

WROCŁAWSKA WYŻSZA SZKOŁA INFORMATYKI STOSOWANEJ

KARTA OPISU PRZEDMIOTU

Wydział		Informatyki	
Kierunek		Informatyka	
Specjalność		Programowanie	
Semestr	V	Program studiów, dla którego obowiązuje sylabus	2023/2024
Stopień studiów	I		

Nazwa przedmiotu	Tworzenie aplikacji webowych			
Kod przedmiotu	TAW			
Łączna liczba godzin	18	Tryb	stacjonarny	niestacjonarny
Profil kształcenia	Ogólnoakademicki (A)		Praktyczny (P)	
Forma zajęć	laboratorium			
Język przedmiotu	polski			
Liczba punktów ECTS	2			

Prowadzący zajęcia	
Forma prowadzonych zajęć	Laboratorium
Wymiar zajęć	18 h
Stopień (tytuł) naukowy	
Imię	
Nazwisko	

Wymagania wstępne	Podstawowa znajomość języków programowania, takich jak Java lub C++. Umiejętność obsługi komputera i środowisk programistycznych. Znajomość podstawowych algorytmów i struktur danych.
Założenia i cele przedmiotu	Celem przedmiotu jest wprowadzenie studentów w proces tworzenia aplikacji webowych obejmujących integrację warstwy frontendowej i backendowej, zarządzanie stanem aplikacji, testowanie oraz optymalizację pod kątem wydajności i szybkości ładowania. Studenci nauczą się stosować architektoniczne wzorce (MVC, MVVM), wdrażać testy jednostkowe i integracyjne oraz korzystać z technik optymalizacji.
Metody dydaktyczne	<ol style="list-style-type: none"> 1. Prezentacje multimedialne. 2. Pokazy przykładowych rozwiązań problemów. 3. Rozwiązywanie zadań praktycznych.

Efekty uczenia się (odniesienie do charakterystyk poziomów Polskiej Ramy Kwalifikacji)		Odniesienie do efektów dla kierunku	Odniesienie do efektów uczenia się wg Polskiej Ramy Kwalifikacji
WIEDZA – absolwent zna i rozumie:	W01. Architektoniczne wzorce projektowe (MVC, MVP, MVVM) i ich wpływ na strukturę aplikacji webowej.	K_W06 K_W07	P6S_WG P6S_WG_INŻ

WROCŁAWSKA WYŻSZA SZKOŁA INFORMATYKI STOSOWANEJ

	<p>W02. Sposoby integracji warstwy frontendowej z backendem i bazą danych, zapewniające spójność danych i interfejsu użytkownika.</p> <p>W03. Metody zarządzania stanem aplikacji (Redux, Vuex, Context API) oraz ich wpływ na skalowalność i utrzymanie kodu.</p> <p>W04. Zasady testowania aplikacji webowych (testy jednostkowe, integracyjne, end-to-end) i narzędzia do ich realizacji.</p> <p>W05. Techniki optymalizacji ładowania i renderowania stron, w tym minifikację kodu, lazy loading i strategię cache'owania.</p>	<p>K_W10</p> <p>K_W20</p>	
UMIEJĘTNOŚCI – absolwent potrafi:	<p>U01. Zastosować wybrany wzorzec architektoniczny w celu strukturyzacji aplikacji webowej.</p> <p>U02. Zintegrować frontend z backendem, korzystając z API oraz zapewnić płynny przepływ danych między serwerem a interfejsem użytkownika.</p> <p>U03. Wykorzystać narzędzia do zarządzania stanem aplikacji, takie jak Redux czy Vuex, optymalizując przepływ danych.</p> <p>U04. Zaplanować i wykonać testy aplikacji webowych, automatyzując proces testowania w celu zwiększenia jakości oprogramowania.</p> <p>U05. Wdrożyć techniki optymalizacji wydajności, monitorować czasy ładowania i reagować na problemy ze skalowalnością.</p>	<p>K_U01</p> <p>K_U02</p> <p>K_U03</p> <p>K_U04</p> <p>K_U08</p> <p>K_U09</p> <p>K_U13</p> <p>K_U23</p> <p>K_U24</p>	<p>P6S_UW</p> <p>P6S_UW_INŻ</p> <p>P6S_UO</p> <p>P6S_KK</p> <p>P6S_UK</p>
KOMPETENCJE SPOŁECZNE – absolwent jest gotów do	<p>K01. Pracy w zespole, przyjmując w nim różne role.</p> <p>K02. Krytycznej oceny możliwości urządzeń oprogramowania i systemów dostępnych na rynku IT.</p> <p>K03. Ciągłego samokształcenia się w celu dostosowywania się do dynamicznie zmieniających się technologii.</p>	<p>K_K04</p> <p>K_K05</p> <p>K_K06</p>	<p>P6S_UO</p> <p>P6S_KR</p> <p>P6S_KK</p>

Lp.	Tematyka zajęć	Liczba godzin
Forma zajęć – laboratorium		
1	Architektura aplikacji webowych. MVC, MVP, MVVM, mikroserwisy.	3
2	Integracja frontend i backend. Łączenie interfejsu użytkownika z serwerem i bazą danych.	4
3	Zarządzanie stanem aplikacji. Redux, Vuex, Context API.	4

WROCŁAWSKA WYŻSZA SZKOŁA INFORMATYKI STOSOWANEJ

4	Testowanie aplikacji webowych. Testy jednostkowe, integracyjne i end-to-end.	4
5	Optymalizacja i wydajność. Techniki przyspieszania ładowania stron, optymalizacja kodu. Zaliczenie.	3

Forma i warunki zaliczenia przedmiotu	Wykonanie projektów. Częstkowe prezentacje, zdawanie raportów, obrona projektów.	
Metody weryfikacji efektów uczenia się		Nr efektu uczenia się z sylabusu
	Ocena projektów i częściowych prezentacji.	W01-W05, U01-U05, K01-K03

Literatura podstawowa	<ol style="list-style-type: none"> 1. R. C. Martin, <i>Czysty kod. Podręcznik dobrego programisty</i>, Helion, Gliwice 2010. 2. J. Roszkowski, <i>Analiza i projektowanie strukturalne</i>, Helion, Gliwice, 2004. 3. N. Wirth, <i>Algorytmy + struktury danych = programy</i>, WNT, Warszawa 2002. 4. A. Roman, <i>Testowanie i jakość oprogramowania. Modele, techniki, narzędzia</i>, Wydawnictwo Naukowe PWN, Warszawa 2015.
Literatura uzupełniająca	<ol style="list-style-type: none"> 1. S. Prata, <i>Język C++. Szkoła programowania</i>. Wydanie VI, Helion, Gliwice 2019. 2. B. Eckel, <i>Thinking in Java. Edycja polska</i>, Helion, Gliwice 2006.

Nakład pracy studenta	
	Liczba godzin
Zajęcia dydaktyczne	18
Przygotowanie się do zajęć	9
Studiowanie literatury	9
Udział w konsultacjach	2
Przygotowanie projektu / eseju / prezentacji itp.	22
Przygotowanie się do egzaminu / zaliczenia	-
Inne	-
ŁĄCZNY nakład pracy studenta w godz.	60
Liczba punktów ECTS	2