

Management projektů

Programová podpora řízení rizik v projektech

Octopus

Konfigurační řízení

# Úvod

Tento dokument slouží jako celkový popis toho, jaké softwarové prostředky použije tým Octopus pro realizaci projektu „Programová podpora řízení rizik v projektech“. V následujícím textu je popsáno, jaký software byl využit k vyvíjení výsledného programu, v jakém jazyce a jeho frameworcích proběhla realizace a taktéž jakým stylem probíhá koordinace prací více programátorů najednou, resp. jak je hlídán konflikt nově programovaných částí.

# Vývojová platforma

Projekt je realizován jako webová aplikace. V tomto případě bude aplikace dostupná z jakéhokoliv zařízení, které disponuje webovým prohlížečem a internetem (počítač, chytrý telefon, tablet apod.) Díky tomuto přístupu jsme schopni v dnešní době zacílit vyvíjený software na téměř jakoukoliv platformu.

# Softwarové vybavení

Webový projekt bude realizován pomocí tzv. LAMP (Linux, Apache, MySQL, PHP). Linux je hlavním operačním systémem použitým při vývoji, ale bude využit i operační systém od značky Apple (převážně vývoj front-end) a také Windows. Jazyk PHP není využit v jeho čisté formě, ale je obohacen o framework zvaný Nette, konkrétně pak ve verzi jazyka PHP 7.0. Pro front-end část je domluven jazyk JavaScript. Pro psaní back-end části je domluven jednotný software PHP Storm od firmy JetBrains (dostupný na studentskou licenci), který nám dovolí jednotný styl psaní kódu viz. [Štábní kultura](#_Štábní_kultura). Databázová část je realizována na systému MySQL. Pro správu projektu bylo zřízené Trello, které slouží jako agilní nástěnka, ale je taktéž využito propagovaného softwaru EasyProject.

Verzovací systém pro tento projekt je GitHub, který nabízí taktéž plnohodnotné studentské licence. Organizace slučování kódů byla zvolena po přístupu tzv. GitFlow, kdy každý pracovník vlastní svoji větev, kterou poté slučuje s Develop větví. Master větev je použita pak pouze pro finální verzi. Podrobněji viz. [Verzování](#_Verzování).

# Verzování

Základní postup verzování byl popsán v předchozím odstavci. Všichni členové jsou obeznámeni se správnou prací s verzovacím systémem Git. Tento systém umožnuje navrácení projektu do historických stavů. K sáhnutí po takových to změnách by nemělo docházet, protože každý jednotlivý programátor si spravuje svoji větev, kde vyvíjí svoji část kódu, případně si může stáhnout nezbytnou část z jiných větví. Sám systém Git hlásí konflikty, které lze jednoduše řešit v softwaru PHP Storm, který již disponuje pokročilou podporou pro verzovací sytém Git. Při dodržení těchto postupů dochází k minimálním nechtěným zásahům do kódu druhých programátorů.

# Požadavky na změny

Jednotlivé požadavky na změny jsou oznámeny vedoucímu projektu, který následně vytváří jednotlivé úkoly, které vedou k jejich nápravě. I přestože je projekt kratšího rázu, tak se snažíme využívat agilní metodiky SCRUM s výrazně zkrácenými sprinty. Při jakýchkoliv připomínkách je tedy změna zařazena do backlogu věcí, které je třeba vyřešit v dalším sprintu. U takto krátkého projektu to bude znamenat spíše v moment, kdy člověk zodpovědný za jednotlivou část dokončí aktuálně rozdělanou práci.

# Štábní kultura

V kapitole 3 bylo řečeno, že bude využívám jednotný editor pro psaní kódu, a to PHP Storm. Tento fakt umožnuje vynechat nutnost psát jakýkoliv codestyle, který by si museli programátoři importovat do svých editorů, neboť bude využit jednotný codestyle definovaný tímto editorem. Samozřejmě je zapotřebí stanovit jednotný styl psaní proměnných a funkcí. Byl zvolen víceméně standard v dnešních moderních programovacích jazycích, a to tzv. camelCase, kdy první písmeno je vždy malé a další slova z původně zamýšleného názvu proměnné jsou započaty velkým písmenem, např. mojeSkvelaPromenaJeUzasna. V tomto bodě je dobré zmínit, že veškerý kód je psán v angličtině, neboť se toto osvědčilo jako dobrá praxe (teoretické přijmutí česky nemluvící programátora do týmu, sdílení kódu se zahraniční firmou atd.) Složitější funkce by měly být pečlivě blokově komentovány.