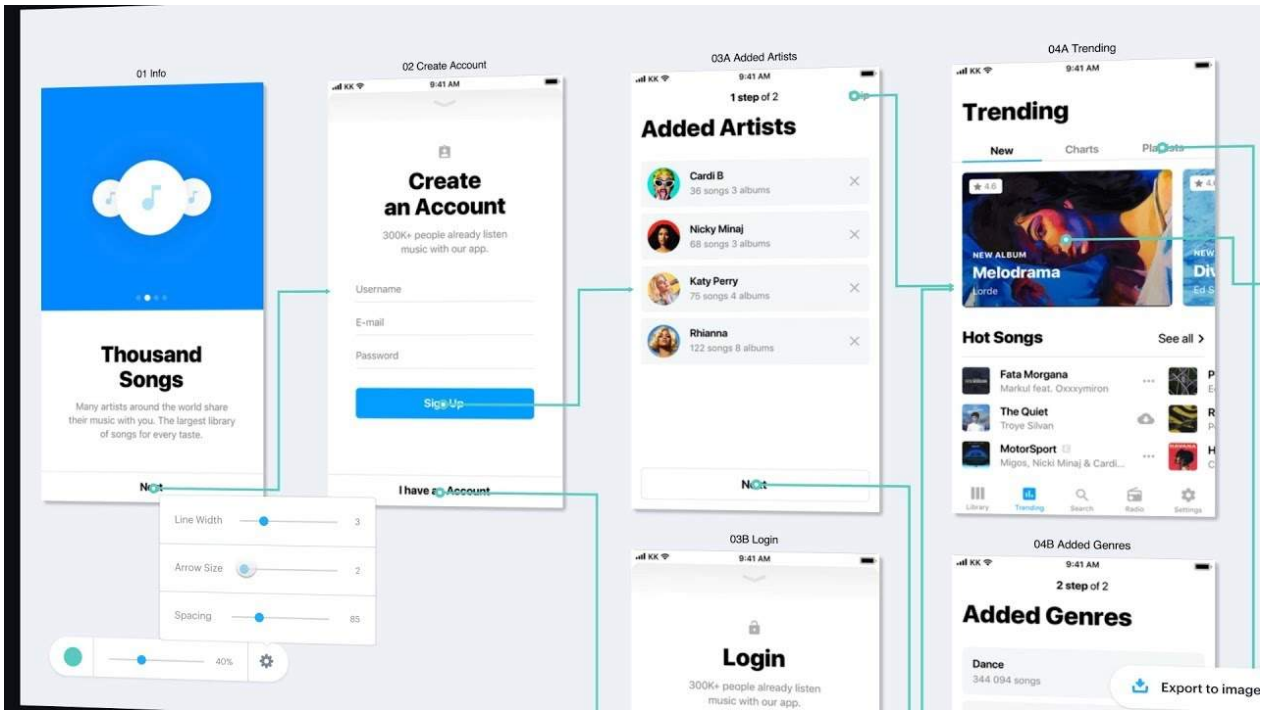
transparente. Además, todo queda centralizado en un único archivo, lo que facilita el seguimiento y la organización.

Marvel: prototipos interactivos simples y enfocados en la experiencia

Marvel es una herramienta digital que permite crear prototipos interactivos a partir de pantallas diseñadas previamente. A diferencia de otras plataformas más completas como Figma, Marvel se enfoca específicamente en el armado de flujos de navegación, permitiendo simular cómo se movería una persona usuaria dentro de una aplicación o sitio web.

En la imagen se observa un ejemplo de prototipo de una app de música, en el que se puede recorrer la interfaz desde la pantalla de bienvenida hasta la selección de artistas y géneros favoritos. Las flechas indican cómo están conectadas las diferentes pantallas, permitiendo simular la experiencia real de uso. Cada paso puede testearse como si la aplicación estuviera en funcionamiento.

Figura 12. Ejemplo de prototipo en Marvel



Fuente: GetApp, s.f., https://surl.li/verash

Una diferencia con Figma es que Marvel no es una herramienta de diseño visual. Es decir, no permite crear las pantallas desde cero dentro de la misma plataforma. En cambio, parte de archivos ya diseñados (por ejemplo, en Photoshop o Sketch), que luego pueden importarse para vincularse entre sí y simular la interacción.

Esto la convierte en una herramienta más simple y accesible para equipos que ya tienen definidas sus pantallas o *wireframes*. Marvel también permite añadir gestos táctiles (como scroll o clics) y animaciones básicas, lo que mejora la fidelidad de la simulación sin requerir conocimientos técnicos avanzados.

Además, su interfaz está pensada para testear flujos y recopilar comentarios de personas usuarias o miembros del equipo, facilitando el

proceso iterativo. Aunque no ofrece la misma flexibilidad para diseñar que Figma, su foco en la creación rápida de prototipos funcionales la convierte en una aliada útil en etapas de validación.

Hasta aquí vimos herramientas de prototipado digital enfocadas en aplicaciones o plataformas web, que son especialmente útiles cuando nuestro proyecto implica el diseño de experiencias interactivas. Sin embargo, si la solución que estamos desarrollando se relaciona con otros campos, existen herramientas específicas según el tipo de proyecto. Por ejemplo, si se trata de un espacio físico o algo vinculado a la arquitectura, se pueden utilizar maquetas tridimensionales, planos, *renders* o recorridos virtuales. Si la propuesta incluye productos físicos, puede recurrirse al uso de materiales reciclados, impresión 3D o modelado con arcilla, cartón u otros recursos. En todos los casos, lo importante es traducir la idea a un formato tangible y accesible, que permita probar, ajustar y compartir la propuesta de forma temprana.

CONTINUAR

Referencias

Activity Timeline, (2025). *Panel de control de Jira: Cómo usarlo y ejemplos.* https://activitytimeline.com/es/blog/jira-dashboard-all-you-need-to-know

Atlassian, (s.f.). *¿Qué es un tablero de kanban?* https://www.atlassian.com/es/agile/kanban/boards

Bahaieva, O. (2020). *Be Successful With Figma: Unlock New Design Level.* https://uxplanet.org/be-successful-with-figma-unlock-new-design-level-bcae09a02624

Coworking FY. (2019). *Metodología Waterfall | Cómo aplicar la gestión de proyectos en cascada*. https://coworkingfy.com/metodologia-waterfall/

Cruz, A. (2025). Lean Startup: cómo validar ideas antes de desarrollar un producto. *Enium*. https://www.eniun.com/lean-startup-como-validar-ideas-antes-desarrollar-producto/

GetApp, (s.f.). Sobre Marvel. https://www.getapp.com.co/software/113601/marvel

Interaction Desing Foundation, (s.f.). *Low-Fidelity Prototypes.* https://www.interaction-design.org/literature/topics/low-fidelity-prototypes

Karimi, M. (2017). *What Does It Mean When Your Technology Is an MVP?* https://jpt.spe.org/twa/what-oil-and-gas-start-ups-need-know-about-being-mvp

[Imagen sin título sobre planificación iterativa], (s.f.). https://www.slideteam.net/blog/los-10-mejores-modelos-iterativos-ejemplos-con-plantillas-y-ejemplos?lang=Spanish

User Focus, (s.f.). *Lean ways to test your new business idea.* https://www.userfocus.co.uk/articles/lean_ways_to_test_your_new_business_idea.html

CONTINUAR

Descarga en PDF