## 7.Uzdevums

Uzraksti pats savu testēšanas plānu – produktam “OnMenu”, kas ļauj aplūkot dažādu ēstuvju ēdienkartes un arī pasūtīt ēdienu.

**TEST PLAN**

**“OnMenu”**

Table of Contents (Saturs)

[Ievads 3](#_Toc109387551)

[1.1 Mērķi 3](#_Toc109387552)

[1.2 Komandas biedri 3](#_Toc109387553)

[2 PĀRSKATS 3](#_Toc109387554)

[3 PIEŅĒMUMI / Riski 4](#_Toc109387555)

[3.1 Pieņēmumi 4](#_Toc109387556)

[3.2 Riski 4](#_Toc109387557)

[4 TestA pieeja 4](#_Toc109387558)

[4.1 Testu Automatizācija 4](#_Toc109387559)

[5 Testa vide 5](#_Toc109387560)

[6 plānotie posma darbi / Piegādes 5](#_Toc109387561)

[6.1 Testēšanas grafiks 5](#_Toc109387562)

[6.2 Piegādes 5](#_Toc109387563)

# Ievads

Testa plāns tiek veidots, lai izskaidrotu testa pieeju visiem komandas biedriem. Testa plāns iekļauj mērķus, pārskatus, izpildes grafikus, riskus un pieeju. Šis dokuments skaidri izskaidro, kādiem jābūt testa rezultātiem un kam ir jābūt iekļautam testa plānā.

## Mērķi

Tiks izmantots testa uzdevumu veidošanas rīks JIRA (web-bāzēts Testa vadības rīks), ar kura palīdzību tisk veidoti un glabāti testi un to rezultāti. Šis rīks ir veidots Java programmēšanas valodā, izmantojot ‘mysql’ databāzi. Testa komanda ir atbildīga par produkta testēšanu tā, lai progamma sasniegtu savu izpildes mērķi. Testa komanda šī projekta gadījumā pārstāv gan klienta, gan testētāja skatupunktu.

JIRA programmā būt iespējams veikt un saglabāt manuālos test casus, kā arī redzēt veicamo test case progresu. Galvenais būtu produkta funkcionalitāte nevis ierobežots produkta izstrādes laiks, tāpēc teste case pārvaldībai izmantosim Agile/ Kanban modeli.

## Komandas dalībnieki

Projektam atvēlētais laiks pie konkrētās OnMenu programmas izstrādes ir ir diezgan liels (1- 1,5 gads), tāpēc šī produkta gadījumā komanda ir maksimāli maza. Turklāt visi esošie darbinieki ir štata darbinieki kādā vienā kopīgā ofisā- uzņēmumā, kas atvieglos savstarpējo komandas komunikāciju. Izņēmums ir progammatūras arhitekts, kurš vienreizēji izveido programmas projektu, balstoties uz stāstiem no lietotāju puese, kas šajā gadījumā ir visi projektā iekļautie komandas dalībnieki.

|  |  |
| --- | --- |
| **Komandas dalībnieki** | **Loma** |
| Programmētājs | Veic atsevišķo programmas moduļu izstrādi |
| Projekta vadītājs | Organizē reviewus (pārskatus), šajā gadījumā tie ir formāli, lai novērtētu progresu. Projekta vadītājs ir cilvēks, kas venlaikus pārzin biznesa intereses. |
| Programmatūras arhitekts | Izstrādā kopējo ieskatu, kā programma strādās un |
| Testētājs | Šajā gadījumā testētājs ir arī galvenais testētājs, kurš komunicē ar projekta vadītāju, lai ievērotu produkta specifikāciju |

# Vispārīgs ieskats, kas tiks testēts

Programma tiks veidota pa moduļiem ar atsevišķu noteiktu funkcionalitāti katram modulim. Veiktajos testos galvenais būs pārliecināties, vai izpildās katrā fāzē nākušā klāt moduļa funkcionalitāte, vai jaunie moduļi integrējas esošajā kopējā sistēmā un tie nav ietekmējuši kopējās sitēmas funkcionalitāti.

Veicot testus, vēlajamajam rezultātam test casos vajadzētu atbilst lietojamības apraksta (user story) rezultātu prasībām. User story prasības veiktajos testos noteikti ir jāņem vērā . Testi tiktu veikti JIRA vadības programmā pa soļiem:

1. Tie tiktu izveidoti ar detalizētu veicamo soļu izklāstu un tiktu saglabāti
2. Tiem būtu iespējams ievadīt rezultātus un atbilstošus komentārus
3. Testu rezultātus būtu iespēajms uzskatāmi apskatīt

Kad tiktu nodrošināta atbilstoša programmas funkcionalitāte, būt jāveic dažāda veida nefunkcionālei testi (piem., slodzes, veiktspējas, drošuma u.c.) produkta veiktspējas uzlabošanai. Diezgan būtisks varētu būt slodzes tests, jo programmas lietotāja daudzums nav prognozējams.

## Pieņēmumi

1. OnMenu produkta izstrādei pēc testa plāna ir paredzēts ilgs laika posms, kas liek domāt, ka līdz produkta piegādei nekādi būtiski riski nevarētu rasties.

## Riski

Tomēr, ja ēdināšanas tirgū rastos vēsmas par līdzīga produkta parādīšanos, varētu rasties atsevišķi proejekta riski.

Zemāk, skat., iespējamos risku un to ietekmi – “smagumu” uz projekta īstenošanu un iespējamās riska radītās sekas, ar kurām būtu nepieciešams pēc tam tikt galā.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Risks | Nopietnība | Ietekme | Rīcības Plāns |
| 1 | Līdz ar laika termiņa iestāšanos produkta izstrādei, jo ir mainījies tirgus, rodas cilvēkresursu trūkums pie izstrādes, testēšanas, lai konkurētu ar tirgus spēlētājiem | Augsta | Projekta aizkavēšanās, testēšanas un izstrādes procesa piespiedu paātrināšana | Jāpieņem darbā papildus programmētājs un testētājs, kas ir papildus izmaksas |
| 2 | Kvalificēta darbaspēka (programmētāju, testētāju) trūkums | Augsta | Projekta aizkavēšanās, papildus izmaksas | Uznēmumam, lai gan vajag tikai šī projekta ietveros papildus “rokas”, nākas pieņemt papildus štata darbiniekus |
| 3 | Jaunie darbinieki (testētāji, programmētāji) nav iepazinušies ar produktu | Medium | Papildus laiks, kamēr jaunie darbinieki iepazīst produktu | Tiek organizētāš “ātrās” apmācības,; testēšanas metodologija ir jāpamaina uz Agile/Scrum |
| 4 |  |  |  |  |

# Testa Pieeja

Jauno risku izaicinājumu ietekmē projekts būt jāīsteno Agile/Scrum pieeju ar iknedēļās – 2 nedēļu atskaites pārbaudes posmiem. Ja radušies riski nav, tad produktu būtu bijis iespējams notestēt arī ar Exploratory testa pieeju. Taču apsaktīsim Agile/Scrum metodoloģijas variantu, kad katras iterācijas beigās tiek pārskatīti nepieciešamie veicamie testi komandai. To, kā tests plāns tiek pildīts, varēs redzēt projekta testa vadības rīkā JIRA.

Kļūdas (BUGU) pieteikumā noteikti jābūt iekļautai šādai informācijai:

* Enviroinment – kurā vidē, kāda produkta versijā tiak atrasta kļūda
* Steps to reproduce –kādi soļi ir jāveic, lai atkārtotu/pieredzētu kļūdu
* Expected result – kam bija jānotiek šo soļu rezultātā
* Actual result – kas notika
* Visual Proof (screenshots, video) – uzskates materiāls, vieglākai kļūdu uztveršanai

Severity/ priority – ko tas ietekmē, nozīmē un cik steidzami jālabo

*Galvenie veicamie testi:*

Komponenšu testēšanu, kurā pārbauda, vai programmas kods un moduļi strādā pareizi veiks izstrādātāji – programmētāji . Integrācijas testus, kuros pārbauda dažādu moduļu un sistēmu savstarpēju mijiedarbību un atbilstību specifikācijai, arī veiks programmētāji. Programmētaji veiks daudz struktūras testus jeb white-box testus.

Sistēmas testus, kuros pārbauda visas sistēmas vai produkta darbību, lai noteiktu atbilstību specifikācijām un vai produkts darbojas pareizi, veiks testētājs testa vidē. Funkcionālos testus un nefunkcionālo testus šajā testēšanas līmenī testētājs izmantos black-box tehniku. Web lapu testēšanā ieteicams izmantot statisko testēšanas tehniku, ko arī testa laikā izmantosim.

Acceptance testos tiks pārbaudīts, vai visas sistēma ir pabeigta un atbilst user story aprakstam. Tiks veikti alfa/beta testi . Tiks pārbaudīts, vai produkts ir gatavs nodošanai klientam saskaņā ar acceptance criteria. Par šo atbildēs Testu līderis.

## Testu Automatizācija

100% automatizētus komponšu testus uzrasktīs programmētāji un tos izpildīs CI/CD sistēma izstrādes un testa vidēs. Regresijas testi būs automatizēti gan komponenšu testēšanas līmenī, gan integrācijas testu līmenī – testus uzrakstīs programmētāji. Nefunkcionālso testus noautomatizēs testētājs testa vidē. Funkcionālajiem testiem automatizācijas nav paredzēta

# Testa Vides

Būs gan izstrādēs vide (to lietos programmētāji), gan testa vide (iespējams pat vairākas tetētāju testiem) un produkcijas vide (šeit tiks izveitots gala produkts, kas ir pieejams klientiem).

Lai būtu iespējams nodrošināt web pakalpojumus un datu glabāšanu datu bāzēs, būs nepieciešami serveri, datori un pat modēmi interneta sakaru nodrošināšanai.

Jānodrošina augsts drošības līmenis testa iekārtu, sistēmas programmatūras, datu un serveru aizsardzībai.

# Plānotie posma darbi / Piegādes

## Testēšanas grafiks

Sākotnējais testa gafiks norādīts uz aptuveno paredzamo izpildes laika posmu.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task Name** | **Start** | **Finish** | **Effort** | **Comments** |
| Test Planning | Oct 2020 | Nov 2020 |  |  |
| Review Requirements documents |  |  | 2 d |  |
| Create initial test estimates |  |  | 1 d |  |
| Staff and train new test resources |  |  |  |  |
| First deploy to QA test environment | April 2021 | May 2021 |  |  |
| Functional testing – Iteration 1 | April 2021 | May 2021 |  |  |
| Iteration 2 deploy to QA test environment | Juny 2021 | July 2021 |  |  |
| Functional testing – Iteration 2 | July 2021 | August 2021 |  |  |
| System testing | August 2021 | September 2021 |  |  |
| Regression testing | Oct 2020 | Oct 2021 |  |  |
| UAT | Oct 2021 | Nov 2021 |  |  |
| Resolution of final defects and final build testing | Nov 2021 | DEC 2021 | 1 week |  |
| Deploy to Staging environment | Nov 2021 | DEC 2021 | 1 week |  |
| Performance testing | Nov 2021 | DEC 2021 |  |  |
| Release to Production | DEC 2021 | DEC 2021 |  |  |

## Nodevumi

|  |  |  |
| --- | --- | --- |
| **Deliverable** | **For** | **Date / Milestone** |
| Test Plan | Project Manager; QA Director; Test Team | Nov 2020 |
| Traceability Matrix | Project Manager; QA Director |  |
| Test Results | Project Manager | All the time |
| Test Status report | QA Manager, QA Director | July 2021 |
|  |  |  |
| Metrics | All team members | During project-all the time |
|  |  |  |