****

**1. INTRODUCCIÓN**

El presente proyecto consiste en el desarrollo de una aplicación de escritorio denominada **KeyBlock**, cuyo objetivo principal es gestionar de forma sencilla y visual una colección de notas personales. La aplicación ha sido desarrollada en lenguaje Java, se ha diseñado aplicando los principios básicos de la **Programación Orientada a Objetos (POO)** y haciendo uso de una base de datos **MySQL** para el almacenamiento persistente de la información.

KeyBlock ha sido desarrollado en el contexto de la asignatura de Programación, como actividad práctica para afianzar contenidos relacionados con interfaces gráficas, acceso a base de datos, gestión de usuarios y estructuración modular del código. Además de su funcionalidad base, el proyecto incorpora elementos visuales inspirados en entornos de desarrollo (modo oscuro, iconos, estilo limpio) y contempla distintos niveles de acceso, como el modo administrador.

El trabajo se ha desarrollado en equipo, dividiendo responsabilidades entre los miembros para fomentar la colaboración y el aprendizaje compartido. Este documento recoge el resultado del trabajo y tiene como objetivo explicar el funcionamiento general del programa, la organización del código, las decisiones de diseño adoptadas **y compartir la experiencia global del desarrollo.**

**2. MOTIVACIÓN / JUSTIFICACIÓN DE LA IDEA**

La idea de crear KeyBlock surge de una necesidad práctica: disponer de una aplicación de escritorio que permita **almacenar notas de forma segura, rápida y bien organizada**, en un entorno claro y a ser posible, agradable a la vista. Aunque existen múltiples soluciones similares, en este proyecto nos planteamos desde el principio crear una alternativa propia que, sin buscar competir directamente, pudiera **mejorar algunos aspectos** que echamos en falta como usuarios de este tipo de herramientas.

Durante la fase inicial del trabajo valoramos varias propuestas, pero esta fue la que más sentido tuvo para nosotros, ya que **utilizamos este tipo de apps en el día a día** y creíamos que podíamos desarrollar una versión ajustada a nuestras preferencias. Incluso más allá del contexto académico, veíamos interesante continuar el proyecto a futuro para seguir puliendo funcionalidades y explorando mejoras reales.

La propuesta encajaba además con los objetivos del módulo, ya que permitía aplicar lo aprendido sobre **interfaces gráficas, bases de datos, estructuras de control y programación orientada a objetos**, todo ello dentro de un proyecto con sentido completo. También se diseñó desde el inicio con dos objetivos clave: una gestión flexible mediante **hashtags** (ya implementada y funcional a nivel visual y de guardado), y una **zona protegida para contraseñas**, que se encuentra planificada como siguiente paso y servirá para explorar mecanismos de verificación de usuario dentro del propio entorno.

**3. OBJETIVOS PROPUESTOS**

El desarrollo de KeyBlock parte de una serie de objetivos claros tanto a nivel funcional como técnico. Estos objetivos han guiado la planificación y la implementación del proyecto desde su inicio:

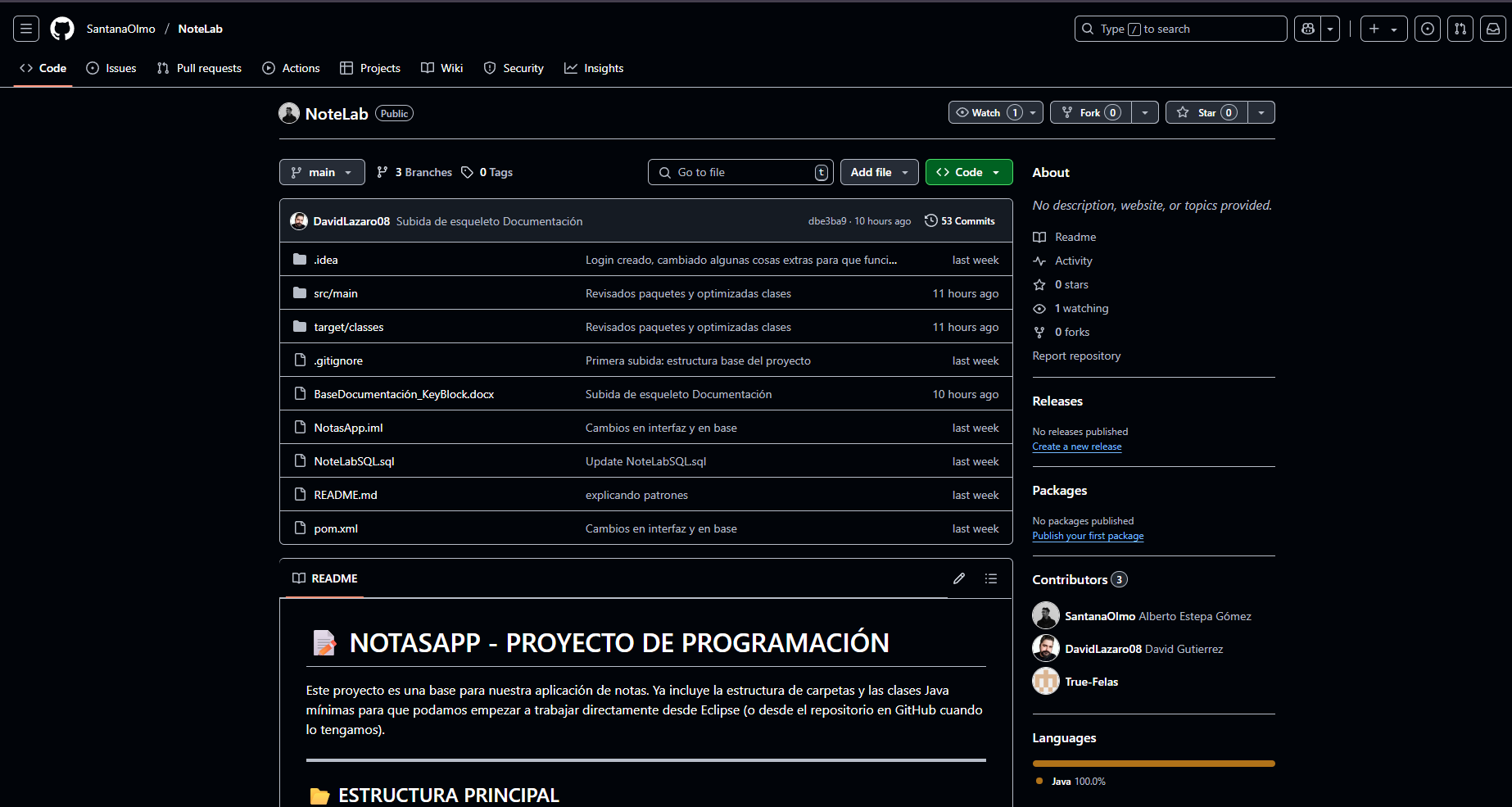
* Diseñar una interfaz gráfica de usuario con estilo moderno, minimalista y modo oscuro, inspirada en los entornos de desarrollo actuales.
* Permitir el registro e inicio de sesión de usuarios, diferenciando entre usuarios normales y administradores mediante roles.
* Ofrecer funcionalidades básicas de gestión de notas: creación, visualización y eliminación.
* Incorporar un sistema de etiquetas o hashtags, asignables a cada nota, los cuales se muestran resaltados con colores aleatorios.
* Incluir un panel de administración con acceso restringido solo a usuarios con perfil de administrador.
* Integrar una sección para almacenar contraseñas de forma segura.

**4. METODOLOGÍA UTILIZADA**

El desarrollo de la app se ha llevado a cabo de forma totalmente colaborativa entre los integrantes del equipo, combinando momentos de trabajo conjunto con fases de reparto de tareas más específicas. Aunque existió una lógica división de tareas según fortalezas o tipo de contenido, la idea siempre fue mantener una implicación conjunta en las decisiones y revisar entre todos el trabajo que se iba completando.

A nivel de organización, se adoptó una filosofía de responsabilidad única por clase, dividiendo el proyecto en paquetes bien diferenciados: vista, controlador, modelo y bbdd, siguiendo así lo que entendimos como buenas prácticas de modularidad y separación de responsabilidades. Esta estructura nos ha permitido avanzar paso a paso, sabiendo qué hacía cada parte del programa y pudiendo trabajar en paralelo sin grandes conflictos.

Además de utilizar herramientas como GitHub para el control de versiones y la sincronización del trabajo en equipo, se realizaron pruebas continuas, ajustes visuales, y pequeñas reuniones para coordinar el enfoque general de la app. Aunque es un proyecto formativo, hemos intentado trabajar con orden y con una idea clara de cómo debía funcionar y organizarse por dentro.



**5. DIAGRAMA DE GANTT**



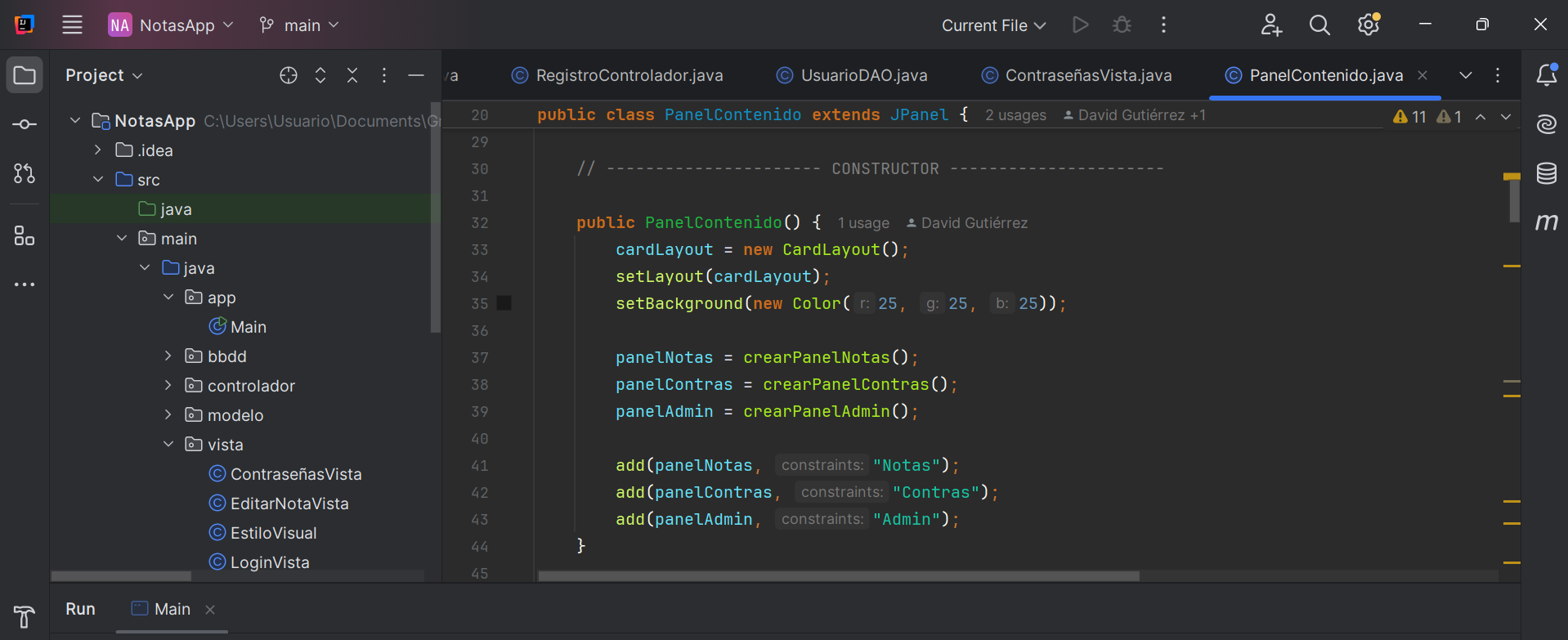
**El gráfico refleja cómo ha sido realmente nuestro proceso de trabajo.** Aunque al principio intentamos seguir una planificación aproximada por semanas, muchas tareas se fueron solapando o ajustando sobre la marcha según el ritmo y la disponibilidad de cada uno. Algunas fases se alargaron más de lo previsto y otras se resolvieron más rápido, pero al final pudimos organizarnos bien y llegar a tiempo con todo lo esencial terminado.

**6. TECNOLOGÍAS Y HERRAMIENTAS UTILIZADAS**

Durante el desarrollo de KeyBlock hemos utilizado un conjunto de herramientas y tecnologías que nos han facilitado tanto la construcción de la aplicación como la organización del trabajo. Algunas nos han servido para la parte visual, otras para conectar con la base de datos y otras para coordinarnos como equipo. A continuación, detallamos las principales:

* Java (JDK 17+): lenguaje principal de programación del proyecto.
* Swing: biblioteca de Java para la creación de interfaces gráficas (GUI), utilizada para diseñar las distintas ventanas de la aplicación.
* MySQL: sistema gestor de bases de datos empleado para almacenar la información de usuarios, notas y hashtags.
* JDBC (Java Database Connectivity): interfaz de conexión entre la aplicación Java y la base de datos MySQL.
* IntelliJ IDEA: entorno de desarrollo utilizado durante todo el proceso.
* XAMPP: paquete que incluye el servidor MySQL local utilizado durante el desarrollo.
* Git y GitHub: sistema de control de versiones utilizado para organizar el trabajo en equipo, hacer seguimiento de los cambios y mantener copias sincronizadas del proyecto.

Todas estas herramientas han sido claves para cumplir con los requisitos técnicos del proyecto y facilitar tanto el desarrollo individual como la colaboración entre compañeros.



**7. ANÁLISIS**

***REQUISITOS FUNCIONALES***

Durante el proceso se identificaron los siguientes requisitos funcionales, es decir, las acciones concretas que la aplicación debía permitir realizar al usuario:

* *RF1: Permitir el registro de nuevos usuarios y el inicio de sesión.*
* *RF2: Crear, visualizar y eliminar notas personales.*
* *RF3: Detectar hashtags escritos en una nota y aplicar color automáticamente.*
* *RF4: Restringir el acceso al panel de administración solo a usuarios con rol admin.*
* *RF5: Añadir un sistema de gestión de contraseñas.*

***REQUISITOS NO FUNCIONALES***

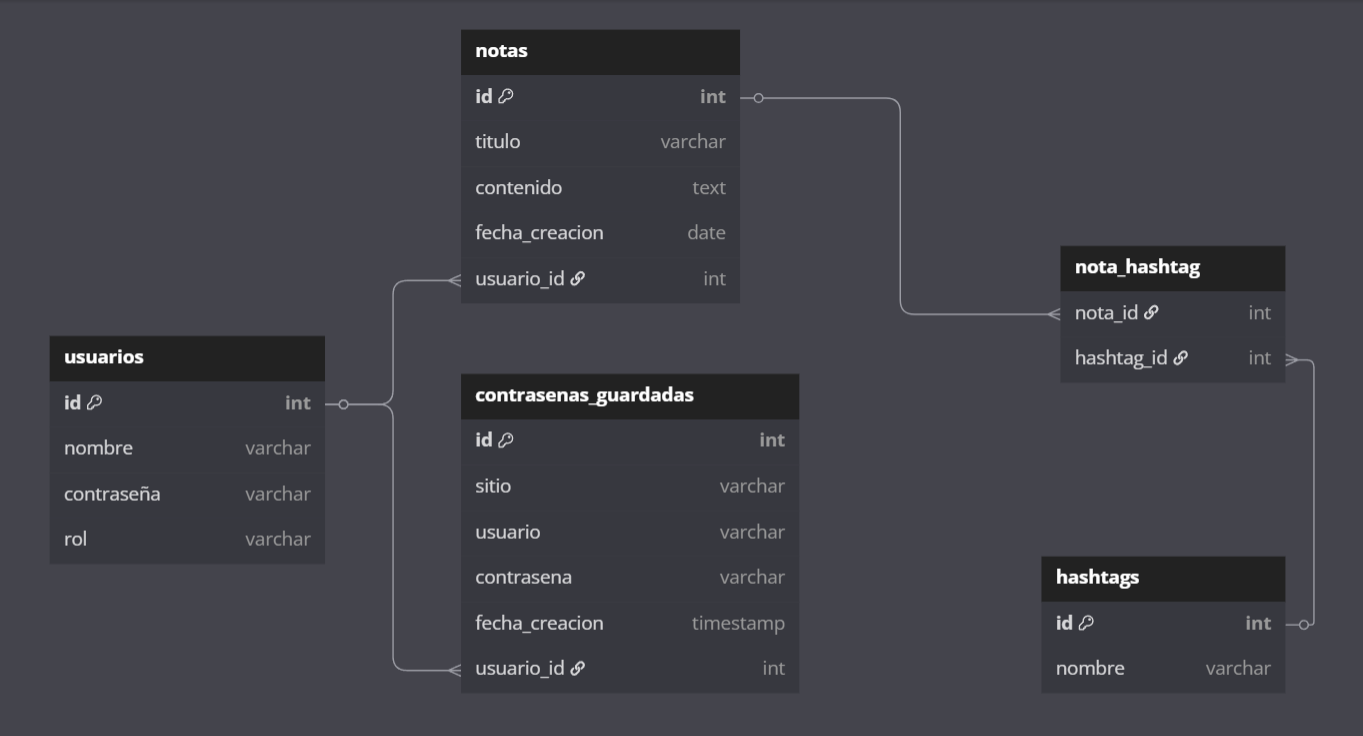
También se tuvieron en cuenta requisitos no funcionales, o esas características *relacionadas con la experiencia, la organización del código y la tecnología empleada:*

* *RNF1: La interfaz debía tener un diseño oscuro, limpio y agradable asemejándose a un Entorno de Desarrollo.*
* *RNF2: Toda la información debía almacenarse de forma persistente en una base de datos MySQL.*
* *RNF3: El código debía estructurarse de manera modular, aplicando buenas prácticas como la separación por paquetes (modelo, vista, controlador, etc.).*

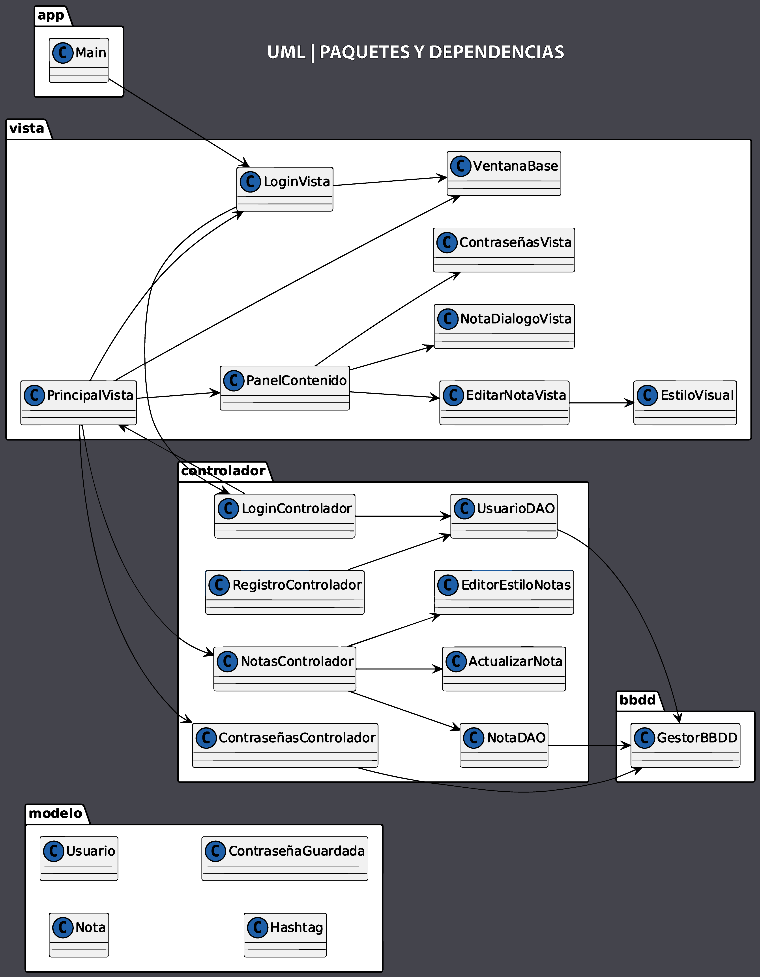
Una vez definidos los requisitos funcionales y no funcionales, se puede representar gráficamente la estructura de la base de datos que permite gestionar la información relacionada con usuarios, notas, hashtags y contraseñas. A continuación, se muestra el modelo entidad-relación utilizado en este proyecto.

***DIAGRAMA ENTIDAD-RELACIÓN***

Diagrama entidad-relación que representa la estructura de la base de datos utilizada.



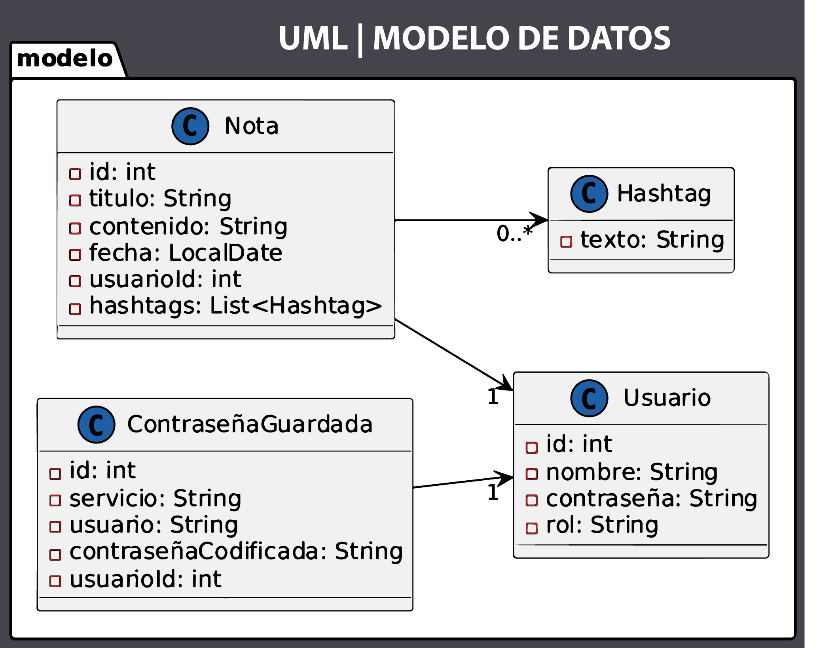
***DIAGRAMA DE CLASES***

******

Para representar de forma clara la estructura interna del proyecto, se ha optado por separar la visualización en dos diagramas:

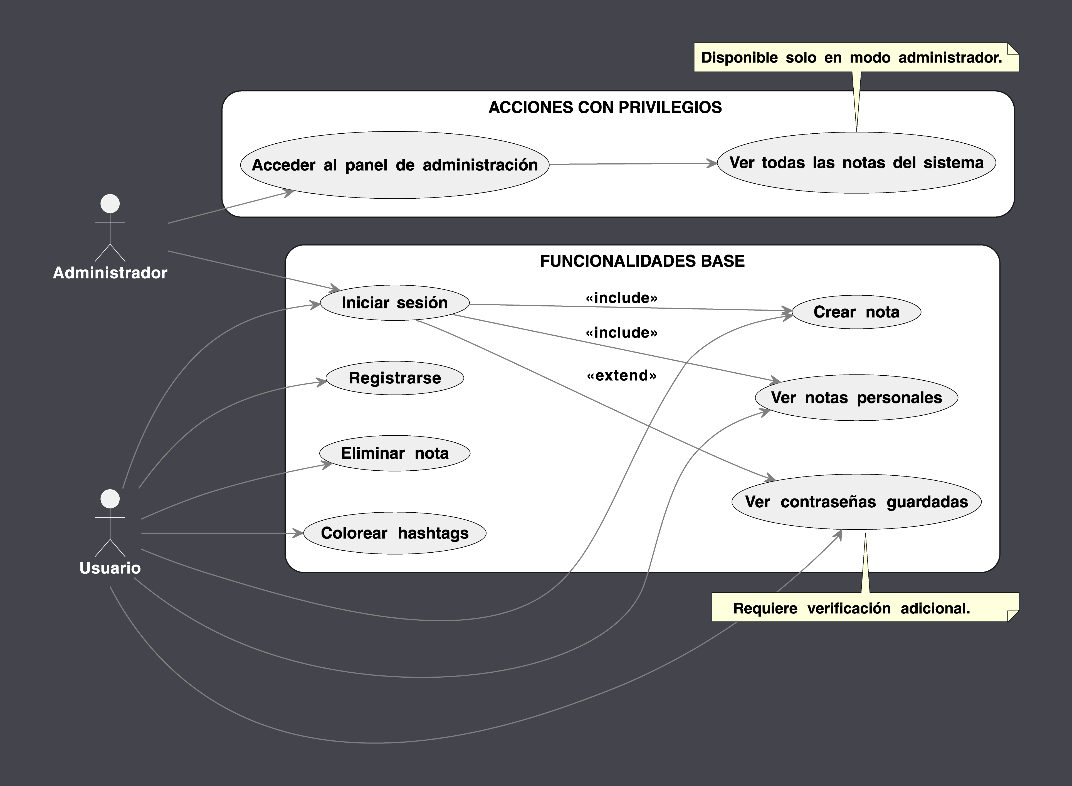
***Diagrama de paquetes y dependencias:*** muestra cómo se organiza el proyecto en paquetes y las relaciones de dependencia entre las clases.

***Diagrama del modelo de datos:*** centrado en las clases que representan la lógica de datos (modelo), detallando sus atributos y relaciones.

******

***CASOS DE USO***

Aquí mostramos las funcionalidades clave de KeyBlock, diferenciando entre las acciones disponibles para cualquier usuario (como crear, eliminar o visualizar sus notas) y aquellas exclusivas del administrador, como acceder al panel de control o ver todas las notas del sistema. Algunas funciones, requieren verificación adicional.

******

**8. DISEÑO**

**Mockups iniciales**

(Incluir bocetos de interfaz - si no existen, se harán basados en capturas actuales)

**Resultado final (capturas)**

* Pantalla de login
* Registro
* Vista de notas
* Panel de administración
* Cuadro de edición de nota

**9. EXPLICACIÓN DEL CÓDIGO DESTACABLE**

* **LoginControlador**: valida credenciales y gestiona el acceso.
* **PanelContenido**: panel con CardLayout que centraliza todas las vistas.
* **ActualizarNota**: sincroniza los campos del editor con una instancia de Nota.
* **EditorEstiloNotas**: aplica colores aleatorios a los hashtags en tiempo real.
* **NotaDAO**: maneja el guardado y borrado de notas y sus relaciones.

**10. CONCLUSIÓN**

El proyecto ha sido una experiencia muy completa donde se han puesto en práctica conceptos clave de Programación Orientada a Objetos, diseño modular y acceso a bases de datos. Hemos aprendido a organizarnos mejor como equipo, estructurar un proyecto desde cero y afrontar errores técnicos de forma colaborativa.

La profesora ha sido comprensiva y exigente a partes iguales, y consideramos que esta actividad ha sido una buena forma de cerrar el curso.

**11. PREPARACIÓN PARA LA EXPOSICIÓN**

Cada miembro conoce el funcionamiento de su parte y hemos repasado el código y la estructura general. Se están preparando capturas y documentos para facilitar la defensa del proyecto.