



Karta przedmiotu

Nazwa i kod przedmiotu	High Availability Distributed Systems, PG_00063936						
Kierunek studiów	Elektronika i telekomunikacja (studia w jęz. angielskim), Informatyka (studia w jęz. angielskim), Automatyka, cybernetyka i robotyka (studia w jęz. angielskim)						
Data rozpoczęcia studiów	luty 2026 r.		Rok akademicki realizacji przedmiotu		2025/2026		
Poziom kształcenia	II stopnia		Grupa zajęć		Grupa zajęć fakultatywnych Grupa zajęć specjalnościowych Grupa zajęć powiązanych z prowadzonymi badaniami naukowymi w dziedzinie nauki związanej z kierunkiem - profil ogólnoakademicki		
Forma studiów	stacjonarne		Sposób realizacji		na uczelni		
Rok studiów	1		Język wykładowy		angielski		
Semestr studiów	1		Liczba punktów ECTS		3.0		
Profil kształcenia	ogólnoakademicki		Forma zaliczenia		zaliczenie		
Jednostka prowadząca	Wydziały Politechniki Gdańskiej -> Wydział Elektroniki, Telekomunikacji i Informatyki -> Katedra Architektury Systemów Komputerowych						
Imię i nazwisko wykładowcy (wykładowców)	Odpowiedzialny za przedmiot		dr Paweł Kozyra				
	Prowadzący zajęcia z przedmiotu		dr Paweł Kozyra				
Formy zajęć	Forma zajęć	Wykład	Ćwiczenia	Laboratorium	Projekt	Seminarium	RAZEM
	Liczba godzin zajęć	15.0	0.0	0.0	15.0	0.0	30
	W tym liczba godzin zajęć na odległość: 0.0						
	Adresy kursu na platformie eNauczanie: Moodle ID: 4796 High Availability Distributed Systems/Rozproszone systemy o wysokiej dostępności https://enauczanie.pg.edu.pl/2025/course/view.php?id=4796						
Aktywność studenta i liczba godzin pracy	Aktywność studenta	Udział w zajęciach dydaktycznych, objętych planem studiów	Udział w konsultacjach	Praca własna studenta	RAZEM		
	Liczba godzin pracy studenta	30	9.0	36.0	75		
Cel przedmiotu	Celem przedmiotu jest wprowadzenie studentów w tematykę wytwarzania aplikacji rozproszonych, a także systemów gromadzenia i przetwarzania danych w sposób rozproszony. Ponadto w ramach przedmiotu studenci zostaną zaznajomieni z platformami wdrożeniowymi stosowanymi w przemyśle, których zadaniem jest zarządzanie zbiorami rozproszonych komponentów aplikacyjnych.						

Efekty uczenia się przedmiotu	Efekt kierunkowy	Efekt z przedmiotu	Sposób weryfikacji i oceny efektu
	[K7_W11] zna i rozumie w pogłębionym stopniu ogólne zasady tworzenia i rozwoju form indywidualnej przedsiębiorczości oraz ekonomiczne, prawne i inne uwarunkowania różnych rodzajów działań związanych z nadaną kwalifikacją, w tym zasady ochrony własności przemysłowej i prawa autorskiego	potrafi zastosować zasady indywidualnej przedsiębiorczości	[SW1] Ocena wiedzy faktograficznej
	[K7_W10] zna i rozumie w pogłębionym stopniu podstawowe procesy zachodzące w cyklu życia urządzeń, obiektów i systemów technicznych oraz metody wspomagania procesów i funkcji, specyficzne dla kierunku studiów	potrafi zastosować wiedzę dotyczącą cyklu życia obiektów technicznych	[SW1] Ocena wiedzy faktograficznej
	[K7_U12] potrafi w pogłębionym stopniu analizować działanie elementów, układów i systemów związanych z kierunkiem studiów oraz mierzyć ich parametry i badać charakterystyki techniczne, a także planować i przeprowadzać eksperymenty związane z kierunkiem studiów, w tym symulacje komputerowe, interpretować uzyskane wyniki i wyciągać wnioski	potrafi skonfigurować środowisko równoległe, uruchamiać i profilować wykonanie aplikacji równoległych	[SU1] Ocena realizacji zadania
[K7_W03] zna i rozumie w pogłębionym stopniu budowę i zasady działania komponentów i systemów związanych z kierunkiem studiów, w tym teorie, metody i złożone zależności między nimi oraz wybrane zagadnienia szczegółowe – właściwe dla programu kształcenia	Student zna i opisuje różne architektury wytwarzania aplikacji. Zna różnice, zalety i wady ze stosowania monolitycznych architektur warstwowych i docelowych architektur rozproszonych.	[SW1] Ocena wiedzy faktograficznej	
Treści przedmiotu	<p>Treści przedmiotu - wykład Skalowalność aplikacji, deployment; Architektury aplikacji rozproszonych (monolit -> mikro usług (CQRS/ Event Sourcing/Saga); konteneryzacja usług -> docker, docker-compose, docker swarm, kubernetes, wdrażaniu i utrzymaniu aplikacji rozproszonej -> monitorowanie (klastry/chmury obliczeniowe OpenStack / AWS) Monitorowanie -> Sentry / Jaeger / Prometheus + Grafana / Load balancery / Systemy kolejkowe; Narzędzia do testowania obciążenia Locust.io / Jmeter Rozproszone systemy plikowe HDFS (Hive) / IPFS Rozproszone bazy danych (Hbase / Neo4j, ArangoDB) Blockchain -> Bitcoin / Ethereum / Stellar / GRP (grafowa) Rozproszone środowisko obliczeniowe (Apache Spark/ YARN -> JupyterLab -> PySparku -> .net context submit)</p> <p>Treści przedmiotu - projekt Definiowanie koordynatora Kubernetes. Wykonywanie operacji za pomocą kubectl. Implementowanie i konfigurowanie wielu usług. Lokalne wdrażanie całego systemu. Konfigurowanie rejestru Dockera. Tworzenie i automatyczne skalowanie klastra. GitOps. Kontrolowanie klastra Kubernetes. Konfigurowanie GitHub. Konfigurowanie wielu środowisk. Skalowanie przepływu pracy. Konfigurowanie logów i metryk. Obsługa zależności usług.</p>		
Wymagania wstępne i dodatkowe	<ol style="list-style-type: none"> 1. Student musi posiadać wiedzę i umiejętności programistyczne w technologiach .net lub java. 2. Student musi rozumieć metody komunikacji stosowane w Internecie 3. Student musi posiadać wiedzę i umiejętności w zakresie wdrażania aplikacji na serwer docelowy w wersji skonteneryzowanej. 		
Sposoby i kryteria oceniania osiągniętych efektów uczenia się	Sposób oceniania (składowe)	Próg zaliczeniowy	Składowa ocena końcowej
	wykład	50.0%	50.0%
	projekt	50.0%	50.0%
Zalecana lista lektur	Podstawowa lista lektur	<ol style="list-style-type: none"> 1. Cloud Native DevOps with Kubernetes, John Arundel, Justin Domingus. 2. Kubernetes Patterns: Reusable Elements for Designing Cloud-Native Applications, Bilgin Ibryam, Roland Huß. 3. KUBERNETES: A Simple Guide to Master Kubernetes for Beginners and Advanced Users (2020 Edition), Brian Docker. 4. Hands-On Docker for Microservices with Python: Design, deploy, and operate a complex system with multiple microservices using Docker and Kubernetes, Jaime Bueta. 5. gRPC: Up and Running: Building Cloud Native Applications with Go and Java for Docker and Kubernetes, Kasun Indrasiri, Danesh Kuruppu. 	

	Uzupełniająca lista lektur	<ol style="list-style-type: none"> 1. The Kubernetes Book, Nigel Poulton. 2. Hands-On Microservices with C# and .NET Core 3: Refactor you monolith architecture into microservices using Azure, 3rd Edition, Gaurav Aroraa, Ed Price. 3. Pro ASP.NET Core 3: Develop Cloud-Ready Web Applications Using MVC, Blazor, and Razor Pages, Adam Freeman. 4. Practical Microservices Architectural Patterns - Event-Based Java Microservices with Spring Boot and Spring Cloud, Binildas Christudas. 5. Monolith to Microservices: Evolutionary Patterns to Transform Your Monolith, Sam Newman. 6. Practical Microservices: Build Event-Driven Architectures with Event Sourcing and CQRS, Ethan Garofolo. 7. Architecting Modern Data Platforms, Jan Kunigk, Ian Buss, Paul Wilkinson & Lars George. 8. Advanced Analytics with Spark, Sandy Ryza, Uri Laserson, Sean Owen & Josh Wills. 9. Big Data Analytics with Hadoop 3, Sridhar Alla. 10. Modern Big Data Processing with Hadoop, V. Naresh Kumar Prashant Shindgikar.
	Adresy eZasobów	
Przykładowe zagadnienia/ przykładowe pytania/ realizowane zadania	<ol style="list-style-type: none"> 1. Wyjaśnij czym jest CNCF 2. Czym jest Infrastrcture as a Code 3. RDD vs DataFrame 4. HDFS vs IPFS 5. pySpark vs python 6. Rola Yarna 	
Zajęcia praktyczne w ramach przedmiotu	Nie dotyczy	

Dokument wygenerowany elektronicznie. Nie wymaga pieczęci ani podpisu.