

Automatizované testy UK

Obsah

Abstrakt.....	4
SmartDriver – framework pro automatizaci testů	4
Dodavatel.....	4
Popis řešení.....	4
Artefakty.....	4
Licence – SmartDriver.....	5
Licence – kódy v codebase projektu.....	5
Návody a technická dokumentace.....	5
Infrastruktura	5
Návrh infrastruktury vycházející z PoC	5
Projektové řízení.....	7
Úkoly	7
Plánování.....	7
Typy úkolů.....	7
Testovací scénáře	8
Správa testovacích scénářů	8
Navrhované řešení.....	9
Návrh procesů pro automatizaci testů	9
Vydefinování seznamu scénářů pro testování.....	9
Proces zpracování jednotlivých scénářů.....	9
Kontrola výsledků běhu automatických testů	10
Zpracování nalezených chyb	10
Akceptace.....	10
Kvalita automatizačních skriptů.....	10
Vlastní akceptace.....	11
Odpovědnost a záruka.....	11
Lidé	11
Inženýr test automatizace.....	11
David Waller.....	11
Architekt, technický garant, kontakt za dodavatele.....	12
Tomáš Novák.....	12
Fakturace.....	12



Reference	12
Konzultační služby	12
Další možná spolupráce	13
Automatické testy na dalších platformách	13
Reporting	13
DataCreator	13



Abstrakt

Nabídka na spolupráci pro vývoj automatických testů pro moduly aplikace IS Věda pro UK. Nabídka je vytvořena na základě Proof of concept (dále "PoC"), které proběhlo jako prerekvizita. Během PoC byla provedena analýza a návrh řešení pro možný přístup k tvorbě automatických testů. Výstupní dokument k PoC byl vytvořen a sdílen na SharePointu UK. Tato nabídka vychází z navrženého řešení během PoC a dalším krokem je toto řešení rozšiřovat.

Cílem této aktivity je odhalovat chyby v aplikaci po nasazování nových verzí – typicky se jedná o regresní chyby, kde je změnou v aplikaci způsobeno chybné chování již existujících a dříve fungujících funkcionalit.

Tato nabídka je strukturována do jednotlivých okruhů pro větší přehlednost.

SmartDriver – framework pro automatizaci testů

Dodavatel

Jedná se o interní řešení vyvíjené firmou **SmarTest Automation**.

Popis řešení

Jedná se o framework (dále "FW"), který obaluje technologii Selenium WebDriver a přidává další funkcionality jako jsou logování, reportování, podporu pro další platformy, podporu pro Selenium Grid a paralelní spouštění testů. FW je navržen a používán pro testování webových aplikací z pohledu koncového uživatele na End to end (dále "E2E") scénářích – je používán ve společnostech **Sazka**, **Raiffeisenbank** a **Vodafone** pro uživatelské akceptační testy a E2E testování. Vývoj FW včetně implementace testů probíhá v jazyce Java. Toto je výhodou i nevýhodou zároveň – je třeba časová investice pro studium základů programování (základní úroveň stačí), nicméně je také možné FW jakkoliv rozšiřovat interně a přidávat specifické funkcionality pro potřeby testů (nejedná se o uzavřené prostředí a nejsme limitováni čekáním na to, než bude implementována funkcionality do jádra frameworku). FW podporuje i testování na dalších platformách (Android, iOS).

Artefakty

SmartDriver je šířen v podobě maven artefaktů. Jedná se o několik modulů, které se přidávají jako závislost do automatizačního projektu. Artefakty jsou šířeny přes dodavatelem spravovanou instanci Nexus repozitáře.



Licence – SmartDriver

SmartDriver bude poskytnut bez licenčního poplatku. Licence umožní použití pro UK, ale není přenositelná mimo UK.

Pro UK bude nasazena verze 2.0-beta.xxx. UK bude mít k dispozici všechny nové verze do doby první release verze (nejdříve začátek roku 2024). Po celou dobu bude k dispozici podpora na opravu chyb ze strany SmarTest Automation. V případě aktivní spolupráce bude pro UK zaručen update na novější verze po dobu celé spolupráce.

Licence – kódy v codebase projektu

Veškeré kódy, které vzniknou v codebase automatizačního projektu na straně UK budou vlastnictvím UK.

Návody a technická dokumentace

Budou k dispozici na sdíleném prostoru ze strany SmarTest Automation – Confluence.

Infrastruktura

V rámci PoC byla nadefinována základní podoba infrastruktury. V rámci další spolupráce by se naše společnost podílela na správě této infrastruktury – případných aktualizací, nastavování dalších exekučních serverů, další nastavení v rámci nově přidávaných požadavků, které budou vznikat v průběhu spolupráce atp.

Návrh infrastruktury vycházející z PoC

Infrastruktura bude postavena na technologii využívající Docker. V rámci tohoto PoC bude po schválení a doladění detailů s Janem Krajícem infrastruktura nastavena tak, aby bylo možná spouštět první testy.

UK poskytuje server, na kterém bude možné tento návrh realizovat. Infrastrukturu bude možné do budoucna rozšiřovat přidáním další výpočetní kapacity.

SmartDriver – Framework

- Java 11, Maven, Selenium, Chromedriver
- Moduly pro automatizaci webu
- K dispozici na stažení ze SmarTest Automation Nexus – konfigurace pro maven a přístupy do Nexus budou dodány

Git

- Správa a verzování kódu
- Bude využito řešení, která vybere UK

Chrome

- Prohlížeč, na kterém bude ze začátku testováno

Selenium Grid

- <https://www.selenium.dev/documentation/grid/>
- Standardní nástroj pro distribuci testů na různé instance dle požadavků testu
- Jednotlivé instance mohou ovládat různé prohlížeče na různých operačních systémech

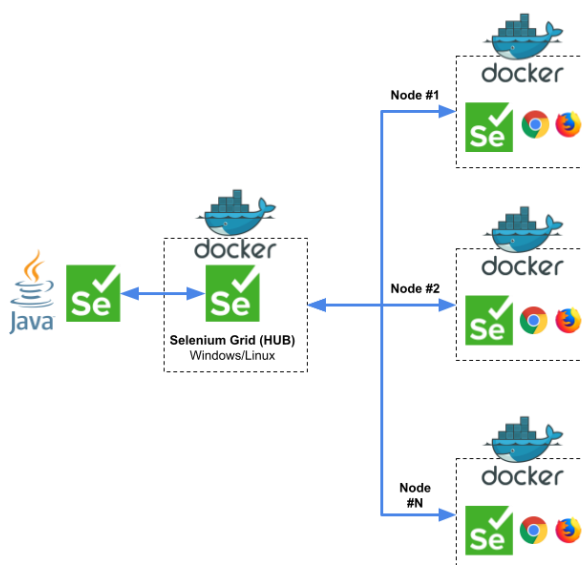
CI (Continuous integration)

- Automatizovaná exekuce testů na infrastruktuře
- Bude upřesněno na základě výběru UK

Docker

- Celá infrastruktura bude využívat Docker kontejnery
- Snadná škálovatelnost, udržovatelnost

Schéma:



Projektové řízení

Pro projektové řízení by mělo být využito již existující řešení – Redmine.

Úkoly

Pro projekt automatického testování bude vytvořen backlog, ze kterého se budou vybírat jednotlivé úkoly. Navrhovaným přístupem je použít kanban board, v kterém bude vidět současný stav projektu – na jakých úkolech se pracuje, které úkoly byly hotovy nebo které úkoly jsou blokovány. Tyto úkoly budou sloužit také k zaznamenávání odpracovaného času.

Plánování

Pro každý úkol je důležité, aby bylo proveden odhad, jak dlouho bude trvat jeho zpracování.

Typy úkolů

Příprava testovacího scénáře

- Pro přehled, jaké scénáře se mají automatizovat
- Sledování postupu pro přípravu jednotlivých scénářů
- Zpracován na straně UK
- Výstupem je test case – po jednotlivých krocích rozepsaný scénář s očekávanými výsledky
- Součástí je vydefinování potřebných rolí, případně testovacích dat
- Po vypracování testovacího scénáře proběhne kontrola ze strany vývojového týmu, zda je v této podobě scénář automatizovatelný a všechny kroky jsou srozumitelné

Implementace automatického testu podle scénáře

- Možné zařadit do zpracování po vytvoření testovacího scénáře
- Prvním krokem je manuální průchod scénářem

Vývoj na infrastruktuře

- Úpravy, implementace na infrastruktuře

Vývoj technické funkcionality pro podporu implementace automatických testů

- Úkoly, které usnadní implementaci automatických testů
- Například funkcionality, které zobecní často opakované úkony v systémech



Úprava skriptů

- V případě, že je do systému dodána funkcionality, která má dopad na již existující skripty, je potřeba tyto skripty upravit
- V případě, že testy selhávají chybou špatné robustnosti, je třeba je upravit k větší spolehlivosti

Podpora

- Příprava testovacích dat
- Příprava testovacích uživatelů a nastavování potřebných rolí

Testovací scénáře

Testovací scénáře jsou potřeba jako definice a reference implementace automatického testu. Podle předpisu testovacího scénáře se krok za krokem implementují automatické testy včetně očekávaných výsledků.

Správa testovacích scénářů

Při použití Redmine je možnost využít pluginů na správu testovacích scénářů. Problémem je, že Redmine nebyl navržen na tuto funkcionality a existující open source pluginy jsou velmi zastaralé a většinou bez podpory. Existují nějaké licencované varianty, ale u nich se musí platit licenční poplatky.

Možné postupy:

- Použít existující open source plugin
 - Riziko toho, že bez nějakých úprav nebude fungovat
- Použít placenou variantu
 - Náklady navíc
- Použití jiného nástroje
 - Například centrální GitLab – zatím není
- Definice v excel souborech na SharePointu
 - Snadno udržitelné v menších počtech scénářů
 - Ve větším měřítku nepoužitelné



Navrhované řešení

Začít s excel soubory a v mezičase vytipovat několik možných řešení, vyzkoušet je a vybrat to, které bude nejlépe splňovat potřeby projektu.

Návrh procesů pro automatizaci testů

V této kapitole je popsán návrh na vlastní proces přípravy testovacích scénářů a následné automatizace. Odhady jsou poskytnuty na základě dat nasbíraných během PoC fáze. Po nastartování procesu se budou časové odhady zpřesňovat na základě skutečného trvání jednotlivých kroků.

Vydefinování seznamu scénářů pro testování

Jako první krok je potřeba vydefinovat seznam scénářů pro automatizaci – na základě tohoto seznamu budou plánovány další kroky.

Proces zpracování jednotlivých scénářů

Použité zkratky:

- [UK] – Univerzita Karlova
- [SA] – SmarTest Automation

1. [UK] Příprava scénáře
 - a. Po jednotlivých krocích rozepsaný scénář s očekávanými výsledky
 - b. Definice potřebných dat a přístupů pro daný scénář
 - c. Očekávaná náročnost je 1–2 MD na scénář dle náročnosti
2. [UK] Validace testovacího scénáře (druhýma očima)
3. [SA] Vývojový team musí zvalidovat, že scénář je automatizovatelný a všechny kroky srozumitelné
4. [UK] Příprava testovacích dat a případné zajištění přístupů
5. [SA] Manuální průchod scénáře
6. [SA] Implementace automatického testu
 - a. Časová náročnost implementace scénáře o rozsahu scénáře “Vytvoření záznamu v OBD” je odhadována na 2 MD
 - b. Každý scénář bude implementován do vlastní větve v rámci použití verzovacího systému git

- c. Bude použito Git Feature Branch Workflow
- 7. [SA] Code review (implementační)
- 8. [UK] Akceptace
 - a. Validace, že je scénář správně zpracován – kroky odpovídají zadání
 - b. Ideálně formou code review na straně [UK]

Kontrola výsledků běhu automatických testů

Pro pravidelný běh testů je potřeba jejich výsledky kontrolovat – pokud budou testy spouštěny každý den, je potřeba jejich výsledky kontrolovat a validovat ty, které skončí nevalidním stavem. Průběžná exekuce nezávisle na dodávkách do prostředí je nutná ke stabilizaci testů, monitorování stavu prostředí a testovacích dat. Pokud by exekuce a údržba testů neprobíhaly kontinuálně, je zde velké riziko, že po nasazení nové verze aplikace a spuštění testů dojde k chybám na straně skriptů, které budou muset být nejdříve opraveny. To značně zpomalí získání informací o tom, jaký vliv mělo nasazení na chod aplikace.

V první fázi je nutná ruční kontrola výsledků. Při vydefinování a zavedení reportingu může být částečně automatizováno.

Zpracování nalezených chyb

V případě nalezených chyb je potřeba kontaktovat dodavatele s bug reportem. V ideální situaci by měl dodavatel dodat odhad, do kdy bude chyba opravena a test může být následně využit na validaci opravy od dodavatele.

Automatické testy by se v případě nalezení chyby **neměly měnit**, aby reflektovaly současný stav testované aplikace.

Akceptace

Formou týdenních callů – v rozsahu 30 min. Tyto cally by měly sloužit zároveň na plánování dalších kroků, postupu a validace postupu.

Kvalita automatizačních skriptů

Součástí procesu akceptace a zaručení kvality dodávaných kódů je i proces code review. Kódy prochází přes review ostatních členů teamu na odhalení případných problémů.



Vlastní akceptace

Nominovaná osoba na straně UK akceptuje implementované řešení - 1 test (1 úkol). V případě, že skript nesplňuje očekávání, je vrácen na dopracování. Akceptace může být součástí code review procesu a může být například pokryto pravidlem, že když jsou kódy zařazeny do test suites, tak je akceptováno.

Odpovědnost a záruka

Dodavatel je odpovědný za případné opravy chyb způsobených na jeho straně – například nesprávná validace výstupu, chybějící krok, atd. Dodavatel není odpovědný za úpravu již akceptovaných skriptů po nasazení nové verze testované aplikace, které upravují původní zadání. Tento scénář je řešen implementací definovaného ticketu na úpravu.

Lidé

Na projekt nabízíme 2 vývojáře s následujícími profily/rolemi a cenami. V případě zájmu UK o dodávku více zdrojů ze strany dodavatele je možné nabídku vývojářů rozšířit – je potřeba počítat s tím, že splnění požadavku bude trvat cca 1 měsíc.

Dále je počítáno s tím, že UK dodá svoje vlastní zdroje pro vývoj automatických testů – pro ně bude poskytnuto zaškolení na vyžádání.

Inženýr test automatizace

David Waller

- Student
- SmartDriver používá již několik let - na projektu pro společnost Sazka
- Spolupracoval na implementaci testovacích scénářů pro PoC
- K dispozici v rozsahu maximálně 0.5 FTE
- **Cena: 5000,-Kč/MD bez DPH**

Architekt, technický garant, kontakt za dodavatele

Tomáš Novák

- Projektové řízení
- Technická garance řešení a kvality kódu
- Code review
- Podpora pro vývoj
- Zpracování požadavků ze strany UK
- Návrh koncepčních řešení a celé architektury test automatizace
- Školení
- K dispozici v rozsahu max 2MD/měsíc
- **Cena: 8500,-Kč/MD bez DPH**

Fakturace

Fakturace na měsíční bázi na základě schválených timesheetů od UK v maximálním rozsahu specifikovaném v detailu u jednotlivých vývojářů.

Reference

SmarTest Automation by na základě uzavřené spolupráce chtěla mít možnost zveřejnit UK nebo ÚVT UK jako jednu z referencí na svém webu. Podmínky pro zveřejnění a použití reference by měly být nadefinovány ze strany UK.

Konzultační služby

Mimo předmět této nabídky nabízíme i další konzultační služby – definované vždy dle aktuální potřeby s odpovídajícím cenovým ohodnocením.

Služby:

- Validace procesu kvality dodávaného SW



- Školení v test automatizaci
- Audit existujícího procesu test automatizace
- Rozšíření test automatizace v rámci dalších oddělení

Další možná spolupráce

Prozatím zaparkovaná témata na základě domluvy s UK.

Automatické testy na dalších platformách

Například na mobilních zařízeních.

Reporting

Posbírání požadavků a návrh centrálního řešení reportingu test automatizace. Formou PoC.

DataCreator

Nástroj pro automatické generování testovacích dat.

