LATVIJAS UNIVERSITĀTE
DATORIKAS FAKULTĀTE

SOCIĀLĀ PORTĀLA EXS.LV LIETOTNE ANDROID OPERĒTĀJSISTĒMAI

KVALIFIKĀCIJAS DARBS

Autors: Edgars Peļņa
Studenta apliecības Nr.: ep12028
Darba vadītājs: Dr. dat. Guntis Arnicāns

RĪGA 2014

ANOTĀCIJA

Kvalifikācijas darba mērķis ir sociālā portāla exs.lv lietotnes izstrāde Android operētājsistēmai. Pēdējos gados lapai aizvien vairāk lietotāju slēdzas klāt tieši no mobilajām ierīcēm, tāpat daudzi lietotāji ir izteikuši vēlmi redzēt lapas lietotni, kas būtu paredzēta Android operētājsistēmai. Ņemot vērā Android ierīču popularitāti to vidū, šī ir iespēja izstrādāt produktu, kas tiktu aktīvi izmantots.

Atslēgvārdi: exs.lv, Android, Java, Agile, lietotne

ABSTRACT

The goal of the Qualification work is to develop an Android application for exs.lv social portal. In the last years, more and more users have started connecting to the portal from mobile devices, and many have expressed a wish for an application for exs.lv that was aimed at Android operating system. Taking into account how popular Android devices are in the midst of them, this is an opportunity to develop a product that many would use.

Keywords: exs.lv, Android, Java, Agile, application

SATURS

IEVADS

No sākotnējiem pāris desmitiem līdz pāris desmitiem tūkstošu lietotāju portāls izaudzis deviņu gadu laikā. Pirmssākumi meklējami tālajā 2005. gadā, kad mājaslapā, kas tolaik slēpās zem runescape.ex.lv domēnvārda, cilvēki aktīvi diskutēja par RuneScape spēlīti. Lietotājiem augot un gadiem ejot, saturs ievērojami mainījies un, attīstoties pašai mājaslapai, pulcējis kopā jauniešus, kas nu jau diskutē par visdažādākajām tēmām.

Mājaslapai attīstoties, parādījušās aizvien jaunas iespējas, kā ievietot kādu ziņu un komentēt citu cilvēku veidotus rakstus. Lapā vēl bez vienkāršiem rakstiem un pamācībām forumā un citās sadaļās ir atrodami arī miniblogi, attēlu galerijas, domubiedru grupas u.c.

Zinot, ka lapas saturs nav ierobežots pēc „draudzēšanās" principa, kā daudzos citos sociālajos portālos, bet gan katram lietotājam redzams teju viss pievienotais, kā arī satura pievienošana ir gana dinamiska, lai būtu ērta arī no mobilajām ierīcēm, cilvēki sākuši lapu izmantot arī tāpēc, lai ziņotu, kas ar viņiem notiek attiecīgajā brīdī. Redzot plašo mājaslapas izmantošanu no mobilajām ierīcēm un vēlmi socializēties, ir acīmredzama nepieciešamība arī pēc mobilās lietotnes, kas to atvieglotu vēl vairāk.

Darba mērķis ir izveidot Android lietotni ar lietotājiem visvilinošāko funkcionalitāti, kas jau eksistē mājaslapā. Galvenokārt tie būtu raksti un tā saucamie miniblogi, kurus mājaslapas apmeklētāji jau šobrīd izmanto visvairāk. Pēc spējās izstrādes pieejas lietotne ar laiku tiks uzlabota un piedāvās arvien vairāk iespēju, balstoties uz lietotnes lietotāju vēlmēm un viedokļiem.

APZĪMĒJUMU SARAKSTS

Šeit uzskaitīti kvalifikācijas darba sadaļās pieminētie apzīmējumi, kuru nozīme var nebūt acumirklī skaidra.

Aktivitāte – lietotnes redzamajā daļā ietilpstošo *fragmentu* konteineris. Vienā lietotnē var būt daudz aktivitāšu, starp kurām pārslēgties.
Android – Google Inc. izstrādāta operētājsistēma skārienjutīgām mobilajām ierīcēm.
Android Honeycomb – Android operētājsistēmas versija, kas izlaista 2011. gada februārī.
Android lietotne – programma, kas paredzēta lietošanai Android operētājsistēmā.
Android SDK („Software development kit") – rīku kopums, kas nepieciešams Android lietotņu izstrādei, kompilēšanai, atkļūdošanai u.tml. To nodrošina Google.
Android skats – katrs *Aktivitātes* saskarnes elements, piemēram, teksta lauks vai attēls.
API level 11 – Android operētājsistēmas versija ar nosaukumu *„Honeycomb".*
Commit – programmā veikto izmaiņu apstiprināšana *versiju kontroles sistēmā*.
Fragments – dinamiska Android lietotnes *aktivitāšu* sastāvdaļa, kuru aktivitātē programmas izpildes laikā var gan ievietot, gan dzēst, gan aizvietot ar kādu citu fragmentu. Tajā atrodas galvenais redzamais saturs.
GIT – plaši pazīstama *versiju kontroles sistēma*.
Ietvars – iepriekš definētu iespēju kopums ar mērķi piedāvāt izstrādātājiem jau gatavas funkcijas, izvairoties no nepieciešamības tās atkārtoti realizēt no nulles.
Izklājumi – faili, kuros *XML* formātā aprakstīta kāda *Android skata* struktūra.
JSON („JavaScript Object Notation") – plaši pazīstams formāts, kādā strukturēt datus un informāciju, lai tā būtu gan viegli lasāma cilvēkiem, gan ērti ielasāma dažādās programmās.
Kešs/kešatmiņa – atmiņa, kurā parasti tiek noglabāta bieži pieprasīta informācija tās ātrai nolasīšanai.
Key-value – formāts, kādā glabāt vienkāršu, nelielu informāciju. Sastāv no atslēgas („key"), caur kuru atsaukties uz attiecīgo informāciju, un atslēgas vērtības jeb glabājamās informācijas. Piemēram, *„vārds":"Edgars";"uzvārds":"Peļņa";*
Lietotājstāsts – pāris teikumos aprakstīta programmizstrādes produkta funkcija ērti saprotamā ikdienas valodā, pieņemot, ka lietotājam nav tehnisku zināšanu.

Miniblogs – exs.lv mājaslapai specifisks bloga formāts ar parasti salīdzinoši īsu saturu. Ar miniblogu saprotams gan pats miniblogs, gan tam pievienotie komentāri, kuriem ir tieši tāds pats izskats.

MVC modelis (*„Model-View-Controller”)* – šablons efektīvai programmizstrādes produkta saskarnes sasaistīšanai ar iekšējiem datu modeļiem (klasēm, resursu failiem, datubāzi u.c.).

NavigationDrawer – Android definēts aktivitātes elements, kas ar pavilkšanas kustību „izvelkams” no aktivitātes sāna un ļauj pārvietoties uz citām lietotnes aktivitātēm.

PPA – Programmatūras projektējuma apraksts.

PPS – Programmatūras prasību specifikācija.

Repozitorijs – projekta failu uzglabāšanas vieta *versiju kontroles sistēmā*.

SharedPreferences – faili, kuros iespējams glabāt *key-value* formāta datus.

Spējā izstrāde – programmizstrādes metodoloģija, kurā prasības un projektējums attīstās laika gaitā, izstrādātājiem un pasūtītājam savstarpēji sadarbojoties un regulāri apspriežoties.

Thread – programmas daļa, kas savu darbību veic paralēli un neatkarīgi no citām daļām, tās neietekmējot.

Versiju kontroles sistēma – sistēma, kas pārvalda projektā veiktās izmaiņas, glabā to versijas un ļauj vienlaicīgi pie produkta strādāt daudziem izstrādātājiem.

XML („Extensible Markup Language”) – plaši pazīstams formāts, kādā strukturēt datus un informāciju, lai tā būtu gan viegli lasāma cilvēkiem, gan ērti ielasāma dažādās programmās.

Žurnālfaili – faili, kuros secīgi uzskaitītas veiktās darbības, kas palīdz izsekot tam, kā strādā sistēma. Ieraksti žurnālfailos atvieglo sistēmas atkļūdošanu.

1 PROGRAMMATŪRAS PRASĪBU SPECIFIKĀCIJA

1.1 Ievads

1.1.1 Nolūks

Programmatūras prasību specifikācijas (PPS) nolūks ir aprakstīt exs.lv sociālā portāla Android lietotnes prasības. Dokuments apraksta to, kas ir jārealizē, nespecificējot, kā tas ir jārealizē. Realizācija aprakstīta kvalifikācijas darbā iekļautajā dokumentā „Programmatūras projektējuma apraksts".

Dokuments ir paredzēts lietotnes izstrādātājam un pasūtītājam.

1.1.2 Darbības sfēra

Šī lietotne paredzēta lietotājiem, kas aktīvi izmanto exs.lv sociālo portālu un vēlas tā saturam ērti piekļūt arī no mobilās ierīces, kas aprīkota ar Android operētājsistēmu.

Ņemot vērā portāla visai sarežģīto struktūru ar lielo funkcionalitāšu klāstu, sākotnēji lietotne nebūs pilnvērtīgā mājaslapas aizstājēja, bet vairāk kā palīgs būtiskā satura piekļūšanai no mobilās ierīces. Sākotnēji tā paredzēta jau reģistrētiem lietotājiem.

1.1.3 Saistība ar citiem dokumentiem

Šis dokuments ir kvalifikācijas darba „Sociālā portāla exs.lv lietotne Android operētājsistēmai" sastāvdaļa un tādējādi izmantojams kopā ar to.

PPS sastādīšanā izmantots standarts LVS 68:1996 „Programmatūras prasību specifikācijas ceļvedis", tādējādi garantējot to, ka dokuments atbilst noteiktai struktūrai.

PPS izmantojams kopā ar Programmatūras projektējuma aprakstu (PPA), kurā minētie punkti atsaucas uz šajā dokumentā definētajām prasībām.

1.2 Vispārējais apraksts

1.2.1 Produkta perspektīva

Lietotnei sākotnēji jānodrošina tikai tā funkcionalitāte, kas visbiežāk tiek izmantota mājaslapā. Tā tiek izstrādāta ar domu nākotnē spēt ērti pielāgot saturu arī lielizmēra ierīcēm, kas aprīkotas ar Android operētājsistēmu (piemēram, planšetdatoriem). Būtiski lietotnē parādīt to saturu, kas lietotājam jāredz, lai tas nejustos palaidis garām galveno.

Tā kā lietotne atspoguļos eksistējošas mājaslapas saturu, tā ir atkarīga arī no mājaslapā veiktām izmaiņām. Tādām izmaiņām, kas mājaslapā ietekmē lietotāja piekļuvi kādai funkcijai, jāietekmē arī iespēja funkcijas piekļuvei lietotnē.

1.2.2 Produkta funkcijas

Lietotnē jārealizē šādas iespējas:

- pieteikšanās sistēmā ar exs.lv reģistrētu profilu
- jaunāko miniblogu saraksta skatīšana
    - izvēlēta minibloga atvēršana un lasīšana
    - miniblogam pievienoto komentāru skatīšana
    - komentāru vērtēšana pozitīvi/negatīvi
    - jauna komentāra pievienošana
    - atbildes pievienošana kādam komentāram
- jaunāko rakstu saraksta skatīšana
    - izvēlēta rakstu atvēršana un lasīšana
    - rakstam pievienoto komentāru skatīšana
    - komentāru vērtēšana pozitīvi/negatīvi
    - jauna komentāra pievienošana
- atteikšanās no sistēmas

1.2.3 Lietotāja raksturiezīmes

Lietotājs ir reģistrēts portālā exs.lv. Tam jābūt iemaņām darbā ar internetu un mobilo ierīci, kas aprīkota ar Android operētājsistēmu. Nepieciešamas zināšanas par to, kā ierīcei pievienot jaunu lietotni. Jāsaprot vispārējā mobilo lietotņu darbība un izmantojamība. Vēlama iepriekšēja pieredze ar skārienjutīgām ierīcēm.

Tālāko informāciju par lietotnes iekšējām iespējām jānodrošina saskarnei. Ja saskarne var būt neskaidra, tai jānodrošina paskaidrojoši paziņojumu un atgādinājumi.

1.2.4 Vispārējie ierobežojumi

Lietotne tiek programmēta Java programmēšanas valodā, izmantojot Android piedāvāto ietvaru.

Tā jāizstrādā ar domu, lai nākotnē to būtu ērti pielāgot ierīcēm ar lielāku ekrānu, piemēram, planšetdatoriem. Saskarnēm un programmkodam jābūt veidotam atbilstoši Google Android vadlīnijām.

Lai nodrošinātu veiksmīgu un mūsdienīgu tehnisko pusi un būtu iespējams izmantot jaunākās Android iespējas, lietotne jāprogrammē, atbalstot ne zemāk par API 11. līmeni (Android *Honeycomb* versija).

Lietotnei jābūt atļaujai pieslēgties tīklam un no tā saņemt atbildi.

1.2.5 Pieņēmumi un atkarības

Tiek pieņemts, ka lietotājs iepriekš ir lietojis kādu citu Android operētājsistēmai paredzētu lietotni, tāpēc ir pazīstams ar piedāvātajiem standarta saskarnes elementiem (rīkjosla, izbīdāma navigācija, izlecoši paziņojumi u.tml.) un iespējamo pārvietošanos pa tiem, tāpēc lietotnē saistībā ar tiem nav jāparāda izmantošanas paskaidrojumi.

Lai arī lietotne visticamāk strādās arī uz planšetdatoriem, tiek pieņemts, ka lietotājs izmanto viedtālruni un tieši tam lietotne ir jārealizē.

Raugoties no izstrādātāja puses, tiek pieņemts, ka servera puses nodrošinājums pieprasījumiem jau ir ieviests.

1.3 Lietotājstāsti

1.3.1 Lietotājstāsta struktūra

Lietotājstāsts	Apraksta veicamo darbību vai notikumu. Formāts: Kā <lietotājs> vēlos <funkcija>[, lai <no funkcijas gūtais labums>].
Pieņemšanas kritēriji	Kritēriju uzskaite, kuru izpilde nozīmē, ka lietotājstāsts ir izpildīts

1.3.2 Vispārīgi lietotājstāsti

1.3.2.1 Pieteikšanās sistēmai

Lietotājstāsts	Kā neautorizēts viesis vēlos pieteikties lietotnē, lai piekļūtu mājaslapas saturam.
Pieņemšanas kritēriji	Viesim redzams pieteikšanās skats ar aizpildāmu lietotājvārda un paroles lauku. Ievadot datus, lietotne no servera saņem atbildi par to pareizību. Datiem esot pareiziem, viesis nonāk lietotāja statusā, saņem tiesības skatīt pārējos lietotnes skatus un tiek pārvirzīts uz skatu ar jaunākajām ziņām. Neveiksmīgas pieteikšanās gadījumā aplūkojams atbilstošs kļūdas paziņojums.

1.3.2.2 Lietotnes navigācija

Lietotājstāsts	Kā lietotājs vēlos skatīt lietotnes navigāciju, ar pirkstu „izvelkot” to no skata kreisās puses.
Pieņemšanas kritēriji	Bīdot pirkstu no skata kreisās puses uz labo, no sāna izbīdās *Navigation Drawer*, no kuras iespējams pārvietoties uz citiem skatiem vai veikt kādu darbību. Navigācija nav pieejama neautorizētam lietotājam.

1.3.2.3 Atteikšanās no sistēmas

Lietotājstāsts	Kā pieteicies lietotājs vēlos no sistēmas atteikties, lai caur manu profilu pieejamais saturs vairs nebūtu skatāms.
Pieņemšanas kritēriji	Atverot sāna navigāciju un nospiežot uz „Iziet”, lietotājs nonāk viesa statusā un tiek pārvirzīts uz pieteikšanās skatu. Viesim neviens cits skats vairs nav pieejams.

1.3.3 Miniblogu lietotājstāsti

1.3.3.1 Jaunāko miniblogu skatīšana

Lietotājstāsts	Kā pieteicies lietotājs pēc lietotnes atvēršanas vēlos redzēt skatu ar jaunākajiem miniblogiem, kārtotiem dilstošā secībā pēc to pēdējā komentāra pievienošanas laika.
Pieņemšanas kritēriji	Atverot lietotni un esot pieteikta lietotāja statusā, aplūkojams skats, kurā tiek ielādēti jaunākie miniblogi. Miniblogi sarakstā kārtoti pēc laika, kurā tajos pievienots pēdējais komentārs. Pie katra minibloga aplūkojama būtiskākā ar to saistītā informācija, piemēram, autors, teksta fragments un pievienošanas laiks.

1.3.3.2 Senāku miniblogu skatīšana

Lietotājstāsts	Kā lietotājs vēlos skatīt senākus miniblogus, slidinot jaunāko miniblogu sarakstu lejup.
Pieņemšanas kritēriji	Noslidinot sarakstu līdz pēdējam miniblogam tajā, tiek ielādēti nākamie senākie miniblogi un pievienoti aiz jau esošā saraksta.

1.3.3.3 Jaunāko miniblogu kategorizācija

Lietotājstāsts	Kā lietotājs jaunāko miniblogu sarakstā vēlos atšķirt parastos un grupu miniblogus.
Pieņemšanas kritēriji	Miniblogiem, kuri pieder kādai no domubiedru grupām, ir redzams apakšējais rāmis dzeltenā krāsā. Pie minibloga informācijas ir arī nosaukums grupai, kurai miniblogs pievienots. Autora bildītes vietā ir redzama grupas bildīte.

1.3.3.4 Jaunāko miniblogu saraksta atjaunošana

Lietotājstāsts	Kā lietotājs vēlos atjaunot miniblogu sarakstu, lai redzētu jaunākos pievienotos.
Pieņemšanas kritēriji	Lietotnes augšējā rīkjoslā ir redzama atjaunošanas poga, pēc kuras nospiešanas jaunāko miniblogu saraksts tiek atjaunots. Nospiežot atjaunošanas pogu, lietotājs redz paziņojumu, ka notiek ielāde, kas, ielādei beidzoties, pazūd. Ja ielāde nav bijusi veiksmīga, tiek saņemts atbilstošs kļūdas paziņojums.

1.3.3.5 Minibloga satura skatīšana

Lietotājstāsts	Kā lietotājs vēlos skatīt sarakstā esošu miniblogu, uz tā nospiežot.
Pieņemšanas kritēriji	Nospiežot uz kāda no sarakstā esošajiem miniblogiem, lietotājs tiek pārvirzīts uz attiecīgā minibloga skatu. Skatā tiek parādīts paziņojums par notiekošu ielādi. Izdevušās ielādes gadījumā lietotājam aplūkojams minibloga saturs un citu lietotāju pievienotie komentāri. Neveiksmīgas ielādes gadījumā lietotājs tiek pārvirzīts uz jaunāko miniblogu saraksta skatu un saņem atbilstošu kļūdas paziņojumu.

1.3.3.6 Minibloga iespēju skatīšana

Lietotājstāsts	Kā lietotājs vēlos skatīt minibloga iespējas.
Pieņemšanas kritēriji	Īsu brīdi pieturot nospiestu minibloga saturu, parādās izlecošs skats ar tā iespējām. Nospiežot laukumā ārpus skata malām, tas tiek aizvērts.

1.3.3.7 Minibloga komentāra iespēju skatīšana

Lietotājstāsts	Kā lietotājs vēlos skatīt minibloga komentāra iespējas.
Pieņemšanas kritēriji	Īsu brīdi pieturot nospiestu minibloga komentāru, parādās izlecošs skats ar komentāra iespējām. Nospiežot laukumā ārpus skata malām, tas tiek aizvērts.

1.3.3.8 Komentāra pievienošana miniblogam

Lietotājstāsts	Kā lietotājs vēlos pievienot komentāru miniblogam, lai citi lietotāji to redzētu.
Pieņemšanas kritēriji	Zem pievienotajiem komentāriem redzama poga, pēc kuras nospiešanas atveras jauna komentāra pievienošanas forma. Komentārus var pievienot arī kā atbildes zem jau esošiem komentāriem. Atbilžu pievienošanas forma atverama, veicot „long-press" uz kāda no komentāriem un izvēloties iespēju „komentēt". Aizpildot formu un nospiežot pogu „Pievienot", komentārs tiek pievienots un ir redzams pievienotajā vietā.

1.3.3.9 Minibloga vērtēšana

Lietotājstāsts	Kā lietotājs vēlos novērtēt atvērto miniblogu ar pozitīvu vai negatīvu vērtējumu.
Pieņemšanas kritēriji	Pie minibloga satura redzams tā vērtējums. Pozitīvs vērtējums iekrāsots zaļā krāsā, negatīvs – sarkanā, bet, ja miniblogs nav vērtēts, redzama 0 pelēkā krāsā. Minibloga iespējās var novērtēt miniblogu ar pozitīvu vai negatīvu vērtējumu. Pēc vērtēšanas atvērtais skats tiek aizvērts, bet minibloga vērtējums – atjaunots. Lietotājs saņem paziņojumu par vērtējumu.

1.3.3.10 Minibloga komentāra vērtēšana

Lietotājstāsts	Kā lietotājs vēlos vērtēt citu lietotāju pievienotos komentārus.
Pieņemšanas kritēriji	Pie katra komentāra redzams vērtējums. Pozitīvi vērtējumi iekrāsoti zaļā krāsā, negatīvi – sarkanā, bet, ja komentārs nav vērtēts, redzama 0 pelēkā krāsā. Minibloga komentāra iespējās ir iespējams novērtēt to ar pozitīvu vai negatīvu vērtējumu. Pēc vērtēšanas atvērtais skats tiek aizvērts, bet minibloga komentāra vērtējums – atjaunots. Lietotājs saņem paziņojumu par vērtējumu.

1.3.3.11 Atvērta minibloga satura atjaunošana

Lietotājstāsts	Kā lietotājs vēlos atjaunot atvērta minibloga saturu.
Pieņemšanas kritēriji	Lietotnes augšējā rīkjoslā (*ActionBar)* ir redzama atjaunošanas poga. Nopiežot atjaunošanas pogu, redzams paziņojums par notiekošu ielādi. Veiksmīgas ielādes gadījumā esošais saturs tiek izdzēsts un tā vietā pievienots ielādētais. Neveiksmīgas ielādes gadījumā tiek parādīts atbilstošās kļūdas paziņojums.

1.3.4 Rakstu lietotājstāsti

1.3.4.1 Jaunāko rakstu saraksta skatīšana

Lietotājstāsts	Kā lietotājs vēlos redzēt skatu ar jaunākajiem rakstiem, kārtotiem dilstošā secībā pēc to pēdējā komentāra pievienošanas laika.
Pieņemšanas kritēriji	Esot pieteikta lietotāja statusā un jaunāko ziņu skatā pārejot uz rakstu cilni, tiek ielādēti jaunākie raksti. Raksti sarakstā kārtoti pēc laika, kurā tajos pievienots pēdējais komentārs. Pie katra raksta aplūkojama būtiskākā ar to saistītā informācija, piemēram, autors, raksta nosaukums un sadaļa.

1.3.4.2 Senāku rakstu skatīšana

Lietotājstāsts	Kā lietotājs vēlos skatīt senākus rakstus, slidinot jaunāko rakstu sarakstu lejup.
Pieņemšanas kritēriji	Noslidinot sarakstu līdz pēdējam rakstam tajā, tiek ielādēti nākamie senākie raksti un pievienoti aiz jau esošā saraksta.

1.3.4.3 Jaunāko rakstu saraksta atjaunošana

Lietotājstāsts	Kā lietotājs vēlos atjaunot rakstu sarakstu, lai redzētu jaunākos pievienotos.
Pieņemšanas kritēriji	Lietotnes augšējā rīkjoslā (*ActionBar)* ir redzama atjaunošanas poga, pēc kuras nospiešanas jaunāko rakstu saraksts tiek atjaunots. Nospiežot atjaunošanas pogu, lietotājs redz paziņojumu, ka notiek ielāde.

1.3.4.4 Raksta satura skatīšana

Lietotājstāsts	Kā lietotājs vēlos skatīt sarakstā esošu rakstu, uz tā nospiežot.
Pieņemšanas kritēriji	Nospiežot uz kāda no sarakstā esošajiem rakstiem, lietotājs tiek pārvirzīts uz attiecīgā raksta skatu. Skatā tiek parādīts paziņojums par notiekošu ielādi. Izdevušās ielādes gadījumā lietotājam aplūkojams raksta saturs ar pievienotajiem citu lietotāju komentāriem. Neveiksmīgas ielādes gadījumā lietotājs tiek novirzīts atpakaļ uz jaunāko rakstu sarakstu un saņem atbilstošu kļūdas paziņojumu.

1.3.4.5 Komentāra pievienošana rakstam

Lietotājstāsts	Kā lietotājs vēlos pievienot komentāru rakstam, lai citi lietotāji to redzētu.
Pieņemšanas kritēriji	Zem pievienotajiem komentāriem redzama poga, pēc kuras nospiešanas atveras komentāra pievienošanas forma. Aizpildot formu un nospiežot pogu „Pievienot”, komentārs tiek pievienots un ir redzams zem pārējiem komentāriem.

1.3.4.6 Raksta komentāra iespēju skatīšana

Lietotājstāsts	Kā lietotājs vēlos skatīt raksta komentāra iespējas.
Pieņemšanas kritēriji	Īsu brīdi pieturot nospiestu raksta komentāru, parādās izlecošs skats ar komentāra iespējām. Nospiežot laukumā ārpus skata malām, tas tiek aizvērts.

1.3.4.7 Raksta komentāra vērtēšana

Lietotājstāsts	Kā lietotājs vēlos vērtēt citu lietotāju pievienotos komentārus.
Pieņemšanas kritēriji	Pie katra komentāra redzams vērtējums. Pozitīvi vērtējumi iekrāsoti zaļā krāsā, negatīvi – sarkanā, bet, ja komentārs nav vērtēts, redzama 0 pelēkā krāsā. Raksta komentāra iespējās ir iespējams novērtēt to ar pozitīvu vai negatīvu vērtējumu. Pēc vērtēšanas atvērtais skats tiek aizvērts, bet raksta komentāra vērtējums – atjaunots. Lietotājs saņem paziņojumu par vērtējumu.

1.3.4.8 Atvērta raksta satura atjaunošana

Lietotājstāsts	Kā lietotājs vēlos atjaunot atvērta raksta saturu.
Pieņemšanas kritēriji	Lietotnes augšējā rīkjoslā (*ActionBar)* ir redzama atjaunošanas poga. Nopiežot atjaunošanas pogu, redzams paziņojums par notiekošu ielādi. Veiksmīgas ielādes gadījumā esošais saturs tiek izdzēsts un tā vietā pievienots ielādētais. Neveiksmīgas ielādes gadījumā tiek saņemts atbilstošs kļūdas paziņojums.

1.3.5 Jaunākā satura lietotājstāsti

1.3.5.1 Pārvietošanās no miniblogiem uz rakstiem un otrādi

Lietotājstāsts	Kā lietotājs vēlos pārslēgties no jaunākajiem miniblogiem uz jaunākajiem rakstiem un otrādi.
Pieņemšanas kritēriji	Jaunāko miniblogu un rakstu skatā redzama miniblogu un rakstu cilne. Starp tām var pārslēgties. Atvērtā cilne tiek iekrāsota. Starp cilnēm var pārvietoties arī, ar pirkstu velkot no skata viena sāna uz otru.

1.6 Drošības prasības

Piesakoties sistēmā ar lietotājvārdu un paroli, saziņai ar serveri jānotiek, izmantojot HTTPS savienojumu, garantējot drošu piekļuves datu apmaiņu ar serveri. Tādējādi lietotne varēs tikt izmantota arī publiskos *wifi* tīklos.

Serveris nepiedāvā HTTPS savienojumu citās lapas sadaļas kā vien pie pieteikšanās, bet tam nav būtiskas nozīmes, jo citi slepeni dati nav jāpārsūta.

1.7 Citas prasības

Lietotnei jānodrošina darbība tā, lai netiktu traucēta lietotāja saskarsme ar to. Veicot darbības vai ierosinot procesus, kas var darboties ilgāku laika periodu, tie jādarbina atsevišķos *thread*, lai tādējādi netiktu bloķēta lietotājam redzamā saskarne un lietotne neizskatītos apstājusies. Šī prasība sevišķi attiecināma uz darbībām tīklā un informācijas lasīšanu no tā.

Datu apmaiņai ar serveri jānotiek JSON formātā. Dati, kas jāuzglabā lietotnē, jāglabā *SharedPreferences* failos.

Dažādie lietotnes skati jāveido kā *aktivitātes*, savukārt tajās esošais saturs jāievieto kā *fragments*, kuru viegli gan ievietot, gan izņemt un aizstāt ar citu, saturam mainoties. Tas ir būtiski, lai uz lielākiem ekrāniem nākotnē varētu vienlaicīgi blakus rādīt vairākus fragmentus.

Lai pārvietošanās starp skatiem būtu ērta, katrā no tiem, atskaitot pieteikšanās skatu, no kreisās uz labo pusi jāvar izbīdīt *NavigationDrawer* skatu.

2 PROGRAMMATŪRAS PROJEKTĒJUMA APRAKSTS

2.1 Ievads

2.1.1 Nolūks

Programmatūras projektējuma apraksta (PPA) nolūks ir aprakstīt kvalifikācijas darba nodaļā „Programmatūras prasību specifikācija” (PPS) uzskaitīto prasību realizāciju.

Dokuments paredzēts sociālā portāla exs.lv Android lietotnes izstrādātājam.

2.1.2 Darbības sfēra

Dokuments apraksta sociālā portāla exs.lv Android lietotnē realizējamās funkcionalitātes projektējumu.

Tas paredzēts izstrādātāja darba strukturēšanai un PPS aprakstītajām prasībām atbilstošai lietotnes realizācijai.

2.1.3 Saistība ar citiem dokumentiem

Šis dokuments ir kvalifikācijas darba „Sociālā portāla exs.lv lietotne Android operētājsistēmai” sastāvdaļa un tādējādi izmantojams kopā ar to.

„Programmatūras projektējuma apraksts” lietojams kopā ar kvalifikācijas darbā iekļauto „Programmatūras prasību specifikāciju”, kurā definētās prasības tas apraksta.

2.2 Dekompozīcijas apraksts

2.2.1 Moduļu dekompozīcija

Tā kā lietotne tiek veidota Android operētājsistēmas piedāvātajā ietvarā, tā izmanto specifisku Android lietotņu realizācijas struktūru, kas, lai arī velk zināmas paralēles ar pazīstamo MVC modeli, tomēr no tā atšķiras.

Android lietotņu struktūrā ietilpst šādas sastāvdaļas:

Klases: tās pārvalda lietotnē notiekošos procesus, sasaistot kopā visus lietotnes elementus. Klases var gan darboties atsevišķi, nodrošinot specifiskas funkcijas, gan raksturot kādu Android izklājuma elementu, atvasinot tā attiecīgo Android klasi, gan izvietot lietotnes skatā kādu izklājumu, gan veikt to visu kopā. Klases arī izmanto pieejamos resursus un iekļauj tos skatos.

Izklājumi: XML formāta faili, kas apraksta saskarnes elementu struktūru. Tie var būt pilni izklājumi, kādi aplūkojami kādā lietotnes skatā, un tie var būt arī atsevišķi izklājuma elementi, piemēram, kāds daudzviet izmantojams, īpaši stilizēts teksta lauks, kas jāiekļauj kādā citā izklājumā.

Resursi: palīgfaili saskarņu papildināšanai.

To piemēri: dažādu formātu un izmēru attēli, saskarņu elementu noformējumu aprakstoši faili, saraksti ar citur izmantotām krāsām, masīviem, simbolu virknēm u.c., tādā veidā izvairoties no to iekodēšanas klašu failos. No klasēm tikai nepieciešams atsaukties uz nepieciešamo resursu.

Papildu iespējams resursu failus izvietot mapēs, kuru nosaukums raksturo resursa izmantojamības brīdi, piemēram, atkarībā no izvēlētās valodas vai ierīces orientācijas (portreta vai ainavformāts).

Nākamajās tabulās uzskaitīts minimums, kādi faili ir nepieciešami katrā no šīm sastāvdaļām.

2.2.1.1 Klases

lv.exs.android.classes	
App	Caur šo klasi jāvar saņemt atsauci uz aktivitātes kontekstu, kuram piekļūt no statiskām metodēm
Auth	Jāpārvalda ar lietotāju un tā sesiju saistīta informācija
Network	Jāsatur ar tīklu saistītas metodes
lv.exs.android.activities	
BaseActivity	Šajā jāinicializē visos skatos nepieciešamie objekti. Pārējās aktivitātes to atvasina
DrawerActivity	Atvasinās *BaseActivity* un nodrošinās izklājumu ar *NavigationDrawer*, kam jābūt redzamam visos skatos
lv.exs.android.screens	
LoginScreen	Ar pieteikšanos sistēmai saistītus fragmentus saturoša aktivitāte
HomeScreen	Jaunāko rakstu un miniblogu ciļņu skata aktivitāte
MiniblogScreen	Ar atvērtu miniblogu saistītus fragmentus saturoša aktivitāte
ArticleScreen	Ar atvērtu rakstu saistītus fragmentus saturoša aktivitāte
lv.exs.android.fragments	
MiniblogsFragment	Fragments ar jaunāko miniblogu sarakstu
MiniblogFragment	Fragments ar atvērta minibloga saturu
ArticlesFragment	Fragments ar jaunāko rakstu sarakstu
ArticleFragment	Fragments ar atvērta raksta saturu
lv.exs.android.adapters	
DrawerAdapter	Jānodrošina *NavigationDrawer* elementu saraksts. To izmanto *DrawerActivity*
HomeTabs	Jānodrošina lietotnes sākumskata ciļņu saturs, ievietojot aktivitātē nepieciešamos fragmentus
HomeMiniblogs	Jāaizpilda saraksts ar jaunākajiem miniblogiem
HomeArticles	Jāaizpilda saraksts ar jaunākajiem rakstiem
MiniblogRows	Jānodrošina saraksts ar minibloga komentāriem
ArticleRows	Jānodrošina saraksts ar raksta komentāriem

2.2.1.2 Izklājumi

Šajā dokumentā uzskaitīti tikai būtiskākie lietotnei nepieciešamie izklājumi. Tādi izklājumi, kas sastāv no viena daudz izmantota elementa vai kā cita līdzīgi mazsvarīga šī dokumenta kontekstā, var būt ļoti daudz un veidojami pēc nepieciešamības.

drawer_activity	Izklājums aktivitātei ar *NavigationDrawer*
drawer_row	*NavigationDrawer* vienas rindas izklājums
tab_activity	Izklājums ar vairākām cilnēm jaunāko ziņu aktivitātei
login_activity	Izklājums aktivitātei, kurā lietotājs piesakās sistēmai
home_miniblogs	Saraksts jaunākajiem miniblogiem
home_miniblogs_row	Minibloga rindas izklājums
home_articles	Saraksts jaunākajiem rakstiem
home_articles_row	Raksta rindas izklājums
miniblog_activity	Atvērta minibloga aktivitātes izklājums
miniblog_content	Izklājums minibloga saturam
miniblog_comments	Izklājums minibloga komentāru sarakstam
miniblog_comment	Izklājums minibloga komentāra rindai
article_activity	Atvērta raksta aktivitātes izklājums
article_content	Izklājums raksta saturam
article_comments	Izklājums raksta komentāru sarakstam
article_comment	Izklājums raksta komentāra rindai

2.2.1.3 Resursi

theme	Vispārīgais lietotnes dizains (rīkjoslas, pogas u.tml.)
styles	Izklājumu elementu stils (formas, rindas, teksta lauki u.tml.)
colors	Saraksts ar izklājumos izmantotām krāsām
strings	Teksti, kas tiek izmantoti lietotnē un uz kuriem ir atsauces klašu failos

2.2.2 Datu dekompozīcija

exs.lv lietotnē aplūkojamais saturs nav jāglabā pašā lietotnē, bet gan pie katra pieprasījuma jāsaņem no exs.lv servera. Tādējādi lietotne neizmantos datubāžu tabulas.

Android piedāvātajā ietvarā ir arī citi veidi, kā glabāt informāciju, tai skaitā glabāšana failos un *SharedPreferences*. Ar *SharedPreferences* Android nodrošina iespēju ērti saglabāt un lasīt *key-value* tipa informāciju.

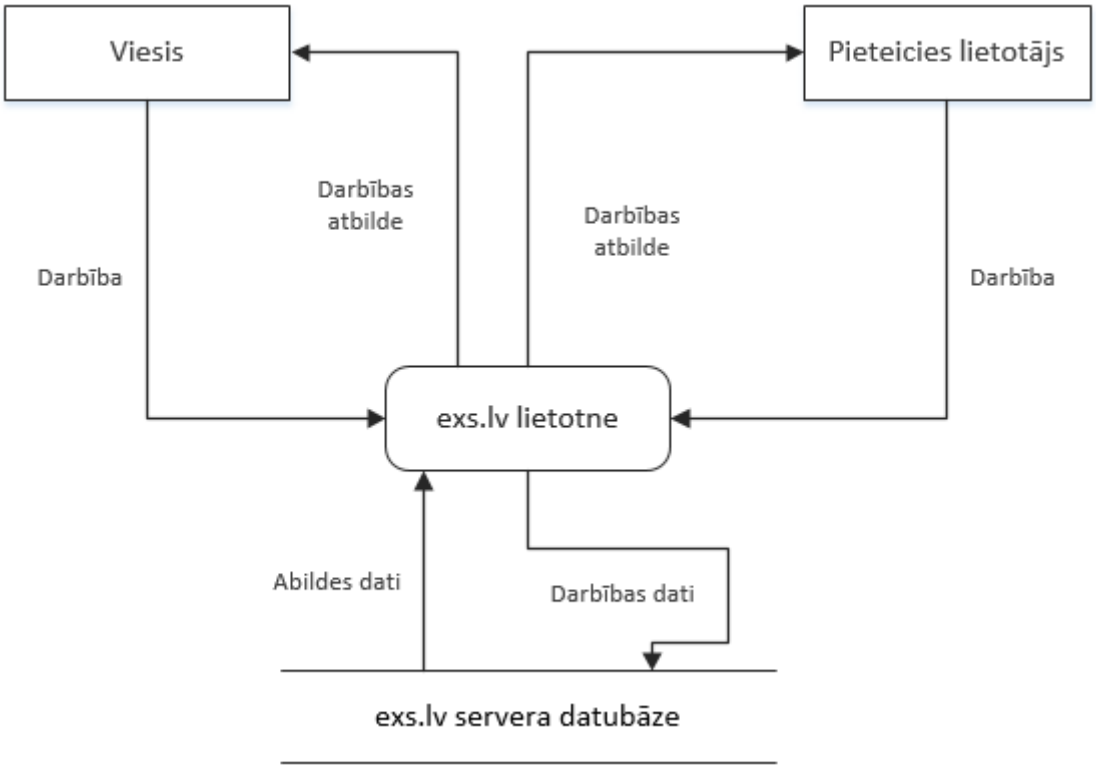
Tā kā lietotnei nepieciešams saglabāt tikai nelielu informāciju, kas saistīta ar sistēmai pieteikto lietotāju un sesijas uzturēšanu, tā izmantos tieši *SharedPreferences*.

Lai nodrošinātu ērtu attēlu lejuplādi tīklā (piemēram, lietotāju attēlus), jāizmanto *loopj* „*Android Smart Image View*" bibliotēka. Tā liek veikt tikai vienas īsas funkcijas izsaukumu, kā parametru norādot attēla tīkla adresi, savukārt iekšienē veic gan attēla īslaicīgu saglabāšanu kešatmiņā, gan lejuplādi citā *thread*, nebloķējot saskarni.

2.3 Datu projektējums

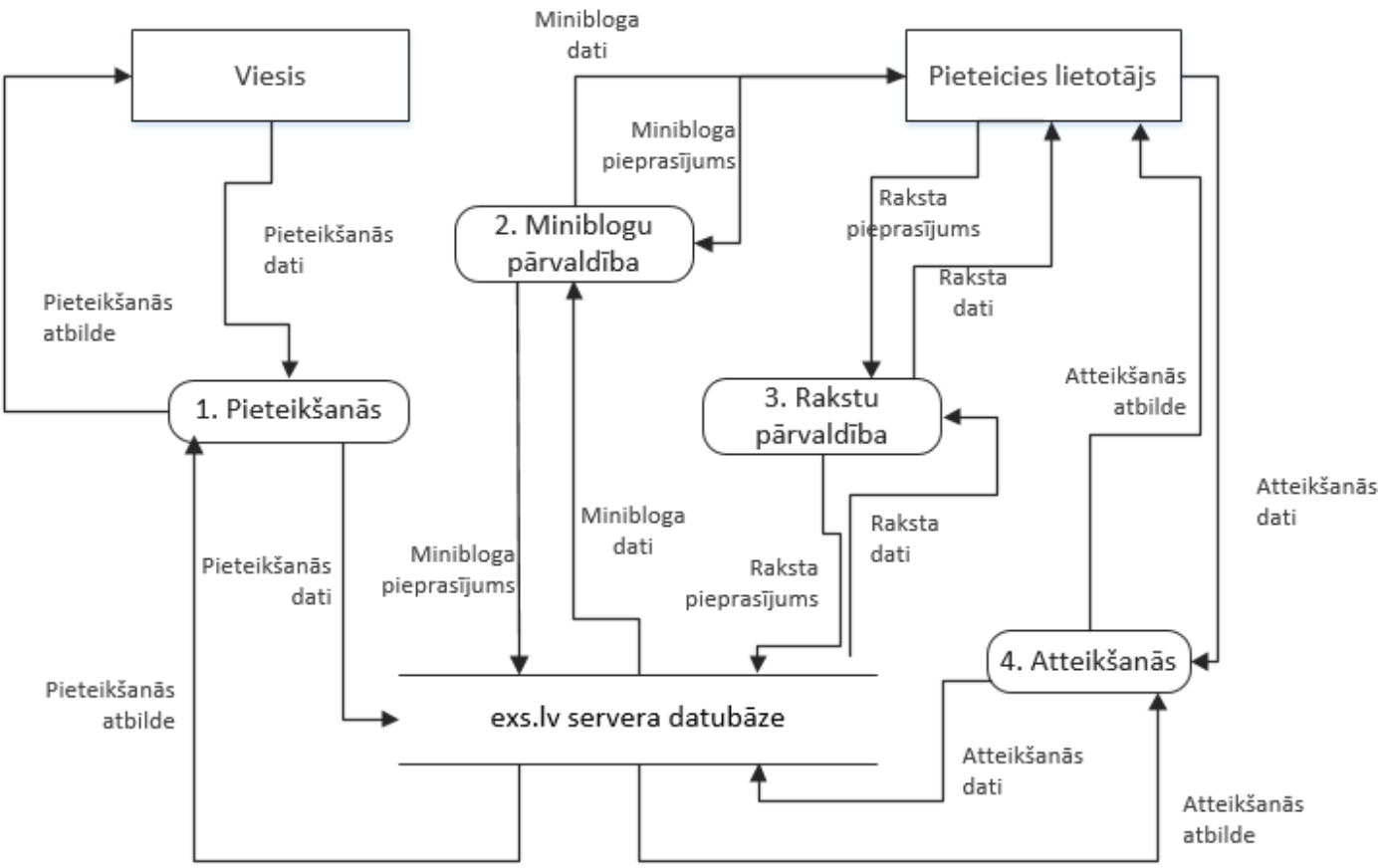
2.3.1 Datu plūsmu diagrammas

2.3.1.1 0. līmeņa diagramma



Ilustrācija 1

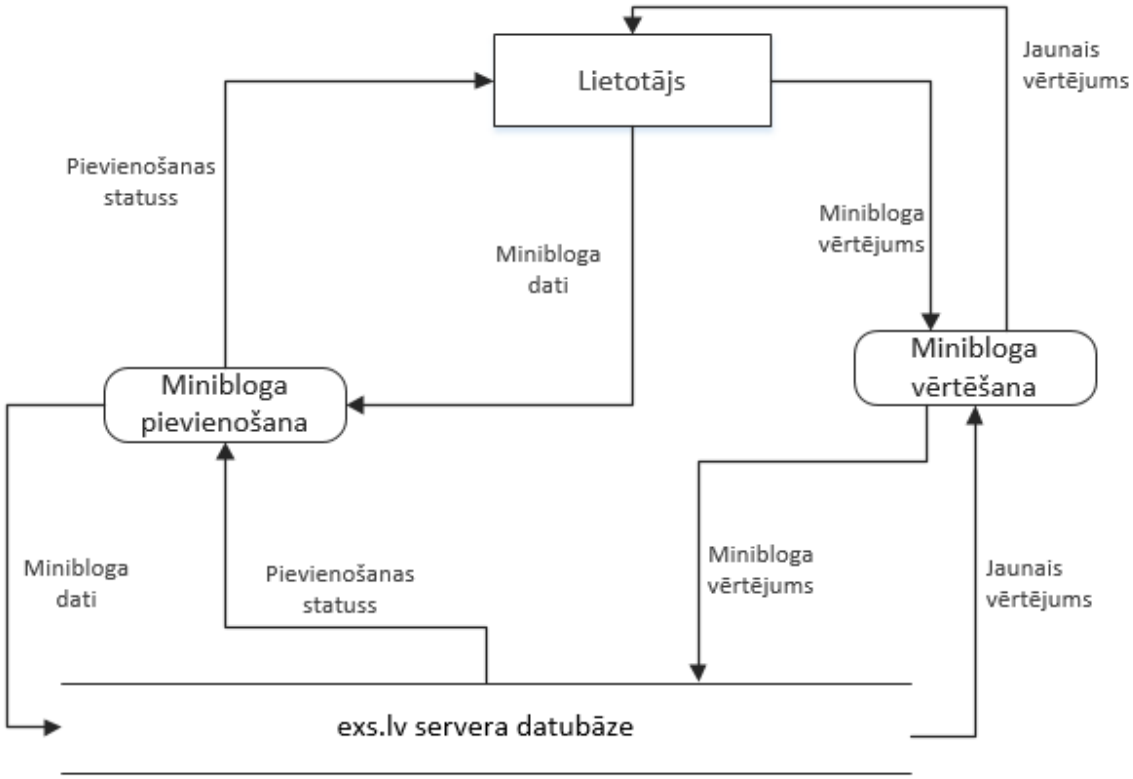
2.3.1.2 1. līmeņa diagramma



Ilustrācija 2

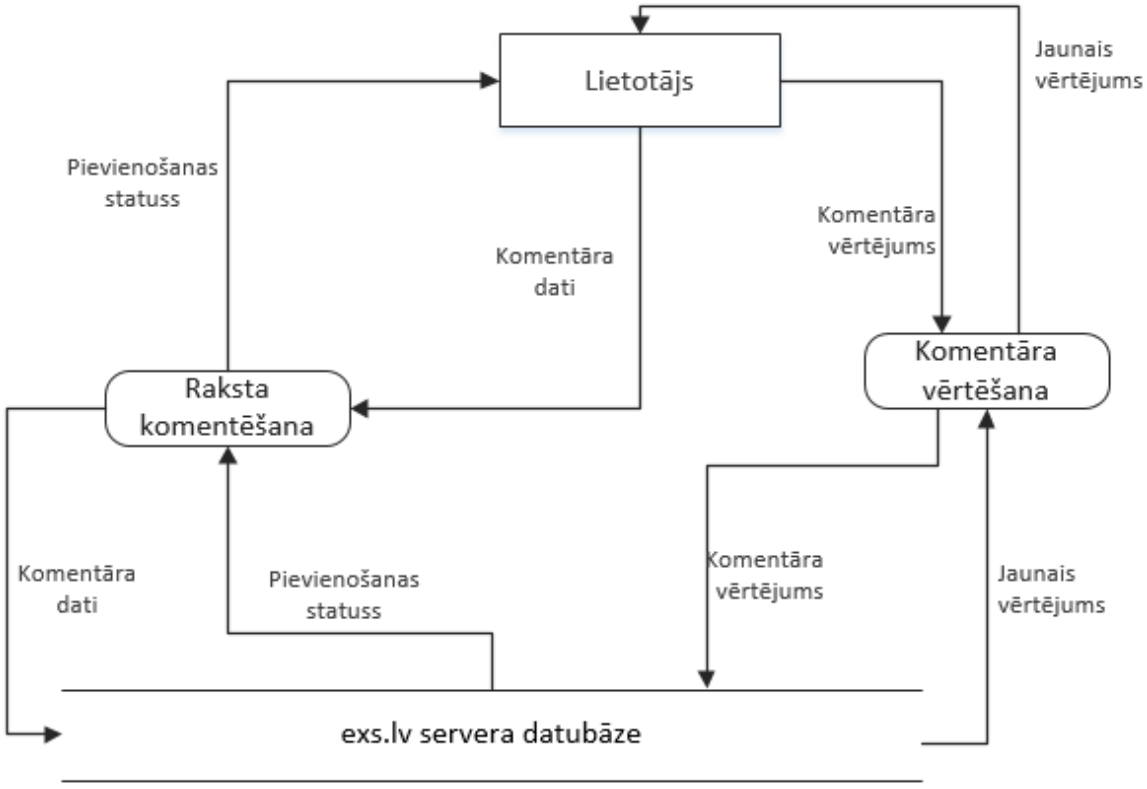
2.3.1.3 2. līmeņa diagrammas

2.3.1.3.1 Miniblogu pārvaldība



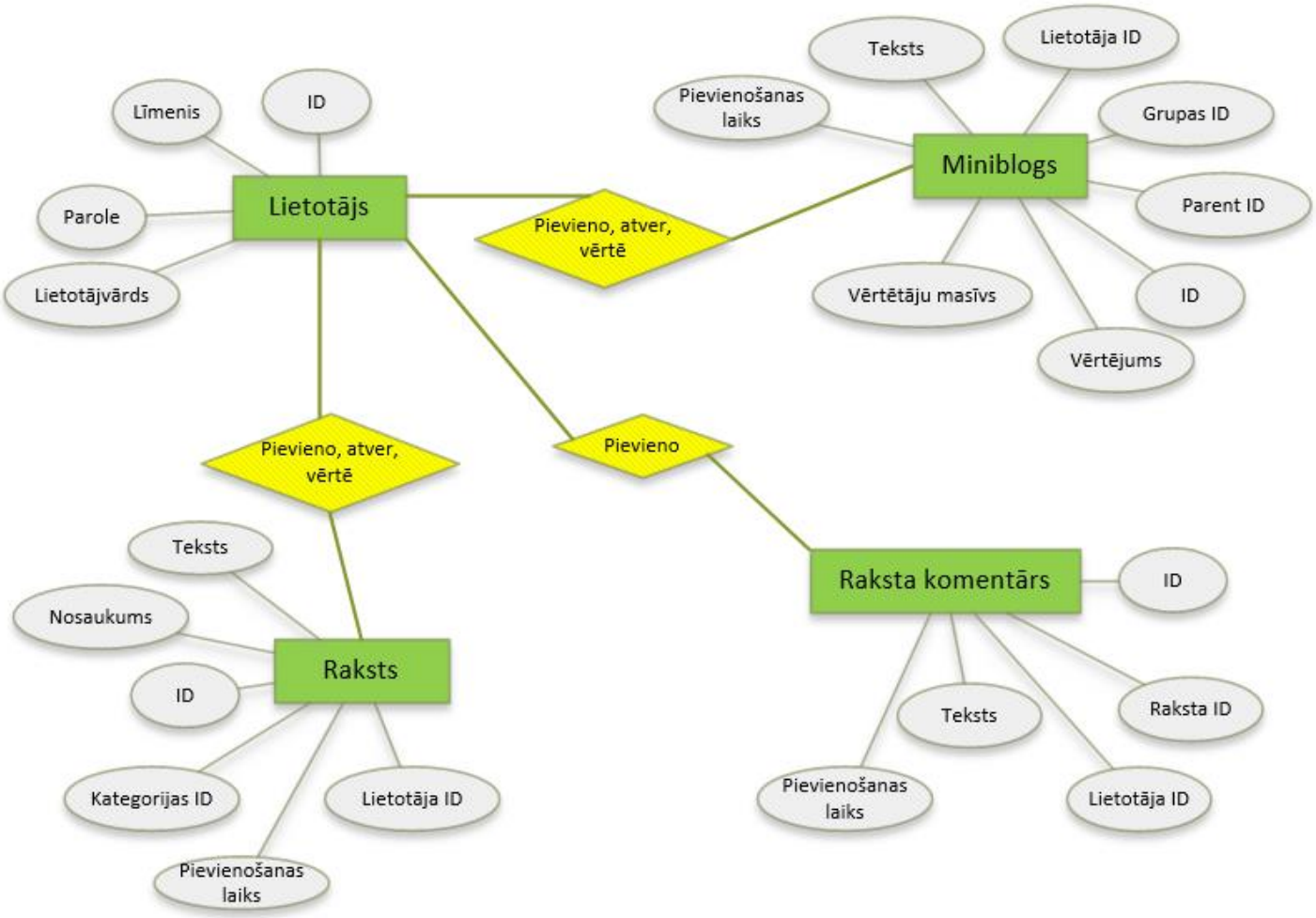
Ilustrācija 3

2.3.1.3.2 Rakstu pārvaldība



Ilustrācija 4

2.3.2 Konceptuālais ER modelis



Ilustrācija 5

2.4 Saskarnes apraksts

2.4.1 Pieteikšanās skats

Šajā skatā reģistrēti lietotāji var pieteikties sistēmā, ierakstot sava profila piekļuves informāciju. Ja lietotājs jau ir pieteicies sistēmā, atverot šo skatu, tas tiek pārvirzīts uz jaunāko miniblogu skatu, kurā nonāk arī tad, ja pieteikšanās formā ievada pareizus datus.

2.4.2 Jaunākie miniblogi

Šajā skatā aplūkojams saraksts ar jaunākajiem miniblogiem, kādus lapā pievienojuši lietotāji. Miniblogi kārtoti dilstošā secībā pēc laika, kurā tajos pievienots pēdējais komentārs.

No šī skata iespējams pārvietoties arī uz skatu ar jaunākajiem rakstiem, vai nu nospiežot uz atbilstošās cilnes, vai arī ar pirkstu velkot no labās uz kreiso pusi.

Veicot pretēju darbību – ar pirkstu velkot no kreisās uz labo pusi -, atvērsies *NavigationDrawer*, no kuras arī var pārvietoties uz citiem skatiem.

2.4.3 Atvērts miniblogs

Šeit aplūkojams atvērtā minibloga saturs. Ja rakstam citi lietotāji ir pievienojuši komentārus, tie aplūkojami zem satura. Komentāru pievienot var arī lietotājs, kas šo miniblogu šobrīd atvēris. Pie katra komentāra aplūkojams tā vērtējums.

2.4.4 Jaunākie raksti

Šajā skatā aplūkojams saraksts ar jaunākajiem rakstiem, kādus pievienojuši lapas lieotāji. Jaunākie raksti kārtoti dilstošā secībā pēc laika, kad tajos pievienots pēdējais komentārs.

Līdzīgi kā jaunāko miniblogu skatā, arī šeit ir iespēja pārslēgties uz jaunākajiem miniblogiem, nospiežot uz atbilstošās cilnes vai velkot ar pirkstu no kreisās uz labo pusi.

2.4.5 Atvērts raksts

Šeit aplūkojams atvērtā raksta saturs. Ja lietotāji rakstam pievienojuši komentārus, tie aplūkojami zem satura. Komentāru pievienot var arī lietotājs, kas šo rakstu šobrīd atvēris. Pie katra raksta komentāra aplūkojams tā vērtējums.

2.4.6 NavigationDrawer

Pieteicies lietotājs jebkurā skatā var skatīt arī *NavigationDrawer*, kas izvelkams, ar pirkstu bīdot no kreisās uz labo pusi. Veicot pretēju darbību, to var atkal aizvērt.

Šajā skatā aplūkojamas lietotnē pieejamās sadaļas, starp kurām pārslēgties.

No šī skata lietotājs var arī no sistēmas atteikties.

3 TESTĒŠANAS DOKUMENTĀCIJA

3.1 Ievads

Apsverot Android lietotnes uzbūvi un tajā izmantoto resursu struktūru, tika nolemts projektu izstrādes gaitā testēt ar integrācijas testiem.

Lietotnes izstrādes laikā katrai realizētajai funkcijai tests tika veikts uzreiz, tā pārliecinoties, ka tā strādā kā iecerēts. Lietotnes izstrādi noslēdzot, tika sastādīts testējamo vienumu un īpašību saraksts, un tad tās visas iztestētas, lai pārliecinātos, ka nekas nav mainījies. Attiecīgi iepriekš iztestēti vienumi tika testēti atkārtoti pēc daudzām izmaiņām.

Veicot testēšanu, izstrādātājs pārliecinājās, ka realizēti visi lietotājstāsti un esošais rezultāts atbilst visiem izvirzītajiem lietotājstāstu kritērijiem un PPS nosauktajām prasībām.

3.2 Testēšanas žurnāls

3.2.1 Lietotnes rīkjosla

Nr.	Nosacījums	Izpilde
1	Rīkjoslā redzams lietotnes nosaukums "Gaming Community".	✓
2	Blakus lietotnes nosaukumam redzams tās attēls/logo.	✓
3	Rīkjoslas labajā pusē redzamas iespēju podziņas, kurām norādīts, ka tām vienmēr vai tad, kad pietiek vieta, jābūt redzamām.	✓
4	Rīkjoslas iespējas, kurām nepietiek vietas rīkjoslā, aplūkojamas, nospiežot mobilās ierīces "Iespēju" pogu.	✓

3.2.2 Tīkla pieprasījumi

ID	Nosacījums	Izpilde
1	Veicot pieprasījum, paziņo, ja nav pieejams tīkla savienojums.	✓
2	Veicot pieprasījumu serverim, paziņo, ja nevar ar to sazināties.	✓
3	Paziņo, ja no servera saņemtā atbilde nav JSON formātā.	✓
4	Paziņo, ja servera atbilde JSON formātā nesatur nepieciešamos identifikācijas laukus.	✓
5	Ja no servera saņemts kļūdas paziņojums, parāda to kā [Toast] paziņojumu.	✓
6	Veicot ielādi, procesa laikā rādās notiekošas ielādes paziņojums, kurš, ielādei beidzoties, pazūd. *Skaidrojums*: *paziņojums rādījās tikai atsevišķiem pieprasījumiem. Izpētot situāciju, konstatēts, ka kļūme rodas, ja vienlaicīgi tiek veikti vairāki pieprasījumi. Sarežģītības dēļ šim brīdim nolemts šādus paziņojumus atsevišķās vietās nerādīt.*	✗

3.2.3 Lietotāja sesija

ID	Nosacījums	Izpilde
1	Ja pēc satura ielādes konstatēts, ka lietotājs vairs servera pusē nav autorizēts, sesiju dzēš un pārvirza uz pieteikšanās aktivitāti.	✓

3.2.4 Vispārīgas pārbaudes

ID	Nosacījums	Izpilde
1	Atbilstoši lietotāja klasei tiek iekrāsots tā lietotājvārds.	✓
2	Lietotājam esot tiešsaistē, tā lietotājvārda priekšā redzama zvaigznīte. *Skaidrojums: zvaigznīte nerādījās lietotājiem, kuri tiešsaistē bija no mobilajām ierīcēm. Kļūme novērsta.*	✗

3.2.5 Navigation Drawer

ID	Nosacījums	Izpilde
1	Atverot lietotni no jauna, iekrāsota ir pirmā rinda ar tekstu "Sākums". *Skaidrojums: pēc izmaiņām navigācijas izklājumā netika iekrāsota neviena rinda. Kļūme novērsta.*	✗
2	Nospiežot uz kādas rindas, tās fons iekrāsojas, savukārt iepriekšējai izvēlētajai rindai iekrāsotais fons pazūd.	✓
3	*Drawer* atverama un aizverama, pavelkot ar pirkstu un vienu vai otru pusi. *Skaidrojums: jaunāko rakstu fragmenta izklājuma īpatnību dēļ Drawer ar pirkstu nebija aizverama. Kļūme novērsta.*	✗
4	*Drawer* aizverama, nospiežot laukumā ārpus tās.	✓
5	Aktivitātēs, kurās ir pieejama *Drawer*, augšējā rīkjoslā ir redzama tās bildīte, kas, *Drawer* atverot vai aizverot, pārvietojas uz sāniem.	✓
6	Nekāda darbība netiek veikta, nospiežot uz *Drawer* rindas, kas jau atzīmēta kā atvērtā.	✓

3.2.6 HomeScreen

ID	Nosacījums	Izpilde
1	Atverot lietotni no jauna, redzama HomeScreen aktivitāte.	✓
2	HomeScreen aktivitātē redzamas divas cilnes - "Miniblogi" un "Raksti".	✓
3	Ielādējot aktivitāti, atvērtā ir "Miniblogu" cilne.	✓
4	Nospiežot uz kādas no cilnēm, par aktīvo kļūst nospiestā un tiek parādīts atbilstošais fragments.	✓
5	Atvērtajai cilnei ir biezāks apakšējais rāmītis.	✓
6	Pārvietošanās starp cilnēm iespējama arī ar "Swipe" kustību, bīdot no viena sāna uz otru.	✓
7	Ar "Swipe" kustību pārvietoties starp cilnēm var tikai tik reižu, cik ciļņu arī ir, nevis vairāk.	✓
8	Rotējot ekrānu horizontālā stāvoklī un cilnēm nonākot rīkjoslā, tām saglabājas caurspīdīgs fons.	✓

9	Nospiežot uz rīkjoslā redzamās satura atjaunošanas podziņas, fragmenta saturs tiek atjaunots.	✓

3.2.7 Jaunākie miniblogi

ID	Nosacījums	Izpilde
1	Veicot saraksta atjaunošanu, jau ielādētie miniblogi tiek aizstāti ar jauniem.	✓
2	Ja ielāde nav bijusi veiksmīga, iepriekš ielādētie miniblogi netiek no saraksta dzēsti.	✓
3	Pārvietojoties līdz saraksta pēdējai rindai, zem tās tiek ielādēti nākamie jaunākie miniblogi.	✓

3.2.8 Atvērts miniblogs

ID	Nosacījums	Izpilde
1	Atverot minibloga aktivitāti, tiek ielādēts minibloga saturs.	✓
2	Minibloga saturs atbilst tam, kurš miniblogs tiešām tika atvērts.	✓
3	Nospiežot rīkjoslā redzamo atjaunošanas pogu, saturs tiek pārlādēts pa jaunam.	✓
4	Pārlādējot saturu, vertikālā ritjosla paliek tajā pašā pozīcijā, ja vien klāt nav nācis kāds jauns komentārs.	✓
5	Ja miniblogam nav pievienotu komentāru, zem tā redzams teksta lauks ar attiecīgu paziņojumu.	✓
6	Nospiežot uz komentāra pievienošanas pogas, atveras dialogs ar komentēšanas formu.	✓
7	Pieturot īsu brīdi pirkstu uz kāda no komentāriem (vai minibloga), parādās logs ar tā iespējām.	✓
8	No iespēju loga izvēloties atbildēšanas iespēju, atveras dialogs ar komentēšanas formu. *Skaidrojums: pēc nesen veiktām izmaiņām forma nesaņēma pareizos datus no aktivitātes. Kļūme novērsta.*	✗
9	Nospiežot ārpus dialoga, tas neaizveras un formā esošais ievadītais teksts nepazūd.	✓

10	Nospiežot iespēju "Atcelt", dialogs pazūd.	✓
11	Nospiežot iespēju "Pievienot", komentārs tiek nosūtīts serverim un minibloga saturs pārlādēts.	✓
12	Izvēloties pozitīvas vērtēšanas pogu, vērtējums tiek palielināts un atjaunots.	✓
13	Izvēloties negatīvas vērtēšanas pogu, vērtējums tiek samazināts un atjaunots.	✓
14	Paziņo, ja vērtēts tiek savs komentārs.	✓
15	Paziņo, ja vērtēts tiek komentārs, kuru šis lietotājs jau ir vērtējis.	✓
16	Atjaunojot vērtējumu, tam piemēro pareizo krāsu (> 0 - zaļā krāsā; < 0 - sarkanā krāsā, 0 - pelēkā krāsā). *Skaidrojums: vērtējot pašu miniblogu, atjaunotais vērtējums vienmēr bija pelēkā krāsā. Kļūme novērsta.*	✗

3.2.9 Jaunākie raksti

ID	Nosacījums	Izpilde
1	Veicot saraksta atjaunošanu, jau ielādētie raksti tiek aizstāti ar jauniem.	✓
2	Ja ielāde nav bijusi veiksmīga, iepriekš ielādētie raksti netiek no saraksta dzēsti.	✓
3	Pārvietojoties līdz saraksta pēdējai rindai, zem tās tiek ielādēti nākamie jaunākie raksti.	✓

3.2.10 Atvērts raksts

ID	Nosacījums	Izpilde
1	Atverot raksta aktivitāti, tiek ielādēts raksta saturs.	✓
2	Raksta saturs atbilst tam rakstam, kurš tiešām tika atvērts.	✓
3	Nospiežot rīkjoslā redzamo atjaunošanas pogu, saturs tiek pārlādēts pa jaunam.	✓
4	Pārlādējot saturu, vertikālā ritjosla paliek tajā pašā pozīcijā, ja vien klāt nav nācis kāds jauns komentārs.	✓
5	Ja rakstam nav pievienotu komentāru, zem tā redzams teksta lauks ar attiecīgu paziņojumu.	✓
6	Nospiežot uz komentāra pievienošanas pogas, atveras dialogs ar komentēšanas formu.	✓
7	Pieturot īsu brīdi pirkstu uz kāda no komentāriem, parādās logs ar tā iespējām.	✓
8	Nospiežot uz zem komentāriem esošās pogas, atveras dialogs ar komentēšanas formu.	✓
9	Nospiežot ārpus dialoga, tas neaizveras un formā esošais ievadītais teksts nepazūd.	✓
10	Nospiežot iespēju "Atcelt", dialogs pazūd.	✓
11	Nospiežot iespēju "Pievienot", komentārs tiek nosūtīts serverim un raksta saturs pārlādēts.	✓
12	Izvēloties pozitīvas vērtēšanas pogu, vērtējums tiek palielināts un atjaunots.	✓
13	Izvēloties negatīvas vērtēšanas pogu, vērtējums tiek samazināts un atjaunots.	✓
14	Paziņo, ja vērtēts tiek savs komentārs.	✓
15	Paziņo, ja vērtēts tiek komentārs, kuru šis lietotājs jau ir vērtējis.	✓
16	Atjaunojot vērtējumu, tam piemēro pareizo krāsu (> 0 - zaļā krāsā; < 0 - sarkanā krāsā, 0 - pelēkā krāsā).	✓

Testēšanas rezultātu kopsavilkums

Tā kā testēšana tika veikta jau izstrādes gaitā, noslēgumā veiktie testēšanas rezultāti bija visai iepriecinoši un atklāja tikai nelielas un ātri novēršamas kļūdas.

4 PROGRAMMATŪRAS PROJEKTA ORGANIZĀCIJA

Projekts izstrādāts, izmantoto Spējās izstrādes pieeju. Šī pieeja izvēlēta, jo bija skaidrs, ka prasības laika gaitā mainīsies un tās nāksies realizēt pakāpeniski, jo izstrādātājam sākotnēji nebija zināšanu par Android lietotņu uzbūvi un to nācās apgūt. Iesākumā bija tikai vīzija par lietotnē vēlamajām funkcijām, nezinot to realizēšanas sarežģītību. Projektam attīstoties, tika gūtas arī zināšanas un attiecīgi papildinātas prasības.

Lietotne realizēta Android operētājsistēmai, tāpēc tā izmanto specifisku Google piedāvātu Android projekta moduļu organizāciju, kas, lai arī ir līdzīga pazīstamajam MVC modelim, tomēr atšķiras. Tomēr, balstoties uz programmizstrādes labo praksi, visas Android modeļa iespējas arī ir izmantotas, tā, piemēram, saskarņu elementu stilu atdalot un atstājot atsevišķos resursu failos, uz kuriem pēc tam var atsaukties. Līdzīgi arī dažādi informatīvi paziņojumu teksti nav iestrādāti izklājumu failos, bet gan atstāti resursu failā, no kura tos var iegūt ar pareizajām atsaucēm. Arī izmantoto attēlu faili veidoti dažādos formātos, lai to kvalitāte būtu vienlīdz kvalitatīva uz dažāda izmēra ierīcēm.

Android lietotne programmēta, izmantojot Java programmēšanas valodu. Izstrādē izmantota *Eclipse IDE*, kurai pievienots *Android SDK* un kas atbalsta *GIT* versiju kontroles sistēmas izmantošanu.

Lai lietotni testētu uz lokāli esoša datora, uz tā tika uzlikts *Apache webserveris* un exs.lv mājaslapas kopija. Testējot lietotnes darbību, tā slēdzās klāt lokālajam datoram, bet pašā lietotnē bija iestrādāta administrēšanas iespēja pārslēgt izmantoto adresi uz īstā exs.lv servera adresi, lai to pašu uzreiz pārbaudītu arī dzīvajā versijā, nepārkompilējot pašu lietotni.

Projekta gaitā izmantoti arī teksta faili, kuros pierakstītas izstrādātājam nepieciešamas piezīmes, vēl paveicamo darbu saraksts u.c. Šādi faili tika veidoti, izmantojot *Sublime Text 2* teksta redaktoru, bet paši faili savukārt atradās mapē, kas tika sinhronizēta ar *Dropbox* serveriem, tādējādi glabājot to kopiju.

5 KVALITĀTES NODROŠINĀŠANA

Mainoties programmatūras projektam vai ieviešot jaunas prasības, regulāri tika salīdzināta PPS un PPA atbilstība vienam pret otru. To rakstīšanas gaitā ievēroti arī PPS un PPA atbiltošie standarti, kas šajā kvalifikācijas darbā nosaukti pie atbilstošā dokumenta nodaļas „Saistība ar citiem dokumentiem".

Izstrādājot lietotni, tika ievērota programmēšanas labā prakse un attiecīgās Android izstrādātāju definētās vadlīnijas. Atbilstoši Android piedāvātajai lietotnes moduļu arhitektūrai visi izstrādātie faili un resursi kārtoti attiecīgos failos un mapēs.

Lai vieglāk analizētu lietotnē notiekošas kļūdas un vērotu tās darbību, tajā tika izmantota arī funkcija, kas veic ierakstus žurnālfailos. *Android SDK* piedāvā žurnalēšanas funkcijai norādīt veicamā ieraksta svarīgumu, tādējādi tie tiek kategorizēti pēc tādām īpašībām, vai tie ir kā parasti ziņojumi, brīdinājumi vai kļūdas, kurām jāaptur lietotnes darbība. Veiktie ieraksti žurnālfailos ir aplūkojami *Eclipse IDE* tam paredzētā logā.

Izstrādājot lietotnes funkcijas, tām uzreiz tika veikta arī integrācijas testēšana, lai redzētu, ka tās veic uzdevumu atbilstoši projektējumam.

Izstrādātais programmatūras pirmkods ir komentēts tā, lai to saprast nebūtu grūti arī citiem Android izstrādātājiem. Komentāri rakstīti latviešu valodā ar atsevišķiem Android specifiskiem terminiem angļu valodā.

6 KONFIGURĀCIJU PĀRVALDĪBA

Projekta izstrādē ērtai konfigurāciju pārvaldībai izmantota GIT versiju kontroles sistēma. Visas izmaiņas pēc katra veiktā *commit* tika glabātas uz lokālās atmiņas, savukārt vismaz reizi dienā visas izmaiņas sinhronizētas ar BitBucket servisu, uz kura atrodas lietotnes repozitorija kopija.

Izmantot GIT servisu bija izdevīgi, jo arī Eclipse IDE to automātiski uztvēra un izcēla failus, kuros veiktās izmaiņas vēl nebija saglabātas ar *commit*.

Tā kā izstrāde notika uz datora ar Windows operētājsistēmu, GIT pārvaldībai tika izmantota Windows GitHub klienta programma ar grafisko saskarni, kas spēja sazināties ne tikai ar GitHub, bet arī BitBucket servisu. Izmantot tieši BitBucket servisu bija izdevīgi un likumsakarīgi, jo uz tā atrodas arī pati exs.lv mājaslapa, tādējādi saistīti produkti glabājas vienuviet un programmētāji var aplūkot abus projektus.

Versiju kontroles sistēmas izmantošana projekta izstrādes gaitā atmaksājās, jo bija iespējams atrast pirmkoda fragmentus, kas pirms kāda laika dzēsti, kā arī salīdzināt, kādas izmaiņas noteiktā laikā veiktas, lai analizētu, kurā brīdī rādusies atrasta funkcijas problēma. Dažkārt radās situācijas, kad kāds no pirmkoda failiem tika pārdēvēts vai pārvietots uz citu pakotni, kā rezultātā atsevišķos lietotnes binārajos failos neveiksmīgi nepārveidojās atsauces, tāpēc programma vairs nestrādāja. Ar GIT palīdzību bija iespējams veiktās izmaiņas atsaukt un pakāpeniski ieviest pa jaunam.

Tā kā lietotne neizmantoja lokālu datubāzi, bet gan veica pieprasījumus exs.lv portāla dzīvās versijas serverim, nebija nepieciešams veikt rezerves kopijas lokālai datubāzei.

Author	Commit	Message
edgarspelna	0460ddb	Mainīta app ikona
edgarspelna	538183c	Lietotāju zvaigznītes un pasvītroti grupu miniblogi
edgarspelna	2173415	Salabota miniblogu komentēšana, vērtēšana un floating ctx menu
edgarspelna	340a805	Response klase
edgarspelna	e9359d4	Iepriekšējā commit piemirsti faili
edgarspelna	078a47b	Pilnībā pārveidota atvērta minibloga struktūra

Ilustrācija 6

7 DARBIETILPĪBAS NOVĒRTĒJUMS

Kvalifikācijas darba produkta darbietilpības novērtēšanai izmantota eksperta metode. Pirms darba uzsākšanas tika fiksētas vairākas galvenās darba izstrādes stadijas:

- Java programmēšanas valodas apguve
- Android lietotņu izstrādes vides un iespēju apgūšana
- Izstrādājamās lietotnes skatu un iespēju projektēšana
- Lietotnes izstrāde

Ņemot vērā faktu, ka darba autors savā programmēšanas pieredzē jau labā līmenī apguvis vairākas programmēšanas valodas, tai skaitā C++ valodu, Java apgūšana tika novērtēta kā ātri apgūstama. Tika paredzēts, ka tā neprasīs vairāk par nedēļu, lai caurskatītu pamācības un iepazītos ar Java sintaksi, iespējām un atšķirībām, salīdzinot to ar citām, jau zināmām valodām. Apgūstot Javu, tika secināts, ka novērtējums nebija īsti precīzs, jo Java šķita tik līdzīga citām valodām, ka beigu beigās tās apgūšana neprasīja vairāk par 5 dienām (kas gan ir visai tuvu sākotnējam novērtējumam).

Ieskatoties Google veidotajā Android pamācībā, tika secināts, ka tik apjomīga dokumentācijas izlasīšana prasīs parāk daudz laika un ne viss no tā būs nepieciešams lietotnes izstrādei. Tika novērtēts, ka nepieciešamo nodaļu izlasīšana, apgūšana un izmēģināšana prasīs vismaz 3 nedēļas.

Darba noslēgumā secināms, ka laiks tika novērtēts salīdzinoši precīzi, jo vairāk laika bez mērķtiecīgas lietotnes izstrādes apgūšanai nebija vērts tērēt. Tālāk tika uzsākts darbs pie izstrādes, kuras laikā gan dokumentācija kā palīglīdzeklis vēl aizvien tika daudz izmantota.

Tā kā lietotnes iespējas nebija jādomā no jauna, jo tās mājaslapā jau eksistē, kā arī ieskats Android lietotņu izstrādē bija gūts, projektēšanai patērējamais laiks tika novērtēts tikai 5 dienām. Aptuveni šāds arī bija projektēšanai veltītais laika periods.

Rēķinoties, ka lietotne tiek veidota pirmo reizi un aizķeršanās būs pie daudzām niansēm, sākotnēji projektēšanā tika fiksētas tikai pavisam nedaudzas iespējas, kas turklāt savā starpā pēc uzbūves ir visai līdzīgas, tādējādi spriežot, ka atlikušajā laikā tās būtu realizējamas.

Projektētās iespējas, lai arī izrādījušās grūtākas nekā iepriekš paredzēts, atlikušajā laikā tiešām tika realizētas, izstrādi pilnībā pabeidzot vien divas dienas pirms darba iesniegšanas.

REZULTĀTI

Portāla exs.lv lietotnes Android operētājsistēmai minimums ir izstrādāts un teorētiski tā arī būtu publicējama Google Play, lai citi lietotāji to varētu sākt izmantot. Tomēr izstrāde izrādījās sarežģītāka nekā gaidīts un ir palikušas daudzas sīkas nianses, kurām pietrūka laika un kuras darba autors vēlas uzlabot, lai lietotnes izmantošana šķistu *draudzīgāka*.

Tā kā autors Android lietotņu izstrādi apguva no nulles, mācību procesa gaitā zināšanas tika palēnām uzlabotas, tāpēc ik pa laikam nācās atgriezties pie esoša koda un to pārveidot, balstoties uz šīm jaunajām zināšanām. Šādas izmaiņas arī prasīja papildu laiku, tomēr autors vēlējās tās veikt laicīgi, nevis daudz par vēlu. Savukārt atsevišķas iespējas sagādāja daudz darba, jo ilgstoši nestrādāja kā nākas.

Kopumā par spīti daudziem sarežģījumiem darba autors ar paveikto ir apmierināts, jo projekts ļāvis tā izstrādātājam gūt jaunas zināšanas gan programmēšanā Java valodā, gan lietotņu izstrādē viedtālruņiem, gan tieši Android operētājsistēmas darbībā.

PROGRAMMATŪRAS PIRMKODA FRAGMENTI

Kods fragmentam, kurā aplūkojams atvērta minibloga saturs.

```
1  package lv.exs.android.fragments;
2
4  *  Atvērts miniblogs..
9
10 import java.util.ArrayList;..
36
37 public class MiniblogFragment extends Fragment
38                              implements MbCommentForm.DialogListener,
39                                         Network.AsyncListener {
40
41     // tīkla pieprasījumus raksturojošas vērtības, pēc kurām nosaka,
42     // kādai metodei jāpadod no servera saņemtā atbilde
43     private static final int LOAD_CONTENT = 0;
44     private static final int LOAD_VOTE = 1;
45
46     private BaseActivity activity;
47
48     // izklājuma skats, kas no fragmenta jāatgriež,
49     // lai aktivitāte zinātu, ko ievietot kopējā izklājumā
50     private LinearLayout lly_miniblog;
51
52     private long miniblog_id;
53     private MiniblogComments adp_rows;
54     private ArrayList<JSONObject> arr_comments;
55
56     // nospiestā elementa sarakstā pozīcija
57     private int item_position = 0;
58
59     private JSONObject content;
60
61     // pozīcija sarakstā specifiskiem skatiem
62     private int rowid_content = 0;
63     private int rowid_extra = 0;
64
65
66     /**
67      *  Izveido un atgriež fragmenta izklājumu
68      *
69      *  @param inflater
70      *  @param container
71      *  @param savedInstanceState
72      *  @return View    minibloga fragmenta izklājums
73      */
74     @Override
75     public View onCreateView(LayoutInflater inflater, ViewGroup container,
76                                                 Bundle savedInstanceState) {
77
78         try {
79             activity = (BaseActivity)getActivity();
80         } catch (ClassCastException e) {
81             Notify.t(R.string.internal_error);
82             return null;
83         }
84
85         // lai parādītu saturu, jābūt zināmam minibloga datubāzes id
86         miniblog_id = activity.getIntent().getLongExtra("id", 0);
87         if (miniblog_id == 0) {
88             Notify.t(R.string.wrong_param);
89             return null;
90         }
91
92         adp_rows = null;
93
94         // inicializē jau šeit un nevis pie katras ielādes, lai, veicot satura
95         // atjaunošanu, ritjosla paliktu tajā pašā vietā
96         arr_comments = new ArrayList<JSONObject>();
97
98         lly_miniblog = (LinearLayout) inflater.inflate(
```

```
                                        R.layout.miniblog_container, null);
        loadContent();

        return (View)lly_miniblog;
    }


    /**
     *  Tīkla interfeisa metode
     *
     *  Tiek izsaukta, kad tīkla pieprasījums saņēmis atbildi.
     *  Pēc "task" nosaka, kāds uzdevums šo atbildi pieprasīja
     *
     *  @param task     uzdevuma identifikators
     *  @param result   saņemtie dati
     */
    public void onTaskComplete(int task, JSONObject result) {

        if (task == LOAD_CONTENT) { // ielādēts minibloga saturs
            parseContent(result);
        } else if (task == LOAD_VOTE) { // vērtēts kāds no komentāriem
            parseVoteResponse(result, item_position);
        }
    }


    /**
     *  Ielādē fragmenta saturu
     *
     *  Tiek izsaukta arī no MiniblogScreen aktivitātes,
     *  kad nospiesta refresh poga
     */
    public void loadContent() {
        Notify.showLoading(activity);
        Network.fetch(this, "/miniblogs/" + miniblog_id, LOAD_CONTENT);
    }


    /**
     *  Piepilda izklājumu ar minibloga saturu
     *
     *  @param response minibloga satura dati
     */
    private void parseContent(JSONObject response) {

        JSONObject comments;
        try {
            JSONObject obj  = response.getJSONObject("pagedata");
            content = obj.getJSONObject("content");

            // servera pusē vienmēr tiek pievienots "safe" komentārs,
            // lai viena komentāra gadījumā uz lietotni netiktu sūtīts
            // masīvs ar komentāra datiem, bet gan objekts; ja nebūs neviena
            // komentāra, savukārt būs tikai safeguard ieraksts (tātad masīvs),
            // tiks izmests JSONException un comments paliks "null"
            comments = obj.getJSONObject("comments");

        } catch (JSONException e) {
            Notify.t(R.string.network_parsing_error);
            return;
        }

        // sarakstā vēl atrodas iepriekšējie ieraksti, ja ielāde ir atkārtota
        arr_comments.clear();
        // saturam, protams, jābūt pirmajā pozīcijā
        arr_comments.add(content);
```

```
166         orderRows(comments, 0, 0);
167
168         LayoutInflater inflater = LayoutInflater.from(activity);
169         ListView lst_comments =
170                 (ListView)inflater.inflate(R.layout.miniblog_comments, null);
171
172         // nospiežot uz kāda no komentāriem,
173         // atvērsies floating context menu ar papildiespējām
174         registerForContextMenu(lst_comments);
175
176         // pārbauda, vai miniblogs nav slēgts, lai zem komentāriem to parādītu
177         boolean is_closed;
178         try {
179             is_closed = content.getBoolean("closed");
180         } catch (JSONException e) {
181             // ja nu kas, labāk, lai nevar komentēt,
182             // nevis iekomentē, kad nedrīkst
183             is_closed = true;
184         }
185
186         // norādīs adapterim, vai pēdējā rindā likt komentēšanas pogu,
187         // vai paziņojumu, ka raksts slēgts
188         JSONObject extra_field = new JSONObject();
189         try {
190             if (is_closed) {
191                 extra_field.put("is_closed", true);
192             } else {
193                 extra_field.put("is_closed", false);
194             }
195         } catch (JSONException e) {
196             Notify.t(R.string.internal_error);
197             return;
198         }
199         arr_comments.add(extra_field);
200         rowid_extra = arr_comments.size() - 1;
201
202         // šāds sazarojums nodrošina to, ka pēc atkārtotas ielādes ritjosla
203         // paliks esošajā vietā, nepārlecot uz pašu augšu, kā būtu, ja
204         // adp_rows katru reizi inicializētu pa jaunu
205         if (adp_rows == null) {
206             adp_rows = new MiniblogComments(activity, arr_comments);
207             lst_comments.setAdapter(adp_rows);
208             lly_miniblog.addView(lst_comments);
209         } else {
210             adp_rows.notifyDataSetChanged();
211         }
212     }
```

Koda fragments skatam ar minibloga komentāra pievienošanas formu.

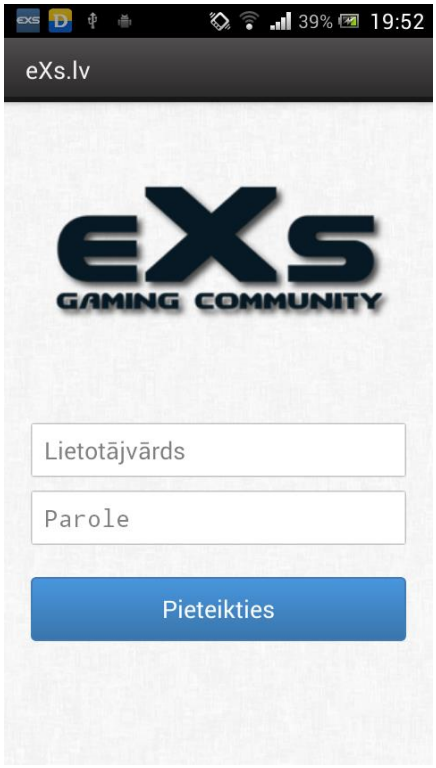
```
1  package lv.exs.android.dialogs;
2
4  *  Minibloga komentāra pievienošanas dialogs
13
14 import lv.exs.android.R;
36
37 public class MbCommentForm extends DialogFragment
38                             implements Network.AsyncListener {
39
40     private View form;
41     private BaseActivity activity;
42
43     // id komentāram, kuram tiek atbildēts; ja neveic atbildēšanu,
44     // bet gan komentē, tad šis sakritīs ar minibloga id
45     private int comment_id;
46     private int miniblog_id;
47
48
49     /**
50      *  Šis jārealizē parent fragmentam, lai tas saņemtu dialoga datus
51      */
52     public interface DialogListener {
53
54         /**
55          *  Paziņo parent fragmentam, ka notika komentāra pievienošana
56          *
57          *  @param status       vai pievienot komentāru izdevās?
58          *  @param error_msg    iespējams pievienošanas kļūdas paziņojums
59          */
60         public void onCommentAdded(boolean status, String error_msg);
61     }
62
63
64     /**
65      *  Saņem atsauci uz aktivitāti un atgriež dialoga izklājumu
66      *
67      *  @param savedInstanceState
68      */
69     @Override
70     public void onCreate(Bundle savedInstanceState) {
71         super.onCreate(savedInstanceState);
72
73         activity = (BaseActivity)getActivity();
74
75         LayoutInflater inflater = activity.getLayoutInflater();
76         form = (View) inflater.inflate(R.layout.miniblog_form, null);
77
78         // formai pievieno atbildamā ieraksta autoru un īsu fragmentu;
79         // pievieno tādēļ, lai vienmēr redzētu, kam atbild, un izvairītos
80         // no nejaušībām, kad nospiež uz nepareizā ieraksta
81         try {
82
83             miniblog_id = getArguments().getInt("miniblog_id");
84             comment_id = getArguments().getInt("comment_id");
85
86             // UserData.colorUser gaida šāda veida objektu ar lietotāja datiem
87             JSONObject author = new JSONObject();
88             author.put("nick", getArguments().getString("nick"));
89             author.put("level", getArguments().getInt("level", 0));
90             author.put("online", getArguments().getBoolean("online", false));
91             author.put("type", getArguments().getInt("type", 0));
92
93             ((TextView)form.findViewById(R.id.mb_form_user))
94                 .setText(UserData.colorUser(author));
95
96             // pārāk garu tekstu rādīt nevar, jo dialogam jau tā
97             // vietas nav pārlieku daudz, atverot klaviatūru
98             String txt_parent = getArguments().getString("text");
```

```
            if (txt_parent == null) {
                txt_parent = "--";
            } else {
                int txt_new_length = txt_parent.length();
                if (txt_parent.length() > 70) {
                    txt_new_length = 70;
                }
                txt_parent = txt_parent.substring(0, txt_new_length) + "...";
            }

            // fragments bija html formātā, turklāt tika apgraizīts,
            // tāpēc noteikti jāizlaiž cauri html funkcijai
            ((TextView)form.findViewById(R.id.mb_form_fragm))
                .setText(Html.fromHtml(txt_parent).toString().trim());

        } catch (NullPointerException e) {
            Notify.t(R.string.error_opening_form);
        } catch (JSONException e) {
            Notify.t(R.string.error_parsing);
        }
    }


    /**
     *  Izveido dialogu
     *
     *  @param savedInstanceState
     *  @return Dialog  pievienošanas formas dialogs
     */
    @Override
    public Dialog onCreateDialog(Bundle savedInstanceState) {

        AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());

        builder.setView(form)

           // pievieno komentāru
           .setPositiveButton(R.string.button_add,
                              new DialogInterface.OnClickListener() {
               @Override
               public void onClick(DialogInterface dialog, int id) {

                   // aizvērs atvērto dialogu
                   getDialog().cancel();

                   postComment();
               }
           })

           // atceļ darbību, aizverot dialogu
           .setNegativeButton(R.string.button_cancel,
                              new DialogInterface.OnClickListener() {
                   @Override
                   public void onClick(DialogInterface dialog, int id) {
                       MbCommentForm.this.getDialog().cancel();
                   }
               }
           );

        // nospiežot ārpus dialoga, tas netiks aizvērts
        // (lai nepazaudētu komentāra tekstu, ja tāds ievadīts)
        setCancelable(false);

        return builder.create();
    }
```

```
156             );
157
158         // nospiežot ārpus dialoga, tas netiks aizvērts
159         // (lai nepazaudētu komentāra tekstu, ja tāds ievadīts)
160         setCancelable(false);
161
162         return builder.create();
163     }
164
165
166     /**
167      *  Sūta komentāru uz serveri
168      */
169     private void postComment() {
170
171         String comment = ((EditText)form.findViewById(R.id.edt_comment_form))
172                             .getText().toString().trim();
173
174         RequestParams params = new RequestParams();
175         params.put("comment_id", Integer.toString(comment_id));
176         params.put("comment", comment);
177
178         Network.post(this, "/miniblogs/" + miniblog_id, params, 0);
179     }
180
181
182     /**
183      *  Tīkla interfeisa metode
184      *
185      *  Tiek izsaukta, kad tīkla pieprasījums saņēmis atbildi.
186      *  Pēc "task" nosaka, kāds uzdevums šo atbildi pieprasīja
187      *
188      *  @param task     uzdevuma identifikators
189      *  @param result   saņemtie dati
190      */
191     public void onTaskComplete(int task, JSONObject result) {
192
193         // pievienots komentārs
194         if (task == 0) {
195             ((MiniblogFragment)getTargetFragment())
196                 .onCommentAdded(true, "");
197         }
198     }
199 }
200
```

SASKARŅU ATTĒLI

1 Pieteikšanās aktivitāte

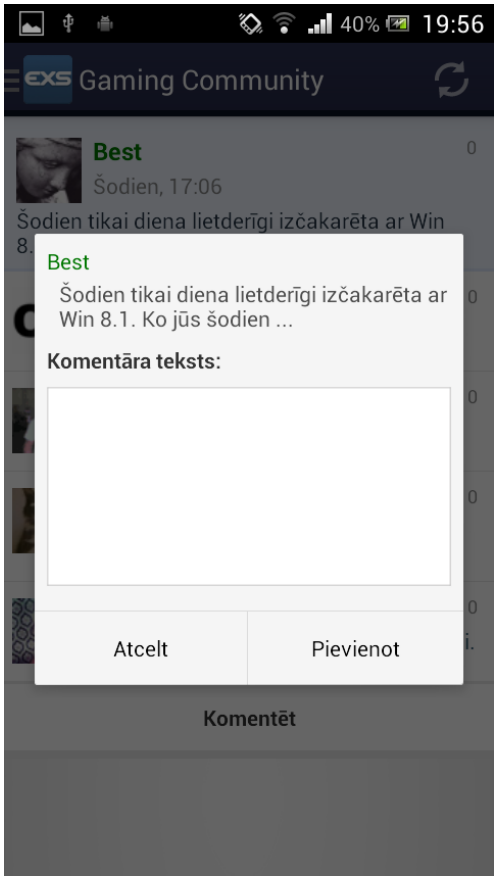


2 Jaunākie miniblogi

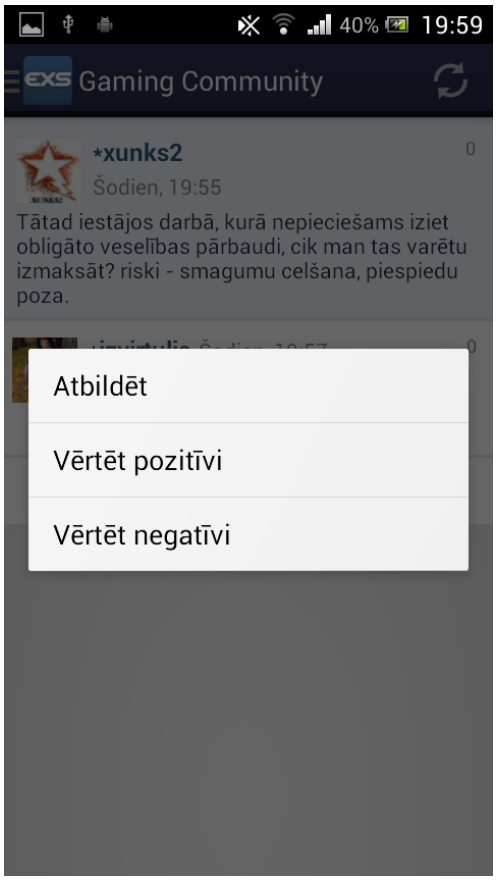
3 Jaunākie raksti

4 Atvērts miniblogs

5 Minibloga komentēšanas forma

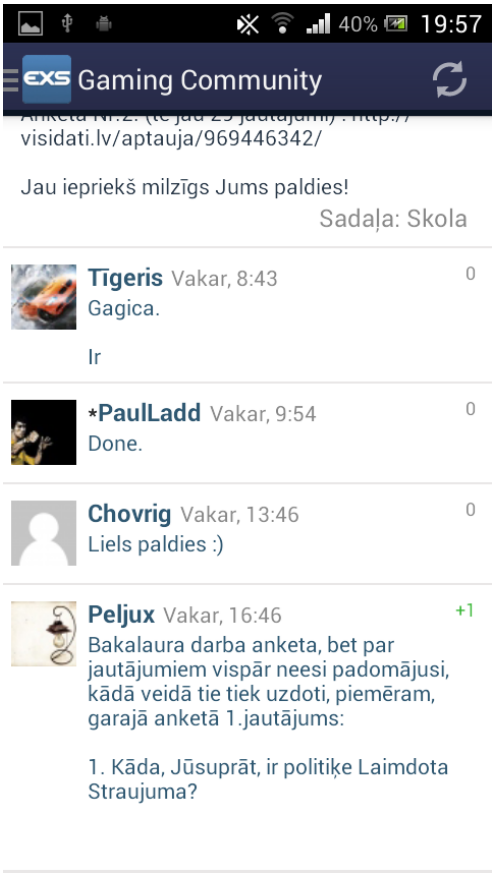


6 Komentāra iespējas



7 Atvērts raksts

8 Raksta komentāri



IZMANTOTĀ LITERATŪRA UN AVOTI

- Lietotnē iekļautās bibliotēkas:
  - „Android Asynchronous Http Client”, © loopj
    http://loopj.com/android-async-http/
  - „Android Smart Image View”, © loopj
    http://loopj.com/android-smart-image-view/

- Java pamācība
  http://www.tutorialspoint.com/java/

- Android lietotņu izstrādes pamācība
  https://developer.android.com/index.html

- LVS 68:1996 „Programmatūras prasību specifikācijas ceļvedis”
  http://estudijas.lu.lv/mod/resource/view.php?id=131427

- LVS 72:1996 „Ieteicamā prakse programmatūras projekta aprakstīšanai”
  http://estudijas.lu.lv/mod/resource/view.php?id=131428

- Android izstrādātāju blograksti (http://android-developers.blogspot.com/)
  http://android-developers.blogspot.com/2013/06/google-play-developer-8-step-checkup.html
  http://android-developers.blogspot.com/2011/06/things-that-cannot-change.html

DOKUMENTĀRĀ LAPA

Kvalifikācijas darbs „Sociālā portāla eXs.lv lietotne Android operētājsistēmai” 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: ***Edgars Peļņa*** _____ **___.06.2014**.

Rekomendēju darbu aizstāvēšanai
Darba vadītājs: ***Dr. dat. Guntis Arnicāns*** _____ **___.06.2014.**

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): _____