### Wymagania

Zaprojektuj system informatyczny w wybranych technologiach.

Kompletny projekt powinien zawierać implementację oraz dokumentację (dokument pdf):

Dokumentacja powinna zawierać:

* Wymagania funkcjonalne i niefunkcjonalne – 1 pkt.
* Diagramy klas – 1 pkt.
* Opis sposobów i metod testowania – 1 pkt.

Implementacja powinna obejmować:

* Warstwa Logiki biznesowej (ocena 3 - element wymagany) – 5 pkt.
* Warstwa prezentacji – 5 pkt.
* Baza danych – 4 pkt.

### Punkty ujemne:

### Brak określonych, lub błędne wymagania funkcjonalne i niefunkcjonalne: – 0,25 pkt.

### Brak diagramów klas: – 0,25 pkt.

### Brak testów jednostkowych: od - 0,25 pkt. do – 1 pkt.

### Brak dokumentacji technicznej kodu: od - 0,25 pkt. do – 1 pkt.

### Ocena końcowa: 2 – < 9 pkt. 3 – 9-10 pkt. 3,5 – 11-12 pkt. 4 – 13-14 pkt. 4,5 – 15-16 pkt. 5 – 17 pkt. 6 – 17 pkt. + wykorzystanie docker compos

### Na koniec zajęć projekt należy przedstawić oraz obronić. Wersja wstępna (alpha) – 01.06.2024r., termin oddania ostatecznej wersji, po uwagach prowadzącego do sesji (koniec czerwca).

### Dokumentacja powinna zawierać: wymagania funkcjonalne i niefunkcjonalne, diagram klas – tylko przy językach obiektowych, jeśli język jest nie obiektowy należy zastosować diagram użycia; opis sposobów i metod stosowania. Testy jednostkowe to minimum, które trzeba zrobić, jeśli ich nie będzie to będą punkty ujemne do oceny za projekt.

### Implementacja: minimum do zaliczenia jest aplikacja trójwarstwowa (wyraźne rozdzielenie na warstwę logiki biznesowej, prezentacji i warstwy danych). Warstwa logiki biznesowej jest „totalnym minimum” dla prowadzącego, bez tego zespół nie zaliczy. Do uzyskania 5 punktów nie może być backend pojedynczy, należy zastosować backend rozproszony. Warstwa prezentacji – ważne, aby jej zaprojektowanie i implementacja była zgodna z dobrymi praktykami programistycznymi i zasadami inżynierii. Nie musi być piękna i cukierkowa, musi być prawidłowo zaimplementowana. Warstwa danych – mamy korzystać z dowolnej bazy danych. Trzeba będzie uzasadnić jaką bazę i dlaczego wybraliśmy, czy typ naszej bazy jest odpowiednia do danych czynności i danego systemu, jakie obsługujemy.

### Punkty ujemne – wskazówka co jest „must have” w projekcie.

### Dokumentacja kodu = czysta dokumentacja kodu, zapisywane komentarze do klas/metod. Komentarze na bazie których tworzy się później dokumentację. Po wybraniu języka programowania trzeba znaleźć narzędzie wygodne do dokumentacji tego kodu, np. w Java jest to Javadoc.

### Dobre praktyki pisania kodu i dokumentacji to język angielski, ale jeśli ustandaryzujemy w zespole język polski, to może być po polsku.