VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

MODUL PRO SLEDOVÁNÍ STAVU PROJEKTŮ PRO KANBOARD.ORG

PROJECT STATUS TRACKING PLUGIN FOR KANBOARD.ORG

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **ANDREJ MASÁR**
AUTHOR

VEDOUCÍ PRÁCE **Ing. FRANTIŠEK ŠČUGLÍK, Ph.D.**
SUPERVISOR

BRNO 2019

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav informačních systémů (UIFS) Akademický rok 2018/2019

Zadání bakalářské práce

Student: **Masár Andrej**
Program: Informační technologie
Název: **Modul pro sledování stavu projektů pro Kanboard.org**
Project Status Tracking Plugin for Kanboard.org
Kategorie: Web

Zadání:

1. Seznamte se s nástrojem Kanboard.org a s technologiemi PHP a SQLite.
2. Nastudujte možnosti tvorby vestavných modulů pro Kanboard.org.
3. Navrhněte modul pro sledování stavu projektů včetně plánů na zotavení. V rámci modulu zohledněte možnost vytvoření souhrnné zprávy o stavu projektů.
4. Po konzultaci s vedoucím navržené řešení implementujte a otestujte. Dosažené výsledky zhodnoťte.

Literatura:

- Dokumentace Kanboard.org [online], Dostupné z: https://docs.kanboard.org/en/latest/
- Welling, L., Thomson, L.: PHP a MySQL Kompletní průvodce vývojáře, CPress, 2017
- Kroenke, D., M., Auer, D. J.: Databáze, CPress, 2014

Podrobné závazné pokyny pro vypracování práce viz http://www.fit.vutbr.cz/info/szz/

Vedoucí práce: **Ščuglík František, Ing., Ph.D.**
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2018
Datum odevzdání: 15. května 2019
Datum schválení: 30. října 2018

Abstrakt

Táto bakalárska práca sa zaoberá návrhom, implementáciou a testovaním vstavaného modulu pre aplikáciu Kanboard.org, ktorý umožňuje sledovanie stavu projektov aplikácie a tvorbu plánov na zotavenie projektu. Práca rozoberá vývoj pomocou metódy Kanban a popisuje aplikácie, ktoré implementujú túto metódu. Obsahuje popis technológií použitých pri implementácii. Výsledný modul je zverejnený a voľne dostupný na portáli GitHub.com.

Abstract

This bachelor thesis is concentrated on design, implementation and testing of plugin for aplication Kanboard.org, that allows project status tracking and creation of recovery plans for project. It analyzes Kanban development and aplications that implemnte it. It contains description of technologies used during plugin development. The result plugin is published and freely available on GitHub.com.

Kľúčové slová

Kanboard.org, kanboard modul, Kanban, sledovanie stavu projektov, plán na zotavenie, Lean metodológia, manažment problémov

Keywords

Kanboard.org, kanboard plugin, Kanban, project status tracking, recovery plan, Lean metodology, issue management

Citácia

MASÁR, Andrej. *Modul pro sledování stavu projektů pro Kanboard.org*. Brno, 2019. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. František Ščuglík, Ph.D.

Modul pro sledování stavu projektů pro Kanboard.org

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pána Ing. Františka Ščuglíka Ph.D. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

.....
Andrej Masár
15. mája 2019

Poďakovanie

Ďakujem svojmu vedúcemu Ing. Františkovi Ščuglíkovi Ph.D. za profesionálny prístup a pochopenie, svojej rodine a priateľke za neoceniteľnú podporu.

Obsah

Kapitola 1

Úvod

Táto práca popisuje tvorbu modulu do bezplatnej aplikácie Kanboard.org. Aplikácia sa zaoberá témou spravovania a manažmentu projektov. Systém riadenia projektov čerpá z povojnových metód výroby automobilov vo firme Toyota. Základné prvky týchto metód, ako napríklad vizualizácia pracovného procesu, sú použiteľné v rôznych odvetviach výroby aj v súčasnosti. Tvorený modul má za úlohu umožniť v aplikácii sledovať aktuálny stav v akom sa projekt nachádza a tvorbu plánu na zotavenie. V súčasnej podobe to aplikácia Kanboard.org ani jej dostupné moduly neumožňujú.

Hlavná aplikácia ponúka základnú funkcionalitu, ktorú je možné doplniť prostredníctvom modulov. Všetky časti aplikácie sú bezplatne dostupné, takže ju môže vylepšovať ktokoľvek. Vytvorený modul bude tiež voľne dostupný na stiahnutie a použitie, prípadné vylepšenie. Úspechom modulu bude jeho použitie čo najväčším počtom užívateľov.

V nasledujúcich kapitolách budem postupne rozoberať problematiku metodológie Kanban, na ktorej hlavná aplikácia stojí a manažment problémov 2, ktorý bude použitý pri tvorbe plánov na zotavenie. Ďalej popíšem dostupné alternatívy hlavnej aplikácie ako aj samotnú aplikáciu 3. Kapitola 4 vymenúva použité technológie pri tvorbe modulu. Návrh a tvorba aplikácie je popísaná v kapitole 5. Testovanie a postupné vylepšovanie modulu je vysvetlené v kapitole 6. Posledný je záver dokumentu 7, kde sú zhrnuté výsledky práce.

Kapitola 2

Softvérový vývoj pomocou Kanbanu

V tejto kapitole sú stručne popísané aspekty vývoja za použitia metódy Kanban, jeho zaradenie a časť teórie manažmentu implementácie, ktorá je používaná v návrhu aplikácie. V sekcii 2.1 je jednoduchý popis agilného vývoja, pod ktorý Kanban spadá. Sekcia 2.2 rozoberá princípy Štíhlej výroby a z nej odvodenej metodiky Lean development, pod ktorú spadá práve Kanban. Sekcia 2.3 hovorí konkrétne o metóde Kanban a jej štyroch zásadách. Posledná sekcia 2.4 sa zaoberá teóriou manažmentu problémov v projekte, ktorá nie je priamo súčasťou Kanbanu, ale je dôležitá pre pochopenie ďalších kapitol.

2.1 Agilný vývoj softvéru

Je metodológia vývoju zameraná na postupný, iteratívny vývoj softvéru. Jej etapový charakter umožňuje, za prebiehajúceho vývoja, meniť zadanie projektu, ako aj spôsob jeho riešenia. Na začiatku projektu existuje vízia, z ktorej sa vytvorí zoznam požiadaviek, alebo definuje požadovaná funkcionalita. Tento prístup je veľmi užitočný pri projektoch objednaných konkrétnym užívateľom, ktorý disponuje víziou a požiadavkami na funkcionalitu. Presný postup vývoja a výsledky jednotlivých častí však nie je možné dopredu predpokladať. Pretože iteračná práca na splnenie jednotlivých požiadaviek je riskantná a bez záruky jasného výsledku, je dôležité úzko spolupracovať so zákazníkom na nastavení priorít jednotlivých etáp vývoja. [5]

Po tom, čo sú definované ciele zákazníka, tím pracujúci na projekte začne vykonávať prácu v malých krokoch, ktorých výsledkom je splnenie konkrétnej časti cieľu - funkcionality. Na konci každého kroku môže tím identifikovať prácu potrebnú na splnenie ďalšej časti. Iteračný postup umožňuje prácu viacerých tímov na jednotlivých krokoch vývoja. Tímy pracujúce agilne často používajú metódu rozdelenia času na separátne periódy, ktoré majú vlastný deadline (termín ukončenia), rozpočet a výstupy.

Za hlavný rozdiel agilného manažmentu od tradičného prístupu sa považuje rozdelenie zodpovednosti na jednotlivých členov tímu, ako aj ich roly. Agilné tímy sú z časti samoorganizované, keďže je na každom členovi tímu, ako a kedy vykoná svoju prácu v rámci iterácie. Na rozdiel od tradičného prístupu, kde o tom rozhoduje projektový manažér. Aby bola udržateľná kontrola nad projektom, vykonáva tím prácu transparentne. Každý pracovník tímu má prístup k dennému postupu ostatných členov, takže problémy so súčasťami

projektu môžu byť priamo adresované ich pôvodcovi. V tomto je preferovaná práca tímu v spoločných priestoroch, kde môžu členovia spolu komunikovať priamo.

Agilné metodológie

Agilný vývoj softvéru bol pevne zadefinovaný v manifeste vydanom v roku 2001. Manifest prichádza s dvanástimi princípmi agilného vývoja a stáva sa najviac odkazovanou definíciou agilného vývoja. Na pozadí manifestu sa začalo vyvíjať viacero metodológií, ktoré sa začali súhrne nazývať agilné metodológie. Medzi základné, z ktorých sa neskôr vyvinulo mnoho ďalších, môžeme považovať:

- Adaptívny vývoj softvéru (Adaptive Software Developlemt, ASD)
- Crystal metodiky
- Extrémne programovanie (XP)
- Vývoj riadený vlastnosťami softwaru (Feature driven development, FDD)
- Vývoj riadený testami (Test driven development, TDD)
- SCRUM Development process
- Lean development

2.2 Lean development

Základné myšlienky pochádzjú z Japonskej priemyselnej výroby začiatku 20. storočia. Štíhla výroba (Lean manufacturing) je metodika vyvinutá firmou Toyota. Zameriava sa na minimalizovanie plytvania, zatiaľ čo sa snaží naplno uspokojiť zákazníkove požiadavky. Pojem Lean software development sa objavuje až v roku 2003, odkedy by sa dalo datovať použitie Lean myslenia pri tvorbe softvéru. Lean development je teda postavený na princípoch definovaných v Štíhlej výrobe. [19]

Tu je sedem hlavných princípov popísaných pre vývoj softvéru [14]:

1. Eliminovať plytvanie – zahrňuje nedokončenú prácu, čakanie, presun ľudí či zariadenia, tvorbu extra funkcionality, presúvanie úloh medzi pracovníkmi, používanie nesprávnych nástrojov, všetko čo nie je potrebné.

2. Zlepšiť učenie – zlepšovanie výrobného procesu, ktorý sa deje v cykloch, skúšanie nových stratégií, spätná väzba od kolegov a zákazníka, iteračné testovanie.

3. Rozhodnúť sa čo najneskôr – robiť rozhodnutia na základe faktov, nie dohadov (prognóz), čím sa môže predísť nesprávnym rozhodnutiam, ktoré spôsobia predraženie či meškanie.

4. Rýchle doručenie – skoré dodanie funkčného produktu zákazníkovi, na základe ktorého môže poskytnúť spätnú väzbu, ktorá je použitá ako vstup pre ďalšiu iteráciu. Skrátenie iterácií dáva možnosti zdržania rozhodnutí, zlepšenie učenia a komunikácie v tíme.

5. Posilniť tím – motivovať členov tímu, zlepšovať ich spoluprácu, vytvoriť priateľské pracovné prostredie, dávať slobodu v rozhodovaní, nastaviť správnu organizačnú štruktúru.

6. Budovať integritu – vnímanú integritu (zákazník musí mať predstavu ako dobre projekt rieši problémy, ako moc je intuitívny), koncepčnú integritu (rovnováhu medzi flexibilitou, udržiavateľnosťou, efektívnosťou a škálovateľnosťou).

7. Mať celkový obraz – dekomponovať veľké úlohy, štandardizovať vývoj, nastaviť dobré vzťahy so subdodávateľmi, inými tímami, rozumieť Lean mysleniu a vzdelávať v ňom všetkých členov projektu.

Za hlavnú myšlienku sa považuje odstránenie všetkého nepotrebného, čo v priebehu vývoja vzniká, zvyšuje náklady a znižuje efektivitu práce. Lean development je teda súbor pravidiel, ako by mal vývoj prebiehať, čo by mal minimalizovať, odstraňovať, aký by mal byť prístup k zákazníkovi a členom tímu. Agilný koncept sa nachádza v bodoch 3, 4 a 5, kde sa rozhodovanie deje na konci každej iterácie, čím sa vytvára priestor na čo najlepšie testovanie, diskusiu o splnení požiadaviek, ako aj dodanie funkčnej verzie zákazníkovi na poskytnutie spätnej väzby, s dôrazom na samo-organizované tímy. [5]

Posilňovanie tímu je kľúčovým prvkom práve pri vývoji softvéru. Tradične to boli práve manažéri, ktorí rozhodovali o postupe práce na základe čoho prikazovali pracovníkom v tíme, ako by mali robiť svoju prácu. Tento vzťah sa mení tak, že manažéri by mali počúvať členov tímu, ich názoru na riešenie problému a zlepšenie produktu, podnecovať ich komunikáciu. Dôležité je mať schopný tím, ktorý dokáže samostatne vykonávať svoju prácu.

Za posledné roky sa do popredia metód vyvíjania softvéru dostáva Kanban. Vychádza priamo z metódy Štíhlej výroby.

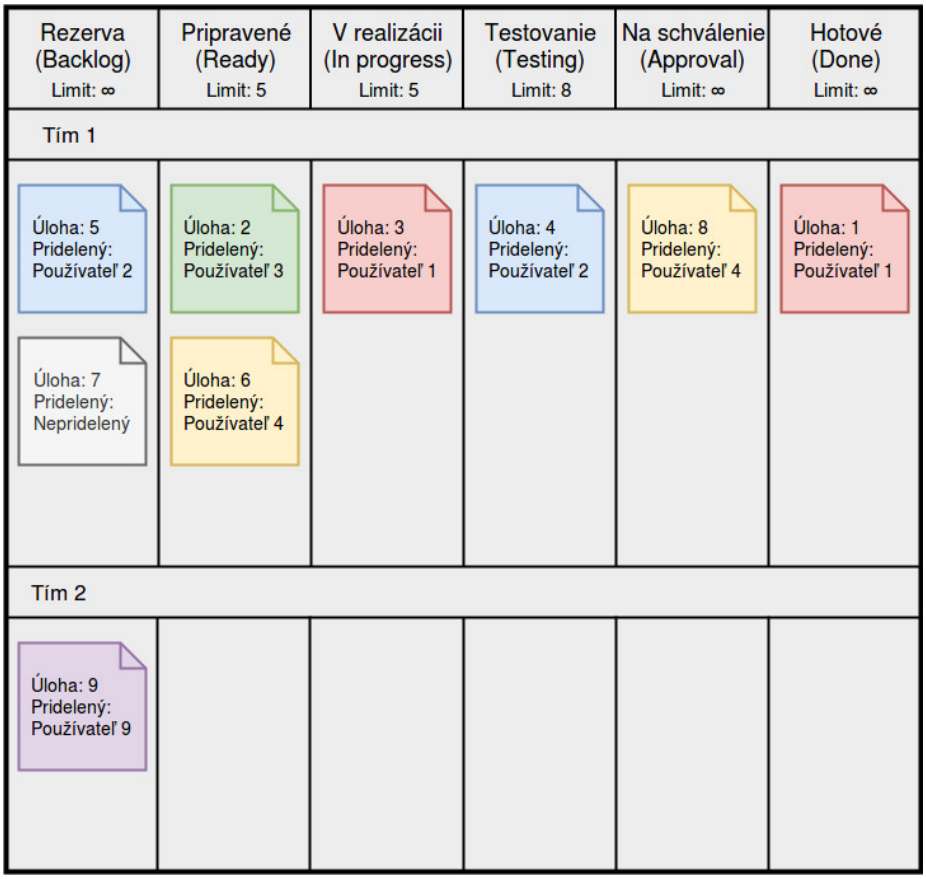
2.3 Kanban

Dá sa z Japončiny preložiť ako signálna karta alebo vývesná tabuľa. Pri výrobe áut to bola práve signálna karta, ktorá sa používala ako mechanizmus, ktorý pomohol zabezpečiť že diel auta dorazil na stanovisko práve včas (Just in time). Fyzická karta sa posielala dodávateľovi ako signál, že výroba potrebuje ďalší diel. Tá istá karta bola potom poslaná s dielom naspäť. Podstatou je, že počet kariet sa nemení bez formálneho rozhodnutia. To zabezpečuje, že počet ešte nepoužitých dielov sa drží na hranici, ktorá maximalizuje tok a zároveň minimalizuje počet vyrobených, ale nepoužitých automobilových dielov. Nepoužité diely sa totiž považujú za plytvanie. [7]

Vizualizácia práce

Ak vychádzame z druhého možného prekladu (vývesná tabuľa) je základným kameňom Kanbanu vizualizácia práce [7]. Je možné použiť elektronickú tabuľu prostredníctvom softvéru, alebo fyzickú tabuľu. Tabuľa zobrazuje prácu v rôznych fázach vývoja pomocou kartičiek reprezentujúcich jednotlivé úlohy. Fázy vývoja sú zastúpené ako stĺpčeky. Kartičky sa hýbu smerom zľava do prava, čo značí postup vývoja a pomáha koordinovať prácu v tíme. Stĺpčeky môžu byť rozdelené aj horizontálne na takzvané swimlanes, reprezentujúce odlišný typ práce alebo prácu iného tímu. Typ a počet fáz nie je fixný. Existuje nekonečné množstvo rôznych tabúľ. Jednoduché tabule majú fázy pre čakajúce úlohy, úlohy v realizácii a dokončené úlohy. Komplexnejšie môžu deliť fázu v realizácii na viacero fáz, aby zachytili proces vývoja detailnejšie. Populárny príklad tabule pre vývoj softvéru je zložený z fáz: Rezerva, Pripravené, V realizácii, Testovanie, Na schválenie a Hotové. Kanban je použiteľný ako pre jednotlivca, tak pre veľké a malé firmy.

Základom k vytvoreniu správnej tabule je rozumieť ako funguje proces plnenia úloh. V Kanban terminológii takejto schéme hovoríme value stream map. Zobrazením práce a pracovného postupu sa dá sledovať tok práce pohybujúci sa systémom Kanban.



Obr. 2.1: Príklad vývesnej tabule so šiestimi fázami vývoja. Tabuľa je rozdelená horizontálne pre tím 1 a tím 2.

Limitácia rozpracovaných úloh

Ďalším zo základných kameňov metódy Kanban je limitovanie počtu úloh, na ktorých sa pracuje. V agilnej metodológii limity vystupujú pod skratkou WiP[1] limity. Nastavením limít sa zabraňuje otváraniu viacerých úloh, ako je tím schopný zvládnuť. Tak isto redukuje počet rozpracovaných, ale nedokončených úloh, čím núti tím zamerať sa na menší počet úloh a buduje kultúru dokončenia úloh. Pokiaľ jeden člen tímu pracuje vo fáze rýchlejšie ako jeho kolegovia v ďalšej a stane sa, že nasledujúca fáza dosiahla svoj limit, musí čakať kým sa uvoľní kapacita. Čakajúci člen tímu môže využiť čas na oddych, alebo v lepšom prípade, sústrediť sa na odstránenie tohto úzkeho hrdla projektu. [7]

WiP limity teda zviditeľňujú blokujúce úlohy a úzke hrdlo projektu. Toto umožňuje tímu pozrieť sa na problémové úlohy spoločne, pochopiť ich, implementovať a vyriešiť, ak existuje jasný ukazovateľ toho, že práca spôsobuje úzke hrdlo. Po tom, čo sa fáza odblokuje, uvoľní sa aj tok práce v tíme. Tieto výhody garantujú, že vykonaná práca sa dostane k zákazníkovi skôr, čo robí z WiP limít dobrý agilný nástroj.

[1]Work in Progress – doslovný preklad je „práca v priebehu“, alebo „rozpracovaná práca“.

Správa toku

Dôležitým parametrom vývoja je čas od začatia práce na úlohe, po jej ukončenie, ako aj čas od zadania požiadavky, po jeho ukončenie. Ideálom je aby boli oba časy čo najnižšie, podľa princípu rýchleho doručenia definovaného v štíhlej výrobe. Hodnota požiadavky s pribúdajúcim časom väčšinou klesá a to je dôvod prečo tieto časy treba znižovať. Pri akejkoľvek zmene organizácie, alebo spôsobu práce na projekte nám práve tieto dva parametre povedia, či mala zmena kladný vplyv na rýchlosť dodania. [7]

Postupné zlepšovanie

Kanban nám hovorí, aby sme použili všetku dostupnú teóriu o toku práce, ktorú môžeme nájsť a aplikovali ju na proces vývoja, aby sa znížil čas dodania. Môže to byť Lean, teória hromadnej obsluhy, teória chaosu, teória hier, teória obmedzení a mnoho iných. Dôležité je dohodnúť sa na zmenách v konsenze, aby bol každý člen tímu vyrovnaný s rozhodnutím. Niektoré zmeny môžu mať negatívny dopad na začiatku, ale sú užitočné z dlhodobého hľadiska. [7]

2.4 Manažment problémov projektu

Je proces identifikácie a riešenia problémov. Problémy s dodávateľmi, chyby vo vývoji, zlyhanie techniky, nedostatok financií, málo pracovníkov, to všetko môže mať negatívny dopad na projekt. Pokiaľ sa problém nerieši hrozí vznik zbytočných konfliktov, oneskorení, alebo dokonca zlyhanie celého projektu. Patria sem však aj otázky vznesené členom tímu, alebo zákazníkom, dodávateľom. Problémy obsahujú rizikové udalosti, ktoré môžu nastať. [2]

Problémy a riziká

Napriek tomu, že pri oboch je pravá podstata pred začiatkom implementácie neznáma, pri rizikách existuje pravdepodobnosť, že takýto stav môže nastať. Problémy sú väčšinou oveľa menej predvídateľné. Môžu prísť bez akéhokoľvek náznaku. Napríklad neschopnosť nájsť kvalifikovaný personál je identifikovateľné riziko. Naopak pokiaľ má člen tímu autonehodu a bude hospitalizovaný niekoľko týždňov, je to problém. Je dôležité identifikovať riziká pred začatím projektu. Dobrým spôsobom môže byť vytvorenie tabuľky pravdepodobností rizík a pripraviť plán manažmentu tých rizík, na ktoré sa dá nájsť riešenie dopredu. Riziká, ktoré neboli odstránené, alebo na ne nebolo pripravené riešenie sa môžu stať neskôr problémom.

Záznam problémov

Problémy, inkonzistencie, alebo konflikty je potrebné zaznamenávať, keď nastanú. Na to slúžia správy o stave projektu, plány na zotavenie alebo čisto záznamy problémov. Je nutné ich zaznamenať, vyšetriť a vyriešiť čo najefektívnejšie. Bez ich zaznamenávania existuje riziko že nebude problém vyriešený úplne, alebo sa objaví znovu a jeho predošlé riešenie nebude zaznamenané.

Pri problémoch je dobré zaznamenávať:

- **Kto objavil problém** – uľahčuje to špecifikáciu problému, pri akej práci k nemu došlo, ako navodiť problém. Riešiteľ problému sa môže objaviteľa spýtať na aj na nepopísané detaily.

- **Čas, kedy sa problém objavil** – môže pomôcť k zisteniu, ako problém nastal. Táto informácia je potrebná pre určenie celkového času nutného na vyriešenie problému.
- **Komu je riešenie problému pridelené** – určuje, kto je zodpovedný za vyriešenie problému. Táto osoba nemusí nutne implementovať riešenie, ale je zodpovedná za nájdenie problému a zaistenie že bude problém vyriešený.
- **Dátum, dokedy je nutné problém vyriešiť** – pokiaľ problém ovplyvňuje vývojový proces, nastaví termín, kedy je očakávané, že sa závislé úlohy môžu pohnúť vpred. Slúži tiež na to, aby sa neodkladalo riešenie problému s malým dopadom.
- **Status** – hovorí o fáze riešenia problému, v ktorej sa nachádza. Môže byť definovaný jednoducho ako: neriešený, v realizácii, hotový, alebo môže byť realizácia rozdelená do viacerých bodov pre lepší prehľad stavu riešenia.
- **Popis problému** – obsahuje detaily zistené o vzniku problému, potencionálny dopad na projekt. Pokiaľ zostane problém nevyriešený, definuje, ktoré časti projektu ním budú ovplyvnené.
- **Popis riešenia** – popisuje, čo bolo potrebné spraviť pre nájdenie a implementáciu riešenia problému. Slúži ako návod, pokiaľ sa problém zopakuje, alebo slúži ako dokumentácia pokiaľ implementácia problému nebola optimálna.

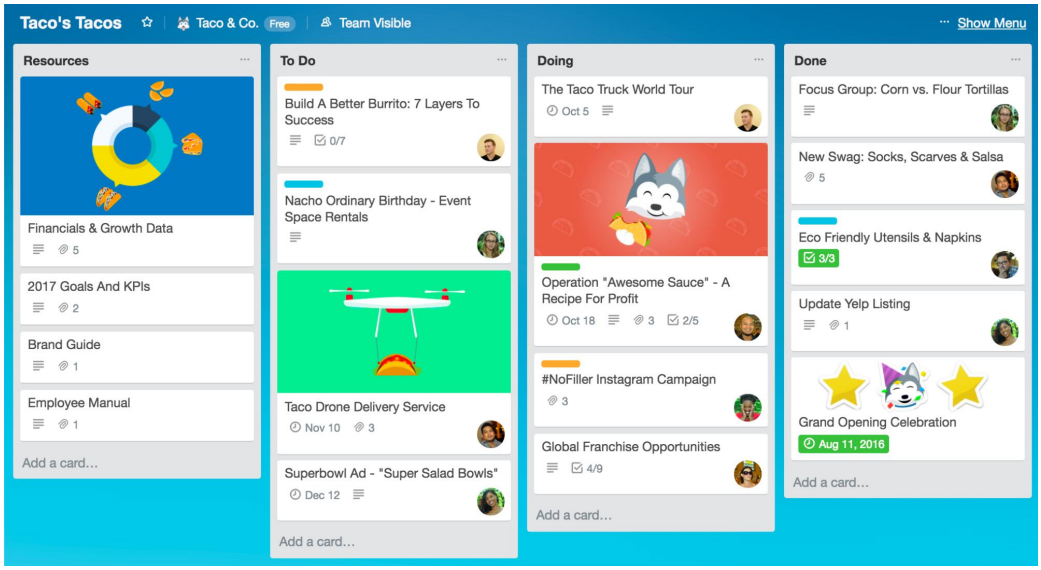
Kapitola 3

Kanboard.org a existujúce aplikácie

Táto kapitola popisuje aplikácie, v ktorých je použitá metóda Kanban, alebo sú ňou inšpirované. Keďže na trhu existuje viac ako 30 takýchto aplikácii, sú z nich vybraté tie, ktoré sú populárne a zároveň prinášajú nejaký nový element. Funkcionalitu aplikácií ako tabule projektov, swilanes, termín splnenia úlohy, farby úloh, status úlohy, podúlohy, komentáre v úlohách atď. nespomínam, pretože nimi disponuje každá z ďalej spomenutých aplikácií a patria k úplne esenciálnym nástrojom aplikácií implementujúcich Kanban. Posledná sekcia 3.4 sa zaoberá detailnejšie aplikáciou Kanboard.org pre ktorú je vyvinutý nový modul sledovania stavu projektov.

3.1 Trello

Je najznámejšia aplikácia poskytujúca Kanban. Svoju popularitu získala hlavne vďaka veľmi príjemnému užívateľskému rozhraniu [Obr. 3.1].

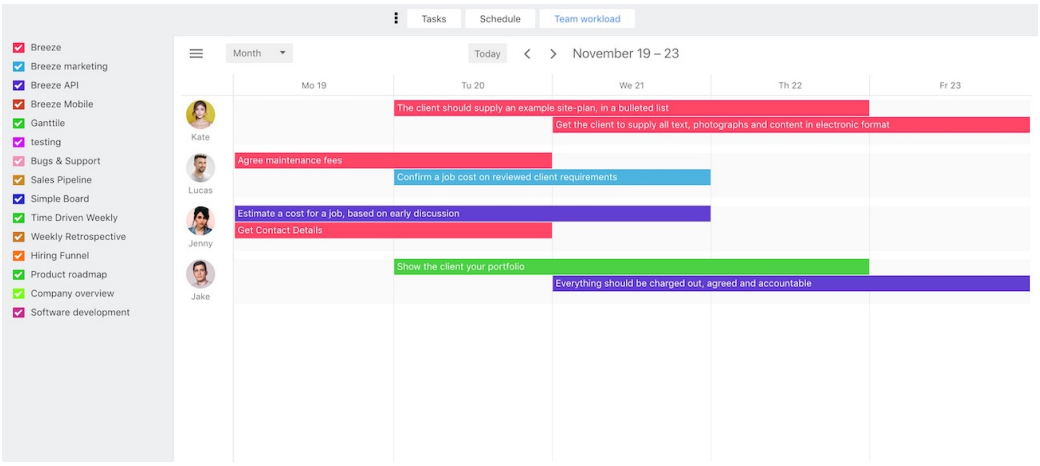


Obr. 3.1: Ukážka vizualizácie práce a pracovného toku v aplikácii Trello. Aplikácia je kladne hodnotená aj vďaka modernému a príjemnému užívatelskému rozhraniu.

Podporuje Markdown syntax, upozornenia emailom, API[1]. Integruje v sebe prepojenia na Salesforce, Slack, GitHub, Google Drive, Evernote a mnohé iné používané aplikácie a platformy pomocou vylepšení nazvaných „Power-Ups“. [18] S IFTTT[2] je možné automatizovať tok práce. Napríklad pokiaľ je v projekte uloženom na portáli GitHub založená nová chyba, Trello automaticky vytvorí novú kartu. V základnom nastavení má Trello veľmi jednoduchú funkcionalitu (chýba tu sledovanie nákladov projektu a času), ktorá sa dá doplniť pomocou vylepšení, prepojení na ďalšie aplikácie. Táto prispôsobiteľnosť je však veľmi obmedzená pri bezplatnej verzii, len na jedno vylepšenie základnej funkcionality pre jednu tabuľu. Trello poskytuje natívne aplikácie pre platformy: Android, iOS, Windows a webovú aplikáciu. Pomocou aplikácii je možné s ním pracovať aj offline, kedy sa zmeny prejavia pri opätovnom pripojení na server. Trello neumožňuje používať vlastný server ako úložisko dát, čo nemusí vyhovovať firmám s vlastnou zabezpečenou lokálnou sieťou.

3.2 Breeze

Jedna z mála online aplikácií podporujúca Kanban tabuľu a zároveň poskytujúca plnú sadu nástrojov na manažment projektov. Nástroje zahrňujú: manažment rozpočtu a času, sledovania výdajov, míľnikov, projektové plánovanie a manažment portfólia. Má na prvý pohľad jednoduché užívateľské rozhranie, ktoré však v sebe skrýva širšiu funkcionalitu oproti väčšine konkurentov. Sledovanie úloh, automatické vytváranie úloh z emailu, zoznam celkovej aktivity, kalendáre projektov, možnosť zálohovania pomocou exportovania dát v HTML, JSON formáte, závislosti úloh, import úloh z tabuľkových formátov, integrácia Google Drive, Dropbox, Slack, GitHub a mnoho ďalších [3]. Okrem integrácie platforiem tretích strán Breeze obsahuje všetku funkcionalitu v sebe, nie je nutné si ju voliť. Je to platená aplikácia a cena sa odvíja od počtu užívateľov, no nijako ich neobmedzuje.

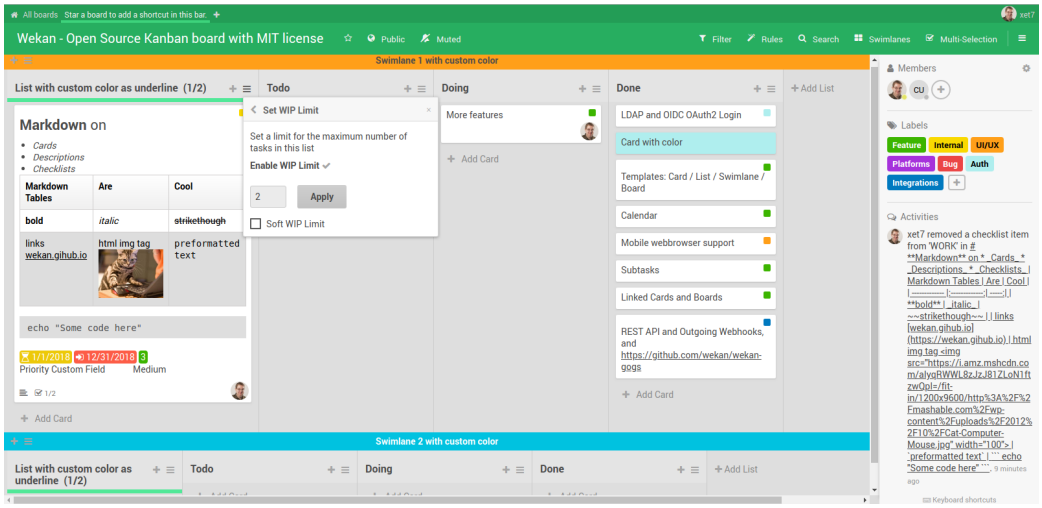


Obr. 3.2: Ukážka kalendára úloh z aplikácie Breeze. Aplikácia poskytuje viacero manažérskych nástrojov na správu projektov.

[1] Application Programming Interface – `https://it-slovnik.cz/pojem/api`
[2] IFTTT – `https://ifttt.com/`

3.3 Wekan

Hneď na úvod treba spomenúť, že aplikácia je pod MIT[3] licenciou, takže je otvorená komunite. Má automatické akcie, notifikácie, import dát aj z aplikácie Trello, REST API, preklady do viac ako 50 jazykov, podporu markdown syntax. Je implementovaná na modernej a rýchlej platforme Meteor. Je potrebná inštalácia, ktorú je možné spraviť aj pomocou platforiem Docker či Sandstorm. Podporuje integráciu platforiem Slack, Rocket.chat. [20] Aplikácia je koncipovaná ako celok a preto je integrácia nových súčastí možná po dohode s vývojármi aplikácie, inak hrozí že nová verzia aplikácie nebude so zákazníkom vytvorenou funkcionalitou kompatibilná. Užívateľské rozhranie je prijateľné, vyzdvihnúť sa dá veľmi intuitívne filtrovanie úloh.



Obr. 3.3: Vizualizácia práce v aplikácii Wekan. Aplikácia je vyvíjaná komunitou autorov. Vpravo je možné vidieť použitie štítkov na jednotlivé úlohy. Podľa štítkov je možné úlohy filtrovať.

3.4 Kanboard.org

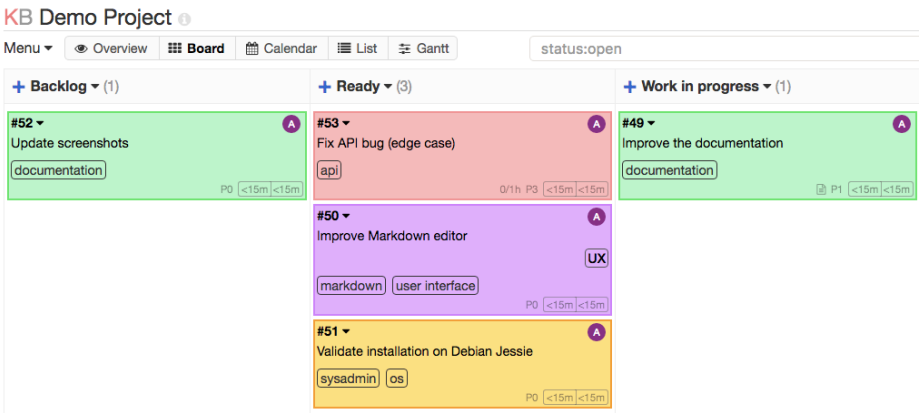
Je to aplikácia otvorená komunite, teda zadarmo. Hlavným znakom aplikácie je minimalizmus 3.4, či sa jedná o grafickú stránku užívateľského rozhrania, alebo množstvo funkcii. Funkcionalita je v prípade Kanboard.org rozšíriteľná pomocou modulov, buď oficiálnych, priamo v aplikácii, alebo vlastných. Aplikácia má výbornú dokumentáciu zameranú ako na používanie aplikácie, tak na vývoj jadra či modulov. Kanboard.org nepoužíva žiadne online knižnice, takže je plne použiteľný aj na lokálnom serveri bez prístupu k internetu. V základe podporuje automatické akcie, markdown syntax, notifikácie, základné analytické grafy, sledovanie času úloh aj podúloh, RSS[4], vytváranie vlastných užívateľských rolí projektu. Pomocou oficiálnych modulov je možné ľubovolne aplikáciu rozšíriť o: prepojenie na GitHub, Bitbucket, Slack, HipChat, ukladanie dát na serveri od firmy Amazon, kalendár a jeho prepojenie na kalendáre od firiem Google, Apple, Microsoft, Mozilla, rozpočty

[3]Massachusetts Institute of Technology – skratka vznikla podľa inštitútu kde vznikla. Jej vysvetlenie je možné nájsť tu: `https://www.kulman.sk/sk/content/maly-sprievodca-open-source-licenciami/`

[4]Rich Site Summary – je zaužívaná technologická skratka. Tu je popis technológie: `https://cs.wikipedia.org/wiki/RSS`

a sledovanie financií projektu, míľniky, overovanie prihlásenia pomocou SMS kódu a ďalšie funkcionality. Inštaláciu je možné previesť manuálne, alebo pomocou aplikácie Docker. [10]

Za najväčšiu výhodu aplikácie sa dá považovať veľmi minimalistický, ale rýchly základ aplikácie, ktorý je možné rozšíriť. Vlastné moduly je možné používať bez toho, aby boli zaregistrované tvorcami aplikácie, čo umožňuje rozšíriť funkcionalitu programu, ktorá je presne na mieru, alebo z dôvodu firemného tajomstva nemôže byť zverejnená.



Obr. 3.4: Časť Kanban tabule z aplikácie Kanboard.org. Aplikácia sa vyznačuje minimalistickým dizajnom.

Kapitola 4

Použité technológie na vývoj modulu

Kapitola uvádza technológie, ktoré je nutné alebo prínosné ovládať pre implementáciu vstavaného modulu do aplikácie Kanboard.org. Technológie sú rozdelené do štyroch skupín. Prvé sú technológie použité pri vývoji aplikácie. Ďalšie sú technológie používané pri návrhu a používaní databázy. Tretia skupina popísaná v sekcii 4.3, sú technológie použité na vývoj používateľského rozhrania. Posledná skupina sú technológie použité pri vývoji serverovej časti modulu.

4.1 Technológie uľahčujúce vývoj

Technológie uvedené v tejto sekcii nie je nutné pri vývoji modulu použiť, ale pri vývoji tohto modulu sú použité. Sú to technológie, ktoré vývoj zľahčili pred začiatkom, ako aj počas implementácie.

Cookiecutter
Je multiplatformová aplikácia príkazového riadku schopná vytvoriť štruktúru projektu podľa šablóny [16]. Program je odporúčaný priamo v dokumentácii Kanboard.org. Šablóna štruktúry na tvorbu modulu je už vytvorená. Jediné čo stačí spraviť, je program nainštalovať a spustiť s parametrom *gh:kanboard/cookiecutter-plugin*. Program sa spýta na niekoľko údajov o novom module, na základe ktorých vygeneruje štruktúru. Súčasťou štruktúry je aj základ registračného súboru modulu, ktorý je najpodstatnejšou súčasťou každého modulu aplikácie Kanboard.org.

Docker
Je technológia na vytvorenie virtuálneho prostredia pomocou kontajnerov. Pri vývoji je technológia Docker používaná na vytvorenie prostredia obsahujúceho najmenej všetky minimálne požiadavky na spustenie hlavnej aplikácie na lokálnom serveri. Špecifikácia prostredia je nastavená pomocou konfiguračného súboru. Výhodou je, že o všetky nutné inštalácie sa postará Docker a rovnaké vývojové prostredie je možné vytvoriť kdekoľvek po nainštalovaní a spustení technológie s vytvoreným konfiguračným súborom. Konfiguračný súbor je možné nájsť v dokumentácii Kanboard.org.

Tracy debuger

Je nástroj poskytujúci vizualizáciu chýb urobených programátorom v jazyku PHP [17]. Tracy pri chybe zobrazí miesto v kóde, kde pravdepodobne nastala chyba a taktiež predchádzajúce volania, ktoré viedli k chybnej časti kódu. Zároveň pri bezchybnom behu meria čas potrebný na načítanie stránky, počet databázových požiadaviek, spotrebovanú pamäť, ktoré môžu slúžiť na optimalizáciu tvorenej aplikácie.

Git

Je voľne dostupný systém na správu verzií. Je multiplatformový a má mnoho konkrétnych implementácií, ktoré sa od seba líšia. Git ukladá projekty do repozitárov. Tie sa delia na lokálne a vzdialené. Zmeny uložené lokálne sa delia na verzie pomocou príkazu *commit*. Jednotlivé verzie je možné preniesť na vzdialené úložisko príkazom *push*. Výhodou verzií je, že je možné sa k akejkoľvek verzii spätne vrátiť. Vzdialené úložisko slúži ako záloha a miesto, kde je projekt dostupný pri vývoji z viacerých počítačov.

Posledná vyvíjaná verzia hlavnej aplikácie je uložená v službe GitHub. Služba je poskytovaná zadarmo a používa sa práve pri vývoji aplikácie viacerými programátormi. Výsledný modul je pre verejné použitie, dostupný práve v tejto službe.

4.2 Databázové technológie

Táto sekcia popisuje databázové technológie používané vo vnútri hlavnej aplikácie. Dôležitá je znalosť SQLite a picoDB, ktorým sa programátor počas implementácie nevyhne.

SQLite

Je softvérový balíček poskytujúci manažment relačnej databázy. Okrem uloženia dát a ich manažmentu umožňuje technológia spracovávanie komplexných databázových požiadavkov, ktoré kombinujú dáta z viacerých tabuliek a generujú ich zhrnutie. SQLite nevyžaduje separátny server, pristupuje k uloženým súborom priamo. Na používanie nie je nutná konfigurácia. [12]

Používa veľmi málo pamäte, takže je často používaná v mobilných aj webových aplikáciách. Tým, že je celá inštancia databázy uložená v jednom súbore umožňuje jednoduchý transport dát.

picoDB

Je minimalistická knižnica na vytváranie databázových požiadaviek v PHP. Je voľne dostupná pod MIT licenciou. Podporuje databázové systémy: SQLite, Mssql, Mysql, Postgresql. Umožňuje vytváranie a použitie migrácie schém databázy, čo je využité aj v hlavnej aplikácii. Migrácie sú organizované do verzií. Pri vytvorení novej aktualizácie aplikácie vyžadujúcej zmenu v databázovej štruktúre, je vytvorená nová verzia migrácie.

4.3 Technológie použité na strane klienta

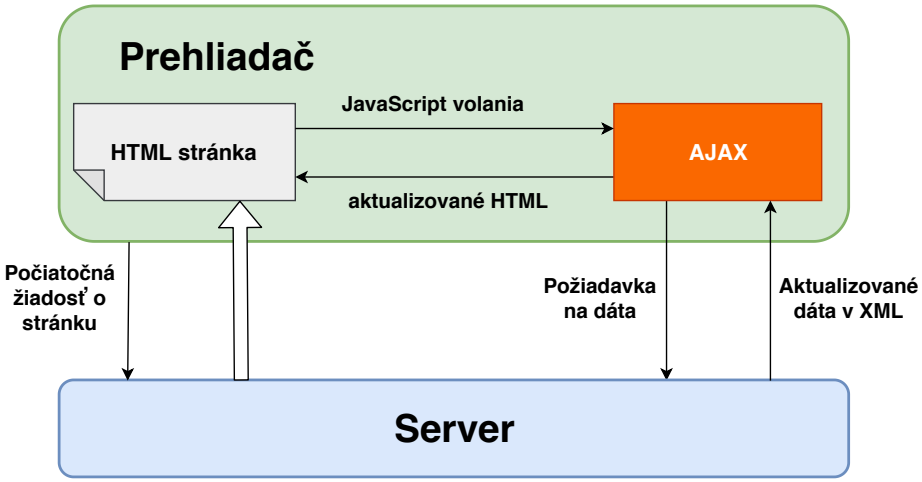
Sekcia popisuje použité technológie pri implementácii užívateľského rozhrania. Boli použité aj technológie HTML a CSS, ale sú pre vývoj webovej aplikácie absolútne esenciálne a netreba ich hlbšie popisovať.

JavaScript

Je interpretovaný, skriptovací jazyk, ktorý je načítaný spolu s obsahom webovej stránky, ale vykonáva sa na strane klienta. Podporuje objektovo orientované programovanie, aj funkcionálne programovanie. Tým, že je vykonaný na strane klienta, je možné meniť obsah načítanej stránky bez nutnosti serverovej podpory. Pomocou tejto technológie je možné zostaviť rozloženie a veľkosť HTML elementov priamo podľa zariadenia, na ktorom si klient webovú aplikáciu spustil. Rýchlymi reakciami na interakciu užívateľa vytvára dojem dynamiky webovej stránky. [8]

AJAX

Nie je technológia, ale pojem popisujúci spojenie viacerých existujúcich technológii ako: HTML, CSS, Javascript, XML a XMLGttpRequest objekt. Spojením týchto a ďalších technológií v AJAX modeli umožňuje webovým aplikáciam rýchle, postupné zmeny v užívateľskom rozhraní, bez nutnosti načítania celej stránky, s možnosťou serverovej podpory. [1] Názov knižnice je skratka *Asynchronous JavaScript and XML*. Aj keď posledným písmenom skratky je X ako XML, v súčastnosti sa väčšinou používa technológia JSON použitá aj v module. JSON je dátový formát, vytvorený ako podmnožina syntax technológie JavaScript. [9] Podporuje ho viacero programovacích jazykov, medzi nimi aj PHP, čo sa využíva v implementácii modulu.



Obr. 4.1: Tok požiadaviek a dát medzi prehliadačom sa serverom. Zatiaľ čo pri načítaní celej stránky ide o veľké množstvo dát, ktoré je nutné poslať, môžu AJAX rutiny žiadať len dáta, ktoré sa zmenili.

jQuery

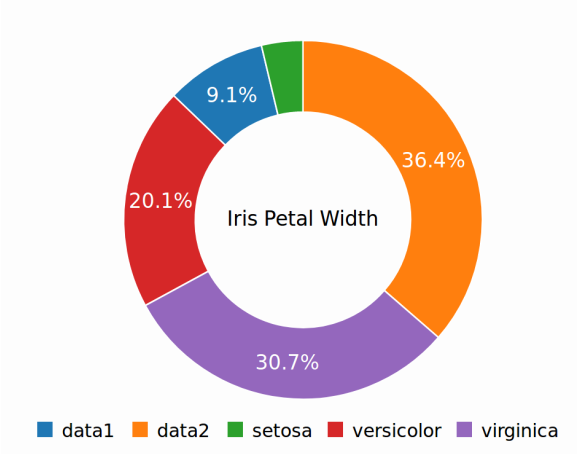
Je knižnica jazyka JavaScript, ktorej hlavnou časťou je selektor HTML elementov. Selektor je implementovaný pomocou takzvanej jQuery funkcie *jQuery()*. Vo väčšine implementácií je reprezentovaná skratkou *$()*. Parametrom funkcie je text v syntax CSS, podľa ktorého sa hľadá DOM[1] element. Ďalším parametrom môže byť kontext, v ktorom sa vyhľadáva. Pokiaľ nie je zadaný, vyhľadávanie sa začne od koreňového uzlu DOM štruktúry.

[1] Document Object Model - je kolekcia objektov a uzlov, ktoré tvoria HTML, XHTML a XML dokumenty

Na vybraný DOM element je potom možné aplikovať jQuery metódy, ktoré je možné skoro všetky postupne reťaziť. Metódy poskytujú jednoduché presuny a modifikácie DOM elementov, syntax na obsluhu udalostí prehliadača, prístup ku všetkým atribútom elementu, animácie a AJAX požiadavky. [11]

c3

Je knižnica na vytváranie grafov, napísaná v jazyku Javascript, voľne dostupná pod MIT licenciou [4]. Poskytuje možnosť na definovanie vlastných štýlov, pridávaním tried ku každému elementu. Disponuje vlastným API, ktoré medzi iným umožňuje zmeny v grafe po tom, čo bol vykreslený. Výhodou je možnosť používania knižnice bez nutnosti pripojenia k internetu. Dáta do grafov je možné nahrávať aj vo formáte JSON.



Obr. 4.2: Ukážka testovacieho grafu vytvoreného pomocou knižnice c3.js. Pomerne menej známa knižnica sa svojimi možnosťami vyrovná aj knižnici Google Charts, ktorú nie je možné používať bez pripojenia k internetu.

Markdown

Je jednoduchý značkovací jazyk, ktorý je preložiteľný do HTML. Umožňuje pomocou značiek definovať odseky, nadpisy rôznych stupňov, meniť rezy písma, tvoriť číslované aj nečíslované zoznamy, vytvárať riadkové referencie, zvýraznenie častí s kódom, ai. [6] V rôznych implementáciách poskytuje rôzne množstvo funkcií, ktoré môžu byť konkrétne prispôsobené. V hlavnej aplikácii sa dá pomocou Markdown syntax v texte odkazovať na iných používateľov či konkrétne úlohy.

4.4 Serverové technológie

Aplikácia Kanboard.org je napísaná v jazyku PHP. Tento fakt predurčuje serverovú časť modulu k implementácii v tomto jazyku. Písať serverovú časť inom jazyku by spôsobilo viaceré nežiaduce re-implementácie kódu hlavnej aplikácie. Hlavná aplikácia je napísaná bez použitia rámca jazyka. Využíva však koncept MVC popísaný neskôr.

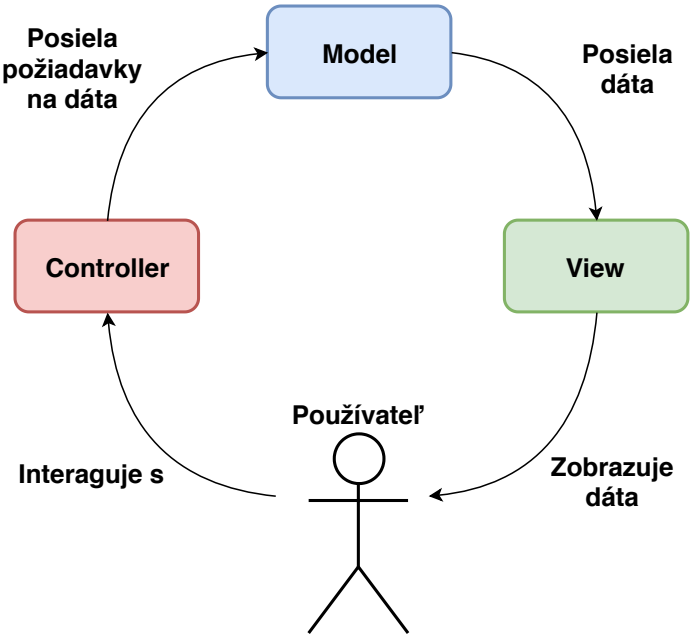
PHP je skriptovací jazyk, ktorého posledná verzia je PHP 7. Má zásadné zmeny v spätnej kompatibilite, pretože sa sústredí na zlepšenie výkonu a rýchlosti. [13] To, že je PHP verzia

7 schopná konkurovať ostatným jazykom dokazuje fakt, že je nasadená u takých gigantov ako sú: Facebook, Wikipedia, Drupal, Wordpress [15].

Model-View-Controller

Je architektúra najčastejšie používaná na vývoj užívateľských rozhraní a webových aplikácií. Je súčasťou mnohých rámcov jazyka PHP, ale je implementovateľná aj v iných jazykoch. Delí vývoj aplikácie na tri oddelené komponenty:

- Model – definuje aplikačné dáta a ich chovanie. Nie je zodpovedný za formátovanie, alebo prezentovanie dát.
- View – zobrazuje užívateľovi dáta poslané modelom, teda užívateľské rozhranie.
- Controller – je zodpovedný za chovanie aplikácie na základe definovaných udalostí alebo interakcie užívateľa tým, že koordinuje *model* k produkcii správneho výstupu.



Obr. 4.3: Princíp architektúry MVC. Používateľovi zobrazuje dáta komponenta *view*, zatiaľ čo jeho akcie spracováva komponenta *controller*. Ten potom posiela požiadavky na uloženie, zmazanie, modifikovanie alebo zobrazenie dát. *Model* posiela požadované dáta na zobrazenie.

Kapitola 5

Návrh a implementácia nového modulu

Táto kapitola popisuje kľúčové časti návrhu a implementácie modulu. Pre vytýčenie používania modulu je tu model prípadov použitia. Návrh databázovej štruktúry určí, aké dáta sa budú ukladať a ako budú spolu previazané. Návrh užívateľského rozhrania ukáže ako dáta a funkcionalitu zobrazovať.

5.1 Prípady použitia

Táto časť návrhu hovorí o skupinách užívateľov, ktorí budú môcť s modulom pracovať a ich možnosti, ako ho môžu používať. Keďže základné rozdelenie užívateľov pri práci s aplikáciou Kanboard.org je do pozícií (rolí užívateľov) v jednotlivých projektoch, delím užívateľov podľa ich úlohy v projekte. Okrem administrátora aplikácie, ktorý má prístup kamkoľvek, ponúka aplikácia v základnom nastavení tri role používateľa priradeného k projektu. Sú to:

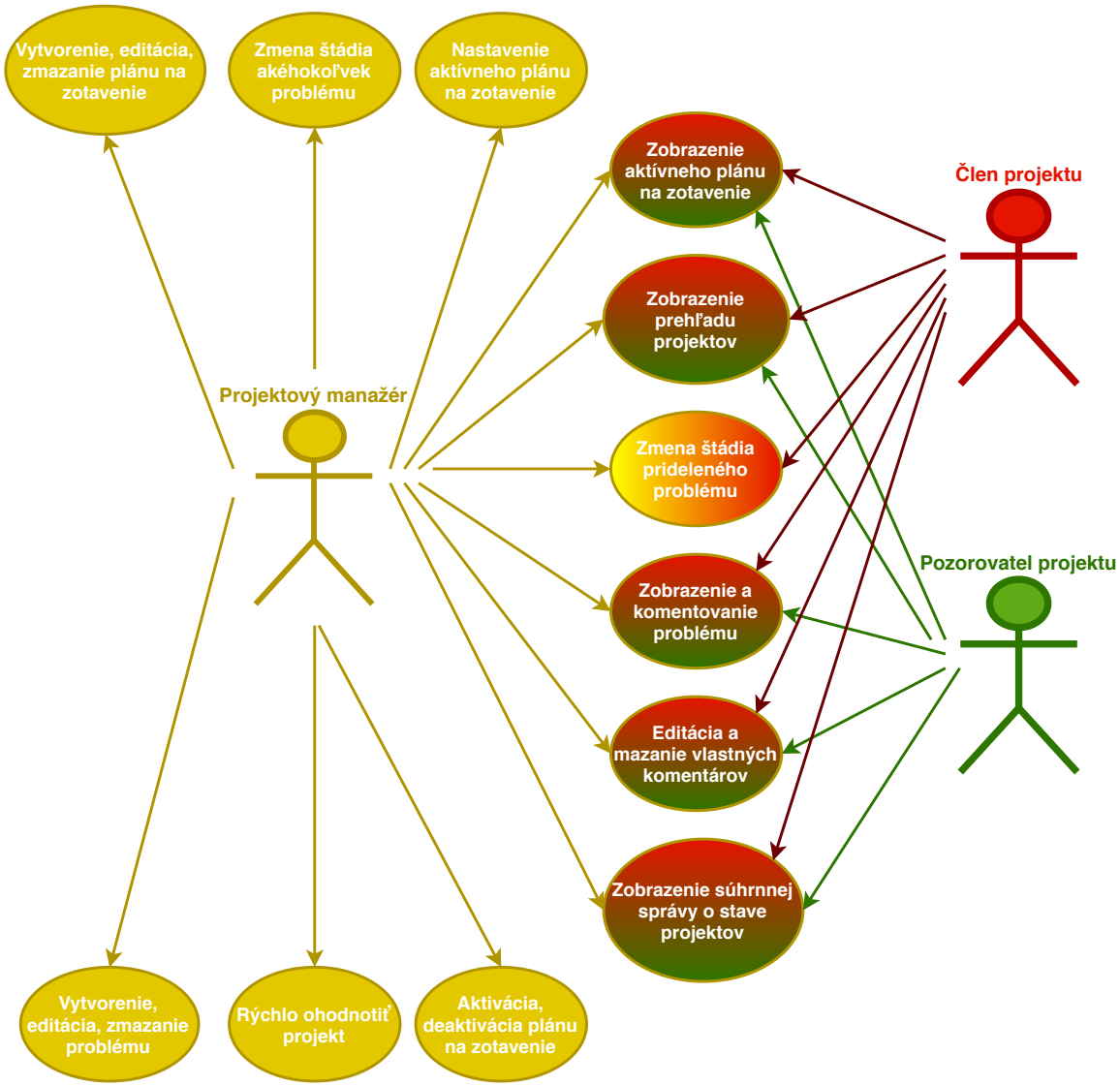
- projektový manažér (project manager),
- člen projektu (project member),
- pozorovateľ projektu (project viewer).

Naviac umožňuje aplikácia aj vytváranie vlastných rolí, ktoré sú základnými právami prístupu na rovnakej úrovni ako členovia projektu, preto sú ich práva zahrnuté do tejto skupiny.

Modul rozširuje funkcionalitu aplikácie o manažment problémov jednotlivých projektov a prehľadné zobrazovanie ich stavu. Dôraz je kladený na možnosť rýchleho či podrobnejšieho zhodnotenia projektu a v kritickom stave vytvorenia plánu na zotavenie. Prípady použitia modulu teda sú:

Zobrazenie prehľadu projektov
: Zobrazí zoznam projektov ku ktorým má užívateľ prístup, ich základné informácie, farebne odlíšený stav projektu, problémové úlohy projektu, informácie o problémových úlohách.

Rýchle ohodnotenie projektu
: Umožní nastavenie stavu v akom sa projekt nachádza. Základná aplikácia neumožňuje hodnotenie projektu ako celku. Vďaka takémuto hodnoteniu bude možné jednoducho odlíšiť problémové projekty od správne fungujúcich.



Obr. 5.1: Diagram prípadov použitia. Modul používa, rovnako ako aplikácia, tri roly užívateľa v projekte.

Vytvorenie, editácia, zmazanie plánu na zotavenie
: Je možnosť vytvoriť podrobnejšiu správu, alebo plán na zotavenie projektu. Správa pomôže k získaniu celkového obrazu o stave projektu.

Vytvorenie, editácia, zmazanie problému
: Je systém zaznamenávania objavujúcich sa problémov za behu projektu. Vytvára sa zoznam problémov.

Zmena stavu problémov
: Pri problémoch je dôležité informovať sa, v akom stave sa nachádzajú, aby užívateľ čakajúci na vyriešenie problému vedel, čo sa s problémom deje.

Nastavenie aktívneho plánu na zotavenie
Je možnosť odlíšenia aktuálneho plánu na zotavenie od starších, alebo ešte len pripravovaných.

Zobrazenie súhrnnej správy o stave projektov
Zobrazí zoznam pridelených projektov, aj ich správy (plány na zotavenie) na jednom mieste. Pomôže identifikovať, v čom sa problémy projektov zhodujú a umožní navrhnúť spoločný postup ich riešenia.

Vytvorenie, editácia, zmazanie komentára k problému
Problémy budú veľakrát vyžadovať spoluprácu viacerých členov tímu. Komentáre problémov dávajú možnosť kolaborácie a zostanú uložené s problémom, takže bude jednoduchšie identifikovať ako tím k riešeniu problému pristúpil, alebo poskytujú priestor na komentovanie riešenia problému. Princíp diskusie k problémom je inšpirovaný službou GitHub popísanej v sekcii 4.1.

Najväčšie možnosti používania modulu majú projektoví manažéri, ktorí za projekt zodpovedajú vyššiemu manažmentu. Majú možnosť hodnotiť projekt, ako aj vytvárať plány na zotavenie a rozhodujú o tom, ktorý plán na zotavenie je prístupný ostatným členom projektu.

5.2 Databázová štruktúra

Ďalším krokom v návrhu, po stanovení čo bude môcť užívateľ v module robiť, bolo nutné navrhnúť aké dáta budú ukladané, aby zabezpečili navrhovanú funkcionalitu. Ako úložisko dát používa aplikácia relačnú databázu. Jej navrhovanú štruktúru ukazuje obrázok 5.2.

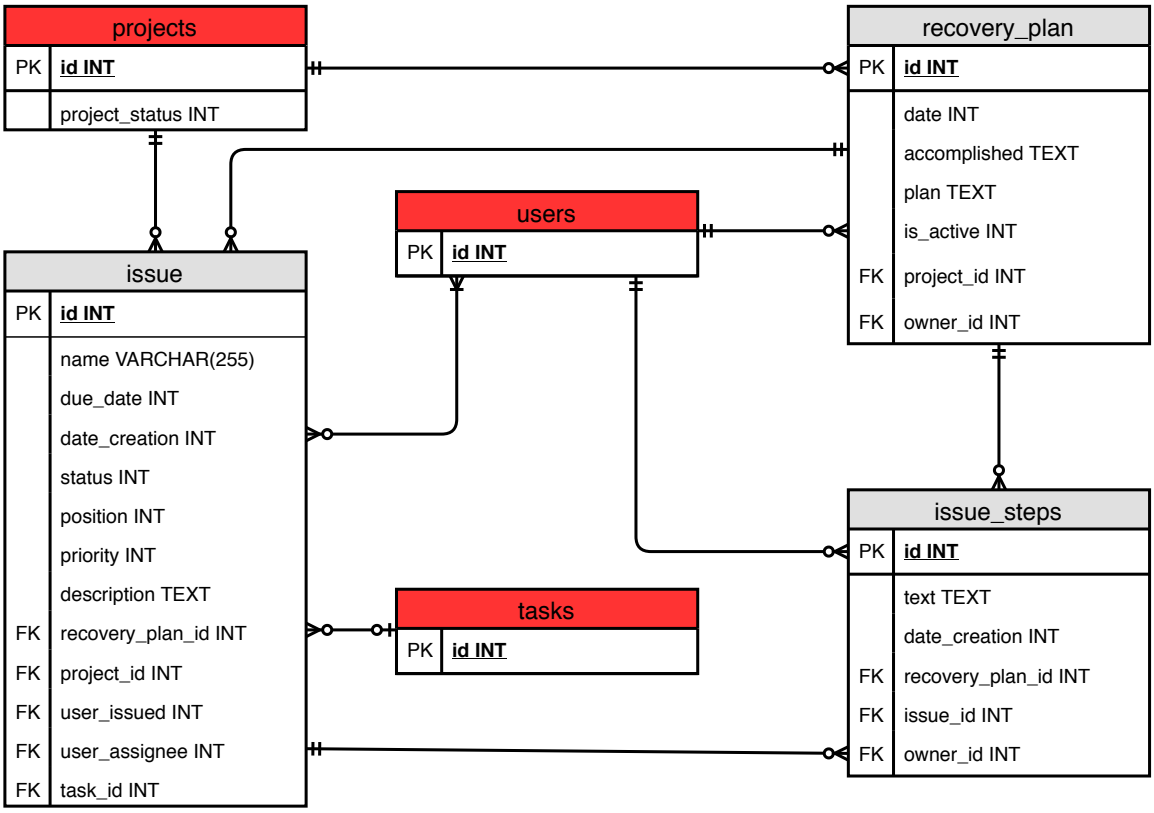
Projects
Je tabuľkou pôvodnej aplikácie. Uchováva základné informácie o projekte, ako názov, popis, kedy bol vytvorený, kedy je termín ukončenia a ďalšie. Aby sa dalo projekt hodnotiť, je nutné do tabuľky pridať položku, ktorá bude hodnotenie ukladať.

Users
Tabuľka pôvodnej aplikácie, ktorá ukladá informácie o užívateľoch aplikácie. Do tejto tabuľky nič pridané nie je. Je uvedená z dôvodu prepojení nových tabuliek na túto tabuľku.

Tasks
Entita obsahujúca informácie o konkrétnej úlohe Kanboard.org tabule. Je už vytvorená v pôvodnej aplikácii a je popísaná kvôli možnému vzťahu konkrétnej úlohy a vzniknutého problému.

Recovery plan
Nová tabuľka uchovávajúca dáta plánu na zotavenie projektu. Obsahuje základné informácie ako: kto plán vytvoril, kedy bol plán vytvorený, či je to aktuálny plán, čo sa projektu podarilo spraviť a aký je plán na nasledujúce obdobie.

Issue
Entita obsahujúca problémy projektu. Pri problémoch je zaznamenané: kto problém našiel, kto je zodpovedný za riešenie, názov, termín dokedy má byť problém vyriešený, kedy bol problém nájdený, popis problému, pozícia v tabuľke a ďalšie.



Obr. 5.2: Schéma relačnej databázy. Schéma zobrazuje štruktúru navrhovaných tabuliek a ich relácie. Tabuľky projects, users, tasks, vyznačené červenou, sú pôvodné tabuľky aplikácie Kanboard.org. Zvyšné entity sú navrhnuté pre ukladanie dát modulu.

Issue steps
: Je tabuľka krokov problému. Uchováva komentáre, diskusiu k problému. Hlavnou časťou je telo textu uložené v poli *text*.

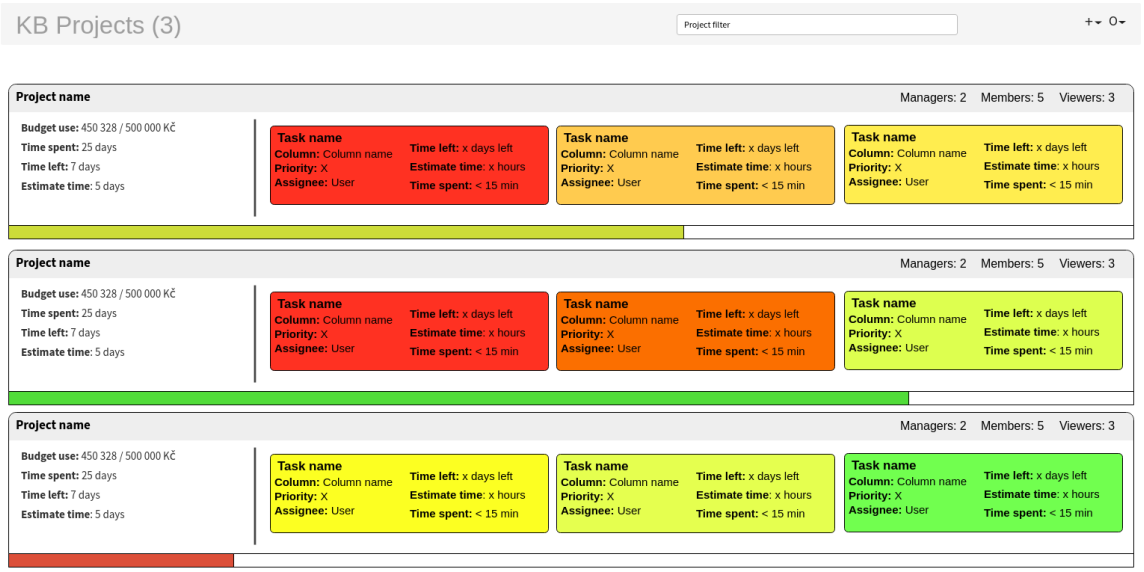
5.3 Návrh užívateľského rozhrania

Na návrh častí užívateľského rozhrania sú použité blokové drôtené modely, pre vytvorenie funkčného prototypu. Modely slúžia na vizualizáciu a ujasnenie funkcionality bezprostredne pred začatím implementácie modulu. Pri tomto návrhu hrá hlavnú rolu konzistencia modulu s hlavnou aplikáciou. Je to preto, aby si užívateľ pri používaním modulu nepripadal, akoby používal úplne inú aplikáciu, s iným užívateľským rozhraním. Pôvodná aplikácia je graficky veľmi minimalistická a zrozumiteľná, čo je aj cieľom návrhu modulu. Na návrh častí užívateľského rozhrania sú použité blokové drôtené modely, pre vytvorenie funkčného prototypu. Modely slúžia na vizualizáciu a ujasnenie funkcionality bezprostredne pred začatím implementácie modulu. Pri tomto návrhu hrá hlavnú rolu konzistencia modulu s hlavnou aplikáciou. Je to preto aby si užívateľ pri používaním modulu nepripadal, akoby používal úplne inú aplikáciu, s iným užívateľským rozhraním. Pôvodná aplikácia je graficky veľmi minimalistická a zrozumiteľná, čo je aj cieľom návrhu modulu.

Za istý deficit hlavnej aplikácie by sa dalo považovať to, ako informuje používateľa o priebehu akcie, ktorú vyvolal. Hlavná aplikácia síce túto spätnú väzbu posiela, no tá sa

zobrazuje v spodnej časti obrazovky, kde sa vo väčšine stránok aplikácie nenachádza nič a používateľ tomuto priestoru nevenuje žiadnu pozornosť.

Drôtené modely sú tvorené pomocou bezplatnej webovej aplikácie MockFlow. Na obrázku 5.3 je úplne prvý model zobrazenia prehľadu projektov.



Obr. 5.3: Drôtený model prehľadu projektov s popisom projektu, problémovými úlohami a časovej osi projektu. Stav projektov je farebne odlíšený.

5.4 Štruktúra modulu

Na vytvorenie základnej kostry programu bol použitý program Cookiecutter popísaný v sekcii 4.1. Hlavné časti modulu sa nachádzajú v priečinkoch *Controller*, *Model* a *Template* podľa MVC modelu. Súbory dynamicky ovplyvňujúce modul a jeho vizuálnu stránku sa nachádzajú v priečinku *Assets*. Novo pridanými priečinkami v štruktúre modulu sú priečinky *Validator* a *Helper*. Ich súbory budú popísané ďalej v popise implementácie.

5.5 Registračný súbor modulu

Tento súbor je základom každého modulu. Obsahuje triedu *Plugin*, ktorá rozširuje triedu *Base*. Táto trieda má viacero metód, ktoré vracajú základné informácie o module, ako: názov modulu, meno autora, popis modulu, verziu modulu, odkaz na webovú stránku, kde je modul dostupný. Pre fungovanie modulu sú však dôležitejšie metódy:

initialize() – registrujú sa v nej súbory, ktoré nie sú súčasťou MVC štruktúry, obsahujúce kaskádové štýly, skripty spúšťané na strane klienta a ďalšie súbory, ktoré rozširujú triedy hlavnej aplikácie a používajú sa naprieč ostatnými súbormi modulu. Ďalej sa môže definovať prepojenie modulu s aplikáciou, alebo definovať prístupové práva užívateľov.

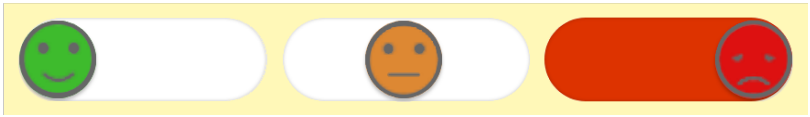
getClasses() – slúži na registrovanie tried, ktoré potrebujú byť uložené do Dependency Injection kontajneru. Ich definovaním sa zabezpečí, že sa vytvorí len jedna inštancia triedy a budú dostupné v celej aplikácii.

5.6 Rýchle ohodnotenie projektu

Deficitom aplikácie Kanboard.org je, že neposkytuje možnosť ohodnotenia projektu. Ak ponúka zoznam projektov, zobrazuje v ňom buď distribúciu úloh do stĺpčekov tabuľky, alebo len dátumy začatia a ukončenia projektu či zoznam členov. Pokiaľ by chcel užívateľ zodpovedný za viacero tímov, pracujúcich na množstve projektov, zistiť či je všetko v poriadku, musí preveriť projekt po projekte.

Hodnotenie projektov je implementované pomocou troj-stavového zaškrtávacieho políčka. Políčko je vytvorené pomocou externého doplnku *jToggler*[1]. Doplnok bol upravený, keďže neumožňoval identifikáciu konkrétneho políčka pri zobrazení viacerých políčok na jednej stránke. Po zaškrtnutí iného stavu políčka sa spustí udalosť, ktorá pomocou technológie AJAX pošle požiadavku na uloženie konkrétneho stavu projektu na server.

Dizajn políčka bol tiež zmenený. Stavy sú odlíšené farebne na: zelenú, žltú, červenú, podľa farieb semaforu, ktoré intuitívne vyjadrujú hodnotenie. Farby boli doplnené piktogramom smajlíka, ktorý môže napovedať stav projektu pre ľudí s poruchou farebného videnia.



Obr. 5.4: Troj-stavové zaškrtávacie políčko na rýchle ohodnotenie projektu, so zmeneným dizajnom. Nachádzajú sa tu farby pripomínajúce semafor doplnené piktogramom smajlíka. Slúži na jednoduché vyjadrenie stavu v akom sa projekt nachádza.

5.7 Zobrazovanie stavu projektov

Všetky projekty užívateľa sú zobrazené na jednom mieste, čo mu dáva celkový prehľad nad jeho prácou. Pri projektoch sú zobrazené ich základné informácie, graf distribúcie úloh členom projektu, graf zobrazujúci rozloženie úloh do stĺpčekov tabule, stav projektu vyjadrený pomocou troj-stavového políčka, odkaz na vytvorenie, alebo vytvorený plán na zotavenie, vybrané úlohy projektu a ukazovateľ časovej osi, kde sa aktuálne projekt nachádza.

Dáta pre grafy sú načítané z databázy a uložené vo formáte JSON do dátového atribútu obalového HTML elementu. Po načítaní stránky sa spustí opakovacia metóda, ktorá postupne prejde všetky obalové HTML elementy, a pomocou anonymnej funkcie nastaví vstup metódy na generovanie grafov knižnice *c3* 5.1.

Úlohy, ktoré sa zobrazujú pri projekte, reflektujú či tím plní úlohy načas. Nedodržiavanie termínov úloh je dôležitý ukazovateľ „zdravia“ projektu. Zobrazené sú úlohy, ktoré sú v danom momente už po termíne ukončenia, ako aj úlohy, ktorých rozdiel odhadovaného času na splnenie a stráveného času presahujú termín odovzdania. Poradie úloh závisí od ich nastavenej priority a náročnosti.

[1] jToggler – `https://github.com/sinneren/jToggler`

```
$.each($(".user_chart"), function () {
    c3.generate({
        bindto: '#user_chart'+$(this).data('chartid'),
        data: {
            columns: [
                ['active tasks'].concat($(this).data('params').tasks),
                ['closed tasks'].concat($(this).data('params').closed_tasks)
            ],
            type: 'bar'
        },
        ...
```

Výpis 5.1: Generovanie viacerých grafov na jednej stránke. Identifikátor grafu je uložený v dátovom atribúte *chartid* a hodnoty grafu v atribúte *params*. Formát dát JSON nie je nutné v jazyku Javascript rozoberať a je možné ku kľúčom pristúpiť bodkovou notáciou.

Stránkovanie

Zobrazovanie všetkých projektov na jednom mieste vyžaduje implementáciu stránkovania. V dokumentácii aplikácie nie je stránkovanie spomenuté, aj keď sa viac krát v aplikácii vyskytuje. Zabezpečuje ho Helper *Paginator*, ktorý sa bude dať použiť aj v module aplikácie. Hlavnými metódami sú *setUrl* a *setQuery*. Pri nastavení URL je treba v parametroch nastaviť, meno modulu a názov všetkých parametrov predaných pomocou URL, inak načítanie stránky skončí s výnimkou. Metóda *setQuery* prijíma jeden parameter čo je inštancia *PicoDB table*, teda neodoslaná požiadavka na databázu.

```
$paginator = $this->paginator
    ->setUrl('StatusController', 'index', array('plugin' => 'Status'))
    ->setMax($paginator_limit)
    ->setOrder('name')
    ->setQuery($query)
    ->calculate();
```

Výpis 5.2: Použitie stránkovania pri zobrazení stavu projektov.

Vyhľadávanie pomocou filtrov

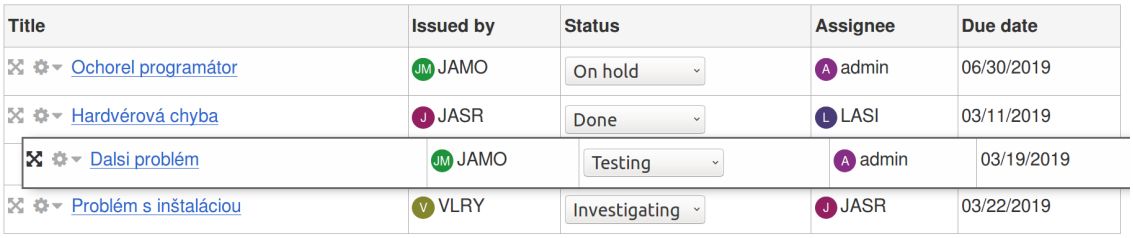
Aby bolo možné filtrovať iba projekty, ktoré majú zlé hodnotenie, alebo pre vyhľadanie konkrétneho projektu, je implementované vyhľadávanie pomocou pokročilej syntax. Syntax je kvôli konzistentnosti prebratá z hlavnej aplikácie. Implementácia je založená na rozbore syntax na jednotlivé kľúčové slová a ich hodnotu. Z nich sú vytvorené podmienky, ktoré sú pripojené k inštancií *PicoDB table*. Tá končí ako vstupný parameter stránkovania.

5.8 Plán na zotavenie

Plán na zotavenie má tri základné časti: popis dosiahnutých bodov projektu, plán čo ešte treba urobiť a zoznam problémov, ktoré projekt spomaľujú. Prvé dve časti sú implementované pomocou Markdown editorov (sekcia 4.3) poskytujúcich podporu štruktúrovaného textu a odkazov. Manažment problémov je implementovaný ako dynamická tabuľka. Riadky je možné medzi sebou presúvať (obrázok 5.5), sú spätne editovateľné. Technológie AJAX

umožňuje okamžite meniť status problému. Problém je schopný odkazovať na aktívne úlohy projektu. Dá sa zaznamenať, kto problém objavil a kto je zodpovedný za jeho riešenie. Je možné nastaviť prioritu problému a termín ukončenia. Stupnica priority je zhodná so stupnicou priority úlohy projektu, pre ktorú je plán vytvorený. Na konci tabuľky je popis problému, ktorý by mohol byť veľmi rozsiahly, preto sa zobrazuje iba jeho úryvok.

Pre zaznamenanie postupu riešenia a možnú spoluprácu viacerých členov projektu na jednom probléme je implementovaný systém komentárov. Komentáre podporujú Markdown syntax, takže sa dajú použiť aj ako návod na riešenie problému. Prístup k písaniu komentárov má každý člen projektu. Spätná editácia a mazanie problémov je povolená len na vlastných príspevkoch používateľa.



Obr. 5.5: Ľavá časť tabuľky problémov pri presúvaní riadku. Tabuľka je naplnená testovacími dátami.

Modálne okná

V hlavnej aplikácii je veľká väčšina vstupov, alebo vstupných formulárov, zobrazovaná v modálnych oknách, otvorených z načítanej stránky. V rámci zachovania konzistencie to tak je aj v novom module. Vstupy sú použité pri implementácii vytvárania, editácie problémov a plánov na zotavenie. Sú pomocou nich implementované aj konfirmácie mazania, či aktivácie. Ich použitie znižuje počet nutných znovu načítaní stránky. Napríklad ak užívateľ omylom klikne na zobrazenie formulára, alebo si svoju akciu rozmyslí, jednoducho zavrie modálne okno a zostáva na pôvodnej stránke bez nutnosti znovunačítania obsahu.

Validácia formulárov

Pre zjednodušenie validácie vstupov formulárov som vytvoril triedy obsahujúce ich pravidlá. Triedy rozširujú triedu *BaseValidator* hlavnej aplikácie, ktorú žiaľ dokumentácia nespomína. Súbory sú pomenované podľa mnou vytvorených databázových tabuliek *recovery_plan* a *issue*, pretože väčšia časť pravidiel pre vstupy formulárov je pri práci s jednou tabuľkou rovnaká. Výstupom validácie je pole, kde prvý prvok poľa hovorí, či je validácia úspešná, alebo nie a v druhom prvku poľa sú chyby vstupov. Tieto triedy sú uložené v priečinku *Validator* spomenutom v sekcii 5.4.

```
1 class IssueValidator extends BaseValidator {
2     public function commonRules() {
3         return array(
4             new Validators\Integer('id',t('This value must be an integer')),
5             new Validators\Required('name',t('The issue name is required'))
6         );
7     }
```

```
8     public function validateCreation(array $values){
9         $v = new Validator($values, $this->commonRules());
10        return array(
11            $v->execute(),
12            $v->getErrors()
13        );
14    }
```

Výpis 5.3: Trieda IssueValidátor určená na validovanie vstupov formulárov. Metóda commonRules() vráti pole spoločných pravidiel pre všetky operácie. Pole je pre ukážku skrátené. Metóda validateCreation() je volaná po odoslaní formulára na vytvorenie problému.

5.9 Access Control List

Je zoznam prístupových práv užívateľa. Prístup je rozdelený podľa rolí užívateľov v projekte uvedenom v sekcii 5.1. Aplikácia ponúka pomerne jednoduché definovanie práv na prístup k metódam riadiacich tried. Definujú sa v registračnom súbore modulu. Stále je však nutné zamedziť zobrazovaniu odkazov a položiek menu, ktoré vedú k zakázaným častiam.

Kapitola 6

Iteratívne testovanie prototypu a postupné zlepšovanie

Skupine ľudí bol predložený modul a aplikácia na vyskúšanie. Vzorka ľudí bola síce obmedzená, no všetci sa zaoberajú vývojom softvéru a majú skúsenosti s prácou na projekte v tíme. Väčšine bol známy koncept Kanban tabule a časť z nich aj požívala niektoré konkurenčné aplikácie, no s aplikáciou Kanboard.org sa nikdy nestretli. Preto som počas testovania na otázky ohľadom fungovania aplikácie, nie samotného modulu, odpovedal.

Takto vyzeral zoznam úloh, ktoré mal užívateľ testujúci modul vykonať:

1. Otvor modul na zhodnotenie stavu projektov.
2. Nájdi kladne hodnotený projekt, ktorý má úlohy po termíne ukončenia a zmeň mu hodnotenie na zlé.
3. Pre tento projekt vytvor neaktívny plán na zotavenie.
4. Založ v ňom 3 problémy pričom vyplň viac ako polovicu vlastností problému.
5. Napíš komentár k druhému problému.
6. Presun tretí problém na vrch tabuľky.
7. Zmaž problém na druhej pozícii.
8. Aktivuj plán na zotavenie.
9. Zobraz správu o projektoch a nájdi svoj plán.
10. Zmeň názov druhého problému.
11. Zmaž komentár, ktorý si vytvoril k druhému problému.
12. Deaktivuj plán na zotavenie.
13. Zmaž plán na zotavenie.

6.1 Prototyp

Prototyp obsahoval viaceré chyby, ktoré obmedzovali pohodlné používanie aplikácie. Používatelia väčšinovo nerozumeli princípu, že sa všetká funkcionalita modulu vyskytuje v jednej časti. Často preto zachádzali do hlavnej aplikácie, kde nemohli splniť úlohy testu. Problém robil aj fakt, že v aplikácii chýbali tlačidlá, ktoré by používateľa dostali na predchádzajúcu stránku. Napríklad po vytvorení komentára sa mohol používateľ vrátiť na projekt len prostredníctvom „späť tlačidla“ prehliadača.

Pri testovaní bola objavená logická chyba vyhľadávania. Pri vyhľadávaní, za použitia viac ako jedného rovnakého filtra, bol napriek správnym hodnotám filtrov výsledok prázdny. Problém spôsoboval fakt, že sa rovnaké filtre spájali do databázového požiadavku pomocou logického *AND* a ich hodnoty sa navzájom vyrušili.

Viacerý užívatelia vyslovili názor, že stránka zobrazujúca stav projektov začína byť neprehľadná 6.1. Pre niektorých bolo otravný presun na ďalšiu stranu stránkovania. Riešením tohto problému bolo vytvorenie stránky nastavení modulu. V nastaveniach modulu je možné zvoliť, či sa majú zobrazovať grafy, súhrnné informácie o projekte a kritické úlohy. Tým si môže používateľ nechať zobrazené pre neho relevantné informácie. Zakázaním zobrazenia všetkých troch častí zostáva stále zobrazený stav projektu a jeho plán na zotavenie. Zredukovaním obsahu sa zmenší aj zabratá plocha obrazovky. Preto nastavenia dovoľujú nastaviť počet projektov, ktoré sa majú zobraziť na jednej stránke.

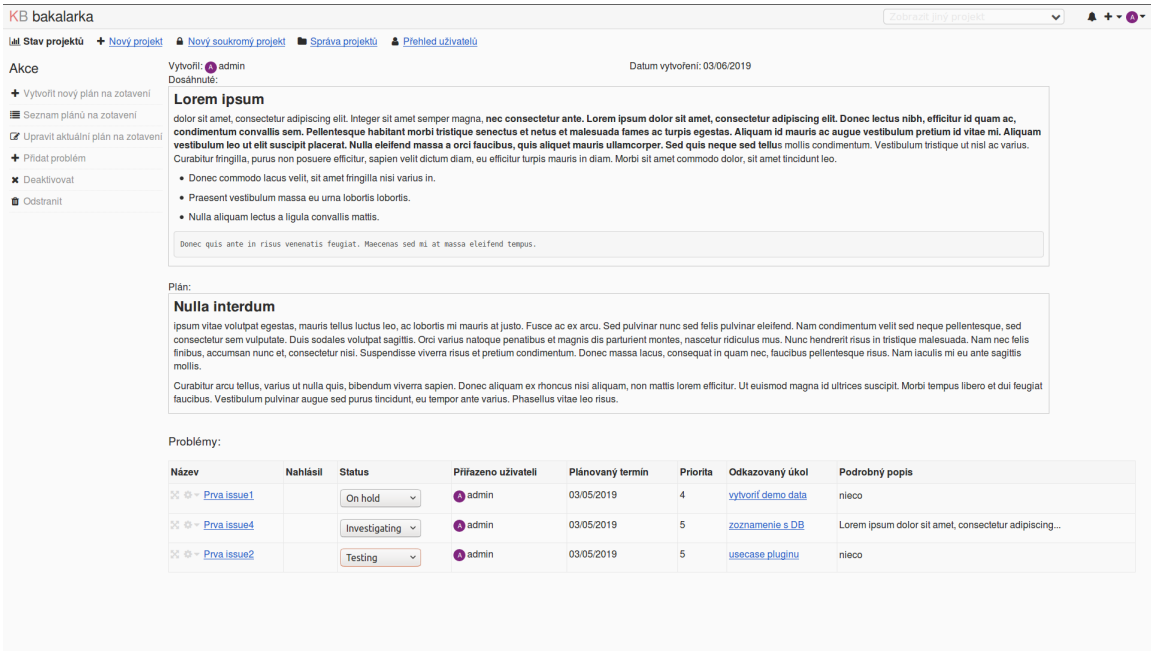
Obr. 6.1: Zobrazenie stavov projektov sa používateľom zdalo neprehľadné. Veľa informácii o projekte zabralo veľkú plochu stránky, preto bolo stránkovanie pevne nastavené na dva projekty na stranu. Nastavenia aplikácie povoľujú zmeny v zobrazení, ako aj počtu projektov, ktoré sa majú zobraziť na jednej stránke.

Užívateľom, ktorý prototyp testovali robil problém aj anglický jazyk, v ktorom bol modul vytvorený. Preto som sa rozhodol vytvoriť preklad všetkých textov modulu do Českého jazyka[1].

6.2 Verzia 1.0

Rieši všetky problémy prototypu, ktoré boli testovaním odhalené. Pridáva zlepšenie navigácie v module prostredníctvom tlačidiel späť. Do súhrnnej správy o stave projektu bola doplnená možnosť na exportovanie vo formátoch *CSV* a *HTML*. To môže byť nápomocné pri tvorení správ pre zákazníka mimo aplikácie Kanboard.org, kde môže byť použitá tabuľka problémov projektu alebo ďalšie informácie plánu na zotavenie. Ďalej bola pridaná podpora databázových technológii *MySQL* a *PostgreSQL*. Keďže technológia *picoDB* 4.2 podporuje aj tieto databázové systémy, bolo treba napísať príkazy na vytvorenie tabuliek v ich syntaxi.

Po testovaní tejto verzie neobjavili testovací užívatelia skoro žiadne faktické chyby. Tie, ktoré boli objavené, boli skôr malého charakteru. Viac nedostatkov bolo v grafickom rozhraní modulu. Za niektorými prekladmi sa stratili medzery. Niekde neboli texty zarovnané 6.2. Po odstránení týchto chýb bol modul zverejnený na platforme GitHub.com, kde je zverejnená aj hlavná aplikácia.



Obr. 6.2: Zobrazenie jedného z testovacích plánov na zotavenie. Aplikácia je prepnutá do Českého jazyka. Texty tabuľky problémov nie sú správane zarovnané.

Pri nájdení malých chýb nebola vytváraná nová verzia modulu, bol len upravený. Pri testovaní verzie 1.0 používateľom, ktorý testoval aj prototyp bola badateľná omnoho lepšia orientovanosť v module. Na splnenie testovacích úloh nepotreboval viac ako 5 minút.

[1] V čase vývoja aplikácia Kanboard.org nepodporovala preklad do Slovenčiny.

Kapitola 7

Záver

Cieľom tejto práce bolo navrhnúť a vytvoriť modul rozširujúci funkcionalitu aplikácie Kanboard.org o zobrazovanie stavu projektov a tvorbu plánu na zotavenie. Zadanie naznačuje, že by mal modul spolupracovať s technológiou SQLite, čo sa mi podarilo. Naviac som modul pripravil na použitie s technológiami MySQL a PostgreSQL. Modul bol obohatený aj o preklad do českého jazyka podporovaného hlavnou aplikáciou. Zadanie tiež dáva možnosť zohľadniť vytvorenie súhrnnej správy o stave projektov, ktorú vytváram ako tabuľku s možnosťou exportovania vo formátoch CSV a HTML. Tvorba modulu bola čo najviac prispôsobená hlavnej aplikácii a nevyžaduje použitie žiadnych externých knižníc, pri ktorých by bolo nutné sledovať zmeny a reagovať na ne. Výsledný modul bol zverejnený na portáli GitHub a je dostupný na adrese `https://github.com/Menoetius/kanboard-plugin-projects-status`.

Zadanie práce bolo splnené. Zoznámil som sa s aplikáciou Kanboard.org rovnako, ako aj s technológiami SQLite a PHP. Naštudoval som si spôsob vytvárania vstavaných modulov pre Kanboard.org. Navrhol som a implementoval modul, ktorý pokrýva všetky stanovené prípady použitia. Aplikáciu som otestoval s užívateľmi a na základe ich pripomienok vylepšil.

Súčasný stav modulu určite nie je konečný. Bolo by nutné ho otestovať viacerými užívateľmi na väčšom množstve dát. Tiež by potreboval spätnú väzbu od manažérov projektov používajúcich aplikáciu Kanboard.org.

Som otvorený pripomienkam a návrhom používateľov na zlepšenie. Do budúcna sa o modul plánujem starať a na základe pripomienok ho zlepšovať. Modul je uvedený s MIT licenciou, ktorá dovoľuje bezplatné použitie kódu a jeho rozšírenie za predpokladu, že budem uvedený ako jeho autor.

Literatúra

[1] *Ajax*. [online], 2019, [cit. 2019-05-08].
URL `https://developer.mozilla.org/en-US/docs/Web/Guide/AJAX`

[2] autorov, K.: *Project Issue Management*. [online], [cit. 2019-05-10].
URL `https://www.mindtools.com/pages/article/newPPM_69.htm`

[3] *Breeze*. [online], [cit. 2019-04-09].
URL `https://www.breeze.pm/`

[4] *C3.js*. [online], [cit. 2019-05-09].
URL `https://c3js.org/`

[5] Haugan, G. T.: *Project Management Fundamentals : Key Concepts and Methodology*. Oakland: Berrett-Koehler Publishers, Incorporated, 2010, ISBN 9781523097302.

[6] Herrero, A.: *Instant Markdown*. Packt Publishing, Aug 2013, ISBN 9781783559145.

[7] Jannika Björkholm, T. B.: *Kanban In 30 days*. Packt Publishing, Júl 2015, ISBN 9781783000906.

[8] *Javascript*. [online], 2019, [cit. 2019-05-06].
URL
`https://developer.mozilla.org/en-US/docs/Web/JavaScript/About_JavaScript`

[9] *JSON*. [online], 2019, [cit. 2019-05-11].
URL `https://developer.mozilla.org/en-US/docs/Glossary/JSON`

[10] *Kanboard.org*. [online], [cit. 2019-04-10].
URL `https://kanboard.org/`

[11] Keith Wald, J. L.: *Pro PHP and jQuery*. Apress, druhé vydanie, Jan 2016, ISBN 9781484212301.

[12] Kreibich, J. A.: *Using SQLite*. O'Reilly Media, Inc., August 2010, ISBN 9780596521189.

[13] Martin Helmich, J. P.: *PHP 7 Programming Blueprints*. Packt Publishing, Okt 2016, ISBN 9781785889714.

[14] Mary Poppendieck, T. P.: *Lean Software Development: An Agile Toolkit*. Addison-Wesley Professional, 2003, ISBN 0321150783.

[15] Peter MacIntyre, R. L., Kevin Tatroe: *Programming PHP*. O'Reilly Media, Inc., tretie vydanie, Feb 2013, ISBN 9781449392772.

[16] Roy, A.: *Cookiecutter*. [online], [cit. 2019-05-10].
URL https://cookiecutter.readthedocs.io/en/latest/readme.html

[17] *Tracy*. [online], [cit. 2019-05-05].
URL https://tracy.nette.org/en/

[18] *Trello*. [online], [cit. 2019-04-07].
URL https://trello.com

[19] Wang, N. R. I. V. R., Xiaofeng; Ali: *Agile and Lean Service-Oriented Development : Foundations, Theory and Practice*. Hershey: IGI Global, 2012, ISBN 9781466625037.

[20] *Wekan*. [online], [cit. 2019-04-09].
URL https://wekan.github.io/

Príloha A

Obsah CD

- readme.txt – súbor popisujúci inštaláciu modulu
- technicka-sprava.pdf – táto technická správa
- /src – zložka so zdrojovými kódmi modulu
- /technicka_sprava – zložka so zdrojovými súbormi v jazyku LaTeX na vytvorenie tejto technickej správy