Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
Ilkovičova 2, 842 16 Bratislava 4

Tímový projekt

Webable

Dokumentácia k inžinierskemu dielu

Tím: stableFamily (tím č. 10)
Vedúci tímu: Ing. Jakub Šimko, PhD.
Členovia tímu: Bc. Michal Fabiš, Bc. Katarína Rafčíková, Bc. Daniela Sitárová, Bc. Maroš Vašš, Bc. Andrej Zaťko, Bc. Andrej Slaninka, Bc. Martin Žák

Ak. rok: 2018/2019

Obsah

Úvod

V rámci predmetu Tímový projekt sa tím stableFamily zaoberá vývojom webového prehliadača pre zrakovo postihnutých ľudí. V tomto dokumente sa nachádzajú podklady, ktoré opisujú nami vyvíjaný softvér, t.j. inžinierske dielo. V nasledujúcich častiach sa uvádzajú ciele, ohraničenia, globálne ciele softvéru na zimný a letný semester, celkový pohľad na systém v zmysle architektúry softvéru, diagramu komponentov, opis konkrétneho modulu, ktorý sme integrovali do nášho prehliadača.

Webový prehliadač pre nevidiacich Webable je projekt, ktorý vznikol počas minulého akademického roka ako aplikácia určená do súťaže Imagine Cup. Preto je náš tím v odlišnej situácii oproti iným tímom, nakoľko pracujeme na softvéri, ktorý už bol vyvíjaný a preto sa stretávame s aj s inými problémami ako ostatné tímy. Vzhľadom na to, že počas minuloročného vývoja sme nedbali na testovanie nášho softvéru, narazili sme na niekoľko problémov, ktoré nám bránia vytvárať automatizované testy. Ide o problém, ktorý je skôr technického charakteru ako toho, že náš kód nie je testovateľný. Počas všetkých 3 šprintov sme sa snažili riešiť tento problém, no zatiaľ sa nám ho nepodarilo odstrániť.

Ciele a ohraničenia

Vzhľadom na to, že sme na softvéri pracovali počas súťaže ImagineCup 2018/2019, nemuseli sme ho vyvíjať od začiatku. Avšak ani zďaleka nespĺňal to, čo by od neho naši potenciálni zákazníci očakávali. V rámci tímového projektu sme si preto stanovili nasledujúce ciele:

1. zabezpečiť to, aby náš prehliadač disponoval základnou funkcionalitou, na ktorú sú zrakovo postihnuti ľudí zvyknutí pri používaní súčasných moderných prehliadačov. Toto môžeme docieliť tak, že náš prehliadač budeme často testovať s potenciálnymi používateľmi a zapracovávať ich pripomienky
2. zabezpečiť to, aby celková funkcionalita prehliadača bola prístupná (aby zrakovo postihnutý používateľ mohol plnohodnotne využívať náš prehliadač)
3. integrovať a zlepšiť algoritmus modulu *Automatická analýza a korekcia kódu*, ktorý bol vyvíjaný samostatne počas súťaže ImagineCup
4. zlepšiť použiteľnosť modulu *Mapa webovej lokality* a premyslieť, akým spôsobom oboznámiť nového používateľa s tým, na čo presne mapa slúži a ako sa používa
5. mať aspoň 1 používateľa

Globálne ciele pre ZS/LS

Ciele a ohraničenia, ktoré sú uvedené vyššie, sme si stanovili na celý rok. Na konci zimného semestra by sme chceli mať prehliadač v stave MVP. To znamená, že celý zimný semester sa budeme zaoberať bodmi č. 1, 2, 3. Prioritou je však pre nás bod č. 3. A to preto, lebo je to niečo inovatívne, vďaka čomu bude mať náš prehliadač pridanú hodnotu, bude iný oproti ostatným prehliadačom a budeme môcť povedať, že máme MVP.

Totiž takouto funkcionalitou nedisponuje žiadny prehliadač a ani žiadne rozšírenie do prehliadača. Bodmi č. 1, 2, 4, 5 sa chceme zaoberať hlavne v letnom semestri.

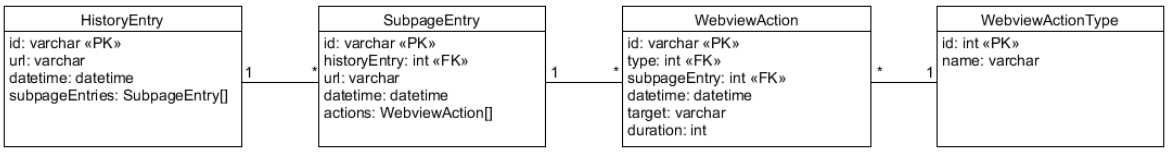
Celkový pohľad na systém

Náš prehliadač je riešený ako klientská aplikácia. Teda ide o spustiteľný súbor, ktorý si bude môcť používateľ stiahnuť z nášho webového sídla a po nainštalovaní hneď začať používať. Súčasťou našej aplikácie je aj čítačka obrazovky, ktorú my neimplementujeme, ale využívame existujúci softvér NVDA, ktorý je k našej aplikácii pribalený.

Primárne je prehliadač Webable programovaný v jazyku JavaScript, pretože využívame rámec Electron, ktorý používa NodeJS (javascript engine). Je niekoľko dôvodov, prečo sme sa rozhodli pre tento rámec. Prvým je ten, že zabezpečuje multiplatformovosť aplikácie a teda je jedno, či chce používateľ spustiť aplikáciu na operačnom systéme Windows, OSX alebo Linux. Ďalším je ten, že sme sa nechceli pri vývoji prehliadača zaoberať samotným spracovávaním HTML kódu, čo by bolo opätovné "*vyvíjanie kolesa*". Rámec Electron využíva Chromium - open-source webový prehliadač - ktorý za nás rieši elementárne veci, ktoré sú súčasťou každého webového prehliadača a preto sa môžeme sústrediť na vývoj našej vlastnej funkcionality.

Dátový model

V našom webovom prehliadači využívame 2 databázy. Prvou je dokumentovo orientovaná databáza PouchDB, v ktorej sú momentálne uložené len nastavenia prehliadača, ktoré si môže používateľ meniť. Druhou je relačná databáza SQLite, v ktorej sa ukladá história všetkých navštívených webových stránok, ktorú si vie používateľ zobraziť a navštíviť vybranú webovú stránku. Dátový model relačnej databázy je zobrazený na Obrázku č. 1.



Obrázok č.1 Dátový model nášho prehliadača

HistoryEntry

Do tabuľky sa vloží záznam vtedy, ak používateľ navštívi stránku zadaním url adresy do vstupného poľa alebo kliknutím na odkaz, ktorý používateľa presmeruje na inú doménu.

SubpageEntry

Do tabuľky sa vloží záznam vtedy, ak sa používateľ dostane na nejakú podstránku domény, ktorú práve navštívil, kliknutím na odkaz na stránke a nie zadaním url adresy do vstupného poľa.

WebviewAction

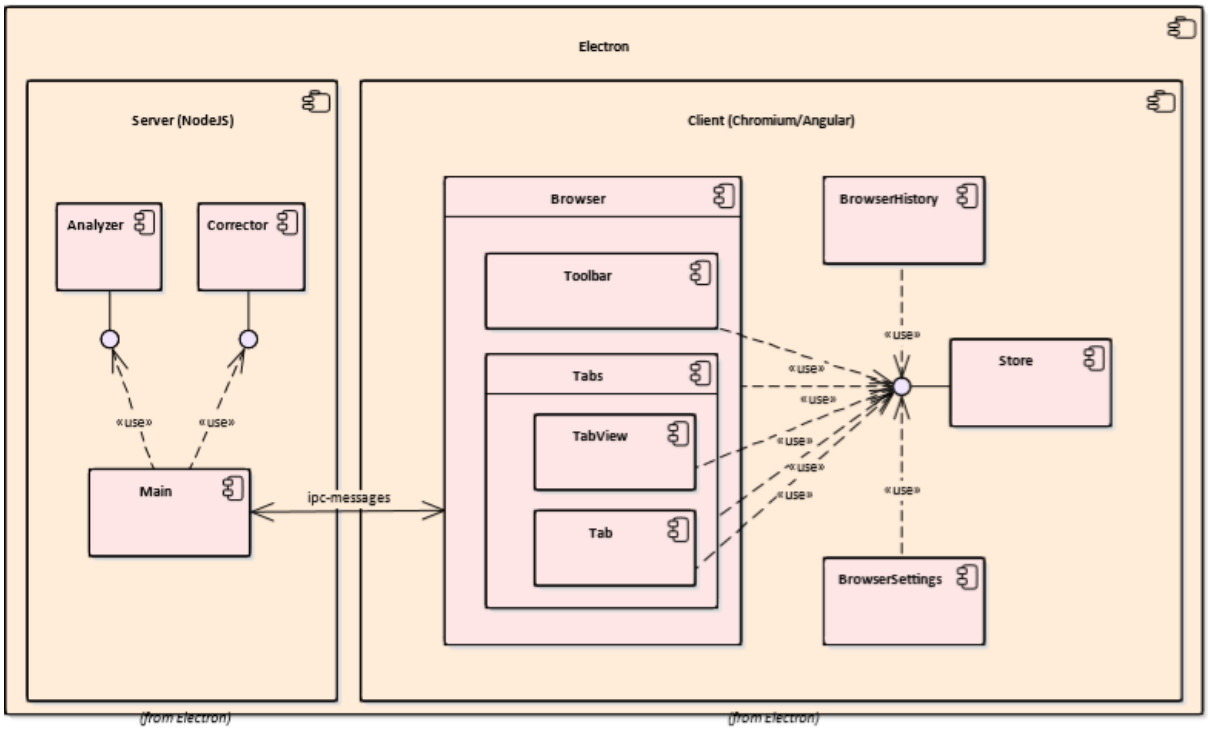
Do tabuľky sa vloží záznam vtedy, ak používateľ vykoná akciu na stránke typu *CLICK*, *SWIPE*, *SCROLL*, *HOVER*

WebviewActionType

Číselník, ktorý obsahuje 4 záznamy - *CLICK*, *SWIPE*, *SCROLL*, *HOVER*

Diagram komponentov

Architektúra prehliadača Webable je vizualizovaná na obrázku č. 2 pomocou diagramu komponentov.



Obrázok č.2 Diagram komponentov prehliadača Webable

Hoci ide o klientskú aplikáciu, je rozdelená na klientskú a serverovú časť.

Klient

Do klientskej časti patrí všetko, čo vidí používateľ. To znamená okno prehliadača s celou jeho funkcionalitou vrátane zobrazovania webových stránok. Z obrázku je možne vyčítať, že všetky komponenty využívajú rozhranie komponentu *Store*. Je to tak preto, lebo stav prehliadača je jeden veľký objekt, ktorý je týmto komponentom reprezentovaný. V prípade, že chcú ľubovoľné komponenty medzi sebou komunikovať, nekomunikujú priamo medzi sebou, ale cez komponent *Store* (ďalej centrálny objekt). A to tak, že sa vytvorí požiadavka nad centrálnym objektom, ktorá zmení jeho stav a tento objekt následne oboznámi príslušne komponenty o zmene svojho stavu. Architektúru softvéru máme takto navrhnutú preto, aby sme zabezpečili modulárnosť systému.

Server

Do serverovej časti patrí časť aplikácie, ktorá zabezpečuje jej spustenie, inicializáciu spojenia s databázou SQLite, komunikáciu s rôznymi modulmi a kontrolu toho, či existuje čítačka obrazovky na počítači. V prípade, že neexistuje, spýta sa používateľa, či ju chce nainštalovať. V opačnom prípade ju naša aplikácia automatický spustí.

Modul analýza a úprava zdrojového kódu

Momentálne náš prehliadač komunikuje len s modulom, ktorý analyzuje a upravuje kód. Ide o funkcionalitu na ktorej sa podieľala členka nášho tímu v rámci svojej bakalárskej práce. Preto analýzu, návrh a implementáciu v tomto dokumente nerozpracujeme ale uvedieme odkaz na bakalársku prácu, v ktorej sú všetky 3 časti podrobne rozpracované. V nasledujúcej podkapitole uvedieme len to, ako sme modul integrovali do nášho prehliadača.

Integrácia

Keďže sa tento modul od začiatku vyvíjal nezávisle od nášho prehliadača, na to, aby fungoval v našom prehliadači, ho bolo potrebné integrovať. Integráciu sme mohli spraviť buď na klientskej časti alebo na serverovej časti. My sme sa rozhodli vytvoriť komunikačné rozhranie s modulom na serverovej časti preto, lebo chceme, aby fungoval nezávisle od nášho prehliadača (klientskej časti aplikácie) a do budúcna by sme chceli z neho spraviť rozšírenie pre súčasné webové prehliadače.

Testovanie

Počas uplynulého akademického roka sme síce pracovali na vývoji prehliadača, ale pretože cieľom bolo vytvoriť softvér určený na súťaž v istom časovom období, príliš sme sa nezaoberali testovaním. Na začiatku boli vytvorené testovacie konfiguračné súbory, avšak postupom času sa stali neaktuálnymi a už viac nie sú kompatibilné s vytváraným softvérom. V súčasnosti nie je jednoduché zakomponovať do vyvíjaného softvéru testovací nastroj, pretože táto zmena si vyžaduje rozsiahle zásahy do existujúceho kódu.

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
Ilkovičova 2, 842 16 Bratislava 4

Tímový projekt

Webable

Dokumentácia k riadeniu

Tím: stableFamily (tím č. 10)
Vedúci tímu: Ing. Jakub Šimko, PhD.
Členovia tímu: Bc. Michal Fabiš, Bc. Katarína Rafčíková, Bc. Daniela Sitárová, Bc. Maroš Vašš, Bc. Andrej Zaťko, Bc. Andrej Slaninka, Bc. Martin Žák

Ak. rok: 2018/2019

Úvod

V rámci predmetu Tímový projekt sa tím stableFamily zaoberá tvorbou webového prehliadača pre nevidiacich s názvom Webable. V tomto dokumente sa nachádza podrobný opis spôsobu riadenia tímu v tomto projekte. V nasledujúcich častiach sa uvádza, aké zodpovednosti majú jednotliví členovia tímu, popis jednotlivých manažmentov, ktoré sú využívané, podrobný záznam z prvých štyroch šprintov, ktoré zatiaľ boli uskutočnené, globálna retrospektíva, motivačný dokument a jednotlivé metodiky, ktorými sa tím riadi.

Webový prehliadač pre nevidiacich Webable je projekt, ktorý vznikol počas minulého akademického roka, ako aplikácia určená do súťaže Imagine Cup. Preto je náš tím v odlišnej situácii oproti iným tímom, nakoľko pracujeme na softvéri, ktorý už bol vyvíjaný a preto sa stretávame s aj s inými problémami ako ostatné tímy. Táto situácia spoločne s nedostatkom skúseností s agilným prístupom spôsobila v prvých týždňoch, že sme pracovali neefektívne a darilo sa nám plniť iba malé množstvo úloh.

Role členov tímu a podiel práce

V rámci tímu vystupujú všetci členovia ako vlastníci produktu (product owner). Tiež sa všetci podieľajú na vývoji a okrem toho má každý ešte ďalšie špecifické úlohy, ktoré sa uvádzajú nižšie. Podiel práce jednotlivých členov tímu na častiach dokumentácie je uvedený v tabuľke 1.

Meno	**Časti dokumentácie, na ktorých sa daný člen podieľal**
Michal Fabiš	Sumarizácie šprintov, Modul analýza a úprava kódu
Katarína Rafčíková	Definition of Done, Celkový pohľad na systém
Daniela Sitárová	Aplikácie manažmentov
Andrej Slaninka	Definition of Ready, Globálne ciele pre ZS/LS
Maroš Vašš	Globálna retrospektíva, Úvod
Andrej Zaťko	Pravidlá pre verziovanie, Ciele a ohraničenia
Martin Žák	Komunikácia, Úvod, dokumentácia k inžinierskemu dielu

Tabuľka 1: Podiel práce jednotlivých členov tímu na častiach dokumentácie

Michal Fabiš

Michal je zodpovedný za administráciu nástroja JIRA, ktorý je používaný na podporu správy úloh pri agilnom vývoji. Správa nástroja JIRA obnáša počiatočnú konfiguráciu, to znamená vytvorenie kont pre jednotlivých členov tímu, prispôsobenie rozhrania, aby vyhovovalo potrebám tímu a zadanie prvých úloh, a v priebehu práce na projekte vytváranie jednotlivých úloh a zapisovanie.

Katarína Rafčíková

Kataríninou úlohou je správa verzií kódu a vývoj pre Mac platformu. Je zodpovedná za úložisko kódu na Githube, s čím bola spojená počiatočná inicializácia a vytvorenie spôsobu manažovania jednotlivých verzií - vytváranie vetiev, nastavenie integrácie jednotlivých modulov a vykonávanie prehliadok kódu.

Daniela Sitárová

Daniela vykonáva funkciu zapisovateľa na jednotlivých tímových stretnutiach. Je zodpovedná za celkovú dokumentáciu - za vytváranie jednotlivých metodík, zaznamenávanie priebehu šprintov a retrospektív, dokumentáciu riadenia v tíme a dokumentáciu inžinierskeho diela. Je tiež jednou

z členov tímu, ktorí sa venovali vývoju webového prehliadača pre nevidiacich v rámci svojich bakalárskych prác.

Andrej Slaninka

Andrejovou úlohou v tíme je správa webového sídla. Navrhol a vytvoril webovú stránku a stará sa o to, aby bola vždy aktuálna. Tiež ma na starosti celý manažment komunikácie, t.j. spravuje nástroj na komunikáciu v tíme (Slack) a v prípade potreby vykonáva rôzne zmeny. Vývoju prehliadača sa venoval už v predošlom akademickom roku, preto sa tiež zaoberá analýzou riešení z pohľadu prístupnosti a je zodpovedný za navrhovanie nových funkcionalít.

Maroš Vašš

Maroš je zodpovedný za úlohy súvisiace s obchodnými stránkami nášho produktu. Zaoberá sa možnosťami propagácie, predaja, šírenia nášho webového prehliadača a komunikáciou s mentormi a potenciálnymi partnermi. Tiež sa podieľa na analýzach potrebných na vytváranie novej funkcionality v rámci softvéru.

Andrej Zaťko

Andrejovou úlohou je dohliadať na vytváranie testov a spôsob testovania. Táto úloha obnáša nakonfigurovanie testovacieho prostredia, vytvorenie šablón a zaškolenie ostatných členov tímu.

Martin Žák

Martin vystupuje v tíme v úlohe SCRUM mastera a preto má na starosti manažment tímových stretnutí. Vedie jednotlivé stretnutia a určuje ich priebeh. Počas šprintov kontroluje, či si ostatní členovia tímu plnia svoje povinnosti a ako si ich plnia. Zároveň tiež pracoval na projekte v rámci svojej bakalárskej práce, preto pozná celkovú architektúru systému a dokáže poradiť ostatným, menej skúseným členom tímu.

Aplikácie manažmentov

Pri riadení projektu je kľúčové, akým spôsobom prebiehajú jednotlivé procesy. Je dôležité dodržiavať určité pravidlá, ktoré si tím stanoví. V nasledujúcej časti sa podrobne zameriavame na tie, ktoré zatiaľ aplikujeme pri riadení nášho projektu. V rámci tímu máme roly rozdelené tak, aby jedna osoba bola zodpovedná za daný typ manažmentu. V rámci tímu sme si pri plnení úloh, ktoré spadali pod konkrétny typ manažmentu, vzájomne pomáhali.

Manažment úloh

Riadenie celého procesu vytvárania, hodnotenia a vykonávania úloh je najdôležitejšou a najobsiahlejšou časťou riadenia. Jednotliví členovia tímu predstavujú zároveň aj vlastníkov produktu, preto sa tiež podieľajú na vytváraní úloh.

Vytváranie úloh

Úlohy sú vytvárané priebežne, počas trvania jednotlivých šprintov. Všetky novovytvorené úlohy sa zhromažďujú v tzv. **backlogu**. Do backlogu môže pridať úlohy každý člen tímu v akomkoľvek čase a aj vedúci, ktorý predstavuje tiež vlastníka produktu a reprezentuje ho na tímových stretnutiach. Pre potreby nášho projektu sme definovali viaceré typy úloh:

- ***Používateľská úloha*** **(User story)** - úlohy tohto typu sú tie, ktoré sa bežne využívané pri agilnom prístupe vývoja softvéru, jedná sa o scenár, ktorý by mal po vykonaní úlohy prehliadač splniť.
- ***Analytická úloha*** **(Analytic story)** - úlohy, v ktorých je potrebné vykonať analýzu nejakého problému alebo oblasti, výsledkom je vytvorenie nových používateľských príbehov
- ***Úloha slúžiaca na tímový manažment*** **(Team management story)** - úlohy súvisiace s riadením procesov v tíme, napr. vytváranie metodík a podobne.

Jednotlivé úlohy sú zároveň združované do väčších celkov, tzv. **epics**. Tie sú definované nasledovne:

- *UX vylepšenia*
- *Výpočtové vylepšenia*
- *Nová funkcionalita*
- *Inteligentný webový prehliadač*
- *Infraštruktúra a refaktoring*
- *Chyby*

Každá úloha by mala spĺňať pravidlá, ktoré tím vytvoril vo vlastnej definícii pripravenosti úlohy (Definition of Ready), viď časť Metodiky - Definition of Ready. Pre jednotlivé typy úloh sú vytvorené odlišné definície.

Prioritizovanie úloh (Backlog grooming)

Počas jednotlivých tímových stretnutí je potrebné úlohy usporiadať do vhodného poradia. Poradie úloh určuje ich priorita, ktorá sa môže v priebehu jednotlivých šprintov meniť. Úlohy s najvyššou prioritou sa nachádzajú vo vrchnej časti backlogu.

Vyberanie úloh do šprintu

Do šprintu sa vyberajú úlohy s najvyššou prioritou, to znamená tie, ktoré sú na vrchu backlogu. Náročnosť jednotlivých úloh je vyjadrená pomocou tzv. story pointov. Analytické úlohy a úlohy slúžiace na tímový manažment sú hodnotené nulovým počtom story pointov, pretože neprinášajú žiadnu pridanú hodnotu pre zákazníka.

Ohodnotenie používateľských úloh určujú členovia tímu pomocou plánovacieho pokeru. Pri tejto metóde sa všetci členovia tímu musia zhodnúť na počte bodov, ktorým danú úlohu ohodnotia. Pokiaľ sa nezhodnú, diskutujú o svojich názoroch a potom opäť zvolia iné bodové ohodnotenie. Tento proces sa opakuje, kým sa nezhodnú. Na základe počtu story pointov vie tím odhadnúť, koľko úloh môže zaradiť do ďalšieho šprintu. Počet úspešne splnených story pointov za jeden šprint reprezentuje výkonnosť tímu.

Vykonávanie úloh v šprinte

Po výbere úloh do šprintu sa rozdeľujú medzi jednotlivých členov tímu. Ak je to potrebné, vytvoria sa k úlohám podúlohy. Každá úloha je priradená jednému členovi tímu, ktorý je zodpovedný za jej vykonanie. K úlohe môže byť priradený aj spolupracovník, ktorý sa na nej bude podieľať a tiež sú k nej priradení členovia tímu, ktorí budú zodpovední za prehliadku kódu.

Počas šprintu sa môže úloha nachádzať v týchto stavoch:

- *Urobiť* (To do)
- *Prebieha práca* (In progress)
- *Pripravené na prehliadku* (Ready to review)
- *Vykonáva sa prehliadka* (In review)
- *Pripravené na akceptáciu* (Ready to Done)
- *Hotovo* (Done)

Na to, aby mohla byť úloha presunutá do stavu *Hotovo*, musí spĺňať pravidlá definície hotovej úlohy (Definition of Done), viď časť Metodiky - Definition of Done.

Manažment komunikácie

Komunikácia v tíme je zabezpečená viacerými spôsobmi. Najviac sa využíva komunikačný nástroj Slack, zápisy v nástroji JIRA, Github a osobné stretnutia.

Slack je rozdelený na viacero kanálov, kde každý slúži na iný účel. Kanály sú nasledovné:

- *general* - slúžiaci na komunikáciu s vedúcim a oficiálne správy, ktoré sa týkajú všetkých,
- *standups* - kanál, do ktorého sa píšu on-line reporty raz za týždeň, aby jednotliví členovia vedeli, na čom robia ostatní, podrobné pravidlá sa nachádzajú v časti Metodiky - Pravidlá komunikácie,
- *privatechat* - komunikácia medzi členmi tímu,
- *jira-notifications* - upozornenia z nástroja JIRA,
- *perry-space* - komunikácia ohľadom biznisu a mentoringu v akcelerátore Perry Space.

V nástroji JIRA je možné vidieť progres jednotlivých členov tímu a zároveň je možné k úlohám pridávať komentáre, ak je to potrebné. Nastroj slúži predovšetkým na sledovanie stavu šprintu a komunikáciu výhradne k úlohám.

Na Githube komunikujú jednotliví členovia tímu pri vykonávaní prehliadok kódu, vo forme komentovania kódu. Predovšetkým za účelom pochopenia, na čo kód slúži.

Stand-upy sú súčasťou spoločných stretnutí. Je tokrátke zhrnutie na začiatku alebo na konci tímového stretnutia podľa toho, ako sa tím dohodne. Prestný postup je zahrnutý v časti Metodiky - Pravidlá komunikácie.

E-mailová komunikácia je určená predovšetkým na oficiálnu komunikáciu za tím. Najčastejšie sa využíva na komunikáciu s Perry Space, mentormi a garantom predmetu tímový projekt.

Manažment verzií a prehliadok kódu

Pri agilnom spôsobe vývoja softvéru je potrebné dôkladne spravovať jednotlivé verzie softvéru. Nakoľko na projekte pracuje viacero ľudí, všetci musia dodržiavať pravidlá, aby tím pracoval efektívne a správne. V našom tíme je používaný nástroj GIthub, ktorý slúži tiež na riadenie prehliadok kódu. Pravidlá na vytváranie verzií a spôsob vykonávania prehliadok kódu (Code reviews) sú popísané v časti Metodiky - Pravidlá pre verzionovanie.

Manažment dokumentácie

Dobrá dokumentácia je základom celého projektu. Na každom stretnutí je vytváraný podrobný zápis o priebehu, ktorý má vopred definovanú štruktúru. Obsahuje informácie o tom, kedy a kde sa stretnutie konalo, kto sa ho zúčastnil a čo bolo jeho obsahom. Ak ide o stretnutie, na ktorom sa ukončuje šprint, súčasťou zápisu je retrospektíva skladajúca sa z troch častí:

- V čom pokračovať?
- S čím prestať?
- S čím začať?

Pokiaľ sa v priebehu šprintov objaví nový proces, ktorý potrebuje mať určité pravidlá, je vytvorená metodika. Ďalšou formou dokumentácie sú zápisy v nástroji JIRA.

Súčasťou je aj dokumentácia inžinierskeho diela a technická dokumentácia softvéru. V minulosti nebola funkcionalita opisovaná, preto je náročnejšie vytvoriť komplexnú technickú dokumentáciu. Pri vytváraní funkcionality sú nové moduly popisované podľa navrhnutej štruktúry. Moduly vzniknuté pred začiatkom práce na tímovom projekte by mali byť priebežne popisované a doplnené do technickej dokumentácie.

Manažment tímových stretnutí

Riadenie tímového stretnutia by malo mať svoju štruktúru, aby na stretnutí nevznikali zbytočné prestoje a priebeh stretnutia bol efektívny. Riadenie tímových stretnutí má na starosti SCRUM master. Jeho priebeh bol vždy zaznamenaný v písomnej podobe osobou, ktorá má na starosti zapisovanie poznámok zo stretnutí. Počas týždňa vždy vytvoril na zdieľanom úložisku (Google Drive) dokument, do ktorého mohol ktokoľvek z tímu počas týždňa spisovať pripomienky, ktoré chcel na najbližšom stretnutí prediskutovať. Tieto pripomienky potom SCRUM master zahrnul do agendy najbližšieho tímového stretnutia.

Tímové stretnutia mali 2 typy priebehu:

1. V prípade, že sa na tímovom stretnutí uzatváral šprint a otváral nový, priebeh stretnutia vyzeral nasledovne:
    - *Koniec šprintu* - spolu s vedúcim tímového projektu (product owner) sme skontrolovali, v akom stave sú úlohy v našom nástroji manažmentu úloh (JIRA). Ak sa nachádzali v stave *Ready to Done*, ich splnenie sa potvrdilo tým, že daná funkcionalita spĺňala pravidlá definície hotovej úlohy (Definition of Done), viď časť Metodiky - Definition of Done. Potom sa mohli presunúť do stavu *Done*.
    - *Retrospektíva* - Retrospektíva k všeobecnému tímovému fungovaniu. Jednotlivý členovia sa postupne vyjadrovali k otázkam *v čom pokračovať, s čím začať, s čím prestať*. Ak vznikli otázky mimo týchto tém, zahrnuli sa do sekcie diskusia, ktorá nasledovala po retrospektíve
    - *Diskusia* - diskutovali sa otázky, ktoré vznikli buď počas týždňa od jednotlivých členov tímu alebo z retrospektívy

- *Backlog grooming* - celý tím prioritizoval a ohodnocoval jednotlivé úlohy, ktoré sa nachádzali v backlogu
- *Začiatok šprintu* - na základe diskusie tímu, kto koľko úloh zvládne a z predošlých skúsenosti šprintov, sa vybralo N najprioritnejších úloh z backlogu a zaradili sa do aktívneho šprintu

2. V prípade, že tímové stretnutie sa konalo uprostred trvania šprintu, priebeh stretnutia vyzeral nasledovne:
   - *Standup* - ich priebeh mal presne takú istú štruktúru, akú majú online standup-y, ktoré sú popísané v časti Metodiky - Pravidlá komunikácie.
   - *Diskusia* - diskutovali sa otázky, ktoré vznikli počas týždňa od jednotlivých členov tímu
   - *Backlog grooming* - celý tím prioritizoval a ohodnocoval jednotlivé úlohy, ktoré by mohli byť súčasťou nasledujúceho šprintu

Biznis manažment

Zviditeľniť Webable vo svete mimo tímový projekt vyžaduje úsilie navyše. Keďže skúsenosti tímu s biznisom sú zatiaľ na nízkej úrovni, rozhodli sme sa zapojiť do podnikateľskej akadémie. Účasťou na workshopoch a konzultáciami s mentormi sme identifikovali oblasti, na ktoré sa musíme ako tím viac zamerať:

- Analýza trhu
- Analýza potrieb potenciálnych zákazníkov
- Definovanie MVP
- Hľadanie vhodného biznis modelu

V aktuálnej fáze analyzujeme rôzne zdroje s cieľom rozšíriť si znalosti. Zaujímavé články sú zdieľané v spoločných dokumentoch na zdieľanom úložisku (Google drive), do ktorých môže pridávať obsah každý člen tímu. Pre pridanie nového článku je dôležité:

- Stručne popísať o čom je článok/skopírovať jeho najzaujímavejšiu časť
- URL odkaz na článok

Nové poznatky konzultujeme na konci šprintu v podnikateľskej akadémii, kde dostaneme spätnú odozvu a definujeme si biznis ciele na nasledujúci šprint. Na stretnutiach a konzultáciách s mentormi sa účastnia vždy najmenej dvaja členovia tímu, aby sa predišlo subjektivite.

Manažment testovania

Testovanie je dôležitou súčasťou vývoja softvéru, preto sme sa mu začali venovať už od prvého šprintu. Na problémy sme však narazili už pri konfigurovaní a výbere testovacieho frameworku. Vzniknuté problémy sa nám doteraz nepodarilo vyriešiť, preto sme sa manažmentu testovania ešte patrične nevenovali. V ďalších krokoch bude odstránenie tohto problému našou prioritou.

Sumarizácie šprintov

V tejto kapitole sa nachádza prehľad úloh, ktoré boli zaradené do jednotlivých šprintov. Ak sa nepodarilo úlohu splniť, je pri nej uvedené, prečo sa to stalo.

Prvý šprint - Alveola

Prvý šprint slúžil tímu na to, aby sa zoznámil s agilným spôsobom vývoja softvéru. Počas tohto šprintu boli nastavované rozličné pravidlá a postupy, ako má tím fungovať tak, aby bol efektívny. Zároveň sa členovia tímu, ktorí nemali predošlú skúsenosť s vyvíjaným softvérom oboznamovali so štruktúrou, architektúrou a fungovaním nášho webového prehliadača. Celkový počet story pointov zaradených do šprintu bol 11, avšak nepodarilo sa splniť ani jeden. Dôvodom nesplnenia bola nedostatočná vedomosť členov o tom, ako funguje softvér a najmä príliš prísne stanovená definícia dokončenej úlohy. V definícii bolo jednou z podmienok, že nový kód musí byť otestovaný, avšak počas tohto šprintu sa nepodarilo nakonfigurovať nástroje na testovanie, z ktorého dôvodu sa ani jedna úloha nedostala do stavu *Done*.

Úlohy

Názov úlohy: Skryť tutoriál

Popis: V prehliadači sa zobrazuje tutoriál pre používateľa. Je potrebné tento tutoriál odstrániť.
Počet story pointov: 1
Splnené: Nie
Dôvod nesplnenia: Príliš prísna definícia ukončenej úlohy a zlé rozloženie času, nestihla sa vykonať prehliadka kódu

Názov úlohy: Fix chyby: Pri znovu pripojení počítača k internetu sa nedá načítať predtým zadaná stránka

Popis: Postup k zreprodukovaniu:

1. Odpojiť počítať od internetu
2. V url bare napísať "sme.sk" a zadať enter
3. Pripojiť počítať k internetu
4. Prejsť na url bar a zadať enter

Akceptačné kritériá:

- Po kroku číslo 4. sa stránka v url bare načíta.

Počet story pointov: 2
Splnené: Nie
Dôvod nesplnenia: Nedostatočná orientácia v kóde.

Názov úlohy: Ako používateľ pri výpadku internetového pripojenia, chcem aby namiesto bielej stránky bola zobrazená primeraná chybová hláška, aby som bol informovaný

Popis: Postup k zreprodukovaniu:

1. Odpojiť počítač od internetu
2. V url bare napísať "sme.sk" a zadať enter

Akceptačné kritériá:

- Zobrazí sa error page, ktorá informuje používateľa o konkrétnom stave. (Buď problém u klienta alebo na serveri)
- Po zobrazení error page na určitej karte chcem, aby som užívateľ mohol navštíviť akúkoľvek webovú stránku
- Nech je error aktuálny po každom obnovení stránky

Počet story pointov: 3
Splnené: Nie
Dôvod nesplnenia: Príliš prísna definícia ukončenej úlohy a zlé rozloženie času, nestihla sa vykonať prehliadka kódu.

Názov úlohy: Ako používateľ po zadaní kľúčových slov, chcem byť presmerovaný na výsledky vyhľadávania googlom, aby som pohodlnejšie našiel to, čo hľadám

Popis: Postup k zreprodukovaniu:

1. V url bare napísať "test test" a zadať enter

Obsah error okna: ERROR LANGUAGES

"\ " Message: undefined Error: [object Object]\""

Akceptačné kritériá:

- pokiaľ používateľ zadá kľúčové slová do URL baru v prehliadači, spustí sa vyhľadávanie cez search engine, ktorý má používateľ vybratý v nastaveniach prehliadača
- Automaticky vyhľadáva v predvolenom vyhľadávacom nástroji.

Počet story pointov: 1
Splnené: Nie

Dôvod nesplnenia: Príliš prísna definícia ukončenej úlohy a zlé rozloženie času, nestihla sa vykonať prehliadka kódu

Názov úlohy: Fix chyby: Pri zatvorení sitemapy má TAB prehliadača stále názov "SITEMAP"

Popis: Postup k zreprodukovaniu:

1. Otvoriť sitemapu(klávesová skratka CTRL + M)
2. Zatvoriť sitemapu(klávesová skratka CTRL + M)

Akceptačné kritériá:

- Názov aktívneho TABU prislúchať otvorenej stránke.

Počet story pointov: 1
Splnené: Nie
Dôvod nesplnenia: Príliš prísna definícia ukončenej úlohy a zlé rozloženie času, lebo nebola vykonaná prehliadka kódu.

Názov úlohy: Uskutočnenie pilotného testovania

Popis: Ak by to bolo možné, je potrebné uskutočniť pilotné testovanie podľa vopred navrhnutého testovacieho scenára.
Počet story pointov: 0
Splnené: Nie
Dôvod nesplnenia: Neboli splnené základné podmienky umožňujúce vykonanie testovania.

Druhý šprint - Biceps

V tomto šprinte tím prispôsobil svoje rozhodnutia podľa toho, aké chyby boli urobené v prvom šprinte. Prispôsobili sa podmienky definície dokončenej úlohy tak, aby bolo možné úlohy označiť ako hotové, aj keď nebolo možné vykonať testovanie. Tiež boli aktualizované pravidlá pre vykonávanie prehliadok kódu.

Náplňou tohto šprintu bolo dokončenie úloh z predošlého šprintu a pokračovanie vo vylepšovaní použiteľnosti prehliadača. Všetky úlohy z predošlého šprintu okrem zobrazovania chybovej hlášky sa podarilo ukončiť. Úloha na vykonanie pilotného testovania nebola zaradená, pretože sa tím rozhodol, že ju zaradí až vtedy, keď bude reálne uskutočniteľná. Táto úloha nebola dokončená, pretože sa nepodarilo zmeny pridať do aktuálnej verzie. Celkový počet story pointov zaradených do tohto šprintu bol 21. Podarilo sa splniť 15.

Úlohy

Okrem úloh prenesených z predošlého šprintu boli pridané aj ďalšie úlohy.

Názov úlohy: Zistiť ako funguje webpack a popísať jeho jednotlivé časti

Popis: Analytická úloha zameraná na lepšie pochopenie fungovania prehliadača.
Počet story pointov: 5
Splnené: Áno

Názov úlohy: Identifikovať a vyriešiť problém, ktorý nás brzdí vo vývoji prehliadača

Popis: Aktuálny stav:

- pri zmene vetvy po zadaní npm install sa nedotiahnutý dependencies. Musím vymazať priečinky *dist* a *node_modules* a nanovo spustiť *npm install*, aby mi veci fungovali (ale nefunguje to vždy)

Očakávaný stav

- po zmene vetvy po zadaní *npm install* sa nainštalujú všetky dependencies

Počet story pointov: 5
Splnené: Áno

Názov úlohy: Zistiť, ktoré balíčky nepoužívame/používame

Popis: Analytická úloha zameraná na lepšie pochopenie fungovania prehliadača.
Počet story pointov: 3
Splnené: Nie
Dôvod nesplnenia: Nedostatočná komunikácia medzi členmi tímu.

Tretí šprint - Clavicula

V tomto šprinte tím zapracoval do procesu riadenia poznatky získané v predošlom šprinte, týkajúce sa najmä prehliadok kódu. Vďaka úpravám môže tím pracovať efektívnejšie.

Počas šprintu sa tím zaoberal najmä ďalším vylepšovaním použiteľnosti prehliadača, integráciou modulu na analýzu a úpravu kódu a analytickými úlohami. Celkový počet zaradených story pointov bol 17, podarilo sa splniť 9.

Úlohy

Do tohto šprintu boli prenesené aj dve úlohy z predchádzajúceho, ktoré sa podarilo splniť. Okrem nich boli pridané aj ďalšie úlohy, na ktorých tím pracoval.

Názov úlohy: Zistiť, ako nakonfigurovať vybraný framework na testovanie tak, aby sme mohli vytvárať unit testy

Popis: Zistiť, nakoľko treba upraviť konfiguráciu vo webpacku tak, aby fungovalo testovanie
Počet story pointov: 0
Splnené: Nie
Dôvod nesplnenia: Zložitý problém, na ktorý sa zatiaľ nepodarilo nájsť riešenie.

Názov úlohy: Fix chyby: Pri navštívení akejkoľvek stránky sa daná stránka načíta dvakrát

Popis: Opis reprodukovania:

- Stránka ktorú chcem načítať, sa reloadne po jej načítaní. Napríklad navštívim stránku www.sme.sk. Prehliadač načíta stránku http://www.sme.sk a potom načíta https//www.sme.sk

Počet story pointov: 3
Splnené: Áno

Názov úlohy: Modul analýza a úprava kódu: Nájdi lepší prístup textovej analýzy aký zatiaľ používame

Popis: Analytická úloha, slúžiaca na podporu vytvárania novej funkcionality
Počet story pointov: 0
Splnené: Áno

Názov úlohy: Ako používateľ, chcem aby prehliadač opravoval formuláre

Popis: Aktuálny stav:

- Existujúci modul zatiaľ funguje mimo nášho prehliadača

Akceptačné kritériá:

- Vymyslená infraštruktúra v prehliadači, aby sa do budúcna ľahšie pridávali moduly

Počet story pointov: 8
Splnené: Nie
Dôvod nesplnenia: Nedodržanie termínu na odovzdanie kódu na prehliadku.

Štvrtý šprint - Dendrit

Skúsenosti z predchádzajúcich šprintov sa odzrkadlili vo štvrtom šprinte, kde sa tímu (okrem úlohy za 0 bodov, pri ktorej sa s nedokončením počítalo) podarilo splniť všetky stanovené úlohy. Pri plánovaní šprintu sa dbalo na dodržanie metodiky Definition of Ready a jasne stanovené zodpovednosti prispievali k celkovej efektivite tímu. Z celkových 17 story pointov tím dodal všetky.

Úlohy

V šprinte sa tím zameral na úlohy, potrebné na vykonanie interného testovania. Z predchádzajúceho šprintu sa preniesli dve úlohy a bolo dodaných ďalších šesť úloh.

Názov úlohy: Ako používateľ, chcem aby prehliadač opravoval formuláre

Popis: Aktuálny stav:

- Existujúci modul zatiaľ funguje mimo nášho prehliadača

Akceptačné kritériá:

- Vymyslená infraštruktúra v prehliadači, aby sa do budúcna ľahšie pridávali moduly

Možné problémy:

- Andrej tomu venoval minulý semester dosť času, čiže to nie je triviálny problém
- Gro práce bude spočívať v integrácii existujúceho modulu
- Navrhnúť spôsob integrácie, musí sa to dobre premysliet
- Nevieme vybrať celý DOM a dať ho späť

Možnosti riešenia:

- manipulovať s celým DOM-om (na to sme zatiaľ neprišli, ako to robiť)
- vypočítať si kde nastali zmeny a len tie zmeny upraviť v prehliadači (vymyslieť interface, prostredníctvom ktoré sa bude opravovať existujúce DOM štruktúra)

Počet story pointov: 8
Splnené: Áno

Názov úlohy: Zistiť, ako nakonfigurovať vybraný framework na testovanie tak, aby sme mohli vytvárať unit testy

Popis: Zistiť, nakoľko treba upraviť konfiguráciu vo webpacku tak, aby fungovalo testovanie
Počet story pointov: 0
Splnené: Nie
Dôvod nesplnenia: Zložitý problém, na ktorý sa zatiaľ nepodarilo nájsť riešenie.

Názov úlohy: Odkazy s target=_blank otvarat na novej karte

Popis: Opis reprodukovania:

- Na facebook.com kliknut na odkaz "terms"

Aktuálny stav:

- Momentálne sa takéto odkazy vôbec neotvoria.

Očakávaný stav:

- Otvorí sa stranka na novej karte.
- Pozri dokumentaciu

Počet story pointov: 3
Splnené: Áno

Názov úlohy: Pridat "s" do homepage http://www.bing.com'

Popis: Pravdepodobne treba spraviť túto zmenu aby si browser aj pri navštívení stranky bing.com doťahoval HTTPS

export const initialState: State =
{ homepage:'http://www.bing.com', onStartUp:ONSTARTUP_OPTIONS[0],
newTab:NEWTAB_OPTIONS[1], searchEngine:SEARCH_ENGINE_OPTIONS[0], }
Počet story pointov: 1
Splnené: Áno

Názov úlohy: V url bare sa nie je možné pohybovať šípkami doprava doľava medzi písmenami

Popis: Možné dôvody problému:

- nejaké eventy inputu sú prekonané
- nechtiac sa zahadzujú eventy vytvorené stlačením šipkami

Počet story pointov: 3

Splnené: Áno

Názov úlohy: Ako používateľ chcem aby som sa po načítaní chybovej stránky automaticky dostal na zobrazenú chybu, aby som nemusel prechádzať cez všetky ostatné elementy.
Popis: Pri chybovej stránke nespraví prehliadač automatický focus na chybovú hlášku.

Možné riešenia:

- dať arial-live atribút html tagu

Počet story pointov: 2
Splnené: Áno

Názov úlohy: Používať prehliadač celý deň

Popis:

- Na základe používania nášho prehliadača by mali vzniknúť úlohy v jire, ktoré sa v nasledovných šprintoch budú riešiť
- písať problémy do kolaboratívneho dokumentu

Počet story pointov: 0
Splnené: Áno

Názov úlohy: Spraviť dokumentáciu a odovzdať ju do AIS-u

Popis:
Počet story pointov: 0
Splnené: Áno

Globálna retrospektíva

Po ukončení každého šprintu tím v retrospektíve zhodnotil, aký bol priebeh šprintu. V tejto časti dokumentu sú opísané poznatky, ktoré boli nadobudnuté počas prvých štyroch šprintov.

Úvod projektu sa niesol v duchu nastavovania pravidiel a vylepšovania jednotlivých procesov v rámci riadenia. Základom sú dobre definované kritériá, ktoré by mala spĺňať každá úloha, či už pri vytváraní alebo pri rozhodovaní, kedy je hotová. Na začiatku boli tieto kritériá stanovené príliš prísne, takže ich nebolo možné splniť. Ukázalo sa, že správne plánovanie je mimoriadne dôležité.

Počas prvých šprintov sme zle odhadli náročnosť jednotlivých úloh, čo sa odrazilo na počte získaných story pointov za jeden šprint. Zároveň sme prišli na to, že je potrebné stanoviť naozaj podrobné pravidlá na jednotlivé procesy. Najviac sme sa zaoberali spôsobom vykonávania prehliadok kódu. Počas druhého šprintu totiž došlo k problémom spôsobeným práve nedostatočne definovaným pravidlami prehliadok kódu.

Po retrospektíve v treťom šprinte sme začali ku každej úlohe explicitne zaznamenávať kto nesie zodpovednosť za danú úlohu, kto na úlohe spolupracuje a kto robí prehliadku kódu. Ďalšou zmenou bolo zaznamenávanie aktuálneho stavu úlohy. Zhodli sme sa, že je užitočné postupne zapisovať progres na danej úlohe, aby ostatní členovia tímu mali prehľad o tom, v akom stave je úloha. Taktiež sme si stanovili interný termín na dokončenie úlohy, aby vznikol čas na prehliadky kódu a na riešenie prípadných nedostatkov.

Po troch šprintoch sa nám podarilo zlepšiť jednotlivé procesy a vykonať úlohy tak, aby sa softvér podarilo dostať do použiteľného stavu a mohli sme začať pracovať na príprave užívateľského testovania, ktoré je mimoriadne dôležité pre ďalší vývoj.

Štvrtý šprint bol teda venovaný príprave na používateľské testovanie. Z retrospektívy po tomto šprinte sme zistili, že chceme pre väčšiu prehľadnosť pri plánovaní šprintu nazývať úlohy stručnejšie a vecnejšie. Prišli sme tiež na to, že spoločná práca je pre nás dôležitá a pomáha nám v efektivite, preto do nej chceme investovať ešte viac času. Taktiež sme sa zhodli na tom, že používaním prehliadača na bežnú prácu počas dňa nám pomáha rýchlejšie odhaliť prípadné problémy a preto sme začali prehliadač viac využívať.

V nasledujúcom období chceme naďalej zlepšovať našu spoluprácu a riadenie v tímovom projekte.

Metodiky

V tejto časti sú spísané jednotlivé metodiky v takej podobe, ako sú používané v rámci práce na projekte. Využitie pri konkrétnych spôsoboch manažmentov je uvádzané v časti Aplikácie manažmentov.

Definition of Ready

User story

- musí byť nezávislá od iných stories
- musí sa dať vytvoriť test
- musí mať nejakú pridanú hodnotu pre koncového používateľa
- musia byť spísané akceptačné kritériá
- musí byť napísaná ako user story (AKO….,CHCEM…,ABY…)
- musí byť zrozumiteľná pre všetkých členov tímu (aj pre produktového vlastníka)
- je spísaný podrobný opis (používateľský scenár, eventuálne nakreslené obrazovky)
- všetci členovia tímu sa musia vedieť zhodnúť na konkrétnom počte story pointov
- každá story musí byť menšia/rovná 13 story points

Analytická story

- musí byť nezávislá od iných stories
- musí mať nejakú pridanú hodnotu
- musí byť zrozumiteľná pre všetkých členov tímu (v čom budem robiť prieskum?)
- musia byť jasne sformulované konkrétne otázky, na ktoré má dať analýza odpoveď
- bude mať 0 story points, keďže analytická úloha nemá žiadnu pridanú hodnotu pre zákazníka

Error story

- chybovosť vyjadrená zlomkom (1/z akého počtu skúšani sa stalo)
- musí byť spísané ako sa dá error zreprodukovať
- vždy, keď sa dá, treba mať obrázok
- očakávaný stav - ako chcem, aby to fungovalo bez chyby
- musí byť nezávislá od iných stories
- každá story musí byť menšia/rovná 13 story points?????
- všetci členovia tímu sa musia vedieť zhodnúť na konkrétnom počte story pointov

Definition of Done

Tento dokument obsahuje DOD pre:

- Definitioin of Done pre user story
- Definition of Done pre analytickú story
- Definition of Done pre error story
- Definition of Done pre šprint
- Definition of Done pre release

Definition of Done pre user story

- Musí byť implementovaná
- Kód musí byť patrične zdokumentovaný (podľa našich pravidiel písania prehľadného kódu)
- Dokumentácia je zaktualizovaná podľa pridanej funkcionality
- Implementácia musí byť pokrytá testami
- Musia zbehnúť všetky testy bez chyby
- Lokálny build prejde bez chýb
- Implementované riešenie a testy su zrevidované aspoň jedným členom z tímu (pri pull requeste)
- Vetva, v ktorej sa vyvíjala nová funkcionalita, musí byť zmergovaná do príslušnej vetvy šprintu
- Po zintegrovaní vetvy danej user story s vetvou sprint musí build prejsť bez chyby (remote)
- Musia byť splnené akceptačné kritériá

Definition of Done pre analytickú story

- Musí byť vytvorený výstupný dokument, ktorý bude obsahovať časti
    - Meno autora dokumentu
    - Zadanie témy, ktorá sa má analyzovať
    - stručný opis postupu riešenia
    - výsledok analýzy (odpovede na položené otázky)
    - V prípade, že je výsledkom analýzy odporučenie zaviesť novú technológiu, potom musí byť táto technológia do podrobna charakterizovaná
- Z dokumentu musí byť jednoznačne jasné, či sa dá problém riešiť a ak áno, tak ako.
- Dokument musí byť uložený na zdielanom úložisku dokumentov vo formáte markdown.
- Dokument musí byť zrevidovaný aspoň jedným členom z tímu
- Z výstupného dokumentu musí byť jasné, ako sa výsledky dajú uplatniť pri implementovaní ďalšej funkcionality

Definition of Done pre error story

- Chyba musí spĺňať očakávaný stav ktorý bol zapísaný v Error story.
- Chyba musí byť opravená pre každý aktuálne podporovaný systém.
- Musia zbehnúť všetky testy bez chyby
- Musia byť napísané testy ktoré pokryjú okrajové prípady
- Lokálny build musí zbehnúť bez chyby
- Implementované riešenie a testy su zrevidované aspoň jedným členom z tímu (pri pull requeste)
- Vetva, v ktorej sa opravovala chyba, musí byť zmergovaná do príslušnej vetvy šprintu
- Po zintegrovaní vetvy danej error story s vetvou sprint musí build prejsť bez chyby (remote)

Definition of Done pre šprint

- Ked sú všetky stories zo šprintu v stave done
- Vetva šprintu bude zmergovaná do vetvy dev

Definition of Done pre release

- Keď je projekt vybuildovaný bez chyby.
- Projekt spĺňa všetky jednotkové testy a funkčné testy sú zelené.
- Všetky akceptačné kritéria sú splnené.
- Product owner a tím odsúhlasili, že môže byť produkt vydaný.
- Používateľská príručka je doplnená o všetky zmeny.

Pravidlá komunikácia

Tento dokument opisuje, ako medzi sebou komunikujeme v tíme.

Standups

Stadups realizujeme keď spoločne pracujeme na projekte(predovšetkým v pondelok). Standups ako súčasť spoločných stretnutí - Ide o krátke zhrnutie na začiatku alebo na konci tímového stretnutia za účelom oboznámiť sa na čom kto pracoval/plánuje pracovať. Z pravidla účastní členovia stoja.

Okrem toho sme zaviedli aj on-line standups prostredníctvom Slacku. Podstatou je, že každý člen tímu napíše do kanála určeného špeciálne na standups, čo urobil a čo plánuje robiť. Tento zápis by každý člen tímu mal urobiť v piatok poobede tak, aby bol celý standup kompletný v sobotu ráno.

Správa na Slack-u by mala mať nasledovnú štruktúru:

UROBIL SOM:

- prvá vec
- druhá vec
- ...

PLÁNUJEM:

- prvá vec
- druhá vec
- ...

MÁM PROBLÉM S: (nemusí byť, iba ak má daný člen tímu nejaký problém)

- prvý problém
- druhý problém
- …

V príspevkoch je dobré označovať ľudí, s ktorými ste pracovali na daných úlohách, prípadne pri problémoch označovať konkrétnych ľudí, ktorí by mohli pomôcť problém vyriešiť.

Pravidlá pre verzionovanie

Commity

- Písané v jazyku angličtina
- Ak je vetva user story - US
    - commity sa budú vytvárať na vetvách user stories nasledovne T{Číslo tasku}: {Popis} (príklad: na vetve US53 bude nasledovný commit -> T35: Fix title on sitemap open/close...)
    - text commitu by mal obsahovať čo najviac kľúčových slov (dať do neho názvy všetkých modulov, ktorých sa týka zmena)
- Ak je vetva analytic story - AS
    - commity sa budú vytvárať na vetvách analytic stories nasledovne T{Číslo tasku}: {Popis} (príklad: na vetve AS53 bude nasledovný commit -> T35: Create list of packages, which are used in Webable and describe them...)
- Commit message by mala obsahovať čo najviac kľúčových slov (názvy všetkých modulov, ktorých sa týka zmena a pod.)

Vetvy

- master
    - hlavná vetva, ktorá je vždy funkčná
    - je možne jú meniť len cez pull requesty
    - kód v tejto vetve je nasadený v produkcii
    - build prebieha bez chyby
    - všetky testy sú úspešné
- dev
    - vetva z mastra
- sprint
    - vetva z devu
    - merguje sa vždy na konci šprintu
- ID user story z Jiry (príklad názvu: US30 - UserStory ID 30)
    - vetva zo sprintu
    - developer, ktorý bude počas šprintu pracovať na konkrétnej user story, si vytvorí vetvu s názvom {ID user story}. V tejto vetve bude pracovať dovtedy, pokým nebude spĺňať body Definition of Done. Po dosiahnutí DOD môže vytvoriť Pull request na vetvu pre aktuálny šprint.
    -

Pull requesty

;Každý task v Jire musí mať prideleného človeka zodpovedného za review, ktorý merguje pull request po splnení podmienok. Človek zodpovedný za review nemôže byť tá istá osoba ako tá, ktorá pracovala na tasku.

Kód pred podaním požiadavky na pull request musí byť:

- otestovaný so sprint vetvou (lokálne u seba sa spraví merge sprint vetvy do vetvy user story, ktorá je dokončená a tým sa zistí či user story bude fungovať so sprint vetvou)
- primerane komentovaný (pre účely dokumentácie).

Minimálne požiadavky pre metódy:

- krátke popísanie každej metódy
- opísanie parametrov
- návratu metód
- ďalšie vhodné informácie

Príklad:

```
/**
  * Normalizes the url according to the conventions if the user does not specify all of its parts and also handles requests for special Webable url
  * @param  {string}  path url we want to normalize and find out if it is not special webable one
  * @returns  {{ label: string, url: string }} label is what user sees and url the real address
  * @memberof  BrowserComponent
  */
```

Podmienky pre pull requesty do vetiev, počet členov tímu, ktorý musí schváliť PR:

- sprint
    - minimálne 2 členovia tímu
- dev
    - minimálne 3 členovia tímu
- master
    - všetci členovia tímu

Code review

- Komentáre pri code review budú v angličtine so snahou písať ich čo najmenej ofenzívne s cieľom niečo sa dozvedieť alebo poradiť.

- Pri code review je povinný reviewer otestovať nové zmeny lokálne, vybuildovaním prehliadača z vetvy, ktorá sa bude mergovat. Nestačí len “pozrieť kód”.
- Po napísaní pripomienok, ktoré treba zapracovať, reviewer v jire presunie úlohu zo stavu *In Review* do stavu *To Do*

Štandardné fungovanie prehliadača

Predtým, ako je vytvorený pull request musí byť skontrolované, či nová funkcionalita neovplyvní štandardné fungovanie prehliadača. To znamená, že musia byť vyskúšané nasledovné scenáre:

- dá sa otvoriť viacero tabov bez toho, aby prehliadač začal sekať
- je možné ľubovoľne sa prepínať medzi tabmi a navštevovať na nich stránky
- taby sa dajú otvárať a zatvárať v náhodnom poradí
- po zadaní ľubovoľného textu do vyhľadávania sa zobrazia korektné výsledky
- po zadaní URL sa zobrazí požadovaná stránka
- po prepísaní URL sa zobrazí požadovaná stránka
- po stlačení tlačidla Reload sa daná stránka znova načíta