ZÁPADOČESKÁ UNIVERZITA V PLZNI

FAKULTA EKONOMICKÁ

Bakalářská práce

Návrh a implementace webového informačního systému

Design and implementation of a web-based information system

Jiří Andrlík

Plzeň 2022

Čestné prohlášení

Prohlašuji, že jsem bakalářskou práci na téma

„Návrh a implementace webového informačního systému“

vypracoval samostatně pod odborným dohledem vedoucího bakalářské práce za použití pramenů uvedených v přiložené bibliografii.

Plzeň dne 24. 4. 2022 v. r. Jiří Andrlík

Poděkování

Rád bych poděkoval vedoucímu bakalářské práce doc. RNDr. Mikuláši Gangurovi, Ph.D. za odborné vedení, ochotu, trpělivost a čas, který mi věnoval při psaní této bakalářské práce.

Obsah

Úvod

Práce na různých typech projektů umožňuje studentům vysokých škol praktické využití teoretických znalostí, které získávají v průběhu svého studia. Tato dovednost je v dnešní době pro studenty a absolventy velmi důležitá a její osvojení může pomoci v konkurenceschopnosti na trhu práce. Dokončený projekt, jehož výstup splní stanovený cíl, nejlépe dokazuje schopnosti studenta aplikovat získané teoretické znalosti a dovednosti. Další, v dnešní době velmi důležitou, dovedností je schopnost týmové spolupráce. Projekt, který je zpracováván v týmech různé velikosti, může umožnit všem zúčastněným spolupracovníkům osvojit si obě tyto poměrně zásadní dovednosti.

Při studiu na vysoké škole jsou studenti ve většině předmětů vedeni k vypracování jak individuálních, tak i skupinových semestrálních prací. Mohlo by však vznikat více studentských projektů rozvíjejících vlastní nápady studentů a vytvořených jejich vlastní iniciativou. Tím by se mohla potenciálně zvýšit pravděpodobnost, že již v období studia na univerzitě vznikne v rukou studentů zárodek úspěšného startupu. Autor z vlastní zkušenosti potvrzuje, že bez jakéhokoliv podpůrného nástroje může být velmi obtížné najít mezi studenty napříč fakultami jedince, kteří by měli o případnou spolupráci na týmovém projektu zájem.

Hlavním cílem této bakalářské práce je vytvořit webový informační systém pro zveřejňování informací o týmových projektech a hledání možností spolupráce na těchto projektech. Vytvořený informační systém by měl sloužit studentům jako nástroj pro podporu hledání spolupráce na aktuálně vypsaných týmových projektech. Ze stanoveného hlavního cíle jsou odvozeny následující dílčí cíle:

- Seznámit se s existujícími informačními systémy souvisejícími s tématem zveřejňování informací o týmových projektech a hledání možností spolupráce.
- Provést analýzu požadavků na webový informační systém.
- Navrhnout funkcionality a obsah webového informačního systému.
- Implementovat navržený webový informační systém.
- S využitím uživatelských scénářů otestovat použitelnost implementovaného webového informačního systému.

- Na základě výsledků testů použitelnosti navrhnout možná zlepšení informačního systému.
- Diskutovat možnosti budoucího rozšíření funkcionality vytvořeného webového informačního systému.

Bakalářská práce je rozdělena na tři části. První část práce, která je teoreticky a rešeršně zaměřená, obsahuje představení základních pojmů, popis vybraných témat souvisejících s návrhem a implementací webového informačního systému a rešerši současné situace týkající se hledání možností spolupráce na týmových projektech.

Druhou částí je metodika řešení, která popisuje metody a postupy konkrétního řešení návrhu a implementace webového informačního systému.

Třetí částí práce je řešení. Tato prakticky zaměřená část obsahuje popis vize a rozsahu navrhovaného informačního systému, zabývá se specifikací požadavků a popisem případů užití. Na základě definovaných požadavků je provedena analýza procesní orientace informačního systému. Důležitou součástí řešení je návrh datové a softwarové architektury spolu s návrhem uživatelského rozhraní. Dle vytvořených návrhů je následně provedena implementace webového informačního systému. Závěr třetí části je věnován realizaci a vyhodnocení testování použitelnosti a popisu návrhů na rozšíření funkcionality implementovaného webového informačního systému.

V závěru bakalářské práce je provedeno shrnutí dosažených výsledků řešení a zhodnocení splnění stanoveného hlavního cíle a jednotlivých dílčích cílů.

1 Rešerše zdrojů

První část bakalářské práce zabývající se rešerší zdrojů je rozdělena do devíti kapitol. Kapitola 1.1 obsahuje definici pojmu webový informační systém a klasifikaci informačních systémů. Kapitola 1.2 se věnuje problematice definice požadavků na informační systém, stručně představuje jednotlivé typy, zdroje a autory požadavků. Kapitola 1.3 se zabývá definicí architektury informačního systému spolu s představením jednotlivých dílčích architektur. Kapitola 1.4 se věnuje návrhu datové architektury informačního systému a datovému modelování na třech úrovních abstrakce. V kapitole 1.5 je rozebrána problematika návrhu softwarové architektury. Kapitola 1.6 se zabývá představením vybraných technologií pro implementaci IS. Kapitola 1.7 se věnuje představení vybraného PHP frameworku. Kapitola 1.8 obsahuje představení problematiky testování použitelnosti informačního systému. Kapitola 1.9 se zabývá současnou situací hledání možností spolupráce na projektech.

1.1 Webový informační systém

Kapitola obsahuje vymezení pojmu informačního systému, představuje klasifikaci informačních systémů a zabývá se definicí pojmu webový informační systém.

1.1.1 Definice informačního systému

Aby bylo možné definovat webový informační systém jako pojem, je nutné se nejprve zaměřit na pojem v jistém slova smyslu nadřazený. Tímto pojmem je informační systém. Jednoznačně definovat informační systém není vůbec jednoduché. Gangur (Gangur, 2014) uvádí, že jednotná definice tohoto pojmu neexistuje a vždy záleží na pohledu, na který je při popisu informačního systému kladen důraz. V odborné literatuře se tak můžeme setkat s velkým množstvím různých definic tohoto pojmu.

„Informační systém je druh systému, jehož prvky jsou lidé, potenciální informace (dokumenty, data), technické prostředky a metody a také pravidla zajišťující shromažďování, zpracování, uchovávání a vyhledávání těchto potenciálních informací za účelem jejich využití. Jeho vazby jsou pak definovány jako potencionální informace a prvky jako místa transformace těchto informací." (Cejpek, 2005, str. 37)

Autoři van der Aalst a Stahl (van der Aalst & Stahl, 2011) pak definují informační systém jako softwarový systém pro zachycování, přenos, ukládání, vyhledávání, manipulaci nebo zobrazování informací, který podporuje lidi, organizace nebo jiné softwarové systémy.

Ačkoliv se v jiné literatuře můžeme setkat i s dalšími, na první pohled jinak formulovanými, definicemi informačního systému, hlavní myšlenka práce informačního systému s informacemi za účelem jejich využití uživateli systému je vždy zachována.

1.1.2 Klasifikace informačních systémů

Problém s jednoznačnou klasifikací informačních systémů spočívá v tom, že klasifikace vyvinutá před několika lety nemusí být stále aktuální i v dnešní době. Dalším a hlavním omezujícím faktorem je, že klasifikační kategorie nejsou obvykle jednoznačně odděleny neboli jeden typ informačního systému může patřit do více kategorií (van der Aalst & Stahl, 2011).

Autoři van der Aalst a Stahl (van der Aalst & Stahl, 2011) uvádějí následující vysokoúrovňovou klasifikaci, která rozlišuje tři třídy informačních systémů:

- Osobní informační systémy – spravují informace pro osobní účely,
- Podnikové informační systémy – slouží jako podpora organizace,
- Veřejné informační systémy – spravují informace, které mají být veřejně přístupné v určité komunitě lidí.

1.1.3 Definice webového informačního systému

Nyní je možné zaměřit se na definici webového informačního systému. Odpověď můžeme nalézt ve způsobu implementace informačního systému. Pokud je informační systém implementován jako webová aplikace využívající pro zobrazení výstupů IS v prezentační vrstvě webový prohlížeč, používáme k jeho označení termín webový informační systém (Gangur, 2014).

1.2 Požadavky na informační systém

Kapitola obsahuje vymezení pojmu požadavek na informační systém, představuje jednotlivé typy požadavků, možné zdroje, autory a krátce zmiňuje problematiku jejich dokumentace.

1.2.1 Definice požadavků

V odborné literatuře se můžeme setkat s různými definicemi pojmu požadavek. Dle SEVOCAB (SEVOCAB, 2017) můžeme požadavek na informační systém definovat následujícími dvěma způsoby:

1. Požadavek je schopnost systému, kterou uživatel potřebuje k vyřešení určitého problému.
2. Požadavek můžeme definovat jako schopnost, kterou musí splňovat nebo vlastnit systém nebo komponenta systému, aby splnily smlouvu, normu, specifikaci nebo jiný formálně dohodnutý dokument.

Chlapek, Řepa a Stanovská (Chlapek, Řepa, & Stanovská, 2011) uvádí, že požadavky je vhodné správně specifikovat již v prvních fázích vývoje informačního systému, protože náklady na zjištění požadavků a analýzu systému jsou zanedbatelné oproti nákladům na opravu systému vytvořeného podle chybně specifikovaných požadavků.

1.2.2 Typy požadavků

Požadavky na informační systém můžeme dělit na dvě základní skupiny. Funkční požadavky vychází z podstaty, proč má informační systém existovat, a popisují, co musí systém dělat. Nefunkční požadavky naopak vyjadřují nutné vlastnosti či kvalitativní charakteristiky informačního systému. (Chlapek, Řepa, & Stanovská, 2011)

Do nadřazené skupiny tzv. funkčních požadavků řadíme kromě samotných funkčních (behaviorálních) požadavků i požadavky podnikatelské a uživatelské (Chlapek, Řepa, & Stanovská, 2011).

Podnikatelské požadavky vyjadřují cíle, kterých by zákazník chtěl prostřednictvím vyvinutého informačního systému dosáhnout. Mohou pocházet od hlavního investora, nabývajícího zákazníka nebo i od produktového vizionáře. (Wiegers, 2008)

Uživatelské požadavky vyjadřují cíle a úkoly, které by měly být uživatelé s výsledným informačním systémem schopni provést (Wiegers, 2008).

Funkční (behaviorální) požadavky, známé také jako požadavky na chování, se využívají k popisu softwarové funkcionality, kterou musí informační systém obsahovat, aby mohli uživatelé splnit své úkoly a zároveň tak mohly být splněny požadavky

podnikatelské. Funkční požadavky ve své podstatě určují, co musí vývojáři do informačního systému naprogramovat. (Wiegers, 2008)

Do skupiny tzv. nefunkčních požadavků můžeme zařadit požadavky na výkonnost softwaru, požadavky na vnější rozhraní softwaru, omezení návrhu softwaru nebo atributy kvality softwaru (SEVOCAB, 2017).

1.2.3 Zdroje a autoři požadavků

Chlapek, Řepa a Stanovská (Chlapek, Řepa, & Stanovská, 2011) uvádí následující zdroje výše uvedených typů požadavků:

- vize,
- poslání,
- strategie,
- procesy, činnosti, funkce a omezení,
- požadavky zadavatelů,
- legislativní požadavky.

Autorů jednotlivých požadavků na informační systém může být velké množství. Může se jednat o zákazníky, uživatele výsledného informačního systému, analytiky požadavků, vývojáře informačního systému, testery, tvůrce dokumentace, vedoucí projektu, odborníky na legislativu, obchodníky, pracovníky marketingu nebo pracovníky zákaznické podpory (Chlapek, Řepa, & Stanovská, 2011).

1.2.4 Dokumentace požadavků

Je zřejmé, že všechny požadavky je potřeba správně a důkladně dokumentovat. Podnikatelské požadavky bývají popsány v dokumentu s popisem vize a rozsahu projektu, uživatelské požadavky jsou většinou zachyceny v podobě případů užití (slovní nebo grafické zpracování s využitím UML diagramů). Podrobné funkční i nefunkční (parametrické) požadavky dohromady tvoří tzv. specifikaci požadavků. Výsledný dokument specifikace požadavků by tedy měl přesně popisovat funkce poskytované výsledným informačním systémem včetně všech požadovaných omezení. (Wiegers, 2008)

1.3 Architektura informačního systému

Kapitola obsahuje vymezení pojmu architektura informačního systému a představuje jednotlivé dílčí architektury.

1.3.1 Definice architektury

„Architektura informačního systému je vyjádření celkové koncepce informačního systému.“ (Rybička, 2021, str. 197).

S pojmem architektura se můžeme setkat ve dvou rovinách, a to na globální a na dílčí úrovni. Globální architektura informačního systému je architektura, která představuje základní schéma vyjadřující hrubou podobu budoucího informačního systému (Danel, 2021). Podle Brucknera, Voříška a Buchalcevové (Bruckner, Voříšek, & Buchalcevová, 2012) na globální architekturu navazují následující dílčí architektury:

- Aplikační architektura,
- Datová/informační architektura,
- Softwarová architektura,
- Technologická architektura.

Je důležité zmínit, že při vývoji informačního systému není nutné vždy navrhovat všechny dílčí architektury (Bruckner, Voříšek, & Buchalcevová, 2012).

1.3.2 Aplikační architektura

Při vývoji nové aplikace, která má být součástí například informačního systému podniku, je nezbytné dívat se na systém jako na celek, a brát tak v úvahu všechny aplikace informačního systému a jejich vzájemné vztahy (Bruckner, Voříšek, & Buchalcevová, 2012).

Aplikační architekturou má tedy smysl se zabývat v případě, pokud je informační systém tvořen větším množstvím podsystémů. S touto situací se můžeme setkat především u komplexních podnikových informačních systémů.

1.3.3 Datová architektura

Datová architektura, která popisuje datovou základnu informačního systému, definuje jednotlivé datové objekty, jejich strukturu a vztahy mezi nimi. Na základě analýzy

potřebných datových objektů a vazeb mezi nimi je proveden konceptuální návrh datové základny, následovaný logickým návrhem. Posledním krokem k finalizaci datové architektury je fyzický návrh datové základny – návrh databázových souborů a jejich fyzického uložení. (Bruckner, Voříšek, & Buchalcevová, 2012)

Detailnějšímu představení problematiky návrhu datové/informační architektury je věnována kapitola 1.4

1.3.4 Softwarová architektura

Softwarovou architekturu můžeme chápat jako dílčí architekturu, která specifikuje, z jakých programových komponent se bude celý systém skládat (Rybička, 2021).

„V případě softwarové architektury je systémem, na kterém architekturu definujeme, jeden softwarový produkt, tedy jedna softwarová aplikace. Hlavními komponentami, jejichž strukturu a vztahy architektura definuje, jsou programové moduly aplikace.“ (Bruckner, Voříšek, & Buchalcevová, 2012, str. 271).

Detailnějšímu představení problematiky návrhu softwarové architektury je věnována kapitola 1.5.

1.3.5 Technologická architektura

Technologickou architekturu si můžeme představit jako dílčí architekturu, jejíž úkolem je propojovat datovou a softwarovou architekturu s hardwarovým vybavením informačního systému (Danel, 2021). Systémem, na kterém technologickou architekturu definujeme je provozní platforma vytvořeného informačního systému (Bruckner, Voříšek, & Buchalcevová, 2012).

„Hlavními komponentami, jejichž strukturu a vztahy architektura definuje, jsou hardwarové komponenty (servery, koncové stanice, počítačové sítě atd.) a komponenty základního programového vybavení (operační systémy, databázové systémy, komunikační systémy, integrační software atd.).“ (Bruckner, Voříšek, & Buchalcevová, 2012, str. 279).

1.4 Návrh datové architektury

Jak již bylo zmíněno v předchozí kapitole, datovou architekturu definujeme na systému zvaném datová základna informačního systému neboli databáze. Databáze definujeme jako specializované struktury, které umožňují počítačovým systémům nejen ukládat, ale i velmi rychle spravovat a vyhledávat data v nich uložená (Coronel, Morris, & Rob, 2011). Ke správě této specializované struktury a jejího obsahu slouží soubor programů nazvaný systém řízení báze dat, jehož hlavním úkolem je řídit přístup k datům uloženým v databázi (Coronel, Morris, & Rob, 2011).

Dle Pokorného & Valenty (Pokorný & Valenta, 2020) existuje několik různých přístupů k tvorbě databází, mezi které můžeme mimo jiné zařadit:

- síťové a hierarchické databáze (vzájemně propojené soubory dat),
- relační databáze (fyzicky nezávislé soubory dat),
- objektové databáze, resp. objektově relační databáze.

Ačkoliv se v dnešní praxi můžeme setkat se všemi uvedenými přístupy k tvorbě databází, dlouhodobým historickým trendem se stal výlučný přechod na relační databáze. Relační databázový model organizuje data do tzv. relací (tabulek) a dává uživatelům silné prostředky pro práci s takto chápanými daty (Pokorný & Valenta, 2020).

Prvním krokem při návrhu databáze je proces zvaný datové modelování, jehož cílem je vytváření konkrétního datového modelu pro stanovenou předmětnou oblast (Coronel, Morris, & Rob, 2011).

Dle Brucknera, Voříška & Buchalcevové (Bruckner, Vořišek, & Buchalcevová, 2012) rozlišujeme při tvorbě datových modelů tři úrovně popisu datových struktur:

- Konceptuální model – popisuje obsah dat systému na úrovni nezávislé na vlastním implementačním a technologickém řešení.
- Logický (technologický) model – popisuje způsob realizace dat systému závisle na technologickém řešení (např. na síťové, hierarchické nebo relační databázi).
- Fyzický (implementační) model – popisuje vlastní realizaci databáze v konkrétním implementačním prostředí.

Konceptuální model

Cílem konceptuálního modelu je přirozený, ale zjednodušený pohled na svět za pomoci modelování entit a vztahů mezi nimi (Pokorný & Valenta, 2020). Datový model na konceptuální úrovni můžeme zachytit pomocí tzv. ERA diagramů, které se využívají k vyjádření datových objektů (entit), jejich podstatných vlastností (atributů) a vzájemných vztahů mezi entitami (Bruckner, Vořišek, & Buchalcevová, 2012).

Logický (technologický) model

Logický model reprezentující relační schéma databáze zobrazuje entitní i vztahové typy ve formě relací (tabulek) spolu s relačními atributy odpovídajícími atributům objektů konceptuálního modelu. Jednotlivým relacím jsou přiřazovány primární klíče, relačním atributům vhodné domény a doplněny jsou i referenční integrity. Dalším cílem logického modelu je, aby výsledné tabulky byly normalizované, tj. aby splňovaly alespoň třetí normální formu. (Pokorný & Valenta, 2020)

Fyzický (implementační) model

Fyzický model reprezentuje datový model na nejnižší úrovni abstrakce a popisuje konkrétní způsob uložení dat v paměti (Coronel, Morris, & Rob, 2011). Důležité je zmínit i fakt, že fyzický model je vždy závislý na konkrétním relačním databázovém systému (Bruckner, Vořišek, & Buchalcevová, 2012).

1.5 Návrh softwarové architektury

Salas (Salas, 1995) uvádí, že aplikace zpravidla obsahuje tři základní skupiny funkcí:

- datové funkce pro ukládání, výběry, aktualizace a ochranu dat,
- věcně orientované funkce zajišťující logiku aplikace,
- komunikační funkce pro zobrazení výsledků na obrazovce.

Základním typem dělení softwarové architektury je dělení na architektu monolitickou, dvouvrstvou a třívrstvou, v závislosti na počtu samostatných programů, které zajišťují uvedené tři základní skupiny funkcí. Zatímco u dvouvrstvé a třívrstvé architektury jsou funkce vykonávány příslušným počtem samostatných programů, pro monolitickou architekturu je typické, že tři skupiny funkcí jsou vzájemně provázány do jednoho programu. (Bruckner, Vořišek, & Buchalcevová, 2012)

Softwarová aplikace respektující monolitickou architekturu může být navrhována s využitím architektonických vzorů, jako je například Model-View-Controller (Figueroa, 2019). Architektonické vzory zobecňují základní strukturu systému skládající se z jednotlivých subsystémů, jejich odpovědností a vzájemných vztahů (Bruckner, Voříšek, & Buchalcevová, 2012).

Architektonický vzor MVC rozděluje aplikaci do tří hlavních komponent: Model představuje datové objekty aplikace, Controller definuje způsob, jak uživatelské rozhraní reaguje na vstup od uživatele a View zajišťuje prezentaci na obrazovce uživatele. Tento architektonický vzor explicitně odděluje část programu zabývající se vstupem a předzpracováním zadávaných příkazů od části reakcí na zadané příkazy a části mající na starosti výstup výsledků, přičemž každou z těchto činností má na starosti samostatná skupina objektů (též modulů). (Pecinovský, 2007)

Mezi benefity využívání architektonického vzoru MVC lze zařadit například přehlednou organizaci i rozsáhlejších webových aplikací, jednoduchou modifikovatelnost, rychlejší vývojový proces, méně duplikovaného kódu a zjednodušení procesu testování aplikace (GeeksforGeeks, 2021).

1.6 Vybrané technologie pro implementaci webového IS

Kapitola je věnována skriptovacímu jazyku PHP a výběru PHP webového frameworku, obsahuje představení vybraného systému řízení báze dat MySQL a front-end frameworku Bootstrap. Kapitola se dále zabývá popisem systémů pro správu balíků Composer a npm.

1.6.1 PHP

PHP je široce používaný skriptovací jazyk pro všeobecné použití, který je vhodný zejména pro vývoj webových aplikací. Program PHP běží na tzv. webovém serveru a prostřednictvím některého z dostupných webových prohlížečů k němu přistupují uživatelé (Sklar, 2018).

Sklar (Sklar, 2018) uvádí následující výhody využití jazyka PHP:

- PHP je open-source projekt – je k dispozici zdarma pro všechny,
- PHP je multiplatformní – lze ho používat s počítačem webového serveru, který běží pod Windows, Mac OS, Linux a mnoha dalšími verzemi Unixu,

- PHP je široce používané – díky početné komunitě uživatelů je k dispozici velké množství knih, časopisů nebo webů, které se věnují výuce PHP a problematice programování v tomto jazyce,
- PHP bylo zrozeno pro webové programování – běžné programovací úlohy (například zpracování formulářů, komunikace s databází, manipulace s datem a časem nebo správa Session a Cookies) jsou v PHP oproti jiným programovacím jazykům snadnější na implementaci.

Poslední zásadní aktualizace jazyka PHP přišla s verzí PHP 8.0 která oproti verzi PHP 7.4 obsahuje například mnoho nových funkcí a optimalizací spolu s vylepšením zpracování chyb a konzistence a využíváním JIT kompilace (PHP Group, 2021).

Na základě statistiky využití server-side programovacích jazyků pro webové stránky vyplývá, že jazyk PHP v současné době využívá 78,1 % všech webových stránek (W3Techs, 2022). Z tohoto důvodu byl i v této bakalářské práci pro implementaci webového informačního systému zvolen programovací jazyk PHP.

1.6.2 Výběr PHP frameworku

PHP frameworky dokáží poskytnout základní strukturu webové aplikace a kompletní sadu rozhraní API, knihoven a rozšíření, a pomáhají tak vývojářům zvýšit produktivitu a omezit opakující se kód v projektu (Laaziri, Benmoussa, Khoulji, & Larbi Kerkeb, 2019). Na trhu existuje mnoho PHP frameworků pro vývoj webových aplikací. Vývojáři se při výběru frameworku mohou setkat například s frameworky Laravel, Symfony, CodeIgniter, Nette, Zend Framework, CakePHP nebo FuelPHP.

Laaziri et al. (Laaziri, Benmoussa, Khoulji, & Larbi Kerkeb, 2019) ve své srovnávací studii porovnávali tři nejpoužívanější PHP frameworky, tj. Symfony, CodeIgniter a Laravel. Kromě obecných porovnávacích kategorií jako jsou například zralost, popularita v komunitě, nabízené služby, kvalita dokumentace či architektura, se zaměřili i na podrobnější výkonnostní testy těchto frameworků. Při testování výkonu autoři sledovali následující kritéria: počet zpracovaných požadavků za sekundu, využití paměti, doba odezvy a množství potřebných souborů. Výsledky této studie ukázaly, že framework Laravel v drtivé většině kategorií ostatní konkurenty překonává.

Pro implementaci webového informačního systému autor na základě výše zmíněné studie zvolí framework Laravel, jehož podrobnějšímu představení je věnována kapitola 1.7.

1.6.3 Výběr relačního databázového systému

Ačkoliv se lze na trhu setkat s velkým množstvím databázových systémů, vybraný webový framework Laravel poskytuje podporu pouze pro čtyři zástupce, konkrétně MySQL, PostgreSQL, SQLite a Microsoft SQL Server.

MySQL je široce používaný systém řízení báze dat založený na relačním databázovém modelu. MySQL je distribuován pro nekomerční využití s otevřeným zdrojovým kódem a řídí se podmínkami General Public License. K vytváření, úpravám, získávání dat z relační databáze a k řízení přístupu uživatelů k databázi je využíván jazyk SQL. Mezi výhody využívání MySQL, který je stále jedním z nejpopulárnějších systémů řízení báze dat na trhu, lze zařadit jeho flexibilitu a jednoduchost používání, relativně vysoký výkon a bezpečnost (Boyett, 2021).

PostgreSQL je výkonný objektově-relační databázový systém, který využívá a rozšiřuje jazyk SQL v kombinaci s mnoha funkcemi, které bezpečně ukládají a škálují nejsložitější datové úlohy. PostgreSQL si dokázal získat dobrou pověst díky své osvědčené architektuře, spolehlivosti, integritě dat, robustní sadě funkcí a rozšiřitelnosti.

Relační databázový systém SQL Server vyvinutý společností Microsoft slouží také pro ukládání a získávání dat. Microsoft SQL Server je nabízen v několika různých edicích určených pro rozličné potřeby a nároky zákazníků.

SQLite je malý a rychlý relační databázový systém obsažený ve speciální knihovně. Na rozdíl od ostatních uvedených zástupců není založen na principu klient-server, ale je přímo zakomponován do koncového programu.

V hodnocení popularity jednotlivých systémů řízení báze dat obdržel z uvedených čtyř zástupců nejvyšší skóre databázový systém MySQL (DB-Engines, 2022).

1.6.4 Bootstrap

Bootstrap je HTML, CSS, and JavaScript frameworkem umožňujícím rychlejší a jednodušší tvorbu responzivních webových aplikací. Bootstrap obsahuje návrhové šablony pro typografii, formuláře, tlačítka, tabulky, navigaci, modální okna, karusely

obrázků a mnoho dalších. Aktuálně nejnovější verze Bootstrap 5 nabízí oproti svým předchůdcům širší nabídku komponent, lepší responzivitu a přechod z jQuery na čistý Javascript. Bootstrap 5 také podporuje nejnovější stabilní verze všech běžně využívaných prohlížečů, kromě Internet Explorer ve verzi 11 a nižší. (W3Schools, 2021)

1.6.5 Composer, npm

Composer je systém pro správu balíků, s jehož pomocí může programátor hledat existující knihovny a integrovat je do své vyvíjené aplikace. Díky využití správce balíků odpadá programátorovi nutnost manuálního stahování souboru archivu obsahujícího knihovnu, rozbalení archivu, ukládání do speciálních adresářů i nutnost manuální správy vzájemných závislostí (Sklar, 2018).

Zkratka npm, odvozená ze slov Node Package Manager, vyjadřuje jak největší registr balíků kódu a open-source software, tak i nástroj správce balíčků a závislostí.

Zatímco Composer je určený spíše pro knihovny PHP, npm je využíván pro získávání knihoven potřebných pro front-end část aplikace.

1.7 Laravel Framework

Kapitola obsahuje bližší představení frameworku Laravel a popis vybraných nástrojů a funkcionalit, které jsou součástí jeho distribuce.

1.7.1 Představení frameworku Laravel

Laravel je PHP webový framework určený pro vývoj webových aplikací založených na architektonickém vzoru MVC (Patel, 2021).

Patel (Patel, 2021) uvádí následující výhody, které framework Laravel nabízí:

- Laravel poskytuje řadu nástrojů, které vývojářům pomáhají zvýšit výkon webové aplikace.
- Laravel je open source projekt s velmi silnou komunitou vývojářů.
- Laravel nabízí podporu pro snadné unit testování jednotlivých komponent webové aplikace v průběhu vývoje.
- S využitím frameworku Laravel lze snadno a rychle vytvořit různé jazykové verze webové aplikace.

- Framework Laravel je navržen tak, aby urychloval proces vývoje webových aplikací, a proto je lze uvést do provozu dříve.

V dnešní době je velmi důležité, aby implementovaná webová aplikace splňovala základní bezpečnostní pravidla. Laravel například využívá předpřipravené SQL dotazy, jejichž cílem je ochránit databázi proti útoku typu SQL Injection. Zároveň framework Laravel nabízí snadný způsob tzv. escapování znaků uživatelského vstupu, čímž dokáže zabránit druhému častému typu útoku na webovou aplikaci, tj. XSS neboli Cross-side scripting. Framework Laravel nabízí i další bezpečnostní prvky jako je podpora pro šifrování hesel nebo ochrana při směrování požadavků. (Patel, 2021)

1.7.2 Blade

Blade je šablonovací systém, který je součástí frameworku Laravel. Šablonovací systém umožňuje pomocí úhledné a zjednodušené syntaxe například vypisovat obsahy proměnných, psát podmínky nebo využívat cykly. Na rozdíl od jiných šablonovacích systémů však Blade programátory neomezuje ani v používání PHP kódu v jednotlivých šablonách. Šablonovací systém Blade nabízí i možnost vytváření vlastních komponent znovupoužitelných napříč webovou aplikací.

1.7.3 Eloquent

Laravel framework nabízí vývojářům nástroj pro objektově-relační mapování zvaný Eloquent. Jeho úkolem je zjednodušit interakci s databází. Při použití nástroje Eloquent má každá tabulka databáze odpovídající objekt typu „model“, který je používán k interakci s danou tabulkou. Kromě načítání záznamů z databázové tabulky umožňují modely Eloquent také vkládat, aktualizovat a mazat záznamy v tabulce.

1.7.4 Artisan

Artisan je nástroj příkazového řádku, který je součástí frameworku Laravel. Skript artisan poskytuje řadu užitečných příkazů, které mohou programátorovi pomoci při vytváření aplikace. Tento nástroj umožňuje jednoduše provádět většinu monotónních a někdy i únavných programátorských úkolů (Patel, 2021).

1.7.5 Middleware

Vrstva Middleware ve frameworku Laravel zajišťuje roli prostředníka mezi přijetím HTTP požadavku a odesláním odpovědi. Účelem použití vrstvy Middleware je kontrola a filtrování HTTP požadavků, které vstupují do webového informačního systému. Příslušný Middleware je zavolán při přijetí HTTP požadavku a na základě logiky, kterou programátor naimplementuje, rozhodne o postoupení požadavku dále, resp. jeho odmítnutí a odeslání příslušné odpovědi zpět uživateli.

1.7.6 Dependency Injection

Technika vkládání závislostí neboli Dependency Injection je založena na vkládání závislostí na jiných třídách do dané třídy zpravidla prostřednictvím jejího konstruktoru. Framework Laravel ke správě závislostí tříd a provádění techniky vkládání závislostí využívá nástroj zvaný Service Container. Pokud třída nemá žádné závislosti nebo závisí pouze na jiných konkrétních třídách (nikoli na rozhraních), nástroj Service Container dokáže vyřešit správu závislostí automaticky. V případě komplikovanějších závislostí na rozhraních či potřebě vytváření návrhových vzorů, je potřeba provést manuální registraci vkládání závislostí prostřednictvím příslušného Service Provider.

1.8 Testování použitelnosti

Kapitola obsahuje popis použitelnosti a jejího testování. Kapitola je dále věnována podrobnějšímu představení metody uživatelského testování použitelnosti a stručně popisuje jednotlivé typy uživatelského testování.

1.8.1 Použitelnost a metody testování

Použitelnost můžeme definovat jako míru, do jaké mohou uživatelé používat daný informační systém, výrobek nebo službu k dosažení určitých cílů s danou účinností, efektivitou a spokojeností v určitém kontextu použití (SEVOCAB, 2017). Za pomoci testování použitelnosti můžeme ověřit, zda se vytvořený informační systém v rukách uživatelů chová tak, jak jsme na základě jeho návrhu předpokládali (Chisnell, 2009).

V závislosti na druhu shromažďovaných dat můžeme metody testování použitelnosti dělit na kvalitativní a kvantitativní. Při kvalitativním uživatelském testování výzkumníci pozorují, jak uživatelé pracují s konkrétními prvky informačního systému a následně

analyzují, které aspekty návrhu jsou problematické a které naopak plní svůj účel správně. Výzkumníci během testování pokládají uživatelům doplňující otázky, které jim pomohou získat podrobnější informace o názoru uživatelů a o problému, na který uživatelé při testování narazili. Naopak kvantitativní uživatelské testování je založené na sběru dat vycházejících z výkonu uživatelů při řešení daného úkolu. Mezi používané metriky patří například čas pro dokončení úkolu, míra úspěšnosti nebo míra chybovosti. Jsou-li kvantitativní data správně interpretována, projeví se jedna z výhod kvantitativního testování, tj. statistická významnost použité metody. (Budiu, 2017)

Mezi metody testování použitelnosti se řadí například (Budinská, 2009):

- uživatelské testování,
- heuristická analýza,
- A/B testování,
- analýza vlastností,
- card sorting,
- dotazníky a ankety.

Metod testování použitelnosti však existuje mnohem více, ale není účelem se v této bakalářské práci všem podrobně věnovat. Vzhledem k faktu, že bude k testování použitelnosti implementovaného informačního systému vybrána metoda uživatelského testování, bude této metodě věnován prostor v následujících podkapitolách.

1.8.2 Uživatelské testování použitelnosti

Uživatelské testování použitelnosti lze definovat jako experiment, během kterého se snažíme s uživateli či zákazníky procházet vytvořený webový informační systém za účelem zjištění, zda je informační systém pro uživatele pochopitelný a použitelný, nebo pro objevení případných úskalí a nedostatků v jeho používání (Voják, Vyhodnocení testu použitelnosti, 2020).

Výběr jednotlivých testerů by měl odpovídat cílové skupině uživatelů vytvořeného informačního systému, kteří mají odpovídající potřeby a požadavky (Voják, Jak dělat uživatelské testování, 2020). Při kvalitativním testování použitelnosti stačí otestovat tři až pět uživatelů, kdy ještě nedochází k plýtvání zdroji (Nielsen, 2001). Nielsen (Nielsen, 2000) ve své studii uvádí, že pouhých pět testerů dokáže odhalit více jak 75 % zásadních

problémů s použitelností. Naopak při kvantitativním testování použitelnosti založeném na sběru metrik je doporučeno testovat 20 uživatelů, abychom získali přiměřeně úzký interval spolehlivosti (Nielsen, 2001).

Před samotným procesem uživatelského testování použitelnosti je důležité si připravit jednotlivé uživatelské scénáře spolu s kontrolními seznamy úkolů reprezentujícími realistické cíle uživatelů (Chisnell, 2009). Samotný proces testování je pak založen na zadávání jednotlivých připravených úkolů testerům, sledování a zaznamenání, jak zvládají tyto úkoly plnit (Voják, Příprava testu použitelnosti, 2020).

Na základě analýzy napozorovaných dat a formulovaných teoriích o příčinách jednotlivých problémů je vhodné určit, jak jednotlivé zjištěné problémy opravit (Chisnell, 2009).

1.8.3 Typy uživatelského testování

Existují čtyři základní typy uživatelského testování použitelnosti. Prvním typem je moderované testování, kdy moderátor zadává respondentovi předem vytvořené úkoly v rámci ucelených scénářů a vede s ním dialog týkající se plnění jednotlivých úkolů. Celý proces je možné nahrávat pro následnou analýzu a odhalování nedostatků. Moderované testování může probíhat osobně i online bez nutnosti osobního setkání. Druhým typem je testování použitelnosti bez moderátora, které je založeno na automatizovaných scénářích a dotaznících. K tomuto účelu lze využít specializované webové aplikace jako jsou například *Maze* nebo *UsabilityHub*. Třetím typem uživatelského testování je tzv. guerillové testování, které je založeno na snadné a nenáročné přípravě testů. Při guerillovém testování lze získat rychlou bezprostřední zpětnou vazbu například od kolegy v práci, členů rodiny nebo hosta u vedlejšího stolu v kavárně. Posledním typem uživatelského testování je dotazníkové testování běžící přímo na webu. Zveřejněním anketních otázek na web lze získat od reálných návštěvníků webu zpětnou vazbu na provedené změny uživatelského rozhraní nebo funkcionality. (Pilka, 2019)

1.9 Současná situace hledání možností spolupráce na projektech

Prvním ze zástupců dostupných informačních systémů souvisejících s tématem hledání možností spolupráce na projektech pro studenty je web *Spolupráce ZČU* (Spolupráce ZČU, 2021). Ten má sloužit k podpoře spolupráce studentů ZČU a firem ze západních

Čech. Cílem tohoto projektu je umožnit studentům vybírat v průběhu studia témata semestrálních nebo kvalifikačních prací a poskytnout firmám prostor pro zveřejňování témat semestrálních a kvalifikačních prací, včetně zveřejňování nabídek odborné praxe pro studenty. Celý projekt je však primárně určen jen pro studenty Fakulty aplikovaných věd ZČU, konkrétně Katedry informační a výpočetní techniky.

Webový informační systém *spoluprace.zcu.cz* nabízí možnost registrace jen pro firmy. Registrace však neprobíhá přes webový formulář. Zástupce firmy musí nejprve stáhnout registrační formulář ve formátu .doc a vyplněný ho zaslat emailem nebo papírově na adresu KIV. Katedra po zpracování žádosti zašle firmě přihlašovací údaje. Po přihlášení firma může vkládat inzeráty témat semestrálních a kvalifikačních prací. Studenti pak využívají systém jen pro prohlížení aktuálně vypsaných inzerátů. V případě semestrálních a kvalifikačních prací má student možnost rezervace vybraného tématu. Ta však neprobíhá na webové stránce samotné. Student je přesměrován na web příslušné katedry, kde danou rezervaci provede vyplněním a odesláním formuláře.

Web Spolupráce ZČU nabízí studentům a absolventům díky napojení na známý kariérní portál *Jobs.cz* i hledání pracovních nabídek zaměstnání a brigád filtrovaných dle studované katedry.

Jak už bylo zmíněno, tento informační systém slouží k navázání spolupráce mezi firmami a ZČU. Spolupráce je však realizována pouze individuálními projekty určenými vždy jen pro jednoho studenta. Tento web tedy neslouží k přímé podpoře spolupráce mezi studenty. Jeho cílem není hledání možností spolupráce na vypsaných týmových studentských projektech.

Ačkoliv na českém trhu žádné jiné informační systémy určené pro hledání možností spolupráce na projektech autor nenašel, v zahraničí se s několika malými projekty s tímto zaměřením setkat můžeme. Mezi vybrané zahraniční zástupce patří americký systém *CollabFinder* a německý sytém *Findnlink*. Oba tyto informační systémy slouží k propojení kreativních lidí, kteří by rádi realizovali určitý projekt, ale chybí jim konkrétní potřebné znalosti a dovednosti, a zájemců, kteří hledají možnosti spolupráce na dostupných týmových projektech a rádi by svými znalostmi a dovednostmi pomohli k realizaci jiných projektů. Platformy *CollabFinder* i *Findnlink* jsou obě založeny na obdobném principu. Prvním krokem k využití služeb obou platforem je úspěšná registrace

a vyplnění potřebných informací v nastavení profilu, jako jsou osobní údaje, popis i znalosti a dovednosti v různých oborech. Následně mohou uživatelé vyhledávat nabídky na spolupráci na zveřejněných projektech, reagovat na ně a projevit svůj zájem o spolupráci s autorem daného projektu. Nástroj *Findnlink* nabízí na rozdíl od svého konkurenta větší možnosti personalizace nových záznamů o projektech. Uživatel může kromě zadání základních údajů jako jsou název a popisek, vkládat k projektům i tagy a obrázky, vybírat různé možnosti nastavení barev pozadí, přidávat odkazy na webové stránky projektů i na sociální sítě. Druhý rozdíl mezi oběma nástroji se týká samotné funkcionality. Systém *Findnlink* umožňuje registrovaným uživatelům procházet nejen zveřejněné projekty a nabídky spolupráce, ale i jednotlivé profily uživatelů. Každý uživatel se tak v systému veřejně prezentuje svým profilem obsahujícím informace jako jsou jméno, osobní popis, znalosti a dovednosti v jednotlivých oborech a realizované projekty.

2 Metodika řešení

Tato část práce je rozdělena do dvou kapitol. Kapitola 2.1 obsahuje shrnutí nástrojů, které budou využity pro implementaci webového informačního systému. V kapitole 2.2 je věnován prostor pro stanovení osnovy řešení a bližšímu popisu jednotlivých bodů, podle kterých bude v následující části práce postupováno.

2.1 Technologie využité pro implementaci

Webový informační systém bude implementován v programovacím jazyce PHP s využitím frameworku Laravel ve verzi 9. Pro ukládání dat bude využit systém řízení báze dat MySQL uplatňující relační datový model. Uživatelské rozhraní informačního systému bude implementováno za pomoci front-end frameworku Bootstrap a šablonovacího systému Blade.

Implementace webového informačního systému bude probíhat na lokálním webovém serveru. Pro jednodušší vývoj a odstranění nutnosti složité konfigurace bude využit multiplatformní softwarový balík XAMPP, který obsahuje webový server Apache, interpret programovacího jazyka PHP 8.0 a systém řízení báze dat MariaDB respektive MySQL. Pro vývoj webového informačního systému bude využito vývojové PhpStorm od společnosti JetBrains.

2.2 Osnova řešení

Při řešení bakalářské práce bude autor postupovat podle následující osnovy, která definuje a popisuje jednotlivé kroky potřebné k dosažení stanoveného cíle:

1. **Vize a rozsah informačního systému**: Kapitola se bude zabývat představením podnikatelských požadavků popisujících cíle, kterých by mělo být s využitím implementovaného webového informačního systému dosaženo.
2. **Specifikace požadavků**: Definovány budou jak funkční (behaviorální) požadavky definující funkcionalitu informačního systému, tak i nefunkční požadavky definující omezení, které by informační systém měl splnit.
3. **Případy užití**: Tato kapitola bude věnována jednotlivým případům užití, které budou popsány slovně i graficky s využitím UML diagramu případů užití.

4. **Analýza**: Kapitola se bude zabývat definicí procesní orientace navrhovaného webového informačního systému.
5. **Návrh datové architektury**: Datová architektura bude v první fázi navržena s využitím konceptuálního modelu, který bude sloužit pro prvotní definici objektů. Konceptuální model bude transformován nejprve do relačního modelu, následně do fyzického modelu a výsledné tabulky budou podrobněji popsány.
6. **Návrh softwarové architektury**: Softwarová architektura webového informačního systému bude navržena s využitím architektonického vzoru MVC. Podrobněji bude popsán postup, podle kterého bude architektura informačního systému pracovat.
7. **Návrh uživatelského rozhraní**: Návrh uživatelského rozhraní webového informačního systému bude spočívat ve tvorbě tzv. wireframes popisujících strukturu a rozložení jednotlivých prvků webové aplikace.
8. **Implementace webového informačního systému**: V této kapitole bude podrobněji popsána implementace výsledného webového informačního systému. Prostor bude věnován bližšímu představení implementačních detailů klíčových funkcionalit systému.
9. **Testování použitelnosti**: Součástí řešení bude i testování použitelnosti navrženého a implementovaného informačního systému. Použitelnost bude otestována metodou uživatelského testování v moderované formě založené na postupném předkládání předem vytvořených scénářů jednotlivým zástupcům cílové skupiny uživatelů. Při testování budou sbírána data kvantitativního charakteru.

3 Řešení

Tato část je rozdělena do deseti kapitol. Kapitola 3.1 se zabývá vizí a rozsahem navrhovaného informačního systému. Kapitola 3.2 obsahuje specifikaci požadavků. Kapitola 3.3 je věnována popisu případů užití. Kapitola 3.4 obsahuje analýzu procesní orientace informačního systému a popis stavů objektů. Kapitola 3.5 se zabývá návrhem datové architektury. Kapitola 3.6 obsahuje návrh softwarové architektury. Kapitola 3.7 představuje návrh uživatelského rozhraní. Kapitola 3.8 popisuje implementaci webového informačního systému. Kapitola 3.9 je věnována testování použitelnosti. Kapitola 3.10 obsahuje návrhy na rozšíření funkcionality implementovaného informačního systému.

3.1 Vize a rozsah informačního systému

Na základě rešerše současné situace a dostupných nástrojů týkajících se hledání možností spolupráce na projektech vyplývá, že na trhu není k dispozici informační systém, který by byl přímo určen pro podporu hledání možností spolupráce studentů na týmových projektech. Ačkoliv se na trhu můžeme setkat s podobnými nástroji, žádný není určen přímo pro akademické prostředí s cílem podpořit a rozvinout mezi vysokoškolskými studenty zájem o spolupráci na týmových projektech. Navržený informační systém by vedle zveřejňování a hledání nabídek spolupráce na týmových projektech mohl sloužit i pro prezentaci jednotlivých projektů, na kterých studenti pracují nebo které již úspěšně dokončili. Kombinace těchto dvou základních funkcionalit tak přináší příležitost navrhnout a implementovat nový webový informační systém.

Webový informační systém nazvaný ***studenti | spolu*** bude určen pro studenty vysokých škol, kteří potřebují pro své autorské projekty získat nové spolupracovníky nebo mají zájem aplikovat své znalosti a dovednosti při práci na týmových projektech ostatních autorů. Předpokladem pro využití navrženého informačního systému je, že se studentovi zrodí v hlavě nápad na nový týmový projekt, na kterém by chtěl začít spolupracovat se zájemci z řad ostatních studentů. Tento student bude moci využít výsledný webový informační systém k vytvoření záznamu o zahájení práce na novém týmovém projektu spolu se zveřejněním nabídek spolupráce pro ostatní uživatele dle potřebných znalostí a dovedností. Záznam o novém týmovém projektu, který daný student vytvoří, bude zároveň sloužit jako prezentace jeho práce ostatním uživatelům systému. Uživatelé, kteří

nemají vlastní nápad na nový projekt, ale přesto by rádi svými znalostmi a dovednostmi přispěli k práci na jiném týmovém projektu, budou moci webový informační systém využít pro hledání nabídek spolupráce na týmových projektech vytvořených ostatními uživateli. Na nabídku, která bude odpovídat jejich požadavkům budou moci zareagovat zasláním žádosti o spolupráci autorovi projektu. Ten následně rozhodne, zda žádost přijme a získá tak nového člena týmu.

3.2 Specifikace požadavků

3.2.1 Funkční (behaviorální) požadavky

FP-1: Informační systém musí umožnit registraci nového uživatele. Mezi povinné registrační údaje bude patřit jméno a příjmení uživatele, e-mail a heslo. Při registraci bude dále potřeba vybrat jednotlivé obory, ve kterých mají znalosti a dovednosti potřebné pro budoucí spolupráce na týmových projektech.

FP-2: Informační systém bude umožňovat neautentizovanému uživateli přihlášení k aktivnímu účtu. Autentizace při přihlašování bude realizována porovnáním zadaného unikátního uživatelského jména a hesla s hodnotami uloženými v databázi.

FP-3: Informační systém musí umožnit přihlášeným uživatelům měnit zadané osobní informace v nastavení jejich profilu.

FP-4: Informační systém musí umožnit přihlášeným uživatelům vytvořit záznam o novém týmovém projektu. Mezi informace, které bude potřeba vyplnit, bude patřit název projektu spolu s jeho abstraktem a popisem.

FP-5: Informační systém musí umožnit přihlášeným uživatelům s rolí *Autor / Spolupracovník* vkládat k vytvořenému týmovému projektu soubory přibližující dosavadní práci na projektu.

FP-6: Informační systém musí umožnit přihlášenému uživateli s rolí *Autor* vytvořit a zveřejnit nabídky na spolupráci v konkrétních oborech dle požadovaných znalostí a dovedností.

FP-7: Informační systém musí umožnit přihlášeným uživatelům s rolí *Autor* upravovat nebo mazat jejich vytvořené záznamy o týmových projektech včetně nabídek na spolupráci pro ostatní uživatele.

FP-8: Informační systém musí umožnit přihlášeným i nepřihlášeným uživatelům zobrazit všechny zveřejněné záznamy o týmových projektech.

FP-9: Informační systém musí umožnit přihlášeným uživatelům zobrazit zveřejněné nabídky spolupráce na týmových projektech v konkrétních oborech.

FP-10: Informační systém musí umožnit přihlášeným uživatelům posílat žádosti o spolupráci na vybraném týmovém projektu.

FP-11: Informační systém musí umožnit přihlášeným uživatelům s rolí *Autor* zobrazit a spravovat přijaté žádosti o spolupráci na jejich autorských týmových projektech.

FP-12: Informační systém musí umožnit přihlášeným uživatelům s rolí *Autor* schválit nebo odmítnout žádosti o spolupráci na jejich autorském týmovém projektu, které ostatní uživatelé odeslali.

FP-13: Informační systém musí umožnit přihlášeným uživatelům zobrazit a spravovat odeslané žádosti o spolupráci na týmových projektech ostatních uživatelů. Odeslané žádosti by měly jít upravovat a případně mazat, pokud jsou ve stavu *Čeká na schválení*.

FP-14: Informační systém musí umožnit přihlášeným uživatelům prohlížet seznam všech aktivních uživatelů a zobrazit si podrobné informace o vybraném uživateli.

FP-15: Informační systém by měl umožnit kontaktovat ostatní aktivní uživatele pomocí uvedené e-mailové adresy s využitím přesměrování na e-mailovou aplikaci uživatele.

FP-16: Informační systém bude přihlášeným uživatelům autorizovaným jako Administrátor nebo SuperAdministrátor umožňovat administraci celé aplikace.

FP-17: Informační systém musí umožnit přihlášeným uživatelům autorizovaným jako Administrátor nebo SuperAdministrátor aktivovat a deaktivovat uživatele.

FP-18: Informační systém bude přihlášeným uživatelům autorizovaným jako SuperAdministrátor umožňovat měnit práva jednotlivým uživatelům.

3.2.2 Nefunkční požadavky

NP-1: Informační systém bude implementován jako webová aplikace. Uživatel bude se systémem komunikovat s využitím internetového prohlížeče.

NP-2: Informační systém bude pro svoji datovou základnu využívat relační databázový systém. Všechna potřebná data budou uložena v relační databázi.

NP-3: Hesla pro přihlašovaní uživatelů budou v relační databázi uložena v zašifrované podobě.

NP-4: Uživatelem informačního systému s nejvyššími právy bude Super Administrátor. Díky tomu bude zajištěna vzájemná neovlivnitelnost administrátorů.

NP-5: Obsah webového informačního systému bude pouze v českém jazyce.

NP-6: Pro všechny uživatele webového informačního systému bude k dispozici uživatelská dokumentace. Ta bude obsahovat seznámení s webovým informačním systémem a bude zároveň popisovat důležité funkce systému.

NP-7: Grafické uživatelské rozhraní webového informačního systému bude plně responzivní.

3.3 Případy užití

Kapitola obsahuje představení aktérů komunikujících se systémem pomocí případů užití a je věnována podrobnému popisu jednotlivých případů užití.

3.3.1 Aktéři

S navrženým informačním systémem budou interagovat následující aktéři:

- **Neregistrovaný uživatel** – uživatel, který ještě nemá vytvořený uživatelský účet.
- **Aktivní uživatel** – uživatel, který úspěšně dokončil registraci a v nastavení profilu zadal svůj popis a vybal obory, ve kterých má znalosti a dovednosti.
- **Autor projektu** – aktivní uživatel, který vytvořil týmový projekt.

- **Spolupracovník** – aktivní uživatel, jehož odeslaná žádost o spolupráci na týmovém projektu byla autorem projektu přijata (tj. je součástí týmu, který spolupracuje na některém z týmových projektů).
- **Administrátor** – uživatel s nadřízenými právy oproti ostatním uživatelům.
- **Super Administrátor** – uživatel v informačním systému, s nejvyššími právy.

3.3.2 Seznam případů užití

UC-1: **Registrovat se do informačního systému**

Aktéři: Neregistrovaný uživatel

Scénář:

1. Neregistrovaný uživatel vyplní všechny povinné údaje v registračním formuláři.
2. Informační systém zkontroluje, zda se zadaná e-mailová adresa nebo heslo neshoduje s některým ze záznamů již zaregistrovaného uživatele. V případě, že zadaná e-mailová adresa a heslo jsou unikátní, systém provede registraci nového uživatele a jeho údaje uloží do databáze.

UC-2: **Přihlásit se do informačního systému**

Aktéři: Aktivní uživatel

Scénář:

1. Aktivní uživatel vyplní přihlašovací formulář.
2. Informační systém zkontroluje zadané přihlašovací údaje. V případě, že se shodují se záznamem v databázi, systém provede přihlášení uživatele.

UC-3: **Vytvořit nový týmový projekt**

Aktéři: Aktivní uživatel

Scénář:

1. Aktivní uživatel vyplní formulář určený pro vytvoření nového týmového projektu.
2. Aktivní uživatel požádá systém o vytvoření nového týmovém projektu.
3. Systém uloží nový projekt do databáze.

UC-4: **Upravit vytvořený týmový projekt**

Aktéři: Autor projektu

Scénář:

1. Autor projektu požádá systém o zobrazení autorských týmových projektů.
2. Autor projektu vybere konkrétní autorský týmový projekt.
3. Autor projektu upraví informace o týmovém projektu v zobrazeném formuláři.
4. Autor projektu požádá systém o úpravu daného záznamu o týmovém projektu.
5. Systém uloží úpravu informací o týmovém projektu do databáze.

UC-5: **Nahrát soubory k vytvořenému týmovému projektu**

Aktéři: Autor projektu, Spolupracovník

Scénář:

1. Autor projektu nebo Spolupracovník požádá systém o zobrazení autorských nebo spolupracovnických týmových projektů.
2. Autor projektu nebo Spolupracovník vybere konkrétní projekt.
3. Autor projektu nebo Spolupracovník vybere soubory, které chce k vybranému projektu nahrát.
4. Systém uloží nahrané soubory na server a údaje o souborech uloží do databáze.

UC-6: **Zveřejnit nabídky spolupráce v konkrétním oboru na týmovém projektu**

Aktéři: Autor projektu

Scénář:

1. Autor projektu požádá systém o zobrazení autorských týmových projektů.
2. Autor projektu vybere konkrétní autorský týmový projekt.
3. Autor projektu vyplní informace o nové nabídce spolupráce v zobrazeném formuláři.
4. Autor projektu požádá systém o uložení a zveřejnění nabídky na spolupráci.

5. Systém uloží nabídky na spolupráci pro daný týmový projekt do databáze a zveřejní je ostatním uživatelům.

UC-7: **Prohlížet své týmové projekty**

Aktéři: Autor projektu, Spolupracovník

Scénář:

1. Autor projektu nebo Spolupracovník požádá systém o zobrazení autorských nebo spolupracovnických týmových projektů.
2. Systém zobrazí seznam autorských nebo spolupracovnických projektů.

UC-8: **Prohlížet všechny zveřejněné týmové projekty**

Aktéři: Aktivní uživatel, Neregistrovaný uživatel

Scénář:

1. Aktivní uživatel nebo Neregistrovaný uživatel požádá systém o zobrazení všech zveřejněných týmových projektů.
2. Systém zobrazí seznam všech zveřejněných týmových projektů.

UC-9: **Vyhledat nabídky spolupráce na týmových projektech**

Aktéři: Aktivní uživatel

Scénář:

1. Aktivní uživatel požádá systém o zobrazení všech aktivních nabídek na spolupráci na týmových projektech ostatních uživatelů.
2. Systém zobrazí uživateli seznam nabídek na spolupráci na týmových projektech.

UC-10: **Odeslat žádost o spolupráci reagující na aktivní nabídku spolupráce**

Aktéři: Aktivní uživatel

Scénář:

1. Aktivní uživatel požádá systém o zobrazení všech aktivních nabídek na spolupráci na týmových projektech ostatních uživatelů.
2. Aktivní uživatel vybere konkrétní nabídku spolupráce, na kterou chce reagovat.

3. Aktivní uživatel vyplní v zobrazeném formuláři určeném pro poslání žádosti o spolupráci zprávu určenou pro autora projektu.
4. Systém uloží nově vytvořenou žádost o spolupráci do databáze.

UC-11: **Zobrazit přijaté žádosti o spolupráci**

Aktéři: Autor projektu

Scénář:

1. Autor projektu požádá systém o zobrazení přijatých žádostí o spolupráci reagujících na vytvořené nabídky spolupráce.
2. Systém zobrazí seznam přijatých žádostí o spolupráci.

UC-12: **Schválit žádost o spolupráci na týmovém projektu**

Aktéři: Autor projektu

Scénář:

1. Autor projektu si zobrazí přijaté žádosti o spolupráci na svých vytvořených projektech.
2. Autor projektu požádá systém o schválení konkrétní přijaté žádosti o spolupráci na týmovém projektu v daném oboru.
3. Systém změní v daném týmovém projektu roli uživatele, jehož žádost o spolupráci byla přijata, na Spolupracovník. Systém změní stav žádosti na *Schválená.*

UC-13: **Zamítnout žádost o spolupráci na týmovém projektu**

Aktéři: Autor projektu

Scénář:

1. Autor projektu si zobrazí přijaté žádosti o spolupráci na svých vytvořených projektech.
2. Autor projektu požádá systém o odmítnutí konkrétní přijaté žádosti o spolupráci na týmovém projektu v daném oboru.
3. Systém změní stav žádosti na *Zamítnutá*.

UC-14: **Zobrazit odeslané žádosti o spolupráci**

Aktéři: Aktivní uživatel

Scénář:

1. Autor projektu požádá systém o zobrazení odeslaných žádostí o spolupráci reagujících na vytvořené nabídky spolupráce.
2. Systém zobrazí uživateli seznam odeslaných žádostí o spolupráci.

UC-15: **Prohlížet všechny aktivní uživatele systému**

Aktéři: Aktivní uživatel

Scénář:

1. Aktivní uživatel požádá systém o zobrazení všech aktivních uživatelů systému.
2. Systém zobrazí uživateli seznam aktivních uživatelů systému.
3. Aktivní uživatel si může zobrazit podrobnější informace o vybraném uživateli ze seznamu.

UC-16: **Kontaktovat ostatní uživatele e-mailem**

Aktéři: Aktivní uživatel

1. Aktivní uživatel si zobrazí seznam všech aktivních uživatelů systému.
2. Aktivní uživatel požádá systém o zobrazení informací o konkrétním uživateli.
3. Aktivní uživatel vybere možnost kontaktovat daného uživatele s využitím přesměrování na e-mailovou aplikaci.

UC-17: **Upravit osobní informace v nastavení profilu**

Aktéři: Aktivní uživatel

Scénář:

1. Aktivní uživatel požádá systém o zobrazení svého uživatelského profilu.
2. Aktivní uživatel upraví informace týkající se uživatelského profilu v zobrazeném formuláři.
3. Aktivní uživatel požádá systém o úpravu informací v osobním profilu.

4. Systém uloží upravené informace o uživateli do databáze.

UC-18: **Spravovat informační systém**

Aktéři: Administrátor, Super Administrátor

Scénář:

1. Administrátor nebo Super Administrátor požádá systém o zobrazení prostředí pro administraci aplikace.
2. Administrátor nebo Super Administrátor provede potřebné změny (týkající se např. profilů uživatelů, vytvořených záznamů o týmových projektech, nabídek spolupráce nebo žádostí o spolupráci).
3. Administrátor nebo Super Administrátor požádá systém o uložení změn do databáze.

UC-19: **Změnit práva uživatele**

Aktéři: Super Administrátor

Scénář:

1. Super Administrátor požádá v prostředí pro administraci aplikace systém o zobrazení aktivních uživatelů systému.
2. Super Administrátor vybere konkrétního uživatele, jehož práva chce změnit.
3. Super Administrátor provede změnu práva uživatele.
4. Super Administrátor požádá systém o změnu práv uživatele.
5. Systém uloží provedenou změnu práv uživatele do databáze.

UC-20: **Aktivace uživatele**

Aktéři: Administrátor, Super Administrátor

Scénář:

1. Administrátor nebo Super Administrátor požádá v prostředí pro administraci aplikace systém o zobrazení aktivních uživatelů systému.
2. Administrátor nebo Super Administrátor vybere konkrétního uživatele, jehož stav chce změnit.

3. Administrátor nebo Super Administrátor provede změnu stavu uživatele na „Aktivní“.
4. Systém uloží provedenou změnu stavu uživatele do databáze.

UC-21: **Deaktivace uživatele**

Aktéři: Administrátor, Super Administrátor

Scénář:

1. Administrátor nebo Super Administrátor požádá v prostředí pro administraci aplikace systém o zobrazení aktivních uživatelů systému.
2. Administrátor nebo Super Administrátor vybere konkrétního uživatele, jehož stav chtějí změnit.
3. Administrátor nebo Super Administrátor provede změnu stavu uživatele na „Neaktivní“.
4. Systém uloží provedenou změnu stavu uživatele do databáze.

3.3.3 Diagram případů užití

Pro celkovou grafickou reprezentaci jednotlivých aktérů a jejich případů užití byl vytvořen UML diagram případů užití.

Obrázek 1: UML diagram případů užití

Zdroj: vlastní zpracování, 2022

3.4 Analýza

Kapitola je věnována rozboru a znázornění procesů, které bude navržený a implementovaný webový informační systém podporovat. Kapitola dále obsahuje definici stavů vybraných objektů informačního systému.

3.4.1 Procesní orientace informačního systému

Procesní orientaci navrhovaného informačního systému můžeme popsat ze dvou základních pohledů. K zobrazení těchto procesů jsou využity UML diagramy aktivit.

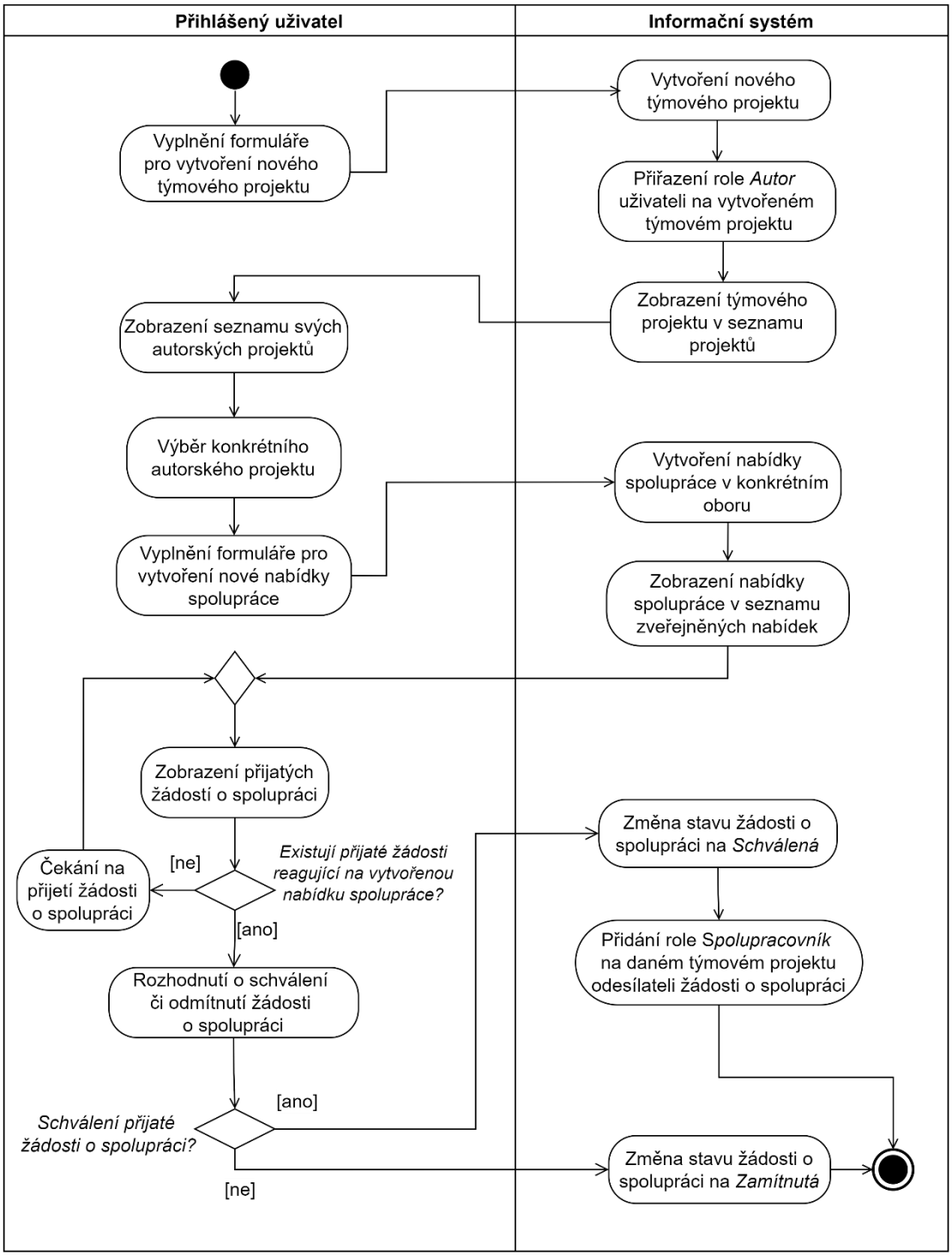
Proces z pohledu studenta jako autora nového projektu:

Diagram aktivit (Obrázek 2: Proces z pohledu studenta jako autora nového projektu) zobrazuje proces z pohledu studenta jako autora nového projektu. Tento pohled popisuje proces od vytvoření nového záznamu o týmovém projektu, přes zveřejnění nabídek spolupráce v konkrétních oborech dle požadovaných znalostí a dovedností, až po rozhodnutí o schválení či odmítnutí přijatých žádostí o spolupráci reagujících na zveřejněnou nabídku spolupráce.

Proces z pohledu studenta jako zájemce o spolupráci:

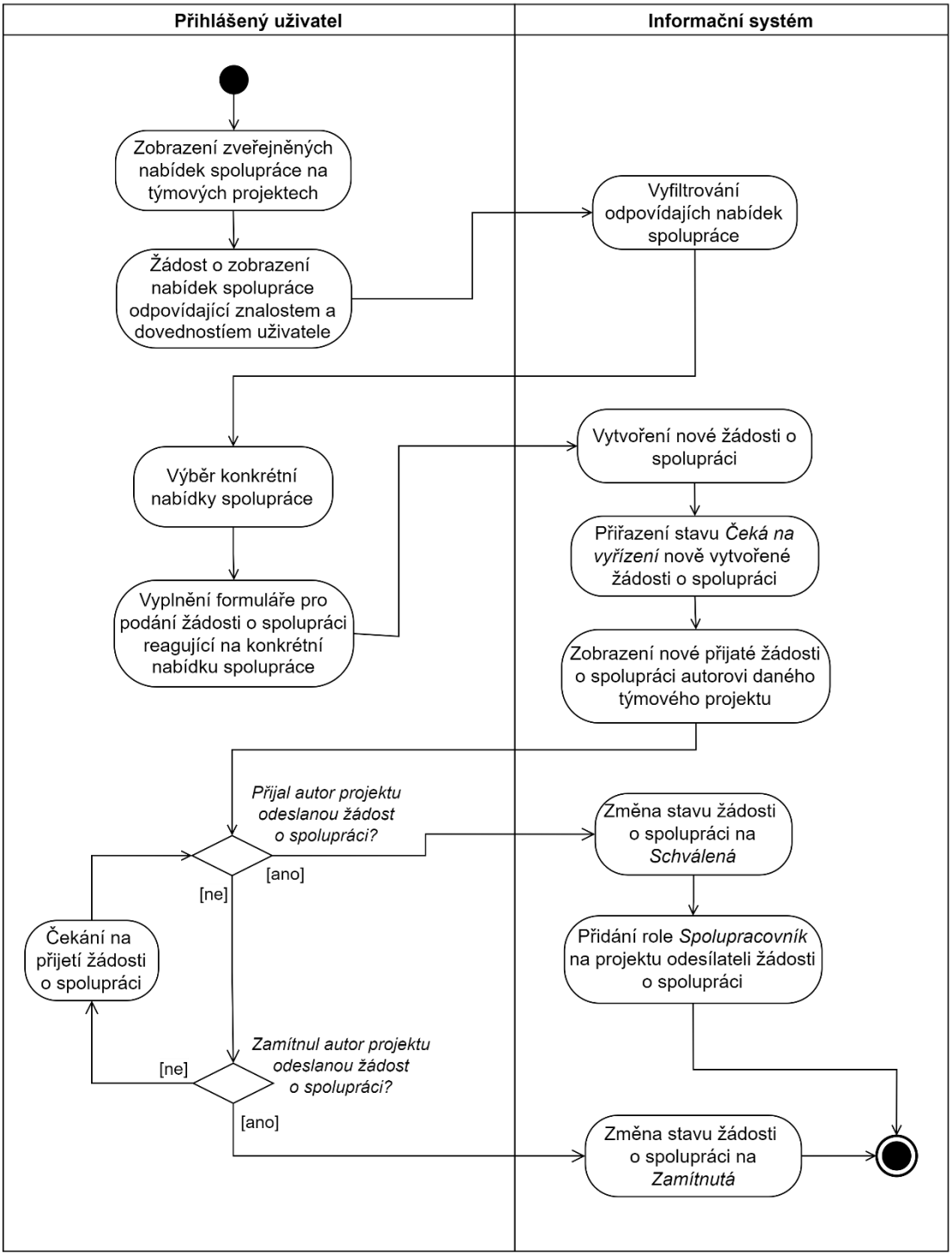
Diagram aktivit (Obrázek 3: Proces z pohledu studenta jako zájemce o spolupráci) zobrazuje proces z pohledu studenta jako zájemce o spolupráci. Tento pohled popisuje proces od zobrazení zveřejněných nabídek spolupráce, výběr konkrétní nabídky odpovídající oboru znalostí a dovedností studenta, přes podání žádosti o spolupráci, až po získání role *Spolupracovník* na daném týmovém projektu v případě rozhodnutí o schválení žádosti ze strany autora projektu.

Obrázek 2: Proces z pohledu studenta jako autora nového projektu



Zdroj: vlastní zpracování, 2022

Obrázek 3: Proces z pohledu studenta jako zájemce o spolupráci



Zdroj: vlastní zpracování, 2022

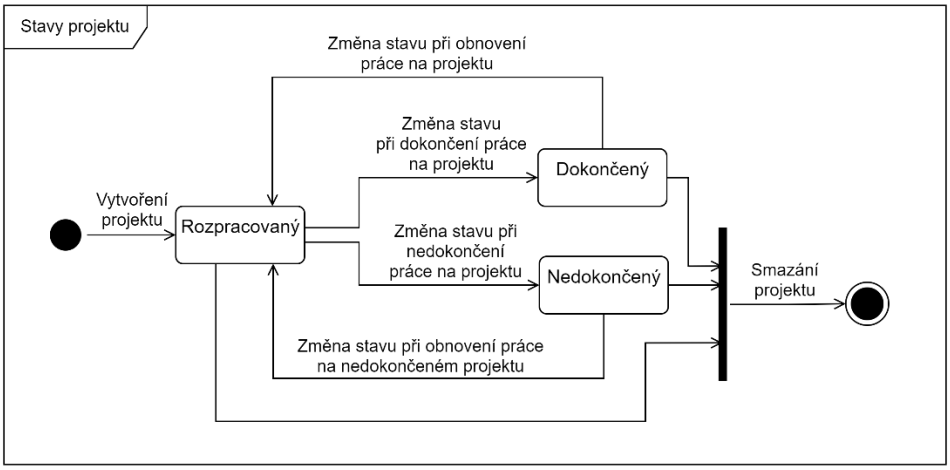
3.4.2 Stavy vybraných objektů informačního systému

Vybrané objekty (Projekt, Nabídka spolupráce, Žádost o spolupráci, Uživatel) používané v navrhovaném informačním systému nabývají v průběhu svého životního cyklu různých stavů. Pro lepší představu, jakých stavů mohou tyto objekty nabývat a jakým způsobem mezi nimi mohou přecházet, slouží následující UML stavové diagramy.

Stavy projektu:

Vytvořený projekt může během svého životního cyklu nabývat stavů: „Rozpracovaný“, „Dokončený“ a „Nedokončený“.

Obrázek 4: Stavový diagram projektu

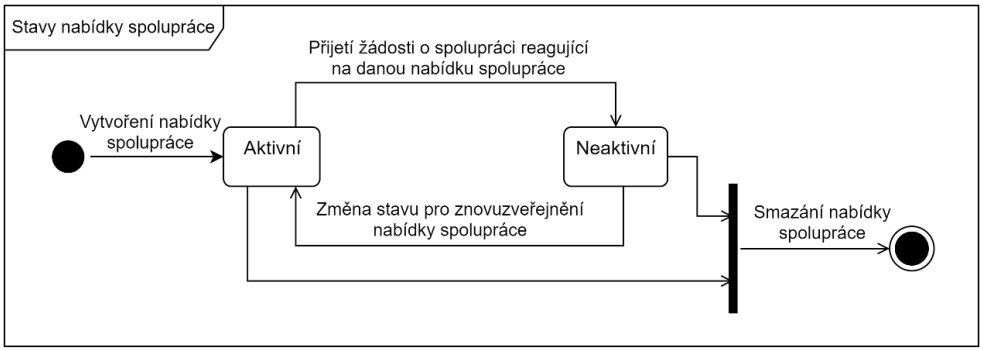


Zdroj: vlastní zpracování, 2022

Stavy nabídky spolupráce:

Vytvořená a zveřejněná nabídka spolupráce může během svého životního cyklu nabývat stavů: „Aktivní“ a „Neaktivní“.

Obrázek 5: Stavový diagram nabídky spolupráce

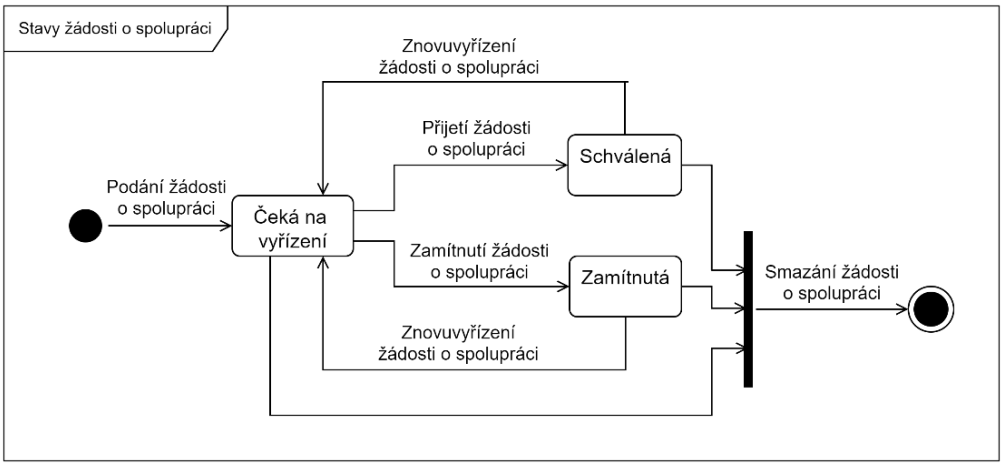


Zdroj: vlastní zpracování, 2022

Stavy žádosti o spolupráci:

Odeslaná žádost o spolupráci může během svého životního cyklu nabývat stavů: „Čeká na vyřízení“, „Schválená“ a „Zamítnutá“.

Obrázek 6: Stavový diagram žádosti o spolupráci

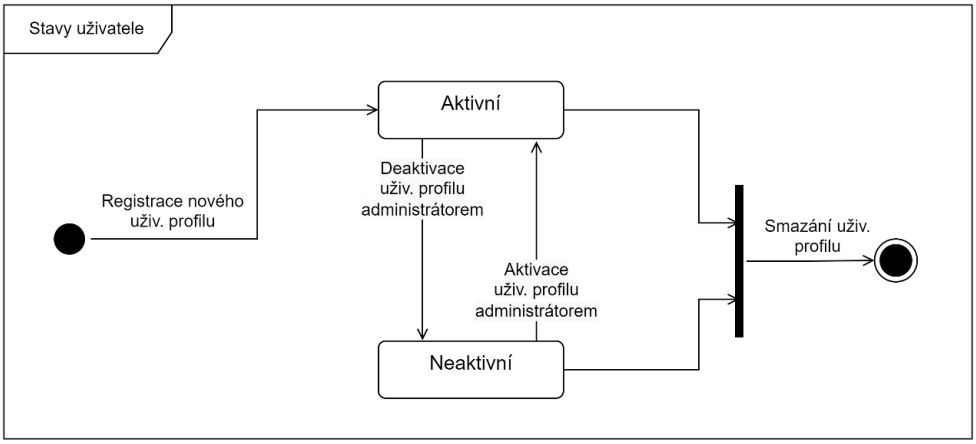


Zdroj: vlastní zpracování, 2022

Stavy uživatele:

Registrovaný uživatel může během svého životního cyklu nabývat stavů: „Aktivní“ a „Neaktivní“.

Obrázek 7: Stavový diagram uživatele



Zdroj: vlastní zpracování, 2022

3.5 Návrh datové architektury

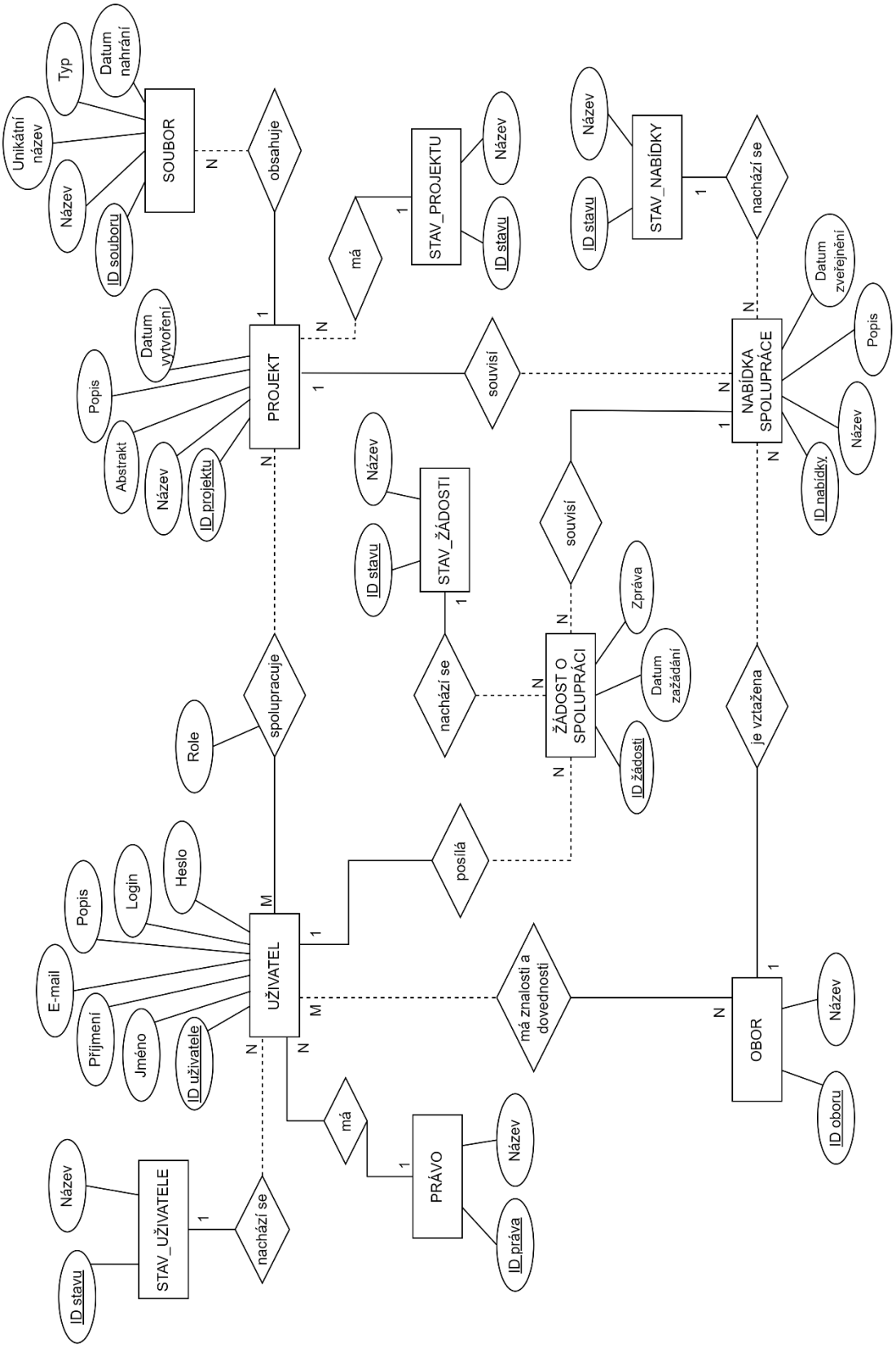
Kapitola je věnována návrhu datové architektury a podrobnějšímu představení všech tabulek výsledného datového modelu.

3.5.1 Konceptuální datový model

Konceptuální datový model (Obrázek 8: Konceptuální datový model) je navržen formou ERA diagramu s využitím Chenovy notace. V tomto modelu na nejvyšší úrovni abstrakce jsou definovány základní potřebné entitní množiny, jejich atributy spolu se vzájemnými vazbami. Některé z uvedených vazeb mezi entitními množinami mají kardinalitu M:N. Tyto vazby budou následně rozloženy v datovém modelu s nižší úrovní abstrakce, tj. v logickém datovém modelu. Konceptuální datový model je navržen na základě následujícího textového popisu problematiky.

Informační systém a jeho funkcionality využívají jeho uživatelé, kteří mají přidělena práva definující rozsah jejich pravomocí. Uživatelé mají znalosti a dovednosti v určitých oborech a spolupracují na týmových projektech. Nový týmový projekt je vždy vytvořen jeho autorem. K vytvořenému týmovém projektu bude možné vkládat soubory, které mohou obsahovat doplňující informace o práci na realizaci projektu a sloužit pro jeho názornější prezentaci. Autor projektu může vytvářet nabídky spolupráce na daném týmovém projektu pro ostatní uživatele. Každá nabídka spolupráce je vztažena k některému z oborů, ve kterém by hledaný spolupracovník měl mít znalosti a dovednosti. Ostatní uživatelé, kteří mají o spolupráci zájem, mají možnost na každou nabídku spolupráce reagovat zasláním žádostí o spolupráci. Autor daného týmového projektu žádost o spolupráci posoudí a rozhodne o jejím přijetí nebo odmítnutí. V případě přijetí žádosti o spolupráci se uživatel stává spolupracovníkem na daném týmovém projektu. Projekty, nabídky spolupráce, žádosti o spolupráci i uživatelé jsou popsány stavem, ve kterém se aktuálně nacházejí.

Obrázek 8: Konceptuální datový model



Zdroj: vlastní zpracování, 2022

3.5.2 Logický datový model

Pro realizaci datové základny je v navrhovaném informačním systému zvoleno technologické řešení ve formě relační databáze. Jednotlivé entitní množiny a jejich atributy definované na konceptuální úrovni v předchozí podkapitole jsou tak v rámci logického datovém modelu převedeny na relace (tabulky). Rozkladem dvou vazeb s kardinalitou M:N z konceptuální úrovně datového modelu vznikly dvě nové tabulky. Jedna tabulka vznikla i pro reprezentaci role, kterou uživatel na daném týmovém projektu zastává. Samozřejmostí je navržení a případná dekompozice relací tak, aby celý relační datový model splňoval 3NF.

Logický datový model je prezentován s využitím relačních schémat v textové formě:

Uživatel (id uživatel, jméno, příjmení, e-mail, popis, login, heslo, id právo, id stav)

Stav uživatele (id stav, název)

Právo (id právo, název)

Obor (id obor, název)

Uživatel_Obor (id uživatel, id obor)

Spolupráce (id uživatel, id projekt, id role)

Role (id role, název)

Projekt (id projekt, název, abstrakt, popis, datum vytvoření, id stav)

Stav projektu (id stav, název)

Soubor (id soubor, název, unikátní název, typ, datum nahrání, id projekt)

Nabídka spolupráce (id nabídka, název, popis, datum zveřejnění, id obor, id projekt, id stav)

Stav nabídky (id stav, název)

Žádost o spolupráci (id žádost, zpráva, datum vytvoření, id uživatel, id nabídka, id stav)

Stav žádosti (id stav, název)

3.5.3 Fyzický datový model

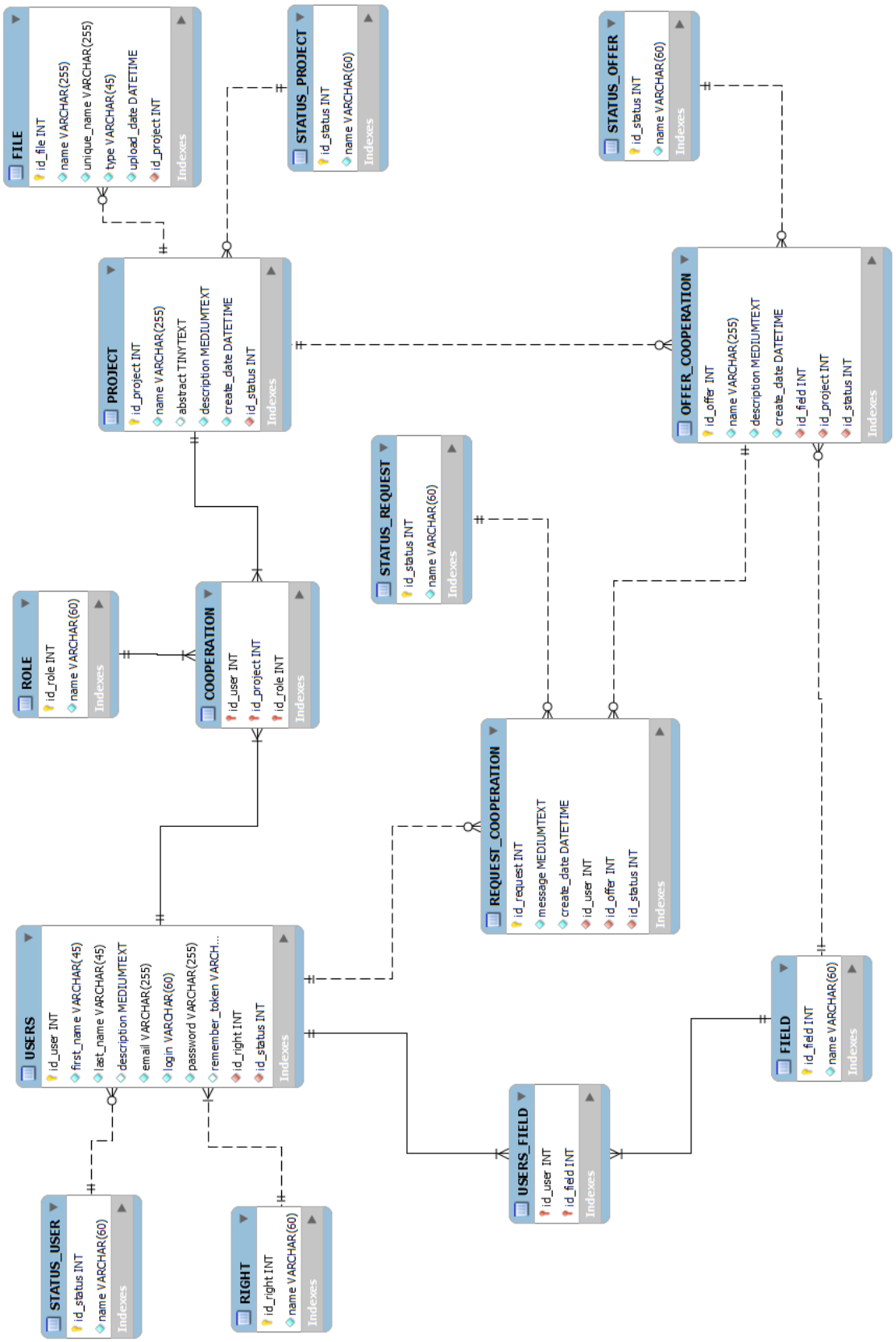
Fyzický datový model (Obrázek 9: Fyzický datový model) je posledním krokem při návrhu datové architektury informačního systému. U všech relací jsou doplněny informace související s konkrétním implementačním prostředím vybraným pro realizaci datové základny. Jednotlivé relace mají definované entitní integritní omezení s využitím primárních klíčů, doménové integritní omezení volbou vhodného datového typu pro jednotlivé atributy a referenční integritní omezení ve formě cizích klíčů reprezentujících vztahy mezi relacemi. Na základě navrženého fyzického datového modelu bude následně vygenerován SQL skript s DDL příkazy pro vytvoření výsledné struktury databáze. Jelikož při implementaci webového informačního systému bude databáze vytvořena v anglickém jazyce, jsou v angličtině definovány i tabulky fyzického modelu.

Přehled definovaných tabulek:

- USERS – Tabulka slouží pro ukládání informací o registrovaných uživatelích. Mezi uchovávané informace patří jméno, příjmení, popis uživatele, e-mailová adresa, login a heslo. Heslo uživatele je v databázi uchováno v zašifrované podobě.
- STATUS_USER – Tabulka typu číselník slouží pro uchování stavů, ve kterých se může nacházet uživatel systému.
    - Hodnoty uložené v číselníku: Aktivní, Neaktivní
- RIGHT – Tabulka typu číselník uchovává názvy práv uživatelů.
    - Hodnoty uložené v číselníku: Super Administrátor, Administrátor, Student
- FIELD – Tabulka typu číselník uchovává obory, ve kterých mohou mít uživatelé znalosti a dovednosti. K těmto oborům jsou zároveň vztaženy i nabídky spolupráce na týmových projektech.
- USERS_FIELD – Tabulka vzniklá rozkladem jedné z vazeb M:N. Tabulka slouží pro uchování informací o tom, v jakých oborech má který uživatel znalosti a dovednosti.

- COOPERATION – Tabulka vzniklá rozkladem druhé z vazeb M:N. Tabulka slouží pro uchování informací o tom, na jakých týmových projektech jednotliví uživatelé spolupracují a v jaké roli.
- ROLE – Tabulka typu číselník uchovává názvy rolí, které může uživatel zastávat v rámci spolupráce na týmovém projektu.
    - Hodnoty uložené v číselníku: Autor, Spolupracovník
- PROJECT – Tabulka slouží pro ukládání informací o vytvořených týmových projektech.
- STATUS_PROJECT – Tabulka typu číselník slouží pro uchování stavů, ve kterých se může nacházet týmový projekt.
    - Hodnoty uložené v číselníku: Rozpracovaný, Dokončený, Nedokončený
- FILE – Tabula slouží pro ukládání názvů a typů souborů přiřazených k jednotlivým projektům.
- OFFER_COOPERATION – Tabulka slouží pro ukládání informací o nabídkách spolupráce na týmových projektech.
- STATUS_OFFER – Tabulka typu číselník slouží pro uchování stavů, ve kterých se může nacházet nabídka spolupráce.
    - Hodnoty uložené v číselníku: Aktivní, Neaktivní
- REQUEST_COOPERATION – Tabulka slouží pro ukládání informací o žádostech o spolupráci, které slouží k reakci na zveřejněné nabídky spolupráce na týmových projektech.
- STATUS_REQUEST – Tabulka typu číselník slouží pro uchování stavů, ve kterých se může nacházet žádost o spolupráci.
    - Hodnoty uložené v číselníku: Čeká na vyřízení, Schválená, Zamítnutá

Obrázek 9: Fyzický datový model



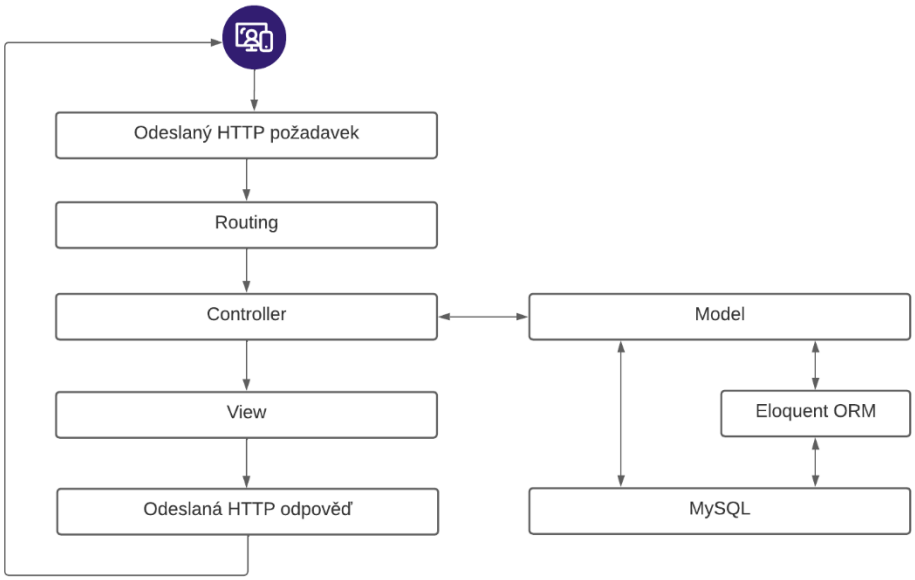
Zdroj: vlastní zpracování, 2022

3.6 Návrh softwarové architektury

Softwarová architektura webového informačního systému bude navržena dle architektonického vzoru Model-View-Controller. Návrh architektury respektuje i konkrétní technologie, které budou využity pro implementaci webového informačního systému. Softwarová architektura navrženého webového informačního systému je zobrazena na obrázku (Obrázek 10: Návrh softwarové architektury). Díky využití architektonického vzoru MVC pro návrh softwarové architektury bude webový informační systém pracovat podle následujícího postupu:

1. Uživatel informačního systému odešle HTTP požadavek z webového prohlížeče na webový server.
2. HTTP požadavek je směrován na odpovídající Controller.
3. Controller interaguje s příslušným Modelem o požadované výměně dat. Komunikace mezi Modelem a SŘBD MySQL probíhá přímo pomocí abstraktní databázové vrstvy PDO, případně přes vrstvu objektově relačního mapování Eloquent, která je součástí frameworku Laravel.
4. View pak následně vyrenderuje příslušnou šablonu.
5. Z webového serveru je nakonec odeslána HTTP odpověď se všemi potřebnými informacemi a vyžádanými daty.

Obrázek 10: Návrh softwarové architektury



Zdroj: vlastní zpracování, 2022

3.7 Návrh uživatelského rozhraní

V rámci návrhu uživatelského rozhraní jsou vytvořeny tzv. wireframes jednotlivých obrazovek webového informačního systému. Tyto náčrty uživatelského rozhraní slouží k zobrazení základních grafických prvků, které bude obsahovat prezentační vrstva výsledného informačního systému. Na základě navržených wireframes jsou následně vytvořeny konkrétní stránky grafického uživatelského rozhraní informačního systému.

Pro návrh uživatelského rozhraní bylo zvoleno klasické rozložení prvků s hlavičkou a menu umístěným v horní části stránky, s kontejnerem pro hlavní obsah stránky a s patičkou umístěnou v dolní části stránky. Odkazy na stránky, které jsou určené pro všechny uživatele, jsou dostupné přímo z menu umístěného v hlavičce, zatímco odkazy na stránky vztažené ke konkrétnímu přihlášenému uživateli je možné nalézt v rozevíracím seznamu se jménem přihlášeného uživatele, umístěném v pravé části hlavičky stránky. Návrh uživatelského rozhraní probíhal v souladu s respektováním možností front-end frameworku Bootstrap, který bude využit při implementaci webového informačního systému.

Vytvořené návrhy obrazovek webového informačního systému jsou součástí přílohy bakalářské práce (Příloha A: Návrh uživatelského rozhraní – wireframes).

3.8 Implementace webového informačního systému

Kapitola se věnuje popisu vybraných implementačních detailů vyvíjeného webového informačního systému.

3.8.1 Instalace frameworku a knihoven

Jelikož byl pro implementaci navrženého webového informačního systému zvolen framework Laravel, bylo nutné jako první krok celého vývoje provést jeho instalaci. Instalace frameworku byla provedena s využitím nástroje *Composer* a příkazu `composer create-project laravel/laravel studenti-spolu`. Nově vytvořený projekt pak bylo možné otevřít v příslušném vývojovém prostředí, konkrétně v PhpStorm IDE. S využitím nástroje *npm* byla provedena instalace několika vybraných knihoven potřebných pro implementaci informačního systému. Kromě knihoven Bootstrap 5 a PopperJS potřebných pro tvorbu uživatelského rozhraní, byly staženy

i knihovny sass a sass-loader určené pro práci s CSS preprocesorem SASS a jeho kompilaci do CSS s využitím nástroje *webpack*, který je rovněž součástí vybraného frameworku.

K instalaci nových balíků slouží příkazy `composer require <package>`, resp. `npm install <package>`. Názvy balíků určených ke stažení včetně potřebných verzí jsou uloženy v souborech `composer.json`, resp. `package.json`.

3.8.2 Adresářová struktura projektu

Využitý framework Laravel pro vývoj webového informačního systému poskytl předem definovanou adresářovou strukturu projektu. Pro lepší orientaci je tato podkapitola věnována krátkému představení jednotlivých složek:

- `/app` – adresář obsahující hlavní kód webového informačního systému, tj. všechny vytvářené třídy aplikace:
  - `/Console` – podadresář obsahuje všechny příkazy nástroje *Artisan*,
  - `/Exceptions` – podadresář obsahuje soubor pro obsluhu výjimek aplikace,
  - `/Http` – podadresář obsahující soubory reprezentující logiku aplikace:
    - `/Controllers` – podadresář obsahuje soubory vztažené ke komponentě *Controller,*
    - `/Middleware` – podadresář obsahuje soubory vrstvy pro kontrolu a filtrování HTTP požadavků vstupujících do aplikace,
  - `/Interfaces` – podadresář obsahuje rozhraní pro vybrané třídy,
  - `/Models` – podadresář obsahuje modely pro objektově-relační mapování,
  - `/Providers` – podadresář obsahuje tzv. *Service Providers*, které slouží pro správu závislostí,
  - `/Repositories` – podadresář obsahuje obalové třídy pro práci s objekty.
- `/bootstrap` – adresář obsahující kromě souboru zavádějícího celý framework i adresář `cache`, který obsahuje soubory generované pro optimalizaci výkonu.

- `/config` – adresář, ve kterém jsou uloženy všechny konfigurační soubory aplikace.
- `/database` – adresář určený pro migrační soubory databáze nebo soubory pro tvorbu složitých modelů.
- `/lang` – adresář obsahuje jazykové soubory aplikace, zpravidla se jedná o definici výpisů zpráv a hlášek v různých jazykových mutacích.
- `/node-modules` – adresář, ve kterém jsou uloženy všechny balíky a závislosti instalované nástrojem *npm*.
- `/public` – veřejný adresář obsahující soubor *index.php*, který je vstupním souborem pro všechny požadavky na informační systém. Dále jsou zde uloženy kompilované obrázky, CSS a JavaScript soubory.
- `/resources` – adresář určený pro ukládání *views* v podobě šablon Blade spolu s nezkompilovanými CSS a JavaScript soubory.
- `/routes` – adresář obsahuje definice všech `routes` webového informačního systému.
- `/storage` – adresář obsahující uložené soubory aplikace:
    - `/app` – podadresář určený pro ukládání všech souborů generovaných aplikací či souborů, které mají být v aplikaci dostupné,
    - `/framework` – podadresář sloužící k ukládání souborů generovaných rámcem a mezipaměti,
    - `/logs` – podadresář, který obsahuje logy aplikace.
- `/tests` – adresář určený pro automatizované testy webové aplikace.
- `/vendor` – adresář, ve kterém jsou uloženy všechny balíky a závislosti instalované nástrojem *Composer*.

3.8.3 Routing

Princip procesu zvaný routing (neboli směrování) spočívá v přiřazování konkrétních metod z příslušného Controlleru přijatým HTTP požadavkům. Pro implementaci webového informačního systému byla využita definice `routes` pro webové rozhraní.

K jednotlivým definovaným `routes` pak lze přistupovat zadáním příslušné URL adresy do webového prohlížeče. Součástí definice `routes` je volání příslušné statické metody třídy `Route` odpovídající konkrétnímu HTTP požadavku. Mezi parametry metody patří specifikovaná adresa, ze které budou přicházet požadavky, spolu s názvem volané metody konkrétního `Controlleru`. Jednotlivé `routes` lze pojmenovat pro lepší přehlednost a snadnější odkazování na danou `route`.

```
use App\Http\Controllers\ProjectsController;

Route::get('/projekty', [ProjectsController::class, 'index'])
    ->name('projekty.index');
```

3.8.4 Obsah vybraných souborů projektu

Z důvodu lepší orientace ve výsledném programu kapitola obsahuje představení obsahu vybraných souborů, které tvoří základ funkčnosti implementovaného webového informačního systému.

Třídy z komponenty Model:

- `FieldRepositoryInterface`, `FieldRepository` – rozhraní a obalová třída obsahující metody pro práci s obory znalostí a dovedností,
- `FileRepositoryInterface`, `FileRepository` – rozhraní a obalová třída obsahující metody pro práci se soubory,
- `OfferCooperationRepositoryInterface`, `OfferCooperationRepository` – rozhraní a obalová třída obsahující metody pro práci s nabídkami spolupráce,
- `ProjectRepositoryInterface`, `ProjectRepository` – rozhraní a obalová třída obsahující metody pro práci s projekty,
- `RequestCooperationRepositoryInterface`, `RequestCooperationRepository` – rozhraní a obalová třída obsahující metody pro práci se žádostmi o spolupráci,
- `RightRepositoryInterface`, `RightRepository` – rozhraní a obalová třída obsahující metody pro práci s uživatelskými právy,

- `StatusOfferRepositoryInterface`, `StatusOfferRepository` – rozhraní a obalová třída obsahující metody pro práci se stavy nabídek spolupráce,
- `StatusProjectRepositoryInterface`, `StatusProjectRepository` – rozhraní a obalová třída obsahující metody pro práci se stavy projektů,
- `StatusRequestRepositoryInterface`, `StatusRequestRepository` – rozhraní a obalová třída obsahující metody pro práci se stavy žádostí o spolupráci,
- `StatusUserRepositoryInterface`, `StatusUserRepository` – rozhraní a obalová třída obsahující metody pro práci se stavy uživatelů,
- `UserFieldRepositoryInterface`, `UserFieldRepository` – rozhraní a obalová třída obsahující metody pro práci s obory znalostí a dovedností uživatelů,
- `UserRepositoryInterface`, `UserRepository` – rozhraní a obalová třída obsahující metody pro práci s uživateli informačního systému.

Třídy z komponenty Controller:

- `AdminProjectsController` – zajišťuje získání příslušných dat o všech projektech spolu s příslušnými nabídkami spolupráce a slouží pro jejich správu v administrátorském rozhraní.
- `AdminUsersController` – zajišťuje získání příslušných dat o všech uživatelích a slouží pro jejich správu v administrátorském rozhraní.
- `AdminRequestsController` – zajišťuje získání příslušných dat o všech žádostech o spolupráci a slouží pro jejich správu v administrátorském rozhraní.
- `Controller` – defaultní předek všech vlastních tříd z komponenty Controller.
- `FilesController` – zajišťuje zobrazení, popř. stažení souboru nahraného ke konkrétnímu projektu.
- `LoginController` – slouží pro kontrolu zadaných přihlašovacích údajů a zajišťuje přihlášení uživatele do informačního systému.
- `LogoutController` – zajišťuje odhlášení uživatele z informačního systému.

- `MyProfileController` – zajišťuje získání příslušných dat o uživatelském profilu a jejich předání vybrané šabloně. Třída zároveň slouží pro zpracování formulářů umístěných na stránce *Můj profil* informačního systému.
- `MyProjectsController` – zajišťuje získání příslušných dat o autorských a spolupracovnických projektech a jejich předání vybraným šablonám. Třída zároveň slouží pro zpracování formulářů umístěných na stránce *Moje projekty* informačního systému.
- `OffersCooperationController` – zajišťuje získání příslušných dat o aktivních nabídkách spolupráce a jejich předání vybraným šablonám. Třída zároveň slouží pro zpracování formulářů umístěných na stránce *Nabídky spolupráce* informačního systému.
- `ProjectsController` – zajišťuje získání příslušných dat o zveřejněných projektech a jejich předání vybraným šablonám pro zobrazení stránky *Projekty* informačního systému.
- `RegistrationController` – slouží pro kontrolu zadaných údajů z registračního formuláře a zajišťuje registraci nového uživatele do informačního systému.
- `RequestsCooperationController` – zajišťuje získání příslušných dat o přijatých a odeslaných žádostech o spolupráci a jejich předání vybrané šabloně. Třída zároveň slouží pro zpracování formulářů umístěných na stránce *Žádosti o spolupráci* informačního systému.
- `UsersController` – zajišťuje získání příslušných dat o aktivních uživatelích systému a jejich předání vybrané šabloně pro zobrazení stránky *Uživatelé* informačního systému.

Šablony z komponenty View:

- `admin/index.blade.php` – slouží pro vykreslení úvodní stránky administračního rozhraní aplikace,
- `admin/projekty/index.blade.php` – slouží pro vykreslení stránky *Administrace projektů a nabídek spolupráce* zobrazující seznam všech projektů a možnosti upravit a mazat je,

- `admin/projekty/show.blade.php` – slouží pro vykreslení stránky pro úpravu vybraného projektu administrátorem aplikace,
- `admin/uzivatele/index.blade.php` – slouží pro vykreslení stránky *Administrace uživatelů* zobrazující seznam všech uživatelů a možnosti upravit a mazat je,
- `admin/uzivatele/show.blade.php` – slouží pro vykreslení stránky pro úpravu vybraného uživatele administrátorem aplikace,
- `admin/zadosti/index.blade.php` – slouží pro vykreslení stránky *Administrace žádostí o spolupráci* zobrazující seznam všech žádostí o spolupráci a možnosti upravit a mazat je,
- `admin/zadosti/show.blade.php` – slouží pro vykreslení stránky pro úpravu vybrané žádosti o spolupráci administrátorem aplikace,
- `layouts/layout.blade.php` – obsahuje základní rozložení uživatelského rozhraní WIS. Šablona slouží pro vykreslení hlavičky s navigačním menu a patičky stránky, které jsou společné pro všechny ostatní šablony.
- `index.blade.php` – slouží pro vykreslení domovské (úvodní) stránky WIS,
- `moje-projekty/index.blade.php` – slouží pro vykreslení stránky *Moje projekty* zobrazující seznam všech autorských nebo spolupracovnických projektů uživatele. Šablona také obsahuje modální okno pro vytvoření nového autorského projektu.
- `moje-projekty/show.blade.php` – slouží pro vykreslení stránky s detailním zobrazením vybraného autorského nebo spolupracovnického projektu,
- `muj-profil/index.blade.php` – slouží pro vykreslení stránky *Můj profil*, která je určena pro zobrazení a úpravu informací z uživatelského profilu,
- `nabidky-spoluprace/index.blade.php` – slouží pro vykreslení stránky *Nabídky spolupráce* zobrazující seznam všech aktivních nabídek spolupráce,
- `nabidky-spoluprace/show.blade.php` – slouží pro vykreslení stránky s detailním zobrazením vybrané aktivní nabídky spolupráce. Šablona také

obsahuje modální okno pro odeslání žádosti o spolupráci reagující na vybranou nabídku spolupráce.

- `prihlaseni/index.blade.php` – slouží k vykreslení stránky pro přihlašování uživatelů,
- `projekty/index.blade.php` – slouží pro vykreslení stránky *Projekty* zobrazující seznam všech zveřejněných projektů,
- `projekty/show.blade.php` – slouží pro vykreslení stránky s detailním zobrazením vybraného zveřejněného projektu,
- `registrace/index.blade.php` – slouží k vykreslení stránky pro registraci uživatelů,
- `uzivatele/index.blade.php` – slouží pro vykreslení stránky *Uživatelé* zobrazující seznam všech aktivních uživatelů WIS,
- `uzivatele/show.blade.php` – slouží pro vykreslení stránky s detailním zobrazením vybraného aktivního uživatele.

3.8.5 Vkládání závislostí

Vkládání závislostí tříd za pomoci nástroje `Service Container`, který je součástí distribuce frameworku Laravel, je využito na mnoha místech celého webového informačního systému. Za zmínku stojí například řešení vkládání závislostí na třídě zajišťující interakci s konkrétním HTTP požadavkem. Pro interakci s aktuálním HTTP požadavkem zpracovávaným webovým informačním systémem, je využívána třída `Illuminate\Http\Request`.

Téměř každá třída z komponenty `Controller` potřebuje pracovat s instancí třídy `Request`. Pro zajištění automatického vložení závislosti nástrojem `Service Container`, stačí jako parametr příslušné metody uvést instanci třídy `Request`.

```
use Illuminate\Http\Request;

class ProjectsController extends Controller
{
      public function index(Request $request)
      {
```

```
            ...
            $project_name = $request->input('project_name');
      }
}
```

Dalším místem, kde byla využita tato technika, tentokrát i s vlastní registrací vkládání závislosti na konkrétním rozhraní, bylo řešení vkládání závislostí na třídách komponenty `Model` v příslušných třídách komponenty `Controller`. Vložení závislosti proběhlo v tomto případě pomocí konstruktoru třídy komponenty Controller, ve kterém byla předaná instance vkládané třídy přiřazena připravenému datovému atributu třídy.

```
use App\Interfaces\OrderRepositoryInterface;
class ProjectsController extends Controller
{
      private IUsersRepository $usersRepository;
      public function __construct(IUsersRepository $usersRepository) {
      $this->usersRepository = $usersRepository;
      }
      ...
}
```

Manuální registrace vložení závislosti na konkrétním rozhraní je provedena vytvořením třídy `RepositoryServiceProvider` a napsáním těla metody `register()`, ve které je definováno, jakou třídu má nástroj `Service Container` za příslušné rozhraní vložit.

```
class RepositoryServiceProvider extends ServiceProvider
{
      public function register()
      {
            $this->app->bind(IUsersRepository::class,
                        UsersRepository::class);
            ...
      }
      ...
}
```

Nakonec byla přidána nová třída `RepositoryServiceProvider` do pole `$providers` v souboru `/config/app.php`:

```
'providers' => [
    ...
    App\Providers\RepositoryServiceProvider::class,
];
```

3.8.6 Autentizace a autorizace

Framework Laravel nabízí přístup k autentizačním službám, které dokáží při programování usnadnit práci. K těmto službám je přístup zajištěn prostřednictvím návrhového vzoru fasáda pojmenovaném `Auth`. Jelikož jsou autentizační služby založeny na práci s modely nástroje *Eloquent*, byl pro jednodušší práci s registrací, přihlašováním a odhlašováním uživatelů vytvořen příslušný model `User` svázaný s tabulkou `users`.

Aby bylo možné uživatele webového informačního systému autentizovat při přihlašování, je nejprve potřeba vyřešit jejich registraci za účelem vytvoření nového uživatelského profilu. Registrace nového uživatele se provádí vytvořením nové instance třídy `User` ve třídě `RegistrationController` v metodě `handle()` zajišťující zpracování registračního formuláře.

```
...
$user = User::create([
      'first_name' => $request->input('first_name'),
      'last_name' => $request->input('last_name'),
      'email' => $request->input('email'),
      'login' => $request->input('login'),
      'password' => Hash::make($request->input('password')),
]);
...
```

K přihlašování uživatelů založeném na kontrole údajů zadaných do přihlašovacího formuláře s údaji uloženými v databázi slouží třída `LoginController`. Samotná autentizace je v metodě `handle()` zajišťující zpracování přihlašovacího formuláře řešena voláním metody `attempt()` fasády `Auth`, které jsou jako parametr předány získané přihlašovací údaje.

```
...
$credentials = $request->validate([
      'login' => ['required'],
      'password' => ['required'],
]);
if (Auth::attempt($credentials)) {
      ...
      return redirect()->route('index');
}
return back()->withErrors([
      'login' => 'Chybně zadané přihlašovací údaje',
]);
...
```

Odhlášení je zajištěno voláním metody `auth()->logout()` v příslušné třídě `LogoutController`.

Framework Laravel nabízí jednoduchý způsob autorizace uživatelů pomocí fasády `Gate`, která určuje, zda je přihlášený uživatel oprávněn provést danou akci. `Gates` se vytváří ve třídě `/app/Providers/AuthServiceProvider` v těle metody `boot()`.

```
public function boot()
{
    $this->registerPolicies();
    Gate::define('isAdmin', function(User $user) {
        return $user->id_right == 2;
    });
    Gate::define('isSuperAdmin', function(User $user) {
        return $user->id_right == 1;
    });
}
```

Definované `Gates` lze využít pro autorizaci uživatele v definici konkrétních `routes` přiřazením `middleware('can:isAdmin')`, v třídách komponenty Controller pomocí metody `Gate:allows('isAdmin')` i v šablonách Blade s využitím příkazu `@can('isAdmin') @endcan`.

3.8.7 Middleware

Při implementaci bylo využito několik tříd vrstvy `Middleware`. V první řadě se jedná o třídu `Authenticate` pro přesměrování nepřihlášeného uživatele na stránku pro přihlášení při požadavku o zobrazení zabezpečené stránky. Dále pak byla využita třída `DeleteUrlCookie.php`, která slouží jako doplněk třídy `Authenticate.php`, pro odstranění uložené `Cookie` s hodnotou URL adresy pro zpětné přesměrování. Vytvořena byla i třída `PreventBackHistory.php`, jejímž úkolem je deaktivovat cache při vykreslování stránek, která by za určitých podmínek způsobovala nekonzistenci při využívání tlačítek zpět v prohlížeči.

Prvním krokem pro vytvoření vlastního `Middleware` je zavolání příkazu `php artisan make:middleware <název>`, který zajistí tvorbu nové třídy patřící právě do vrstvy `Middleware`. Druhý krok spočívá v napsání těla veřejné metody `handle`, která reprezentuje onu logiku filtrování přijatého požadavku. Aby bylo možné vytvořenou třídu vrstvy `Middleware` přiřadit konkrétním `routes`, které přijímají HTTP požadavky, je potřeba provést jeho registraci v souboru `/app/Http/Kernel.php`. Třetím krokem je přidání nového prvku do asociativního pole `$routeMiddleware` v registračním souboru. Prvek tohoto asociativního pole musí jako klíč obsahovat název, kterým se bude daný `Middleware` volat, a jako hodnotu danou vytvořenou třídu.

```
...
protected $routeMiddleware = [
   ...
   'delete-url-cookie' => \App\Http\Middleware\DeleteUrlCookie::class,
];
...
```

Následně je již možné přiřadit novou třídu `Middleware` ke konkrétním `routes`:

```
Route::get('/nabidky-spoluprace', [OffersCooperationController::class,
'index'])
    ->middleware('auth')
    ->name('nabidky-spoluprace.index');
```

3.8.8 Nahrávání souborů

Naprogramování funkcionality nahrávání souborů umožňuje studentům přidávat k jejich autorským nebo spolupracovnickým projektům soubory, které slouží pro prezentaci jejich dosavadní práce. Každému nahranému souboru odpovídá příslušný záznam v databázové tabulce FILES. Soubory jsou nahrávány na cloudové úložiště Amazon S3 prostřednictvím rozhraní webové služby. Jelikož framework Laravel nabízí mimo jiné podporu pro cloudové úložiště Amazon S3, je možné využívat pro práci se soubory fasádu `Storage`. Pro využívání služby S3 bylo nutné vytvořit si účet u Amazon Web Services. Po vytvoření nového kontejneru pro ukládání souborů zvaného Bucket a nastavení přístupových práv, bylo potřeba přiřadit hodnoty příslušným proměnným prostředí, které slouží ke konfiguraci připojení ke cloudovému úložišti. Fasáda `Storage` nabízí velké množství metod pro práci se soubory, z nichž byly využity především metody pro uložení, získání a odstranění souborů.

3.8.9 Nasazení webového informačního systému na server

Posledním krokem implementace webového informačního systému bylo jeho nasazení na veřejný server. Zvolena byla cloudová platforma Amazon Web Services. Pro nasazení vyvinutého webového informačního systému byla využita služba AWS Elastic Beanstalk. Nasazení webové aplikace do cloudu spočívalo v pronájmu Linux serveru s nainstalovaným interpretem jazyka PHP 8.0 a softwarovým webovým serverem NGINX. Platforma AWS nabízí i podrobné monitorování chodu webové aplikace i provozu na virtuálním serveru.

Implementovaný webový informační systém je veřejně dostupný na adrese:

http://studentispolu-env.eba-9qkcehhg.eu-west-2.elasticbeanstalk.com/.

3.9 Testování použitelnosti

Kapitola popisuje postup testování použitelnosti a definuje uživatelské scénáře včetně úkolů, které budou využity po testování použitelnosti. Kapitola je dále věnována vyhodnocení provedeného testu použitelnosti.

3.9.1 Postup pro testování použitelnosti

Pro testování použitelnosti navrženého a implementovaného informačního systému byla zvolena metoda uživatelského testování v moderované formě. Celý postup testování použitelnosti byl založen na postupném zadávání jednotlivých úkolů testerům a měření doby (v sekundách) potřebných pro jejich splnění. Pro testování použitelnosti byla dána přednost kvantitativnímu způsobu testování před testování kvalitativním. Testování založené na sběru metrik kvantitativního charakteru bude mít větší vypovídací schopnost o zdatnosti uživatelů využívat navržený informační systém k plnění úkolů vycházejících z procesů reálného užití. Pro testování použitelnosti bylo osloveno dvacet studentů vysoké školy odpovídajících cílové skupině uživatelů navrženého informačního systému.

3.9.2 Uživatelské scénáře

Scénář pro autora nového týmového projektu:

U1 – Registrujte se do webového informačního systému.

U2 – Vytvořte nový projekt, ke kterému potřebujete najít spolupracovníka.

U3 – K nově vytvořenému týmovému projektu nahrajte soubor, který bude sloužit k prezentaci vaší práce na projektu.

U4 – Vytvořte a zveřejněte novou nabídku spolupráce v některém z oborů na vytvořeném týmovém projektu.

Scénář pro studenta, který by rád spolupracoval na týmovém projektu:

U5 – Přihlaste se do informačního systému.

U6 – Vyhledejte konkrétní zveřejněnou nabídku spolupráce a pošlete žádost o spolupráci.

U7 – Upravte odeslanou žádost o spolupráci.

Scénář pro autora schvalujícího přijatou žádost o spolupráci:

U8 – Proveďte schválení nově přijaté žádosti o spolupráci.

U9 – Proveďte úpravu informací v nastavení uživatelského profilu.

U10 – Vyhledejte libovolného aktivního uživatele informačního systému a kontaktujte ho e-mailem.

3.9.3 Vyhodnocení testování použitelnosti

V tabulce (Tabulka 1: Naměřené hodnoty doby [s] pro zpracování úkolů při testování použitelnosti) jsou zaznamenány doby, které byly naměřeny během procesu testování použitelnosti. Sloupce tabulky odpovídají zadávaným úkolům a v řádcích jsou uvedeny doby plnění těchto úkolů jednotlivými testery v sekundách.

Tabulka 1: Naměřené hodnoty doby [s] pro zpracování úkolů při testování použitelnosti

Úkol / Tester	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10
T1	3	11	14	14	3,5	8,5	13	10	4,5	11,5
T2	3	5	11	16	3	11	7	7,5	4	8
T3	3	12	12	18	3,5	8	12	8	4,5	8
T4	3,5	12	16,5	23,5	3,5	16,5	11,5	11	5	11
T5	3	11	11	23	2	14	10	9	4	10
T6	3	14,5	9	18	3	9	9	7,5	3	7
T7	3,5	7	18	12	2	8,5	10	9	5	8
T8	3	6,5	11	11	2	9	10	10	5	11
T9	3,5	15,5	10	25	2	7	9,5	17	3,5	11
T10	2,5	10	14,5	22,5	2,5	12,5	10,5	14,5	3	10
T11	3	8	15	15,5	3,5	16	10	10	5	7
T12	2,5	7	13	19	3,5	14	11,5	13	4	11,5
T13	2,5	9	11	19	2	8	8	6,5	5	11
T14	3	13	10,5	12	2	8,5	10	15,5	4,5	6
T15	3	9	12	21	3	12	8	12	5	9
T16	3,5	8	13	20	3	14	9,5	12	5	7,5
T17	2,5	7	10	18	2	8	9	9	4	11
T18	3	7	15,5	17	2,5	11	10,5	13,5	3	10
T19	2,5	7	12	15,5	2,5	10	11	9	3,5	7,5
T20	3,5	10	14	16	3	12	11,5	12,5	5	9,5

Zdroj: vlastní zpracování, 2022

Pro znázornění rozložení hodnot a identifikaci případných odlehlých hodnot naměřených dob byl s využitím software TIBCO Statistica vygenerován krabicový graf. V tabulce (Tabulka 2: Vypočtené hodnoty pro vyhodnocení testu použitelnosti) jsou zároveň

dopočítány konkrétní hodnoty vyznačené ve vygenerovaném krabicovém grafu – medián, horní kvartily, dolní kvartily, horní vnitřní hradby a dolní vnitřní hradby.

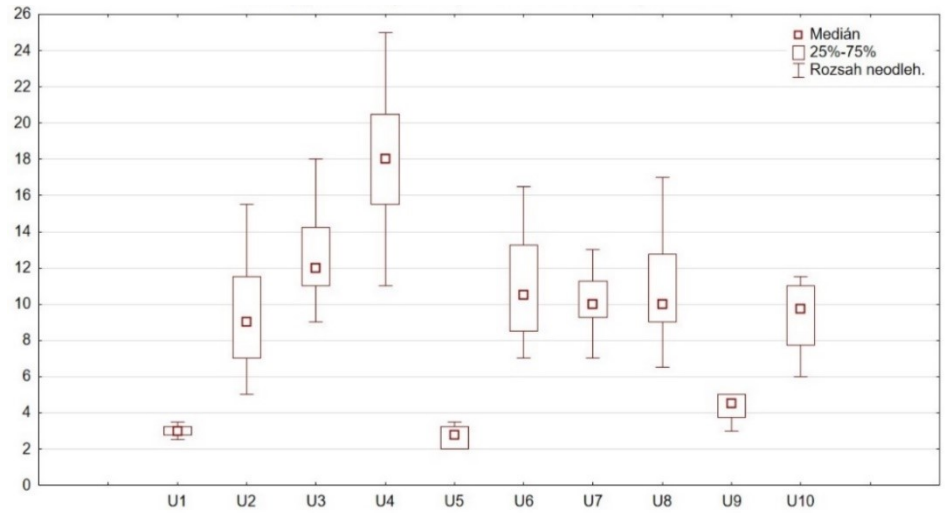
Tabulka 2: Vypočtené hodnoty pro vyhodnocení testu použitelnosti

	U1	**U2**	**U3**	**U4**	**U5**	**U6**	**U7**	**U8**	**U9**	**U10**
Medián $\tilde{x}$	3	9	12	18	2,75	10,5	10	10	4,5	9,75
Dolní kvartil x_{25}	2,75	7	11	15,5	2	8,5	9,25	9	3,75	7,75
Horní kvartil x_{75}	3,25	11,5	14,25	20,5	3,25	13,25	11,25	12,75	5	11
Mezikvartilové rozpětí $IQR = x_{75} - x_{25}$	0,5	4,5	3,25	5	1,25	4,75	2	3,75	1,25	3,25
Dolní vnitřní hradba $= x_{25} - 1{,}5 \cdot IQR$ / minimální hodnota statistického znaku	2,5	5	9	11	2	7	7	6,5	3	6
Horní vnitřní hradba $= x_{75} + 1{,}5 \cdot IQR$ / maximální hodnota statistického znaku	3,5	15,5	18	25	3,5	16,5	13	17	5	11,5

Zdroj: vlastní zpracování, 2022

Z výsledného grafu (Obrázek 11: Krabicový graf naměřených dob při testování použitelnosti) lze vyčíst, že žádnou z naměřených hodnot nelze považovat za odlehlou, protože se žádná z nich nenachází za horní vnitřní hradbou, resp. dolní vnitřní hradbou. V rámci celého procesu testování nenastala situace, při které by některý z testerů vůbec nepochopil zadání úkolu a nezvládl zadaný úkol dokončit. Implementovaný webový informační systém fungoval spolehlivě a nevyskytl se žádný problém s jeho funkčností.

Obrázek 11: Krabicový graf naměřených dob při testování použitelnosti



Zdroj: vlastní zpracování (s použitím SW Statistica), 2022

Po provedené analýze odlehlých hodnot následuje test normality pro doby potřebných pro splnění jednotlivých úkolů, který bude sloužit jako rozhodovací nástroj pro volbu vhodného statistického testu o shodě středních hodnot, resp. mediánů. Pro testování normality byl zvolen Shapiro-Wilkův test, který je vhodný i pro malé výběry $3 \leq n \leq 5$ (Svoboda, Gangur, & Mičudová, 2019). V tabulce (Tabulka 3: Výsledky Shapiro-Wilkova testu normality) jsou pro jednotlivé úkoly stanoveny nulové a alternativní hypotézy a spočteny hodnoty testového kritéria W spolu s příslušnou $p - hodnotou$. Pro hladinu významnosti $\alpha = 0{,}05$ platí, že výběrové soubory naměřených dob pro úkoly U1, U5, U6 a U9 nemají normální rozdělení.

Tabulka 3: Výsledky Shapiro-Wilkova testu normality

Úkol	$\bar{x}$	s_x^2	H_0	H_1	W	p-hodnota
U1	3	0,13	$X \sim N(3; 0{,}13)$	$X \sim nonN(3; 0{,}13)$	0,81534	0,00148
U2	9,48	8,22	$X \sim N(9{,}48; 8{,}22)$	$X \sim nonN(9{,}48; 8{,}22)$	0,94182	0,25949
U3	12,7	5,71	$X \sim N(12{,}7; 5{,}71)$	$X \sim nonN(12{,}7; 5{,}71)$	0,95384	0,42924
U4	17,8	15,6	$X \sim N(17{,}8; 15{,}6)$	$X \sim nonN(17{,}8; 15{,}6)$	0,97233	0,80309
U5	2,7	0,38	$X \sim N(2{,}7; 0{,}38)$	$X \sim nonN(2{,}7; 0{,}38)$	0,82719	0,00226
U6	10,9	8,26	$X \sim N(10{,}9; 8{,}26)$	$X \sim nonN(10{,}9; 8{,}26)$	0,91516	0,07999
U7	10,1	2,14	$X \sim N(10{,}1; 2{,}14)$	$X \sim nonN(10{,}1; 2{,}14)$	0,97726	0,89399
U8	10,8	8,17	$X \sim N(10{,}8; 8{,}17)$	$X \sim nonN(10{,}8; 8{,}17)$	0,95663	0,47886
U9	4,28	0,57	$X \sim N(4{,}28; 0{,}57)$	$X \sim nonN(4{,}28; 0{,}57)$	0,83509	0,00303
U10	9,28	3,09	$X \sim N(9{,}28; 3{,}09)$	$X \sim nonN(9{,}28; 3{,}09)$	0,90487	0,05094

Zdroj: vlastní zpracování (s použitím SW Statistica), 2022

Vzhledem k výsledku testu normality, který prokázal, že neexistuje normalita dat ve všech výběrových souborech, byl zvolen neparametrický Kruskal-Wallisův test o shodě mediánů. Kruskal-Wallisův test je založen na seřazení hodnot X_{ij} do rostoucí posloupnosti, určení jejich pořadí R_{ij} a určení součtů pořadí pro jednotlivé výběry T_i (Svoboda, Gangur, & Mičudová, 2019).

Stanovení hypotéz pro Kruskal-Wallisův test:

$$H_0: x_{0,5(1)} = x_{0,5(2)} = x_{0,5(3)} = x_{0,5(4)} = x_{0,5(5)} = x_{0,5(6)} =$$
$$= x_{0,5(7)} = x_{0,5(8)} = x_{0,5(9)} = x_{0,5(10)}$$

$$H_1: \neg H_0$$

Následující obrázek (Obrázek 12: Výsledek Kruskal-Wallisova testu) obsahuje výsledek Kruskal-Wallisova testu provedeného s využitím software TIBCO Statistica. Pro Kruskal-Wallisův test vychází testová statistika $H = 155{,}7803$. Této hodnotě testové statistiky odpovídá $p - hodnota$ nižší než stanovená hladina významnosti $\alpha = 0{,}05$. Vzhledem k zamítnutí nulové hypotézy platí, že mediány naměřených dob pro jednotlivé úkoly se nerovnají.

Obrázek 12: Výsledek Kruskal-Wallisova testu

	Kruskal-Wallisova ANOVA založ. na poř.; Doba Nezávislá (grupovací) proměnná : Úkol Kruskal-Wallisův test: H (9, N= 200) =155,7803 p < 0,05			
Závislá: Doba	Kód	Počet platných	Součet pořadí	Průměr Pořadí
U1	101	20	500,000	25,0000
U2	102	20	2143,500	107,1750
U3	103	20	3012,000	150,6000
U4	104	20	3663,000	183,1500
U5	105	20	382,500	19,1250
U6	106	20	2506,000	125,3000
U7	107	20	2341,500	117,0750
U8	108	20	2498,500	124,9250
U9	109	20	951,500	47,5750
U10	110	20	2101,500	105,0750

Zdroj: vlastní zpracování (s použitím SW Statistica), 2022

Po zamítnutí nulové hypotézy, je potřeba pomocí post hoc analýzy určit, pro které dvojice úkolů se mediány naměřených dob statisticky významně liší. Post hoc analýza byla provedena pomocí Dunnové metody vhodné pro Kruskal-Wallisův test, která je založena na principu vícenásobného porovnání jednotlivých úkolů mezi sebou na základě jejich průměrného pořadí. V případě, že absolutní hodnota rozdílů průměrného pořadí je větší než tabelovaná kritická hodnota, mediány se od sebe statisticky významně liší (Svoboda, Gangur, & Mičudová, 2019).

Stanovení hypotéz pro post hoc analýzu

$$H_0: x_{0,5(i)} = x_{0,5(j)}$$

$$H_1: x_{0,5(i)} \neq x_{0,5(j)}$$

Následující obrázek (Obrázek 13: Výsledek post hoc analýzy) obsahuje výsledné $p - hodnoty$ pro jednotlivé dvojice porovnávaných úkolů. V případě, že $p - hodnota$ je

nižší než stanovená hladina významnosti $\alpha = 0{,}05$, mediány naměřených dob příslušné dvojice úkolů se statisticky významně liší.

Obrázek 13: Výsledek post hoc analýzy

Závislá: Doba	Vícenásobné porovnání p hodnot (oboustr.); Doba Nezávislá (grupovací) proměnná : Úkol Kruskal-Wallisův test: H (9, N= 200) =155,7803 p =0,000 U1 R:25,000	U2 R:107,17	U3 R:150,60	U4 R:183,15	U5 R:19,125	U6 R:125,30	U7 R:117,08	U8 R:124,92	U9 R:47,575	U10 R:105,08
U1		0,000321	0,000000	0,000000	1,000000	0,000002	0,000022	0,000002	1,000000	0,000547
U2	0,000321		0,794936	0,001490	0,000068	1,000000	1,000000	1,000000	0,050794	1,000000
U3	0,000000	0,794936		1,000000	0,000000	1,000000	1,000000	1,000000	0,000001	0,579231
U4	0,000000	0,001490	1,000000		0,000000	0,070830	0,013776	0,066006	0,000000	0,000897
U5	1,000000	0,000068	0,000000	0,000000		0,000000	0,000004	0,000000	1,000000	0,000119
U6	0,000002	1,000000	1,000000	0,070830	0,000000		1,000000	1,000000	0,000977	1,000000
U7	0,000022	1,000000	1,000000	0,013776	0,000004	1,000000		1,000000	0,006586	1,000000
U8	0,000002	1,000000	1,000000	0,066006	0,000000	1,000000	1,000000		0,001070	1,000000
U9	1,000000	0,050794	0,000001	0,000000	1,000000	0,000977	0,006586	0,001070		0,075623
U10	0,000547	1,000000	0,579231	0,000897	0,000119	1,000000	1,000000	1,000000	0,075623	

Zdroj: vlastní zpracování, 2022

Na základě provedené post hoc analýzy budou v následujícím kroku identifikovány homogenní skupiny úkolů. Postup identifikace homogenních skupin je založen na sestupném seřazení jednotlivých úkolů podle mediánu, označení úkolu s nejvyšší hodnotou mediánu písmenem *„a“* a následném porovnání s úkoly, jejichž medián je nižší. Pokud mezi porovnávanou dvojicí neexistuje statisticky významná odchylka, druhý z dvojice úkolů je označen stejným písmenem. Po porovnání všech dvojic, byl písmenem „b“ označen úkol s druhým největším mediánem a byl opakován obdobný postup s porovnáváním dvojic úkolů. Dále je potřeba vyloučit písmena, která jsou všechna obsažena ve skupinách úkolů již označených jiným písmenem. Na konci tohoto procesu každé písmeno identifikuje jednu homogenní skupinu. Každý z úkolů musí být pouze v jedné homogenní skupině.

Obrázek 14: Výsledek analýzy homogenních skupin

Úkol	Medián	Homogenní skupiny			
U4	18	**a**			
U3	12	**a**	b		
U6	10,5	**a**	b		
U7	10		**b**		
U8	10	**a**	b		
U10	9,75		**b**	c	
U2	9		**b**	c	
U9	4,5			**c**	d
U1	3				**d**
U5	2,75				**d**

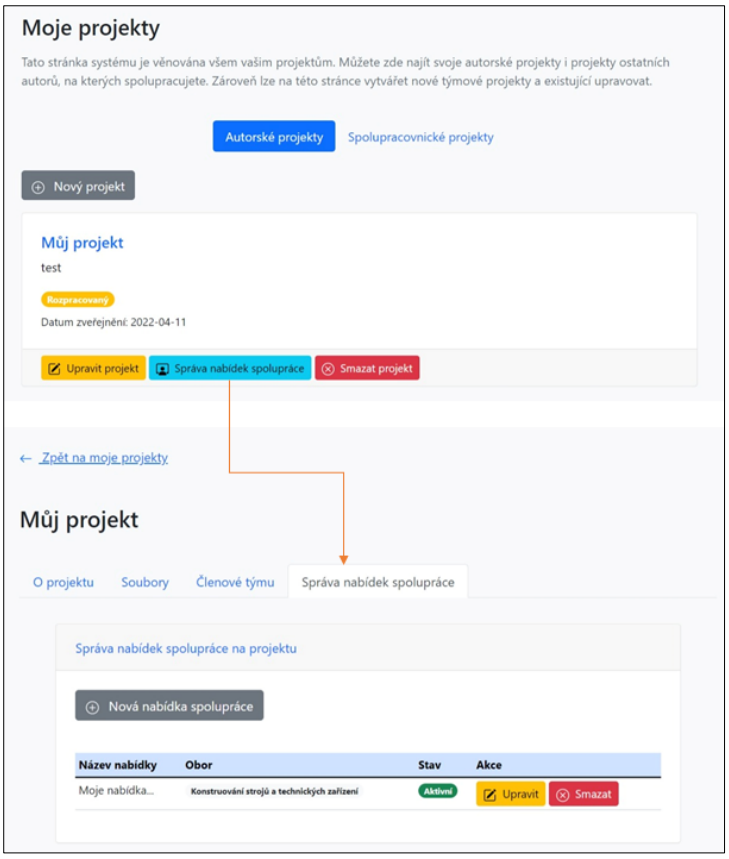
Zdroj: vlastní zpracování, 2022

Homogenní skupinu s nejvyššími naměřenými hodnotami doby tvoří úkoly U4, U3, U6 a U8. Tyto úkoly byly oproti ostatním obsáhlejší svým rozsahem a k jejich vyřešení byla z jejich podstaty potřeba delší doba. Při uživatelském testování však bylo možné u testerů pozorovat při plnění úkolu U4 jisté zaváhání, zda se *Správa nabídek spolupráce* nachází na stránce *Upravit projekt*. Návrhu na vyřešení tohoto identifikovaného problému je věnována následující podkapitola

3.9.4 Návrhy na zlepšení na základě testu použitelnosti

Současný webový informační systém je navržen tak, že nabídky spolupráce na určitém projektu lze vytvářet, upravovat a mazat v sekci *Správa nabídek spolupráce* na stránce *Moje projekty – Upravit projekt.* Tato vlastnost však při plnění úkolu U4 vedla ke znejistění testerů, zda se na stránce pro úpravu projektu nachází možnost správy nabídek spolupráce. Pro odstranění tohoto problému by bylo vhodné přidat k jednotlivým projektům na stránce *Moje projekty* nové tlačítko s názvem *Správa nabídek spolupráce* odkazující přímo do příslušné sekce stránky. Na obrázku je proveden návrh řešení.

Obrázek 15: Návrh na zlepšení na základě testu použitelnosti



Zdroj: vlastní zpracování, 2022

3.10 Návrhy na rozšíření funkcionality

Webový informační systém byl již od počátku navržen a následně implementován tak, aby nabízel plnou nabídku funkcionalit potřebných pro reálné využití a zajišťoval tak maximální podporu procesu zveřejňování informací o autorských projektech studentů vysokých škol, hledání aktivních nabídek spolupráce a zasílání reakcí v podobě žádostí od ostatních uživatelů. I přes to lze seznam nabízených funkcionalit dále rozšiřovat.

Příklady funkcionalit, které by bylo možné do implementace informačního systému doplnit:

- Přiřazování tagů při vytváření nového autorského projektu, které by usnadnily a zpřehlednily následné vyhledávání a filtrování v seznamu zveřejněných projektů na stránce *Projekty*.
- Zobrazení doporučení autorovi projektu ze strany informačního systému o schválení, resp. zamítnutí přijaté žádosti o spolupráci. Kritériem pro doporučení by mohla být shoda oborů znalostí a dovedností uživatele a oboru, ke kterému byla zveřejněna nabídka spolupráce, na kterou uživatel žádostí o spolupráci reagoval.
- Vytvoření notifikačního centra, kde by informační systém uživateli zobrazoval zprávy o novinkách, které se během jeho neaktivity staly. Mezi tyto notifikace by patřily například zprávy o počtu nově zveřejněných projektů, o počtu nových nabídek spolupráce odpovídající znalostem a dovednostem uživatele či o počtu nově přijatých žádostí o spolupráci na autorských projektech uživatele.
- Možnost přidat k vytvořenému autorskému projektu odkaz na web pro zveřejnění a ukázku dalších informací jako např. video s ukázkou vytvářeného produktu, odkaz na kód programu uloženého na GitHub apod.
- V nastavení uživatelského profilu dát uživateli možnost vložit odkaz na sociální sítě (např. LinkedIn) nebo na osobní webovou stránku s podrobnějšími informacemi o uživateli informačního systému.

Závěr

Hlavním cílem této bakalářské práce bylo vytvořit webový informační systém pro zveřejňování informací o týmových projektech a hledání možností spolupráce na těchto projektech. Nejprve byla provedena rešerše zdrojů souvisejících s vybranými tématy pro návrh a implementaci webového informačního systému. Dále byla popsána současná situace a dostupné nástroje týkající se možností hledání spolupráce na týmových projektech. V části metodiky řešení byl stanoven postup řešení návrhu a implementace webového informačního systému. Významnou součástí řešení byla specifikace požadavků, popis případů užití, analýza procesní orientace systému, návrh datové a softwarové architektury a samotná implementace navrženého webového informačního systému. Byl proveden kompletní vývoj s respektováním všech specifikovaných požadavků. Na implementovaném webovém informačním systému bylo provedeno uživatelské testování použitelnosti založené na postupném zadávání jednotlivých úkolů testerům a měření doby (v sekundách) potřebné pro jejich splnění. S využitím krabicového grafu bylo zjištěno, že žádnou z naměřených hodnot dob nelze považovat za odlehlou. Po provedení Kruskal-Wallisova testu o shodě mediánů a následné post hoc analýzy byla identifikována homogenní skupina úkolů s nejvyššími naměřenými hodnotami doby. Na základě výsledků testu použitelnosti byl za účelem snazšího nalezení sekce pro správu nabídek spolupráce proveden návrh na přidání příslušného tlačítka k autorským projektům na stránce Moje projekty. Při celém procesu testování fungoval webový informační systém spolehlivě bez výskytu problémů s jeho funkčností.

V poslední fázi práce na webovém informačním systému bylo provedeno nasazení navrženého a implementovaného informačního systému do ostrého provozu s využitím cloudové platformy Amazon Web Services. Pro uživatele je připravena detailní uživatelská dokumentace popisující práci s webovým informačním systémem ***studenti | spolu***.

Seznam použitých zdrojů

Boyett, R. (2021). *What is MySQL: MySQL Explained For Beginners*. Dostupné 13. 1. 2021 z https://www.hostinger.com/tutorials/what-is-mysql

Bruckner, T., Voříšek, J., & Buchalcevová, A. (2012). *Tvorba informačních systémů: Principy, metodiky, architektury.* Praha, Česko: Grada Publishing.

Budinská, I. (2009). *Klasifikace a porovnání metod testování (Diplomová práce).* Univerzita Pardubice, Fakulta ekonomicko-správní, Česká republika.

Budiu, R. (2017). *Quantitative vs. Qualitative Usability Testing*. Dostupné 18. 2. 2022 z https://www.nngroup.com/articles/quant-vs-qual/

Cejpek, J. (2005). *Informace, komunikace a myšlení : úvod do informační vědy.* Praha, Česko: Karolinum.

Coronel, C., Morris, S., & Rob, P. (2011). *Database Systems: Design, Implementation, and Management.* Course Technology.

Danel, R. (2021). *Architektura informačních systémů.* Dostupné 28. 12. 2021 z http://homel.vsb.cz/~dan11/is_skripta/IS%202011%20-%20Architektura%20IS.pdf

DB-Engines. (2022). *DB-Engines Ranking*. Dostupné 4. 2. 2022 z https://db-engines.com/en/ranking

Figueroa, A. (2019). *Monolithic versus microservices, and all in between*. Dostupné 14. 12. 2022 z https://medium.com/swlh/monolithic-vs-micro-services-and-all-in-between-7d496408ad02

Gangur, M. (2014). *Základy implementace webového informačního systému: Praktický průvodce.* Plzeň, Česko: Západočeská univerzita.

GeeksforGeeks. (2021). *Benefit of using MVC*. Dostupné 14. 12. 2021 z https://www.geeksforgeeks.org/benefit-of-using-mvc/

Chisnell, D. (2009). *Usability Testing Demystified*. Dostupné 16. 2. 2022 z https://alistapart.com/article/usability-testing-demystified/

Chlapek, D., Řepa, V., & Stanovská, I. (2011). *Analýza a návrh informačních systémů.* Praha, Česko: Oeconomica.

Laaziri, M., Benmoussa, K., Khoulji, S., & Larbi Kerkeb, M. (2019). A Comparative study of PHP frameworks performance. *Procedia Manufacturing*, stránky 864-871.

Nielsen, J. (2000). *Why You Only Need to Test with 5 Users*. Dostupné 10. 1. 2022 z https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/

Nielsen, J. (2001). *Usability Metrics*. Dostupné 10. 1. 2022 z https://www.nngroup.com/articles/usability-metrics/

Patel, J. (2021). *10 Reasons to Use Laravel PHP Framework For Web Development*. Dostupné 28. 12. 2021 z https://www.monocubed.com/why-laravel-php-framework/

Pecinovský, R. (2007). *Návrhové vzory : [33 vzorových postupů pro objektové programování].* Brno: Computer Press.

PHP Group. (2021). *PHP: Hypertext Preprocessor*. Dostupné 27. 12. 2021 z https://www.php.net/

Pilka, L. (2019). *Velký průvodce uživatelským testováním webů a aplikací*. Dostupné 10. 1. 2022 z https://www.blueghost.cz/clanek/velky-pruvodce-uzivatelskym-testovanim-webu-a-aplikaci/

Pokorný, J., & Valenta, M. (2020). *Databázové systémy.* Praha, Česko: Česká technika - nakladatelství ČVUT.

Rybička, J. (2021). *Informační systémy.* Dostupné 28. 12. 2021 z https://akela.mendelu.cz/~rybicka/prez/infsyst.pdf

Salas, A. (1995). Computing Architectures: Choosing the right way for you. Sborník Systémová integrace ’95. Praha.

SEVOCAB. (2017). *Software and Systems Engineering Vocabulary*. Dostupné 5. 12. 2021 z https://pascal.computer.org/sev_display/index.action

Sklar, D. (2018). *PHP 7: praktický průvodce nejrozšířenějším skriptovacím jazykem pro web.* Brno: Zoner Press.

Spolupráce ZČU. (2021). Dostupné 12. 11. 2021 z https://spoluprace.zcu.cz/

van der Aalst, W., & Stahl, C. (2011). *Modeling Business Processes: A Petri Net-Oriented Approach.* Cambridge, MA: The MIT Press.

Voják, M. (2020). *Jak dělat uživatelské testování*. Dostupné 10. 1. 2022 z https://designdev.cz/jak-delat-uzivatelske-testovani

Voják, M. (2020). *Příprava testu použitelnosti*. Dostupné 10. 1. 2022 z https://designdev.cz/priprava-testu-pouzitelnosti

Voják, M. (2020). *Vyhodnocení testu použitelnosti*. Dostupné 10. 1. 2022 z https://designdev.cz/vyhodnoceni-testu-pouzitelnosti

W3Schools. (2021). *Bootstrap 5 Tutorial*. Dostupné 28. 12. 2021 z https://www.w3schools.com/bootstrap5/index.php

W3Techs. (2022). *Usage statistics of server-side programming languages for websites*. Dostupné 4. 2. 2022 z https://w3techs.com/technologies/overview/programming_language

Wiegers, K. E. (2008). *Požadavky na software.* Brno: Computer Press.

Seznam tabulek

Seznam obrázků

Seznam použitých zkratek

AWS	Amazon Web Services
CSS	Cascading Style Sheets
DDL	Data Definition Language
ERA	Entity-Relationship-Attribute
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IDE	Integrated Development Environment
IS	Informační systém
JIT	Just in time
MVC	Model-View-Controller
npm	Node Package Manager
PDO	PHP Data Objects
PHP	Hypertext Preprocessor
SEVOCAB	Software and Systems Engineering Vocabulary
SQL	Structured Query Language
SŘBD	Systém řízení báze dat
S3	Simple Storage Service
UML	Unified Modeling Language
WIS	Webový informační systém
XSS	Cross-site scripting
ZČU	Západočeská univerzita v Plzni

Seznam příloh

Příloha A: Návrh uživatelského rozhraní – wireframes

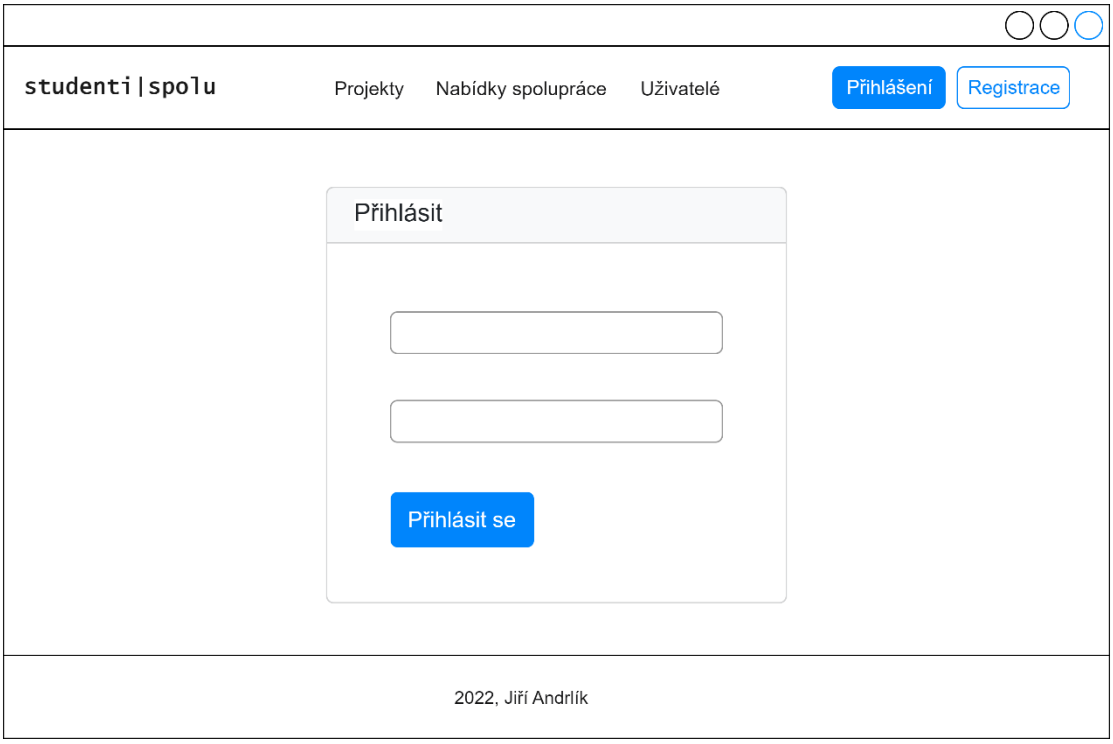
Příloha B: Uživatelská dokumentace

Příloha C: CD-R obsahující zdrojový kód implementovaného webového informačního systému

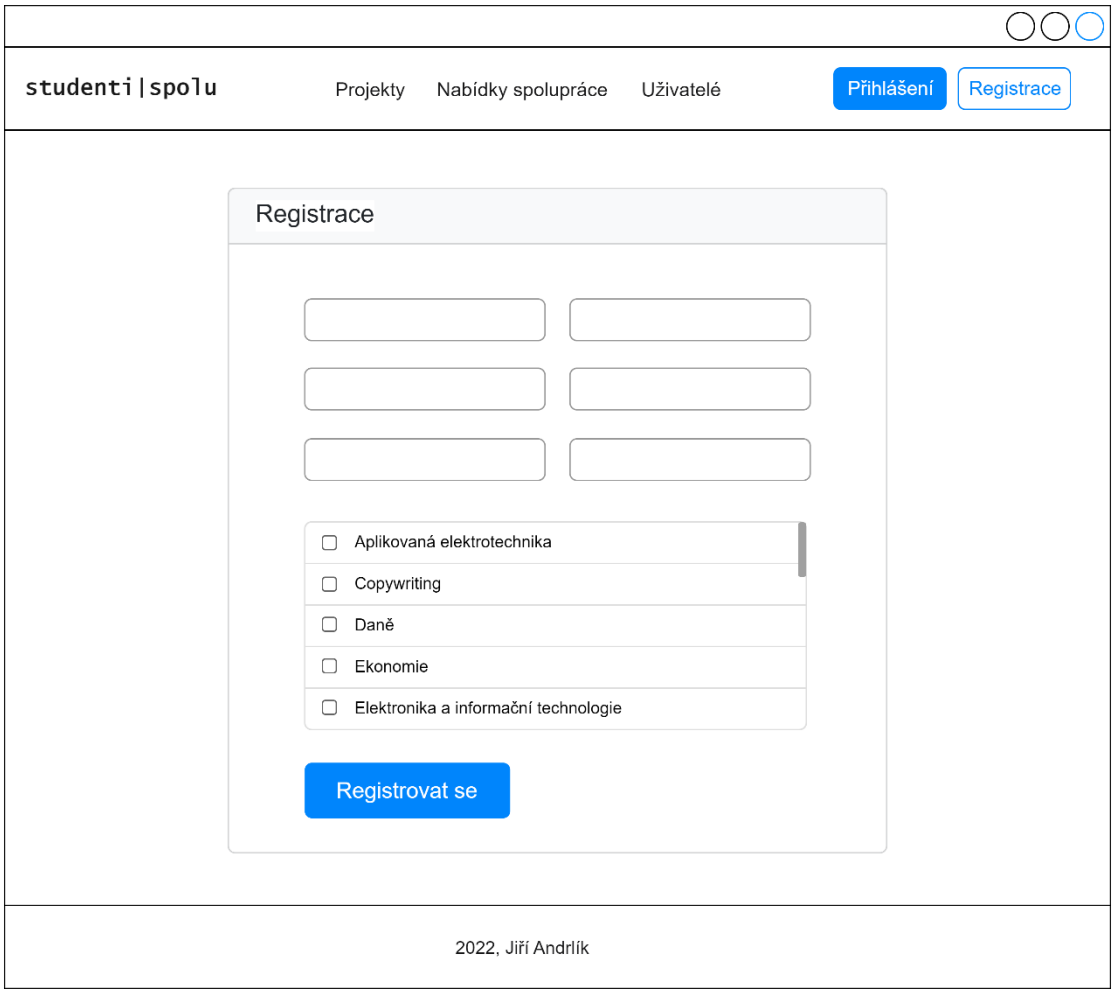
Příloha A: Návrh uživatelského rozhraní – wireframes

Domovská stránka:

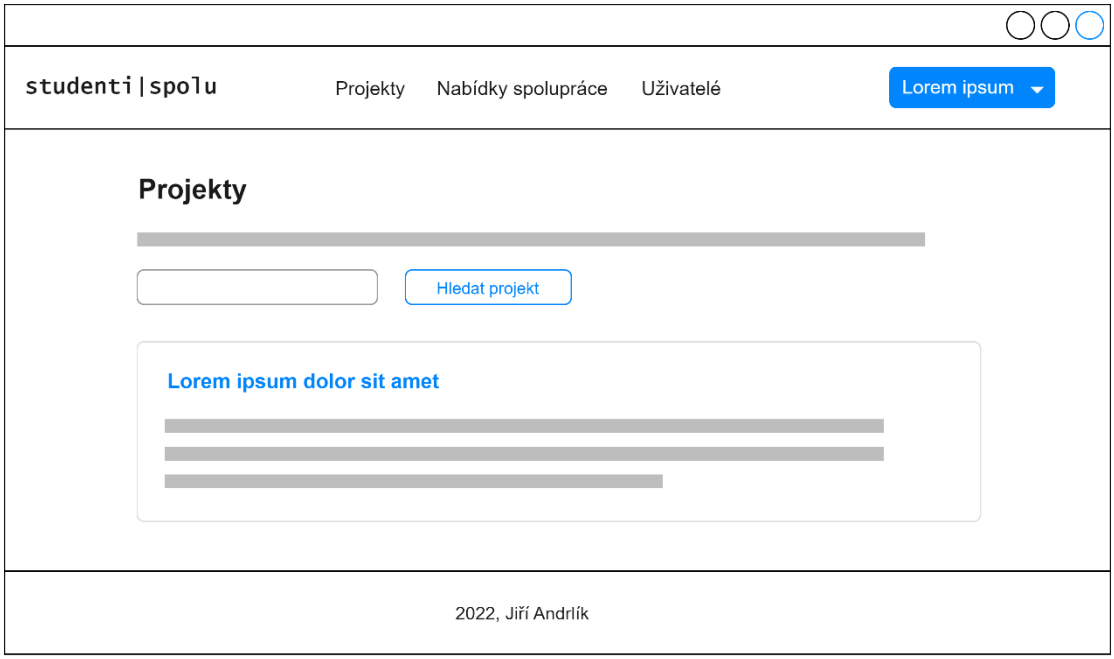
Přihlášení:



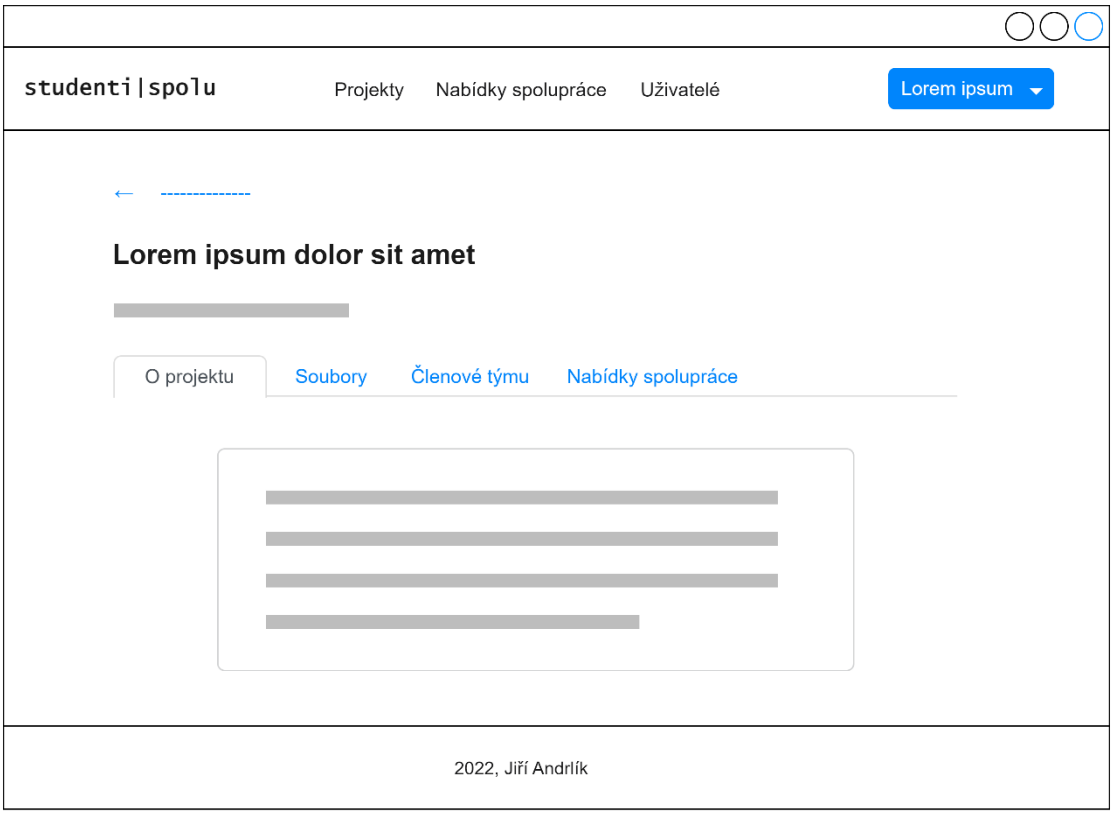
Registrace:



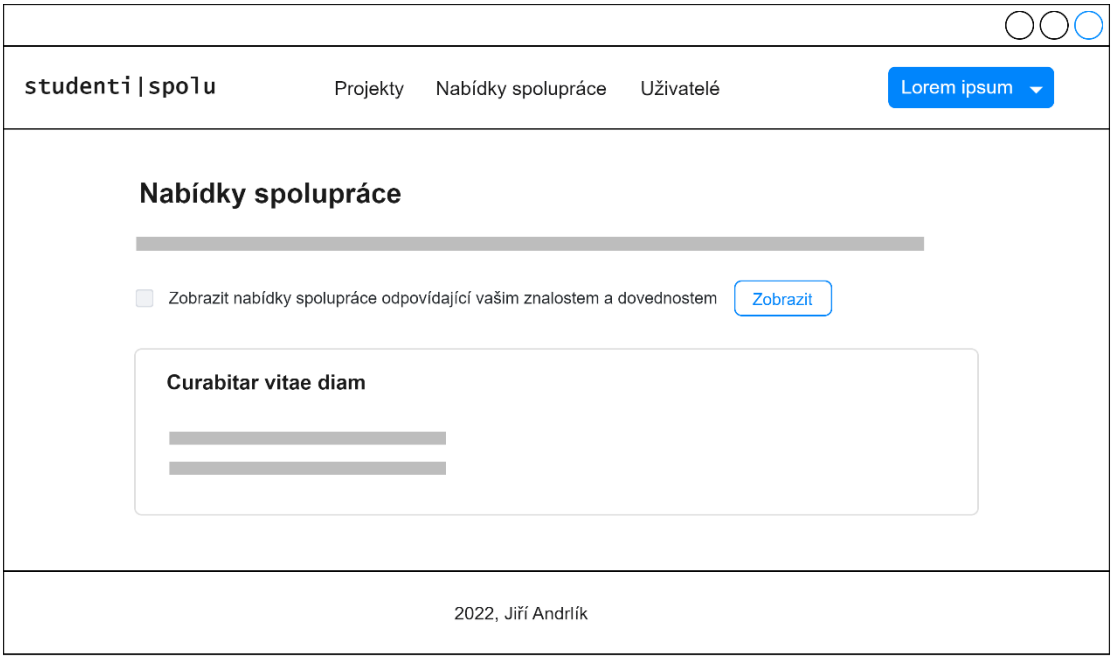
Projekty:



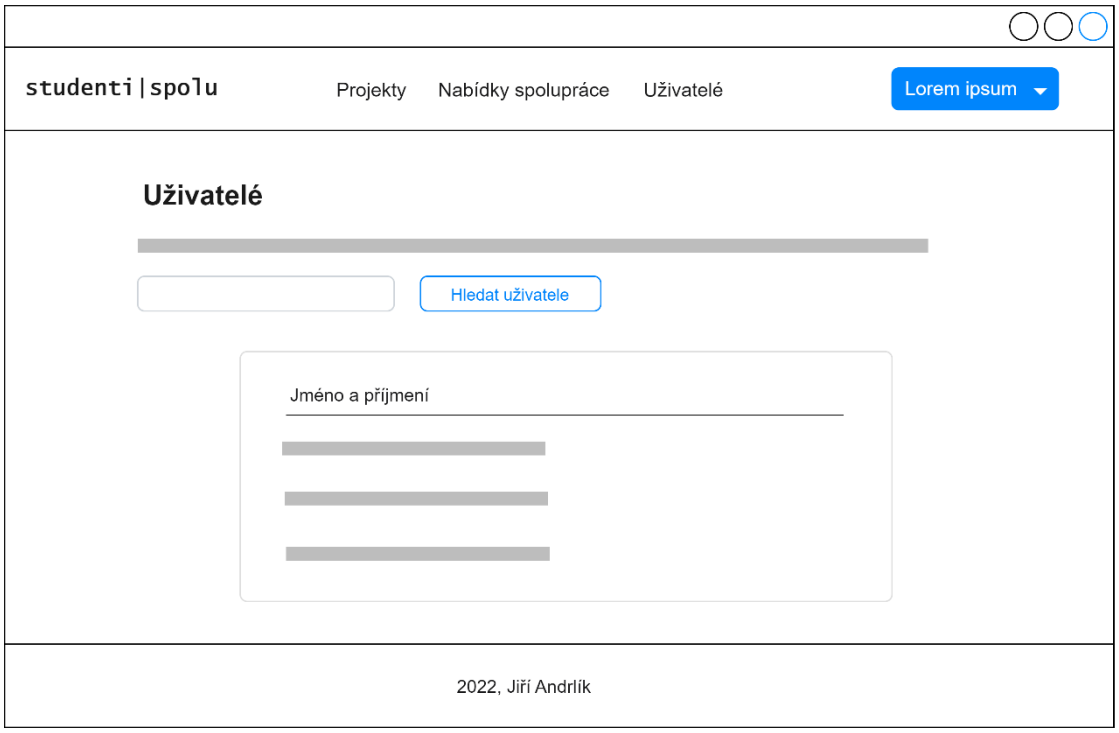
Detail projektu:



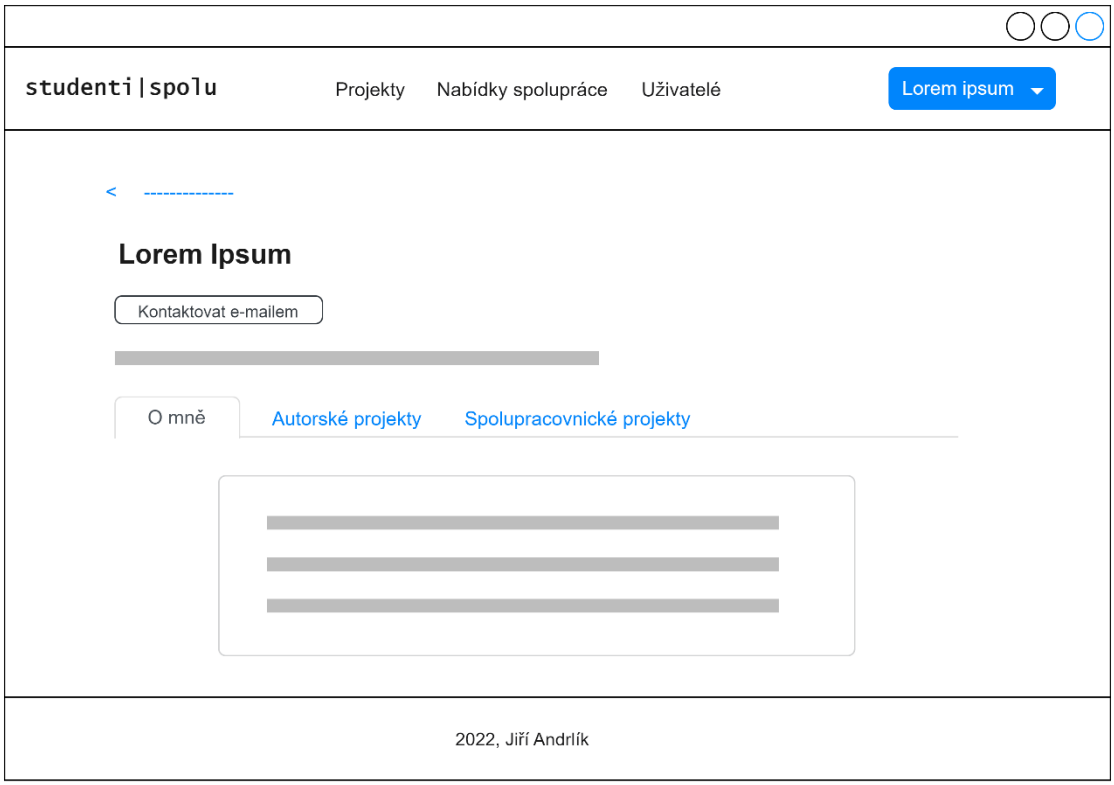
Nabídky spolupráce:



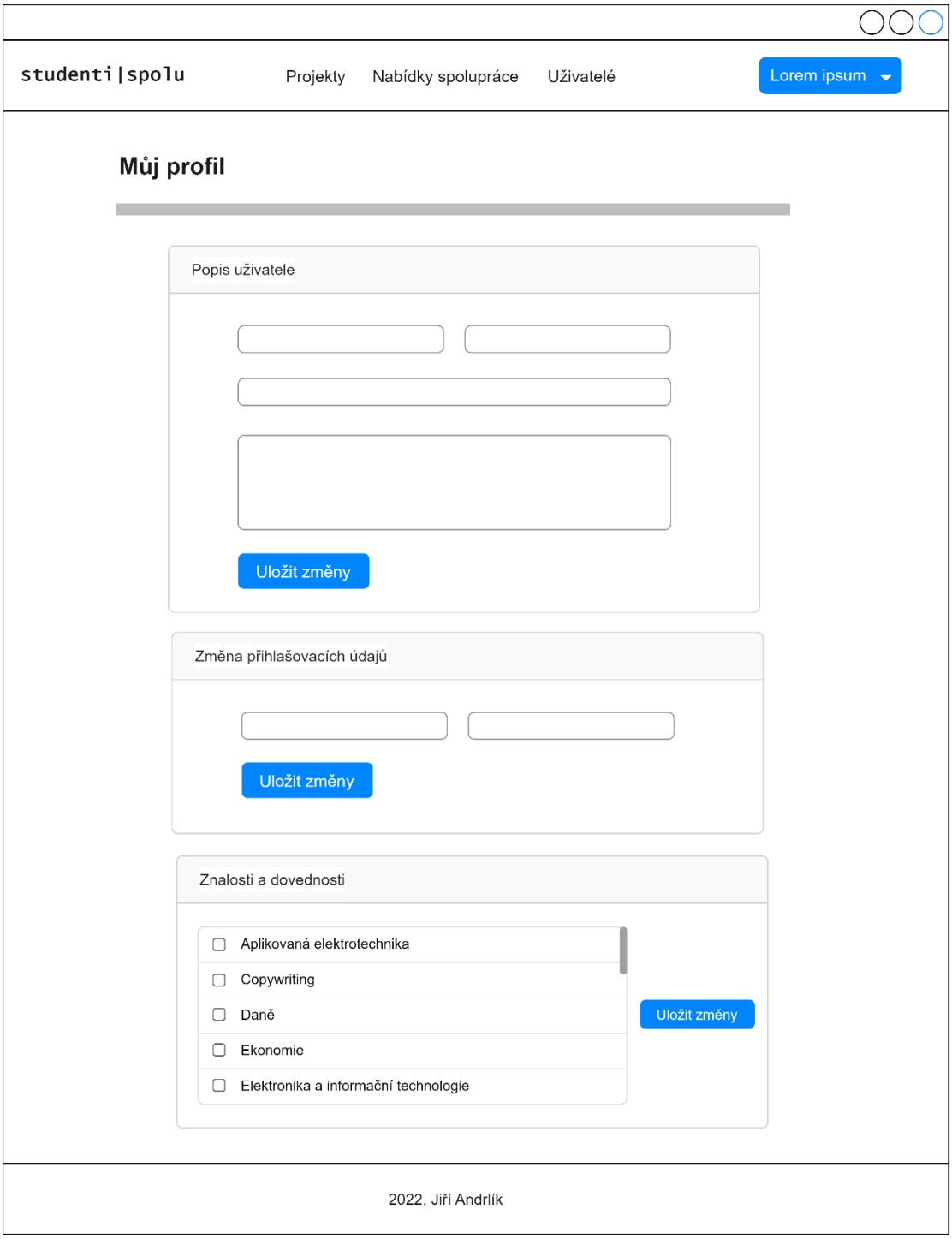
Uživatelé:



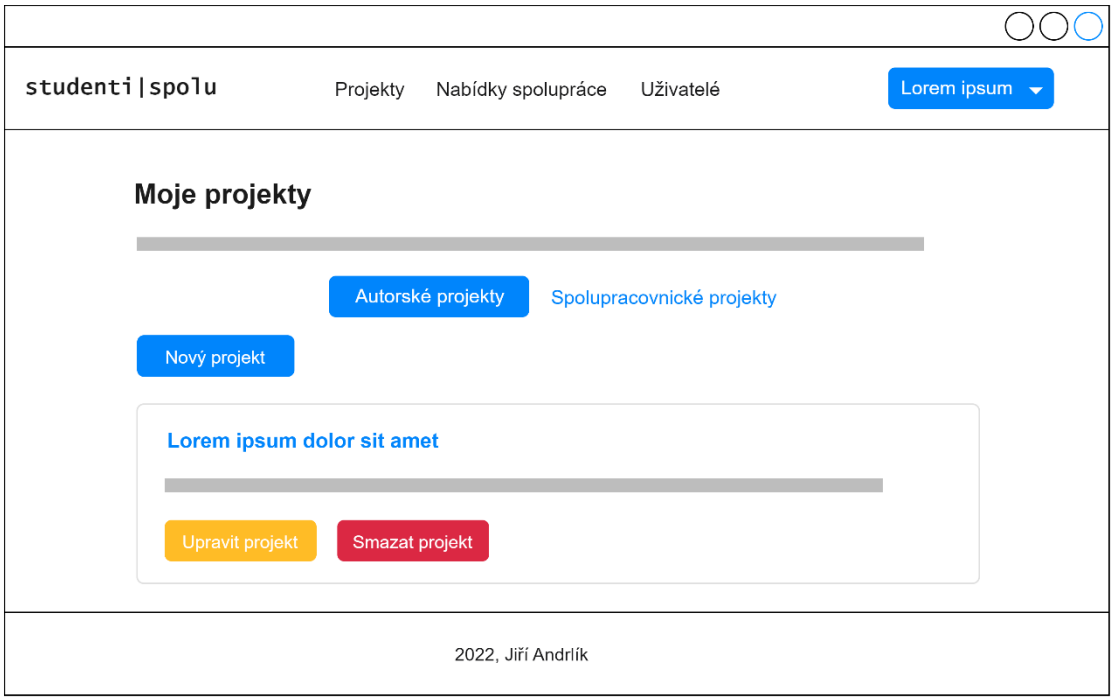
Detail uživatele:



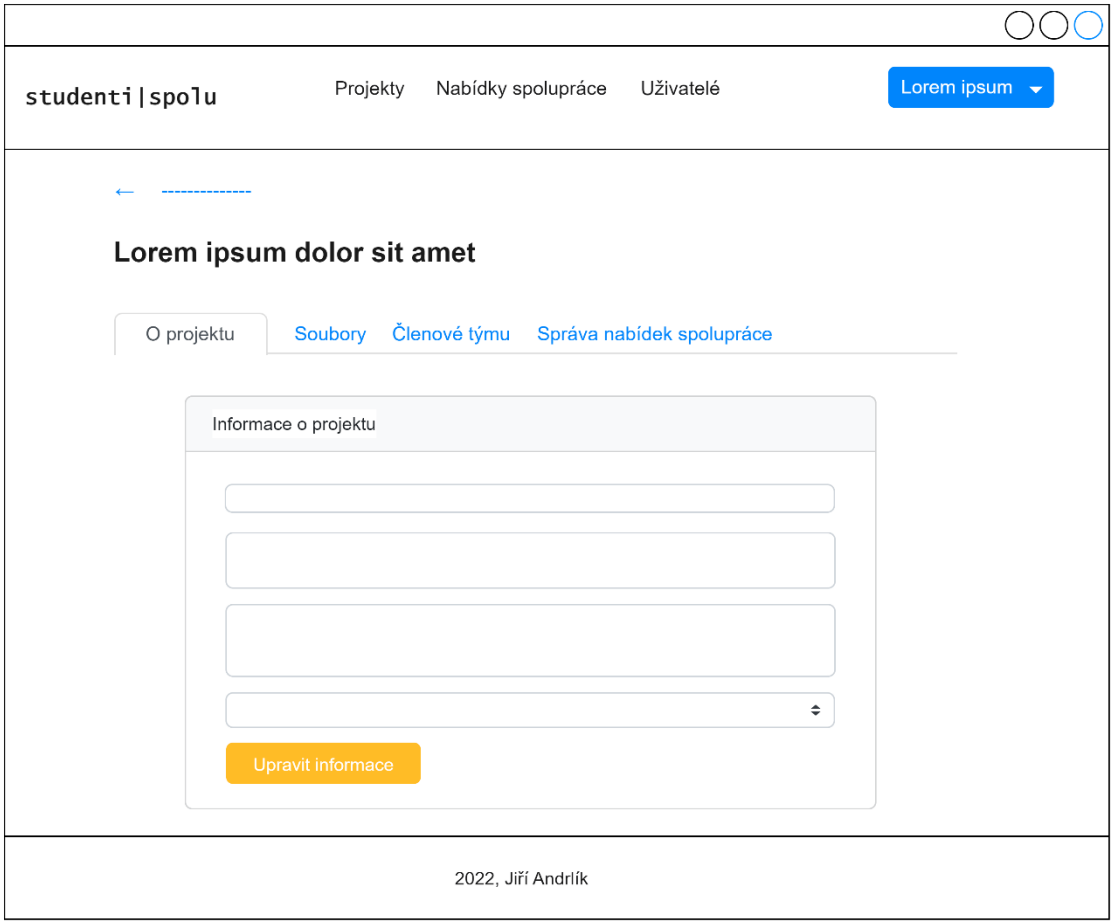
Můj profil:


studenti|spolu
Projekty
Nabídky spolupráce
Uživatelé
Lorem ipsum
Můj profil
Popis uživatele
Uložit změny
Změna přihlašovacích údajů
Uložit změny
Znalosti a dovednosti
Aplikovaná elektrotechnika
Copywriting
Daně
Ekonomie
Elektronika a informační technologie
Uložit změny
2022, Jiří Andrlík

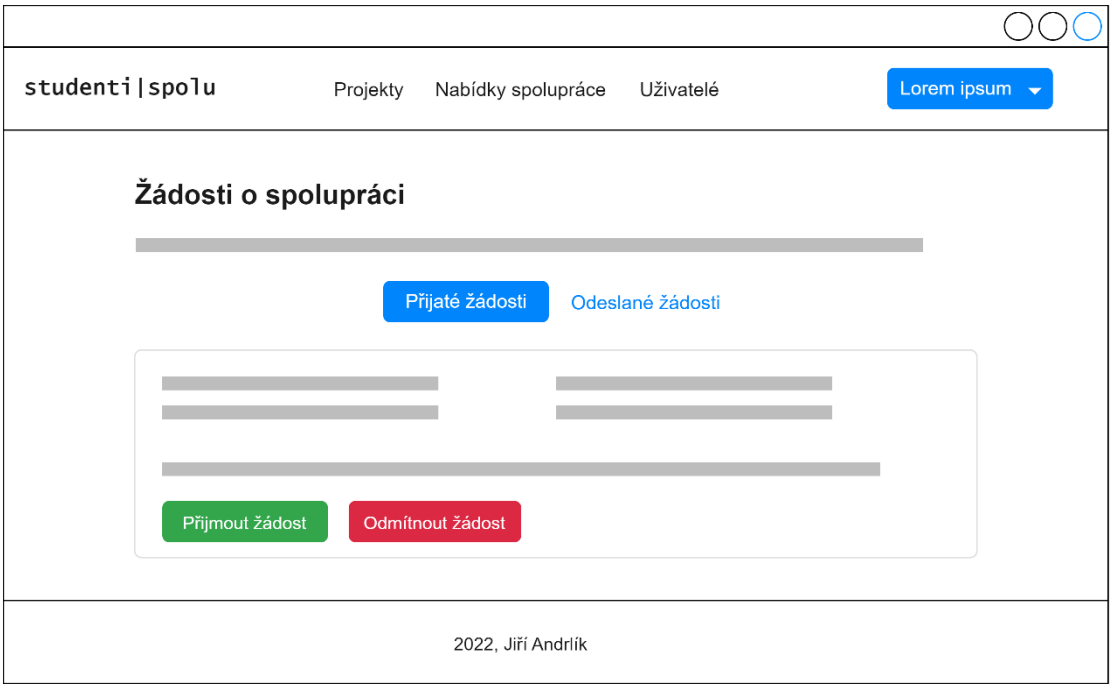
Moje projekty:



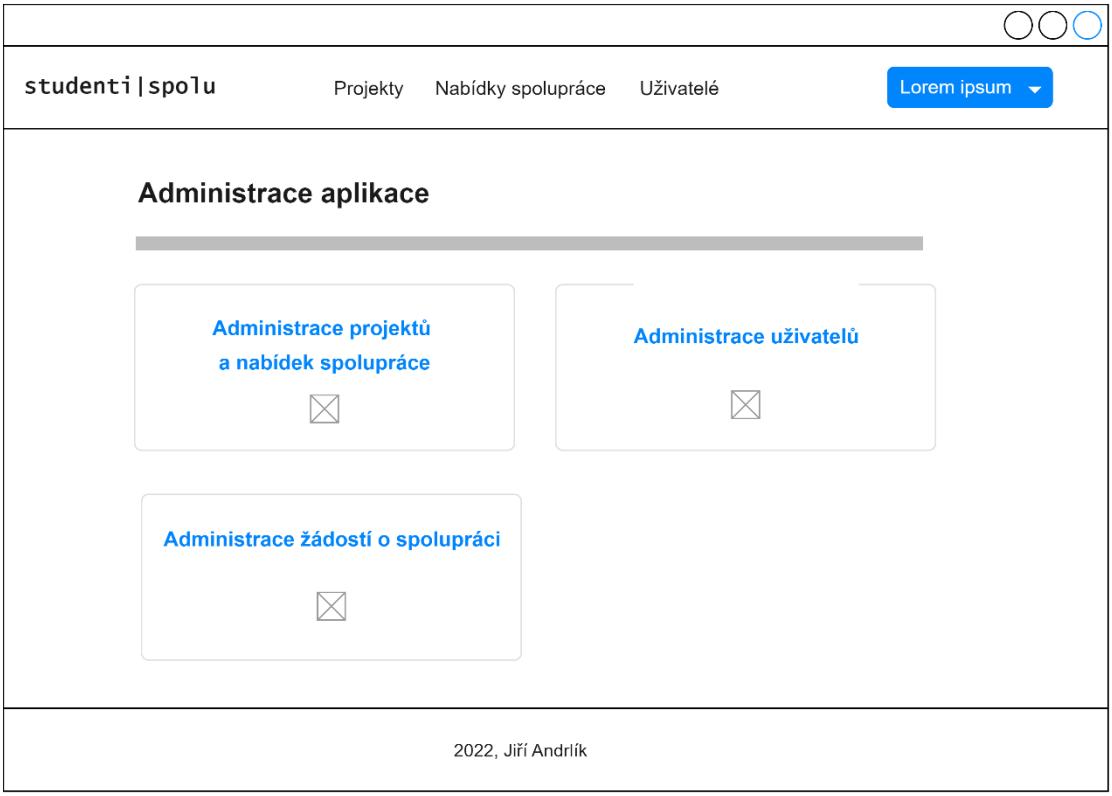
Úprava mého projektu:



Žádosti o spolupráci:



Administrace aplikace:

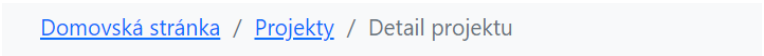


Příloha B: Uživatelská dokumentace

Domovská stránka systému

Uživatelské rozhraní obsahuje klasické rozložení prvků s hlavičkou a menu umístěným v horní části stránky, s kontejnerem pro hlavní obsah stránky a s patičkou umístěnou v dolní části stránky. Odkazy na stránky, které jsou určené pro všechny uživatele, jsou dostupné přímo z menu umístěného v hlavičce, zatímco odkazy na stránky vztažené ke konkrétnímu přihlášenému uživateli je možné nalézt v rozevíracím seznamu se jménem přihlášeného uživatele, umístěném v pravé části hlavičky stránky.

Na všech stránkách systému naleznete navigační lištu s odkazy na předcházející stránky, která slouží pro lepší orientaci, na jaké stránce se právě nacházíte.

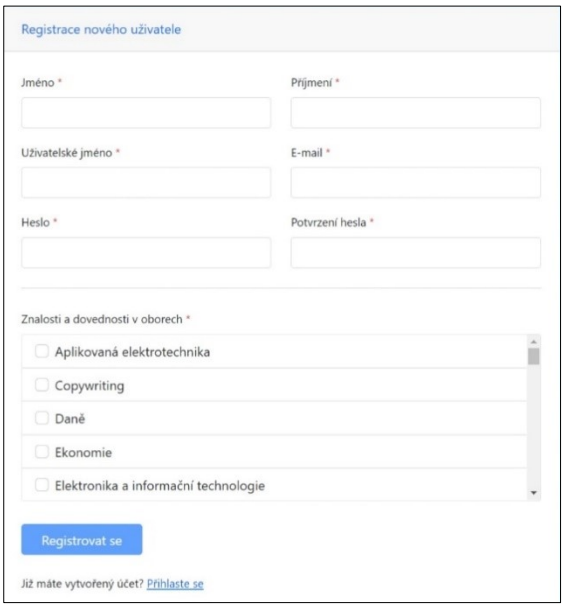


Registrace nového uživatelského profilu

1. Přejděte na úvodní stránku informačního systému.

2. Do informačního systému se můžete registrovat přes tlačítko *Registrace* umístěné v pravé části hlavičky stránky.

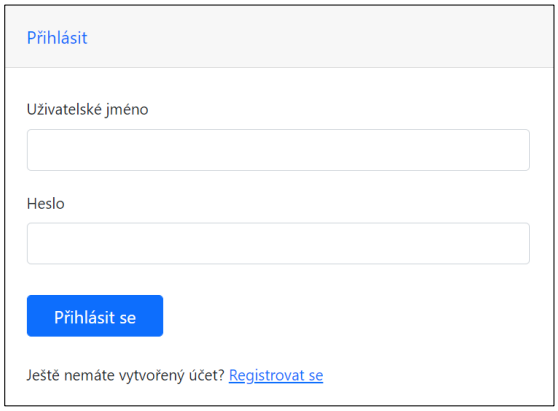
3. Vyplňte registrační formulář a klikněte na tlačítko *Registrovat se*. Aby bylo možné dokončit registraci, je potřeba zvolit alespoň jeden obor, ve kterém máte znalosti a dovednosti.



Přihlášení do informačního systému

1. Do informačního systému se můžete registrovat přes tlačítko *Přihlásit* umístěné v pravé části hlavičky stránky.

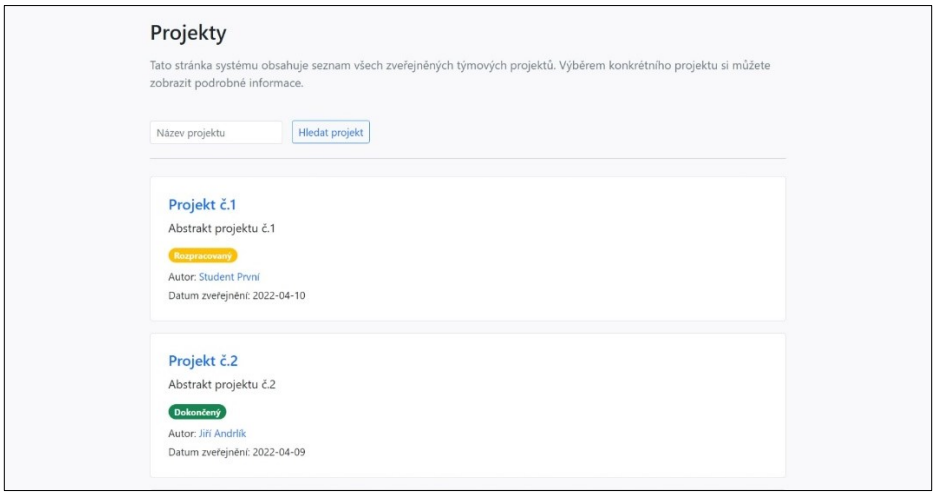
2. Vyplňte přihlašovací formulář a klikněte na tlačítko *Přihlásit se*.



Zobrazení seznamu všech zveřejněných projektů

1. V horním menu vyberte odkaz *Projekty*.

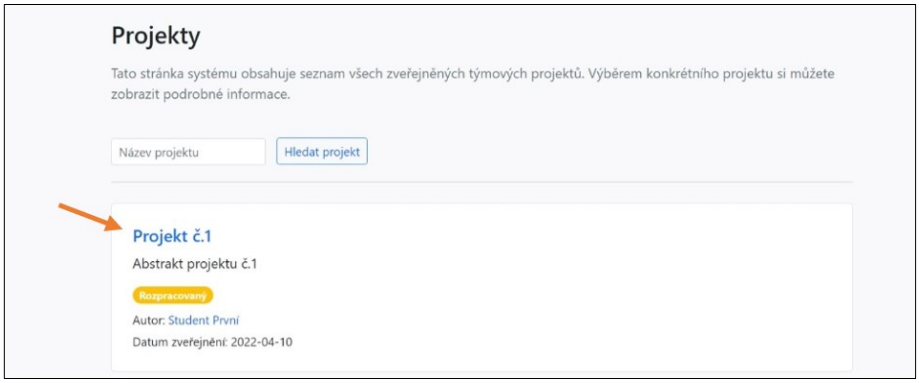
2. Zobrazí se stránka se seznamem všech zveřejněných projektů. Tato stránka nevyžaduje přihlášení uživatele.



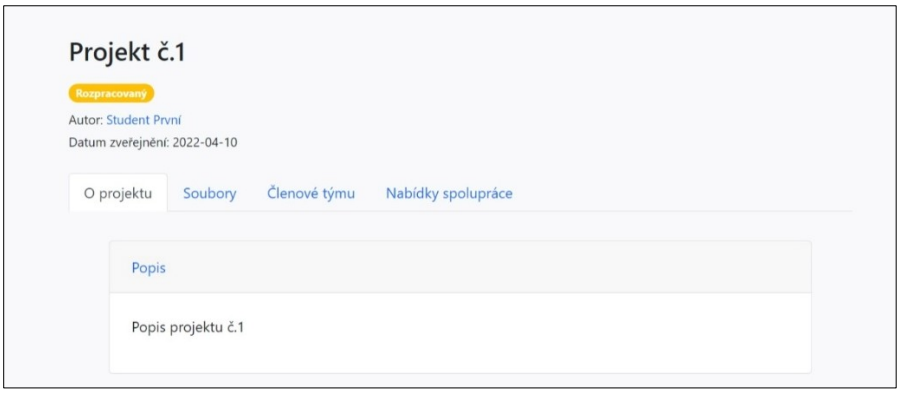
3. V seznamu zveřejněných projektů můžete vyhledávat projekty dle jejich názvu.

Zobrazení detailu vybraného zveřejněného projektu

1. Detail projektu zobrazíte kliknutím na jeho název v seznamu všech zveřejněných projektů.



2. Zobrazí se stránka s detailními informacemi o vybraném projektu.

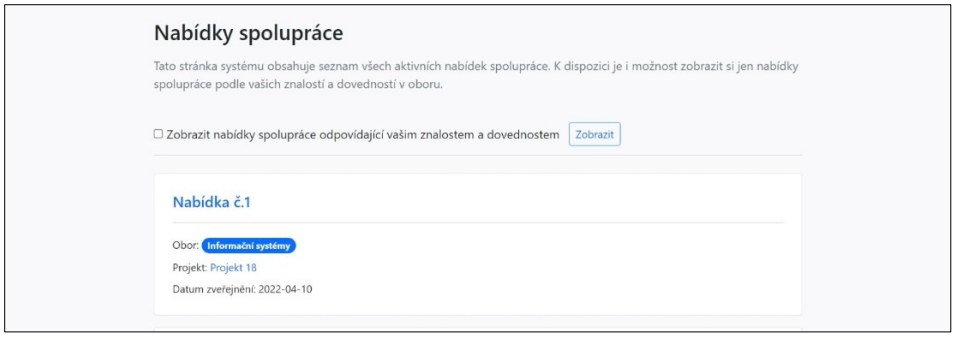


3. Stránka obsahuje sekce, mezi kterými lze přepínat pro zobrazení dalších informací.

Zobrazení seznamu aktivních nabídek spolupráce

1. V horním menu vyberte odkaz *Nabídky spolupráce.*

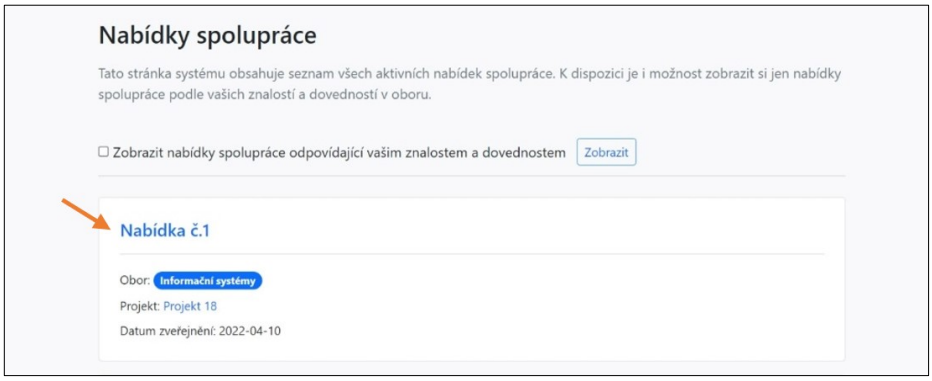
2. Zobrazí se stránka se seznamem všech aktivních nabídek spolupráce.



3. Vybráním checkboxu a potvrzením pomocí tlačítka *Zobrazit* si můžete zobrazit vyfiltrované nabídky spolupráce odpovídající vašim znalostem a dovednostem.

Zobrazení detailu vybrané aktivní nabídky spolupráce

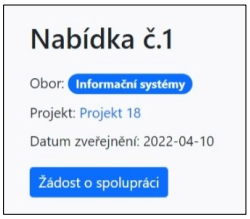
1. Detail nabídky spolupráce zobrazíte kliknutím na její název v seznamu všech aktivních nabídek spolupráce.



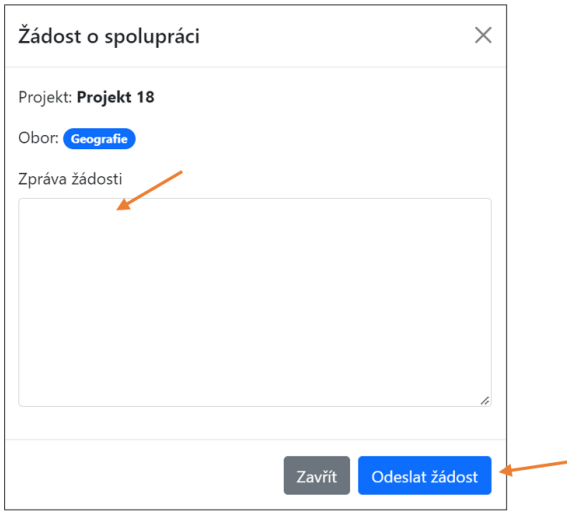
2. Zobrazí se stránka s detailními informacemi o vybrané nabídce spolupráce.

Odeslání žádosti o spolupráci reagující na vybranou nabídku spolupráce

1. V zobrazení detailu vybrané nabídky spolupráce vyberte tlačítko *Žádost o spolupráci*.



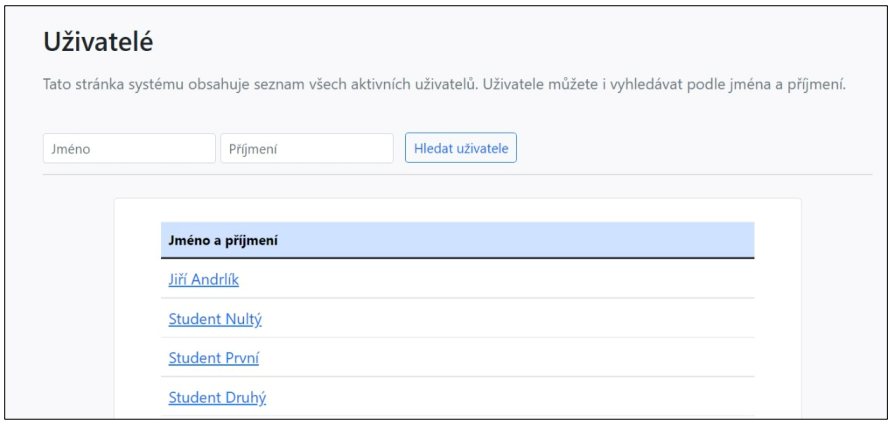
2. Vyplňte zprávu, kterou odešlete autorovi projektu spolu se žádostí o spolupráci a potvrďte odeslání kliknutím na tlačítko *Odeslat žádost*.



Zobrazení seznamu všech uživatelů informačního systému

1. V horním menu vyberte odkaz *Uživatelé.*

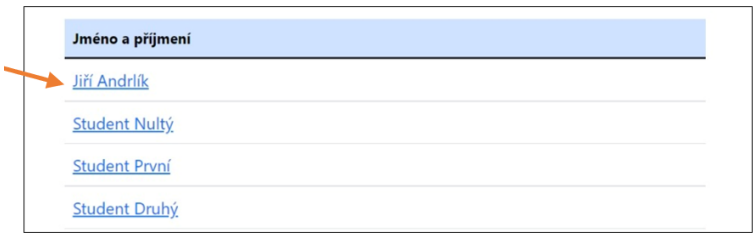
2. Zobrazí se stránka se seznamem všech aktivních uživatelů informačního systému.



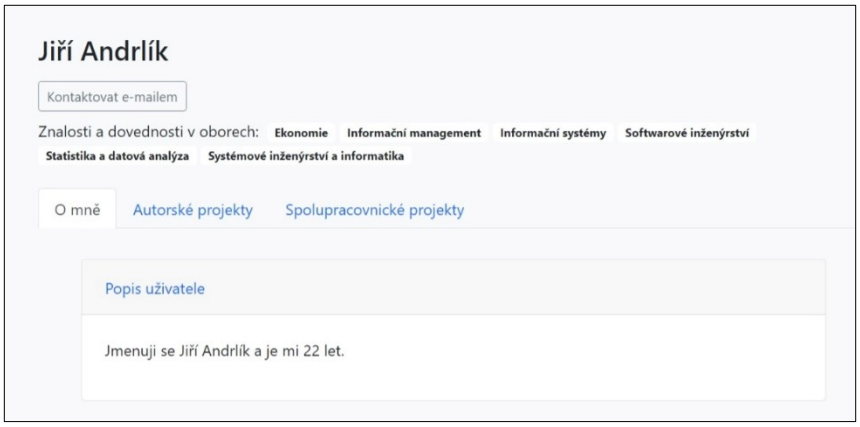
3. V seznamu zveřejněných uživatelů můžete vyhledávat jejich jména a příjmení.

Zobrazení detailu vybraného uživatele

1. Detail uživatele zobrazíte kliknutím na jeho jméno a příjmení v seznamu všech uživatelů.



2. Zobrazí se stránka s detailními informacemi o vybraném uživateli.

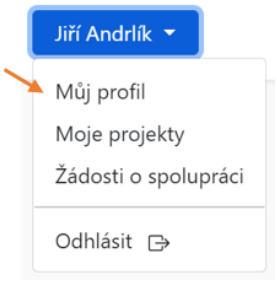


3. Stránka obsahuje sekce, mezi kterými lze přepínat pro zobrazení dalších informací.

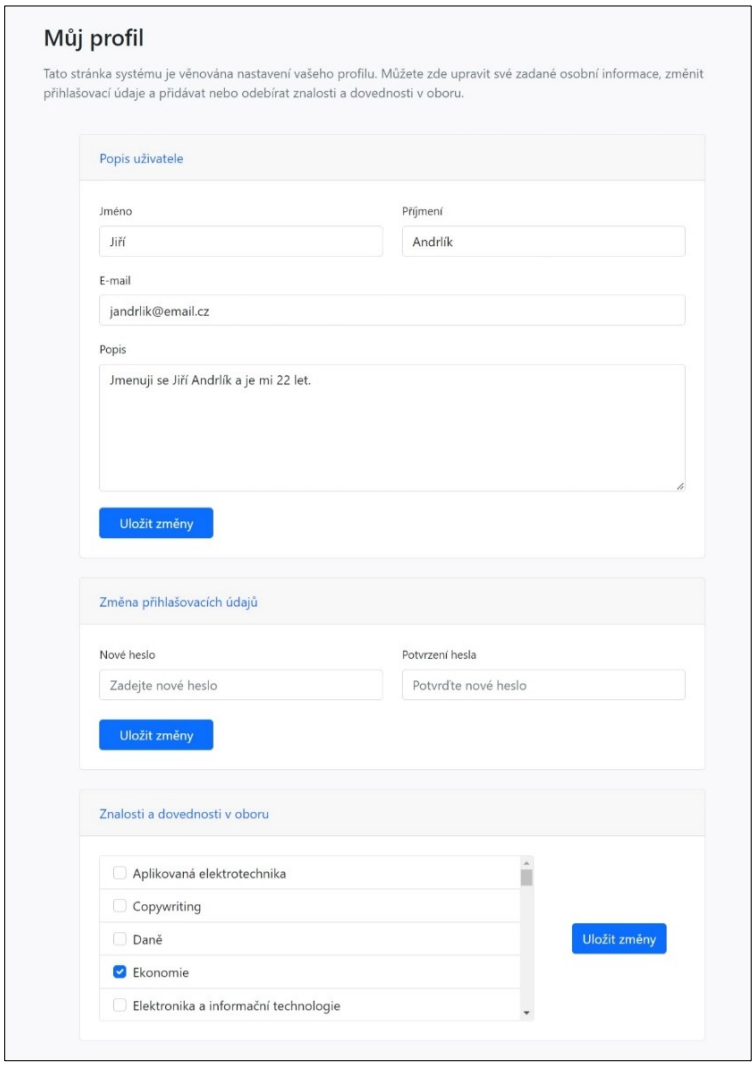
4. Vybraného uživatele můžete kontaktovat e-mailem.

Zobrazení uživatelského profilu

1. V rozbalovacím seznamu vyberte *Můj profil.*



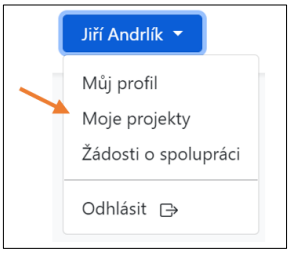
2. Zobrazí se stránka pro úpravu uživatelského profilu, která je rozdělena do tří sekcí – Popis uživatele, Změna přihlašovacích údajů, Znalosti a dovednosti v oboru.



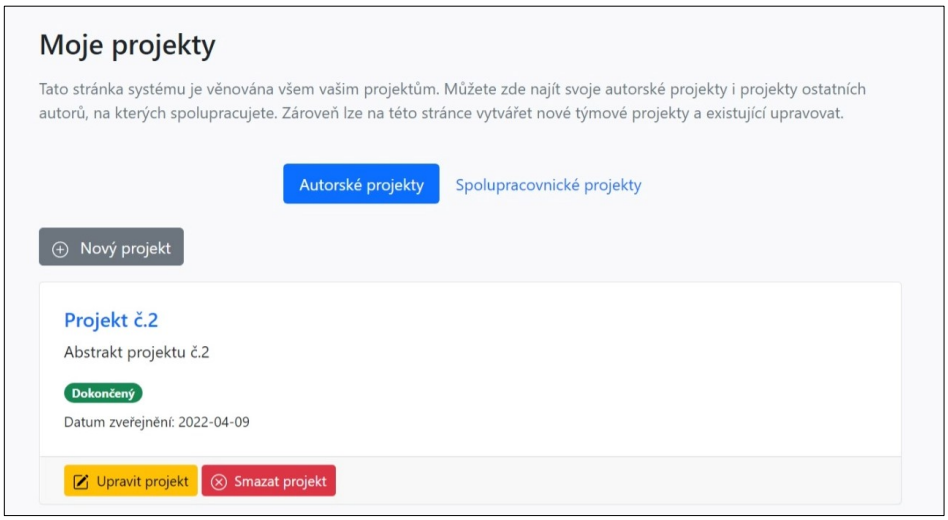
3. V každé sekci můžete změnit informace. Úpravu potvrďte stisknutím příslušného tlačítka *Uložit změny.*

Zobrazení stránky s vašimi projekty

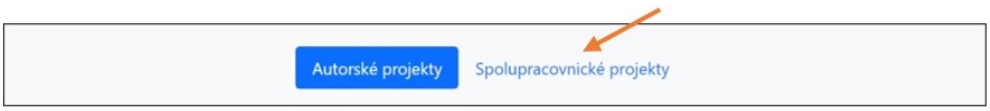
1. V rozbalovacím seznamu vyberte *Moje projekty.*



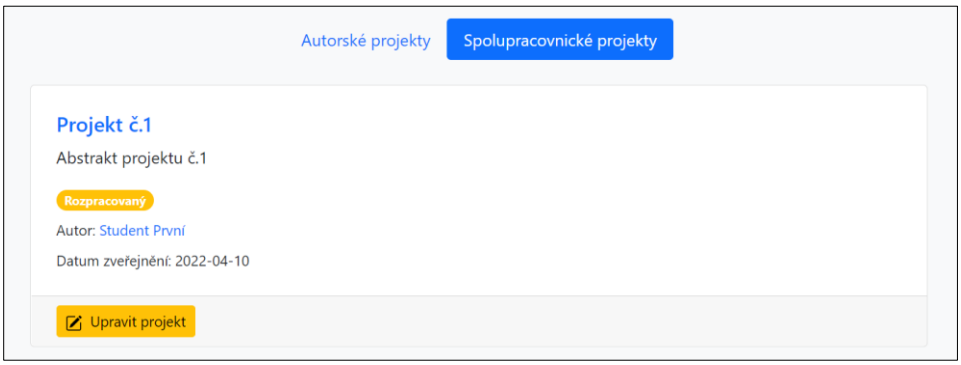
2. Zobrazí se stránka s vašimi autorskými a spolupracovnickými projekty. Jako první se vám zobrazí sekce s autorskými projekty, tj. projekty, které jste vytvořili. Tyto projekty můžete upravovat nebo smazat kliknutím na tlačítko *Upravit projekt* nebo *Smazat projekt*.



3. Pro přechod do sekce spolupracovnických projektů klikněte na *Spolupracovnické projekty.* Spolupracovnické projekty jsou takové projekty, které jste nevytvořili, ale jste součástí projektového týmu s rolí Spolupracovník, (tj. existuje vámi odeslaná a autorem projektu schválená žádost o spolupráci).

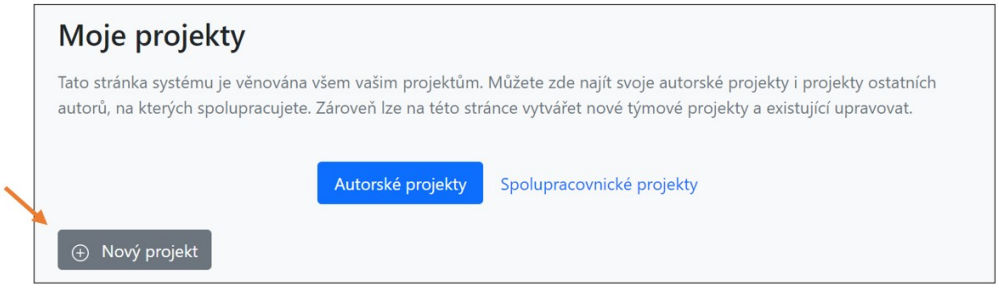


4. Zobrazí se seznam spolupracovnických projektů, tj. projekty, na kterých pracujete s rolí Spolupracovník. Po stisku tlačítka *Upravit projekt* můžete jako spolupracovník provádět úpravy pouze v sekci *Soubory*.

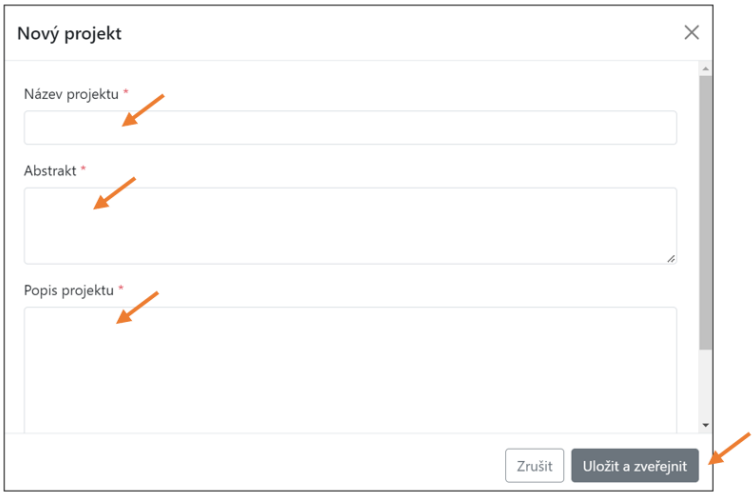


Vytvoření nového týmového projektu

1. Přejděte na stránku *Moje projekty*.
2. Klikněte na tlačítko *Nový projekt* v sekci *Autorské projekty*.



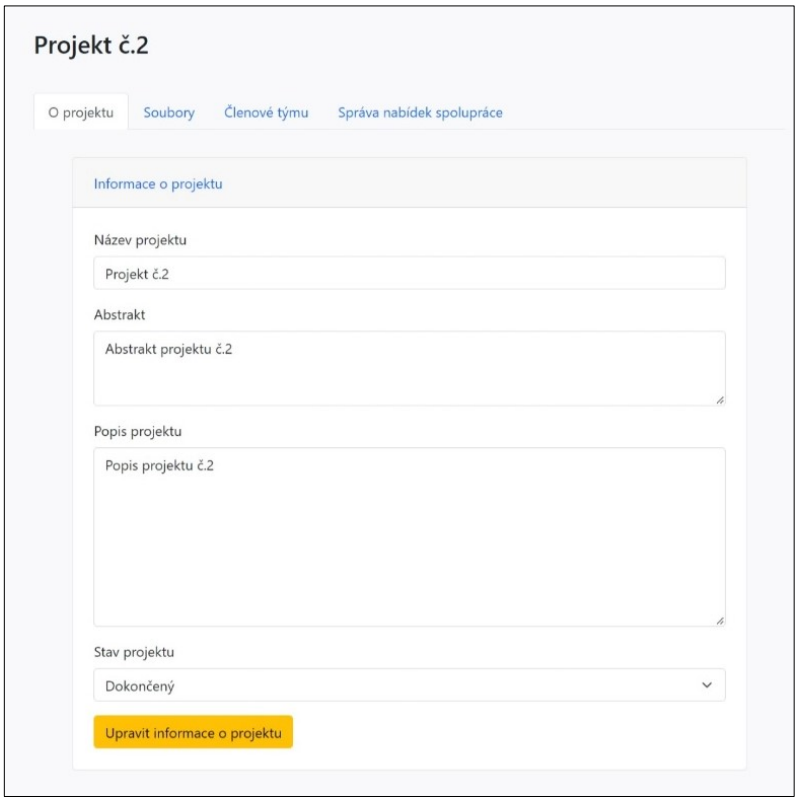
3. Vyplňte formulář s informacemi o novém projektu a potvrďte kliknutím na tlačítko *Uložit a zveřejnit.*



Zobrazení stránky pro úpravu vašeho projektu

1. V seznamu autorských nebo spolupracovnických projektů stiskněte u konkrétního projektu tlačítko *Upravit projekt.*

2. Zobrazí se stránka určená pro úpravu konkrétního projektu. Můžete změnit informace o projektu a úpravu potvrdit stisknutím tlačítka *Upravit informace o projektu.*



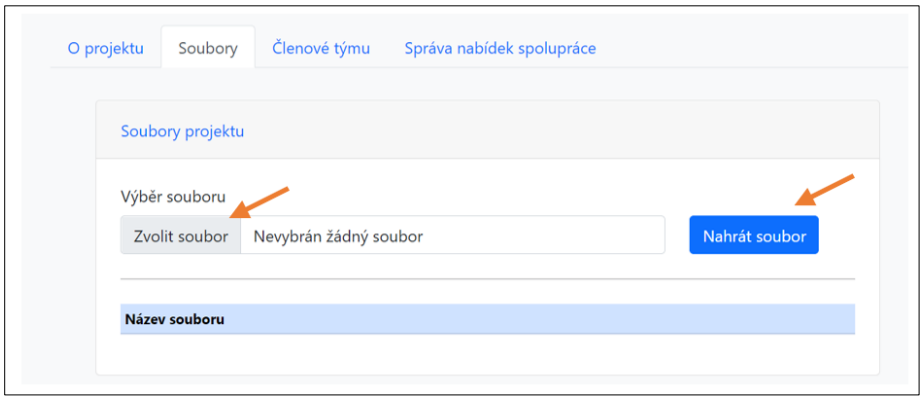
3. Stránka obsahuje sekce, mezi kterými lze přepínat pro zobrazení a úpravu dalších informací.

Nahrání souboru k vašemu (autorskému nebo spolupracovnickému) projektu

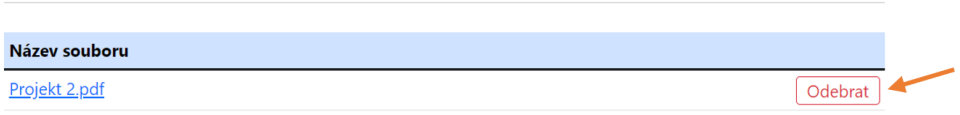
1. Přejděte na stránku pro úpravu vašeho konkrétního projektu.

2. Přejděte do sekce *Soubory.*

3. Vyberte soubor, který chcete nahrát a potvrďte stiskem tlačítka *Nahrát soubor.*



4. Nahraný soubor můžete odstranit stisknutím tlačítka *Odebrat* a potvrzením vyskakovacího okna.

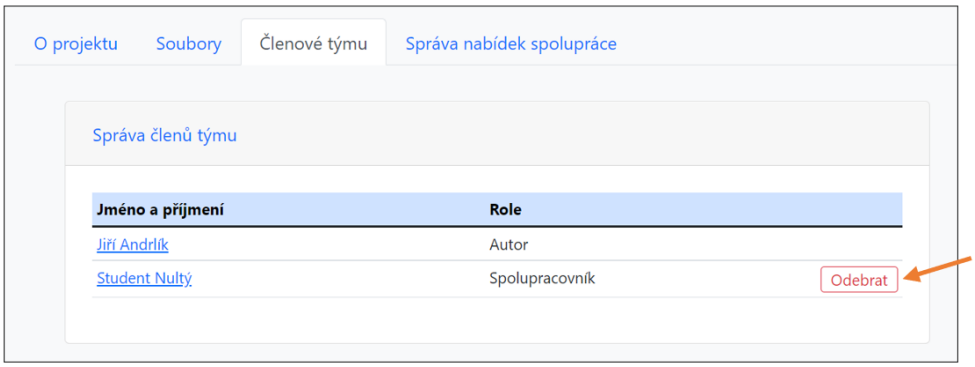


Odebrání člena projektového týmu

1. Přejděte na stránku pro úpravu vašeho konkrétního projektu.

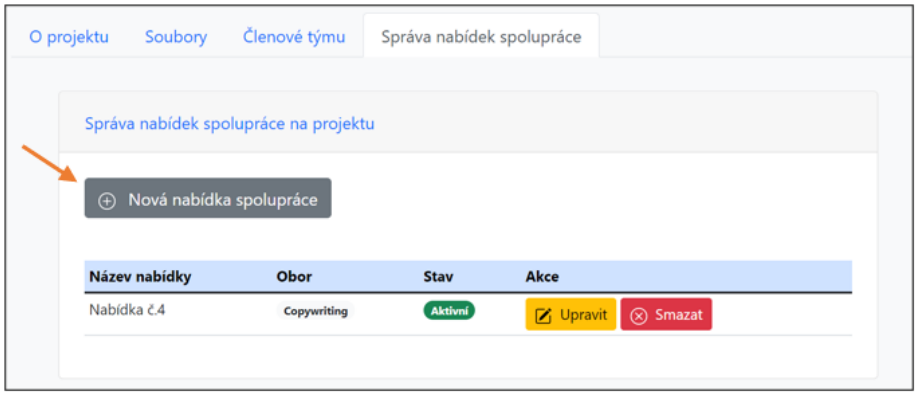
2. Přejděte do sekce *Členové týmu.*

3. Odebrání člena týmu provedete stiskem tlačítka *Odebrat* a potvrzením vyskakovacího okna. Odebrání člena projektového týmu může provést pouze autor projektu.

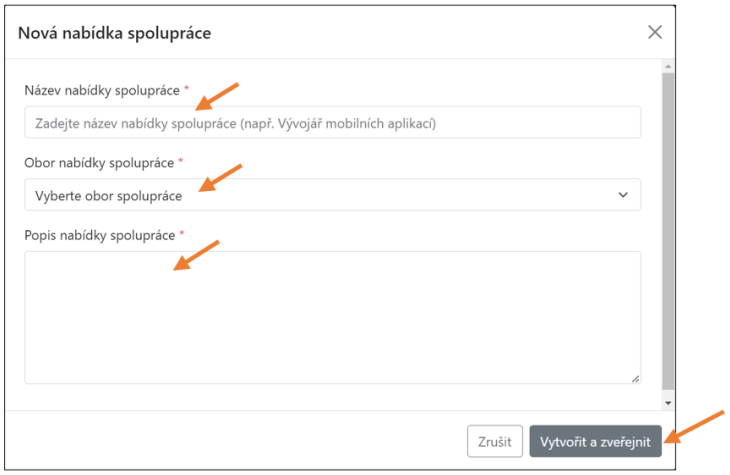


Správa nabídek spolupráce

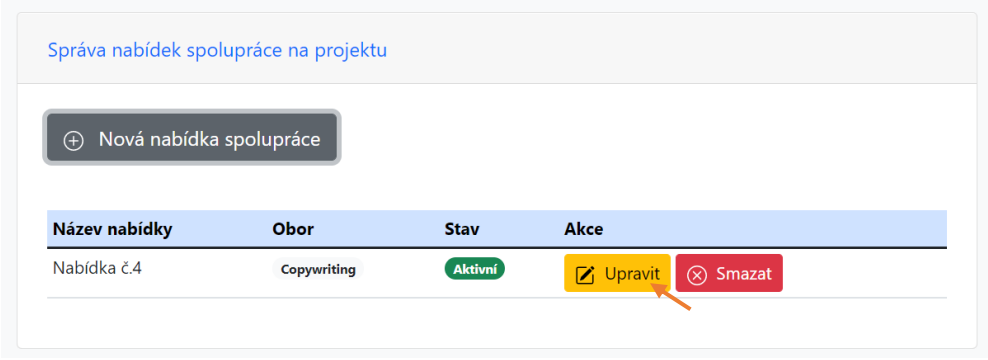
1. Přejděte na stránku pro úpravu vašeho konkrétního projektu.
2. Přejděte do sekce *Správa nabídek spolupráce.*
3. Novou nabídku spolupráce vytvoříte stisknutím tlačítka *Nová nabídka spolupráce.*



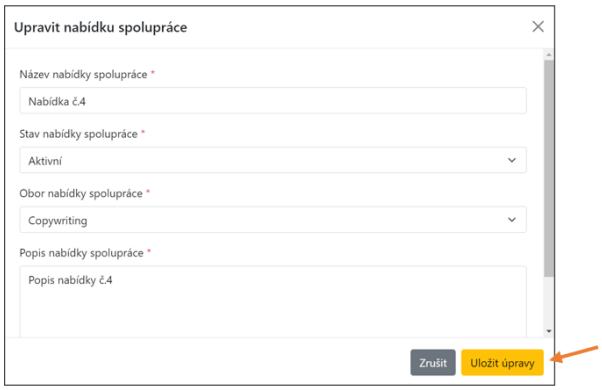
4. Vyplňte údaje o nové nabídce spolupráce a potvrďte kliknutím na tlačítko *Vytvořit a zveřejnit*. Obor, ke kterému je nabídka spolupráce vztažena, vyberte z rozbalovacího seznamu.



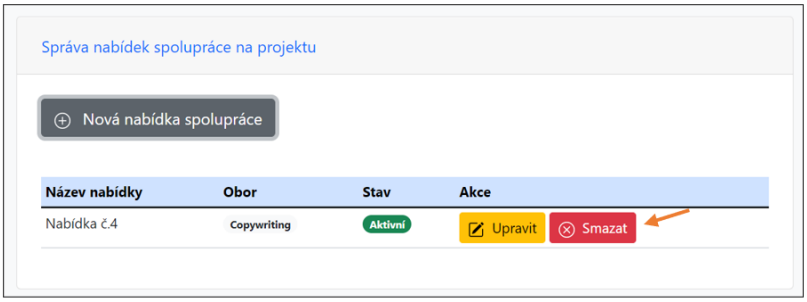
5. Vytvořenou nabídku spolupráce můžete upravit stiskem tlačítka *Upravit.*



6. Změňte údaje o nabídce spolupráce a potvrďte úpravu kliknutím na tlačítko *Uložit úpravy*.

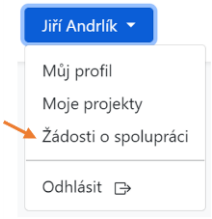


7. Vytvořenou nabídku spolupráce můžete odstranit stisknutím tlačítka *Smazat* a potvrzením vyskakovacího okna.

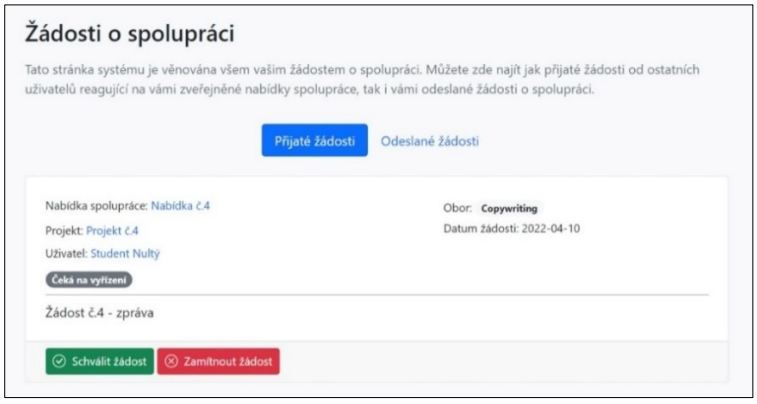


Správa přijatých žádostí

1. V rozbalovacím seznamu vyberte *Žádosti o spolupráci.*



2. Zobrazí se stránka se seznamem přijatých žádostí o spolupráci v sekci *Přijaté žádosti*.



3. Schválení žádostí provedete stiskem tlačítka *Schválit žádost* u konkrétní žádosti o spolupráci.

4. Schválením žádosti získáte nového člena projektového týmu, kterému bude automaticky přiřazena role *Spolupracovník.*

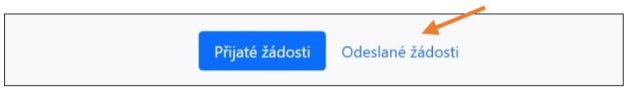
5. Žádost, která je ve stavu *Schválená* nebo *Zamítnutá*, lze znovu vrátit do stavu *Čeká na vyřízení* stiskem tlačítka *Znovu posoudit*.

6. Zamítnutí žádostí provedete stiskem tlačítka *Zamítnout žádost* u konkrétní žádosti o spolupráci.

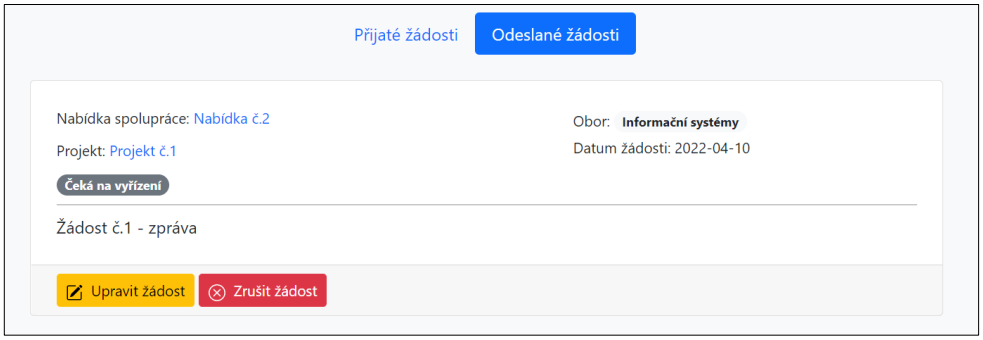
Správa odeslaných žádostí

1. Přejděte na stránku *Žádosti o spolupráci.*

2. Pro přechod do sekce odeslaných žádostí klikněte na tlačítko *Odeslané žádosti*.

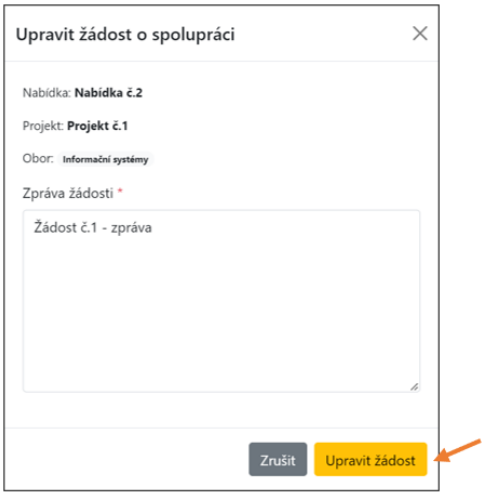


3. Zobrazí se sekce se seznamem odeslaných žádostí o spolupráci.



4. U odeslané žádosti, která se nachází ve stavu *Čeká na vyřízení*, je možné upravit zprávu, která byla autorovi odeslána spolu s danou žádostí o spolupráci.

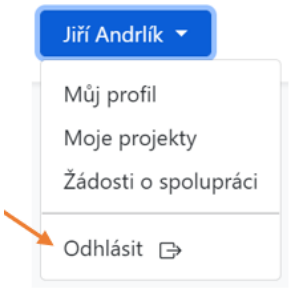
5. Změňte obsah zprávy a potvrďte úpravu stiskem tlačítka *Upravit žádost*.



6. Žádost o spolupráci, která se nachází ve stavu *Čeká na vyřízení*, je možné zrušit stisknutím tlačítka *Zrušit žádost.* Daná žádost o spolupráci se již autorovi projektu nebude zobrazovat.

Odhlášení z informačního systému

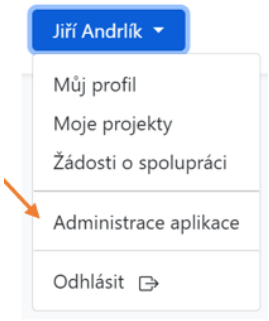
1. Odhlášení je možné provést z jakékoliv stránky informačního systému stisknutím *Odhlásit* v rozbalovacím seznamu s vaším jménem.



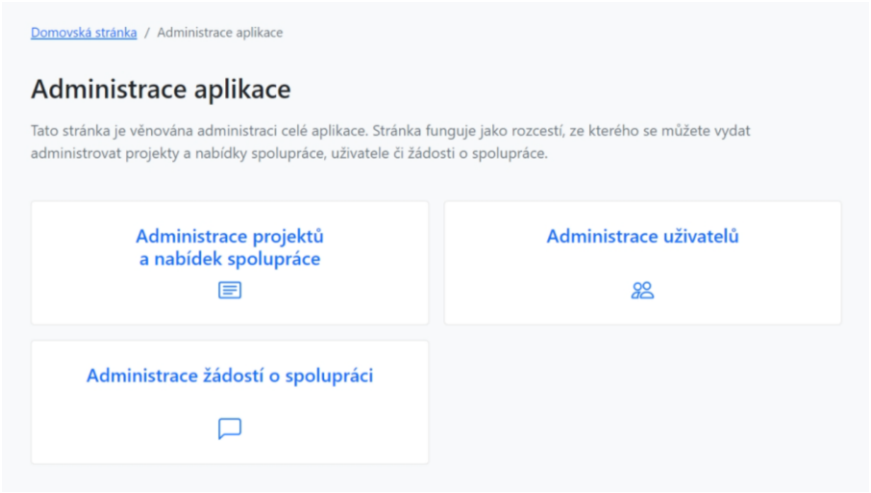
Administrace aplikace

Sekce určená pro administraci aplikace je dostupná pouze pro uživatele s právem Administrátor nebo Super Administrátor.

1. V rozbalovacím seznamu vyberte *Administrace aplikace.*



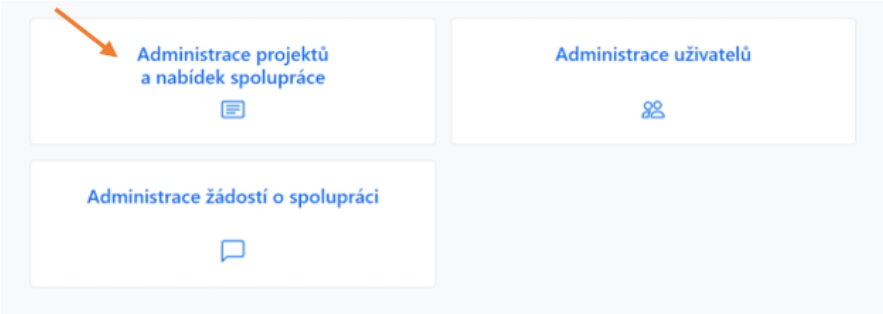
2. Zobrazí se úvodní stránka administračního rozhraní aplikace.



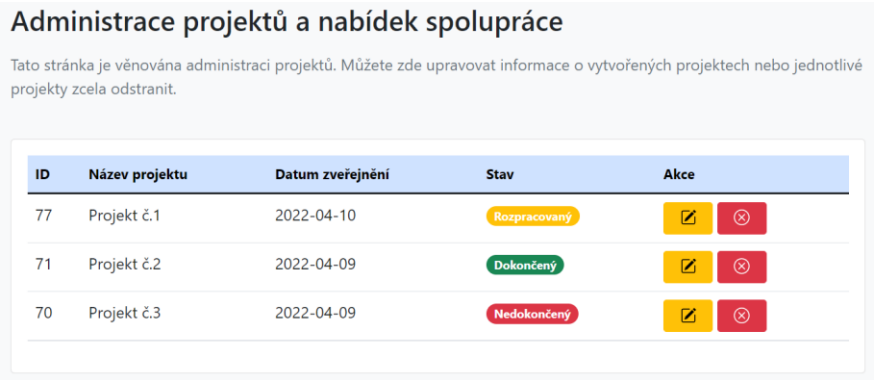
Administrace projektů a nabídek spolupráce

1. Přejděte na úvodní stránku administračního rozhraní aplikace.

2. Klikněte na odkaz *Administrace projektů a nabídek spolupráce*.



3. Zobrazí se seznam všech projektů, které jsou uloženy v informačním systému.



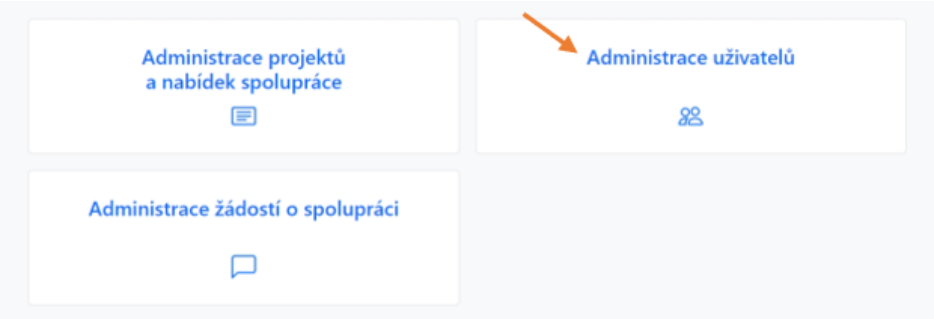
4. Vybraný projekt můžete upravit stiskem tlačítka *Akce* určeného pro úpravu. Následná administrátorská úprava projektu probíhá stejně jako úprava autorského projektu na stránce *Moje projekty.* Administrace nabídek spolupráce je prováděna na stránce pro úpravu projektu v sekci *Správa nabídek spolupráce*.

5. Vybraný projekt můžete zcela odstranit z informačního systému stiskem tlačítka *Akce* určeného pro odstranění. Odstraněním vybraného projektu dojde k odstranění i všech ostatních záznamů, které jsou na něj navázány.

Administrace uživatelů

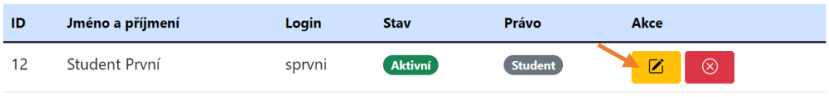
1. Přejděte na úvodní stránku administračního rozhraní aplikace.

2. Klikněte na odkaz *Administrace uživatelů*.

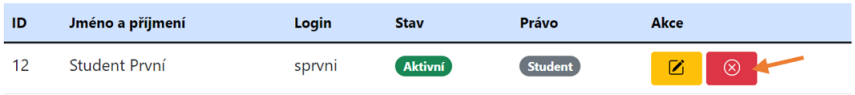


3. Zobrazí se seznam všech uživatelů, které jsou v informačním systému registrováni.

4. Vybraného uživatele můžete upravit stiskem tlačítka *Akce* určeného pro úpravu. Následná administrátorská úprava uživatele probíhá stejně jako úprava uživatelského profilu na stránce *Můj profil*. Super Administrátor je na stránce administrátorské úpravy uživatele oprávněn měnit práva jednotlivých uživatelů.



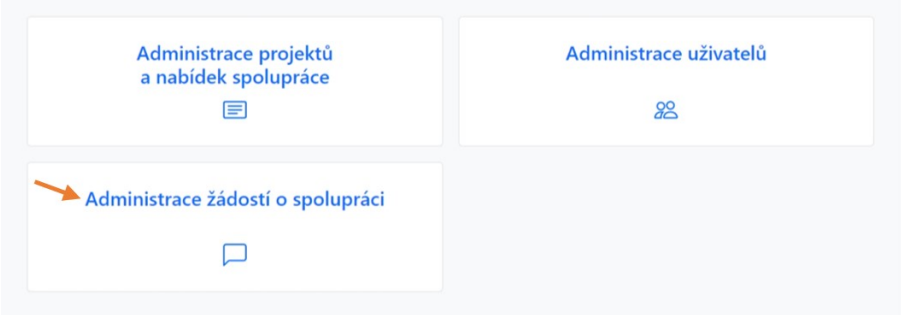
5. Vybraného uživatele můžete zcela odstranit z informačního systému stiskem tlačítka *Akce* určeného pro odstranění. Odstraněním uživatele dojde k odstranění i všech ostatních záznamů, které jsou na něj navázány.



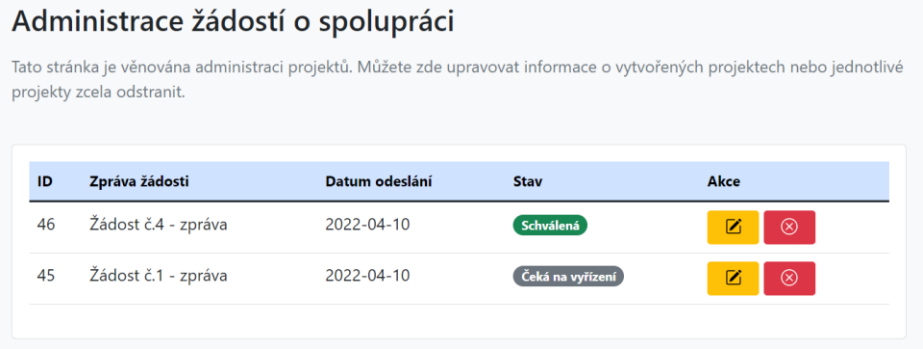
Administrace žádostí o spolupráci

1. Přejděte na úvodní stránku administračního rozhraní aplikace.

2. Klikněte na odkaz *Administrace žádostí o spolupráci*.



3. Zobrazí se seznam všech žádostí o spolupráci.



4. Vybranou žádost o spolupráci můžete upravit stiskem tlačítka Akce určeného pro úpravu. Následná úprava zahrnuje možnost upravit zprávu, která byla se žádostí o spolupráci odeslána, spolu s možností měnit stav žádosti o spolupráci.

5. Vybraného uživatele můžete zcela odstranit z informačního systému stiskem tlačítka *Akce* určeného pro odstranění.

Abstrakt

Andrlík, J. (2022). *Návrh a implementace webového informačního systému* (Bakalářská práce), Západočeská univerzita v Plzni, Fakulta ekonomická, Česko.

Klíčová slova: informační systém, webová aplikace, týmové projekty, týmová spolupráce, PHP, Laravel, MVC, testování použitelnosti

Bakalářská práce se zabývá návrhem a implementací webového informačního systému, který bude určen pro studenty vysokých škol ke zveřejňování informací o týmových projektech a hledání možností spolupráce v konkrétním oboru na těchto projektech. Navržený informační systém bude sloužit i pro prezentaci práce studentů na jednotlivých projektech.

Rešeršní část bakalářské práce je zaměřena na představení základních pojmů, popis vybraných témat souvisejících s návrhem a implementací webového informačního systému a rešerši současné situace a dostupných nástrojů týkajících se hledání možností spolupráce na projektech. Na základě postupů stanovených v části metodiky řešení bude provedena specifikace požadavků, analýza procesní orientace, návrh datové a softwarové architektury spolu s implementací a následným testováním použitelnosti webového informačního systému.

Abstract

Andrlík, J. (2022). *Design and implementation of a web-based information system* (Bachelor Thesis). University of West Bohemia, Faculty of Economics, Czech Republic.

Key words: information system, web application, team projects, team collaboration, PHP, Laravel, MVC, usability testing

This bachelor thesis focuses on the design and implementation of a web-based information system which will allow university students to publish information about their team projects and search for opportunities to collaborate on these projects in specific fields. The designed information system will also be used for sharing the work done by students on individual projects.

The research part of the bachelor thesis introduces basic concepts and terminology, and provides the description of selected topics related to the design and implementation of the web-based information system. This part of the thesis also explores the tools that are currently available when searching for opportunities to collaborate on projects. Based on the defined research methodology, the results chapter of the thesis is devoted to requirements specification, analysis of process orientation, design of data architecture and software architecture together with implementation and usability testing.