Ohjelman kuvaus

Toteutettu Android applikaatio, DaBank, on mobiiliverkkopankkisovellusta mukaileva simulaatiosovellus. Sovellus sisältää samankaltaisia toimintoja kuten oikeat mobiilipankkisovellukset, mutta ei kommunikoi verkon yli oikeiden pankkien kanssa vaan toimii tavallaan omassa hiekkalaatikossa. Sovellus käyttöliittymän kannalta simuloi asiakkaan näkökulmaa, Applikaatio ei hyödynnä Internet oikeuksia ollenkaan, vaan sovellus luo tietokannan tietojen tallettamista varten ensimmäisellä asennuskerralla laitteeseen.

Sovelluksessa on neljä eri pankkia; DaBank, Star Bank, Flash Bank ja Sun Bank. Kullakin pankilla on omat asiakkaansa ja kullakin asiakkaalla omat pankkitilinsä. Asiakas voi kirjautua mihin tahansa pankkiin testikäyttäjänä tai järjestelmänvalvojana (Admin -käyttäjänä). Järjestelmänvalvoja voi lisätä uusia asiakkaita pankkeihin sekä olemassa oleville asiakkaille pankkitilejä. Asiakas voi kuitenkin itse liittää tileihinsä mielivaltaisesti pankkikortteja sekä muokata niitä.

- Testikäyttäjän tunnukset: Username: username, Password: password
- Järjestelmänvalvojan tunnukset: Username: admin, Password: Administrator123!

Huomautettakoon jo tässä vaiheessa, että sovellus **ei ole suojattu SQL -injektioita vastaan.** Tämä tarkoittaa sitä, että tietokannan voi mahdollisesti vioittaa syöttämällä tekstikenttiin erikoismerkkejä tai SQL -kielen varattuja sanoja. Mikäli tietokanta kuitenkin vioittuu, sovellus on poistettava (mukaan lukien tietokanta, joka sijaitsee Android laitteella polussa /data/data/com.jhprog.dabank/databases/data.db ja asennettava uudelleen.

Ohjelman toteutus

Luonnollisesti ainoana ryhmä jäsenenä toteutin yksin projektin kaikki ominaisuudet. Suunnitteluvaiheessa pyysin isääni auttamaan tietokantojen kanssa, sillä tietokantoja ei ole vielä ensimmäisenä opiskeluvuotena opetettu. Suunnitteluvaiheessa oli jo selvää, että tietokanta on SQL -pohjainen, joten ensimmäinen ongelma oli selvittää miten SQL -pohjaisen tietokannan ylipäätään saa implementoitua Android applikaatioon. Android Developers sivustolla oli kuitenkin erinomaisesti dokumentoitu tietokannan toteuttaminen:

- https://developer.android.com/training/data-storage/sqlite

Suurimman osan ohjelmiston suunnittelusta kuitenkin sain sisällytettyä alustavaan harjoitustyösuunnitelmaan. Joitain rajauksia oli kuitenkin tehtävä alkuperäiseen suunnitelmaan nähden, kuten pankkitilien korollisuuden implementoiminen, aikarajoitteiden puitteissa. Sovelluksen järkevää rakennetta lähdin kasaamaan toteuttamalla ensin suurimman osan käyttöliittymästä. Kun käyttöliittymä oli valmis, niin pala kerrallaan työstäminen oli kohtuullisen suoraviivaista. Eniten kuitenkin työtä aiheutti Androidin arkkitehtuurin tutkiminen. Karkeasti arvioituna tein harjoitustyötä kuukauden verran, yhteensä noin 150 tuntia.

Mitä opin harjoitustyöstä?

Eniten opin suunnittelemaan ja toteuttamaan projektimaisia laajempia ohjelmistoja. Olen aikaisemmin jo harrastusmielessä tehnyt useita projekteja ennen opintojani (aloitin itseasiassa Java ohjelmoinnin opiskelun yhdeksännellä luokalla), mutta tämä taisi viedä mittakaavassa voiton. Huomasin myös et paljon joutuu rajaamaan ominaisuuksia, sillä tavoitteet nopeasti riistäytyvät käsistä ja aikataulu tulee vastaan.

Listatut toiminnallisuudet

Ominaisuus	Pisteet
Olio-ohjelmoitu	**Pakollinen**
Vähintään viisi erilaista luokkaa & oliota (käyttöliittymäluokkia ei lasketa)	**Pakollinen**
Tärkeiden tietojen kirjoitus tiedostoon (XML, JSON tai CSV)	**Pakollinen**
Aihe spesifit perustoiminnot (löytyvät jokaisen aiheen alta listattuna)	**Pakollinen**
Ohjelma on rakennettu hyvin suunnitelluista UI-komponenteista	2 pistettä
Tietokanta (esim. MySQL) (**Huom!** 2 pisteen tietojen kirjoitus & lukemis -ominaisuutta ei tämän toteutuksen kanssa lasketa enää lisäpisteiksi ts. sen voi jättää tekemättä)	5 pistettä
Admin-käyttäjä	3 pistettä
Useampi käyttäjä (ja niiden luominen), tietojen tallennus järkevästi jonnekin	3 pistettä
Kirjautuminen applikaatioon	3 pistettä
Kirjautumisen salasana noudattaa hyvän salasanan sääntöjä (sisältää vähintään yhden numeron, erikoismerkin, ison ja pienen kirjaimen, on vähintään 12 merkkiä pitkä).	1.5 pistettä
Salasanan tallennus käyttää jonkinlaista hash-menetelmää ja suolausta (esim SHA-512 + salt)	3 pistettä
Käyttöliittymä noudattaa Googlen Material Designia sekä kaikki käyttöliittymäkomponentit ovat itse luotuja (grafiikoiltaan). Ohjelma noudattaa nykypäivän ”best practices” (LiveData, SharedViewModel, ViewBinding, DatabaseContract).	4 pistettä

(Seuraava sivu)

Pistemäärä	Toiminto
Pakollinen	**Voi muokata käyttäjän tietoja (nimi, yhteystiedot, jne.)**
Pakollinen	**Voi luoda käyttäjille tilejä (oikeasti pitäisi tehdä pyyntö pankille, tässä kuitenkin voi tehdä suoraan)**
Pakollinen	**Voi muokata tilien ominaisuuksia esim.** **1. tilityyppi (Voi luoda käyttämällä luokkia tai muuttujia)** **2. voiko tililtä maksaa,** **3. rahan siirto tililtä toiselle** **4. jne.**
Pakollinen	**Voi luoda tileihin pankkikortteja (oikeasti pitäisi tehdä pyyntö pankille, tässä kuitenkin voi tehdä suoraan)**
Pakollinen	**Voi tarkastella tilitapahtumia (siirrot, nostot, talletukset, maksut, jne.)**
Pakollinen	**Voi lisätä rahaa tilille**
1	Voi muokata pankkikorttien oikeuksia (nostoraja, maksuraja, toimivuusalue, jne.)
1	Voi siirtää rahaa omalta tililtä toiselle
1	Voi tehdä rahanostoja tai korttimaksuja (jos tilillä on kortti, muista maksuraja), simuloitu
1	Voi tehdä tilisiirtoja ulkopuoliselle tilille JA Voi siirtää rahaa käyttäjien välillä (maksutapahtuma)
2	Korteissa on maarajoitukset ja rahanostot ja korttimaksut voi testata eri maissa
2	Voi tehdä "maksetaan eräpäivänä" maksun, joka veloitetaan automaattisesti tililtä kun päivä on mennyt
3	Järjestelmässä voi olla useampi pankki, joilla kaikilla on omat käyttäjänsä ja tilinsä. Tilitiedoissa, maksuissa ja tapahtumissa pitää näkyä pankin BIC koodi (esim. Osuuspankin OKOYFIHH koodi)
3	Voi tehdä toistuvia automaattisia veloitustapahtumia (veloitetaan tililtä automaattisesti) kuten esimerkiksi automaattiset laskujen maksut

YHTEENSÄ: 40