

Two thick, horizontal, slightly wavy bars. The top one is teal and the bottom one is yellow, both spanning the width of the page.

KeyCraft

Autores: Iván García Espaliu y Lucas Bernabé Fernández

Tutor: Javier Martín Rivero

Fecha de entrega: 2025/04/01

Convocatoria: 2024 2025

ÍNDICE:

- 1. MOTIVACIÓN**
- 2. ABSTRACT**
- 3. OBJETIVOS PROPUESTOS**
- 4. METODOLOGÍA UTILIZADA**
- 5. TECNOLOGÍA Y HERRAMIENTAS**
- 6. PLANIFICACIÓN**
- 7. REQUISITOS FUNCIONALES Y NO FUNCIONALES**
- 8. MOCKUP**
- 9. DIAGRAMA DE CLASES**
- 10. CASOS DE USO**
- 11. EXPLICACIÓN DEL CÓDIGO**
- 12. PRUEBAS DE CAJA NEGRA (JUNIT)**

1. Motivación

El mundo de los teclados mecánicos personalizados ha experimentado un crecimiento notable en los últimos años. Cada vez más usuarios buscan crear sus propios teclados a medida, eligiendo cada componente como: switches, placas PCB, carcasas, keycaps o estabilizadores; según sus preferencias estéticas y funcionales. Sin embargo, este proceso no es sencillo: muchos de estos componentes no son universales, y la compatibilidad entre ellos puede generar confusión, especialmente para quienes se inician en este hobby.

Actualmente, no existe una herramienta centralizada que permita a los usuarios seleccionar piezas y verificar de forma clara si son compatibles entre sí. Esto provoca errores de compra, pérdidas de tiempo y frustración. Por ello, esta aplicación surge como solución: ofrecer una plataforma donde los usuarios puedan explorar componentes, crear combinaciones personalizadas y asegurarse de que todo encaja antes de realizar cualquier adquisición.

Además de ayudar a principiantes, también busca optimizar la experiencia de los más experimentados, permitiendo organizar su colección de componentes y descubrir nuevas configuraciones de forma rápida y fiable.

2. Abstract

The world of custom mechanical keyboards is rapidly expanding, but many users still face difficulties finding compatible components. Not all parts—such as switches, PCBs, cases, and keycaps—follow universal standards, which often leads to confusion and incompatible purchases.

This application aims to address that problem by allowing users to select different keyboard components and verify their compatibility before buying. The system offers a user-friendly interface where users can explore, combine, and validate components based on real compatibility data.

The project has been developed using *JavaFX* for the graphical interface, *Hibernate* for data persistence, and *MySQL* as the relational database system. *IntelliJ IDEA* has been used as the primary development environment, with *GitHub* for version control. The interface design was prototyped in *Figma*, while *Canva* was used to create supporting visuals and documentation.

This tool is designed to assist both beginners and experienced keyboard enthusiasts, reducing the risk of errors and streamlining the building process.

Keywords: mechanical keyboards, compatibility, JavaFX, Hibernate, MySQL, user interface, custom keyboards

3. Objetivos Propuestos

Objetivo general:

Desarrollar una aplicación de escritorio utilizando JavaFX como interfaz gráfica y Hibernate como herramienta de persistencia, con el objetivo de facilitar a los usuarios la selección de diferentes componentes de teclados mecánicos personalizados, permitiendo verificar la compatibilidad entre las piezas antes de realizar una compra. Esta solución busca minimizar errores comunes en la elección de componentes no compatibles, optimizar el proceso de diseño personalizado de teclados y ofrecer una experiencia de usuario intuitiva y eficiente. El proyecto está orientado tanto a usuarios principiantes como a entusiastas experimentados, promoviendo un acceso más organizado, visual y práctico a la construcción de teclados mecánicos. Para ello, se integrarán herramientas y tecnologías modernas como MySQL para la gestión de datos, GitHub para el control de versiones, y se apoyará el desarrollo visual y documental con herramientas como Figma y Canva.

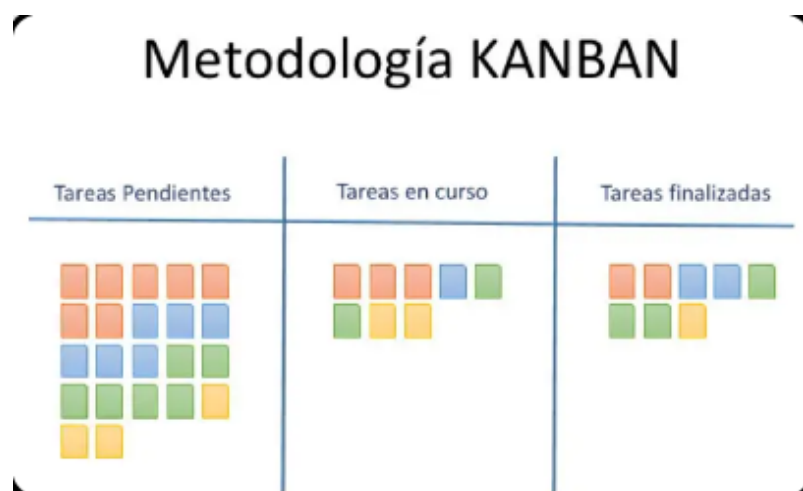
Objetivos específicos:

- Diseñar y gestionar una base de datos relacional mediante **Hibernate** y **MySQL**, capaz de almacenar de forma estructurada información detallada sobre los distintos componentes de teclados mecánicos, incluyendo sus características técnicas, relaciones de compatibilidad y precios. La base de datos deberá permitir consultas eficientes y relaciones claras entre los distintos tipos de piezas, posibilitando su uso para análisis de compatibilidad y futuras ampliaciones del sistema.
- Desarrollar un sistema de verificación lógica que analice las relaciones de compatibilidad entre los componentes seleccionados por el usuario, aplicando reglas definidas por el estándar técnico de cada pieza. Este sistema deberá identificar combinaciones no válidas y advertir al usuario antes de que proceda con la elección o adquisición de los elementos.

- Implementar un sistema de preselección o simulación de compra, que permita al usuario guardar configuraciones válidas de teclados personalizados, visualizar el conjunto de componentes compatibles seleccionados y obtener información útil antes de realizar la compra real a través de plataformas externas o integradas.

4. Metodología Utilizada

Para el desarrollo del presente proyecto se ha optado por utilizar la metodología **Kanban**, un enfoque ágil de gestión de proyectos que permite visualizar el flujo de trabajo, limitar el trabajo en curso y fomentar la mejora continua. Esta metodología ha sido especialmente adecuada para este proyecto por su flexibilidad y capacidad de adaptación a cambios frecuentes, lo cual es fundamental en el desarrollo de aplicaciones que requieren iteración constante sobre los requisitos y el diseño funcional.



5. Tecnología y Herramientas

A lo largo del desarrollo del proyecto se emplearán diversas herramientas, tanto de programación como de diseño y documentación, con el objetivo de optimizar el flujo de trabajo, asegurar una buena organización del código y mejorar la presentación final del aplicativo. A continuación, se detallan las más relevantes:

- **Java:** Para el desarrollo del proyecto, se ha elegido **Java** como lenguaje principal debido a su **versatilidad, estabilidad y amplia adopción en entornos profesionales**. Otra razón clave para su elección ha sido la compatibilidad con herramientas como **JavaFX**, que permite el desarrollo de interfaces gráficas.
- **JavaFX:** Para la creación de la interfaz gráfica de usuario, se ha optado por el uso de **JavaFX**, una tecnología moderna y robusta desarrollada específicamente para aplicaciones de escritorio en Java. Una de las principales ventajas de JavaFX es su capacidad para **separar el diseño de la lógica** mediante archivos FXML. Esta elección también garantiza una buena integración con otras tecnologías empleadas en el proyecto, como Hibernate y MySQL, dentro del ecosistema Java.
- **Hibernate:** Para la gestión de la persistencia de datos en el proyecto, se ha optado por el uso de **Hibernate**, un framework de mapeo objeto-relacional (ORM). Hibernate permite establecer una conexión directa entre las clases del modelo de dominio escritas en Java y las tablas de la base de datos, facilitando así el acceso, manipulación y gestión de los datos sin necesidad de escribir consultas SQL de forma manual.
- **IntelliJ IDEA:** Es el entorno de desarrollo integrado (IDE) principal, seleccionado por su compatibilidad con Java y sus potentes funciones de asistencia al desarrollo y la integración con sistemas de control de versiones.
- **GitHub:** Se utilizará como plataforma de control de versiones y colaboración. A través de repositorios remotos, se gestionará el historial de cambios del proyecto, asegurando trazabilidad, control y posibilidad de retroceso en caso de errores.
- **MySQL:** Es el sistema de gestión de bases de datos empleado para almacenar la información de los componentes, usuarios y relaciones de compatibilidad. Su robustez, rendimiento y facilidad de integración con Hibernate son

determinantes para su elección.

- **Figma:** Se utilizará para el diseño de prototipos de interfaz y la planificación visual de algunas funcionalidades (**Mockup**), lo que nos permite tener una visión clara y estructurada de la experiencia del usuario antes de pasar al desarrollo en JavaFX.
- **Canva:** Es empleada para la creación de materiales gráficos complementarios, como diagramas y presentaciones permitiendo mantener una coherencia estética y profesional en toda la documentación.

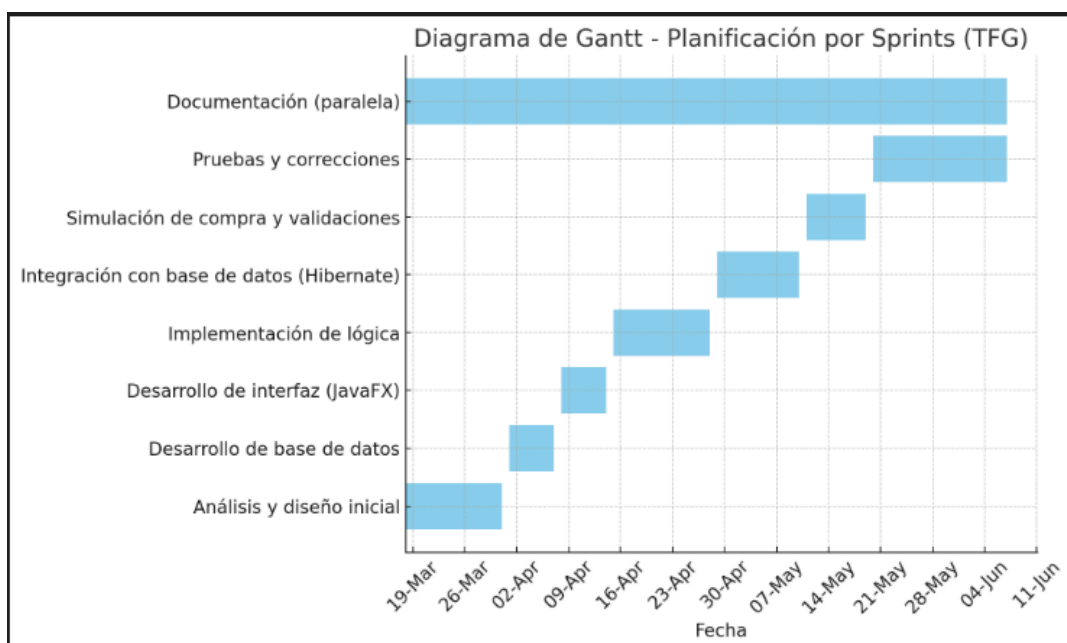
6. Planificación

La planificación del proyecto se ha estructurado mediante un **diagrama de Gantt**, donde se refleja la secuenciación temporal de las tareas clave desde **el 18 de marzo hasta el 7 de junio**. Esta organización sigue un enfoque ágil basado en **sprints**, permitiendo dividir el trabajo en fases manejables con entregables concretos por cada bloque de tiempo.

Durante el primer sprint (18-31 de marzo), se abordarán tareas de análisis, diseño preliminar de la base de datos y revisión de herramientas. A continuación, se dedicaron semanas específicas al desarrollo de cada parte fundamental del sistema: la base de datos (1-7 de abril), la interfaz gráfica (8-14 de abril), y la lógica interna de la aplicación (15-28 de abril).

En las fases posteriores (del 29 de abril al 19 de mayo), se centrará el trabajo en la integración con la base de datos utilizando Hibernate, así como en la implementación de la lógica de verificación de compatibilidad y simulación de compra. Finalmente, del 20 de mayo al 7 de junio, se reservó el tiempo para pruebas, correcciones y ajustes finales.

Paralelamente a todo el desarrollo, se fue elaborando la documentación del proyecto, asegurando coherencia entre lo implementado y lo redactado en la memoria.



7. Análisis del sistema

Antes de comenzar el desarrollo del proyecto, se ha llevado a cabo un análisis detallado del problema que se busca resolver. En este caso, se ha identificado que muchos usuarios interesados en el montaje de teclados mecánicos personalizados encuentran dificultades al intentar combinar distintos componentes, debido a la falta de estandarización entre marcas y modelos.

El objetivo principal del sistema es permitir a los usuarios seleccionar diferentes piezas (switches, PCB, carcasas, keycaps, etc.) y verificar automáticamente su compatibilidad. Para ello, ha sido necesario definir con precisión las entidades involucradas, las relaciones entre ellas, y los criterios de validación que rigen dicha compatibilidad.

Este análisis también ha permitido identificar los distintos tipos de usuarios que podrían interactuar con la aplicación, así como sus necesidades y expectativas. A partir de ello, se han definido los **requisitos funcionales** y **no funcionales** que guiarán el desarrollo y validación del sistema.

Requisitos funcionales:

Los requisitos funcionales describen las funcionalidades esenciales que el sistema debe ofrecer para cumplir su propósito. A continuación, se detallan los más relevantes:

1. El sistema debe permitir al usuario registrarse e iniciar sesión de forma segura.
2. El usuario podrá explorar un catálogo de componentes clasificados por tipo (switches, PCBs, keycaps, etc.).
3. El sistema debe permitir seleccionar distintos componentes para crear una configuración personalizada.
4. El sistema debe verificar automáticamente la compatibilidad entre los componentes seleccionados.
5. El usuario podrá guardar configuraciones compatibles como simulaciones de compra.
6. El sistema debe validar que no se pueda guardar una configuración incompatible.

Requisitos no funcionales:

Los requisitos no funcionales definen restricciones y cualidades del sistema que afectan su rendimiento, usabilidad, seguridad, entre otros aspectos clave. Los principales son:

1. La interfaz gráfica debe ser intuitiva, clara y accesible para usuarios con distintos niveles de experiencia.
2. El sistema debe estar desarrollado íntegramente en Java utilizando **JavaFx**, **Hibernate** y **MySQL**.
3. La respuesta del sistema ante la verificación de compatibilidad no debe superar los 2 segundos.
4. El código debe ser modular y fácil de mantener, con pruebas unitarias.

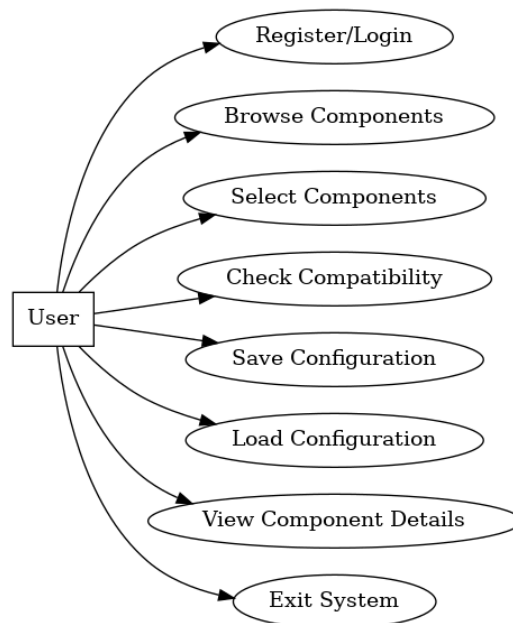
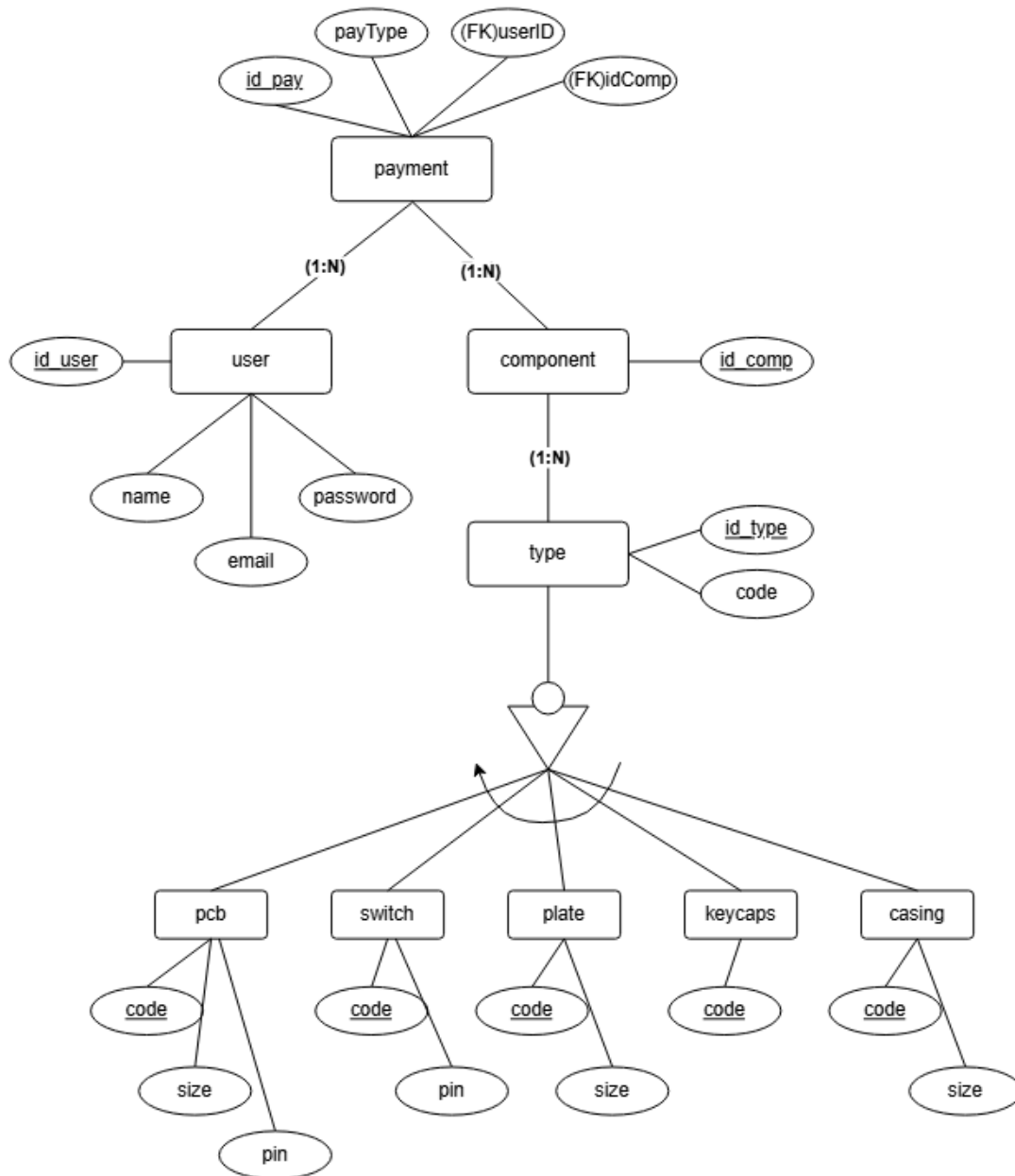
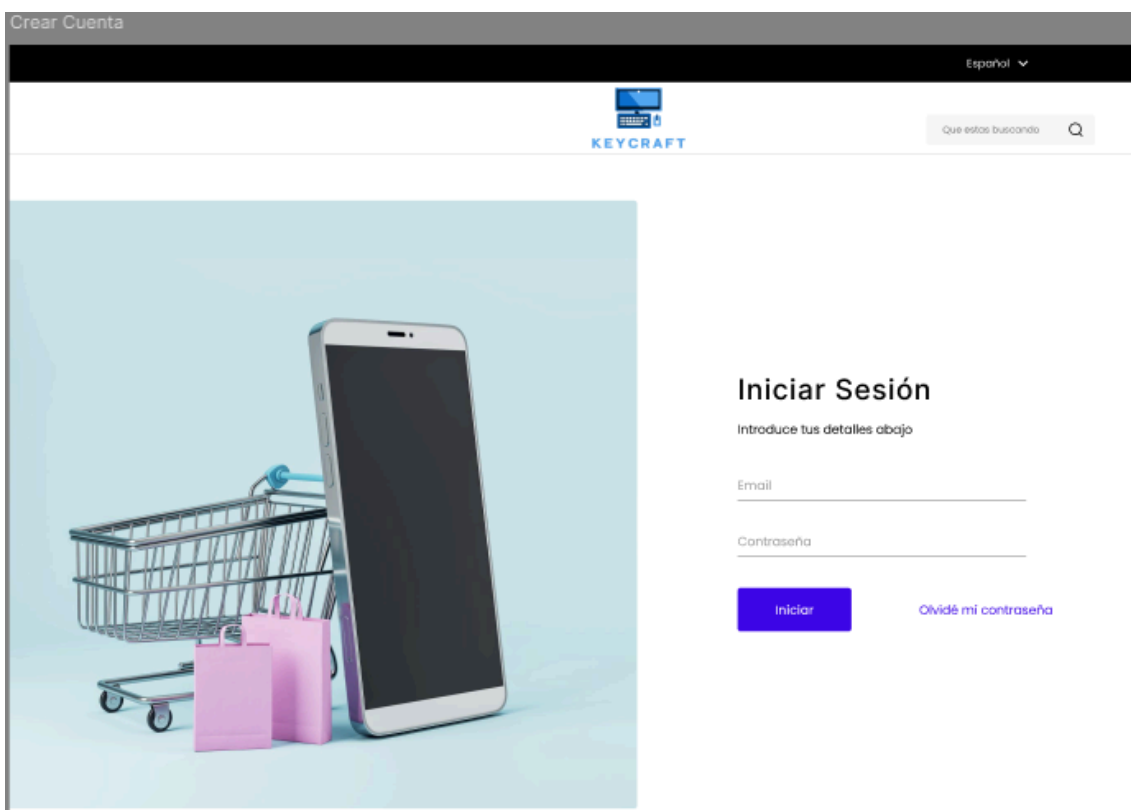
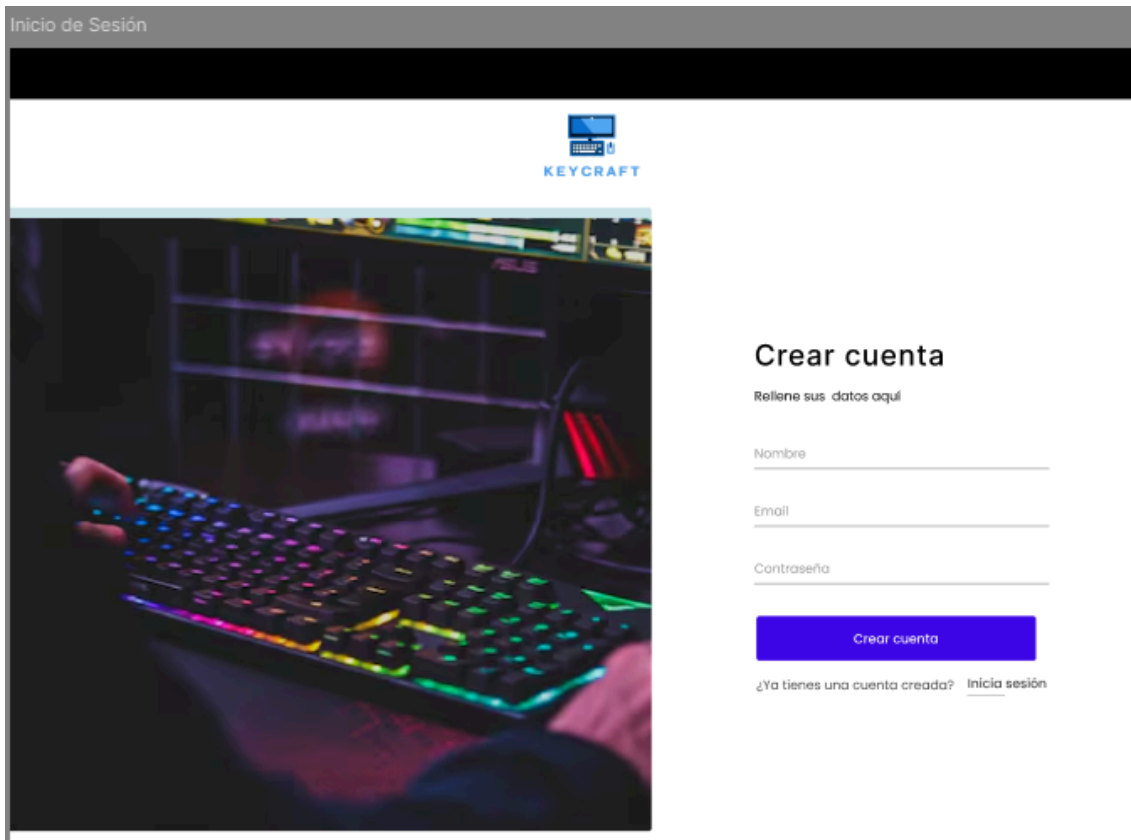


Diagrama Entidad-Relación.



8. Mockup



Página principal

Home

KEYCRAFT

Switches
Plates
Casing
PCB's
Keycaps

Los componentes mas actualizados del mercado

Comprar Ahora →

Switches

Image	Product Name	Price	Rating
	Alko V3 Yellow Cream	17,84€	★★★★★ (88)
	NEWSKILL Switches Gateron Optomecánicos	8,40€	★★★★★ (78)
	EPOMAKER Sea Salt Silent	20,99€	★★★★★ (99)
	ELUTENG Mechanical Switches	20,15€	★★★★★ (99)
	EPOMAKER Zebra	24,99€	★★★★★ (99)


Plates

Image	Product Name	Price	Rating
	Placa de latón Q6	31,50€	★★★★★ (88)
	Placa Q3 FR4	13,50€	★★★★★ (75)
	Placa de aluminio Q3	13,50€	★★★★★ (99)
	Placa de aluminio K2 Pro / K2 Max	15,00€	★★★★★ (99)
	S-Series Comfort Chair	35,00€	★★★★★ (99)

Casing

Image	Product Name	Price	Rating
	Casing de madera	89,99€	★★★★★ (88)
	Casing Aluminio 5075 VIA	72,59€	★★★★★ (75)
	Casing GMK104	58,99€	★★★★★ (99)
	Casing GMK87 con pantalla	51,99€	★★★★★ (99)
	Casing Tofu60 Redux Case	188,99€	★★★★★ (99)


PCB



DZ65 RGB V3 Hot-Swap RGB PCB

58,00€


★★★★★ (88)



Tofu60 2.0 PCB

45,00€


★★★★☆ (76)



Pluto PCB

49,00€

★★★★★ (99)




♥

👁

KBD6X MKIII PCB

30,00€

★★★★★ (99)




KBDfans TET PCB

33,55€

★★★★☆ (99)

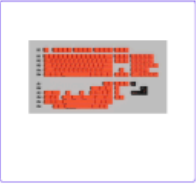
Keycaps



S9000 Cherry profile Keycaps

36,00€


★★★★★ (88)



GMK Flare

130,00€


★★★★☆ (75)



GMK GYL Modern Sono

100€

★★★★★ (98)




♥

👁

Oitsso Afternoon Soirée OSA profile keycaps

59€

★★★★★ (99)



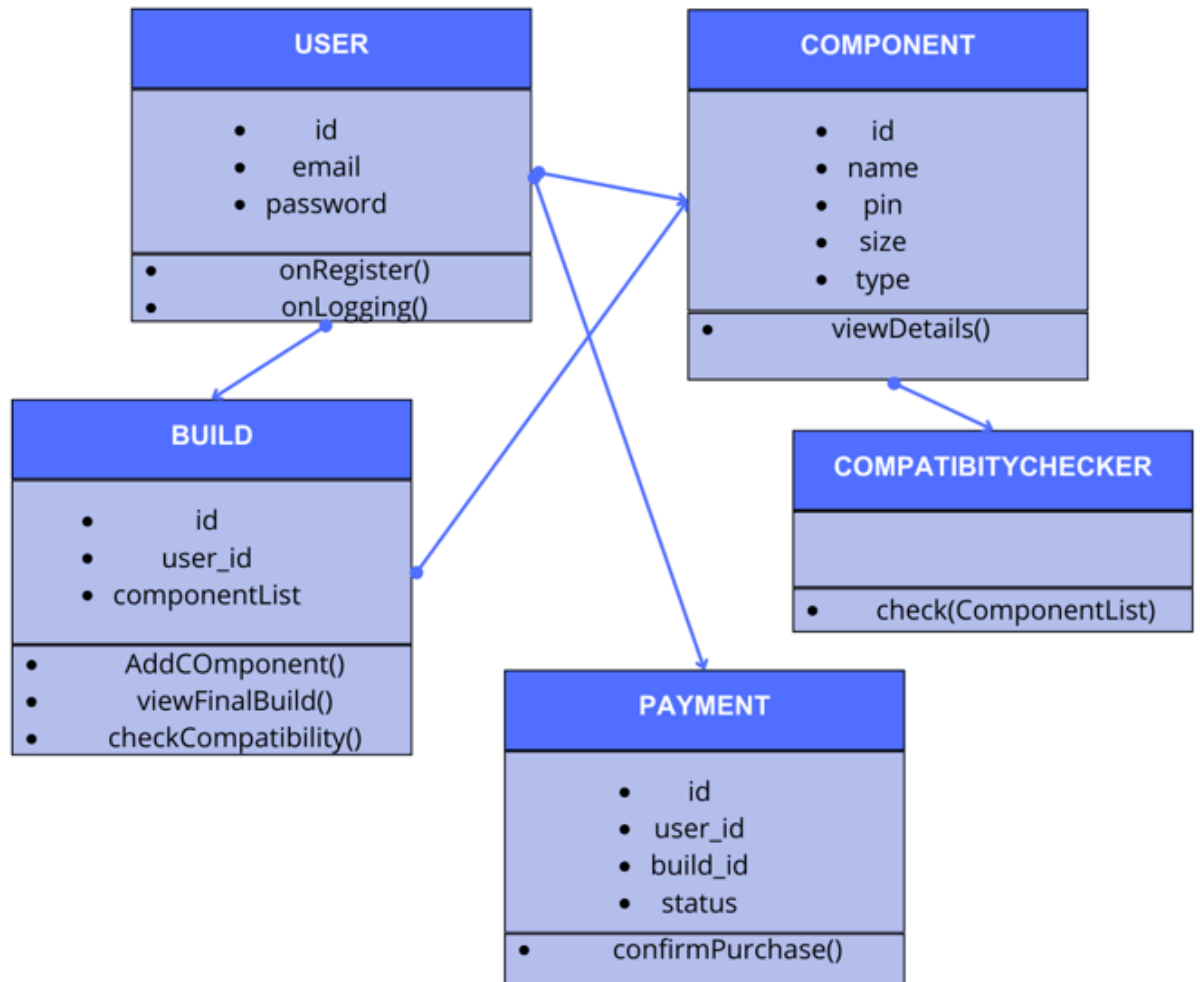
JTK Zen Cherry Profile keycaps

75€

★★★★★ (99)

13

9. Diagrama de clases



10. Casos de Uso.

