ČESKÉ VYSOKÉ
UČENÍ TECHNICKÉ
V PRAZE

F3

Fakulta elektrotechnická
Katedra počítačů

Bakalářská práce

Řízení kroků k úspěšnému dokončení studia

Daniel Groschup
Softwarové inženýrství a technologie

Srpen 2020
Vedoucí práce: Lukáš Zoubek

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Groschup** Jméno: **Daniel** Osobní číslo: **457813**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Softwarové inženýrství a technologie**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Řízení kroků k úspěšnému dokončení studia

Název bakalářské práce anglicky:

Managing steps to successfully complete studies

Pokyny pro vypracování:

1) Popište nezbytné kroky pro úspěšné dokončení studia v bakalářských a magisterských studijních programech na FEL ČVUT z pohledu studenta.
2) Definujte požadavky na aplikaci podporující úspěšné provedení těchto kroků (zakládání projektu a úkolů, komentáře, změny úkolů a termínů, notifikace, správa studentů).
3) Zpracujte analytický návrh pro implementaci takové aplikace, aplikaci implementujte a nasaďte k testovacímu provozu.
4) Definujte testovací scénáře pro ověření správné funkčnosti aplikace.
5) Proveďte testování a vyhodnoťte.

Seznam doporučené literatury:

[1] TAYNTOR, Christine B. Project management tools and techniques for success: CRC Press, c2010. ISBN 978-143-9816-301.
[2] AHMED, Ashfaque. Software project management: a process-driven approach: CRC Press, c2012. ISBN 978-143-9846-551.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Lukáš Zoubek, Centrum znalostního managementu FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **14.02.2020** Termín odevzdání bakalářské práce: **14.08.2020**

Platnost zadání bakalářské práce: **30.09.2021**

Ing. Lukáš Zoubek podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. Datum převzetí zadání

Podpis studenta

Poděkování / Prohlášení

Chtěl bych především poděkovat svému vedoucímu práce Ing. Lukáši Zoubkovi za pevné nervy při mentoringu a řízení mé závěrečné práce. Dále bych chtěl poděkovat Ing. Jiřímu Fryčovi za pomoc při řešení obsahu implementace a základním seznámením s funkcionalitami aplikace Moodle. Rád bych také poděkoval Bc. Janu Veselému za pomoc s instalací aplikace Moodle a za jeho rady při implementaci.Také všem testerům, kteří mojí aplikaci testovali. Zuzaně Pavlatové za pravopisnou kontrolu a rady během psaní. Poslední poděkování patří mé rodině za podporu, které se mi během studia dostalo a za důvěru v dokončení mého studia.

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 14. 8. 2020

.....

Abstrakt / Abstract

Tato práce se věnuje podpoře úspěšného ukončení studia studentů v bakalářských a magisterských programech na ČVUT FEL. Podporu zajišťuje implementovaný plugin spolu s kurzem ve fakultní aplikaci Moodle, které společně řídí kroky studenta v posledním ročníku jeho studia. Práce zahrnuje analýzu posloupnosti kroků vedoucích k dokončení studia, mezi které patří i absolvování předmětů souvisejících se závěrečnou prací. Na základě analýzy a shromážděných dotazů studentů bylo možné identifikovat požadavky na výsledné řešení. Výstupem je pak tedy plugin naplňující definovaný cíl. Jeho vytvoření předcházela rešerše možných řešení a dostupných systémů, které by bylo možné podobným způsobem rozšířit. Na závěr bylo provedeno testování implementovaného řešení. Realizováno bylo formou manuálního testování zaměřeného na funkčnost pluginu a uživatelského testování ověřujícího obsah a funkcionality kurzu v aplikaci Moodle.

Klíčová slova: Moodle, dokončení studia, závěrečná práce, semestrální projekt, bakalářská práce, diplomová práce, vývoj aplikací, projekt, proces, plugin

This work is devoted to the support of successful completion of students' studies in bachelor's and master's programs at CTU FEE. Support is provided by an implemented plugin together with a course in the faculty application Moodle. Together, they manage student's steps in the last year of his studies. The thesis includes an analysis of the sequence of steps leading to the completion of the studies, including the completion of subjects related to the final thesis. Based on the analysis and collected questions from students, it was possible to identify the requirements for the final solution. The output is then a plugin fulfilling the defined goal. Its creation was preceded by research of possible solutions and available systems, which could be extended in a similar way. Finally, testing of the implemented solution was performed. It was realized by manual testing focused on the functioning of the plugin and user testing verifying the content and functionality of the course in the Moodle application.

Keywords: Moodle, completion of studies, final thesis, semestral project, bachelor thesis, master thesis, application development, project, process, plugin

Title translation: Managing steps to successfully complete studies

Obsah /

Tabulky / Obrázky

Kapitola 1
Úvod

1.1 Předmluva a motivace

Na Fakultě elektrotechnické na ČVUT v Praze studovalo v roce 2018 celkem 2613 studentů [1] na magisterských a bakalářských studijních programech. Z toho 614 dokončovalo studium [1] a muselo řešit úkoly a povinnosti spojené s úspěšným zakončením studia. Některé základní úkoly jsou pro všechny studenty stejné a definované školním řádem. Tyto úkoly se však obtížně hledají a studenti neví, jak mají postupovat. Z toho důvodu by měla vzniknout aplikace, která by studentům pomohla s úspěšným dokončením studia.

1.2 Cíle a výstupy práce

Cílem je pomoci studentům řídit kroky k úspěšnému dokončení studia a seznámit je s jednotlivými úkoly, které je na této cestě čekají.

Výstupem této práce bude aplikace, která se bude snažit naplnit cíle definované v požadavcích. Aplikace tak bude jakýmsi průvodcem posledním ročníkem studia a informace, které poskytne studentům, by měly vést k úspěšnému dokončení jejich studia. Aplikace nicméně nebude suplovat oficiální zdroje a bude potřeba její obsah aktualizovat dle platných nařízení fakulty.

1.3 Struktura práce

Na začátku se zaměříme na vývoj aplikací a také předvedeme, že proces dokončení studia je projektem každého studenta. Poté se zaměříme na poslední rok studia, konkrétně na semestrální projekt, závěrečnou práci a státní závěrečné zkoušky. Sepíšeme seznam jednotlivých úkolů, které studenty v rámci plnění těchto částí čekají a problémy, se kterými se potýkají. Definujeme zde také podmínky, za kterých je možné určit, že student vstupuje do poslední fáze studia a detekovat ho tak v systému.

Následně přejdeme od analýzy posledního ročníku k sepsání požadavků na aplikaci, které jsme na základě této analýzy, častých dotazů studentů a konzultací se zástupcem fakulty mohli detekovat. Dále postoupíme k analýze možných řešení. Zde se budeme rozhodovat mezi implementací nové aplikace a integrací do již existujících systémů, které jsou na Fakultě elektrotechnické používány studenty při výuce.

Poslední kapitoly se týkají implementace, instalace a nastavení vytvořeného pluginu a testování výsledného řešení. V kapitole implementace popíšeme, jakým způsobem jsme provedli vývoj a nasazení výsledného systému. Testování bylo realizováno formou manuálního testování zaměřeného na funkčnost pluginu a uživatelského testování ověřujícího obsah a funkcionality kurzu v aplikaci Moodle.

Kapitola 2
Vývoj aplikací

V této kapitole jsou popsány nejčastější metodiky pro vývoj software, které se ve světě používají. Dříve, než je popíšeme a vybereme tu nejvhodnější, kterou použijeme pro vývoj aplikace, tak definujeme základní pojmy jako je projekt, projektové řízení, základní projektové role, projektové fáze a životní cyklus software. Jako příklad použijeme projekt úspěšného zakončení studia, na kterém některé definice vysvětlíme. Dokážeme tak, že je možné ho považovat za projekt a aplikovat tak na něj projektové řízení a další metodiky.

2.1 Projekt

Existuje mnoho definic, které nám popisují, co je projekt.

Norma ISO 10006 popisuje projekt jako jedinečný proces sestávající z řady koordinovaných a řízených činností s daty zahájení a ukončení, prováděný pro dosažení cíle, který vyhovuje specifickým požadavkům, včetně omezení daných časem, náklady a zdroji. [2] Dle standardu PMBOK je projekt dočasné úsilí s cílem vytvořit unikátní produkt nebo službu. [2] Další definice nám říká, že projekt je jedinečná a konečná množina aktivit, které jsou potřebné k dosažení určitého cíle. [3] Lze najít i jiné, které popisují projekt jinak, ale ze všech víceméně vyplývá, že projekt je jedinečná sada aktivit, která musí být konečná a díky kterým je za pomoci zdrojů dosaženo předem stanoveného cíle.

Jako projekt bychom mohli definovat i úspěšné zakončení studia. Tento projekt sice musí splnit všichni studenti, kteří chtějí studium úspěšně zakončit, avšak výstupem tohoto projektu je bakalářská práce, která musí být unikátní. Projekt obsahuje řadu aktivit, kterými jsou výběr vedoucího závěrečné práce, dokončení základních předpokladů pro studium jako je například počet kreditů, napsání závěrečné práce a absolvování státních závěrečných zkoušek spolu s obhajobou práce. Za zdroje, které jsou použity pro úspěšné zakončení studia, můžeme považovat čas a znalosti, které student získal nebo získá během psaní práce. Cíl projektu je jasný - je jím úspěšné zakončení studia. Nyní jsme si ukázali, že úspěšné zakončení studia lze považovat za projekt a tudíž na něj mohou být použity metodiky projektového řízení.

2.2 Projektové řízení

Principy projektového řízení se používají pro dosažení požadované kvality projektu v daném čase. Osoba, která projekt řídí a která nese odpovědnost za jeho úspěch či neúspěch, se nazývá projektový manažer, který je popsán v kapitole 2.3.1. V téměř každé firmě, která dbá na projektové řízení, je základní metodika, která je používána pro řízení projektů. Každý projekt je ale jiný a je proto nutné tuto metodiku vzít a upravit ji pro daný projekt. Kvůli tomu dochází k řízení projektů, protože neexistuje obecně platný postup, který by byl aplikovatelný na všechny projekty. Projektové řízení je tedy nalezení optimální cesty k dosažení cíle. [4]

Požadovaný cíl našeho modelového projektu je úspěšné zakončení studia. Pokud by studium bylo ukončeno jinak, než úspěšně, byla by nejspíše chyba v řízení samotného projektu. Čas, který máme na ukončení projektu můžeme uvažovat jako standardní dobu studia. Pokud bychom ale studium prodloužili, nemusí to nutně znamenat neúspěch projektu.Je pak pouze na projektovém manažerovi, aby posoudil, nakolik byl projekt úspěšný.

2.3 Projektové role vývojového týmu dle fází vývoje

Za vývojem každého software se nachází několik skupin lidí, jejichž společným výstupem je finální aplikace. Tu by nebylo možné vytvořit, pokud by nebylo jasně definované, jaké jsou potřeby a požadavky na aplikaci. Zároveň by nebylo bezpečné aplikaci poskytnout k používání, pokud by ji kompetentní osoba nezkontrolovala. Z toho vyplývá, že každý má v projektu svoji roli, které se během vývoje musí držet. Může se stát, že někdo bude mít rolí více. V tom případě by ale měl umět fungovat pouze v rámci role, ve které se v dané chvíli nachází a jedna role by neměla mít vliv na fungování role druhé. [5]

2.3.1 Projektový manažer

Z hlediska fungování projektu je asi nejdůležitější rolí projektový manažer, jehož povinností je dohlížet na chod projektu a hlídat dodržování termínů. Zároveň je zodpovědný za rozdělení zdrojů a využití kapacit. Projektový manažer by měl mít vždy přehled o tom, v jakém stavu se projekt nachází, měl by mít rozhodující slovo při řešení problémů a otázek. Často také sestavuje projektový tým, který se na projektu podílí. [6]

2.3.2 Analytik

Prací analytika je analýza požadavků, potřeb a problémů, které má projekt zohlednit. Mezi analytiky řadíme například byznys analytika, systémového analytika a další. Výstupem jejich práce je analýza, která popisuje daný problém, návrh řešení a postup, jakým má být řešení dosaženo. Pro jednoduchost do této skupiny budeme řadit i UX/UI designéra, který navrhuje vzhled aplikace.

2.3.3 Vývojář

Vývojář vezme analýzu, kterou vytvořil analytik a implementuje podle ní danou aplikaci. Má také za úkol opravit chyby, které vznikly při vývoji aplikace. Mezi vývojáře počítáme i osoby, které nepřímo pracují na implementaci aplikace - například správce sítě, montážní technik a další.

2.3.4 Tester

Tato osoba je zodpovědná za to, že výsledná aplikace nebude obsahovat chyby a bude fungovat tak, jak bylo definováno analýzou. Pokud tester najde chybu, tak ji vrátí zpět buďto analytikovi, nebo vývojáři k opravení.

2.4 Vývojové fáze software

Během vývoje se software nachází vždy v některé z fází vývojového cyklu. Pokud by člověk hledal, jaké fáze existují, našel by jich patrně mnoho. Pro jednoduchost budeme uvažovat pouze 4, které jsou obecné a jelikož jsou skoro stejné jako projektové role, nebudeme jejich význam dále rozvádět. [5] Jedná se o:

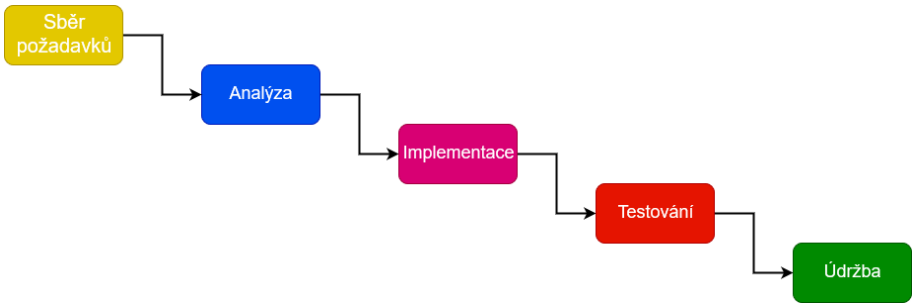
- analýzu,
- implementaci,
- testování,
- údržbu.

2.5 Metodiky vývoje software

Existuje mnoho metodik, které se používají k vývoji software. Zde jsou uvedeny dvě, které jsme vybrali ke zvážení pro vývoj aplikace. První je vodopádový přístup a druhá je metoda SCRUM. Tyto metodiky zanalyzujeme a na základě analýzy vybereme takovou, kterou použijeme pro implementaci aplikace.

2.5.1 Vodopádový přístup

Tento model se řadí mezi sekvenční vývojové procesy a má mnoho výhod i nevýhod, které je potřeba zvážit. Jednotlivé fáze vývoje na sebe navazují a jedna fáze nemůže začít, pokud nebyla ukončena předchozí. Při použití tohoto modelu je potřeba dopředu naplánovat jednotlivé aktivity s definovaným začátkem a koncem. První dvě aktivity obrázku 2.1 bychom mohli určit jako analýzu z kapitoly 2.4 a zbytek bodů odpovídá jejich názvům. [7–8]



Obrázek 2.1. Diagram popisující vodopádový přístup vývoje software. [8]

Výhody:

- Jedna činnost začíná ve chvíli, kdy skončila předchozí.
- Lze na začátku naplánovat termíny jednotlivých úkolů.
- V průběhu se lépe odhaduje, v jaké fázi se projekt nachází.

Nevýhody:

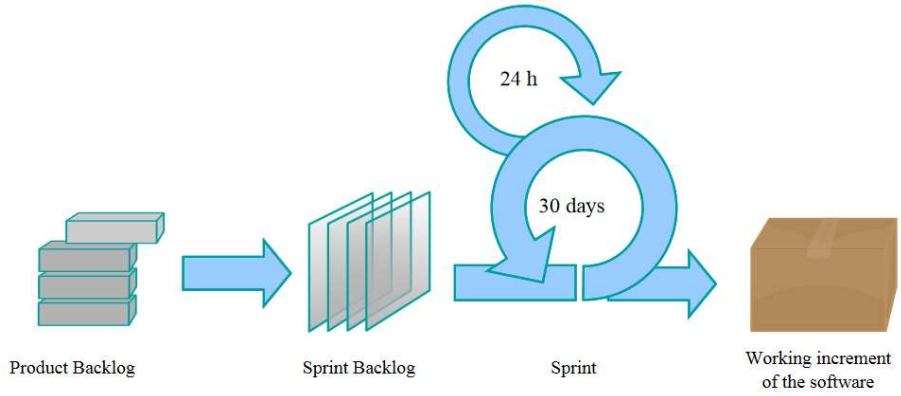
- Nutnost dodržet termíny kvůli předem stanovené časové ose.
- Špatná reakce na změny v zadání během realizace.

2.5.2 Metoda SCRUM

Metodu SCRUM řadíme mezi agilní iterativní metodiky. Agilní metodika znamená, že jsou zákazníkovi pravidelně posílány výstupy vývoje a on tak může měnit požadavky a díky tomu je možné rychleji reagovat na změny. Iterativní znamená, že dochází k vývoji v cyklech, které mají pevnou délku, obvykle dva až čtyři týdny. Projektové týmy by měly být malé s počtem členů od čtyř do deseti lidí, jelikož podstatou této metodiky jsou časté schůzky, kde se řeší, co bylo během cyklu uděláno, dochází k akceptaci výstupů a určuje se práce na další cyklus. Během cyklu může docházet k pravidelným

kontrolám, kolik práce bylo uděláno, kolik jí zbývá a jestli není některý z týmů blokován jiným týmem. Oproti jiným agilním metodám je u této možné určit, kolik požadavků zbývá k dokončení projektu. Požadavky jsou shromažďovány v backlogu, což je forma evidování důležitých informací pro daný projekt, a zde slouží jako jakýsi pomyslný zásobník požadavků. [9]

V metodice SCRUM se vyskytují další role, které se na vývoji podílí. Jedna z nových rolí je SCRUM master, který má podobnou roli a funkce, jako projektový manažer z kapitoly 2.3.1. Další rolí je takzvaný product owner, který má zásadní roli v rozhodování, které požadavky budou implementovány a kontroluje kvalitu předaných výstupů. Poslední rolí jsou stakeholdeři, kam patří lidé, kteří projekt financují, budoucí uživatelé, vedení firmy a další. [10]



Obrázek 2.2. Diagram popisující metodiku SCRUM. [11]

Výhody:

- Rychlá reakce na změny v zadání.
- Výstupy jsou průběžně kontrolovány a odsouhlaseny zákazníkem.

Nevýhody:

- Častá potřeba komunikovat s lidmi v týmu.
- Je nutné na začátku vytvořit podrobnou analýzu.
- Je náročné odhadnou pracnost před začátkem projektu.

2.5.3 Výběr metodiky pro tuto práci

Dle webu medium.com je úspěšnost implementace software pomocí vodopádového modelu 49 %, zatímco u agilních metodik je úspěšnost 64 %. Díky vyšší úspěšnosti jsou agilní metodiky více používané, než sekvenční vodopádový přístup. Dle webu managementmania.com je vodopádový model použitelný pro případ, kdy je přesně definovaný cíl a termín projektu. [8, 7]

Po zvážení všech možností jsme zvolili vodopádový přístup vývoje software, jelikož výsledná aplikace bude odevzdána na konci a není možné ji odevzdávat průběžně.

2.6 Závěr

Pro implementaci jsme zvolili vodopádový model vývoje software, který se jeví jako optimální. Zároveň jsme ukázali, že úspěšné zakončení studia je možné považovat za

projekt a uplatnit na něj tak metodiky projektového řízení. Můžeme tedy definovat jedny z prvních požadavků na systém, který dále rozvedeme v kapitole 4.4. Těmito požadavky je mít možnost vytvářet jednotlivé úkoly, měnit stav úkolů, přiřazovat jim prioritu a termín dokončení.

Kapitola 3
Analýza studia v posledním ročníku

Jak již bylo zmíněno v úvodu, tato práce se zabývá pouze bakalářským a magisterským studiem. Doktorské studium není součástí této práce a tudíž nebude analyzováno. Každý student, který se dostal do posledního ročníku, musel řešit, jak studium úspěšně dokončit. Nejprve se podíváme na předměty posledního ročníku, které musí student absolvovat a následně na to, kdy se student dostane do posledního ročníku a je vhodné u něj o možném úspěšném konci studia uvažovat.

3.1 Semestrální projekt

Z hlediska procesu dokončení studia se jedná o první předmět, který student musí absolvovat. Jedná se o nutnou prerekvizitu předmětu Závěrečná práce definovaném v kapitole 3.2. Může dojít k výjimce a student může mít tento předmět zapsaný souběžně se závěrečnou prací.

Cílem předmětu je naučit studenta pracovat samostatně, vyhledávat si potřebné informace a sepsat je do uceleného textu. Dalo by se tedy říci, že se jedná o přípravu na psaní závěrečné práce. Díky práci na projektu se může student zamyslet, zda ho dané téma opravdu zajímá a je pro závěrečnou práci vhodné a dostatečně obsáhlé.

Úkoly, které musí student splnit a jejich vhodné pořadí:

1. Výběr vedoucího a tématu,
2. schválení tématu,
3. vypracování projektu,
4. odevzdání projektu,
5. vypracování prezentace,
6. prezentace a obhajoba projektu.

Další úkol, který bychom mohli také přidat do seznamu, je vyhledání zdrojů a literatury. Na rozdíl od ostatních úkolů je tento krok volitelný, proto ho do seznamu přidávat nebudeme.

Seznam všech předmětů Semestrální projekt, které jsou v současné době vyučovány na Fakultě elektrotechnické:

Kódy předmětů			
B1BPROJ4	AD1B13IND	AD1B14IND	AD0M14DIP
AD1B15IND	B2BPROJ6	B2BPROJ4	BD5B99IN2 P
A2B17IN2	A2B31IN2	A2B32IND	A2B34IN2
A2B37IN2	B3BPROJ4	A3B33IND	A3B35IND
A3B38IND	BBPROJ4	B4BPROJ6	A4B31SVP
A4B33SVP	A4B35SVP	A4B36SVP	A4B38SVP
A4B39SVP	A8BPROJ2	B6B36PRO	BD6B36PRO
A7B36PRO	A7B39PRO	AD7B16PRO	AD7B17PRO
AD7B33PRO	AD7B35PRO	AD7B36PRO	AD7B38PRO
AD7B39PRO	A7B36PRO	A7B39PRO	B1MPROJ
B1M16IND	A1M16IND	AD1M16IND	B1M14IND
A1M14TP1	A1M14IND	B1M15IND	A1M15IND
AD1M15IND	A1M13TP1	A1M13IND	B1M13IND
B2MPROJ6	B3MPROJ8	A3M33IND	A3M35IND
A3M38IND	BMPROJ6	B4MSVP	A4M31SVP
A4M33SVP	A4M35SVP	A4M36SVP	A4M38SVP
A4M39SVP	125PIB2	2163034	A5M99PR2
A6M31IP	A6M33IP	2163034	A5M99PR2

Tabulka 3.1. Tabulka jednotlivých předmětů zabývajících se předmětem Semestrální projekt. [12]

Tyto studijní programy nemají ve studijním plánu předmět Semestrální projekt:

- Letectví a kosmonautika,
- Otevřené elektronické systémy,
- Otevřená informatika - obor Počítačové systémy,
- Otevřená informatika - obor Softwarové systémy.

Informace k dokončení studia tak budou potřebovat až v posledním semestru, kdy mají zapsaný předmět Závěrečná práce. Při návrhu řešení bude potřeba definovat podmínku tak, aby jim byl před posledním semestrem umožněn vstup do kurzu.

Procesní diagram znázorňující proces průběhu předmětu Semestrální projekt je součástí Přílohy A.

3.2 Závěrečná práce

Dle typu studia si student v posledním semestru studia, ve kterém chce studium dokončit, zapisuje předmět bakalářská nebo diplomová práce. Pokud by tento předmět neměl zapsaný, tak nemůže být připuštěn ke SZZ. Podobně jako u Semestrálního projektu popsaného v kapitole 3.1 je potřeba projít specifické kroky definované procesem Vypracování a odevzdání závěrečné práce z Přílohy C spolu s jeho podprocesy. Cílem předmětu je vypracovat a obhájit závěrečnou práci, jejíž téma může vycházet ze semestrálního projektu, nebo může být úplně jiné. [13]

Úkoly, které vyplývají ze zmíněných procesů:

1. Výběr vedoucího a tématu
   - Dle procesu výběru vedoucího práce z Přílohy B by mělo dojít nejprve k výběru vedoucího a poté tématu, ale je možné mít již vybrané téma a hledat vedoucího, který je ochoten práci vést.

2. Sestavení a přihlášení tématu práce v aplikaci KOS
3. Schválení práce studijním oddělením nebo vedoucím příslušné katedry
4. Vypracování závěrečné práce
5. Odevzdání elektronické verze práce
6. Odevzdání papírové verze práce
7. Udělení zápočtu vedoucím
8. Vypracování oponentského posudku oponentem
9. Vypracování prezentace na obhajobu
10. Obhajoba

Tabulka všech předmětů Závěrečná práce, které jsou v současné době akreditovány a vyučovány na Fakultě elektrotechnické:

Kódy předmětů			
BBAP15	ABAP9	BDIP25	AD0M14DIP
BBAP16	BBAP20	ADIP25	AD0M15DIP
ABAP20	A0B01BAP	A0M13DIP	AD0M16DIP
A0B13BAP	A8B01BAP	A0M14DIP	AD0M17DIP
A0B14BAP	A8B14BAP	A0M15DIP	AD0M31DIP
A0B15BAP	A8B15BAP	A0M16DIP	AD0M32DIP
A0B16BAP	A8B16BAP	A0M17DIP	AD0M33DIP
A0B17BAP	A8B17BAP	A0M31DIP	AD0M34DIP
A0B31BAP	A8B31BAP	A0M32DIP	AD0M35DIP
A0B32BAP	A8B32BAP	A0M33DIP	AD0M36DIP
A0B33BAP	A8B33BAP	A0M34DIP	AD0M37DIP
A0B34BAP	A8B34BAP	A0M35DIP	AD0M38DIP
A0B35BAP	A8B35BAP	A0M36DIP	AD0M39DIP
A0B36BAP	A8B36BAP	A0M37DIP	BDIP30
A0B37BAP	A8B37BAP	A0M38DIP	ADIP26
A0B38BAP	A8B38BAP	A0M39DIP	A5M99DIP
A0B39BAP	A8B39BAP	AD0M13DIP	

Tabulka 3.2. Tabulka jednotlivých předmětů zabývajících se Závěrečnou prací. [12]

Procesní diagram průběhu předmětu Závěrečná práce je součástí Přílohy D.

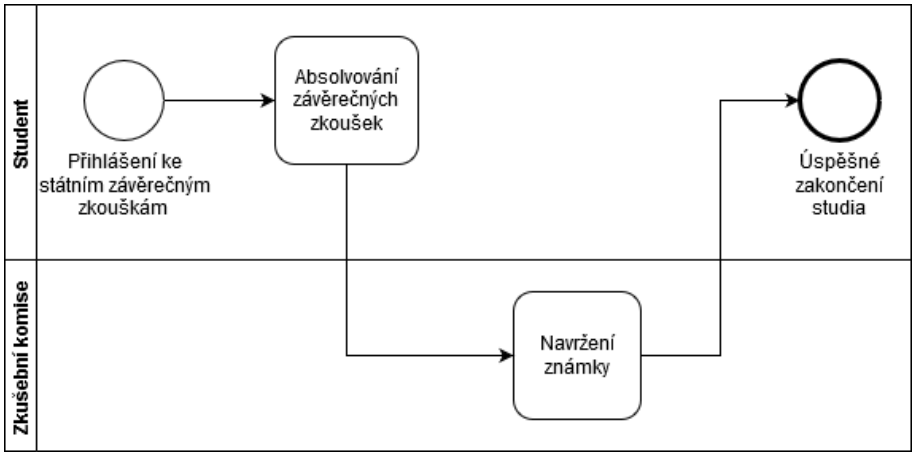
3.3 Státní závěrečné zkoušky

Poslední fází, kterou student musí projít, aby mohl studium úspěšně zakončit, je SZZ. Ke zkoušce se student přihlašuje přes aplikaci KOS. Před zkouškou musí dojít ke splnění a uzavření studijního plánu. Pokud student plán nesplní, je mu zkouška přeložena do nejbližšího následujícího termínu SZZ. **Podmínky pro splnění studijního plánu jsou:**

1. Získání určeného počtu kreditů dle typu studia
2. Složení zkoušky z anglického jazyka úrovně B2
3. Úspěšné absolvování předmětu Semestrální projekt
4. Úspěšné absolvování předmětu Závěrečná práce
5. Podmínky definované studijním plánem daného programu fakulty
6. Uzavření studijního plánu 3 pracovní dny před začátkem SZZ

Úkoly, které musí student vykonat:

1. Podání přihlášky k SZZ
2. Splnění studijního plánu dle výše uvedených podmínek
3. Složení SZZ



Obrázek 3.1. Diagram popisující proces dokončení studia.

3.4 Vstup do závěrečné fáze studia

Co se myslí termínem závěrečná fáze studia nelze obecně definovat. Pro tuto práci však můžeme však říct, že se jedná o období posledního roku dle předmětů obsažených ve studijním plánu. První, na co je potřeba se zaměřit je, kdy nastává ideální doba, kdy by měl student dokončení studia řešit. Jako rozumný milník se tu může jevit doba, kdy student vstupuje do posledního ročníku. Avšak v tu chvíli by již měl mít vybraného vedoucího a téma práce. Proto musíme tento milník posunout na **předběžné zápisy**, kdy si studenti zapisují předměty a je možné na jejich základě detekovat, že se chystají v následujícím ročníku studium ukončit.

Dále musíme vyřešit, jak vypadá student, kterého chceme do aplikace zahrnout. Definujeme proto tyto základní typy studentů:

- Klasický student
- Student, překračující klasickou dobu studia
- Opakující nebo přecházející studenti
  - Student, který znovu nastoupil do 1. ročníku a nechá si uznat kredity
  - Student, který znovu nastoupil a nemá uznané kredity z minulého studia
  - Student, který přestoupil v rámci programu nebo školy
- Zahraniční student
- Student studující více programů najednou

V následujících podkapitolách definujeme jednotlivé studenty a určíme podmínky, které student splňuje, když vstupuje do posledního ročníku studia. V **poslední podkapitole tyto podmínky spojíme a na jejich základě definujeme obecné podmínky**, které studenti splňují v předběžných zápisech, které jsme zvolili jako milník, kdy student začíná řešit dokončení studia.

Student, který opakuje studium a nemá uznané žádné předměty je stejný, jako řádný student, jelikož musí splnit studijní plán od začátku. **Studenty s individuálním studijním plánem a zahraniční studenty** můžeme považovat za výjimky, jichž jsou na škole řádově jednotky. Z toho důvodu je **v analýze nebudeme uvažovat**. Podmínky by měly

být definované tak, aby je systém automaticky detekoval, případně bude jejich zapsání řešeno manuálně.

3.4.1 Klasický student

Za klasického studenta považujeme takového, který **plní studijní plán nebo opakuje malý počet předmětů**. Student tak nemusí mít přesně 120 kreditů pro bakalářské studium a 60 pro magisterské studium, ale celkový počet by se měl pohybovat okolo této hranice. Pokud má kreditů výrazně méně, je jeho ukončení studia v následujícím ročníku velmi nepravděpodobné. Zároveň školní řád povoluje zápis maximálně 40 kreditů za semestr. Studenti si mohou zapsat kreditů více, ale musí podat žádost ke studijnímu proděkanovi, který musí vyšší počet kredit schválit.

Jednou z dalších důležitých podmínek je, že vstupuje do 3. ročníku v případě bakalářského a 2. ročníku v případě magisterského studia. Pokud se student nachází v jiném, než 3. nebo 2. ročníku studia a má výše definovaný počet kreditů, tak se nemůže jednat o řádného studenta. Tyto studenty definujeme a rozebereme v následujících kapitolách.

Poslední podmínkou, která nám u takového studenta detekuje, že chce v následujícím ročníku studium úspěšně dokončit, je zápis předmětu Semestrální projekt. Tento předmět dle je nutnou prerekvizitou, která musí předcházet předmětu Závěrečná práce. Mohou existovat výjimky, kdy má student oba předměty zapsány současně, ale opět by se tak nejednalo o klasického studenta.

Souhrn podmínek:

- Zisk okolo 120 kreditů v případě bakalářského a 60 v případě magisterského studia.
- Vstupuje se ve 3. ročníku bakalářského nebo 2. magisterského studia.
- Má zapsaný předmět Semestrální projekt.

3.4.2 Student, překračující klasickou dobu studia

Student může délku studia překročit z mnoha důvodů. My ale do této kategorie budeme počítat pouze ty, kteří překračují standardní dobu studia v rámci jednoho studijního období. **Nejsou to studenti, kteří znovu nastoupili ke studiu**, ale jsou to tací, kteří **nemají dostatečný počet kreditů** , aby mohli studium ukončit v řádné studijní době. Dále se jedná o studenty, kteří vyjeli na studijní pobyt Erasmus. Takový student se tak nachází ve vyšším, než 3. ročníku bakalářského a 2. ročníku magisterského studia.

Souhrn podmínek:

- Získal okolo 120 kreditů v případě bakalářského a 60 v případě magisterského studia.
- Má zapsaný předmět Semestrální projekt.
- Nachází se ve vyšším než 3. ročníku bakalářského a 2. ročníku magisterského studia.

3.4.3 Opakující nebo přestupující studenti

Do této kategorie patří všichni studenti, kteří již za sebou mají alespoň 1 neúspěšné studium nebo se rozhodli pro změnu programu nebo školy. Mohou mít uznány některé předměty z předchozích studií nebo nastupují znovu do 1. ročníku a musí splnit celý studijní plán, takové ale budeme počítat jako klasické studenty. U těchto lidí bude pak záležet na tom, kolik předmětů z předchozího studia jim bylo uznáno a kolik kreditů jim zbývá do dokončení studia. **Z hlediska analýzy se jedná o nejsložitější množinu lidí, protože počtem kreditů můžou spadat do 1., 2. nebo 3. ročníku a určujícím bodem u nich tak bude počet kreditů nebo zápis konkrétního předmětu**.

Tím jsme uvedli jednu z podmínek, jak je možné tyto studenty identifikovat. Další z nich je zápis jednoho z předmětů týkajících se Semestrálního projektu nebo Závěrečné práce. Může se totiž jednat o studenta, kterému chybí k úspěšnému dokončení pouze vypracovat, odevzdat a obhájit závěrečnou práci a složit SZZ. Například se jedná o studenta, který dvakrát neudělal SZZ a zbytek předmětů mu bylo možné uznat.

Souhrn podmínek:

- Získal okolo 120 kreditů v případě bakalářského a 60 v případě magisterského.
- Má zapsaný předmět Semestrální projekt.
- Má zapsaný předmět Závěrečná práce.

3.4.4 Studenti studující více programů najednou

Tito studenti jsou naprosté výjimky ve studijním plánu. Z hlediska analýzy není možné je detekovat, jelikož počet dosažených kreditů může být během studia o dost vyšší, než u klasického studijního plánu a je velice pravděpodobné, že budou studovat delší, než klasickou dobu definovanou školním řádem. Z toho důvodu je do analýzy zahrnovat nebudeme. Úspěšné dokončení studia budou tito studenti muset řešit v každém případě a tyto informace budou muset hledat, z toho důvodu není problém je do aplikace automaticky nahrát.

3.4.5 Definice podmínek pro detekci vstupu do poslední fáze studia

Již jsme definovali obecné podmínky, které musí student splnit, pokud chce v následujícím roce studium úspěšně dokončit. Tyto podmínky však splňuje až při vstupu do poslední ročníku. Z hlediska analýzy je potřeba tohoto uživatele detekovat ještě dříve a informovat ho tom, že může v následujícím roce studium dokončit a poskytnout mu návod, jak je toho možné dosáhnout.

Pokud podmínky sloučíme, získáme tento seznam podmínek:

- Má **120 kreditů** v případě bakalářského a **60 kreditů** v případě magisterského studia.
- Má zapsaný předmět **Semestrální projekt**.
- Má zapsaný předmět **Závěrečná práce**.
- Nachází se ve **3. a vyšším ročníku** bakalářského nebo **2. ročníku** magisterského studia.

Tyto podmínky musíme upravit, aby je student splnil v námi definovaném termínu. Kredity musíme snížit, jelikož předběžné zápisy se konají přibližně v polovině semestru a **uživatelé budou mít k dispozici kredity pouze za 3 semestry v případě bakalářského a 1 semestr v případě magisterského studia**. Minimální počet kreditů pro postup do 2. ročníku je 30 kreditů a pro postup ze třetího semestru potřebuje student 20 kreditů. To dává v součtu 50 kreditů, avšak lze těžko předpokládat, že student s 50 kredity dokončí studium v řádné době, protože mu zbývá 130 kreditů na následující 3 semestry a i když je možné žádat o výjimku a zapsat si více než 40 kreditů v jednom semestru, tak časová náročnost je taková, že nelze předpokládat dokončení v řádné době. Z toho důvodu snížíme hodnotu kreditů na **75**. V případě magisterského definujeme počet kreditů na 20, jelikož pro postup z prvního semestru potřebuje student 15 kreditů, ale to je minimální počet a student, který dosáhne minimálního počtu, tak málokdy dokončí studium v řádné době.

Dále musíme snížit i ročník, ve kterém se student nachází. Bylo by navíc dobré tuto hranici zdola omezit, jelikož se jedná o spodní hranici, kdy může student o dokončení studia uvažovat. Bohužel v **případě opakujících se studentů nelze ročník brát v úvahu**, protože se mohou nacházet v prvním ročníku, ale studium dokončit v následujícím

roce. Detekce takovýchto studentů není pomocí systému možná. Z toho důvodu hodnotu **snížíme na 1. ročník a uvedeme ji pouze pro pořádnost**, z hlediska aplikace není potřeba tuto podmínku kontrolovat, protože každý student se může nacházet v 1. ročníku. Pouze zápisy daných předmětů zde necháme.

Výsledné podmínky tedy jsou:

1. Má **75 kreditů** v případě bakalářského a **20 kreditů** v případě magisterského studia.
2. Má zapsaný předmět **Semestrální projekt** nebo **Závěrečná práce**.
3. Nachází se v **1. a vyšším ročníku bakalářského a 1. ročníku magisterského studia**.

Podmínky 1 a 2 musí platit současně. Bod 3 je pro bakalářské studium natolik obecný, že jej nelze použít. Proto jsme jako podmínky pro detekci příslušného studenta vybrali ty, které jsou uvedeny v bodech 1 a 2 v seznamu výsledných podmínek. Zápis předmětů Semestrální projekt a Závěrečná práce probíhá automaticky dle studijního plánu a studentovi tak nemusí být připomínáno, aby si tyto předměty zapsal.

3.5 Závěr

V této kapitole jsme prošli náležitosti, které student musí pro úspěšné zakončení studia splnit. Definovali jsme kritéria, pomocí kterých takového studenta identifikujeme. Vypsali jsme jednotlivé předměty, které se v současné době na fakultě vyučují a jsou součástí akreditovaných programů. Vytvořili jsme tak základ pro analýzu aplikace.

Kapitola 4
Požadavky na aplikaci

V této kapitole budeme vycházet především z předpokladu, že zakončení studia je projektem každého studenta a ze závěrů Kapitoly 3 o analýze posledního ročníku studia, kde jsme definovali stav, kdy je možné u studenta detekovat, že má v úmyslu studium dokončit. Jako výstup diskuze se studenty zde uvedeme problémy, se kterými se potýkají během procesu dokončení studia. Popíšeme současný a budoucí stav a postup, jakým se chceme k budoucímu stavu dostat. Nakonec definujeme požadavky a případy užití, které budou sloužit jako základ pro následující vývoj.

4.1 Problémy studentů se zakončením studia

Na základě zpětné vazby a dotazů na studenty jsme definovali tyto problémy. Některé z těchto problému použijeme v následující podkapitole 4.4, kde už budou sepsány jednotlivé požadavky z nich vycházející, které má aplikace splňovat.

Základní informace, které studenti nevědí:

- Co je v posledním ročníku čeká a kdy.
- V čem a jak se píše semestrální projekt.
- Jaké aplikace lze pro psaní použít.
- Jak souvisí semestrální projekt s bakalářskou prací.
- Jaký má být rozsah semestrálního projektu.
- Jak má vypadat prezentace.
- Zda musí bakalářská práce souviset se semestrálním projektem.
- Jaké jsou termíny pro výběr tématu prací.
- Kdy a jak se mají přihlásit k závěrečným zkouškám.

Tyto problémy mají studenti zejména proto, že **neexistuje jednotný zdroj**, kde by bylo možné informace najít. V tom by jim pak měli pomoci zejména vedoucí semestrálních projektů a závěrečných prací. Bohužel ani ti občas tyto informace nemají a student si tak musí vše dohledat sám. Informace o dokončení studia jsou k dohledání na stránkách fakulty [14] nebo na Procesním portále [15] , kde jsou jednotlivé činnosti znázorněné pomocí procesů. Případně jim tyto informace poskytují pomocí emailů referentky studijního oddělení, zástupci katedry, nebo vedoucí předmětů.

Bohužel nejsou tyto zdroje jednotné a informace, které spolu souvisí, jsou k dispozici na různých místech. **Studenti tak jsou zmatení** a neví, co mají dělat a co a kdy je přesně čeká.

4.2 Aktuální stav

V současné době jsou informace o semestrálním projektu, závěrečné práci a státních závěrečných zkouškách dostupné na různých místech na stránkách fakulty a nebo posílány mailem studentům. Tento stav je špatný zejména proto, že **webové stránky fakulty jsou nepřehledné** a navíc nejsou všechny informace na jednom místě. Vedoucí předmětů,

semestrálních projektů nebo závěrečných prací musí trávit mnoho času odpovídáním na dotazy studentů. **Studenti jsou na začátku posledního ročníku zmatení a neví, co je čeká.**

4.3 Budoucí stav

Bude vytvořena aplikace, která informace sjednotí na jedno místo, kde studenti naleznou vše o tom, co je v posledním ročníku studia čeká a dá jim návod, jak mají postupovat. Díky tomu bude možné vést jejich kroky k úspěšnému dokončení studia. Nahrání studentů do aplikace proběhne automaticky. Současně se jim vytvoří příslušné úkoly s termíny splnění. Bude možné odevzdávat průběžná řešení jednotlivých úkolů. Aplikace bude splňovat všechny požadavky definované v následující kapitole.

4.4 Požadavky

Zde je vyspán seznam požadavků na aplikaci, které jsme definovali na základě Kapitoly 4.1 a Kapitoly 3. Požadavky byly sestaveny na základě identifikovaných problémů a konzultace se zástupcem fakulty.

Id	Název
1	Vytvoření projektu
2	Vytvoření podprojektů
3	Vytvoření úkolů
4	Správa úkolů
5	Přehled termínů
6	Zobrazení termínů
7	Evidence změn
8	Nahrání posledních verzí
9	Nahrání uživatelů
10	Zobrazení dat
11	Správa projektu

Tabulka 4.1. Jednotlivé požadavky na aplikaci.

Požadavky s jejich popisem jsou součástí přílohy E.

4.5 Případy užití

Na základě požadavků můžeme definovat jednotlivé případy užití (anglicky use cases, zkráceně UC). U jednotlivých UC evidujeme jejich ID, název, popis, aktéra, cíl a požadavek, který daný UC plní. Pro přehlednost zde uvedeme pouze tabulku s ID a názvem, kompletní seznam UC je pak jako součást přílohy I.

Id	Název
1	Vytvoření projektu
2	Vytvoření popisu projektu
3	Vytvoření podprojektu Semestrální projekt
4	Vytvoření podprojektu Závěrečná práce
5	Vytvoření podprojektu Státní závěrečné zkoušky
6	Vytvoření popisu podprojektům
7	Vytvoření úkolů Semestrálního projektu
8	Vytvoření úkolů Závěrečné práce
9	Vytvoření úkolů Státní závěrečné zkoušky
10	Zobrazení úkolů
11	Smazání úkolu
12	Vytvoření úkolu
13	Editace úkolu
14	Stažení termínů
15	Zobrazení termínů
16	Evidence změn
17	Návrat změn
18	Nahrání projektu
19	Uložení projektu
20	Nahrání uživatelů
21	Zobrazení projektu
22	Změna dat projektu

Tabulka 4.2. Všechny UC pro budoucí aplikaci.

Aktéři jednotlivých UC jsou **Uživatel**, **Administrátor** a **Systém**. Uživatel bude vždy přihlášený. Nepřihlášeného uživatele nebudeme v rámci aplikace vůbec uvažovat. Administrátor má vyšší práva než klasický uživatel. Mapování je k dispozici v rámci přílohy **F**. Neuvedli jsme nefunkční požadavky, jelikož by jejich počet byl malý a pro naší aplikaci nejsou prioritní.

4.6 Závěr

V této kapitole jsme definovali jednotlivé požadavky, které má výsledný systém splňovat. Z těchto požadavků budeme vycházet zejména při výběru řešení v analýze systémů a v Kapitole 6 o implementaci.

Kapitola 5
Analýza systémů na ČVUT FEL

Předtím, než se podíváme na implementaci řešení našeho problému, tak musíme zvážit využití již existujících systémů, které jsou na fakultě používány a samozřejmě i vytvoření systémů nového. Při hledání řešení se zaměříme na to, aby vybraný systém splňoval největší množství definovaných požadavků z kapitoly 4.4, správu systému v provozu a jeho využívanost cílovými uživateli, tedy studenty.

Ze školních systémů jsme k analýze vybraly systémy Moodle, CourseWare, GitLab, FELSight a Témata.

5.1 GitLab

Jako první se zaměříme na fakultní systém GitLab. Jedná se o verzovací systém, který je využíván k vývoji projektů po celém světě. Na fakultě ho mohou využívat studenti i vyučující. Systém umožňuje vytvoření jednotlivých projektů jednotlivým uživatelům. Do nich je možné vytvářet další adresáře a soubory, tím pádem by uživatelé mohli nahrávat svá vlastní řešení. Lze také upravovat role uživatelů a sledovat historii. Dále systém umožňuje management projektu, tedy vytváření, mazání, úpravu a zobrazení jednotlivých projektů.

Splňuje tak všechny požadavky, které jsme definovali v kapitole 4.4 a je to vhodný kandidát, kterého bychom mohli pro náš systém použít. GitLab však funguje pouze pro verzování systému, není možné do něj umístit modul nebo aplikaci, která by fungovala samostatně. To bychom mohli vyřešit tak, že bychom naši aplikaci spustili samostatně a používali ji pouze pro nahrávání projektů a přidávání uživatelů k projektu. Studenti by pak pracovali pouze se systémem GitLab.

5.2 Moodle

Moodle je framework sloužící pro podporu výuky. Fakulta ho využívá v rámci open GNU licence a celá aplikace včetně obsahu běží na fakultních serverech. Do systému přistupují vyučující, kteří zde spravují kurzy pro výuku svých předmětů. Moodle nabízí celou škálu funkcí a splňuje téměř všechny požadavky, které jsme definovali v kapitole 4.4.

Z hlediska funkčnosti a možností se jedná nejspíš o nejlepšího kandidáta, jelikož pokrývá všechny důležité požadavky, které jsme definovali. I z hlediska správy a využívanosti je na tom Moodle dobře, protože ho používá většina vyučujících a studentů.

5.3 CourseWare

Jedná se o systém spravovaný a provozovaný fakultou. Funguje podobně jako systém Moodle a funkce mají stejné. Využívají ho jak studenti tak vyučující. Dle dostupných informací není možné do CourseWare implementovat další pluginy. Společně s Moodlem se však jedná o nejlepšího kandidáta, kam bychom mohli implementovat naší aplikaci.

5.4 FELSight

FELSight je aplikace vyvíjená Centrem znalostního managementu, kterou používají studenti a učitelé pro zobrazování rozvrhu, správu událostí a projektů. Největší předností aplikace je zobrazení rozvrhu, termínů událostí a úkolů. Hlavní nedostatky aplikace FELSight z hlediska našich požadavků jsou, že neumožňuje vytvářet editovatelné seznamy úkolů a nahrávat vypracovaná řešení. Všechna data by navíc musela být v projektu jako text, a z toho důvodu FelSight nesplňuje další požadavek, kterým je správa úkolů u projektu.

5.5 Témata

Aplikace slouží učitelům pro zadávání témat a zadání semestrálních projektů. Zároveň slouží studentům pro přihlášení k tématům. V aplikaci je možné vyhledávat témata podle zadaných parametrů. Bohužel aplikace nedisponuje žádnou z funkcionalit, které pro aplikaci požadujeme. Z toho důvodu je aplikace velmi nevhodný kandidát pro implementaci našeho řešení.

5.6 Nová aplikace

Na závěr zvážíme možnost vytvoření nové aplikace. Z hlediska požadavků z Kapitoly 4.4 by se jednalo o nejoptimálnější řešení. Mohli bychom totiž od základu vybrat ideální technologie a postavit aplikaci takzvaně na zelené louce.

Bohužel z hlediska využívání aplikace by se jednalo o velmi špatnou cestu, jelikož fakulta v současné době disponuje desítkami systémů a aplikací a přidávat další aplikaci, kterou by uživatelé museli používat, je špatný nápad. Počet uživatelů, kteří by reálně aplikaci používali by tak byl nižší. Z toho důvodu je lepší zvolit již existující systém.

5.7 Výběr řešení

Na základě jednotlivých analýz a tabulky v Obrázku 5.1 bychom jako nejlepšího kandidáta zvolili systém GitLab. GitLab je využíván zejména programátory a na mnoha studijních programech by s jeho použitím neměli studenti problém. Systém je však potřeba napsat tak, aby byl použitelný na celé fakultě a proto musíme zvolit jiný systém, se kterým jsou studenti více seznámeni. Zároveň implementace aplikace do systému GitLab má svá úskalí a bude jednodušší zvolit jiný systém.

Proto jsme vybrali vybrali systém **Moodle** z podkapitoly 5.2, jelikož sám o sobě pokrývá všechny požadavky. Implementace bude realizována pomocí pluginu, který bude možné do fakultního systému Moodle nainstalovat.

Vzhledem k tomu, že nebudeme implementovat novou aplikaci, ale vydáme se cestou integrace do již fungující aplikace, tak nemusíme dělat kompletní rešerši technologií a budeme používat technologie, které využívá systém Moodle. Vyhneme se tak náročnému nastavování technologií a celého systému.

5.8 Závěr

V této kapitole jsme prošli používané systémy na FEL ČVUT, se kterými se mohou studenti setkat. Z těchto systémů jsme vybrali ten nejvhodnější pro implementaci našeho

ID	Název	Moodle	CourseWare	FELSight	Témata	GitLab
HR01	Vytvoření projektu	Ano	Ano	Ano	Ano	Ano
HR02	Vytvoření podprojektů			Ne	Ne	Ano
HR03	Vytvoření úkolů	Ano	Ano		Ne	Ano
HR04	Správa úkolů	Ano	Ano		Ne	Ano
HR05	Přehled termínů	Ano	Ano	Ano	Ne	Ano
HR06	Zobrazení termínů	Ano	Ano	Ano	Ne	Ano
HR07	Evidence změn	Ano	Ano	Ne	Ne	Ano
HR08	Nahrání posledních verzí	Ano		Ne	Ne	Ano
HR09	Nahrání uživatelů	Ano	Ano	Ano	Ne	Ano
HR10	Zobrazení dat	Ano	Ano	Ano	Ne	Ano
HR11	Správa projektu	Ano	Ano	Ano	Ne	Ano

Obrázek 5.1. Tabulka popisující, jak jednotlivé systémy splňují námi definované požadavky. Zelená značí, že plně splňuje, žlutá částečně a červená vůbec.

řešení. Systém budeme implementovat ve formě pluginu pro aplikaci Moodle. Vybrané řešení jsme porovnali i s vytvořením nové aplikace a zdůvodnili, proč je vytvoření nového systému na fakultě nevýhodné.

Kapitola 6
Implementace

V této kapitole popíšeme, jakým způsobem probíhala implementace našeho řešení. Jako řešení jsme vybrali vytvoření pluginu pro aplikaci Moodle, tento výběr je odůvodněný v Kapitole5. V následujících sekcích tedy popíšeme systém Moodle, použitá Moodle API, komunikaci s jednotlivými školními API, která jsme v rámci práce implementovali a databázový model. Jako součást implementace vznikl kurz, který je ve formátu MBZ a je součástí přiloženého CD.

6.1 Moodle

V této části popíšeme obecné fungování výukové platformy Moodle a nastíníme implementaci na FEL. Na ČVUT je provozován také rektorátní systém Moodle, toho se ale naše implementace netýká.

Systém Moodle je open source výuková platforma, která je používána po celém světě. Dle dokumentace se jedná o flexibilní, plně nastavitelný, robustní a bezpečný systém. Aplikace využívá web server s databází, pro emailovou komunikaci potřebuje přístup k SMTP serveru. Aplikace je multiplatformní a je možné jí užívat na mobilních zařízeních i počítačích. Programovací jazyk, ve kterém je aplikace napsána, je jazyk PHP, který je potřeba mít na serveru nainstalovaný. Součástí aplikace je i takzvaný CRON skript, který má za úkol spouštět v požadovaném čase vytvořené skripty, které mohou sloužit například k zapsání studentů do kurzů nebo dalším aktivitám.

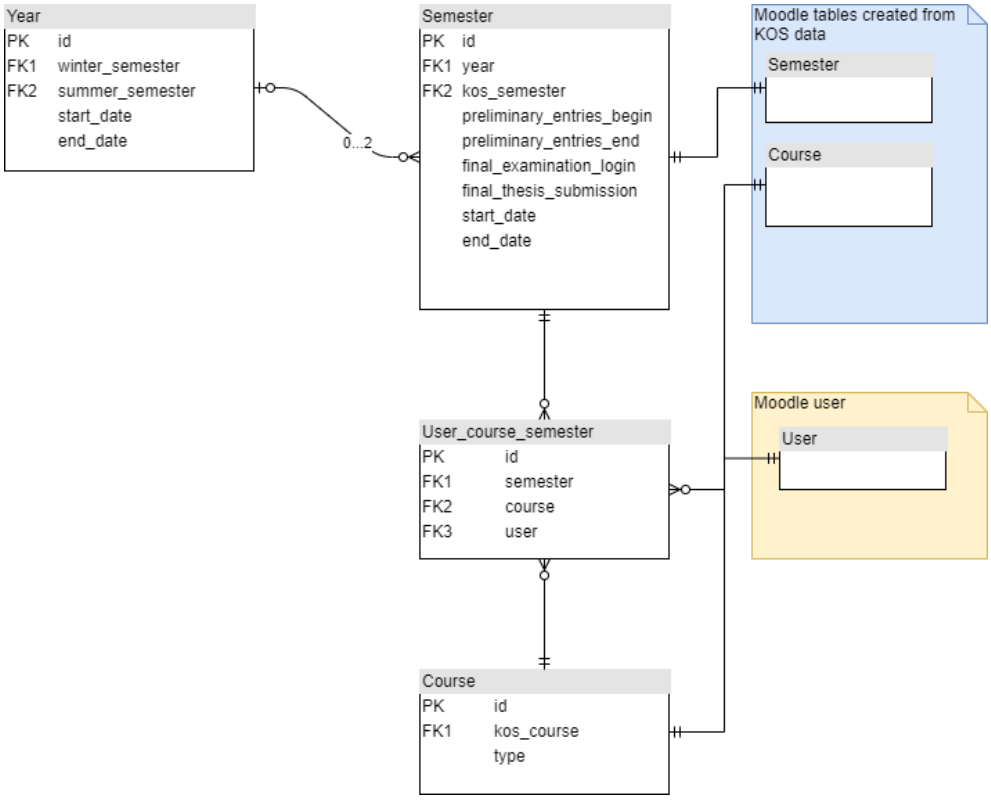
Moodle podporuje instalaci pluginů. Tyto pluginy mohou být staženy z oficiálních stránek aplikace nebo je mohou implementovat samotní uživatelé pro lokální použití. To je pro nás klíčové, jelikož naše aplikace bude implementována jako plugin pro použití na FEL. Instalaci pluginů mohou provádět administrátoři aplikace.

Pro komunikaci se školní databází je potřeba implementovat rozhraní, které pomocí REST komunikuje s takzvaným KOSapi [16] .Díky tomuto rozhraní lze získat data ze školní databáze. Fakultní Moodle má implementovaný a fungující plugin pro komunikaci s tímto rozhraním. Tento plugin můžeme použít, jelikož jednotlivé pluginy a jádro aplikace mohou být provázány. Bez dat, které jsme schopní díky KOSapi získat, bychom nebyli schopní aplikaci napsat. Podrobnější popis využití KOSapi se nachází v sekci 6.3. Další API, které během implementace využijeme, je Calendar API, které poskytuje jádro systému Moodle. Toto API vytváří události a notifikace pro uživatele.

Kvůli volbě systému Moodle jsme se vyhnuli hledání technologií, které bychom při implementaci použili a musíme použít to, co umí. Systém je napsán v jazyku PHP a využívá PostgreSQL databázi.

6.2 Databázový model

Pro plugin bude potřeba vytvořit vlastní databázi, která bude uložená do již existující databáze. Současně s tím ale bude systém využívat již existující databázi a pluginová databáze bude na tuto databázi napojená z několika tabulek.



Obrázek 6.1. Databázový model pluginu.

Databáze je napojena na hlavní KOS databázi přes tabulky semester, course a user_course_semester.

6.3 Komunikace s externími systémy

Aplikace bude komunikovat s některými školními systémy. Prvním, který v této kapitole uvedeme, je **KOSapi** [16] . Toto API slouží ke komunikaci se školní databází. Není možné díky němu zapisovat data do školní databáze, ale je možné data získávat a číst. Komunikaci s tímto API nám zajistí již implementovaný plugin v Moodle.

Metody, které přes tento plugin budeme volat a dotazovat se na data jsou:

- GET /courses/code/students
- GET /semesters/current
- GET /courses/code

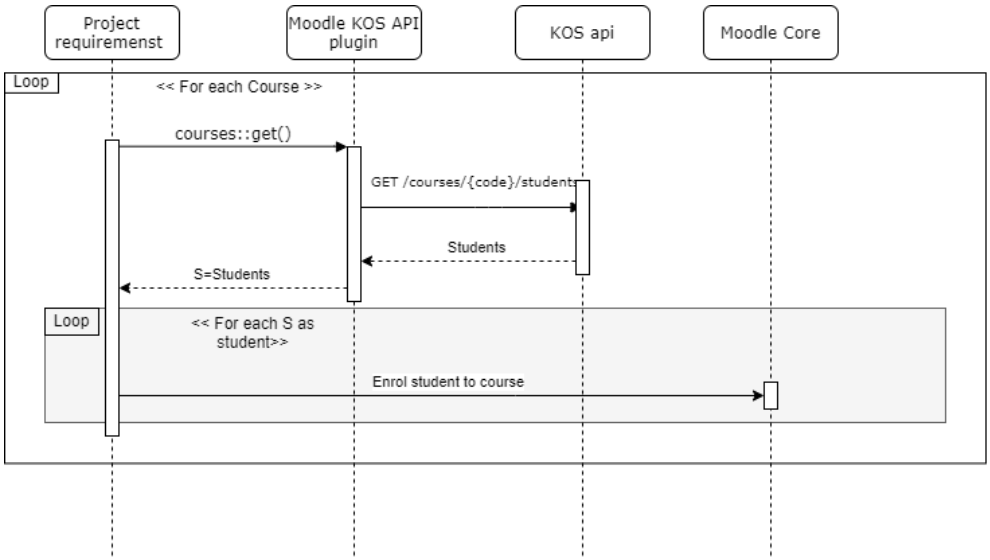
Dále budeme komunikovat s **aplikací FELSight pomocí již existujícího pluginu Rabbit Notification**. Do této aplikace budou nahrány termíny, které budou vytvořeny jednotlivým uživatelům. Uživatelé tak tyto termíny uvidí na 2 místech. Využití tohoto pluginu je automatické s vytvořením termínů a není potřeba volat žádné speciální metody.

6.4 Zápis studentů do kurzu

Zápis studentů do kurzu bude probíhat každý den přes skript, který bude spouštěn pomocí CRON. Podmínky pro vstup do kurzu jsou definovány v sekci 3.4.5. Bohužel jednu z podmínek nebude možné ověřit a to počet kreditů. Pro získání počtu kreditů je potřeba najít všechny studenty, k nim získat všechny absolvované předměty a spočítat

jejich kredity. Tento výpočet by byl velmi náročný a provádět ho každý den je vysoce neefektivní.

Proto se budeme řídit pouze zápisem jednoho z předmětů Závěrečná práce nebo Semestrální projekt, které jsou dostatečně směrodatné, protože se tak stává automaticky v předběžných zápisech předmětů. Pokud si student tento předmět zapíše, tak bude v dohledné době studium dokončovat a tyto informace jsou pro něj tedy relevantní.



Obrázek 6.2. Sekvenční diagram zápisu studentů do kurzu.

Nahrání studentů do kurzu proběhne v době předběžných zápisů, kdy si studenti volí předměty na následující semestr. Už v té době by měl mít student informace o tom, co, kdy a jak má udělat. Bylo potřeba ale ohlídat, aby bylo možné zapsat studenty i v již probíhajícím semestru, kdy mohou přes studijní oddělení změnit zapsané předměty. Proto zápis do kurzu probíhá jak pro následující, tak pro současný semestr.

6.5 Vytváření událostí

V současné době poskytuje KOSapi pouze omezené možnosti, jaká data je možné z databáze dostat. Museli jsme proto najít jinou cestu, jak data získat. Jedna z možností je **zadat data ručně** do systému, to by ale znamenalo dělat každý rok ruční aktualizaci. To by bylo časově náročné.

Druhá a efektivnější možnost je **získat data parsováním z harmonogramu na stránkách fakulty** [17] , kde jsou data k dispozici. Bude potřeba tedy napsat HTML parser, který data získá. Nebude tak potřeba dělat každý rok ruční aktualizaci, ale bude možné data aktualizovat automaticky.

Jakmile budou jednou data z harmonogramu získána, nebude potřeba je parsovat znovu. **Parsování proběhne vždy při změně školního roku**. Harmonogram je schvalován dopředu a změny v akademickém roce musí schválit fakultní senát. Pro nás důležité termíny jsou ale pevně dané a zpravidla se nemění. Po vytvoření harmonogramu v databázi budou zapsaným studentům vytvořeny události do Moodle databáze (tabulka mdl_events).

Důležitá data:

- Začátek letního semestru
- Konec letního semestru

- Začátek zimního semestru
- Konec zimního semestru
- Začátek předběžných zápisů v obou semestrech
- Konec předběžných zápisů v obou semestrech
- Termín přihlášení ke státním závěrečným zkouškám
- Termín odevzdání závěrečných prací

6.6 Kurz v aplikaci Moodle

V rámci implementace jsme vytvořili kurz v aplikaci Moodle, ve kterém student posledního ročníku nalezne všechny potřebné informace a odkazy. Struktura kurzu je k dispozici v příloze J a samotný kurz je součástí přílohy ve formátu MBZ. Obsah celého kurzu byl implementován pomocí rozhraní aplikace Moodle.

První sekci, kterou je potřeba zmínit je sekce úkoly, kde student najde strukturovaný seznam úkolů seřazených tak, jak by je měl plnit. Do seznamu může přidávat svoje položky a jednotlivé položky editovat. Při sestavování úkolů jsme vycházeli z procesů dostupných v přílohách H, G a dokumentu [18] s návodem na dokončení studia a zkušeností studentů.

Další sekcí, kterou je potřeba zmínit, je sekce V čem psát. Zde jsou popsána doporučení, jak mohou studenti psát svoji práci. Je zde také k dispozici upravená šablona LaTeX od profesora Olšáka, kterou mohou studenti použít. [19]

Na závěr kurzu jsou uvedeny odkazy, kde studenti najdou více informací. Pokud mají jiný specifický problém, musí věc řešit se studijním oddělením, zástupci příslušné katedry nebo vedoucím práce.

6.7 Použitá Moodle API

Pro vytváření událostí jednotlivým uživatelům jsme použili **Moodle Calendar API**. Pomocí tohoto API je možné spravovat uživatelské události, které se zobrazují v kalendáři. Aplikace pak umožní uživateli pouze zobrazení těchto událostí, ale ne jejich editaci. [20]

Dalším API, které jsme při implementaci využili, je **Moodle Data manipulation API**. Toto API, jak již jeho název napovídá, slouží k manipulaci s daty uvnitř databáze. Jeho použití je možné vidět například u jednotlivých entit, ze kterých dochází k přístupu do databáze. [21]

6.8 Spouštění pluginu

Jedná se o takzvaný scheduled_task, který je pouštěn automaticky každou noc okolo 3. hodiny ráno díky systému CRON. Plugin je možné spustit i ručně přes vyhledání jednotlivých tasků. To ale může dělat pouze administrátor aplikace. Změnit nastavení spouštění tasku je možné upravením souboru task ve složce db.

```php
defined( name: 'MOODLE_INTERNAL') || die;
$tasks = [
    [
        'classname' => 'local_projectrequirements\task\load_students',
        'blocking' => 0,
        'minute' => 'R',
        'hour' => '3',
        'day' => '*',
        'dayofweek' => '*',
        'month' => '*'
    ]
];
```

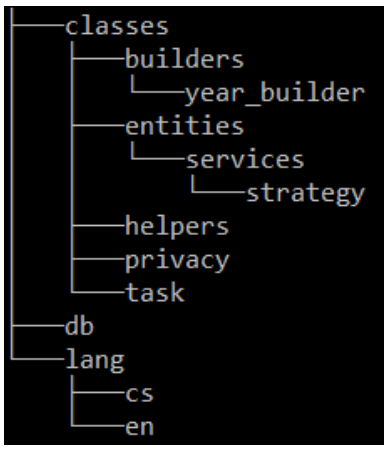
Obrázek 6.3. Nastavení spouštění tasku pluginu.

6.9 Struktura a obsah projektu

Hlavní adresář projektu je rozdělen na 3 základní složky. Jsou to **classes**, **db** a **lang**. V hlavním adresáři se nachází soubory **settings** a **version**. Setting slouží k vytvoření nastavení projektu v administrátorské sekci. Version určuje verze pluginu.

Složky lang a db jsou jednoduché. Ve složce lang jsou jazykové soubory pro uživatelské rozhraní. Je zde angličtina a čeština. Složka db obsahuje instalační soubor databáze, který je použit při instalaci programu a vytváření databáze pluginu. Posledním souborem, který se ve složce db nachází, je **soubor task**, kde je definováno, kdy a který soubor bude automaticky spouštěn pomocí CRON.

Souborová struktura classes obsahuje všechny zdrojové kódy projektu. V podsložce **builders** je použit návrhový vzor builder pro vytváření konkrétního roku. Složka **entities** slouží pro komunikaci s databází. Jednotlivé DTO jsou opsány z již existujícího kódu aplikace. Konkrétně jsme při jejich psaní vycházeli z **KOSapi pluginu**. Dále se ve složce entities nacházejí **services** pro vytváření a čtení náročnějších databázových vztahů a v ní je složka **strategy**, kde jsou jednotlivé strategie pro vytváření událostí a kurzů v databázi. Jedna z posledních složek je **helpers**, kde jsou pomocné soubory jako například parsování harmonogramu ze stránek FEL, vytváření notifikací a také třída custom_date pro vlastní typ data. Poslední dvě složky jsou **privacy** a **task**. Složka task obsahuje spouštěnou třídu task, která je spouštěna pomocí CRON a jedná se o hlavní třídu projektu.



Obrázek 6.4. Stromová struktura projektu.

Při implementaci jsme použili **návrhové vzory Strategy** a **Builder**. Builder je použitý pro vytváření struktury roku. Nejdřív je vytvořený rok a poté jednotlivé semestry se všemi termíny. Strategy jsme využili při vytváření notifikací a jednotlivých typů kurzů.

Zdrojový kód pluginu i kurzu jsou k dispozici na fakultní aplikaci GitLab ?? nebo v příloze.

6.10 Možnosti rozšíření pluginu

V případě, že by se plugin osvědčil a z vedení jednotlivých studijních programů by byla zaznamenána vůle k tomu kurz spravovat používat, tak je možné ho upravit tak, aby umožňoval správu kurzů pro jednotlivé programy. Nyní je nastaven pouze na obecné užívání všemi studijními programy na fakultě. Personalizace obsahu kurzu ze strany vedení studijních programů by pak vedla ke zvýšení přehlednosti požadavků u semestrálních projektů a závěrečných prací. Další rozšíření, které je možné implementovat je **vytváření vlastních událostí administrátorem kurzu**, který by mohl vytvářet vlastní události nebo upravovat již existující, které sám vytvořil nebo byly automaticky vytvořeny aplikací.

Posledním rozšířením, které by bylo možné realizovat, je **napojení kurzu na GitLab repozitáře s projekty studentů**. Muselo by být vytvořeno API pro komunikaci s tímto systémem a jednoduchý interface, přes který by studenti mohli svůj projekt nahrávat a stahovat pomocí commitů. Toto rozšíření by ocenili zejména studenti programů s programováním, kteří se se systémem GitLab setkali a většinou ho při vytváření závěrečné práce využívají.

6.11 Problémy při implementaci

V rámci implementace jsme narazili na zajímavý problém a tím je, že aktuálně **není možné ze školních systému získat harmonogram fakulty**. Jediné místo, kde se tento harmonogram nachází, jsou v tuto chvíli stránky fakulty. Bylo by dobré aktualizovat rozhraní KOSapi, aby bylo možné tyto data ze systému dostat.

Dalším problémem je, že **entita kurzu v KOS databázi nemá atribut pro identifikaci typu předmětu**. Nejde tak určit, jestli se jedná o semestrální projekt, nebo závěrečnou práci. Z toho důvodu je nutné tento seznam vytvořit ručně a aktualizovat. Tuto práci bude muset provádět administrátor při reakreditaci jednotlivých studijních programů a vytváření nových kurzů týkajících se semestrálního projektu a závěrečné práce v jednotlivých studijních plánech.

6.12 Naplnění cílů analýzy

V analýze jsme definovali jednotlivé požadavky na systém, které jsme při implementaci museli naplnit. K dispozici proto dáváme přehled jednotlivých požadavků a jejich stav vzhledem k implementaci v tabulce níže.

Id	Název	Implementován(ano/ne)
1	Vytvoření projektu	Ano
2	Vytvoření podprojektů	Ano
3	Vytvoření úkolů	Ano
4	Správa úkolů	Ano
5	Přehled termínů	Ano
6	Zobrazení termínů	Ano
7	Evidence změn	Ano
8	Nahrání posledních verzí	Ano
9	Nahrání uživatelů	Ano
10	Zobrazení dat	Ano
11	Správa projektu	Ano

Tabulka 6.1. Tabulka implementace jednotlivých požadavků do pluginu.

Požadavky 7 a 11 umožňuje samotná podstat aplikace Moodle. Zobrazení termínů, které je pod ID 6, je možné pouze při vývoji, aplikace totiž nemá kalendář pro uživatele dostupný. Termíny by ale měly být pomocí pluginu RabbitMQ posílány do aplikace FELSight, která je zobrazí pomocí notifikací. Je tak možné prohlásit tento požadavek za splněný.

6.13 Závěr

Hlavním výstupem této kapitoly je hotový a fungující plugin. Tento plugin je k dispozici na přiloženém CD jako příloha. Jak plugin do aplikace nainstalovat a nakonfigurovat popíšeme v následující Kapitole 7. V rámci implementace jsme museli rozlišit, kdy studenty do kurzu pustit a kdy jim vytvořit příslušné události.

Dalším výstupem implementace je vytvořený kurz v aplikaci Moodle, na který se zaměříme během testování v Kapitole 8. Tento kurz vznikl z dokumentů dostupných na stránkách fakulty a procesních diagramů v Procesním portálu. [14–15] .

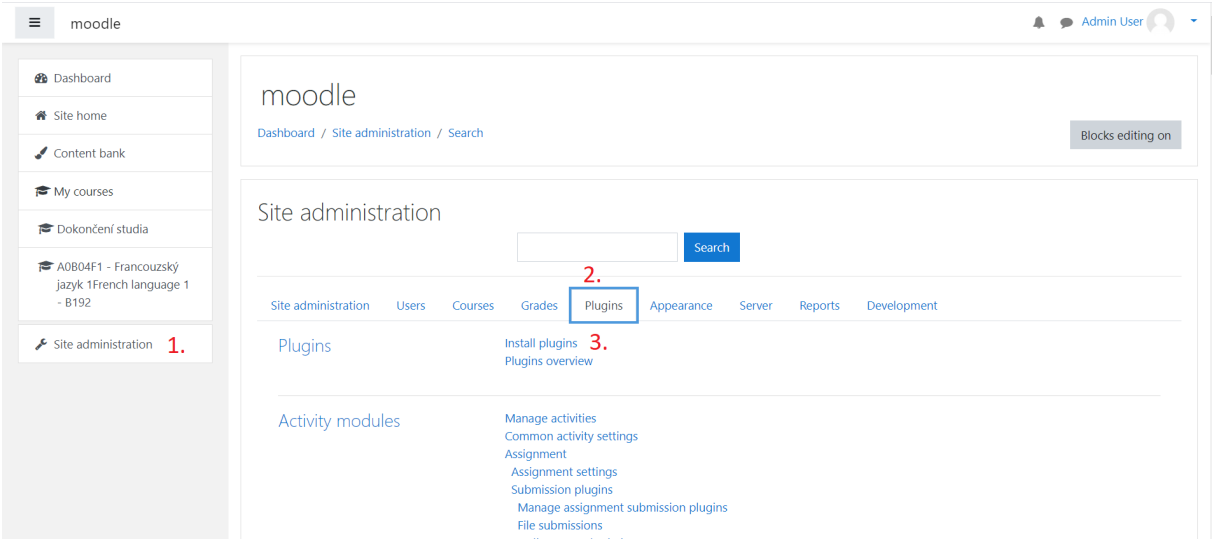
Kapitola 7
Instalace a nastavení

V této kapitole popíšeme, jakým způsobem je možné nainstalovat projekt, který jsme pojmenovali **Projectrequirements**, do aplikace Moodle. Nejprve popíšeme instalaci samotného pluginu a poté jeho nastavení. Tento postup je potřebný pro provozování na Fakultě elektrotechnické. Aktuálně na fakultě používáme Moodle ve verzi 3.9. Aby mohla aplikace běžet, tak potřebuje mít k dispozici server s minimální verzí **PHP 7.2.0** a **PostgreSQL 9.5**. Pro zobrazení aplikace a její spuštění je pak potřeba prohlížeč na straně klienta. Instalace a konfigurace se týká pouze instalací pluginu. Instalací samotné aplikace Moodle se v této kapitole věnovat nebudeme.

7.1 Instalace

Nejprve je potřeba stáhnout a nainstalovat fakultní verzi systému Moodle. Tato verze je přístupná pouze pro vývojáře a osoby, kteří tuto aplikaci spravují. Instalaci pluginu může z těchto lidí provádět pouze ten, který má přiřazenou roli administrátora v aplikaci. Na jiných verzích Moodle nebude plugin fungovat, jelikož pro komunikaci se školní databází využívá lokální plugin KOSapi.

Plugin se do aplikace instaluje pomocí webového rozhraní. Nejprve přejdeme do nastavené přes tlačítko Site aministration. Poté se musíme dostat do sekce Plugins a zde je možnost Install plugins, což je znázorněno v Obrázku 7.1.



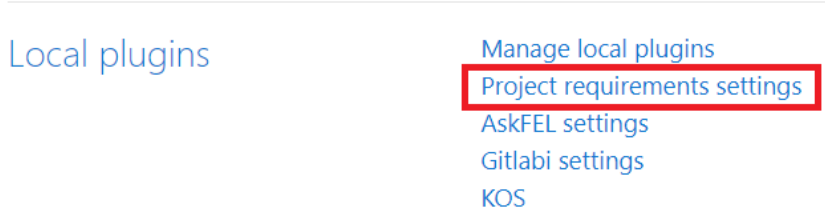
Obrázek 7.1. První krok instalace.

Nyní je potřeba mít stažený plugin k dispozici ve formátu ZIP. Pomocí drag and drop je možné plugin nahrát, kliknout na Instalovat a následně potvrdit další krok, kde dojde k ověření instalovaného pluginu. V dalším kroku Moodle ověří, jestli plugin splňuje veškeré požadavky na běh, pomocí tlačítka Continue pokračujeme dále. Na další stránce se zobrazí přehled pluginů, které jsme nahráli nebo které systém nabízí

k instalaci nebo aktualizaci. Pomocí Upgrade Moodle database pokračujeme a tím je instalace pluginu u konce a přejdeme ke konfiguraci.

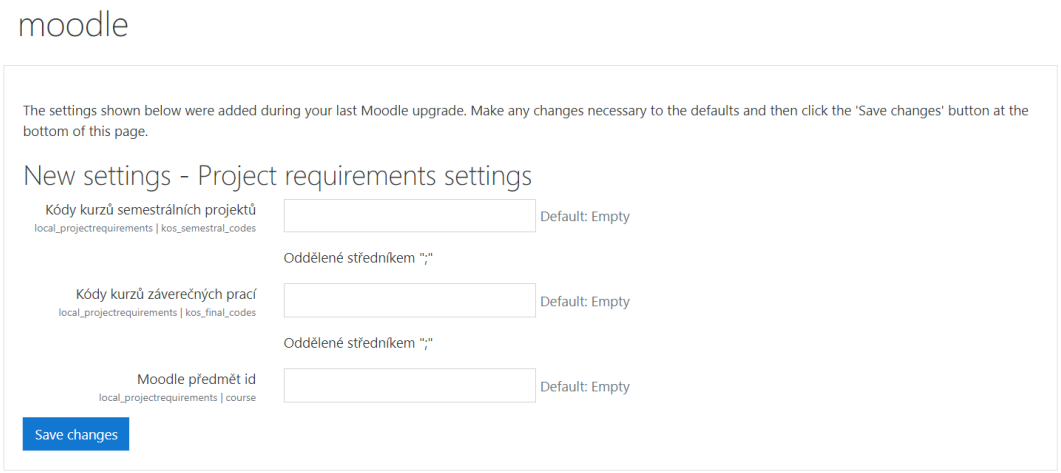
7.2 Konfigurace

Plugin je po nainstalování možné konfigurovat buď bezprostředně po instalaci, nebo je možné upravit již vytvořené nastavení pluginu tak, že v nastavení a sekci Plugins najdeme část locals a vybereme možnost Project requirements setting, jak je vidět na obrázku 7.2.

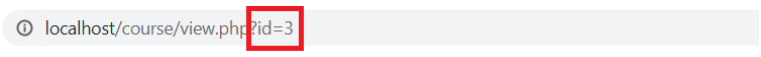


Obrázek 7.2. Umístění nastavení konfigurace v nastavení.

V samotné konfiguraci, která vypadá jako na Obrázku 7.3, je pak potřeba vyplnit kódy všech předmětů semestrálních projektů a závěrečných prací, které jsou na fakultě vyučovány. Jednotlivé položky seznamu je potřeba oddělit středníkem. Do posledního textového pole se vyplní ID příslušného kurzu v aplikaci Moodle. Toto ID je možné získat tak, že administrátor přejde do kurzu a ID se mu zobrazí za URL adresou - viz Obrázek 7.4.



Obrázek 7.3. Vyplnění konfigurace pluginu.



Obrázek 7.4. Zobrazení ID kurzu v systému Moodle.

Kapitola 8
Testování

Poslední kapitolou uzavřeme celou práci a popíšeme, jakým způsobem probíhalo testování výsledného řešení a výstupů implementace, kterým je kurz v aplikaci Moodle. Nejprve projdeme dostupné možnosti testování a vybereme tu, která je optimální. Poté určíme testovací scénáře a napíšeme výstupy z testování.

8.1 Výběr testů

Existuje několik druhů testů, které bychom mohli pro testování zvolit. Na začátek vybereme mezi manuálním a automatizovaným testováním. Automatizované testování se hodí zejména, pokud se jedná o velkou aplikaci, která obsahuje mnoho tříd, byznys logiky a komunikuje s velkým množstvím systémů pomocí různých rozhraní. Vzhledem k tomu, že se jedná o plugin, který není z hlediska byznys logiky náročný, tak zvolíme manuální testování.

Nejprve proto musíme otestovat samotný kód aplikace. Plugin je potřeba ručně spustit a vyzkoušet některé scénáře, které mohou nastat. Pro testování samotného kurzu a jeho obsahu je nejlepší použít uživatelské testování, kdy předem vybereme určitou množinu uživatelů, kteří pokryjí velké množství případů.

Uživatele jsme zvolili následovně:

- Studenti, kteří už **prošli procesem dokončení studia** a mohou tak zhodnotit, jestli jsou informace v kurzu správné a případně doplnit, co chybí.
- Studenti, které **dokončení studia teprve čeká**. Nezáleží, jestli v následujícím roce nebo později. Tito studenti nám mohou zhodnotit celkovou strukturu kurzu a pomoci upravit ji tak, aby dalším generacím co nejvíce pomohla.
- Studenti, kteří se **nachází v procesu dokončení studia** mají tak aktuální zkušenosti s dokončením a mohou přinést to, co předchozí 2 skupiny.

Dále pro uživatelské testování musíme sestavit jednotlivé scénáře, kterými se budeme zabývat v podkapitole 8.4.

8.2 Průběh testování funkčnosti pluginu

Pomocí manuálního testování je potřeba ověřit, jestli se plugin chová správně.

Z hlediska fungování pluginu je klíčové:

- Proběhne nahrání studentů při zápisu některého z předmětů Semestrální projekt nebo Závěrečná práce do následujícího semestru.
- Vytvoří se úkoly pouze studentům, kteří mají v aktuálním semestru zapsaný jeden z předmětů Semestrální projekt nebo Závěrečná práce.
- Pouze po nastavení harmonogramu na další rok se tento rok uloží do databáze.
- Pokud není v databázi aktuální rok, tak nedojde k vytvoření událostí uživatelům.

Na základě těchto údajů jsme schopní vytvořit testovací scénáře. Jelikož jsme vybrali manuální testování, musíme si ručně připravit data do databáze. K tomu využijeme nástroj zvaný PGAdmin, který dokáže komunikovat s databází a umožňuje nad ní veškeré CRUD operace. Tyto scénáře je pak potřeba ověřit ručně.

8.3 Uživatelské testování kurzu

Stejně jako při testování funkčnosti pluginu, i pro uživatelské testování bude potřeba rozumně zvolit testovací scénáře na základě požadavků na aplikaci z Kapitoly 4.4.

Pro užívání kurzu jsou důležité následující činnosti:

- Uživatel může zobrazit seznam úkolů.
- Uživatel může přidávat a mazat položky v seznamu úkolů.
- Uživatel může nahrát soubory pro Semestrální projekt a Závěrečnou práci.
- Uživatel může zobrazit kurz v aplikaci Moodle.

Administrátorské UC není potřeba testovat, jelikož vychází ze samotné podstaty aplikace Moodle a jsou testovány jádrem aplikace.

8.4 Testovací scénáře

Na základě podkapitoly 8.3 o uživatelském testování a testování funkčnosti pluginu z podkapitoly 8.2 můžeme sestavit jednotlivé testovací scénáře. Scénáře UT jsou k dispozici v příloze K, zde uvedeme pouze názvy a cíle jednotlivých scénářů. Scénáře pro testování funkčnosti jsou k dispozici v příloze L. Veškeré výstupy a přehled scénářů je také součástí příloh na CD.

Název	Cíl
Zobrazení úkolů	Uživatel si zobrazí sekci Úkoly.
Přidání úkolů	Uživatel přidá úkol mezi již existující úkoly a systém úkol uloží.
Nahrání Závěrečné práce	Uživatel nahraje svůj postup ve formě souboru do systému, který ji uloží.
Nahární Semestrálního projektu	Uživatel nahraje svůj postup ve formě souboru do systému, který ho uloží.

Tabulka 8.1. Tabulka scénářů uživatelského testování.

8.5 Závěry testování

Na základě výstupů UT jsme upravili vzhled kurzu a jeho obsah. Testování proběhlo manuálním nahráním testovaných subjektů do kurzu a následně uživatelé vzdáleně testovali aplikaci pomocí formuláře, který obsahoval jednotlivé scénáře.

Celkem se testování zúčastnilo 6 subjektů, které pokryly všechny možnosti definované v Kapitole 8.1.

1. Celkem **3 subjekty**, které procesem prošli.
2. Celkem **2 subjekty**, které do procesu právě vstoupili, z toho 1 již procesem předtím úspěšně prošel a spadá i do bodu č. 1.
3. Pouze **1 subjekt**, který se nachází v průběhu procesu.
4. Pouze **1 subjekt**, který do procesu vstoupí následující rok.

Manuální testování probíhalo v průběhu vývoje a na jeho základě byly upravovány jednotlivé funkce pluginu tak, aby splnily zadání.

Kapitola 9
Závěr

Cílem práce bylo pomoci studentům s řešením problémů s dokončením studia. Výstupem práce je plugin do aplikace Moodle, který slouží k nahrávání studentů do vytvořeného kurzu obsahujícího všechny potřebné informace a tím řídit jejich kroky k úspěšnému dokončení studia. Nejprve však byla vytvořena analýza celého procesu dokončení studia, ve které jsme našli důležité termíny z hlediska jednotlivých předmětů a definovali podmínky pro detekci studentů, kteří do tohoto procesu vstupují. Na základě výstupů této kapitoly jsme definovali požadavky na výsledný systém, který měl být výstupem celé práce.

Dále jsme se v rámci analýzy systémů na fakultě rozhodovali, jakou cestou implementace se vydat a zvolili jsme systém Moodle. Mohla vzniknout úplně nová aplikace, kterou bychom navrhli od základů a mohli tak použít jiné technologie. Z hlediska používání by ale aplikace trpěla aktuálním prostředím na fakultě, kde v současné době běží desítky aplikací a proto jsme se rozhodli právě pro Moodle.

Z hlediska implementace se nejednalo o tak jednoduchý úkol, jak se na první pohled může zdát. U nové aplikace bychom mohli rozvrhnout a vymyslet rozložení jednotlivých částí a obsahu a vše si tak přizpůsobit podle sebe. Museli jsme projít již napsaný kód aplikace a najít, co všechno můžeme využít. Zároveň bylo potřeba zjistit, jakým způsobem se provádí instalace pluginu a projít dokumentaci aplikace, která je sice obsáhlá, ale ne moc přehledná.

Právě kvůli zakomponování pluginu do Moodle jsme museli často volit to, co je a neměli jsme na výběr. Nejnáročnější bylo vytvořit podmínky pro vstup do kurzu a vytvoření termínů. Vše se nám nakonec povedlo a výstupem tak je funkční plugin. Jediné, co brání produkčnímu spuštění a uvedení do provozu je souhlas ze strany vedení fakulty a kontrola informací v kurzu studijními proděkany.

Práce také obsahuje informace, jakým způsobem je možné systém nainstalovat a zprovoznit na fakultě. Při nastavování pluginu je potřeba vyplnit kódy jednotlivých předmětů, které jsou určující pro detekci studentů. Tyto 2 seznamy jsou součástí Kapitoly 3 v Tabulkách 3.1 a 3.2.

Poslední fází je kapitola testování. Pro testování jsme vybrali uživatelské testování informací v kurzu projektu. Samotný plugin není složitý a spíše komunikuje s databází a provádí operace pro určité množiny studentů, proto stačilo otestovat ho manuálně. Pokud bychom volili unit testy, tak bychom mohli otestovat 2-3 metody, které jsou komplexnější a složitější.

Hlavní přínosem celé aplikace je sjednocení informací o dokončení studia všem studentům na fakultě a jejich představení v momentě, kdy je studenti začínají řešit. Celé řešení by tak mělo budoucím studentům posledních ročníku zpříjemnit toto náročné období a pomoci jim k úspěšnému dokončení celého studia.

Literatura

[1] *Výroční zpráva ČVUT FEL*. 2018.
http://www.fel.cvut.cz/cz/rozvoj/vyrocni-zprava2018.pdf. Navštíveno: 10.1.2020.

[2] *Projekt*. 2015.
https://managementmania.com/cs/projekt. Navštíveno: 6.1.2020.

[3] Christine B. Tayntor. *Project Management Tools and Techniques for Success : Definition, Planning, Execution, and Control* . Edition 1 vydání. CRC Press LLC , 2010 . ISBN 9781439816301.

[4] *Metody řízení projektu*. 2016.
https://managementmania.com/cs/metody-rizeni-projektu. Navštíveno: 6.1.2020.

[5] *Životní cyklus software*. 2012.
https://kalabovi.org/pitel:isz:zivotni_cyklus_softwaru. Navštíveno: 7.1.2020.

[6] *Účastníci projektu*. © 2009-2019.
https://www.potifob.cz/Klicove-zucastnene-strany-role-a-odpovednosti-na-projektu. Navštíveno: 7.1.2020.

[7] *Vodopádový model*. 2015.
https://managementmania.com/cs/vodopadovy-model-waterfall-model. Navštíveno: 8.1.2020.

[8] 2018.
https://medium.com/@lizparody/waterfall-vs-agile-methodology-in-software-development-1e19ef168cf6. Navštíveno: 8.1.2020.

[9] 2016.
http://www.knesl.com/co-je-agile. Navštíveno: 8.1.2020.

[10] 2016.
http://www.knesl.com/co-je-scrum. Navštíveno: 8.1.2020.

[11] 2012. Navštíveno: 8.1.2020.

[12] 2020.
https://www.fel.cvut.cz/cz/education/bk/. Navštíveno: 10.8.2020.

[13] *Směrnice děkana pro závěrečné práce a státní zkoušky v bakalářských a magisterských studijních programech na ČVUT FEL*. 2018.
https://www.fel.cvut.cz/cz/rozvoj/smerniceSZZ.pdf. Navštíveno: 9.1.2020.

[14]
https://www.fel.cvut.cz/cz/. Navštíveno: 27.4.2020.

[15]
https://procesy.cvut.cz/bpm/. Navštíveno: 25.7.2020.

[16] © 2006-2015.
https://kosapi.fit.cvut.cz/projects/kosapi/wiki. Navštíveno: 3.8.2020.

[17] https://www.fel.cvut.cz/cz/education/harmonogram.html. Navštíveno: 9.8.2020.

[18] 2017. https://www.fel.cvut.cz/cz/education/bachelor/ukonceni-zaklad.html. Navštíveno: 9.1.2020.

[19] 2020. http://petr.olsak.net/ctustyle.html. Navštíveno: 14.8.2020.

[20] 18.6.2020. https://docs.moodle.org/dev/Calendar_API. Navštíveno: 9.8.2020.

[21] 15.1.2020. https://docs.moodle.org/dev/Data_manipulation_API. Navštíveno: 9.8.2020.

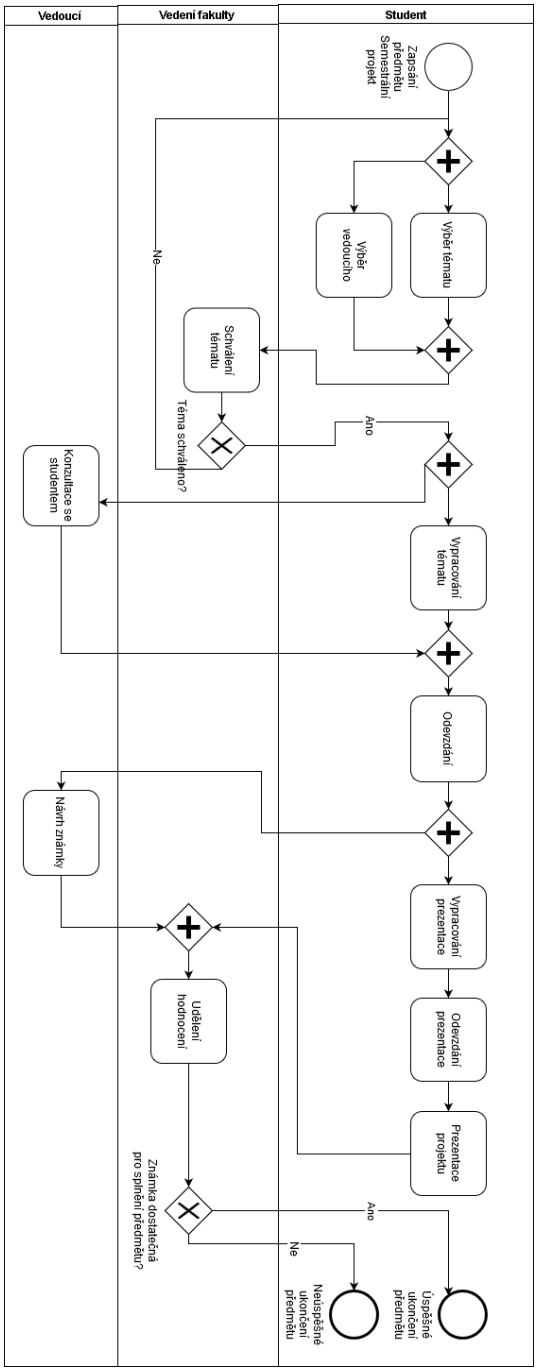
[22] https://procesy.cvut.cz/bpm/process.xhtml?pid=350&tenant=7&remember=true. Navštíveno: 23.7.2020.

[23] https://procesy.cvut.cz/bpm/process.xhtml?pid=411&tenant=7. Navštíveno: 23.7.2020.

[24] https://procesy.cvut.cz/bpm/process.xhtml?pid=412&tenant=7. Navštíveno: 27.4.2020.

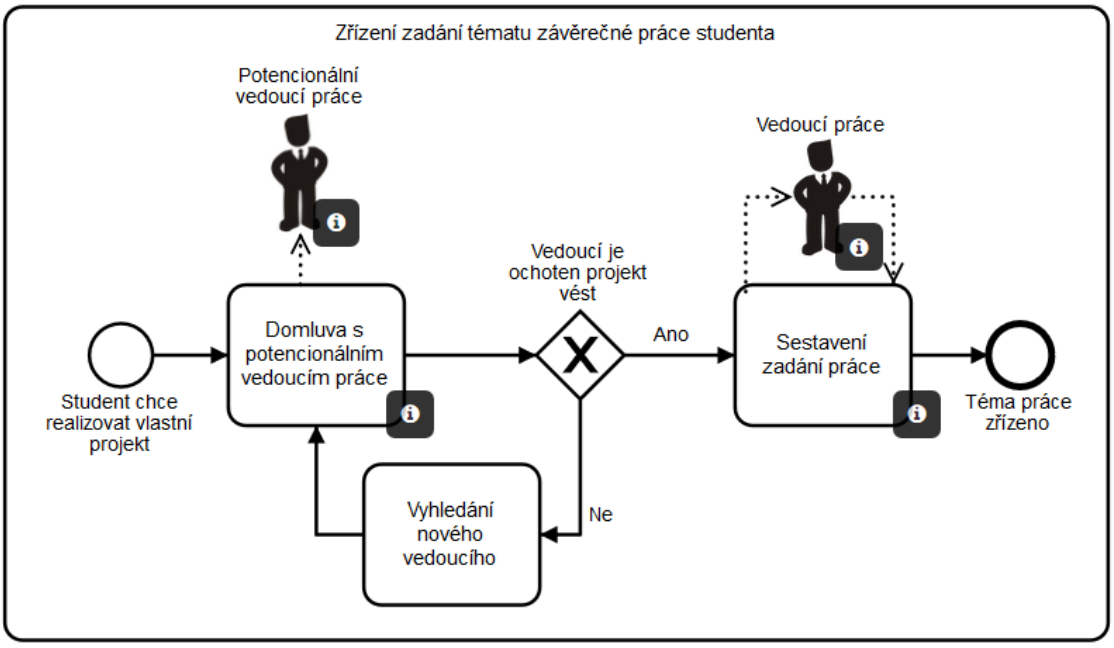
[25] https://procesy.cvut.cz/bpm/process.xhtml?pid=329&tenant=7. Navštíveno: 27.4.2020.

Příloha A
Procesní diagram semestrálního projektu



Obrázek A.1. Diagram popisující proces dokončení semestrálního projektu.

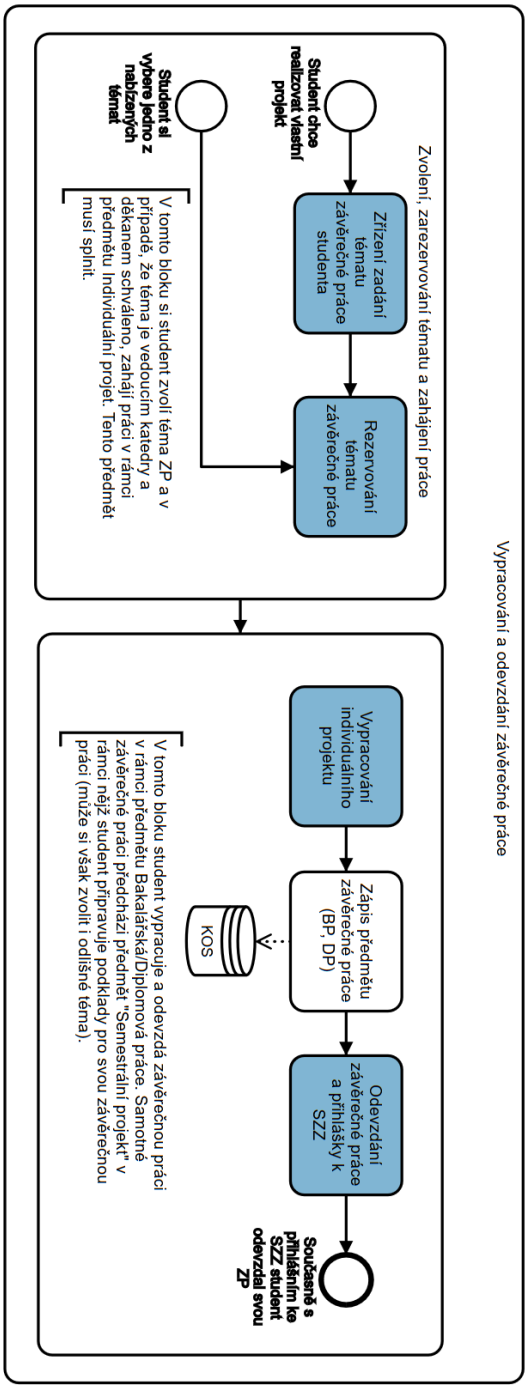
Příloha B
Procesní výběru vedoucího práce



Obrázek B.2. Diagram popisující proces výběru vedoucího práce [22].

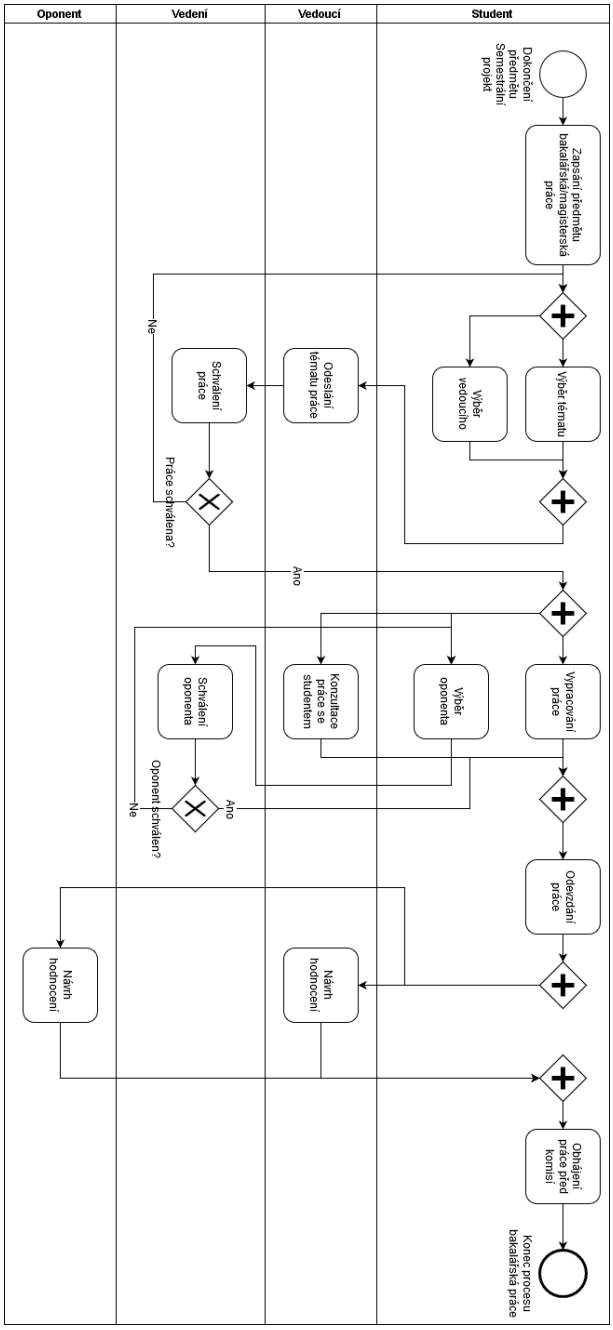
Příloha C

Procesní diagram závěrečné práce z procesního portálu FEL ČVUT



Obrázek C.3. Diagram popisující proces dokončení závěrečné práce z procesního portálu FEL ČVUT [23].

Příloha D
Procesní diagram závěrečné práce



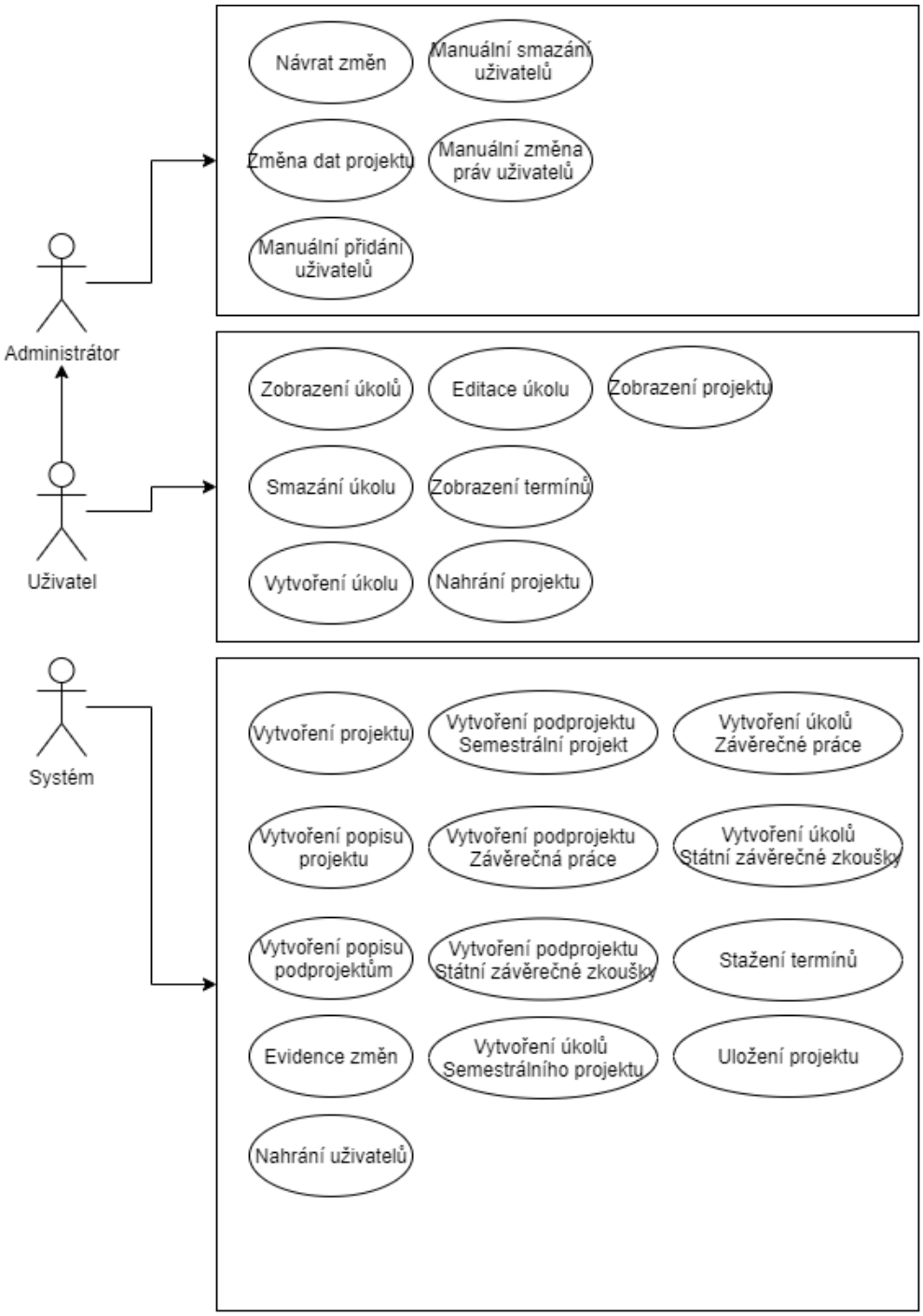
Obrázek D.4. Diagram popisující proces dokončení závěrečné práce, vypracovaný dle zkušeností a procesu z procesního portálu [23].

Příloha E
Seznam požadavků

ID	Název	Popis
HR01	Vytvoření projektu	Systém automaticky vytvoří projekt Zakončení studia.
HR02	Vytvoření podprojektů	Systém automaticky vytvoří podprojekty k projektu Zakončení studia.
HR03	Vytvoření úkolů	Systém automaticky vytvoří jednotlivé úkoly studentům do sekce Seznam úkolů.
HR04	Správa úkolů	Systém umožní uživatel spravovat úkoly, které jsou v sekci Seznam úkolů.
HR05	Přehled termínů	Systém automaticky vytvoří uživateli seznam termínů.
HR06	Zobrazení termínů	Systém umožní uživateli zobrazit termíny, které pro něj byly automaticky vytvořeny.
HR07	Evidence změn	Systém umožní verzovat stav celého projektu.
HR08	Nahrání posledních verzí	Systém umožní uživateli nahrát poslední verzi jednotlivých projektů. Tyto verzi půjdou nahrávat opakovaně a přemažou tak původní verzi.
HR09	Nahrání uživatelů	Systém automaticky stáhne a nahraje uživatele do projektu.
HR10	Zobrazení dat	Systém umožní uživateli zobrazit projekt Zakončení studia.
HR11	Správa projektu	Systém umožní administrátorovi spravovat projektu Zakončení studia.

Obrázek E.5. Diagram s jednotlivými požadavky na systém.

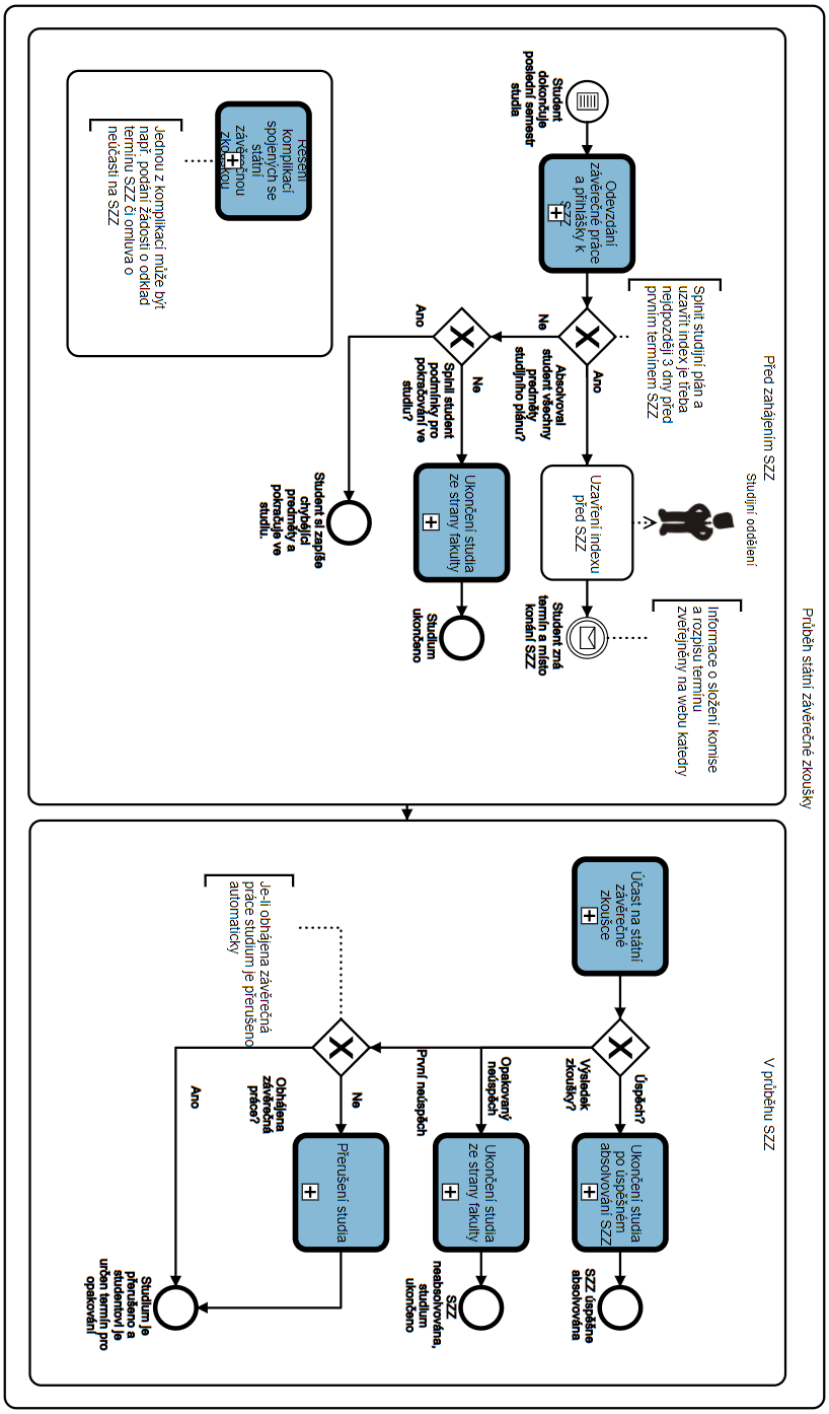
Příloha F
Diagram případů užití



Obrázek F.6. Diagram případů užití.

Příloha G

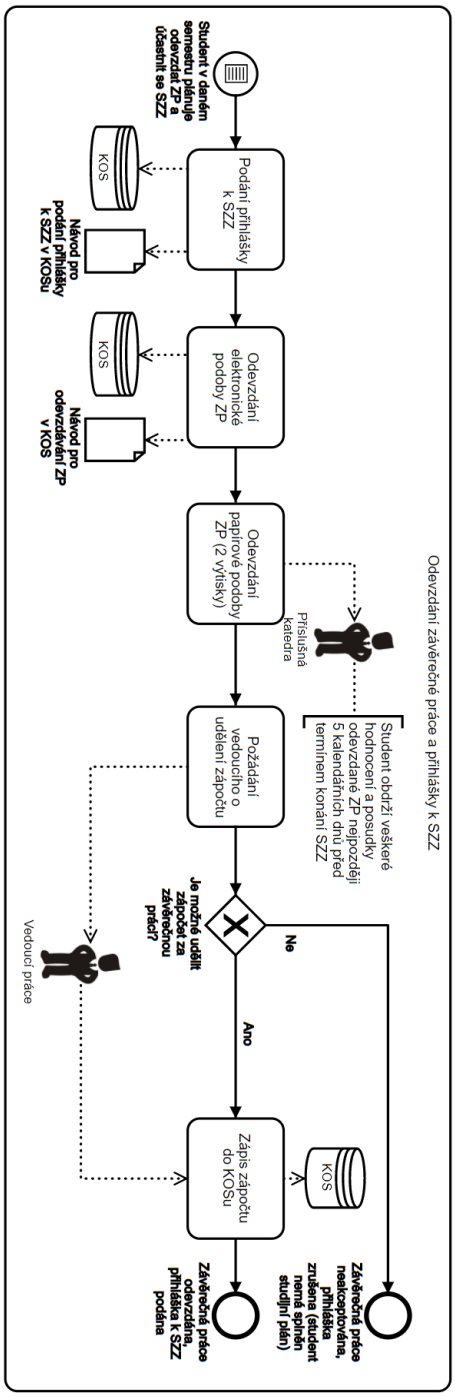
Diagram průběhu státních závěrečných zkoušek



Obrázek G.7. Diagram průběhu státních závěrečných zkoušek. [24]

Příloha H

Diagram odevzdání závěrečné práce a přihlášky k státním závěrečným zkouškám



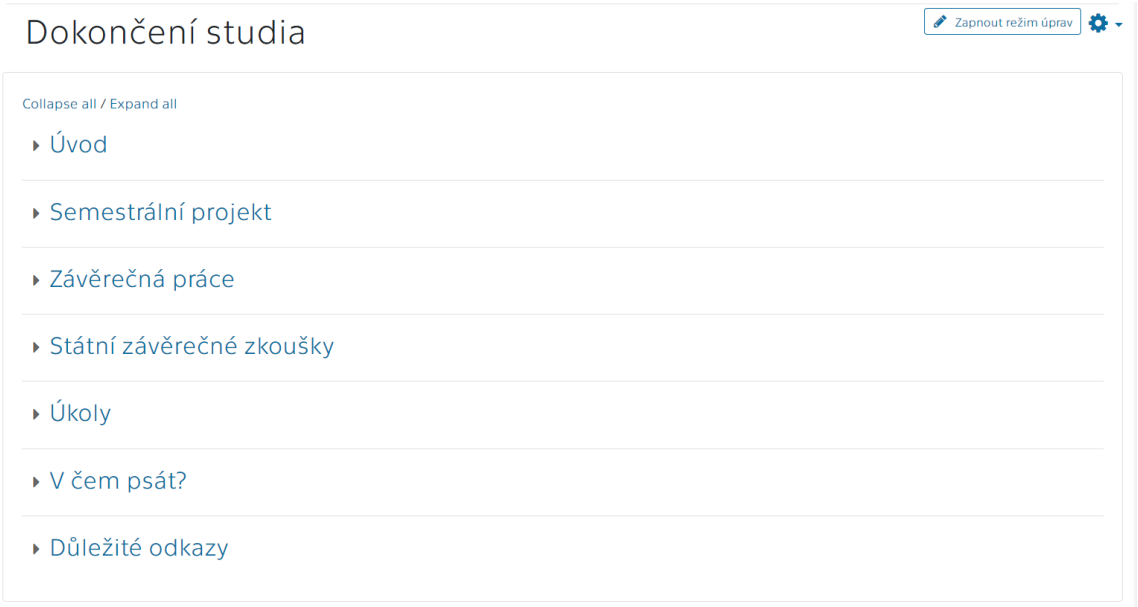
Obrázek H.8. Diagram odevzdání závěrečné práce a přihlášky k státním závěrečným zkouškám. [25]

Příloha I
Tabulka případů užití

ID	Název	Popis	Aktéři	Cíl	Požadavek
UC01	Vytvoření projektu	Systém automaticky vytvoří projekt dokončení studia pro uživatele.	Systém	Projekt byl vytvořen	HR01
UC02	Vytvoření popisu projektu	Systém automaticky vytvoření popis projektu Zakončení studia.	Systém	Data byla nahrána do projektu	HR01
UC03	Vytvoření podprojektu Semestrální projekt	Systém automaticky vytvoří podprojekt Semestrální projekt do projektu Zakončení studia.	Systém	Podprojekt byl vytvořeny	HR02
UC04	Vytvoření podprojektu Závěrečná práce	Systém automaticky vytvoří podprojekt Závěrečná práce do projektu Zakončení studia.	Systém	Podprojekt byl vytvořeny	HR02
UC05	Vytvoření podprojektu Státní závěrečné zkoušky	Systém automaticky vytvoří podprojekt Státní závěrečné zkoušky do projektu Zakončení studia.	Systém	Podprojekt byl vytvořeny	HR02
UC06	Vytvoření popisu podprojektům	Systém automaticky vytvoří popisy jednotlivým podprojektům projektu Zakončení studia.	Systém	Popisy podprojektů byly vytvořeny	HR02
UC07	Vytvoření úkolů Semestrálního projektu	Systém automaticky vytvoří uživatelům seznam úkolů: 1. Výběr vedoucího a tématu 2. Schválení tématu 3. Vypracování projektu 4. Odevzdání projektu 5. Vypracování prezentace 6. Prezentace a obhajoba projektu	Systém	Úkoly jsou vytvořeny	HR03
UC08	Vytvoření úkolů Závěrečné práce	Systém automaticky vytvoří uživatelům seznam úkolů: 1. Výběr vedoucího a tématu 2. Přihlášení tématu závěrečné práce 3. Schválení práce studijním oddělení nebo vedoucím příslušné katedry 4. Vypracování ZP 5. Odevzdání elektronické verze 6. Odevzdání papírové verze 7. Udělení zápočtu vedoucím 8. Vypracování oponentského posudku oponentem 9. Vypracování prezentace na obhajobu 10. Obhajoba	Systém	Úkoly jsou vytvořeny	HR03
UC09	Vytvoření úkolů Státní závěrečné zkoušky	Systém automaticky vytvoří uživatelům seznam úkolů: 1. Podání přihlášky k SZZ 2. Splnění studijního plánu 3. Složení SZZ	Systém	Úkoly jsou vytvořeny	HR03
UC10	Zobrazení úkolů	Uživatel bude moct zobrazit úkoly z UC07, UC08, UC09.	Uživatel	Uživatel zobrazí úkoly	HR04
UC11	Smazání úkolu	Uživatel bude moct smazat jím úkoly z UC07, UC08, UC09.	Uživatel	Úkol bude smazán	HR04
UC12	Vytvoření úkolu	Uživatel bude moct vytvořit úkol z UC07, UC08, UC09.	Uživatel	Úkol bude vytvořen	HR04
UC13	Editace úkolu	Uživatel bude moct editovat úkol z UC07, UC08, UC09.	Uživatel	Úkol bude editován	HR04
UC14	Stažení termínů	Systém automaticky stáhne termíny celého roku z webových stránek fakulty nebo z lokální databáze.	Systém	Termíny budou staženy	HR05

UC15	Zobrazení termínů	Systém umožní uživateli zobrazit přehled termínů vytvořených v rámci UC14.	Uživatel	Uživateli se zobrazí termíny	HR06
UC16	Evidence změn	Systém automaticky umožní evidovat změny na jednotlivých studentských projektech.	Systém	Projekt bude verzován	HR07
UC17	Návrat změn	Systém umožní administrátorům nahrát staré verze projektu Zakončení studia do systému.	Administrátor	Verze projektu bude změněna	HR07
UC18	Nahrání souborů	Systém umožní uživateli nahrát soubor s projektem Závěrečné práce a Semestárlní projekt.	Uživatel	Závěrečná práce bude nahrána a uložení	HR08
UC19	Uložení projektu	Systém uloží data nahráná uživatelem z UC18.	Systém	Data budou uložena	HR08
UC20	Nahrání uživatelů	Systém automaticky nahraje uživatele do projektu. Kritéria pro výběr uživatelů jsou: 1. Má 75 kreditů v případě bakalářského a 20 v případě magisterského studia 2. Má zapsaný předmět Semestrální projekt nebo Závěrečná práce 3. Nachází se v 1. a vyšším ročníku u bakalářského a 1. ročníku u magisterského studia	Systém	Uživatelé budou nahrání	HR09
UC21	Zobrazení projektu	Uživatel bude moct po nahrání v rámci UC20 do projektu zobrazit celý projekt Zakončení studia.	Uživatel	Uživatel zobrazí projekt Zakončení studia	HR10
UC22	Změna dat projektu	Systém umožní administrátorovi upravit data v projektu Zakončení studia a jeho podprojektů.	Administrátor	Data budou změněna všem uživatelům	HR11
UC23	Manuální přidání uživatelů	Systém umožní administrátorovi manuálně přidat uživatele do projektu.	Administrátor	Uživatel bude přidán do projektu	HR11
UC24	Manuální smazání uživatelů	Systém umožní administrátorovi manuálně smazat uživatele z projektu.	Administrátor	Uživatel bude smazán z projektu	HR11
UC25	Manuální změna práv uživatelů	Systém umožní administrátorovi manuálně měnit práva jednotlivým uživatelům projektu.	Administrátor	Práva budou upravena	HR11

Příloha J
Struktura kurzu v aplikaci Moodle



Obrázek J.9. Struktura kurzu v aplikaci Moodle.

Příloha K
Scénáře UT

Test case 1

Přejděte prosím do kurzu v aplikaci Moodle - https://moodle.fel.cvut.cz/course/view.php?id=4997
Cílem tohoto TC je zobrazit seznam úkolů.
*Povinné pole

1. 1. Podařilo se ti zobrazit seznam úkolů? *

   Označte jen jednu elipsu.

   - ◯ Ano
   - ◯ Ne *Přeskočte na otázku 11*

2. 2. Přišlo ti označení sekce přehledné? *

   Označte jen jednu elipsu.

   - ◯ Ano
   - ◯ Ne
   - ◯ Jiné: _____

3. Dokážeš identifikovat, co znamená barevné rozlišení úkolů? V odpovědi zkus popsat, co si myslíš, že to znamená. *

   _____
   _____
   _____
   _____
   _____

Doplňující otázky

Obrázek K.10. Strana 1.

4. Pokud jsi v otázce č.1 vybral ne, popiš, proč se ti nepodařilo seznam úkolů najít.

5. Ostatní - popiš své postřehy/nejasnosti/problémy při řešení Test casu.

Test case 2

Cílem TC je přidat do seznamu úkolů 2 úkoly.
1) Do vypracování semestrálního projektu přidejte úkol: Napsat úvod.
2) Do odevzdání závěrečné práce přidejte: Požádat o náhradní termín odevzdání

6. 1. Podařilo se ti přidat úkol č.1? *

Označte jen jednu elipsu.

- Ano
- Ne

7. 2. Podařilo se ti přidat úkol č.2? *

Označte jen jednu elipsu.

- Ano
- Ne

Doplňující otázky

Obrázek K.11. Strana 2.

8. Pokud jsi v otázce č.1 vybral ne, popiš, proč se ti nepovedlo přidat úkol.

9. Pokud jsi v otázce č.2 vybral ne, popiš, proč se ti nepovedlo přidat úkol.

10. Ostatní - popiš své postřehy/nejasnosti/problémy při řešení Test casu.

Test case 3

Cílem TC je nahrát projekt Závěrečné práce. Data jsou pouze testovací, můžeš tak odevzdat libovolný validní soubor.

11. 1. Podařilo se ti nahrát projekt Závěrečné práce? *

Označte jen jednu elipsu.

- Ano
- Ne

Obrázek K.12. Strana 3.

12. 2. Přišlo vám označení sekce přehledné? *

Označte jen jednu elipsu.

- ◯ Ano
- ◯ Ne
- ◯ Jiné: _____

Doplňující otázky

13. Pokud jsi v otázce č.1 vybral ne, popiš, proč se ti nepodařilo přidat řešení Závěrečné práce.

_____

_____

_____

_____

_____

14. Ostatní - popiš své postřehy/nejasnosti/problémy při řešení Test casu.

_____

_____

_____

_____

_____

Test case 4

Cílem TC je nahrát projekt Semestrálního projektu. Data jsou pouze testovací, můžeš tak odevzdat libovolný validní soubor.

Obrázek K.13. Strana 4.

15. 1. Podařilo se ti nahrát projekt Semestrálního projektu? *

Označte jen jednu elipsu.

- ◯ Ano
- ◯ Ne

16. 2. Přišlo vám označení sekce přehledné? *

Označte jen jednu elipsu.

- ◯ Ano
- ◯ Ne
- ◯ Jiné: _____

Doplňující otázky

17. Pokud jsi v otázce č.1 vybral ne, popiš, proč se ti nepodařilo přidat řešení Semestrálního projektu.

_____

_____

_____

_____

_____

18. Ostatní - popiš své postřehy/nejasnosti/problémy při řešení Test casu.

_____

_____

_____

_____

_____

Obrázek K.14. Strana 5.

Doplňující otázky

Nyní si zkus projít obsah kurzu, následující otázky jsou pouze doplňující a týkají se obsahu kurzu.

19. V jakém ročníku studia se nacházíš? *

_____

20. Řešil jsi už dokončení studia? *

Označte jen jednu elipsu.

- ◯ Ano
- ◯ Ne

21. Hledal jsi informace o dokončení studia? Pokud Ano, tak popiš, jestli ti kurz dal něco navíc nebo ti něco chybělo.

_____

_____

_____

_____

_____

22. Přišli ti informace v kurzu srozumitelné? *

Označte jen jednu elipsu.

- ◯ Ano
- ◯ Ne

Obrázek K.15. Strana 6.

23. Chyběly ti některé informace v obsahu kurzu?

24. Přišla ti struktura kurzu přehledná?

25. Připomínky/postřehy/nejasnosti..

Obrázek K.16. Strana 7.

Příloha L

Scénáře funkčního testování

ID	Název	Popis	Počáteční stav	Cílový stav
1	Uložení nového roku	Tento TC zjišťuje, jestli po aktualizaci harmonogramu na stránkách fakulty dojde k jeho stažení, vyparsování a správnému uložení do DB.	V databázi neexistuje záznam právě probíhajícího roku a nejsou tam ani oba semestry.	V databázi je vytvořen aktuální rok s počátečním a koncovým datem. Dále jednotlivé semestry, které jsou svázány pomocí cizího klíče s příslušným rokem. U semestrů jsou vytvořeny data začátku a konce předběžných zápisů, termínu odevzdání závěrečných prací a termín přihlášení k závěrečným zkouškám.
2	Pokus o opětovné uložení aktuálního roku	Je potřeba ověřit, jestli se nevytvoří stejná struktura daného roku v databázi 2x a dojde tam pouze k jednomu uložení roku.	V databázi již existuje záznam právě probíhajícího roku s oběma semestry.	V databázi nedojde k nahrání probíhajícího roku.
3	Nahrání studentů do kurzu	TC ověřuje, že došlo k nahrání studentů do kurzu.	V databázi neexistuje záznam o studentech předmětů SP nebo ZP v aktuálním nebo budoucím semestru.	Uživatelé jsou nahrání do kurzu a mají do něj uživatelský přístup. Mohou tedy číst data a měnit pouze předem definované části kurzu. Nemohou však měnit jeho obsah nebo strukturu.
4	Vytvoření úkol uživatelům SP v probíhajícím semestru	Studentům, kteří mají v aktuálním semestru zapsaný předmět SP, **je** vytvořen jeden úkol začátku a konce předběžných zápisů s popisem.	Uživatel **nemá** v databázi vytvořený termín k danému předmětu do probíhajícího semestru.	Uživateli **je** vytvořen příslušný úkol s popisem.
5	Opakované vytvoření úkolu uživatelům SP v probíhajícím semestru	Studentům, kteří mají v aktuálním semestru zapsaný předmět SP, **není** vytvořen jeden úkol začátku a konce předběžných zápisů s popisem.	Uživatel **má** v databázi vytvořený úkol k danému předmětu do probíhajícího semestru.	Uživateli **není** vytvořen příslušný úkol s popisem.
6	Vytvoření úkol uživatelům ZP v probíhajícím semestru	Studentům, kteří mají v aktuálním semestru zapsaný předmět ZP, **je** vytvořen jeden úkol začátku a konce předběžných zápisů s popisem.	Uživatel **nemá** v databázi vytvořený termín k danému předmětu do probíhajícího semestru.	Uživateli **je** vytvořen příslušný úkol s popisem.
7	Opakované vytvoření úkolu uživatelům ZP v probíhajícím semestru	Studentům, kteří mají v aktuálním semestru zapsaný předmět ZP, **není** vytvořen jeden úkol začátku a konce předběžných zápisů s popisem.	Uživatel **má** v databázi vytvořený úkol k danému předmětu do probíhajícího semestru.	Uživateli **není** vytvořen příslušný úkol s popisem.

Obrázek L.17. Scénáře funkčního testování.

Příloha M
Seznam použitých zkratek

API	Application Programming Interface
BS	Bakalářské studium
CD	Compact Disc
CRON	Softwarový démon pro spouštění procesů v definovaný čas
ČVUT	České vysoké učení technické v Praze
DTO	Data transfer object
FEL	Fakulta elektrotechnická ČVUT
HTML	Hypertext Markup Language - značkovací jazyk pro tvorbu webových stránek
ISO	International Organization for Standardization
KOS	Komponenta studium - studijní informační systém na ČVUT FEL
MS	Magisterské studium
PHP	Hypertext Preprocessor - programovací jazyk
PMBOK	Project Management Body of Knowledge
SP	Semestrální projekt
SZZ	Státní závěrečné zkoušky
TC	Test case
UC	Use case
UI	User Interface
URL	Unified Resource Identifier
UT	User testing
UX	User Experience
ZIP	Kompresní souborový formát
ZP	Závěrečná práce