SYLABUS

DOTYCZY CYKLU KSZTAŁCENIA 2026/2027 - 2029/2030

Rok akademicki 2027/2028

1. PODSTAWOWE INFORMACJE O PRZEDMIOCIE

Nazwa przedmiotu	*programowanie wspomagane AI*
Kod przedmiotu*	
Nazwa jednostki prowadzącej kierunek	*Instytut Informatyki, Wydział Nauk Ścisłych i Technicznych*
Nazwa jednostki realizującej przedmiot	*Instytut Informatyki, Wydział Nauk Ścisłych i Technicznych*
Kierunek studiów	*sztuczna inteligencja*
Poziom studiów	*studia I stopnia*
Profil	*ogólnoakademicki*
Forma studiów	*stacjonarne*
Rok i semestr/y studiów	*rok II, semestr 4*
Rodzaj przedmiotu	*przedmiot obieralny kierunkowy 1*
Język wykładowy	*polski*
Koordynator	*dr hab. Jan G. Bazan, prof. UR*
Imię i nazwisko osoby prowadzącej / osób prowadzących	*dr hab. Jan G. Bazan, prof. UR*

***** - *opcjonalnie, zgodnie z ustaleniami w Jednostce*

1.1.Formy zajęć dydaktycznych, wymiar godzin i punktów ECTS

Semestr (nr)	Wykł.	Ćw.	Konw.	Lab.	Sem.	ZP	Prakt.	Inne (jakie?)	**Liczba pkt. ECTS**
4	15			15					3

1.2. Sposób realizacji zajęć

zajęcia w formie tradycyjnej

1.3 Forma zaliczenia przedmiotu (z toku)

wykład – zaliczenie bez oceny, laboratoria – zaliczenie z oceną

2.WYMAGANIA WSTĘPNE

Znajomość zagadnień realizowanych na przedmiotach: algorytmy i struktury danych, bazy danych, programowanie obiektowe, programowanie w języku Python

3. CELE, EFEKTY UCZENIA SIĘ, TREŚCI PROGRAMOWE I STOSOWANE METODY DYDAKTYCZNE

3.1 Cele przedmiotu

C1	Celem przedmiotu jest zapoznanie studentów z nowoczesnymi metodami programowania wspieranymi przez narzędzia sztucznej inteligencji.
C2	Studenci nabędą umiejętność efektywnego wykorzystania modeli AI (w tym modeli językowych i narzędzi generatywnych) w procesie projektowania, implementacji, testowania oraz dokumentowania oprogramowania, z uwzględnieniem aspektów etycznych, prawnych i bezpieczeństwa.

3.2 Efekty uczenia się dla przedmiotu

EK (efekt uczenia się)	Treść efektu uczenia się zdefiniowanego dla przedmiotu	Odniesienie do efektów kierunkowych [1]
EK_01	Student zna podstawowe pojęcia związane z AI w kontekście inżynierii oprogramowania oraz rozumie możliwości i ograniczenia narzędzi AI wspierających programowanie.	K_W07
EK_02	Student projektuje rozwiązania programistyczne z wykorzystaniem wsparcia AI oraz wykorzystuje narzędzia AI do generowania, analizy kodu, tworzenia testów i dokumentacji.	K_U09
EK_03	Student potrafi krytycznie oceniać jakość kodu wygenerowanego przez AI.	K_U09

3.3 Treści programowe

A. Problematyka wykładu

1. Wprowadzenie do AI w inżynierii oprogramowania.
2. Modele językowe i generatywne – zasady działania
3. Inżynieria promptów w programowaniu.
4. Generowanie kodu źródłowego z użyciem AI
5. Testowanie, debugowanie i refaktoryzacja wspierane AI
6. Bezpieczeństwo, etyka i aspekty prawne.

B. Problematyka laboratoriów

1. Konfiguracja środowiska programistycznego z narzędziami AI.
2. Generowanie kodu w wybranych językach programowania.
3. Analiza i poprawa jakości kodu przy użyciu AI.
4. Tworzenie testów jednostkowych z pomocą AI
5. Dokumentowanie projektów programistycznych z pomocą AI
6. Wykrywanie błędów i podatności bezpieczeństwa aplikacji generowanych z użyciem AI.

[1] W przypadku ścieżki kształcenia prowadzącej do uzyskania kwalifikacji nauczycielskich uwzględnić również efekty uczenia się ze standardów kształcenia przygotowującego do wykonywania zawodu nauczyciela.

3.4 Metody dydaktyczne

Wykład: wykład z prezentacją multimedialną
Laboratorium: wykonywanie zadań przy komputerze

4. METODY I KRYTERIA OCENY

4.1 Sposoby weryfikacji efektów uczenia się

Symbol efektu	Metody oceny efektów uczenia się (np.: kolokwium, egzamin ustny, egzamin pisemny, projekt, sprawozdanie, obserwacja w trakcie zajęć)	Forma zajęć dydaktycznych (w, ćw, ...)
EK_01	kolokwium pisemne	w
EK_02	zaliczenie projektu	lab
EK_03	zaliczenie projektu	lab

4.2 Warunki zaliczenia przedmiotu (kryteria oceniania)

Zadania i pytania weryfikujące poszczególne efekty kształcenia formułowane są w taki sposób, że:

- na ocenę dst dotyczą znajomości lub umiejętności zastosowania metod i technologii,
- na ocenę db dotyczą porównywania pojawiających się metod i technologii,
- na ocenę bdb dotyczą optymalizowania jakości i efektywności metod i technologii dla konkretnych zagadnień.

Zasady uzyskania oceny końcowej:
Zaliczenie laboratorium następuje na podstawie zaliczenia wszystkich efektów weryfikowanych przez planowane w danym okresie metody weryfikacji. Przy tym zakłada się, że każda metoda weryfikacji dostarcza osobne oceny dla każdego z weryfikowanych przez nią efektów uczenia się. Jeśli dany efekt jest weryfikowany przez więcej niż jedną metodę, to ocena weryfikująca osiągnięcie tego efektu jest obliczana jako średnia arytmetyczna ocen uzyskanych w poszczególnych metodach weryfikowania tego efektu.
Student otrzymuje z zaliczenia ocenę **niedostateczny,** gdy metody weryfikacji wykażą, iż co najmniej jeden z efektów nie został osiągnięty (średnia ocena dla tego efektu jest niższa niż 3.0);

Student otrzymuje ocenę **dostateczny,** gdy przeciętnie każdy z efektów zostanie osiągnięty na poziomie co najmniej 3.0, ale chociaż jeden z efektów został osiągnięty na poziomie mniejszym od 3.75;

Student otrzymuje ocenę **dobry,** gdy przeciętnie każdy z efektów zostanie osiągnięty na poziomie co najmniej 3.75, ale chociaż jeden z efektów został osiągnięty na poziomie mniejszym od 4.75;

Student otrzymuje ocenę **bardzo dobry**, gdy przeciętnie każdy z efektów zostanie osiągnięty na poziomie co najmniej 4.75

5. CAŁKOWITY NAKŁAD PRACY STUDENTA POTRZEBNY DO OSIĄGNIĘCIA ZAŁOŻONYCH EFEKTÓW W GODZINACH ORAZ PUNKTACH ECTS

Forma aktywności	Średnia liczba godzin na zrealizowanie aktywności
Godziny z harmonogramu studiów	30
Inne z udziałem nauczyciela akademickiego (udział w konsultacjach, egzaminie)	5
Godziny niekontaktowe – praca własna studenta (przygotowanie do zajęć, egzaminu, napisanie referatu itp.)	40
SUMA GODZIN	75
SUMARYCZNA LICZBA PUNKTÓW ECTS	**3**

** Należy uwzględnić, że 1 pkt ECTS odpowiada 25-30 godzin całkowitego nakładu pracy studenta.*

6. PRAKTYKI ZAWODOWE W RAMACH PRZEDMIOTU

wymiar godzinowy	–
zasady i formy odbywania praktyk	–

7. LITERATURA

Literatura podstawowa: 1. Christoffer Noring, Anjali Jain, Marina Fernandez, Ayşe Mutlu, Ajit Jaokar: Programowanie wspomagane AI. Automatyzacja pracy programisty dzięki ChatGPT i GitHub Copilot, Helion, 2025. 2. Mark Winteringham, Nicola Martin (Przedmowa): Testowanie oprogramowania z wykorzystaniem generatywnej AI, Helion, 2025. 3. Leo Porter, Daniel Zingaro: Programuj ze sztuczną inteligencją. Twórz kod w Pythonie z wykorzystaniem GitHub Copilot i ChatGPT. Wydanie II, Helion, 2025.
Literatura uzupełniająca: 1. Beazley, D.: Python, zwięzłe kompendium dla programisty, Helion, 2022. 2. Mark Lutz: Python. Wprowadzenie. Wydanie VI, Helion, 2025.