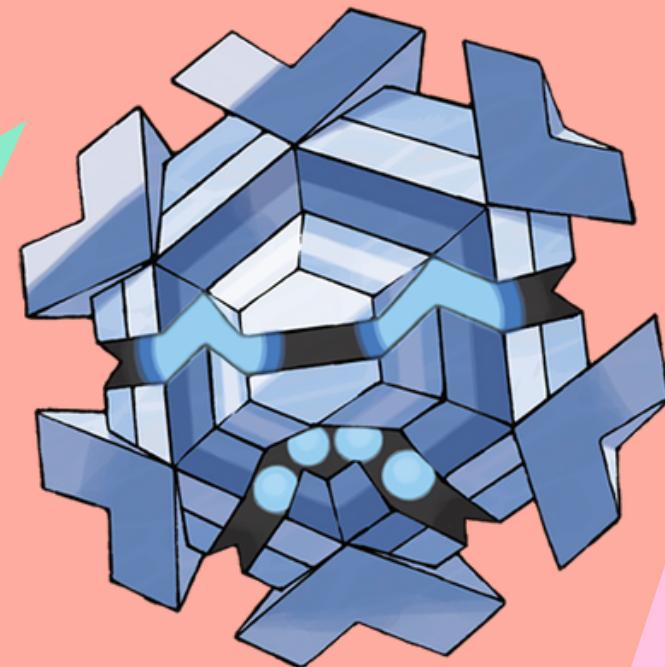
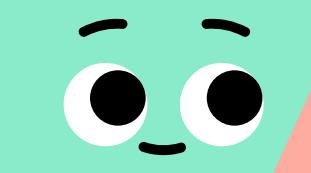
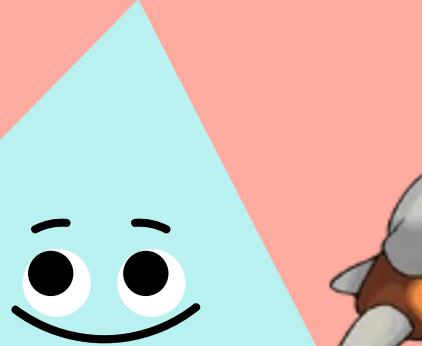
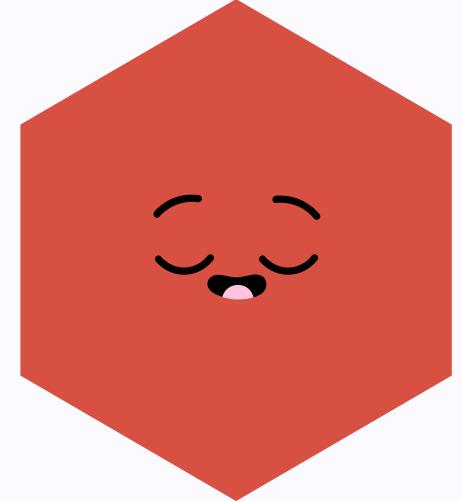


ARQUITECTURA HEXAGONAL

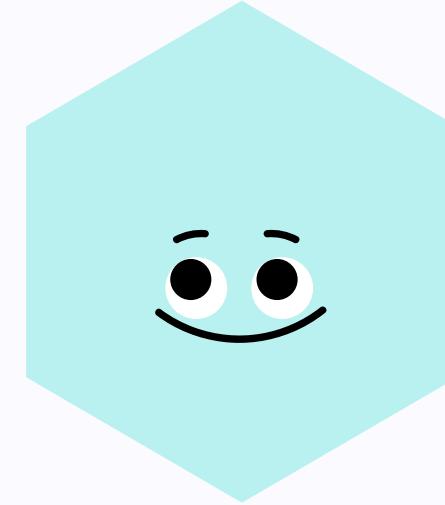
Grupo 2: Andrés García, Sophia Aristizábal y Juan Gómez



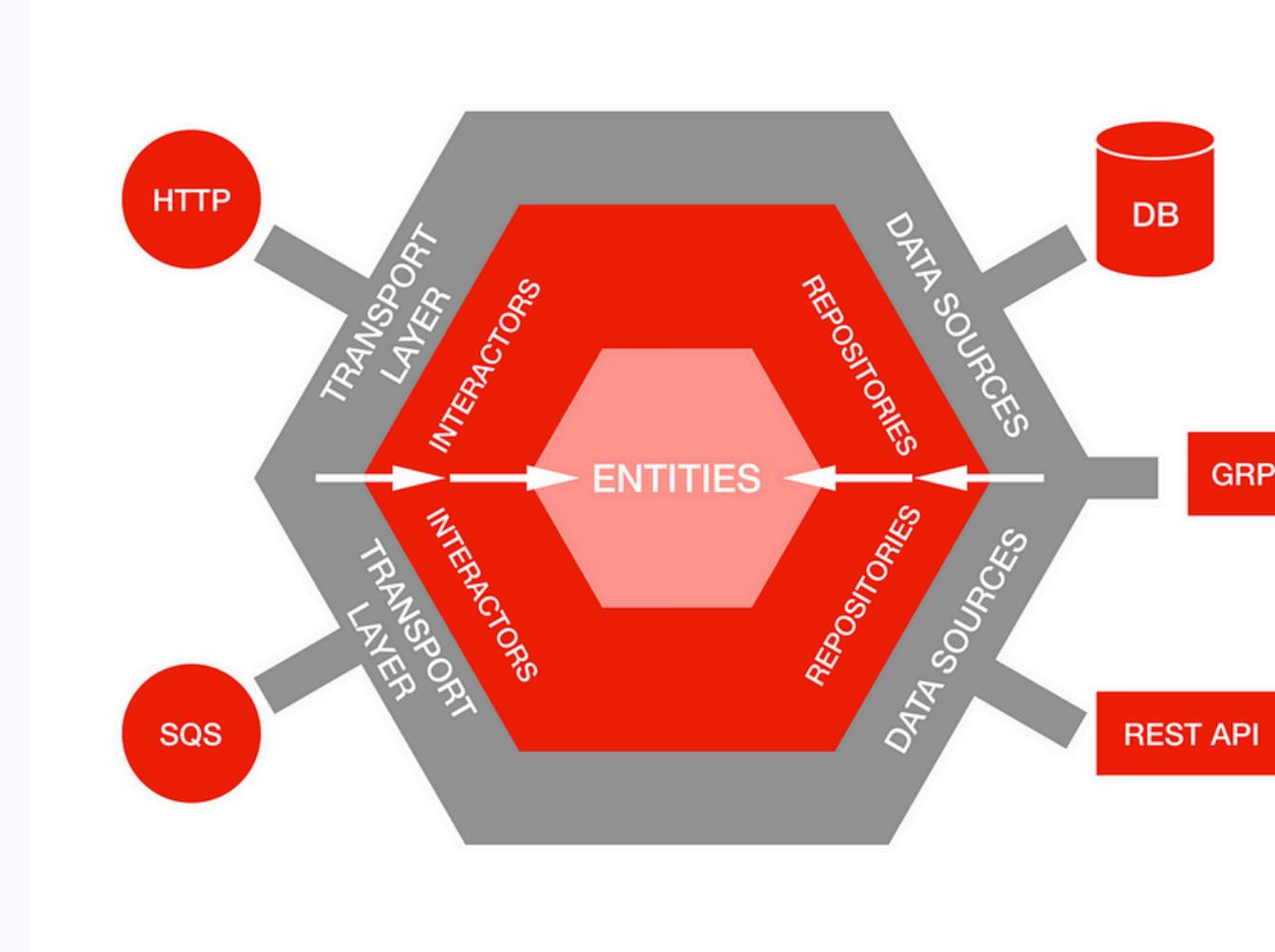


¿QUÉ ES?

Arquitectura hexagonal

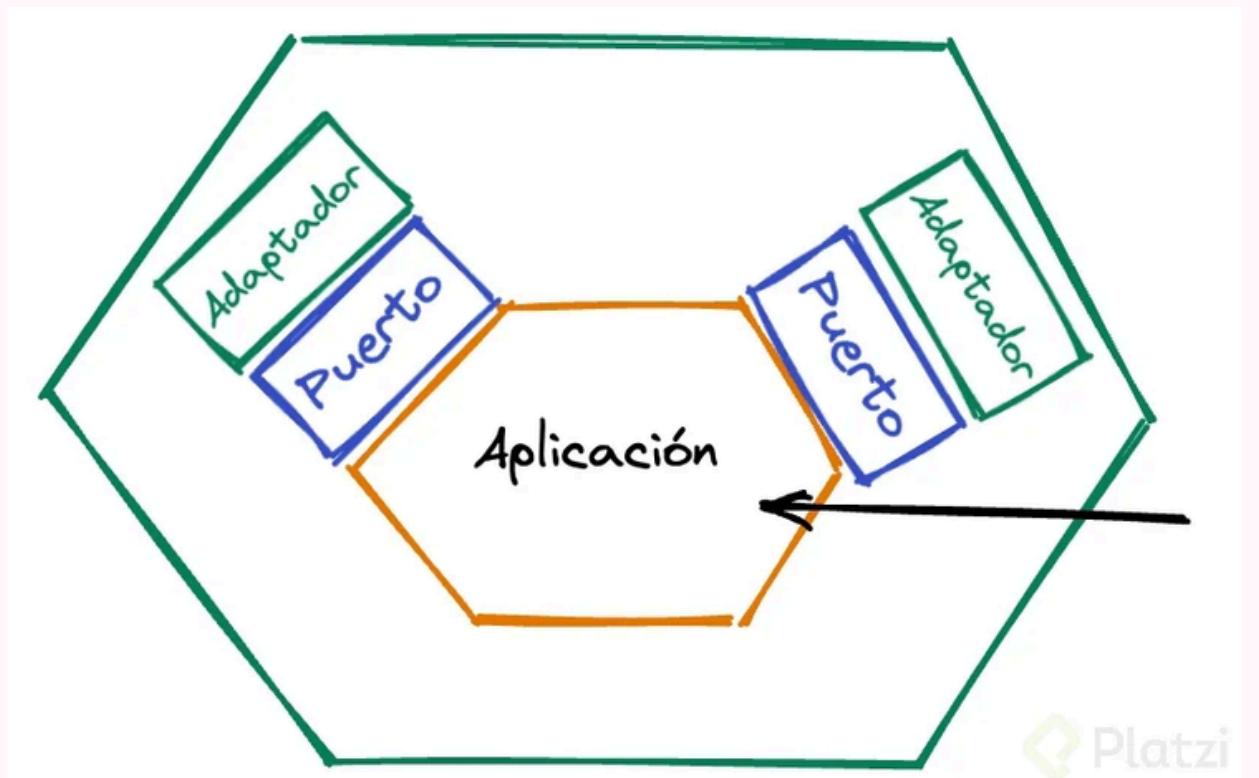


DEFINICIÓN



La **arquitectura hexagonal**, también conocida como **arquitectura de puertos y adaptadores**, se define por su separación clara entre el núcleo de negocio (lógica de la aplicación) y las interfaces externas, lo que permite la independencia de tecnologías y facilita el cambio de herramientas o frameworks.

CARÁCTERÍSTICAS



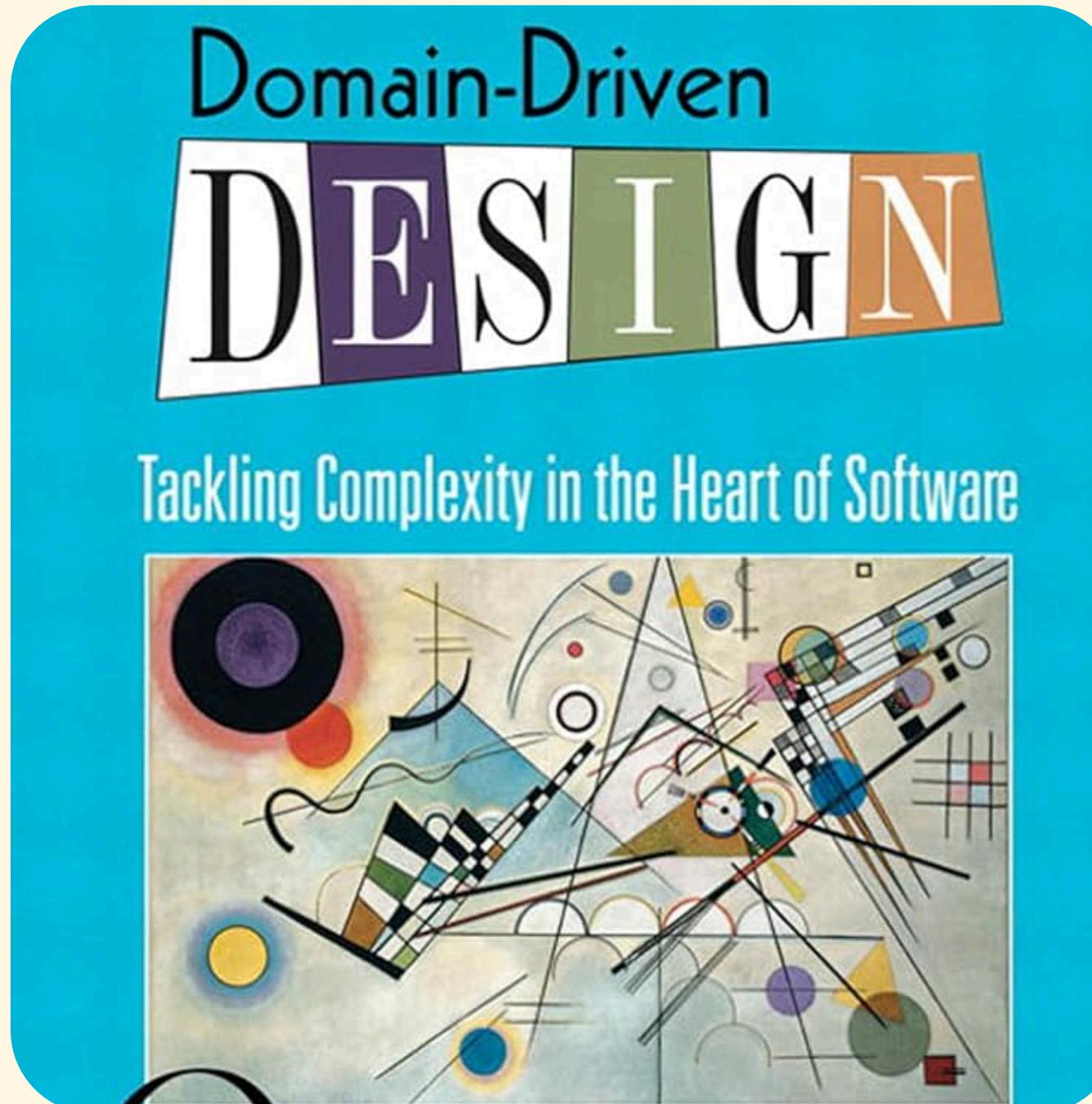
- **Separación de responsabilidades:** Divide la aplicación en capas independientes. El núcleo de negocio es independiente de cualquier infraestructura.
- **Puertos y adaptadores:** Los puertos son las interfaces que permiten la interacción con el núcleo, y los adaptadores son las implementaciones específicas para tecnologías.
- **Enfoque centrado en el dominio:** El núcleo de negocio contiene la lógica y reglas del dominio, separándolo de aspectos técnicos, permitiendo una independencia tecnológica.

HISTORIA



- La arquitectura hexagonal fue propuesta por **Alistair Cockburn** en **2005** con el objetivo de mejorar el manejo de dependencias en sistemas complejos.
- Surgió como una **evolución de las arquitecturas en capas tradicionales**, centrada en desacoplar la lógica de negocio de los frameworks y tecnologías externas.
- Cockburn buscaba una solución que permitiera una **mayor flexibilidad y mantenibilidad** al permitir cambiar componentes externos sin afectar el núcleo de la aplicación.

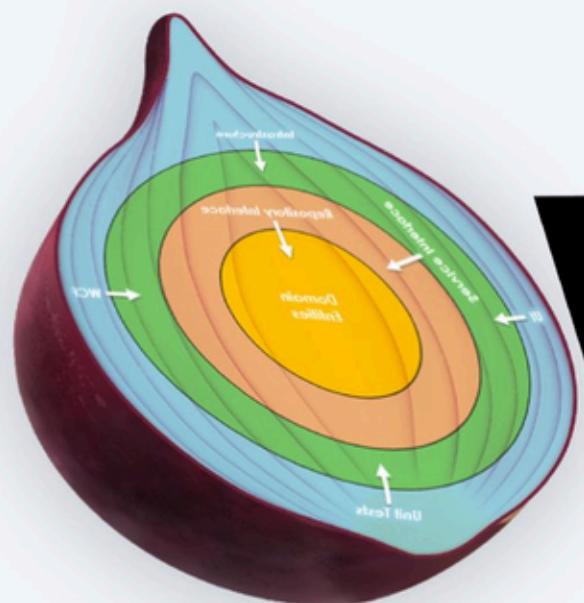
EVOLUCIÓN



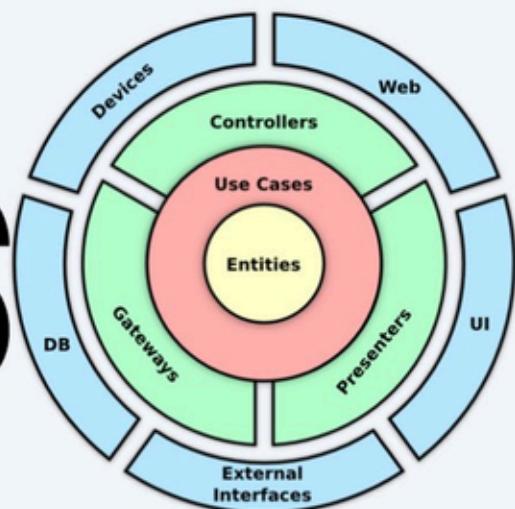
- **2010:** La arquitectura hexagonal empezó a ganar relevancia entre la comunidad de desarrollo ágil y los proponentes de Domain-Driven Design (DDD), dado que favorece un enfoque centrado en el dominio de negocio.
- **Clean Architecture y Onion Architecture:** La arquitectura hexagonal sentó las bases para otras arquitecturas, como la Clean Architecture (de Robert C. Martin) y la Onion Architecture. Ambas comparten principios clave: la independencia del núcleo de negocio respecto a la infraestructura.

EVOLUCIÓN

Onion Clean



VS



How are these **architectures** different?

- **Hexagonal** se centra en puertos y adaptadores para las interacciones externas, mientras que Clean y Onion se organizan en capas concéntricas.
- **Onion Architecture** es más estricta en evitar que el núcleo tenga cualquier dependencia de las capas externas.
- **Clean Architecture** es un enfoque más amplio y flexible, aplicando capas externas para manejar diferentes tecnologías, pero permitiendo una organización más modular.



@MilanJovanović

SWIPE >>>

VENTAJAS Y DESVENTAJAS

Ventajas:

- **Mantenibilidad:** Separar el núcleo de negocio de las interfaces permite modificar una parte sin afectar la otra.
- **Testabilidad:** Facilita las pruebas unitarias al no depender de la infraestructura externa.
- **Flexibilidad:** Cambiar tecnologías o integrar nuevos servicios es sencillo sin impacto en la lógica de negocio.

Desventajas:

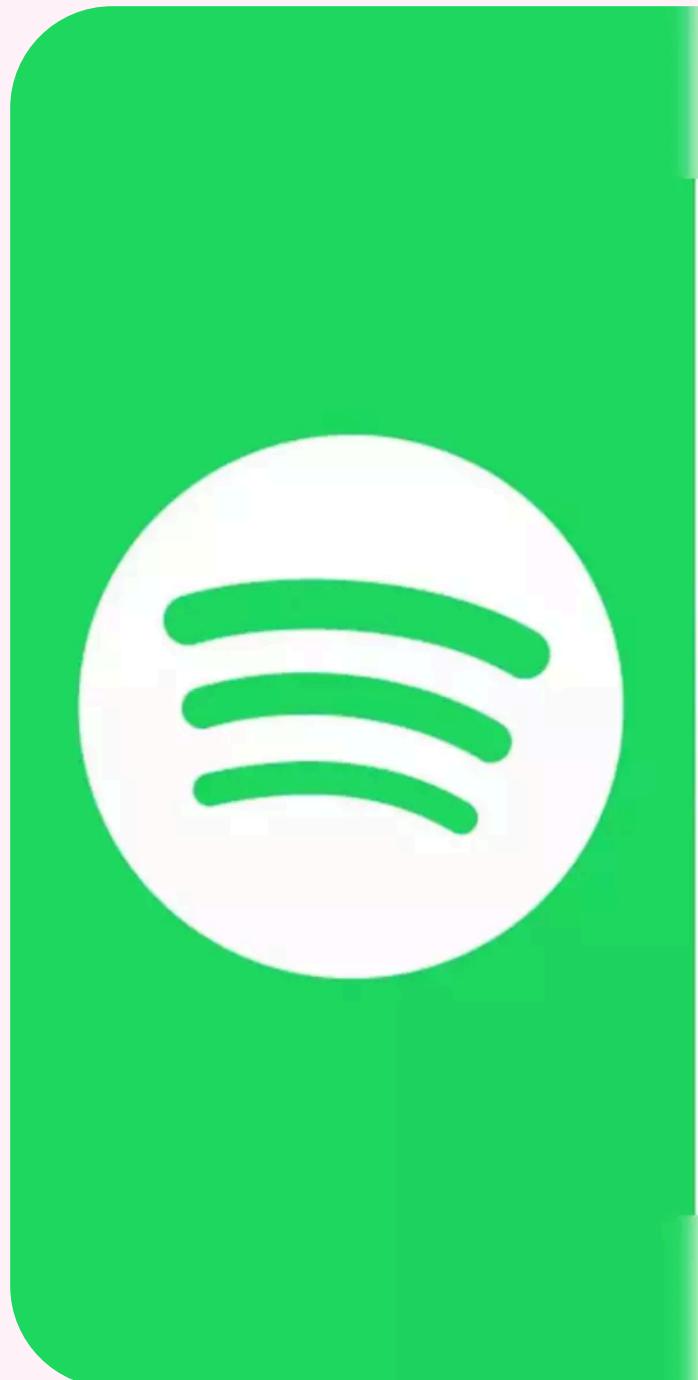
- **Complejidad inicial:** Requiere un diseño más cuidadoso y detallado desde el principio, lo que puede ser costoso en proyectos pequeños.
- **Confusión general:** Es difícil encontrar información consistente entre Hexagonal, Onion y Clean Architecture, ya que cada fuente presenta estructuras diferentes.

CASOS DE USO



- **Sistemas con alta frecuencia de cambios en las tecnologías externas:** Ideal cuando se prevén cambios en bases de datos, interfaces, o servicios externos.
- **Aplicaciones que requieren alta testabilidad:** Si es crucial tener una cobertura de pruebas sólida sin depender de tecnologías externas.
- **Desarrollo ágil y DDD:** En entornos donde la lógica de negocio es crítica y se busca que esté desacoplada de la infraestructura.

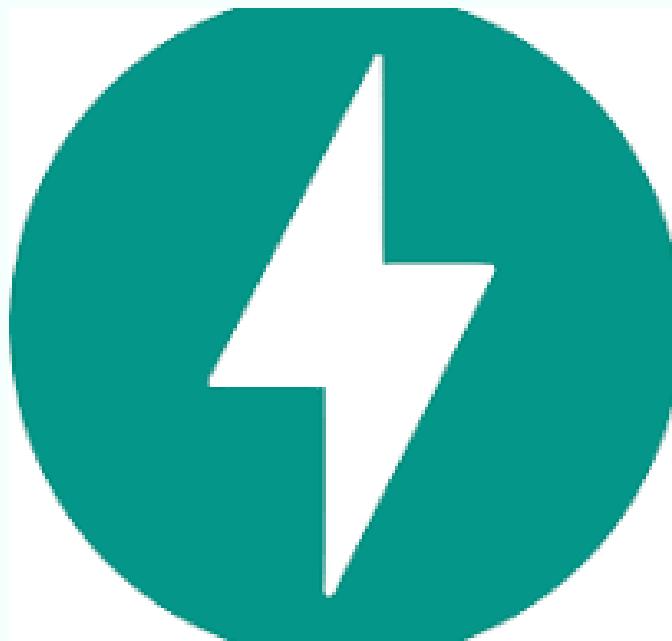
CASOS DE APLICACIÓN



- **Google:**
 - Utiliza microservicios que se basan en la arquitectura hexagonal.
 - Modulariza componentes, creando un sistema resistente a cambios.
 - En su sistema de publicidad, aísla la lógica de negocio de los sistemas de entrada y salida.
- **Spotify:**
 - Aísla la lógica de streaming de música de las interfaces de usuario y sistemas externos.
 - Cambios en la interfaz (web, escritorio, móvil) no afectan la lógica central.
 - Garantiza una experiencia de usuario consistente y sin interrupciones.

STACK DESIGNADO

FAST API



FastAPI es popular entre los científicos de datos y los profesionales del aprendizaje automático para crear API.

Lo utilizan empresas importantes como Uber y Netflix para crear sus aplicaciones.

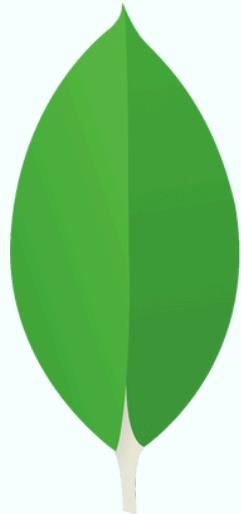
SVELTE



Svelte impulsa el 0,1 % de los sitios web

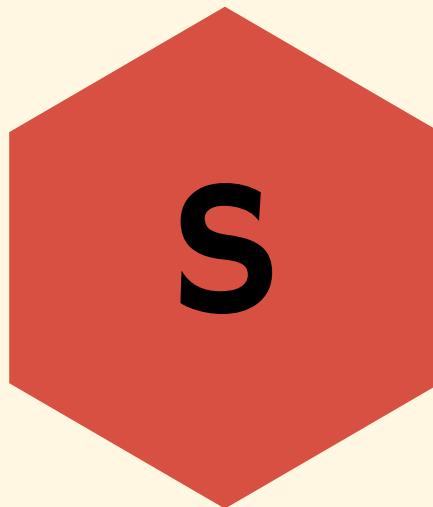
El 20% de los desarrolladores de JavaScript utilizan Svelte.

MONGO

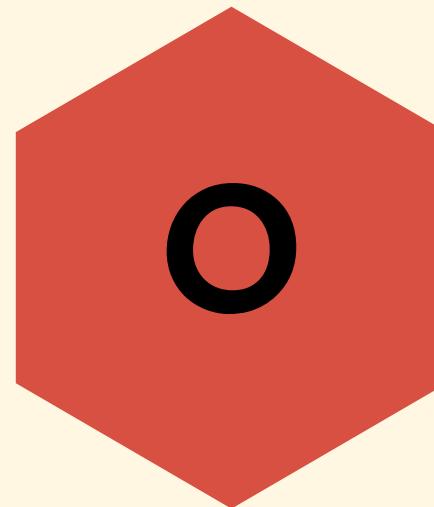


DB-Engines clasifica constantemente a MongoDB como la base de datos NoSQL más popular mes a mes.

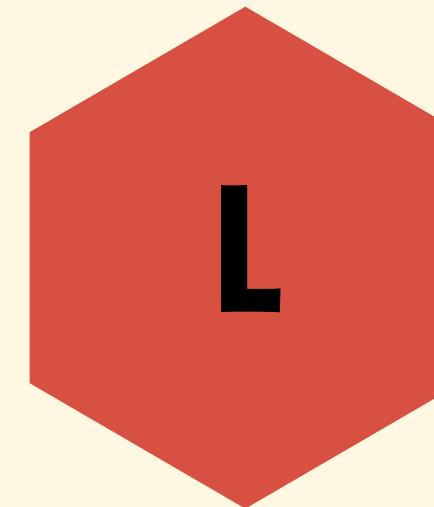
SOLID



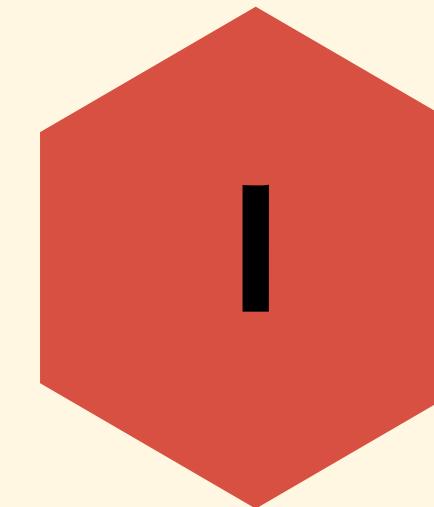
Se separan las responsabilidades del sistema en diferentes componentes:
núcleo interno para la lógica de negocio
los adaptadores externos para las interacciones con sistemas externos



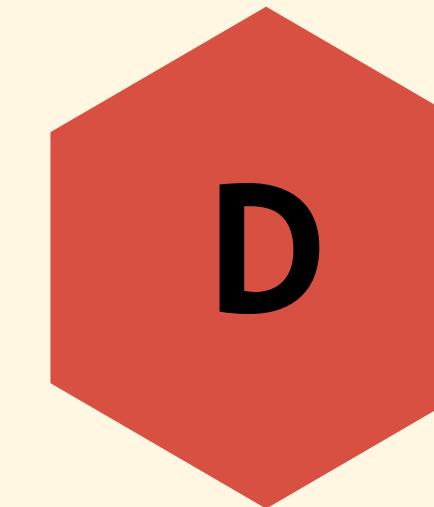
Se logra mediante interfaces definidas entre el núcleo y los adaptadores externos, permitiendo la adición de nuevos adaptadores sin modificar el núcleo interno.



asegurando que los adaptadores externos sean intercambiables sin impactar la funcionalidad del sistema, al implementar interfaces definidas por el núcleo interno para garantizar su sustituibilidad



al definir interfaces específicas para cada tipo de interacción entre el núcleo y los adaptadores externos, evitando dependencias innecesarias



El núcleo depende de interfaces en lugar de implementaciones concretas

ATRIBUTOS DE CALIDAD



ESCALABILIDAD



MANTENIBILIDAD



TESTEABILIDAD



FLEXIBILIDAD

TÁCTICAS

SEPARACIÓN DE
RESPONSABILIDADES

USO DE DTOS

INYECCIÓN DE DEPENDENCIAS

SEPARACIÓN DE
RESPONSABILIDADES

MOCKEO PARA PRUEBAS

MERCADO LABORAL

Technology | 2024 Stack Overflow Developer Survey

JavaScript has been a mainstay in the developer survey and on Stack Overflow since our first survey. The most popular programming language has been JavaScript every year we have done the survey except for 201...

 stackoverflow

The State of Developer Ecosystem 2023



The State of Developer Ecosystem in 2023 Infographic

Learn about the latest trends in tools, technologies, AI, and programming languages.

 JetBrains

Svelte

Sabiendo que el 20 % de los programadores de Javascript usan svelte y JS es una de los lenguajes mas utilizados segun Stack Overflow, con un 62.3% de usuarios de la plataforma haciendo que Svelte es utilizado en gran medida, pero el cambio de frameworks para algunos desarrolladores dificulta su adquisición

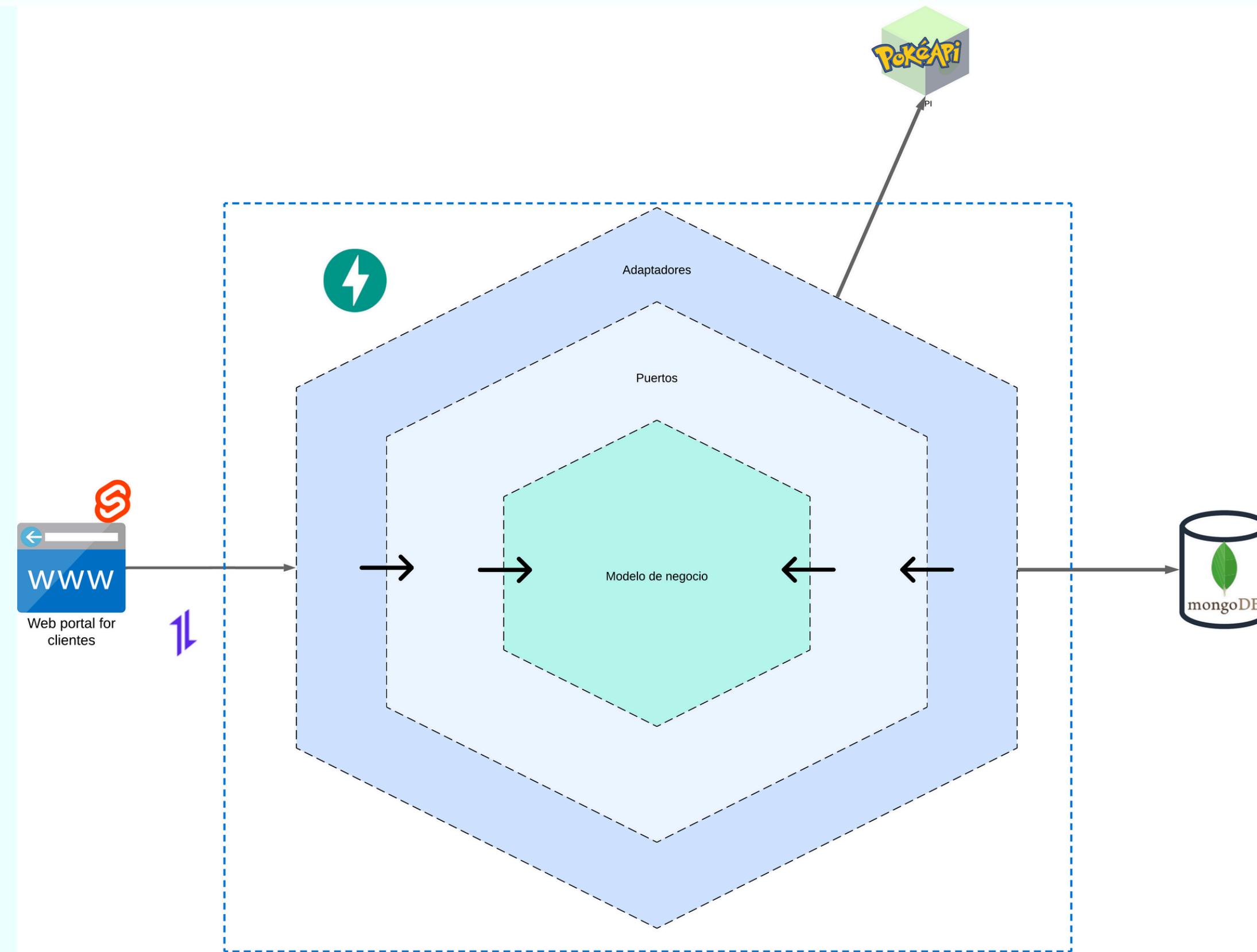
FastAPI

FastAPI es un marco basado en Starlette diseñado para crear APIs con Python,

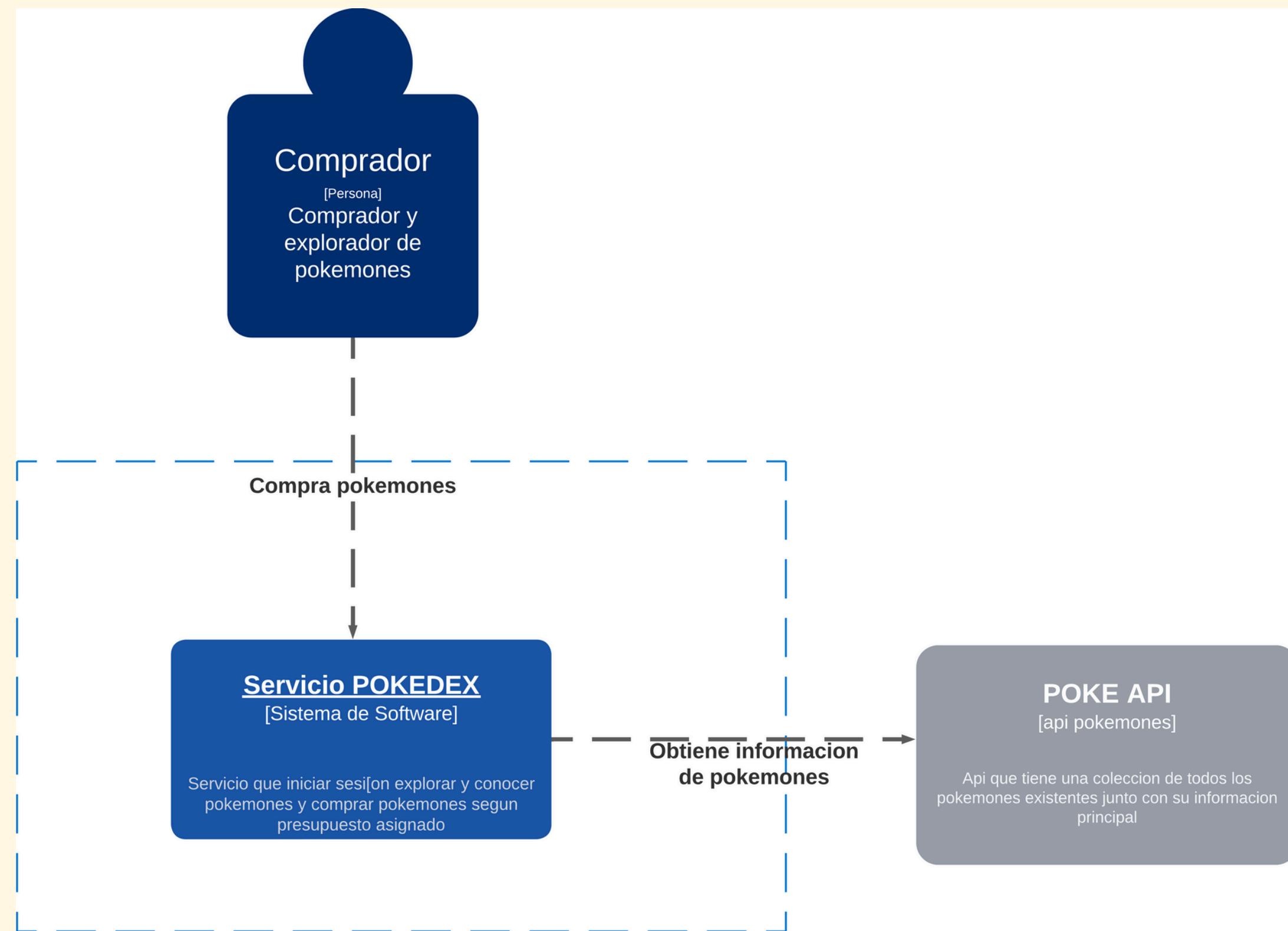
MongoDB

Mongo es el tercer tipo de base de datos mas utilizada en la industria por los desarrolladores segun jetbrains, donde se establecio que el 27% de los usuarios de la plataforma utilizan este stack tecnologico, no obstante MySql y las BDD relacionales aun son mas usadas

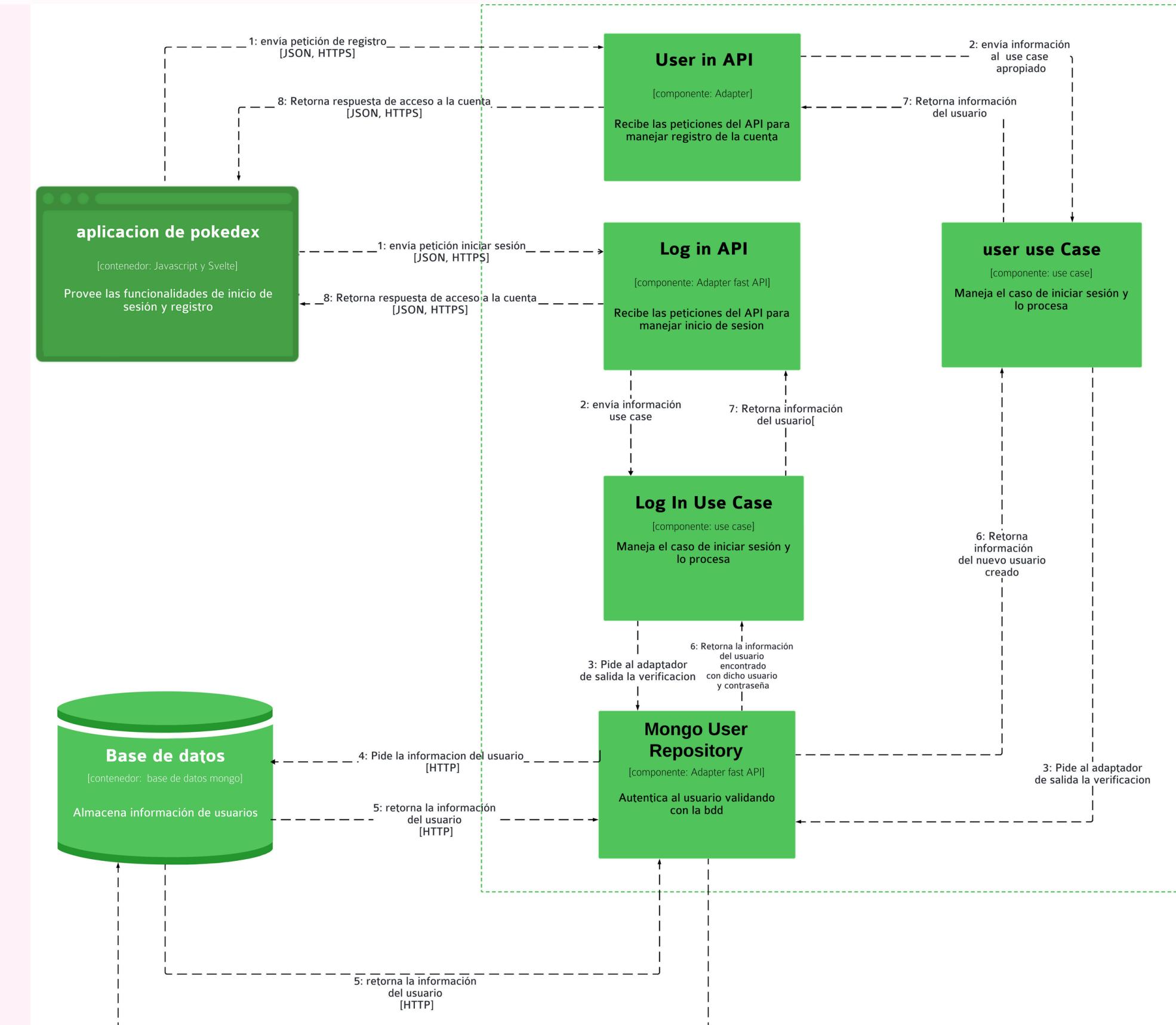
ARQUITECTURA ALTO NIVEL



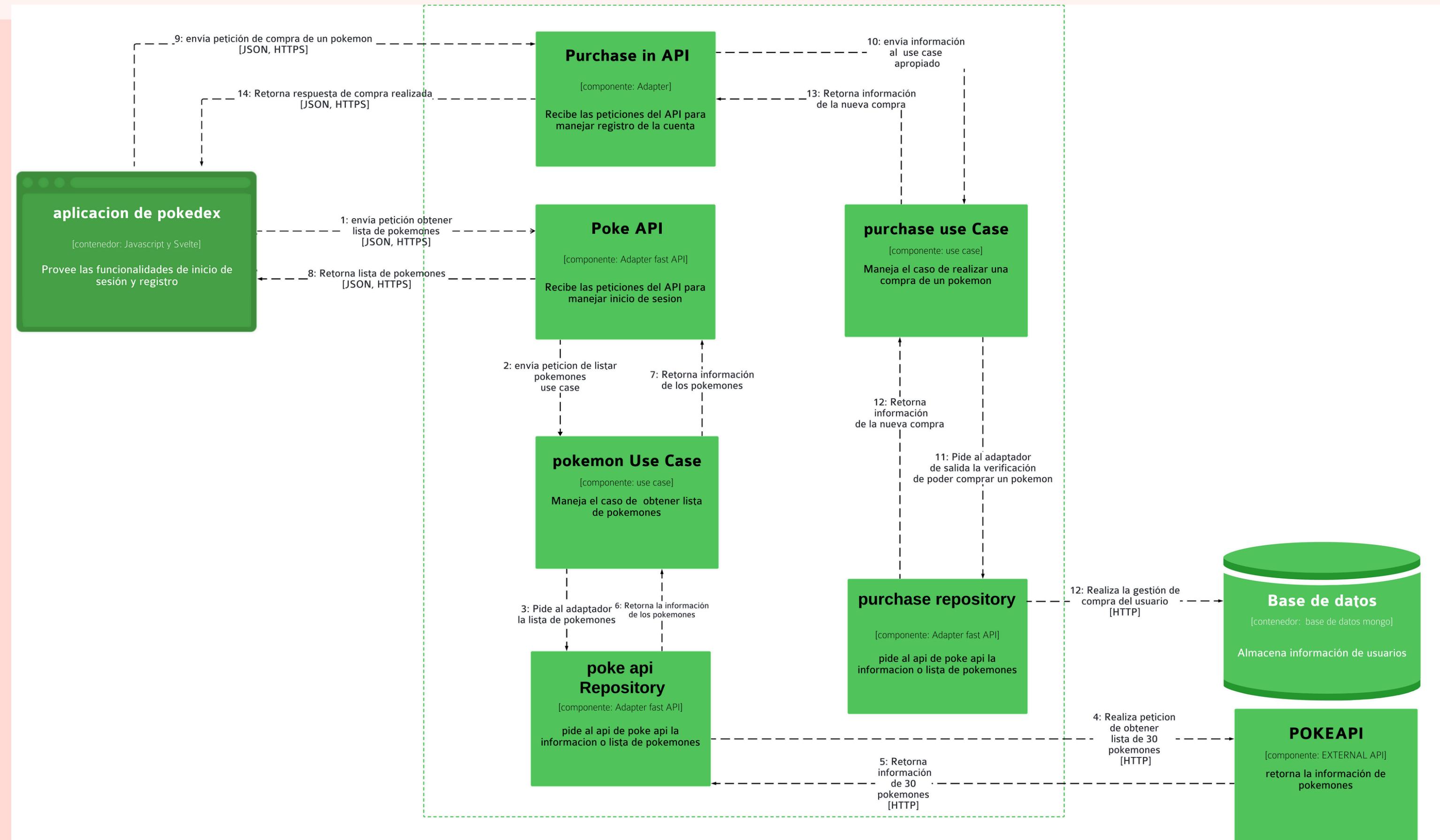
C4 CONTEXTO



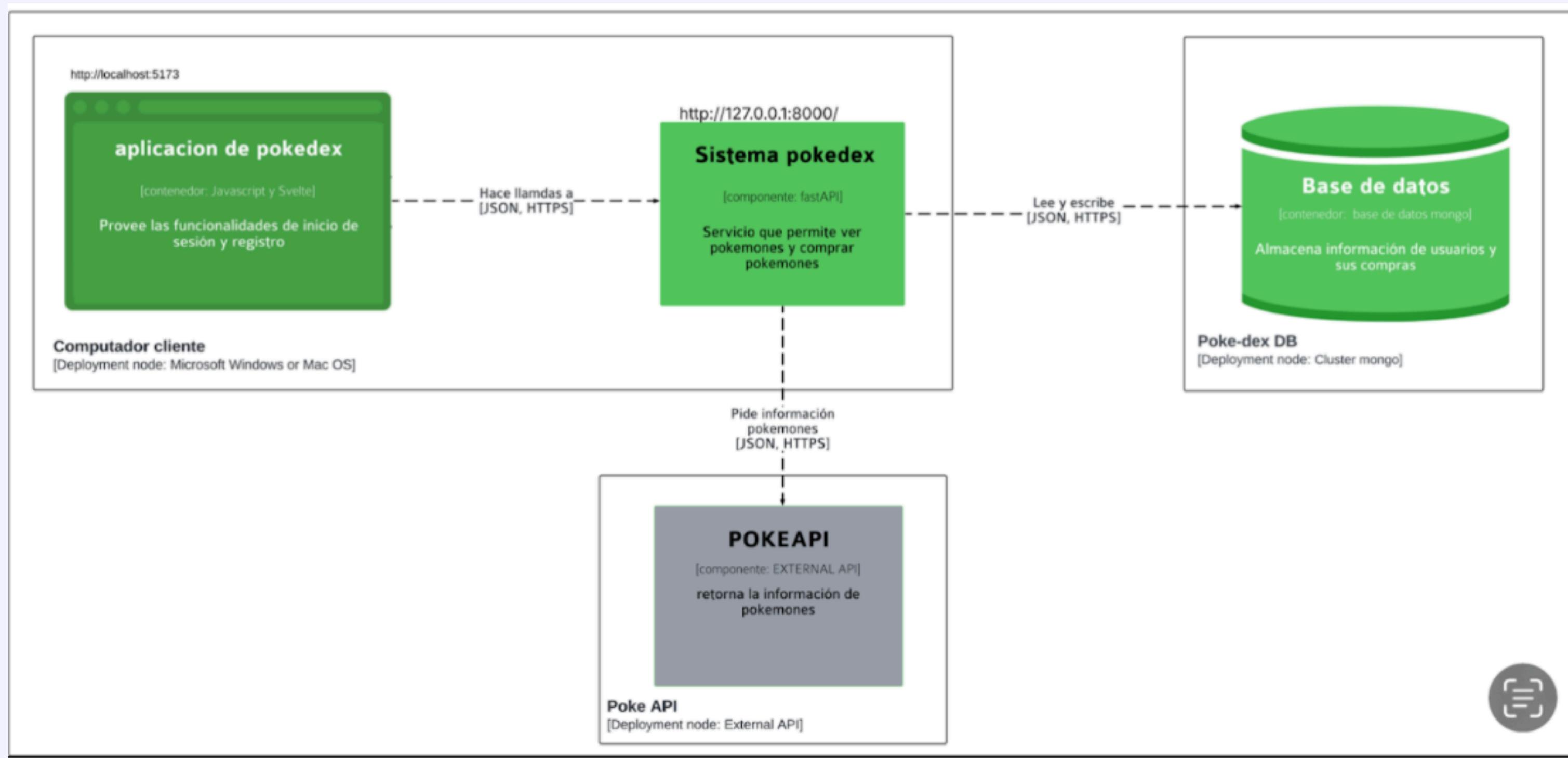
C4 DINÁMICO USUARIO



C4 DINÁMICO COMPRA



C4 DESPLIEGUE



UML PAQUETES

