Slovenská technická univerzita v Bratislave Fakulta informatiky a informačných technológií

Tím 25 - Bystander Effect

3D-UML Improved

Projektová dokumentácia - riadenie

Členovia tímu
Bc. Štefan Motko
Bc. Ronald Demeter
Bc. Marcel Furucz
Bc. Patrik Sčensný
Bc. Martin Dieška
Bc. Peter Psota
Bc. Martin Gembec

Vedúci
doc. Ing. Ivan Polášek, PhD.

Akademický rok: 2017/2018

Obsah

Úvod

Tento dokument predstavuje dokumentáciu riadenia projektu, ktorý je vytváraný v rámci predmetu Tímový projekt v akademickom roku 2017/2018. Dokument zahŕňa roly členov tímu, podiel ich práce na dokumentácii projektu, sumarizácie šprintov, použitých metodík a globálnu retrospektívu pre zimný semester.

Náš tím sa volá *Bystander Effect* a téma s ktorou pracujeme je 3D-UML. Cieľom projektu je vytvorenie webového nástroja na vytváranie 3D sekvenčných diagramov, ktorý dovolí používateľovi vytvárať sekvenčný diagram s viacerými vrstvami. Interakcia s diagramom bude podobná interakcii v známych komerčných nástrojoch ako napr.: Enterprise Architect.

Prvá kapitola obsahuje prehľad rolí jednotlivých členov tímu a ich zodpovednosti na projekte. Takisto tu je tabuľka zobrazujúca podiel práce na dokumentácií riadenia projektu.

V druhej kapitole je popísaná aplikácia jednotlivých manažmentov v projekte. V tejto časti je opis činností potrebných pre riadenie projektu.

Tretia kapitola dokumentu slúži ako sumarizáciu jednotlivých šprintov. V sekcií šprintu sú vidieť úlohy ktoré boli, resp. neboli ukončené a burndown grafy.

Štvrtá kapitola dokumentu obsahuje zoznam a stručný opis metodík, ktorými sme sa riadili pri práci na projekte.

V piatej kapitole dokumentu je uvedená globálna retrospektíva pre zimný semester. Táto časť zahŕňa zoznam činností, ktoré boli vyhodnotené za pozitívne, zoznam činností, ktorým by sme sa mali v budúcnosti vyhnúť.

1 Role členov tímu a podiel práce

1.1 Role členov tímu

Meno člena tímu	Zodpovednosť v tíme
Štefan Motko	Projektový manažér
Ronald Demeter	Hlavný vývojár
Marcel Furucz	Scrum a komunikácia
Patrik Sčensný	Testovanie a kvalita
Peter Psota	Grafický odborník
Martin Gembec	Administrácia databáz a dokumentácia
Martin Dieška	Vývoj serverovej strany

Tabuľka 1: Role členov tímu

1.2 Pridelené zodpovednosti členov tímu

Štefan Motko

- Prezentácia projektu
- Doménový expert
- Návrh systémových komponentov

Ronald Demeter

- Dohliadať na vývojový tím
- Rozhodovať o zdrojoch
- Dodávať kód, ktorý je dobre otestovaný a bezchybný
- Napísať čistý, štruktúrovaný a dobre zdokumentovaný kód

Marcel Furucz

- Udržiavať Scrum a Šprinty
- Spravovať tímovú komunikáciu

Patrik Sčensný

- Sledovať písanie testov a testovania
- Dohliadať nad kvalitou systému
- Vytvárať, štylizovať a spravovať tímovú webstránku
- Spravovať tímový server

Peter Psota

- Spravovať vizualizácie diagramov
- Spravovanie používateľského zážitku

Martin Gembec

- Vytváranie zápisníc zo stretnutí
- Vytváranie a spravovanie dokumentácie
- Správa databázy

Martin Dieška

- Zabezpečiť funkčnú serverovú stranu aplikácie.
- Poskytnúť front-endu všetky potrebné služby pre získanie a zapisovanie dát.
- Zabezpečiť komunikáciu medzi databázou a serverom a medzi serverom a klientom.

1.3 Podiel práce

Časť dokumentácie	Vypracovali
Úvod	Patrik Sčensný
Role členov tímu	relevantní členovia tímu
Aplikácie manažmentov	Ronald Demeter, Patrik Sčensný
Sumarizácie šprintov	Martin Gembec
Používané metodiky	relevantní členovia tímu
Globálne retrospektíva	Martin Dieška

Tabuľka 2: Podiel práce na dokumentácií riadenia projektu

2 Aplikácie manažmentov

2.1 Manažment dokumentácie

Každé tímové stretnutie vzniká zápisnica, ktorú píše každý týždeň zodpovedný člen tímu. Táto zápisnica obsahuje opísaný priebeh stretnutia a taktiež zoznam úloh, ktoré je potrebné spraviť. Na túto zápisnicu bola v úvode práce na tímovom projekte vytvorená šablóna aby každá mala jednotný tvar. Zápisnica sa publikuje každý týždeň aj na tímovú stránku, aby bola dostupná každému členovi tímu.

2.2 Manažment komunikácie

Komunikácia mimo stretnutí prebieha cez nástroje Slack a Trello. Slack je rozdelený na viaceré kanály(anglicky: channels), ktoré slúžia na konkretizovanie diskusie a dva automatizované kanály ktoré slúžia na logovanie informácií a udalostí z Trella a githubu. Trello umožňuje komentovať jednotlivé používateľské príbehy, reprezentované kartami(anglicky: cards), čo slúži na konkretizovanie diskusie.

Počas stretnutí je komunikácia voľná. Na konci šprintu sa vytvorí retrospektíva kde každý člen vyjadrí svoje pozitíva a negatíva, ktoré zaznamenal počas šprintu.

2.3 Manažment plánovania

Na manažovanie plánovania sa používa Trello. V Trelle máme zadefinovaných šesť stĺpcov:

- Epics - obsahuje príbehy, ktoré sa sťahujú na viaceré šprinty
- Backlog - obsahuje používateľské príbehy sťahujúce na šprint
- Development - obsahuje používateľské úlohy, na ktorých sa momentálne pracuje. Vo vnútri týchto príbehov sú definované úlohy, ktoré sa musia splniť aby príbeh bol splnený
- Testing - Príbehy, na ktorých prebiehajú prehliadky kódu a testy
- Pending - Príbehy, ktoré majú vytvorený pull request a čakajú na zlúčenie do dev konára
- Done - Dokončené používateľské príbehy

Tím pracuje v šprintoch dĺžky dvoch týždňov. Na začiatku šprintu zadefinujeme úlohy, ktoré by na konci šprintu mali byť hotové.

2.4 Manažment verziovania

Každý člen tímu je osobne zodpovedný za dodržiavanie pravidiel verziovania kódu, ktoré sú zdokumentované v Metodike verziovania.

2.5 Manažment testovania

Každý člen tíme je zodpovedný za požiadanie napísania testu. Člen tímu zodpovedný za písanie testu bude návrh testu konzultovať so žiadateľom návrhu testu. Testy sú rozdelené na backend a frontend. Testy sú písané v nástroji Mocha. Pre detaily si preštudujte metodiku testovania.

2.6 Manažment kvality písania zdrojového kódu

Každý člen tímu, ktorý píše zdrojový kód je zodpovedný za to, aby bol čitateľný, funkčný a dodržiaval metodiku písania zdrojového kódu.

Všetok napísaný kód podlieha prehliadke zdrojového kódu, ktorý sa vytvorí počas stretnutia. Na prehliadku zdrojového kódu dohliadajú dvaja alebo

viacerí členovia tímu. Postup je opísaný v metodike prehliadke zdrojového kódu.

3 Sumarizácie šprintov

3.1 Predšprintová príprava

Úlohou predšprintovej prípravy bolo definovanie technológií, ktoré by mohli byť využité v projekte, ako aj oboznámenie sa s technológiami použitými v minuloročnom riešení tohto projektu.

V prvom týždni boli rozdelené úlohy jednotlivým členom tímu a vytvorený prvotný návrh plagátu. Každý člen tímu dostal za úlohu preštudovať si SCRUM a prezrieť si dokumentácie starších projektov, webových stránok a zápisníc.

Témou stretnutia v druhom týždni bolo bližšie oboznámenie sa s projektom prezentáciou minuloročných zadaní. Prezentáciu viedol Štefan Motko. Následne boli uvedené možnosti, ako by sme zmenou niektorých technológií mohli zefektívniť prezentované riešenia. Nakoniec sme spojazdnili nástroj na vytváranie a manažovanie user stories a taskov (Trello), vytvorili sme repozitár v GitHube, formálne sme ukončili prvý šprint a priradili sme si prvé úlohy. Do ďalšieho stretnutia sme sa rozhodli preskúmať konektivitu medzi C# a MongoDB a vygenerovať JSONy pre účely testovania. Každý člen tímu dostal za úlohu naštudovať si odprezentovaný metamodel.

- ukončené:

- príprava prezentácie - Štefan Motko
- prezretie dokumentácii, stránok a zápisníc starších projektov - všetci
- príprava plagátu - Marcel Furucz
- preštudovanie scrumu - všetci
- rozdelenie rol medzi členov tímu - všetci

3.2 Šprint 1 - Layer

V prvom šprinte sme začali s počiatočnou implementáciou služieb a prvkov grafického rozhrania.

Prvý týždeň prvého šprintu bol venovaný prezentácií metamodelu. Prezentáciu viedol Štefan Motko spolu s vedúcim tímu. Po odprezentovaní prebehla diskusia o možnom prvotnom modeli JSONu. Následne sme zostavili počiatočnú schému JSONu. Dôležitou úlohou do ďalšieho týždňa sa stalo vytvorenie webovej stránky a jej nasadenie na server. Ešte počas stretnutia bola celým tímom vybraná vizuálna schéma stránky.

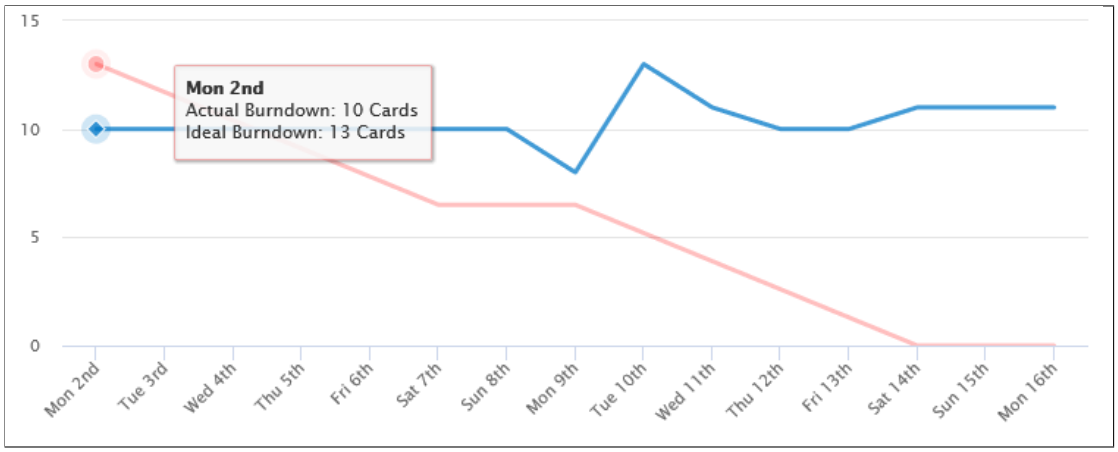
Na začiatku druhého stretnutia prvého šprintu boli odprezentované dosiahnuté výsledky vedúcemu tímu. Následne boli zosumarizované splnené a nesplnené úlohy a boli pomenované chyby, ako napr zlé pushovanie do GitHubu a pod., spolu s riešeniami, ako sa takýmto chybám vyvarovať. Po hraní Scrum Poker hry boli vybrané a pridelené úlohy do ďalšieho šprintu.

-ukončené:

- preskúmanie konektivity C# a MongoDB - Martin Dieška, Martin Gembec
- vygenerovanie testovacích jsonov - Martin Gembec
- vytvorenie statického prototypu WebGL cez ThreeJS - Marcel Furucz, Peter Psota
- vytvorenie webstránky - Martin Dieška
- nasadenie webstránky na server - Patrik Sčensný
- vytvorenie prvotných služieb na získavanie dat z MongoDB - Martin Gembec, Martin Dieška
- vizualizácia objektov pomocou THreeJS - Peter Psota

-neukončené:

- definícia modelových schém - Štefan Motko, Ronald Demeter, Patrik Sčensný
- preštudovanie metamodelu - všetci
- implementácia ovládania kamery pomocou ThreeJS - Marcel Furucz



Obr. 1: Burndown chart pre šprint Layer

3.3 Šprint 2 - Lifeline

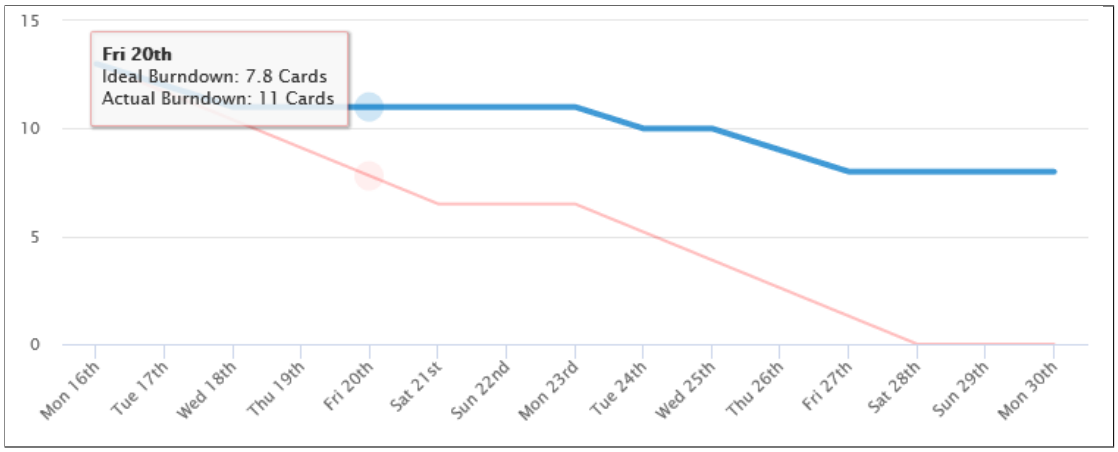
Druhý šprint bol zameraný na implementáciu pokročilejších grafických prvkov, ako napríklad lifeline, a spojením backendu s frontendom do jedného celistvého prototypu.

-ukončené:

- vytvorenie služby pre ukladanie dat v MongoDB - Martin Dieška, Martin Gembec
- dokončenie implementácie ovládania kamery pomocou ThreeJS - Marcel Furucz
- vloženie lifeline na prvú intersect layeru - Peter Psota
- implementácia metamodelu do dátového modelu - Štefan Motko, Ronald Demeter, Peter Psota, Martin Gembec, Martin Dieška
- preštudovanie metamodelu - všetci

-neukončené:

- definícia modelových schém - Štefan Motko, Ronald Demeter, Patrik Sčensný



Obr. 2: Burndown chart pre šprint Lifeline

3.4 Šprint 3 - Message

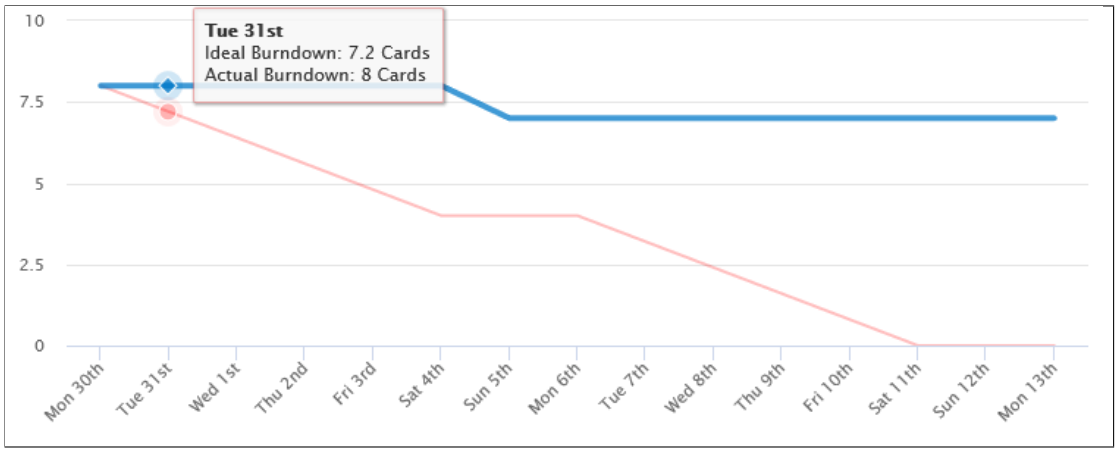
Ako cieľ tretieho šprintu sme si určili pridávanie správ do diagramu a komunikáciu frontendu s backendom. Počas implementácie sme narazili na komplikácie pri orderingu správ, ktoré sme vyriešili diskusiou v druhom týždni šprintu a dohodnutím sa na ďalšom postupe pri riešení.

-ukončené:

- jednoduchý ordering - Štefan Motko, Ronald Demeter, Peter Psota
- automatické layoutovanie prvkov - Peter Psota
- riešenie komplikácii s kamerou z dôvodu migrácie na typescript - Marcel Furucz

-neukončené:

- serializér objektov - Martin Dieška, Martin Gembec
- deserializér objektov - Martin Dieška, Martin Gembec
- komunikácia pomocou GET/PUT/POST - Martin Dieška, Martin Gembec
- definícia modelových schém - Štefan Motko, Ronald Demeter, Patrik Sčensný



Obr. 3: Burndown chart pre šprint Message

4 Používane metodiky

4.1 Metodika plánovania

Táto metodika popisuje metódy používané pri plánovaní, opisuje nástroje a metodiky na to používané. Obsahuje pravidlá pre vytváranie a narábanie s používateľskými príbehmi a úlohami.

4.2 Metodika konvencií písania zdrojového kódu

Pri backend sa používa štandardná c# anotácia odporúčaná Microsoftom. Pri frontende sa používa podobná anotácia ako pri backende.

4.3 Metodika testovania

Pri testovaní používame JavaScript testovací framework Mocha bežiaci na Node.js a v prehliadači. Testy sú písané popri vytváraní písaní zdrojového kódu a konzultácií s autorom funkcionality a projektovým manažérom.

4.4 Metodika prehliadky zdrojového kódu - code review

Pri prehliadky zdrojového kódu sa využíva funkcionalita poskytovaná githubom (Pull request). Žiadateľ vytvori pull request. Ten request sa zhodnotí a podľa zhodnotenia sa vykonajú zmení.

4.5 Metodika pre mapovanie služieb

Metodika určujú architektúru, spôsoby pomenovania a miesto uloženia služby. Takisto je v nich napísaný spôsob mapovania volania týchto služieb.

4.6 Metodika verziovania zdrojového kódu

Táto metodika opisuje pravidla pre vetvenie, verziovanie a zálohovanie zdrojového kódu v rámci Github. Píše sa v nej na akých vetvách sa vypracujú používateľské príbehy, ako tie vetvy vytvárať. Zároveň obsahuje sadu pravidiel, ktoré sa majú dodržiavať počas práci s githubom.

4.7 Metodika písania dokumentácie

Táto metodika popisuje základné pravidlá a postupy pri písaní dokumentácie a zápisníc po každom stretnutí. V tejto metodike nájdeme pravidlá, ako napr.: použité textové editory, veľkosť písma, formát tabuliek a grafov.

5 Globálna retrospektíva

V tejto kapitole je opísaná globálna retrospektíva celého zimného semestra. Našou snahou bolo celý semester zlepšovať tímovú komunikáciu a spoluprácu. Snažili sme sa zabezpečiť čo najlepšiu efektivitu riešenia problémov a rozdelenia práce medzi členov tímu. Vždy na konci šprintu, čo bolo každé dva týždne sme sa venovali retrospektíve v ktorej sme sa rozprávali o tom čo sme dosiahli a čo by sme chceli zlepšiť. Každý člen tímu mal možnosť sa vyjadriť k problémom s ktorými sa stretol a vysvetliť čo sa mu podarilo spraviť. Dôležitou súčasťou retrospektívy bolo poukázať na ďalšie smerovanie projektu a upovedomiť ostatných členov tímu a častiach projektu, ktoré nefungujú správne.

Prvé týždne semestra sme sa zameriavali na zvolenie správnych technológií. Bolo pre nás veľmi dôležité zanalyzovať všetky dostupné technológie a zvoliť tie najvhodnejšie, pomocou ktorých sme schopný naplniť všetky požiadavky zákazníka. Práve zle zvolená technológia bola často dôvodom zdržania a neúspechu predchádzajúcich tímov, ktoré sa venovali Web UML tématike. Výber technológii prebiehal na základe dohody členov tímu a o každej technógii sme spolu diskutovali a ukázali si jej výhody a nevýhody.

Niektorí členovia tímu nemali väčšie skúsenosti s verziovaním kódu a preto vedúci tímu urobil prednášku zameranú na GIT technológiu v ktorej všetkým členom tímu vysvetlil nevyhnutné základy. Oboznámili sme sa s technológiou na komunikáciu, konkrétne služba Slack a nástrojom na riadenie užívateľských príbehov s názvom Trello.

Vďaka predchádzajúcim skúsenostiam niektorých členov tímu v danej tématike sme sa rýchlo prepracovali k prvým návrhom projektu. Na základe retrospektív sme vždy dokázali odhaliť aktuálne problémy s ktorými sme sa následne potýkali v ďalších šprintoch. Jednalo sa o rôzne problémy:

- nedostatočná komunikácia tímu počas šprintu, kde členovia spolu komunikovali len málo a radšej problém riešili sami
- technické problémy spôsobené novými technológiami, s ktorými sa členovia tímu nikdy predtým nestretli
- odkladanie práce na stretnutie
- nedostatočné vysvetlenie problému spôsobujúce zlý postup pri jeho riešení

Podarilo sa nám vytvoriť systém, ktorý poskytuje prvotné grafické rozhranie, ktoré zobrazuje niekoľko vrstvový diagram, ktorý je priamo pomocou servera načítaný z databázy a zobrazovaný na klientovi, konkrétne vo webovom prehliadači. Používateľ môže otáčať kamerou a zobrazovať si diagram z rôznych uhlov a vzdialeností. Taktiež je možné pridávať do diagramu layers, lifelines a messages. Každá zmena sa ukladá do databázy a dáta sú perzistentné.

Slovenská technická univerzita v Bratislave Fakulta informatiky a informačných technológií

Tím 25

Motivačný dokument

Členovia tímu
Bc. Štefan Motko
Bc. Ronald Demeter
Bc. Marcel Furucz
Bc. Patrik Sčensný
Bc. Martin Dieška
Bc. Peter Psota
Bc. Martin Gembec

Akademický rok: 2017/2018

1 Predchádzajúce skúsenosti tímu

- vizualizácia a interakcia s vizualizovanými dátami, prezentácia dát
  - vizualizácia vo webovom prostredí
    - Štefan Motko
    - Ronald Demeter
    - Marcel Furucz
    - Martin Dieška
- manipulácia s dátami(XML, JSON)
- skúsenosti vo vývoji webových aplikácií
  - HTML, CSS, JavaScript, ASP.NET
    - Štefan Motko
    - Martin Dieška
    - Ronald Demeter
    - Marcel Furucz
    - Patrík Sčensný
- skúsenosti v práci s grafickými prostrediami
  - Three.js, D3.js, DevExtreme
    - Ronald Demeter
    - Martin Dieška

Téma prioritného záujmu:

2 14. 3D-UML

Viacerí členovia tímu sa v rámci svojich bakalárskych projektov venovali vizualizáciám v prehliadači, pričom niektorí aj konkrétne vizualizáciám správania softvéru. Viacerí sa zároveň chcú v inžinierskom štúdiu ďalej venovať oblastiam modelovania a architektúry softvéru, ďalší majú záujem o činnosť v oblasti vizualizácie. Tiež sme absolvovali predmety zamerané na modelovanie v UML z ktorých získané poznatky vieme využiť v danom projekte.

3 Príloha A

Zoradenie tém podľa priority od najvyššej po najnižšiu

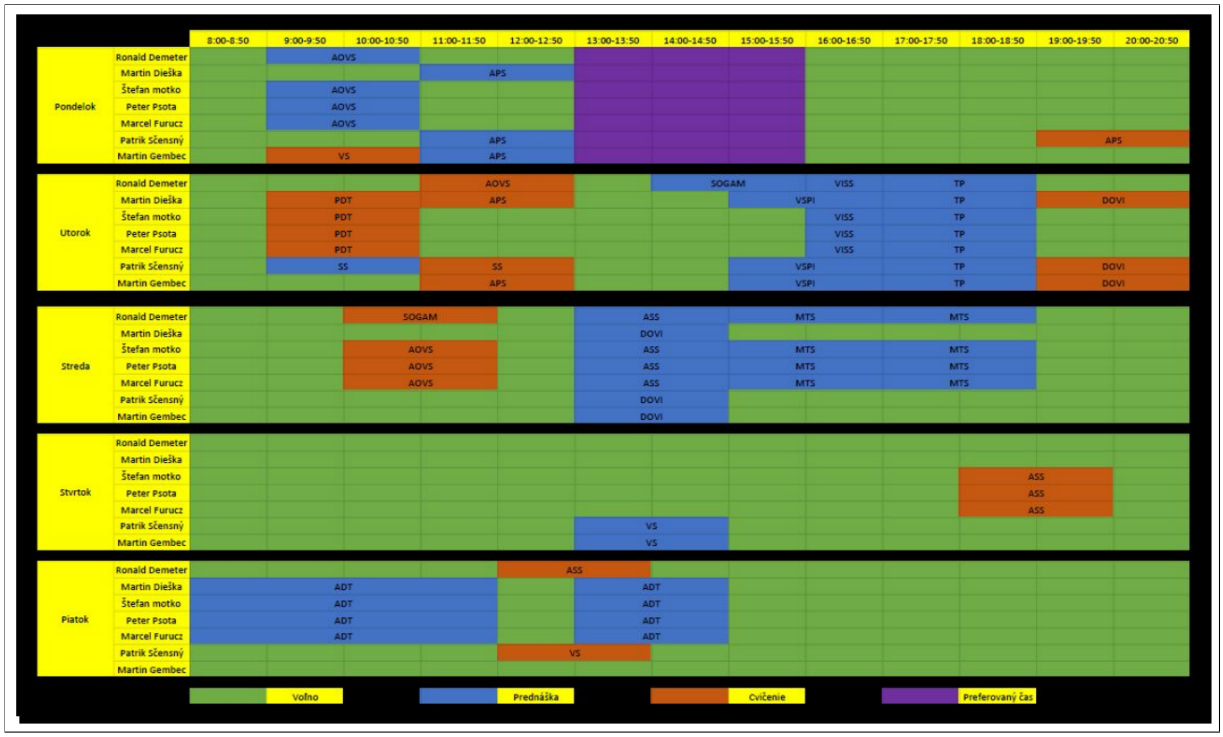
- 3D-UML(14)
- IBazar(4)
- reCommers(9)

Následne hociktorá z nasledujúcich tém s rovnakou prioritou.

- Group(2)
- Zmluvy(3)
- Invest(24)
- EduVirtual(8)
- Look-Inside-Me(10)
- Collab-UI(12)
- DeepSearch(15)
- iWeb(19)
- BehaMetrics(20)
- SmartParking(26)
- OntoSEC(18)
- Mob-UX(13)
- FIIT-DU(11)
- PubDatasets(1)

A v poslednom rade hociktorá zo zostávajúcich tém.

4 Priloha B



Obr. 1: Rozvrh kazdeho clena tymu

Slovenská technická univerzita v Bratislave Fakulta informatiky a informačných technológií

Tím 25 - Bystander Effect

3D-UML Improved

Metodika pre prehliadku kódu

Členovia tímu
Bc. Štefan Motko
Bc. Ronald Demeter
Bc. Marcel Furucz
Bc. Patrik Sčensný
Bc. Martin Dieška
Bc. Peter Psota
Bc. Martin Gembec

Vedúci
doc. Ing. Ivan Polášek, PhD.

Akademický rok: 2017/2018

1 Prehliadka kódu

Človek zodpovedný za používateľský príbeh vytvorí pull request s názvom Code Review_MenoŽiadateľa a vyberie jedného alebo viacerých členov tímu, ktorý bude mať rolu posudzovateľa.

Posudzovateľ po žiadosti o review vykoná prehliadku kódu a komentármi označí miesta nesprávne napísaného kódu. Popritom berie do úvahy metodiku písania kódu.

Po prehliadke posudzovateľ pošle žiadateľovi informáciu ohľadom nedostatkov. Žiadateľ potom ide vyriešiť nedostatky po ich odstránení to oznámy posudzovateľovi. Ak posudzovateľ nenájde ďalšie nedostatky tak schváli pull request. Predchádzajúce kroky sa opakujú pokiaľ sa neodstránia všetky nedostatky.

Slovenská technická univerzita v Bratislave Fakulta informatiky a informačných technológií

Tím 25 - Bystander Effect

3D-UML Improved

Metodika dokumentácie

Členovia tímu
Bc. Štefan Motko
Bc. Ronald Demeter
Bc. Marcel Furucz
Bc. Patrik Sčensný
Bc. Martin Dieška
Bc. Peter Psota
Bc. Martin Gembec

Vedúci
doc. Ing. Ivan Polášek, PhD.

Akademický rok: 2017/2018

1 Dokumentácia

Táto sekcia obsahuje pravidlá a postupy, ktoré musia byť dodržiavané pri písaní dokumentácie.

1.1 Použité nástroje

Dokumentácia ku tímovému projektu je vytváraná v textovom editore ShareLatex.

1.2 Veľkosť a štýl písma

Font a veľkosť písma boli ponechané na východiskový stav article v Latex-u. Pre vytvorenie nadpisov boli použité príkazy ShareLatex-u: hlavné nadpisy príkazom section, nadpisy druhej úrovne príkazom subsection a nadpisy tretej úrovne príkazom subsubsection.

1.3 Tabuľky

Pre tabuľky v dokumente je špecifické modré zafarbenie hlavičky tabuľky. Číslo tabuľky spolu s jej názvom je zadávaný pod tabuľku, centrovaný do stredu dokumentu.

Meno člena tímu	Zodpovednosť v týme
Štefan Motko	Projektový manažér
Ronald Demeter	Hlavný vývojár
Marcel Furucz	Scrum a komunikácia
Patrik Sčensný	Testovanie a kvalita
Peter Psota	Grafický odborník
Martin Gembec	Administrácia databáz a dokumentácia
Martin Dieška	Vývoj serverovej strany

Tabuľka 1: Role členov tímu

Obr. 1: Príklad tabuľky zo zápisnice

2 Zápisnica

V tejto časti sú uvedené pravidlá pri písaní zápisnice.

2.1 Použité nástroje

Pre vytváranie zápinsníc zo stretnutí bol použitý textový editor Microsoft Word.

2.2 Veľkosť a štýl písma

V zápisniciach zo stretnutí je použitý font Calibri, veľkosti 11. V prípade nadpisov je použitý font Calibri Light, veľkosti 13 a modrej farby. Jedná sa o prednastavené písmo pre nadpisy, ktoré ponúka textový editor Microsoft Word.

2.3 Tabuľky

Tabuľky majú predpísanú formu, do ktorej sa po stretnutí dopĺňajú informácie. Na obrázku nižšie je uvedený príklad vyplnenej tabuľky zo zápisnice.

#	Úloha	Zodpovedné osoby	Dátum ukončenia	Status
1	Vytvorenie webstránky	Martin Dieška	16.10.2017	Dokončená
2	Vytvorenie prvotných služieb na získavanie dát z MongoDB	Martin Dieška, Martin Gembec	16.10.2017	Dokončená
3	Implementácia ovládania kamery pomocou ThreeJS	Marcel Furucz	16.10.2017	Nedokončená
4	Vizualizácia objektov pomocou ThreeJS	Peter Psota	16.10.2017	Dokončená
5	Nasadenie webstránky na server	Štefan Motko, Ronald Demeter, Patrik Sčensný	16.10.2017	Dokončená
6	Definícia modelových schém	Štefan Motko, Ronald Demeter, Patrik Sčensný	16.10.2017	Nedokončená
7	Preštudovanie metamodelu	Všetci	16.10.2017	Dokončená

Obr. 2: Príklad tabuľky zo zápisnice

Slovenská technická univerzita v Bratislave Fakulta informatiky a informačných technológií

Tím 25 - Bystander Effect

3D-UML Improved

Metodika písania zdrojového kódu

Členovia tímu
Bc. Štefan Motko
Bc. Ronald Demeter
Bc. Marcel Furucz
Bc. Patrik Sčensný
Bc. Martin Dieška
Bc. Peter Psota
Bc. Martin Gembec

Vedúci
doc. Ing. Ivan Polášek, PhD.

Akademický rok: 2017/2018

1 Serverový kód

Pri písaní serverového kódu (od teraz backend) sme sa rozhodli použiť štandardnú metodiku písania kódu jazyka C# odporúčanú Microsoftom.

1.1 Konvencia nazývania

Triedy a metódy sa píšu vo formáte PascalCase každé slovo začína veľkým písmenom. Premenné a atribúty sa píšu camelCase prvé slovo začína malým písmenom a každé ďalšie slovo veľkým. Privátne premenné začínajú "_".

1.2 Rozmiestnenie

Jedna deklarácia a vyraz na riadok. Používanie tabulátora veľkosťou 4 medzery. Jeden prázdny riadok po definícii funkcie, metódy a podmienky.

1.3 Komentáre

Komentáre sa píšu na samostatný riadok a nie na koniec. Komentáre začínajú veľkým písmenom a sú oddelene od // jednou medzerou a zakončené bodkou.

1.4 Jazyk

Jazyk písania kódu je angličtina.

2 Klientský kód

2.1 Konvencia nazývania

Triedy a metódy sa píšu vo formáte PascalCase každé slovo začína veľkým písmenom. Premenné a atribúty sa píšu camelCase prvé slovo začína malým písmenom a každé ďalšie slovo Veľkým. Privátne premenné začínajú "_"

2.2 Rozmiestnenie

Jedna deklarácia a vyraz na riadok. Používanie tabulátora veľkosťou 4 medzery. Jeden prázdny riadok po definícii funkcie, metódy a podmienky.

2.3 Komentáre

Komentáre sa píšu na samostatný riadok a nie na koniec. Komentáre začínajú veľkým písmenom a sú oddelene od // jednou medzerou a zakončené bodkou.

2.4 Jazyk

Jazyk písania kódu je angličtina.

Slovenská technická univerzita v Bratislave Fakulta informatiky a informačných technológií

Tím 25 - Bystander Effect

3D-UML Improved

Metodika plánovania

Členovia tímu
Bc. Štefan Motko
Bc. Ronald Demeter
Bc. Marcel Furucz
Bc. Patrik Sčensný
Bc. Martin Dieška
Bc. Peter Psota
Bc. Martin Gembec

Vedúci
doc. Ing. Ivan Polášek, PhD.

Akademický rok: 2017/2018

Metodika plánovania hovorí o vytváraní, prideľovaní a manažmente úloh a používateľských príbehov.

1 Šprint

Cieľom šprintu je okrem iného aj vytvoriť príbehy s dostatočnou granularitou. Každý používateľský príbeh je vytvorený, ohodnotený príbehovými bodmi a poprípade následne rozdelený na menšie. Pri vytvarani príbehov je pár pravidiel, ktoré sa dorživajú:

- Príbeh je dobre a jasne popísaný
- Príbeh ma určené aspoň približné úlohy

2 Vytváranie uloh

Po vytvorení príbehov sú prideľované členom tímu na vývoj podľa sekcie. Je možné, aby na jednom príbehu pracovalo viacej členov tímu, nie však na jednej úlohe.

3 Stavy používateľského príbehu

V priebehu zivota používateľský príbeh prechádza rôznymi stavmi:

- Backlog - východiskový stav. Príbeh sa takto označí po vytvorení, pred zaradením do šprintu.
- In development - Príbeh je zaradneý do šprintu a je aktívne vyvýjaný
- Testing - Príbeh je funkčný, prebieha kontrola kvality kódu a testovanie
- Pending - Príbeh bol skontrolovaný a má vytvorený pull request.
- Done - Príbeh bol zlúčený do dev vetvy.

Prechody medzi stavmi majú niekoľko pravidiel:

- Z backlogu do in development sa úlohy presúvajú len priradenými členmi tímu, potom ako na ňom začnú pracovať
- Pull request musí byť schválený všetkými členmi tímu, ktorý boli na príbeh priradený.

4 Spravovanie používateľských príbehov

Každý používateľský príbeh je zaradený do sekcie podľa jeho zamerania. Vedúci sekcie dohliada na používateľský príbeh pod jeho sekciou a organizuje stretnutia členov k ním priradeným.

5 Nástroj Trello

Na plánovanie a spravovanie používateľských príbehov používame voľne dostupný nástroj Trello. Pouzivatelske príbehy sú karticky, ktoré sa presúvaju medzi stĺpcami, reprezentujúce ich stav. Členovia tímu sú registrovaný na Trelle a priraďujú sa k príbehom, ktoré su im určené. Príbehove karticky majú zoznamy položiek, reprezentujuce úlohy. Sekcie používateľských príbehov sú reprezentovane farbami kartičiek.

Slovenská technická univerzita v Bratislave Fakulta informatiky a informačných technológií

Tím 25 - Bystander Effect

3D-UML Improved

Metodika pre mapovanie služieb

Členovia tímu
Bc. Štefan Motko
Bc. Ronald Demeter
Bc. Marcel Furucz
Bc. Patrik Sčensný
Bc. Martin Dieška
Bc. Peter Psota
Bc. Martin Gembec

Vedúci
doc. Ing. Ivan Polášek, PhD.

Akademický rok: 2017/2018

1 Architektúra

Metodika webových služieb vychádza z REST architektúry. Každá služba musí obsahovať niektorú z RESTových metód, aby sme mohli hovoriť o RESTovej architektúre webových služieb. Jedná sa o metódy:

- GET - čítanie a len čítanie zdrojov
- PUT - aktualizácia existujúceho zdroja
- DELETE - odstránenie zdroja
- POST - pridanie nového zdroja

2 Pomenovanie služieb

Služby pomenúvame podľa objektu s ktorým pracujú. Pokiaľ sa jedná o prácu s viacerými objektami, pomenúva službu podľa funkcionality, ktorú poskytuje nad danými objektami.

3 Smerovanie

Pre poskytovanie služieb sa využívaju objekty Controller, keďže serverová architektúra je typu MVC. Každý controller obsahuje niekoľko metód a môžeme hovoriť o službe. Metódy sú mapované nasledovne: Na začiatku controllera sa nastaví URL cesta na ktorej sa controller nachádza. Následne každá jednotlivá metóda ďalej špecifikuje, ako sa bude volať:

api/[controller]/[action]/parameters

4 Framework

ASP .NET Core 2 nám poskytuje ASP.NET MVC 6 Controller, ktorý spája funkcionalitu predchádzajúceho MVC modelu a Web API. Obsahuje niekoľko typov smerovaní, vie vracať View aj objekt, ktorý dokáže serializovať.

Slovenská technická univerzita v Bratislave Fakulta informatiky a informačných technológií

Tím 25 - Bystander Effect

3D-UML Improved

Metodika testovania

Členovia tímu
Bc. Štefan Motko
Bc. Ronald Demeter
Bc. Marcel Furucz
Bc. Patrik Sčensný
Bc. Martin Dieška
Bc. Peter Psota
Bc. Martin Gembec

Vedúci
doc. Ing. Ivan Polášek, PhD.

Akademický rok: 2017/2018

1 Unit testing

V projekte rozdelíme testovanie na dve časti:

- **testovanie backendu** - na tetovanie serverového kódu sa používajú štandardné anotácie pre testovanie dostupné v jazyku C#
- **testovanie frontendu** - na testovanie klientskeho kódu prebieha pomocou nástroja Mocha

Na základe požiadaviek a špecifikácií od člena týmu, osoba zodpovedná za písanie testov navrhne a napíše test, ktorý sa bude využívať na zistenie správnej funkcionality.

Slovenská technická univerzita v Bratislave Fakulta informatiky a informačných technológií

Tím 25 - Bystander Effect

3D-UML Improved

Metodika verziovania

Členovia tímu
Bc. Štefan Motko
Bc. Ronald Demeter
Bc. Marcel Furucz
Bc. Patrik Sčensný
Bc. Martin Dieška
Bc. Peter Psota
Bc. Martin Gembec

Vedúci
doc. Ing. Ivan Polášek, PhD.

Akademický rok: 2017/2018

1 Vetvenie zdrojového kódu

Na verziovanie zdrojového kódu používame verziovací systém Git. Spoločné úložisko zdrojových kódov udržiavame na stránke GitHub. Vytvorili sme dve primárne vetvy(angl. branch).

- master - produkčná vetva - obsahuje poslednú prezentovanú a akceptovanú verziu vyvíjaného softvéu
- dev - vývojová vetva - obsahuje najnovšie dokončené črty

Pre každý používateľský príbeh sa vytvorí nová vetva s názvom feature/"popis"(kde popis je názov vyvíjanej črty), v ktorej sa pracuje na danom používateľskom príbehu. V prípade zistenia chýb vo vetvách master alebo dev sa vytvára nová vetva s názvom fix/"popis".

1.1 Postup vytvárania vetvy pre používateľský príbeh

Na používateľskom príbehu sa pracuje na samostatnej vetve. Vetvu vytvorí jeden z členov tímu, ktorému bolo priradené riešenie daného používateľského príbehu. Pre vetvy platia nasledujúce pravidla:

- Názov sa píše po anglicky.
- Názov má tvar feature/"príbeh"napr. "feature/state-machine"
- Viacslovné pomenovania sa oddeľujú pomlčkou.

Postup pri vytvorení novej vetvy:

1. Otvoriť konzolu schopnú spustiť program git
2. Prepnúť sa na vetvu dev pomocou príkazu "git checkout dev"
3. Vytvoriť novú vetvu pomocou príkazu "git checkout -b <meno novej vetvy>"
4. Publikovať novú vetvu na vzdialené úložisko pomocou príkazu "git push –set-upstream origin <meno vytvorenej vetvy>"

1.2 Pravidla pracovania vo vetve

1. Do vetvy master a dev sa neprispieva priamo
2. Pracuje sa exkluzívne vo vetvách feature/* a fix/*
3. Webová prezentácia projektu má vlastné úložisko nezávislé od vyvíjaného softvéru
4. Pred vytvorením novej feature vetvy je potrebné aktualizovať lokálnu verziu vetvy dev
5. Revízie zdrojového kódu sa ukladajú po čo najmenších logických celkoch pre udržiavanie čitateľnosti histórie
6. Akýkoľvek text zadávaný do verziovacieho systému je v anglickom jazyku
7. Pri ukončení práce na používateľskom príbehu vytvorí spracovateľ pull request do vetvy dev, pričom všetkých kolaborátorov zaradí do vetvy ako schvaľovateľov
8. Po integrácii vývojovej vetvy do vetvy dev sú členovia tímu pracujúci na ostatných vývojových vetvách povinní spätne integrovať zmeny vo vetve dev do svojich vývojových vetiev