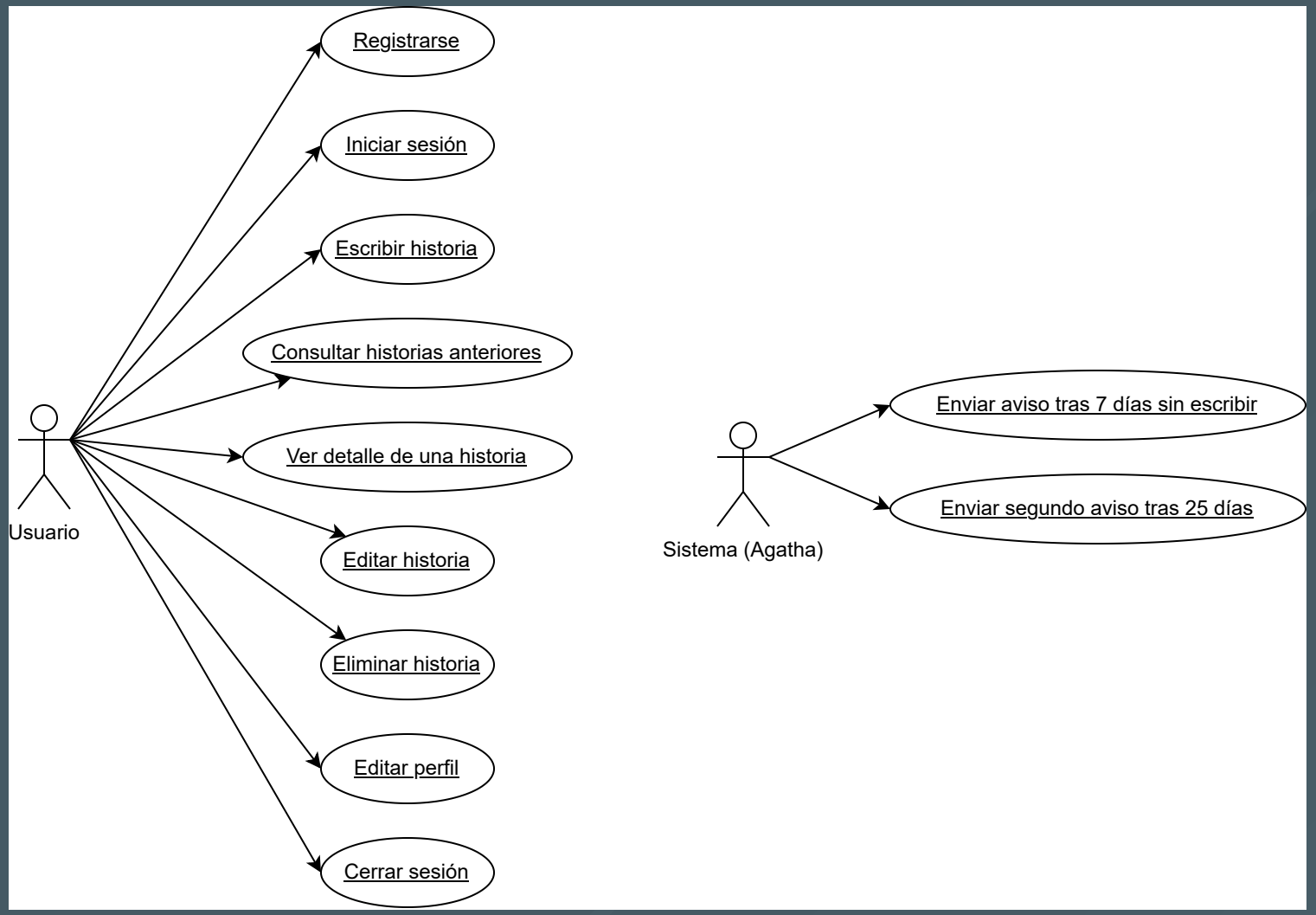
3. Análisis

En este capítulo se recogen los elementos clave que definen cómo debe funcionar la aplicación Agatha a nivel funcional y estructural. Incluye los casos de uso principales, el diagrama entidad-relación lógico de la base de datos y varios diagramas que ayudan a entender los flujos internos, como el sistema automático de avisos por inactividad.

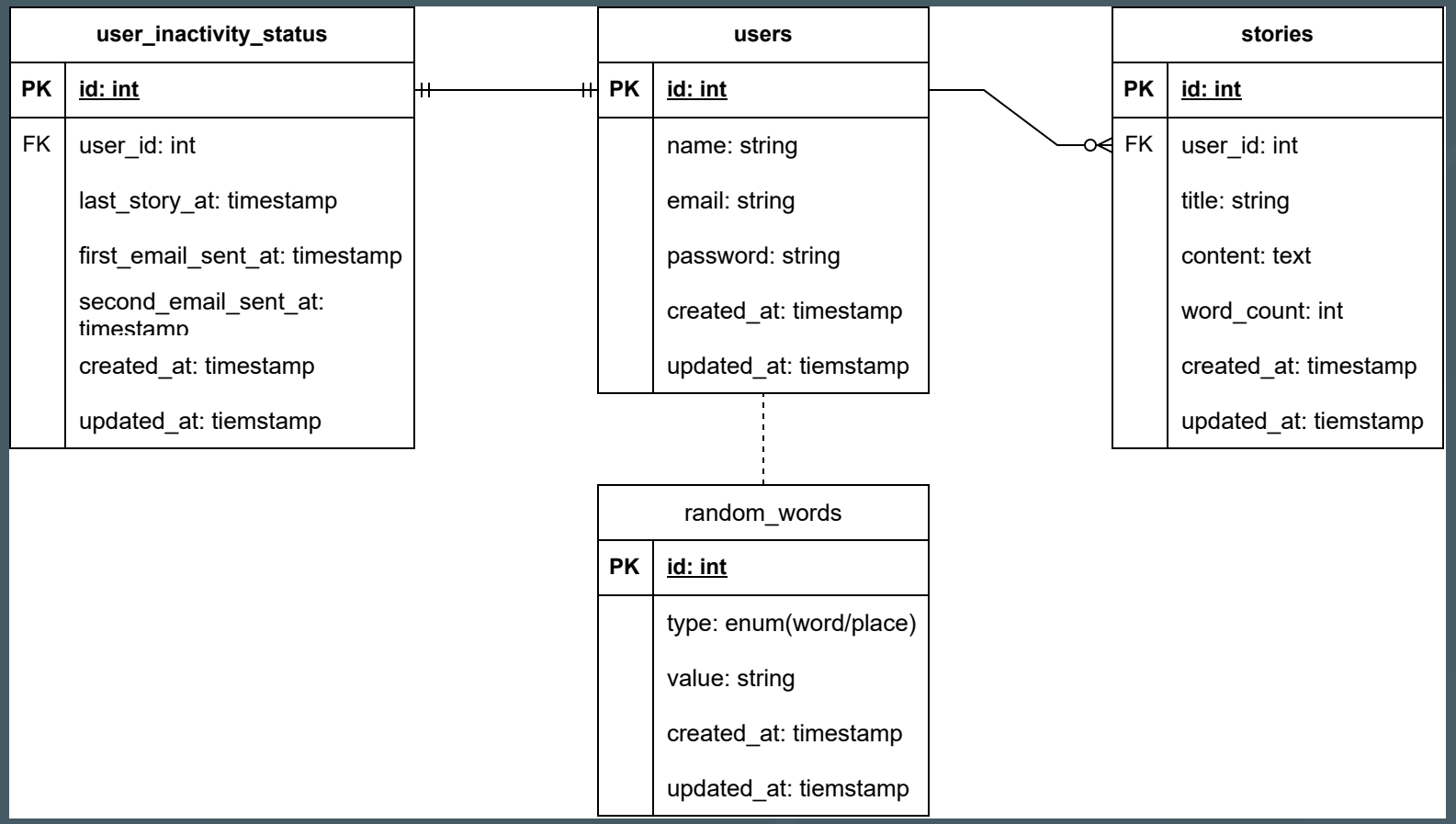
3.1. Casos de Uso

A continuación se representan los casos de uso básicos del sistema, centrándonos en la interacción entre el usuario y la plataforma.


Registrarse
Iniciar sesión
Escribir historia
Consultar historias anteriores
Ver detalle de una historia
Editar historia
Eliminar historia
Editar perfil
Cerrar sesión
Usuario
Enviar aviso tras 7 días sin escribir
Enviar segundo aviso tras 25 días
Sistema (Agatha)

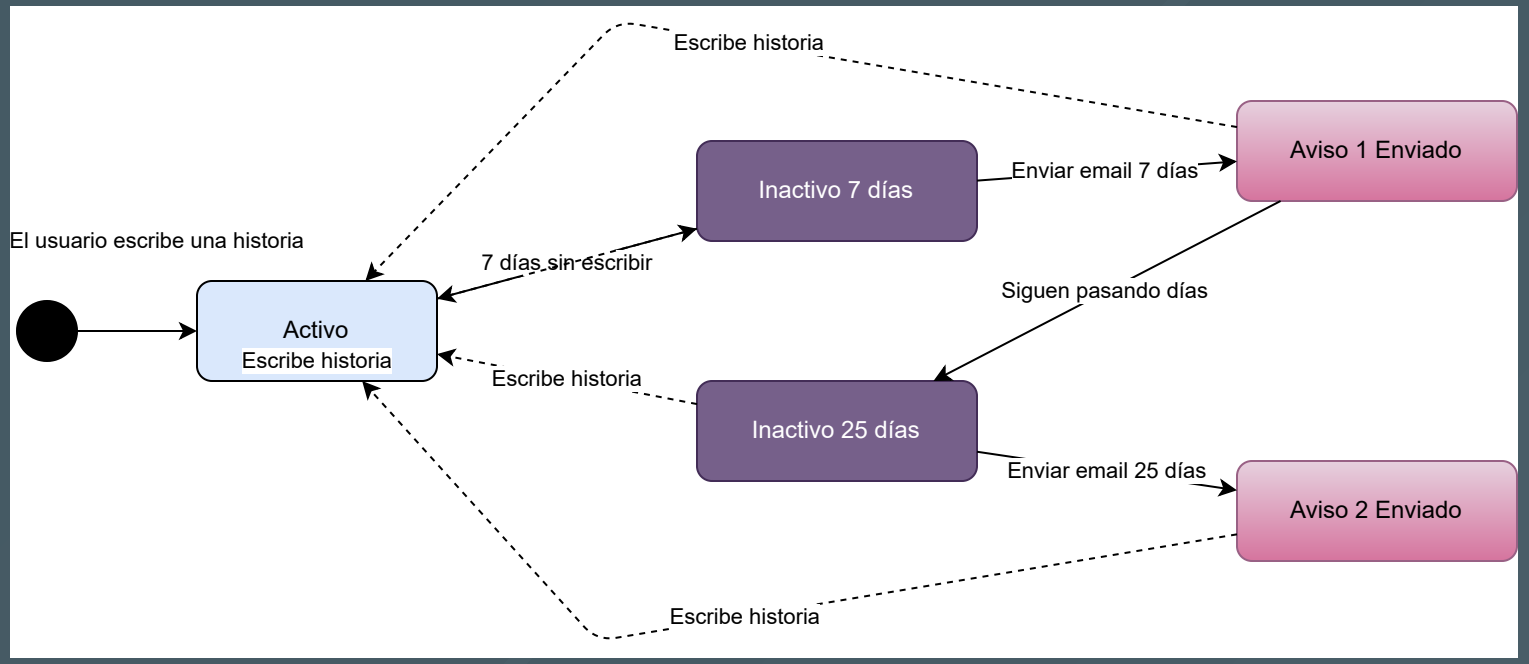
3.2. Diagrama Entidad-Relación (Versión Lógica)

Este diagrama representa la estructura principal de la base de datos utilizada en Agatha. Está adaptado al modelo real del proyecto.


user_inactivity_status
PK
id: int
FK
user_id: int
last_story_at: timestamp
first_email_sent_at: timestamp
second_email_sent_at: timestamp
created_at: timestamp
updated_at: tiemstamp
users
PK
id: int
name: string
email: string
password: string
created_at: timestamp
updated_at: tiemstamp
stories
PK
id: int
FK
user_id: int
title: string
content: text
word_count: int
created_at: timestamp
updated_at: tiemstamp
random_words
PK
id: int
type: enum(word/place)
value: string
created_at: timestamp
updated_at: tiemstamp

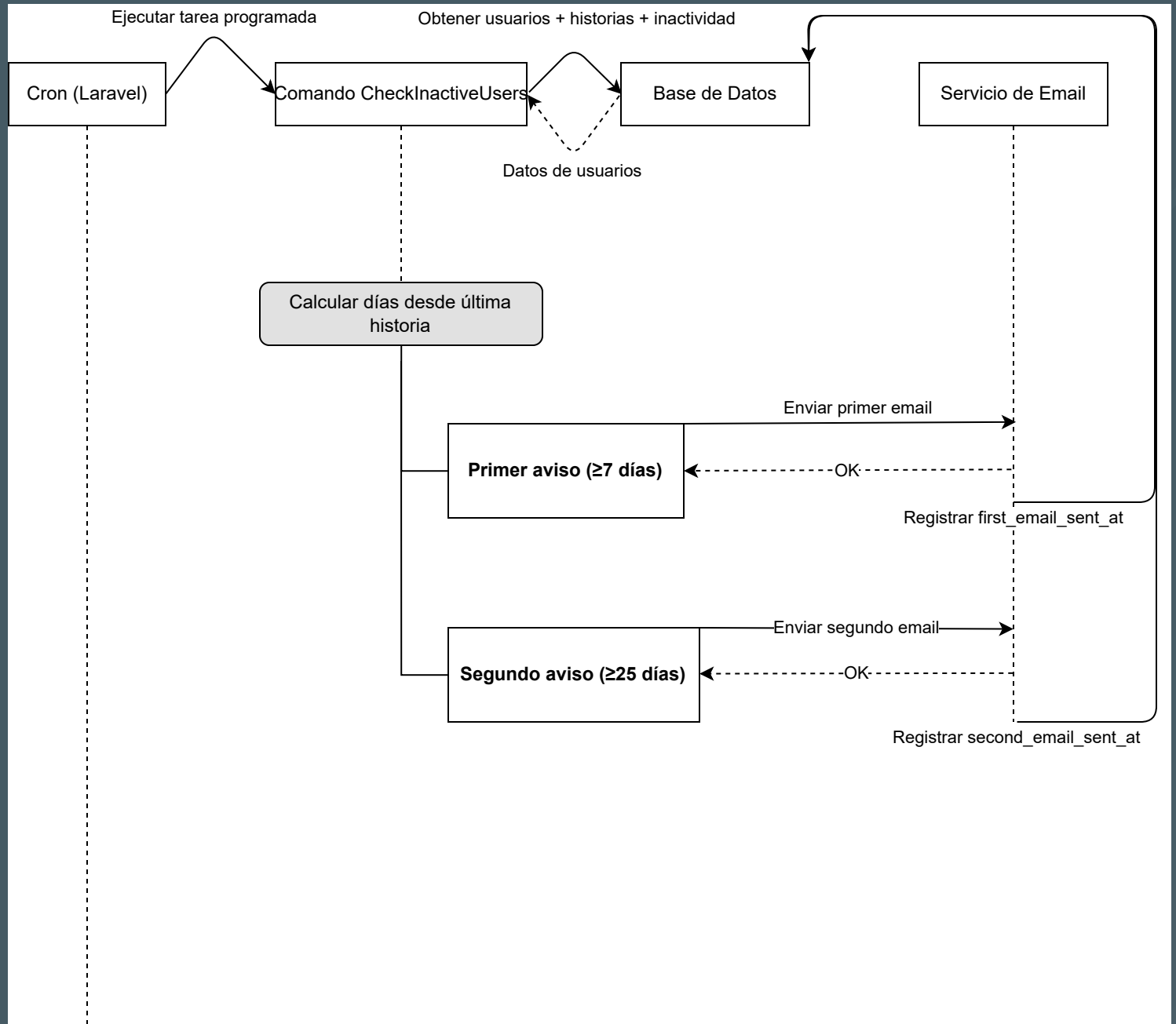
3.3. Diagrama de Estados (Actividad del usuario)

Este diagrama muestra cómo evoluciona el estado de un usuario dentro del sistema en función de su actividad al escribir historias. Es especialmente ilustrativo para entender el sistema automático de correos.



3.4. Diagrama de Secuencia (Proceso de envío automático)

Este diagrama explica cómo se ejecuta el comando users:check-inactive y cómo se envían los correos.


Ejecutar tarea programada
Obtener usuarios + historias + inactividad
Cron (Laravel)
Comando CheckInactiveUsers
Base de Datos
Servicio de Email
Datos de usuarios
Calcular días desde última historia
Enviar primer email
Primer aviso (≥7 días)
OK
Registrar first_email_sent_at
Enviar segundo email
Segundo aviso (≥25 días)
OK
Registrar second_email_sent_at

3.5. Requisitos del Proyecto

Requisitos funcionales:

- El usuario debe poder registrarse, iniciar y cerrar sesión.
- El usuario puede escribir nuevas historias.
- Puede consultar la lista de historias escritas.
- Puede ver el detalle de una historia.
- Puede editar o eliminar historias.
- El sistema debe enviar un email tras 7 días sin actividad.
- El sistema debe enviar un segundo email tras 25 días sin actividad.

- Los correos deben enviarse solo una vez por cada periodo.

Requisitos no funcionales:

- La API debe seguir el estándar REST.
- La base de datos debe garantizar integridad y relaciones.
- La aplicación debe ser segura mediante tokens Sanctum.
- Debe ser desplegable tanto en entorno local como remoto.
- Los tiempos de respuesta deben ser inferiores a 200ms en operaciones básicas.

3.6 Tabla Resumen de Endpoints de la API

Categoría	Endpoint	Método	Auth	Descripción	Campos
Autenticación	`/register`	POST	No	Registro de usuario	name, email, password, password_confirmation
Autenticación	`/login`	POST	No	Inicio de sesión. Devuelve token Sanctum	email, password
Autenticación	`/logout`	POST	Sí	Cierra sesión y elimina todos los tokens del usuario	—
Usuario	`/me`	GET	Sí	Datos del usuario autenticado	—
Usuario	`/user`	PUT	Sí	Actualiza el perfil del usuario	name?, email?, password?, password_confirmation?
Historias	`/story/random`	GET	Sí	Devuelve palabra y lugar aleatorios	—
Historias		GET	Sí	Lista todas las historias del	

3.7. Configuración Base

URL base de la API:

`http://agatha-api.test/api`

Formato de datos:

JSON (tanto para peticiones como para respuestas)

Autenticación:

Bearer Token obligatorio para todos los endpoints protegidos (se envía en la cabecera `Authorization: Bearer <token>`)

Duración del token:

12 horas (generado mediante Laravel Sanctum)