KARTA KURSU (realizowanego w specjalności)

Inżynieria Oprogramowania

(nazwa specjalności)

Nazwa	**Projekt inżynierski**
Nazwa w j. ang.	*Engineering Project*

Koordynator	dr Roman Czapla	Zespół dydaktyczny
		dr inż. Magdalena Andrzejewska mgr Wojciech Baran dr Roman Czapla mgr Michał Frontczak dr Wojciech Gwizdała dr Beata Krzaczek mgr Łukasz Przybytek mgr Patryk Mazurek mgr inż. Patryk Mieczkowski dr inż. Grzegorz Sokal
Punktacja ECTS*	5	

Opis kursu (cele kształcenia)

Celem kursu jest przygotowanie studentów kierunku Informatyka do kompleksowej realizacji projektu inżynierskiego, który stanowi praktyczne podsumowanie wiedzy i umiejętności zdobytych podczas studiów. Zajęcia mają formę seminarium projektowego i koncentrują się na pełnym cyklu życia rozwiązania informatycznego: analizie problemu, projektowaniu architektury, implementacji, testowaniu, dokumentowaniu oraz prezentacji wyników.

Studenci pracują zespołowo (w 3-osobowych grupach) nad wybranym zagadnieniem praktycznym, tworząc działające oprogramowanie, system, usługę lub inne rozwiązanie informatyczne. W trakcie kursu uczą się planowania pracy projektowej, podejmowania decyzji technologicznych, zarządzania zadaniami, współpracy zespołowej oraz przygotowywania dokumentacji technicznej i użytkowej.

Kurs rozwija także umiejętności miękkie związane z komunikacją w zespole, prezentacją rezultatów oraz krytyczną analizą podejmowanych decyzji projektowych. Realizacja projektu przygotowuje studentów do pracy zawodowej w rolach programisty, administratora, projektanta systemów, testera lub inżyniera rozwiązań IT.

Warunki wstępne

Wiedza	Student posiada ogólną wiedzę z zakresu informatyki, w szczególności dotyczącą podstaw programowania, działania systemów komputerowych, baz danych i zasad projektowania oprogramowania. Rozumie sposób działania typowych narzędzi wykorzystywanych w pracy informatyka i potrafi korzystać z dokumentacji technicznej.
Umiejętności	Student potrafi posługiwać się co najmniej jednym językiem programowania, korzystać z systemów kontroli wersji (np. Git) oraz pracować w środowiskach programistycznych. Umie samodzielnie wyszukiwać informacje, analizować wymagania oraz rozwiązywać podstawowe problemy techniczne związane z implementacją i konfiguracją systemów.
Kursy	Wymagane jest wcześniejsze zaliczenie kursów kierunkowych przygotowujących do realizacji projektów informatycznych, w szczególności obejmujących podstawy programowania, inżynierię oprogramowania, bazy danych oraz inne przedmioty specjalistyczne wynikające z toku studiów.

Efekty uczenia się

Wiedza	Efekt uczenia się dla kursu	Odniesienie do efektów kierunkowych
	Po zakończeniu kursu student: **W01:**	S1_W01 S1_W02
Umiejętności	Efekt uczenia się dla kursu	Odniesienie do efektów kierunkowych
	Po zakończeniu kursu student: **U01:**	S1_U01 S1_U03 S1_U06 S1_U07 S1_U11 S1_U12
Kompetencje społeczne	Efekt uczenia się dla kursu	Odniesienie do efektów kierunkowych
	Po zakończeniu kursu student: **K01:**	S1_K03 S1_K04

Studia stacjonarne

Organizacja							
Forma zajęć	Wykład (W)	Ćwiczenia w grupach					
		A	K	L	S	P	E
Liczba godzin					60		

Studia niestacjonarne

Organizacja

Forma zajęć	Wykład (W)	Ćwiczenia w grupach					
		A	K	L	S	P	E
Liczba godzin					60		

Opis metod prowadzenia zajęć

Zajęcia mają formę seminarium projektowego i są prowadzone w sposób warsztatowy oraz konsultacyjny. Podczas spotkań studenci prezentują postępy prac, omawiają napotkane problemy oraz wspólnie z prowadzącym analizują możliwe rozwiązania techniczne. Każdy zespół projektowy pracuje nad własnym projektem inżynierskim, a prowadzący wspiera studentów w zakresie planowania prac, doboru technologii, projektowania architektury, implementacji i przygotowania dokumentacji.

Praca odbywa się w 3-osobowych zespołach, zgodnie z harmonogramem projektu, z wykorzystaniem repozytorium kodu i dobrych praktyk pracy zespołowej. Istotnym elementem zajęć są przeglądy etapowe, konsultacje indywidualne i zespołowe, analiza przypadków, a także prezentacje postępów prac projektowych.

Formy sprawdzania efektów uczenia się

	E – learning	Gry dydaktyczne	Ćwiczenia w szkole	Zajęcia terenowe	Praca laboratoryjna	Projekt indywidualny	Projekt grupowy	Udział w dyskusji	Referat\prezentacja etapowa	Praca pisemna (dokumentacja)	Egzamin ustny	Egzamin pisemny	Zadania problemowe
W01							x	x	x	x			
W02							x	x	x	x			
W03							x	x	x	x			
W04							x	x	x	x			
U01							x	x	x	x			
U02							x	x	x	x			
U03							x	x	x	x			
U04							x	x	x	x			
U05							x	x	x	x			
K01							x	x	x				
K02							x	x	x				
K03							x	x	x				

Kryteria oceny	Osiągnięcie minimalnych efektów uczenia się podanych powyżej uprawnia studentów do uzyskania oceny nie wyższej niż dostateczna. Warunkiem zaliczenia kursu jest: 1. realizacja kompletnego projektu inżynierskiego w zespole 3-osobowym, 2. oddanie dokumentacji projektowej (technicznej i użytkowej), 3. uczestnictwo w przeglądach etapowych i regularnych konsultacjach, 4. prezentacja i obrona wykonanego projektu. Ocena końcowa jest ustalana na podstawie: • jakości technicznej i kompletności projektu - 50 procent, • dokumentacji projektowej - 15 procent, • organizacji pracy zespołu, terminowości oraz realizacji harmonogramu - 20 procent, • prezentacji końcowej oraz umiejętności uzasadnienia przyjętych rozwiązań - 15 procent. Zaliczenie na ocenę dobrą lub bardzo dobrą otrzymuje student, który - poza spełnieniem wymogów oceny dostatecznej - dodatkowo:

	• samodzielnie i poprawnie rozwiązuje bardziej złożone problemy techniczne, • stosuje dobre praktyki inżynierskie w projektowaniu, implementacji i testowaniu, • świadomie dobiera technologie oraz potrafi uzasadnić wybór rozwiązań, • przygotowuje dokumentację i prezentację na wysokim poziomie jakościowym.

Uwagi	

Treści merytoryczne (wykaz tematów)

1. Wprowadzenie do projektu inżynierskiego
   - omówienie wymagań, zakresu i celów projektu,
   - zasady pracy zespołowej, harmonogram i organizacja pracy,
   - narzędzia wykorzystywane w projekcie (repozytorium kodu, dokumentacja, komunikacja).
2. Analiza problemu i określenie wymagań
   - identyfikacja problemu i celów projektu,
   - definiowanie wymagań funkcjonalnych i niefunkcjonalnych,
   - analiza możliwych rozwiązań i wybór podejścia projektowego.
3. Projektowanie rozwiązania informatycznego
   - projekt architektury systemu (moduły, komponenty, interfejsy, przepływy danych),
   - dobór języków programowania, bibliotek i narzędzi,
   - modelowanie danych i projekt struktury systemu.
4. Implementacja i integracja
   - praca nad modułami i integracja elementów systemu,
   - współpraca zespołowa przy wykorzystaniu systemów kontroli wersji,
   - rozwiązywanie problemów technicznych i iteracyjne ulepszanie rozwiązania.
5. Testowanie i weryfikacja rozwiązania
   - tworzenie przypadków testowych,
   - testy jednostkowe, integracyjne i funkcjonalne,
   - analiza błędów, poprawki oraz ocena jakości systemu.
6. Dokumentacja projektowa
   - struktura i sposób przygotowania dokumentacji technicznej,
   - dokumentacja użytkowa i opis wdrożenia,
   - przygotowanie raportu końcowego projektu.
7. Prezentacja projekt
   - przygotowanie prezentacji technicznej,
   - demonstracja działania projektu,
   - omówienie zastosowanych rozwiązań i podjętych decyzji projektowych.
8. Podsumowanie projektu
   - analiza przebiegu prac,
   - wnioski z realizacji projektu,
   - ocena współpracy i jakości pracy zespołu.

Wykaz literatury podstawowej

Wskazane przez prowadzącego rozdziały:

1. I. Sommerville, Inżynieria oprogramowania. Wydanie X, PWN, Warszawa 2020;
2. R. S. Pressman, B. R. Maxim, Software Engineering: A Practitioner’s Approach. 9th Edition, McGraw-Hill Education, New York 2019;
3. R. C. Martin, Czysta architektura. Struktura i design oprogramowania. Przewodnik dla profesjonalistów. Helion, Gliwice 2019;
4. M. Śmiałek, K. Rybiński, Inżynieria oprogramowania w praktyce. Od wymagań do kodu z językiem UML. Helion, Gliwice 2023;
5. **Inne publikacje, dokumentacje techniczne oraz materiały wskazane przez prowadzącego i studentów - zależnie od specyfiki realizowanego projektu.**

Wykaz literatury uzupełniającej

1. A. Harmel-Law, Tworzenie architektury oprogramowania. Wspieranie zespołów w podejmowaniu trafnych decyzji. Helion, Gliwice 2025;
2. D. Thomas, A. Hunt, Pragmatyczny programista. Od czeladnika do mistrza. Wydanie II. Helion, Gliwice 2021;
3. E. Gamma, R. Helm, R. Johnson, J. Vlissides, Wzorce projektowe. Elementy oprogramowania obiektowego wielokrotnego użytku. Helion, Gliwice 2010;
4. R. C. Martin, Czysty kod. Podręcznik dobrego programisty, Helion, Gliwice 2010.

Bilans godzinowy zgodny z CNPS (Całkowity Nakład Pracy Studenta) – **studia stacjonarne**

Liczba godzin w kontakcie z prowadzącymi	Wykład	0
	Seminarium (ćwiczenia, laboratorium itd.)	60
	Pozostałe godziny kontaktu studenta z prowadzącym	5
Liczba godzin pracy studenta bez kontaktu z prowadzącymi	Lektura w ramach przygotowania do zajęć	10
	Przygotowanie dokumentacji technicznej i użytkowej	15
	Praca własna i w grupie nad projektem	70
	Przygotowanie do prezentacji projektu	5
Ogółem bilans czasu pracy		150
Liczba punktów ECTS w zależności od przyjętego przelicznika		5

Bilans godzinowy zgodny z CNPS (Całkowity Nakład Pracy Studenta) – **studia niestacjonarne**

Liczba godzin w kontakcie z prowadzącymi	Wykład	0
	Seminarium (ćwiczenia, laboratorium itd.)	60
	Pozostałe godziny kontaktu studenta z prowadzącym	5
Liczba godzin pracy studenta bez kontaktu z prowadzącymi	Lektura w ramach przygotowania do zajęć	10
	Przygotowanie dokumentacji technicznej i użytkowej	15
	Praca własna i w grupie nad projektem	85
	Przygotowanie do prezentacji projektu	5
Ogółem bilans czasu pracy		150
Liczba punktów ECTS w zależności od przyjętego przelicznika		5