ContentWire

Váradi Kristóf

2022. október 30.

1. Rövid Leírás

A ContentWire egy **Java Swing alapú asztali alkalmazás E-Mail marketinges célokra**, amelynek segítségével egy szervezeten belül a ilyen tevékenységért felelős személyek egy egyszerűen használható **felhasználói interfészen keresztül kommunikálhatnak a szervezet változatos összetételű, változékony közönségeivel**. Az alkalmazás fő előnye, hogy könnyedén illeszthető különböző E-Mail szolgáltatókhoz, továbbá a felhasználók sokféle ügyfélcsoportot hozhatnak létre, feléjük különböző módokon kommunikálhatnak.

2. Strukturális áttekintés

A ContentWire az E-Mailek küldéséhez az **SMTP protokollt** használja, mivel a legtöbb szolgáltató rendelkezik SMTP szerverrel, valamint egyéb, a protokollt támogató megoldásokkal is. Az alkalmazás üzleti logikai szinten strukturálisan két lényeges részre bontható:

1. **Resource Management**
2. **E-Mail Management**

Míg az E-Mail Management Service az SMTP szerver felé közvetíti a kiküldeni kívánt leveleket, a Resource Management Service kezeli a tárolt E-Mail fiókokat, a kampánymenedzserek adatait és fiókjait, a közönségeket, valamint magukat a közönség tagjait is.

2.1. Resource Management

A *Resource Management*-ben az adott szervezeten belül kijelölt személy menedzseli a *Campaign*-eket, az *Audience*-eket, az *Audience Member*-eket, valamint a *Campaign Responsible*-öket. Itt lehetőség adódik az előbb felsoroltak hozzáadására, törlésére, módosítására, valamint egymáshoz rendelésére is. A *Resource Management* két további komponensre bontható fel, ezek a *Campaign Management Service*, valamint a *User Management Service*. Míg az előbbi szolgáltatás a kampányok, valamint a közönségek és közönségtagokat és a köztük fennálló kapcsolatokon végez műveleteket, a *User Management Service* a *Campaign Responsible*-ök adataiért, autentikációjáért, stb... felelős.

2.2. E-Mail management

Az E-Maileket a *Customer Responisble*-ök komponálhatják egyszerű, formázatlan szövegként az alkalmazáson belül, vagy importálhatnak HTML formátumú dokumentumot, amelyet az E-Mail Management Service ugyanúgy postáz, mint a plain-text üzenetet. HTML-tartalmú üzenet esetén lehetőséget kell biztosítanunk arra, hogy a *Customer Responsible* megjelenítse a feltöltött fájlt.

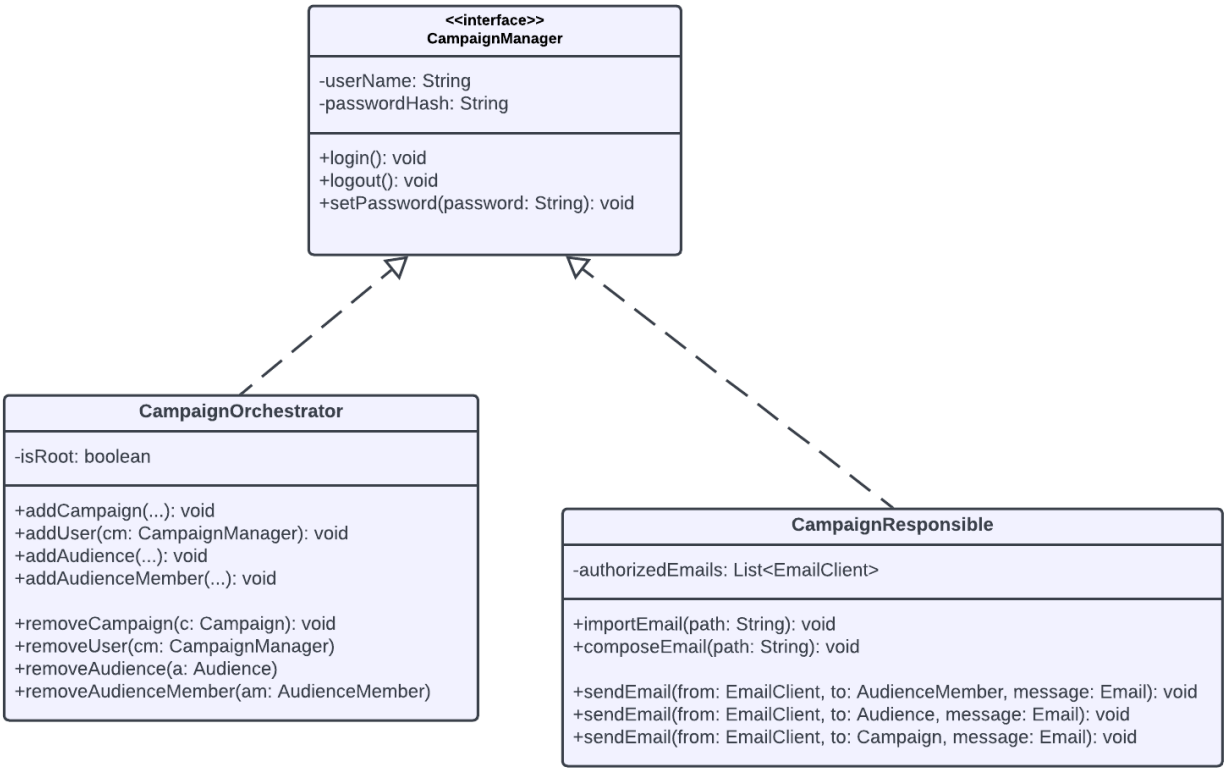
2.3. Felhasználói osztályok

A feladat helyes kivitelezése szempontjából a legfontosabb osztályok azok, amelyek a felhasználók modellezéséhez szükségesek.

Minden felhasználónak tárolnunk kell az egyedi felhasználónevét, a hashelt jelszavát, be-, és ki kell tudnunk őket jelentkeztetni, valamint lehetőséget kell adnunk rá, hogy megváltoztassák a saját jelszavukat.

A *CampaignOrchestrator*-okról tudnunk kell, hogy az adott objektum root felhasználóe (ezt nem törölhetjük), valamint az ilyen személyek hozzá tudnak adni, és törölni tudnak kampányokat, *CampaignManager*-eket, közönségeket és közönségtagokat is.

A *CampaignResponsible*-ök



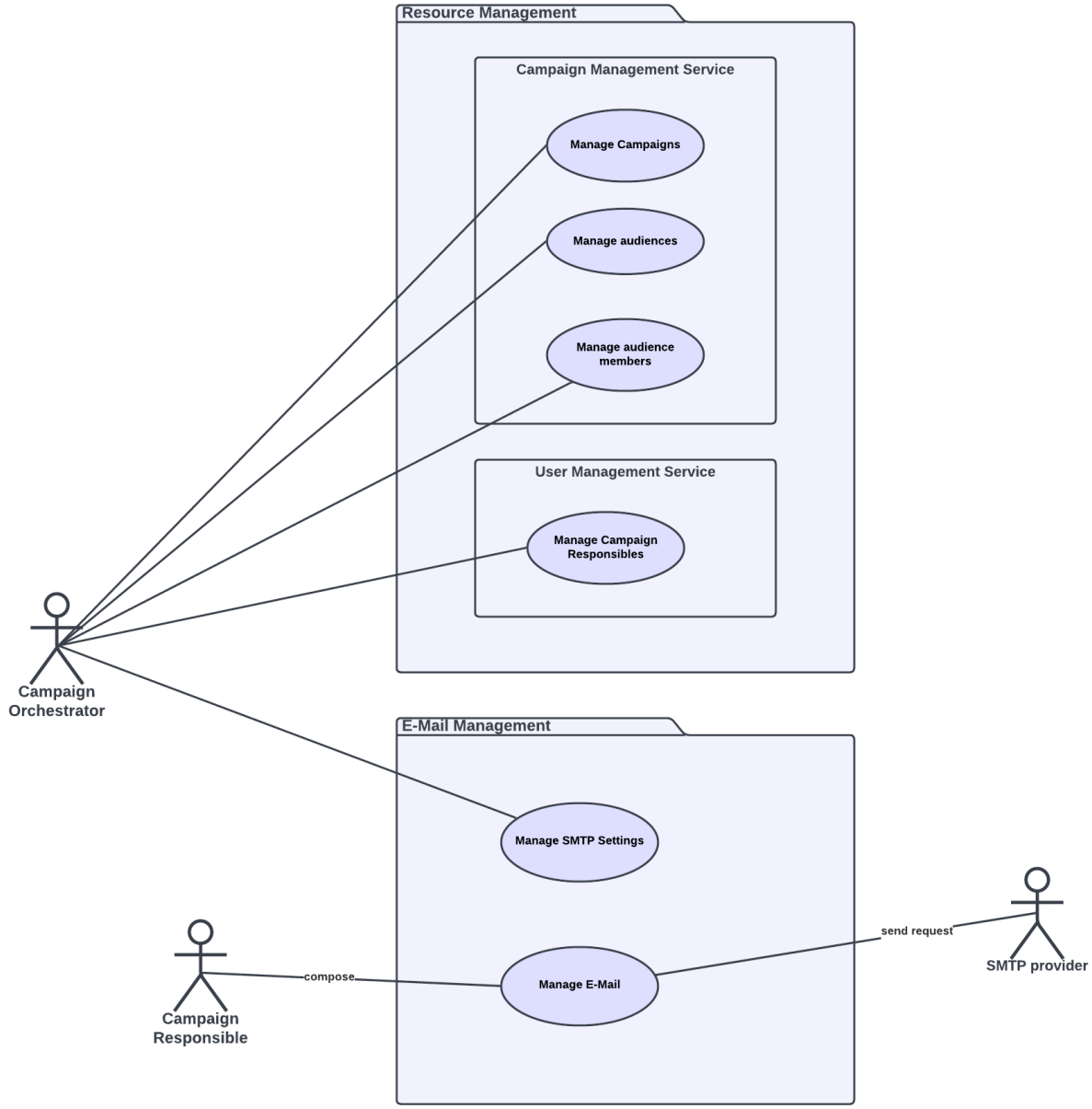
1. ábra. Osztálydiagram a *Campaign Manager*-ek struktúrájának szemléltetésére

3. Az applikáció nézetei

A terméket (attól függően, hogy az aktuális felhasználó *Campaign Orchestrator*, vagy *Campaign Responsible*-e) különböző módokon használhatjuk.

A *Campaign Orchestrator*-nak rálátása van az összes közönségekkel és *Campaign Responsible*-ökkel kapcsolatos adatra, és az adott objektumokat, valamint az SMTP beállításokat módosítani tudja, az E-Mailek menedzselése viszont *nem* tartozik a hatáskörébe.

A *Campaign Responsible* nem rendelkezik a *Campaign Orchestrator* jogosultságaival, azonban képes E-Maileket komponálni, valamint ki tudja küldeni azokat változatos közönségeknek.



2. ábra. Use-Case Diagram a felhasználók hatásköreiről

4. Használt technológiák

4.1. Adatbázis-kezelés

A felhasználók bejelentkezési adatait, a mentett kampányokat, közönségeket, és közönségtagokat, stb... tárolni kell, ezért szükségünk van egy olyan adatbázisra, amelybe beszúrhatjuk, illetve ahonnan lekérdezhetjük ezeket az információkat. Ehhez a **JDBC API-t**, valamint az **SQLite-ot** (vagy ha nem bizonyul megfelelőnek az adatok tárolásához, egyéb RDBMS-t) használhatjuk.

4.2. Titkosítás

Mivel a rendszer többfelhasználós, és kritikus információkat is tartalmazhat, az adatbázisban tárolt jelszavakat és egyéb bejelentkezési adatokat hashelnünk kell, amelyhez használhatunk egy beépített java.security megoldást, vagy egyéb, külső algoritmusokat is. Továbbá megfontolandó, hogy mennyire biztonságos lokálisan tárolni jelszó-hasheket nagy mennyiségben.

4.3. SMTP megvalósítás

Az E-Mailek küldéséhez, mint ahogy korábban szót ejtettünk róla, szükség van egy olyan API-ra, amely hozzáférést nyújt az E-Mail szolgáltatók SMTP szervereihez. Ehhez célszerű a javax.mail API-t használnunk, de természetesen léteznek alternatívák.