aktualizacja: 2020-03-21, zmiany uwzględniają konieczność prowadzenia zajęć w trybie zdalnym

zespoły 1-3 osobowe

Temat ogólny:

aplikacja pobierająca dane z serwisu zewnętrznego i wyświetlająca je po odpowiednim przetworzeniu

wymagane:

1. możliwość parametryzowania zapytań (np. ustalenie miejscowości dla pogodynki wg wpisu użytkownika i odczytu GPS)
2. nieblokujące się GUI (przetwarzanie długich operacji w wątkach, wskaźnik postępu, odporność na brak połączenia sieciowego)
3. co najmniej 3 widoki/okienka, w tym co najmniej dwa zależne od siebie (np. widok master-detail)
4. dostosowanie do różnych rozmiarów urządzeń (tablety i telefony)
5. odporność na błędy użytkownika (w tym obsługa znaków narodowych), przynajmniej częściowa zgodność ze standardami UX dla iOS (poprawne wyświetlanie list, obsługa dowolnej ilości wyników, właściwe ikony)
6. wykorzystanie **wybranego narzędzia wieloplatformowego** lub języka Swift i środowiska xcode (w miarę dostępności własnego sprzętu)
7. projekt może zakładać opracowanie własnego API i części serwerowej, wtedy liczba członków zespołu może zostać zwiększona o 2 osoby (czyli maksymalnie do 5 osób); wszystkie osoby w zespole muszą posiadać wiedzę i umiejętności związane z oprogramowaniem klienckim (na iOS), ponieważ tylko tej części projektu będzie dotyczyć egzamin!

Opcjonalnie:

8. obsługa mapy/aparatu/akcelerometru/GPS/BLE/ innych czujników urządzenia
9. testy na symulatorze iOS działającym na macOS (analizuję możliwość udostępnienia komputerów z MacOS z jakąś formą „zdalnego pulpitu”, w tej chwili nie mogę określić szczegółów ze względu na zmieniające się przepisy, np. ograniczenie dostępu do laboratorium)

uwaga: w projektach nie wolno używać gotowych modułów/bibliotek/komponentów realizujących w całości postawione zadania lub ich istotne części (np. dla konwertera walut zabronione jest wykorzystanie modułu, który podaje kursy walut z Internetu po wskazaniu kodów walut jako argumentów).

Prezentacja projektów:

Każdy zespół otrzyma dostęp do folderu na dysku google, w którym należy umieszczać dokumentację poszczególnych efektów kształcenia.

W celu zaliczenia efektu 1 („zna zasady projektowania aplikacji mobilnych dla systemu iOS z uwzględnieniem najnowszych trendów”) należy opracować analizę porównawczą wybranego narzędzia developerskiego z co najmniej dwoma rozwiązaniami alternatywnymi (rozwiązanie natywne: xcode+Swift/ObjectiveC, przykładowe narzędzia wieloplatformowe: React Native, Adobe Phonegap/Cordova, Flutter, Xamarin, Embarcadero RAD Studio, B4x, wszystkie wymienione są open-source lub mają wersje bezpłatne/community z wyjątkiem B4x Anywhere Software, które jest bezpłatne wyłącznie dla Androida, ale zasady projektowania i język dla iOS są analogiczne). Analiza porównawcza powinna zostać zrealizowana jako prezentacja z informacjami o szczegółach technologicznych (sposób generowania GUI, sposób współdziałania z systemem operacyjnym, dostępne języki/kompilatory/interpretery itd., podgląd aplikacji w trakcie projektowania itp.) oraz porównanie wygody i wydajności przy zakodowaniu prostej aplikacji typu „hello world” z jednym przyciskiem, zmieniającym tekst na etykiecie.

Ponadto na końcu prezentacji należy umieścić adresy: **repozytorium plików projektu** (np. w serwisie github) oraz **systemu zarządzania zadaniami** w projekcie (np. tablica Trello).

Prezentację należy wgrać swój folder na dysku google i wysłać powiadomienie mailem. Nie ustalam terminów z racji wyjątkowej sytuacji.

(pozostałe efekty i wymagania w kolejnych aktualizacjach)

Przykłady tematów projektów:

1. Pogodynka
2. Konwerter walut
3. Tłumacz/gra językowa
4. Słownik/słownik wyrazów bliskoznacznych
5. Plan zajęć (np. z plan.polsl.pl)
6. Lokalizator POI (restauracje, bary, sklepy, stacje benzynowe)
7. Rozkłady jazdy komunikacji miejskiej/pociągów/samolotów
8. Terminarze imprez/meetupów/programy kinowe/teatralne
9. Cytaty
10. Dane statystyczne dla państw/województw/miast/obszarów/dziedzin
11. Czytnik kodów kreskowych/sprawdzanie ceny/lista zakupów
12. Rozpoznawanie obiektów na obrazie (ew. obiektów z pewnej kategorii, być może z własnym API)
13. Rozpoznawanie znaków drogowych (być może z własnym API)
14. Rozpoznawanie figur geometrycznych (być może z własnym API)
15. Analiza sentymentu/streszczanie/inne algorytmy NLP (być może z własnym API)