KARTA OPISU PRZEDMIOTU - SYLABUS

Nazwa przedmiotu
Frontend Development [S2Inf1-GiTI>FDEV]

Przedmiot

Kierunek studiów
Informatyka

Rok/Semestr
1/1

Studia w zakresie (specjalność)
Gry i technologie internetowe

Profil studiów
ogólnoakademicki

Poziom studiów
drugiego stopnia

Język oferowanego przedmiotu
polski

Forma studiów
stacjonarne

Wymagalność
obligatoryjny

Liczba godzin

Wykład
30

Laboratorium
30

Inne (np. online)
0

Ćwiczenia
0

Projekty/seminaria
0

Liczba punktów ECTS

4,00

Koordynatorzy

dr inż. Marcin Borowski
marcin.borowski@put.poznan.pl

Wykładowcy

Wymagania wstępne

Student rozpoczynający ten przedmiot powinien posiadać podstawową wiedzę z programowania strukturalnego oraz obiektowego, programowania z wykorzystaniem schematu MVC, podstawowej wiedzy na temat technologii internetowych (HTML, CSS, JS), oraz podstawową wiedzę z zakresu projektowania baz danych. Powinien posiadać umiejętność rozwiązywania podstawowych problemów związanych z procesem projektowania systemów informatycznych oraz umiejętność pozyskiwania informacji ze wskazanych źródeł. Powinien również rozumieć konieczność poszerzania swoich kompetencji / mieć gotowość do podjęcia współpracy w ramach zespołu. Ponadto w zakresie kompetencji społecznych student musi prezentować takie postawy jak uczciwość, odpowiedzialność, wytrwałość, ciekawość poznawcza, kreatywność, kultura osobista, szacunek dla innych ludzi.

Cel przedmiotu

1. Przekazanie studentom podstawowej wiedzy dotyczącej technologii wykorzystywanych przy budowie aplikacji webowych w szczególności technik frontendowych, w zakresie podejść do projektowania, doboru technologii oraz implementacji (w tym również rozwiązań przeznaczonych dla urządzeń mobilnych). 2. Rozwijanie u studentów umiejętności rozwiązywania problemów związanych z projektowaniem aplikacji internetowych również działających w czasie rzeczywistym (reaktywność), wykorzystywanie frameworków, bibliotek oraz innych narzędzi wspierających budowę serwisów. 3. Kształtowanie u studentów umiejętności pracy zespołowej jak również samodzielności w rozwiązywaniu problemów.

Przedmiotowe efekty uczenia się

Wiedza:
• ma zaawansowaną i pogłębioną wiedzę z zakresu technologii internetowych, podstaw teoretycznych ich budowania oraz metod, narzędzi i środowisk programistycznych wykorzystywanych do ich implementacji
• ma zaawansowaną wiedzę szczegółową dotyczącą technologii frontendowych
ma zaawansowaną i szczegółową wiedzę o procesach zachodzących w cyklu życia aplikacji internetowych oraz przesyłu informacji
• zna zaawansowane metody, techniki i narzędzia stosowane przy rozwiązywaniu złożonych zadań inżynierskich podczas budowy aplikacji internetowych

Umiejętności:
• potrafi ocenić przydatność i możliwość wykorzystania nowych osiągnięć oraz nowych produktów informatycznych (dedykowane narzędzia, dedykowane języki itd.)
• potrafi — przy formułowaniu i rozwiązywaniu zadań inżynierskich — integrować wiedzę z różnych obszarów informatyki oraz zastosować podejście systemowe, uwzględniające także aspekty pozatechniczne
• potrafi posługiwać się technikami informacyjno-komunikacyjnymi wykorzystywanymi przy realizacji aplikacji frontendowych
• potrafi ocenić przydatność metod i narzędzi służących do rozwiązania zadania inżynierskiego, polegającego na budowie lub ocenie systemu informatycznego lub jego składowych, w tym dostrzec ograniczenia tych metod i narzędzi - w efekcie potrafi dobrać odpowiednią technologię wytwarzania aplikacji w zależności od wymagań
• potrafi - zgodnie z zadaną specyfikacją - zaprojektować i zaimplementować złożone aplikacje internetowe - co najmniej w części, używając właściwych metod, technik i narzędzi, w tym przystosowując do tego celu istniejące lub opracowując nowe narzędzia

Kompetencje społeczne:
• rozumie, że w informatyce wiedza i umiejętności bardzo szybko stają się przestarzałe, w szczególności technologie internetowe
• rozumie potrzeby wykorzystywania najnowszych osiągnięć techniki oraz zna przykłady i rozumie przyczyny wadliwie działających systemów informatycznych, które doprowadzić mogą do poważnych strat finansowych, wizerunkowych lub społecznych

Metody weryfikacji efektów uczenia się i kryteria oceny

Efekty uczenia się przedstawione wyżej weryfikowane są w następujący sposób:

Ocena formująca
1. wykład - na podstawie aktywności podczas interaktywnych części wykładów;
2. laboratorium - na podstawie oceny bieżącego postępu realizacji zadań;
Ocena podsumowująca
1. wykład - ocenę samodzielnie przygotowanej przez studenta prezentacji dotyczącej wybranej technologii, biblioteki lub frameworku wykorzystywanego przy budowie aplikacji webowych;
2. laboratorium - ocenę i obronę przez studentów przygotowanych zadań - 4 mini-projektów;

Przy wystawianiu oceny końcowej, student może uzyskać podwyższenie oceny za:
- omówienie dodatkowych aspektów prezentowanych zagadnień, nie prezentowanych na zajęciach;
- wykorzystania umiejętności i wiedzy spoza programu studiów do rozwiązywania realizowanych zadań;
- pomoc w doskonaleniu materiałów dydaktycznych związanych z przedmiotem;

Treści programowe

Wykład:
Program wykładu obejmuje następujące zagadnienia:
Protokół komunikacji HTTP. Wprowadzenie do technologii node.js. Budowa prostych serwerów popularnych usług sieciowych (echo, chat, http). Wprowadzenie do frameworka Express.js. Wprowadzenie do frameworka Angular. Wprowadzenie do frameworka React. Wprowadzenie do frameworka SvelteKit. Omówienie narzędzi wspomagających takich jak Grunt, Gulp, Webpack, Vite, SASS, Less i bibliotek TailwindCSS, Bootstrap. Języki definiowania szablonów aplikacji i komponentów EJS, HAML, JSX.
Laboratorium:
Zajęcia laboratoryjne prowadzone są w formie piętnastu 2-godzinnych ćwiczeń, odbywających się w laboratorium. Ćwiczenia realizowane są samodzielnie przez studentów. Program laboratorium obejmuje następujące zagadnienia:
Przygotowanie szablonów stron i widoków komponentów z wykorzystaniem HTML5, CSS, LESS, SASS oraz wykorzystania frameworków i bibliotek komponentów (m.in. Bootstrap, SemanticUI, Tailwindcss). Instalacja i konfiguracja środowiska node.js. Uruchamianie aplikacji napisanych w node.js. Proste serwery usług. Realizacja prostych aplikacji zrealizowane w frameworku Express.js z Angular i MongoDB, React, Svelte. Przykłady wykorzystania narzędzi wspomagających oraz modułów dla node.js: Vite, Nodemon, Skeleton itp.

Tematyka zajęć

Wykład:
- Protokół komunikacyjny HTTP.
- Wprowadzenie do technologii Node.js.
- Budowa prostych serwerów i popularnych usług webowych (echo, chat, HTTP).
- Wprowadzenie do frameworka Express.js.
- Wprowadzenie do frameworków Angular, React i SvelteKit.
- Narzędzia wspierające: Grunt, Gulp, Webpack, Vite, SASS, LESS, TailwindCSS, Bootstrap.
- Języki szablonów i definicji komponentów: EJS, HAML, JSX.
Laboratorium:
- Piętnaście 2-godzinnych zajęć w laboratorium, realizowanych samodzielnie przez studentów.
Tematy:
- Przygotowanie szablonów stron i widoków komponentów z użyciem HTML5, CSS, LESS, SASS, oraz bibliotek i frameworków (np. Bootstrap, SemanticUI, TailwindCSS).
- Instalacja i konfiguracja środowiska Node.js.
- Uruchamianie aplikacji napisanych w Node.js.
- Budowa prostych serwerów usługowych.
- Implementacja prostych aplikacji z użyciem Express.js, Angular, MongoDB, React, Svelte.
- Przykłady użycia narzędzi i modułów wspierających Node.js: Vite, Nodemon, Skeleton, itp.

Metody dydaktyczne

Wykład: prezentacja multimedialna, ilustrowana przykładami podawanymi na tablicy.
Ćwiczenia laboratoryjne: prezentacja multimedialna, prezentacja ilustrowana przykładami podawanymi na tablicy, live coding oraz wykonanie zadań podanych przez prowadzącego - ćwiczenia praktyczne.

Literatura

Dokumentacje techniczne wymienionych narzędzi dostępne w internecie

Bilans nakładu pracy przeciętnego studenta

	Godzin	ECTS
Łączny nakład pracy	100	4,00
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	60	2,50
Praca własna studenta (studia literaturowe, przygotowanie do zajęć laboratoryjnych/ćwiczeń, przygotowanie do kolokwiów/egzaminu, wykonanie projektu)	40	1,50