**INŻYNIERIA OPROGRAMOWANIA PROJEKT – DOKUMENTACJA I OPIS**

**Zadanie**

Celem zadania projektowego będzie stworzenie działającego i w pełni funkcjonalnego oprogramowania.

**Źródła:**

1. <https://github.com> – repozytorium
2. <https://javastart.pl/> - Java
3. <https://docs.oracle.com/javase/9/> - Java, InteliJ
4. <https://www.visual-paradigm.com/support/documents/vpuserguide/> - dokumentacja dotycząca narzędzia Visual Paradigm
5. <https://brasil.cel.agh.edu.pl/~09sbfraczek/> - materiały dotyczące Inżynierii Oprogramowania
6. <https://msdn.microsoft.com/en-us/library/dd409393.aspx> - diagram komponentów
7. <https://msdn.microsoft.com/en-us/library/dd409416.aspx> - diagram klas
8. Pragmatic Unit Testing in Java 8 with JUnit by Jeff Langr, with Andy Hunt and Dave Thomas –Javy oraz testowania
9. <http://code.makery.ch/library/javafx-8-tutorial/> - JavaFx, SceneBuilder

**Testy:**

Według (8), str.99, podpunkt 8.1 - Where to Put Test Code, "In general, you don't want to break any encapsulation for the sake of testing (or as Mom used to say, "don't expose your privates!"). Most of the time, you should be able to test a class by exercising its public methods. If there is significant functionality that is hidden behind private or protected access, that might be a warning sign that there's another class in there struggling to get out." Oznacza to, że zgodnie z przyjętymi dobrymi praktykami w naszym kodzie nie testowaliśmy prywatnych metod.

Jedyny test wykrywający błędy zawarty w mojej części programu dotyczy prawidłowo wpisywanych danych do odpowiedniego pola – gdy wpiszemy dane w nieodpowiednim formacie – pokazuje się błąd.

**Dobrane narzędzia oraz technologie:**

Computer Aided Software Engineering - Visual Paradigm – licencja akademicka: <https://ap.visual-paradigm.com/agh-university-of-science-and-technology2>

Repozytorium Git (<https://github.com/>)

IntelliJ IDEA: The Java IDE for Professional Developers by JetBrains - <https://www.jetbrains.com/idea/>

Jezyk programowania – Java

**Diagramy ULM:**

Diagram klass – „jest statycznym diagramem, przedstawiającym strukturę aplikacji bądź systemu w paradygmacie programowania obiektowego”(9). Taki diagram zawiera informacje o statycznych związkach między elementami (klasami).

Symbolem klasy jest prostokąt, zwykle podzielony poziomymi liniami na trzy sekcje: nazwy, atrybutów, operacji.

Związki między klasami:



Diagram komponentów - służy do ilustracji organizacji i zależności pomiędzy komponentami. Diagram komponentów prezentuje system na wyższym poziomie abstrakcji niż diagram klas, gdyż każdy z komponentów może być implementacją jednej lub większej liczby klas. Diagramy komponentów służą do określania szczegółów niezbędnych do budowy systemu.

Opisane diagramy znajdują się w repozytorium Git.