Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Softwarová podpora organizace předmětů TSP

Plzeň 2022 Hinterholzinger Jan

Místo této strany bude zadání práce.

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 27. března 2022

Hinterholzinger Jan

Abstract

The text of the abstract (in English). It contains the English translation of the thesis title and a short description of the thesis.

Abstrakt

Text abstraktu (česky). Obsahuje krátkou anotaci (cca 10 řádek) v češtině. Budete ji potřebovat i při vyplňování údajů o bakalářské práci ve STAGu. Český i anglický abstrakt by měly být na stejné stránce a měly by si obsahem co možná nejvíce odpovídat (samozřejmě není možný doslovný překlad!).

Obsah

V souboru `literatura.bib` jsou uvedeny příklady, jak citovat knihu [3], článek v časopisu [1], webovou stránku [2].

1 Úvod

2

2.1 Motivace

Na Katedře informatiky a výpočetní techniky vzniká nový předmět Týmový softwarový projekt (KIV/TSP1 a KIV/TSP2) určený pro studenty navazujícího studia. Podstatou předmětu je vypracování zadaného tématu ve skupinkách studentů, kdy studenti přijdou do styku s týmovou prací, řízením projektu a dalšími interními procesy.

Zvláštností předmětu je doba pro řešení projektu, dva semestry rozdělené do dvou předmětů. Budou zapojeny čtyři skupiny lidí. Garant, zadavatelé, mentoři a studenti, což bude klást poměrně značné nároky na organizaci předmětu. Pro takový rozsah bylo rozhodnuto o vytvoření webové aplikace, která všem zúčastněným stranám zjednoduší prohlížení obsahu a evidenci postupu prací projektu. Cílem aplikace je umožnit lepší informovanost a zapojení studentů, řešení evidence zadavatelů a možnosti sdílení mezi mentorem a garantem. Aplikace tak má nahradit způsob organizace pomocí Excel tabulky, jako je tak tomu u jiných obdobných předmětů.

3 Specifikace požadavků

3.1 Organizace předmětů TSP

Výuka předmětů TSP je rozdělena do dvou semestrů, a tedy do dvou předmětů KIV/TSP1 vyučovaného v letním semestru 1. ročníku a KIV/TSP2 v zimním semestru 2. ročníku. Řešení týmového projektu tedy bude překračovat hranice ročníku a v mnoha částech se organizace předmětů bude překrývat.

Jednotlivý cyklus se v nejčastějším případě bude probíhat následovně.

- Garant přijímá od domluvených zadavatelů jejich témata, která mají studentské týmy řešit. Toto zadání společně s možným PDF souborem vloží do podpůrného softwaru pro zvolené období předmětu.
- Před začátkem TSP budoucí vedoucí týmu zažádá garanta o založení účtu v podpůrném softwaru s rolí „Vedoucí“. Vedoucí týmu společně s dalšími studenty sestaví tým a vloží informace o týmu do této podpory.
- Poté co jsou témata zadání projektů zveřejněny, vedoucí týmu (na základě rozhodnutí v týmu) projeví zájem o zvolené téma.
- Mentor ohodnotí vhodnost zvoleného tématu pro daný tým na základě zájmu, který vedoucí týmu učinil.
- Garant následně podle rozhodnutí mentora a zájmu týmu definitivně přiřadí nebo nepřiřadí téma týmu. Přiřazením tématu vzniká projekt.
- Tým řeší projekt a postup řešení vkládá do softwarové podpory
- Mentor na základě postupu řešení týmu kontroluje a potvrzuje postup týmu
- Na závěru předmětu TSP se koná obhajoba projektu, jejíž výsledek se zapíše do softwarové podpory.

3.2 Uživatelé

3.2.1 Nepřihlášený uživatel

3.2.2 Vedoucí týmu

3.2.3 Mentor

Mentor je osoba, která kontroluje plnění postupu týmu.

3.2.4 Garant

3.3 Případy užití

4 Dostupné technologie

4.1 PHP frameworky

Existuje mnoho úspěšných PHP frameworků, které mají různé typy zaměření. Mezi nejúspěšnější a nejpoužívanější patří Laravel, Symphony a Nette. Všechny tyto frameworky si zakládají na vytváření znovupoužitelných komponent a služeb. Kromě toho v sobě obsahují rozsáhlé nástroje pro usnadnění například práce s databází, přesměrování, ošetření bezpečnosti, atd. Díky tomu ulehčují vývojářům vlastní vývoj aplikace a nemusí tak vyvíjet úsilí pro tvorbu vlastních řešení. Výrazné zjednodušení představuje funkce dependency injection, která zajišťuje propojení mezi jednotlivými komponentami a službami a hlídá dostupnost všech závislostí.

Součástí těchto frameworků jsou také šablonovací enginy, které zjednodušují tvorbu front-endu. Tyto šablonovací systémy umožňují dědičnost jednotlivých pohledů, jejich členění na sekce a obecně jejich cílem je jednodušeji prezentovat data z back-endu. Jednotlivé šablonovací enginy se liší svými funkcemi a rozšířeními, ale typicky je vybíráme dle osobních preferencí nebo dle použitého back-end frameworku.

4.1.1 Laravel

Framework Laravel je možné považovat jako za ten nejrozšířenější. Mezi jeho hlavní východy patří jeho jednoduchost používání a rychlost. Pro svůj přístup k jednoduchému použití je Laravel doporučován jako vhodný pro začátečníky ale i pro profesionály. Framework se hodí pro vytváření méně komplexních projektů.

Laravel využívá šablonovací engine Blade, který je standardně dodáván společně se samotným frameworkem. Tento engine umožňuje oproti jiným rozšiřovat PHP kód, a tak provádět různé jednoduché operace pro přizpůsobení dat k samotnému front-endu.

4.1.2 Symphony

Tento framework se vyznačuje zakládáním si na striktním dodržování nejen PHP standardů a snaží se maximálně využívat různé návrhové vzory. Díky tomu jsou komponenty frameworku robustnější, což může znamenat větší časovou náročnost, ale také výraznou stabilitu frameworku, a proto je vhodný

pro použití na komplexnějších projektech. Další předností mohou být rozsáhlé možnosti pro vývojáře, který si může prostředí přizpůsobit svým potřebám. To však vyžaduje hlubší znalosti jazyka PHP a struktury frameworku. Pro nováčky se tedy Symphony více náročný na naučení.

Jako výchozí šablonovací engine je využíván Twig, který se také řadí mezi nejpoužívanější šablonovací systémy. Oproti systému Blade obsahuje navíc další bezpečnostní vrstvu a další funkce. Twig je často využíván i samostatně, tedy bez použití back-end frameworku. To podtrhuje jeho flexibilitu.

4.1.3 Nette

S frameworkem společně přichází i šablonovací engine Latte.

4.2 Front-end

4.2.1 HTML a CSS

Základem webových aplikací je způsob zobrazování. Webové technologie nabízejí tvorbu elementů pomocí formátu HTML založeného na XML. Vlastností tohoto formátu je tvorba webových elementů pomocí tagů, kterým se specifikují jejich vlastnosti. Dnes již naprostá většina prohlížečů zobrazuje webové stránky pomocí tohoto formátu a stává prakticky synonymem pro internetové stránky.

Další obdobně oblíbenou technologií je stylovací kaskádový jazyk CSS. Díky tomuto formátu je možné nastavovat atributy jednotlivých elementů a upravovat tak zejména jejich vzhled. Tyto styly lze aplikovat přímo do HTML tagu daného elementu nebo do blokového elementu společný pro celou stránku. Nejčastěji se však využívá importování přiloženého `.css` souboru. CSS umožňuje vzhled stránky upravovat i podle velikosti displeje zařízení a tak přizpůsobit obsah i pro menší obrazovky.

4.2.2 Bootstrap

Bootstrap představuje nadstavbu nad CSS styly. Umožňuje upravovat prvky pomocí tříd a dát jim tak moderní vzhled bez potřeby upravovat styly ručně. Při správném použití tohoto frameworku je velice snadno zajištěna responzivita webu.

4.2.3 AdminLTE

4.3 Nástroje pro řízení projektu

Aplikace, kterou vyvíjíme, bude patřit mezi rozsahově náročnější, proto je její řešení rozděleno do dvou prací. Jedna práce (tato) se věnuje samotnému vývoji aplikace. Druhá je zaměřená na důkladné otestování aplikace, čímž má být zajištěna její kvalita spolehlivost.

Ze skutečnosti, že na aplikaci takového rozsahu pracuje více lidí, je potřeba využít takové procesy, které usnadní jednotlivé části vývoje a domlovu mezi vývojem, testováním a vedoucím práce.

4.3.1 Verzovací systém

V oblasti vývoje informačních projektů je častým nástrojem pro efektivní a produktivní vývoj verzovací systém ze skupiny Git.

Tato technologie totiž umožňují práci více vývojářů na jednom projektu současně. Jednotlivé změny v repozitáři vývojáři seskupují do tzv. commitů. Ty totiž v systémech Git představují nejmenší jednotku změny v úložišti. Commit v sobě obsahuje informace o upravených řádkách textových a změny dalších souborů.

Další důležitou funkcí je větvení. V základu je v repozitáři hlavní větev „main“ nebo „master“, který obsahuje centrální vývoj projektu. Od této větve se se oddělují další větve, dle určení. Typicky se do jednotlivých větví postupně commitují úpravy rozsáhlejší funkcionality aplikace. Díky tomu ostatní vývojáři nejsou těmito změnami zasaženi a tak si nepřekážejí.

Po dokončení úprav nebo přidávání funkcionality jsou commity větve sloučeny do rodičovské větve, kde tvoří větší celek. Tato funkce se jmenuje Merge request a její starostí je přemístit commity jedné větve do jiné větve tak, aby byly změny zaneseny. Může se stát, že se změny v obou větvích dostanou do konfliktu, poté záleží na uživateli jak konflikt vyřeší.

Dostupných nadstaveb verzovacích systémů existuje celá řada. Mezi nejvýznamnější na trhu patří GitHub a GitLab.

GitHub je pravděpodobně nejúspěšnější systém pro tvorbu projektů od jednotlivců nebo malých skupin. Obsahuje však výrazná omezení, která jsou sice odstraněny v prémiovém plánu, ale v našem případě výhody ostatních nadstaveb převažují nad výhodami GitHubu.

Z tohoto důvodu se zdá být přínosnější použít systém GitLab, který pokrývá veškeré naše požadavky.

Důležitou zmínkou je, že projekt GitLab je veden jako open-source. Díky

tomu Katedra informatiky a výpočetní techniky provozuje svoji vlastní instalaci tohoto verzovacího nástroje na katedrálním serveru.

To má pro studenty nesmírnou výhodu, protože během dosavadního studia již GitLab používali a mají s ním zkušenosti. Navíc všichni zúčastnění mají na této platformě vytvořené své účty, nemusíme tedy organizačně řešit vstup na novou platformu.

Z těchto důvodů jsme se v naší volbě verzovacího systému rozhodli zvolit právě variantu s katedrálním serverem.

4.3.2 Plánování úkolů

S rozsahem práce přichází i potřeba rozvržení práce na časové úseky a stanovení cílů. Zároveň je žádoucí, aby k plánu měli přístup všechny zúčastněné osoby. Tyto požadavky sice splňuje množství známých nástrojů nebo sdílených úložišť. My jsme se ale rozhodli využít již námi používaný nástroj GitLab.

Tento nástroj obsahuje možnosti vytváření wiki dokumentace ve formátu MarkDown, který navíc zajišťuje základní formátování.

V této dokumentátorské sekci máme vytvořenou jednu stránku, ve které je po týdnech rozdělena struktura naplánovaných úkolů a vzniklých dotazů.

4.3.3 Issues

Různé požadavky evidujeme jako jednotlivé issues. Issues je záznam o požadavku na aplikaci nebo úkolu, který je potřeba vykonat. Issue tak eviduje přiřazení k člověku, který daný požadavek řeší, kompletní historii řešení požadavku, termíny pro jeho splnění, atd.

Systém také poskytuje štítkování těchto issues a tím pomáhá v jejich řazení, filtrování, seskupování i výběru. Díky štítkům můžeme totiž přiřazovat prioritu, závažnost, druh požadavku na aplikaci a jiné označení.

Issues nám pomáhají ve fragmentaci jednotlivých požadavků na aplikaci do přibližně stejně náročných dílů, které se poté ve vývoji dobře plní a je tak zajištěna přehlednost. Zároveň má tento princip i pozitivní psychologický dopad, kdy vývojáři mají dobrý pocit z dokončeného úkolu a mají tak chuť pokračovat dalším issue.

V systému GitLab tyto issues používáme také ve funkci Board, který využívá principy kanbanu.

4.3.4 Kanban

Systém kanban je strategie řízení projektu, kdy si mezi sebou části výroby předávají výrobek. Cílem je využívat pouze ty nejnutnější zdroje.

V praxi se tento systém zobrazuje jako tabule, kde jsou rozvržené jednotlivé sloupce dle možných stavů vývoje. Následně zde máme rámečky představující výrobek, které obsahují popis jak má být produkt upraven. Podstatou věci je, že tyto rámečky následně přemisťujeme mezi jednotlivými sloupci v závislosti na reálném stavu produktu.

V informačních technologiích se kanban používá ve velké míře pro organizaci různých úkolů a požadavků. Různé části vývoje si mezi sebou požadavek vyměňují a tím mění jeho stav.

Například když vývojář dokončí práci na nové funkcionalitě, tak přesune příslušný rámeček ze sloupce „Ve vývoji“ do sloupce „Připraveno k testování“. Tím se tester dozví, že je daný úkol připraven k testování a může si úkol přiřadit a pracovat na něm.

V GitLabu je systém kanbanu implementován v souvislosti s issues a jejich štítky. Každý issue je zobrazen jeden rámeček na tabuli kanban, kam se automaticky po vytvoření issue promítne a po přidělení štítků zařadí do připravených sloupců.

Propojenost s issue nám ulehčí práci s přepisováním jednotlivých požadavků do jiných nástrojů, které mají obdobné funkce.

4.3.5 MantisBT

Mantis Bug Tracker je webová aplikace pro nahlašování a evidenci chyb (defektů) vytvořených v průběhu vývoje aplikace. Přidávané defekty lze velmi podrobně popsat, určit prioritu a štítkovat.

MantisBT využíváme pro nahlašování objevených selhání nalezených především, ale ne výhradně pomocí komplexního automatizovaného testování aplikace. Na základě reportování defektů jsou následně žádány jejich opravy chyb a po opravě jsou znovu testovány. Pro naše potřeby používáme vlastní instalaci MantisBT, která je přístupná pro anonymně přihlášené, aby mohli v budoucnu nahlašovat defekty i uživatelé aplikace.

5 Návrh aplikace

5.1 Architektura aplikace

5.1.1 MVC architektura

5.1.2 Komponenty aplikace

5.2 Rozdělení aplikace dle způsobu užití

5.3 Databázová struktura

5.4 Uživatelské rozhraní

6 Realizace

6.1 Adresářová struktura

7 Testování

8 Závěr

Literatura

[1] Hoare, C. A. R. Algorithm 64: Quicksort. *Commun. ACM*. July 1961, 4, 7, s. 321. ISSN 0001-0782. doi: 10.1145/366622.366644. Dostupné z: http://doi.acm.org/10.1145/366622.366644.

[2] *Class Graphics2D* [online]. Oracle, 2016. [cit. 2016/03/09]. Java SE Documentation. Dostupné z: https://docs.oracle.com/javase/7/docs/api/java/awt/Graphics2D.html.

[3] Knuth, D. E. *The Art of Computer Programming, Volume 2 (3rd Ed.): Seminumerical Algorithms*. Addison-Wesley Longman Publishing Co., Inc., 1997. ISBN 0-201-89684-2.