

queri

Plan testiranja

Verzija 1.0

Istorija revizija

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Verzija** | **Opis** | **Autor** |
| 01.04.2023. | 1.0 | Prva verzija dokumenta. | Milan Vlaški |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Sadržaj

1. Uvod 4

1.1 Svrha 4

1.2 Obim 4

1.3 Ciljana publika 4

1.4 Terminologija i skraćenice dokumenta 4

1.5 References 4

2. Motivacija za testiranje 4

2.1 Pozadina 4

3. Elementi testiranja 4

4. Pristup testiranju 4

5. Isporuke 5

Plan testiranja

# Uvod

## Svrha

Cilj ovog *Plana testiranja* je da sabere sve bitne informacije vezane za upravljanje testiranjem softverskog alata *queri*. Plan testiranja obuhvata:

* Elemente koji će biti ciljani testovima
* Motivaciju za testiranje određenih oblasti
* Pristup testiranju
* Listu isporučivih elemenata testiranja projekta

## Obim

U sklopu projekta koristiće se jedinično i manuelno testiranje. Jedinični testovi pokriće *Model* dio aplikacije, dok će ostale funkcionalnosti, vidljive korisniku, biti ručno testirane od strane članova projektnog tima.

## Ciljana publika

Ovaj dokument namijenjen je kao artefakt koji daje uvid svima zainteresovanima u način na koji je vođeno testiranje aplikacije *queri*.

## Terminologija i skraćenice dokumenta

## References

* *Dokument softverske arhitekture*

# Motivacija za testiranje

Test driven development se pokazao kao alat koji donosi bolji dizajn, ponovnu iskoristivost i čitkost koda. Njime nam je omogućeno bezbjedno refaktorisanje koda i brza provjera rezultata, jer ne zahtijeva pokretanje i korištenje same aplikacije.

## Pozadina

U MVC arhitekturi softverski elementi se naslanjaju na *Model* kao izvor tačnih podataka, i funkcionalnosti vezanih za promjenu stanja. Takođe, bez *View* dijela, stanje i podaci u Modelu se ne mogu lako provjeriti. Potreban je veliki setup u svakom pokušaju testiranja neke funkcionalnosti. Da ne bi pribjegavali amaterskim tehnikama kao što su printanje podataka na standardni ulaz smatramo da je *Model* dio softverskog rješenja najbitnije testirati.

# Elementi testiranja

Funkcionalnost svake klase *Modela* biće pokrivena jediničnim testovima. Saradnja *Modela* i *View*a biće testiranja ručno.

# Pristup testiranju

Pristup testiranju vođen je u 3 poznata koraka *Test driven developmen*t-a: *Red, Green, Refactor.* Prije pisanja metode ili skupa funkcionalnosti, potrebno je napisati test koji u svakom slučaju pada. Ovim je definisan ulaz/izlaz funkcije. Onda je potrebno napisati minimalnu količinu koda da funkcionalnost prođe. Sve dok smo u prolaznom stanju, refaktorišemo rješenje i činimo ga elegantnijim i efikasnijim. Ponavljamo korake dok funkcionalnost ne bude dovršena. Ovim je prvo obezbjeđen rad glavnog scenarija funkcije, te onda onih ne toliko očiglednih slučajeva.

Manuelno testiranje će biti sprovođeno nad trenutno izvedenim funkcionalnostima, gdje će se testeri truditi da replikuju scenario u kojem je došlo do izuzetka ili greške. Korištenje debagera u sklopu radnog okruženja Eclipse je preporučeno.

# Isporuke

Za pregled zainteresovanim stranama biće dostupan sam source code testova, *Plan testiranja* i *Rezime ocjene testiranja.*