

# AKADEMIA TECHNICZNO-INFORMATYCZNA W NAUKACH STOSOWANYCH

## KARTA OPISU PRZEDMIOTU

<b>Wydział</b>		<b>Informatyki</b>	
<b>Kierunek</b>		<b>Informatyka</b>	
<b>Specjalność</b>		<b>Programowanie</b>	
<b>Semestr</b>	<b>V</b>	<b>Program studiów, dla którego obowiązuje sylabus</b>	<b>2025/2026</b>
<b>Stopień studiów</b>	<b>I</b>		

Nazwa przedmiotu	Tworzenie aplikacji webowych			
Kod przedmiotu	TAW			
Łączna liczba godzin	30	Tryb	stacjonarny	niestacjonarny
Profil kształcenia	Ogólnoakademicki (A)		Praktyczny (P)	
Forma zajęć	laboratorium			
Język przedmiotu	polski			
Liczba punktów ECTS	2			

<b>Prowadzący zajęcia</b>	
<b>Forma prowadzonych zajęć</b>	<b>Laboratorium</b>
<b>Wymiar zajęć</b>	<b>30 h</b>
<b>Stopień (tytuł) naukowy</b>	
<b>Imię</b>	
<b>Nazwisko</b>	

<b>Wymagania wstępne</b>	Podstawowa znajomość języków programowania, takich jak Java lub C++. Umiejętność obsługi komputera i środowisk programistycznych. Znajomość podstawowych algorytmów i struktur danych.
<b>Założenia i cele przedmiotu</b>	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.
<b>Metody dydaktyczne</b>	<ol style="list-style-type: none"> <li>1. Prezentacje multimedialne.</li> <li>2. Pokazy przykładowych rozwiązań problemów.</li> <li>3. Rozwiązywanie zadań praktycznych.</li> </ol>

<b>Efekty uczenia się (odniesienie do charakterystyk poziomów Polskiej Ramy Kwalifikacji)</b>		<b>Odniesienie do efektów dla kierunku</b>	<b>Odniesienie do efektów uczenia się wg Polskiej Ramy Kwalifikacji</b>
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

## AKADEMIA TECHNICZNO-INFORMATYCZNA W NAUKACH STOSOWANYCH

	<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 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 K_U02 K_U03 K_U04 K_U08 K_U09 K_U13 K_U23 K_U24</p>	<p>P6S_UW P6S_UW_INŻ P6S_UO P6S_KK 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 K_K05 K_K06</p>	<p>P6S_UO P6S_KR P6S_KK</p>

Lp.	Tematyka zajęć	Liczba godzin
<b>Tworzenie aplikacji webowych</b>		
1	Architektura aplikacji webowych. MVC, MVP, MVVM, mikroserwisy.	6
2	Integracja frontend i backend. Łączenie interfejsu użytkownika z serwerem i bazą danych.	8
3	Zarządzanie stanem aplikacji. Redux, Vuex, Context API.	6

## AKADEMIA TECHNICZNO-INFORMATYCZNA W NAUKACH STOSOWANYCH

4	Testowanie aplikacji webowych. Testy jednostkowe, integracyjne i end-to-end.	6
5	Optymalizacja i wydajność. Techniki przyspieszania ładowania stron, optymalizacja kodu. Zaliczenie.	4

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

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

Nakład pracy studenta	
	Liczba godzin
Zajęcia dydaktyczne	30
Przygotowanie się do zajęć	5
Studiowanie literatury	5
Udział w konsultacjach	2
Przygotowanie projektu / eseju / prezentacji itp.	18
Przygotowanie się do egzaminu / zaliczenia	-
Inne	-
<b>ŁĄCZNY nakład pracy studenta w godz.</b>	<b>60</b>
<b>Liczba punktów ECTS</b>	<b>2</b>