LATVIJAS UNIVERSITĀTE
DATORIKAS FAKULTĀTE

Android lietotne restorāniem

KVALIFIKĀCIJAS DARBS

Autors: **Mārtiņš Andersons**
Studenta apliecības Nr.: ma12027
Darba vadītājs: asoc. profesors Dr. dat. Uldis Straujums

RĪGA 2014

ANOTĀCIJA

Mārtiņa Andersona kvalifikācijas darbā "Android lietotne restorāniem" ir izstrādāta lietotņu kopa MArestorans. Lietotņu kopa sastāv no trīs lietotnēm, kuras lietos restorāna klients, restorāna viesmīlis un restorāna pavārs. Lietotņu mērķis - atvieglot pasūtījumu veikšanu. Klienta lietotnē ir ēdienu izvēles, ēdienu pasūtīšanas, viesmīļa izsaukšanas, rēķina apskates un servisa novērtēšanas opcijas. Viesmīļa lietotnē ir pasūtījumu apstiprināšanas, ēdienu pievienošanas un noņemšanas, pagatavoto ēdienu apstiprināšanas, apmaksāto rēķinu apstiprināšanas un galdiņu saraksta, kuri izsaukuši viesmīli, apskates opcijas. Pavāra lietotnē ir iespēja norādīt, ka kāds no apstiprinātajiem ēdieniem ir pagatavots.

Visas lietotnes ir izstrādātas Android programmēšanas valodā, izmantojot Parse aizmugursistēmu.

Atslēgas vārdi: Android, Agile, Restorānu pārvaldība, Parse API, MArestorans

ABSTRACT

MArestorans is an application set, which has been developed by Mārtiņš Andersons as a final project "Android application software for restaurants". The application set consists of three applications, which is for a client, a restaurant waiter and a restaurant's cook. Applications target is to ease order placing. Client's application has menu, food ordering, waiter call, a view of the bill and service rating features. Waiter's application have features: to validate the order, to add or remove the food, to validate prepared food, to validate paid bill, table list, which have called for the waiter. Waiter's application has a feature to indicate food, which has been prepared.

The applications are developed in Android programming language, using Parse back end.

Keywords: Android, Agile, Restaurant management, Parse API, MArestorans

SATURS

1. IEVADS

1.1. Nolūks

Programmatūras prasību specifikācijas nolūks ir aprakstīt Android visu trīs lietotņu prasības. Šis dokuments ir paredzēts moduļa projektētājam un pasūtītājam produkta funkcionalitātes prasību saskaņošanai.

1.2. Darbības sfēra

Šajā dokumentā ir definētas prasības restorāna Android lietotnēm. Programmatūras produkta mērķis ir veikt restorāna pasūtījumu izveidi, apstrādi. Produkta paredzētais pielietojums ir visu trīs lietotņu pieejamība Google Play veikalā - klienta lietotne pieejama pilnīgi visiem, viesmīļa un pavāra tikai ar iepriekš noteiktām atļaujām.

1.3. Definīcijas, akronīmi un saīsinājumi

NFC - Near Field Communication – ir standartu kopa telefoniem un līdzīgām ierīcēm, kas nodrošina īsā rādiusā bezvadu savienojumu.

Google Play – interneta veikals Android lietotnēm.

Android – mobīlo ierīču operētājsistēma.

Eclipse – izstrādes vide.

Parse – bezmaksas aizmugursistēma.

ORMLite – bibliotēka, kas nodrošina datu bāzes tabulu veidošanu un saglabāšu telefona atmiņā.

Universal Image Loader – bibliotēka, kas nodrošina bilžu ielādi, apstrādi un saglabāšanu telefona atmiņā.

SVN – Subversion – versiju kontroles rīks.

JavaDoc – dokumentācijas lauku automātiskais izveidotājs Java programmēšanas valodai.

2. VISPĀRĒJAIS APRAKSTS

2.1. Produkta perspektīva

Izstrādātās trīs lietotnes ir komplekts, kuru var piedāvāt ikvienam restorānam, veicot nelielas izmaiņas lietotņu dizainā un specifikācijā.

2.2. Produkta funkcijas

Tālāk aprakstīts kādas funkcijas jāpilda katrai lietotnei.

2.2.1. Klienta lietotne

Programmatūrai jāattēlo ēdienkarte, jādod iespēja klientam pievienot ēdienus pasūtījumam un veikt pasūtījumu.

2.2.2. Viesmīļa lietotne

Programmatūrai jāattēlo pasūtījumi, jādod iespēja pievienot un noņemt ēdienus pasūtījumam, kā arī apstiprināt pasūtījumu.

2.2.3. Pavāra lietotne

Programmatūrai jāattēlo saraksts ar apstiprināto pasūtījumu ēdieniem.

2.3. Lietotāja raksturiezīmes

Restorāna apmeklētāji, viesmīļi un pavāri. Lietotājam nav jābūt ar īpašām priekšzināšanām, vienīgi ar Android ierīci, kuras versija ir sākot no 2.3.3. vai augstāka.

2.4. Vispārējie ierobežojumi

Viesmīļu un pavāru epastiem jābūt pievienotiem Google Groups restorāna grupā, lai būtu iespējams lejupielādēt lietotnes no Google Play veikala.

2.5. Pieņēmumi un atkarības

Tiek pieņemts, ka ēdienu skaits dienas piedāvājumos nepārsniegs 20 ēdienu.

3. LIETOTĀJA STĀSTI

Projekta sākumā pasūtītājs precīzi nenoteica lietotņu prasības un funkcijas, līdz ar to projekts bija jāizstrādā laika gaitā, pamatojoties uz lietotāju stāstiem, kuros tika skaidrāk definēts, ko lietotājs sagaida no programmatūras.

„Lietotāju stāstos" izstrādātājs noskaidroja, ko vēlas un sagaida lietotājs, kā arī, kas ir primārais, sekundārais utt. priekš lietotāja, piemēram, primārais ir fona krāsa vai fonta izmērs.

„Lietotāja stāsts" ir veidots tabulas veidā, kurā ir attēloti visi lietotāja stāsti noteiktā secībā – pamatojoties uz to prioritātes pakāpi, kā arī novērtēts, kāda ir to izstrādes sarežģītība punktos. Punkti tika noteikti, pamatojoties uz izstrādātāja pieredzi un zināšanām, līdz ar to visvairāk punktu ir tiem stāstiem, ar kuriem izstrādātājs iepriekš nebija saskāries vai nebija pietiekami daudz zināšanu.

3.1. Lietotāju stāsti sadalīti trijās daļās

Katrai lietotnei ir savi stāsti, kuros programmatūras topošais lietotājs izskaidroja savas prasības.

Klienta lietotnes lietotājstāsti ir attēloti tabulā 3.1.1.

3.1.1. tabula

Klienta lietotnes stāsti

Lietotāja stāsts	Prioritāte	Sarežģītības punkti
1. Kā lietotājs, es vēlos redzēt restorāna ēdienkarti.	1	21
2. Kā lietotājs, es vēlos iespēju pasūtīt ēdienus.	2	8
3. Kā lietotājs, es vēlos iespēju izsaukt viesmīli.	2	5
4. Kā lietotājs, es vēlos, lai dienas piedāvājumi ir izcelti no ēdienkartes.	3	13
5. Kā lietotājs, es vēlos iespēju redzēt savu rēķinu.	3	5
6. Kā lietotājs, es vēlos novērtēt servisu.	4	2

Viesmīļa lietotnes lietotājstāsti ir attēloti tabulā 3.1.2.

3.1.2. tabula

Viesmīļa lietotnes stāsti

Lietotāja stāsts	Prioritāte	Sarežģītības punkti
1. Kā lietotājs, es vēlos redzēt un apstiprināt pasūtītos ēdienus.	1	8
2. Kā lietotājs, es vēlos redzēt pagatavotos ēdienus.	2	3
3. Kā lietotājs, es vēlos redzēt katra galdiņa rēķinu.	2	3
4. Kā lietotājs, es vēlos redzēt galdiņus, kuri izsaukuši viesmīli.	3	5
5. Kā lietotājs, es vēlos pievienot ēdienus pasūtījumam.	4	8
6. Kā lietotājs, es vēlos noņemt ēdienus no rēķina.	5	5

Pavāra lietotnes lietotājstāsti ir attēloti tabulā 3.1.3.

3.1.3. tabula

Pavāra lietotnes stāsti

Lietotāja stāsts	Prioritāte	Sarežģītības punkti
1. Kā lietotājs, es vēlos redzēt apstiprinātos ēdienus.	1	8
2. Kā lietotājs, es vēlos, lai ēdieni tiktu sakārtoti augošā secībā atkarībā no laika.	2	5

3.2. Lietotāju stāstu sadalījums pa iterācijām

Lietotājstāstu sadalījumi pa iterācijām attēloti tabulās. Katra tabula sastāv no mērķa un uzdevuma. Mērķis attēlo klienta prasības, savukārt uzdevuma daļa attēlo, kā lietotnes izstrādātājs izpildīs, ar kādu metodi.

Pirmās iterācijas lietotājstāsts ir attēlots tabulā 3.2.1.

3.2.1. tabula

Pirmā iterācija

Lietotāja stāsts	Uzdevums
1. Kā lietotājs, es vēlos redzēt restorāna ēdienkarti.	Izveidot servisā tabulu, kurā glabāsies ēdienkartes dati.
	Izveidot aktivitāti, kura ielādēs ēdienkartes datus no servisa.
	Izveidot skatu un objektus, kuros tik ielādēti ēdienkartes dati.
	Izveidot datu bāzē tabulu, kurā tiks saglabāti ēdienkartes dati ielādes laikā.
	Izveidot viena ēdiena skatu, kur attēlo arī ēdiena bildi un sastāvdaļas.

Otrās iterācijas lietotājstāsti ir attēloti tabulā 3.2.2.

3.2.2. tabula

Otrā iterācija

Lietotāja stāsts	Uzdevums
2. Kā lietotājs, es vēlos iespēju pasūtīt ēdienus.	Izveidot servisā tabulu, kurā glabāsies pasūtījuma dati.
	Izveidot pasūtīšanas iespēju ēdienkartes un viena ēdiena skatā.
	Izveidot datu bāzē tabulu, kurā glabāsies visi pasūtījumam pievienotie ēdieni.
	Izveidot aktivitāti, kura attēlos sarakstu ar pasūtījumam

		pievienotajiem ēdieniem un dos iespēju veikt pasūtījumu.
		Izveidot NFC mikroshēmas datu ielasītāju.
3.	**Kā lietotājs, es vēlos iespēju izsaukt viesmīli.**	Izveidot servisā tabulu, kurā glabāsies viesmīļa izsaukuma dati.
		Izveidot skatu, kurš dos iespēju izsaukt viesmīli.
4.	**Kā lietotājs, es vēlos, lai dienas piedāvājumi ir izcelti no ēdienkartes.**	Izveidot servisā tabulu, kurā glabāsies dienas piedāvājumu dati.
		Izveidot aktivitāti, kura ielādēs dienas piedāvājumu datus no servisa.

Trešās iterācijas lietotājstāsti ir attēloti tabulā 3.2.3.

3.2.3. tabula

Trešā iterācija

Lietotāja stāsts		Uzdevums
4.	**Kā lietotājs, es vēlos, lai dienas piedāvājumi ir izcelti no ēdienkartes.**	Izveidot skatu un objektus, kuros tiks ielādēti dienas piedāvājuma dati.
		Izveidot datu bāzē tabulu, kurā tiks saglabāti dienas piedāvājuma dati ielādes laikā.
		Pārveidot viena ēdiena skatu, lai tajā attēlo arī cenu iepriekš.
5.	**Kā lietotājs, es vēlos iespēju redzēt savu rēķinu.**	Izveidot aktivitāti, kura ielādēs rēķina datus no servisa.
		Izveidot skatu un objektus, kuros tik ielādēti rēķina dati.
6.	**Kā lietotājs, es vēlos novērtēt servisu.**	Izveidot servisā tabulu, kurā glabāsies novērtējuma dati.
		Izveidot skatu, kurš dos iespēju novērtēt servisu.

1. Kā lietotājs, es vēlos redzēt un apstiprināt pasūtītos ēdienus.	Izveidot aktivitāti, kura ielādēs tikai tos pasūtījuma datus no servisa, kuriem tips būs 0.
	Izveidot skatu un objektus, kuros tik ielādēti pasūtījuma dati.
	Izveidot viena pasūtījuma skatu, kur attēlos iepriekš izvēlētā galdiņa sarakstu ar pasūtītajiem ēdieniem.
	Izveidot iespēju apstiprināt pasūtījumu.

Ceturtās iterācijas lietotājstāsti ir attēloti tabulā 3.2.4.

3.2.4. tabula

Ceturtā iterācija

Lietotāja stāsts	Uzdevums
2. Kā lietotājs, es vēlos redzēt pagatavotos ēdienus.	Izveidot aktivitāti, kura ielādēs tikai tospasūtījuma datus no servisa, kuriem tips būs 2.
	Izveidot skatu un objektus, kuros tik ielādēti pagatavoto ēdienu dati.
3. Kā lietotājs, es vēlos redzēt katra galdiņa rēķinu.	Izveidot aktivitāti, kura ielādēs visus pasūtījuma datus no servisa.
	Izveidot skatu un objektus, kuros tik ielādēti visu pasūtījumu dati.
	Viena pasūtījuma skatā izveidot papildus informāciju pie katra ēdiena par tā stāvokli (neapstiprināts, pagatavots).
4. Kā lietotājs, es vēlos redzēt galdiņus, kuri izsaukuši viesmīli.	Izveidot aktivitāti, kura ielādēs visus viesmīļa izsaukumu datus no servisa.
	Izveidot skatu un objektus, kuros tik ielādēti visu viesmīļa izsaukumu dati.
	Izveidot iespēju noņemt viesmīļa izsaukumu.

5.	**Kā lietotājs, es vēlos pievienot ēdienus pasūtījumam.**	Izveidot aktivitāti, kura ielādēs ēdienkartes un dienas piedāvājuma datus no servisa.
		Izveidot skatu un objektus, kuros tik ielādēti ēdienkartes un dienas piedāvājumu dati.
		Izveidot datu bāzē tabulu, kurā glabāsies visi ēdienkartes un dienas piedāvājumu dati.
		Izveidot iespēju viena pasūtījuma skatā atvērt ēdienkartes skatu un pievienot ēdienus pasūtījumam.

Piektās iterācijas lietotājstāsti ir attēloti tabulā 3.2.5.

3.2.5. tabula

Piektā iterācija

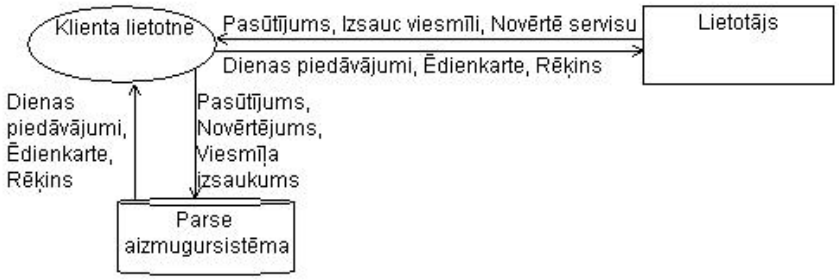
	Lietotāja stāsts	Uzdevums
6.	**Kā lietotājs, es vēlos noņemt ēdienus no rēķina.**	Izveidot iespēju, ka ēdienu var noņemt gan pasūtījuma, gan pagatavoto, gan rēķina skatā.
1.	**Kā lietotājs, es vēlos redzēt apstiprinātos ēdienus.**	Izveidot aktivitāti, kura ielādēs tikai tospasūtījuma datus no servisa, kuriem tips būs 1.
		Izveidot skatu un objektus, kuros tik ielādēti apstiprināto ēdienu dati.
		Izveidot iespēju apstiprināt, ka ēdiens ir pagatavots.
		Izveidot iespēju, lai saraksts ar ēdieniem tiek atjaunots ik pēc 10 sekundēm.
2.	**Kā lietotājs, es vēlos, lai ēdieni tiktu sakārtoti augošā secībā atkārībā no laika.**	Izveidot skatā papildus lauku, kurā attēlo laiku, cik pagājis, kopš ēdiens tika apstiprināts.
		Izveidot, lai visi dati tiek sakārtoti augošā secībā pēc laika.

4. PROGRAMMATŪRAS PROJEKTĒJUMA APRAKSTS

Programmas projektējuma apraksta nolūks ir aprakstīt, kā lietotājstātu nodaļā minētās prasības tika realizētas programmprodukta izstrādē. Projekta sākumā nebija zināmi visi lietotājstāsti, līdz ar to arī visu lietotņu prasības, tāpēc nebija iespējams izstrādāt datu plūsmu diagramas. Katru reizi pēc jaunu lietotājstāstu iegūšanas un to prasību noprecizēšanas, programmatūras projektējuma apraksts tika papildināts ar jaunu moduļu projektējumiem.

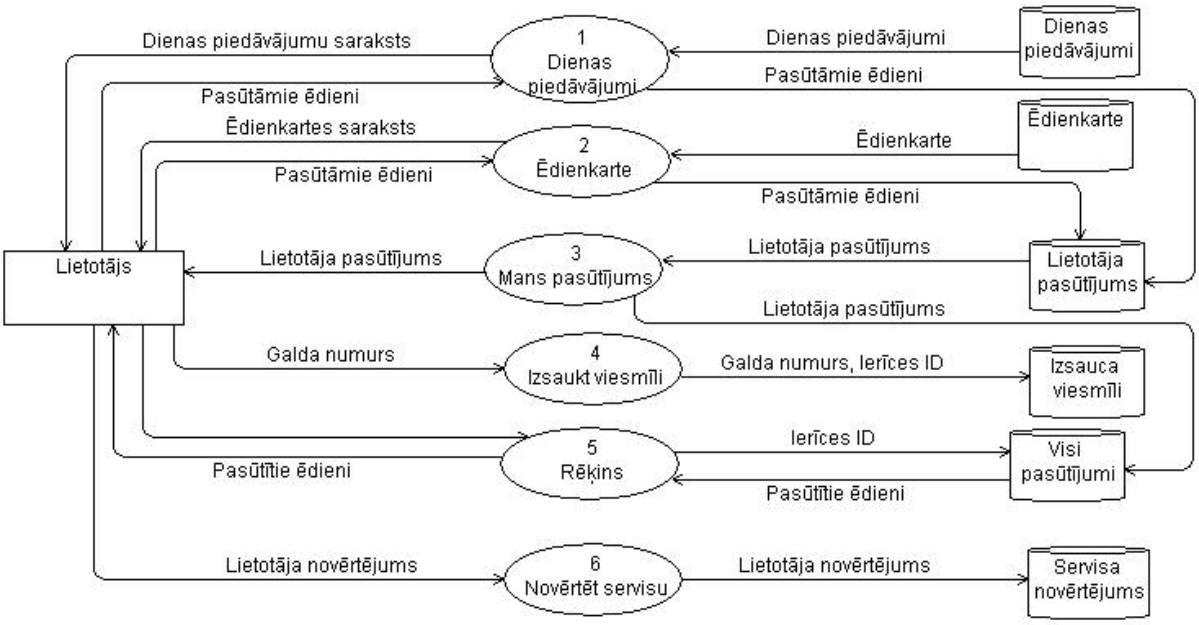
4.1. Programmatūras datu plūsmu diagrammas

Klienta lietotnes 0. līmeņa datu plūsmu diagramma redzama attēlā 4.1.1.



4.1.1. att. **Klienta lietotnes 0. līmeņa datu plūsmu diagramma**

Klienta lietotnes 1. līmeņa datu plūsmu diagramma redzama attēlā 4.1.2.



4.1.2. att. **Klienta lietotnes 1. līmeņa datu plūsmu diagramma**

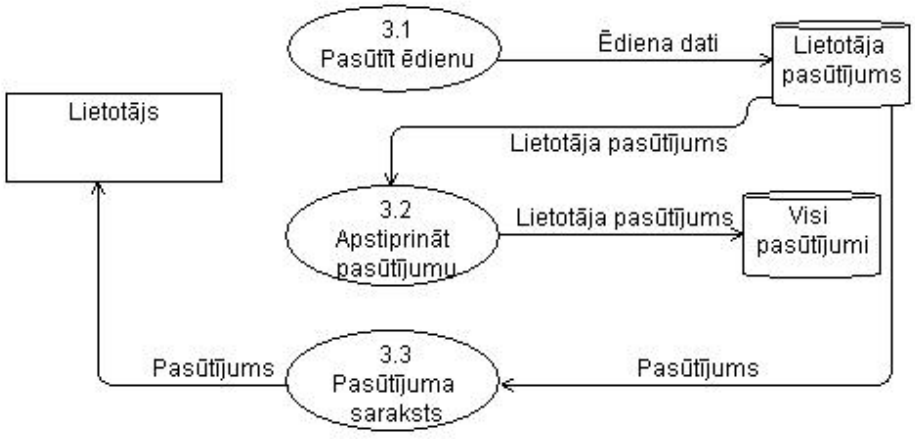
Klienta lietotnes 2. līmeņa „Dienas piedāvājumi” datu plūsmu diagramma redzama attēlā 4.1.3.

4.1.3. att. **Klienta lietotnes 2. līmeņa „Dienas piedāvājumi” datu plūsmu diagramma**

Klienta lietotnes 2. līmeņa „Ēdienkarte” datu plūsmu diagramma redzama attēlā 4.1.4.

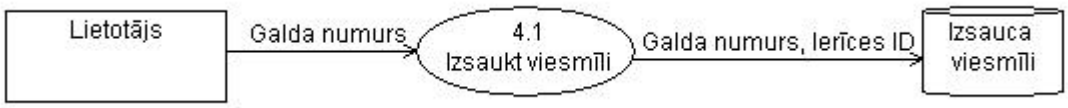
4.1.4. att. **Klienta lietotnes 2. līmeņa „Ēdienkarte” datu plūsmu diagramma**

Klienta lietotnes 2. līmeņa „Mans pasūtījums” datu plūsmu diagramma redzama attēlā 4.1.5.



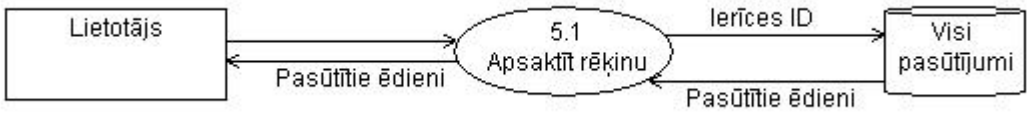
4.1.5. att. **Klienta lietotnes 2. līmeņa „Mans pasūtījums” datu plūsmu diagramma**

Klienta lietotnes 2. līmeņa „Izsaukt viesmīli” datu plūsmu diagramma redzama attēlā 4.1.6.



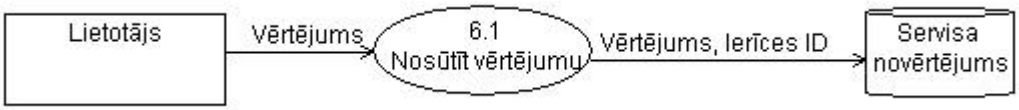
4.1.6. att. **Klienta lietotnes 2. līmeņa „Izsaukt viesmīli” datu plūsmu diagramma**

Klienta lietotnes 2. līmeņa „Rēķins” datu plūsmu diagramma redzama attēlā 4.1.7.



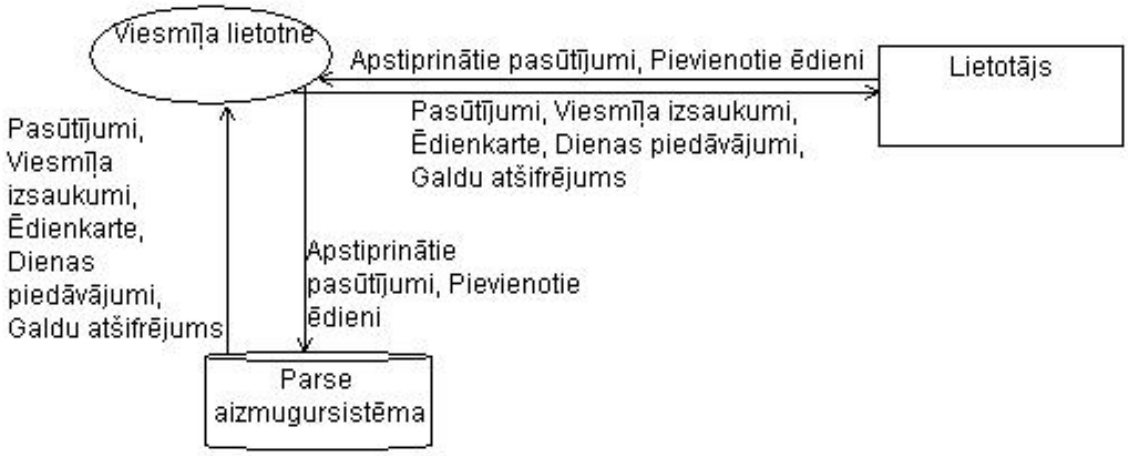
4.1.7. att. **Klienta lietotnes 2. līmeņa „Rēķins” datu plūsmu diagramma**

Klienta lietotnes 2. līmeņa „Novērtēt servisu” datu plūsmu diagramma redzama attēlā 4.1.8.



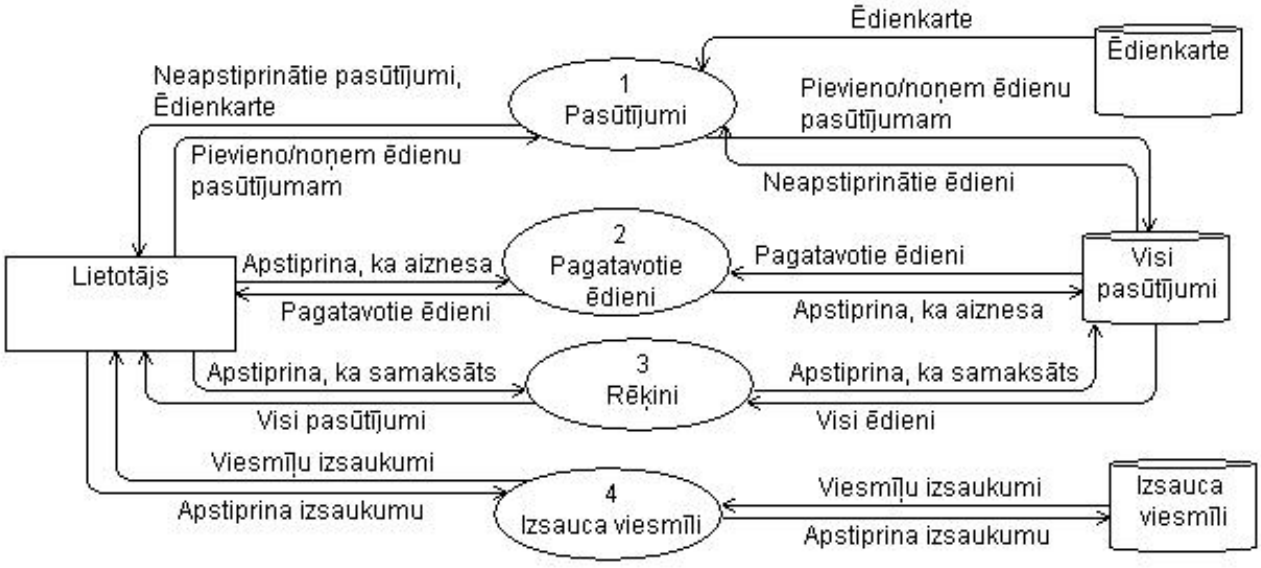
4.1.8. att. **Klienta lietotnes 2. līmeņa „Novērtēt servisu” datu plūsmu diagramma**

Viesmīļa lietotnes 0. līmeņa datu plūsmu diagramma redzama attēlā 4.1.9.



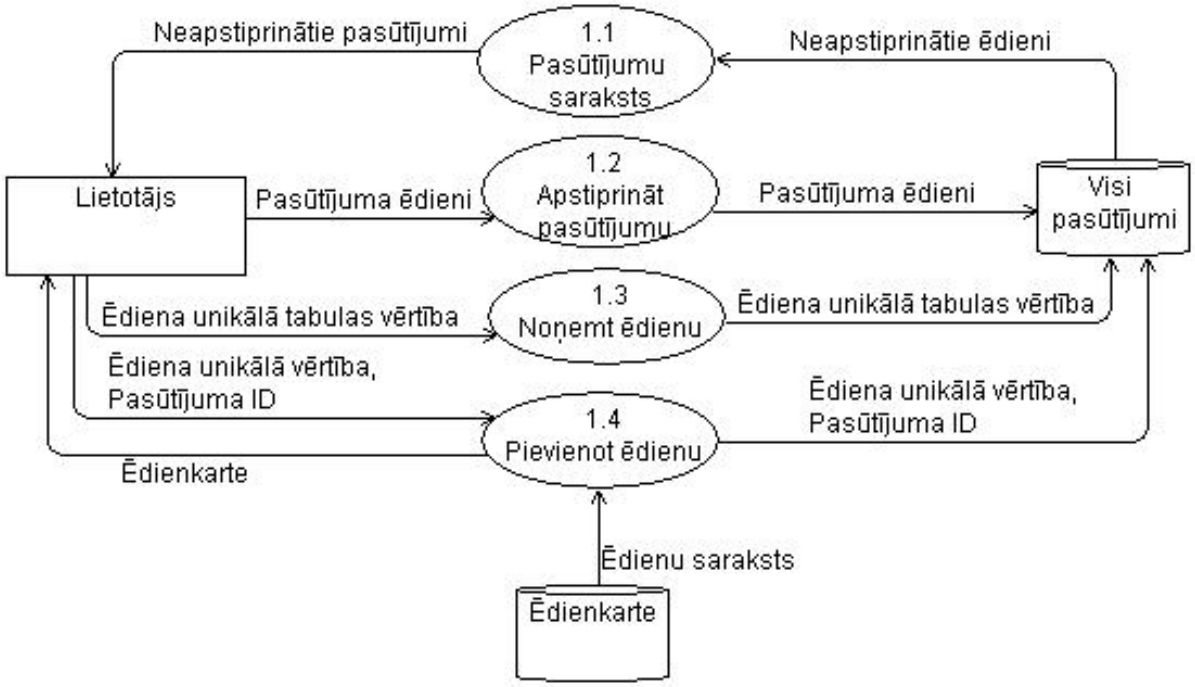
4.1.9. att. **Viesmīļa lietotnes 0. līmeņa datu plūsmu diagramma**

Viesmīļa lietotnes 1. līmeņa datu plūsmu diagramma redzama attēlā 4.1.10.



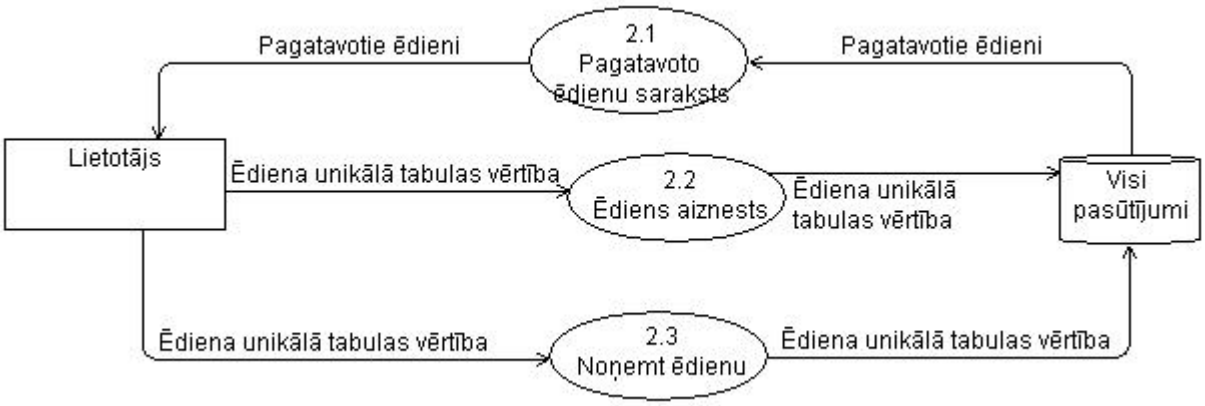
4.1.10. att. **Viesmīļa lietotnes 1. līmeņa datu plūsmu diagramma**

Viesmīļa lietotnes 2. līmeņa „Pasūtījumi" datu plūsmu diagramma redzama attēlā 4.1.11.



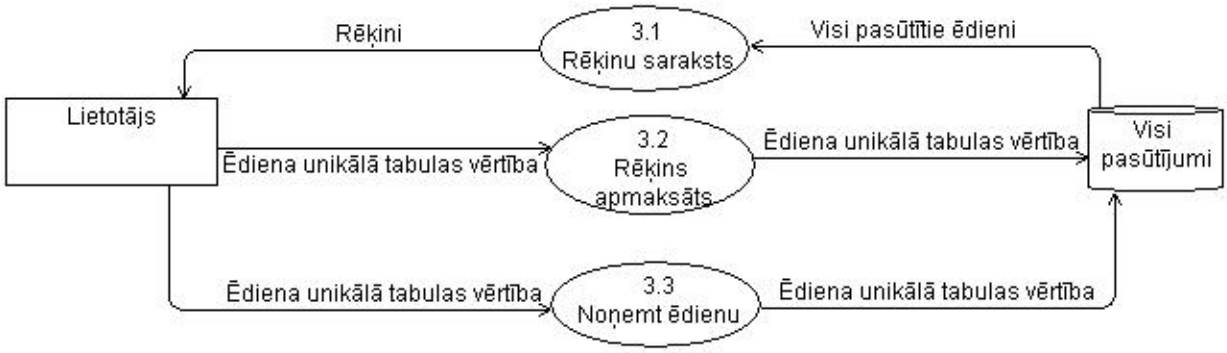
4.1.11. att. **Viesmīļa lietotnes 2. līmeņa „Pasūtījumi" datu plūsmu diagramma**

Viesmīļa lietotnes 2. līmeņa „Pagatavotie ēdieni” datu plūsmu diagramma redzama attēlā 4.1.12.



4.1.12. att. **Viesmīļa lietotnes 2. līmeņa „Pagatavotie ēdieni” datu plūsmu diagramma**

Viesmīļa lietotnes 2. līmeņa „Rēķini” datu plūsmu diagramma redzama attēlā 4.1.13.

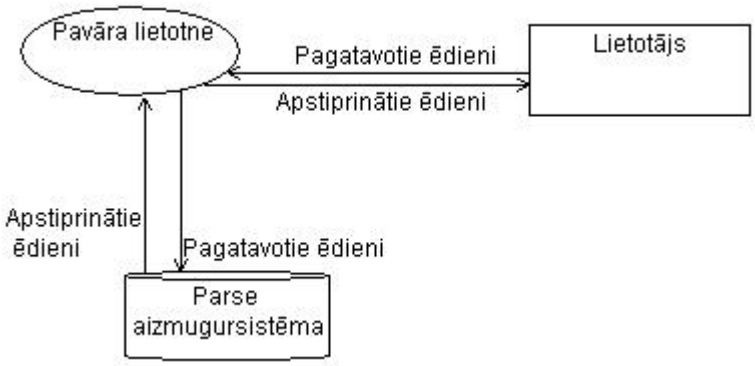


4.1.13. att. **Viesmīļa lietotnes 2. līmeņa „Rēķini” datu plūsmu diagramma**

Viesmīļa lietotnes 2. līmeņa „Izsauca viesmīli” datu plūsmu diagramma redzama attēlā 4.1.14.

4.1.14. att. **Viesmīļa lietotnes 2. līmeņa „Izsauca viesmīli” datu plūsmu diagramma**

Pavāra lietotnes 0. līmeņa datu plūsmu diagramma redzama attēlā 4.1.15.



4.1.15. att. **Pavāra lietotnes 0. līmeņa datu plūsmu diagramma**

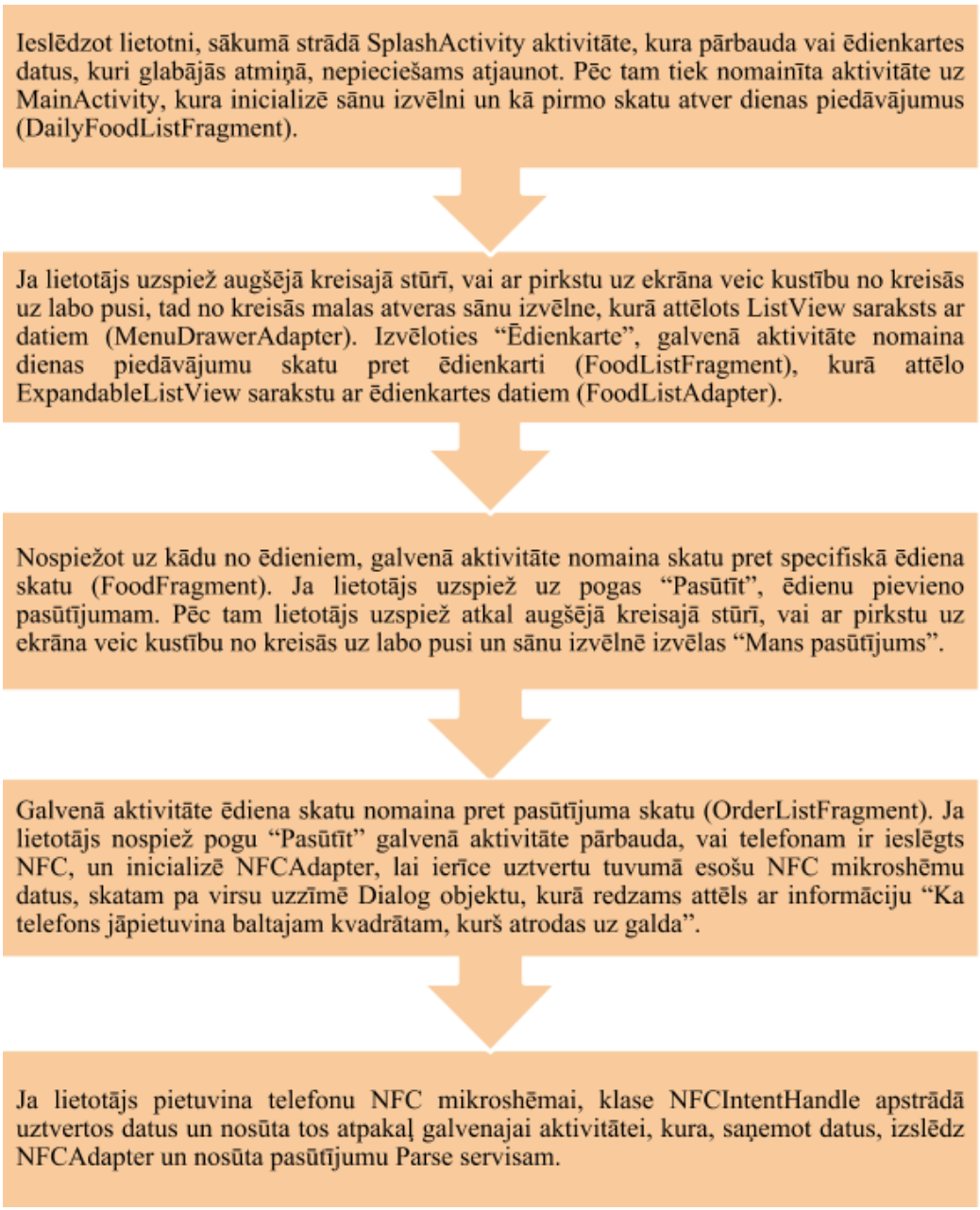
Pavāra lietotnes 1. līmeņa datu plūsmu diagramma redzama attēlā 4.1.16.

4.1.16. att. **Pavāra lietotnes 1. līmeņa datu plūsmu diagramma**

Datu plūsmu diagrammas tika izveidotas, kad bija zināmas visas lietotājstāstu prasības. Veidošanā tika izmantots SimpleMod (9. avots) rīks.

4.2. Funkcionalitātes piemērs

Lai izprastu lietotnes darbību, tika izveidots un apskatīts detalizēts un grafisks funkcionalitātes piemērs (*4.2.1. attēls*), kurā ir apskatīta reāla lietotāja darbība, kad lietotājs veic pasūtījumu.



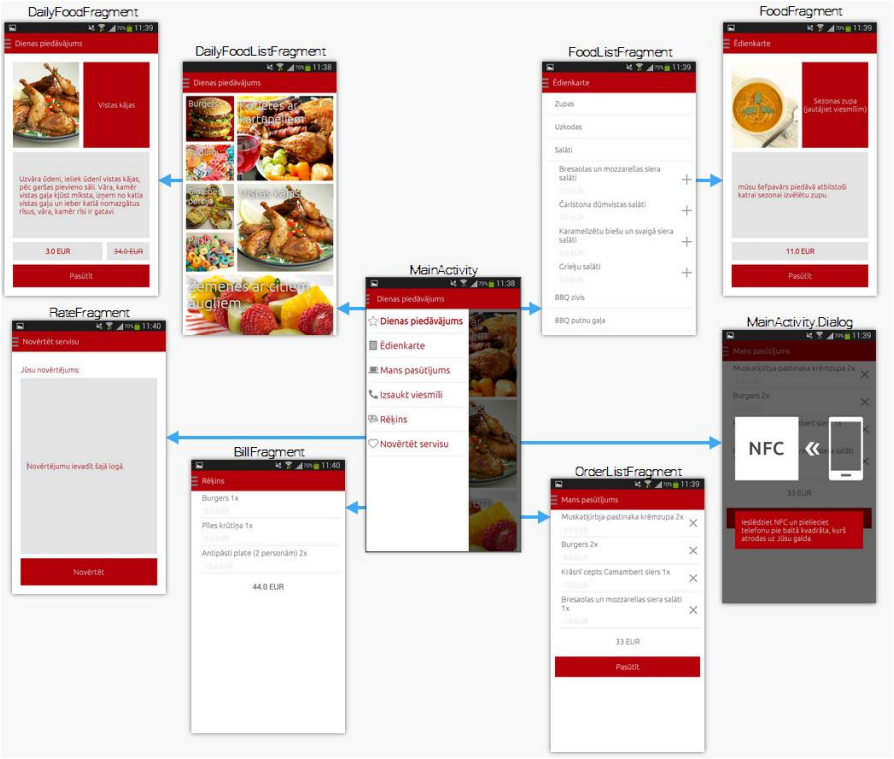
4.2.1. *att.* Funkcionalitātes piemērs pasūtījuma veikšanai

Funkcionalitātes piemērā ir iekļauti trīs lietotājstāsti, ieslēdzot lietotni, sākumā ir dienas piedāvājumi, tālāk lietotājs var apskatīt ēdienkarti un pievienot ēdienus pasūtījumam, veikt pasūtījumu.

4.3. Lietotņu saskarņu diagrammas

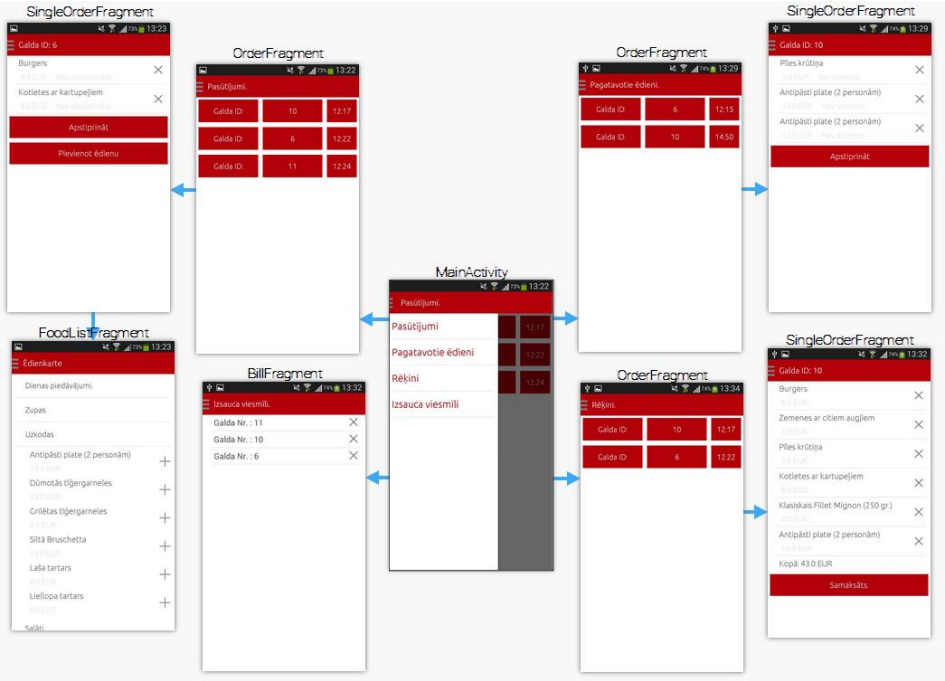
Diagrammās ir attēlots, kā katra lietotne strādā noteiktu soļu gaitā. Tā kā klienta un pavāra ir visietilpīgākās, tad abām attēlots, kā savstarpēji lietotnes skati komunicē, kādas opcijas ir iespējamas.

Klienta lietotnes saskarnes diagramma redzama attēlā 4.3.1.



4.3.1. att. **Klienta lietotnes saskarnes diagramma.**

Viesmīļa lietotnes saskarnes diagramma redzama attēlā 4.3.2.



4.3.2. att. **Viesmīļa lietotnes saskarnes diagramma.**

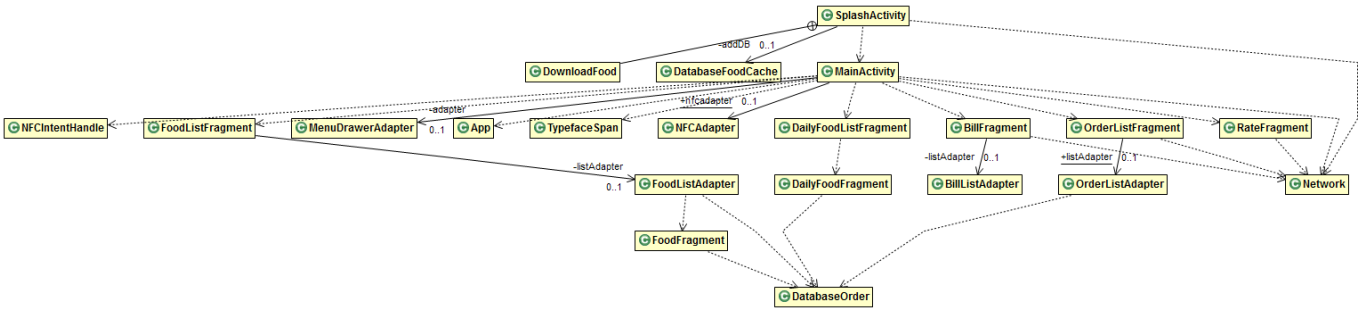
Pavāra lietotnes saskarnes diagramma redzama attēlā 4.3.3.

4.3.2. att. **Pavāra lietotnes saskarnes diagramma.**

4.4. Klašu hierarhijas diagramma

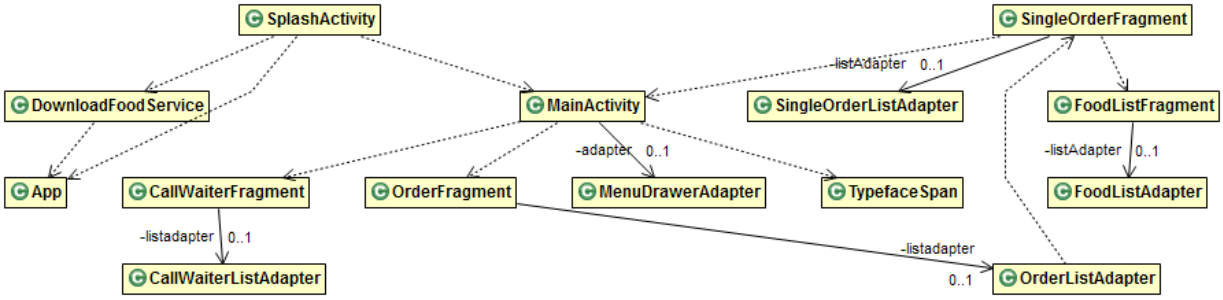
Lai izveidotu klašu hierarhijas diagrammas, Eclipse izstrādes vidē tika izmantots ObjectAid spraudnis (12. avots).

Klienta lietotnes klašu hierarhijas diagramma redzama attēlā 4.4.1.



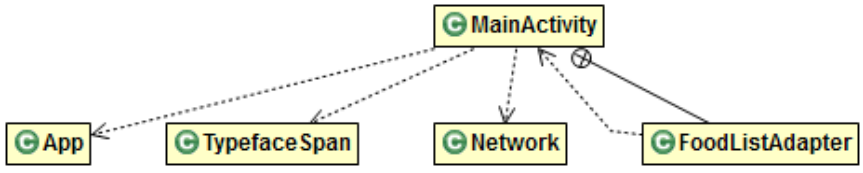
4.4.1. att. **Klienta lietotnes klašu hierarhijas diagramma.**

Viesmīļa lietotnes klašu hierarhijas diagramma redzama attēlā 4.4.2.



4.4.2. att. **Viesmīļa lietotnes klašu hierarhijas diagramma.**

Pavāra lietotnes klašu hierarhijas diagramma redzama attēlā 4.4.3.



4.4.3. att. **Pavāra lietotnes klašu hierarhijas diagramma.**

Klašu hierarhijas diagrammā nav attēlotas tādas klases kā C, kur nodefinētas visas konstantās vērtības, DatabaseHelper, kura dod pieeju datu bāžu tabulām, un klases, kurās nodefinētas datu bāzes tabulas un to lauki.

4.5. Klienta lietotnes moduļu projektējums

4.5.1. Klase SetTitleCallback

Klase, kura ir atbildīga par lietotnes virsraksta nomaiņu. To implementē galvenā aktivitāte un izsauc visi fragmentu skati.

Metodes:

Tips	Metode	Apraksts
void	setTitle(CharSequence title)	Nomaina aktivitātes virsraksta tekstu.

4.5.2. Klase FoodListFragment

Klase izveido ēdienkartes skatu, kur attēlo ExpandableListView sarakstu ar ēdienu kategorijām un ēdieniem zem tām, dod iespēju pievienot kādu no ēdieniem pasūtījumam. Ja lietotājs uzspiež uz kādu no ēdieniem, klase izsauc FoodFragment klasi, kura attēlo tieši specifisko ēdienu.

Metodes:

Tips	Metode	Apraksts
void	onAttach(Activity activity)	Kad fragments pievienots aktivitātei, tad inicializē mainīgo callback.
void	onDetach()	Kad fragments tiek atvienots, tad iztukšo mainīgo callback.
View	onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)	Izveido skatu, inicializē FoodListAdapter un pievieno to foodlist mainīgajam.

Mainīgie:

Tips	Mainīgais	Apraksts
SetTitleCallback	callback	Norāde uz galveno aktivitāti.
ExpandableListView	foodlist	Ēdienkartes saraksts.
FoodListAdapter	listAdapter	Datu sniedzējs ēdienkartes sarakstam.
static int	prev	Mainīgais glabā iepriekš izvērstās

		kategorijas pozīciju.
OnGroupExpandListener	onGroupExpandListener	Funkcija aizver iepriekš atvērto kategoriju.

4.5.3. Klase DailyFoodListFragment

Klase izveido dienas piedāvājumu skatu, kur attēlo sarakstu ar ēdieniem. Ja lietotājs uzspiež uz kādu no ēdieniem, klase izsauc DailyFoodFragment klasi, kura attēlo tieši specifisko ēdienu.

Metodes:

Tips	Metode	Apraksts
void	onAttach(Activity activity)	Kad fragments pievienots aktivitātei, tad inicializē mainīgos callback un food_callback.
void	onDetach()	Kad fragments tiek atvienots, tad iztukšo mainīgos callback un food_callback.
View	onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)	Dinamiski izveido režģa skatu.
ImageView	initImage(final SquareItem item)	Dinamiski ielādē objekta attēlu un uzstāda parametrus
TextView	initText(SquareType squareType, SquareItem item)	Dinamiski ielādē objekta tekstu un uzstāda parametrus.
SquareType	translateRandomIdToSquareType(int randomID)	Atgriež režģa objekta izmēru.
RowStyle	translateRandomIdToRowLayoutEnum(int randomID)	Atgriež režģa rindas izkārtojumu.
List<SquareItem>	initItems()	Ielādē visus dienas piedāvājumu objektus no datubāzes.

Mainīgie:

Tips	Mainīgais	Apraksts
List<SquareItem>	items	Saraksts ar dienas piedāvājumu objektiem.
int	padding	Glabā dinamiski izveidoto objektu sānu atstarpes izmērus.
boolean	avoidDuplicateRandomLayouts	Tiek izmantots, lai būtu mazāka iespēja, ka otrajā reizē būs tāds pats izkārtojums.
View	view	Norāde uz izveidoto skatu.
SetTitleCallback	callback	Norāde uz galveno aktivitāti.
FoodCallback	food_callback	Norāde uz galveno aktivitāti.
enum	RowStyle	Glabā režģa rindas izkārtojuma veidus.
enum	GridCreationState	Glabā režģa objektu izmēru veidus.

4.5.4. Klase FoodFragment

Klase izveido viena ēdiena skatu, kur attēlo ēdiena bildi, nosaukumu, cenu, nelielu informāciju par ēdienu un dod iespēju pievienot ēdienu pasūtījumam.

Metodes:

Tips	Metode	Apraksts
View	onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)	Inicializē viena ēdiena skata izkārtojumu, no datubāzes pēc iegūtās unikālās vērtības iegūst ēdienu un aizpilda laukus ar informāciju.

4.5.5. Klase DailyFoodFragment

Klase izveido specifiska ēdiena skatu, kur attēlo ēdiena bildi, nosaukumu, cenu, cenu ar atlaidi, nelielu informāciju par ēdienu un dod iespēju pievienot ēdienu pasūtījumam.

Metodes:

Tips	Metode	Apraksts
View	onCreateView(LayoutInflater inflater, ViewGroup container, Bundle	Inicializē viena ēdiena skata izkārtojumu, no datubāzes pēc

	savedInstanceState)	iegūtās unikālās vērtības iegūst ēdienu un aizpilda laukus ar informāciju.

4.5.6. Klase OrderListFragment

Klase izveido pasūtījuma skatu, kur attēlo ListView sarakstu ar visiem pasūtītajiem ēdieniem, to skaitu un kopējo cenu. Ja lietotājs nospiež pasūtīt, tad visi ēdieni tiek aizsūtīti servisam, kā pasūtījums.

Metodes:

Tips	Metode	Apraksts
void	onAttach(Activity activity)	Kad fragments pievienots aktivitātei, tad inicializē mainīgo callback.
void	onDetach()	Kad fragments tiek atvienots, tad iztukšo mainīgo callback.
View	onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)	Izveido skatu, inicializē OrderListAdapterun pievieno to orderlist mainīgajam.
static void	setTotalSum()	No datubāzes ielasa visus pasūtījuma objektus un aprēķina to kopējo summu.

Mainīgie:

Tips	Mainīgais	Apraksts
SetTitleCallback	callback	Norāde uz galveno aktivitāti.
ListView	orderlist	Pasūtījuma saraksts.
OrderListAdapter	listAdapter	Datu sniedzējs pasūtījuma sarakstam.
TextView	total	Norāde uz teksta vietu, kurā parāda kopējo summu.
DatabaseHelper	databaseHelper	Norāde uz datubāzes piekļuves klasi.
View	view	Norāde uz izveidoto skatu.
Toast	toast	Norāde uz ekrānziņu izvadi.

4.5.7. Klase BillFragment

Klase iegūst rēķina datus no servera un izveido rēķina skatu, kur attēlo ListView sarakstu ar visiem ēdieniem, to skaitu un kopējo cenu.

Metodes:

Tips	Metode	Apraksts
void	onAttach(Activity activity)	Kad fragments pievienots aktivitātei, tad inicializē mainīgo callback.
void	onDetach()	Kad fragments tiek atvienots, tad iztukšo mainīgo callback.
View	onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)	Izveido skatu, inicializē BillListAdapter un pievieno to billList mainīgajam.

Mainīgie:

Tips	Mainīgais	Apraksts
SetTitleCallback	callback	Norāde uz galveno aktivitāti.
ListView	billList	Rēķina saraksts.
BillListAdapter	listAdapter	Datu sniedzējs rēķina sarakstam.
TextView	total	Norāde uz teksta vietu, kurā parāda kopējo summu.
View	view	Norāde uz izveidoto skatu.
Toast	toast	Norāde uz ekrānziņu izvadi.
ParseQuery<ParseObject>	query	Tiek nodefinēts servisa pieprasījums.
FindCallback<ParseObject>	parseback	Tiek nodefinēta datu apstrāde pēc pieprasījuma izpildes.

4.5.8. Klase RateFragment

Klase izveido servisa novērtēšanas skatu, kur dod lietotājam iespēju ievadīt vērtējumu. Ja lietotājs uzspiež novērtēt, ievadītais teksts tiek nosūtīts serverim.

Metodes:

Tips	Metode	Apraksts
void	onAttach(Activity activity)	Kad fragments pievienots aktivitātei,

		tad inicializē mainīgo callback.
void	onDetach()	Kad fragments tiek atvienots, tad iztukšo mainīgo callback.
View	onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)	Izveido servisa novērtēšanas skatu.
boolean	isEmpty(EditText etText)	Pārbauda vai EditText laukā ir ievadīts teksts.

Mainīgie:

Tips	Mainīgais	Apraksts
SetTitleCallback	callback	Norāde uz galveno aktivitāti.
Toast	toast	Norāde uz ekrānziņu izvadi.

4.5.9. Klase MenuDrawerAdapter

Klase atgriež sānu izvēlnes datus un tās skatu.

Metodes:

Tips	Metode	Apraksts
konstruktors	MenuDrawerAdapter(Context context)	Inicializē sānu izvēlnes datu sniedzēju, pievieno sānu izvēlnes objektus drawerItemList sarakstam.
View	getView(int position, View convertView, ViewGroup parent)	Atgriež katras sānu izvēlnes objekta izskatu.
int	getCount()	Atgriež objektu skaitu.
DrawerItem	getItem(int position)	Atgriež objektu pēc pozīcijas.
long	getItemId(int position)	Android automātiski ģenerēta funkcija.

Mainīgie:

Tips	Mainīgais	Apraksts
Context	context	Norāde uz galveno aktivitāti.
List<DrawerItem>	drawerItemList	Saraksts ar sānu izvēlnes objektiem.

4.5.10. Klase FoodListAdapter

Klase atgriež ēdienkartes datus un tās skatu.

Metodes:

Tips	Metode	Apraksts
konstruktors	FoodListAdapter(Context context)	Inicializē ēdienkartes datu sniedzēju, ielasa no datubāzes ēdienu kategorijas un to ēdienus.
View	getChildView(final int groupPosition, final int childPosition, boolean isLastChild, View convertView, ViewGroup parent)	Atgriež katra ēdiena skatu pēc grupas pozīcijas un skata pozīcijas.
int	getChildrenCount(int groupPosition)	Atgriež kategorijas ēdienu skaitu.
Food	getChild(int groupPosition, int childPosititon)	Atgriež ēdienu pēc grupas pozīcijas un skata pozīcijas.
long	getChildId(int groupPosition, int childPosition)	Android automātiski ģenerēta funkcija.
View	View getGroupView(int groupPosition, boolean isExpanded, View convertView, ViewGroup parent)	Atgriež katras ēdiena kategorijas skatu pēc pozīcijas.
int	getGroupCount()	Atgriež ēdienu kategoriju skaitu.
FoodCategory	getGroup(int groupPosition)	Atgriež ēdienu kategoriju pēc pozīcijas.
long	getGroupId(int groupPosition)	Android automātiski ģenerēta funkcija.
boolean	hasStableIds()	Android automātiski ģenerēta funkcija.
boolean	isChildSelectable(int groupPosition, int childPosition)	Android automātiski ģenerēta funkcija.

Mainīgie:

Tips	Mainīgais	Apraksts
Context	ctx	Norāde uz galveno aktivitāti.
List<FoodCategory>	listFoodCategory	Saraksts ar ēdienu kategorijām.
HashMap<FoodCategory, List<Food>>	listFood	Karte ar ēdienu kategorijām un to ēdieniem.
DatabaseHelper	databaseHelper	Norāde uz datubāzes piekļuves klasi.

4.5.11. Klase OrderListAdapter

Klase atgriež pasūtījuma datus un tās skatu.

Metodes:

Tips	Metode	Apraksts
konstruktors	OrderListAdapter(Context context)	Inicializē pasūtījuma datu sniedzēju.
void	load()	Ielādē pasūtījuma objektus no datubāzes.
View	getView(int position, View convertView, ViewGroup parent)	Atgriež katrapasūtījuma objekta izskatu.
int	getCount()	Atgriež objektu skaitu.
Order	getItem(int position)	Atgriež objektu pēc pozīcijas.
long	getItemId(int position)	Android automātiski ģenerēta funkcija.

Mainīgie:

Tips	Mainīgais	Apraksts
Context	ctx	Norāde uz galveno aktivitāti.
List<Order>	order	Saraksts ar pasūtījuma objektiem.
DatabaseHelper	databaseHelper	Norāde uz datubāzes piekļuves klasi.

4.5.12. Klase BillListAdapter

Klase sakārto pasūtījuma datus un atgriež tā skatu.

Metodes:

Tips	Metode	Apraksts
konstruktors	BillListAdapter(Context context, List<ParseObject> object)	Inicializē rēķina datu sniedzēju, noņem vienādos ēdienus no object saraksta un saskaita katra ēdiena skaitu.
View	getView(int position, View convertView, ViewGroup parent)	Atgriež katra rēķina objekta izskatu.
int	getCount()	Atgriež objektu skaitu.
ParseObject	getItem(int position)	Atgriež objektu pēc pozīcijas.
long	getItemId(int position)	Android automātiski ģenerēta funkcija.

Mainīgie:

Tips	Mainīgais	Apraksts
Context	ctx	Norāde uz galveno aktivitāti.
List<ParseObject>	bill	Saraksts ar rēķina objektiem.
Map<Integer, Integer>	count	Karte, kurā katram ēdienam tiek glabāts skaits.

4.5.13. Klase DatabaseHelper

Klase izveido datubāzē tabulas un atgriež pieeju tabulām.

Metodes:

Tips	Metode	Apraksts
konstruktors	DatabaseHelper(Context context)	Inicializē pieeju datubāzei.
void	onCreate(SQLiteDatabase db, ConnectionSource connectionSource)	Izveido visas nodefinētās tabulas.
void	onUpgrade(SQLiteDatabase db, ConnectionSource connectionSource, int oldVersion, int newVersion)	Izdzēš visas nodefinētās tabulas un izsauc onCreate funkciju.

Dao<FoodCategory, Integer>	getFoodCategoryDao()	Atgriež pieeju ēdienu kategoriju tabulai.
Dao<Food, Integer>	getFoodDao()	Atgriež pieeju ēdienu tabulai.
Dao<DailyFood, Integer>	getDailyFoodDao()	Atgriež pieeju dienas piedāvājumu tabulai.
Dao<Order, Integer>	getOrderDao()	Atgriež pieeju pasūtījumu tabulai.
void	close()	Aizver pieeju visām tabulām.

Mainīgie:

Tips	Mainīgais	Apraksts
static final String	DATABASE_NAME	Datubāzes nosaukums.
static final int	DATABASE_VERSION	Datubāzes versija.
Dao<FoodCategory, Integer>	foodCategoryDao	Norāde uz ēdienu kategoriju tabulu.
Dao<Food, Integer>	foodDao	Norāde uz ēdienu tabulu.
Dao<DailyFood, Integer>	dailyFoodDao	Norāde uz dienas piedāvājumu tabulu.
Dao<Order, Integer>	orderDao	Norāde uz pasūtījumu tabulu.

4.5.14. Klase DatabaseFoodCache

Klase saglabā datubāzē dienas piedāvājumu datus, ēdienkartes datus un ielādē visus attēlus.

Metodes:

Tips	Metode	Apraksts
konstruktors	DatabaseFoodCache(Context ctx)	Inicializē pieeju datubāzei.
void	AddFoodList(List<ParseObject> object)	Funkcija pievieno ēdienu karti datubāzei.
void	addFoodToDB(JSONArray array, String objectid)	Funkcija pievieno ēdienus datubāzei.
void	AddDailyList(List<ParseObject> object)	Funkcija pievieno dienas

		piedāvājumus datubāzei.
class	DownloadImage extends AsyncTask<ArrayList<String>, Void, Void>	Lejupielādē visus ēdienu attēlus asinhroni.

Mainīgie:

Tips	Mainīgais	Apraksts
DownloadCallBack	callback	Norāde uz SplashActivity klasi.
DatabaseHelper	databaseHelper	Norāde uz datubāzes piekļuves klasi.
Dao<FoodCategory, Integer>	daoCategory	Norāde uz ēdienu kategoriju tabulu.
Dao<Food, Integer>	daoFood	Norāde uz ēdienu tabulu.
Dao<DailyFood, Integer>	daoDaily	Norāde uz dienas piedāvājumu tabulu.
UpdateBuilder<FoodCategory, Integer>	updateCategoryBuilder	Glabā ēdienu kategorijas atjaunošanas pieprasījumu.
DeleteBuilder<DailyFood, Integer>	removeDailyBuilder	Glabā dienas piedāvājumu izdzēšanas pieprasījumu.
DeleteBuilder<Food, Integer>	removeFoodBuilder	Glabā ēdienu izdzēšanas pieprasījumu.
DeleteBuilder<FoodCategory, Integer>	removeCategoryBuilder	Glabā ēdienu kategoriju izdzēšanas pieprasījumu.
ArrayList<String>	foodpictures	Saraksts ar ēdienu attēlu adresēm.

4.5.15. Klase DatabaseOrder

Klase saglabā un izņem pasūtījumu objektus no datubāzes.

Metodes:

Tips	Metode	Apraksts
void	AddOrder(DatabaseHelper databaseHelper, int nid, String food, double cost)	Pievieno objektu pasūtījuma tabulai.
void	RemoveOrder(DatabaseHelper databaseHelper, int nid, int count)	Noņem objektu pasūtījuma tabulai.

4.5.16. Klase DrawerItem

Klasē nodefinēti sānu izvēlnes objektu dati.

Metodes:

Tips	Metode	Apraksts
konstruktors	DrawerItem(String name, int imgID)	Izveido jaunu sānu izvēlnes objektu.
String	getName()	Atgriež objekta nosaukumu.
int	getImgID()	Atgriež objekta attēla id.

Mainīgie:

Tips	Mainīgais	Apraksts
String	name	Sānu izvēlnes objekta nosaukums.
int	imgID	Sānu izvēlnes attēla id.

4.5.17. Klase DailyFood

Klasē nodefinēti dienas piedāvājumu objektu dati.

Metodes:

Tips	Metode	Apraksts
konstruktors	DailyFood(String food, String about, double costNow, double costBefore, int nid, String picture, String objectid, Date	Izveido jaunu dienas piedāvājumu objektu.

	updatedat)	
	DailyFood()	Nepieciešams OrmLite bibliotēkai.
String	getFood()	Atgriež objekta nosaukumu.
double	getCostNow()	Atgriež objekta cenu tagad.
double	getCostBefore()	Atgriež objekta cenu iepriekš.
String	getPicture()	Atgriež objekta bildes adresi.
String	getAbout()	Atgriež objekta informāciju.
int	getNid()	Atgriež objekta unikālo vērtību.
String	getObjectid()	Atgriež objekta unikālo vērtību, kura izveidojas automātiski servisā.
Date	getUpdatedAt()	Atgriež objekta pēdējo atjaunošanas datumu.

Mainīgie:

Tips	Mainīgais	Apraksts
int	id	Datubāzes unikāls ID.
String	food	Dienas piedāvājuma objekta nosaukums.
String	about	Dienas piedāvājuma objekta informācija.
double	costNow	Dienas piedāvājuma objekta cena tagad.
double	costBefore	Dienas piedāvājuma objekta cena iepriekš.
int	nid	Dienas piedāvājuma objekta unikālā vērtība.
String	picture	Dienas piedāvājuma objekta attēla adrese.
String	objectid	Dienas piedāvājuma objekta

		unikālā vērtība, kura izveidojas automātiski servisā.
Date	updatedat	Dienas piedāvājuma objekta pēdējais atjaunošanas datums.

4.5.18. Klase Food

Klasē nodefinēti ēdienkartes ēdienu objektu dati.

Metodes:

Tips	Metode	Apraksts
konstruktors	Food(String food, String about, double cost, int nid, String picture, String parent)	Izveido jaunu ēdiena objektu.
	Food()	Nepieciešams OrmLite bibliotēkai.
String	getFood()	Atgriež objekta nosaukumu.
double	getCost()	Atgriež objekta cenu tagad.
String	getPicture()	Atgriež objekta bildes adresi.
String	getAbout()	Atgriež objekta informāciju.
int	getNid()	Atgriež objekta unikālo vērtību.
String	getParentid()	Atgriež objekta vecāka unikālā vērtība.

Mainīgie:

Tips	Mainīgais	Apraksts
int	id	Datubāzes unikāls ID.
String	food	Ēdiena objekta nosaukums.
String	about	Ēdiena objekta informācija.
double	cost	Ēdiena objekta cena.
int	nid	Ēdiena objekta unikālā vērtība.
String	picture	Ēdiena objekta attēla adrese.
String	parentId	Ēdiena objekta vecāka unikālā vērtība.

4.5.19. Klase FoodCategory

Klasē nodefinēti ēdienkartes kategoriju objektu dati.

Metodes:

Tips	Metode	Apraksts
konstruktors	FoodCategory(String food_category, Date updatedAt, String objectId)	Izveido jaunu ēdienu kategorijas objektu.
	FoodCategory()	Nepieciešams OrmLite bibliotēkai.
String	getFood_category ()	Atgriež objekta nosaukumu.
Date	getUpdatedAt ()	Atgriež objekta pēdējo atjaunošanas datumu.
String	getObjectid()	Atgriež objekta unikālo vērtību.

Mainīgie:

Tips	Mainīgais	Apraksts
int	id	Datubāzes unikāls ID.
String	food_category	Ēdienu kategorijas objekta nosaukums.
Date	updatedAt	Ēdienu kategorijas objekta pēdējais atjaunošanas datums.
String	objectId	Ēdienu kategorijas objekta unikālā vērtība, kura automātiski izveidota servisā.

4.5.20. Klase Order

Klasē nodefinēti pasūtījumu objektu dati.

Metodes:

Tips	Metode	Apraksts
konstruktors	Order(int nid, String name, double cost)	Izveido jaunu pasūtījuma objektu.
	Order()	Nepieciešams OrmLite bibliotēkai.

int	getNid()	Atgriež pasūtījuma objekta unikālo vērtību.
String	getName ()	Atgriež objekta nosaukumu.
double	getCost ()	Atgriež objekta cenu.
int	getCount ()	Atgriež objekta skaita vērtību.

Mainīgie:

Tips	Mainīgais	Apraksts
int	id	Datubāzes unikāls ID.
int	nid	Pasūtījuma objekta unikālā vērtība.
String	name	Pasūtījuma objekta nosaukums.
double	cost	Pasūtījuma objekta cena.
int	count	Pasūtījuma objekta skaits.

4.5.21. Klase ShadowProperties

Klasē nodefinēti dienas piedāvājuma skata objektu ēnu dati.

Metodes:

Tips	Metode	Apraksts
konstruktors	ShadowProperties(float radius, int dx, int dy, int color)	Izveido jaunu ēnu parametru objektu.
float	getRadius ()	Atgriež objekta rādiusu.
void	setRadius(float radius)	Uzstāda objekta rādiusu.
int	getDx()	Atgriež objekta attālumu no centra pa X asi.
void	setDx(int dx)	Uzstāda objekta attālumu no centra pa X asi.
int	getDy()	Atgriež objekta attālumu no centa pa Y asi.
void	setDy(int dy)	Uzstāda objekta attālumu no centra pa Y asi.

int	getColor()	Atgriež objekta krāsu.
void	setColor(int color)	Uzstāda objekta krāsu.

Mainīgie:

Tips	Mainīgais	Apraksts
float	radius	Ēnas objekta rādiuss.
int	dx	Ēnas objekta attālums no centra pa X asi.
int	dy	Ēnas objekta attālums no centra pa Y asi.
int	color	Ēnas objekta krāsa.

4.5.22. Klase SquareItem

Klasē nodefinēti dienas piedāvājuma skata objektu dati.

Metodes:

Tips	Metode	Apraksts
konstruktors	SquareItem(String label, String string, int nid)	Izveido jaunu dienas piedāvājumu objektu.
void	applyFontStyle(int fontsize_large, int fontsize_medium, int fontsize_small, Typeface font, int fontColor, ShadowProperties shadowProperties)	Uzstāda šriftu tekstam.
String	getImage()	Atgriež objekta attēla adresi.
void	setImage(String image)	Uzstāda objekta attēla adresi.
String	getTitle()	Atgriež objekta nosaukumu.
void	setTitle(String title)	Uzstāda objekta nosaukumu.
int	getNid()	Atgriež objekta unikālo vērtību.
int	getFontSize(SquareType squareType)	Atgriež objekta šrifta izmēru.
Typeface	getFont(SquareType squareType)	Atgriež objekta šriftu.
int	getFontColor(SquareType squareType)	Atgriež objekta šrifta krāsu.
ShadowProperties	getShadowProperties(SquareType squareType)	Atgriež objekta ēnas parametrus.

Mainīgie:

Tips	Mainīgais	Apraksts
String	title	Dienas piedāvājuma nosaukums.
String	image	Dienas piedāvājuma attēla adrese.
int	nid	Dienas piedāvājuma unikālā vērtība.
Typeface	font	Šrifts.
int	fontsize_small	Maza šrifta izmērs.
int	fontsize_medium	Vidēja šrifta izmērs.
int	fontsize_large	Liela šrifta izmērs.
int	fontColor	Šrifta krāsa.
ShadowProperties	shadowProperties	Ēnu parametri.

4.5.23. Klase NFCAdapter

Klase ieslēdz, izslēdz NFC lasītāju.

Metodes:

Tips	Metode	Apraksts
konstruktors	NFCAdapter(Activity activity, Toast toast)	Inicializē jaunu NFC lasītāju.
void	enableForegroundDispatch()	Ieslēdz NFC lasītāju.
void	disableForegroundDispatch()	Izslēdz NFC lasītāju.
boolean	isInitialized()	Pārbauda vai NFC lasītājs ir inicializēts.

Mainīgie:

Tips	Mainīgais	Apraksts
NfcAdapter	mNfcAdapter	Norāde uz NFC lasītāju.
Activity	activity	Norāde uz galveno aktivitāti.
Toast	toast	Norāde uz ekrānziņu izvadi.

4.5.24. Klase NFCIntentHandle

Klase iegūst NFC ielasītos datus, apstrādā un atgriež tos galvenajai aktivitātei.

Metodes:

Tips	Metode	Apraksts
konstruktors	NFCIntentHandle()	Ieslēdz servisu.
void	onHandleIntent(Intent intent)	Saņem ielasītās NFC mikroshēmas datus, apstrādā tos un atgriež atpakaļ galvenajai aktivitātei.
String	readText(NdefRecord record)	Pārveido NDEF uz tekstu.
String	dumpTagData(Parcelable p)	Iegūst NFC mikroshēmas birku.
long	getDec(byte[] bytes)	Pārveido mikroshēmas ielasītos baitus par decimālu vērtību.

4.5.25. Klase Network

Klase pārbauda vai ir pieejams interneta pieslēgums.

Metodes:

Tips	Metode	Apraksts
static final boolean	isOnline(Context cxt)	Pārbauda vai pieejams interneta savienojums.

4.5.26. Klase TypefaceSpan

Klase nomaina lietotnes virsraksta šriftu.

Metodes:

Tips	Metode	Apraksts
konstruktors	TypefaceSpan(Context context, String typefaceName)	Ielādē šriftu un saglabā to kešatmiņā.
void	updateMeasureState(TextPaint p)	Atjauno izmēru stāvokli.
void	updateDrawState(TextPaint tp)	Atjauno zīmējuma stāvokli.

Mainīgie:

Tips	Mainīgais	Apraksts
Typeface	mTypeface	Šrifts.
static LruCache<String, Typeface>	sTypefaceCache	Kešatmiņa, kur glabāsies šrifts.

4.5.27. Klase App

Klase brīdī, kad lietotne tiek ieslēgta, inicializē visus nepieciešamos objektus.

Metodes:

Tips	Metode	Apraksts
void	onCreate()	Funkcija ielādē šriftu, noskaidro ierīces ID, inicializē attēlu ielādētāju un Parse servisu.

Mainīgie:

Tips	Mainīgais	Apraksts
Typeface	face	Šrifts.
String	deviceid	Ierīces ID.

4.5.28. Klase C

Klase nodefinētas visas konstantās String vērtības, kuras tiek izmantotas lietotnes darbības laikā.

Mainīgie:

Tips	Mainīgais	Apraksts
static final String	APP_ID	Parse servisa lietotnes ID.
static final String	CLIENT_KEY	Parse servisa klienta ID.
static final String	PARSE_FOOD, PARSE_DAILY, ORDER, WAITER, USER_RAITIN.	Parse servisa tabulu nosaukumi.

static final String	PARSE_TYPE, PARSE_GALDA_ID, PARSE_FOOD_NID, PARSE_DEVICE_ID, FOOD_CATEGORY, PARENT, OBJECTID, UPDATEDAT, FOOD_PICTURE, FOOD_ARRAY, FOOD, FOOD_ABOUT, FOOD_COST, NID, COST_NOW, COST_BEFORE, DAILY, COUNT.	Vērtības, kuras tiek izmantotas, lai parsētu iegūtos datus no Parse servisa.
static final String	PREFS	Glabā koplietojamo uzstādījumu nosaukumu.
static final long	FIVEMINS	Glabā 5 minūšu vērtību.
static final String	EXTRAID, EXTRATAG.	Tiek izmantoti lai atšifrētu NFC mikroshēmas datus.
static final String	BROADCAST, BROADCAST_NFC, BROADCAST_SUCCESSFUL, BROADCAST_FAIL.	Tiek izmantoti pārsūtot NFC mikroshēmas datus.

4.5.29. Klase SplashActivity

Sākuma aktivitāte, kura attēlojas, kamēr ielādē, saglabā visus ēdienkartes, dienas piedāvājumu datus un bildes.

Metodes:

Tips	Metode	Apraksts
void	onCreate(Bundle savedInstanceState)	Ielādē skatu un, ja ir interneta pieslēgums, sāk ēdienkartes un dienas piedāvājumu ielādi.
void	onDownloaded()	Funkcija, kura beidz SplashActivity un sāk MainActivity. Tiek izsaukta, kad visi dati ielādēti.
void	setLoadingText(final String text)	Funkcija nomaina ielādes tekstu.
class	DownloadFood extends AsyncTask<String, Void, Void>	Funkcija asinhroni ielādē un saglabā datubāzē ēdienkarti un dienas piedāvājumus.

Mainīgie:

Tips	Mainīgais	Apraksts
Intent	startMain	Mainīgais, kurš glabā norādi uz MainActivity.
DatabaseFoodCache	addDB	Norāde uz DatabaseFoodCache klasi.
TextView	loading_text	Teksts, kurā attēlo ielādes informāciju.
ImageView	splash	Fona bilde.
Typeface	face	Šrifts.

4.5.30. Klase MainActivity

Galvenā aktivitāte, kura izveido sānu izvēlni, maina skatus, uzliek lietotnes virsraksta šriftu, atbild par NFCAdapter klasi.

Metodes:

Tips	Metode	Apraksts
void	onCreate(Bundle savedInstanceState)	Ielādē skatu.
boolean	onCreateOptionsMenu(Menu menu)	Android automātiski ģenerēta funkcija.
void	SelectItem(int position)	Funkcija pēc pozīcijas nomaina skatus.
void	callWaiter()	Funkcija pārbauda vai viesmīlis nav ticis izsaukts jau piecas minūtes atpakaļ, ieslēdz NFC lasītāju un gadījumā, ja NFC mikroshēmas dati ir veiksmīgi ielasīti nosūta servisam viesmīļa izsaukumu.
void	onNewIntent(Intent intent)	Saņemot jaunu Intent objektu, izsauc handleIntent funkciju.
void	onResume()	Funkcija, ja NFC lasītājs ir ticis inicializēts, ieslēdz to.
void	handleIntent(Intent intent)	Apstrādā Intent objektu un nosūta to NFCIntentHandle servisam.
void	setTitle(CharSequence title)	Nomaina lietotnes virsraksta tekstu.
void	onPostCreate(Bundle savedInstanceState)	Android automātiski ģenerēta funkcija.
void	onConfigurationChanged(Configuration newConfig)	Android automātiski ģenerēta funkcija.
boolean	onOptionsItemSelected(MenuItem item)	Funkcija atver sānu izvēlni.
void	onBackPressed()	Ja sānu izvēlne nav atvērta, tad atver to, citādi beidz galveno aktivitāti.

void	onFoodClick(Fragment fragment)	Funkcija nomaina skatu un pievieno to stekā.
class	DrawerItemClickListener	Funkcija, kad tiek nospiests uz sānu izvēlnes izsauc SelectItem funkciju.

Mainīgie:

Tips	Mainīgais	Apraksts
DrawerLayout	mDrawerLayout	Sānu izvēlnes izskats.
ListView	mDrawerList	Sānu izvēlnes saraksts.
ActionBarDrawerToggle	mDrawerToggle	Poga lietotnes augšējā kreisajā stūrī, kura atver/aizver sānu izvēlni.
MenuDrawerAdapter	adapter	Sānu izvēlnes datu sniedzējs.
NFCAdapter	nfcadapter	Norāde uz NFCAdapter klasi.
Toast	toast	Norāde uz ekrānziņu izvadi.

4.5.31. Skatu izkārtojumi.

activity_main.xml – satur galvenās aktivitātes izkārtojumu.
activity_splash.xml – satur ielādes aktivitātes izkārtojumu.
bill_list.xml – satur rēķina izkārtojumu.
custom_drawer_item.xml – satur sānu izvēlnes objekta izkārtojumu.
food_list_fragment.xml – satur ēdienkartes izkārtojumu.
food_list_group.xml – satur ēdienkartes kategorijas izkārtojumu.
food_list_item.xml – satur ēdienkartes kategorijas bērna izkārtojumu.
food.xml – satur viena ēdiena izkārtojumu.
fragment_staggered_view.xml – satur dienas piedāvājumu izkārtojumu.
nfc_alert_dialog.xml – satur NFC brīdinājuma izkārtojumu.
order_list_footer.xml – satur pasūtījuma kājenes izkārtojumu.
order_list_item.xml – satur pasūtījuma objekta izkārtojumu.
order_list.xml – satur pasūtījuma izkārtojumu.
rate_service.xml – satur novērtēšanas izkārtojumu.

4.5.32. Iepriekš nodefinētās vērtības.

color.xml – satur krāsu kodus, kurus izmanto lietotnē.
strings.xml – satur lietotnes nosaukumu un izvadziņu tekstus.
styles.xml – satur teksta stilus, kurus izmanto lietotnē.

4.6. Viesmīļa lietotnes moduļu projektējums

4.6.1. Klase FoodListFragment

Klase izveido ēdienkartes skatu, kur attēlo ExpandableListView sarakstu ar ēdienu kategorijām un ēdieniem zem tām, dod iespēju pievienot kādu no ēdieniem pasūtījumam.

Metodes:

Tips	Metode	Apraksts
void	onAttach(Activity activity)	Kad fragments pievienots aktivitātei, tad inicializē mainīgo callback.
void	onDetach()	Kad fragments tiek atvienots, tad iztukšo mainīgo callback.
View	onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)	Izveido skatu, inicializē FoodListAdapter un pievieno to foodlist mainīgajam.

Mainīgie:

Tips	Mainīgais	Apraksts
TitleCallback	callback	Norāde uz galveno aktivitāti.
ExpandableListView	foodlist	Ēdienkartes saraksts.
FoodListAdapter	listAdapter	Datu sniedzējs ēdienkartes sarakstam.
static int	prev	Mainīgais glabā iepriekš izvērstās kategorijas pozīciju.
OnGroupExpandListener	onGroupExpandListener	Funkcija aizver iepriekš atvērto kategoriju.

4.6.2. Klase CallWaiterFragment

Klase izveido skatu, kur attēlo sarakstu ar galdu numuriem, kuri izsauca viesmīli.

Metodes:

Tips	Metode	Apraksts
void	onAttach(Activity activity)	Kad fragments pievienots aktivitātei, tad inicializē mainīgo callback.
void	onDetach()	Kad fragments tiek atvienots, tad iztukšo mainīgo callback.
View	onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)	Izveido skatu, ielādē visus izsaukto viesmīļu tabulas datus no servisa, inicializē CallWaiterListAdapter un pievieno to call_list mainīgajam.

Mainīgie:

Tips	Mainīgais	Apraksts
TitleCallback	callback	Norāde uz galveno aktivitāti.
ListView	call_list	Ēdienkartes saraksts.
CallWaiterListAdapter	listadapter	Datu sniedzējs ēdienkartes sarakstam.

4.6.3. Klase OrderFragment

Klase atgriež skatu, kurā attēlo pasūtījumus, ja uzspiež uz kādu no pasūtījumiem, klase izsauc SingleOrderFragment klasi, kura attēlo tieši specifisko pasūtījumu.

Metodes:

Tips	Metode	Apraksts
void	onAttach(Activity activity)	Kad fragments pievienots aktivitātei, tad inicializē mainīgo callback.
void	onDetach()	Kad fragments tiek atvienots, tad iztukšo mainīgo callback.
View	onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)	Izveido skatu, ielādē pasūtījumu datus no servisa pēc to tipa, sadala visu pasūtījumu objektus pa galdiem, inicializē

		OrderListAdapter un pievieno to order_list mainīgajam.

Mainīgie:

Tips	Mainīgais	Apraksts
TitleCallback	callback	Norāde uz galveno aktivitāti.
ListView	order_list	Pasūtījumu saraksts.
OrderListAdapter	listadapter	Datu sniedzējs pasūtījuma sarakstam.
List<List<ParseObject>>	parseobject	Saraksts ar galdu numuriem un to pasūtījumiem.

4.6.4. Klase SingleOrderFragment

Klase atgriež skatu, kurā attēlo sarakstu ar specifiskā pasūtījuma datiem.

Metodes:

Tips	Metode	Apraksts
void	onAttach(Activity activity)	Kad fragments pievienots aktivitātei, tad inicializē mainīgo callback.
void	onDetach()	Kad fragments tiek atvienots, tad iztukšo mainīgo callback.
View	onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)	Izveido skatu, ielādē specifiskā pasūtījuma datus no servisa, inicializē SingleOrderListAdapter un pievieno to order_list mainīgajam.

Mainīgie:

Tips	Mainīgais	Apraksts
TitleCallback	callback	Norāde uz galveno aktivitāti.
ListView	order_list	Pasūtījumu saraksts.
SingleOrderListAdapter	listAdapter	Datu sniedzējs pasūtījuma sarakstam.

4.6.5. Klase FoodListAdapter

Klase atgriež ēdienkartes datus un tās skatu.

Metodes:

Tips	Metode	Apraksts
konstruktors	FoodListAdapter(Context context)	Inicializē ēdienkartes datu sniedzēju, ielasa no datubāzes ēdienu kategorijas un to ēdienus.
View	getChildView(final int groupPosition, final int childPosition, boolean isLastChild, View convertView, ViewGroup parent)	Atgriež katra ēdiena skatu pēc grupas pozīcijas un skata pozīcijas.
int	getChildrenCount(int groupPosition)	Atgriež kategorijas ēdienu skaitu.
Food	getChild(int groupPosition, int childPosititon)	Atgriež ēdienu pēc grupas pozīcijas un skata pozīcijas.
long	getChildId(int groupPosition, int childPosition)	Android automātiski ģenerēta funkcija.
View	View getGroupView(int groupPosition, boolean isExpanded, View convertView, ViewGroup parent)	Atgriež katras ēdiena kategorijas skatu pēc pozīcijas.
int	getGroupCount()	Atgriež ēdienu kategoriju skaitu.
FoodCategory	getGroup(int groupPosition)	Atgriež ēdienu kategoriju pēc pozīcijas.
long	getGroupId(int groupPosition)	Android automātiski ģenerēta funkcija.
boolean	hasStableIds()	Android automātiski ģenerēta funkcija.
boolean	isChildSelectable(int groupPosition, int childPosition)	Android automātiski ģenerēta funkcija.

Mainīgie:

Tips	Mainīgais	Apraksts
Context	ctx	Norāde uz galveno aktivitāti.
List<FoodCategory>	listFoodCategory	Saraksts ar ēdienu kategorijām.
HashMap<FoodCategory, List<Food>>	listFood	Karte ar ēdienu kategorijām un to ēdieniem.
DatabaseHelper	databaseHelper	Norāde uz datubāzes piekļuves klasi.
String	galda_id	Glabā galda unikālo vērtību.
String	device_id	Glabā ierīces unikālo vērtību.

4.6.6. Klase CallWaiterListAdapter

Klase atgriež viesmīļu izsaukumu datus un skatu.

Metodes:

Tips	Metode	Apraksts
konstruktors	CallWaiterListAdapter(List<ParseObject> parseobject, Context context, Map<String, String> table_id)	Inicializē viesmīļu izsaukumu datu sniedzēju.
int	getCount()	Atgriež viesmīļu izsaukumu skaitu.
ParseObject	getItem(int position)	Atgriež viesmīļu izsaukumu pēc tā pozīcijas.
long	getItemId(int arg0)	Android automātiski ģenerēta funkcija.
View	getView(final int position, View arg1, ViewGroup container)	Atgriež katra viesmīļa izsaukuma skatu pēc pozīcijas.

Mainīgie:

Tips	Mainīgais	Apraksts
Context	ctx	Norāde uz galveno aktivitāti.
List<ParseObject>	child	Saraksts ar viesmīļu izsaukumiem.
Map<String, String>	table_id	Karte ar NFC mikroshēmas ID un tās galda ID atšifrējumu.

4.6.7. Klase MenuDrawerAdapter

Klase atgriež sānu izvēlnes datus un tās skatu.

Metodes:

Tips	Metode	Apraksts
konstruktors	MenuDrawerAdapter(Context context)	Inicializē sānu izvēlnes datu sniedzēju, pievieno sānu izvēlnes objektus drawerItemList sarakstam.
View	getView(int position, View convertView, ViewGroup parent)	Atgriež katras sānu izvēlnes objekta izskatu.
int	getCount()	Atgriež objektu skaitu.
DrawerItem	getItem(int position)	Atgriež objektu pēc pozīcijas.
long	getItemId(int position)	Android automātiski ģenerēta funkcija.

Mainīgie:

Tips	Mainīgais	Apraksts
Context	context	Norāde uz galveno aktivitāti.
List<DrawerItem>	drawerItemList	Saraksts ar sānu izvēlnes objektiem.

4.6.8. Klase OrderListAdapter

Klase atgriež pasūtījumu datus un tā skatu.

Metodes:

Tips	Metode	Apraksts
konstruktors	OrderListAdapter(List<List<ParseObject>> parseobject, Context context, int id, Map<String, String> table_id)	Inicializē pasūtījumu datu sniedzēju, sakārto pasūtījumus augošā secībā pēc to izveides laika.
View	getView(final int position, View arg1, ViewGroup container)	Atgriež katra pasūtījuma objekta izskatu.
int	getCount()	Atgriež objektu skaitu.
List<ParseObject>	getItem(int position)	Atgriež objektu pēc pozīcijas.
long	getItemId(int position)	Android automātiski ģenerēta funkcija.

Mainīgie:

Tips	Mainīgais	Apraksts
Context	ctx	Norāde uz galveno aktivitāti.
List<List<ParseObject>>	child	Saraksts ar pasūtījumiem.
Map<String, String>	table_id	Karte ar NFC mikroshēmas ID un tās galda ID atšifrējumu.
int	id	Tips, pēc kura atlasa pasūtījumus.

4.6.9. Klase SingleOrderListAdapter

Klase atgriež specifiska pasūtījuma datus un tā skatu.

Metodes:

Tips	Metode	Apraksts
konstruktors	SingleOrderListAdapter(List<ParseObject> parseobject, Context context)	Inicializē specifiska pasūtījumu datu sniedzēju, izsauc cost() funkciju.
View	getView(final int position, View arg1, ViewGroup container)	Atgriež katra pasūtījuma objekta izskatu, ja visi ēdieni ir piegādāti, tad atgriež arī kopējo summu.
int	getCount()	Atgriež objektu skaitu.
List<ParseObject>	getItem(int position)	Atgriež objektu pēc pozīcijas.
long	getItemId(int position)	Android automātiski ģenerēta funkcija.
void	cost()	Aprēķina pasūtījuma kopējo summu.

Mainīgie:

Tips	Mainīgais	Apraksts
Context	ctx	Norāde uz galveno aktivitāti.
List<ParseObject>	child	Saraksts ar pasūtījuma objektiem.
double	cost	Mainīgais glabā kopējo pasūtījuma summu.
boolean	delivered	Mainīgais, kuru izmanto, lai atzīmētu, ka visi ēdieni ir piegādāti pie galda.

4.6.10. Klase DatabaseHelper

Klase izveido datubāzē tabulas un atgriež pieeju tabulām.

Metodes:

Tips	Metode	Apraksts
konstruktors	DatabaseHelper(Context context)	Inicializē pieeju datubāzei.
void	onCreate(SQLiteDatabase db, ConnectionSource connectionSource)	Izveido visas nodefinētās tabulas.
void	onUpgrade(SQLiteDatabase db, ConnectionSource connectionSource, int oldVersion, int newVersion)	Izdzēš visas nodefinētās tabulas un izsauc onCreate funkciju.
Dao<FoodCategory, Integer>	getFoodCategoryDao()	Atgriež pieeju ēdienu kategoriju tabulai.
Dao<Food, Integer>	getFoodDao()	Atgriež pieeju ēdienu tabulai.
void	close()	Aizver pieeju visām tabulām.

Mainīgie:

Tips	Mainīgais	Apraksts
static final String	DATABASE_NAME	Datubāzes nosaukums.
static final int	DATABASE_VERSION	Datubāzes versija.
Dao<FoodCategory, Integer>	foodCategoryDao	Norāde uz ēdienu kategoriju tabulu.
Dao<Food, Integer>	foodDao	Norāde uz ēdienu tabulu.

4.6.11. Klase Food

Klasē nodefinēti ēdienkartes ēdienu objektu dati.

Metodes:

Tips	Metode	Apraksts
konstruktors	Food(String food, double cost, int nid, String parent)	Izveido jaunu ēdiena objektu.
	Food()	Nepieciešams OrmLite bibliotēkai.
String	getFood()	Atgriež objekta nosaukumu.
double	getCost()	Atgriež objekta cenu tagad.
int	getNid()	Atgriež objekta unikālo vērtību.
String	getParentid()	Atgriež objekta vecāka unikālā vērtība.

Mainīgie:

Tips	Mainīgais	Apraksts
int	id	Datubāzes unikāls ID.
String	food	Ēdiena objekta nosaukums.
double	cost	Ēdiena objekta cena.
int	nid	Ēdiena objekta unikālā vērtība.
String	parentId	Ēdiena objekta vecāka unikālā vērtība.

4.6.12. Klase FoodCategory

Klasē nodefinēti ēdienkartes kategoriju objektu dati.

Metodes:

Tips	Metode	Apraksts
konstruktors	FoodCategory(String food_category, Date updatedAt, String objectId)	Izveido jaunu ēdienu kategorijas objektu.
	FoodCategory()	Nepieciešams OrmLite bibliotēkai.
String	getFood_category ()	Atgriež objekta nosaukumu.
String	getObjectid()	Atgriež objekta unikālo vērtību.

Mainīgie:

Tips	Mainīgais	Apraksts
int	id	Datubāzes unikāls ID.
String	food_category	Ēdienu kategorijas objekta nosaukums.
String	objectId	Ēdienu kategorijas objekta unikālā vērtība, kura automātiski izveidota servisā.

4.6.13. Klase DownloadFoodService

Klase saglabā datubāzē dienas piedāvājumu un ēdienkartes datus.

Metodes:

Tips	Metode	Apraksts
konstruktors	DownloadFoodService()	Ieslēdz servisu.
void	onHandleIntent(Intent intent)	Saņem ziņojumu, ka nepieciešams ielādēt visu ēdienkarti un dienas

		piedāvājumus, ielādē tos un izsauc AddDailyList un AddFoodList funkcijas.
void	AddFoodList(List<ParseObject> object)	Funkcija pievieno ēdienu karti datubāzei.
void	addFoodToDB(JSONArray array, String objectid)	Funkcija pievieno ēdienus datubāzei.
void	AddDailyList(List<ParseObject> object)	Funkcija pievieno dienas piedāvājumus datubāzei.

Mainīgie:

Tips	Mainīgais	Apraksts
DatabaseHelper	databaseHelper	Norāde uz datubāzes piekļuves klasi.
Dao<FoodCategory, Integer>	daoCategory	Norāde uz ēdienu kategoriju tabulu.
Dao<Food, Integer>	daoFood	Norāde uz ēdienu tabulu.

4.6.14. Klase Network

Klase pārbauda vai ir pieejams interneta pieslēgums.

Metodes:

Tips	Metode	Apraksts
static final boolean	isOnline(Context cxt)	Pārbauda vai pieejams interneta savienojums.

4.6.15. Klase TypefaceSpan

Klase nomaina lietotnes virsraksta šriftu.

Metodes:

Tips	Metode	Apraksts
konstruktors	TypefaceSpan(Context context, String typefaceName)	Ielādē šriftu un saglabā to kešatmiņā.
void	updateMeasureState(TextPaint p)	Atjauno izmēru stāvokli.
void	updateDrawState(TextPaint tp)	Atjauno zīmējuma stāvokli.

Mainīgie:

Tips	Mainīgais	Apraksts
Typeface	mTypeface	Šrifts.
static LruCache<String, Typeface>	sTypefaceCache	Kešatmiņa, kur glabāsies šrifts.

4.6.16. Klase App

Klase brīdī, kad lietotne tiek ieslēgta, inicializē visus nepieciešamos objektus.

Metodes:

Tips	Metode	Apraksts
void	onCreate()	Funkcija ielādē šriftu, noskaidro ierīces ID un inicializē Parse servisu.

Mainīgie:

Tips	Mainīgais	Apraksts
Typeface	face	Šrifts.

String	deviceid	Ierīces ID.

4.6.17. Klase C

Klasē nodefinētas visas konstantās vērtības, kuras tiek izmantotas lietotnes darbības laikā.

Mainīgie:

Tips	Mainīgais	Apraksts
static final String	APP_ID	Parse servisa lietotnes ID.
static final String	CLIENT_KEY	Parse servisa klienta ID.
static final String	PARSE_FOOD, PARSE_DAILY, PARSE_ORDER, PARSE_TABLE_ID, PARSE_WAITER.	Parse servisa tabulu nosaukumi.
static final String	PARSE_TYPE, PARSE_GALDA_ID, PARSE_FOOD_NID, PARSE_DEVICE_ID, PARSE_NFC_ID, PARSE_GALDA_NR, FOOD_CATEGORY, PARENT, FOOD_ARRAY, FOOD, FOOD_COST, NID, COST_NOW, DAILY_ID.	Vērtības, kuras tiek izmantotas, lai parsētu iegūtos datus no Parse servisa.
static final String	ACTION_DOWNLOAD _ALL,	Tiek izmantoti, lai nosūtītu paziņojumus starp aktivitāti un

	ACTION_DOWNLOAD_ALL_OK, ACTION_DOWNLOAD_ALL_FAIL	DownloadFoodService

4.6.18. Klase SplashActivity

Sākuma aktivitāte, kura attēlojas, kamēr ielādē, saglabā visus ēdienkartes un dienas piedāvājumu datus.

Metodes:

Tips	Metode	Apraksts
void	onCreate(Bundle savedInstanceState)	Funkcija ielādē sākuma skatu un ieslēdz DownloadFoodService.

Mainīgie:

Tips	Mainīgais	Apraksts
BroadcastReceiver	receiver	Uztvērējs, kurš saņems paziņojumu no DownloadFoodService, ka viss ir veiksmīgi ielādēts.
TextView	loading_text	Norāde uz teksta lauku, kurā attēlo ielādes informāciju.

4.6.19. Klase MainActivity

Galvenā aktivitāte, kura izveido sānu izvēlni, maina skatus, uzliek lietotnes virsraksta šriftu.

Metodes:

Tips	Metode	Apraksts
void	onCreate(Bundle savedInstanceState)	Ielādē skatu.

boolean	onCreateOptionsMenu(Menu menu)	Android automātiski ģenerēta funkcija.
void	SelectItem(int position)	Funkcija pēc pozīcijas nomaina skatus.
void	setTitle(CharSequence title)	Nomaina lietotnes virsraksta tekstu.
void	onPostCreate(Bundle savedInstanceState)	Android automātiski ģenerēta funkcija.
void	onConfigurationChanged(Configuration newConfig)	Android automātiski ģenerēta funkcija.
boolean	onOptionsItemSelected(MenuItem item)	Funkcija atver sānu izvēlni.
void	onBackPressed()	Ja sānu izvēlne nav atvērta, tad atver to, citādi beidz galveno aktivitāti.
class	DrawerItemClickListener	Funkcija, kad tiek nospiests uz sānu izvēlnes izsauc SelectItem funkciju.

Mainīgie:

Tips	**Mainīgais**	**Apraksts**
DrawerLayout	mDrawerLayout	Sānu izvēlnes izskats.
ListView	mDrawerList	Sānu izvēlnes saraksts.
ActionBarDrawerToggle	mDrawerToggle	Poga lietotnes augšējā kreisajā stūrī, kura atver/aizver sānu izvēlni.
MenuDrawerAdapter	adapter	Sānu izvēlnes datu sniedzējs.
FragmentManager	frgManager	Norāde uz fragmentu pārvaldnieku.

4.6.20. Skatu izkārtojumi.

activity_main.xml – satur galvenās aktivitātes izkārtojumu.
activity_splash.xml – satur ielādes aktivitātes izkārtojumu.
custom_drawer_item.xml – satur sānu izvēlnes objekta izkārtojumu.
food_list_fragment.xml – satur ēdienkartes izkārtojumu.
food_list_group.xml – satur ēdienkartes kategorijas izkārtojumu.
food_list_item.xml – satur ēdienkartes kategorijas bērna izkārtojumu.
order_list_item.xml – satur pasūtījumu objekta izkārtojumu.
order_list.xml – satur pasūtījumu izkārtojumu.
single_order_list_footer.xml – satur specifiska pasūtījuma kājenes izkārtojumu.
single_order_list_item.xml – satur specifiska pasūtījuma objekta izkārtojumu.

4.6.21. Iepriekš nodefinētās vērtības.

color.xml – satur krāsu kodus, kurus izmanto lietotnē.
strings.xml – satur lietotnes nosaukumu un izvadziņu tekstus.
styles.xml – satur teksta stilus, kurus izmanto lietotnē.

4.7. Pavāra lietotnes moduļu projektējums

4.7.1. Klase Network

Klase pārbauda vai ir pieejams interneta pieslēgums.

Metodes:

Tips	Metode	Apraksts
static final boolean	isOnline(Context cxt)	Pārbauda vai pieejams interneta savienojums.

4.7.2. Klase TypefaceSpan

Klase nomaina lietotnes virsraksta šriftu.

Metodes:

Tips	Metode	Apraksts
konstruktors	TypefaceSpan(Context context, String typefaceName)	Ielādē šriftu un saglabā to kešatmiņā.
void	updateMeasureState(TextPaint p)	Atjauno izmēru stāvokli.
void	updateDrawState(TextPaint tp)	Atjauno zīmējuma stāvokli.

Mainīgie:

Tips	Mainīgais	Apraksts
Typeface	mTypeface	Šrifts.
static LruCache<String, Typeface>	sTypefaceCache	Kešatmiņa, kur glabāsies šrifts.

4.7.3. Klase App

Klase brīdī, kad lietotne tiek ieslēgta, inicializē visus nepieciešamos objektus.

Metodes:

Tips	Metode	Apraksts
void	onCreate()	Funkcija ielādē šriftu un inicializē Parse servisu.

Mainīgie:

Tips	Mainīgais	Apraksts
Typeface	face	Šrifts.

4.7.4. Klase MainActivity

Galvenā aktivitāte, uzliek lietotnes virsraksta šriftu, iegūst gatavojamo ēdienu datus un attēlo tos sarakstā.

Metodes:

Tips	Metode	Apraksts
void	onCreate(Bundle savedInstanceState)	Funkcija ielādē skatu, izveido cilpu, kura ik pēc 10 sekundēm ielādē apstiprinātos ēdienus.
void	setTitle(CharSequence title)	Uzstāda lietotnes virsrakstu.
void	onDestroy()	Beidz iepriekš izveidoto cilpu.
class	FoodListAdapter	Nodefinēts apstiprināto ēdienu datu sniedzējs.
konstruktors	FoodListAdapter(List<ParseObject> object)	Inicializē apstiprināto ēdienu datu sniedzēju.
int	getCount()	Atgriež apstiprināto ēdienu skaitu.
ParseObject	getItem(int position)	Atgriež apstiprināto ēdienu pēc

		tā pozīcijas.
long	getItemId(int position)	Android automātiski ģenerēta funkcija.
View	getView(final int position, View convertView, ViewGroup parent)	Atgriež apstiprinātā ēdiena skatu pēc tā pozīcijas.

Mainīgie:

Tips	Mainīgais	Apraksts
ParseQuery<ParseObject>	query	Parse servisa apstiprināto ēdienu pieprasījums.
ListView	foods_list	Apstiprināto ēdienu saraksts.
FoodListAdapter	listAdapter	Datu sniedzējs apstiprināto ēdienu sarakstam.
TextView	error	Norāde uz teksta lauku, kur izvada paziņojumus.
Date	last_time_refreshed	Mainīgais glabā datumu, kad pēdējo reizi saraksts atjaunots.
Handler	handleRefreshLoop	Serviss, kurš ik pēc 10 sekundēm izsauc runnableRefresh.
Runnable	runnableRefresh	Izsauc apstiprināto ēdienu pieprasījumu.
List<ParseObject>	child	Saraksts ar apstiprinātajiem ēdieniem.

4.7.5. Skatu izkārtojumi.

activity_main.xml – satur galvenās aktivitātes izkārtojumu.
food_list_item.xml – satur apstiprinātā ēdiena izkārtojumu.

4.7.6. Iepriekš nodefinētās vērtības.

color.xml – satur krāsu kodus, kurus izmanto lietotnē.
strings.xml – satur lietotnes nosaukumu un izvadziņu tekstus.

5. PROJEKTA ORGANIZĀCIJA

Projektā tika izvēlēta spējās programmēšanas (Agile) pieeja (11. avots), jo sākumā nebija definētas programmatūras iespējas. Programmprodukta plānošanas un izstrādes procesa laikā produkts tika atrādīts iespējamajiem lietotājiem un tādējādi tika noskaidroti lietotājstāsti.

Neskatoties uz to, ka spējā izstrāde neparedz, ka lietotājstāstu akcepttestēšanu veic izstrādātājs, tā kā pasūtītāja kā tāda nebija, tad testēšanu veica pats izstrādātājs, ik pēc laika pārbaudot lietotnes ar akcepttestiem un vienībtestiem.

Programmatūra tika izstrādāta pēc Android labās programmēšanas principiem, klases un skatu izkārtojumu nosaukumi tika veidoti pēc iepriekš nodefinētiem principiem, lai citam izstrādātājm pirmo reizi skatoties uz projektu būtu vieglāk to saprast.

Programmatūras izstrādē tika izmantota Android valoda, Parse (5. avots) aizmugursistēma, ORMLite (4. avots) un Universal Image Loader (6. avots) bibliotēkas.

5.1. Konfigurācijas pārvaldība

SVN sistēma tika izmantota, lai kontrolētu konfigurāciju, pielietojot iztrādes vidē Subclipse (14. avots) versijkontroles spraudni.

Stabilai konfigurācijas kontrolei izmantoja SVN Branch (15. avots) metodi, kuras stumbrā atradās strādājoša lietotne, savukārt stumbra zari - papildus lietotnes iespējas - tika vēlāk pievienoti pamatam - stumbram.

5.2. Kvalitātes nodrošināšana

Lai nodrošinātu kvalitāti:

- Izstrāde tika veikta pēc Objektorientētās programmēšanas principiem.
- Komentāru lauki tika izveidoti, izmantojot izstrādes vides JavaDoc spraudni.
- Lietotņu vienībtestēšana tika veikta ne retāk, kā divreiz dienā.
- Izstrādes vidē koda noformatēšanā tika izmantots Android standarts.
- Izmaiņas tika nodotas konfigurācijas pārvaldības sistēmai katras darba dienas beigās.
- Lietotnes tika notestētas uz vairākām ierīcēm un Android versijām.

6. DARBIETILPĪBAS NOVĒRTĒJUMS

Darbietilpības novērtējums tika izstrādāts projekta izpildīšanas sākumā, pamatojoties uz katra lietotājstāsta novērtēšanu, izmantojot Fibonači punktu skalu (10. avots). Ar 2, 3, 5 punktiem tika novērtēti tie lietotājstāsti, kurus izstrādātājs uzskatīja par vieglākiem, 8, 13, 21 punktiem tika novērvērtēti laikietilpīgākie. Kopsummā visi lietotājstāstu sarežģītība tika novērtēta 99 punktos.

Tabulā 6.1. redzams sarežģītības punktu sadalījums pa iterācijām.

6.1. tabula

Sarežģītības punktu sadalījums pa iterācijām

Iterācijas	Sarežģītības punkti
Pirmā	21
Otrā	21
Trešā	20
Ceturtā	19
Piektā	18

Izstrādes ātrums tika pieņemts projekta sākumā - 10 punkti nedēļā, līdz ar to visam projektam ir nepieciešamas apmēram 10 nedēļas. Daži lietotājstāsti neaizņēma tik daudz laika, cik tam bija atvēlēts, tāpēc izstrādes laikā lietotnes tika papildinātas ar papildus iespējām, kuras lietotājstātos nebija pieminētas.

Vidējais izstrādes ātrums bija 10,41 punkti nedēļā - to izstrādātājs aprēķināja projekta beigās, līdz ar to neplānotā brīvā laikā izstrādātājs papildināja lietotnes ar papildus iespējām un rūpīgākai testēšanai.

7. TESTĒŠANAS DOKUMENTĀCIJA

Testēšana tika nodrošināta izmantojot akcepttestus un vienībtestus. Pēc katras reizes, kad programmētājs lietotni papildināja ar jaunām funkcijām, tika veikta pārbaude vai tas nav izmainījis iepriekšējo funkciju izpildi. Vienībtestēšanā tika izmantota Robotium (16. avots) bibliotēka.

7.1. Klienta lietotne.

Klienta lietotnē katram lietotājstāstam tika uzrakstīts akcepttests un arī vienībtests, ja tāds bija iespējams.

Klienta lietotnes pirmā lietotājstāsta „Kā lietotājs, es vēlos redzēt restorāna ēdienkarti." akcepttestu skatīt 7.1.1. tabulā.

7.1.1. tabula

Klienta lietotnes pirmā lietotājstāsta akcepttests

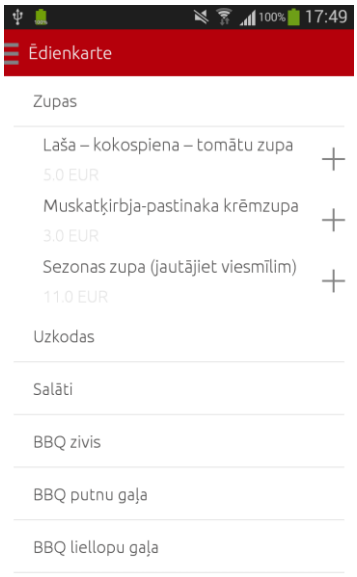
Izpilde:	Gaidāmais rezultāts:
Atveram lietotni.	Lietotne atveras.
Atveram sānu izvēlni.	Attēlojas sānu izvēlne.
Izvēlamies "Ēdienkarte"	Skatā attēlojas ēdienkarte.
Pārbaudam vai attēlo ēdienu kategorijas.	Skatā attēlojas ēdienu kategorijas.
Pārbaudam vai uzspiežot uz ēdiena kategorijas atver sarakstu ar ēdieniem.	Skatā zem ēdienu kategorijas attēlojas tās ēdieni.

Lietotājstāsta vienībtests (skat. 7.1.1. attēlā) izpildījās sekmīgi vidēji 9,950 sekundēs.

```
public void testEdienkarte() throws Exception {
solo.clickOnText("Dienas piedāvājums"); // Atver sānu izvēlni
solo.clickOnText("Ēdienkarte"); // Sānu izvēlnē izvēlas ēdienkarti
solo.clickOnText("Zupas"); // Ēdienkartē uzspiež uz "Zupas"
// Pārbauda vai attēlojas kategorija
// "Zupas" un ēdiens "Sezonas zupa"
boolean categoryAndFoodFound = solo.searchText("Zupas") &&
solo.searchText("Sezonas zupa");
assertTrue("Neuzrādas vai nu kategorija, vai ēdiens zem tās.",
categoryAndFoodFound);
    }
}
```

7.1.1. att. **Klienta lietotnes pirmā lietotājstāsta vienībtests**

Izpildot klienta lietotnes pirmā lietotājstāsta vienībtestu redzama saskarne attēlā 7.1.2.



7.1.2. att. **Klienta lietotnes ēdienkartes saskarne**

Klienta lietotnes otrā lietotājstāsta „Kā lietotājs, es vēlos iespēju pasūtīt ēdienus.” akcepttestu skatīt 7.1.2. tabulā.

7.1.2. tabula

Klienta lietotnes otrā lietotājstāsta akcepttests

Izpilde:	Gaidāmais rezultāts:
Atveram lietotni.	Lietotne atveras.
Atveram sānu izvēlni un izvēlamies "Ēdienkarte"	Attēlojas ēdienkartes skats.
Uzspiežam uz kādas no kategorijām.	Skatā zem ēdienu kategorijas attēlojas tās ēdieni.
Uzspiežam uz kāda no ēdieniem.	Attēlojas specifiskā ēdiena skats.
Nospiežam pogu pasūtīt.	Parādās ziņa, ka ēdiens ir pievienots pasūtījumam.
Atveram sānu izvēlni un izvēlamies "Mans pasūtījums"	Attēlojas pasūtījumu skats.
Pārbaudam vai pasūtījuma sarakstā atrodas pievienotais ēdiens.	Sarakstā attēlojas iepriekš pievienotais ēdiens.
Nospiežam pogu pasūtīt.	Izvadās ziņa, ka jāpievieno telefons pie NFC mikroshēmas.
Pievienojam telefonu NFC mikroshēmai.	Parādās ziņa, ka pasūtījums nosūtīts.

Lietotājstāsta vienībtests (skat. 7.1.3. attēlā) izpildījās sekmīgi vidēji 17,097 sekundēs.

```
public void testPasutijums() throws Exception {
solo.clickOnText("Dienas piedāvājums"); // Atver sānu izvēlni
solo.clickOnText("Ēdienkarte"); // Sānu izvēlnē izvēlas ēdienkarti
solo.clickOnText("Zupas"); // Ēdienkartē uzspiež uz "Zupas"
// Ēdienkartes sarakstā uzspiež uz "Sezonas zupa"
solo.clickOnText("Sezonas zupa");
// Specifiskā ēdiena skatā nospiež uz pogas pasūtīt.
solo.clickOnButton("Pasūtīt");
solo.goBack(); // Dodas atpakaļ uz ēdienkarti.
solo.goBack(); // Atver sānu izvēlni.
// Sānu izvēlnē izvēlas "Mans pasūtījums"
solo.clickOnText("Mans pasūtījums");
// Pārbauda vai pasūtījuma sarakstā ir iepriekš pasūtītais ēdiens.
boolean foodAdded =  solo.searchText("Sezonas zupa");
assertTrue("Ēdiens nepievienojās pasūtījumam.", foodAdded);
}
```

7.1.3. att. **Klienta lietotnes otrā lietotājstāsta vienībtests**

Klienta lietotnes trešā lietotājstāsta „Kā lietotājs, es vēlos iespēju izsaukt viesmīli.” akcepttestu skatīt 7.1.3. tabulā.

7.1.3. tabula

Klienta lietotnes trešā lietotājstāsta akcepttests

Izpilde:	Gaidāmais rezultāts:
Atveram lietotni.	Lietotne atveras.
Atveram sānu izvēlni un izvēlamies "Izsaukt viesmīli"	Izvadās ziņa, ka jāpievieno telefons pie NFC mikroshēmas.
Pievienojam telefonu pie NFC mikroshēmas.	Parādās ziņa, ka viesmīlis ir izsaukts.

Klienta lietotnes ceturtā lietotājstāsta „Kā lietotājs, es vēlos, lai dienas piedāvājumi ir izcelti no ēdienkartes.” akcepttestu skatīt 7.1.4. tabulā.

7.1.4. tabula

Klienta lietotnes ceturtā lietotājstāsta akcepttests

Izpilde:	Gaidāmais rezultāts:
Atveram lietotni.	Lietotne atveras.
Pārbaudam vai attēlojas dienas piedāvājumu skats.	Attēlojas dienas piedāvājumu skats.

Izpildot klienta lietotnes ceturtā lietotājstāsta akcepttestu redzama saskarne attēlā 7.1.4.

7.1.4. att. **Klienta lietotnes dienas piedāvājumu saskarne**

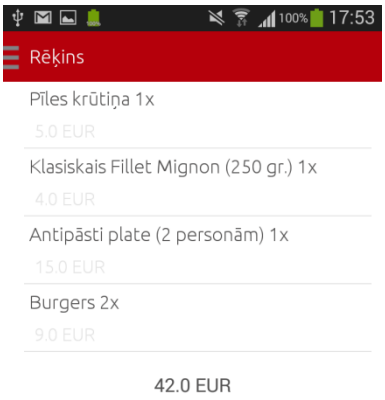
Klienta lietotnes piektā lietotājstāsta „Kā lietotājs, es vēlos iespēju redzēt savu rēķinu.” akcepttestu skatīt 7.1.5. tabulā.

7.1.5. tabula

Klienta lietotnes piektā lietotājstāsta akcepttests

Izpilde:	Gaidāmais rezultāts:
Atveram lietotni.	Lietotne atveras.
Atveram sānu izvēlni un izvēlamies "Rēķins".	Ja iepriekš tika veikts pasūtījums, tad attēlojas saraksts ar ēdieniem, citādi skats ir tukšs.

Izpildot klienta lietotnes piektā lietotājstāsta akcepttestu redzama saskarne attēlā 7.1.5.



7.1.5. att. **Klienta lietotnes rēķina saskarne**

lienta lietotnes sestā lietotājstāsta „Kā lietotājs, es vēlos novērtēt servisu.” akcepttestu skatīt 7.1.6. tabulā.

7.1.6. tabula

Klienta lietotnes sestā lietotājstāsta akcepttests

Izpilde:	Gaidāmais rezultāts:
Atveram lietotni.	Lietotne atveras.
Atveram sānu izvēlni un izvēlamies "Novērtēt servisu".	Attēlojas servisa novērtēšanas skats.
Novērtējuma laukā ievadam tekstu un nospiežam pogu "Novērtēt".	Parādās ziņa "Paldies par vērtējumu.".

Lietotājstāsta vienībtests (skat. 7.1.6. attēlā) izpildījās sekmīgi vidēji 8,555 sekundēs.

```
public void testVertejums() throws Exception {
solo.clickOnText("Dienas piedāvājums"); // Atver sānu izvēlni.
// Sānu izvēlnē izvēlas novērtēt servisu
solo.clickOnText("Novērtēt servisu");
// Ievada novērtējumu.
solo.enterText(0, "Jūsu serviss ir ļoti ērts.");
solo.clickOnButton("Novērtēt"); // Nospiež pogu novērtēt.
// Pārbauda vai parādās ziņa.
boolean ratingSent =  solo.searchText("Paldies par vērtējumu");
assertTrue("Ēdiens nepievienojās pasūtījumam.", ratingSent);
}
```

7.1.6. att. **Klienta lietotnes sestā lietotājstāsta vienībtests**

7.2. Viesmīļa lietotne.

Viesmīļa lietotnē katram lietotājstāstam tika uzrakstīts akcepttests un arī vienībtests, ja tāds bija iespējams.

Viesmīļa lietotnes pirmā lietotājstāsta „Kā lietotājs, es vēlos redzēt un apstiprināt pasūtītos ēdienus.” akcepttestu skatīt 7.2.1. tabulā.

7.2.1. tabula

Viesmīļa lietotnes pirmā lietotājstāsta akcepttests

Izpilde:	Gaidāmais rezultāts:
Atveram lietotni.	Lietotne atveras un parādas pasūtījumu skats.
Atveram kādu no saraksta objektiem.	Atveras specifiskā pasūtījuma skats, kur attēlojas visi pasūtītie ēdieni.
Nospiežam pogu "Apstiprināt".	Izvadās ziņa, ka pasūtījums apstiprināts.

Lietotājstāsta vienībtests (skat. 7.2.1. attēlā) izpildījās sekmīgi vidēji 7,420 sekundēs.

```
public void testApstiprinatPasutijumu() throws Exception {
      // Tā kā pasūtījuma skats atveras pirmais, tad izvēlamies
pirmo pasūtījumu.
      solo.clickInList(1);
      // Nospiežam pogu.
      solo.clickOnButton("Apstiprināt");
      // Pārbaudām vai izvadās ziņa "Pasūtījums pieņemts!"
      boolean orderAccepted =  solo.searchText("Pasūtījums
pieņemts!");
      assertTrue("Pasūtījums netika apstiprināts.", orderAccepted);
      }
```

7.2.1. att. **Viesmīļa lietotnes pirmā lietotājstāsta vienībtests**

Viesmīļa lietotnes otrā lietotājstāsta „Kā lietotājs, es vēlos redzēt pagatavotos ēdienus.” akcepttestu skatīt 7.2.2. tabulā.

7.2.2. tabula

Viesmīļa lietotnes otrā lietotājstāsta akcepttests

Izpilde:	Gaidāmais rezultāts:
Atveram lietotni.	Lietotne atveras.
Atveram sānu izvēlni un izvēlamies "Pagatavotie ēdieni"	Atveras pagatavoto ēdienu skats.
Izvēlamies kādu no galdiņiem.	Atveras skats, kur attēlojas izvēlētā galdiņa pagatavotie ēdieni.

Viesmīļa lietotnes trešā lietotājstāsta „Kā lietotājs, es vēlos redzēt katra galdiņa rēķinu.” akcepttestu skatīt 7.2.3. tabulā.

7.2.3. tabula

Viesmīļa lietotnes trešā lietotājstāsta akcepttests

Izpilde:	Gaidāmais rezultāts:
Atveram lietotni.	Lietotne atveras.
Atveram sānu izvēlni un izvēlamies "Rēķini"	Atveras rēķinu skats.
Izvēlamies kādu no galdiņiem.	Atveras skats, kur attēlojas izvēlētā galdiņa rēķins.

Viesmīļa lietotnes ceturtā lietotājstāsta „Kā lietotājs, es vēlos redzēt galdiņus, kuri izsaukuši viesmīli.” akcepttestu skatīt 7.2.4. tabulā.

7.2.4. tabula

Viesmīļa lietotnes ceturtā lietotājstāsta akcepttests

Izpilde:	Gaidāmais rezultāts:
Atveram lietotni.	Lietotne atveras.
Atveram sānu izvēlni un izvēlamies "Izsauca viesmīli"	Atveras skats, kur attēlojas saraksts ar galdiņiem, kuri izsaukuši viesmīli.

Viesmīļa lietotnes piektā lietotājstāsta „Kā lietotājs, es vēlos pievienot ēdienus pasūtījumam.” akcepttestu skatīt 7.2.5. tabulā.

7.2.5. tabula

Viesmīļa lietotnes piektā lietotājstāsta akcepttests

Izpilde:	Gaidāmais rezultāts:
Atveram lietotni.	Lietotne atveras.
Atveram kādu no saraksta objektiem.	Atveras specifiskā pasūtījuma skats, kur attēlojas visi pasūtītie ēdieni.
Nospiežam pogu "Pievienot ēdienu"	Atveras ēdienkartes skats.
Atveram kādu no kategorijām un pievienojam kādu no ēdieniem.	Izvadās ziņa, ka ēdiens veiksmīgi pievienots.
Dodamies atpakaļ un pārbaudam vai pasūtījuma sarakstā atrodas pievienotais ēdiens.	Pasūtījuma sarakstā ir klāt nācis pievienotais ēdiens.

Lietotājstāsta vienībtests (skat. 7.2.2. attēlā) izpildījās sekmīgi vidēji 10,466 sekundēs.

```
public void testPievienotEdienu() throws Exception {
solo.clickInList(1); // Atver pirmo pasūtījumu.
solo.clickOnButton("Pievienot ēdienu"); // Nospiež pogu.
solo.clickInList(4); // Atver 4. ēdienu kategoriju.
// Nospiež uz pirmā attēla, šajā gadījumā tas ir atvērtās ēdienu
kategorijas, pirmā ēdiena pievienošanas poga.
solo.clickOnImage(0);
// Pārbauda vai izvadās ziņa.
boolean foodAdded = solo.searchText("pievienots pasūtījumam");
assertTrue("Ēdiens netika pievienots.", foodAdded);
solo.goBack(); // Dodas atpakaļ uz pasūtījuma skatu.
// Pārbauda vai ēdiens pievienots.
foodAdded = solo.searchText("Bresaolas un mozzarellas siera
salāti");
assertTrue("Ēdiens netika pievienots.", foodAdded);
}
```

7.2.2. att. **Viesmīļa lietotnes piektā lietotājstāsta vienībtests**

Viesmīļa lietotnes sestā lietotājstāsta „Kā lietotājs, es vēlos noņemt ēdienus no rēķina.” akcepttestu skatīt 7.2.6. tabulā.

7.2.6. tabula

Viesmīļa lietotnes sestā lietotājstāsta akcepttests

Izpilde:	Gaidāmais rezultāts:
Atveram lietotni.	Lietotne atveras.
Atveram sānu izvēlni un izvēlamies "Rēķini"	Atveras rēķinu skats.
Izvēlamies kādu no galdiņiem.	Atveras skats, kur attēlojas izvēlētā galdiņa rēķins.
Noņemam kādu no ēdieniem.	Ēdiens vairs neuzrādās rēķina sarakstā.

Lietotājstāsta vienībtests (skat. 7.2.3. attēlā) izpildījās sekmīgi vidēji 14,320 sekundēs.

```
public void testNonemtEdienu() throws Exception {
solo.clickOnText("Pasūtījumi"); // Atver sānu izvēlni
solo.clickOnText("Rēķini"); // Izvēlas "Rēķini"
solo.clickInList(1); // Atver pirmo rēķinu
// Pārbauda vai ēdiens ir rēķinā.
boolean foodInBill = solo.searchText("Bresaolas un mozzarellas siera
salāti");
assertTrue("Ēdiens nav rēķinā.", foodInBill);
solo.clickOnImage(0); // Pirmajam ēdienam uzspiež "Noņemt" pogu.
solo.clickOnText("Jā"); // Piekrīt, ka vēlas noņemt.
// Pārbauda vai ēdiens vēl ir rēķinā.
foodInBill = !solo.searchText("Bresaolas un mozzarellas siera
salāti");
assertTrue("Ēdiens ir rēķinā.", foodInBill);
}
```

7.2.3. att. **Viesmīļa lietotnes sestā lietotājstāsta vienībtests**

7.3. Pavāra lietotne.

Pavāra lietotnē katram lietotājstāstam tika uzrakstīts akcepttests un arī vienībtests, ja tāds bija iespējams.

Pavāra lietotnes pirmā lietotājstāsta „Kā lietotājs, es vēlos redzēt apstiprinātos ēdienus.” akcepttestu skatīt 7.3.1. tabulā.

7.3.1. tabula

Pavāra lietotnes pirmā lietotājstāsta akcepttests

Izpilde:	Gaidāmais rezultāts:
Atveram lietotni.	Lietotne atveras un parādās saraksts ar apstiprinātajiem ēdieniem.

Pavāra lietotnes otrā lietotājstāsta „Kā lietotājs, es vēlos, lai ēdieni tiktu sakārtoti augošā secībā pēc laika.” akcepttestu skatīt 7.3.2. tabulā.

7.3.2. tabula

Pavāra lietotnes otrā lietotājstāsta akcepttests

Izpilde:	Gaidāmais rezultāts:
Atveram lietotni.	Lietotne atveras un parādās saraksts ar apstiprinātajiem ēdieniem.
Pārbaudam vai ēdieni tiek sakārtoti pēc laika augošā secībā.	Katram ēdienam tiek attēlots laiks un tie ir sakārtoti augošā secībā.

SECINĀJUMI

Lai gan iepriekš autoram bija tikai neliela pieredze Android lietotņu izstrādē, visas lietotājstāstu prasības tika izpildītas un palika vēl laiks, lai pievienotu papildus iespējas. Autoram zināmu laika daļu aizņēma ORMlite un Parse dokumentāciju izpēte, lai izstrādes laikā nebūtu jāmeklē informācija par to iebūvētajām metodēm.

Izstrādes laikā autors konstatēja, ka Android programmēšanas valodai ir ļoti daudz materiālu un bibliotēku, kuras atvieglo izstrādātāja darbu. Liela daļa no tām ir atklātā pirmkoda programmatūra tādējādi dodot izstrādātājam iespēju tās pielāgot specifisku uzdevumu risināšanai.

Nākotnē autors plāno izveidot citu aizmugursistēmu, lai būtu iespējams lietotnēs pievienot jaunas iespējas, un piedāvāt šo lietotņu kopu restorāniem nedaudz izmainot lietotņu dizainu un specifikāciju.

IZMANTOTĀ LITERATŪRA

1. Android Developer. [tiešsaiste]. [Skatīts 04.02.2014]. Pieejams: http://developer.android.com/reference/packages.html
2. How to Set a Custom Font in the ActionBar Title? [tiešsaiste]. [Skatīts 02.14.2014]. Pieejams: http://stackoverflow.com/questions/8607707/how-to-set-a-custom-font-in-the-actionbar-title
3. How to check internet access on Android? [tiešsaiste]. [Skatīts 02.10.2014]. Pieejams: http://stackoverflow.com/questions/1560788/how-to-check-internet-access-on-android-inetaddress-never-timeouts
4. OrmLite.[tiešsaiste]. [Skatīts 02.21.2014]. Pieejams: http://ormlite.com/
5. Parse Android Guide.[tiešsaiste]. [Skatīts 02.17.2014]. Pieejams: https://parse.com/docs/android_guide
6. AndroidUniversalImageLoader. [tiešsaiste]. [Skatīts 03.02.2014]. Pieejams: https://github.com/nostra13/Android-Universal-Image-Loader
7. StaggeredView. [tiešsaiste]. [Skatīts 02.16.2014]. Pieejams: https://github.com/kiteflo/Android_StaggeredView
8. Reading NFC Tags with Android. [tiešsaiste]. [Skatīts 03.14.2014]. Pieejams: http://code.tutsplus.com/tutorials/reading-nfc-tags-with-android--mobile-17278
9. SimpleMod rīks. [tiešsaiste]. [Skatīts 03.02.2014]. Pieejams: http://estudijas.lu.lv/pluginfile.php/291638/mod_page/content/35/SimpleMod.ppt
10. Darbietilpības prognozēšana [tiešsaiste]. [Skatīts 02.15.2014]. Pieejams: http://estudijas.lu.lv/pluginfile.php/258971/mod_resource/content/1/Darbietilpiibas%20prognozeeshana%20-%20Liva%20Steinberga%20-%2029%2010%202012.pdf
11. Spējā izstrāde [tiešsaiste]. [Skatīts 02.14.2014]. Pieejams: http://estudijas.lu.lv/mod/page/view.php?id=128545
12. ObjectAid UML Explorer for Eclipse [tiešsaiste]. [Skatīts 05.12.2014]. Pieejams: http://www.objectaid.com/
13. Parse [tiešsaiste]. [Skatīts 02.26.2014]. Pieejams: https://parse.com/
14. Subclipse [tiešsaiste]. [Skatīts 02.16.2014]. Pieejams: http://subclipse.tigris.org/
15. SVN trunk, branches and tags [tiešsaiste]. [Skatīts 02.16.2014]. Pieejams: http://blog.jmfeurprier.com/2010/02/08/svn-trunk-branches-and-tags/
16. Robotium [tiešsaiste]. [Skatīts 03.12.2014]. Pieejams: https://code.google.com/p/robotium/

PIELIKUMS

Klase DatabaseFoodCache

```
public class DatabaseFoodCache {
/**
    * Objekts, kurš izsauc onDownloaded() un setLoadingText funkcijas iekš
    * SplashActivity klases.
    */
public interface DownloadCallBack {
/**
        * Tiek izsaukta, kad viss ir ielādēts.
        */
public void onDownloaded();
/**
        * Maina ielādes tekstu.
        *
        * @param teksts
        */
public void setLoadingText(String text);
   };

private DownloadCallBack callback;
/** Datubāzes objekts ar kuru iegūst datus. */
private DatabaseHelper databaseHelper = null;
/** Ēdiena kategoriju datubāzes objekts. */
private Dao<FoodCategory, Integer>daoCategory;
/** Ēdienu datubāzes objekts. */
private Dao<Food, Integer>daoFood;
/** Dienas piedāvājumu datubāzes objekts. */
private Dao<DailyFood, Integer>daoDaily;
/** Ēdiena kategoriju datubāzes atjaunošanas objekts. */
private UpdateBuilder<FoodCategory, Integer>updateCategoryBuilder;
/** Dienas piedāvājumu datubāzes dzēšanas objekts. */
private DeleteBuilder<DailyFood, Integer>removeDailyBuilder;
/** Ēdienu datubāzes dzēšanas objekts. */
private DeleteBuilder<Food, Integer>removeFoodBuilder;
/** Ēdiena kategoriju datubāzes dzēšanas objekts. */
private DeleteBuilder<FoodCategory, Integer>removeCategoryBuilder;
/** Masīvs ar bilžu adresēm. */
private ArrayList<String>foodpictures;
/**
    * Inicializē jaunu DatabaseFoodCache
    * @param aktivitātes konteksts
    */
public DatabaseFoodCache(Context ctx) {
databaseHelper = new DatabaseHelper(ctx);
```

```
callback = (DownloadCallBack)(Activity)ctx;
foodpictures = new ArrayList<String>();
   }
/**
    * Funkcija, kura pievieno ēdienkarti datubāzei.
    * @param saraksts ar ēdieniem un kategorijām
    */
public void AddFoodList(List<ParseObject> object) throws SQLException, JSONException {
       HashMap<String, Date> category_list = new HashMap<String, Date>();
daoCategory = databaseHelper.getFoodCategoryDao();
daoFood = databaseHelper.getFoodDao();
// Saglabā visu objektu ObjectID un UpdateAt iekš HashMap, lai datubāzei
// jāslēdzas klāt tikai create, update, remove gadījumā.
       List<FoodCategory> db_category_list =
databaseHelper.getFoodCategoryDao().queryForAll();
for (int i = 0; i < db_category_list.size(); i++) {
          category_list.put(db_category_list.get(i).getObjectid(), db_category_list.get(i)
                .getUpdatedAt());
       }
// Apstaigā visus saņemtos objektus (ēdienu kategorijas)
for (int i = 0; i < object.size(); i++) {
          JSONArray food_list = object.get(i).getJSONArray(C.FOOD_ARRAY);
          ParseObject singleobj = object.get(i);

          FoodCategory category = new
FoodCategory(singleobj.getString(C.FOOD_CATEGORY),
                singleobj.getUpdatedAt(), singleobj.getObjectId());

// Ja pievienotais objekts (ēdienu kategorija) nav datubāzē, tas
// tiek pievienots, kā arī tā bērni (ēdieni)
if (!category_list.containsKey(category.getObjectid())) {
daoCategory.create(category);

if (food_list != null)
                addFoodToDB(food_list, category.getObjectid());

             category_list.remove(category.getObjectid());
             App.log("Pievienoja kategoriju " + category.getFood_category() + " !");
// Ja pievienotais objekts (ēdienu kategorija) jau ir datubāzē,
// bet tā iepriekš
// atjaunotais laiks ir vecāks par tagad iegūto objektu, tas
// tiek atjaunots, kā arī tā bērni (ēdieni)
          } elseif (category_list.containsKey(category.getObjectid())
&& category_list.get(category.getObjectid()).before(singleobj.getUpdatedAt())) {
updateCategoryBuilder = daoCategory.updateBuilder();
updateCategoryBuilder.where().eq(C.OBJECTID, singleobj.getObjectId());
updateCategoryBuilder.updateColumnValue(C.FOOD_CATEGORY,
                   singleobj.getString(C.FOOD_CATEGORY));
```

```
updateCategoryBuilder.updateColumnValue(C.UPDATEDAT, singleobj.getUpdatedAt());
updateCategoryBuilder.update();

removeFoodBuilder = daoFood.deleteBuilder();
removeFoodBuilder.where().eq(C.PARENT, category.getObjectid());
removeFoodBuilder.delete();

if (food_list != null)
                addFoodToDB(food_list, category.getObjectid());

            category_list.remove(category.getObjectid());
            App.log("Atjaunoja kategoriju " + category.getFood_category() + " !");
          } elseif (category_list.containsKey(category.getObjectid())
&& category_list.get(category.getObjectid()).equals(singleobj.getUpdatedAt())) {
            category_list.remove(category.getObjectid());
            App.log("Kategorija " + category.getFood_category() + " nav mainījusies!");
          }

      }

// Ja iekš HashMap ir pāri palikuši "key", tad jāizdzēš objekti (ēdienu
// kategorijas), kuri
// serverī noņemti no saraksta un šī objekta bērni (ēdieni)
if (category_list.size() > 0) {
removeFoodBuilder = daoFood.deleteBuilder();
removeCategoryBuilder = daoCategory.deleteBuilder();

for (String s : category_list.keySet()) {
              App.log("Izdzēsa objektu ar ID " + s + " !");
removeCategoryBuilder.where().eq(C.OBJECTID, s);
removeCategoryBuilder.delete();
removeFoodBuilder.where().eq(C.PARENT, s);
removeFoodBuilder.delete();
          }
      }
new DownloadImage().execute(foodpictures);

   }
/**
    * Pievieno datubāzē ēdienus katrai kategorijai.
    * @param JSON masīvs ar ēdieniem
    * @param vecāka ID
    */
private void addFoodToDB(JSONArray array, String objectid) throws JSONException,
SQLException {
for (int i = 0; i < array.length(); i++) {
          JSONObject food_item = array.getJSONObject(i);
```

```
daoFood = databaseHelper.getFoodDao();
        Food food = new Food(food_item.getString(C.FOOD),
food_item.getString(C.FOOD_ABOUT),
                food_item.getDouble(C.FOOD_COST), food_item.getInt(C.NID),
                food_item.getString(C.FOOD_PICTURE), objectid);
daoFood.create(food);
foodpictures.add(food.getPicture());

        App.log("Pievienoja ēdienu: " + food.getFood());
      }
   }
/**
    * Pievieno datubāzē dienas piedāvājumus.
    * @param saraksts ar dienas piedāvājumu objektiem
    */
public void AddDailyList(List<ParseObject> object) throws SQLException {
      HashMap<String, Date> daily_list = new HashMap<String, Date>();
daoDaily = databaseHelper.getDailyFoodDao();

// Saglabā visu objektu ObjectID un UpdateAt iekš HashMap, lai datubāzei
// jāslēdzas klāt tikai create, update, remove gadījumā.
      List<DailyFood> db_daily_list = databaseHelper.getDailyFoodDao().queryForAll();
for (int i = 0; i < db_daily_list.size(); i++) {
        daily_list.put(db_daily_list.get(i).getObjectid(), db_daily_list.get(i).getUpdatedAt());
      }

for (int i = 0; i < object.size(); i++) {
        ParseObject singleobj = object.get(i);

        DailyFood daily = new DailyFood(singleobj.getString(C.FOOD),
            singleobj.getString(C.FOOD_ABOUT), Double.parseDouble(singleobj
                .getString(C.COST_NOW)), Double.parseDouble(singleobj
                .getString(C.COST_BEFORE)),
            Integer.parseInt(singleobj.getString(C.NID)),
            singleobj.getString(C.FOOD_PICTURE), singleobj.getObjectId(),
            singleobj.getUpdatedAt());

// Ja pievienotais objekts (dienas piedāvājums) nav datubāzē, tas
// tiek pievienots
if (!daily_list.containsKey(daily.getObjectid())) {
daoDaily.create(daily);

foodpictures.add(daily.getPicture());
            daily_list.remove(daily.getObjectid());
            App.log("Pievienoja dienas piedāvājumu " + daily.getFood() + " !");
```

```
// Ja pievienotais objekts (dienas piedāvājums) jau ir datubāzē,
// bet tā iepriekš
// atjaunotais laiks ir vecāks par tagad iegūto objektu, tas
// tiek atjaunots
        } elseif (daily_list.containsKey(daily.getObjectid())
&& daily_list.get(daily.getObjectid()).before(singleobj.getUpdatedAt())) {
removeDailyBuilder = daoDaily.deleteBuilder();
removeDailyBuilder.where().eq(C.OBJECTID, daily.getObjectid());
removeDailyBuilder.delete();
daoDaily.create(daily);
foodpictures.add(daily.getPicture());

            daily_list.remove(daily.getObjectid());
            App.log("Atjaunoja dienas piedāvājumu " + daily.getFood() + " !");
        } elseif (daily_list.containsKey(daily.getObjectid())
&& daily_list.get(daily.getObjectid()).equals(singleobj.getUpdatedAt())) {
            daily_list.remove(daily.getObjectid());
            App.log("Dienas piedāvājums " + daily.getFood() + " nav mainījies!");
        }
      }
// Ja iekš HashMap ir pāri palikuši "key", tad jāizdzēš objekti (ēdienu
// kategorijas), kuri
// serverī noņemti no saraksta
if (daily_list.size() > 0) {
removeDailyBuilder = daoDaily.deleteBuilder();
for (String s : daily_list.keySet()) {
            App.log("Izdzēsa objektu ar ID " + s + " !");
removeDailyBuilder.where().eq(C.OBJECTID, s);
removeDailyBuilder.delete();
        }
      }
   }
/**
    * Ielādē asinhroni attēlus un SplashActivity klasē maina ielādes tekstu.
    */
public class DownloadImage extends AsyncTask<ArrayList<String>, Void, Void> {
@Override
protected Void doInBackground(ArrayList<String>... params) {
        List<String> pictures = params[0];
for (int i = 0; i < pictures.size(); i++) {
            ImageLoader.getInstance().loadImageSync(pictures.get(i));
callback.setLoadingText("Ielādē attēlus " + i + " / " + pictures.size());
        }
callback.onDownloaded();
return null;
      }
   }
}
```

Kvalifikācijas darbs „Android lietotne restorāniem” izstrādāts Latvijas Universitātes Datorikas fakultātē.

Ar savu parakstu apliecinu, ka darbs izstrādāts patstāvīgi, izmantoti tikai tajā norādītie informācijas avoti un iesniegtā darba elektroniskā kopija atbilst izdrukai.

Autors: *Mārtiņš Andersons* _____ ___.05.2014.

Rekomendēju darbu aizstāvēšanai

Darba vadītājs*: Dr. dat. Uldis Straujums*_____ ___.05.2014.

Recenzents: *M.dat. Aleksandrs Zeļenkovs*

Darbs iesniegts 02.06.2014.

Kvalifikācijas darbu pārbaudījumu komisijas sekretārs: *Imants Gorbāns* _____

Darbs aizstāvēts kvalifikācijas darbu pārbaudījuma komisijas sēdē

___.06.2014. prot. Nr. _____

Komisijas sekretārs(-e): _____