Aplikacje sieciowe i webowe

dr inż. Rafał Brociek

Katedra Zastosowań Matematyki i Metod Sztucznej Inteligencji
Wydział Matematyki Stosowanej
Politechnika Śląska

10.10.2023

Zdefiniowanie problemu

Dobra architektura jest kluczem do budowania skalowalnych, modułowych i łatwych w utrzymaniu aplikacji. Różne architektury mogą różnić się szczegółami, ale wszystkie mają te same cele, którymi są rozdzielenie modułów aplikacji. Można osiągnąć ten podział, dzieląc aplikację na warstwy.

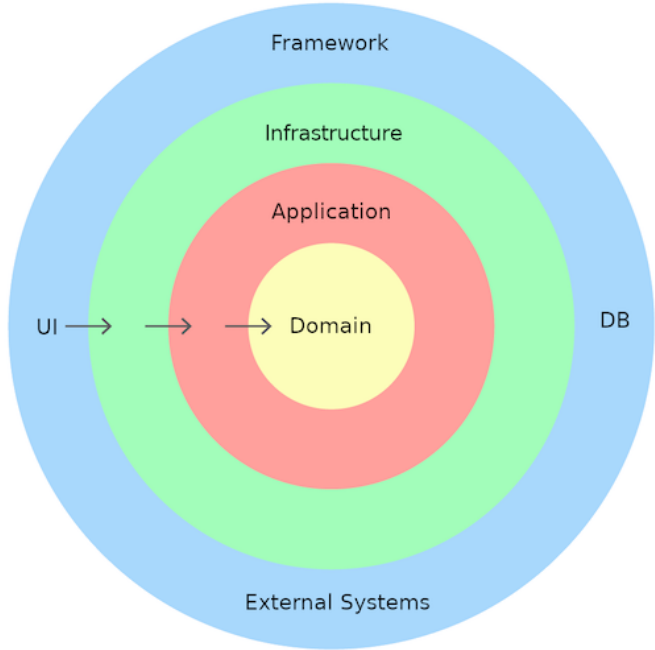
Czysta architektura

Czysta architektura (ang. clean architecture) to architektura oprogramowania, której celem jest utrzymywanie kodu pod kontrolą oraz łatwe zarządzanie nim. Główną koncepcją czystej architektury jest niezmienny (lub bardzo rzadko zmieniający się) główny kod/rdzeń/logika aplikacji (tzw. core). Powinien on być napisany bez żadnych bezpośrednich zależności. Oznacza to, że jeśli zmienia się struktura aplikacji, baza danych lub interfejs użytkownika, rdzeń systemu (reguły biznesowe/domena) nie powinien zostać zmieniony. Oznacza to, że zewnętrzne zależności są całkowicie zastępowalne.

Czysta architektura

Czysta architektura składa się z następujących warstw:

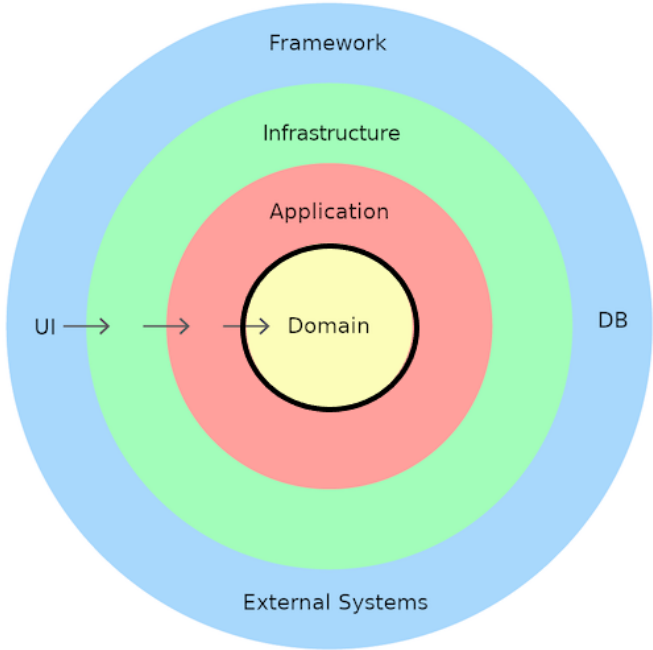
- *domeny* (ang. domain),
- *warstwy aplikacji* (ang. application layer),
- *warstwy infrastruktury* (ang. infrastructure layer),
- *warstwy framework*.



Rysunek: Schemat wzorca czystej architektury (źródło: `https://www.c-sharpcorner.com/article/introduction-to-clean-architecture-and-implementation-with-asp-net-core/`)

Domain layer

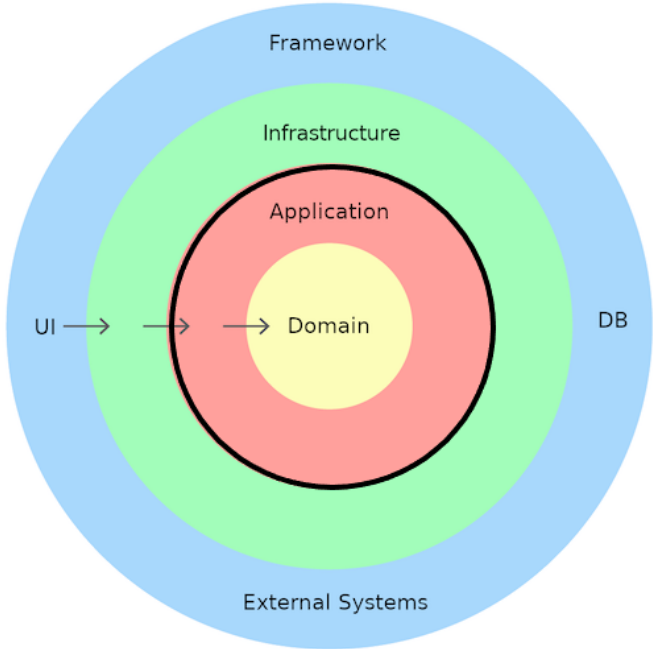
Warstwa ta znajduje się w centrum architektury i zawiera model danych aplikacji (tzw. ang. entities) w postaci klas modelu aplikacji lub klas modelu bazy danych. W przypadku podejścia typu „*code first*” (w ASP.NET Core) model ten posłuży do utworzenia tabel w bazie danych.



Rysunek: Schemat wzorca czystej architektury (źródło: `https://www.c-sharpcorner.com/article/introduction-to-clean-architecture-and-implementation-with-asp-net-core/`)

Application layer

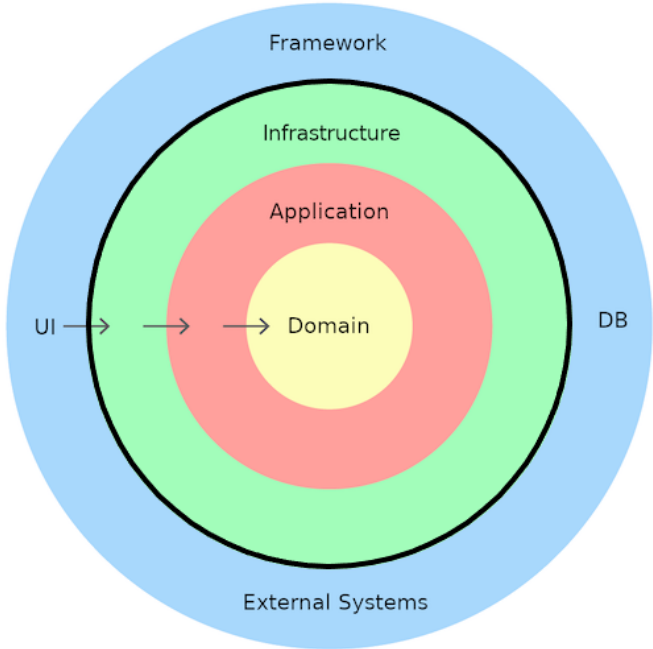
Warstwy domeny i aplikacji pozostają w centrum projektu, który jest znany jako rdzeń systemu (ang. core). Warstwa domeny zawiera model, a warstwa aplikacji zawiera logikę biznesową. Model może być współdzielony przez wiele systemów, ale logika biznesowa będzie zazwyczaj używana tylko w danym systemie/projekcie. Rdzeń będzie niezależny od dostępu do danych i innych problemów związanych z infrastrukturą. Możemy to osiągnąć za pomocą interfejsów i abstrakcji w rdzeniu i zaimplementować je przez inne warstwy poza rdzeniem.



Rysunek: Schemat wzorca czystej architektury (źródło: `https://www.c-sharpcorner.com/article/introduction-to-clean-architecture-and-implementation-with-asp-net-core/`)

Infrastructure layer

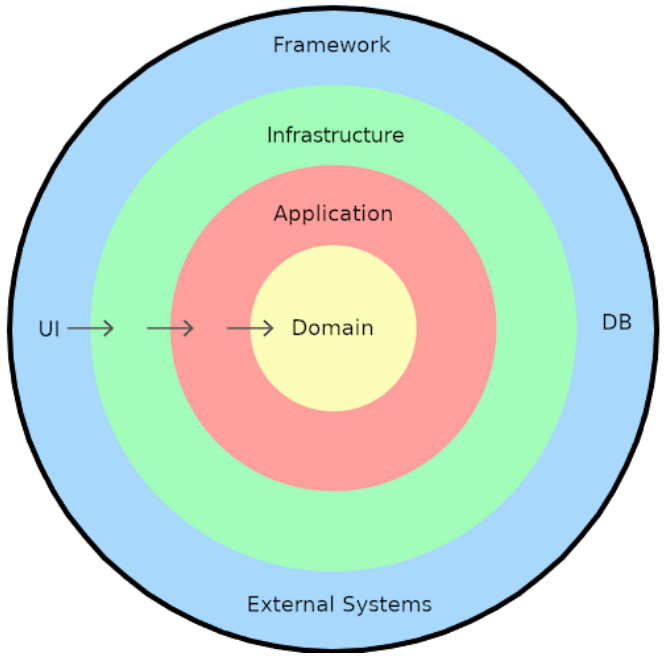
W warstwie infrastruktury znajdują się obiekty dotyczące wszystkich migracji bazy danych i obiekty kontekstu bazy danych. W tej warstwie znajdują się również repozytoria wszystkich obiektów modelu.



Rysunek: Schemat wzorca czystej architektury (źródło: `https://www.c-sharpcorner.com/article/introduction-to-clean-architecture-and-implementation-with-asp-net-core/`)

Framework layer

W tym kręgu znajduje się warstwa prezentacji, która prezentuje nam obiekty danych (najczęściej w jsonie) uzyskane z bazy poprzez żądania HTTP. W przypadku aplikacji front-endowych dane prezentujemy za pomocą interfejsu użytkownika.



Rysunek: Schemat wzorca czystej architektury (źródło: `https://www.c-sharpcorner.com/article/introduction-to-clean-architecture-and-implementation-with-asp-net-core/`)

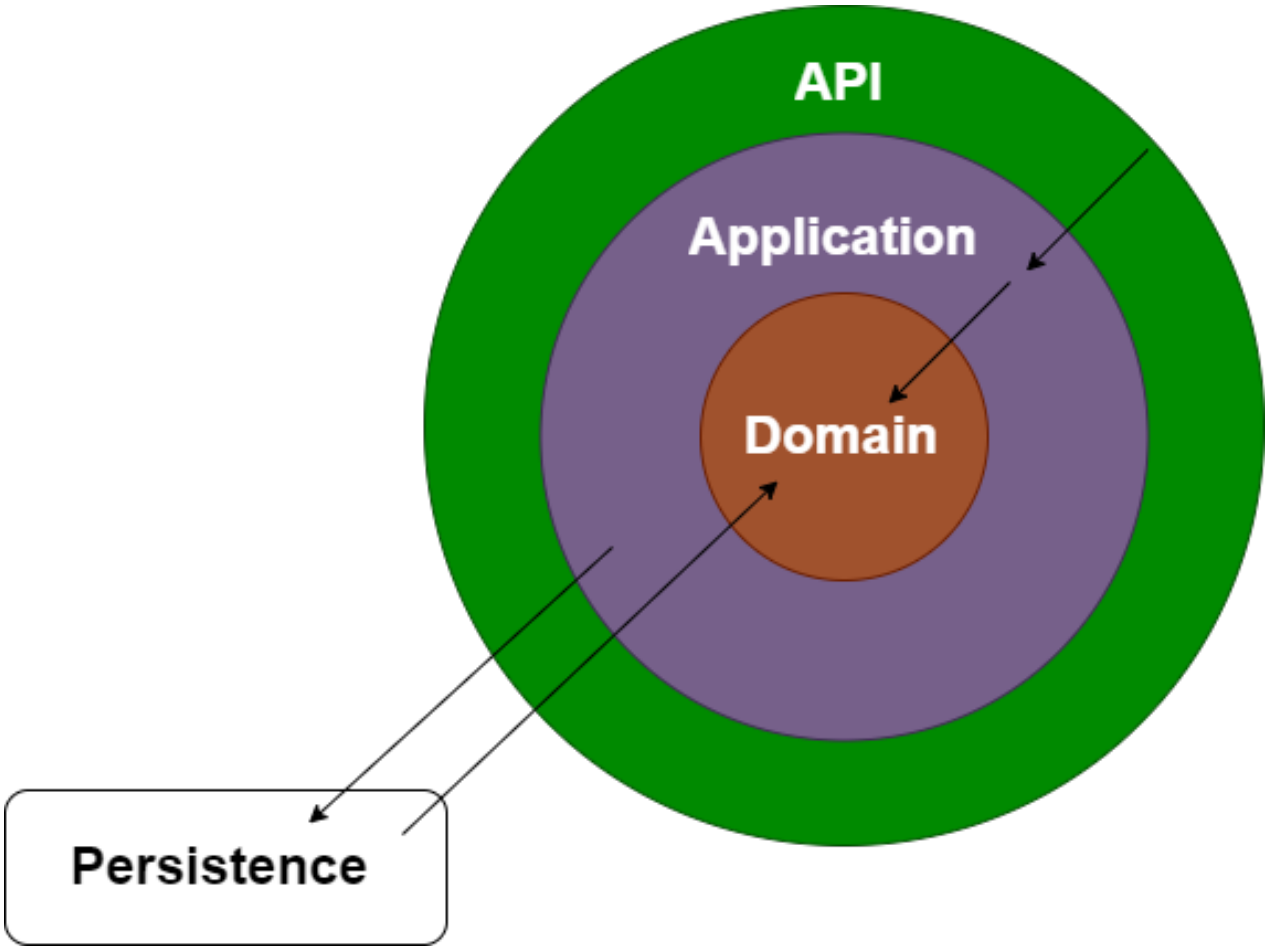
Założenia czystej architektury

- W czystej architekturze wszystkie zależności mają kierunek do wewnątrz, a rdzeń nie jest zależny od żadnej innej warstwy.
- Kluczową koncepcją czystej architektury jest to, że warstwa wewnętrzna nie powinna wiedzieć o warstwie zewnętrznej, ale warstwa zewnętrzna powinna wiedzieć o warstwie wewnętrznej.
- Warstwa domeny nie będzie miała odniesienia do żadnego projektu.
- Architektura ta pozwala na zmianę zewnętrznych elementów systemu bez wpływu na rdzeń systemu.

Omówienie tworzonego projektu

Celem pierwszego zadania jest utworzenie aplikacji webowej dotyczącej samochodów (lub dowolnych innych obiektów). W pierwszej kolejności utworzona będzie aplikacja serwerowa (tzw. backend) w technologii ASP.NET Core Web API. W projekcie wykorzystana zostanie relacyjna baza danych SQLite, a sam projekt będzie zaimplementowany z użyciem wzorca czystej architektury.

Omówienie tworzonego projektu



Rysunek: Schemat wzorca czystej architektury w tworzonej aplikacji

Warstwy systemu

Domain layer. Warstwa, w której znajdować się będzie model danych. Model ten nie powinien być zależny od żadnej innej warstwy.

Application layer. Warstwa ta będzie odpowiedzialna za logikę biznesową aplikacji i jest zależna od warstwy domeny (domain) oraz warstwy trwałości (persistence). W tej warstwie docelowo umieszczone będą klasy odpowiedzialne za obsługę CRUDa, tzw. handlery.

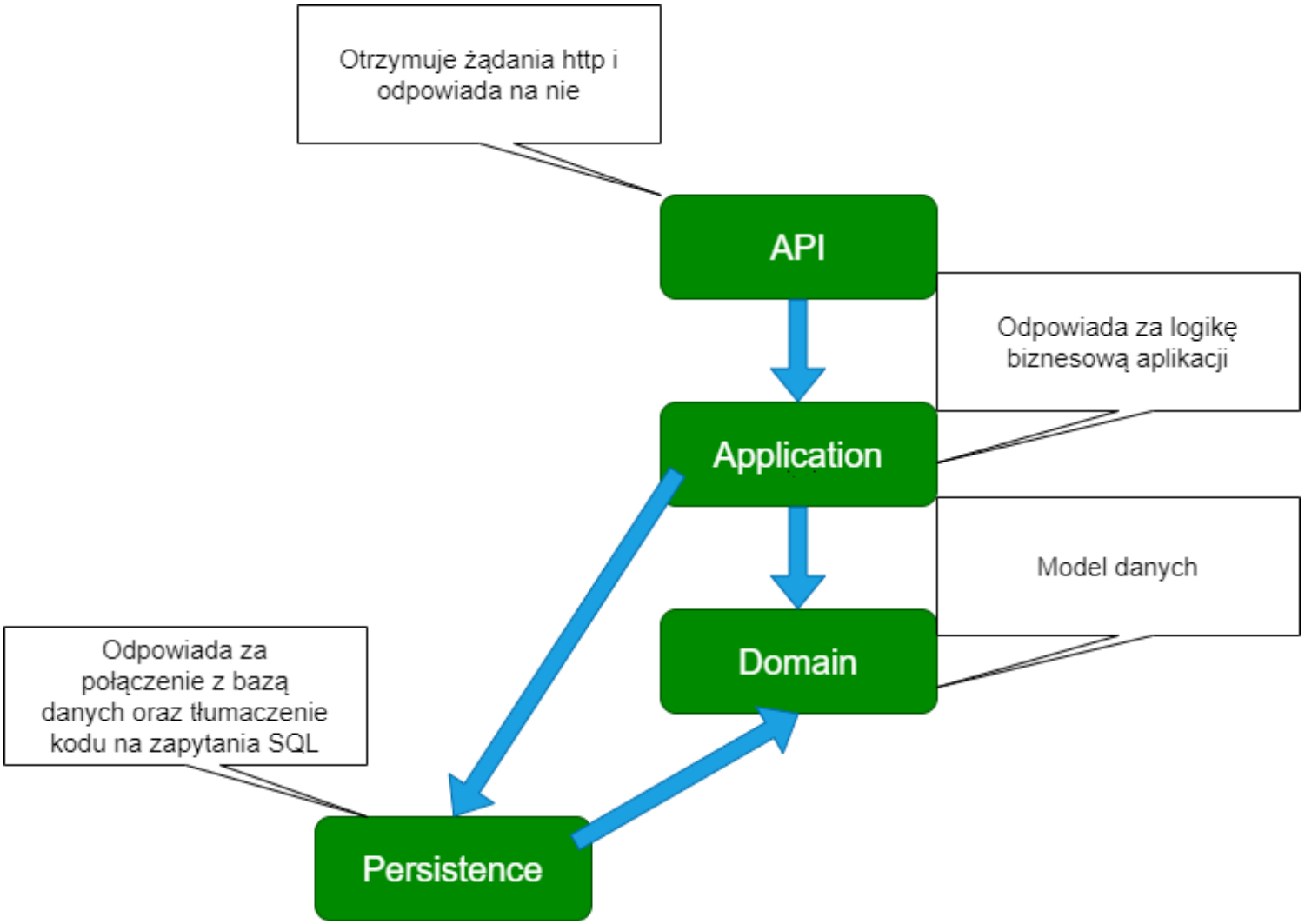
API layer. Warstwa zależna od warstwy aplikacji. To właśnie tam zostaną umieszczone kontrolery.

Persistence layer. Warstwa trwałości aplikacji jest odpowiedzialna za przechowywanie i pobieranie danych aplikacji. Używa się jej, aby odizolować logikę biznesową aplikacji od mechaniki magazynu, takiego jak używana przez nas baza danych. W tej warstwie znajdować się będzie klasa `DataContext` odpowiedzialna za kontekst bazy danych. Warstwa ta w naszym rozwiązaniu będzie zależna od warstwy domeny.

Tworzenie solucji

- Najpierw stworzymy warstwę API (projekt API) i na początku będzie ona odpowiedzialna za odbieranie żądań http za pośrednictwem endpointów i odpowiedzi na te żądania.
- Następnie utworzymy projekt z warstwy aplikacji, który będzie odpowiedzialny za logikę biznesową i będzie zależny od modelu/domeny.
- Kolejny projekt dotyczyć będzie modelu danych (warstwa domeny).
- Kolejny projekt należeć będzie do warstwy trwałości (persistence layer). Projekt trwałości zapewnia połączenie z bazą danych i tłumaczy kod, który piszemy, na zapytania SQL.

Schemat tworzonego systemu



Rysunek: Schemat tworzonych projektów, ich odpowiedzialności oraz zależności między nimi

Dziękuję za uwagę