POLSKO-JAPOŃSKA AKADEMIA TECHNIK KOMPUTEROWYCH

SYLABUS PRZEDMIOTU

Nazwa przedmiotu:	**Projekt**
Kod przedmiotu:	PRO
Kierunek / Profil:	Informatyka / praktyczny
Tryb studiów:	stacjonarny
Rok / Semestr:	3 / 5
Charakter:	obowiązkowy
Odpowiedzialny:	dr hab. inż. Marta Łabuda
Wersja z dnia:	19.02.2026

1. Godziny zajęć i punkty ECTS

Wykłady	Ćwiczenia	Laboratori	Z prowa-dzącym	Praca własna	Łącznie	ECTS
—	—	45 h	45 h	30 h	75 h	**3**

2. Forma zajęć

Forma zajęć	Sposób zaliczenia
Projekt	Zaliczenie z oceną

3. Cel dydaktyczny

Celem zajęć jest zwiększenie umiejętności związanych z podstawowymi zagadnieniami i praktyką planowania i prowadzenia projektu informatycznego, w szczególności – projektu mającego na celu wytworzenie produktu programowego w technologii obiektowej oraz zdobycie przez studentów doświadczeń pracy w zespole. Zajęcia projektowe są skorelowane z wykładem przedmiotu Budowa i integracja systemów (BYT), zapewniając kompleksowość przedstawianych zagadnień inżynierii oprogramowania i dając studentom możliwość samodzielnego, metodycznego przeprowadzenia niedużych projektów informatycznych

– od zdefiniowania problemu, poprzez dobór strategii i zaplanowanie jego rozwiązania do wykonania praktycznych elementów produktu, z użyciem wzorców i komponentów projektowych oraz przeprowadzeniem testów systemowych i walidacyjnych, a także ich udokumentowaniem. Wzorcowym projektem jest wykorzystanie technologii obiektowej w cyklu kaskadowym lub komponentowym, oraz zwinne strategie realizacji projektów. Istotnymi czynnikami doboru tematów są praktyczna przydatność, a także, w miarę możliwości, zapewnienie współpracy studentów z rzeczywistym klientem projektu. Tematy są prowadzone są w 2-4 osobowych grupach.

4. Przedmioty wprowadzające

Przedmiot

- Programowanie 2

- Relacyjne bazy danych

- Umiejętność programowania obiektowego na poziomie podstawowym

5. Treści programowe

1. Projekt
2. Konstrukcja aplikacji w oparciu o przyjętą metodykę, wzorce (analityczne, projektowe) frameworki i komponenty + dokumentacja projektu
3. 1. Wprowadzenie, zasady zaliczenia przedmiotu. Założenia prac projektowych, ich prowadzenia i dokumentowania. Zapoznanie się z dostępnym środowiskiem. Dyskusja nad sformułowaniami projektów i grup projektowych. Opis projektu, założenia projektowe Analiza dokumentacji przykładowych rozwiązań projektowych.
4. 2. Praca nad projektem: Identyfikacja problemu. Kontekst i udziałowcy problemu; klient projektu. Wzbogacony wizerunek, definicja i wizja systemu, zakres systemu, rodzaj produktu. Przygotowanie Dokumentu Założeń Wstępnych
5. 3. Specyfikacja wymagań na system, podział i analiza wymagań. User stories. Studium wykonalności projektu; analiza organizacyjna, mikro i makrootoczenie projektu; dyskusja wariantów rozwiązania. Dobór strategii realizacji projektu w zależności od rodzaju produktu. Debata
6. 4. Modelowanie systemu (przypadki użycia, diagramy sekwencji i/lub czynności). Logiczny diagram klas
7. 5. Modelowanie systemu. Architektura systemu: podział na podsystemy i komponenty. Projekt bazy danych.
8. 6. Harmonogram prac implementacyjnych, integracji, testowania oraz dodatkowej dokumentacji z podziałem na członków zespołu i ich role.
9. 7. Konfiguracja środowiska programistycznego i technologicznego.
10. 8. Prace implementacyjne. Realizacja zadań projektowych wg. Harmonogramu.
11. 9. Prace implementacyjne. Realizacja zadań projektowych wg. Harmonogramu.
12. 10. Prace implementacyjne. Wstępna ocena postępów, analiza zagrożeń.
13. 11. Implementacja i integracja oprogramowania.
14. 12. Implementacja i integracja. Inspekcja kodu. Analiza bezpieczeństwa.
15. 13. Ocena i poprawa jakości oprogramowania. Scenariusze testów. Testowanie i walidacja oprogramowania; przygotowanie raportu z przeprowadzonych testów.
16. 14. Finalizacja prac implementacyjnych i dokumentacyjnych; odbiór techniczny, przygotowanie raportu końcowego.
17. 15. Prezentacja, omówienie i podsumowanie projektów. Zaliczenie przedmiotu.

6. Efekty kształcenia

Wiedza

- Student zna i rozumie zaawansowane pojęcia z zakresu zagadnień inżynierii oprogramowania, standardów i kształtu cykli wytwórczych oraz ewolucji oprogramowania; zna podstawy zarządzania przedsięwzięciem programistycznym i rozumie problem jakości oprogramowania; rozumie rolę modelowania i ma szczegółową wiedzę o obiektowym wytwarzaniu oprogramowania i notacji UML, zna i rozumie zasady

korzystania z wzorców programowych i standardowych API; ma wiedzę o typowych narzędziach i środowiskach wspomagających;
- Student zna i rozumie podstawowe pojęcia z zakresu kluczowych zagadnień inżynierii wymagań, rozumie potrzebę systematycznego budowania i pielęgnacji specyfikacji wymagań; ma szczegółową wiedzę dotyczącą ich specyfikacji, analizy i modelowania z użyciem dostępnych narzędzi.
- Student zna i rozumie kluczowe pojęcia z zakresu walidacji i testowania oprogramowania.
- Student zna i rozumie pojęcia z zakresu planowania przedsięwzięcia informatycznego, wstępnej oceny ekonomicznej, aspektów społecznych oraz analizy wykonalności.

Umiejętności

- Student potrafi pozyskiwać specjalistyczne informacje z literatury, baz danych, systemów patentowych, Internetu oraz innych źródeł, w języku polskim i angielskim w zakresie informatyki; potrafi dokonywać oceny, krytycznej analizy i syntezy tych informacji, a także wyciągać wnioski oraz formułować i uzasadniać opinie.
- Student potrafi przygotować w języku polskim i języku obcym dobrze udokumentowane opracowanie problemów z zakresu informatyki lub dokumentację realizacji zadania inżynierskiego.
- Student potrafi pracować w zespole; potrafi oszacować czas i koszty potrzebne na realizację zleconego zadania; potrafi planować, opracować i realizować harmonogram prac zapewniający dotrzymanie terminów.
- Student potrafi analizować i wyjaśniać obserwowane zjawiska; tworzyć i weryfikować modeli świata rzeczywistego oraz posługiwać się nimi w celu predykcji zdarzeń i stanów; potrafi posłużyć się właściwe dobranymi środowiskami programistycznymi, symulatorami oraz narzędziami wspomagania komputerowego do symulacji, projektowania i analizy prostych systemów.
- Student potrafi wyspecyfikować, zaprojektować, zaimplementować, przetestować oraz debuggować program; potrafi korzystać z bibliotek, środowisk programistycznych, integrujących i uruchomieniowych.
- Student potrafi zaplanować i zrealizować prosty system oprogramowania zgodnie z metodyką obiektową, posługując się wzorcami programowymi, standardami i dobrymi praktykami programistycznymi; potrafi dobrać model procesu wytwarzania oprogramowania do specyfiki przedsięwzięcia, a także dobrać narzędzia wspomagające budowę oprogramowania.
- Student potrafi zaplanować i przeprowadzić procesy pozyskiwania, analizy, specyfikacji i modelowania wymagań wobec oprogramowania oraz ich pielęgnacji.
- Student potrafi dokonać przeglądu projektu oprogramowania i poprawić jego jakość.
- Student potrafi zaplanować i przeprowadzić proces integracji, oceny i realizacji planu testowania oraz dokonać diagnozy defektów.
- Student potrafi zaplanować i wytworzyć podstawowe dokumenty związane z realizacją prostego przedsięwzięcia informatycznego, wstępnie ocenić efekty ekonomiczne i społeczne przedsięwzięcia oraz ich wpływ na udziałowców;
- Student potrafi zaplanować i przeprowadzić proces instalacji i uruchomienia całości prostego systemu (system operacyjny, baza danych, aplikacja, oprogramowanie

współdziałające).

Kompetencje społeczne

- Student jest gotów do współdziałań i współpracy w zespole, przyjmując różne role, m.in. zamawiającego, klienta, analityka, projektanta, wykonawcy.
- Student jest gotów do ustalania priorytetów zadań.

7. Kryteria oceny

- Wykład (powiązany treściami z przedmiotem BYT):
- dokumentowanie produktów projektowych
- burza mózgów
- praca grupowa nad projektem z wykorzystaniem nowoczesnych technik i narzędzi informatycznych
- Kryteria oceny
- prezentacja projektu i dokumentacji
- obrona projektu
- raport z wykonanego zadania
- ocena sporządzonego dokumentu: szablony dokumentów, raporty
- ocena sporządzonego oprogramowania: weryfikacja powstałych komponentów oraz produktu końcowego
- ocena doboru i wykonania rozwiązań projektowych i powiązanej dokumentacji
- Uwaga: Ocena na podstawie oceny produktu realizowanego projektu oraz ocen częściowych, otrzymywanych za wykonanie udokumentowanych komponentów projektowych

8. Metody dydaktyczne

Wykład, laboratoria, praca własna studenta.

9. Literatura

Podstawowa:

- 1. Richard Hundhausen Profesjonalne wytwarzanie oprogramowania z zastosowaniem Scruma i usług Azure DevOps, Prima 2021
- 2. Krzysztof Sacha, Inżynieria Oprogramowania, PWN 2022
- 3. Ian Sommerville, Inżynieria Oprogramowania, wyd.10, PWN 2020
- 4. Max Kanat-Alexander, Zrozumieć oprogramowanie, Helion 2019
- 5. Robert C. Martin, Czysta architektura. Struktura i design oprogramowania. Przewodnik dla profesjonalistów, Helion 2022

- 6. Mike Cohn, Agile. Metodyki zwinne w planowaniu projektów, Helion 2019
- 7. Robert C. Martin, Zwinnewytwarzanie oprogramowania. Najlepsze zasady, wzorce i praktyki, Helion 2017

Uzupełniająca:

- 1. Bernd Bruegge, Allen H. Dutoit, Inżynieria oprogramowania w ujęciu obiektowym UML, wzorce projektowe i Java Helion 2011
- 2. Keeling Michael,Zostań architektem oprogramowania, PWN 2019
- 3. Piotr Gaczkowski, Adrian Ostrowski, Architektura oprogramowania bez tajemnic, Helion 20224.
- 4. Arnon Axelrod, Automatyzacja testów. Kompletny przewodnik dla testerów oprogramowaniaPWN 2022
- 5. Strony domowe narzędzi i technologii. Źródła internetowe.