SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE

FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Automatizované testovanie (unit, integračné, End-to-EnD A Výkonnostné testy)

Dokumentácia k semestrálnemu Zadaniu Predmetu Architektúra Softvérových systémov 2022/2023

Obsah

[Zoznam skratiek a značiek 2](#_Toc121876911)

[Úvod 1](#_Toc121876912)

[1 Technická špecifikácia 2](#_Toc121876913)

[1.1 Informácie o kóde softvéru 2](#_Toc121876914)

[1.2 Testovacie nástroje použité v ukážkach 2](#_Toc121876915)

[2 Unit testy 3](#_Toc121876916)

[3 Integračné testy 4](#_Toc121876917)

[4 End-to-end testy 5](#_Toc121876918)

[5 Výkonnostné testy 6](#_Toc121876919)

[Zoznam použitej literatúry 7](#_Toc121876920)

Zoznam skratiek a značiek

**Úvod**

Dokument súži ako návod pre realizáciu praktických ukážok z prezentácie na tému Automatizované testovanie (unit, integračné, end-to-end a výkonnostné testy), ktorá bola odprezentovaná autormi v rámci semestrálneho zadania na predmete Architektúra softvérových systémov.

Kapitola Technická špecifikácia obsahuje informácie o kóde demonštračného softvéru a zoznam použitých testovacích nástrojov. V ďalších častiach je dokumentácia delená do kapitol podľa typu vykonaných testov.

1. Technická špecifikácia
   1. Informácie o kóde softvéru

Použitý programovací jazyk: Java

Použité frameworky:

Kód je v čase publikácie tohto dokumentu dostupný na:

<https://github.com/j-puszter/testovanie>

* 1. Testovacie nástroje použité v ukážkach

Unit testy:

Integračné testy:

End-to-end testy

Výkonnostné testy: Jmeter

1. Unit testy
2. Integračné testy

Cieľom integračných testov je overenie, či časti softvéru spolu fungujú. Tento typ testovania sa využíva hlavne pri väčších projektoch. Integračnými testami môžeme testovať napríklad endpointy aplikácie a porovnať očakávaný výstup.

3.1 Konfigurácia

@SpringBootTest – potrebný pre spustenie celej aplikácie, vytvorí ApplicationContext potrebný pre načítanie tried pomocou dependancy injection

@AutoConfigureMockMvc – Automatické nakonfigurovanie MockMvc



3.2 MockMvc

Hlavný vstupný bod pre komunikáciu s aplikáciou. Vďaka MockMvc môžeme volať requesty bez toho aby sa musel spúšťať server.



* 1. Testovanie

Endpointy voláme pomocou metódy perform v ktorej môžeme volať rôzne metódy a následne môžeme zadať čo očakávame, aký status by mal endpoint vrátiť, aké hlavičky by mal vrátiť, aké dáta má vrátiť a ďalšie parametre.

* + 1. Metódy

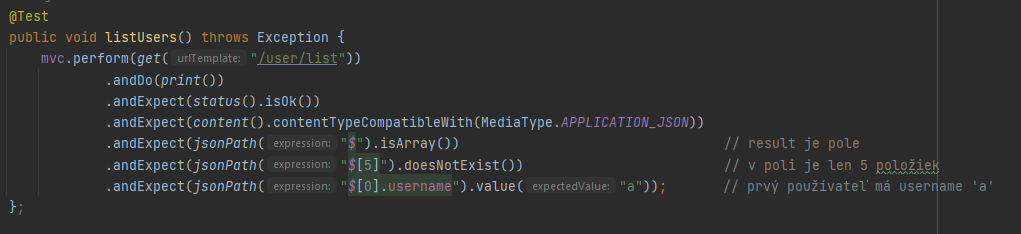
Metódy voláme pomocou triedy MockMvcRequestBuilder ktorý obsahuje statické metódy ako get a post na ktoré je možné následne zadať aký typ dát posielame a vložiť dané dáta.

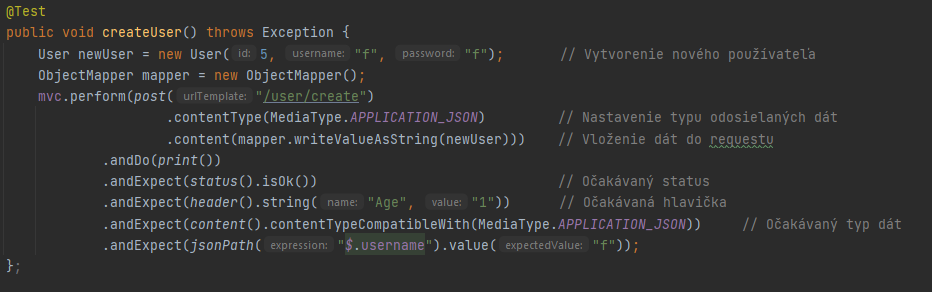
|  |  |
| --- | --- |
| GET |  |
| POST |  |

* + 1. Výstup

Pomocou triedy MockMvcResultHandler a jej statickej metódy print() test zobrazí všetky podrobnosti o requeste. Zároveň je možné porovnávať výsledky s očakávaným výsledkom pomocou triedy MockMvcResultMatchers a jej metód:

* status() – status odpovede
* content()
  + formu odpovede (typ dát, kódovanie...)
  + samotnú odpoveď - porovnanie základného stringu (string())
* header() – hlavičku odpovede
* jsonPath() – porovnanie konkrétnej hodnoty v json-e
  1. Testy





1. End-to-end testy
2. Výkonnostné testy

**Zoznam použitej literatúry**

**There are no sources in the current document.**