Platforma de examinare a cunostiintelor pe device-urile mobile

Student: Costea Radu
Profesor coordonator: Conf. Mihai Horia Zaharia

Descriere

Proiectul isi propune sa standardizeze procesul de examinare a cunostiintelor, intr-o maniera simpla si intuitiva, putand avea astfel o aplicabilitate practica foarte mare. Principala provocare consta in a putea abstractiza cat mai bine procesul, astfel incat prin standardizearea lui sa nu se piarda din aplicabilitate. O alta provocare o constituie necesitatea de a implementa procesul intr-o maniera sigura dar care sa permita accesul de la distanta. Al treilea aspect important, este faptul ca rezultatul final trebuie sa fie intuitiv si usor de folosit, astfel incat curba de invatare sa fie cat mai mica.

Ca si feature-uri, aplicatia va fi capabila de urmatoarele:
+ crearea de intrebari cu layout diferit, sau cu mod de raspuns diferit
+ creeare, editare de teste
+ generarea automata a testelor avand predefinit un set de posibile intrebari, cu ponderi diferite
+ generare manala a testelor alegand dintr-un set de posibile intrebari
+ intrebari cu pondere variata
+ formula de calcul a notei finale tinand cont de ponderile intrebarilor
+ simularea testelor pentru intrebarile proprii, sau sustinerea de teste la initiativa autorului intrebarilor
+ validarea raspunsului de catre autorul intrebarii pentru intrebari cu raspuns liber sau interpretabil
+ istoric al testelor sustinute, si eventual cu statistici
+ nivele diferite de acces la date functie de tipul utilizatorului (autor / profesor, elev / evaluat)

Platforme hardware

Unul din aspectele cele mai importante ale aplicatiei il constituie tehnologia. Aplicatia trebuie sa fie utilizabila pe device-urile mobile, iar pentru a restrange aria device-urilor mobile am ales device-urile compatibile iOS (iPhone / iPad).
Ca si limitare aici vom considera iOS 9.0 ca fiind minimul iOS compatibil..

Limbajul de programare

Pentru a putea scrie aplicatii native iOS se va folosi unul din limbajele de programare: Objective C sau Swift, sau o combinatie intre cele doua, functie de necesitati. Si pentru ca Swift incepe sa se contureze ca un limbaj de programare matur, ne-am propus ca cea mai mare parte din aplicatie sa fie cod pur Swift, iar la nevoie vom apela si la Objective C.

Tehnologii

Pentru a putea duce la bun sfarsit ceea ce ne-am propus avem nevoie sa ne folosim mai multe tehnologii astfel:

- Partea de stocare locala va fi facuta folosindu-ne CoreData (O baza de date de tip graf de obiecte, avand in implementarea interna o baza de date SQLite)
- Partea de cloud va fi facuta folosindu-ne de CloudKit (Un framework prin care putem stoca datele in cloud)
- Pentru simplitate, imaginile vor fi stocate ca stringuri in baza 64
- Partajarea testelor se va face posibila prin utilizarea deeplink-rilor, prin mail
- Pentru partea de Unit teste se va folosi suportul din XCode – XCTest
- Partea de UI Teste va fi facuta de asemenea ce ofera XCode
- Criptarea datelor cu caracter sensibil va fi facuta folosind algoritmi de criptare precum SHA256, si Keychain

Proiectarea aplicatiei

Aplicatia va fi realizata in trei etape:

I. Suport minimal local: Se va crea scheletul aplicatiei, care sa permita operatiile de baza ce se pot realiza local (autentificare, persistenta locala, creare de teste, simulare a unui test, etc). Aceasta este cea mai importanta etapa, deoarece pe baza a ceea ce se construieste acum se va putea dezvolta in urmatoarele etape.
II. Suport minim remote: Peste ceea ce s-a creeat la prima etapa se va adauga suportul pentru lucrul la distanta (persistenta in cloud, distribuirea de teste intre utilizatori, etc)
III. Extinderea functionalitatilor: Aceasta etapa este etapa in care se finiseaza aplicatia, se adauga functionalitati noi, se adauga moduri noi de testare, tipuri noi de intrebari, etc.

Arhitectura

In inima aplicatiei se afla baza de date, si mai exact modul in care sunt abstractizate elementele specifice unui test astfel:

Orice test este compus din una sau mai multe intrebari. Fiecare intrebare are unul sau mai multe posibile raspunsuri corecte.
Asadar putem extrage 3 componente de baza si relatiile dintre ele:

1. Intrebarea – poate fi reprezentata printr-un text, imagine etc.
2. Raspusul – poate fi un text, o formula, o imagine, etc
3. Algoritmul de evaluare a raspunsului – reprezinta cum validam ca un raspuns este sau nu corect pentru o anumita intrebare (de exemplu pentru intrebari cu raspuns liber, a compara cu un string predefinit poate fi gresit de vreme ce raspunsul este supus diferitelor forme de exprimare si specifice fiecarui limbaj in parte)

Fiecare dintre aceste 3 componente poate fi extinsa. Si prin existenta diferitelor variatii ale acestora vom reusi sa acoperim o gama cat mai larga de forme de examinare.

Mai jos se poate observa un exemplu al organizarii obiectelor intr-o baza de date prototip: