MINISTERUL EDUCAȚIEI NAȚIONALE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

ANDROCHEF: SISTEM PENTRU EFICIENTIZAREA COMENZILOR CĂTRE RESTAURANTE ÎN CADRUL GRUPURILOR DE PERSOANE

LUCRARE DE LICENȚĂ

Absolvent: **Oana Raluca RUSU**

Coordonator științific: **Şef lucrări ing. Cosmina IVAN**

2014

MINISTERUL EDUCAȚIEI NAȚIONALE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

DECAN,
Prof. dr. ing. Liviu MICLEA

DIRECTOR DEPARTAMENT,
Prof. dr. ing. Rodica POTOLEA

Absolvent: **Oana Raluca RUSU**

ANDROCHEF: SISTEM PENTRU EFICIENTIZAREA COMENZILOR CĂTRE RESTAURANTE ÎN CADRUL GRUPURILOR DE PERSOANE

1. **Enunțul temei:** *Proiectul își propune realizarea unei aplicații, ale cărei funcționalități au rolul de a facilita și de a eficientiza efectuarea de comenzi către restaurante, în cadrul grupurilor de persoane.*

2. **Conținutul lucrării:** *Cuprins, Introducere, Obiectivele proiectului, Studiu bibliografic, Analiză și fundamentare teoretică, Proiectare de detaliu și implementare, Testare și validare, Manual de instalare și utilizare, Concluzii, Bibliografie, Anexe.*

3. **Locul documentării**: Universitatea Tehnică din Cluj-Napoca, Departamentul Calculatoare

4. **Consultanți**: Șef lucrări ing. Cosmina IVAN

5. **Data emiterii temei:** 1 noiembrie 2013

6. **Data predării:** 3 Iulie 2014

Absolvent: _____

Coordonator științific: _____

MINISTERUL EDUCAȚIEI NAȚIONALE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

Declarație pe proprie răspundere privind autenticitatea lucrării de licență

Subsemnatul(a)_____
_____,
legitimat(ă) cu _____ seria _____ nr. _____
CNP _____, autorul lucrării
_____
_____
_____elaborată în vederea susținerii examenului de finalizare a studiilor de licență la Facultatea de Automatică și Calculatoare, Specializarea _____ din cadrul Universității Tehnice din Cluj-Napoca, sesiunea _____ a anului universitar _____, declar pe proprie răspundere, că această lucrare este rezultatul propriei activități intelectuale, pe baza cercetărilor mele şi pe baza informațiilor obținute din surse care au fost citate, în textul lucrării, și în bibliografie.

Declar, că această lucrare nu conține porțiuni plagiate, iar sursele bibliografice au fost folosite cu respectarea legislației române şi a convențiilor internaționale privind drepturile de autor.

Declar, de asemenea, că această lucrare nu a mai fost prezentată în faţa unei alte comisii de examen de licență.

In cazul constatării ulterioare a unor declaraţii false, voi suporta sancţiunile administrative, respectiv, *anularea examenului de licenţă*.

Data | Nume, Prenume

_____ | _____

Semnătura

Cuprins

Capitolul 1. Introducere 1
1.1. Contextul general 1
1.2. Contextul aplicației 2
1.3. Conținutul lucrării 3
Capitolul 2. Obiectivele Proiectului 5
2.1. Obiectiv principal – Comanda de grup 5
2.2. Obiective secundare 5
2.2.1. Comanda individuală 6
2.2.2. Înregistrare, Autentificare, Recuperare parolă 6
2.2.3. Obiective auxiliare 7
Capitolul 3. Studiu Bibliografic 8
3.1. Aplicații mobile vs. Aplicații web 8
3.2. Platforma Android vs. iOS 10
3.3. Sisteme similare 11
3.3.1. AndroChef vs. Urbanspoon 12
3.3.2. AndroChef vs. FoodPanda 14
3.3.3. AndroChef vs. EatFab 17
3.3.4. Concluzii pe baza studiului sistemelor similare 19
Capitolul 4. Analiză și Fundamentare Teoretică 20
4.1. Arhitectura conceptuală a sistemului 20
4.2. Cerințele aplicației 22
4.2.1. Cerințele funcționale 22
4.2.2. Cerințele non-funcționale 24
4.3. Modelul cazurilor de utilizare 25
4.3.1. Diagrama cazurilor de utilizare 25
4.3.2. Înregistrare, Autentificare, Recuperare parolă 27
4.3.3. Efectuarea comenzii individuale 30
4.3.4. Efectuarea comenzii de grup 31
4.4. Tehnologii utilizate 33
4.4.1. Tehnologii pe parte de Server 34
4.4.2. Comunicarea între Client și Server 37
4.4.3. Tehnologii pe parte de Client 39

Capitolul 5. Proiectare de Detaliu si Implementare 43
5.1. Structura generală a sistemului 43
5.2. Diagrama de deployment 44
5.3. Aplicația Server – Detalii de Implementare 45
5.3.1. Diagrama de module 45
5.3.2. Componenta de servicii web 46
5.3.3. Componenta Util 51
5.3.4. Componenta de module DBIx 51
5.4. Aplicația Client – Detalii de Implementare 52
5.4.1. Diagrama de pachete 52
5.4.2. Diagrama de clase 53
5.4.3. Descrierea claselor importante 56
5.4.4. Diagramele de secvență pentru comanda de grup 59
5.5. Baza de date MySQL 62
Capitolul 6. Testare şi Validare 63
6.1. Testarea manuală a aplicației 63
6.1.1. Black Box Testing (Functionality Testing) 63
6.1.2. White Box Testing 63
6.1.3. Regression Testing 64
6.2. Testarea automată a aplicației 64
6.2.1. Sikuli Test 64
6.2.2. Robotium 65
6.2.3. Testarea automată a comenzii individuale cu Robotium 65
Capitolul 7. Manual de Instalare si Utilizare 67
7.1. Instalarea aplicației AndroChef 67
7.1.1. Instalarea aplicației Server 67
7.1.2. Instalarea aplicației Client 68
7.2. Manual de utilizare 68
7.2.1. Înregistrare, Autentificare, Recuperare parolă 68
7.2.2. Comanda de grup 70
Capitolul 8. Concluzii 74
8.1. Realizări 74
8.2. Dezvoltări ulterioare 75
Bibliografie 76

Anexa 1 – Lista de figuri 78
Anexa 2 – Lista de tabele..... 80
Anexa 3 – Glosar de termeni..... 81

Capitolul 1. Introducere

În acest capitol, se vor prezenta câteva aspecte referitoare la evoluția sistemelor de operare pentru dispozitive mobile, precum și o scurtă introducere a contextului în care aplicația AndroChef a fost dezvoltată.

1.1. Contextul general

Articolul [1] prezintă evoluția tuturor sistemelor de operare pentru dispozitive mobile, fiind sursa de inspirație a următoarei prezentări.

Vorbim despre o istorie de 19 ani în care, sistemele de operare pentru dispozitive mobile au avut o evoluție spectaculoasă. Totul a început cu prima versiune a sistemului de operare Palm OS. Timp de aproape 5 ani de zile Palm OS a fost singurul sistem de operare pentru terminale mobile. Aceasta s-a întâmplat până în anul 2000, când Microsoft lansa Pocket PC. Sistemul de operare Pocket PC avea să supraviețuiască doar 3 ani de zile, până la lansarea primei versiuni a Windows Mobile.

În 2005, Google achiziționa Android Inc., însă abia 3 ani mai târziu avea să lanseze sistemul de operare cu același nume. Apple deja lansase prima versiune a iOS în anul 2007. Apple a apărut târziu în lumea companiilor care au lansat sisteme de operare pentru terminale mobile, însă dispozitivele sale erau cele mai bine vândute din lume în anul 2011, iar compania încasa două treimi din profiturile vânzătorilor de telefoane mobile.

În ciuda faptului că în anul 2011 Apple domina piața dispozitivelor mobile cu sistemul de operare iOS, statisticile pe următorii ani arătau o scădere pe piață pentru aceștia. Sistemul de operare Android al celor de la Google a început să se răspândească tot mai mult printre utilizatorii de telefoane inteligente, iar în prezent, Android domină piața de terminale mobile.

Sistemul de operare Android a cunoscut o dezvoltare tot mai mare de la an la an pe cele 12 piețe mari din întreaga lume. În Statele Unite, a înregistrat 50.6 % din vânzările de telefoane inteligente în lunile octombrie, noiembrie și decembrie ale anului 2013. Pe următoarele poziții s-au clasat iOS cu 43.9 % si Windows Phone cu 4.3 %. iOS a suferit un declin de 5.8 procente față de anul 2012, în timp ce Windows Phone a cunoscut o creștere de 1.9 procente. Android a avut cel puțin 50 de procente în piața din Europa, America Latină, China si Australia pe perioada celor trei luni mai sus menționate. În aceeași perioadă, sistemele de operare produse de către Microsoft au avut creșteri în toate regiunile menționate, mai puțin America Latină, în Europa având o creștere de 4.7 procente.

După toate cele prezentate mai sus, reiese că sistemul de operare Android este cel mai răspândit sistem pentru dispozitive mobile din întreaga lume. În acest context, aplicația AndroChef este dezvoltată pentru a putea fi utilizată pe platforma Android. Astfel, aplicația se aliniază tendințelor actuale.

1.2. Contextul aplicației

Dispozitivele mobile au devenit indispensabile în viața de zi cu zi. Dacă la început acestea erau utilizate doar cu scopul de a păstra legătura între persoane, prin intermediul apelurilor telefonice si al mesajelor, în prezent, dispozitivele mobile au tendința de a înlocui calculatoarele clasice atunci când vine vorba de entertainment sau alte activități zilnice.

În mod cert, aplicațiile pentru telefoane inteligente au devenit și ele indispensabile pentru posesorii de dispozitive mobile, mărturie în acest sens fiind numărul lor aflat in permanență creștere. Numărul aplicațiilor disponibile în prezent pe Google Play este de 1,249,713, aproape dublu față de începutul anului 2013. Odată cu această impresionantă creștere, tot mai multe aplicații pentru eficientizarea comenzilor către restaurante și-au făcut apariția, venind în ajutorul celor care obișnuiesc să își comande micul dejun, prânzul sau cina prin intermediul aplicațiilor web sau a dispozitivelor mobile.

În acest context, am ales să dezvolt o aplicație pentru persoanele care dețin un telefon inteligent pe care rulează sistemul de operare Android. Această aplicație vine în ajutorul celor care doresc să comande de la un anumit restaurant, într-un mod cât mai simplu și eficient. Utilitatea aplicației se poate observa cel mai bine, atunci când este folosită în cadrul grupurilor de persoane. Cu cât grupul este mai numeros, cu atât crește și mai mult utilitatea funcționalităților oferite.

Pentru ca aplicația să poată fi utilizată, conexiunea la Internet este absolut necesară, fie ea prin wireless sau prin conexiunea de date oferită de către furnizorul de telefonie mobilă. Astfel, locația în care se află persoana care dorește să plaseze o anumită comandă, nu mai este o problemă atâta timp cât există și conexiune la internet. În acest mod, este eliminată constrângerea ca persoana în cauză să se afle acasă sau la birou, în fața calculatorului conectat la internet, sau chiar la restaurant cu meniul la îndemână, pentru a putea efectua o comandă către unul dintre restaurantele preferate.

În zilele noastre, fiecare restaurant are propriul site în care își prezintă întreaga ofertă de produse. O persoană care dorește să își comande de la un anumit restaurant, trebuie mai întâi să acceseze site-ul restaurantului, apoi să își aleagă meniul, după care să plaseze comanda online sau telefonic. Dacă persoana în cauză are acces doar la dispozitivul mobil, atunci plasarea comenzii prin intermediul aplicației web poate deveni foarte greoaie, deoarece mare parte din aplicațiile web nu sunt optimizate pentru dispozitive mobile. Apare cazul în care persoana, după ce consulta meniul, constată că nu este interesată de nimic din această ofertă. Ceea ce trebuie să facă este să părăsească aplicația web actuală și să se gândească la un alt restaurant care ar putea să ii ofere ceea ce ea dorește în acel moment. Foarte mult timp se consumă în acest proces de accesare a paginilor web. O soluție imediată este transformarea aplicației web într-o aplicație pentru Android. Această soluție elimină parțial problema timpului, deoarece nu elimină timpul în care persoana trebuie să se gândească care restaurant ar putea să îi satisfacă cerințele și apoi să instaleze pe rând fiecare aplicație a restaurantului căruia dorește să îi vizualizeze meniul.

Toate neajunsurile mai sus menționate sunt eliminate de către aplicația care face obiectul acestui document. Aplicația pune la dispoziția utilizatorilor o listă de restaurante care sunt înscrise în cadrul ei. Un mare avantaj este faptul că utilizatorii au posibilitatea să aleagă dintr-o gamă diversificată de restaurante, având un acces foarte ușor la meniul

acelor restaurante, precum si la informațiile de contact ale acestora. Meniurile sunt prezentate în același format, nemaifiind necesar ca utilizatorul să se obișnuiască cu regulile fiecărei aplicații proprii ale unui restaurant. Dacă cineva dorește să comande de la cel mai apropiat restaurant față de locația în care se află, i se va pune la dispoziție o listă cu toate restaurantele dintr-o anumită zonă, în ordine crescătoare a distanței față de utilizator. În acest fel, el va ştii care este restaurantul cel mai apropiat şi va putea să efectueze o comandă pentru care timpul de livrare sau timpul în care se deplasează către restaurant, va fi mai scurt. De asemenea, utilizatorul va putea să vizualizeze o listă cu restaurantele cele mai des accesate. Astfel, persoana care dorește să plaseze o comandă poate să aleagă din restaurantele pe care le vizitează de cele mai multe ori, fiind astfel restaurantele ei preferate. Metodologia de plasare a unei comenzi este aceeași pentru fiecare restaurant în parte, eliminând problema de acomodare care apare la aplicațiile proprii restaurantelor.

Cea mai importantă şi inovatoare funcționalitate a aplicației este posibilitatea plasării *comenzilor de grup*. Această funcționalitate face diferența între aplicațiile deja existente şi aplicația de față. Este foarte utilă atunci când mai multe persoane se află în aceeași locație şi doresc să comande de la același restaurant. În loc să se facă o listă lungă cu selecțiile fiecărei persoane, iar apoi o singură persoană să trebuiască să efectueze o comandă foarte mare, fiecare participant la grup va avea posibilitatea să facă o comandă simplă în cadrul unui grup creat de un așa-zis administrator responsabil cu crearea grupului şi cu plasarea finală a comenzii. Această funcționalitate vine cel mai mult în ajutorul angajaților companiilor care, de obicei, comandă de la unu sau mai multe restaurante într-o anumită zi. Comenzile din cadrul companiilor se efectuează de cele mai multe ori prin simple formulare care mai apoi trebuie comunicate angajaților restaurantelor. Această metodă este una dificilă si neelegantă de a comanda de la un restaurant.

Aplicația Android prezentată în acest document încearcă să elimine majoritatea neajunsurilor identificate în aplicațiile asemănătoare şi să ofere utilizatorilor acesteia o experiență cât mai utilă şi plăcută, prin funcționalitățile existente şi modul de prezentare al acestora.

1.3. Conținutul lucrării

Structura lucrării este următoarea:

Capitolul 1 (Introducere) prezintă contextul general şi contextul specific proiectului. Contextul general arată evoluția aplicațiilor pentru dispozitive mobile, demonstrând avantajele ca aplicația AndroChef să fie dezvoltată pentru platforma Android. Contextul specific se focalizează pe o descriere generală a aplicației, cu îmbunătățirile pe care aceasta le aduce, comparativ cu celelalte aplicații asemănătoare existente deja, şi pe explicarea necesității apariției acestui tip de aplicație.

Capitolul 2 (Obiectivele Proiectului) prezintă tema propriu-zisă a lucrării şi stabilirea obiectivelor.

Capitolul 3 (Studiu Bibliografic) descrie contextul apariției aplicațiilor pentru comenzi către restaurante şi abordările existente ale aplicațiilor de acest tip.

Capitolul 4 (Analiză și Fundamentare Teoretică) explică principiile funcționale ale aplicației implementate. În acest capitol sunt descrise tehnologiile folosite cu scopul atingerii obiectivelor lucrării. Capitolul cuprinde și cazurile de utilizare ale aplicației și arhitectura conceptuală a unui sistem de tip client-server. De asemenea, este realizată o analiză detaliată a ceea ce va fi utilizat la implementarea sistemului.

Capitolul 5 (Proiectare de Detaliu și Implementare) cuprinde schema generală a aplicației și descrierea componentelor implementate la nivel de modul. Conține și diagrama de clase împreună cu o descrierea a claselor și a metodelor mai importante.

Capitolul 6 (Testare și Validare) cuprinde partea de testare şi validare a sistemului. Evaluarea sistemului va măsura performanța și conformitatea unor funcționalități oferite de aplicație.

Capitolul 7 (Manual de Instalare și Utilizare) prezintă modalitățile prin care aplicația poate fi instalată și instrucțiunile de utilizare.

Capitolul 8 (Concluzii) reprezintă un set de concluzii împreună cu îmbunătățirile care se pot aduce acestui proiect.

Capitolul 2. Obiectivele Proiectului

Acest capitol prezintă obiectivul principal al aplicației, precum și obiectivele secundare. Obiectivul principal al acestei aplicații este reprezentat de *Comanda de grup*.

2.1. Obiectiv principal – Comanda de grup

Scopul acestui proiect este definirea si construirea unui sistem care să faciliteze și să eficientizeze comandarea mesei de la restaurante de către grupurile de persoane, fie ele în cadrul unor companii sau pur si simplu grupuri de prieteni care se află in aceeași locație și doresc să își comande masa de la un anumit restaurant.

Obiectivul de bază este implementarea unei aplicații care să permită crearea de grupuri de persoane cărora li se oferă posibilitatea de plasare a unor comenzi către restaurante în cadrul acelui grup. Acest obiectiv presupune implementarea unor funcționalități prin care să se poată gestiona grupul. Orice utilizator autentificat în aplicație va avea dreptul să inițieze o comandă de grup.

Crearea unui grup se efectuează în momentul în care utilizatorul simplu lansează invitații către alți utilizatori de a participa la comanda sa. Fiecare invitație lansată specifică restaurantul căruia i se va plasa comanda. Cei invitați vor primi un mesaj pe care îl acceptă dacă doresc să ia parte la comanda de grup sau îl ignoră în cazul în care nu sunt interesați.

Fiecare membru al grupului și administratorul de grup își aleg individual produsele, însă în momentul în care se plasează comanda, toate produsele vor fi centralizate într-o comandă unică.

Toți membrii unui grup au dreptul să vizualizeze componența grupului și să observe care dintre membrii grupului și-au finalizat propria comandă în cadrul grupului. Membrii grupului nu vor avea însă dreptul de a vizualiza produsele comandate de către ceilalți membrii. Acest drept îi revine doar administratorului grupului.

Plasarea comenzii de grup este responsabilitatea administratorului, adică a celui care a inițiat-o. Finalizarea comenzii de grup se va materializa prin trimiterea unui email către restaurant cu toate informațiile necesare pentru a putea livra comanda. Un email cu informații legate de ceea ce fiecare membru al grupului a comandat va fi trimis către administratorul grupului, pentru a știi suma de bani pe care trebuie să o primească de la fiecare.

2.2. Obiective secundare

Majoritatea obiectivelor secundare vin în ajutorul implementării funcționalității de comandă de grup sau comandă simplă, însă unele au scopul doar de a oferi utilizatorului anumite funcționalități de care ar putea avea nevoie în timpul utilizării aplicației, cum ar fi contactarea telefonică a restaurantului sau vizualizarea acestuia pe hartă.

2.2.1. Comanda individuală

Comanda de grup este formată din comenzi individuale, cu diferența că o comandă individuală în cadrul grupului nu va mai conține pasul în care trebuie precizat tipul de livrare și adresa la care să se livreze comanda. De asemenea, membrul unui grup nu va avea dreptul să comande de la un alt restaurant față de cel care a fost ales inițial pentru crearea comenzii de grup. Dacă unul dintre membrii grupului selectează un alt restaurant pentru comanda sa, atunci acesta va părăsi automat grupul și va putea efectua o comandă individuală.

Comanda individuală constă în parcurgerea următorilor pași de către utilizator:

a. selectarea restaurantului;
b. selectarea categoriei de produse a restaurantului;
c. selectarea unui anumit produs din lista de produse;
d. vizualizarea detaliată a produsului;
e. adăugarea produsului în coșul de produse, prin precizarea instrucțiunilor speciale și a numărului de itemi din acel produs;
f. pașii b-e vor fi reparcurși până când utilizatorul și-a ales toate produsele dorite;
g. utilizatorul poate să editeze sau să șteargă produse din coș în orice moment;
h. selectarea tipului de livrare (livrare la domiciliu, ridicare de la restaurant, comandă la restaurant);
i. furnizarea adresei la care să se livreze comanda în cazul în care a fost ales tipul de livrare la domiciliu;
j. plasarea comenzii prin trimiterea unui email la adresa restaurantului;

2.2.2. Înregistrare, Autentificare, Recuperare parolă

Sistemul va implementa și funcționalitățile de înregistrare, autentificare și recuperare parolă. Înregistrarea și autentificarea sunt necesare pentru ca membrii unui grup să se poată identifica între ei. De asemenea, datele de contact ale utilizatorului vor trebui trimise împreună cu comanda către restaurant pentru ca angajații restaurantului să poată contacta clientul pentru situații de excepție.

Înregistrarea utilizatorului presupune completarea unui formular cu datele sale personale, iar autentificarea constă în completarea credențialelor de către utilizator și validarea acestora de către sistem.

Funcționalitatea de recuperare a parolei vine în ajutorul utilizatorilor autentificați care au uitat numele de utilizator sau parola. Aceștia vor furniza aplicației email-ul cu care s-au autentificat, iar apoi vor primi de la sistem un email care va conține numele de utilizator și o parolă temporară. Odată ce s-au locat cu parola temporară, utilizatorii vor trebui să aleagă o nouă parolă cu care să se legheze pe viitor.

2.2.3. Obiective auxiliare

Obiectivele auxiliare nu sunt absolut necesare pentru a putea fi utilizate funcționalitățile de comandă de grup sau comandă simplă, însă vin în ajutorul utilizatorilor pentru a le facilita anumite servicii de care ar putea avea nevoie.

Următoarea listă prezintă funcționalitățile auxiliare ale sistemului:

- ✓ furnizarea unei liste de restaurante în ordine crescătoare a distanței dintre restaurant și locația curentă a utilizatorului;
- ✓ furnizarea tuturor restaurantelor dintr-un anumit oraș selectat;
- ✓ posibilitatea de a contacta telefonic restaurantul direct din aplicație;
- ✓ posibilitatea de a localiza restaurantul pe hartă;
- ✓ posibilitatea de vizualizare și editare a profilului utilizatorului;
- ✓ posibilitatea de vizualizare a unui istoric al comenzilor efectuate;
- ✓ posibilitatea de vizualizare a restaurantelor favorite;

Capitolul 3. Studiu Bibliografic

În acest capitol vor fi prezentate o serie de concepte relevante pentru domeniul proiectului de față, care au stat la baza proiectării și implementării cerințelor aplicației. Totodată, se vor descrie pe scurt referințele utilizate în această lucrare și se va efectua o comparație între aplicațiile existente deja pe piață și aplicația de față.

3.1. Aplicații mobile vs. Aplicații web

În acest subcapitol se va prezenta o comparație între aplicațiile mobile și aplicațiile web, pe parcursul ultimilor ani. Evoluția aplicațiilor mobile explică de ce aplicația AndroChef va fi dezvoltată pentru platforma Android și nu va fi o simplă aplicație web.

În articolul [2] apare următoarea afirmație: "Web-ul, așa cum îl știm noi este înlocuit încet, dar sigur, de către aplicațiile mobile." Această tendință se datorează faptului că aplicațiile mobile sunt mult mai ușor de utilizat în comparație cu aplicațiile web, pe un telefon mobil. Aplicațiile mobile sunt create special pentru dispozitivele mobile și prin urmare, se pot folosi de toate avantajele pe care platforma mobilă le are, în timp ce aplicațiile web se pot accesa doar din browser, ceea ce face mai dificilă utilizarea lor de pe un dispozitiv mobil.

Figura 3.1, preluată din articolul [2], evidențiază faptul că utilizatorii de aplicații mobile au depășit utilizatorii de PC-uri, în ceea ce privește utilizarea web-ului.

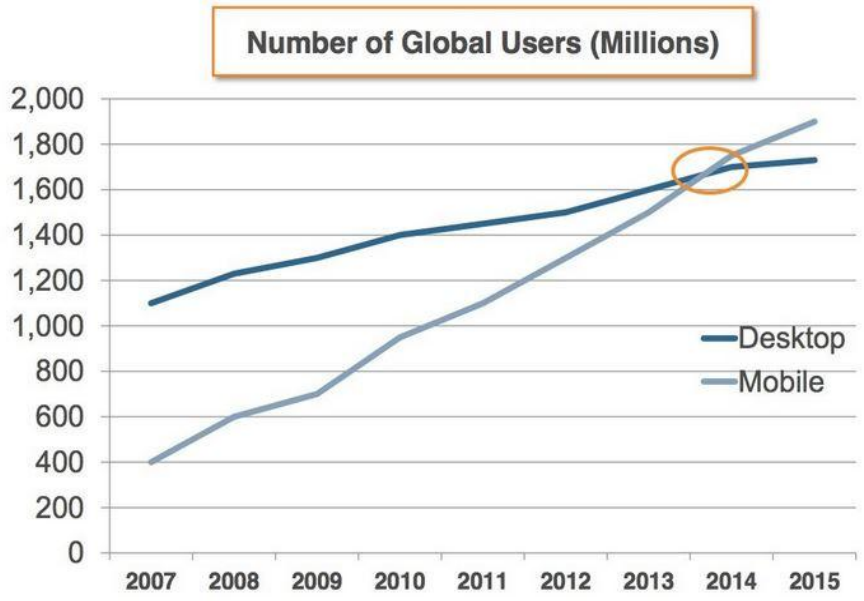

Figura 3.1 Mobile vs. Desktop Users

Din Figura 3.1 se observă preferința utilizatorilor de Internet către dispozitivele mobile, în detrimentul Desktop-urilor. Această preferință pentru dispozitive mobile determină o utilizare tot mai intensă și a aplicațiilor mobile care se pot instala pe acestea.

Figura 3.2, preluată din articolul [2], ilustrează tendința de creștere a utilizării aplicațiilor mobile, în defavoarea aplicațiilor web.

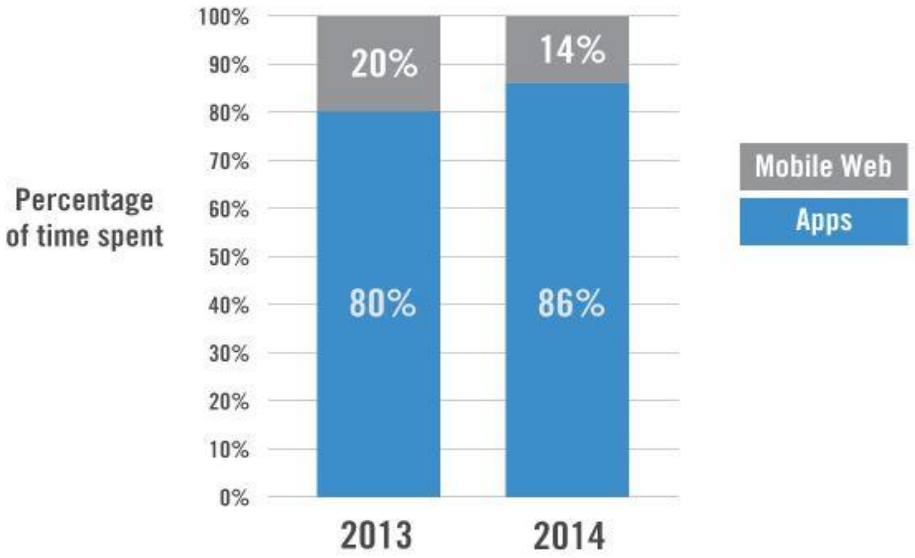

Figura 3.2 Mobile Web vs. Apps

Deși utilizatorii de web pe dispozitive mobile au crescut atât de mult încât i-au depășit pe utilizatorii de web pe Desktop, totuși procentajul de utilizare al web-ului pe dispozitive mobile este în scădere față de cel al aplicațiilor mobile, dacă se ia în calcul anul 2014 față de anul 2013. Un posesor de dispozitiv mobil petrece 86% din timpul total în care folosește dispozitivul, cu aplicațiile mobile, și doar 14% din timp folosește aplicațiile web prin intermediul browser-ului.

Cele prezentate mai sus demonstrează că aplicațiile mobile sunt un domeniu în care merită investit, deoarece utilizatorii de aplicații preferă tot mai mult să utilizeze aplicațiile instalate pe telefonul mobil. Având la dispoziție o gamă diversificată de aplicații gratis care se pot descărca și instala din Play Store, posesorii de dispozitive mobile cu Android preinstalat, au ocazia să vizualizeze informații financiare, informații despre vreme, precum și ultimele știri apărute. De asemenea, există aplicații care se ocupă de managementul timpului, care asistă la plata facturilor, care ajută la gestionarea afacerilor, aplicații cu scop de entertainment și nu în ultimul rând, aplicații care asistă la comandarea de meniuri de la restaurante.

Aplicația AndroChef face parte din categoria aplicațiilor cu ajutorul cărora utilizatorii pot să comande de la un anumit restaurant, facilitând într-o mare măsură întregul proces de comandă.

3.2. Platforma Android vs. iOS

În acest subcapitol se vor prezenta câteva informații legate de platforma Android, împreuna cu beneficiile pe care aceasta le oferă, comparativ cu platforma iOS.

Tabel 3.1 Android vs. iOS

	Android	**iOS**
Widget-uri	Da	Nu, exceptând în NotificationCenter
Compania/Dezvoltator	Google	Apple Inc.
Familia de OS-uri	Linux	OS X, UNIX
Personalizare	Foarte multă. Se poate schimba aproape orice.	Limitată
Release inițial	23 Septembrie 2008	29 Iulie 2007
Dezvoltat în	C, C++, Java	C, C++, Objective-C
Dependent de PC sau Mac	Nu	Nu
Transfer ușor de media	Depinde de model	Cu aplicație desktop
Tipul de sursă	Open source	Closed, cu componente open source
Open source	Kernel, UI și câteva aplicații standard	iOS Kernel nu e open source, dar e bazat pe open-source Darwin OS
Feature-uri de apel suportate	Auto-respond	Auto-respond, call-back reminder, do not disturb mode
Internet browsing	Google Chrome (sau Android Browser pe versiuni mai vechi; și alte browsere disponibile)	Mobile Safari (alte browsere sunt disponibile)
Disponibil pe	Multe telefoane și tablete, incluzând Kindle Fire, LG, HTC, Samsung, Sony, Motorola, Nexus, și altele	iPod Touch, iPhone, iPad, Apple TV
Interface	Touch screen, Smartwatch	Touch screen
Messaging	Google Hangouts	iMessage
Voice commands	Google Now	Siri
App store	Google Play – 1,000,000+ Apps Alte aplicații cum ar fi Amazon și Getjar.	Apple app store – 1,000,000+ Apps
Market share	81% din smartphone-uri, 3.7% din tabletele din America de Nord și 44.4% din tabletele din Japonia (Ian. 2013). În Statele Unite în Q1 2013 – 52.3% telefoane, 47.7% tablete.	12.9% din smartphone-uri, 87% din tabletele din America de Nord și 40.1% din tabletele din Japonia (Ian. 2013)
Ultimul release stabil	Android 4.4 KitKat (Octombrie 2013)	7.1 (10 Martie 2014)

Tabelul 3.1, ilustrează principalele diferențe și asemănări dintre cele 2 platforme, Android și iOS, fiind preluat din sursa [3].

Din articolul [4] se poate extrage o definiție a platformei Android. Android este un sistem de operare pentru smartphone-uri, asemeni platformelor iOS care rulează pe iPhone și BlackBerry OS care funcționează pe telefoanele BlackBerry. Platforma a fost creată de Google și a rulat pentru prima dată pe telefonul T Mobile G1, lansat în 2008.

Platforma Android este de tipul open source. Spre deosebire de iOS-ul iPhone-ului, care este strict sub controlul Apple, sau de cel al BlackBerry-ului, care se rezumă strict la dispozitivele RIM, Android nu este legat de un dispozitiv anume, orice producător hardware putând dezvolta pe această platformă.

Un alt avantaj al platformei Android este interfața intuitivă și flexibilă, dotată cu o multitudine de opțiuni, fiind la fel de rapidă ca și cea a iOS-ul. În plus, cei de la Google lucrează în permanență la îmbunătățirea platformei.

În articolul [5] se prezintă plusurile pe care Android le are, comparativ cu iOS-ul. Android-ul este un sistem deschis, în care oricine poate utiliza produsul gratuit pentru a construi aplicații, oricine poate instala orice și de oriunde pe echipament. Pe lângă magazinul oficial Google Play Store, de pe care se pot instala aplicații, mai există o mulțime de alte surse de pe care se pot descărca aplicații. Android-ul poate fi utilizat gratuit pentru construirea dispozitivelor, produselor și serviciilor. A ajuns să fie folosit chiar și în roboți, drone și alte echipamente fără legătură cu smartphone-urile sau tabletele.

Toate facilitățile mai sus menționate nu se regăsesc și la Apple, deoarece iOS reprezintă un sistem închis, în care există o singură companie care dezvoltă produsul și nimeni nu poate construi aplicații noi fără aprobarea celor de la Apple. Există o singură sursă de unde se pot descărca și instala aplicații, și anume, Apple Store. Prin urmare, iOS nu este cel mai potrivit mediu pentru dezvoltarea de noi aplicații, spre deosebire de Android.

Aplicația AndroChef va fi dezvoltată special pentru platforma Android, având șanse mult mai mari de a intra pe piața aplicațiilor pentru Android, dacă se iau în calcul cele mai sus menționate privind strictețea celor de la Apple în comparație cu libertatea oferită de cei de la Google.

3.3. Sisteme similare

Odată cu dezvoltarea tot mai multor aplicații mobile pe diferite domenii, au fost dezvoltate și o serie de aplicații pentru eficientizarea comenzilor către restaurante. Din acest domeniu face parte și aplicația AndroChef, care pe lângă funcționalitățile clasice oferite și de către alte aplicații din această categorie, se remarcă prin cea mai importantă funcționalitate a sa, și anume comanda de grup. Această funcționalitate diferențiază aplicația AndroChef de restul aplicațiilor Android deja existente, care se axează în general pe comenzi simple.

În articolul [6] se prezintă un top 22 al celor mai bune aplicații pentru restaurante, disponibile pe dispozitive mobile. Aceste aplicații asistă utilizatorul la găsirea restaurantului cel mai potrivit de la care să comande, la crearea unor rezervări, la alegerea unor meniuri sănătoase și la alegerea celei mai potrivite opțiuni din lista de produse.

Există și aplicații care permit utilizatorilor să încarce poze cu felul lor de mâncare preferat de la un anumit restaurant, astfel încât o persoană care primește o recomandare de a lua masa la acel restaurant să știe care opțiuni sunt de luat în considerare.

Alte aplicații oferă posibilitatea de a afla unde sunt localizate restaurantele cu un anumit specific, de exemplu, cu specific vegetarian, indiferent de locația în care utilizatorul se află. Unele au capacitatea de a recomanda restaurante pe baza preferințelor utilizatorului. Pentru această funcționalitate este nevoie ca mai întâi aplicația să ia la cunoștință de aceste preferințe.

Din toată această diversitate de aplicații pentru restaurante disponibile, alegerea celei mai potrivite aplicații care să satisfacă nevoile utilizatorului, poate deveni o decizie dificilă.

Alegerea celei mai bune aplicații se efectuează pe baza mai multor criterii:

- Ușurința de utilizare – aplicațiile de top trebuie să se focuseze pe o interfață ușor de folosit, care să nu pune obstacole utilizatorului.
- Listare completă de restaurante – aplicațiile care merită luate în considerare trebuie să dispună de o listă vastă și diversificată de restaurante.
- Prețul aplicației – aplicațiile pentru restaurante sunt de regula gratis și orice aplicație de acest tip care nu ar fi gratis, nu ar merita să fie luată în considerare.

Cele trei criterii mai sus menționate sunt preluate din articolul [6], pe baza lor, efectuându-se topul celor 22 cele mai bune aplicații pentru restaurante. Aplicația care a ieșit câștigătoare în acest top se numește Urbanspoon. Interfața sa se poate vizualiza în Figura 3.3.

3.3.1. AndroChef vs. Urbanspoon

Aplicația Urbanspoon este o aplicație “all-in-one” de luat masa de la restaurant pentru orice tip de utilizator. Aceasta se remarcă prin mai multe funcționalități:

- De a furniza sugestii random în momentul în care utilizatorul mișcă telefonul.
- De a găsi restaurantele cele mai apropiate pe baza GPS-ului integrat in telefon.
- De a căuta într-o listă mare de restaurante rating-urile utilizatorilor, precum și recomandările oferite de către aceștia.

Aplicația AndroChef nu se va ocupa de efectuarea de recomandări și de plasarea de rating-uri restaurantelor de către utilizatori, ci are ca și obiectiv principal crearea de comenzi de grup. Cele două funcționalități menționate pot fi integrate și ele la un moment dat în sistem, însă deocamdată nu au o implicație importantă în realizarea obiectivului principal.

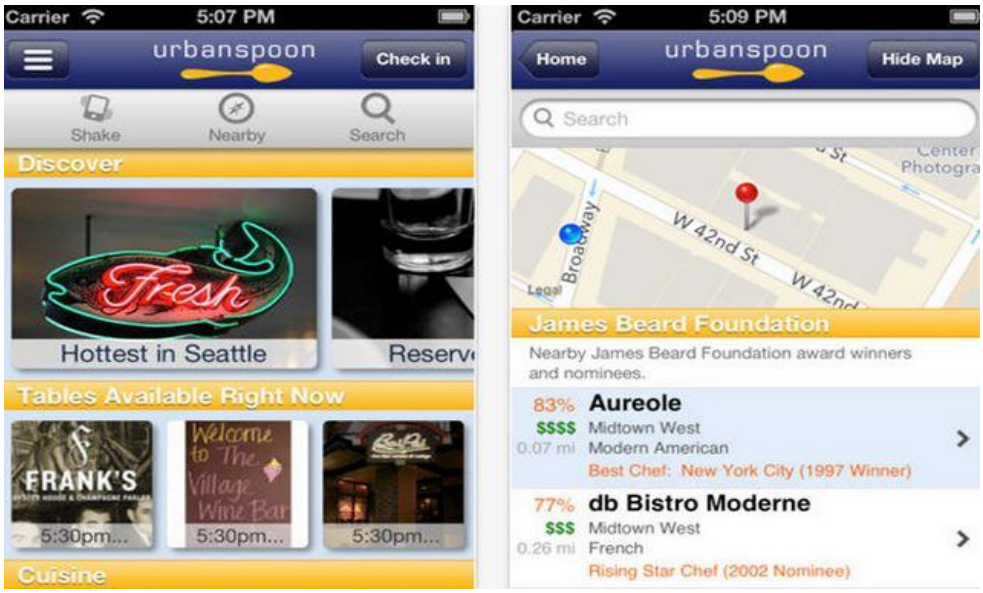

Figura 3.3 Interfața aplicației Urbanspoon

Tabel 3.2 AndroChef vs. Urbanspoon

Funcționalități	AndroChef	Urbanspoon
Înregistrare	Da	Da
Autentificare	Da	Da
Recuperare parolă	Da	Da
Căutarea de restaurante pe bază distanței și locației curente	Da	Da
Adăugarea de rating pentru restaurante	Nu	Da
Efectuarea recomandărilor	Nu	Da
Efectuarea de comenzi de grup	Da	Nu
Efectuarea de comenzi simple	Da	Da
Invitarea unui utilizator la comanda de grup	Da	Nu
Adăugarea unui prieten în lista de prieteni	Nu	Da
Localizarea restaurantului pe hartă	Da	Da
Apelarea restaurantului direct din aplicație	Da	Nu
Căutarea de meniuri împreună cu poze ale acestora	Da	Da
Schimbarea locației pe baza GPS-ului în timpul mersului	Da	Da
Afișarea restaurantelor favorite	Da	Da

Tabelul 3.2 prezintă o comparație pe mai multe criterii decât cele prezentate mai sus, a aplicațiilor AndroChef și Urbanspoon.

Informațiile cu privire la funcționalitățile pe care Urbanspoon le oferă au fost extrase din [7], fiind site-ul lor de prezentare.

Din Tabelul 3.2 observăm că Urbanspoon se focusează pe rating-uri și pe recomandări, în timp ce AndroChef are ca și principală funcționalitate comanda de grup, fără a avea un sistem prin care să monitorizeze părerile utilizatorilor. Aceasta este principala diferență dintre cele două aplicații.

Deși Urbanspoon este în topul preferințelor utilizatorilor, datorită funcționalităților pe care le pune la dispoziție și a modului lor de prezentare, AndroChef se menține la un nivel apropiat în ceea ce privește funcționalitățile de bază pe care le pune la dispoziție, având în plus, funcționalitatea de comandă de grup.

3.3.2. AndroChef vs. FoodPanda

Un top al celor mai utile aplicații de pe mobil, efectuat în orașul București și care se regăsește în articolul [8], prezintă pe locul doi, o aplicație pentru comenzi către restaurante, demonstrând încă o dată importanța acestui tip de aplicație. Pe locul întâi în acest top, este aplicația Urban Hero, care reprezintă cea mai ușoară modalitate de a semnala neregulile de pe strada.

Aplicația de pe locul doi este FoodPanda, care reușește să adune sute de restaurante într-un singur loc, pe același principiu, bazându-se și AndroChef. Utilizatorii au posibilitatea să comande de la cele mai apropiate restaurante din zona lor, prin intermediul acestei aplicații. Este, totodată, cea mai mare platformă globală de livrări de mâncare, care nu demult s-a lansat și la București, Cluj-Napoca, Timișoara și Iași. Mâncarea se poate comanda prin această platformă de pe orice dispozitiv conectat la internet (desktop, laptop, tabletă, smartphone, etc).

FoodPanda colaborează în prezent cu peste 15.000 de restaurante din întreaga lume și este lider de piață în majoritatea țărilor în care operează. Brandul FoodPanda s-a extins foarte rapid, doar în anul 2013, lansându-se în 17 țări din cele peste 40 în care activează. În prezent, FoodPanda, împreună cu brandul afiliat HelloPanda, se situează pe primul loc în topul mondial al celor mai descărcate aplicații pentru comenzi de mâncare.

Unul dintre marile avantaje ale acestei aplicații este faptul că se poate utiliza atât platforma online, cât și platforma mobilă FoodPanda.

FoodPanda operează în prezent în următoarele țări:

- Asia-Pacific: India, Thailand, Indonesia, Pakistan, Singapore, Malaysia, Vietnam, Taiwan, Bangladesh, Kazakhstan, Azerbaijan;
- Europa: Russia, Poland, Ukraine, Hungary, România, Czech Republic, Slovakia, Bulgaria, Croația, Serbia;
- Africa: Nigeria, Morocco, Kenya, Ghana, Senegal, Ivory Coast, Rwanda, Tanzania, Uganda, Algeria;
- Middle East: Saudi Arabia, Jordan, Lebanon;
- Latin America: Brazil, Mexico, Argentina, Chile, Colombia, Peru.

Prin accesarea aplicației, se detectează automat strada pe care se află clientul și se afișează o listă cu cele mai apropiate restaurante care fac livrări în acea zonă, specificul acestora, meniul, comanda minimă și taxa de livrare. AndroChef dispune de această funcționalitate, iar localizarea celor mai apropiate restaurante se efectuează prin intermediul GPS-ului telefonului care detectează locația curentă a clientului.

După efectuarea unei comenzi, clienții primesc un SMS care confirmă comanda și timpul de livrare. Plata comenzii se face momentan, doar cash la livrare, însă pentru perioada următoare, clienții vor putea opta și pentru plata online, prin card bancar.

În Figura 3.4 se poate vizualiza interfața aplicației FoodPanda.

Figura 3.4 Interfața aplicației FoodPanda

Aplicația FoodPanda este una dintre cele mai răspândite aplicații pentru comenzi către restaurante din lume, fiind renumită în foarte multe țări și având în plan să se extindă și în alte țări unde momentan nu este disponibilă. Această aplicație oferă utilizatorilor toate funcționalitățile necesare pentru a putea efectua o comandă către un anumit restaurant, însă nu oferă posibilitatea efectuării unei comenzi de grup.

Tabelul 3.3 prezintă o comparație între aplicația AndroChef și FoodPanda, pe baza mai multor funcționalități.

Tabel 3.3 AndroChef vs. FoodPanda

Funcționalități	**AndroChef**	**FoodPanda**
Înregistrare	Da	Da
Autentificare	Da	Da
Recuperare parolă	Da	Da
Afișarea restaurantelor din orașul selectat	Da	Da
Afișarea restaurantelor în ordine crescătoare a distanței față de locația curentă	Da	Nu
Precizarea tipurilor de comandă (livrare, ridicare, la restaurant) la vizualizarea listei de restaurante	Da	Nu
Posibilitatea de localizare pe hartă a restaurantului	Da	Da
Permite adăugarea de review-uri pentru restaurante	Nu	Da
Afișarea restaurantelor în ordine alfabetică	Da	Da
Afișarea restaurantelor în funcție de timpul de livrare, evaluări, taxa de livrare, comanda minimă	Nu	Da
Posibilitatea de a suna direct la restaurant	Da	Nu
Posibilitatea de a comanda direct în restaurant	Da	Nu
Filtrarea restaurantelor după un anumit specific	Nu	Da
Vizualizarea restaurantelor favorite	Da	Nu
Permite plasarea unei comenzi simple	Da	Da
Permite plasarea unei comenzi de grup	Da	Nu

AndroChef dispune de toate funcționalitățile de bază necesare pentru a efectua o comandă simplă sau o comandă de grup, având ca și dezvoltări ulterioare o parte din funcționalitățile oferite de FoodPanda, cum ar fi filtrarea după timpul de livrare, după taxa de livrare, după comanda minimă sau înregistrarea părerilor utilizatorilor.

3.3.3. AndroChef vs. EatFab

EatFab este o aplicație web creată special pentru companii, care vine în ajutorul angajaților atunci când aceștia doresc să își comande de la unul dintre restaurantele cu care compania are un contract. Restaurantele care sunt înscrise în această aplicație semnează un contract cu compania, prin care acestea se angajează să introducă zilnic meniul în aplicație. Această aplicație nu are scopul de a facilita efectuarea de comenzi individuale, ci doar în cadrul companiei.

Interfața aplicației EatFab se poate vizualiza în Figura 3.5, iar aplicația se poate accesa prin adresa [9].

Figura 3.5 Interfața aplicației EatFab

Unul dintre dezavantajele acestei aplicații este faptul că angajații restaurantelor trebuie să introducă doar meniul special din acea zi, fără a oferi angajaților companiei posibilitatea de a comanda și alte produse care se pot găsi în acel restaurant. Însă această restricție poate fi benefică pentru cei care obișnuiesc să își comande doar meniul zilei de la restaurant, nemaifiind nevoiți să caute prin întreaga listă de produse. Un avantaj este, totodată, faptul că angajații companiilor pot să vizualizeze meniul pentru întreaga săptămână, știind din timp dacă vor fi nevoiți să comande de la un alt restaurant, în cazul în care meniul dintr-o anumită zi nu este pe placul lor.

O altă funcționalitate a aplicației EatFab este gestionarea unui cont, în care angajatul companiei trebuie să depună bani pentru a-și putea comanda de la unul dintre restaurante. Acest cont vine în ajutorul celor care obișnuiesc să comande des din această aplicație și nu doresc ca de fiecare dată când comandă, să fie nevoiți să plătească responsabilului suma de bani. Un neajuns al acestei funcționalități este că angajatul trebuie să depună banii din timp pentru a putea efectua o comandă, fapt care îl restricționează pe utilizator să își comande de la unul dintre restaurantele aplicației pentru a-și consuma banii deja introduși în cont, nemaiavând posibilitatea de a alege un restaurant dintr-o altă aplicație. Existența unui cont presupune ca în permanență, unul

dintre angajați să se ocupe de colectarea banilor și de evidența acestora. Dacă acest cont nu ar exista, nu ar mai fi necesar ca o persoană să păstreze banii angajaților până aceștia se decid să comande, lucru care implică o atenție în plus.

O îmbunătățire a acestui cont ar fi, ca să permită și utilizatorilor care încă nu au depus banii în cont, să comande meniul zilei de la unul dintre restaurantele disponibile, acesta fiind nevoit să achite suma de plată până la livrarea comenzii.

Tabel 3.4 AndroChef vs. EatFab

Funcționalități	**AndroChef**	**EatFab**
Înregistrare	Da	Da
Autentificare	Da	Da
Recuperare parolă	Da	Da
Afișarea restaurantelor în ordinea crescătoare a distanței față de locația curentă	Da	Nu
Identificarea locației curente prin GPS-ul telefonului	Da	Nu. Nu este o aplicație Android
Afișarea tuturor categoriilor restaurantelor	Da	Nu
Afișarea tuturor produselor pe care restaurantul le oferă	Da	Doar meniul zilei
Afișare meniului zilei pentru întreaga săptămână	Nu	Da
Vizualizarea unui istoric al comenzilor	Nu	Da
Vizualizarea restaurantelor favorite	Da	Nu
Legătură cu rețelele de socializare	Nu	Da
Efectuarea unor comenzi simple	Da	Nu
Efectuarea comenzilor de grup	Da. În mod flexibil	Da. În mod strict
Vizualizarea restaurantului pe hartă	Da	Da
Posibilitatea de a suna direct la restaurant din interiorul aplicației	Da	Nu. Nu este o aplicație Android
Vizualizarea coșului de cumpărături	Da	Nu
Gestionarea unui cont	Nu. Nu se poate aplica în cadrul grupurilor temporare	Da

Tabelul 3.4 prezintă o comparație sintetică între aplicațiile AndroChef și EatFab în privința funcționalităților pe care le pun la dispoziție fiecare.

Gestionarea unui cont este o funcționalitate care se poate aplica doar sistemelor care utilizează grupuri fixe. AndroChef se ocupă cu gestionarea grupurilor temporare, unde nu este posibilă existența unei asemenea funcționalități.

Comanda de grup este funcționalitatea de bază a aplicației EatFab, funcționalitate care se regăsește și la aplicația AndroChef. Conceptul de grup din cadrul EatFab este însă unul inflexibil, deoarece angajații companiei vor putea comanda în cadrul acestui grup doar atunci când se află la firmă, fără a putea folosi aplicația și în timpul liber, în cadrul grupurilor de prieteni. AndroChef oferă această flexibilitate, de a putea crea grupuri noi fără ca utilizatorul să depindă de o locație fixă și de un grup fix. Totodată, aplicația AndroChef poate fi utilizată cu ușurință și în cadrul companiilor.

O altă problemă a acestui tip de aplicație este că puține restaurante doresc să se înscrie în cadrul lor, deoarece preferă tipul de aplicație din care și AndroChef face parte, în care utilizatorii nu trebuie să fie angajații unei companii și să se afle în sediul ei atunci când doresc să comande. Utilizatorii preferă o gamă variată de produse din care pot să aleagă, iar restaurantele se îndreaptă către acele aplicații care pot să le promoveze toate produsele pe care ele le pot oferi.

3.3.4. Concluzii pe baza studiului sistemelor similare

Pe baza studiului sistemelor similare se poate observa că majoritatea sistemelor care oferă funcționalitatea de efectuare a comenzilor către restaurante, nu permit și efectuarea comenzilor de grup.

Aplicația web EatFab, spre deosebire de celelalte aplicații studiate, este creată special pentru comenzi de grup în cadrul companiilor, însă nu permite crearea de grupuri în alt context, cum ar fi, în cadrul unui grup de prieteni sau în cadrul anumitor evenimente. Nu oferă flexibilitate în gestionarea grupurilor.

AndroChef va implementa funcționalitatea de comandă de grup. Grupul va putea fi creat de orice utilizator și va fi șters în momentul în care comanda de grup este plasată. Aplicația AndroChef oferă flexibilitate în gestionarea comenzilor de grup, existența unui grup fiind condiționată doar de o locație comună pentru toți membrii grupului.

Capitolul 4. Analiză și Fundamentare Teoretică

Acest capitol va prezenta arhitectura conceptuală a sistemului, cerințele funcționale captate sub forma cazurilor de utilizare și cerințele non funcționale care specifică atributele sistemului și sunt corelate cu cerințele funcționale. De asemenea, vor fi detaliate tehnologiile alese în dezvoltarea aplicației.

4.1. Arhitectura conceptuală a sistemului

Aplicația AndroChef este un sistem Client-Server, în care partea de Server va fi creată cu ajutorul framework-ului Dancer, în Perl, iar partea de Client va fi dezvoltată în framework-ul Android, având la dispoziție SDK-ul care include un set complet de instrumente de dezvoltare. La final, aplicația va putea fi rulată doar pe dispozitivele pe care este instalat sistemul de operare Android.

Pentru centralizarea informațiilor despre utilizatori, restaurante, meniuri și comenzi de grupuri de persoane, se va utiliza o bază de date, care va fi gestionată prin intermediul sistemului de gestiune al bazei de date relaționale, MySQL.

Diagrama conceptuală generală a sistemul este ilustrată în Figura 4.1, unde este evidențiată comanda în cadrul unui grup de persoane.

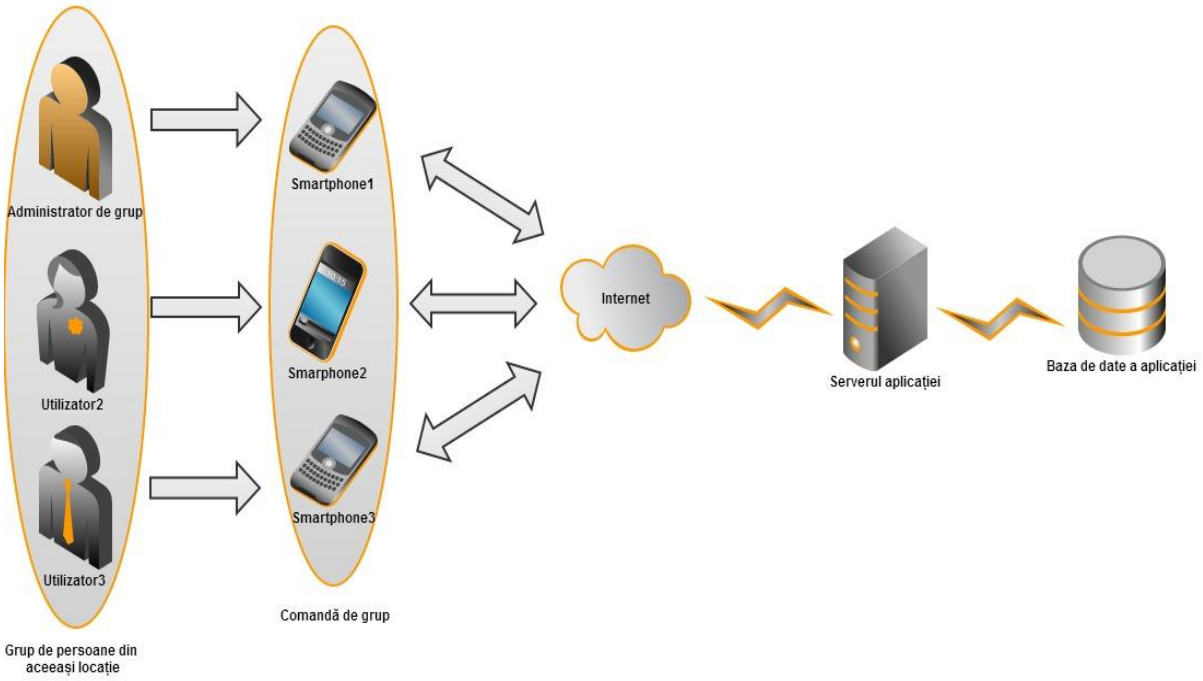

Figura 4.1 Arhitectura conceptuală a aplicației

Pentru ca utilizatorii să poată efectua o comandă de grup, aceștia trebuie să se afle în **aceeași locație**, având fiecare la dispoziție un **telefon inteligent** pe care rulează sistemul de operare Android.

Pentru a se putea realiza conexiunea dintre **smartphone-uri** și **server** este nevoie ca fiecare dintre acestea să dispună de acces la **internet**. Telefonul se poate conecta prin wireless la unul dintre punctele de acces la internet din apropiere sau prin conexiunea de date oferită de către furnizorul de servicii mobile.

În Figura 4.1 se poate identifica **administratorul de grup**, el fiind cel care inițiază comanda de grup prin trimiterea unor invitații celorlalți utilizatori (**utilizator1** și **utilizator2**) interesați de comandă. Cei care primesc invitația la grup, au ocazia să se decidă dacă mai doresc sau nu să participe la comandă. Dacă se decid să participe la grup, mai trebuie doar să accepte invitația pentru a putea fi înregistrați ca făcând parte din grup. Toți ceilalți utilizatori, care fac parte din grupul creat, vor putea să vizualizeze componența grupului până la momentul actual.

Odată ce administratorul de grup a trimis câte o invitație celorlalți doi utilizatori, se efectuează automat și un request către **server**, prin care se creează o nouă comandă de grup. Comanda creată pe server îl are ca și administrator de grup, pe cel care a trimis invitațiile celorlalți utilizatori. Restaurantul căruia i se va trimite în final comanda de grup, este cel ales de inițiatorul grupului. Comanda de grup cu toate atributele setate de către server, este mai apoi salvată în baza de date, prin intermediul unui interfețe de acces.

Aplicația AndroChef are la bază arhitectura Client-Server, clientul fiind reprezentat de aplicația Android, iar serverul de aplicația care oferă web serviciile aplicației client. Serverul, la rândul lui, apelează la o baza de date pentru asigurarea persistenței. Arhitectura Client-Server se poate vizualiza în Figura 4.2.

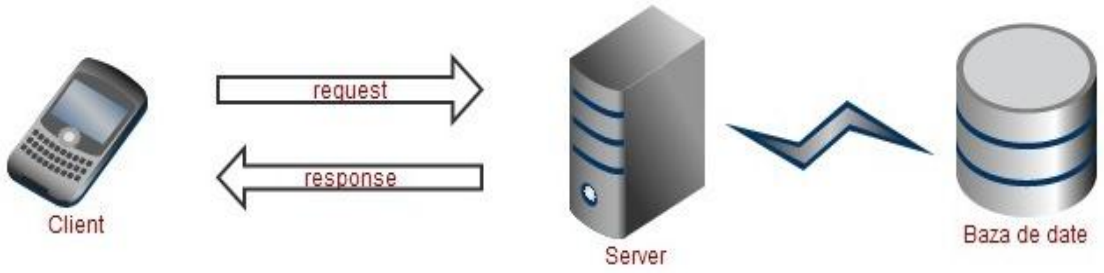

Figura 4.2 Arhitectura Client-Server

Clientul și serverul sunt într-o relație de cooperare pentru a putea îndeplini cerințele funcționale ale întregului sistem. Serverul este cel care oferă servicii clienților, care mai întâi trebuie să inițieze o cerere pentru a beneficia de serviciile disponibile.

Clienții și serverul fac schimb de mesaje prin intermediul modelului request-response. Clientul trimite o cerere, iar serverul returnează un răspuns. Acest schimb de mesaje este un exemplu de comunicare între procese. Pentru a comunica, aplicațiile trebuie să utilizeze un limbaj comun și să urmeze anumite reguli pentru ca și clientul și serverul să știe ceea ce așteaptă. Limbajul și regulile de comunicare sunt definite printr-un protocol de comunicare. Toate protocoalele dintre client și server operează la nivelul layer-ului aplicație. Serverul poate implementa și un API pentru a formaliza și mai mult schimbul de date, cum ar fi web serviciile. Principiul de funcționare al arhitecturii Client-Server a fost preluat din sursa [10].

4.2. Cerințele aplicației

Cerințele aplicației reprezintă ceea ce utilizatorul se așteaptă de la sistem, și anume, capabilități și constrângeri la care un sistem trebuie să se conformeze. Identificarea cerințelor este pasul cel mai important în dezvoltarea sistemului, pentru a putea finaliza cu succes aplicația.

Cerințele prezentate în acest subcapitol descriu funcționalitățile sistemului care facilitează și eficientizează comenzile către restaurante. O parte din cerințele sistemului se regăsesc la majoritatea aplicațiilor de acest tip, însă există și alte funcționalități care diferențiază aplicația AndroChef de ceea ce există momentan pe piață.

4.2.1. Cerințele funcționale

Cerințele funcționale descriu capabilitățile la care trebuie să se supună sistemul. Aceste descrieri trebuie să fie complete și să fie prezentate din perspectiva utilizatorului.

Cea mai importantă funcționalitate a aplicației AndroChef este efectuarea comenzilor de grup. Această funcționalitate constă în mai mulți pași, pasul de creare a comenzii de grup, pasul de invitare a altor utilizatori la comandă, pasul prin care utilizatorii își aleg produsele și pasul prin care inițiatorul comenzii finalizează comanda de grup.

Nu este necesar ca toți utilizatorii invitați la comanda de grup să își aleagă minim un produs, administratorul având posibilitatea să finalizeze comanda în orice moment. Fiecare utilizator care aparține unui grup are posibilitatea să vizualizeze componența acestui grup, iar administratorul are dreptul și de a vizualiza care dintre membrii grupului și-a finalizat comanda individuală. Nici unul dintre membrii grupului nu are dreptul să vadă ceea ce un alt membru și-a ales din lista de produse a restaurantului.

Comanda de grup are asignat un unic administrator, și anume cel care o inițiază. Administratorul este și singura persoană care are dreptul de a finaliza comanda. Totodată, în momentul creării comenzii de grup, îi este asignat automat acesteia, restaurantul selectat de către inițiatorul grupului.

Tabelul 4.1 prezintă cerințele funcționale ale aplicației, în ceea ce privește gestionarea unei comenzi simple și a unei comenzi de grup.

Tabel 4.1 Comandă simplă și comandă de grup

	Descrierea funcționalităților
CF-1	**Sistemul trebuie să permită efectuarea unei comenzi simple**.
CF-2	**Sistemul trebuie să permită efectuarea unei comenzi de grup.**
CF-2.1	Utilizatorul are posibilitatea de a crea un grup prin invitarea altor persoane la comanda sa.
CF-2.2	Utilizatorii invitați la comanda pot sa accepte sau să refuze invitația. Odată acceptată, aceștia vor face parte din comanda de grup.
CF-2.3	Doar inițiatorul comenzii de grup va avea posibilitatea de a finaliza comanda.

Tabelul 4.1 prezintă un sumar al cerințelor funcționale ale aplicației AndroChef.

Tabel 4.2 Sumar al cerințelor funcționale de bază

	Descrierea funcționalităților
CF-1	**Utilizatorul trebuie să aibă posibilitatea de a se înregistra în aplicație.**
CF-2	**Utilizatorul trebuie să poată să se autentifice în aplicație.**
CF-2.1	Odată ce utilizatorul s-a autentificat în aplicație, trebuie să rămână autentificat până în momentul în care se deloghează.
CF-2.2	Utilizatorul care nu este autentificat în aplicație are dreptul doar să vizualizeze restaurantele și meniurile acestora.
CF-3	**Dacă utilizatorul nu-și mai amintește parola contului, acesta are posibilitatea de a și-o recupera.**
CF-4	**Sistemul trebuie să furnizeze o listă cu toate orașele în care aplicație este disponibilă.**
CF-5	**Sistemul trebuie să furnizeze o listă cu mai multe restaurante din care utilizatorul să aibă dreptul să selecteze unul dintre acestea.**
CF-5.1	Utilizatorul trebuie să aibă posibilitatea de a ordona restaurantele în ordine alfabetică.
CF-5.2	Utilizatorul trebuie să aibă posibilitatea de a ordona restaurantele în ordine crescătoare a distanței dintre restaurant și locația utilizatorului.
CF-5.3	Utilizatorul trebuie să aibă posibilitatea de a vizualiza toate restaurantele dintr-un anumit oraș selectat de către acesta.
CF-5.4	Utilizatorul poate să vizualizeze restaurantele favorite, de la cel de la care a efectuat cele mai multe comenzi la cel cu cele mai puține comenzi.
CF-6	**Sistemul trebuie să furnizeze informații despre fiecare restaurant în parte.**
CF-6.1	Pentru fiecare restaurant trebuie să se poată vizualiza numele, specificul restaurantului, adresa, poza.
CF-6.2	Pentru fiecare restaurant trebuie specificate tipurile de livrare de care dispune: livrare la domiciliu, ridicare, comandare la restaurant.
CF-6.3	Dacă utilizatorul efectuează o filtrare crescătoare în funcție de distanță, atunci trebuie afișată și distanța aferentă.
CF-7	**Sistemul trebuie să permită vizualizarea pe hartă a locației unui restaurant.**
CF-8	**Utilizatorul trebuie să aibă posibilitatea de a suna direct la restaurant.**
CF-9	**Sistemul trebuie să prezinte toate categoriile de produse ale unui restaurant selectat de către utilizator.**
CF-10	**Sistemul trebuie să prezinte toate produsele dintr-o categorie selectată, împreună cu denumirea, cantitatea, prețul și poza acestora.**
CF-11	**Utilizatorul trebuie să-și gestioneze coșul de produse, prin adăugarea de produse în coș, specificând numărul de itemi și instrucțiunile speciale, prin ștergerea și editarea produselor deja existente.**

4.2.2. Cerințele non-funcționale

Spre deosebire de cerințele funcționale, cele non-funcționale reprezintă proprietăți și impun constrângeri asupra sistemului. Acestea specifică atributele sistemului și nu ceea ce sistemul trebuie să facă.

Aplicația AndroChef este proiectată astfel încât să suporte un număr mare de utilizatori care-și folosesc propriile dispozitive concurent. Utilizatorii nu vor avea nevoie de o prealabilă instruire înainte de utilizarea aplicației, deoarece interfața va fi una intuitivă și ușor de folosit.

Utilizabilitatea descrie ușurința cu care un sistem poate fi învățat și utilizat. Este o cerință non-funcțională la fel de importantă ca și cele funcționale, deoarece aceasta conduce la creșterea numărului de utilizatori ai aplicației, în detrimentul altor aplicații de același tip. **Suportabilitatea, performanța** și **disponibilitatea** se află și ele printre cele mai importante cerințe non-funcționale ale sistemului.

Deoarece utilizatorii aplicației nu pot fi instruiți înainte de folosirea aplicației, **utilizabilitatea** devine o caracteristică critică a sistemului. Interfața aplicației trebuie să fie proiectată astfel încât să fie intuitivă și user friendly. Procesul de creare și gestionare a grupurilor de persoane trebuie să conțină pași cât mai simpli, pentru a reduce timpul necesar efectuării unei comenzi de grup.

Odată ce utilizatorul s-a înregistrat și autentificat în aplicație, chiar dacă părăsește aplicația pentru o anumită perioadă de timp, sesiunea va fi reținută, iar acesta nu va fi nevoit să se autentifice din nou la următoarea accesare a aplicației. Această funcționalitate este o caracteristică de utilizabilitate, scutind utilizatorul de efectuarea unor pași inutili.

Mentenabilitate se referă la abilitatea sistemului de a fi ușor de modificat și întreținut pentru a putea integra ușor noi funcționalități sau a se putea modifica unele funcționalități deja existente. Partea de server trebuie structurată astfel încât să permită adăugarea de noi servicii necesare pentru aplicația angajaților restaurantelor, prin care pot vizualiza și gestiona comenzile primite și pentru aplicația CMS, prin care sunt gestionate meniurile restaurantelor. Partea de client trebuie structurată pe module și trebuie să respecte șabloanele de design coeziune înaltă si cuplare slabă.

Cerințele de **performanță** se referă de obicei la timpul de răspuns al aplicației și la capacitatea acesteia. Comenzile efectuate de către utilizatorii aplicației trebuie să ajungă într-un timp cât mai scurt la angajații restaurantelor pentru a putea onora comenzile. Aplicația trebuie să păstreze timpul de acces la lista restaurantelor în limite normale, chiar dacă numărul restaurantelor înscrise în aplicație crește considerabil. De asemenea, creșterea numărului de utilizatori nu va afecta timpul de răspuns al sistemului. Timpul necesar pentru ca o comandă să ajungă la unul dintre angajații restaurantului trebuie să fie de cel mult 1 minut, iar procesul de efectuare a unei comenzi de către utilizator va dura în medie 2 minute.

Disponibilitatea descrie măsura în care sistemul trebuie să funcționeze pentru utilizatori. Acesta trebuie să fie disponibil 24 de ore din 24, utilizatorilor, iar timpul de recuperare în cazul căderii unuia dintre servere să fie sub limita de 15 minute. Sistemul trebuie să fie implementat iterativ pentru ca eventualele buguri să poată fi descoperite din timp și fixate, astfel încât sistemul să funcționeze la parametrii normali.

Tabelul 4.3 prezintă un sumar al cerințelor non-funcționale ale aplicației.

Tabel 4.3 Sumar al cerințelor non-funcționale

	Descriere
CNF-1	Utilizabilitate – interfață intuitivă și user friendly
CNF-2	Mentenabilitate – structură pe module coezive și slab cuplate
CNF-3	Performanță – comenzi efectuate într-un timp cât mai scurt
CNF-4	Disponibilitate – dezvoltare iterativă pentru fixarea din timp a bugurilor

4.3. Modelul cazurilor de utilizare

Modelul cazurilor de utilizare este definit de către Procesul Unificat în cadrul disciplinei de Cerințe și cuprinde întregul set de cazuri de utilizare. Este un model al funcționalităților sistemului și al mediului în care sistemul este utilizat, conform [11].

Un scenariu este o secvență specifică de acțiuni și interacțiuni între actori și sistem. Se mai numește și instanță de caz de utilizare.

Un caz de utilizare este o colecție de scenarii de succese și eșecuri care descriu actorii utilizând sistemul pentru a atinge un anumit scop. Cazurile de utilizare ale aplicației AndroChef vor fi captate atât sub formă scrisă, cât și sub forma de diagramă de cazuri de utilizare.

4.3.1. Diagrama cazurilor de utilizare

Diagrama UML a cazurilor de utilizare ilustrează numele acestora și ale actorilor sistemului, împreună cu relațiile dintre aceștia.

Actorul este un obiect cu un comportament specific (persoană, sistem computerizat). Actorii sistemului AndroChef sunt utilizatorul simplu, administratorul de comandă de grup și membrul unui grup. Administratorul de comandă de grup reprezintă o specializare a utilizatorului simplu, adică toate funcționalitățile de care dispune utilizatorul simplu sunt disponibile și administratorului de grup.

Orice utilizator simplu poate deveni administrator de comandă de grup dacă dorește ca și alte persoane să facă parte din comanda sa. Atunci când utilizatorul simplu trimite invitații de participare la comanda sa și altor utilizatori, acesta devine administrator de comandă de grup, fiind cel care se ocupă și de finalizarea comenzii de grup. În momentul în care comanda de grup este finalizată de către administrator, el devine din nou utilizator simplu care are posibilitatea de a efectua comenzi simple sau de a crea alte comenzi de grup.

Invitațiile sunt trimise de către administratorul de grup, utilizatorilor simpli. Aceștia trebuie să se decidă daca vor sau nu să participe la comanda de grup. Dacă decizia finală este de a nu participa la grup, invitații vor rămâne tot utilizatori simpli. În cazul în care se decid să accepte invitația la grup, aceștia vor deveni membrii ai grupului.

Membrii grupului sunt asemenea administratorilor de grup, specializări ale utilizatorilor simpli, având posibilitatea de a utiliza oricare din cazurile de utilizare specifice utilizatorilor simpli.

Membrii unui grup sunt constrânși să își comande de la restaurantul pentru care a fost creată comanda de grup. Procesul de selectare al produselor de la acel restaurant rămâne același până în momentul în care membrul grupului își finalizează comanda individuală în cadrul acestuia. Membrul simplu al grupului nu mai trebuie să parcurgă pașii de selectare a metodei de livrare (livrare la domiciliu, ridicare, comandă la restaurant) și de furnizare a adresei complete la care să fie livrată comanda, în cazul în care a fost selectată livrarea la domiciliu. Această sarcină îi revine administratorului de grup.

Figura 4.3 ilustrează diagrama cazurilor de utilizare mai importante ale întregului sistem. Această diagramă surprinde cei trei actori ai sistemului împreună cu cazurile de utilizare specifice fiecăruia.

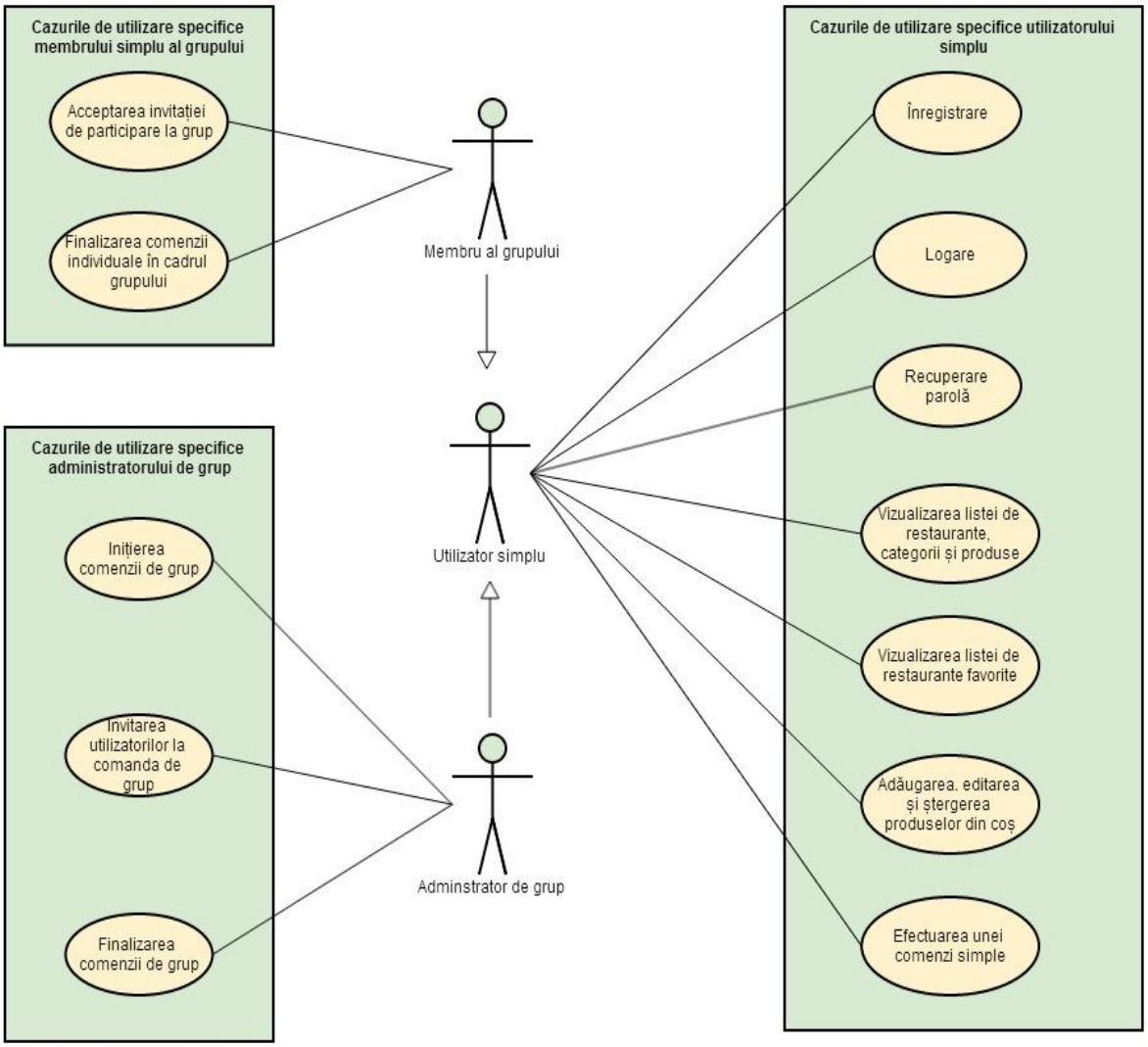

Figura 4.3 Diagrama cazurilor de utilizare

Din diagrama cazurilor de utilizare se poate observa că administratorul de grup si membrul unui grup sunt specializări ale utilizatorului simplu, deoarece primii doi pot utiliza funcționalitățile specifice celui din urmă.

4.3.2. Înregistrare, Autentificare, Recuperare parolă

a) Înregistrare utilizator

Toți utilizatorii trebuie să se înregistreze pentru a putea efectua o comandă către unul dintre restaurante. Utilizatorii care doresc doar să vizualizeze lista de restaurante și meniurile fiecăruia nu trebuie să fie înregistrați în aplicație.

Pentru a se înregistra în aplicație, utilizatorii trebuie să completeze un formular, conținând datele personale ale acestora. În cele ce urmează va fi prezentat cazul de utilizare Înregistrare utilizator:

Actor principal: Utilizatorul simplu

Participanți și interese: Utilizatorul simplu: dorește să se înregistreze în aplicație pentru a putea efectua comenzi către restaurante.

Precondiții: Aplicația trebuie să fie instalată și pornită.

Post condiții: Utilizatorul simplu este înregistrat în aplicație.

Principalul scenariu de succes (Flux de Bază):

1. Utilizatorul accesează secțiunea de înregistrare.
2. Utilizatorul introduce datele sale personale, împreună cu numele de utilizator și parola aleasă.
3. Utilizatorul trimite datele completate pentru a fi salvate de către sistem.
4. Noul cont este salvat cu succes de către sistem.
5. Utilizatorul primește un mesaj de înregistrare cu succes. Utilizatorul are acum posibilitatea de a se autentifica în aplicație.

Extensii (Fluxuri Alternative):

4a. Sistemul detectează că nu au fost completate toate datele obligatorii:

1. Sistemul anulează înregistrarea utilizatorului și trimite acestuia un mesaj corespunzător.

4b. Emailul trimis este invalid:

1. Sistemul anulează înregistrarea și trimite un mesaj de tipul “Emailul introdus nu este valid”.

4c. Emailul sau numele de utilizator trimise sunt deja salvate în sistem:

1. Sistemul informează utilizatorul ca este deja înscris în aplicație cu acel email sau că numele de utilizator este deja folosit.

4d. Parolă trimisă nu este destul de puternică (lungimea caracterelor >= 8 și să conțină cel puțin o cifră și o literă mare):

1. Sistemul anulează înregistrarea începută și informează utilizatorul.

b) Autentificare utilizator

Orice tip de utilizator trebuie să se autentifice în aplicație pentru a putea efectua o comandă simplă sau o comandă de grup. Pentru a se autentifica, acesta trebuie să furnizeze sistemului numele de utilizator și parola alese în momentul înregistrării.

Utilizatorul simplu poate să vizualizeze lista de restaurante și lista cu produsele fiecărui restaurant chiar dacă nu este autentificat în sistem. Odată ce utilizatorul s-a autentificat în sistem, iar apoi a părăsit aplicația fără a selecta opțiunea de delogare, la redeschiderea aplicației, acesta va fi redirecționat către lista de restaurante.

În continuare, este prezentat cazul de utilizare Autentificare utilizator.

Actor principal: Utilizatorul simplu

Participanți și interese: Utilizatorul simplu: dorește să se autentifice în sistem pentru a putea efectua comenzi simple sau comenzi de grup.

Precondiții: Aplicația trebuie să fie instalată și pornită.
Utilizatorul simplu trebuie să fie înregistrat în aplicație.

Post condiții: Utilizatorul simplu este autentificat în sistem.

Principalul scenariu de succes (Flux de Bază):
1. Utilizatorul simplu introduce numele de utilizator și parola în aplicație.
2. Utilizatorul trimite datele completate pentru a fi validate de către sistem.
3. Sistemul validează datele trimise de către utilizator.
4. Sistemul redirecționează utilizatorul către lista de restaurante înscrise în sistem.

Extensii (Fluxuri Alternative):
1a. Utilizatorul este deja autentificat în aplicație la deschiderea acesteia:
1. Se trece la efectuarea pasului 4 din fluxul principal.
3a. Parola furnizată de către utilizator este o parolă temporară trimisă prin email acestuia după efectuarea pașilor 1-4 ai fluxului de bază din cazul de utilizare Recuperare parolă:
1. Se parcurg pașii 6-10 din fluxul cazului de utilizare mai sus menționat
3b. Nu au fost furnizate unul dintre numele de utilizator sau parola:
1. Utilizatorul nu este autentificat și primește un mesaj corespunzător.
3c. Numele de utilizator sau parolă nu se potrivesc cu baza de date:
1. Utilizatorul nu este autentificat și primește un mesaj de eroare de la sistem.

c) Recuperare parolă

Utilizatorul simplu are posibilitatea de a-și recupera parola uitată prin intermediul acestei secțiuni. Utilizatorul primește un email cu o parolă temporară care poate fi folosită o singură dată de către acesta. Această parolă temporară are rolul de a redirecționa utilizatorul către secțiunea de resetare a parolei. În cele ce urmează, va fi descris în detaliu acest caz de utilizare.

Actor principal: Utilizatorul simplu

Participanți și interese: Utilizatorul simplu: dorește să își recupereze parola pierdută pentru a putea efectua din nou comenzi simple sau comenzi de grup.

Precondiții: Aplicația trebuie să fie instalată și pornită.
Utilizatorul simplu trebuie să fie înregistrat în aplicație.

Post condiții: Utilizatorul simplu are o nouă parolă cu care se poate autentifica.

Principalul scenariu de succes (Flux de Bază):
1. Utilizatorul înregistrat în aplicație accesează secțiunea de recuperare a parolei.
2. Utilizatorul furnizează sistemului adresa de email.
3. Sistemul validează adresa de email.
4. Sistemul trimite un email utilizatorului, conținând numele de utilizator inițial și parola temporară.
5. Utilizatorul se autentifică în sistem cu parola temporară primită.
6. Sistemul validează numele de utilizator și parola temporară furnizată.
7. Sistemul redirecționează utilizatorul către secțiunea de resetare a parolei.
8. Utilizatorul furnizează sistemului noua sa parolă.
9. Sistemul validează noua parolă și actualizează contul utilizatorului cu aceasta.
10. Sistemul redirecționează utilizatorul către secțiunea de autentificare și trimite un mesaj de succes.

Extensii (Fluxuri Alternative):
2a. Adresa de email nu este furnizată:
    1. Sistemul informează utilizatorul că adresa de email nu a fost completată.
3a. Adresa de email furnizată este invalidă:
    1. Sistemul informează utilizatorul că emailul introdus nu există în baza de date.
6a. Numele de utilizator sau parola temporară nu corespund cu baza de date:
    1. Utilizatorul este informat de incorectitudinea datelor furnizate.
9a. Noua parolă nu este validă:
    1. Procesul de resetare este oprit și utilizatorul este informat că parola furnizată nu este suficient de puternică sau că parola introdusă nu se potrivește cu parola de confirmare.

4.3.3. Efectuarea comenzii individuale

Această funcționalitate poate fi utilizată doar de către utilizatorii autentificați în aplicație. Pentru a efectua o comandă individuală, aceștia trebuie să selecteze restaurantul de la care doresc să comande, dintr-una din listele de restaurante.

Lista de restaurante poate fi filtrată de către utilizator în ordine crescătoare a distanțelor față de locația curentă, în ordine alfabetică, în funcție de orașul selectat sau în funcție de restaurantele favorite. Pentru ca un utilizator să poată vizualiza această listă, nu este necesar să fie autentificat în aplicație, însă pentru a putea efectua o comandă propriu-zisă, acesta trebuie mai întâi să se autentifice.

Pentru fiecare restaurant din listă vor fi disponibile informații specifice, cum ar fi: numele, adresa, metoda de livrare a comenzii pe care restaurantul o aplică. De asemenea, tot aici se găsesc și opțiunile de a suna direct la restaurant, de a localiza restaurantul pe hartă și de a accesa restaurantul respectiv în vederea plasării unei comenzi.

Următoarea descriere de caz de utilizare cuprinde pașii necesari efectuării unei comenzi individuale:

Actor principal: Utilizatorul simplu

Participanți și interese:

Utilizatorul simplu: dorește să ajungă în posesia produselor dorite de la un anumit restaurant, prin efectuarea unei comenzi individuale.

Angajatul restaurantului: dorește să vizualizeze cât mai repede comanda pentru a o putea onora în cel mai scurt timp.

Precondiții: Aplicația trebuie să fie instalată și pornită.
Utilizatorul simplu trebuie să fie autentificat în aplicație.

Post condiții: Comanda efectuată de către utilizator este vizualizată de către angajatul restaurantului.

Principalul scenariu de succes (Flux de Bază):

1. Utilizatorul alege restaurantul de la care dorește să comande sau dorește doar să vizualizeze produsele din lista de restaurante.
2. Utilizatorul alege una dintre categoriile de produse pe care restaurantul le oferă.
3. Utilizatorul vizualizează lista de produse din categoria aleasă și selectează unul dintre produsele oferite.
4. Sistemul prezintă informațiile detaliate despre produs: imagine, denumire, cantitate, ingredientele din care se compune produsul și prețul acestuia.
5. Utilizatorul selectează numărul de temi din acest produs și furnizează instrucțiunile speciale pe care angajații restaurantelor vor încerca să le urmeze, însă nu este obligatoriu să fie îndeplinite de către aceștia.
6. Utilizatorul adaugă cu succes respectivul produs în coșul de cumpărături.
7. Utilizatorul parcurge din nou pașii 2-6 în cazul în care dorește să selecteze altă categorie, pașii 3-6 dacă dorește să aleagă un produs din categoria actuală, pașii 4-5 dacă dorește să mai adauge itemi din același produs anterior adăugat. Această buclă este efectuată până când lista de produse conține tot ceea ce utilizatorul dorește să comande.

8. Utilizatorul accesează coșul de produse pentru a vizualiza din nou ceea ce a ales în pașii anteriori: produsele împreună cu numărul de itemi, prețul fiecărui produs și suma totală de plată.

9. Utilizatorul trece la următorul pas în care trebuie să aleagă unul dintre tipurile de livrare pe care restaurantul le pune la dispoziție (livrare la domiciliu, ridicare de la restaurant și comanda la masă). Nu este obligatoriu ca restaurantul să ofere toate aceste trei tipuri, însă trebuie să pună la dispoziție cel puțin unul.

10. Utilizatorul trebuie să furnizeze adresa completă la care să îi fie livrată comanda. Acest pas este efectuat doar dacă la pasul 9 a fost ales tipul de livrare la domiciliu.

11. Sistemul adaugă comanda în baza de date și furnizează angajaților restaurantului ales toate datele privitoare la comandă. Toate produsele din coș sunt eliminate.

12. Utilizatorul primește un mesaj cu detaliile despre produsele comandate și cu suma totală de plată.

Extensii (Fluxuri Alternative):

1a. Utilizatorul dorește să contacteze telefonic restaurantul selectat:

1. Utilizatorul apelează restaurantul dorit.

1b. Utilizatorul dorește să vizualizeze locația restaurantului.

1. Sistemul localizează restaurantul pe hartă.

2-10a. Utilizatorul dorește să aleagă alt restaurant de la care să comande:

1. Utilizatorul se întoarce la lista de restaurante.
2. Toate produsele existente în coș sunt eliminate.

2-10b. Utilizatorul dorește să modifice coșul de produse:

1. Utilizatorul accesează coșul de produse, unde are posibilitatea să modifice numărul itemilor pentru un anumit produs sau să îl șteargă.

4.3.4. Efectuarea comenzii de grup

Comanda de grup este cea mai importantă funcționalitate a sistemului. Este cea care face diferența între aplicațiile Android de acest tip deja existente și aplicația AndroChef. Această funcționalitate vine în ajutorul persoanelor care se află în aceeași locație și doresc să efectueze o comandă unică pe grup, către un anumit restaurant.

Fiecare utilizator participant la grup, efectuează o comandă individuală în cadrul grupului. Toate comenzile individuale sunt grupate de către sistem într-o singură comandă care este mai apoi furnizată către restaurantul de la care fiecare participant la grup și-a ales produsele.

Angajații restaurantului nu vor ști dacă au primit o comandă simplă sau o comandă de grup, deoarece vor primi comanda de grup în același format ca și o comandă simplă. Aceștia sunt interesați doar de produsele comandate împreună cu numărul lor, cu instrucțiunile speciale și adresa de livrare.

În momentul în care comanda de grup este onorată de către unul dintre angajații restaurantului, administratorul de grup este cel care se ocupă de plata sumei de bani aferente și de ridicarea produselor. Administratorul de grup trebuie să cunoască suma

finală de plată înainte de livrarea efectivă a comenzii. De asemenea, trebuie să știe suma de bani pe care fiecare membru al grupului trebuie să o achite.

Fiecare membru al grupului va primi un mesaj cu detalii legate de produsele comandate și cu suma totală de plată pentru comanda individuală din cadrul grupului, în momentul în care comanda individuală este finalizată și sistemul o înregistrează în baza de date.

Administratorul grupului va primi un mesaj la finalizarea comenzii de grup, care va conține suma de bani pe care fiecare membru al grupului trebuie să o plătească pentru comanda sa și suma totală pe care administratorul trebuie să o înmâneze angajatului restaurantului.

Actor principal: Administratorul de grup

Participanți și interese:

Administratorul de grup: dorește să inițieze o comandă de grup pentru a efectua o comandă unică pe întregul grup.

Membrul grupului: dorește să ia parte la comanda de grup pentru a efectua o comandă individuală în cadrul lui. Dacă ar efectua o comandă simplă și nu ar depăși o anumită sumă minimă, ar trebui să plătească și transportul.

Angajatul restaurantului: dorește să vizualizeze cât mai repede comanda de grup într-un format cât mai accesibil pentru a o putea onora în cel mai scurt timp.

Precondiții:

Aplicația trebuie să fie instalată și pornită pe dispozitivul *administratorului de grup*. Acesta trebuie să fie autentificat și să selecteze unul dintre restaurantele din lista pentru a putea inița comanda de grup.

Dispozitivele *membrilor grupului* trebuie să aibă instalată aplicația pentru a o putea porni în momentul acceptării invitației de participare la comandă.

Post condiții: Comanda de grup finalizată de către administratorul grupului este vizualizată de către angajatul restaurantului.

Principalul scenariu de succes (Flux de Bază):

1. Administratorul de grup invită persoane la grup prin trimiterea unui mesaj care conține numele restaurantului și un link prin care cel invitat poate porni aplicația.
2. Invitații la grup acceptă invitația trimisă de către administrator și devin membrii ai grupului.
3. Membrii grupului sunt direcționați către categoriile restaurantului selectat de către administrator.
4. Administratorul grupului și fiecare membru efectuează pașii 2-8 din fluxul principal al cazului de utilizare 4.3.3 Efectuarea comenzii individuale.
5. Membrii grupului finalizează comanda lor individuală în cadrul acestuia. Membrii nu mai trebuie să parcurgă pasul de selectare a tipului de livrare și de completare a adresei.
6. Administratorul de grup observă că toți membrii și-au finalizat propria comandă și decide și finalizarea comenzii de grup. Administratorul și fiecare membru

poate vizualiza componența grupului și poate observa membrii care au finalizat comanda individuală și cei care nu au finalizat-o.

7. Administratorul efectuează pașii 9-10 din fluxul principal al cazului de utilizare 4.3.3 Efectuarea comenzii individuale.

8. Sistemul adaugă comanda în baza de date și furnizează angajaților restaurantului ales toate datele privitoare la comandă. Toate produsele din coșul administratorului și al membrilor grupului sunt eliminate.

9. Administratorul grupului primește un mesaj cu suma pe care fiecare membru al grupului trebuie să o plătească, iar membrii grupului primesc un mesaj la fel cu cel în cazul comenzii individuale.

10. Administratorul și membrii grupului devin utilizatori simpli, având posibilitatea de a efectua oricând o nouă comandă.

Extensii (Fluxuri Alternative):

2a. Invitația este acceptată doar de către o parte dintre cei invitați:

1. Grupul va fi format doar din cei care au acceptat invitația.

2b. Nici un invitat nu participă la comanda de grup:

1. Grupul rămâne format doar din administrator.

3a. Cei care acceptă invitația nu sunt autentificați în aplicație:

1. Vor fi direcționați la secțiunea de autentificare și se vor autentifica.
2. După autentificare, vor putea alege din lista de categorii ale restaurantului ales de către administrator.

4a. Cel puțin unul dintre membrii dorește să părăsească grupul:

1. Membrul grupului selectează opțiunea de părăsire a acestuia pentru a efectua o comandă individuală sau pentru a amâna comanda pentru altă dată.
2. Membrul grupului devine utilizator simplu și este direcționat către lista de restaurante.

5-6a. Cel puțin unul dintre membrii nu își finalizează propria comandă:

1. Administratorul grupului poate să finalizeze comanda de grup chiar dacă nu și-a finalizat fiecare membru comanda individuală.
2. Cei care nu și-au finalizat propria comandă nu vor primi mesaj la finalizarea comenzii de grup.

4.4. Tehnologii utilizate

Acest proiect are ca și scop construirea unei aplicații Client-Server pe Android, care să eficientizeze efectuarea de comenzi de grup către restaurante. Tehnologiile alese pentru partea de Server sunt Dancer și MySQL, iar pe partea de Client este Android.

Aplicația va fi construită pentru dispozitive mobile, deoarece acestea au marele avantaj de a fi la îndemâna posesorilor oriunde s-ar deplasa. Din acest motiv, desktop-urile sau laptop-urile sunt înlocuite încetul cu încetul de către dispozitivele mobile, atunci când vine vorba de a naviga pe internet sau de a folosi aplicații.

4.4.1. Tehnologii pe parte de Server

Aplicația Server va fi dezvoltată în limbajul Perl, având la bază framework-ul Dancer. Baza de date va fi gestionată prin intermediul SGBD-ul MySQL. Accesul la baza de date de pe Server se va realiza prin intermediul ORM-ului DBIX.

Mediul de dezvoltare utilizat pentru implementarea părții de Server este Sublime.

4.4.1.1. Limbajul Perl

Limbajul Perl este un limbaj de programare de nivel înalt, interpretat, inițial dezvoltat de Larry Wall în 1987. Limbajul Perl a fost dezvoltat pentru prelucrarea de texte și generarea de rapoarte, de aici acronimul: Practical Extraction and Report Language. Limbajul împrumută caracteristici ale limbajelor C, Pascal, AWK, sed, sh și chiar BASIC. Perl este un limbaj portabil, rulând în peste 100 de platforme de la sisteme portabile la sisteme distribuite.

Perl a fost folosit de la începuturile Web-ului în programarea dinamică a site-urilor, utilizând protocolul CGI. Pe lângă Python și PHP, Perl este unul dintre cele mai populare limbaje folosite în dezvoltarea site-urilor dinamice. Site-uri populare ca imdb.com, bbc.co.uk, amazon.com, booking.com sau dslreports.com au fost realizate cu ajutorul perl-ului sau mod perl-ului.

Pe sistemele UNIX limbajul Perl este de obicei preinstalat deoarece numeroase componente Linux/Unix sunt implementate folosind Perl. Pentru sistemele Microsoft Windows, interpretorul Perl trebuie descărcat și instalat.

Informațiile de mai sus referitoare la limbajul Perl au fost preluate din lucrarea [12]. Tot aici se precizează că Perl este un acronim, însă din cartea referită la [13] se deduce faptul că Perl este de fapt un backronym, adică inversul acronimului. Perl a mai fost numit și “Pathologically Eclectic Rubbish Lister” pe lângă denumirea precizată mai sus. Larry Wall a venit cu ideea numelui mai întâi și apoi cu semnificația acestuia.

Cartea [13] explică și avantajele limbajului Perl față de alte limbaje de programare. La momentul creării limbajului, Larry a avut nevoie de un limbaj rapid asemenea shell-ului sau limbajului *awk*, care să aibă și puterea unor tool-uri mai avansate, cum ar fi grep, cut, sort și sed fără a fi nevoit să recurgă la un limbaj precum C.

Perl încearcă să umple golul dintre programarea de nivel jos (cum ar fi C, C++ sau asamblare) și cea de nivel înalt (cum ar fi programare shell). Programele de nivel jos sunt de obicei mai greu de scris și mai neplăcut, însă sunt rapide și nelimitate. Este foarte greu de depășit viteza unui program bine scris de nivel jos pe o anumită mașină. Pe de altă parte, programele de nivel ridicat tind să fie încete, neplăcute și limitate. Sunt multe lucruri care nu se pot face cu programele shell sau batch.

Perl adună avantajele limbajelor mai sus menționate, rezultând într-un limbaj ușor, aproape nelimitat și de cele mai multe ori rapid. Însă Perl păstrează, din păcate și unul dintre dezavantajele celorlalte limbaje, și anume faptul că poate să aibă un aspect neplăcut. În cele ce urmează sunt explicate mai în detaliu cele patru caracteristici ale limbajului:

- ✓ Perl este un **limbaj ușor**. Aceasta înseamnă că este ușor de utilizat. Nu înseamnă neapărat că este și ușor de învățat. De exemplu, pentru a conduce o mașină se

petrec săptămâni sau chiar luni pentru a învăța procedurile, iar apoi va fi ușor de condus. Pentru a învăța Perl durează tot atât de mult cât durează învățatul condusului, iar apoi va fi foarte ușor de folosit.

- ✓ Perl este **aproape nelimitat**. Sunt foarte puține lucruri care nu se pot realiza în Perl. Se pot realiza de la cele mai mici și rapide programe la cele mai mari și puternice aplicații.
- ✓ Perl este **rapid** de cele mai multe ori. Dacă cineva dorește să adauge un feature în limbaj care ar fi util, dar ar încetini alte programe, Larry cu siguranța ar refuza acel feature până s-ar găsi o soluție mai rapidă.
- ✓ Perl este destul de **neplăcut**. Simbolul Perl-ului a devenit cămila de pe venerabila copertă a cărții *Programming Perl*. Cămilele nu sunt frumoase, însă muncesc din greu chiar și în condiții extreme. Cămilele sunt pregătite să își facă treaba în ciuda dificultăților existente, fără să conteze că nu arată cel mai bine sau chiar au un miros neplăcut. Perl-ul se poate asemăna cu o cămilă.

Unele porțiuni de cod în Perl pot să fie scrise doar în 3 linii, în timp ce în alte limbaje ar trebui scrise 20 de linii pentru a realiza aceeași funcționalitate. Această caracteristică salvează din timpul programatorilor. Totodată, codul este mai ușor de întreținut și de depanat prin folosirea mai redusa a variabile.

Limbajul utilizat la implementarea Server-ului va fi Perl, datorită caracteristicilor pozitive mai sus menționate.

4.4.1.2. Framework-ul Dancer

Framework-ul care va sta la baza aplicației Server va fi Dancer. Dancer este un framework web simplu și de dimensiune mică, dar puternic. Acesta este inspirat dintr-un framework scris în Ruby și care se numește Sinatra. Dancer permite crearea rapidă a site-urilor web, spre deosebire de framework-ul Catalyst care este foarte complex și mai dificil de utilizat. Dancer oferă suport pentru crearea de aplicații mici și mijlocii, în comparație cu Catalyst care este în special utilizat pentru aplicații de dimensiuni mari.

Următoarele sunt caracteristicile cheie alea framework-ului Dancer, conform site-ului [14]:

- ✓ **Simplitatea** sa constă în faptul că este intuitiv, minimalist și are o sintaxă foarte expresivă.
- ✓ Este **flexibil**. Suportul PSGI, plugin-urile și design-ul modular permit o puternică scalabilitate.
- ✓ **Puține dependențe**. Dancer depinde de cât mai puține module de pe CPAN, făcând astfel ușoară instalarea sa.

Construirea aplicațiilor în Dancer pornește de la definirea verbelor HTTP, URL-urilor (denumite și rute) și a metodelor care să trateze cererile spre rute.

Noțiunea de rută, conform site-ului [15], specifică una dintre căile pe care utilizatorul o poate accesa în aplicație. Rutele suportă variabile, expresii regulate și chiar și potrivire condițională.

De exemplu:

```
get '/' => sub {
        return 'hello world!';
 };
```

Ruta de mai sus reprezintă calea de bază a aplicației. Cuvântul get specifică metoda HTTP (GET) pentru care calea există. Pe lângă get se mai pot folosi și cuvintele post, del sau put, care reprezintă celelalte metode HTTP. De asemenea, se poate utiliza și cuvântul any pentru a furniza o singură rută pentru toate metodele HTTP sau doar pentru o parte. Ceea ce urmează după '=>' reprezintă metoda care tratează accesul la această rută.

Pentru definirea rutelor se pot folosi și prefixe cu scopul de a evita repetarea aceluiași cuvânt. Sistemul de rutare este implementat pentru a suporta scalabilitatea, printr-un mecanism de caching. Astfel, chiar dacă numărul rutelor este crescut, viteza este la fel de mică ca și cum ar fi un număr scăzut de rute.

Exemplul următor reprezintă o aplicație web funcțională, demonstrând cât de ușor este să creezi aplicații web cu ajutorul lui Dancer:

```
#!/usr/bin/perl
use Dancer;

get '/hello/:name' => sub {
        return "Why, hello there " . param('name');
};

dance;
```

Pentru dezvoltarea aplicației Server a fost ales Dancer, deoarece se pretează aplicațiilor mici, așa cum este AndroChef, fiind ușor de învățat și simplu de utilizat. Deși Catalyst este un framework bogat în feature-uri, flexibil și matur, având o mulțime de documente și tutoriale disponibile, nu a fost ales pentru implementare, deoarece este un framework complex, greu de învățat și de utilizat.

4.4.1.3. Accesul la baza de date: DBIx

Serverul este cel responsabil de accesul la baza de date. Accesul se efectuează prin intermediul unui ORM. ORM (Object-relational mapping) este o tehnică de programare pentru convertirea datelor între tipuri de sisteme incompatibile. Aceasta creează o bază de date virtuală care reprezintă interfața dintre aplicație și baza de date. Figura 4.4 ilustrează interacțiunea cu baza de date prin intermediul ORM-ului.

DBIx::Class este ORM-ul folosit pentru accesul la baza de date. Acesta mapează rândurile din baza de date relațională pe obiecte Perl și generează interogări SQL în mod transparent. DBIx implementează pattern-ul Active Record.

DBI este un alt modul Perl, reprezentând un layer de abstractizare la baza de date. Acesta definește un set de metode, variabile și convenții care oferă o interfață consistentă

la baza de date, independentă de tipul bazei de date folosită. DBI este doar o interfață de acces, spre deosebire de DBIx care este o mapare.

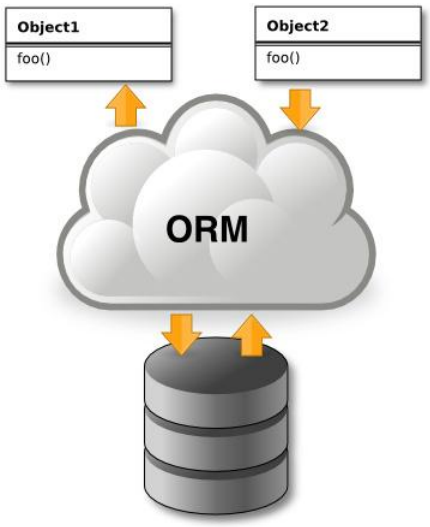

Figura 4.4 Baza de date – ORM - Aplicație

ORM-ul DBIx a fost ales ca și metodă de acces la partea persistentă a sistemului, în detrimentul tehnicilor tradiționale de acces la baza de date precum DBI, deoarece primul reduce considerabil cantitatea de cod care trebuie scrisă. Oferă o vizualizare a bazei de date ca și o listă de obiecte în loc de rânduri și coloane.

Developerii nu mai trebuie să scrie manual interogări. Nivelul de abstractizare introdus între aplicație și baza de date permite schimbarea tipului de bază de date fără să fie nevoie de efectuarea de modificări în aplicație (de exemplu din mysql spre postgres sau oracle).

DBIx nu oferă doar o simplă bază de date către layer-ul de obiecte mapate, ci poate implementa diferite alte funcții de baze de date de nivel mai înalt (integritate referențială, ștergere în cascadă etc.), la nivelul aplicației și nu la nivelul bazei de date.

Informațiile referitoare la DBIx au fost extrase din sursa [16].

4.4.2. Comunicarea între Client și Server

Protocolul prin care se realizează comunicarea dintre Client și Server este protocolul HTTP. În cele ce urmează, vor fi prezentate aspectele importante ale acestui protocol, pe baza sursei [17].

HTTP (Hypertext Transfer Protocol) este metoda cea mai des utilizată pentru accesarea informațiilor pe Internet care sunt păstrate pe servere World Wide Web (WWW). Protocolul HTTP este un protocol de tip text, fiind protocolul "implicit" al WWW. Este un așa-zis limbaj între cele două aplicații care comunică, prin care se pot transfera documente HTML, fișiere grafice, fișiere de sunet, animații, video sau chiar și programe executabile. Modelul de referință OSI clasifică protocolul HTTP ca fiind unul la nivel de aplicație.

În prezent se utilizează două versiuni ale protocolului, HTTP/1.0 și HTTP/1.1. La versiunea HTTP/1.0 se stabilește o nouă conexiune TCP înaintea cererii, iar după transmiterea răspunsului conexiunea trebuie închisă. Astfel dacă un document HTML cuprinde 10 imagini, vor fi necesare 11 conexiuni TCP, pentru ca pagina să fie afișată complet (în browser). La versiunea 1.1 se pot emite mai multe cereri și răspunsuri pe aceeași conexiune TCP. Astfel pentru documentul HTML cu 10 imagini este necesară doar o singură conexiune TCP.

Metodele de bază pentru realizarea comunicării sunt următoarele:

- ✓ **GET**: este cea mai folosită metodă, fiind utilizată atunci când serverului i se cere o resursă.
- ✓ **PUT**: metoda este folosită pentru a depune documente pe server, fiind inversul metodei GET.
- ✓ **POST**: a fost proiectată pentru a trimite date de intrare către server.
- ✓ **DELETE**: se ocupă de ștergerea unei resurse.

Aplicația Server va fi construită pe modelul arhitectural **REST**. REST (Representational State Transfer) este un stil arhitectural software constând dintr-un set de constrângeri arhitecturale aplicate componentelor, conectorilor și elementelor de date, într-un sistem distribuit, conform [18]. REST ignoră detaliile legate de implementarea componentelor și de sintaxa protocolului pentru a se concentra pe rolul componentelor, pe constrângerile dintre interacțiunile cu alte componente și pe interpretarea elementelor de date.

O definiție informală precizează că REST este un mod de a crea, citi, actualiza sau șterge informații pe un server prin intermediul apelurilor HTTP. Este o alternativă la mecanismele mai complexe, cum ar fi SOAP, CORBA sau RPC. Un apel REST este un simplă cerere HTTP către server.

SOAP (Simple Object Access Protocol) este o alternativă la modelul REST. Este un protocol pentru schimbul de informații structurate prin intermediul XML, în implementarea web serviciile.

O analiză comparativă a celor 2 arhitecturi este prezentată în articolul [19], unde se precizează faptul că SOAP nu este cea mai bună soluție la dezvoltarea aplicațiilor mobile, fiind identificate o serie de neajunsuri:

- ➢ **Controlul schimbării** – Modificarea serviciilor care folosesc SOAP de cele mai multe ori implică modificări complicate pe partea de client. Când clientul este o aplicație web, aceste modificări nu sunt neapărat o problemă, dar dacă clientul SOAP rulează pe un dispozitiv mobil, devine o problemă importantă.
- ➢ **Complexitatea** – Generarea de cod client SOAP din WSDL și XSD poate deveni destul de complexă, iar această problemă este amplificată de faptul că multe organizații trebuie să producă aceeași aplicație mobilă pentru mai multe platforme (iOS, Android, etc.).
- ➢ **Reutilizarea** – Majoritatea aplicațiile moderne care folosesc Ajax au adoptat serviciile RESTful cu formatul JSON ca fiind cea mai bună soluție de a accesa servicii web la distanță. Deci reutilizarea devine o limitare dacă aplicațiile mobile se folosesc de SOAP și nu de REST.

De asemenea, în articolul [19] sunt descrise beneficiile pe care arhitectura REST le oferă, și anume:

- **Simplitate –** REST a fost proiectat să opereze cu clienți de dimensiuni mai mici, cum ar fi browser-ul web și cu toate tipurile de servicii (Java, .NET, PHP, Perl). RESTful se caracterizează prin simplitate și flexibilitate în implementare.
- **Infrastructură adaptată** – load balancer-ul de rețea, firewall-uile și proxi-urile sunt toate optimizate pentru traficul REST, deoarece ele au fost deja optimizate pentru traficul HTTP/S.
- **Scalabilitate** – Cererile de tip REST nu rețin starea, ceea ce face ca aceste cereri să fie foarte scalabile.
- **Stateless sau Stateful –** Deși prin definiție apelurile REST se referă la apeluri care nu păstrează starea, se pot utiliza și stări prin intermediul sesiunii. Sesiunile oferă o securizare a comunicării.
- **Eficiență** – Serviciile SOAP returnează întotdeauna XML, în timp ce REST oferă flexibilitatea de alege din mai multe structuri de date. Standardul "de facto" al datelor transmise prin REST este JSON. Inițial, Android și iOS nu conțineau parsere de JSON ca parte sistemului, însă mai târziu aceste funcționalități au fost adăugate. Structurile JSON sunt de obicei mai mici decât cele XML, iar daca se mai adaugă și învelitoarea SOAP peste structura XMP, REST împreună cu JSON vor fi mult mai mici decât competitoarele lor.

Pentru implementarea comunicării au fost alese serviciile REST, datorită simplității, performanței și flexibilității pe care le oferă, în comparație cu SOAP.

4.4.3. Tehnologii pe parte de Client

Clientul este reprezentat de o aplicație care va rula pe platforma Android și va implementa pattern-ul MVC (Model-View-Controller).

4.4.3.1. Platforma Android

Android este o platformă software și un sistem de operare pentru dispozitive și telefoane mobile bazată pe nucleul Linux, dezvoltată mai întâi de către compania Google, iar mai târziu de consorțiul comercial Open Hanset Alliance.

Android permite dezvoltatorilor să scrie cod în limbajul Java, având la dispoziție bibliotecile Java dezvoltate de către Google. Aplicațiile pot fi scrise și în alte limbaje, pot fi compilate în cod mașină ARM și executate, însă acest model de dezvoltare nu este sprijinit oficial de către cei de la Google.

Android este o platformă open source, acesta fiind cel mai important factor care îl diferențiază de alte sisteme de operare pentru dispozitive mobile. Aceasta înseamnă că dacă un developer dorește să mai adauge o nouă capabilitate, poate să o construiască și să o încorporeze în OS.

Figura 4.5 , preluată din sursa [20], ilustrează arhitectura platformei Android.

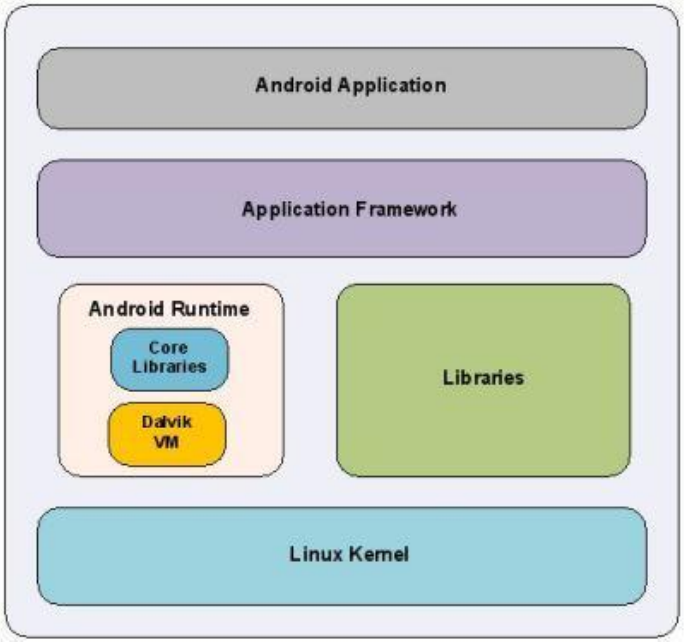

Figura 4.5 Arhitectura Android

Arhitectura platformei Android este structurată sub forma unei stive. La baza ei se află nucleul Linux. Nucleul este principala componentă a unui sistem de operare, care creează o punte între partea software și hardware, oferind cel mai de jos nivel de abstractizare. Nucleul include programe de gestionare a memoriei, de gestionare a energiei și drivere pentru a controla componentele hardware, cum ar fi camera, tastatura, componenta audio, etc.

Deasupra nucleului se găsesc librăriile Android, care dau instrucțiuni nucleului pentru a executa un task specific. De exemplu, librăria care conține framework-ul media suportă redarea și înregistrarea formatelor audio, video sau a imaginilor. Pe acelaşi nivel se află și Android Runtime. Acesta conține un set de librării Java de bază care permit dezvoltatorilor Android să-și creeze propriile aplicații, folosind limbajul de programare Java.

Mașina virtuală Dalvik se află în categoria Android Runtime. O mașină virtuală este o aplicație software care poate executa programe ca și mașinile fizice, comportându-se ca și un dispozitiv independent împreună cu propriul sistem de operare. Sistemul Android folosește mașini virtuale pentru a rula fiecare aplicație în propriul lor proces. Aceasta face ca aplicațiile să fie independente, iar dacă o aplicație cade, celelalte aplicații vor putea rula în continuare.

Deasupra librăriilor se află layer-ul Application Framework. Acesta include programe care controlează funcțiile de bază ale telefonului, cum ar fi alocarea de resurse, gestionarea locației, gestionarea notificațiilor, etc. Fiind o platformă de dezvoltare deschisă, dezvoltatorii au acces complet la framework-ul aplicație, permițând crearea unor aplicații ample.

În vârful stivei se află chiar aplicațiile care interacționează cu nivelele de mai jos pentru efectua anumite operații, ca de exemplu, efectuarea unui apel telefonic. Întreaga stiva se află în memoria ROM (Read Only Memory).

Detaliile legate de arhitectura platformei Android au fost sintetizate din articolul [20].

Figura 4.6, preluată din articolul [21], prezintă toate versiunile Android apărute de-a lungul timpului. Acestea poartă denumiri inspirate de produsele de cofetărie, venind în sprijinul utilizatorilor. Aceștia pot identifica mai ușor versiunile și aplicațiile compatibile. Cel mai recent, Nestle a încheiat un acord cu Google astfel că ultima versiune de Android se numește KitKat.

Version	Codename	API	Distribution
2.2	Froyo	8	1.3%
2.3.3 - 2.3.7	Gingerbread	10	20.0%
3.2	Honeycomb	13	0.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	16.1%
4.1.x	Jelly Bean	16	35.5%
4.2.x		17	16.3%
4.3		18	8.9%
4.4	KitKat	19	1.8%

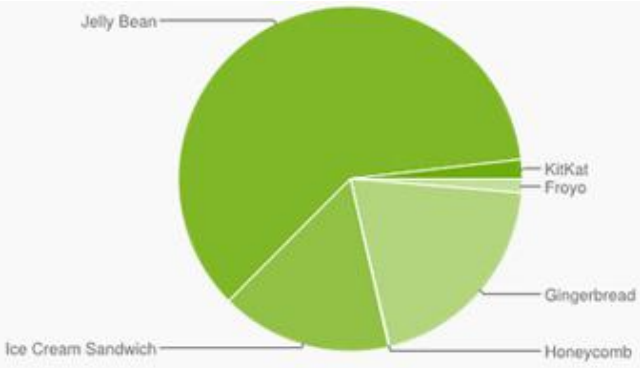

Figura 4.6 Versiunile platformei Android

4.4.3.2. Android Passive MVC

În zilele noastre cererea pentru dezvoltarea aplicațiile mobile este foarte mare. Pentru a fi competitivă, o aplicație mobilă trebuie să fie redusă ca și cost și de calitate bună, iar alegerea arhitecturii este importantă pentru a asigura calitatea aplicației în timp și pentru a reduce timpul de dezvoltare.

În timp ce arhitectura iOS se bazează pe design pattern-ul Model-View-Controller, sistemul Android nu specifică un model de design. Acest lucru poate încetini dezvoltarea, întreținerea, extinderea și performanța aplicației. Lucrarea [22] propune un model arhitectural unic pe care să se dezvolte aplicațiile Android, deoarece puține lucrări se concentrează pe acest aspect. Lucrarea prezintă arhitectura MVC adaptată la sistemul Android.

Arhitectura MVC pentru Android a fost denumită Android Passive MVC. Activitatea, specifică Android, s-a dovedit a nu se potrivi în arhitectura MVC, deși pare să se adapteze nevoilor dezvoltatorilor. De aceea s-a decis crearea unui model MVC care să facă din activitate a patra componentă a arhitecturii. Activitatea devine o componentă intermediară între View și Controller, Controller-ul preluând responsabilitatea de a trata evenimente, iar View-ul de a se ocupa de logica de prezentare, astfel Activitatea rămânând coezivă.

Figura 4.7, preluată din lucrarea [22], ilustrează schema arhitecturii Android MVC. Săgețile gri punctate arată interacțiunea prin metodele native Android. Săgețile negre indică apeluri directe, iar săgețile gri reprezintă evenimente pe listener.

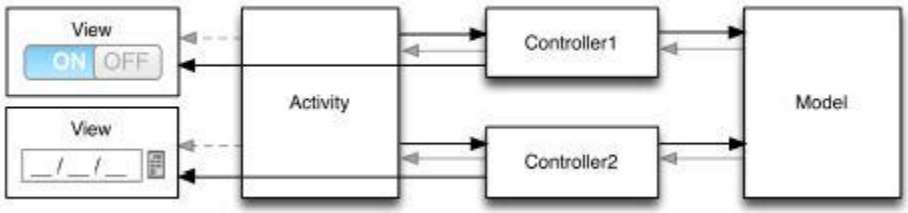

Figura 4.7 Android Passive MVC

Activitatea este ca și un controler de ecran. Activitatea pornită creează o legătură între View și Controller pentru a le permite să comunice direct. Comunicarea între Controllere se face prin intermediul Activității.

View-urile sunt componentele de interfață, cum ar fi formulare, meniuri sau liste de elemente. Componentele View conțin metode care permit setarea sau obținerea de informații de la interfața utilizatorului la cererea Controller-ului, setarea ascultătorilor de evenimente pe componentele vizuale și modificarea componentelor vizuale (setarea erorilor, schimbarea culorii, etc.). View-urile sunt independente și nu comunică între ele.

Controller-ul tratează evenimentele declanșate de utilizator (apăsarea unui buton), apelează metodele necesare din Model și apoi notifică View-ul pentru a fi actualizat conform răspunsului. Controllerele sunt independente unul de celălalt și nu comunică direct.

Această soluție face ca Activitatea să nu aibă prea multe responsabilități, transferând tratarea de evenimente Controller-ului și gestionarea interfeței View-ului. Aplicația poate fi ușor modificată doar prin ștergerea sau modificarea perechilor View-Controller. De asemenea, poate fi și ușor extinsă cu alte perechi View-Controller. Modelul este independent de View, de Controller și de Activitate. Deci interfața utilizator poate fi înlocuită fără a avea vreun impact asupra Modelului, crescând astfel mentenabilitatea sistemului.

Există soluții și pentru dezvoltarea de aplicații pe mai multe platforme, adică o aplicație dezvoltată poate rula și pe Android, dar și pe iOS. Printre acestea se numără Phone-Gap, Rhodes Rhomobile și Titanium Appcelerator. Utilizarea lor reduce timpul de dezvoltare, însă au posibilități limitate, având nevoie de multe ori de plugin-uri native. Soluțiile multi-platformă adaugă și complexitate codului nativ, care scade performanța aplicației. În lucrarea [22] se precizează că suportul pentru soluții non-native ar putea fi abandonat, soluțiile native având la dispoziție toate opțiunile platformei cu o mai bună performanță și un cod mai ușor. De aceea dezvoltatorii aleg de cele mai multe ori kit-ul de dezvoltare nativ (SDK).

Datorită celor mai sus menționate, aplicația AndroChef va fi dezvoltată în limbajul nativ al Android - Java, având la dispoziție tool-urile Android SDK. Arhitectura sistemului va fi cea propusă în lucrarea [22], și anume, Android Passive MVC.

Capitolul 5. Proiectare de Detaliu si Implementare

Acest capitol cuprinde schema generală a aplicației și descrierea componentelor implementate la nivel de modul. De asemenea, conține și diagrama de clase împreună cu o descrierea a claselor și a metodelor mai importante.

5.1. Structura generală a sistemului

Sistemul AndroChef este o aplicație Client-Server, în care partea de Server a fost implementată în limbajul Perl, având la bază framework-ul Dancer, iar partea de Client a fost implementată în Java, cu ajutorul framework-ului Android. Persistența sistemului este asigurată prin intermediul SGBD-ului MySQL.

Figura 5.1 prezintă structura de nivel înalt a sistemului, cu principalele tehnologii utilizate.

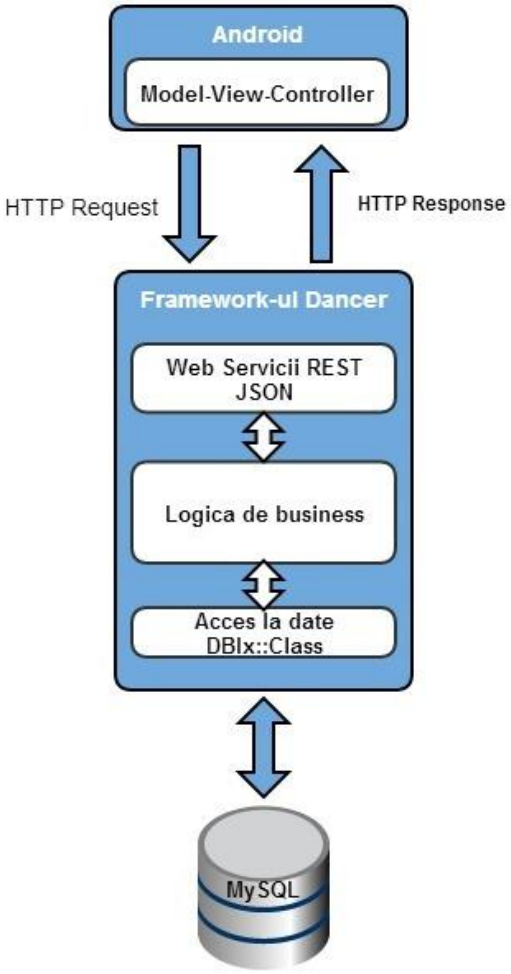

Figura 5.1 Schema generală a sistemului

Aplicația AndroChef este structurată pe trei componente, și anume aplicația Client, aplicația Server și baza de date.

Aplicația Server este cea care furnizează servicii aplicației Client prin intermediul protocolului HTTP. Datele care se trimit între Client și Server sunt în format JSON. Aplicația Server are acces la baza de date MySQL cu ajutorul ORM-ului DBIx. Fiecare tabel din baza de date este mapat pe un modul care moștenește modului Perl DBIx::Class.

Aplicația Client poate rula doar pe dispozitivele Android. Acesta este implementată pe modelul de design Model-View-Controller. View-urile sunt reprezentate de fișierele XML, care pot fi meniuri, liste de elemente sau alte componente de interfață. Controller-ele reprezintă Activitățile, care generează View-urile corespunzătoare și se ocupă de tratarea evenimentelor. Modelele implementează logica aplicației, fiind independente de View și de Controller.

5.2. Diagrama de deployment

Diagrama de deployment descrie componentele hardware (nodurile) pe care rulează componentele software, denumite și artefacte ale sistemului și arată modul în care aceste componente sunt conectate. Scopul acestei diagrame este de arăta structura hardware a sistemului.

Figura 5.2 prezintă diagrama de deployment a aplicației AndroChef.

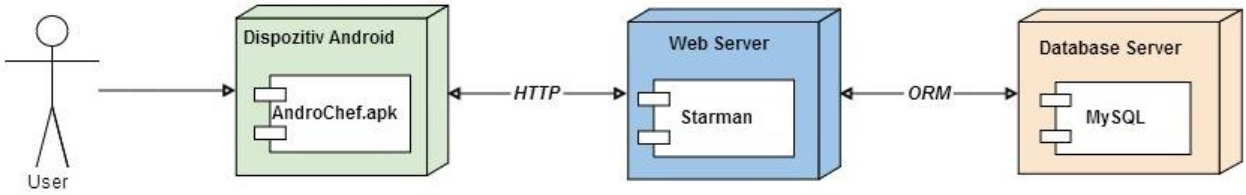

Figura 5.2 Diagrama de deployment

Utilizatorul este cel care beneficiază de pe urma întregului sistem. El utilizează dispozitivul mobil Android pe care este instalată aplicația AndroChef. Aplicația Android comunică cu web serverul Starman pe care este lansată aplicația care furnizează web servicii Clientului. Aplicația Server comunică la rândul ei cu serverul de baze de date MySQL prin intermediul ORM-ului.

5.3. Aplicația Server – Detalii de Implementare

Acest subcapitol cuprinde diagrama de module a aplicației Server împreună cu descrierea fiecărui modul în parte. Vor fi prezentate pe scurt și subrutinele modulelor.

5.3.1. Diagrama de module

În limbajul Perl nu există conceptul de clasă, deoarece nu este un limbaj Orientat-Obiect. În locul claselor există însă modulele care nu conțin atribute, ci doar subrutine. În continuare va fi ilustrată diagrama de module a aplicației Server.

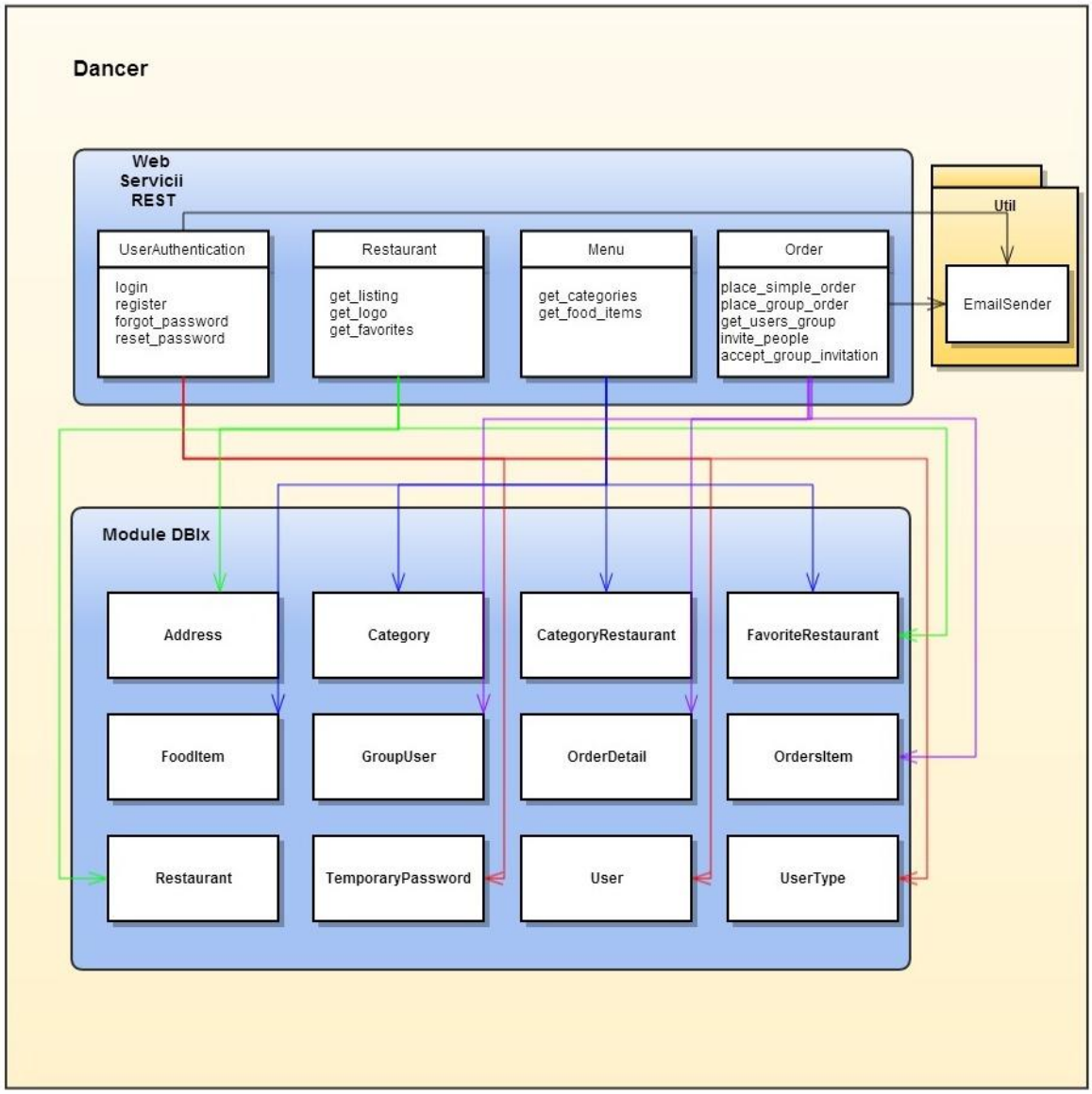

Figura 5.3 Diagrama de module pentru aplicația Server

Diagrama de module a aplicației Server se structurează pe 3 componente mari:

- Componenta de Web Servicii care se ocupă de furnizarea serviciilor către Client și de logica de business. Aceasta comunică cu Clientul prin protocolul HTTP și implementează arhitectura REST.
- Componenta Util care conține un singur modul EmailSender care se ocupă de trimiterea de email-uri.
- Componenta de module DBIx reprezintă maparea tabelelor și relațiilor din baza de date cu scopul de a oferi o interfață de acces la aceasta.

5.3.2. Componenta de servicii web

Această componentă reprezintă serviciile Web implementate pe arhitectura REST și se compune din patru module: UserAuthentication, Restaurant, Menu, Order.

5.3.2.1. Modulul UserAuthentication

Acest modul furnizează servicii pentru înregistrarea, autentificarea și recuperarea parolei utilizatorului și implementează patru subrutine:

- login
- register
- forgot_password
- reset_password

a) Login

Subrutina *login* are rolul de a autentifica utilizatorul care dorește să efectueze comenzi prin intermediul aplicației. Aceasta așteaptă să primească doi parametrii, numele de utilizator și parola. În cazul în care unul dintre parametrii nu este furnizat, se va genera un mesaj de eroare de forma “Username and password are required fields.”. Pentru verificarea furnizării ambilor parametrii se folosește funcția *trim* care face parte din librăria String::Util. Această funcție elimină spațiile de la începutul și de la sfârșitul unui cuvânt și apoi verifică dacă șirul de caractere trimis este vid sau nu.

Dacă ambii parametrii conțin caractere, atunci se trece la pasul de criptare a parolei în format *md5*. MD5 (Message Digest Algorithm 5) este o funcție criptografică de tip hash unidirecțional, care livrează ca rezultat o valoare fixă ca lungime de 128 de biți. Această funcție se asigură de faptul că parola nu va fi vizibilă în baza de date de către persoane neautorizate.

Următorul pas este de verificare în baza de date a existenței respectivului utilizator. Dacă acesta există, se returnează un mesaj de succes către Client împreună cu datele utilizatorului.

Dacă doar numele de utilizator se găsește în tabela User, atunci se verifică dacă parola trimisă este una temporară, primită prin email la cererea de recuperare a parolei. În

caz afirmativ se returnează id-ul utilizatorului împreună cu un flag din care aplicația Client va ştii că trebuie să acceseze secțiunea de resetare a parolei.

Dacă nici unul dintre cazurile de mai sus nu au fost efectuate cu succes, atunci se furnizează un mesaj eroare, sub forma "User name and/or password are incorrect, please re-enter and submit."

b) Register

Subrutina *register* se ocupă de înregistrarea utilizatorilor în sistem. Dacă datele furnizate de către utilizator sunt valide, atunci acesta este salvat cu succes în baza de date. Parametrii trimiși de către aplicația Client sunt numele de utilizator, parola, numele, prenumele, emailul și numărul de telefon. Se folosește din nou subrutina *trim* pentru a verifica dacă toate șirurile de caractere trimise au cel puțin un caracter. În caz contrar, se returnează un mesaj conținând câmpurile care au rămas necompletate.

Dacă toate câmpurile au fost completate de către utilizator, se trece la eliminarea spațiilor de la începutul și de la sfârșitul numelui și a prenumelui, precum și la eliminarea spațiilor multiple din interiorul acestor cuvinte. Pentru aceasta se folosește subrutina crunch din librăria String::Util.

Apoi urmează pasul de validare a emailului furnizat. Pentru aceasta se folosește subrutina *address* din modulul Email::Valid importat în aplicație. Dacă emailul nu este valid se returnează un mesaj de eroare, 'Email is not valid.'

Următorul pas verifică dacă emailul sau numele de utilizator există deja în baza de date. Dacă da, atunci se returnează mesajul de eroare 'Email/Username already used.' Dacă numele de utilizator și emailul sunt unice, urmează validarea parolei. O parola validă trebuie să conțină cel puțin opt caractere, o literă mare și o cifră. Dacă aceste criterii nu sunt îndeplinite, utilizatorul nu este înregistrat în baza de date, furnizându-se mesajul de eroare "Passwords not strong enough. Password must be at least 8 characters, and contain at least 1 upper case letter and 1 number."

Dacă nu apare nici unul dintre cazurile menționate mai sus, atunci utilizatorul este salvat cu succes în baza de date și se returnează un mesaj de succes.

c) Forgot_password

Această subrutină se ocupă de generarea unei parola aleatoare și de inserarea ei în tabela TemporaryPassword. Așteaptă ca și unic parametru, emailul utilizatorului. Primul pas constă în verificarea ca emailul trimis să nu fie vid, cu ajutorul subrutinei *trim*. Dacă emailul trimis este vid se returnează un mesaj de eroare corespunzător. Altfel se trece la căutarea respectivului email în baza de date. Dacă emailul nu se află în baza de date se iese din subrutina cu un mesaj de eroare.

În cazul în care emailul se găsește în baza de date, urmează pasul de generare a parolei, prin intermediul constructorului din modulul Session::Token. Parola generată este apoi criptată cu md5.

Dacă utilizatorul deja deține o parolă temporară, atunci aceasta va fi înlocuită cu cea nouă. După generarea parolei, această va fi trimisă prin email utilizatorului, împreună cu instrucțiunile de resetare. Emailul este trimis prin intermediul modului existent în aplicație FoodApp::Util::EmailSender.

d) Reset_password

Subrutina *reset_password* se ocupă de modificarea parolei. Aceasta primește ca parametrii id-ul utilizatorului, parola temporară, noua parola și noua parolă confirmată. Primul pas este de verificare ca ambii parametrii care conțin noua parolă să nu fie vizi. Apoi se verifică ca ambii parametrii să coincidă, după care trebuie validată noua parolă, astfel încât să fie puternică (greu de ghicit). Dacă una dintre aceste situații nu este îndeplinită se returnează un mesaj de eroare corespunzător.

Urmează pasul de ștergere a parolei temporare din tabela TemporaryPassword și de criptare a noii parole. Noua parolă este actualizată în tabela User și se returnează un mesaj de confirmare.

5.3.2.2. Modulul Restaurant

Acest modul se ocupă de extragerea listelor de restaurante și constă în 3 subrutine:

- get_listing
- get_logo
- get_favorite_restaurants

a) Get_listing

Subrutina *get_listing* are rolul de a furniza o listă cu restaurantele dintr-un anumit oraș trimis ca parametru. Pentru a se popula lista, se creează un join între tabela Restaurant și tabela Address, având ca filtru orașul trimis ca parametru. În cazul în care nu trimite nici un oraș atunci vor fi extrase toate restaurantele înscrise în aplicație.

Pentru fiecare restaurant vor fi returnate următoarele atribute: idrestaurant, name, description, email, phone, image, country, street, street_no, latitude, longitude, delivery_order, takeout_order, at_restaurant_order.

b) Get_logo

Această subrutină returnează o imagine cu sigla restaurantului a cărui id este trimis ca parametru. Pentru returnarea imaginii se utilizează metoda send_file care aparține framework-ului Dancer. Formatul în care este trimisă imaginea este JPG.

c) Get_favorite_restaurants

Subrutina *get_favorite_restaurants* se ocupă de returnarea restaurantelor favorite ale unui utilizator. Aceasta primește ca și parametru id-ul utilizatorului, iar pentru a extrage restaurantele din baza de date, creează un join între tabela FavoriteRestaurant, Restaurant și Address, având ca filtru id-ul utilizatorului. Atributele returnate pentru fiecare restaurant sunt aceleași cu cele returnate de către subrutina get_listing.

5.3.2.3. Modulul Menu

Acest modul furnizează servicii pentru extragerea categoriilor și a produselor corespunzătoare unei categorii și este format din două subrutine:

- get_categories
- get_food_items

a) Get_categories

Subrutina *get_categories* se ocupă de returnarea tuturor categoriilor de produse pe care un anumit restaurant le oferă. Aceasta primește ca și parametrii id-ul restaurantului și id-ul utilizatorului. Se așteaptă și id-ul utilizatorului pentru a putea fi implementată funcționalitatea de restaurante favorite.

Odată ce utilizatorul a accesat un anumit restaurant pentru prima dată, va fi creată o nouă intrare în tabela FavoriteRestaurant care va avea contorul care reține numărul de accesări, setat pe 1. În cazul în care perechea (id_restaurant, id_user) trimisă ca și parametru există deja în baza de date, atunci contorul respectiv va fi incrementat.

Apoi urmează pasul de extragere al categoriilor restaurantului. Pentru aceasta, se creează un join între tabelele Restaurant, CategoryRestaurant și Category, având ca și filtru id-ul restaurantului. Categoriile sunt extrase în ordine alfabetică direct din baza de date. Pentru fiecare categorie, se trimite id-ul și numele acesteia.

b) Get_food_items

Subrutina get_food_items are rolul de a furniza o listă cu toate produsele dintr-o anumită categorie. Aceasta primește ca și parametrii id-ul restaurantului și id-ul categoriei. Pentru a se extrage din baza de date lista de produse, se creează un join între tabela FoodItem și tabela CategoryRestaurant, având ca și filtre cei doi parametrii primiți. Produsele vor fi extrase în ordine alfabetică.

Pentru fiecare produs se vor returna următoarele atribute: idfood_item, code, name, description, price, quantity, unit_measure, id_restaurant și id_category.

5.3.2.4. Modulul Order

Acest modul se ocupă de efectuarea comenzilor simple și a comenzilor de grup, precum și de gestionarea tuturor funcționalităților care țin de comanda de grup. Este format din următoarele subrutine:

- place_simple_order
- place_group_order
- get_users_group
- invite_people
- accept_group_invitation

a) Place_simple_order

Subrutina *place_simple_order* se ocupă de plasarea efectivă a unei comenzi. Aceasta primește o structură JSON în body-ul request-ului care este mai apoi decodificată cu ajutorul subrutinei decode_json a modulului JSON. Structura primită conține coșul de produse, id-ul utilizatorului, id-ul restaurantului, adresa la care să se efectueze comanda, tipul de comandă și prețul total.

În pasul următor se creează o nouă comandă în baza de date în tabela OrderDetail. Fiecare produs din coș este inserat și el în baza de date în tabela OrdersItem.

b) Place_group_order

Subrutina *place_group_order* se ocupă de finalizarea comenzii de grup. Aceasta primește o structură JSON, care conține produsele comandate de către administrator, id-ul comenzii, tipul de livrare și adresa de livrare. Produsele comandate de către administrator sunt adăugate în tabela OrdersItem, și este setat prețul total în tabela GroupUsers. Apoi se extrag toți membrii grupului din tabela GroupUsers și toate produsele comandate de către aceștia din tabela OrdersItem. Se calculează prețul total pe comanda de grup și se introduce în tabela OrderDetails, iar flagul placed_yn din aceeași tabelă se setează pe true.

c) Get_users_group

Subrutina *get_users_group* extrage toți utilizatorii care aparțin unui anumit grup. Vor fi returnați toți utilizatorii care aparțin aceleiași comenzi ca și utilizatorul trimis ca parametru. Mai întâi se verifică dacă utilizatorul trimis ca parametru aparține unui grup. Dacă nu aparține, se va returna doar un singur utilizator, și anume cel transmis ca parametru. În cazul în care utilizatorul aparține unui grup, vor fi returnați toți ceilalți utilizatori care fac parte din aceeași comandă împreună cu utilizatorul în cauză.

d) Invite_people

Subrutina *invite_people* este apelată atunci când un utilizator dorește să inițieze o comandă de grup prin invitarea altor persoane la comanda sa sau când dorește să invite persoane la o comandă de grup deja existentă.

Subrutina primește ca și parametrii id-ul utilizatorului și id-ul restaurantului, iar apoi se verifică dacă utilizatorul face parte deja dintr-un grup. Dacă utilizatorul aparține unui grup, atunci vor fi returnate linkul, numele restaurantului și id-ul comenzii.

Altfel, se introduce o nouă intrare în tabela OrderDetail, marcându-se comanda ca fiind una de grup. Inițiatorul comenzii de grup este inserat în tabela GroupUser. Vor fi returnați aceiași parametrii ca și în cazul de mai sus.

e) Accept_group_invitation

Subrutina *accept_group_invitation* are rolul de a insera utilizatorul care acceptă invitația în tabela GroupUsers, în cazul în care acesta nu aparține deja unui grup.

Subrutina primește ca parametrii id-ului utilizatorului și id-ul comenzii la care va fi adăugat.

5.3.3. Componenta Util

Componenta Util conține un singur modul, EmailSender, cel care se ocupă de trimiterea de emailuri. Modulul EmailSender este format dintr-un constructor și din metoda send. Constructorul creează un obiect de tipul EmailSender prin cuvântul *bless.* Metoda *send* creează un obiect de tipul Email::Send::SMTP::Gmail, iar apoi apelează metoda send a modului respectiv pentru a trimite emailul.

Modulul Email::Send::SMTP::Gmail este descărcat de pe CPAN, instalat și utilizat în aplicația de față.

5.3.4. Componenta de module DBIx

Componenta de module DBIx conține toate tabelele din baza de date mapate pe aplicația Server. Pentru generarea modulelor s-a folosit un script de Perl, care folosește modulul DBIx::Class::Schema::Loader prin metoda make_schema_at. Scriptul arată în felul următor:

```
use DBIx::Class::Schema::Loader qw/ make_schema_at rescan /;
make_schema_at(
	'FoodApp::Schema',
	{
		debug => 1,
		dump_directory => '.',
		overwrite_modifications => 1,
	},
	[ 'dbi:mysql:androchef_db:127.0.0.1', 'root', 'root',
	],
);
```

Primul parametru reprezintă începutul denumirii fiecărui modul. Cel de-al doilea parametru este un hash care conține trei perechi cheie-valoare. Cheia debug setată pe true înseamnă că vor fi semnalate warning-uri de fiecare dată când va fi executat un statement al modulului DBIx::Class. Cheia dump_directory reprezintă calea relativă unde vor fi salvate modulele generate. În cazul de față va fi directorul curent.

Dacă overwrite_modifications este setat pe false, atunci Loader-ul va refuza să actualizeze orice cod generat anterior care a fost modificat de la ultima sa rulare. În cazul în care este setat pe true, acesta nu va lua în considerare nici o modificare făcută manual de la ultima sa rulare.

Cel de-al treilea parametru este un array care conține pe prima poziție serverul bazei de date, numele acesteia și ip-ul la care se găsește. Pe a doua și a treia poziție se află numele de utilizator și parola cu care se accesează baza de date.

5.4. Aplicația Client – Detalii de Implementare

Aplicația Client rulează pe dispozitivele Android. Aceasta a fost implementată în limbajul Java, având la dispoziție tool-urile din Android SDK. Android SDK conține debugger, librării, emulatoare, documentație, exemple de cod și tutoriale. IDE-ul suportat oficial este Eclipse, care folosește plugin-ul ADT (Android Development Tool) pentru dezvoltare de aplicații Android.

În acest subcapitol vor fi prezentate diagrama de pachete și diagrama de clase a aplicației Client. Totodată, vor fi descrise și clasele împreună cu metodele mai importante ale sistemului.

5.4.1. Diagrama de pachete

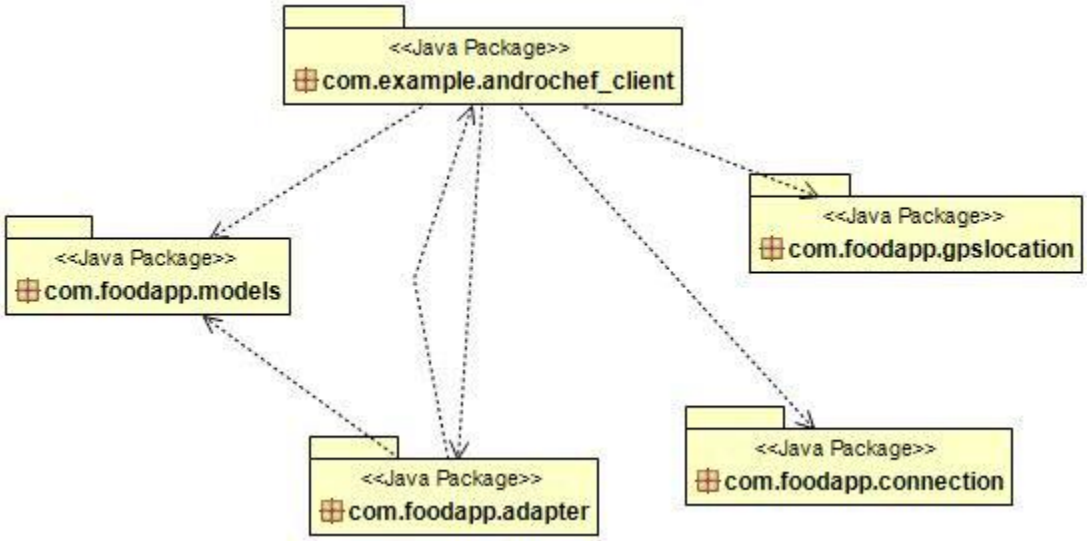

Figura 5.4 Diagrama de pachete a aplicației Client

În Figura 5.4 sunt ilustrate cele 5 pachete ale aplicației Client împreună cu dependențele dintre ele.

Pachetul com.example.androchef_client depinde de toate celelalte 4 pachete. Acest pachet conține clasele care extind Activity. Fiecare clasă din acest pachet are asociat un layout și un meniu, reprezentând Controller-ul pentru aceste View-uri.

Pachetul com.foodapp.adapter este cel care face legătura între listele de obiecte și View-urile pe care aceste liste se vor mapa. Fiecare clasă din acest pachet extinde clasa BaseAdapter.

Pachetul com.foodapp.gpslocation conține o singură clasă care se ocupă extragerea locației curente a dispozitivului. Această clasă extinde Service și implementează LocationListener.

Pachetul com.foodapp.connection conține metode pentru parsarea request-urilor trimise și a response-urilor primite.

Pachetul com.foodapp.model conține partea de model a aplicației care reprezintă logica de business a acesteia. Clasele din care este compus pachetul se ocupă de gestionarea coșului de produse, de produse, de categorii de produse, de restaurante, de comenzi și de utilizatori.

5.4.2. Diagrama de clase

În cele ce urmează va fi ilustrată diagrama tuturor activităților din aplicația Client în Figura 5.5. Diagrama cuprinde activitățile propriu-zise împreună cu relațiile de dependență dintre ele. Activitățile se ocupă de tratarea evenimentelor care apar pe View-urile asociate lor. Fiecare activitate extinde clasa de bază Activity.

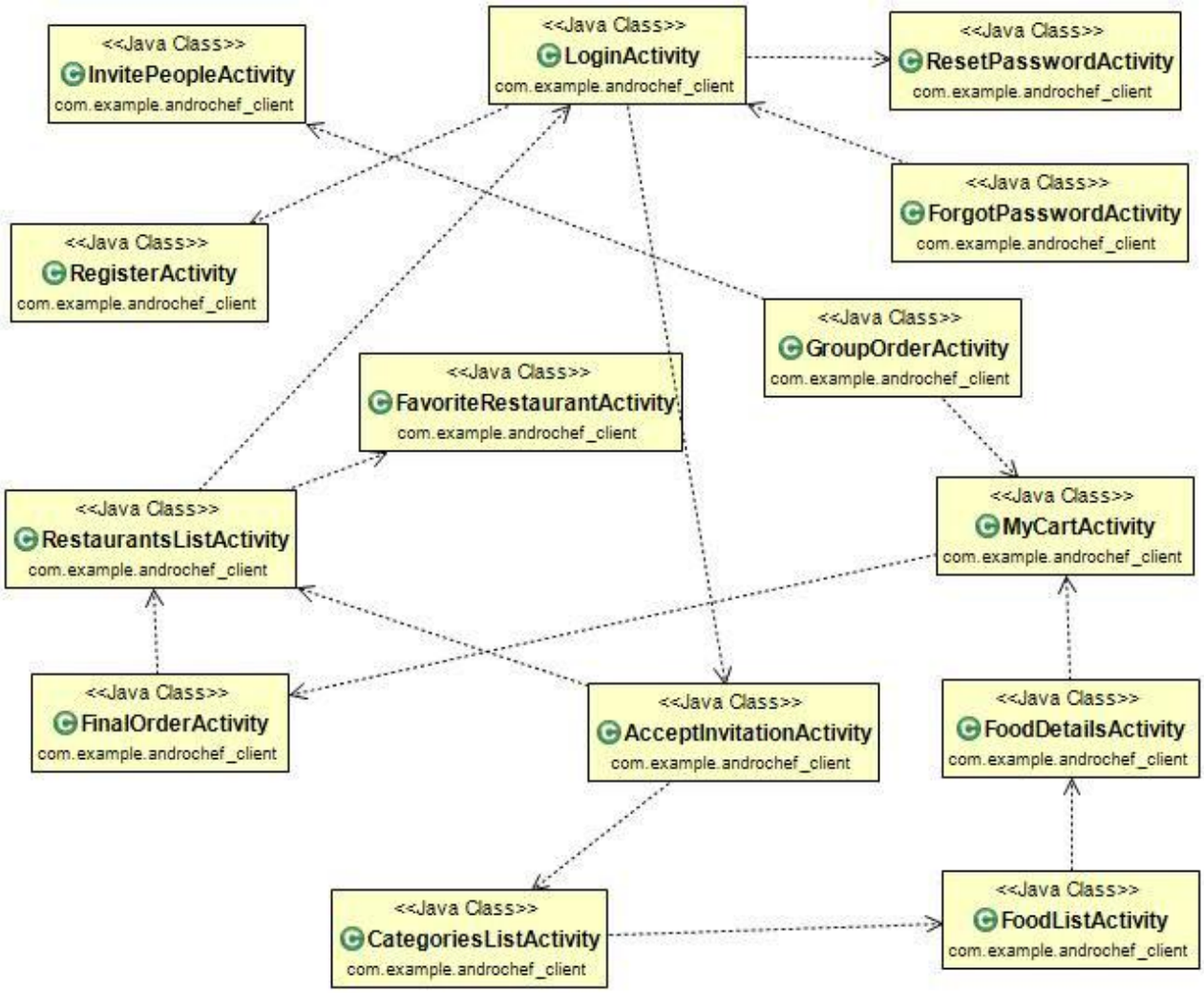

Figura 5.5 Diagrama de activități

Fiecare activitate are asociat un meniu și un layout. Layout-urile care conțin liste (ListView, GridView) sunt compuse din elementele de listă care reprezintă layout-uri separate. Pentru a face legătura între listele de obiecte și lista asociată View-ului se folosesc adaptoare care mapează obiectul pe elementul din listă.

Următoarea listă conține perechi de forma Activity – Adapter - Model, cuprinzând activitățile care au asociate liste împreună cu adaptoarele care mapează modelul pe view:

- RestaurantsListActivity - ItemRestaurantListBaseAdapter - Restaurant
- FavoriteRestaurantsActivity – ItemFavoriteRestaurantAdapter - Restaurant
- CategoryListActivity – ItemCategoryAdapter – Category
- FoodListActivity – ItemFoodListAdapter - FoodItem
- MyCartActivity – ItemCartAdapter - FoodItemOrder
- GroupOrderActivity – UserAdapter – User

Figura 5.6 ilustrează vizual una dintre relațiile menționate mai sus, și anume relația RestaurantsListActivity - ItemRestaurantAdapter - Restaurant.

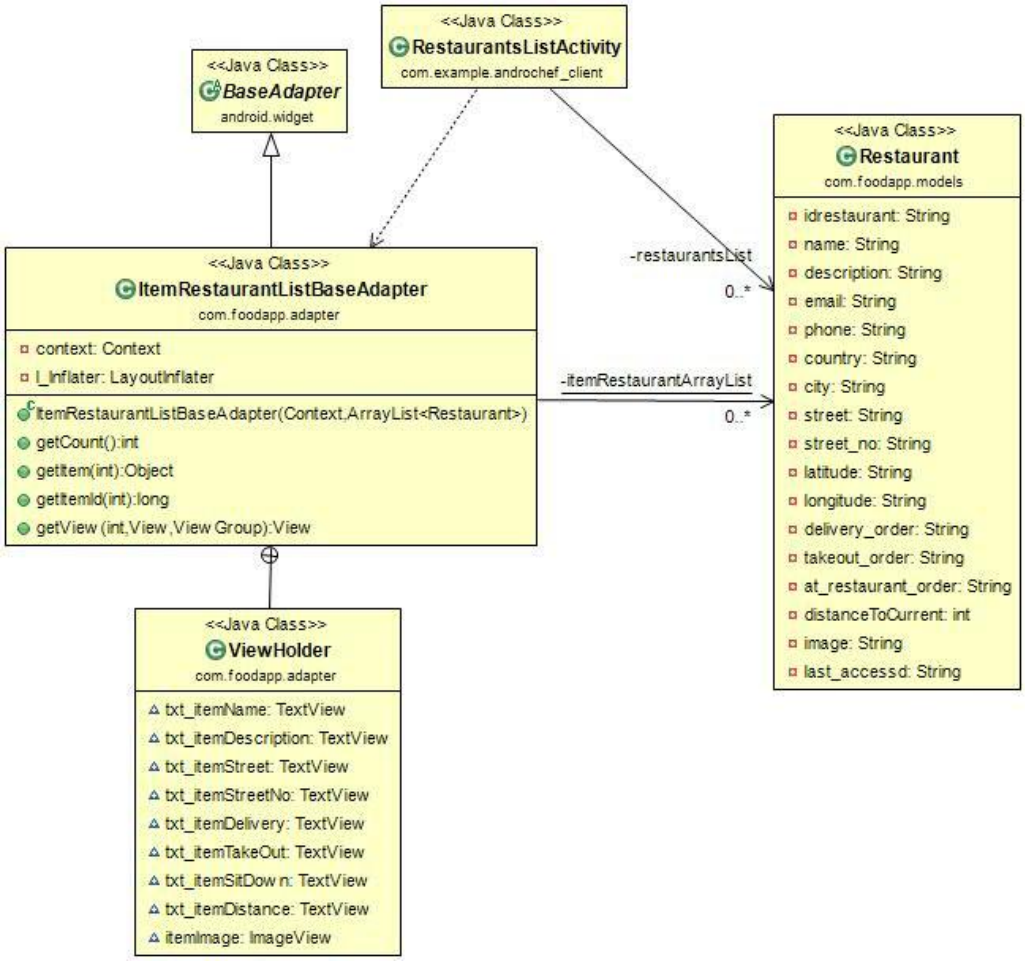

Figura 5.6 Mapare RestaurantsListActivity - ItemRestaurantAdapter - Restaurant

Din Figura 5.6 se observă că ItemRestaurantListBaseAdapter extinde clasa BaseAdapter și conține o clasă internă ViewHolder care are ca și atribute elementele de interfață ale layout-ului unui item din listă. Activitatea RestaurantsListActivity este în relație de dependență cu adaptorul ItemRestaurantListBaseAdapter și în relație de asociere one-to-many cu clasa Restaurant.

Figura 5.7 prezintă clasa statică CartOrder împreună cu atributele sale statice și cu clasele cu care se află în relație de asociere sau de dependență, FoodItemOrder și FoodItem. Activitatea MyCartActivity este cea care încarcă View-ul corespunzător coșului de produse și apelează metodele clasei CartOrder.

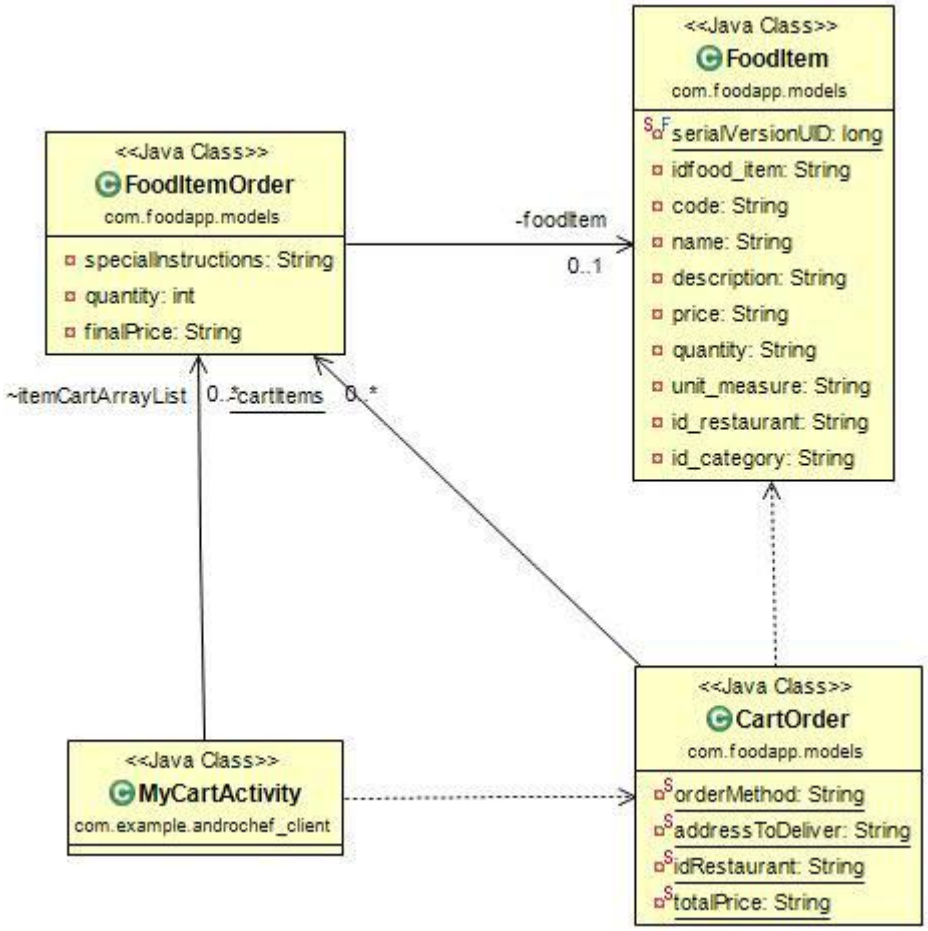

Figura 5.7 Clasa CartOrder și clasele relaționate

Clasa CartOrder este cea mai importantă clasă a pachetului com.foodapp.models. Aceasta cuprinde un ArrayList de obiecte de clasă FoodItemOrder în care sunt reținute toate produsele pe care un utilizator le-a selectat în vederea plasării unei comenzi. Clasa FoodItemOrder este în relație one-to-one cu clasa FoodItem. Pe lângă obiectul de clasă FoodItem, clasa FoodItemOrder mai conține alte atribute specifice, cum ar fi: instrucțiuni

speciale, numărul de itemi din acest produs și prețul total calculat pe baza numărului de itemi și a prețului unitar.

5.4.3. Descrierea claselor importante

Acest subcapitol cuprinde detaliile de implementare ale celor mai importante clase ale aplicației. Va fi detaliată clasa CartOrder care se ocupă de gestionarea coșului de produse, clasa HttpAsyncTask care efectuează request-urile asincrone către Server și clasa GPSTracker care accesează GPS-ul telefonului pentru a detecta locația curentă.

5.4.3.1. Clasa CartOrder

Clasa CartOrder conține toate atributele și metodele necesare pentru efectuarea operațiilor asupra coșului de produse. Atributele de clasă sunt descrise pe scurt în următoarea listă:

- ✓ *cartItems : ArrayList<FoodItemOrder>* - reprezintă coșul de produse propriu-zis, conținând obiecte de clasă FoodItemOrder;
- ✓ *orderMethod : String* – reține metoda de livrare către restaurant (livrare la domiciliu, ridicare din restaurant, comandă la restaurant);
- ✓ *addresssToDeliver : String* – se completează cu adresa la care să se livreze comanda doar în cazul în care metoda de livrare este la domiciliu;
- ✓ *idRestaurant : String* – conține id-ul restaurantului de care aparțin produsele din coș. Este completat doar în cazul în care există produse în coș;
- ✓ *totalPrice : String* – reține prețul total al produselor din coș;

Următoarea listă conține fiecare metodă a acestei clase împreună cu o scurtă descriere asociată:

- ✓ *getAllItems() : ArrayList<FoodItemOrder>* - extragerea produselor din coș pentru vizualizare;
- ✓ *setNewItem(newItem : FoodItemOrder) : void* - adăugarea de produse;
- ✓ *removeItem(position : int) : void* - ștergerea de produse;
- ✓ *checkItemById(itemId : String) : int* - verificarea dacă un anumit obiect FoodItemOrder se găsește în coș pentru a putea fi editat;
- ✓ *deleteAllItems() : void* - resetarea întregului coș atunci când utilizatorul părăsește actualul restaurant;
- ✓ *computeTotalPrice() : String* - calcularea prețului total pentru produsele din coș;
- ✓ *setRestaurant(idRestaurant : String) : void* – setarea noului restaurant pe baza primului produs adăugat în coș;
- ✓ *settere* și *gettere* pentru atributele de clasă;

5.4.3.2. Subclasa HttpAsyncTask

HttpAsyncTask este subclasă a tuturor activităților care efectuează request-uri către Server. Această clasă extinde clasa AsyncTask<String, Void, String> care permite efectuarea de operații în background prin intermediul thread-urilor, fiind o clasă Helper pentru clasele Thread și Handler.

Clasa AsyncTask este recomandată pentru operații mai scurte de 5 secunde și mai lungi de 20 milisecunde. Pentru a evita ca aplicația să devină unresponsive, este necesară efectuarea task-ului pe alt thread decât cel principal pe care rulează toate activitățile. Request-urile către Server durează mai puțin de 5 secunde și în acest caz, clasa AsyncTask poate fi utilizată cu succes.

HttpAsyncTask suprascrie trei dintre metodele clasei AsyncTask, și anume:

- *onPreExecute() : void* – afișează un ProgressDialog pe perioada în care se procesează request-ul către Server;
- *doInBackground(params : String[]) : String* – efectuează un request de tip GET sau POST, în funcție de necesitate, cu ajutorul unui obiect de tip HttpClient și HttpGet sau HttpPost. Params reprezintă parametrii care vor fi trimiși către Server;
- *onPostExecute(result : String) : void* – este apelată în momentul în care ajunge response-ul de la Server și primește ca și parametru response-ul respectiv.

Figura 5.8 reprezintă activitatea LoginActivity, subclasa HttpAsyncTask și relația dintre acestea. Acest exemplu este ilustrativ și pentru celelate activități care conțin subclasa HttpAsyncTask.

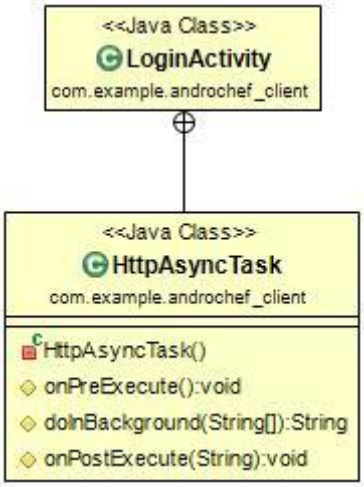

Figura 5.8 Exemplu de subclasă HttpAsyncTask

Clasa privată HttpAsyncTask este apelată în momentul în care utilizatorul completează numele de utilizator și parola, iar apoi apasă butonul de logare. Se

efectuează un request către Server pentru a verifica dacă respectivul utilizatorul a fost înregistrat în aplicație. Clasa este apelată prin următoarea linie de cod
:

```
new HttpAsyncTask().execute(getResources().getString(R.string.server_address) +
        "/user/login", usernameS, passwordS);
```

Primul parametru reprezintă URL-ul, iar usernameS și passwordS sunt numele de utilizator și parola de logare. Prima care se execută este metoda onPreExecute, urmată de doInBackground și apoi de onPostExecute.

Dacă response-ul Server-ului are flagul success setat pe 1, utilizatorul va fi direcționat către lista de restaurante. Altfel, utilizatorul va primi un mesaj de eroare și va trebui să-și completeze din nou credențialele.

5.4.3.3. Clasa GPSTracker

Clasa GPSTracker se ocupă de detectarea automată a locației utilizatorului prin integrarea modulelor GPS în aplicație. Detectarea locației este necesară pentru ordonarea restaurantelor în funcție de locația utilizatorului.

Pentru a putea utiliza această clasă trebuie adăugate permisiuni în fișierul AndroidManifest.xml. Accesarea locației pe baza GPS-ului necesită adăugarea *ACCESS_FINE_LOCATION*, iar accesarea pe baza rețelei necesită adăugarea permisiunii *INTERNET*.

Clasa GPSTracker va accesa serviciile sistemului, deci va trebui să extindă clasa Service. Constructorul are ca și parametru o variabilă de tip Context cu care instanțiază variabila *mContext*. Din constructor se apelează metoda *getLocation()* care extrage o instanță de clasă Location pe baza GPS-ului sau a rețelei și completează variabilele de instanță *location, latitude, longitude* cu valorile obținute. Dacă locația a putut fi detectată cu succes, aceasta însemnând că cel puțin una dintre cele două metode de detectare sunt active, variabila de instanță canGetLocation va fi setată pe true.

Metodele care returnează latitudinea și longitudinea obținută prin metoda getLocation() sunt: *getLatitude()* și *getLongitude()*.

Dacă GPS-ul nu este pornit pe telefon se poate apela funcția *showSettingsAlert()* care deschide un Dialog prin care informează utilizatorul că GPS-ul trebuie pornit pentru a beneficia de funcționalitățile aplicației. Se oferă posibilitatea de a deschide pagina cu setările pentru a activa GPS-ul.

GPSTracker este instanțiat în activitatea RestaurantsListActivity pentru a putea ordona restaurantul după locația utilizatorului. Următoarea secvență de cod arată cum este utilizată clasa GPSTracker în RestaurantsListActivity:

```
// create class object
gps = new GPSTracker(RestaurantsListActivity.this);

// check if GPS enabled
if(gps.canGetLocation()) {
```

```
        currentLatitude = gps.getLatitude();
        currentLongitude= gps.getLongitude();
} else {
        // can't get location
        // GPS or Network is not enabled
        // Ask user to enable GPS/network in settings
        gps.showSettingsAlert();
}
```

5.4.4. Diagramele de secvență pentru comanda de grup

Diagrama de secvență arată interacțiunile dintre obiecte și ordinea în care acestea comunică. Pentru un anumit scenariu sunt extrase clasele și obiectele implicate, precum și secvența de mesaje schimbate între obiecte pentru a duce la îndeplinire funcționalitatea scenariului. Acestea sunt asociate cu realizarea cazurilor de utilizare.

5.4.4.1. Invitarea utilizatorilor la comanda de grup

Figura 5.9 reprezintă primul pas al cazului de utilizare *Comandă de grup,* cel în care sunt lansate invitațiilor către alți utilizatori.

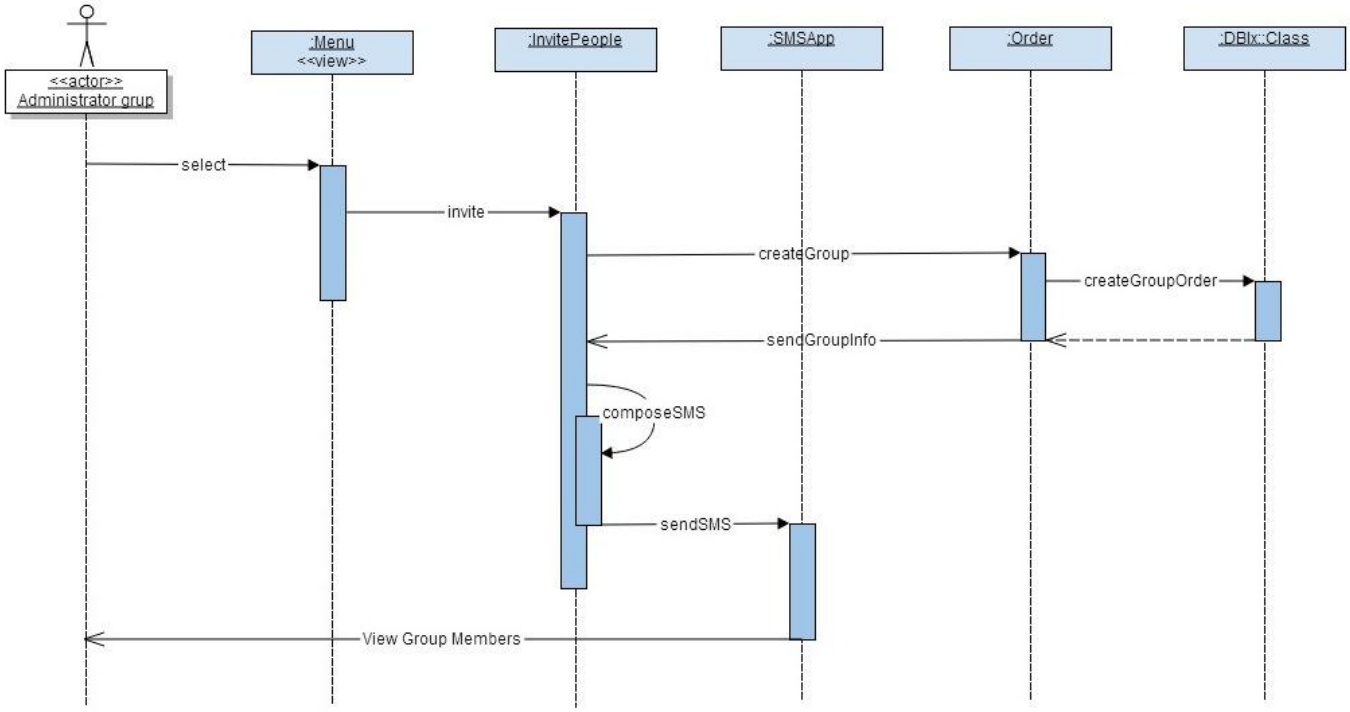

Figura 5.9 Invitarea persoanelor la comanda de grup

Cel care declanșează cazul de utilizare este administratorul de grup. Pentru a putea invita persoane la comanda de grup, acesta trebuie să fi selectat în prealabil restaurantul pentru care să se efectueze comanda. Apoi trebuie să selecteze din meniu opțiunea *Invite People* pentru a deschide aplicația de trimitere de SMS-uri a telefonului.

După aceasta, se efectuează un request către modulul Order de pe Server pentru a înregistra comanda de grup în baza de date.

Între timp, administratorul trebuie să aleagă din lista de contacte persoanele sau grupurile de persoane (se permite crearea de grupuri de contacte) pe care dorește să le invite. După trimiterea SMS-urilor, administratorul de grup va reveni în aplicație pentru a continua cumpărăturile și pentru a vizualiza membrii care s-au înscris până acum în aplicație.

5.4.4.2. Acceptarea invitațiilor la comanda de grup

Figura 5.10 reprezintă pasul de acceptate a invitațiilor de către cei care au primit SMS de la administratorul grupului.

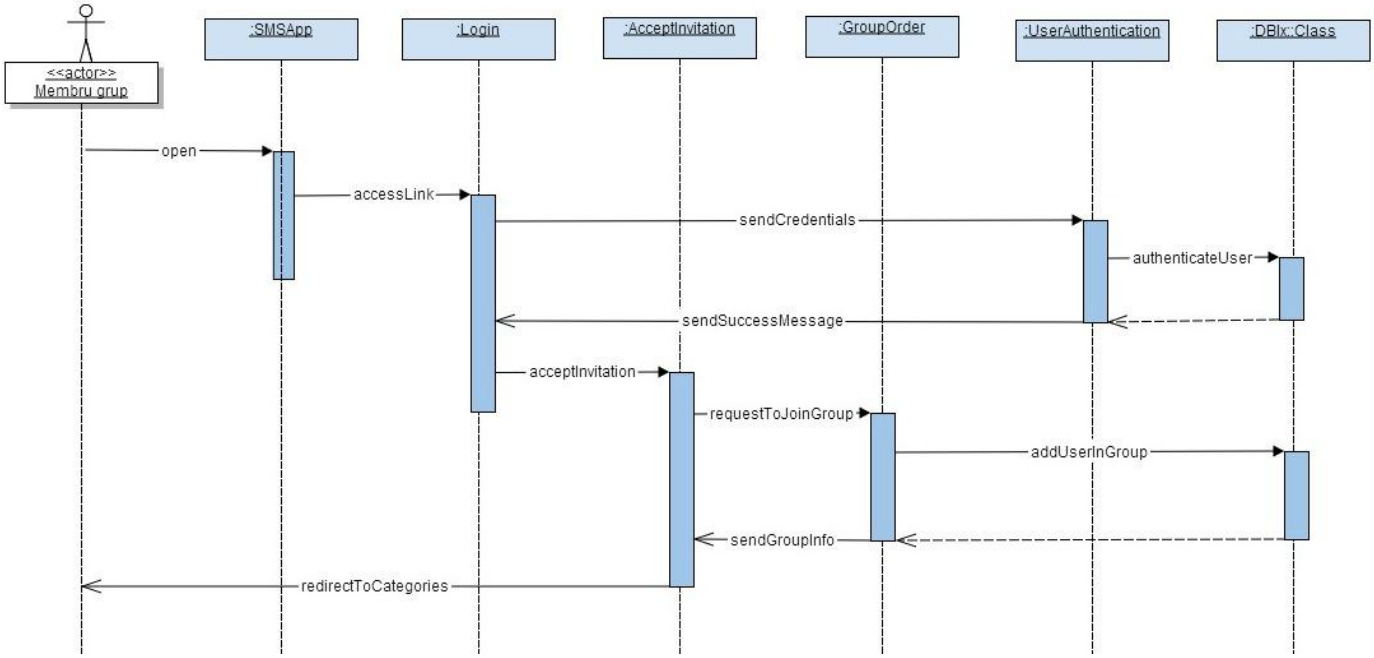

Figura 5.10 Acceptarea invitațiilor

Acceptarea invitațiilor de către cei care doresc să facă parte din comanda de grup este cel de-al doilea pas din cazul de utilizare *Comandă de grup*, după cel în care administratorul de grup lansează invitațiile.

Utilizatorul simplu primește un link care conține informații legate de comanda de grup, precum și un link care trebuie accesat pentru înregistrarea la comandă. După accesarea linkului, se va deschide aplicația AndroChef la pagina de logare, în cazul în

care utilizatorul nu este deja autentificat în aplicație. Dacă deja este autentificat, nu va mai fi necesar să se autentifice din nou.

Urmează pasul de acceptare a invitației. Se deschide un dialog prin care utilizatorul este întrebat dacă sigur dorește să participe la comanda de grup. În cazul în care răspunsul este afirmativ Server-ul va fi înștiințat de înregistrare pentru a adăuga utilizatorul în grupul din baza de date. Utilizatorul simplu devine în acest moment membru al grupului.

Dacă utilizatorul se răzgândește cu privire la participarea la grup, acesta poate selecta butonul Cancel din dialog pentru a putea continua cumpărăturile individual.

Membrul grupului este direcționat în același timp în care Serverul îl înregistrează în baza de date, către categoriile restaurantului pentru care se efectuează comanda de grup.

5.4.4.3. Plasarea finală a comenzii de grup

Plasarea finală a comenzii de grup este cel de-al treilea pas al cazului de utilizare Comandă de Grup și ultimul. Este pasul în care administratorul de grup decide trimiterea comenzii către angajații restaurantului prin email.

Înainte de efectuarea acestui pas toți membrii grupului și-au ales produsele dorite. Figura 5.11 reprezintă diagrama de secvență pentru acest ultim pas, cuprinzând și selectarea produselor de către administrator care conține aceiași pași care trebuie parcurși și la selectarea produselor de către membrii grupului.

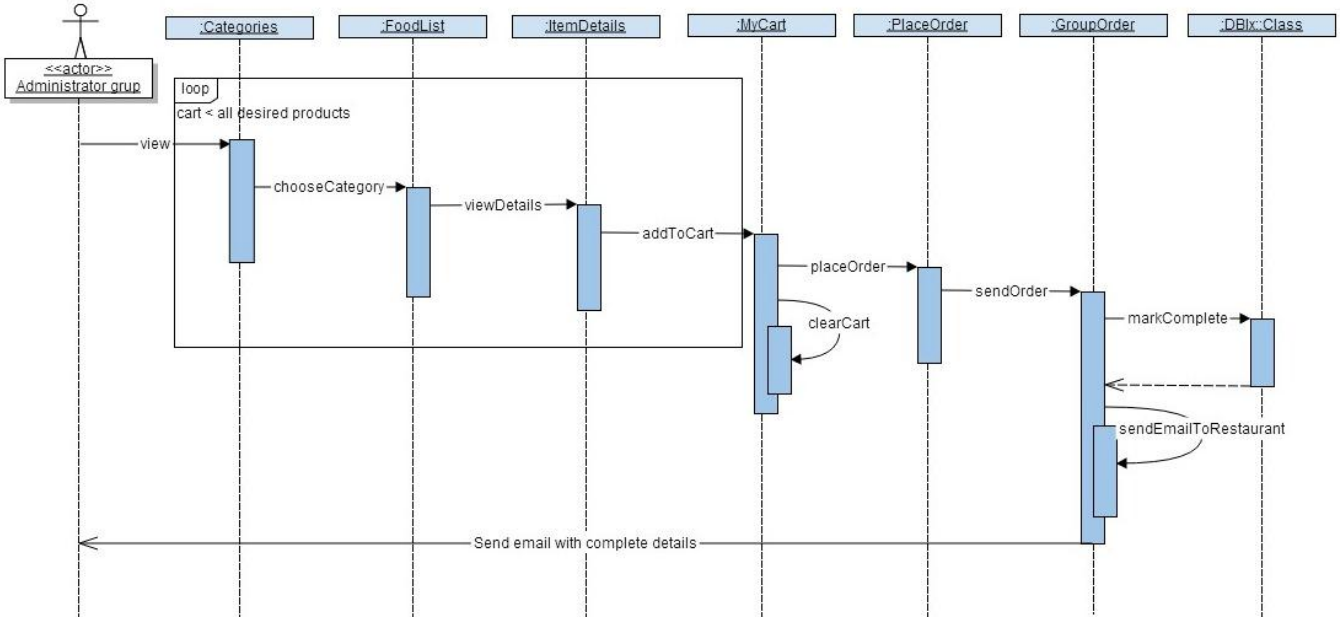

Figura 5.11 Plasarea comenzii de grup

5.5. Baza de date MySQL

Baza de date care asigură partea de persistență a sistemului este gestionată prin intermediul SGBD-ului MySQL. MySQL este cel mai popular SGBD open-source la ora actuală, fiind una dintre componentele cheie a stivei LAMP (Linux, Apache, MySQL, PHP).

Baza de date a aplicației AndroChef are în componența sa 12 tabele, după cum urmează: address, category, category_restaurant, favorite_restaurant, food_item, group_users, order_details, orders_items, restaurant, temporary_password, user și user_type. Diagrama bazei de date se poate vizualiza în Figura 5.12, unde se pot observa tabelele și relațiile dintre ele.

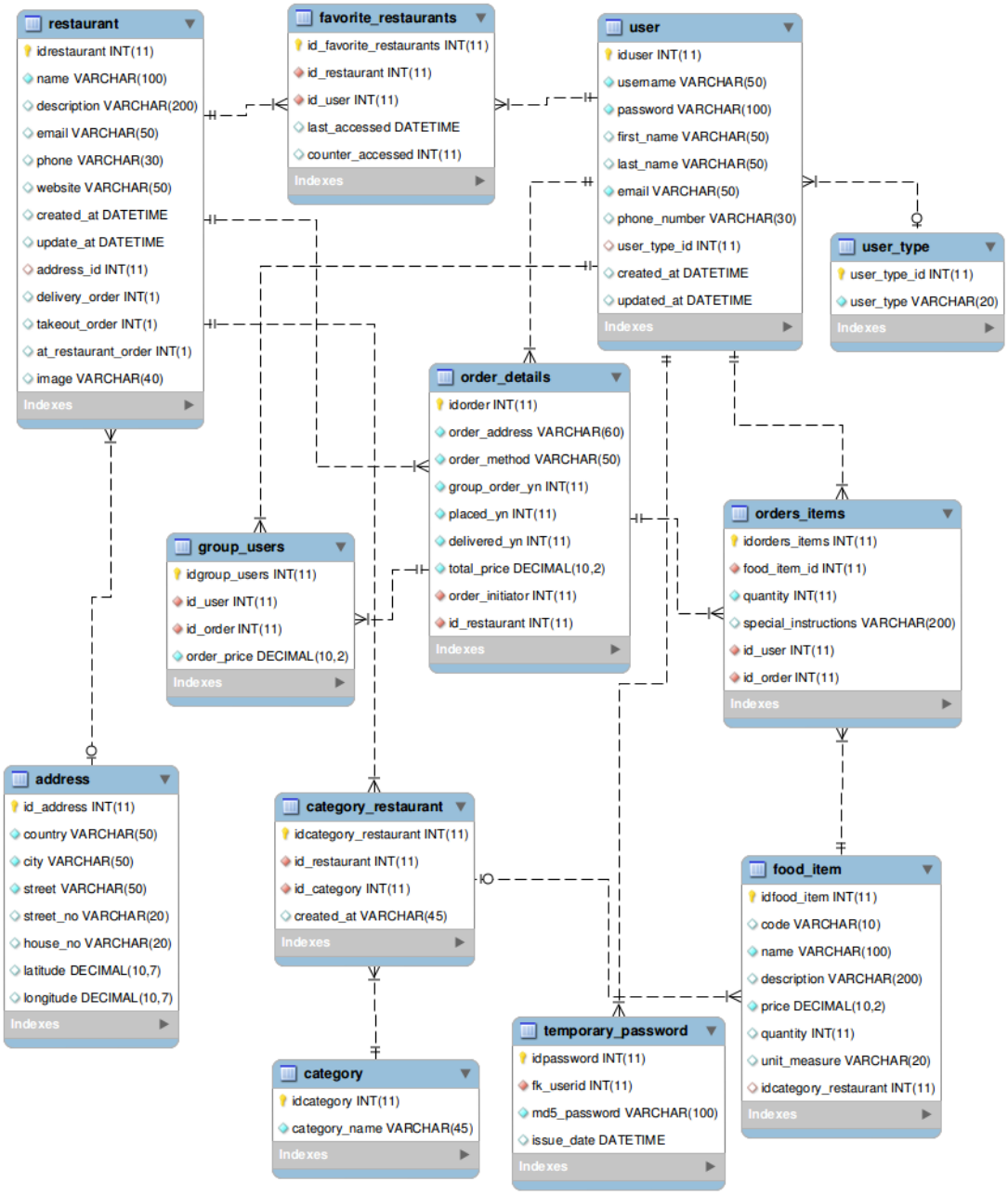

Figura 5.12 Diagrama bazei de date

Capitolul 6. Testare şi Validare

Device-urile mobile devin din ce în ce mai complexe cu o sumedenie de trăsături și funcționalități noi. Multe aplicații, care au fost dezvoltate inițial pentru a servi drept ca și aplicații desktop sau web, sunt acum dezvoltate ca și aplicații pentru device-uri mobile. Deși aplicațiile mobile folosesc resurse limitate, calitatea lor principală trebuie să fie fiabilitatea la fel ca și în cazul aplicațiilor tradiționale.

Metoda principală de a măsura calitatea tuturor aplicațiilor și de a decide dacă acestea sunt de încredere, este testarea manuală sau automată. Acest capitol prezintă cele două metode utilizate pentru validarea sistemului dezvoltat.

6.1. Testarea manuală a aplicației

Testarea manuală a aplicației s-a efectuat după implementarea fiecărei noi funcționalități. Beneficiul testării iterative este reprezentat de găsirea din timp a anumitor bug-uri și de fixarea lor imediată. Rezolvarea din timp a problemelor care apar în aplicație are mai puține costuri în ceea ce privește timpul și complexitatea metodei de rezolvare decât dacă acestea ar fi fixate la sfârșitul implementării.

Deşi tehnica de testare a aplicațiilor mobile diferă de cea a testării dispozitivelor tradiționale, în esență, aceleași metode de testare se folosesc. În procesul de dezvoltare a aplicației mobile AndroChef, numeroase tehnici de testare manuală au fost utilizate, precum: Black Box Testing, White Box Testing și Regression Testing.

6.1.1. Black Box Testing (Functionality Testing)

Prin testarea funcțională se înțelege testarea funcționalităților de bază ale aplicațiilor mobile, conform specificațiilor, pentru a stabili dacă ceea ce s-a cerut s-a și implementat. Această metodă de testare poate implica testarea interfeței utilizator, API-urile, testarea bazelor de date, securitatea sau instalarea. Metoda de testare Black Box se referă la testarea aplicațiilor, fără acces la detalii interne, în particular la codul sursă.

Pentru fiecare funcționalitate nou implementată s-a testat dacă aplicația Client se conectează la Server, dacă baza de date se actualizează corespunzător cerințelor și dacă interfața răspunde corect interacțiunii cu utilizatorul. De asemenea, au fost testate toate fluxurile alternative ale funcționalităților autentificare, înregistrare și recuperare parolă și ale funcționalităților comandă de grup și comandă simplă.

6.1.2. White Box Testing

White Box Testing este opusul metodei Black Box Testing. În acest caz, se presupune că tester-ul aplicației are acces la sursele programului (structuri, cod, algoritmi). De multe ori, testarea prin această metoda implică scrierea de cod sau cel

puțin urmărirea celui existent. Practic, luând la cunoștință construcția programului și modul său de lucru, se testează fiecare metodă în parte inițial, iar apoi interoperarea metodelor.

Fiecare subrutină de pe Server, dezvoltată pentru a răspunde request-urilor HTTP de la Client, a fost mai întâi testată în browser, iar dacă au fost găsite buguri, acestea au fost fixate înainte de integrarea subrutinei în Client-ul Android. După integrarea cu Client-ul s-a testat conexiunea dintre cele două subsisteme și s-a verificat ca request-ul către Server să fie formulat corect.

6.1.3. Regression Testing

Regression Testing reprezintă orice tip de testare software care încearcă să descopere erorile de soft prin retestarea parțiala a unui program modificat. Prin intermediul Regression Testing se urmărește să nu se introducă erori suplimentare în procesul de corecție a altor probleme. Unul dintre principalele motive pentru realizarea testării de regresie este că adesea pentru un programator este extrem de dificil să-și dea seama cum o schimbare într-o parte a aplicației va avea efecte în alte părți ale acesteia.

După implementarea unei noi funcționalități s-au testat toate celelalte funcționalități deja implementate și testate care au fost în strânsă legătură cu noua funcționalitate dezvoltată. De exemplu, după implementarea cazului de utilizare comandă de grup s-a verificat ca funcționalitatea comandă individuală să nu fie afectată, deoarece comanda de grup are la bază mai multe comenzi individuale care sunt efectuate de către membrii grupului.

6.2. Testarea automată a aplicației

Pe lângă mijloacele de testare manuală folosite pentru asigurarea calității aplicației AndroChef, s-a aplicat, de asemenea, și testarea automată, având posibilitatea de a alege dintre mai multe framework-uri existente. Au fost selectate două framework-uri care vor fi analizate: Sikuli Test si Robotium. Unul dintre acestea va fi ales pentru testarea automată a aplicației.

6.2.1. Sikuli Test

Sikuli Test este un GUI framework, care oferă posibilitatea de a crea și rula teste automate. Permite scrierea de teste și automatizarea lor, prin intermediul cărora se verifică conformitatea funcționalităților implementate. Testele automate folosind Sikuli Test au fost dezvoltate în principal pentru a testa aplicațiile desktop pe platformele Windows sau Mac OSX, dar și pentru testarea aplicațiilor mobile instalate pe sistemele de operare Android sau iOS.

Sikuli Test folosește următoarele două funcții de aserțiune pentru efectuarea testelor:

- assertExist(image or string [, region]) – metodă de aserțiune care permite verificarea dacă o imagine sau anumite informații apar pe ecranul device-ului sau într-o anumită regiune specificată.
- assertNotExist(image or string [, region]) – metodă de aserție care permite verificarea dacă o imagine sau anumite informații nu apar pe ecranul device-ului.

Sikuli Test are momentan o limitare foarte importantă, din cauza căreia a fost necesară utilizarea altui framework. Aceasta limitare se referă la faptul că Sikuli Test nu dispune de o modalitate de testare a funcționalităților propriu-zise ale aplicațiilor, ci doar testarea vizuală a acestora.

6.2.2. Robotium

Robotium este un framework de testare automată a device-urilor Android, fiind dezvoltat și lansat în 2010. Ceea ce deosebește Robotium de alte framework-uri este faptul că dispune de un recorder eficient care oferă posibilitatea înregistrării acțiunilor realizate pe device-urile mobile deschise în emulatoare. Aceste acțiuni înregistrate pot fi salvate ca și scenarii de test care vor putea fi rulate pentru a stabili conformitatea aplicației.

6.2.3. Testarea automată a comenzii individuale cu Robotium

În cele ce urmează va fi prezentat un test automat aplicat funcționalității comandă individuală. Testarea funcționalității comandă de grup diferă de cea individuală doar prin existența funcționalităților de gestionare a grupului. Va fi prezentat cazul de test al comenzii individuale și nu al comenzii de grup, deoarece comanda de grup are la bază mai multe comenzi individuale, comanda individuală reprezentând funcționalitatea esențială a aplicației.

Pentru testarea automată a comenzii individuale s-a folosit framework-ul Robotium. Cazurile de test au fost înregistrate prin intermediul lui Robotium, în momentul în care au fost testate manual funcționalități ale aplicației. De exemplu, după implementarea comenzii individuale s-a aplicat testarea manuală pentru validarea scenariilor de utilizare. Cazurile de test efectuate prin testarea manuală au fost înregistrate cu ajutorul lui Robotium Recorder, acesta creând un nou proiect în care au fost salvate cazurilor de test importante.

După implementarea funcționalității comandă de grup, cazurile de test înregistrate pentru comanda individuală au fost rulate pentru a verifica păstrarea integrității sistemului. Proiectul care conține cazurile de test salvate poate fi rulat ca și Android JUnit Test. Poate fi executat întregul proiect sau doar anumite cazuri de test.

Figura 6.1 prezintă execuția cazului de test al comenzii individuale.

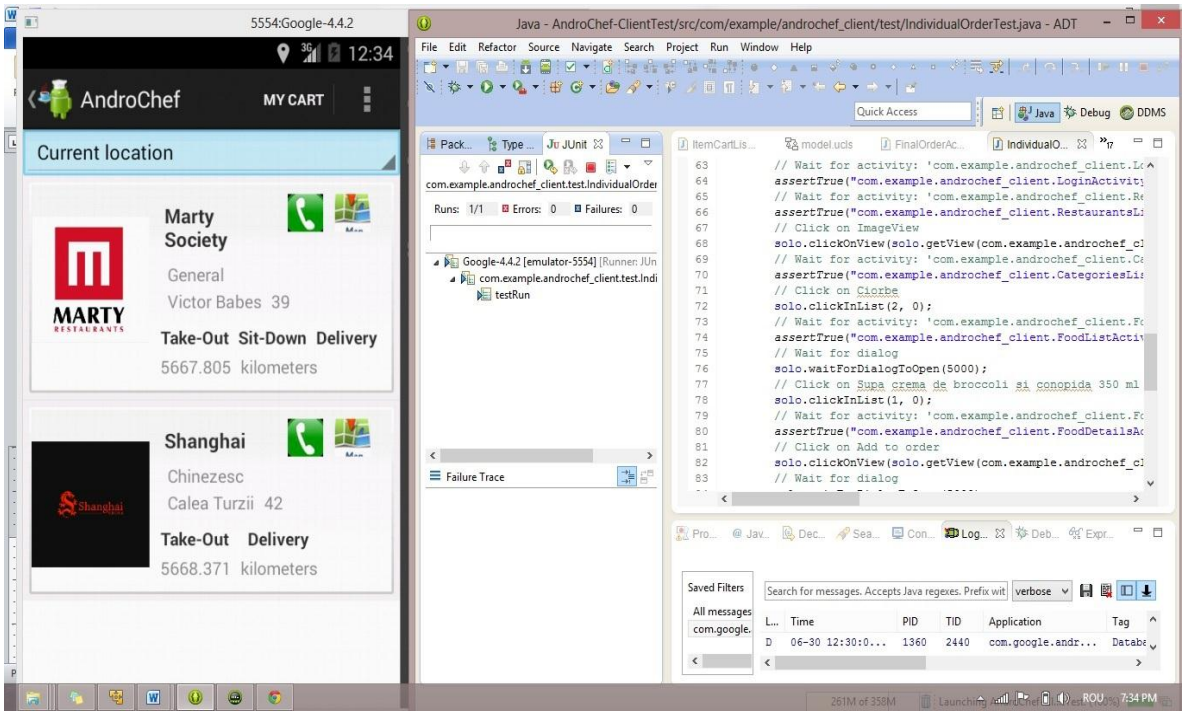

Figura 6.1 Caz de test pentru comanda individuală

Capitolul 7. Manual de Instalare si Utilizare

Această secțiune prezintă resursele hardware și software necesare pentru instalarea și rularea aplicației Server și a aplicației Client, precum și pașii care trebuiesc parcurși pentru instalarea celor două.

De asemenea, va fi detaliat procesul de efectuare a unei comenzi de grup, comanda de grup fiind compusă din mai multe comenzi individuale. Vor fi prezentate și metodele de autentificare în sistem.

7.1. Instalarea aplicației AndroChef

7.1.1. Instalarea aplicației Server

1) Cerințele hardware pe care le impune aplicația Server sunt următoarele:

- ✓ Un computer pe care să fie instalat unul dintre sistemele de operare Linux sau Windows (de preferat Linux deoarece oferă suport pentru aplicațiile scrise în Perl);
- ✓ Minim 1 GB de memorie RAM;
- ✓ Minim 2 GB de memorie fizică;

2) Cerințele software necesare rulării aplicației Server împreună cu pașii de instalare sunt:

- ✓ Modulele importate în aplicație trebuie mai întâi instalate de pe CPAN: Digest::MD5, Email::Valid, Session::Token și Email::Send::SMTP::Gmail
  - o pentru a instala un modul Perl se folosesc următoarele linii de comandă în Terminal:

    cpan App::cpanminus
    cpanm Email::Send::SMTP::Gmail

- ✓ Trebuie să fie instalat framework-ul Dancer
  - o se efectuează prin următoarea linie de comandă în Terminal:

    curl -L http://cpanmin.us | perl - --sudo Dancer
    cd FoodApp-AndroChef
    plackup ./bin/app.pl – pornirea aplicației Server

- ✓ Trebuie instalat serverul de baze de date MySQL;
  - o instalarea serverului de baze de date se efectuează prin secvența:

    sudo apt-get install mysql-server
    sudo netstat -tap | grep mysql – se verifică dacă serverul a pornit

sudo service mysql restart – repornirea serverului
sudo dpkg-reconfigure mysql-server-5.5 – pentru a seta parola *root*
mysql -u root -p root < androchef_db.sql – se importă baza de date a aplicației AndroChef în serverul MySQL

7.1.2. Instalarea aplicației Client

1) Cerințele hardware pe care le impune aplicația Client sunt următoarele:

- ✓ Un telefon inteligent sau o tabletă pe care să fie instalat Android OS;
- ✓ Cel puțin 15 MB de memorie fizică;
- ✓ Conexiune la Internet;

2) Cerințele software necesare rulării aplicației Client sunt următoarele:

- ✓ Dispozitivul Android trebuie să aibă instalată versiunea Android 3.0 sau versiunile apărute după aceasta;

3) Pașii de instalare a aplicației Client:

a. Descărcarea și instalarea aplicației ASTRO File Manager de pe Google Play;
b. Conectarea dispozitivului la calculatorul pe care se află AndroChef.apk și copierea pachetului pe dispozitiv;
c. Se deschide AndroChef.apk de pe dispozitiv și se apasă butonul Install pentru instalarea aplicației;

7.2. Manual de utilizare

7.2.1. Înregistrare, Autentificare, Recuperare parolă

După efectuarea pașilor de instalare a aplicației se poate trece la utilizarea ei. Pentru aceasta, trebuie găsită aplicația cu numele AndroChef printre celelalte aplicații. După deschiderea aplicației, utilizatorul este întâmpinat de pagina Login din Figura 7.1. Dacă acesta nu este încă înregistrat în aplicație, poate accesa secțiunea de înregistrare prin apăsarea butonului Register. Formularul de înregistrare se poate vizualiza tot în Figura 7.1. Toate câmpurile pe care le conține trebuie completate.

Utilizatorul înregistrat în aplicație poate să se autentifice prin completarea numelui de utilizator și a parolei furnizate la înregistrare.

Dacă utilizatorul înregistrat nu-și amintește numele de utilizator sau parola, acesta poate accesa secțiunea Forgot Password prin apăsarea butonului cu același nume. În pagina Forgot Password din Figura 7.2, utilizatorul trebuie să completeze adresa de email la care să i se trimită numele de utilizator și parola temporară. Parola temporară poate fi utilizată o singură dată, iar după autentificarea utilizatorului cu acea parolă, acesta va

direcționat către pagina Reset Password, unde va trebui să furnizeze o nouă parola. Secțiunea Forgot Password se poate observa în Figura 7.2.

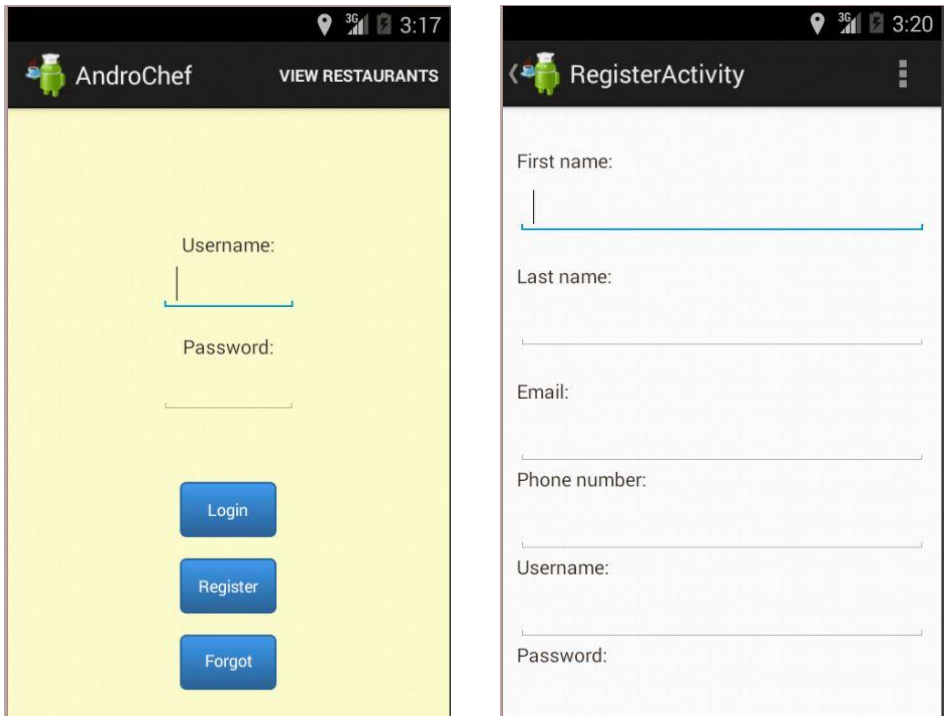

Figura 7.1 Pagina de Login și Register

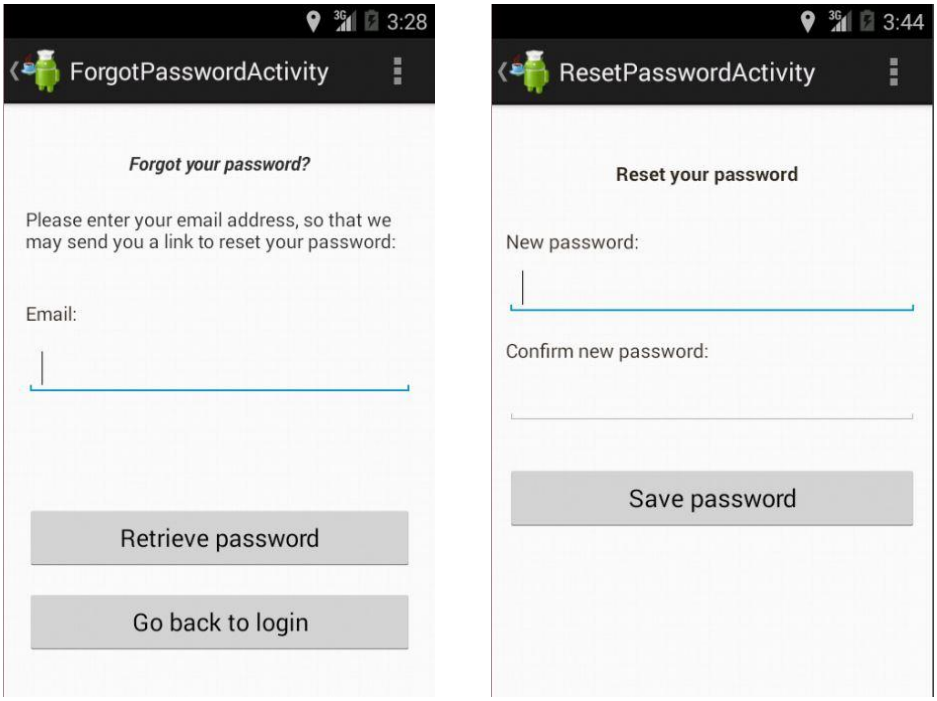

Figura 7.2 Pagina Forgot Password

7.2.2. Comanda de grup

Plasarea unei comenzi individuale sau de grup poate fi efectuată doar de către utilizatorii autentificați în sistem, în timp ce utilizatorii neautentificați au dreptul să vizualizeze restaurantele împreună cu produsele pe care le oferă.

Efectuarea unei comenzi de grup începe prin selectarea restaurantului de către administratorul de grup, de la care acesta împreună cu membrii grupului doresc să comande produse. Restaurantul poate fi selectat în funcție de locația la care se află grupul, din lista ordonată crescător după distanța dintre locația curentă și restaurant (Figura 7.3) sau din lista de restaurante dintr-un anumit oraș.

Pentru acest exemplu va fi selectat cel de-al doilea restaurant din lista, deoarece distanța dintre cele două este doar de 1 km și acest fapt nu influențează timpul de livrare al comenzii. După selectarea restaurantului, va fi prezentată lista de categorii de produse de care restaurantul dispune în Figura 7.3.

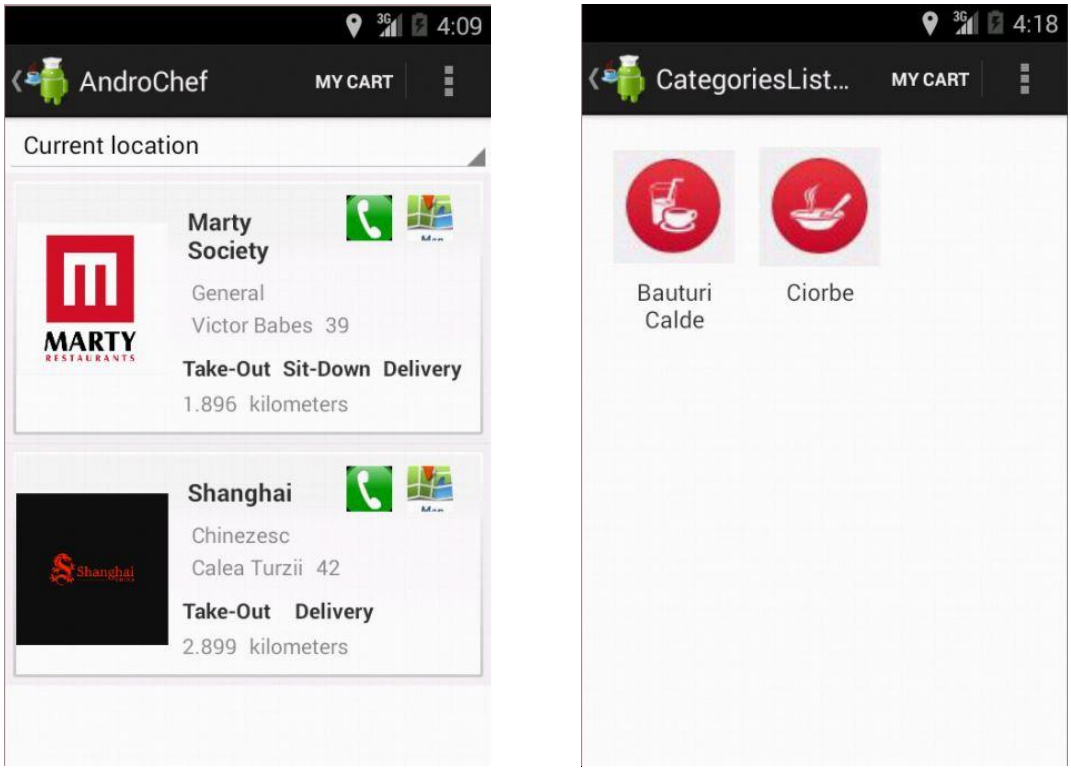

Figura 7.3 Lista de restaurante și categorii

Categoria selectată în continuare este categoria de ciorbe. După selectarea acesteia, se deschide lista de produse din această categorie, care poate fi vizualizată în Figura 7.4. Apoi urmează selectarea unui produs din lista pentru a vedea toate detaliile acestuia. Se va selecta primul produs din listă. Detaliile acestui produs pot fi vizualizate tot în Figura 7.4.

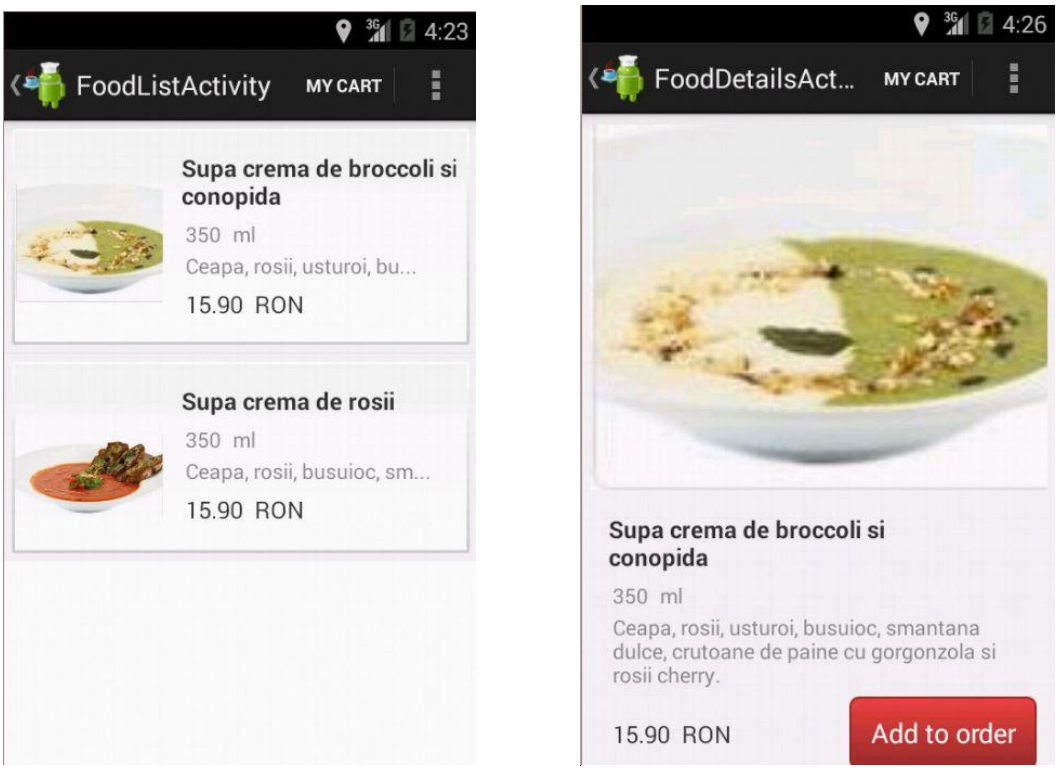

Figura 7.4 Lista de produse și detaliile unui produs selectat

Pentru adăugarea produsului în coș se va apăsa butonul Add to order din Figura 7.4. După aceasta, se va deschide un dialog în care se pot completa instrucțiuni speciale legate de produs, precum și numărul de itemi din acest tip de produs, după cum se poate observa în Figura 7.5.

După apăsarea butonului Add to order, produsul este adăugat în coș. Coșul de produse poate fi vizualizat în Figura 7.5. Din această secțiune începe efectiv funcționalitatea de comandă de grup atunci când utilizatorul decide să invite și alți utilizatori la comanda sa. Acesta selectează din meniu opțiunea Invite People. În acel moment se deschide secțiunea care conține lista cu toți membrii grupului de până acum. Deoarece utilizatorul nu a lansat nici o invitație până acum, grupul va fi format doar dintr-o singură persoană și anume administratorul de grup.

Administratorul lansează invitații la comanda de grup prin apăsarea butonului Invite People din secțiunea de grup. Se deschide aplicația de trimitere de SMS-uri a dispozitivului, în care este completat mesajul care trebuie trimis invitaților. Mai trebuie selectate doar contactele din lista de contacte, la care să fie trimise invitațiile. Mesajul trimis către ceilalți utilizatori este ilustrat în Figura 7.6.

Cei care primesc acest mesaj și doresc să ia parte la comanda de grup pentru acel restaurant trebuie doar să acceseze link-ul trimis pentru a deschide aplicația direct la restaurantul ales de către administrator. Fiecare membru al grupului va putea vizualiza componența acestuia prin selectarea opțiunii View Group din meniu.

Fiecare membru al grupului parcurge aceiași pași pe care i-a parcurs și administratorul de grup pentru selectarea produselor dorite, cu mențiunea că membrul

grupului nu mai trebuie să precizeze tipul de livrare și adresa de livrare pentru livrarea la domiciliu.

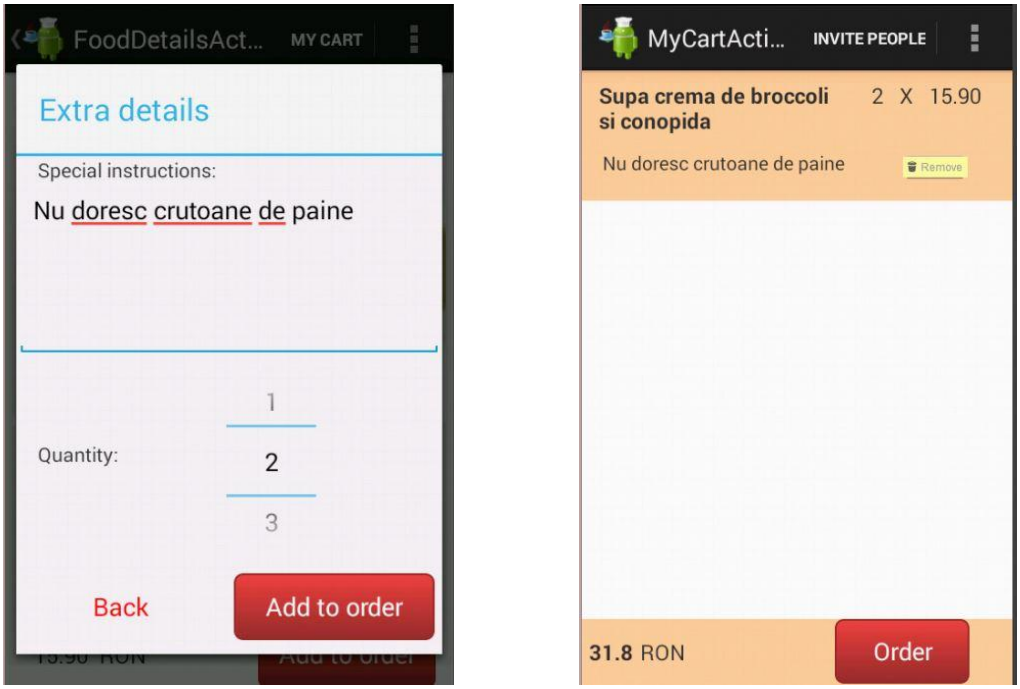

Figura 7.5 Selectarea detaliilor suplimentare și coșul de produse

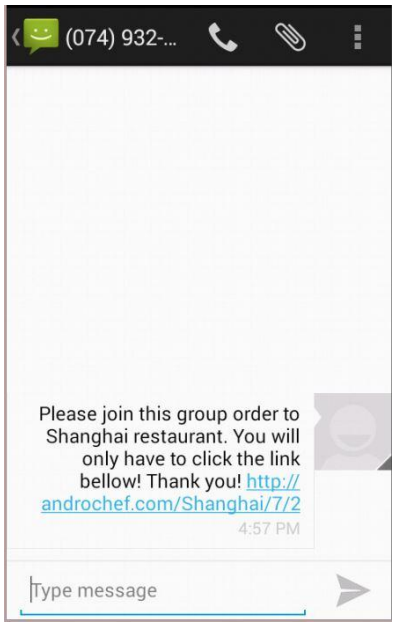

Figura 7.6 Mesajul de invitare la comanda de grup

Finalizarea comenzii poate fi efectuată doar de către administratorul grupului, chiar dacă doar o parte din membrii grupului și-au finalizat comanda individuală în cadrul grupului. Acesta trebuie să precizeze tipul de livrare și adresa la care să se livreze produsele în cazul în care a fost ales tipul livrare la domiciliu. În Figura 7.7 se observă ultimii pași care trebuiesc parcurși înainte de plasarea comenzii către restaurant.

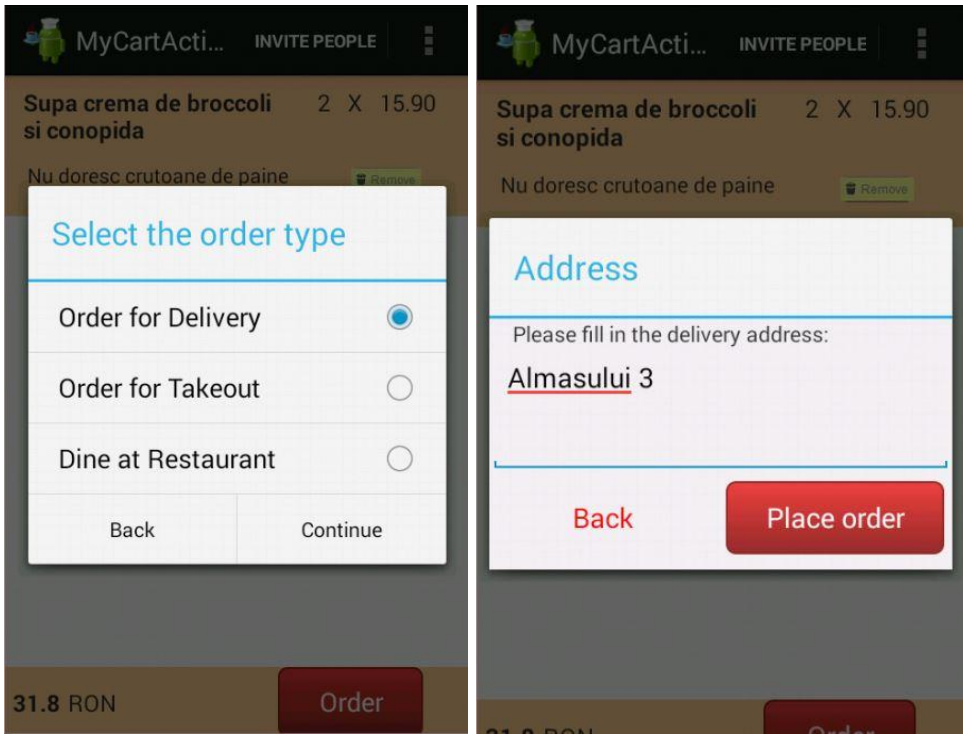

Figura 7.7 Selectarea tipului de livrare și a adresei

După finalizarea comenzii, va fi trimis un email la adresa restaurantului, conținând toate informațiile necesare onorării comenzii. De asemenea, fiecare membru al grupului va primi un email cu produsele comandate și prețul acestora. Administratorul de grup va primi un email cu suma pe care fiecare membru al grupului trebuie să o plătească.

.

Capitolul 8. Concluzii

Acest ultim capitol prezintă rezultatele obținute și măsura în care obiectivele propuse au fost atinse. De asemenea, sunt enumerate dezvoltările ulterioare posibile pentru îmbunătățirea aplicației.

8.1. Realizări

Aplicația AndroChef a fost dezvoltată pentru a facilita și a eficientiza comanda de produse de la restaurante de către grupurile de persoane. Realizarea acestui principal obiectiv a adus cu sine și dezvoltarea funcționalității de comandă individuală, comanda de grup având la bază comenzi individuale efectuate de către membrii grupului.

Așadar, aplicația va permite utilizarea sa cu succes atât în cadrul grupurilor de persoane, cât și în cazul comenzilor individuale. Această realizare este foarte importantă, având în vedere că aplicațiile mobile de acest tip existente deja pe piață nu permit plasarea de comenzi de grup.

Comanda de grup presupune ca fiecare membru al grupului să-și aleagă produsele de la restaurantul selectat de către administrator prin intermediul aplicației, iar la final să se plaseze o comandă unică pe grup către acel restaurant. Sarcina de efectuare a comenzii nu mai revine doar unei singura persoane care trebuie să adauge în coș toate produsele pe care fiecare membru dorește să le comande, ci este distribuită pe întregul grup cu ajutorul funcționalității de comandă de grup.

Celelalte obiective secundare propuse au fost realizate aproape în totalitate. A fost dezvoltat un sistem de autentificare în aplicație care este necesar atunci când utilizatorul dorește să plaseze o comandă de grup sau una individuală, pentru ca angajații restaurantului sau ceilalți membrii ai grupului să poată identifica persoana. Autentificarea în sistem nu este necesară dacă utilizatorul dorește doar să vizualizeze lista de restaurante împreună cu produsele pe care le oferă. De asemenea, utilizatorul care s-a autentificat în aplicație nu mai trebuie să se autentifice din nou la redeschiderea acesteia, salvând astfel din timpul utilizatorului.

Printre obiectivele secundare realizate se numără și vizualizarea unei liste cu restaurantele preferate ale utilizatorului. Această listă este ordonată descrescător pe baza comenzilor efectuate către restaurantele din listă. Totodată, se permite și vizualizarea unei liste ordonate crescător în funcție de distanța dintre locația curentă a clientului și locația restaurantului, astfel încât clientul să știe la care restaurant poate să ajungă cât mai repede sau care dintre restaurante poate să îi livreze comanda într-un timp cât mai scurt.

Localizarea pe hartă a restaurantului și posibilitatea de a contacta telefonic angajații restaurantului sunt obiective atinse care aduc un plus sistemului dezvoltat, deoarece utilizatorii nu mai trebuie să efectueze pași în plus necesari deschiderii altor aplicații pentru a beneficia de aceste servicii.

Posibilitatea de a vizualiza un istoric al comenzilor efectuate este un obiectiv secundar nerealizat. În lista cu istoricul comenzile efectuate, utilizatorul ar trebui să aibă posibilitatea de a clasifica produsele comandate în funcție de gradul de satisfacție pe care produsul l-a oferit clientului. Această funcționalitate ar ajuta utilizatorul să aleagă pentru viitoarele comenzi, produse pe care le-a mai consumat și de care a fost mulțumit.

Totodată, în cazul în care nu a fost mulțumit de unele dintre produsele comandate, va ştii pe baza istoricului comenzilor care dintre acestea nu vor mai face parte din viitoarele sale comenzi.

Printre realizările de natură tehnică se numără învățarea dezvoltării de aplicații mobile pe platforma Android şi a dezvoltării de aplicații de tip web server în limbajul Perl cu ajutorul framework-ului Dancer.

Produsul final este o aplicație Android care poate fi utilizată de către orice persoană care deține un dispozitiv cu Android OS instalat şi care dorește să efectueze o comandă către unul dintre restaurantele preferate, într-un mod cât mai eficient.

8.2. Dezvoltări ulterioare

Sistemul dezvoltat poate fi îmbunătățit prin adăugarea de noi funcționalități şi prin îmbunătățirea funcționalităților deja dezvoltate. De asemenea, printre dezvoltările ulterioare se numără şi creşterea performanței aplicației. Următoarea listă conține noile funcționalități care ar putea fi adăugate pe viitor sistemului:

- Furnizarea unui istoric al comenzilor efectuate în care să se permită clasificarea produselor comandate pe baza gradului de satisfacție al clientului;
- Posibilitatea de adăugare de rating-uri pentru restaurantele înscrise în sistem şi produsele pe care acestea le pun la dispoziție. Această funcționalitate vine în ajutorul clienților atunci când doresc să aleagă un anumit produs de la un anumit restaurant de la care nu au mai comandat niciodată;
- Încărcarea imaginilor la profilul utilizatorului prin accesarea camerei telefonului sau prin selectarea unei fotografii din galerie. Această funcționalitatea permite o mai uşoară identificare a persoanelor în cadrul grupului;
- Furnizarea de informații legate de fiecare restaurant, cum ar fi: orarul de funcționare, o scurtă descriere, taxa de livrare şi suma minimă pentru care nu se percepe taxa de livrare;
- Posibilitatea de a selecta limba română ca fiind metoda prin care sistemul comunică informații utilizatorilor. Momentan, aplicația AndroChef dispune doar de limba engleză;
- În cazul comenzii de grup, linkul trimis către cei invitați la comandă ar putea să fie trimis şi prin email, nu doar prin SMS;

Legat de performanța sistemului, aceasta ar putea fi crescută prin scăderea traficului dintre Client şi Server. Acest lucru ar implica memorarea unor răspunsuri deja primite la accesarea unor pagini, iar la reaccesarea acestora, conținutul lor ar fi extras din memoria dispozitivului.

Interfața utilizator ar putea fi şi ea îmbunătățită, astfel încât să fie mai intuitivă şi mai uşor de utilizat. În cazul funcționalității Comandă de grup dezvoltată, s-ar putea oferi posibilitatea ca fiecare membru al grupului să poată vizualiza ceea ce şi-au ales ceilalți membrii.

În concluzie, aplicația AndroChef dezvoltată poate fi utilizată cu succes, deoarece obiectivele de bază au fost realizate. Dezvoltările ulterioare sunt menite doar ca să ofere utilizatorilor şi alte funcționalități ajutătoare.

Bibliografie

1] [X]CUBE LABS, „Evoluția sistemelor de operare pentru dispozitive mobile," 2011. [Interactiv]. Available: http://www.xcubelabs.com/evolution-of-mobile-operating-systems.php. [Accesat 3 Aprilie 2014].

2] J. Edward, „Statistici de utilizare ale aplicațiilor web și ale aplicațiilor mobile," 7 Aprilie 2014. [Interactiv]. Available: http://www.businessinsider.com/mobile-web-vs-app-usage-statistics-2014-4. [Accesat 13 Aprilie 2014].

3] C. Cernat, „Batalia dintre Android si iOS: Care este pentru mine platforma castigatoare," 2 Aprilie 2014. [Interactiv]. Available: http://www.gazetalibera.ro/android-vs-ios-care-este-pentru-mine-platforma-castigatoare/. [Accesat 16 Aprilie 2014].

4] „Android, pe intelesul tuturor - ce este si ce face?," 20 Martie 2011. [Interactiv]. Available: http://despreandroid.5g.ro/. [Accesat 14 Aprilie 2014].

5] „Android vs. iOS," 15 Septembrie 2013. [Interactiv]. Available: http://www.diffen.com/difference/Android_vs_iOS. [Accesat 17 Aprilie 2014].

6] P. Daniels, „The 22 Best Free Restaurant Apps," 12 Septembrie 2013. [Interactiv]. Available: http://www.thedailymeal.com/22-best-free-restaurant-apps. [Accesat 19 Aprilie 2014].

7] „Frequently asked questions," 2013. [Interactiv]. Available: http://www.urbanspoon.com/faq. [Accesat 20 Aprilie 2014].

8] A. Bujor, „Top 10 cele mai utile aplicații pe mobil, în București," 15 Noiembrie 2013. [Interactiv]. Available: http://www.b365.ro/top-10-cele-mai-utile-aplica-ii-pe-mobil-in-bucure-ti_196741.html. [Accesat 22 Aprilie 2014].

9] „Aplicație web pentru comenzi online," 2013. [Interactiv]. Available: http://www.eatfab.com/.

10] „Client-server model," 22 Martie 2011. [Interactiv]. Available: http://en.wikipedia.org/wiki/Client-server_model. [Accesat 26 Aprilie 2014].

11] O. Pop, „PDFSlide Curs 4-5 Use Cases," UTCN, Calculatoare, An 4 - Cursul Sisteme Informatice, Cluj-Napoca, 2014.

12] A. Gudiu, „Lucrarea 1 - Introducere în limbajul Perl," 12 Martie 2014. [Interactiv]. Available: http://www.b624.net/retele-intranet/laboratoare/lucrarea-1. [Accesat 28 Aprilie 2014].

13] Randal L. Schwartz, brian d foy și Tom Phoenix, Learning Perl, 6th Edition, O'Reilly Media, 2011.

14] A. Sukrieh, „Quick Start," 2009. [Interactiv]. Available: http://www.perldancer.org/quickstart. [Accesat 29 Aprilie 2014].

15] „A Website in a Minute Using Dancer, the Effortless Web Framework," 6 Aprilie 2010. [Interactiv]. Available: http://www.perl.com/pub/2010/04/a-website-in-a-minute-using-dancer-the-effortless-web-framework.html. [Accesat 2 Mai 2014].

16] P. Rabbitson, „DBIx::Class::Manual::Intro," 2013. [Interactiv]. Available: http://search.cpan.org/~ribasushi/DBIx-Class-0.08270/lib/DBIx/Class/Manual/Intro.pod. [Accesat 4 Mai 2014].

17] „Protocolul de comunicare HTTP," 14 Aprilie 2013. [Interactiv]. Available: http://ro.wikipedia.org/wiki/HTTP. [Accesat 4 Mai 2014].

18] Fielding, Roy Thomas, *Architectural Styles and the Design of Network-based Software Architectures*, Irvine: University of California, 2000.

19] J. Cox, „SOAP vs. REST For Mobile Services," 17 Septembrie 2011. [Interactiv]. Available: http://captechconsulting.com/blog/jack-cox/soap-vs-rest-mobile-services. [Accesat 6 Mai 2014].

20] M. Arora, „Deciphering phone and embedded security - Part 1: Fundamentals of the Android architecture and terminologies," 11 Iunie 2012. [Interactiv]. Available: http://www.eetimes.com/document.asp?doc_id=1279698. [Accesat 7 Mai 2014].

21] J. Fingas, „Jelly Bean claims 59.1 percent of Android device share as KitKat inches forward," 11 Ianuarie 2014. [Interactiv]. Available: http://www.engadget.com/2014/01/11/android-device-share-january-2014/. [Accesat 9 Mai 2014].

22] Karina Sokolova și Marc Lemercier, „Android Passive MVC: a Novel Architecture Model for Android Application Development," în *The Fifth International Conferences on Pervasive Patterns and Applications*, France, 2013.

Anexa 1 – Lista de figuri

FIGURA 3.1 MOBILE VS DESKTOP USERS 8
FIGURA 3.2 MOBILE WEB VS APPS 9
FIGURA 3.3 INTERFAȚA APLICAȚIEI URBANSPOON 13
FIGURA 3.4 INTERFAȚA APLICAȚIEI FOODPANDA 15
FIGURA 3.5 INTERFAȚA APLICAȚIEI EATFAB 17
FIGURA 4.1 ARHITECTURA CONCEPTUALĂ A APLICAȚIEI 20
FIGURA 4.2 ARHITECTURA CLIENT-SERVER 21
FIGURA 4.3 DIAGRAMA CAZURILOR DE UTILIZARE 26
FIGURA 4.4 BAZA DE DATE – ORM - APLICAȚIE 37
FIGURA 4.5 ARHITECTURA ANDROID 40
FIGURA 4.6 VERSIUNILE PLATFORMEI ANDROID 41
FIGURA 4.7 ANDROID PASSIVE MVC 42
FIGURA 5.1 SCHEMA GENERALĂ A SISTEMULUI 43
FIGURA 5.2 DIAGRAMA DE DEPLOYMENT 44
FIGURA 5.3 DIAGRAMA DE MODULE PENTRU APLICAȚIA SERVER 45
FIGURA 5.4 DIAGRAMA DE PACHETE A APLICAȚIEI CLIENT 52
FIGURA 5.5 DIAGRAMA DE ACTIVITĂȚI 53
FIGURA 5.6 MAPARE RESTAURANTSLISTACTIVITY - ITEMRESTAURANTADAPTER - RESTAURANT 54
FIGURA 5.7 CLASA CARTORDER ȘI CLASELE RELAȚIONATE 55
FIGURA 5.8 EXEMPLU DE SUBCLASĂ HTTPASYNCTASK 57
FIGURA 5.9 INVITAREA PERSOANELOR LA COMANDA DE GRUP 59
FIGURA 5.10 ACCEPTAREA INVITAȚIILOR 60
FIGURA 5.11 PLASAREA COMENZII DE GRUP 61
FIGURA 5.12 DIAGRAMA BAZEI DE DATE 62
FIGURA 6.1 CAZ DE TEST PENTRU COMANDA INDIVIDUALĂ 66
FIGURA 7.1 PAGINA DE LOGIN ȘI REGISTER 69
FIGURA 7.2 PAGINA FORGOT PASSWORD 69
FIGURA 7.3 LISTA DE RESTAURANTE ȘI CATEGORII 70

FIGURA 7.4 LISTA DE PRODUSE ȘI DETALIILE UNUI PRODUS SELECTAT..... 71
FIGURA 7.5 SELECTAREA DETALIILOR SUPLIMENTARE ȘI COȘUL DE PRODUSE 72
FIGURA 7.6 MESAJUL DE INVITARE LA COMANDA DE GRUP..... 72
FIGURA 7.7 SELECTAREA TIPULUI DE LIVRARE ȘI A ADRESEI..... 73

Anexa 2 – Lista de tabele

TABEL 3.1 ANDROID VS IOS 10
TABEL 3.2 ANDROCHEF VS URBANSPOON 13
TABEL 3.3 ANDROCHEF VS FOODPANDA 16
TABEL 3.4 ANDROCHEF VS EATFAB 18
TABEL 4.1 COMANDĂ SIMPLĂ ŞI COMANDĂ DE GRUP 22
TABEL 4.2 SUMAR AL CERINȚELOR FUNCȚIONALE DE BAZĂ 23
TABEL 4.3 SUMAR AL CERINȚELOR NON-FUNCȚIONALE 25

Anexa 3 – Glosar de termeni

ADT - Android Development Tool
API - Application Programming Interface
CGI - Common Gateway Interface
CMS - Content Management System
CORBA - Common Object Request Broker Architecture
CPAN - The Comprehensive Perl Archive Network
GPS - Global Positioning System
GUI - Graphical User Interface
HTTP - Hypertext Transfer Protocol
JSON - JavaScript Object Notation
LAMP - Linux, Apache, MySQL, PHP
MD5 - Message Digest Algorithm 5
ORM - Object-relational mapping
OSI - Open Systems Interconnection
Perl - Practical Extraction and Report Language
PHP - Php: Hypertext Preprocessor
PSGI - Perl Web Server Gateway Interface
REST - Representational State Transfer
RPC - Remote Procedure Call
SDK - Software Development Kit
SGBD - Sistem de Gestiune a Bazelor de Date
SMTP - Simple Mail Transfer Protocol
SOAP - Simple Object Access Protocol
TCP - Transmission Control Protocol
UML - Unified Modeling Language
URL - Uniform Resource Locator
WSDL - Web Services Description Language
WWW - World Wide Web
XML - Extensible Markup Language
XSD - XML Schema Definition