

# Języki skryptowe

## Laboratorium

### Zasady i wymagania dotyczące projektu

1. Celem projektu jest wykorzystanie umiejętności zdobytych podczas laboratoriów w ramach kursu "Języki skryptowe" w warunkach laboratoryjnych, zbliżonych do rzeczywistych, tj. efektem projektu powinien być reprezentatywny prototyp aplikacji komputerowej.
2. Projekt może być realizowany samodzielnie lub w grupach dwuosobowych.
3. Projekt powinien stanowić aplikację wykonaną w języku Python. W przypadku, gdy wykonanie całej aplikacji w tym języku jest niemożliwe lub utrudnione (np. część front-endowa uruchamiana jest w przeglądarce internetowej), należy pokryć kodem w języku Python możliwie największą część systemu.
4. Projekt powinien być wykonany zgodnie z zasadami poprawnego konstruowania kodu w zakresie [podziału na warstwy](#), tzn. powinien uwzględniać separację części odpowiedzialnej za interfejs użytkownika od części realizującej logikę działania systemu, może być oparty o architektoniczne wzorce projektowe (w zależności od typu projektu mogą to być np. [MVC](#), [ECS](#), itp.). Podczas pisania kodu należy również uwzględnić zasady takie jak: [SOLID](#), [DRY](#), czy [POLA](#).
5. Projekt powinien odpowiadać na rzeczywiste potrzeby użytkowników. Oznacza to, że wymagania powinny dotyczyć faktycznych problemów, a opracowane oprogramowanie powinno być komukolwiek przydatne, nie powinno stanowić jedynie demo umiejętności programistycznych autora. Przed rozpoczęciem realizacji projektu, jego temat i zakres funkcjonalny należy uzgodnić z prowadzącym zajęcia. Przykładowe projekty (tylko do inspiracji):
  - a. Wyszukiwarka treści w dokumentach PDF i DOCX
  - b. Lokalny tagger zdjęć z możliwością filtrowania
  - c. Czytnik i analizator paragonów (OCR + kategoryzacja)
  - d. Interpreter minijęzyka z edytorem kodu
  - e. Monitor aukcji (np. Allegro, OLX)
  - f. Bot Discordowy/Slackowy
6. Klasy i funkcje zapewniające kluczowe funkcjonalności projektu należy pokryć testami jednostkowymi.
7. Projekt powinien posiadać interfejs użytkownika, preferowany jest interfejs graficzny. Ewentualnie, można również wykonać aplikację webową, gdzie aplikacja napisana w Pythonie będzie służyć jako back-end.
8. Projekt powinien posiadać wykonane narzędzia ułatwiające przeprowadzenie wdrożenia aplikacji poprzez tworzenie pakietów/obrazów. W zależności od modelu wdrożenia, można skorzystać z np.:
  - a. obrazu Docker'owego,
  - b. [PyInstaller](#)
  - c. c. własnego skryptu.
9. Podczas pracy nad projektem należy korzystać z wybranego, ulubionego systemu kontroli wersji (np. git).

10. Projekt powinien zawierać dokumentację użytkownika oraz dokumentację techniczną (np. plik README.md opisujący strukturę, sposób uruchomienia i zależności).