

SYLABUS
DOTYCZY CYKLU KSZTAŁCENIA 2023-2025
 Rok akademicki 2024/2025

1. PODSTAWOWE INFORMACJE O PRZEDMIOCIE

Nazwa przedmiotu	<i>modelowanie i analiza systemów informatycznych</i>
Kod przedmiotu	
Nazwa jednostki prowadzącej kierunek	<i>Instytut Informatyki, Kolegium Nauk Przyrodniczych</i>
Nazwa jednostki realizującej przedmiot	<i>Instytut Informatyki, Kolegium Nauk Przyrodniczych</i>
Kierunek studiów	<i>informatyka</i>
Poziom studiów	<i>studia II stopnia</i>
Profil	<i>ogólnoakademicki</i>
Forma studiów	<i>stacjonarne</i>
Rok i semestr/y studiów	<i>rok II, semestr 3</i>
Rodzaj przedmiotu	<i>przedmiot kształcenia kierunkowego</i>
Język wykładowy	<i>polski</i>
Koordynator	<i>dr inż. Maksymilian Knap</i>
Imię i nazwisko osoby prowadzącej / osób prowadzących	<i>dr inż. Maksymilian Knap, mgr inż. Ewa Żestawska</i>

1.1. Formy zajęć dydaktycznych, wymiar godzin i punktów ECTS

Semestr (nr)	Wykł.	Ćw.	Konw.	Lab.	Sem.	ZP	Prakt.	Inne (jakie?)	Liczba pkt. ECTS
3	15			30					4

1.2. Sposób realizacji zajęć

wykład w formie kontaktu zdalnego z prowadzącym
 ćwiczenia laboratoryjne w formie tradycyjnej

1.3 Forma zaliczenia przedmiotu (z toku)

zaliczenie z oceną

2. WYMAGANIA WSTĘPNE

Student posiada ogólną wiedzę o zagadnieniach, których dotyczy inżynieria oprogramowania, zna etapy tworzenia oprogramowania, standardy dotyczące analizy oprogramowania, różne rodzaje testów; zna co najmniej dwie metody zarządzania projektami i różne rodzaje ryzyk związanych z realizacją projektów programistycznych.

Student potrafi tworzyć dokumentację techniczną, w tym diagramy UML, czytać dokumentację techniczną i na jej podstawie realizować oprogramowanie; potrafi korzystać z narzędzi do automatyzacji wytwarzania oprogramowania i jego dokumentacji.

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

3.1 Cele przedmiotu

C ₁	Poznanie metod analizy i modelowania wymagań użytkownika: strukturalnego i obiektowego
C ₂	Zdobycie pogłębionej wiedzy i umiejętności z zakresu stosowania nowoczesnych narzędzi do wytwarzania oprogramowania, projektowania, tworzenia, testowania, wdrażania i utrzymania aplikacji
C ₃	Uzyskanie wiedzy i umiejętności dotyczących prezentacji i promocji oprogramowania

3.2 Efekty uczenia się dla przedmiotu

EK (efekt uczenia się)	Treść efektu uczenia się zdefiniowanego dla przedmiotu	Odniesienie do efektów kierunkowych
EK_01	Student zna procesy, metody i narzędzia inżynierii oprogramowania, potrafi je zastosować w realizowanym projekcie informatycznym	K_Wo1, K_Wo2,
EK_02	Student zna znaczenie dokumentów stosowanych w trakcie przygotowania procesu tworzenia oprogramowania. Rozumie prawne uwarunkowania ich wykorzystania oraz potrafi na ich podstawie tworzyć specyfikacje techniczne	K_Wo1
EK_03	Student potrafi stosować różne metody analizy oprogramowania oraz nowoczesne – zwinne – podejście do zarządzania ich przebiegiem. Stosuje nowoczesne narzędzia wspomagające ww. procesy: analizy, tworzenia i testowania oprogramowania. Potrafi analizować i projektować systemy informatyczne oraz ich architekturę, stosując nowoczesne narzędzia i bieżące rozwiązania sprzętowe	K_Uo3

3.3 Treści programowe

A. Problematyka wykładu

Formułowanie, analiza i realizacja wymagań użytkownika
Architektury systemów informatycznych. Projektowanie architektury systemu na podstawie wymagań pozafunkcjonalnych
Analiza strukturalna i związane z nią diagramy UML
Analiza obiektowa i związane z nią diagramy UML
Analiza domenowa jako wariant analizy obiektowej
Testowanie tworzonego oprogramowania. Testy: jednostkowe, integracyjne i akceptacyjne
Narzędzia i środowiska wytwarzania oprogramowania. Narzędzia ciągłego dostarczania oprogramowania, procesy CI/CD

B. Problematyka ćwiczeń laboratoryjnych

Przygotowanie dokumentów RFI i RFP będących inicjatorem planowanego przedsięwzięcia informatycznego
Przygotowanie (na podstawie RFP) koncepcji projektu aplikacji: nazwa, cele, wymagane funkcjonalności, potencjalni klienci, prototyp interfejsu, itp.
Prezentacja koncepcji aplikacji na forum grupy. Wspólna dyskusja nad prezentowanym pomysłem - określenie zalet, wad, propozycje zmian; Wprowadzenie zmian w koncepcji uwzględniających wyniki dyskusji
Przygotowanie szczegółowych wymagań systemu. Sporządzenie dokumentacji projektowej wraz z diagramem przypadków użycia i wstępnym zestawem testów oprogramowania
Dobór technologii wykonania oraz architektury z uwzględnieniem nowoczesnych narzędzi tj. Docker i Kubernetes. Uzupełnienie dokumentacji o opis architektury i diagram wdrożenia
Analiza funkcjonalności z wykorzystaniem analizy obiektowej. Opracowanie diagramów klas, stanów (dla obiektów reaktywnych), sekwencji i aktywności. Uzupełnienie dokumentacji projektu o etap analizy
Projekt bazy danych, diagram ERD
Tworzenie struktury aplikacji z wykorzystaniem wybranego frameworka, implementacja założonych funkcjonalności oraz testów jednostkowych
Opracowanie finalnego zestawu testów akceptacyjnych do walidacji przedwdrożeniowej opracowywanego oprogramowania
Przygotowanie posteru oraz prezentacji promującej aplikację
Prezentacja projektu na forum publicznym

3.4 Metody dydaktyczne

Wykład: wykład z prezentacją multimedialną.

Ćwiczenia w laboratorium komputerowym: metoda projektów (wdrożeniowy, praktyczny), praca w grupach (rozwiązywanie zadań, dyskusja).

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	zaliczenie w formie testu	w
EK_02	zaliczenie w formie testu projekt informatyczny	w lab
EK_03	kolokwium, projekt informatyczny	w lab

4.2 Warunki zaliczenia przedmiotu (kryteria oceniania)

Zaliczenie z wykładu będzie realizowane w formie testu wiedzy. Próg zaliczenia zostaje ustalony na 50% poprawnych odpowiedzi + 1.

Zaliczenie z laboratorium odbędzie się na podstawie aktywności w trakcie zajęć, kolokwium oraz zrealizowanego projektu, w którym student rozwiązuje konkretny, zadany problem. Efekt EK_o4 jest zaliczany binarnie. Efekt EK_o3 jest zaliczany na ocenę. Uznaje się go za osiągnięty, gdy zarówno z kolokwium jak i z projektu student otrzyma ocenę co najmniej dostateczny. Udział oceny z kolokwium i projektu w końcowej ocenie z laboratorium wynosi po 50%.

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

Forma aktywności	Średnia liczba godzin na zrealizowanie aktywności
Godziny kontaktowe wynikające z harmonogramu studiów	45
Inne z udziałem nauczyciela akademickiego (udział w konsultacjach, egzaminie)	2
Godziny niekontaktowe – praca własna studenta (przygotowanie do zajęć, egzaminu, napisanie referatu itp.)	55
SUMA GODZIN	102
SUMARYCZNA LICZBA PUNKTÓW ECTS	4

6. PRAKTYKI ZAWODOWE W RAMACH PRZEDMIOTU

wymiar godzinowy	Nie dotyczy
zasady i formy odbywania praktyk	Nie dotyczy

7. LITERATURA

Literatura podstawowa:

1. Karl E Wieggers, Joy Beatty: Specyfikacja oprogramowania. Inżynieria wymagań. Wydanie III, Helion, 2014
2. Perdita Stevens: UML. Inżynieria oprogramowania. Wydanie II, Helion, 2007
3. Bogdan Bereza-Jarociński, Bolesław Szomański: Inżynieria oprogramowania. Jak zapewnić jakość tworząc aplikacje, Helion, 2008

Literatura uzupełniająca:

1. Computer Society: Guide to the Software Engineering Body of Knowledge (SWEBOK(R)): Version 3.0, Pierre Bourque, Richard E. Fairley (Editors), IEEE, 2014
2. Project Management Institute: A Guide to the Project Management Body of Knowledge (PMBOK® Guide)–Sixth Edition, PMI, 2019