

**VENTSPILS AUGSTSKOLA
INFORMĀCIJAS TEHNOLOĢIJU FAKULTĀTE**

BAKALaura DARBS

**Programmatūras izstrādes rīku izmantošana “Ķelmēnu
maiznīcas” sistēmas automatizētai izveidei**

Autors

Ventspils Augstskolas
Informācijas tehnoloģiju fakultātes
bakalaura studiju programmas
„Datorzinātnes”
3. kursa students
Mārtiņš Bērtulis
Matr.nr. 12020026

(paraksts)

Fakultātes dekāns

asoc.prof., Dr. math. Gaļina Hilkeviča

(paraksts)

Zinātniskais vadītājs

lekt., Dr.sc.ing. Raita Rollande

(paraksts)

Recenzents

(ieņemamais amats, zinātniskais nosaukums, vārds, uzvārds)

(paraksts)

Ventspils
2015

ANOTĀCIJA

Darba nosaukums: Programmatūras izstrādes rīku izmantošana “Ķelmēnu maiznīcas” sistēmas automatizētai izveidei

Darba autors: Mārtiņš Bērtulis

Darba vadītājs: Dr.sc.ing. Raita Rollande

Darba apjoms: 55 lpp., 0 tabulas, 20 attēli, 74 bibliogrāfiskās norādes, 4 pielikumi.

Atslēgas vārdi: SISTĒMAS IZSTRĀDE, SISTĒMAS IZSTRĀDES RĪKI, PROJEKTU VADĪBA

Bakalaura darbā ir apskatīti sistēmas izstrādes rīki, kas izmantoti “Ķelmēnu maiznīcas” sistēmas automatizētai izveidei. Darbā gaitā apgūti un aprakstīti dažādi sistēmas izstrādes rīki, piemēram, “GitHub”, “Asana”, “Toggl”, “Hound”, “Ruby RSpec”, “Travis CI”, “Heroku”, “Capybara”, “Divise”, “Teatro” un “Vagrant”, kas ir ļoti noderīgi sistēmu izstrādes automatizēšanā. Darba gaitā apgūts un izmantots “Agile” “Scrum” izstrādes modelis.

Aprakstīta arī izstrādātā sistēma, tās datu un saskarnes struktūra. Apskatīts projektu izstrādes dzīves cikls un izstrādes metodes, kā arī informācijas tehnoloģijas projektu vadība. Darba nobeigumā parādīti projektā “Demola Latvia” izstrādātās sistēmas ekrānu uzņēmumi, kā arī sniegts to apraksts.

Bakalaura darbs sastāv no ievada, attēlu saraksta, piecām nodaļām un secinājumiem. Tajā ir 55 lappuses, 20 attēli, 74 bibliogrāfiskās norādes un 4 pielikumi.

Bakalaura darbu var izmantot par paraugu līdzīgu sistēmu veidošanā, kā arī apskatot izmantotos rīkus un izmantojot tos citos projektos.

Bakalaura darba rezultātā ir izstrādāta daļēji pabeigta “Ķelmēnu maiznīcas” ražošanas uzskaites sistēma.

ANNOTATION

Title: Using Software Development Tools for the Automated Development of System for *Ķelmēnu maiznīca*

Author: Mārtiņš Bērtulis

Academic Advisor: Dr.sc.ing. Raita Rollande

Volume: 55 pages, 0 tables, 20 images, 74 sources of literature, 4 appendices.

Keywords: SYSTEM DEVELOPMENT, SYSTEM DEVELOPMENT TOOLS, PROJECT MANAGEMENT

This Bachelor's Paper deals with software development tools that have been used for the automated development of a system for *Ķelmēnu maiznīca*. Various system development tools that are very important for the automation of system development, including *GitHub*, *Asana*, *Toggl*, *Hound*, *Ruby RSpec*, *Travis CI*, *Heroku*, *Capybara*, *Divise*, *Teatro* and *Vagrant*, have been acquired and characterised. The *Agile Scrum* development model has also been acquired and applied.

It also explores the system that was developed as well as the structure of its data and interface. Moreover, it features a description of project development life cycle, methods of development and information technology project management. The final chapter provides screenshots of the system that was developed in the course of a *Demola Latvia* project as well as the corresponding descriptions.

This Bachelor's Paper consists of an introduction, a table of figures, five chapters and conclusions. It contains 55 pages, 20 images, 74 sources of literature and 4 appendices.

This Bachelor's Paper can serve as a fundament for developing similar systems as well as for characterising the tools that have been described and using them in other projects.

The result of this Bachelor's Paper is the development of a partly finished production record system for *Ķelmēnu maiznīca*.

SATURA RĀDĪTĀJS

ATTĒLU SARAKSTS	5
IEVADS.....	6
1. INFORMĀCIJA PAR PROJEKTU “DEMOLA LATVIA”	8
2. INFORMĀCIJAS TEHNOLOĢIJAS PROJEKTI UN TO VADĪŠANA	13
2.1. Informācijas tehnoloģijas projekti.....	13
2.2. Informācijas tehnoloģijas projektu vadība	14
2.3. IT projektu izstrādes modeļi.....	16
3. SISTĒMAS IZSTRĀDES RĪKI	21
3.1. Rīku iespējas informācijas sistēmas izstrādē	21
3.2. Biežāk izmantotie sistēmas izstrādes rīki.....	23
3.3. Projektā izmantotie sistēmas izstrādes rīki	27
4. “ĶELMĒNU MAIZNĪCAS” RAŽOŠANAS UZSKAITES SISTĒMAS ANALĪZE UN PROJEKTĒJUMS.....	33
4.1. Ķelmēnu maiznīcas ražošanas uzskaites sistēmas prasības	33
4.2. Sistēmas datu struktūras projektējums	34
4.3. Saskarnes projektējums	36
5. “ĶELMĒNU MAIZNĪCAS” RAŽOŠANAS UZSKAITES SISTĒMAS IZSTRĀDE UN IEVIEŠANA	42
5.1. “Ķelmēnu maiznīcas” ražošanas uzskaites sistēmas izstrāde	42
5.2. Rezultātu apraksts	42
SECINĀJUMI UN PRIEKŠLIKUMI.....	44
IZMANTOTĀS LITERATŪRAS UN AVOTU SARAKSTS	46
PIELIKUMI	51
GALVOJUMS	55

ATTĒLU SARAKSTS

1.1. att. Projekta iedalījums posmos	9
1.2. att. “Demola Latvia” attīstības posmi [1]	9
2.1. att. Projektu realizēšana [6]	15
2.2. att. Projektu pabeigšana [6]	16
2.3. att. Īdēnskrituma modelis [9]	17
2.4. att. “Agile” “SCRUM” metodes cikls [15]	18
2.5. att. Īdēnskrituma metodes projekta izdošanās salīdzinājums ar “Agile” metodi [17]	19
2.6. att. Izstrādes metožu salīdzinājums [18]	20
3.1. att. Izstrādes rīku sadalījums	21
4.1. att. “Ķelmēnu maiznīcas” darba plūsmas diagramma	34
4.2. att. Datu struktūras diagramma	35
4.3. att. Lietošanas gadījumu diagramma	36
4.4. att. Sistēmas sākuma lapa	37
4.5. att. Izejmateriālu logs	38
4.6. att. Izejvielu logs	38
4.7. att. Maizes cepšanas procesa logs	39
4.8. att. Produktu logs	39
4.9. att. Darbinieku ekrānu logs	40
4.10. att. Darbinieku ekrānu apskates logs	40
4.11. att. Darbinieku ekrāna izveide	41

IEVADS

Tēmas aktualitātes pamatojums

Mūsdienās daudzos uzņēmumos projektu vadītāji, sistēmu analītiķi, programmētāji un testētāji neizmanto visas mūsdienu tehnoloģiju piedāvātās priekšrocības, kas varētu ievērojami atvieglot un automatizēt informācijas tehnoloģijas projektu izstrādi [1]. Tā kā projektu izstrādes koordinēšana ir būtiska daļa no informācijas tehnoloģijas projektu vadītāju ikdienas pienākumiem, lai atvieglotu savu ikdienas darbu, projektu vadītājiem ir jāpārvalda dažādi sistēmas izstrādes rīki un metodes. Tāpat ir daudz dažādu rīku, ko izstrādes automatizēšanas nolūkos var izmantot arī sistēmu analītiķi, programmētāji un testētāji [2].

Joprojām ir uzņēmumi, kur sistēmas izstrādes posmi un tajos veicamie uzdevumi tiek veikti manuāli, taču, izmantojot sistēmas izstrādes rīkus, šos darbus varētu veikt daudz kvalitatīvāk un ātrāk [1].

Darba mērķis ir izpētīt dažādus programmatūras izstrādes rīkus un praktiski izmantot tos “Ķelmēnu maiznīcas” sistēmas automatizētai izveidei.

Lai sasniegtu šo mērķi, ir izvirzīti šādi **darba uzdevumi**:

1. Apskatīt projektu izstrādes dzīvescikla stadijas un metodes.
2. Noskaidrot, kādi ir populārākie sistēmas izstrādes rīki.
3. Iepazīties ar “Demola Latvia” projekta koncepciju un sistēmas izstrādes komandu.
4. Iepazīties ar “Ķelmēnu maiznīcas” problēmvidi.
5. Izmantot dažāda veida sistēmas izstrādes rīkus “Ķelmēnu maiznīcas” sistēmas automatizētai izveidei.
6. Aprakstīt sistēmas izstrādes rīkus, kas tika izmantoti “Ķelmēnu maiznīcas” sistēmas izveidei.
7. Aprakstīt “Ķelmēnu maiznīcas” ražošanas uzskaites sistēmas struktūru.
8. Aprakstīt “Ķelmēnu maiznīcas” sistēmas izstrādes procesu un ieviešanu.
9. Aprakstīt “Ķelmēnu maiznīcas” sistēmas gala rezultātu.
10. Aprakstīt, kā izmantotos “Ķelmēnu maiznīcas” sistēmas rīkus iespējams izmantot citu sistēmu izstrādē un kā tas uzlabo izstrādātāju darbu.

Tēmas struktūras apskats

Darba teorētiskajā daļā sniegta informācija par “Demola Latvia” projektu, jo darbā aprakstītā sistēma tika izstrādāta projekta laikā. Tāpat šajā sadaļā autors sniedz viedokli par dalību projektā, kā arī apraksta sistēmas izstrādes komandu. Otrkārt, darba teorētiskajā daļā tiek sniegts ieskats informācijas tehnoloģijas projektos, projektu vadītāju profesijā un projektu vadītāju lomā informācijas tehnoloģijas projektu izstrādē, kā arī divas populārākās projektu izstrādes metodes. Treškārt, darba teorētiskajā daļā tiek aprakstītas sistēmas izstrādes rīku iespējas informācijas sistēmas projektu izstrādē, kā arī to iedalījums.

Darba praktiskajā daļā atrasti un izpētīti biežāk izmantotie sistēmas izstrādes rīki, kā arī aprakstīti rīki, kas tika izmantoti “Ķelmēnu maiznīcas” sistēmas izveidē. Aprakstīta arī “Ķelmēnu maiznīcas” ražošanas uzskaites sistēmas analīze un struktūra. Praktiskās daļas nobeigumā aprakstīta “Ķelmēnu maiznīcas” ražošanas uzskaites sistēmas izveide un gala rezultāts.

Darba pielikumā iespējams aplūkot abus autora projektā “Demola Latvia” iegūtos diplomus — vienu par dalību projektā, savukārt otru — par pirmās vietas iegūšanu. Lai visas sistēmas funkcijas būtu labāk izprotamas, pielikumā ievietoti arī dažādi izveidotās sistēmas ekrānuzņēmumi.

1. INFORMĀCIJA PAR PROJEKTU “DEMOLA LATVIA”

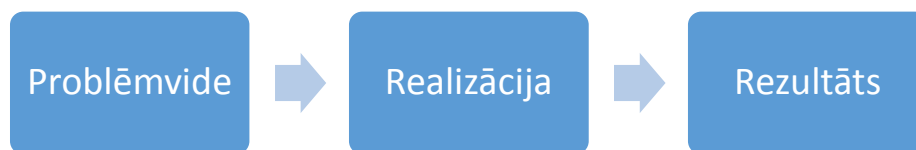
“Demola Latvia” pirmsākumi meklējami Somijā 2008. gadā, taču pašlaik šis projekts ir pārtapis par atvērto inovāciju platformu, kur satiekas studenti un uzņēmēji, un tas darbojas vairākās valstīs, tostarp kopš 2014. gada sākuma arī Latvijā. Pašlaik “Demola Latvia” projekti tiek realizēti deviņās valstīs: Somijā (Oulu, Tampere), Lietuvā (Viļņa), Ungārijā (Budapešta), Slovēnijā (Maribora), Latvijā (Rīga), Krievijā (Sanktpēterburgā), Zviedrijā (Malmo, Nordkopingā), Spānijā (Bilbao), kā arī Meksikā (Guadalaharā) [3]. “Demola Latvia” projekta mērķis ir veicināt uzņēmējdarbības un inovāciju attīstību, starpdisciplināritāti, kā arī dzīvotspēju tautsaimniecībā kopumā. “Demola Latvia” sniedz uzņēmējiem iespēju saņemt inovatīvus un neparastus aktuālu problēmjaudājumu risinājumus, nereti no jauna, neierasta skatupunkta, savukārt studenti, strādājot jauktās komandās (katras komandas sastāvā ir 5–6 dalībnieki; vienu izglītības iestādi un nozari nedrīkst pārstāvēt vairāk kā divi dalībnieki), meklē radošus risinājumus reāliem dažādu nozaru izaicinājumiem [3].

Tāpat kā Ventspils Augstskolā, arī “Demola Latvia” projektā ir divi semestri — pavasara un rudens semestris. Pavasara semestris sākas 25. martā, un tā noslēgums ir 30. jūnijā. Savukārt rudens semestris ilgst no oktobra vidus līdz janvāra pēdējai nedēļai. Lai palīdzētu studentu grupai nonākt pie veiksmīga rezultāta, “Demola Latvia” pārstāvji nodrošina darba plānu, metodiku, mentoru atbalstu, izglītojošus seminārus, kā arī organizē tikšanās reizes ar nozaru ekspertiem un konsultantiem.

2014. gada rudens semestrī tika rīkoti šādi pasākumi: sezonas atklāšana jeb “kick-off”, iepazīšanās ar komandu un projektu, projekta sākšanas un plānošanas seminārs, “DEMOLA JAM” — pilnas dienas praktisks ideju ģenerēšanas seminārs, “No-Slide Pitch” — pirmo ideju prezentēšana, “Mid-Pitch” — starprezultātu prezentēšana, “Final Pitch” un izlaidums. Pēc komandu iniciatīvas notiek arī citas tikšanās reizes, kuru vajadzībām ir iespējams izmantot “Demola Latvia” telpas.

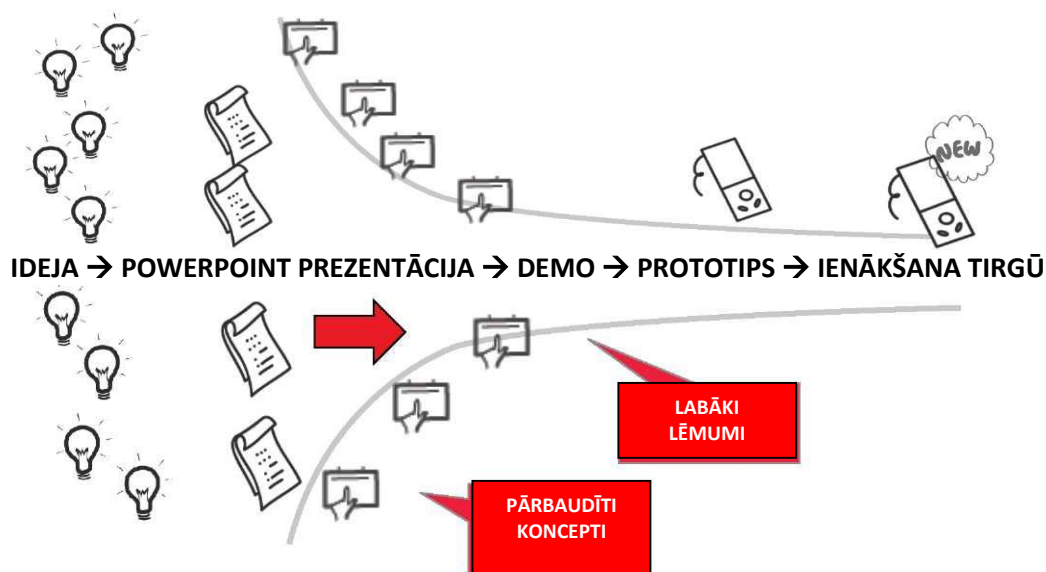
“Demola Latvia” projektus var iedalīt trīs posmos. Vispirms uzņēmums studentu komandu iepazīstina ar problēmu jeb izaicinājumu, ko studentu komanda pēc tam risina. Ja projekta beigās rezultāts uzņēmumu apmierina, tad uzņēmuma darbinieki var licencēt kopīgas tiesības. Jāpiebilst, ka nereti darba gaitā uzdevumi un prioritātes mainās, tāpēc izstrādātais risinājums bieži atšķiras no sākotnēji iecerētā. Tomēr attiecībā uz atvērtajām inovācijām šī nav neierasta parādība. Proti, projekta vide, tostarp komandas redzējums un

pieeja, nereti atšķiras no uzņēmuma vides, līdz ar to sākotnējās idejas iegūst jaunus apveidus, turklāt tās var izmantot arī citās jomās un procesos. 1.1. attēlā redzams projekta iedalījums posmos.



1.1. att. Projekta iedalījums posmos

Piedaloties “Demola Latvia” projektā, dalībnieki var iegūt unikālu praktisko pieredzi jaunu produktu vai risinājumu izstrādē, apgūt jaunas iemaņas, kā arī uzlabot esošās iemaņas (piemēram, angļu valodas un publiskās uzstāšanās prasmes, radošo domāšanu un prasmi darboties starpdisciplinārā komandā, mācoties citam no cita). Viens no būtiskākajiem ieguvumiem ir biznesa kontakti un darba iespējas, kā arī iespēja pēc projekta beigām — ja izstrādātais risinājums ir bijis veiksmīgs — komercializēt izstrādātās idejas un saņemt atlīdzību no uzņēmuma. 1.2. attēlā redzami “Demola Latvia” attīstības posmi.



1.2. att. “Demola Latvia” attīstības posmi [3]

Organizatori uzskata, ka studentus motivē arī tas, ka projekts ir īsts, tas ir, viņu darbs tiks reāli izmantots reālā uzņēmumā vai valsts iestādē. Vēl viens ieguvums ir personīgā izaugsme un labi pamati tiem “Demola Latvia” dalībniekiem, kas apsver sava uzņēmuma izveidi [3].

Galvenais studentu atlases kritērijs ir motivācija. Proti, studentiem jāizvēlas problēma, kas viņus interesē, un jāpiesakās konkrētam projektam. Projekti, kuros iespējams

pieņemot, ir atrodamas "Demola Latvia" mājaslapā [4]. Lai palīdzētu atrast piemērotāko projektu, ir iespējams veikt meklēšanu, atlasot projektam nepieciešamās prasmes, tādējādi izkristalizējas cits būtisks aspekts, tas ir, studentu komandas tiek veidotas, ņemot vērā kopīgas intereses. Savienojot abus atlasas aspektus, rodas komanda, kuras dalībniekus ne vien interesē viena tēma, bet viņiem arī ir personiska motivācija atrisināt konkrētu nozares problēmu.

Kā jau minēts, viss projektā paveiktais ir grupas īpašums, tas ir, īpašumtiesības pieder studentu komandai, un uzņēmējs drīkst izmantot grupas darbu, tikai iegādājoties licenci. Licences tiek iedalītas trīs kategorijās:

- 1) darbā ir paveikts nepieciešamais;
- 2) darbā ir paveikts nepieciešamas, un iekļautas papildu idejas;
- 3) darbā ir īstenota jaunākā tehnoloģija, un rezultāts ietver daudz jaunu un noderīgu ideju.

Katrai no šīm kategorijām ir pielīdzināms zināms atalgojums, ko komanda saņem un pēcāk sadala. Katrs komandas dalībnieks saņem vienlīdzīgu daļu.

Problēmju risināšanu "Demola Latvia" ir uzticējuši tādi atpazīstami uzņēmumi un iestādes kā "Intel", "Nokia", "Nokian", Helsinku pilsētas dome, "SAAB", "Sony", "Canon", kā arī Latvijas Valsts zemes dienests, "SEB banka", "1188", "Daugavpils ložu rūpnīca", „Ķelmēnu maiznīca” kā arī daudzi citi tehnoloģiju, finanšu, būvniecības, loģistikas un ražošanas uzņēmumi. Četras piektdaļas projektu iesniedzēju ir atzinuši, ka iegūtie rezultāti ir vērtīgi un izmantojami biznesā [3]. Tāpat "Demola Latvia" projekti ir sekmējuši inovāciju tīklu attīstību, īpaši starp augstākās izglītības iestādēm, uzņēmumiem un valsts sektora iestādēm, kas savukārt palīdzējis veicināt zināšanu pārnesei un ļāvis tās izplatīt [3].

Tā kā viens no „Demola Latvia” principiem ir novatoriskas metodes, kas arī ir būtiskas augstāko izglītības iestāžu studiju programmās, ir pamats uzskatīt, ka arī augstskolas un universitātes gūst labumu no šādas zināšanu pārneses, jo "Demola Latvia" pietuvina augstākās izglītības iestādēm tādus procesus kā produktu jauninājumi un komercializācija [3]. Piemēram, Somijas universitāšu mācībspēki ir secinājuši, ka "Demola Latvia" projekti ir veicinājuši to, ka valsts universitātēs daudz vairāk uzmanības tiek pievērsts jauninājumiem, kā arī palīdzējuši padziļināt kontaktus un līdz ar to arī sadarbību ar uzņēmējdarbības sektoru [3]. Tā kā Somijas augstākās izglītības sistēma ir atzīta par vienu no kvalitatīvākajām pasaulē un ieņem sesto vietu valstu reitingā, šī valsts ir ļoti labs piemērs ar izglītību saistītos jautājumos [5]. Cits vērtīgs ieguvums ir reģionālo augstskolu sadarbības

veicināšana. “Demola Latvia” pavērs jaunas mācību iespējas, sniedzot studentiem iespēju apgūt praktisku pieeju problēmu risināšanai, kā arī darba pieredzi.

“Demola Latvia” projekta iniciators ir Latvijas IT klasteris, savukārt sadarbības vienošanos parakstījušas 11 augstskolas un četras profesionālās organizācijas. “Demola Latvia” projekts notiek LIAA īstenotās programmas “Pasākumi motivācijas celšanai inovācijām un uzņēmējdarbības uzsākšanai” un Rīgas domes Pilsētas attīstības departamenta projekta “Working 4 Talent” ietvaros, un to līdzfinansē Eiropas Savienība un Eiropas Reģionālās attīstības fonds [3].

2014. gada rudens semestrī projektā “Demola Latvia” iesaistījās arī darba autors. Par iespēju piedalīties šajā pasākumā autors uzzināja, atsaucoties Ventspils Augstskolas mācību prorektora Kārļa Krēsliņa aicinājumam kopā ar viņu doties uz jauna projekta prezentāciju. Pēc šīs prezentācijas autors saprata, ka noteikti vēlētos piedalīties šajā projektā, jo tas sniegtu praktiskas iemaņas informācijas tehnoloģiju projektu vadībā un izstrādē. Autors izvēlējās projektu “Bakery Watchdog”, jo tas šķita visai interesantāks, turklāt šķita, ka tas ļautu iegūt vērtīgu pieredzi projektu izstrādē un vadībā, kas varētu noderēt un sniegt priekšrocības kā darba tirgū, tā veidojot karjeru. No pieteikšanās līdz pozitīvas atbildes saņemšanai pagāja vairāk nekā divas nedēļas. Autora komandā bija seši dalībnieki: divi Rīgas Tehniskās Universitātes studenti (studiju programmas “Dizains un māksla” un “Informāciju tehnoloģijas”, ko apguva ERASMUS students no Francijas), viens dalībnieks no Latvijas Lauksaimniecības Universitātes (studiju programma “Lauksaimniecības inženierzinātnes”), Rīgas biznesa skolas (studiju programma “Ekonomika un vadība”), kā arī students no Uzbekistānas, kas Biznesa augstskolā “Turība” apguva tūrisma.

Tā kā komandas sastāvā bija divi ārvalstu studenti, visa saziņa notika tikai angļu valodā, kas arī bija liels ieguvums. Laika gaitā palielinājās komandas sadarbības prasmes. Darba gaitā notika neskaitāmas tikšanās reizes, tostarp divas reizes komanda devās uz Vidzemes austrumu pilsētu Ranku, kur atrodas “Ķelmēnu maiznīca”, lai klātienē redzētu, kā tiek ražota “Ķelmēnu maiznīcas” maize, un vairotu izpratni par to, kā atrisināt problēmu. Otrajā maiznīcas apmeklēšanas reizē tika aptaujāti darbinieki, lai noskaidrotu prasības pret izstrādājamo sistēmu un varētu padarīt to viņiem parocīgāku, tajā pašā laikā iekļaujot tajā visu uzņēmumam nepieciešamo. Pirmās komandas tikšanās reizes aizritēja, spriežot par prasību izzināšanu un analīzi, kā arī sistēmas datu un saskarnes projektēšanu. Beidzamajās tikšanās reizēs komanda lēma par nepieciešamajām sistēmas izmaiņām un uzlabojumiem. Vērts pieminēt, ka autors ar savu komandu noslēguma prezentācijā pēc žūrijas vērtējuma ieguva pirmo vietu un uzvarēja konkursā. Pēc projekta prezentēšanas “Ķelmēnu maiznīcas” pārstāvjiem viņi nolēma iegādāties produkta licenci, tāpēc, kā minēts iepriekš, katrs

komandas dalībnieks saņēma 20 % no kopējās summas, jo projekta laikā viens no komandas dalībniekiem no dalības projektā atteicās. Abus autora diplomus — par dalību projektā un par pirmās vietas izcīnīšanu — var skatīt pielikumā Nr. 1 un pielikumā Nr. 2.

Piedaloties projektā, darba autors uzlaboja savas angļu valodas un saziņas, kā arī prezentācijas prasmes. Autors noteikti iesaka visiem studentiem piedalīties šajā projektā, jo tā ir lieliska iespēja strādāt pie reāliem projektiem, risināt reālas, uzņēmumiem aktuālas problēmas un, ja galarezultāts ir veiksmīgs, saņemt zināmu atalgojumu.

2. INFORMĀCIJAS TEHNOLOĢIJAS PROJEKTI UN TO VADĪŠANA

2.1. Informācijas tehnoloģijas projekti

Projekta definīcija — “projekts ir process, kurā ir vairāki posmi ar skaidri definētu un apstiprinātu mērķi vai mērķiem, kuri ir jāsasniedz līdz noteiktam laikam, izmantojot ierobežotus resursus” [6].

“Projektu būtība ir darbības esamība. Tas ir gan darbs, gan būvēšana un pārbūvēšana, gan sasniegumi, gan produkcijas izlaide, gan pats darbības rezultāts.” [7]

Projektam parasti ir šādas iezīmes [6]:

- 1) projektam ir skaidri izvirzīts mērķis;
- 2) projekta mērķa sasniegšanai ir paredzēti noteikti resursi;
- 3) projektam ir zināms sākuma datums, un ir noteikts beigu datums;
- 4) projekta mērķis ir unikāls vismaz vienas organizācijas ietvaros;
- 5) projektam ir vairāki posmi.

Uz “Demola Latvia” projektiem ir attiecināmi visi punkti. “Demola Latvia” projekta sākumā autora komandai tika izvirzīts mērķis modernizēt “Ķelmēnu maiznīcas” ražošanas uzskaiti. Lai sasniegtu mērķi, tiek izveidota piecu cilvēku komanda, un cilvēku skaits nevar pieaugt; tas var tikai samazināties, ja kāds no komandas locekļiem atsakās no dalības projektā. “Demola Latvia” projektiem ir noteikts sākuma datums. Piemēram, autora projekts sākās 9. oktobrī, savukārt tā noslēgums bija 29. janvārī, kad bija jāprezentē četros mēnešos paveiktais, kā arī darba rezultāts. Līdz šim uzņēmumam “Ķelmēnu maizīca” sistēmas nebija izstrādātas, tāpēc šī projekta mērķis – izveidot ražošanas uzskaites sistēmu, “Ķelmēnu maiznīcai” noteikti bija unikāls. “Demola Latvia” projektam bija vairāki posmi. Vispirms bija jānoskaidro prasības un jāveic to analīze. Turpinājumā notika darbs pie sistēmas koncepcijas un dizaina izstrādes, pēc tam notika programmēšana un izveidotās sistēmas testēšana. Tā kā projektam tika izvēlēta “Agile” izstrādes metode, visi šie posmi atkārtojās.

Informāciju tehnoloģiju projektus parasti var iedalīt šādos posmos [6]:

- 1) sistēmas prasību apkopošana un analīze;
- 2) sistēmas koncepcijas un dizaina izstrāde;
- 3) kodēšana (koda rakstīšana), sistēmas izveide;
- 4) testēšana;
- 5) ieviešana;
- 6) uzturēšana.

“Demola Latvia” projekta ietvaros autora komanda īstenoja pirmos četrus posmus. Tā kā autora komanda darbojās pēc “Agile” metodes, šie posmi tika veikti vairākas reizes, tomēr sistēma netika pilnībā pabeigta, līdz ar to tā netika ieviesta un uzturēta. Sistēmas prasību apkopošanai komanda devās uz “Ķelmēnu maiznīcu”, lai klātienē redzētu maizes ražošanas procesu un vairāk izprastu pasūtītāja vēlmes. Pēc nepieciešamās informācijas iegūšanas tika veikta datu analīze, pēc kuras komanda noskaidroja, kādus datus būs nepieciešams izmantot ražošanas uzskaites sistēmā. Tika izveidoti lietotāja saskarnes prototipi, kas pēcāk tika parādīti maiznīcas darbiniekiem. Pēc prasību apkopošanas tika sākta sistēmas izstrāde. Pēc katras izmaiņas veikšanas sistēma tika testēta, lai pārliecinātos, ka nav pieļautas kļūdas vai nepilnības.

IT projektu ilgums var būt no vienas dienas līdz pat vairākiem gadiem. Tas ir atkarīgs no projekta apjoma un mēroga, iesaistīto resursu skaita un projekta sarežģītības pakāpes. Līdz ar to projektā ļoti svarīgs un laiktietilpīgs ir sākotnējais posms, proti, sistēmas prasību apkopošana un analīze, kā arī sistēmas koncepcijas un dizaina izstrāde, jo turpmāk visi darbi tiks veikti, balstoties tikai uz šajos divos posmos iegūto informāciju.

IT projektu izstrādē tiek izmantotas dažādas metodes, vairāk par izmantotajām metodēm skatīt sadaļā 2.3.). Īsākos projektos parasti tiek izmantota “Agile” metode (skatīt sadaļā 2.3.), kur darbi tiek veikti, izveidojot sākuma variantu, un pēc tam, to uzlabojot, tiek iegūts gala rezultāts.

Jebkurā projektā ļoti būtiski ir atrast pareizos risinājumus projektu vadīšanai. Pareiza projektu vadīšana var ievērojami ietaupīt gan laiku, gan līdzekļus, turpretim slikta — tieši pretēji — kavēt nodošanas grafikus un palielināt izmaksas.

2.2. Informācijas tehnoloģijas projektu vadība

Projektu vadības definīcija — “projekta vadīšana ir cilvēku grupas mērķtiecīga darbība saskaņā ar noteiktu plānu un prasībām projekta mērķa vai mērķu sasniegšanai” [6].

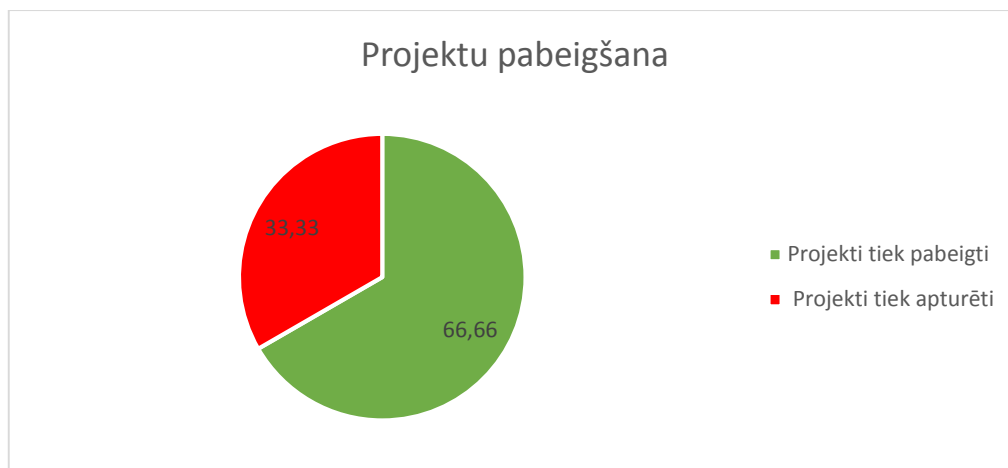
“Projektu vadība ir sarežģīts darba veids, un informācijas tehnoloģijas projektu vadībā tiek izmantotas jau kopš pagājušā gadsimta piecdesmitajiem gadiem, kad aprēķinu veikšanai tika izmantoti lieldatori (angļu valodā — “mainframe”)” [8]. Taču lieldatorus izmantoja tikai aprēķinu veikšanai lielajos projektos [8]. Lai, piemēram, uzzīmētu struktūrplānu, šajā laikā projektu vadībā izmantoja papīru, zīmuli un lineālu [8]. Straujas tehnoloģiju attīstības dēļ šobrīd projektu vadība nav iedomājama bez datora un dažādu rīku izmantošanas.

Informācijas tehnoloģijas projektu vada projektu vadītājs, kas ir kā tilts starp pasūtītāju un izstrādes daļas darbiniekiem. Projektu vadītājiem ir jātiecas ar pasūtītāju un pēc iespējas detalizētāk jācenšas izziņāt visas nepieciešamās prasības, lai projekta beigās galarezultāts būtu tieši tāds, kā klients vēlēties. Tāpat jāveic dažādi izpētes procesi, lai varētu prognozēt projekta darbietilpību un izmaksas. Nereti projektam vadītājam nākas apjomīgus darbus sadalīt sīkākos uzdevumos, lai programmētājiem būtu saprotami visi uzdevumi. Jo mazāki un konkrētāki ir uzdevumi, jo vieglāk kontrolēt padarīto darbu un pārbaudīt, vai padarītie darbi ir paveikti pareizi. Projektam vadītājam ir jāspēj uzraudzīt visus programmētājus, to pabeigtos darbus, kā arī jāpārbauda darba rezultāti, lai pārliecinātos, ka nav pieļautas kļūdas vai nepilnības. Projektam vadītājs ir atbildīgs par projekta starprezultātu pabeigšanu, kā arī projekta nodošanu laikā. 2.1. attēlā redzams projektu realizēšanas grafiks. Kā redzams grafikā tikai desmit procenti no informācijas tehnoloģijas projektiem tiek realizēti tieši tā, kā sākumā iecerēti. Pārējie 90 procenti saskaras ar lielākām, vai mazākām problēmām. No šī attēla var secināt, ka ļoti liela nozīme ir precīzu mērķu un prasību sastādīšanai, lai tos būtu iespējams realizēt kā iecerēts un tam paredzētajā laikā.



2.1. att. Projektu realizēšana [9]

Izstrādes laikā tiek pārtraukts katrs trešais informācijas tehnoloģijas projekts. Projekti tiek pārtraukti, ja mainās prioritātes un prasības, kā arī projekta neveiksmes dēļ [6]. 2.2. attēlā redzams grafisks projektu pabeigšanas attēlojums. Kā redzams attēlā tikai divas trešdaļa no iesāktajiem informācijas tehnoloģijas projektiem tiek pabeigti, bet viena trešdaļa informācijas tehnoloģijas projekti tiek apturēti un paliek nepabeigti. No šī attēla var secināt, ka pirms projektu izstrādes sākuma ir ļoti nopietni jāapsver visi iespējamie riski, kas varētu būt par iemeslu, lai projekts būtu jāpārtrauc.



2.2. att. Projektu pabeigšana [9]

Projektu vadīšanai ir šādi nosacījumi [6]:

- 1) projekta vadība ir darbība saskaņā ar noteiktām prasībām un noteiktu plānu;
- 2) projekta vadība ir darbība, kas orientēta uz projekta mērķa vai mērķu un starpmērķu sasniegšanu;
- 3) projektu vadībā tiek lietotas noteiktas metodes, darbarīki un metodikas;
- 4) projektu vadība vienmēr ir saistīta ar lielākas vai mazākas darba grupas organizēšanu.

Informācijas tehnoloģijas projekti nereti ir pakļauti dažādām grūtībām, tāpēc ir ļoti svarīgi izvēlēties pareizo projekta izstrādes modeli.

2.3. IT projektu izstrādes modeļi

Informāciju tehnoloģiju projekti var atšķirties pēc darbības attīstības veida. Ir daudz uz dažādi varianti, kam pamatā ir divi būtiski atšķirīgi IT projektu darba modeļi, ko raksturo tā dēvētā „ūdenskrituma” [10] un „Agile” [11] jeb ātro prototipu veida darbības attīstība. „Ūdenskrituma” gadījumā raksturīgas izteiktas robežzīmes starp darbu posmiem un visu darbu secīgu izpildi, savukārt „Agile” gadījumā izteiktu atsevišķu posmu sākuma un noslēguma punktu nav.

Ūdenskrituma modeli sāka izmantot 20. gadsimta 70. gadu sākumā, kad šī metode parādījās V. Roisa (W. Royce) publikācijā, kas tika izmantota, lai attēlotu iespēju formalizēt izstrādes procesus [12]. Lai izmantotu ūdenskrituma modeli, ir nepieciešama ļoti apjomīga sākotnējā analīze un ļoti detalizēta informācija par nepieciešamajām prasībām no pasūtītāja. Ūdenskrituma tipa projektos pārejas punkti no vienas fāzes uz otru ir uzskatāmi par atskaites un vērtējuma punktiem jeb robežzīmēm, kas raksturo projekta attīstību. Izmantojot

ūdenskrituma modeli, ir liela iespēja ļoti precīzi sastādīt darba plānu, tā izpildes termiņus, plānoto darbības attīstību, kā arī turpmākos projekta uzlabojumus, jo visas veiktās darbības tiek stingri dokumentētas.

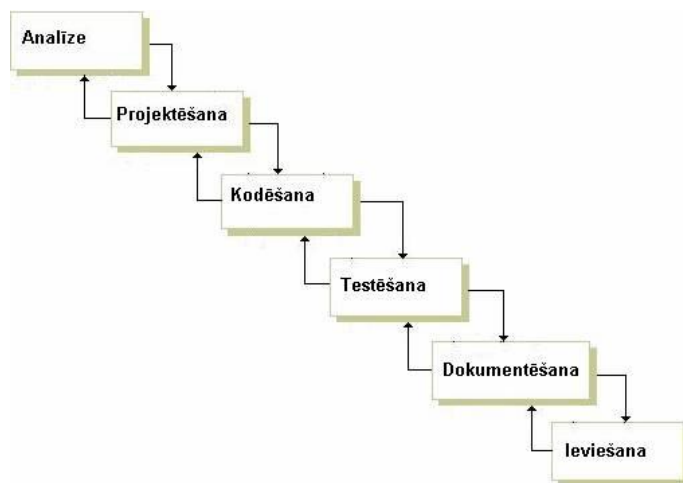
Metodes priekšrocības [13]:

- analogiski kā būvējot māju, visas darbības tiek veiktas secīgi, tas ir, vispirms tiek projektēta sistēma, un tikai pēc tam tiek strādāts pie dizaina un sistēmas izstrādes;
- ļoti precīzi iespējams noteikt izmaksas un darbietilpību, jo visi posmi seko viens otram.

Metodes trūkumi [13]:

- kļūda analīzes fāzē ūdenskrituma modelī patērē ļoti daudz resursu, turklāt tiek pazaudēts laiks, jo viss process ir jāsāk no sākuma un secīgi jāiet cauri visām fāzēm;
- ja pasūtītāja vēlmes nav skaidri zināmas, tad analīzes fāzi nav iespējams veikt pilnīgi, kas bieži rada ilgu projektu aizkavēšanos;
- problēmas iespējams identificēt tikai sistēmas testēšanas fāzē;
- ja tiek aizkavēta viena no darbības fāzēm, tiks aizkavētas arī visas pārējās.

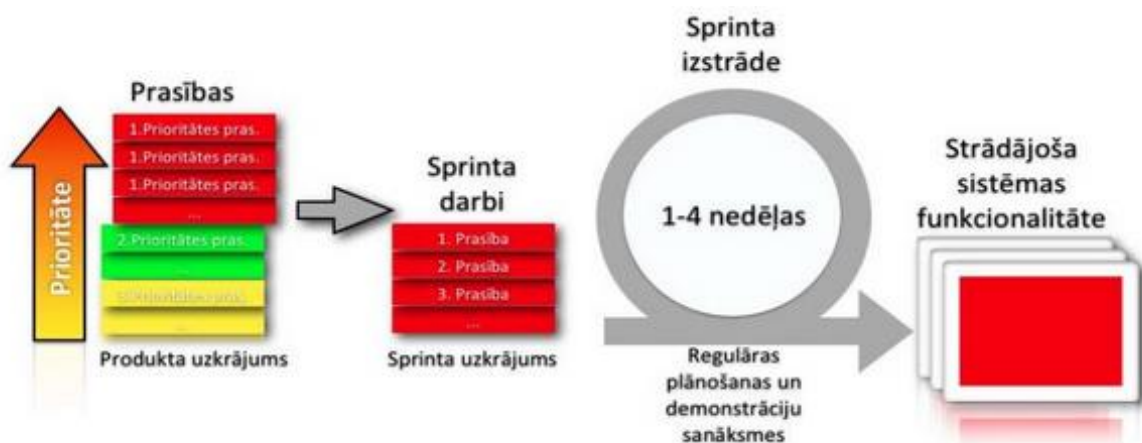
Ūdenskrituma modeli iespējams apskatīt attēlā 2.3



2.3. att. Ūdenskrituma modelis [12]

“Agile” jeb ātrās prototipēšanas metode. Šīs metodes pamatā ir ideja pēc iespējas ātrāk klientam nodot rezultātu, un, turpinot to uzlabot, iegūt gala versiju [14]. Metodes pamatā ir nedēļu līdz mēnesi gari izstrādes cikli, kuru laikā tiek izveidota sistēmas struktūra, kā arī veikta pati izstrāde un testēšana [14] “Agile” metodes “SCRUM” apakšmodeļa pieeja ir, cik vien iespējams, samazināt dokumentēšanu un izstrādes komandas, ātras produkcijas

piegādes, ka arī ikdienas mazās sapulces [14]. “SCRUM” modelī “Agile” “SCRUM” izstrādes ciklus iespējams aplūkot 2.4. attēlā



2.4. att. “Agile” “SCRUM” metodes cikls [15]

“Agile” metode ir ļoti atbilstoša, ja sākumā projekta prasības nav konkrētas, jo tajā ir iespējams vienkārši veikt labojumus un uzlabojumus. Cikli var atkārtoties neierobežoti daudz reižu, kamēr tiek sasniegts vēlamais rezultāts, un tad pēdējais cikls kļūst par gala rezultātu. “Agile” metodē darbus veic 4–9 darbinieki, kas ir savstarpēji ļoti vienoti un saprot viens otru. Lai ietaupītu laiku, izmantojot “Agile” metodi, tiek veikta pēc iespējas mazāka dokumentācija.

Metodes priekšrocības [13]:

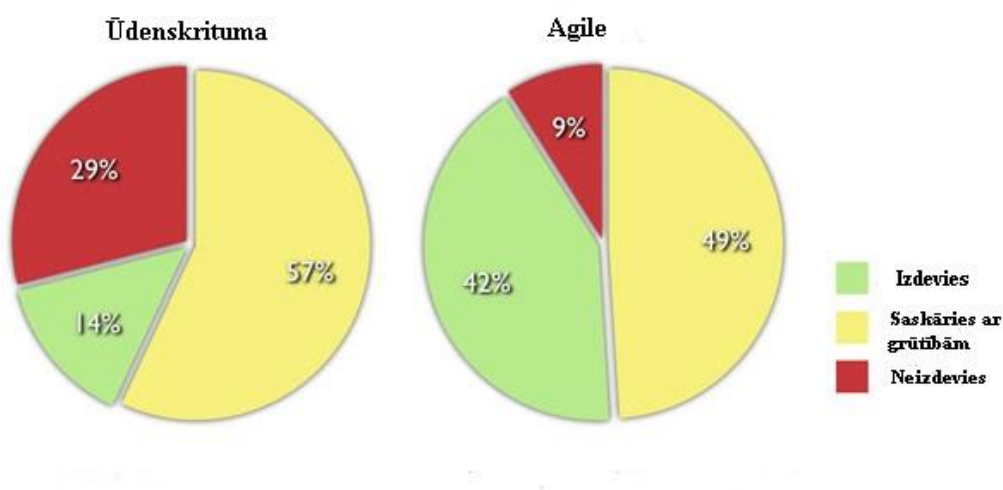
- Ļoti drīz iespējams redzēt kaut sākotnēji, bet gatavu rezultātu.
- Skatoties izveidoto versiju, iespējams saprast, vai izveidotā sistēma atbilst iecerētajam un vajadzībām.
- Klientam ir vieglāk atrast sev vēlamo dizainu, jo tas tiek mainīgs un pielāgots.
- Ja klienta prasības nav zināmas, tad, redzot izveidoto versiju, tam būs vieglāk tās definēt, labojot esošo versiju.

Metodes trūkumi [13]:

- Mazāka dokumentācija, jo jebkurš variants var tikt uzskatīts par pagaidu variantu.
- Ļoti grūti plānot projekta darbietilpību un izmaksas.
- Redzot lietotāja saskarni, lietotājs var kļūdīties, domājot, ka viss darbs jau ir pabeigts.

Galvenās metožu atšķirības ir to pieejā, iespējas izmaiņu veikšanā izstrādes laikā un dokumentācijā. Ūdenskrituma metode notiek secīgi, tas ir, viens notikums seko otram; ja nav pabeigts pirmais uzdevums, tad otro sākt nav iespējams. Turpretim “Agile” modelī var

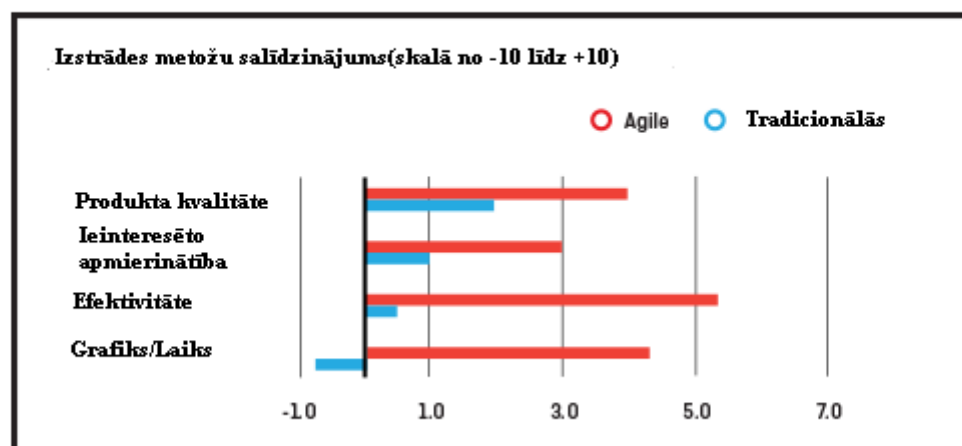
pēkšņi ieviest dažādus labojumus. Ūdenskrituma metodē ļoti svarīga ir dokumentēšana, kas notiek katrā stadijā un kam tiek veltīts daudz laika, jo pareiza un pilnīga dokumentēšana ir svarīga nākamā posma izstrādei. “Agile” modelī dokumentēts tiek daudz mazāk, un pārsvarā dokumentācija sastāv tikai no pēdējā prototipa apraksta. Tāpat “Agile” modelī galvenais uzsvars tiek likts uz ātra prototipu parādīšanu pasūtītājam, lai tas var to novērtēt un lūgt ieviest uzlabojumus, ja tādi ir nepieciešami. “Agile” metodē ir iespējams sistēmas izmaiņas veikt pēdējā brīdī, bet ūdenskrituma metodē šādas iespējas nav [16]. Parasti ūdenskrituma metodi vēlams izmantot, ja pasūtītāja darbiniekiem ir labi zināms, ko tie vēlas no sistēmas, un projekta cena ir fiksēta. “Agile” modeli ieteicams izmantot, ja pasūtītājam ir skaidrs, ka esošā sistēma to neapmierina, bet nav skaidru prasību, ko un kā vajadzētu mainīt. Tāpat “Agile” modeli ieteicams lietot, projekta sākumā tā cena nav fiksēta [13]. Salīdzinot ar pēc jebkuras citas metodes veiktajiem projektiem, tie projekti, kas izstrādāti pēc “Agile” metodes, trīs reizes biežāk sasniedz veiksmīgu rezultātu [17]. Salīdzinājumu par to, cik veiksmīgi tiek pabeigti ūdenskrituma un “Agile” metodes projekti, skatāms 2.5. attēlā.



2.5. att. Ūdenskrituma metodes projekta izdošanās salīdzinājums ar “Agile” metodi [17]

Salīdzinājuma attēlā redzams, ka “Agile” projekti ir izdevušies trīs reizes biežāk nekā projekti, kas īstenoti pēc “ūdenskrituma” metodes. Tāpat iespējams redzēt, ka projekti, kas izstrādāti pēc “Agile” metodes, ir cietuši neveiksmi retāk, nekā pēc “ūdenskrituma” metodes izstrādātie projekti. Līdz ar to no attēla iespējams secināt, ka projekti, kas izstrādāti pēc “Agile” izstrādes metodes, ir veiksmīgāki.

2.6. attēlā iespējams redzēt izstrādes metožu salīdzinājumu.



2.6. att. Izstrādes metožu salīdzinājums [18]

Attēlā redzams salīdzinājums starp “Agile” un tradicionālajām izstrādes metodēm. Kā redzams, izstrādājot projektu pēc “Agile” izstrādes metodes, projekta kvalitāte un ieinteresēto pušu (piemēram, izstrādātāju un pasūtītāju) apmierinātība ir augstāka, nekā izstrādājot projektu pēc tradicionālajām izstrādes metodēm. Vislielākā starpība novērojama efektivitātes un iekļaušanās grafikā/laika stabiņos, kur redzams, ka “Agile” metode ir piecas reizes efektīvāka nekā tradicionālās izstrādes metodes. Stabiņā “Grafiks/Laiks” redzams, ka tradicionālo izstrādes metožu skala ir negatīva, turpretim “Agile” izstrādes metodes skala ir pozitīva. Pēc šī attēla var secināt, ka “Agile” izstrādes metode ir labāka par tradicionālajām izstrādes metodēm.

Abus minētos modeļus var izmantot kopā, piemēram, izmantot “Agile” metodi kāda atsevišķa sistēmas bloka veidošanai, bet pamatā pieturēties pie “ūdenskrituma” metodes.

Autora komanda “Ķelmēnu maiznīcas” ražošanas uzskaites sistēmas izveidē izmantoja “Agile” izstrādes metodi. Šī metode bija piemērotākā, jo projekta sākumā sistēmas prasības bija ļoti skopas un no tām nevarēja veikt analīzi. Tāpat pasūtītājam nebija skaidra redzējuma par to, kādai vajadzētu būt sistēmai. Izstrādes gaitā šīs prasības pamazām kļuva skaidrākas, kas ļāva izstrādāt aizvien precīzākus un pasūtītājam tīkamākus prototipus. Bija ļoti ērti izmantot “Agile” modeli, jo komandā bija tikai pieci cilvēki, tāpēc bija ļoti viegli vienoties par visām izmaiņām un uzlabojumiem. Kā vienīgo problēmu varētu minēt to, ka, izstrādājot sistēmu pēc “Agile” izstrādes metodes, dažkārt rodas sajūta, ka projekts nekad nebeigsies, jo pasūtītājs vienmēr jau gatavā prototipā grib iekļaut jaunumus vai uzlabojumus.

3. SISTĒMAS IZSTRĀDES RĪKI

3.1. Rīku iespējas informācijas sistēmas izstrādē

Pirmo sistēmas izstrādes rīku 1982. gadā radīja Mičiganas programmatūras izstrādes uzņēmums “Nastec Corporation” [19]. “Nastec Corporation” izveidoja grafikas un teksta redaktoru “GraphText” [19].

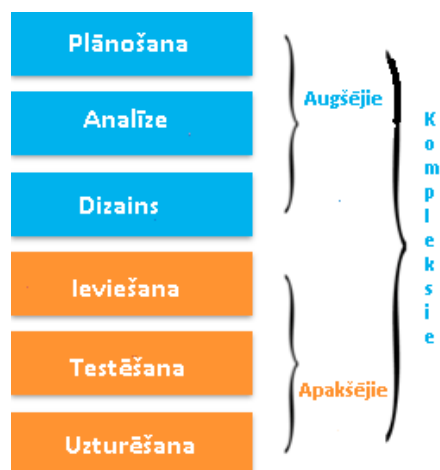
Mūsdienās projekta izstrāde nav iedomājama bez sistēmas izstrādes rīku izmantošanas. Šie rīki attīstās ar katru dienu, kļūstot aizvien parocīgāki un vienkāršāki, bet tajā pašā laikā — ļoti spēcīgi un noderīgi. Katrā projektu izstrādes stadijā iespējams izmantot dažādus izstrādes rīkus, bet ir arī tādi rīki, ko iespējams izmantot vairākos ciklos.

Ir ļoti grūti novērtēt izmaksas un ieguvumus no sistēmas izstrādes rīku izmantošanas, jo daži no avotiem sniedz datus par lielām izmaksām un maziem ieguvumiem, bet citi tieši pretēji — par lieliem ieguvumiem un kvalitāti [20].

Rīku galvenās priekšrocības [20]:

- Produktivitātes un kvalitātes pieaugums
- Atvieglo izmaiņu veikšanu dažādās projekta izstrādes stadijās.
- Ļauj pārstrukturēt slikti uzrakstītu kodu.
- Vienkāršo programmas uzturēšanu
- Īsteno sistēmas izstrādes standartus.
- Nodrošina datu krātuvi un references uz enciklopēdiju
- Vienkāršo sistēmas arhitektūras diagrammu zīmēšanu.
- Samazina aplikācijas izstrādes laiku.
- Automātiskā koda ģenerēšana

Projektu izstrādes rīkus var iedalīt šādās kategorijās (skatīt 3.1. attēlā):



3.1. att. Izstrādes rīku sadalījums

Augšējie sistēmas izstrādes rīki ir paredzēti plānošanai, analīzei un dizaina veidošanai, savukārt apakšējie sistēmas izstrādes rīki ir paredzēti ieviešanai, testēšanai un uzturēšanai. Apakšējie sistēmas izstrādes rīki ir paredzēti programmas ieviešanai, testēšanai un uzturēšanai. Kompleksie sistēmas izstrādes rīki ir piemēroti visu projekta dzīves ciklu īstenošanai, sākot no prasību noskaidrošanas līdz testēšanai un dokumentēšanai.

Sistēmas izstrādes rīkus ir iespējams grupēt, ja tiem ir vienāda funkcionalitāte, kā arī procesa darbības un līdzīgas savienojamības iespējas ar citiem rīkiem.

Autors piedāvā šādas rīku grupas.

Diagrammu rīki ir paredzēti, lai attēlotu sistēmas komponentes, nepieciešamos datus un kontroles plūsmu starp dažādās programmatūras komponentēm un sistēmas struktūru grafiskā veidā. UML (vienotā modelēšanas valoda) diagrammas ir sadalītas trīs kategorijās: struktūras un izmantošanas diagrammās. Kopā vienotajā modelēšanas valodā iespējams zīmēt 14 diagrammu [21]. Diagrammu zīmēšanai tiek izmantota arī BPML (Biznesa procesu modelēšanas valoda) [22]. Diagrammu izmantot tādus rīkus kā “Star UML” [23], “Visual-Paradigm” [24], “Microsoft Visio” [25] un citus.

Projektu vadības rīki ir paredzēti projektu plānošanai, izmaksu un darba ieguldījuma noteikšanai, kā arī projektu grafika un resursu plānošanai. Vadītājam ir stingri jāuzmana projekta izpilde, tostarp vissīkākie procesi, kas jāveic, lai pabeigtu projektu. Vadības rīki uzlabo savstarpējo komunikāciju, lai visa nepieciešamā informācija vienmēr būtu visiem pieejama. Projektu vadības rīkos iespējams pievienot visus projektam nepieciešamos attēlus vai failus, lai tie vienmēr būtu pieejami. Projektu vadībai iespējams lietot tādus rīkus kā “Basecamp” [26], “Jira” [27], “Asana” [28], “Microsoft Project” [29] un citus. “Jira” [27] rīks ir informācijas tehnoloģijas projektu vadības rīks, pārējie minētie rīki ir klasiskie projektu vadības rīki.

Konfigurācijas rīki palīdz izdalīt sistēmas versijas. Šie rīki noder, lai varētu vieglāk pārvaldīt esošo un iepriekšējo versiju un kļūdu gadījumā varētu atgriezties atpakaļ. Tātad šie rīki ir ļoti noderīgi, ja pie vienā projektā darbojas vairāki darbinieki, jo šādi visus izdarītos darbus iespējams savienot vienā un veikt nākamās labojumus jau uzlabotajai versijai. Var izmantot tādus konfigurācijas rīkus kā “Git” [30], “Bitbucket” [31] un citi.

Izmaiņu kontroles rīki darbojas kā konfigurācijas rīku palīgrīki. Tie saglabā visas izmaiņas, kas veiktas sistēmai, kad tās sākotnējā versija ir uzlabota vai kad sistēma pirmo reizi ir izlaista. Tāpat tie palīdz organizācijas izmaiņu īstenošanas gadījumā.

Programmēšanas rīki ietver programmēšanas vides, piemēram, IDE (Integrēto izstrādes vidi), iebūvēto bibliotēku un simulēšanas rīkus. Šie rīki sniedz vispusīgu atbalstu

programmas izveidē un ietver simulēšanu un testēšanu. Kā piemēru var minēt “Cscope” [32], ko izmanto, lai atrastu sev nepieciešamo koda daļu C valodā, izmantojot “Eclipse” [33].

Koda ģenerēšanas rīki palīdz ietaupīt laiku un ģenerēt kodu no izveidotajām UML (vienotās modelēšanas valodas) klašu un secību diagrammām, kā arī ERD diagrammas. Ģenerējot kodu no, piemēram, klašu diagrammas, tiek ģenerētas visas klases, tajās esošie dati un to datu tipi.

Prototipēšanas rīki tiek izmantoti, lai izveidotu lietotāju saskarnes modeli, kurā iespējams redzēt un izmēģināt, kāda būs sistēmā pieejamo grafisko objektu funkcionalitāte. Prototipēšanas rīki ir ļoti būtiski, lai pasūtītājs pārbaudītu un saprastu, vai viņa vēlnes ir saprastas pareizi un viss izskatās tā, kā tam vajadzētu izskatīties. Ir arī prototipēšanas rīki, kas ļauj simulēt prototipus. Kā prototipēšanas rīku piemēru var minēt “Balsamiq mockups” [34].

Vienības testēšanu var veikt pēc baltās kastes strukturālās testēšanas metodes vai melnās kastes strukturālās testēšanas metodes [35]. Kā **vienību testēšanas rīku** piemēru var minēt “Ruby Rspec” [36] kā arī “crosschecknet” [35] piedāvātās iespējas.

Integrācijas testēšanas laikā tiek kombinētas un testētas aparatūras vai programmatūras sastāvdaļas, tādejādi pārliecinoties par to savstarpējo sadarbību atbilstoši izvirzītajām prasībām. Integrācijas testēšana notiek tik ilgi, līdz sistēma ir pilnībā integrēta. Kā **integrācijas testēšanas rīku** piemēru var minēt “Capybara” [37].

Manuālā testēšana ir testēšana, ko veic testētāji, pārbaudot sistēmas darbību pie dažādām ievadītām vērtībām, tostarp pārbaudot, vai ievietotas visas nepieciešamās pārbaudes, lai pārliecinātos, vai kļūdainu datu ievadīšanas gadījumā programma nostrādā tā, kā tas ir gaidāms un nepieciešams [38]. Testētājiem ir jāziņo par visām testu laikā atrastajām kļūdām, kā arī jāmeklē visas šīs kļūdas atkārtot. Kā **manuālās testēšanas rīka** piemēru var minēt “utest” [39] piedāvātās iespējas.

Uzturēšanas un kļūdu meklēšanas rīki tiek izmantoti, lai automātiski meklētu un ziņotu par iespējamajām kļūdām, automātiski ģenerētu kļūdu biļetes un veiktu cēloņu analīzi. Kā uzturēšanas un kļūdu meklēšanas rīku piemēru var minēt “Bugzilla” [40].

3.2. Biežāk izmantotie sistēmas izstrādes rīki

Par vienu no labākajiem rīkiem ar testēšanas iespējām, pēc avotā [41] sastādītā topa ir atzīts “Atlassian” rīks “JIRA” [27], kuram ir divi spraudņi, kas atbalsta testēšanu. Turklāt “JIRA” tiek lietots arī kā informācijas tehnoloģijas projektu vadības rīks, jo tajā iespējams zīmēt procesa diagrammas, kā arī veikt plānošanu. “JIRA” rīkam iespējams pievienot vairāk

kā tūkstoš spraudņu, kas manāmi uzlabo rīka lietojamību [27]. Tāpat “JIRA” ir iespējams lietot kopā ar “Bitbucket” [31] vai “GitHub” [30], kas uztur programmas kodu. Par vienu no spēcīgākajiem rīkiem testēšanā atzīts arī “TestRail” [42], ar kura palīdzību iespējams izveidot testa plānus un novērot to virzību. “TestRail” ir iespējams savienot ar vairāk kā 35 rīkiem.

Populāri klasiskie projektu vadības rīki ir “Microsoft” izstrādātais “Microsoft Project” [29], kā arī “Basecamp” [26], ko izmanto tādi uzņēmumi kā “Adidas”, “Twitter”, “Nike”, “DHL” u. c. “Basecamp” ir izmantojuši vairāk nekā 15 miljoni cilvēku [26]. “Basecamp” tiek plaši izmantots un ir zināms gan rīka ērtās un ātrās lietošanas, gan lietotājiem pievilcīgās grafiskās saskarnes dēļ. Šis rīks ir noderīgs uzņēmumu vadītājiem, kuri var redzēt un pārskatīt visus projektus un darbus, kā arī to, kuri darbinieki ir atbildīgi par šiem darbiem. Uzņēmuma vadītājs katru rītu saņem ziņu no “Basecamp” par to, kas ir padarīts iepriekšējās dienas laikā, kādas izmaiņas veiktas un kurš tās ir veicis. Projektu vadītāji var piešķirt darbus, norādīt nepieciešamos projekta vai darba beigu termiņus, kā arī redzēt progresu. Tāpat projektu vadītāji var organizēt dažādas sapulces un tikšanās, kā arī sniegt atsauksmes par padarītajiem darbiem. Projektos iesaistītās personas var viegli redzēt sev uzdotos uzdevumus un apskatīt pabeigtos uzdevumus. “Basecamp” iespējams glabāt arī visus nepieciešamos attēlus, logo, kā arī prezentācijas un citus failus, lai tie vienmēr būtu visiem viegli pieejami [26]. Kā “Basecamp” galvenos plusus var minēt to, ka, izmantojot viedtālruni vai datoru, šo rīku iespējams lietot jebkurā pasaules vietā, kur ir pieejams internets, un ka tam nav nepieciešama papildu uzstādīšana, bet šis aspekts var pārtapt par mīnusu, ja kādam no projektā iesaistītajiem nav pieejams internets [26]. Kā galvenos “Microsoft Project” plusus var minēt iespēju izveidot dažādas atskaites, kā arī no esošajiem datiem ir iespējams izgūt daudz diagrammu, lai labāk saprastu projekta virzību un darbinieku noslodzi [29]. Šajā rīkā ir iespējams uzstādīt uzdevuma prioritāti, un tādējādi darbiniekiem, kas strādā pie attiecīgā projekta, ir ļoti skaidri redzams, kas ir primāri veicamie darbi. Tāpat “Microsoft Project” iespējama “Microsoft Word” failu, e-pastu un dažādu citu failu veidu ievietošana projektā [29]. “Microsoft Project” iespējams izmantot arī bez interneta pieslēguma. Šī rīka plusi noteikti ir tā daudzās iespējas, bet kā lielāko mīnusu varētu minēt lielās izmaksas. Abi minētie rīki ļoti noder projekta izmaksu plānošanā, jo, ievadot visu nepieciešamo informāciju, tie automātiski aprēķina izmaksas. “Basecamp” ir piemērotāks mazu uzņēmumu ar mazu darbinieku skaitu vajadzībām, savukārt “Microsoft Project” ir vairāk piemērots lieliem projektiem [43]. Abiem rīkiem ir pieejamas bezmaksas izmēģinājuma versijas.

Saskaņā ar [44] informāciju vieni no labākajiem vienotās modelēšanas valodas diagrammu zīmēšanas rīkiem ir “Visual Paradigm for UML” [24] un “Argo UML” [45]. “Visual Paradigm for UML” rīks ir ļoti daudzpusīgs un līdz ar to piemērots lietotājiem ar dažādām vajadzībām. To var izmantot gan programmatūras izstrādātāji, gan sistēmas analītiķi, gan biznesa analītiķi, gan visi, kas ir ieinteresēti veidot liela mēroga programmatūras sistēmas, izmantojot objektorientēto pieeju [43]. Izmantojot šo rīku, iespējams uzzīmēt vienotās modelēšanas valodas diagrammas, tāpat iespējama koda ģenerēšana, dokumentēšana, to ir iespējams izmantot grafiskās saskarnes prototipēšanai, kā arī rīks izstrādā prasību specifikāciju. Šim rīkam ir pieejama 30 dienu izmēģinājuma versija, kas pieejama pat vislabāk nokomplektētajai rīka pakai [23]. Šis rīks ir noderīgs un atvieglo darbu visā projekta izstrādes ciklā. “Argo UML” ir atvērtā koda bezmaksas rīks. Rīkā iespējams uzzīmēt deviņas vienotās modelēšanas valodas diagrammas. Šajā rīkā iespējama koda ģenerēšana, kā arī diagrammu eksportēšana vairākos formātos. Tāpat iespējama koda reversā inženierija (tas ir, no programmas koda tiek ģenerēta diagramma) [44].

Viens no populārākajiem relāciju datu bāzes datu arhitektūras un modelēšanas rīkiem, pēc [46] ir “Embarcadero” izveidotais rīks “ER/Studio” [47]. Šis rīks nodrošina ātru un vienkāršu datu sasaistes veidu datu vadības speciālistiem, lai izveidotu un uzturētu uzņēmuma datu modeļus un metadatu glabātuves. Iebūvētās iespējas automatizēt ikdienas modeļu uzdevumus ļauj lietotājam analizēt un optimizēt datu bāzes [47].

Saskaņā ar [48] informāciju viens no atzītākajiem prototipēšanas rīkiem ir “Balsamiq Mockups” [34]. Ar šī rīka palīdzību ir iespējams uzzīmēt dažādas lietotāja saskarnes, lai sistēmas izstrādātāji, neieguldot lielu darbu, varētu parādīt pasūtītājam, kā varētu izskatīties lietotāja saskarne. Izmantojot “Balsamiq Mockups” iespējams savienot dažādus logus, lai labāk varētu saprast katra loga un to grafisko elementu (pogu, tabulu, attēlu) funkcionalitāti. Piemēram, nospiežot pogu “Uz sākumu”, lietotājam tiek parādīta sākumlapa. Izveidotos prototipus iespējams eksportēt .pdf vai .png formātā, kā arī, izmantojot citus rīkus, — citos formātos [34]. Ir iespējams ģenerēt prototipa koda dažādās programmēšanas valodās, izmantojot rīkus, ko iespējams savienot ar “Balsamiq Mockups” [34]. Savukārt, kā vienu no atzītākajiem prototipēšanas rīkiem atbilstoši avotā norādītajai [48] informācijai varētu minēt arī “Axure” [49], kurā iespējams veidot lietotāja saskarnes un izveidot funkcionējošas pogas. Lai izveidoto prototipu varētu skatīt sev izvēlētajā interneta pārlūkā (“Internet Explorer”, “Google Chrome”, “Mozilla Firefox”, “Safari”), šajā rīkā iespējams arī ģenerēt .html failu, tāpat iespējams izveidot dokumentāciju. Lai veiksmīgu apgūtu šo rīku, ir pieejamas bezmaksas tiešsaistes mācības [49]. Atšķirībā no “Balsamiq Mockups”, “Axure” ļauj imitēt prototipa darbību.

Saskaņā ar internetā pieejamo informāciju populārākās programmēšanas valodas ir C un Java [50]. Saskaņā ar [51] viens no populārākajiem C programmēšanas valodas izstrādes rīkiem ir “CodeBlock” [52]. Šim rīkam ir iebūvētas lietotāja saskarnes un atklūdošanas, kā arī kompilācijas iespējas. “CodeBlock” ir atvērtā koda bezmaksas rīks, kurā iespējams rakstīt programmas kodu C, C++ un Fortran valodā. Lai uzlabotu rīka funkcionalitāti, tam iespējams pievienot dažādus spraudņus [52]. Atbilstoši [53] viens no populārākajiem Java programmēšanas valodas rīkiem ir “Eclipse” [33]. “Eclipse” ir bezmaksas rīks, kam iespējams pievienot dažādus spraudņus, lai šo rīku padarītu vēl universālāku. Šis rīkam ir arī savs veikals, kur iespējams redzēt visus pieejamos spraudņus, cik reizes tie ir lejupielādēti, kā arī lietotāju atsauksmes un komentārus. Šajā veikalā pieejami vairāk kā 1700 rīki, ko iespējams pievienot “Eclipse” rīkam [33].

Atbilstoši [54] viens no labākajiem bezmaksas rīkiem programmas koda konfigurēšanai un uzturēšanai ir “GitHub” [30]. Šis rīks uzlabo koda izmaiņu veikšanu, jo pirms katras jaunās koda izmaiņas kādam citam darbiniekam vai vadītājam šis kods ir jāpārbauda. Tāpat ļoti viegli redzamas kodā veiktās izmaiņas, jo visas mainītās vai dzēstās rindiņas tiek iekrāsotas sarkanā krāsā, bet labotās vai klāt pierakstītās programmas koda rindas — zaļā krāsā. “GitHub” ir lielākais programmas kodu uzturētājs [30], un tajā var glabāt vairāk nekā 22,6 miljonus projektu [30]. Ir iespējams izmantot bezmaksas projektus jeb projektus, kuru kodu iespējams aplūkot visiem, vai kuros iespējams izveidot privātu projektu, kam pieeja atļauta tikai projektā iesaistītajām personām. Rīkā iespējams komentēt jebkuru koda daļu — ja projekta vadītājs vai darbinieks, kas atbild par koda izmaiņu veikšanu, nepiekrīt programmas koda izmaiņu veicējam, kļūdainās vietas ir iespējams komentēt, kā arī liegt šī jaunā koda iekļaušanu pamatkodā [54].

3.3. Projektā izmantotie sistēmas izstrādes rīki

Ķelmēnu aplikācija ir “Ruby On Rails” aplikācija. Tā ir rakstīta “Ruby” [55] programmēšanas valodā.

Programmas koda uzturēšanai, izmaiņu kontrolēšanai un vieglākai programmas koda pārskatīšanai izmantotas “Github” piedāvātās iespējas. Rīks ir bezmaksas, ja savu koda glabātuvi veido kā publisku, tātad visiem pārējiem lietotājiem ir pieeja jebkurai projekta mapei un failam. Tāpat pieejams 45 dienu izmēģinājums privātajai versijai, kur projekts pieejams tikai tā izstrādātājiem [30]. Šādas privātās versijas izmaksas sākas no septiņiem ASV dolāriem mēnesī [30]. Kā šī rīka alternatīvu varu minēt “Atlassian” veidoto rīku “Bitbucket” [31] Šis rīks ir ļoti līdzīgs “Github”. Tā lietošana projektu grupām līdz pieciem lietotājiem ir bezmaksas, bet, sākot no sešiem dalībniekiem, ir jāmaksā desmit ASV dolāru mēnesī. Cena palielinās, pārsniedzot katru lietotāju skaita līmeni [31]. Izmantojām tieši “GitHub” rīku, jo lai izmantotu dažus no pārējiem izmantotajiem rīkiem, bija nepieciešams izmantot tieši šo rīku.

“Ķelmēnu maiznīcas” projektā izmantojām trīs koda zarus (angļu valodā — “branch”), proti, “master”, “staging” un “develop”. Trīs zarus izmantojām, lai “master” vienmēr būtu tikai jau gatavā izstrādes vide, savukārt “develop” vidē tika veiktas visas izmaiņas, kas pēc izmaiņu veikšanas tās tika ievietotas “staging” zarā, lai tām varētu piekļūt testētāji, kas pārbauda, vai ir iekļauts viss nepieciešamais, bet tajā pašā laikā varētu veikt izmaiņas “develop” zarā. Kad “staging” zara kods bija pārbaudīts un problēmas netika atrastas, tas tikai pievienots “master” un “develop” zaram, lai visos zaros kods būtu vienāds. Šis rīks ir ļoti noderīgs projektu izstrādē, jo, šādi darbojoties, vienmēr var pārliicināties un pārbaudīt, vai no jauna uzrakstītajā, papildinātajā sistēmas kodā nav pieļauta kāda kļūda vai izmantota nepareiza metode. Ja šādas kļūdas ir, tad visi, kam ir pieeja šim projektam, var pievienot savus komentārus par šo kļūdu, un tikai pēc kļūdu izlabošanas šo kodu ievieto attiecīgajā zarā. Pie šī rīka sākumā nav viegli pierast, jo šī pieeja atšķiras no pirms tam izmantotajām pieejām, bet, apgūstot un saprotot, kā to izmantot, var sākt izjust rīka priekšrocības.

Sistēmas vienību testēšanai tika izmantota “Ruby Rspec” [36]. Lai lietotu šo rīku, ir nepieciešams izmantot “Rspec” bibliotēku un projektam pievienot “Rspec” mapi, kurā glabājas visi projektam nepieciešamie testi. Šajā mapē tiek izveidoti faili ar .rb (“Ruby”) paplašinājumu, kuros tiek ierakstīti testi, piemēram, kādu reakciju sagaidīt no testējamās klases vai funkcijas, vai jebkuras citas vienības, ko vēlas testēt. Piemēram, varu minēt, ka projektā bija izveidots tests, kas pārbaudīja, vai tiek samazināti glabātvē esošie materiāli,

ja izveidots jauns produkts. Šīs bibliotēkas un mapes pievienošana nebija sarežģīta, bet paša principa izprašana prasīja laiku, jo tas bija kas jauns un nekad iepriekš nedarīts. Tāpat bija jāpierod pie “Ruby” valodas sintakses.

Integrācijas testi veikti ar “Capybara” izmantojot “Selenium” [37]. Lai izmantotu šo rīku, ir nepieciešams pie bibliotēkām pievienot “Capybara webkit” un izveidot failus, kuros testi tiks aprakstīti. Kā piemēru iespējams minēt reģistrēšanās testu. Pēc lietojumprogrammas palaišanas pie lietotājevārda tiek ievietots esošs lietotājs ar tiesībām pievienoties sistēmai, kā arī ievadīta pareiza parole. Turpmāk, pēc pogas “ieiet” nospiešanas, būtu jāatveras sistēmas sākuma lapai. Ja viss notiek pēc minētā plāna, tad, palaižot testu, tas parāda, ka viss ir noticis pareizi, un pēc doto datu ievadīšanas ir atvērusies sākuma lapa. Arī šāda veida testēšanas izmantošana autoram bija jaunums, tāpēc bija ļoti interesanti ar to darboties, veidot dažādus testa piemērus un skatīties to rezultātus. Turklāt darboties ar šo rīku bija salīdzinoši vienkārši.

Automātiskai koda stila pārskatīšanai projekta ietvaros izmantojām rīku “Hound” [56]. Šis rīks palīdz uzlabot koda stilu, lai tas vienmēr būtu viegli pārredzams. Lai izmantotu šo rīku, tas ir jāsaista ar “Github” [30] projekta mapi. Katru reizi, kad tiek veikts kāds koda labojums, pirms tā iekļaušanas zarā “Hound” pārbauda programmas kodu un vai tā stilā nav pieļautas kļūdas. Ja kļūdas ir atrastas, tad tiek nosūtīts kļūdas ziņojums. Šo rīku ir iespējams lietot bez maksas, ja tas ir saistīts ar atvērtā koda “Github” [30] projektu. Par privātā tipa projekta sasaisti ar rīku un tā izmantošanu nepieciešams maksāt 12 ASV dolārus mēnesī [56]. “Hound” par stila pamatu ņem pieņemtos nozares standartus un savu vairāku gadu pieredzi projektu izstrādē. Ja nepieciešams, šos stila standartus ir iespējams pielāgot sava projekta vajadzībām [56]. “Hound” rīka izmantošana ir ļoti vienkārša, jo tā ir tikai jāsaista ar “Github” projektu. Autors šī rīka izmantošanu uzskata par nepieciešamu un ļoti noderīgu, jo viņa sliktajam programmēšanas stilam šī stila kontrole lieti noder.

“Ķelmēnu maiznīcas” lietojumprogramma atrodas “herokuapp” [57] serverī. Šo rīku, ļoti līdzīgi kā “Hound”, ir nepieciešams saistīt ar “Github” projektu. Pēc “Github” un “herokuapp” sasaistes ir iespējams programmas kodu attēlot aplikācijas veidā, turot to uz “herokuapp” servera. Rīks piedāvā dažādas iespējas — ir iespējams aplikāciju automātiski atjaunot pēc katras programmas koda izmaiņu veikšanas vai, kad lietotājs pats to vēlas, to atjaunot manuāli. Rīks “herokuapp” atbalsta “Ruby”, “Python”, “Java”, “PHP” un “Node.js” [57]. Ja datubāzē ir ierobežots rindu skaits, tad šo rīku ir iespējams izmantot bez maksas, tomēr jāreķinās, ka serveru profilakses dēļ, četras stundas mēnesī rīks nav pieejams, turklāt aplikācija ir samērā lēna. Papildus iespējams iegādāties lielāku aplikācijas apmeklētāju skaitu, ātrāku serveri, iespēju izmantot “herokuapp” datubāzi ar neierobežotu rindu

daudzumu, kā arī pievienot aplikācijai dažādus spraudņus vai tehnisko palīdzību [57]. “herokuapp” šobrīd ir viena no deviņām populārākajām mākoņu izstrādes platformām [58]. Saskaņā ar [58] rīkam ir pieejamas tādas alternatīvas kā “Google App Engine” [59] un “Microsoft Windows Azure” [60]. Šī rīka izmantošana bija ļoti vienkārša un noderīga, jo bija iespējams redzēt, kā izskatās aplikācija, to pārbaudīt un paveikto darbu atrādīt pasūtītājam.

Pirms katras programmas koda izmaiņu veikšanas, uzlabotā lietojumprogramma tika apskatīta ar “Teatro” [61] rīka palīdzību. Ar šī rīka palīdzību iespējams redzēt tieši tā zara izmaiņas, kurā tās veiktas. Šis rīks izveido interneta lapu, kurā ir attēlots programmas koda saturs. “Teatro” rīkā ir iespējams izmantot “mysql”, “sqlite” un “postgresql” datubāzi. Šis rīks atbalsta “Ruby on Rails”, “PHP” un “Python/DJANGO” platformas [61]. Ja tiek izmantots publiski pieejams “Github” projekts, tad šo rīku ir iespējams lietot bez maksas. Šo rīku izmantojām, jo pēc pieprasījuma nosūtīšanas “Teatro” nosūta saiti, kurā ir iespējams redzēt un izmēģināt lietojumprogrammu, kā arī, lai pārējie projekta dalībnieki varētu to notestēt un pārbaudīt, vai lietojumprogrammā ir viss nepieciešamais un tā darbojas, kā iecerēts. Šī rīka izmantošana bija vienkārša, un tā noderēja, lai visi projekta dalībnieki varētu novērtēt, vai lietojumprogrammas uzlabojumi ir veikti pareizi.

Pirms katra programmas koda iekļaušanas zarā kods tiek testēts, izmantojot “Travis CI” [62] rīku. Šis rīks automātiski palaiž visus projektā esošos testus — gan vecos, gan jaunus, tikko pievienotos —, lai varētu pārliecināties, ka jaunās izmaiņas nav izmainījušas kādu citu programmas koda daļu. Šo rīku ir nepieciešams sasaistīt ar “Github” [30] projektu, un turpmāk, katru reizi veicot izmaiņas, “Travis CI” saņem ziņu, ka programmas kods ir jānotestē. Ja tests ir veiksmīgs, tad pie šī izmaiņu pieprasījuma tiek pievienots ieraksts zaļā krāsā, apliecinot, kods ir izturējis visus testus un ka šos uzlabojumus ir iespējams iekļaut programmas kodā. Ja testā atrasta kļūda, tad tiek pievienots ieraksts sarkanā krāsā, kas neļauj šo uzlabojumu pievienot pamatkodam. Šo rīku ir iespējams izmantot bez maksas, ja tiek izmantot “Github” publiskais projekts. Par rīka maksas versijas izmantošanu jāmaksā, sākot no 129 ASV dolāriem mēnesī [62]. Kā vienu no šī rīka alternatīvām var minēt “Circle CI” [64], kas piedāvā ļoti līdzīgas iespējas. Šo rīku var lietot bez maksas, ja to izmanto tikai vienam projektam, bet diviem un vairāk projektiem maksa sākas no 50 ASV dolāriem. [63] Šī rīka izmantošana atvieglo izstrādes darbu un laiku, jo nav nepieciešams katru no testiem un to veidiem palaist manuāli, bet tie tiek veikti automātiski pie katras izmaiņu veikšanas. Tāpat rīks noder, jo tam nav iespēja kļūdīties un aizmirst palaist kādu testu, kas var būt ļoti nozīmīgi tālākai izstrādei, jo visām kļūdām ir jābūt izlabotām, pirms tās tiek iekļautas pamatkodā.

Lai visiem būtu vienādas izstrādes vides, tika izmantots “Vagrant” [64]. Šis rīks ir ļoti nepieciešams, jo katrs izstrādātājs var strādāt, izmantojot citu operētājsistēmu (piemēram, “Linux”, “Mac Os X”, “Windows”), tas ļauj visiem izstrādātājiem strādāt vienā izstrādes vidē. Lai izmantotu šo rīku, ir jāizveido fails, kurā jāieraksta visa nepieciešamā informācija par virtuālo mašīnu, kas tiks lietota. Lai failam varētu piekļūt visi izstrādātāji, tas ir jā saglabā projekta mapē, un, izmantojot šo failu, “Vagrant” izveido virtuālo mašīnu atbilstoši failā minētajām prasībām. Šo rīku iespējams izmantot bez maksas. “Demola Latvia” projektā rīks bija ļoti nepieciešams, jo komandas dalībnieki izmantoja atšķirīgas operētājsistēmas. Šī rīka izmantošana bija jaunums, jo darba autors nebija piedalījies nevienā projektā, kur pie viena darba strādā vismaz divi programmētāji. Šī rīka apguve neaizņēma daudz laika, bet sākumā bija jāizprot pati rīka ideja un tas, kā tā darbojas, kā arī rīka priekšrocības.

Lai varētu kontrolēt projekta gaitu — paveiktos un vēl atlikušos uzdevumus —, izmantojām “Atlassian” rīku “Asana” [28]. “Asana” rīks ir bezmaksas rīks projektiem, kuros darbojas ne vairāk kā pieci dalībnieki. Sešiem un vairāk dalībniekiem cena sākas no 21 ASV dolāra mēnesī un pieaug atkarībā no nepieciešamā darbinieku skaita. [28] “Asana” rīks ļauj izveidot dažādus projektus, veidot uzdevumus projektos un piešķirt darbiniekam uzdevumus bez projekta.. Katru uzdevumu iespējams piešķirt nepieciešamajam darbiniekam, kas to redzēs savu uzdevumu lapā. Katram uzdevumam iespējams pievienot arī sekotājus, kas savā “Asana” e-pastā saņem informāciju par veiktajām izmaiņām. Ir iespējams uzdevumiem piešķirt datumu, kurā darbam jābūt pabeigtam. Ir iespējams apskatīt kalendāru, kurā atzīmēti visi uzdevumi, kuriem piešķirts pabeigšanas datums. Tāpat ir iespējams katram uzdevumam izveidot apakšuzdevumus. Visi uzdevumi, kas atzīmēti kā pabeigti, ir pieejami atsevišķā sadaļā. Katram projektam iespējams redzēt grafiku ar pabeigto un vēl veicamo uzdevumu skaitu. Lai pierastu pie šī rīka, nepieciešams laiks. Lai arī sākotnēji šķita, ka rīks ir ļoti vienkāršs un tajā nevar veikt zināmas darbības, ieguvuši vairāk informācijas par rīku, sapratām, ka šī rīka iespējas ir lielākas, nekā šķita sākumā. Rīks mūsu projektam bija ļoti noderīgs, jo tur varēja apspriest visas nepieciešamās nianšes, kā arī pievienot dažādus failus un attēlus, lai vieglāk paskaidrotu savu viedokli. Salīdzinoši lielu jautājumu risināšanai rīku izmantot bija daudz lietderīgāk nekā jebkuru čata saziņas rīku, jo šajā rīkā var komentēt katru atsevišķo jautājumu, un tādējādi visa informācija kļūst daudz pārskatāmāka, jo viss nav jāmeklē vienā garā tekstā. Tāpat šo rīku bija ļoti ērti lietot, jo tam ir pieejama bezmaksas Web versija, kā arī “Android” un “iOS” lietojumprogramma, kas atvieglo saziņu, jo piekļūt viedtālruna lietojumprogrammai un to izmantot ir ļoti vienkārši [28].

Lai noteiktu patērēto laiku, izmantojām “Toggl” [65] rīka piedāvātās iespējas. Šis rīks ir bezmaksas, ja to lieto ne vairāk kā pieci dalībnieki. Ja dalībnieki ir seši un vairāk, tad par katru dalībnieku jāmaksā pieci ASV dolāri mēnesī. Bezmaksas versijā pieejama laika uzskaitīšana, neierobežota projektu izveidošana, kā arī atskaites ziņojumi par to, cik ilgi katrs dalībnieks ir darbojies pie katra uzdevuma un kurā projektā. Maksas versijā ir visas bezmaksas versijas iespējas, bet papildus ir iespējams pievienot katra dalībnieka darbam tā stundas atalgojumu, tādējādi iegūstot dalībnieku samaksas aprēķinu, ko pēc tam iesniedzot pasūtītājam, ir iespējams pierādīt, cik daudz darba ir ieguldīts šajā projektā. Grafikus iespējams eksportēt .csv, .pdf un .xls(x) formātā [65]. Šo rīku ir ļoti viegli lietot, jo tam nav nepieciešama nekāda īpaša apmācība. Dalībniekam ir jāieiet mājaslapā vai jāatver lietojumprogramma savā viedtālrunī un jānospiež viena poga, lai tiktu uzskatīts laiks. Rīks ir viegli pieejams, jo tam iespējams piekļūt gan interneta mājaslapā, gan izmantojot viedtālruņa lietojumprogrammu. Rīku iespējams lietot arī bez interneta pieslēguma, jo, sākot uzskaitīt laiku, tas tiek uzskaitīts arī pēc mājaslapas vai lietojumprogrammas pamešanas, un laiks tiek apstādināts, tikai nospiežot pogu “stop”. Ļoti noderīga iespēja ir laika precizēšana vai pievienošana, ja aizmirsies laicīgi nospiezt laika palaišanu vai — tieši pretēji — aizmirsies nospiezt “stop” pogu, kā arī gadījumos, kuros kādā dienā aizmirsies rīku izmantot. Šo rīku projektā izmantojām, jo, izmantojot pārlūka “Google Chrome” spraudni, to ir iespējams lietot “Asana” rīkā, turklāt, sākot automātiski uzskaitīt laiku, tam tiek piešķirts konkrētā “Asana” uzdevuma nosaukums. Šī spraudņa trūkums ir projekta neatpazīšana, kā arī “Toggl” rīkā to ir nepieciešams ievadīt manuāli, lai gan nosaukums tiek ņemts no “Asana” uzdevuma, kas iekļauts kādā projektā.

Projektu izstrādājām pēc “Agile” “Scrum” metodes. Šī metode bija jaunums, jo iepriekš autors bija projektus izstrādājis tikai pēc ūdenskrituma metodes, tāpēc nācās iepazīties ar šīs metodes pamatprincipiem. Šo metodi izvēlējāmies, jo sākotnēji pasūtītājam nebija precīzu prasību; tās radās, tikai projektam virzoties uz priekšu, savukārt, redzot jau esošos risinājumus, pasūtītājs spēja noteikt, ko un kā būtu nepieciešams uzlabot. Sākumā sarunās ar pasūtītāju uzzinājām vispārējas prasības — kas tieši ir nepieciešams, kādas ir problēmas. Apmeklējot “Ķelmēnu maiznīcu”, prasības kļuva vēl skaidrākas. Pēc pirmā apmeklējuma tapa pirmais lietotāja saskarnes prototips, ko lietos maiznīcas darbinieki. Prototipi darbiniekus apmierināja, taču viņi deva norādījumus, ko vajadzētu uzlabot. Strādājām aptuveni trīs nedēļu ilgos posmos, kuru beigās atrādījām paveikto un ieguvām informāciju par veicamajiem uzlabojumiem. Iepazīšanās un metodes saprašana lielas problēmas neradīja un bija vienkārša, jo bijām ļoti saliedēta komanda, tāpēc visas sarunas un vienošanās neradīja lielas problēmas. Visu jautājumu risināšanā centāmies iesaistīt visus

komandas dalībniekus, lai iegūtu vismaz virspusējas idejas, kā veikt uzlabojumus. Pēc darbošanās saskaņā ar “Agile” metodi darba autors ir sapratis, ka šī metode ir ļoti noderīga, un metodi noteikti izmantos nākotnes projektos, kuros tā būs nepieciešama. Vienīgās problēmas sagādāja pasūtītāja neizpratne par darba gaitu, jo pēc pēdējā prototipa parādīšanas pasūtītājam tas domāja, ka projekts ir pabeigts vai ka tā pabeigšanai ir nepieciešams neilgs laiks, kaut gan sistēmas funkcionalitātes papildināšanai un pabeigšanai bija jāvelta vēl daudz laika. Jāpiebilst, ka šī ir viena no raksturīgākajām problēmām projektos, kur tiek izmantota “Agile” izstrādes metode. Salīdzinot ar “ūdenskrituma” metodi, “Agile” metodes izmantošana šajā projektā bija daudz efektīvāka, jo projekta sākumā “Ķelmēnu maiznīcai” nebija konkrētu sistēmas prasību, līdz ar to “ūdenskrituma” metode būtu ļoti neefektīva, jo būtu nepieciešams ilgs laiks, lai savāktu nepieciešamo informāciju, kā arī nebūtu garantijas, ka šīs sākotnējās prasības būs precīzi definētas.

Sistēmas autentifikācijas nodrošināšanai izmantojām “Devise” rīku [66]. Šis ir bezmaksas rīks autentifikācijas nodrošināšanai. Lietojām šo rīku, lai nebūtu lieki jātērē laiks, izveidojot pašiem savu autentifikācijas sistēmu.

4. “ĶELMĒNU MAIZNĪCAS” RAŽOŠANAS UZSKAITES SISTĒMAS ANALĪZE UN PROJEKTĒJUMS

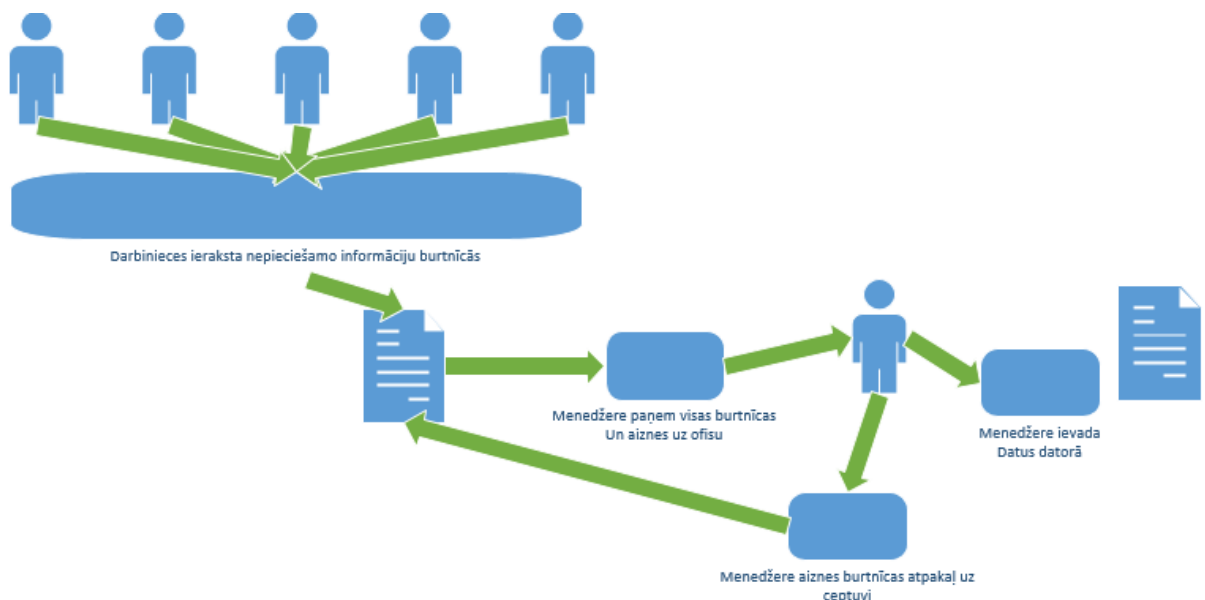
4.1. Ķelmēnu maiznīcas ražošanas uzskaites sistēmas prasības

Pirms “Demola Latvia” projekta “Ķelmēnu maiznīcā” visa ražošanas uzskaite notika “uz papīra”, rakstot ar roku. Piemēram, kad tika sagatavots maizes ieraugs, darbinieki ar roku to ierakstīja burtnīcā, kur glabājās visa informācija par ieraugiem. Tāpat arī pārējos maiznīcas ražošanas procesos tika izmantota šī pati metode. Vakarā “Ķelmēnu maiznīcas” menedžere devās uz maiznīcu, paņēma visas burtnīcas un ievadīja visus datus datorā. Šī sistēma bija ļoti novecojusi un neefektīva, jo nebija iespējams veikt plānošanu, tāpat bija ļoti grūti noteikt izejvielu atlikumu, līdz ar to, lai varētu turpināt ražot maizi, nācās saskarties ar problēmām, kad pēkšņi beidzās cukurs vai sāls un steidzami bija jādodas uz vietējiem veikaliem un jāizpērk attiecīgo izejvielu krājumi. Jāpiemin, ka “Ķelmēnu maiznīcā” maizes ražošana notiek 24 stundas diennaktī un septiņas dienas nedēļā; vienīgās maiznīcas brīvdienas ir valsts noteiktās svētku dienas.

Turklāt šī uzskaites metode bija ļoti neefektīva, jo viena maizes klaipa cepšanas process ilgst piecas dienas. Ar šo metodi nebija iespējams ātri noteikt, cik maizes klaipu būs izcepti pēc trim dienām, jo tad nācās iet uz ceptuvi un no jauna vākt un pārskatīt visu informāciju, tādējādi bija grūti plānot lielos pasūtījumus, ja tādi pēkšņi radās.

“Ķelmēnu maiznīcas” sākotnējais uzstādījums bija atrisināt viņu doto problēmu jebkāda veidā. Vienīgā prasība bija, lai nepieciešamajai informācijai varētu piekļūt no jebkuras pasaules vietas, kur ir pieejams internets.

4.1. attēlā redzama “Ķelmēnu maiznīcas” darba plūsmas diagramma.



4.1. att. “Ķelmēnu maiznīcas” darba plūsmas diagramma

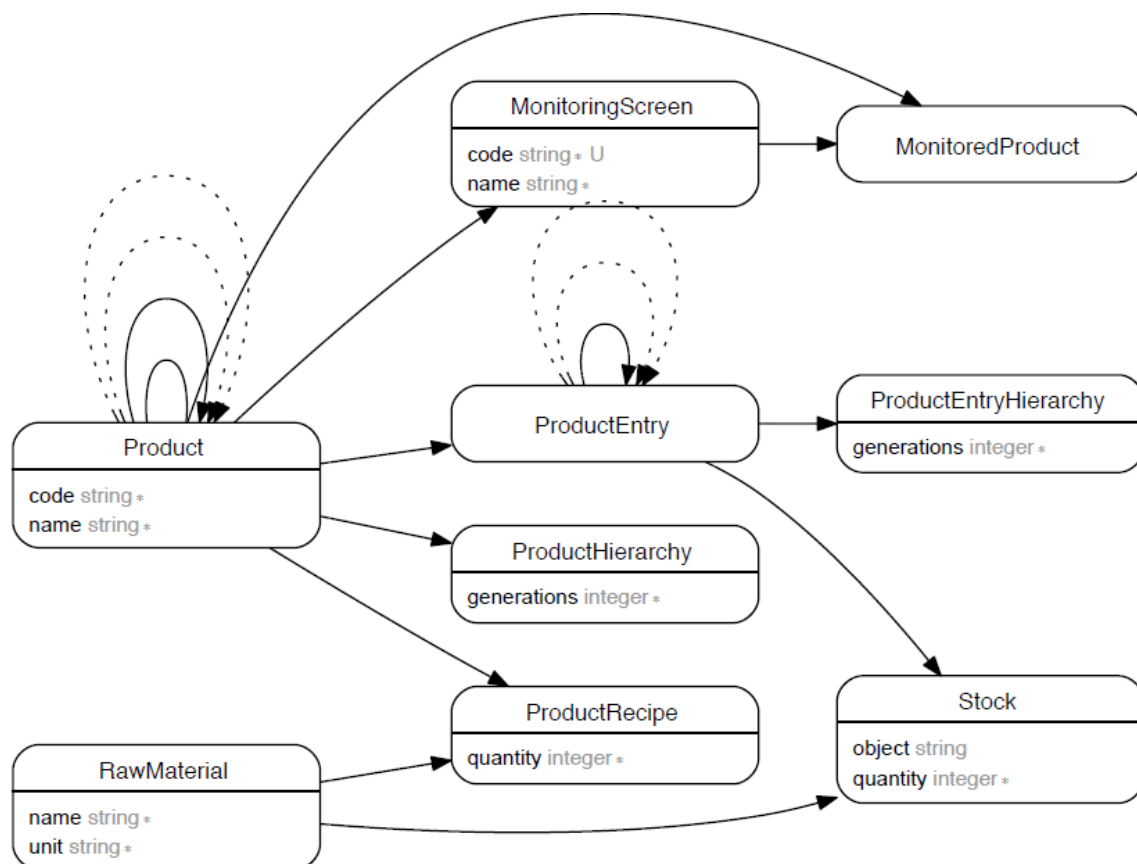
Diagrammā redzams, kā ražošanas uzskaitē “Ķelmēnu maiznīcā” notika pirms sistēmas izstrādes. Šī metode bija neefektīva un novecojusi, jo, lai menedžere varētu apskatīt saražoto, viņai ikreiz bija jādodas uz maiznīcu; dažkārt pat vairākas reizes dienā. Tas apgrūtināja menedžeres darbu, jo viņai visu laiku bija jāatrodas maiznīcas tuvumā.

4.2. Sistēmas datu struktūras projektējums

Datu analīze veikta, atrodoties uz vietas “Ķelmēnu maiznīcā”, kur tika apskatīti visi izmantotie dokumenti, kā arī izejmateriālu uzskaites process. Turpmāk izveidojām nelielu ražošanas procesa aprakstu un analīzi, tāpat apkopojām ieceres, kā gribētu to uzlabot. Šī apraksta ietvaros iekļāvām arī visus datus, kas tiks izmantoti sistēmā.

Sistēmas datu struktūra iegūta ar reversās inženierijas palīdzību no jau izveidotas datu bāzes, jo, uzlabojot programmu, nereti nācās uzlabot arī datu struktūru.

4.2. attēlā redzama datu struktūras diagramma.



4.2. att. Datu struktūras diagramma

Diagrammā ir deviņas tabulas. Maizes izstrādei ir piecas stadijas. Katrā no šīm stadijām izveidotais (ieraugs, mīkla, maizes klaipi, izcepti maizes klaipi, sapakota maize) ir kā produkts.

Galvenā tabula ir tabula Product, kas sevī ietver string laukus code un name. Name laukā tiek rakstīti produkta nosaukums, piemēram, rupjmaizes ieraugs.

Lai vieglāk varētu saprast kurā stadijā produkts atrodas ir izveidota tabula ProductHierarchy, kurā tiek atzīmēti integer tipa mainīgo no 1-5, jo tik daudz stadiju ir maizes cepšanas procesā.

Tāpat, lai iegūtu produktu nepieciešams izmantot kādas izejvielas, kas aprakstīti klasē RawMaterial, kur ir string lauki name un unit.

Lai varētu noskaidrot cik tieši daudz šī izejmateriāla ir nepieciešams ir tabula ProductRecipe, kurā ir viens integer lauks quantity, kurā tiek aprakstīts nepieciešamais daudzums.

Lai varētu noskaidrot cik daudz un kādu izejmateriālu ir, tika izveidota tabula Stock, kurā ir string tipa lauks object, kurā glabājas izejvielas nosaukums, un integer tipa lauks quantity, kurā tiek glabāts izejvielas daudzums.

Tāpat diagrammā ir tabula ProductEntry, kas nodrošina produktu nonākšanu Stock tabulā.

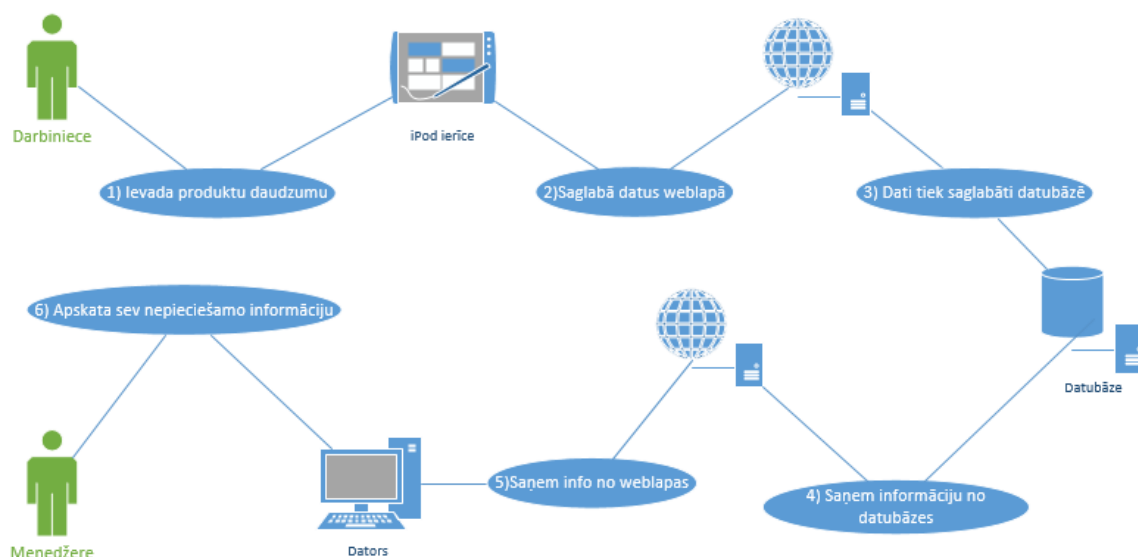
Tabula ProductEntryHierarchy tāpat kā ProductHierarchy atbild par produkta izstrādes stadiju.

Tabulā MonitoringScreen ir string tipa lauki code un name, kurā tiek glabāts šī ekrāna nosaukums, piemēram, saceptās_maizes_ekrāns.

Starp tabulā ir šādas saites. Tabulas RawMaterial un ProductEntry ir saistītas ar Stock tabulu, jo glabātvē tiek ielikti gan gatavie produkti, gan tiem nepieciešamās izejvielas. Product tabula ir saistīta ar MonitoringScreen tabulu, jo MonitoringScreen ekrānos tiek ievadīti un izveidoti jauni produkti. Tāpat MonitoringScreen ir saistīts ar MonitoredProduct, jo šis jaunais produkts tiek izveidots šajā ekrānā. Product tabula ir savienota ar Recipe tabulu, jo katram produktam ir zināma recepte, pēc kuras tas tiek gatavots, savukārt Recipe tabula ir savienota ar RawMaterials, jo tiek izmantoti tieši šie izejmateriāli.

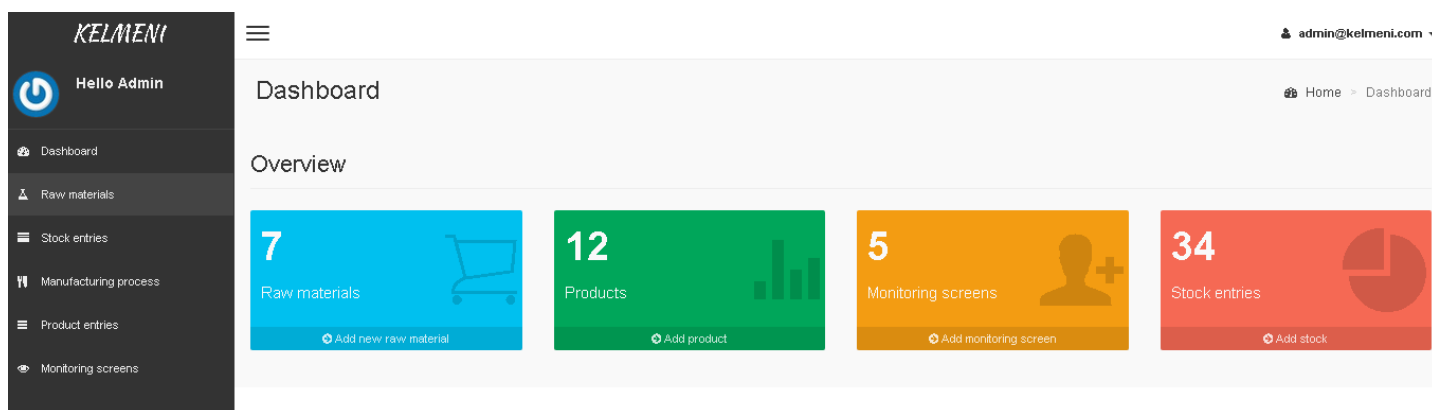
4.3. Saskarnes projektējums

4.3. attēlā ir redzama “Ķelmēnu maiznīcai” izstrādātās sistēmas lietošanas gadījumu diagramma. Jāpiemin, ka “Ķelmēnu maiznīcā” katrā maizes cepšanas stadijas vietā tiks novietota “iPad” ierīce, kurā darbinieces atzīmēs vajadzīgo informāciju. Diagrammā redzams, ka darbiniece konkrētajai ražošanas stadijai ievada izveidoto produktu skaitu “iPad” ierīcē. Tad dati tiek saglabāti datubāzē. Jebkurā brīdī, kad tas ir nepieciešams, menedžere vai jebkurš cits, kam ir izveidots konts, ar kuru iespējams pieslēgties sistēmai, var apskatīt visu nepieciešamo informāciju, piemēram, izejvielu daudzumu vai to, cik produktu šobrīd ir izveidots.



4.3. att. Lietošanas gadījumu diagramma

4.4. attēlā ir redzama sistēmas sākuma lapa, kurā parādīts, cik daudz izejvielu tiek izmantots, cik daudz produktu ir izveidots, cik daudz ekrānu, kur darbinieki ievadīs nepieciešamo informāciju, ir izveidoti, kā arī to, cik daudz izmaiņu veikts izejvielu glabātuvē. Augšējā labajā stūrī ir iespējams redzēt, ar kādu lietotāja vārdu lietotājs ir pieslēdzies sistēmai. Kreisajā malā redzama galvenā izvēlne, kurā atrodas visi “Ķelmēnu maiznīcai” nepieciešamie logi, par kuriem tiek skaidrots darba turpinājumā.



4.4. att. Sistēmas sākuma lapa

4.5. attēlā ir redzams logs, kurā iespējams apskatīties, kādas izejvielas šobrīd izmanto “Ķelmēnu maiznīca”, to nosaukumu, kā arī katras izejvielas šī brīža daudzumu glabātuvē. Tāpat ir iespējams redzēt, kad šī izejviela tikusi pievienota izejvielu logam. Turklāt iespējams apskatīt arī katras konkrētās izejvielas glabātuves izmaiņas, kā arī mainīt izejvielas informāciju vai, ja maiznīcā tā vairs netiks lietota, to izdzēst.

KELMENI

Hello Admin

Dashboard

Raw materials

Stock entries

Manufacturing process

Product entries

Monitoring screens

admin@kelmeni.com

HomeDashboard

Raw materials

HomeDashboard

List

Id	Name	Unit	Stock	Created at	Actions
2	Salt	g	5675	Thu, Jan 15, 2015 at 06:55:54 PM	Show StockEditDestroy
3	Sugar	g	6950	Thu, Jan 15, 2015 at 06:55:54 PM	Show StockEditDestroy
4	Rye Flour	g	1140	Thu, Jan 15, 2015 at 06:55:54 PM	Show StockEditDestroy
5	Wheat Flour	g	5215	Thu, Jan 15, 2015 at 06:55:54 PM	Show StockEditDestroy
6	Sourdough	g	555	Thu, Jan 15, 2015 at 06:55:54 PM	Show StockEditDestroy
7	Yeast	g	2348	Thu, Jan 15, 2015 at 06:55:54 PM	Show StockEditDestroy
8	Cream	10	500	Mon, May 11, 2015 at 05:36:34 PM	Show StockEditDestroy

New

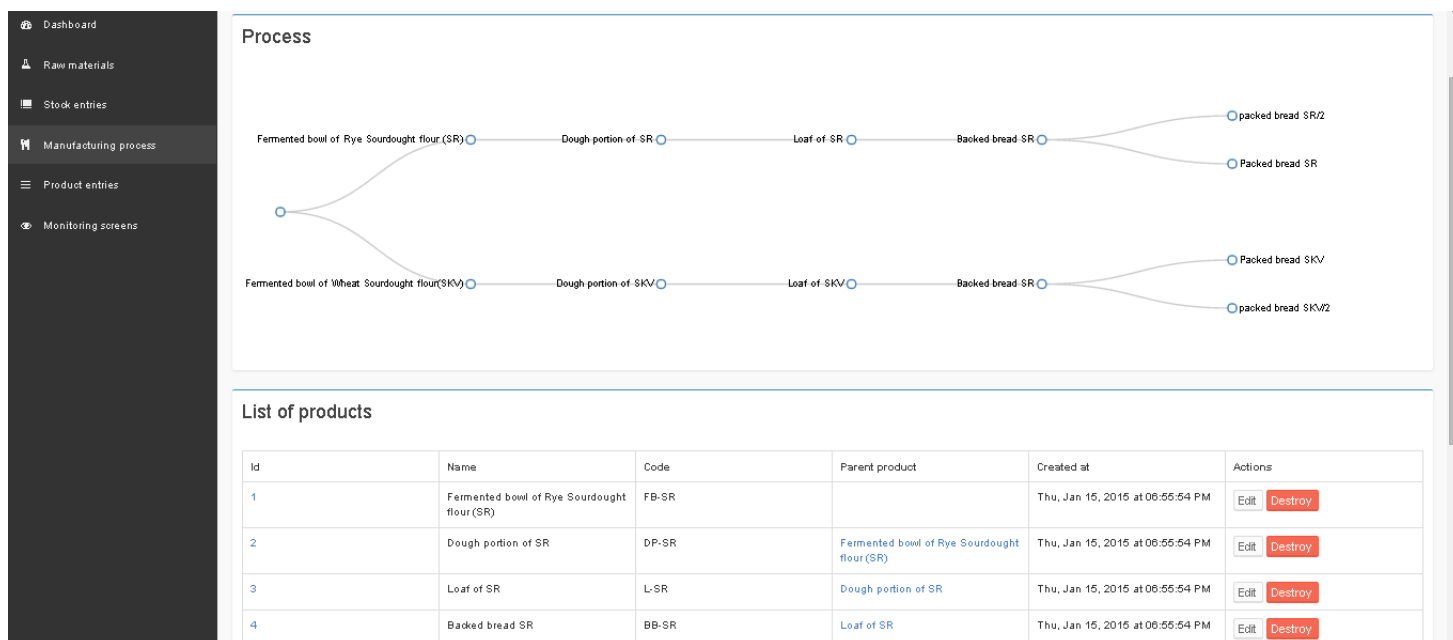
4.5. att. Izejmateriālu logs

4.6. attēlā ir redzams logs, kurā iespējams apskatīt visas izejvielu glabātuvēs veiktās izmaiņas un tādējādi kontrolēt, cik daudz izejvielu tiek ievests, kā arī to, cik daudz un kur tās tiek iztērētas. Kā redzams attēlā, maiznīcā tika ievests 570 kilogramu ierauga. Tāpat redzams, ka no glabātuves tika izņemti pieci kilogrami sāls un desmit kilogrami cukura, lai izveidotu rupjmaizes mīklu. Šajā logā iespējams labot katru glabātuves ierakstu, ja, piemēram, darbinieks ir kļūdījies un ievadījis nepareizu izejvielu daudzumu, vai ierakstu dzēst. Tāpat logā iespējams pievienot jaunas izejvielas izmaiņas. Nospiežot uz attiecīgās izejvielas nosaukuma, atveras logs, kurā iespējams redzēt konkrētās izejvielas izmaiņas izejvielu glabātuvē.

10	Initial stock	570	Sourdough	Thu, Jan 15, 2015 at 06:55:54 PM	Edit Destroy
11	Initial stock	570	Yeast	Thu, Jan 15, 2015 at 06:55:54 PM	Edit Destroy
12	Creation of Fermented bowl of Rye Sourdought flour (SR) #1 ProductEntry#1	-5	Salt	Fri, Jan 16, 2015 at 02:28:46 PM	Edit Destroy
13	Creation of Fermented bowl of Rye Sourdought flour (SR) #1 ProductEntry#1	-10	Sugar	Fri, Jan 16, 2015 at 02:28:46 PM	Edit Destroy
14	Creation of Fermented bowl of Rye Sourdought flour (SR) #1 ProductEntry#1	-15	Rye Flour	Fri, Jan 16, 2015 at 02:28:46 PM	Edit Destroy
15	Creation of Fermented bowl of Rye Sourdought flour (SR) #1 ProductEntry#1	-2	Yeast	Fri, Jan 16, 2015 at 02:28:46 PM	Edit Destroy
16	Creation of Fermented bowl of Wheat Sourdought flour(SKV) #2 ProductEntry#2	-5	Salt	Fri, Jan 16, 2015 at 02:28:54 PM	Edit Destroy
17	Creation of Fermented bowl of Wheat Sourdought flour(SKV) #2 ProductEntry#2	-10	Sugar	Fri, Jan 16, 2015 at 02:28:54 PM	Edit Destroy

4.6. att. Izejvielu logs

4.7. attēlā ir redzams maizes cepšanas process, kā arī visas procesa stadijas. Iespējams redzēt maizes cepšanas stadijas nosaukumu un produkta, no kura tiek izveidots jaunais produkts, nosaukumu, piemēram, lai izveidotu maizes klaipus, ir nepieciešama mīkla. Tāpat iespējams redzēt stadijas izveidošanas laiku. Visas maizes cepšanas stadijas var izmainīt vai izdzēst, ja šāda stadija vairs nav nepieciešama. Tāpat iespējams pievienot jaunu maizes cepšanas stadiju.



4.7. att. Maizes cepšanas procesa logs

4.8. attēlā ir attēlots produktu logs, kur redzams, kāds produkts ir izveidots, kurā datumā un cik tas izveidots, kā arī produkta izveidei nepieciešamās izejvielas. Tāpat redzams, no kā radies šis produkts — no kāda iepriekšējā produkta tas izveidots. Pēc produkta izveidošanas no glabātuves tiek atņemtas attiecīgā daudzuma izejvielas, kas bija nepieciešamas produkta izveidei. Tāpat šajā logā iespējams pievienot jaunu produktu, kā arī izmainīt vai izdzēst jau esošos produktus.

KELMENI
⋮
Hello Admin

admin@kelmeni.com
Home > Dashboard

Product entries

Id	Parent	Product	Created at	Stock entries	Actions
1		Fermented bowl of Rye Sourdought flour (SR)	Fri, Jan 16, 2015 at 02:28:46 PM	Salt Sugar Rye Flour Yeast	<button>Edit</button> <button>Destroy</button>
2		Fermented bowl of Wheat Sourdought flou(SKV)	Fri, Jan 16, 2015 at 02:28:54 PM	Salt Sugar Wheat Flour Sourdough Yeast	<button>Edit</button> <button>Destroy</button>
3		Fermented bowl of Rye Sourdought flour (SR)	Fri, Jan 16, 2015 at 04:26:41 PM	Salt Sugar Rye Flour Yeast	<button>Edit</button> <button>Destroy</button>
4	3	Baked bread SR	Sat, Mar 21, 2015 at 10:17:53 AM		<button>Edit</button> <button>Destroy</button>
5	1	Baked bread SR	Mon, May 11, 2015 at 05:34:34 PM		<button>Edit</button> <button>Destroy</button>
6	5	Fermented bowl of Rye Sourdought flour (SR)	Mon, May 11, 2015 at 05:41:14 PM	Salt Sugar Rye Flour Yeast	<button>Edit</button> <button>Destroy</button>
7	6	Loaf of SR	Mon, May 11, 2015 at 05:47:42 PM		<button>Edit</button> <button>Destroy</button>
8	1	Fermented bowl of Rye Sourdought flour (SR)	Fri, May 15, 2015 at 04:21:13 PM	Salt Sugar Rye Flour Yeast	<button>Edit</button> <button>Destroy</button>
9	3	Dough portion of SR	Fri, May 15, 2015 at 04:22:18 PM		<button>Edit</button> <button>Destroy</button>
10	4	Loaf of SR	Fri, May 15, 2015 at 04:22:56 PM		<button>Edit</button> <button>Destroy</button>

New

4.8. att. Produktu logs

4.9. attēlā redzams logs, kurā iespējams apskatīt esošo darbinieku logu nosaukumu un izveidošanas datumu. Tāpat iespējams pievienot logus, kur darbinieki, izmantojot “iPad” ierīces, ievadīs nepieciešamo informāciju. Šajā logā iespējams izmaniīt vai dzēst jau esošos logus, kā arī, nospiežot uz attiecīgā loga Id skaitļa, iespējams apskatīt saiti, kur pieejams šis darbinieka ekrāns, kā arī kādus produktus iespējams izveidot šajā logā

List

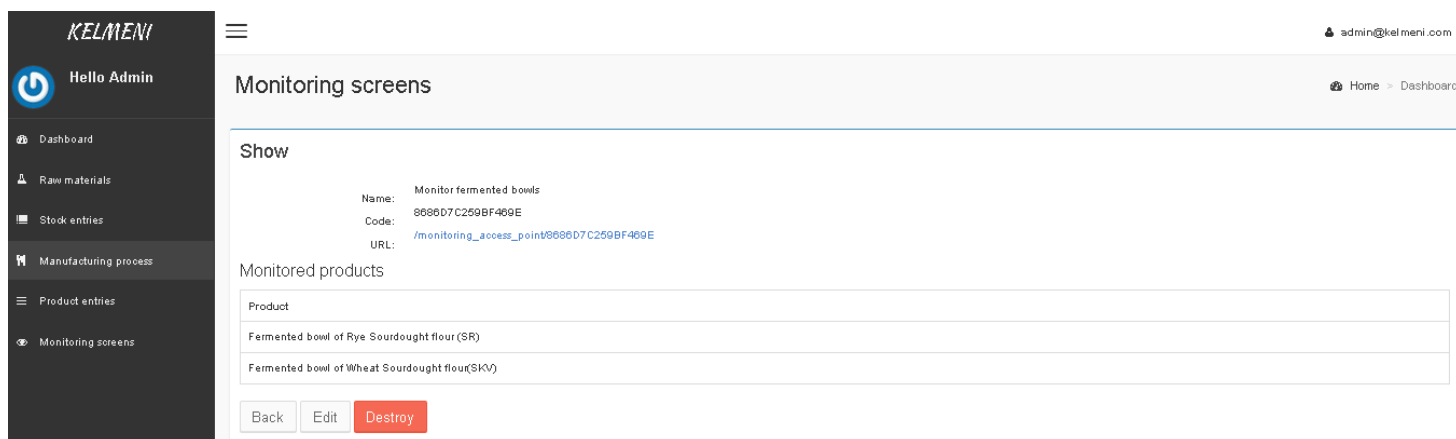
Id	Name	Code	Created at	Actions
1	Monitor fermented bowls	8686D7C259BF469E	Thu, Jan 15, 2015 at 07:08:03 PM	Edit Destroy
2	FERMENTED BOWLS	C47A137F0291D6F7	Fri, Jan 16, 2015 at 04:11:14 PM	Edit Destroy
3	Monitoring sough portion	0975DF6187B2B904	Sat, Mar 21, 2015 at 10:19:45 AM	Edit Destroy
4	Fermented bowl of wheat	8C9EA3FA045AD405	Mon, May 11, 2015 at 05:48:30 PM	Edit Destroy
5	Backed bread SR	924E61FCE013C04F	Fri, May 15, 2015 at 04:24:23 PM	Edit Destroy

New

4.9. att. Darbinieku ekrānu logs

4.10. attēlā redzams logs, kurā iespējams apskatīt darbinieku ekrānu. Šajā logā iespējams apskatīt šī darbinieka ekrāna nosaukumu, kā arī iespējams redzēt kādus produktus iespējams izveidot šajā logā. Tāpatšajā logā iespējams redzēt šī darbinieku ekrāna linku, kurā

tas pieejams, un uz kura uzklikšķinot sistēma pārdresē lietotāju uz tieši šo darbinieka logu. Redzamo logu iespējams izmainīt, kā arī dzēst.



KELMENI

Hello Admin

Dashboard

Raw materials

Stock entries

Manufacturing process

Product entries

Monitoring screens

Monitoring screens

admin@kelmeni.com

Home Dashboard

Show

Name: Monitor fermented bowls

Code: 8686D7C259BF469E

URL: /monitoring_access_point/8686D7C259BF469E

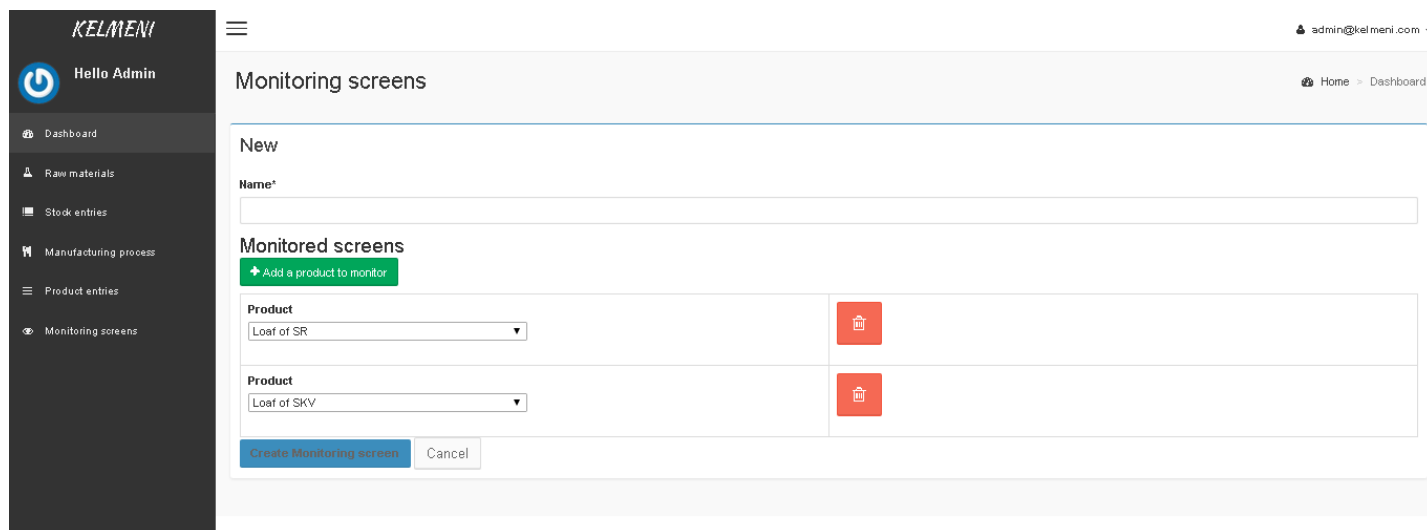
Monitored products

Product
Fermented bowl of Rye Sourdough flour (SR)
Fermented bowl of Wheat Sourdough flour (SKV)

Back Edit Destroy

4.10. att. Darbinieku ekrānu apskates logs

4.11. attēlā redzams, kā iespējams pievienot jaunu darbinieku logu. Ir nepieciešams ievadīt ekrāna nosaukumu, kā arī produktus, ko varēs izveidot šajā ekrānā. Iespējams izvēlēties, cik un kādus produktus varēs pievienot šajā logā. Piemēram, pie izejvielu piegādes ekrāna varēs pievienot visas izejvielas, kas maiznīcai nepieciešamas, turpretim ierauga stadijā būs iespējams pievienot saldskābmaizes vai rupjmaizes ieraugu.



KELMENI

Hello Admin

Dashboard

Raw materials

Stock entries

Manufacturing process

Product entries

Monitoring screens

Monitoring screens

admin@kelmeni.com



Home Dashboard

New

Name*

Monitored screens

+ Add a product to monitor

Product	
Loaf of SR	
Loaf of SKV	

Create Monitoring screen Cancel

4.11. att. Darbinieku ekrāna izveide

Ikdienā darbinieki strādās tikai ar trim no šiem logiem, proti, darbinieces maiznīcā ievadīs izveidotos produktus, un “Kelmēnu maiznīcas” menedžere apskatīsies, cik daudz produktu ir izgatavots un cik daudz izejvielu vēl atlicis.

5. “ĶELMĒNU MAIZNĪCAS” RAŽOŠANAS UZSKAITES SISTĒMAS IZSTRĀDE UN IEVIEŠANA

5.1. “Ķelmēnu maiznīcas” ražošanas uzskaites sistēmas izstrāde

Sistēmas izstrāde ilga vairāk nekā četrus mēnešus, jo pēc projekta beigām sistēma tika pilnveidota. Sistēma tika uzlabota arī pēc “Demola Latvia” gala prezentācijas, jo komanda vēlējās būt izdarījusi maksimāli daudz, lai “Ķelmēnu maiznīca” būtu ieinteresēta iegādāties produkta licenci un tādējādi atalgot komandu par labu padarīto darbu un problēmas atrisināšanu. Jāpiemin, ka sistēmas izstrāde nav pabeigta, jo līdz ar sistēmas prezentēšanu “Ķelmēnu maiznīcai” darbs projektā bija noslēdzies, savukārt visas nepilnīgās sistēmas daļas (nepabeigtās funkcionalitātes), ja “Ķelmēnu maiznīca” to vēlēties, tiks izstrādātas pēc projekta beigām, atsevišķi vienojoties ar personām, kas pabeigs sistēmas izveidi. Sākotnēji pie sistēmas izstrādes darbojās visa komanda sešu cilvēku sastāvā, jo katrs no dalībniekiem varēja sniegt ieguldījumu komandas darbā, sniedzot jaunas idejas. Kad bija noskaidrotas sākotnējās prasības un izveidoti sākotnējie sistēmas prototipi, darbu pie sistēmas turpināja tikai divi dalībnieki: darba autors un ERASMUS students no Francijas. Pārējiem komandas dalībniekiem nebija nepieciešamo zināšanu, lai varētu turpināt darbu pie sistēmas izstrādes, tāpēc viņi vairāk strādāja pie līdzīgu sistēmu meklēšanas un analīzes. Sistēmas izstrāde parasti notika “Demola Latvia” telpās, un abi izstrādātāji, savstarpēji komunicējot, varēja veikt sistēmas izstrādei nepieciešamās darbības.

5.2. Rezultātu apraksts

Projekta laikā tika izstrādāta sistēma, ar kuras palīdzību “Ķelmēnu maiznīca” var labāk pārraudzīt ražošanu. Sistēmā iespējams redzēt izejvielu daudzumu un gatavos produktus. Katrai maizes izstrādes stadijai, kur iepriekš tikai izmantota pildspalva un papīrs, šobrīd ir izstrādāts speciāls logs, kurā būs iespējams ievadīt visu nepieciešamo informāciju, savukārt pēc tam aplikācija šo informāciju saglabās datubāzē, līdz ar to “Ķelmēnu maiznīcas” menedžerim būs iespēja daudz vienkāršāk pārskatīt ražošanas procesu. Ievades ekrāna piemēru iespējams apskatīt pielikumā Nr. 3. Aplikācijā ir iespējams izveidot ievades ekrānus, kā arī pielāgot tos katras stadijas vajadzībām. Logu izveidošanas lapu ir iespējams apskatīt pielikumā Nr. 4. Katras maizes ražošanas stadijas vietā būs “iPad” ierīce, kurā tiks uzstādīts un nobloķēts tieši konkrētajai ražošanas stadijai nepieciešamais ekrāns. Izveidojot

jaunu produktu, no izejvielu glabātuves tiek noņemtas jaunā produkta izgatavošanai nepieciešamās izejvielas vajadzīgajā daudzumā. Tāpat menedžeris vai darbinieks var pievienot izejvielu izmaiņas, piemēram, ja uz maiznīcu tiek atvests cukurs vai sāls, vai — tieši otrādi — šīs izejvielas tiek izlietotas citām vajadzībām, bet glabātnē tās vairs nav. Sistēmā ir iespējams pievienot arī jaunas izejvielas, ja maiznīca sāks tādās izmantot. Tāpat iespējams pievienot arī jaunus produktus, ja “Ķelmēnu maiznīca” nolems sākt ražot jaunus maizes veidus. Ir iespējams redzēt, kur un cik daudz tika izmantota katra izejviela. Sistēmā ir iespējams pievienot jaunas ražošanas stadijas, ja tādās maiznīcā tiks ieviestas.

SECINĀJUMI UN PRIEKŠLIKUMI

Apskatījis un izmantojis sistēmas izstrādes rīkus “Ķelmēnu maiznīcas” sistēmas automatizētai izveidei, autors secina:

1. Sistēmu izstrādes rīku izmantošanai ir būtiska nozīme sistēmu automatizētā izveidē, jo to izmantošana atvieglo sistēmas izstādi, ietaupa laiku un palīdz pārbaudīt sistēmas pareizu darbību.
2. Populārākās izstrādes metodes ir ūdenskrituma metode, ko parasti izmanto, ja projekta sākumā ir skaidri zināmas sistēmas prasības un izstrādes laikā tās netiks mainītas, kā arī “Agile” metode, kas sastāv no neierobežota skaita iterācijām un tiek lietota, ja projekta sākumā informācija par sistēmas prasībām ir ļoti skopa un izstrādes laikā šīs prasības var mainīties.
3. Projektu izstrādei ir vairākas stadijas, un tās nozīmīgumu un ilgumu ietekmē izmantotās izstrādes metodes.
4. Ir ļoti grūti noteikt pašus populārākos sistēmu izstrādes rīkus, jo to ir ļoti daudz un tie ir dažādi. Tāpat ir sarežģīti noteikt populārākos rīkus, izmantojot rīku iedalījumu grupās, jo nav viena noteikta topa, pēc kura tie tiek sarindoti, savukārt dažādos informācijas avotos šie rīki var būt pilnīgi dažādi.
5. “Demola Latvia” projekts ir noderīgs gan iegūto zināšanu, gan kontaktu ziņā. Darba autors iesaka ikvienam studentam, kam ir iespēja šajā projektā piedalīties, pieņemt izaicinājumu un piedalīties “Demola Latvia” projektā.
6. Sistēmas izstrādē liela nozīme ir projekta izstrādes komandai, jo viena komandas dalībnieka kļūda var maksāt ļoti daudz laika, tāpēc komandu ieteicams izvēlēties ļoti rūpīgi. Tāpat ļoti liela nozīme ir komandas sadarbībai un prasmei sastrādāties.
7. “Ķelmēnu maiznīca” sākumā bija izvirzījusi ļoti skopas prasības, tāpēc sistēma tika izstrādāta pēc “Agile” metodes un izstrādes laikā sistēmai tika veikta vairāki labojumi. Ja izstrādes sākumā prasības būtu skaidri zināmas, tad būtu bijis iespējams izmantot “ūdenskrituma” metodi.
8. Kopumā sistēmas izstrādē tika izmantoti 11 dažādi sistēmas izstrādes rīki, kas bija pietiekami, lai manāmi atvieglotu sistēmas izstrādi.
9. Vairums rīku tika izmantoti, lai sistēmas izstrādē varētu darboties vairāki projekta dalībnieki un nebūtu problēmu, ja dalībnieki izmantotu atšķirīgas operētājsistēmas.
10. Sistēmas izstrāde ilga vairāk nekā četrus mēnešus, un to laikā sistēmas izstrādes komanda divas reizes viesojās “Ķelmēnu maiznīcā”, lai iegūtu papildu informāciju, kā arī parādītu sistēmas prototipus un iegūtu turpmāk nepieciešamo uzlabojumu

aprakstu, savukārt projekta noslēgumā tika izveidota strādājoša, taču nepabeigta sistēma.

11. Autors iesaka katra projekta sākumā apskatīt sistēmas izstrādes metodes un izmantojamās sistēmas izstrādes rīkus un izlemēt, kuras metodes un rīkus nepieciešams izmantot sistēmas izstrādes laikā, jo pareizi izvēlēta izstrādes metode un sistēmas izstrādes rīki var būtiski uzlabot darba kvalitāti un laiku, turpretim, izvēloties nepareizus rīkus, var zaudēt daudz laika un līdz ar to arī līdzekļus.
12. Visi sākumā izvirzītie uzdevumi ir izpildīti, un bakalaura darbs ir pabeigts.
13. Darba sākumā izvirzītais mērķis ir izpildīts, proti, sistēmas izstrādes laikā ir apskatīti, izmantoti un vēlāk darbā aprakstīti sistēmas izstrādes rīki.

Darba izstrādes laikā autoram radās šādi priekšlikumi:

1. Autors iesaka Ventspils Augstskolas Informācijas tehnoloģiju fakultātes studiju programmā “Datorzinātnes” iekļaut kursus par testēšanu, jo, veidojot šo darbu, autors ir sapratis, cik liela nozīmes sistēmu izstrādē ir testēšanai un cik ļoti darbs tiek atvieglots, izveidojot automatizētu testēšanu, lai testēšana pie katras izmaiņas nebūtu jāveic manuāli.
2. Autors iesaka Ventspils Augstskolas Informācijas tehnoloģiju fakultātes studiju programmā “Datorzinātnes” iekļaut padziļinātu praktisko kursu par sistēmas koda uzturēšanas rīkiem un vienādas sistēmas vides izstrādes rīku izmantošanu, jo ar šo rīku izmantošanu nākas saskarties katrā sistēmu izstrādē.

IZMANTOTĀS LITERATŪRAS UN AVOTU SARAKSTS

1. Case Tools, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<http://www.umsl.edu/~sauterv/analysis/F08papers/View.html>
2. Hofner A.J., George F. J., Valacich S.J., Chapter 4 Automated Tools for Systems Development [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<http://www.slideshare.net/emolagi/chapter04-automated-tools-for-systems-development>
3. Sparāne, L., Zunde, K., Combining next generation talents with meaningful challenges, 2014
4. “Demola Latvia”, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <http://latvia.demola.net>
5. World top countries for higher education, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<http://www.prnewswire.com/news-releases/ranking-reveals-worlds-top-countries-for-higher-education-206676221.html>
6. Uzulāns, J. Skolotājs mūsdienu informatīvajā telpā IV. Projekts. Rīga : RaKa, 2005. 84. lpp.
7. Geipele, I., Tambovceva, T. Projektu vadīšana studijām un biznesam. Rīga : Valters un Rapa, 2004.187.lpp
8. Uzulāns, J. Projektu vadīšana mūsdienu apstākļos. Rīga : SIA „Drukātava”, 2007. 102. lpp
9. Projektu pārvaldība, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
http://estudijas.lu.lv/pluginfile.php/68978/mod_resource/content/0/EUCIP-5.pdf
10. Waterfall model, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
http://en.wikipedia.org/wiki/Waterfall_model
11. Agile software development, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
http://en.wikipedia.org/wiki/Agile_software_development
12. Dzīves ciklu modeļi, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<http://estudijas.lu.lv/file.php/3582/I%20semestris/Kospekti/1.Iss%20parskats%20par%20kursu.htm>
13. Vītoliņš, V. IT projektu pārvaldība, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<https://odo.lv/ftp/files/ITProjektuParvaldiba.pdf>
14. Dosbergs, D. Programminženierijas standartu sistēma, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<https://www.google.lv/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&>

uact=8&ved=0CCAQFjAA&url=http%3A%2F%2Festudijas.lu.lv%2Fmod%2Fresource%2Fview.php%3Fid%3D131424&ei=91tLVdWkFMOTsgG8yoGwAg&usg=AFQjCNHnS4LFdOB_hz8eVTgg2nIA4TBtkw&sig2=M__ -
xwwj_VPmjRK0kflB1Q&bvm=bv.92765956,d.bGg

15. Springe, R., Granāts, M. Labās prakses vadlīnijas iepirkumiem IT jomā, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<http://www.slideshare.net/rspringe/tapisliktafin>
16. McCormick, M. Waterfall vs. Agile Methodology, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
http://www.mccormickpcs.com/images/Waterfall_vs_Agile_Methodology.pdf
17. Cohn, M. Agile Succeeds Three Times More Often Than Waterfall, February 13, 2012, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<http://www.mountaingoatsoftware.com/blog/agile-succeeds-three-times-more-often-than-waterfall>
18. Agile vs waterfallmethod, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<http://clearcode.cc/2014/12/agile-vs-waterfall-method/>
19. Antony, R. Case Tools, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<http://www.slideshare.net/arnoldindia/case-tools>
20. Lenhart, S. Systems Analysis, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<http://www.umsl.edu/~sauterv/analysis/Term%20Papers/f14/CASE.htm>
21. Unified Modeling Language [tiešsaiste][skatīts 31.05.2015.] Pieejams:
http://en.wikipedia.org/wiki/Unified_Modeling_Language
22. Bizness Process Modeling Language [tiešsaiste][skatīts 31.05.2015.] Pieejams:
http://en.wikipedia.org/wiki/Business_Process_Modeling_Language
23. Star Uml [tiešsaiste][skatīts 31.05.2015.] Pieejams: <http://staruml.io/>
24. Visual-paradigm, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <http://www.visual-paradigm.com/>
25. Microsoft Visio [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<https://products.office.com/en-us/visio/flowchart-software>
26. Basecamp, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <https://basecamp.com/>
27. Jira, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<https://www.atlassian.com/software/jira>
28. Asana, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <https://asana.com/pricing>

29. Microsoft Project, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<https://products.office.com/en-us/Project/project-and-portfolio-management-software>
30. Github, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <https://github.com/features>
31. Bitbucket, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <https://bitbucket.org/features>
32. Cscope, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <http://mylinuxbook.com/cscope/>
33. Eclipse, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <http://www.eclipse.org/>
34. Balsamiq Mockups, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<https://balsamiq.com/products/mockups/>
35. Testing White, Black, Gray, Box, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
http://www.crosschecknet.com/soa_testing_black_white_gray_box.php
36. RSpec, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <http://rspec.info/>
37. Capybara, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<https://github.com/jnicklas/capybara>
38. Dosbergs, D., Testēšana Testēšanas dokumentācija, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
https://www.google.lv/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0CB4QFjAA&url=http%3A%2F%2Festudijas.lu.lv%2Ffile.php%2F1756%2FLekcijas%2F08-Testesana.ppt&ei=56I_Vb-VKYqesAH6sICYAQ&usg=AFQjCNEcWQGANqWXwxylQ_resykAQjN3OA&sig2=w8E0-YvNhMIQ9GaJ7iJ8jQ&bvm=bv.91665533,d.bGg
39. Utest, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <http://www.utest.com/>
40. Bugzilla, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <https://www.bugzilla.org/>
41. Best test anagement tools for sowtware testers [tiešsaiste][skatīts 31.05.2015.] Pieejams: <http://www.softwaretestinghelp.com/15-best-test-management-tools-for-software-testers/>
42. Test Rail, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<http://www.gurock.com/testrail/>
43. Microsoft Project un Basecamp salīdzinājums, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <https://kpakiwal.files.wordpress.com/2014/02/project-management-case-tools.pdf>
44. UML tools, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
http://www.angelfire.com/ultra/all_about_uml/uml_tools.htm
45. Argo UML, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <http://argouml.tigris.org/>

46. Singh, S. K. Database Systems: Concepts, Design and Applications, India, 2011, 912. lpp
47. ER- Studio, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<http://www.embarcadero.com/products/er-studio>
48. Top wireframing tools, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<http://www.creativebloq.com/wireframes/top-wireframing-tools-11121302>
49. Axure, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <http://www.axure.com/features>
50. Programming language popularity, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<http://langpop.com/>
51. Best compilers and i-ides for c programmers, [tiešsaiste][skatīts 31.05.2015.]
Pieejams: <http://codecall.net/2014/02/26/best-compilers-and-ides-for-cc-programmers/>
52. CodeBlock, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <http://www.codeblocks.org/>
53. Tools for java develipers, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<https://blog.newrelic.com/2014/05/21/toolsforjavadevelopers/>
54. Wayner, P., Open source programming tools on the rise, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <http://www.infoworld.com/article/2623734/development-tools/open-source-programming-tools-on-the-rise.html>
55. Ruby programming language, [tiešsaiste][skatīts 31.05.2015.]
Pieejams:<https://www.ruby-lang.org/en/>
56. Hound Ci, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <https://houndci.com/>
57. Heroku, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <https://www.heroku.com/>
58. Best 9 Cloud development platforms, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<http://www.infoworld.com/article/2606630/development-environments/9-cloud-development-platforms-on-the-rise.html#slide3>
59. Google Cloud, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<https://cloud.google.com/appengine/docs>
60. Microsoft Azure, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<http://azure.microsoft.com/en-gb/overview/what-is-azure/>
61. Teatro, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <https://teatro.io/help>
62. Travis CI, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <https://travis-ci.com/plans>
63. Circle CI, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <https://circleci.com/pricing>
64. VagrantUp, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <https://www.vagrantup.com/>
65. Toggl, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <https://toggl.com/features>

66. Devise, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<https://github.com/plataformatec/devise>
67. “Demola Latvia “ grafiks [tiešsaiste][skatīts 31.05.2015.] Pieejams:
<http://latvia.demola.net/news/demola-latvia-fall2014-schedule#.VT5lxSHtmkp>
68. Testēšana, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
http://home.lu.lv/~garnican/Testesan/test_n40.htm
69. Yde, L., M.Sc, Selected Topics in Software Development, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <http://www.diku.dk/forskning/performance-engineering/Courses/Software-development-2008/Slides/testing.pdf>
70. Petridou, M, Software Project Management, Prototyping and CASE Tools, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
https://www.google.lv/url?sa=t&rct=j&q=&esrc=s&source=web&cd=7&cad=rja&uact=8&ved=0CEwQFjAG&url=http%3A%2F%2Fwww.cs.nott.ac.uk%2F~mzp%2FmyLecture%2FLecture13.ppt&ei=kcc_Vc-HG8GqsgGir4G4Bg&usg=AFQjCNFBBytWTN7UYtZ3bzz0RmuYNSysiw&sig2=k_NIte75rn2w3KOJYoarbw
71. Case tools overview, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
http://www.tutorialspoint.com/software_engineering/case_tools_overview.htm
72. Nīkiforova, O. Programmatūras attīstības tehnoloģijas, [tiešsaiste][skatīts 31.05.2015.] Pieejams:
http://ditf.afraid.org/ditf/3%20kurss/Programmatuuras%20Attiistibas%20Tehnologijas/slaidi%202006/PAT_02_Process_Disciplined_studentsiem.pdf
73. Informācijas tehnoloģijas projektu vadītājs, [tiešsaiste][skatīts 31.05.2015.] Pieejams: <http://visc.gov.lv/profizglitiba/dokumenti/standarti/ps0170.pdf>
74. Mikose, M. Informāciju tehnoloģiju projekts uzņēmumā. Rīga : „SIA Biznesa Augstskola Turība”, 2006. 242.lpp.

PIELIKUMI

Pielikums Nr.1



DIPLOMA

I

1st place

BAKERY WATCHDOG

in recognition of outstanding performance during **DEMOLA Fall'2014** season
and presentation delivery during final pitch is awarded to


Lilita Sparāne
Executive Director of Latvian IT Cluster
DEMOLA Operator

January 29th, 2015, Riga


Jury: Māris Millers, Kārlis Valters,
Viesturs Sosārs, Edgaras Kriukonis,
Evija Zvirbule-Mirķe



IEGULDĪGUMS TAVĀ NĀKOTNĒ!





DEMOLA
LATVIA

DIPLOMA

This is to certify that

Mārtiņš Bērtulis
within team *Bakery Watchdog*

has successfully completed **DEMOLA Fall'2014 project** and improved such competences as presentation skills, teamwork, business strategy development, new product design, prototyping and validation.


Lilita Sparāne
Executive Director of Latvian IT Cluster
DEMOLA Operator



January 29th, 2015, Riga


Kristine Zunde
Facilitator



IEGULDĪGUMSTAVĀ NĀKOTNĒ!



Fermented bowl of Rye Sourdought flour (SR)	0	 
Fermented bowl of Wheat Sourdought flour(SKV)	0	 

List

Id	Name	Code	Created at	Actions
1	Monitor fermented bowls	8686D7C259BF469E	Thu, Jan 15, 2015 at 07:08:03 PM	Edit Destroy
2	FERMENTED BOWLS	C47A137F0291D6F7	Fri, Jan 16, 2015 at 04:11:14 PM	Edit Destroy
3	Monitoring sough portion	0975DF6187B2B904	Sat, Mar 21, 2015 at 10:19:45 AM	Edit Destroy
4	Fermented bowl of wheat	8C9EA3FA045AD405	Mon, May 11, 2015 at 05:48:30 PM	Edit Destroy
5	Backed bread SR	924E61FCE013C04F	Fri, May 15, 2015 at 04:24:23 PM	Edit Destroy

[New](#)

GALVOJUMS

Ar šo es, Mārtiņš Bērtulis, galvoju, ka bakalaura darbs ir izpildīts patstāvīgi, konsultējoties ar darba vadītāju. No svešiem pirmavotiem ņemtā informācija ir norādīta ar atsaucēm, dati un definējumi ir uzrādīti darbā. Šis darbs tādā vai citādā veidā nav nekad iesniegts nevienai citai pārbaudījumu komisijai.

Esmu informēts, ka mans bakalaura darbs tiks ievietots un apstrādāts Vienotajā datorizētajā plaģiāta kontroles sistēmā plaģiāta kontroles nolūkos.

20__gada _____

(paraksts)