ČESKÉ VYSOKÉ
UČENÍ TECHNICKÉ
V PRAZE

F3

Fakulta elektrotechnická
Katedra počítačů

Diplomová práce

Testovací systém pro notaci BPMN

Filip Štěpánek
Otevřená informatika - Softwarové inženýrství

Květen 2023
Vedoucí práce: Ing. Pavel Náplava, Ph.D.

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Štěpánek** Jméno: **Filip** Osobní číslo: **483820**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Otevřená informatika**

Specializace: **Softwarové inženýrství**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Testovací systém pro notaci BPMN

Název diplomové práce anglicky:

BPMN notation testing system

Pokyny pro vypracování:

Navrhněte systém, který umožní vyučujícím vytvářet testy pro zkoušení notace BPMN a zjednoduší jejich hodnocení. Postupujte následovně:
1. Seznamte se s notací BPMN – jaká je její historie, pro jaké účely se používá, jaké typy objektů a diagramů používá, porovnejte ji s jinými notacemi atd.
2. Na základě diskuse se zadavatelem a provedené rešerše identifikujte nejčastější problémy a anti-patterny, které se v BPMN diagramech vyskytují
3. Navrhněte způsob, jakým je možné automatizovaně problémy, popsané v bodě 2, hledat a kontrolovat
4. Navrhněte systém, v rámci kterého bude navržený způsob realizovaný a bude možné jej využít při výuce notace BPMN
5. Navržený systém vytvořte.
6. Navrhněte způsob otestování tohoto systému a otestujte jej. Minimálně formou uživatelských testů.

Seznam doporučené literatury:

Václav Řepa. Podnikové procesy: procesní řízení a modelování. 2., aktualiz. a rozš. vyd vyd.. Grada, 2007. ISBN 9788024722528.
Mark von Rosing, Henrik von Scheel a August-Wilhelm Scheer. The Complete Business Process Handbook. Elsevier, 2015. ISBN 9780128004722 .
Grady Booch, James Rumbaugh a Ivar Jacobson. Unified Modeling Language User Guide. Addison Wesley, 1998. ISBN 0-201-57168-4.
Business Process Model and Notation. OMG, prosinec 2010. http://www.omg.org/spec/BPMN/2.0.
Zakhutskyi Viacheslav. Simulátor modelování procesů v notaci BPMN. Bakalářská práce, ČVUT FEL, 2020.
Seliverstov Aleksandr. Aplikace pro ověřování dodržování standardů notace BPMN. Bakalářská práce, ČVUT FEL, 2020

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Pavel Náplava, Ph.D. Centrum znalostního managementu FEL

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

Datum zadání diplomové práce: **30.01.2023** Termín odevzdání diplomové práce: **26.05.2023**

Platnost zadání diplomové práce: **22.09.2024**

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

III. PŘEVZETÍ ZADÁNÍ

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

. _____
Datum převzetí zadání

_____
Podpis studenta

Poděkování / Prohlášení

Děkuji panu doktorovi Náplavovi za podporu a zkušené vedení této práce.

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 16. 5. 2023

.....

Abstrakt / Abstract

Tato diplomová práce se zabývá vývojem systému pro ověřování znalostí notace BPMN, která slouží pro modelování podnikových procesů. Ten v administrátorské části poskytne rozhraní pro tvorbu a kontrolu zadání a zkoušek. V klientské pak studenti budou mít možnost trénovat a za stanovených podmínek skládat zkoušky. Systém následně pokusy automaticky vyhodnotí srovnáním s referenčním řešením a kontrolou dodržení správné specifikace jazyka BPMN.

Klíčová slova: BPMN, testovací systém, zkouškový systém, automatické vyhodnocení

This master's thesis focuses on the development of a system for verifying knowledge of BPMN notation, which is used for modeling business processes. In the administrative part, the system provides an interface for creating and checking assignments and exams. In the client part, students will have the opportunity to practice and take exams under certain conditions. The system then automatically evaluates attempts by comparing them with a reference solution and checking compliance with the correct specification of the BPMN language.

Keywords: BPMN, test system, automatic evaluation

Obsah /

Tabulky / Obrázky

Kapitola 1
Úvod

Tato diplomová práce se zabývá vývojem testovacího systému z notace BPMN pro účely výuky předmětu Procesní řízení. V rámci tohoto předmětu jsou prezentovány metody modelování podnikových procesů v BPMN notaci, avšak v současnosti neexistuje platforma pro automatizované ověření znalostí této notace. Cílem této práce je tedy vyvinout webovou aplikaci pro snadné testování a trénink modelových scénářů.

V následujících kapitolách nejprve prozkoumám, co je to procesní modelování a jaké má účely. Krátce představím nejrozšířenější notace a důvody pro výběr BPMN. Tuto notaci pak podrobně rozeberu a stručně nastíním její strukturu a historii.

Následující kapitoly jsou věnovány vývoji systému. Nejprve představím proces výuky, dostupné výukové systémy a předchozí příbuzné práce. Zbylé kapitoly mají charakter softwarové dokumentace. Představím případy užití a softwarové požadavky, které byly shromážděny od garanta předmětu. Poté podrobně rozeberu technologický aspekt vyvíjené aplikace od návrhu architektury až po detailní výčet použitých technologií. Jedna kapitola je věnovaná nejdůležitějším implementačním detailům, které mi přišli zajímavé a důležité. Rovněž představím přístupy k testování a výsledky uživatelského testování, jehož scénáře jsou dostupné v příloze.

V závěrečné části zhodnotím diplomovou práci z projektového hlediska.

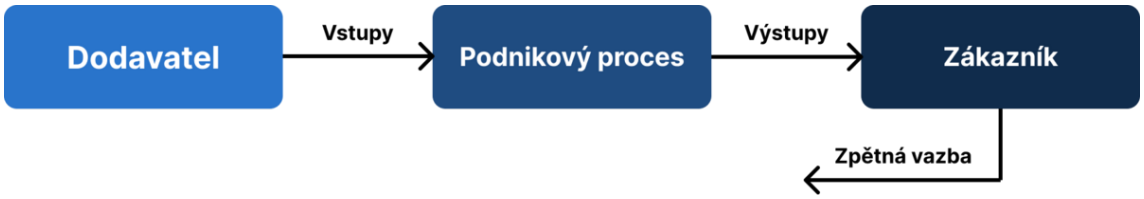
Kapitola 2
Podnikové procesy

V dnešní době se podniky a organizace neustále snaží zlepšovat svůj výkon, zefektivnit své činnosti a optimalizovat své zdroje. Abychom porozuměli, jak těmto cílům dosáhnout, je nezbytné se zaměřit na základní kámen každého podniku – podnikové procesy. V této kapitole stručně představím podnikové procesy, jejich zlepšování a modelování, včetně různých notací a nástrojů používaných pro jejich analýzu a optimalizaci.

2.1 Definice podnikového procesu

V rámci studia podnikových procesů je důležité si uvědomit, že existuje celá řada definic, které popisují jejich povahu a základní principy. Různí autoři a odborníci se snaží představit podnikové procesy z různých úhlů a perspektiv, což může někdy vést k mírným rozdílům v jejich chápání. Pro naše účely a pro lepší pochopení jsem vybral jednu konkrétní definici, kterou považuji za dostatečně srozumitelnou a ucelenou.

Podle Václava Řepy lze podnikový proces popsat následovně: „Jednoduše řečeno, podnikový proces je souhrnem činností, transformujících souhrn vstupů do souhrnu výstupů (zboží nebo služeb) pro jiné lidi nebo procesy, používajíce k tomu lidi a nástroje. Všichni to děláme, přičemž jednou jsme v pozici zákazníka, jindy zase dodavatele.“ [1] Vizualizaci této definice si můžete prohlédnout na obrázku 2.1.



Obrázek 2.1. Model podnikového procesu [1]

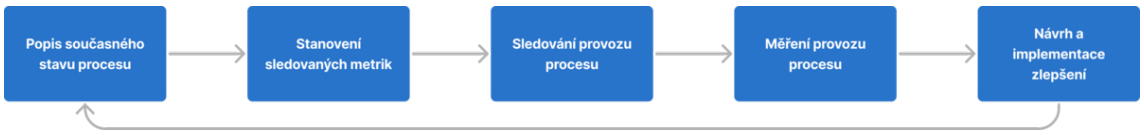
Podnikové procesy lze dále dělit na tři kategorie: [2]

- Provozní procesy - Klíčové procesy, které přináší hodnotu (např. zákaznické objednávky. výroba komponent)
- Řídící procesy - Dohlíží na provozní procesy (např. řízení společnosti, rozpočtový dohled)
- Podpůrné procesy - Podporují provozní procesy (např. technická podpora, účetnictví)

2.2 Zlepšování procesů

Vzhledem ke globálnímu vysoce konkurenčnímu prostředí tržní ekonomiky vzniká tlak na podniky své procesy neustále zlepšovat. Jestliže například konkurenční firma pracuje efektivněji, je schopna produkty vyrábět levněji a zákaznicí přirozeně přejdou k ní. Firmy se tak stále snaží své procesy analyzovat a vylepšovat. Obvykle je nežádoucí provádět radikální změny, a proto firmy přistupují spíše k postupné evoluci. Té je docíleno porozuměním a měřením stávajícího procesu a hledání cest k jeho vylepšení. [1]

Obecný postup zlepšování procesů zachycuje obrázek 2.2. Je důležité zdůraznit zpětnou vazbu mezi posledním a prvním krokem. Zlepšování procesů je neustálý iterativní proces, který se opakovaně provádí, aby umožnil podnikům se neustále zlepšovat. Z této filozofie vychází i tzv. PDCA cyklus, kterým se zabírá následující podkapitola.



Obrázek 2.2. Zlepšování podnikových procesů [1]

2.2.1 PDCA cyklus

PDCA cyklus, známý také jako Demingův cyklus, je metoda pro neustálé zlepšování procesů. Tato metoda byla původně navržena W. Edwardsem Demingem, americkým statistikem. PDCA cyklus se skládá ze čtyř základních kroků: Plánování (Plan), Provedení (Do), Kontrola (Check) a Akce (Act). Níže naleznete krátký popis každého kroku:

Plan (Plánování): V prvním kroku se zaměřujeme na identifikaci problémů a možností ke zlepšení v současných procesech. Je důležité pečlivě analyzovat procesy, sbírat data a vyhodnotit současný stav. Na základě analýzy a zhodnocení se navrhují zlepšovací opatření a stanovují cíle pro jejich realizaci.

Do (Provedení): V druhém kroku se implementují navržená řešení a zlepšovací opatření. Je nezbytné provést změny v procesech, zaškolit zaměstnance a zajistit, aby byly nové postupy náležitě zavedeny do praxe.

Check (Kontrola): Třetí krok spočívá v měření a hodnocení výsledků zavedených změn a zlepšovacích opatření. Je třeba sledovat a vyhodnocovat, zda navržená řešení přinesla očekávané výsledky a zda byly splněny stanovené cíle. Pokud je to možné, je vhodné porovnat výsledky s předchozím stavem procesů.

Act (Akce): Poslední krok zahrnuje přijetí nezbytných kroků k dalšímu zlepšení procesů na základě zjištěných výsledků. Pokud byly výsledky úspěšné, lze nové postupy standardizovat a rozšířit na další oblasti podnikání. Pokud výsledky nebyly dostačující, je třeba provést další analýzu a navrhnout nová zlepšovací opatření.

PDCA cyklus je iterativní proces, který se neustále opakuje, což umožňuje organizacím postupně zlepšovat své procesy a dosahovat vyšší úrovně kvality. [3]

2.2.2 As-is a To-be stav

V kontextu zlepšování procesů se často používají pojmy „as-is“ (stav, jaký je) a „to-be“ (stav, jaký má být). Tyto pojmy jsou klíčové pro analýzu a zlepšování podnikových procesů, protože umožňují porovnat současný stav procesů s jejich požadovaným stavem po provedení změn a optimalizací.

As-is: Tento pojem se používá k popisu současného stavu podnikového procesu. Při analýze as-is stavu se zkoumá, jak probíhá daný proces v reálném prostředí podniku, jaké jsou jeho vstupy a výstupy, jaké činnosti a úkoly se provádějí, jaké nástroje a zdroje se používají a jaká je role jednotlivých aktérů. Cílem analýzy as-is stavu je identifikovat slabiny a nedostatky procesu, které mohou být zlepšeny.

To-be: Tento pojem odkazuje na požadovaný stav podnikového procesu po provedení změn a optimalizací. To-be stav reprezentuje vizi, jak by měl proces fungovat efektivněji. Při modelování to-be stavu se vytváří návrh nového procesu, který zohledňuje zjištěné nedostatky a slabiny as-is stavu a navrhuje řešení pro jejich odstranění. To-be stav tak slouží jako cílová podoba procesu, která by měla vést k lepšímu výkonu a konkurenceschopnosti podniku.

Proces zlepšování podnikových procesů často začíná analýzou a modelováním as-is stavu, který následuje identifikací možností pro zlepšení a nakonec navrhování a implementace to-be stavu. Tímto způsobem lze systematicky a promyšleně zlepšovat podnikové procesy a dosahovat trvalých výsledků. [4]

2.3 Modelování procesů

Z výše uvedeného vyplývá, že procesy je nutné nějakým způsobem zaznamenat pro účely analýzy a zlepšování. Tato činnost se nazývá modelováním procesů. Pod tím si lze představit formalizovaný záznam aktivit a rozhodnutí vykonávaných v rámci procesu, kdy o něm zároveň získáváme další informace jako jsou jeho vstupy, výstupy, podprocesy, atp.

Při modelování procesů se postupuje na základě textových popisů nebo informací shromážděných z různých částí podniku. Je nutné zaručit, že tato vstupní data jsou správná a aktuální. Vzhledem k tomu, že procesy jsou často složité a obsáhlé, nejvhodnějším způsobem záznamu je grafický model. Ten je snadněji pochopitelný pro širší škálu lidí a názornější než pouhý textový popis.

V praxi existuje několik grafických notací, ze kterých lze vybírat.

2.4 Orchestrace a automatizace procesů

Orchestrace procesů označuje snahu o koordinaci aktivit podnikového procesu a jejich vzájemného spojování. Orchestrace procesů pomáhá pracovat s lidmi, systémy a zařízeními, které jsou součástí procesu se snahou dosáhnout co největší automatizace (tj. minimalizovat množství práce vykonané lidmi). [5]

Příkladem může být eshop většího podniku, kde si pod orchestrací procesu objednávky lze představit koordinaci aktivit mezi obchodním oddělením, skladem, skladovým systémem, dopravcem a CRM.

2.5 Grafické notace

Existuje více notací a grafických jazyků, kterými lze procesy zaznamenávat. V následujících odstavcích představím vybrané notace, které považuji za nejvhodnější.

2.5.1 Petriho sítě

Petriho síť je matematický a grafický nástroj pro modelování a analýzu systémů, které se skládají z diskrétních událostí a jejich vzájemných vztahů. Skládají se z míst, přechodů a orientovaných hran, které propojují místa s přechody a naopak. Místa reprezentují stavy systému, zatímco přechody reprezentují události nebo činnosti. Tokeny v místech modelují dostupné zdroje nebo stav systému v daném čase.

Petriho sítě tedy lze použít k vizualizaci a analýze podnikových procesů, které se skládají z úkolů, jejich vzájemných závislostí a prostředků potřebných k jejich realizaci. Rovněž umožňují modelovat paralelní i sekvenční provádění úkolů, což je klíčové pro modelování složitějších podnikových procesů. Navíc umožňují zkoumat vlastnosti procesů, jako je například slabá ukončení, která zaručují, že proces může vždy dosáhnout konečného stavu.

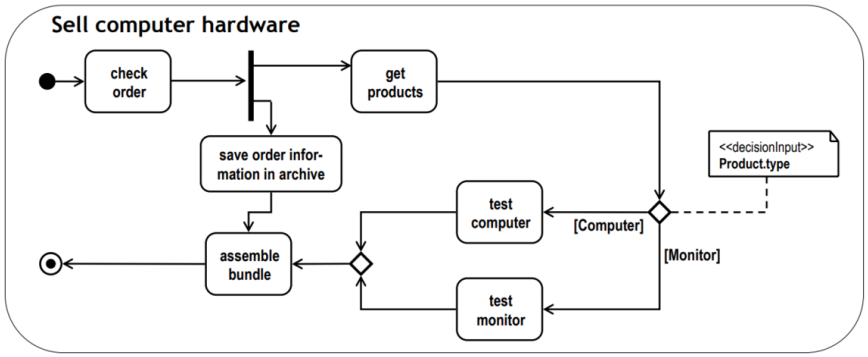
Použití Petriho sítí pro modelování business procesů přináší řadu výhod, mezi něž patří možnost formální analýzy, simulace a optimalizace procesů. Díky formálnímu základu Petriho sítí mohou být identifikovány potenciální problémy, jako jsou závislosti na prostředcích nebo blokující situace. [6]

2.5.2 Unified Modeling Language (UML)

UML je grafický jazyk pro vizualizaci, specifikaci, konstrukci a dokumentaci dílčích částí softwarového systému. UML poskytuje standardní způsob zápisu návrhů systému, který zahrnuje koncepční věci, jakými jsou například právě podnikové procesy nebo funkce systému, ale i konkrétní věci, jako jsou třídy zapsané v určitém programovacím jazyce, databázová schémata a opakovaně použitelné softwarové komponenty. [7] Kořeny UML sahají až do roku 1994, od roku 2005 se však téměř výhradně používá verze UML 2 a jejich podverze.

V kontextu modelování procesů hovoříme o takzvaných UML procesních diagramech. Ty jsou modelovány jako diagram toku (flowchart) jednotlivých událostí v procesu, doplněné o pomocné konektory pro rozhodování, paralelní zpracování a podobně.

Na obrázku 2.3 je znázorněn příklad procesu „Prodej počítačového hardwaru“. Proces začíná ověřením objednávky, poté se současně zajistí produkty a uloží informace o objednávce. Po získání produktů se provede testování jejich funkčnosti podle typu produktu. Pokud testy proběhnou úspěšně a informace o objednávce jsou uloženy, balíček se sestaví a proces je ukončen.



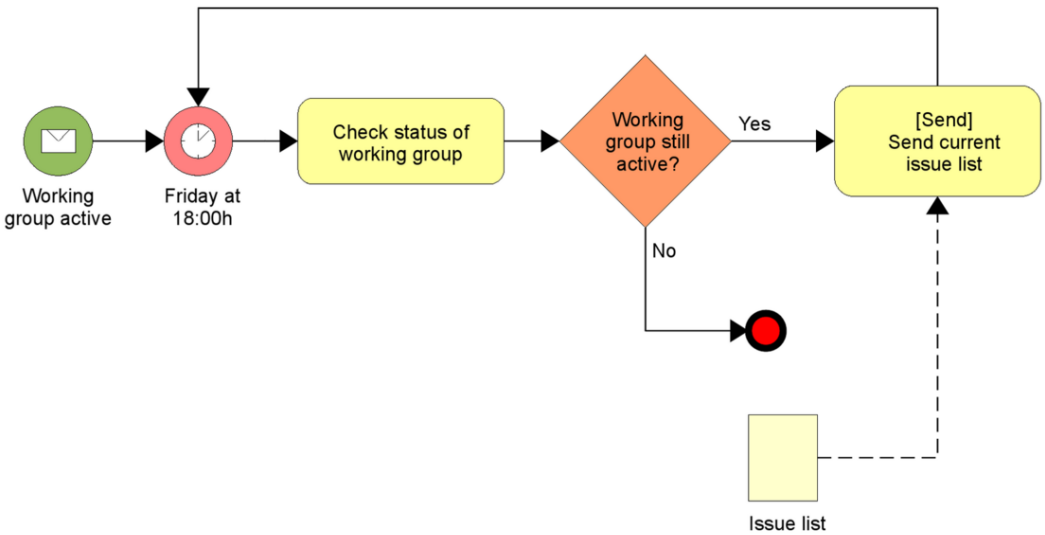
Obrázek 2.3. Příklad modelu UML [8]

2.5.3 Business process modelling notation (BPMN)

BPMN je vyvíjeno od roku 2004 a po několika revizí se od roku 2014 prosazuje verze 2.0.2. Notace je použitelná výhradně pro účely procesního modelování, kdy zachycuje celý proces jako tok aktivit (flowchart) přes všechny možné scénáře a události. Vyznačuje se svou jednoduchostí a snadnou čitelností pro všechny, kdo se jakkoli účastní práce s procesy, tj. od samotných architektů procesů přes vývojáře po lidi, kteří proces řídí a dozorují. [1]

Oproti UML umožňuje přehlednější záznam vnějších aktivit a vlivu časovaných událostí.

Na obrázku 2.4 je zobrazen příklad diagramu v notaci BPMN. Proces se spustí, pokud je pracovní skupina aktivní. Jestliže je skupina aktivní, každý pátek v 18:00 proběhne kontrola aktivity skupiny. Pokud skupina není aktivní, proces končí. V případě, že je skupina stále aktivní, odesílá se e-mail se seznamem aktuálních problémů a proces čeká na další páteční kontrolu.



Obrázek 2.4. Příklad modelu BPMN [9]

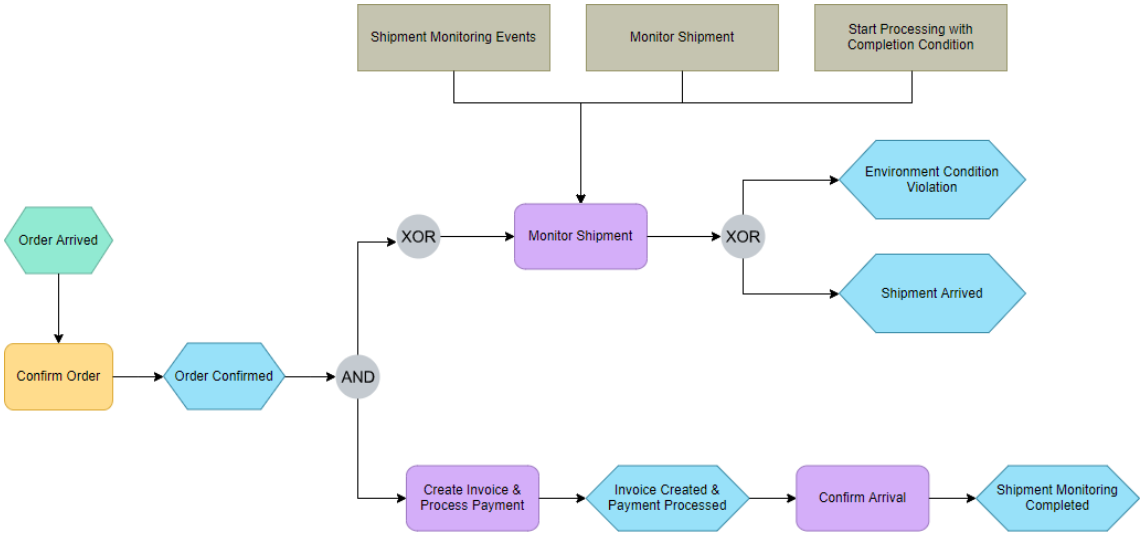
2.5.4 Architecture of Integrated Information Systems (ARIS)

Metodu ARIS vyvinul prof. Dr. Augustem-Wilhelmem Scheerem jako referenční architekturu informačního systému, na kterou se dívá ze tří pohledů. Tím prvním je pohled na organizaci tzv. podnikový pohled, druhým je pohled z perspektivy informačních systémů a technologií, posledním je informační a datový model podniku. Tyto tři části jsou spolu úzce propojeny. [1]

ARIS poskytuje své softwarové nástroje, které mimo jiné umožňují právě procesní modelování. V poslední cloudové verzi podporuje jak vlastní notaci EPC (Event-driven process chain) tak notaci BPMN. Rozdíly a vhodnost těchto dvou notací jsou předmětem vášnivých debat.

EPC byla vyvinuta spolu s ARIS a jak již z názvu vypovídá, stojí na principu uspořádaného grafu událostí (events) a k nim přiřazených funkcí. Spolu s tím poskytuje různě konektory pro větvení, paralelní zpracováni, logické funkce a další. [10] Dá se říct, že EPC zaznamenává změnu stavů procesu v reakci na události.

Příklad EPC diagramu naleznete na obrázku 2.5. Tento proces popisuje objednávku. Jakmile je objednávka přijata, je potvrzena a následně probíhají dvě současné činnosti. První spočívá v neustálém sledování objednávky, zatímco druhá zahrnuje vytvoření faktury, zpracování platby a následné potvrzení doručení.



Obrázek 2.5. Příklad notace EPC [10]

2.6 Volba notace

V rámci předmětu Procesní řízení, pro jehož potřeby je systém primárně vyvíjen, je vyučována notace BPMN, jakožto nejlepší volba pro modelování procesů. [11] Z tohoto důvodu je vyvíjený testovací systém postaven právě na ověřování znalostí této notace. Podrobněji je ji věnována následující kapitola 3.

Kapitola 3
Business process modelling notation (BPMN)

3.1 Historie

BPMN bylo původně vyvinuto organizací Business Process Management Initiative (BPMI), která v květnu 2004 zveřejnila původní verzi 1.0. V červnu 2005 se BPMI sloučila s OMG (Object Management Group), která spravuje i jazyk UML. V únoru 2006 vydala OMG specifikační dokument BPMN. V roce 2010 byla vytvořena verze BPMN 2.0 a aktuální verze specifikace byla vydána o tři roky později v prosinci 2013. Nejnovější verze (2.0.2) byla oficiálně zveřejněna organizací ISO jako standard vydání ISO/IEC 19510:2013. V průběhu posledních BPMN nabíralo na popularitě a stal se z něj v podstatě standard pro procesní modelování. [12]

3.2 Typy diagramů

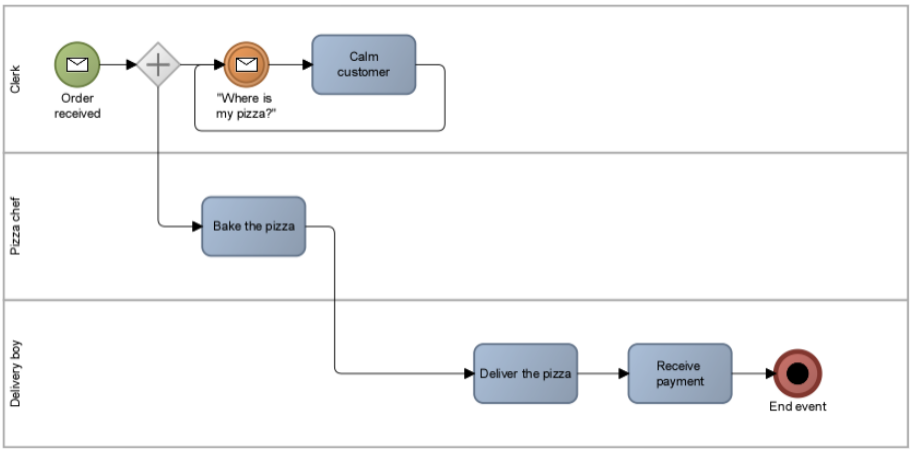
Notace podporuje 3 základní typy standardizovaných diagramů. [13] Pro účely vyvíjeného systému je pro nás nejdůležitější první typ procesního diagramu, na nějž bude zaměřeno ověřování dovedností. Zbylé dva doplňuji pro větší přehlednost.

3.2.1 Procesní diagram

Procesní diagram zachycuje posloupnost a podmínky provedení aktivit prováděných v rámci jednoho procesu spolu s možnými výjimkami. K popisu účastníků, kteří jsou zodpovědní za provádění daných aktivit, lze využít tzv. dráhy (lanes). Účastníkem může být osoba, ale i systém nebo oddělení. Jeho jméno je obvykle uvedeno v levé části pruhu. Použití drah není povinné. [13]

Podobně jako dráhy, některé další elementy jsou volitelné. BPMN je volná notace, nejedná se tedy o metodiku.

Příklad procesního diagramu zachycující proces objednávky dovozu pizzy naleznete na obrázku 3.1.



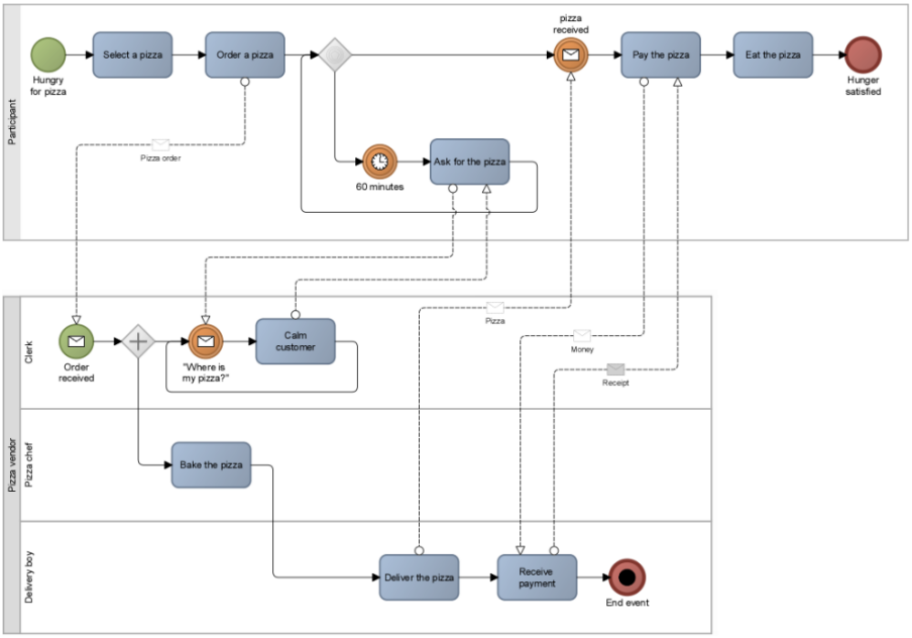
Obrázek 3.1. BPMN - Procesní diagram [13]

3.2.2 Diagram spolupráce

Diagram spolupráce názorně zobrazuje účastníky procesu a jejich interakce, čímž zachycuje spolupráci mezi nimi.

Struktura diagramu spolupráce je podobná procesním diagramům. Stejně jako v procesních diagramech, i zde jsou účastníci spolupráce reprezentováni pomocí drah (lanes). Avšak v kontextu spolupráce jsou účastníci vizualizováni jako podčásti obdélníkového prostoru, ve kterém se proces nachází (pool). Bazén (pool) může, stejně jako dráha, reprezentovat organizační jednotku, roli nebo oddělení. [13]

Rozšířený diagram procesu objednávky dovozu pizzy nalznete na obrázku 3.2. Ten zachycuje spolupráci mezi hladovým objednavatelem a restaurací.

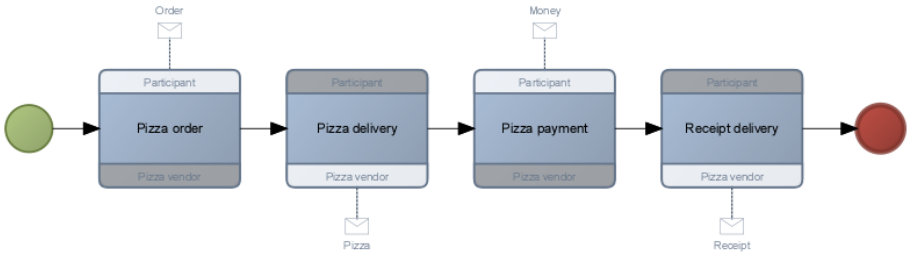


Obrázek 3.2. BPMN - Diagram spolupráce [13]

3.2.3 Diagram choreografie

Zatímco předchozí typy popisují sekvenci aktivit prováděnou účastníky, diagram choreografie vizualizuje způsob, jakým účastníci koordinují své interakce. Nezachycuje úkoly prováděné účastníky, ale popisuje výměnu informací mezi nimi. Diagram zajišťuje, že výměna zpráv je prováděna uspořádaným způsobem. [13]

Na obrázku 3.3 naleznete diagram choreografie pro proces objednávky doručení pizzy. Diagram znázorňuje způsoby komunikace mezi objednavatelem a restaurací.



Obrázek 3.3. BPMN - Diagram choreografie [13]

3.3 BPMN formát a objekty

Dle standardu je BPMN použitelné výhradně na modelování procesů, ostatní typy diagramů, jako např. datové modely, organizační struktury a podobné nejsou podporovány.

BPMN definuje pět hlavních typů objektů, ze kterých jsou vzájemným propojováním komponovány diagramy. Jsou jimi: [14]

- Tokové objekty (flow objects) - Hlavní chování procesů (události, aktivity, brány)
- Data
- Spojovací objekty (connecting objects) - Spojují prvky do grafu
- Dráhy (lanes) a bazény (pools) - Seskupují prvky v procesu
- Artefatky (artifacts) - Dodatečně informace

Konkrétní specifikace je vydávaná standardizační společností OMG a je volně dostupná `online`[1] .

3.4 Nástroje pro práci s BPMN

Vzhledem k širokému rozšíření notace BPMN existuje spousta nástrojů, které jej podporují.

[1] `https://www.omg.org/spec/BPMN/2.0/PDF`

V následujících podkapitolách představím několik nástrojů pro práci s BPMN, které jsem rozdělil do dvou kategorií a to na diagramové editory a systémy pro orchestraci a automatizaci procesů. Toto dělení jsem vymyslel sám, aby bylo snazší porozumět rozmanitosti nástrojů a jejich funkcím.

3.4.1 Diagramové editory

V této podkapitole se budeme zabývat BPMN diagramovými editory, které představují klíčovou kategorii nástrojů pro práci s BPMN notací. Tyto editory umožňují návrh, tvorbu a úpravu BPMN diagramů, které slouží k vizualizaci a dokumentaci podnikových procesů. V následujícím textu stručně představím několik mnou vybraných nástrojů, které po zkoumání považuji za nejlepší. Podmínkou výběru byla jejich dostupnost v bezplatné verzi.

bpmn-js

BPMN-js je open-source knihovna pro práci s BPMN 2.0. Tato knihovna umožňuje vytváření, zobrazování, editaci a uložení BPMN modelů v HTML5 a JavaScriptovém prostředí. Za vývojem a údržbou knihovny bpmn-js stojí společnost Camunda.

Knihovna bpmn-js je založena na technologiích jako JavaScript, SVG a CSS, což umožňuje její snadnou integraci do webových aplikací. Díky svému modulárnímu designu je knihovna rozšiřitelná a lze ji přizpůsobit konkrétním potřebám uživatelů. [15].

Reprezentativní distribuce knihovny bpmn-js je k dispozici přímo od vývojářů na oficiálním webu BPMN.io.

BPMN-js knihovna je velmi oblíbená mezi vývojáři aplikací (v době psaní práce přes 7100 Github stars) a je využívána v mnoha komerčních i nekomerčních projektech.

Diagrams.net

Diagrams.net je k dispozici jako bezplatný open source nástroj, který nabízí uživatelům široké spektrum možností pro tvorbu různých typů diagramů, jako jsou např. BPMN, UML, ER a mnoho dalších. Vyznačuje se nejen možností práce online i offline, ale také silným zaměřením na soukromí uživatelů. Kromě toho umožňuje spolupráci na diagramech v reálném čase a nabízí rozsáhlou škálu možností pro export diagramů. [16]

3.4.2 Systémy pro orchestraci a automatizaci procesů

Camunda Platform

Camunda Platform je software pro orchestraci podnikových procesů. Poskytuje nástroje pro modelování BPMN umožňující jednoduchý design procesů. Dále obsahuje nástroje pro automatizaci a monitorování procesů nehledě na to, zda jejich aktivity vykonávají lidi, systémy nebo zařízení. V poslední řadě pro získání detailního přehledu jsou dostupné analytické nástroje pro dosažení optimalizace podnikových procesů. Jedná se o komerčně vyvíjenou platformu od společnosti Camunda. [17]

Bizagi Platform

Bizagi Platform je cloudová platforma pro nízko-kódovou automatizaci procesů. Skládá se ze tří základních nástrojů. Prvním z nich je Bizagi Modeler, který slouží k modelování procesů právě v notaci BPMN. Bizagi Studio pak poskytuje rozhraní pro interaktivní tvorbu a správu procesů, automatizací a aplikací. Nakonec Bizagi Automation slouží pro provoz a monitorování vytvořených objektů.

Platforma poskytuje širokou škálu předpřipravených integrací pro další aplikace. [18]

IBM Business Process Manager

IBM Business Process Manager (IBM BPM) je software pro správu a automatizaci podnikových procesů, který je vyvíjen společností IBM. Ten získal nástroj akvizicí společnosti Lombardi [19]. Tento nástroj umožňuje organizacím modelovat, provádět, monitorovat a optimalizovat podnikové procesy, aby zlepšily efektivitu a zvyšovaly konkurenceschopnost na trhu. IBM BPM k popisu procesů využívá právě notace BPMN. [20]

IBM BPM nabízí širokou škálu funkcí, které usnadňují práci s BPMN. Některé z těchto funkcí zahrnují:

- Modelování procesů
- Validace a simulace
- Automatizace procesů
- Monitorování a řízení
- Kolaborace

3.5 Závěr

V této kapitole jsme se seznámili s historií a základními koncepty BPMN notace, která se stala standardem pro modelování podnikových procesů. Pro práci s BPMN existuje řada nástrojů, které se dělí do diagramových editorů a nástrojů pro orchestraci a automatizaci procesů. Pro účely mé práce je nejdůležitější knihovna bpmn-js, která umožňuje snadnou integraci do webových aplikací a jejíž vývoj stojí zastřešuje společnost Camunda. Tato knihovna bude hrát klíčovou roli ve zbytku mé práce, kde se budu dále odkazovat na její funkce a možnosti.

Kapitola 4
Výuka a výukové systémy

Z kapitoly číslo 2 vyplývá, že procesní řízení a modelování procesů hrají klíčovou roli ve vývoji a řízení podnikových strategií. Jedním z hlavních nástrojů pro modelování těchto procesů je notace BPMN popsaná v kapitole 3. Výuka BPMN je proto zásadní pro rozvoj dovedností potřebných pro efektivní řízení a optimalizaci podnikových procesů a možná i proto byla vybrána jako součást výuky předmětu Procesní řízení.

V této kapitole se zaměřím na výukové systémy pro notaci BPMN a představím výsledky mého bádání, které se zabývá zkoumáním a analýzou existujících výukových systémů stejně tak jako přístupu k výuce předmětu Procesní řízení. Tato studie staví na předešlých akademických pracích s příbuznou tématikou.

4.1 Současný přístup k výuce

V současné době se výuka procesního řízení zaměřuje na teoretické i praktické aspekty této disciplíny. Výuka probíhá v několika formách, zahrnující přednášky, cvičení a testy, aby studenti získali co největší množství znalostí a dovedností v oblasti procesního řízení a modelování BPMN.

Přednášky

Na přednáškách se studenti seznamují s teoretickým základem procesního řízení a modelování BPMN. Přednášky pokrývají široké spektrum témat, jako jsou základní principy procesního řízení, role a úkoly zainteresovaných stran, metody a nástroje používané pro modelování a analýzu procesů. Zároveň se studenti seznámí s BPMN a jeho základními prvky, které jim umožní efektivně modelovat procesy.

Cvičení

Cvičení mají za cíl prakticky procvičit dovednosti a znalosti získané na přednáškách. Během cvičení se studenti učí mimo jiné modelovat procesy pomocí BPMN a získávají zkušenosti s praktickou aplikací teoretických znalostí. Cvičení probíhá tak, že lektor poskytne zadání s konkrétním scénářem či problémem, který studenti mají vymodelovat pomocí BPMN. Studenti tak trénují své dovednosti a získávají zkušenosti s modelováním různých typů procesů a situací.

Testy

Testování je způsob, jak ověřit, zda studenti zvládli teoretické znalosti a praktické dovednosti potřebné pro efektivní modelování procesů pomocí BPMN. Testy probíhají

tak, že lektor poskytne zadání, které studenti musí vypracovat pomocí nástroje bpmn.io. Výsledkem je BPMN diagram, který by měl odpovídat zadanému scénáři či problému a je zaslán lektorovi emailem k vyhodnocení. Lektor pak hodnotí správnost a úplnost vytvořených diagramů, což pomáhá studentům zlepšovat své dovednosti a pochopení procesního řízení a BPMN modelování.

Výuka procesního řízení je tak komplexním procesem, který kombinuje teoretické a praktické aspekty této disciplíny. Studenti mají možnost nejen rozšiřovat své teoretické znalosti, ale také získávat praktické dovednosti a zkušenosti s modelováním procesů pomocí BPMN. Tento přístup umožňuje studentům lépe pochopit důležitost procesního řízení, jeho význam pro organizace a efektivní řízení jejich činností. Navíc, díky praktickému cvičení a testování, se studenti stávají kompetentnějšími v oblasti BPMN modelování, což jim pomáhá při řešení reálných problémů a výzev v praxi.

4.2 Výukové systémy

V rámci diplomové práce jsem se zaměřil na nalezení aplikace, která by byla primárně navržena pro výuku BPMN, spíše než na její použití v komerčním prostředí jako nástroj pro orchestraci procesů. Cílem bylo najít výukovou platformu, která by mohla usnadnit a automatizovat proces výuky procesního řízení a modelování BPMN, jak byl popsán v předchozí kapitole 4.1.

Výzkum zahrnoval zkoumání různých dostupných systémů a nástrojů, které by mohly být použity pro výuku BPMN, a hodnocení jejich schopnosti poskytovat efektivní výukové prostředí pro studenty, které by zároveň usnadnilo a zefektivnilo proces hodnocení a zpětné vazby ze strany lektorů.

Metodikou zkoumání bylo vyhledávání na internetu pomocí klíčových slov, jako jsou „bpmn teaching platform“, „bpmn exercise system“ a další související fráze, které by mohly vést k nalezení vhodné výukové platformy pro BPMN. Vyhledávání zahrnovalo prozkoumávání odborných článků, blogů a dalších zdrojů informací, které by mohly poskytnout návod na vhodný systém nebo nástroj.

Součástí procesu byla také intenzivní diskuse s vedoucím práce, který je zároveň garantem předmětu a přirozeně má v oblasti BPMN a výukových systémů pro procesní řízení značný přehled. Tato spolupráce pomohla identifikovat klíčové vlastnosti a funkce, které by měl ideální výukový systém pro BPMN obsahovat, a poskytla další směry pro vyhledávání a hodnocení potenciálních řešení. Intenzivní diskuse s ním umožnila získat cenné informace a zkušenosti.

Navzdory pečlivému hledání a intenzivní snaze jsem nebyl schopen najít výukovou platformu pro BPMN, která by alespoň částečně splňovala požadavky. Během výzkumu jsem sice narazil na několik interaktivních kurzů a online materiálů, což ukazuje, že takové systémy skutečně existují a jsou využívány. Avšak, žádný z nalezených nástrojů nebyl speciálně navržen pro univerzitní prostředí a potřeby lektorů a studentů při výuce procesního řízení a modelování BPMN.

4.3 Předchozí práce

Na fakultě elektrotechnické vzniklo několik akademických prací zabývající se tématikou notace BPMN. V následujících odstavcích je krátce představím včetně toho, jak na ně navazuje tato práce.

4.3.1 Simulátor modelování procesů v notaci BPMN

„Tato bakalářská práce se zabývá procesním řízením a modelováním procesů v notaci BPMN. Základem je seznámení se s pojmem procesu a s aspekty procesního řízení. Dále je definovaná oblast modelování procesu a následně je zmíněná notace BPMN. Součástí práce je popis činnosti a nezbytných dovedností procesního analytika a souhrn možností jeho vzdělávání. V další části práce je představen návrh a realizace aplikace, která umí ověřovat schopnost uživatele porozumět slovnímu popisu procesu a následně ho převádět do podoby modelu v notaci BPMN. Poslední části práce obsahuje vyhodnocení jejích výsledků pomocí testování a krátký popis jejího možného budoucího rozvoje. Výslednou aplikaci lze využít při výuce procesního řízení.“ [21]

Z této práce vychází přístup k řešení testů v podobě vypracování BPMN diagramu z textového zadání. Rovněž jsem čerpal inspiraci z použití knihovny bpmn.js jako nástroj pro kreslení diagramů.

4.3.2 Aplikace pro ověřování dodržování standardů notace BPMN

„Tato bakalářská práce se zabývá procesním řízením, modelováním procesů v notaci BPMN a jejich následnou analýzou. V první částí je definována teorie související s procesem a procesním řízením. V části následné je popsán základní úvod do problematiky modelování procesů, dál v části navazující uveden popis notaci BPMN a jsou krátce zmíněny zodpovědnosti procesního analytika. Součástí práce je také rozbor nejčastějších chyb procesních analytiků, které v další části práce je využito pro návrh a implementace aplikace, schopnou tyto chyby detekovat a upozornit uživatele na nich. Poslední část práce obsahuje výsledky uživatelského testování této aplikace.“ [22]

Tato práce slouží jako jistý teoretický podklad pro systém automatického vyhodnocování testů, protože validace bude jednou ze součástí hodnocení. Autor zde definuje několik validačních pravidel, které lze strojově ověřovat. Ačkoli jeho práci nelze fyzicky přepoužít, jedná se o zdroj inspirace.

4.3.3 Virtuální příprava procesních analytiků

„Tato diplomová práce se zabývá procesním řízení, jeho výukou a přípravou procesních analytiků. V první části práce je definováno procesní řízení a teorie s ním související. Popsány jsou rovněž různé oblasti procesní analýzy. Následně jsou popsány činnosti procesního analytika a kompetence nutné k vykonávání této profese. V návaznosti na tyto kompetence je uveden výčet způsobů, pomocí kterých je možné je posbírat a představen možný plán výuky pro začínající procesní analytiky. Jeho součástí je nová aplikace, která je popsána v další části. V posledních dvou částech je popsán vývoj a testování této aplikace.“

Tato diplomová práce představuje fundamentální základ mé vlastní práce, neboť se zaměřuje na vývoj systému určeného pro trénink procesních analytiků. Autor se ve svém výzkumu věnovala řešení několika klíčových pilotních problémů, například jak efektivně implementovat kreslení BPMN diagramů v rámci vlastní aplikace. Má práce tedy nepřímo navazuje na tuto práci, s cílem zobecnit její účel pro výuku studentů předmětu Procesní řízení.

4.4 Závěr

Na základě analýzy současných nástrojů pro modelování BPMN procesů a přehledu existujících akademických prací lze konstatovat, že žádný z dostupných nástrojů není dostatečně ucelený a přizpůsobený specifickým potřebám výuky předmětu procesní řízení. Tento fakt podtrhuje nutnost vývoje nového nástroje, který by byl šitý na míru pro výuku modelování BPMN procesů a zohledňoval specifika výukového prostředí, jako je sledování výsledků studentů a automatické vyhodnocení jejich práce.

Vývoj takového nástroje představuje hlavní cíl této diplomové práce. Navazuje na předchozí akademické práce a jejich poznatky v oblasti tréninku procesních analytiků a implementace BPMN diagramů ve vlastních aplikacích. Vytvořením uceleného a intuitivního nástroje pro výuku modelování BPMN procesů se otevírá cesta k zefektivnění procesu výuky a zvýšení kvality vzdělání budoucích absolventů předmětu.

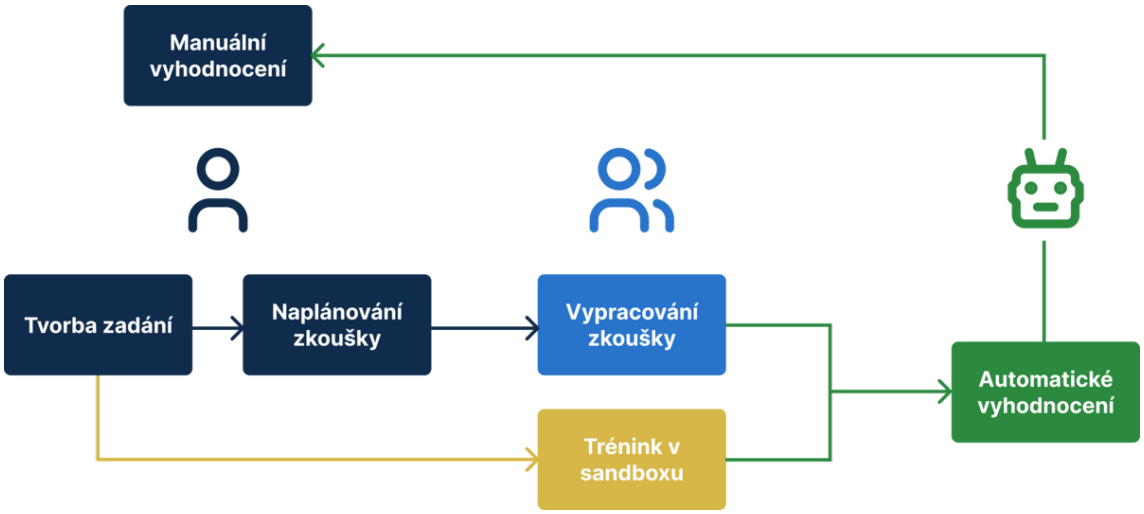
Kapitola 5
Návrh systému

V předchozí kapitole 4 jsem identifikoval potřebu vytvoření a vývoje nového systému pro účely výuky předmětu procesní řízení. Tato kapitola se zaměřuje na návrh tohoto systému, přičemž se budu věnovat nejen softwarovým požadavkům, ale také rozsahu použití, systémovým požadavkům a případům užití.

Kromě toho se budu zabývat vizuálním designem aplikace, aby byla přehledná, intuitivní a přitažlivá pro uživatele. Mým cílem je vytvořit efektivní nástroj, který podpoří jak studenty, tak pedagogy v procesu výuky procesního řízení. V průběhu této kapitoly rovněž prozkoumám jednotlivé aspekty návrhu systému a předložím řešení, která zohledňují potřeby a očekávání uživatelů.

5.1 Rozsah použití

Systém je navržen tak, aby umožňoval snadné testování znalostí studentů z notace BPMN. Lektor tvoří zadání, textový popis procesu, který mají studenti za úkol zpracovat jako BPMN diagram. Systém následně provede automatické vyhodnocení diagramu oproti referenčně zpracovanému zadání zadání a kontrolu anti-patternů. Diagram je následně ještě podroben manuální kontrole a hodnocení. Systém dále umožní zkoušet cvičná zadání.



Obrázek 5.1. Rozsah použití

5.2 Systémové požadavky

Následující sekce definuje požadavky, které musí systém splňovat ze softwarového hlediska.

5.2.1 Funkční požadavky

Funkční požadavky jsou definovány následovně:

FR1 - Registrace uživatele Umožňuje uživatelům vytvořit nový účet v aplikaci a získat přístup na základě registračního kódu.

FR2 - Přihlášení uživatele Umožňuje uživatelům přístup k funkcím dostupným pouze registrovaným uživatelům.

FR3 - Správa tříd Systém bude umožňovat tvorbu a mazání tříd. Každá třída bude opatřena kódem, který studenta při registraci přiřadí do dané třídy.

FR4 - Správa zadání Systém bude umožňovat tvorbu, mazání a aktualizaci zadání. Ty se budou skládat z textového popisu a referenčního řešení. Zadání bude rovněž možné označit jako cvičné.

FR5 - Správa zkoušek Systém bude umožňovat tvorbu, mazání a aktualizaci zkoušek. Každá zkouška bude mít přidělený interval dostupnosti, časový limit a sadu zadání, z které se bude losovat. Lektor bude mít možnost zobrazit a manuálně kontrolovat odevzdané zkoušky.

FR6 - Sandbox Systém bude umožňovat neomezené vyplňování a automatického vyhodnocování cvičných zadání.

FR7 - Podstoupení zkoušky Systém bude umožňovat běžným uživatelům účastnit se zkoušek v dostupném termínu. Po vyplnění proběhne automatické vyhodnocení.

FR8 - Manuální hodnocení zkoušky Systém bude umožňovat oprávněným uživatelům manuálně ohodnotit zkoušku body a přidat komentář.

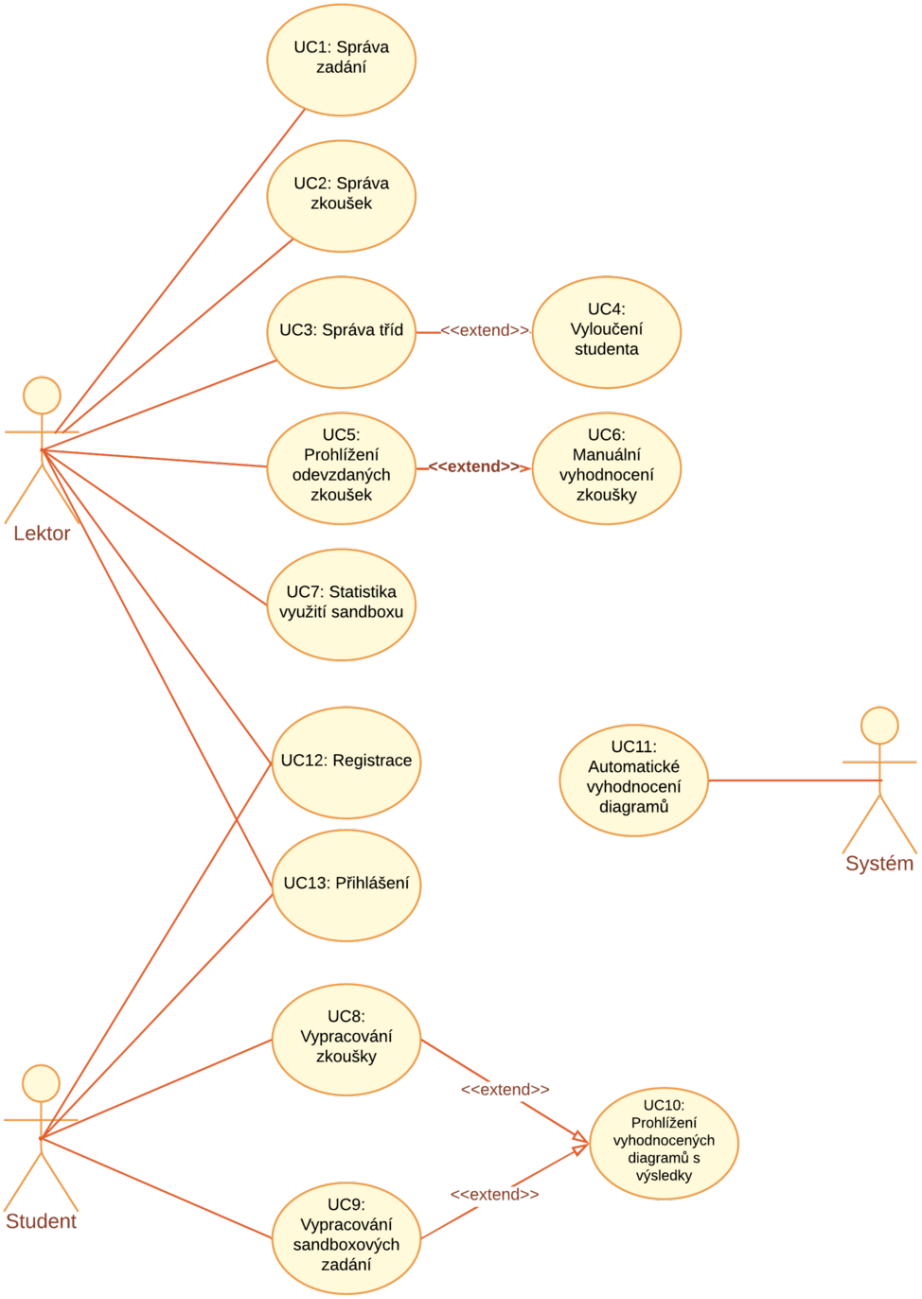
FR9 - Automatické vyhodnocení Systém bude automaticky vyhodnocovat odevzdané zkoušky a pokusy v sandboxu srovnáním s referenčním řešením. Zároveň bude detekovat typické BPMN anti-patterny.

5.2.2 Nefunkční požadavky

NFR1 - Platformní nezávislost Systém bude designován nezávisle na použitém operačním systému a verzi prohlížeče.

NFR2 - Podpora desktopových zařízení Systém bude designován tak, aby byl použitelný na všech běžných dekstopových rozlišení.

5.3 Případy užití



Obrázek 5.2. Use Case Diagram

UC1 - Správa zadání Lektor může vytvářet, upravovat a mazat zadání. Zadání se skládá z textového popisu a referenčního řešení. Každé zadání může být označeno jako sandboxové.

UC2 - Správa zkoušek Lektor může vytvářet, upravovat a mazat zkoušky. Zkouška se skládá z přidělených zadání, tříd a intervalu dostupnosti.

UC3 - Správa tříd Lektor může vytvářet a mazat třídy. Každá třída se skládá z názvu, kódu a studentů.

UC4 - Vyloučení studenta Lektor může vyloučit studenta ze třídy.

UC5 - Prohlížení odevzdaných zkoušek Lektor může prohlížet odevzdané diagramy studentů seskupené podle zkoušek.

UC6 - Manuální vyhodnocení zkoušky Lektor může provést manuální vyhodnocení odevzdaného diagramu studenta.

UC7 - Statistika využití sandboxu Lektor může zobrazit jmennou statistiku využití sandboxového zadání.

UC8 - Vypracování zkoušky Student může vypracovat a odevzdat zkoušku otevřenou v daném časovém intervalu.

UC9 - Vypracování sandboxových zadání Student může neomezeně vypracovávat a odevzdávat sandboxové zadání.

UC10 - Prohlížení odevzdaných diagramů s výsledky Student může prohlížet své odevzdané ohodnocené diagramy.

UC11 - Automatické vyhodnocení Systém automaticky vyhodnotí odevzdaný diagram na základě nastavených pravidel a anti-patternů.

UC12 - Registrace Uživatel se může za pomocí kódu zaregistrovat do systému. Na základě kódu jsou jsou mu přiřazeny role a případně třída.

UC13 - Přihlášení Uživatel se může přihlásit do systému pomocí přihlašovacích údajů.

5.4 Vizuální zpracování

5.4.1 Uživatelské rozhraní klientské aplikace

Struktura uživatelského rozhraní klientské aplikace, kterou lze vidět na obrázku 5.3, se skládá ze tří stěžejních statických komponent a z dynamického obsahu, který se mění v závislosti na aktuálním výběru uživatele. Statické komponenty zahrnují záhlaví, menu a autentizaci.

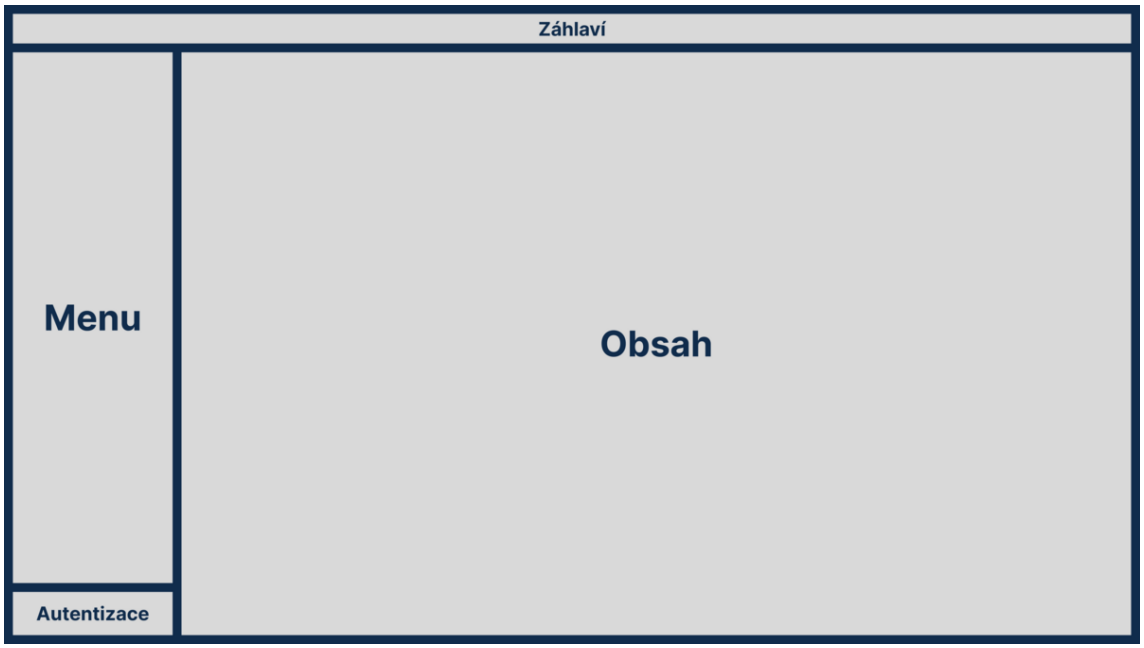
Záhlaví je umístěno v horní části aplikace a má převážně estetický význam. Jeho účelem je představit vizuální identitu aplikace, zobrazit logo a poskytnout jednotný vizuální prvek napříč celou aplikací.

Menu se nachází v bočním panelu aplikace a slouží k navigaci mezi různými částmi aplikace. Umožňuje uživatelům snadno přepínat mezi jednotlivými sekcemi.

Autentizační část je zodpovědná za zobrazení aktuálně přihlášeného uživatele a poskytování možnosti odhlášení.

Dynamický obsah aplikace se nachází v hlavním prostoru aplikace a zobrazuje aktuálně vybranou sekci. Tento obsah se mění v závislosti na tom, jakým směrem se uživatel naviguje prostřednictvím menu a dostupných funkcí. Dynamický obsah umožňuje uživatelům interakci s funkcemi aplikace a zobrazuje informace relevantní pro aktuální kontext.

Celkově základní struktura uživatelského rozhraní klientské aplikace zajišťuje jednoduchou a intuitivní navigaci, zatímco současně poskytuje konzistentní vizuální zážitek napříč celou aplikací.



Obrázek 5.3. Wireframe uživatelského rozhraní rozhraní

5.4.2 Název a vizuální identita

Vizuální identita vyvíjené aplikace je klíčovým prvkem, který ovlivňuje uživatelský zážitek a vizuální vnímání aplikace. Tato identita zahrnuje několik prvků, jako jsou název, logo, ikony a barevná paleta, které společně vytvářejí jednotný a konzistentní vizuální jazyk.

Pro navrhovaný systém jsem zvolil jméno Gamajun, inspirované slovanskou mytologií. Gamajun je v těchto pověstech zobrazován jako moudrý a prorocký pták s lidskou tváří, který slouží jako posel bohů a šiřitel znalostí a moudrosti mezi lidmi. [23] Toto jméno bylo zvoleno pro tento systém, aby zdůraznilo jeho schopnost poskytovat uživatelům hluboké porozumění.

Logo aplikace představuje vizuální symbol aplikace a je umístěno v záhlaví nebo na úvodní obrazovce aplikace. Jeho účelem je poskytnout uživatelům okamžitou identifikaci aplikace a zároveň zdůraznit její profesionální vzhled. Logo si můžete prohlédnout na obrázku 5.4.

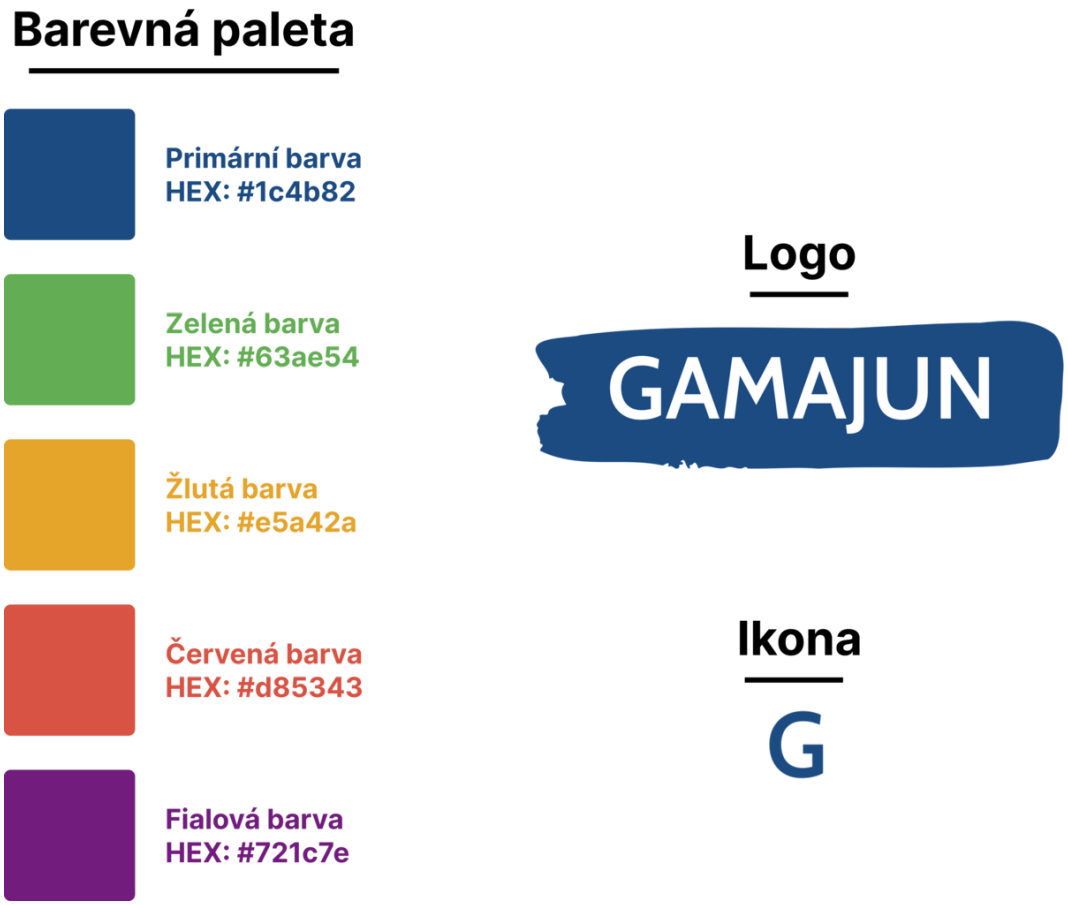
Ikony jsou důležitým prvkem vizuální identity, který usnadňuje navigaci a orientaci v aplikaci. Ikony by měly být navrženy tak, aby byly jednoduché, snadno rozpoznatelné

a konzistentní s celkovým vizuálním jazykem aplikace. Jejich použití zvyšuje srozumitelnost a efektivitu uživatelského rozhraní.

Barevná paleta je soubor pečlivě vybraných barev, které se používají v rámci webové aplikace pro dosažení konzistentního a atraktivního vizuálního vzhledu. Vývojáři webových aplikací vytvářejí barevné palety, aby zajistili harmonii mezi prvky uživatelského rozhraní, podpořili identitu aplikace a zvýšili použitelnost a přístupnost aplikace. Tím posilují celkovou estetiku webového designu.

Barevná paleta je tak zásadním prvkem identity aplikace, který se prolíná celou aplikací a pomáhá identifikovat a rozlišovat jednotlivé prvky. Například zelená barva může symbolizovat úspěch, novou entitu a uložení změn, což uživatelům poskytuje vizuální zpětnou vazbu o provedených akcích. Na druhou stranu červená barva může být použita pro nebezpečné akce, jako je mazání nebo zrušení, čímž upozorňuje uživatele na potenciální rizika spojená s těmito akcemi.

Barevnou paletu využitou při návrhu uživatelského rozhraní vyvíjené aplikace naleznete rovněž na obrázku 5.4.



Obrázek 5.4. Identita

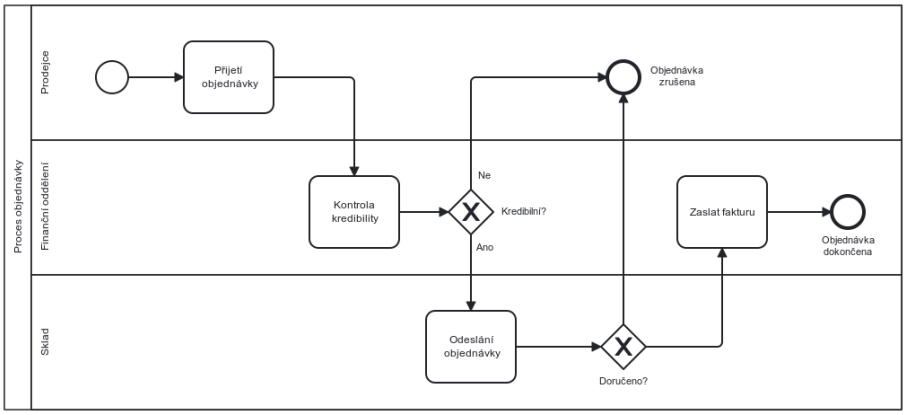
Kapitola 6
Automatické vyhodnocení

Jedním z klíčových požadavků na vyvíjenou aplikaci je co možná největší míra automatizace vyhodnocení odevzdaných řešení. Ta si klade za cíl ušetřit čas lektorovi a zároveň poskytnout okamžitou zpětnou vazbu studentům.

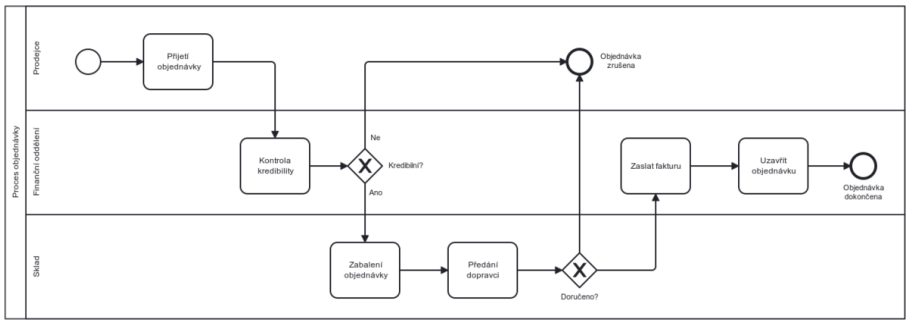
Z technického hlediska se jedná o poměrně složitý problém. Ačkoli lektor při tvorbě zadání vytváří i referenční řešení, nelze vyhodnocení provést jako prosté porovnání obou dvou diagramů. Notace BPMN je z principu volná a stejného výsledku, který je správně, lze dosáhnout více cestami, což ilustrují diagramy 6.1 a 6.2.

Tyto dva příklady popisují stejný proces, ale každý z nich používá jiný přístup k jeho popisu. V řešení A je proces zjednodušen a zahrnuje pouze krok „Odeslání objednávky“, zatímco v řešení B je tento krok rozdělen na dva samostatné kroky: „Zabalení objednávky“ a „Předání přepravci“. Navíc v řešení B je po „Zaslání faktury“ ještě přidán další krok, kterým je „Uzavření objednávky“. I přes tyto odlišnosti oba diagramy popisují v podstatě totožný proces.

Pokud bychom měli modelovou situaci, kde řešení A je referenční a řešení B studentské, pouhým porovnáním obou diagramů bychom nedosáhli shody, i když v podstatě oba popisují stejný proces.



Obrázek 6.1. BPMN diagram objednávky - řešení A



Obrázek 6.2. BPMN diagram objednávky - řešení B

Z předchozího příkladu je zřejmé, že prosté srovnání diagramů jeden ku jednomu není dostatečné pro správné vyhodnocení jejich shody. Tento problém vyžaduje sofistikovanější přístup. Proto byl vyvinut víceúrovňový systém, který umožňuje automaticky vyhodnotit odevzdané diagramy s větší jistotou. Sestává se ze dvou komponent:

- Validátor
- Referenční srovnání

6.1 Validátor

Fakt, že BPMN je spíše volnou notací, vede k vyššímu riziku chyb při tvorbě diagramů. I když některé zjevné nedostatky, jako umístění prvků přes sebe nebo nepřipojené konektory, eliminuje grafický editor diagramů, stále existuje mnoho možných syntaktických, sémantických a pragmatických chyb. Tyto nevhodné modelovací přístupy jsou obecně označovány jako anti-patterny.

Pro lepší klasifikaci chyby dělíme na:

- Syntaktické chyby - Nesprávné použití BPMN notace
- Sémantické chyby - Nesprávný význam diagramu
- Pragmatické chyby - Nesrozumitelné nebo nadbytečné použití notace

Při vytváření validátoru jsem se inspiroval studií T. Rozmana a G. Polančiče nazvanou „Analysis of most common process modelling mistakes in BPMN process models“, která se zaměřuje na analýzu nejběžnějších chyb studentů informačních systémů při tvorbě BPMN diagramů. V rámci této práce autoři na základě analýzy více než dvou tisíc diagramů identifikovali 15 nejčastějších anti-patternů. Tyto anti-patterny jsem převzal a zaměřil jsem se na ty, které mohou nastat. Pokud je možné se jich v grafickém editoru dopustit, automaticky je detekuji v odevzdaných diagramech.

Student je okamžitě po odevzdání seznámen s výsledkem a tak okamžitě ví, zda udělal chybu, jež je popsána, případně názorně vysvětlena. Lektorovi pak validátor usnadní hodnocení obecné znalosti BPMN.

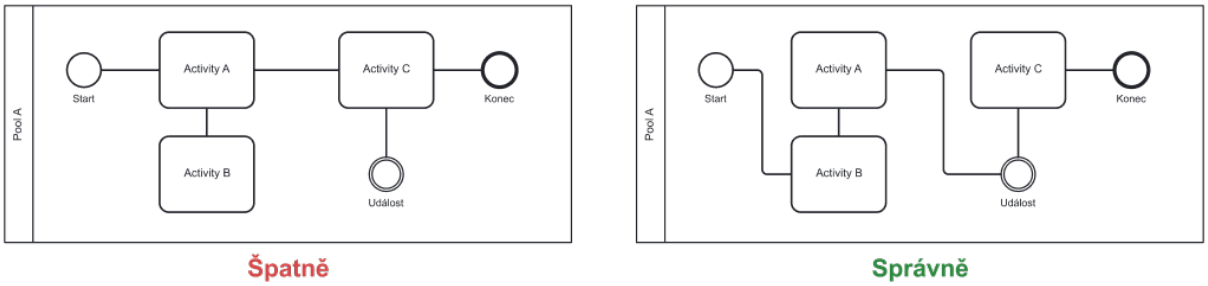
Tím, že jsou kontrolována přesně definovaná pravidla na úrovni jednotlivých objektů nebo menších skupin a není třeba nijak interpretovat logický význam diagramu, je tento systém relativně spolehlivý a riziko falešné pozitivity i negativity je malé.

Ačkoli validátor ověří znalost syntaxe notace BPMN, nedokáže vyhodnotit, zda student namodeloval proces, který byl zadán.

6.1.1 Příklad: Anti-pattern - zavěšení události nebo aktivity

Pokud v diagramu existuje aktivita nebo událost, která má odchozí hranu, ale nemá hranu příchozí, označujeme ji jako zavěšenou. Tento stav představuje sémantickou chybu, protože taková aktivita nebo událost se nikdy nemůže uskutečnit. Pro vyřešení této situace je potřeba přeorganizovat graf tak, aby daný objekt měl příchozí hranu. [24]

Tento stav pak strojově detekujeme tak, že procházíme všechny aktivity a události v diagramu a kontrolujeme, zda mají odpovídající počet vstupních a výstupních hran.



Obrázek 6.3. Anti-pattern zavěšení

6.2 Referenční srovnání

Cílem tohoto modulu je vyhodnotit, zda odevzdaný diagram logicky odpovídá textovému popisu procesu. Ke srovnání slouží referenční řešení vypracované lektorem. Jak již bylo zmíněno, diagramy nelze primitivně porovnat. Tento problém má několik možných řešení, které níže nastíním.

6.2.1 Omezení dostupných BPMN objektů

Za předpokladu, že by došlo ke stanovení množství a typů BPMN objektů, z kterých může student při vypracování daného zadání vybírat, zvýší se značně pravděpodobnost, že výsledný diagram bude shodný s referenčním.

Toto však přináší zásadní nevýhodu v podobě jisté návodnosti v přístupu k řešení. Jednak dojde k omezení kreativity studentů, za druhé se sníží obtížnost, a celkově by se z myšlenky „jak vymodelovat tento proces“ stalo „jak lektor vymodeloval tento proces“. Toto celé za cenu toho, že výsledné řešení nemusí vždy úplně odpovídat, jak ilustruje obrázek níže.

Z výše zmíněných důvodů jsme po konzultaci s vedoucím tento přistup zavrhli.

6.2.2 Strojové učení

Protože se moje odborné pole působnosti zaměřuje na softwarové inženýrství a ne na umělou inteligenci, prezentovaný model je spíše hypotetický a nebyl jsem schopen jej implementovat. Nicméně se domnívám, že by bylo možné vytvořit umělou inteligenci, která by se učila na datech sestávajících z diagramů popisujících stejné procesy různými způsoby. Takový model by poté získal schopnost pochopit význam diagramů a teoreticky by byl schopen určit správnost libovolného řešení na základě referenčního zadání. Nepodařilo se mi však dohledat žádnou existující datovou sadu takového typu a to považuji za klíčový problém tohoto přístupu. Přesto se mi zdá, že tento problém by mohl být zajímavým tématem pro budoucí výzkum v této oblasti.

6.2.3 Redukce na graf

Odmyslíme-li si některé části BPMN notace (např. Bazény), můžeme libovolný BPMN diagram převést na orientovaný cyklický graf. Jestliže je graf vymodelovaný dle specifikací, je tento graf i spojitý. Teoreticky tak můžeme zredukovat problém podobnosti dvou BPMN diagramů na hledání izomorfismu dvou grafů.

Z definice plyne, že dva orientované grafy jsou izomorfní, pokud existuje bijekce mezi jejich množinami vrcholů taková, že zachovává směr a počet hran mezi nimi. Izomorfismus orientovaných grafů je tedy ekvivalencí mezi nimi a ukazuje, že mají stejnou strukturu a vlastnosti.

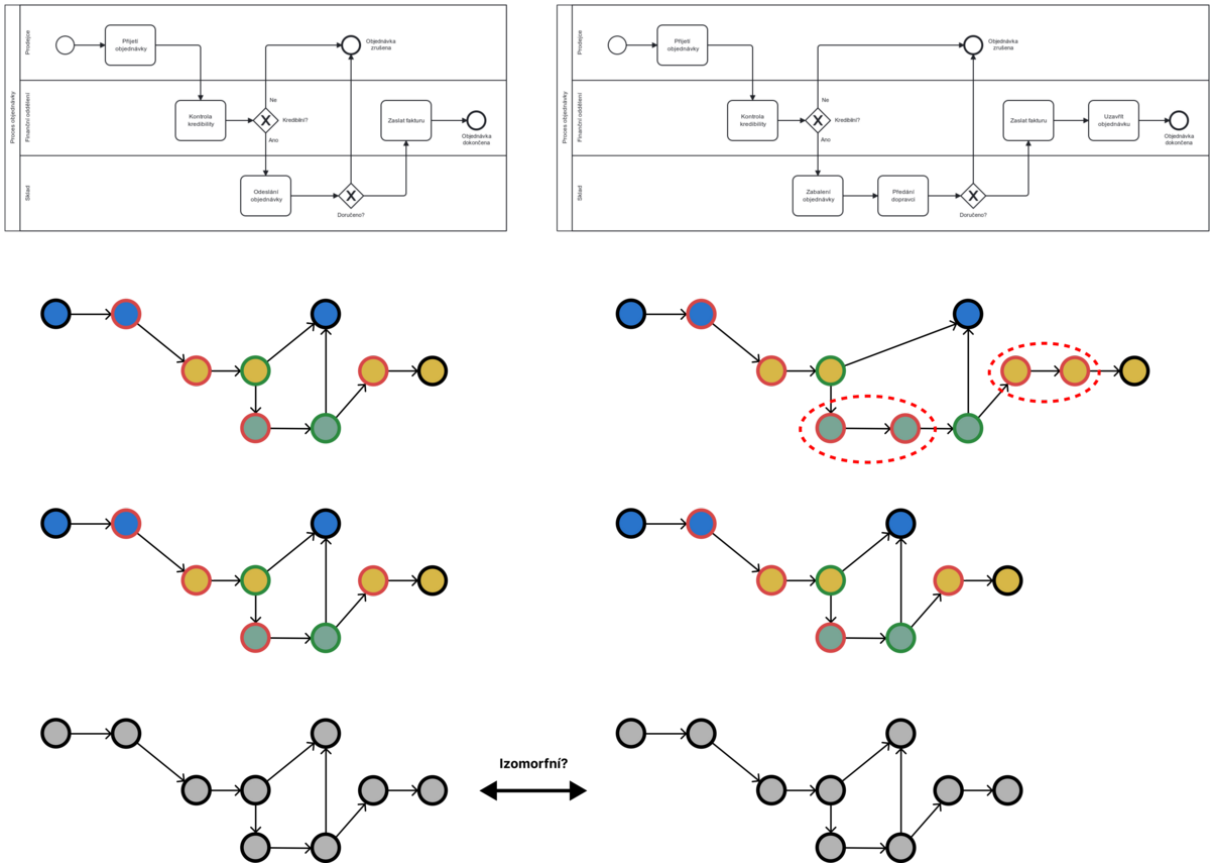
Jsme tedy schopni na graf převedený diagram porovnat, problém s rozdílností grafů však nadále zůstává. Můžeme se ho však snažit minimalizovat redukční operací. Tu provedeme na referenčním i studentském řešení.

Sloučení po sobě jdoucích aktivit v rámci bazénu

Po sobě jdoucí aktivity můžeme sloučit do jedné. Při modelování procesu můžeme aktivity modelovat rozdílnou granularitou. Ta nijak neovlivní správnost diagramu ale komplikuje srovnání. Sloučíme-li v rámci grafu všechny přímo navazující aktivity do jedné, zbavíme se se potencionálních rozdílností.

Celkově tento přístup nedokáže s jistotou garantovat správnost rozhodnutí, avšak je to to nejlepší, s čím jsem přišel. Falešná pozitivita je spíše nepravděpodobná, avšak vzhledem k širokým možnostem přístupu k modelování nelze falešnou negativitu vyloučit.

Vizualizaci algoritmu pro referenční srovnání nalzente na obrázku 6.4.



Obrázek 6.4. Ukázka algoritmu referenčního srovnání

Známé problémy

Jedním z případů, kdy tento přístup nebude fungovat je referenční řešení, které obsahuje dvě po sobě jdoucí aktivity. V případě, že student vymodeluje pouze jednu a zbytek grafu je v pořádku, algoritmus i v referenčním řešení provede redukci a označí diagramy za shodné.

Z výše uvedeného vyplývá, že plně automatické vyhodnocení je komplexní výpočetní problém. Z důvodu toho, že systém není schopen garantovat správnost ohodnocení, finální rozhodnutí činí lektor. Práci má však značně ulehčenou validátorem, který za něj detekuje anti-patterny a referenčním srovnáním, které v případě pozitivní shody s vysokou jistotou garantuje správnost řešení. V praxi se tak lektor musí většinou zabývat zejména řešením s negativní shodou.

Kapitola 7
Technická analýza

Následující kapitola se věnuje technickému aspektu vyvíjeného systému. Popisuje návrh architektury a obsahuje výčet zvolených technologií.

7.1 Architektura systému

Navrhovaný systém využívá typickou architekturu klient-server. V kontextu vývoje webové aplikace je architektura klient-server často používaným přístupem k vytváření internetových aplikací. V této architektuře je webová aplikace rozdělena na dvě hlavní části, klienta a server.

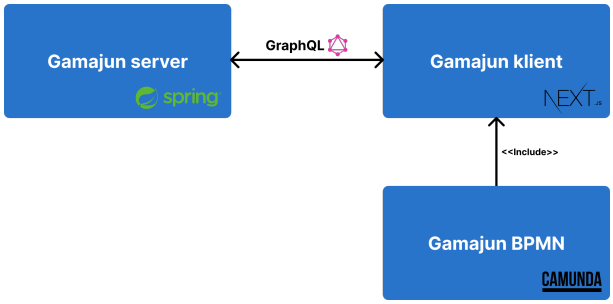
Klient je část systému, který běží v prohlížeči uživatele a zajišťuje zobrazení uživatelského rozhraní a interakci s uživatelem. Klient obvykle komunikuje se serverem pomocí HTTP požadavků a odpovědí.

Server je část sytému, který běží na vzdáleném počítači (serveru) a zajišťuje zpracování požadavků od klientů. Server často pracuje s databází nebo jinými externími systémy a poskytuje klientovi odpovědi v požadovaném formátu, jakým je v našem případě JSON. [25]

Serverová část vyvíjené aplikace je postavena na technologii Spring Frameworku a je programována v Javě. Klient je potom Single-Page webová aplikace postavená na frameworku Next.js. Nabízí uživateli veškerou potřebnou interakci se systémem. Přihlášení probíhá pomocí technologie OAuth2 oproti vyvíjenému serveru.

Pro vytváření a zobrazování BPMN diagramů systém využívá vlastní distribuci bpmn-js od firmy Camunda. Tato distribuce byla upravena tak, aby poskytovala jak editor diagramů, tak prohlížeč diagramů pro potřeby systému, a je distribuována jako NPM balíček.

Grafické znázornění architektury lze vidět na obrázku 7.1.



Obrázek 7.1. Architektura systému

7.1.1 Architektura serveru

Serverová aplikace využívá vrstvenou architekturu. Jedná se o běžný přístup k návrhu a implementaci Java aplikací. Vrstvená architektura umožňuje oddělit různé vrstvy aplikace tak, aby byla snadněji spravovatelná, škálovatelná a udržovatelná. Každá vrstva se soustředí na specifické úkoly a komunikuje s ostatními vrstvami přes definovaná rozhraní, většinou však pouze s vrstvami sousedícími. [26]

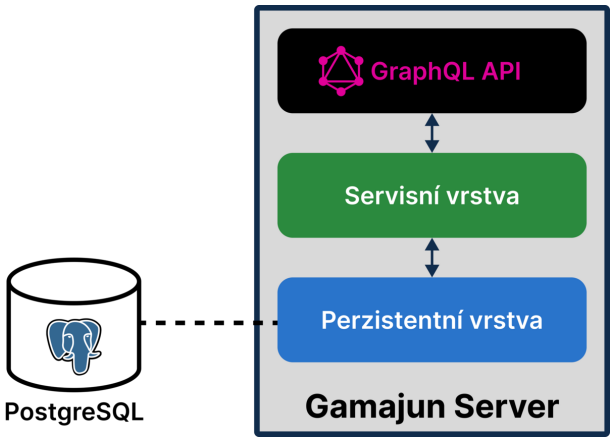
Konkrétně pak aplikace obsahuje tyto tři vrstvy:

Perzistentní vrstva, která se stará o uchovávání dat na databázovém serveru. Tato vrstva se stará o tzv. objektově relační modelování (ORM), což je přístup k mapování objektově orientovaného programování na relační databáze. ORM umožňuje programátorům pracovat s databázovými tabulkami pomocí objektů a metod, místo aby psali přímé SQL dotazy.

Servisní vrstva je zodpovědná za implementaci business logiky aplikace. To znamená, že pracovává požadavky od prezentační vrstvy a využívá datovou vrstvu k práci s daty.

Prezentační vrstva převádí data a objekty mezi aplikačním rozhraním a aplikační vrstvou, a to včetně formátování, validace a zpracování uživatelských interakcí. V našem případě se jedná o vrstvu, která zpracovává požadavky, které dorazí na GraphQL aplikační rozhraní.

Vizualizaci architektury serveru si lze prohlédnout na obrázku 7.2.



Obrázek 7.2. Architektura serveru

7.1.2 Architektura klienta

V kontextu vyvíjené klientské aplikace nelze vzhledem ke zvoleným technologiím mluvit o jedné konkrétní architektuře jako spíše o kombinaci více vývojových patternů a principů. V následujícím textu jsem vybral mezi použitými ty nejdůležitější.

Stránky a komponenty - Rozdělení webové aplikace na stránky a komponenty. Stránky jsou základní jednotkou webové aplikace a představují samostatné zobrazení obsahu. Každá stránka má své vlastní URL, na které je přístupná, a může být navigována z jedné stránky na druhou. Stránky obvykle obsahují komponenty a jsou

odpovědné za vykreslování celého zobrazení. Komponenty jsou menší jednotky webové aplikace, které tvoří jednotlivé části stránky. Tyto komponenty mohou být použity na více stránkách a jsou znovupoužitelné v různých kontextech. Komponenty by měly být co nejvíce nezávislé na stránkách a co nejjednodušší a snadno použitelné. Komponenty lze dále zanořovat s dalšími komponenty k tvorbě komplexnějších komponent.

Data fetching - Vývojový pattern používaný pro získávání dat z externích zdrojů, jako jsou aplikační rozhraní nebo databáze. Tento pattern je založen na principu asynchronního načítání dat, aby se minimalizoval vliv na výkon a user experience.

Automatické přednačítání stránek - Umožňuje webovým aplikacím předem načítat stránky, které budou pravděpodobně navštíveny v blízké budoucnosti. Tento pattern vylepšuje user experience a snižuje čas, který je nutný k načtení při navigaci aplikací.

Stavové proměnné - Stavové proměnné jsou proměnné, které určují, jak se má komponenta zobrazovat. Jsou použity k řízení chování komponenty a jsou aktualizovány v reakci na uživatelské akce nebo jiné události.

Klient-side navigace - Vývojový pattern, který se používá pro implementaci navigace ve webových aplikacích. Tento pattern umožňuje dynamické zobrazování obsahu na stránkách bez nutnosti měnit celý obsah, místo toho se pouze načte část stránky, která se liší.

7.2 Volba technologií

Následující podkapitola se zabývá rozborem konkrétních použitých technologií.

7.2.1 Serverová část

Serverová část je programována v jazyce Java za použití Spring frameworku. K této volbě jsem dospěl zejména z toho důvodu, že kromě toho, že jde o osvědčené řešení, tak se v rámci fakulty jedná o nejvíce používaný framework a aplikaci může snadno převzít a udržovat jiný vývojář.

Spring Framework je aplikační open-source framework, který poskytuje komplexní infrastrukturu pro podnikové aplikace založené na jazyce Java. Je navržen tak, aby pomáhal vývojářům vytvářet aplikace od základů a poskytoval širokou škálu funkcí a nástrojů pro vytváření, testování, nasazování a údržbu aplikací podnikové úrovně. Framework si zakládá na principech jednoduchosti, modularity a snadného použití. Mezi klíčové vlastnosti frameworku patří podpora dependency-injection, kontejner pro správu JavaBeans a flexibilní konfigurační systém. Je vhodný k vytváření široké škály aplikací, od jednoduchých webových aplikací až po složité podnikové systémy. [27]

V následujících odstavcích popíšu Spring moduly použité k tvorbě diplomové práce.

Spring Web MVC - Modul pro vývoj webových aplikací, který zahrnuje podporu pro REST API, Servlets a integraci s populárními webovými technologiemi, jako je například GraphQL. [28]

Spring Data - Modul, který poskytuje abstrakci a podporu pro práci s různými databázovými technologiemi, jako jsou relační databáze, NoSQL databáze nebo distribuované úložiště dat. [29]

Spring Security - Modul pro zabezpečení aplikací, který poskytuje řešení pro autentizaci, autorizaci a ochranu před běžnými bezpečnostními hrozbami, jako je například CSRF nebo SQL Injection. [30]

Spring for GraphQL - Modul poskytuje podporu pro integraci GraphQL API do Java aplikací, zjednodušuje vývoj a zajišťuje efektivní správu datových dotazů a manipulací. [31]

Spring Authorization Server - Modul pro řešení pro implementaci OAuth 2.0 a OpenID Connect autorizačního serveru, umožňující centralizovanou správu autentizace a autorizace pro webové a mobilní aplikace. [32]

Spring Resource Server - Poskytuje nástroje a podporu pro implementaci OAuth 2.0 chráněných zdrojových serverů, které umožňují zabezpečený přístup k API a zdrojům aplikace na základě udělených tokenů a oprávnění. [33]

Pro práci s BPMN diagramy byla využita knihovna **Camunda Model API**, která poskytuje nástroje pro objektové zpracování a manipulaci s procesními modely. Umožňuje vývojářům snadno číst, vytvářet, upravovat a validovat BPMN soubory, což zjednodušuje integraci těchto modelů do softwarových projektů. Camunda Model API a knihovna Bpmn.io umožňují import a manipulaci s BPMN diagramy ve formátu XML, což zjednodušuje interoperabilitu a integraci mezi oběma nástroji. Výhodou XML je i možnost uchovávat jej snadno v databázi. [34]

Pro práci s grafy aplikace využívá knihovnu **JGrapht**. Jedná se o flexibilní knihovnu, která poskytuje nástroje a algoritmy pro práci s grafy. Knihovna slouží ke zjednodušení a zefektivnění vývoje softwarových aplikací, které vyžadují manipulaci s grafy. JGrapht umožňuje vytvářet různé typy grafů, provádět grafové algoritmy (např. nejkratší cesta, minimální kostra, isomorfismy), analyzovat grafy a jiné další. [35]

7.2.2 Klientská aplikace

Klientská webová aplikace je postavena na knihovně React a je tedy programována primárně v JavaScriptu.

React je otevřená JavaScriptová knihovna vyvinutá a udržovaná společností Facebook, který ji i sám používá k vývoji uživatelských rozhraní webových a mobilních aplikací. Jde o deklarativní knihovnu založenou na komponentách, což umožňuje vývojářům snadno vytvářet modulární a znovupoužitelné UI prvky, které se automaticky aktualizují, když se mění stav nebo data. Díky virtual DOM (Document Object Model) a efektivnímu systému aktualizace je React schopen rychle a efektivně manipulovat s reálným DOM, což zlepšuje výkon aplikace. [36]

Vzhledem k tomu, že se jedná pouze o knihovnu, použití pro tvorbu webových aplikací vyžaduje další nástroje a konfiguraci.

Z tohoto důvodu je základem klientské aplikace framework **Next.js**. Jedná se o populární open-source framework postavený na JavaScriptovém běhovém prostředí prostředí Node.js. Umožňuje vytvářet výkonné a optimalizované webové aplikace založené na Reactu. Tento framework byl vyvinut a je udržován společností Vercel a nabízí řadu pokročilých funkcí. Obstarává nástroje a konfigurace potřebné pro chod Reactu a poskytuje další struktury, funkce a optimalizace pro hladký chod aplikace. [37]

Pro implementaci autentizace je využito open-source rozšíření NextAuth.js, které umožňuje snadnou implementaci OAuth2 autentizace do klientské aplikace. [38]

Značná část komponent, které vyvíjená aplikace vyžaduje, jsou standardní. Příkladem můžou být vstupní pole, kalendáře, modální okna, atp. Z tohoto důvodu se v praxi obvykle používá knihovna komponent. Ta usnadňuje a zrychluje tvorbu aplikace přepoužitím připravených komponent. V klientské aplikaci padla volba na knihovnu **Mantine**. Mantine je knihovna Reactových komponent poskytující kvalitní uživatelský, ale i vývojářský zážitek. Jedná se o poměrně novou, open-source knihovnu, která byla publikována v květnu roku 2021. Navzdory tomu, že se jedná o relativně novou knihovnu, obsahuje již velké množství kvalitních a designově líbivých komponent. [39]

Pro volání GraphQL API je použita knihovna **Apollo Client**, která zjednodušuje komunikaci mezi klientskou částí aplikace a GraphQL serverem. Apollo Client nabízí efektivní správu stavu, automatické cachování a normalizaci dat, což zvyšuje výkon aplikace a minimalizuje množství dotazů na server. [40]

Ačkoli přechozí text pojednává o jazyku Javascriptu, zdrojový kód je ve skutečnosti psán v TypeScriptu. **TypeScript** je nadstavba nad jazykem JavaScript, která přidává statické typování a další prvky, jež usnadňují práci s kódem a zvyšují jeho čitelnost a bezpečnost. TypeScript vyvinul a spravuje společnost Microsoft a poprvé byl představen v roce 2012. Přestože je založen na JavaScriptu, poskytuje vývojářům více nástrojů a možností, jak psát robustní a udržitelný kód.

TypeScript nabízí řadu výhod oproti čistému JavaScriptu. Jednou z klíčových výhod je statické typování, které umožňuje vývojářům deklarovat očekávané typy proměnných, funkcí a tříd. Díky statickému typování lze odhalit chyby dříve, než kód dosáhne produkčního prostředí, což zvyšuje kvalitu kódu a snižuje náklady na údržbu.

TypeScript také přispívá k zpřehlednění a čitelnosti kódu. Umožňuje používat moderní jazykové konstrukce, jako jsou třídy, rozhraní, moduly a dekorátory, které zjednodušují a zpřehledňují kód. To vede k lepšímu pochopení kódu ostatními vývojáři a snazší údržbě.

Kompatibilita s JavaScriptem je další významnou výhodou TypeScriptu. Jelikož je TypeScript nadstavbou nad JavaScriptem, existující JavaScriptový kód může být snadno převeden na TypeScript a postupně rozšířen o statické typování a další funkce. To usnadňuje migraci stávajících projektů na TypeScript a integraci s knihovnami a nástroji založenými na JavaScriptu.

TypeScript představuje výrazné vylepšení JavaScriptu, které usnadňuje práci vývojářům a zvyšuje kvalitu kódu. [41]

7.2.3 BPMN editor

Pro tvorbu BPMN diagramů v rámci klientské aplikace je použita vlastní distribuce knihovny BPMN.js vyvíjená společností Camunda jako open-source. Ta poskytuje plnohodnotný diagramový editor, jehož referenční verze je dostupná na stránce BPMN.io. Vlastní distribuce je použita z důvodů potřeby přizpůsobení funkcionalit této knihovny. Tuto vlastní distribuci označuji jako gamajun-bpmn. [42]

Detailní popis implementace práce s diagramy naleznete dále v kapitole 8.1.

7.2.4 GraphQL

GraphQL je dotazovací jazyk pro API a runtime pro provádění těchto dotazů nad daty. Byl vyvinut společností Facebook v roce 2012 a otevřeně zveřejněn v roce 2015. GraphQL slouží jako alternativa k REST architektuře pro vývoj API a umožňuje klientům žádat přesně ta data, která potřebují, zatímco server definuje dostupné datové typy a operace. [43] [44]

REST a GraphQL mají různé přístupy k práci s API. REST je založen na architektuře zdrojů, kde každý zdroj má svůj vlastní endpoint. Naopak GraphQL používá jediný endpoint, na kterém klienti provádějí dotazy a mutace. Co se týče protokolů, tak oba přístupy běžně používají HTTP pro komunikaci mezi klientem a serverem. [45]

Výhody GraphQL oproti RESTu zahrnují:

- Flexibilita - Klienti si mohou vyžádat přesně ta data, která potřebují, místo toho, aby byli omezeni pevně danými endpointy, jako je tomu u RESTu.
- Lepší výkon - GraphQL minimalizuje množství dat přenášených mezi klientem a serverem, což zlepšuje výkon aplikace a snižuje nároky na šířku pásma.
- Silný typový systém - GraphQL poskytuje silně typovaný jazyk, který umožňuje lepší kontrolu nad daty a usnadňuje detekci chyb.

Overfetching je problém, kdy klient získává více dat, než skutečně potřebuje. Tento problém se často vyskytuje u REST API, kde jsou data pevně vázaná na konkrétní endpointy. GraphQL řeší tento problém tím, že klientům umožňuje specifikovat, jaká data a jejich strukturu chtějí získat. Díky tomu klient získá pouze ta data, která potřebuje, což vede ke snížení zátěže na šířku pásma a zlepšení celkového výkonu aplikace. [45]

V GraphQL existují dva hlavní druhy operací: [46]

- Dotazy (Queries): Slouží k načítání dat z API. Dotazy jsou ekvivalentem HTTP GET v RESTu.
- Mutace (Mutations): Slouží k manipulaci s daty, jako je vytváření, aktualizace nebo mazání záznamů. Mutace odpovídají HTTP POST, PUT a DELETE v RESTu.

7.3 Závěr

V rámci kapitoly o technické analýze jsem se zaměřil na celkovou architekturu systému a použité technologie napříč celým projektem. Systém využívá moderní a špičkové technologie pro reaktivní webovou aplikaci, která zaručuje efektivitu a přívětivost pro uživatele. Tato analýza technických aspektů představuje základní kámen pro úspěšnou implementaci a nasazení systému, který je schopen splnit požadavky a zajistit vysokou úroveň výkonnosti a spolehlivosti.

Kapitola 8
Implementace

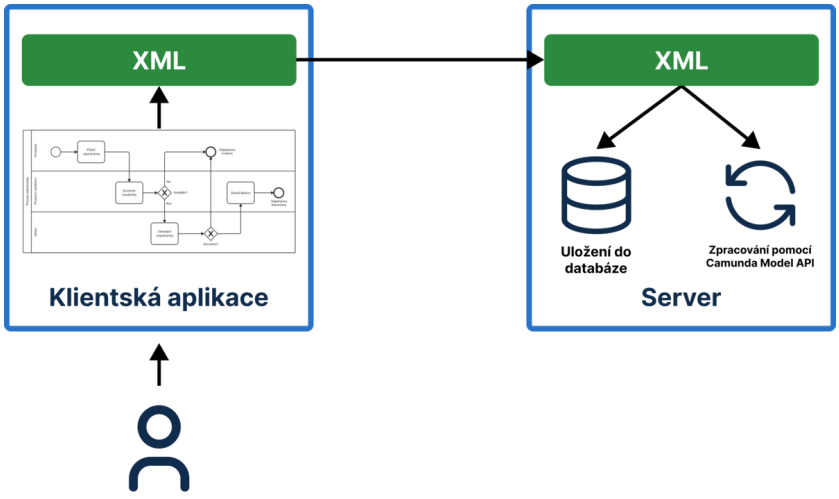
Následující kapitola popisuje nejzajímavější části implementace systému.

8.1 Práce s diagramy

V rámci této diplomové práce se přirozeně manipuluje s mnoha BPMN diagramy. Na straně klienta je implementována moje vlastní distribuce knihovny bpmn-js. Tvůrci této knihovny poskytují veřejně jak zdrojový kód, tak i návod pro vytvoření vlastní distribuce a možnost přímé konfigurace aktivovaných funkcí. V mém případě využívám jak prohlížeč pro zobrazení diagramů, tak jednoduchý editor pro jejich tvorbu a úpravy. Při odevzdávání řešení je BPMN diagram zasílán ve formě XML textového řetězce společně s identifikátorem odevzdání. Tyto údaje jsou následně zpracovány na serveru, kde je diagram po zpracování uložen do databáze ve formátu XML.

Na straně serveru je pro manipulaci s diagramy využívána knihovna Camunda Model API, která poskytuje objektové rozhraní pro práci s BPMN diagramy. Tato knihovna umožňuje načíst XML soubor obsahující BPMN diagram a vytvořit z něj instanci s API pro interakci s daným diagramem. Díky tomuto přístupu je manipulace s BPMN diagramy efektivní a snadná.

V následující podkapitole 8.2 se podrobněji zaměřím na jednu z konkrétních implementací s využitím knihovny Camunda Model API.



Obrázek 8.1. Práce s diagramy

8.2 Automatické vyhodnocení - Validátor

Kód na obrázku 8.2 demonstruje jedno z pravidel validátoru a to konkrétně kontrolu zavěšení. Na podobném principu jsou implementována všechna pravidla.

Základem je abstraktní třída BaseValidatorRule, která slouží jako základ pro všechny třídy pravidel validace. Hlavní metodou této třídy je metoda validate, která přijímá konkrétní diagram v objektové reprezentaci jako parametr. V rámci této metody probíhá logika kontroly, jejímž výsledkem může být buď validní nebo nevalidní stav, doplněný o důvod tohoto stavu.

V ukázkovém případě metoda nejprve prochází všechny aktivity z diagramu a ověřuje, zda při existenci odchozí hrany existuje také hrana příchozí. Pokud tomu tak není, metoda vrátí nevalidní stav spolu s chybovou hláškou. Tento postup se aplikuje i na události diagramu.

Pokud žádný z výše uvedených stavů nenastane, lze konstatovat, že konkrétní pravidlo validace nebylo porušeno.

Zbylé metody z obrázku slouží slouží k identifikaci konkrétního pravidla validace. Abstraktní třída BaseValidatorRule rovněž zajišťuje, že každé pravidlo je vždy uloženo i v databázi. Důvodem ukládání je nutnost uchování výsledků validačních pravidel pro konkrétní odevzdání určité zkoušky v databázi, což logicky vyžaduje existenci relace na dané pravidlo.

Metody valid a invalid, které jsou součástí abstraktní třídy BaseValidatorRule, umožňují snadné vrácení ValidatorRuleResult ve správném formátu.

Důležitá je také anotace třídy @Component. Tímto vytvořím ze třídy bean, která je držena v závislostním kontejneru (DI container). Po automatickém propojení (autowiring) základní třídy BaseValidatorRule ve validační službě získám instance všech validačních pravidel v kolekci pod jedním rozhraním bez nutnosti manuální tvorby instancí.

```java
@Component
public class HangingRule extends BaseValidatorRule {
    //Rozman pattern 8
    public HangingRule(ValidatorRuleDao validatorRuleDao) {
        super(validatorRuleDao);
    }

    @Override
    public ValidatorRuleResult validate(BpmnModelInstance submissionBpmn) {

        for (Activity activity : submissionBpmn.getModelElementsByType(Activity.class)) {
            if (activity.getIncoming().isEmpty() && !activity.getOutgoing().isEmpty())
                return invalid("Aktivita '%s' je zavěšená".formatted(activity.getName()));

        }

        for (IntermediateThrowEvent intermediateThrowEvent : submissionBpmn.getModelElementsByType(IntermediateThrowEvent.class)) {
            if (intermediateThrowEvent.getIncoming().isEmpty() && !intermediateThrowEvent.getOutgoing().isEmpty())
                return invalid("Událost '%s' je zavěšená".formatted(intermediateThrowEvent.getName()));

        }

        return valid();
    }

    @Override
    protected String getId() {
        return "Hanging";
    }

    @Override
    protected String getName() {
        return "Kontrola zavěšení";
    }

    @Override
    protected String getDescription() {
        return "Aktivity/eventy musí obsahovat vstupní hranu";
    }
}
```

Obrázek 8.2. Kontrola zavěšení - kód

8.3 Zabezpečení

Zabezpečení je klíčový aspekt každého informačního systému, zejména v případě, kdy jde o citlivá data a lze předpokládat pokusy o útok. V rámci vývoje diplomové práce bylo nutné klást důraz na zabezpečení, aby studenti nemohli například ovlivňovat data jiných studentů nebo získat přístup k referenčním zadáním.

Jedním z klíčových prvků zabezpečení systému je autentizace a autorizace uživatelů. Té se blíže věnuje podkapitola 8.4. Pro zajištění zabezpečení na serverové části je použita technologie Spring Security.

Uživatelé v systému mají přiřazené role, konkrétně „GAMAJUN_STUDENT“ a „GAMAJUN_TEACHER“. Tyto role určují úroveň přístupu k různým částem systému a funkcím. Na základě těchto rolí jsou povoleny různé metody controlleru, které jsou následně provolávány prostřednictvím GraphQL API.

Roli „GAMAJUN_TEACHER“ je možné získat pouze registrací pomocí tajného kódu, který je nastaven při nasazení aplikace. Tímto způsobem je zabezpečeno, že pouze

oprávněné osoby mohou získat přístup k funkcím a datům, které jsou vyhrazeny pro učitele. Anotace v controllerech pak umožňují snadno definovat, které metody mohou být přístupné pro uživatele s danou rolí.

Díky těmto opatřením je systém schopen efektivně chránit citlivá data a zajišťovat, že studenti nemohou získat neoprávněný přístup k informacím, které by mohly narušit integritu výukového procesu. Zabezpečení je tak zásadní součástí systému vyvinutého v rámci diplomové práce, která napomáhá jeho spolehlivému a bezpečnému provozu.

8.4 Přihlášeni a registrace

Autorizaci mezi klientem a serverem používá protokol OAuth 2.0, což je autorizační protokol, který umožňuje aplikacím získávat omezený přístup k uživatelským účtům na jiných službách, aniž by získávaly heslo uživatele. Je to standardizovaný a bezpečný způsob, jakým mohou třetí strany autentizovat a autorizovat uživatele a provádět operace jejich jménem, například číst nebo zapisovat data. [47]

OAuth 2.0 se primárně zaměřuje na autorizaci, zatímco pro autentizaci je využito OpenID Connect (OIDC). OIDC je tenká vrstva nad OAuth 2.0, která poskytuje rozšíření pro ověřování identity uživatele. OIDC využívá konceptu tzv. „ID tokenů“, které jsou založeny na standardu JSON Web Token (JWT). Tyto tokeny obsahují informace o uživateli a jeho autentizaci u autorizačního serveru. [48]

Jak již bylo zmíněno v kapitole 7.2.1, ve vyvíjené aplikaci je použit Spring Authorization Server na straně serverové aplikace spolu s Spring Resource Serverem. Spring Authorization Server pak zajišťuje správu a vydávání autorizačních tokenů podle OAuth 2.0 a OpenID Connect protokolů. To vše na základě konfigurace autentizačních a autorizačních pravidel a dalších aspektů souvisejících s autorizačním procesem.

Spring Resource Server slouží jako zabezpečený prostředek pro přístup k chráněným zdrojům. Tento modul ověřuje příchozí požadavky na základě platných tokenů a poskytuje přístup ke GraphQL aplikačnímu rozhraní dle uživatelských oprávnění.

Na straně klientské aplikace je tento proces řešen pomocí NextAuth.js, který rovněž zařizuje výměnu tokenů dle standardů OAuth 2.0 a OIDC a vystavuje programátorovi přívětivé rozhraní.

Na serveru jsou uživatelé a oprávnění spravováni v databázi pomocí Java Persistence API a zpřístupněni pomocí vlastní implementace UserDetailService. Tato služba zajišťuje načítání uživatelských údajů a jejich oprávnění z databáze a jejich konverzi do formátu, který je kompatibilní se Spring Security, což umožňuje další zpracování v rámci zabezpečení aplikace.

8.5 End-to-end typesafety

End-to-end typová bezpečnost je koncept, který zajišťuje konzistenci a typovou bezpečnost dat, operací a modelů napříč různými částmi systému, včetně komunikace mezi

serverem a klientem. Díky tomuto přístupu jsou datové modely ze serveru automaticky udržovány i na klientovi, což minimalizuje možnost chyb způsobených nesprávným použitím nebo nedorozuměním ohledně očekávaných datových struktur.

Serverová aplikace nabízí GraphQL API, které poskytuje přesné schéma všech dostupných dotazů, mutací a jejich vstupních a návratových typů. Klientská strana využívá knihovnu **GraphQL Code Generator** pro generování kódu na základě těchto schémat. V aplikaci definuji schéma dotazů a mutací, které plánuji použít a určuji parametry a návratové hodnoty, které mě zajímají.

Na základě těchto schémat Code Generator vytvoří TypeScript kód, který obsahuje React hooky a typované rozhraní. Použitím těchto speciálních metod dochází k volání GraphQL API a získání dat prostřednictvím knihovny Apollo. Díky tomu můžu komponenty úplně izolovat od logiky získávání dat, což zajišťuje lepší čitelnost a udržitelnost kódu.

Kapitola 9
Testování

Abych zajistil kvalitu a spolehlivost vyvíjené aplikace, podrobuji ji různým typům testů. Testování je nezbytné pro odhalení chyb, zlepšení uživatelského rozhraní a ověření funkčnosti aplikace v reálných podmínkách.

Jak již bylo řečeno, aplikace se skládá ze dvou částí - klienta a serveru. Je nutné pečlivě otestovat obě tyto komponenty. Serverová část je proto testována nezávisle pomocí jednotkových testů a spolu s klientskou částí pomocí uživatelských testů.

9.1 Jednotkové testy

Jednotkové testy jsou automatizované testy, které izolovaně ověřují funkčnost jednotlivých komponent programu. V praxi se obvykle testují třídy a jejich metody - předávají jim různé vstupy a zkouší, zda jsou jejich výstupy korektní. Jednotkové testy pomáhají odhalit chyby v kódu, zvýšit jeho kvalitu a zabránit regresivnímu zanášení nových chyb. Jednou z nevýhod jednotkových testů je možné potenciální nesoučinnost komponent. To znamená, že i když jednotlivé komponenty fungují správně podle jejich jednotkových testů, mohou nastat problémy při jejich integraci do celého systému. Tento problém adresují ostatní prováděné testy. [49]

V rámci vyvíjené aplikace pak podléhá jednotkovému testování celá aplikační vrstva, která obsahuje veškerou klíčovou logiku. Testy jsou prováděny nezávisle na databázi s využitím mockování.

Mockování je technika, která se používá v softwarovém testování a umožňuje simulovat chování komponent. Proces spočívá vytvoření umělých (mock) objektů, které nahrazují skutečné objekty a chovají se podobně jako ony. Mock objektům může být přiřazeno konkrétní očekávané chování. [50]

Typicky jsou mnou testované objekty závislé na nějakých jiných. Tuto závislost jsem odstranil tím, že tyto objekty byly nahrazeny mocky s očekovánými výstupy a testuji tak pouze logiku dané komponenty.

Technologicky je v diplomové práci použita kombinace JUnit 5 a Mockito.

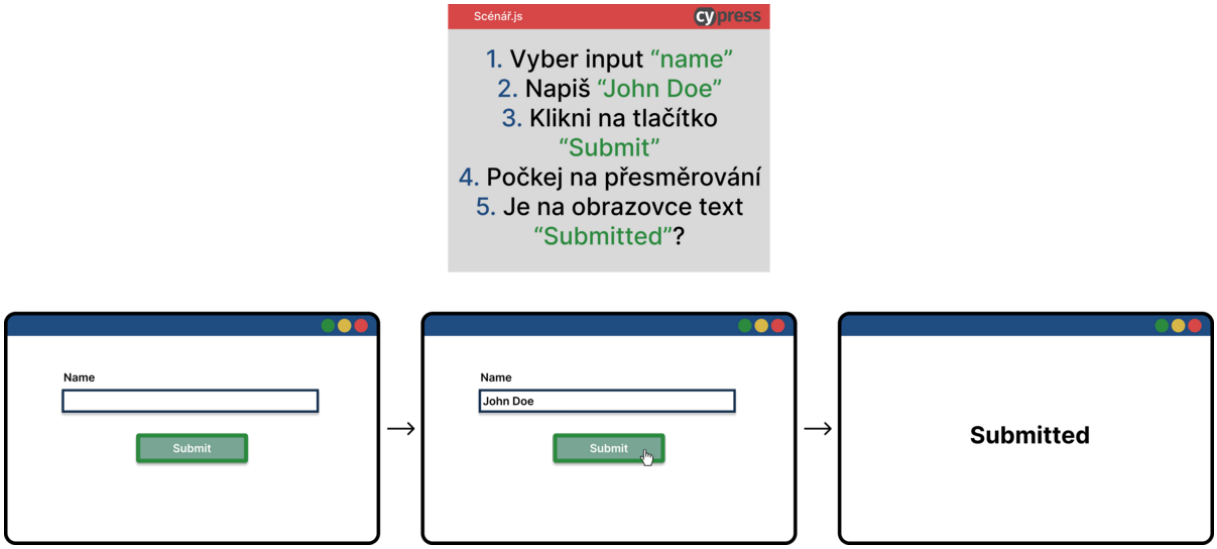
9.2 Automatické end-to-end testování

V diplomové práci jsem vážně zvažoval využít automatické end-to-end testování, které je považováno za efektivní a spolehlivé řešení pro testování komplexních softwarových

systémů. Tyto testy automaticky simuluji celý proces uživatelských interakcí od začátku do konce, což by mohlo být užitečné pro odhalení potenciálních problémů a chyb. V plánu bylo využití knihovny Cypress.

Po několika experimentech a poradě s vedoucí jsem však z několika důvodů tento přístup zavrhl. Hlavním důvodem bylo, že tyto testy neumožňují v rámci aplikace snadno kreslit diagramy, což by mohlo nabourávat účel těchto testů. Diagramy jsou hlavní součástí této diplomové práce a bez schopnosti snadno je kreslit, bych nedokázal plně ověřit integritu a správnou funkčnost celého systému. Zároveň většinu testovacích scénářů lze bez problémů pokrýt pomocí manuálních testů.

I přes některé výhody automatického testování, jako je nižší náročnost na čas a lidské zdroje, jsme dospěli k závěru, že v mém případě představuje lepší volbu pro zajištění kvality a spolehlivosti využití právě uživatelského testování.



Obrázek 9.1. Automatické end-to-end testování

9.3 Uživatelské testování

Uživatelské testování spočívá v tom, že se vybere skupina reprezentativních uživatelů a dá se jim za úkol splnit nějaký cíl pomocí testovaného softwaru. Obvykle je jim poskytnut scénář, který popisuje situace a úkoly, které mají vyřešit. Cílem uživatelského testování se scénáři je zjistit, jak dobře software splňuje potřeby a očekávání uživatelů, jak snadno a rychle se s ním pracuje a jaké jsou jeho silné a slabé stránky. [51]

Scénáře pro uživatelské testování vyvíjené aplikace jsou dostupné v příloze A.1. Při průchodu scénáři testeři zároveň vyplňovali dotazník, který naleznete v příloze C.1.

9.3.1 Vyhodnocení uživatelského testování

V této podkapitole se zaměřím na vyhodnocení uživatelského testování, které bylo provedeno za účelem získání zpětné vazby a identifikace možných nedostatků či oblastí ke zlepšení navrženého systému.

Celkem se testování zúčastnilo pět testerů, kteří reprezentovali různé skupiny potenciálních uživatelů systému od méně technicky zdatných uživatelů až po pokročilé uživatele. Každý z testerů byl vyzván k vyplnění dotazníku po dokončení každého testovacího scénáře.

V následujících částech této podkapitoly podrobně analyzuji a vyhodnocuji zpětnou vazbu získanou od testerů prostřednictvím dotazníků. Tato analýza poskytne cenné informace pro další vývoj a optimalizaci systému.

Scénář 1

V rámci uživatelského testování Scénáře 1 jsem zaznamenal úspěšnost splnění 100 %, což svědčí o vysoké míře funkčnosti a použitelnosti registrace a přihlášení. Uživatelé byli celkově spokojeni s uživatelským prostředím, které si zasloužilo průměrné hodnocení 4/5.

Jeden z testerů upozornil na neobvyklé oddělení registrace a přihlášení, což je v současnosti často centralizováno. Toto oddělení je z technických důvodů nutné vzhledem k použití Spring Authorization Serveru a bylo pohodlnější implementovat registraci na klientovi než na serveru. Uvědomuji si, že tento postup může být pro některé uživatele nezvyklý, ale zároveň věřím, že důvody, proč jsem se rozhodl pro takovou implementaci, jsou opodstatněné.

Na druhou stranu, jeden z testerů ocenil funkci validace kódu, což mi dokázalo, že dvoukroková registrace byla správným implementačním rozhodnutím.

Z uživatelského testování Scénáře 1 vyplývá, že registrace a přihlášení je dobře navrženo a plně funkční. Přesto beru na vědomí zpětnou vazbu týkající se oddělení registrace a přihlášení, jako potenciální prostor pro zlepšení.

Scénář 2

Uživatelské testování Scénáře 2 opět prokázalo vysokou úspěšnost 100 %. Testeři byli celkově spokojeni s uživatelským prostředím a udělili mu průměrné hodnocení 4/5.

Méně technicky zdatný tester měl obtíže s nalezením tlačítka pro ohlášení, což považuji za individuální problém, jelikož ostatní uživatelé s tímto neměli potíže a tlačítko se nachází na místě typickém pro ostatní webové aplikaci. Přesto jsem se rozhodl reagovat na tuto zpětnou vazbu a přidal jsem tooltip pro tlačítko, aby byla jeho funkce jasnější pro všechny uživatele.

Testeři ocenili několik aspektů mé aplikace. Jeden z nich chválil jednoduchý způsob registrace, kdy je student okamžitě zasazen do třídy na základě kódu. Tento způsob zjednodušuje proces registrace a urychluje začlenění studenta do třídy. Další tester oceňoval rychlost, s jakou se mohl zaregistrovat do aplikace.

Z uživatelského testování Scénáře 2 vyplývá, že funkcionalita tvorby tříd a registrace studentů je funkční a uživatelský přívětivá.

Scénář 3

Jsem potěšen, že úspěšnost testování Scénáře 3 byla 100 % a celková spokojenost s uživatelským prostředím dosáhla 3.8/5. Během testování jsem zaznamenal několik problémů, na které mě uživatelé upozornili.

Dva testeři si stěžovali na nedostatečně specifický scénář, kvůli kterému mírně tápali, než ho úspěšně dokončili. Přisuzuji to mému nedostatečnému vysvětlení návrhu a účelu systému, což způsobilo, že testeři nebyli jisti, co mají dělat. Uvědomuji si, že toto je spíše moje chyba než chyba systému, a budu se snažit v budoucích testech poskytnout jasnější pokyny a kontext.

Na druhou stranu, dva testeři ocenili koherentní uživatelské rozhraní napříč celou aplikací. To je pozitivní zjištění, které ukazuje, že aplikace má konzistentní a srozumitelný design.

Méně zkušený tester byl zmatený překlikáváním karet v editoru Zadání. Tuto zpětnou vazbu vezmu v potaz a zvážím možnosti, jak zjednodušit a vylepšit tento aspekt aplikace, aby byl přístupnější pro uživatele s různými úrovněmi zkušeností.

Uživatelské testování Scénáře 3 bylo celkově úspěšné a získalo kladná hodnocení. V budoucích testech se zaměřím na poskytnutí jasnějších pokynů a kontextu, abych minimalizoval zmatení uživatelů. Zároveň budu pracovat na zlepšení uživatelského prostředí, aby bylo co nejpřívětivější pro všechny uživatele.

Scénář 4

Při uživatelském testování Scénáře 4 jsem zaznamenal úspěšnost 80 % a celkovou spokojenost s uživatelským prostředím 4.25/5. Testeři si stěžovali na několik problémů.

První tester nebyl schopen dokončit scénář, protože v systému se objevila chyba zanesená při refactoringu. Chyba znemožňovala spuštění editoru zkoušek. Po nahlášení došlo k okamžité opravě, aby byla aplikace opět plně funkční.

Jeden tester si stěžoval, že scénář obsahuje příliš mnoho odhlašování a přihlašování, což je pravda, ale vzhledem k povaze aplikace s tím nelze nic dělat. Přesto si uvědomuji, že tento aspekt může být pro uživatele nepříjemný.

Dva testeři ocenili obrazovku výsledků odevzdání zkoušky, což je pozitivní zpětná vazba, která ukazuje, že tato část aplikace je pro uživatele užitečná a přehledná.

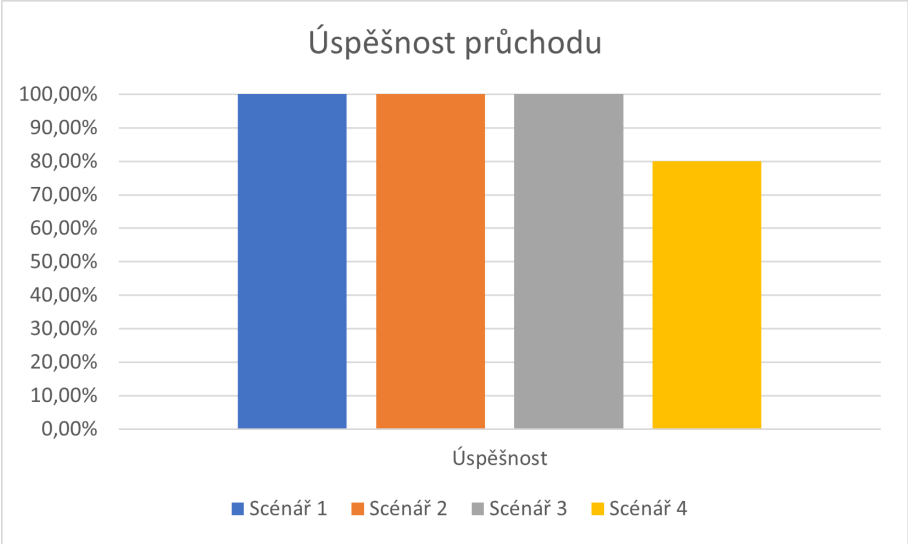
Uživatelské testování Scénáře 4 mělo celkově dobré výsledky, i když nebylo bez problémů. Opravil jsem chybu zanesenou při refactoringu. Kladná zpětná vazba ohledně obrazovky výsledků odevzdání zkoušky je povzbudivá a ukazuje, že aplikace je pro uživatele užitečná.

Shrnutí testování

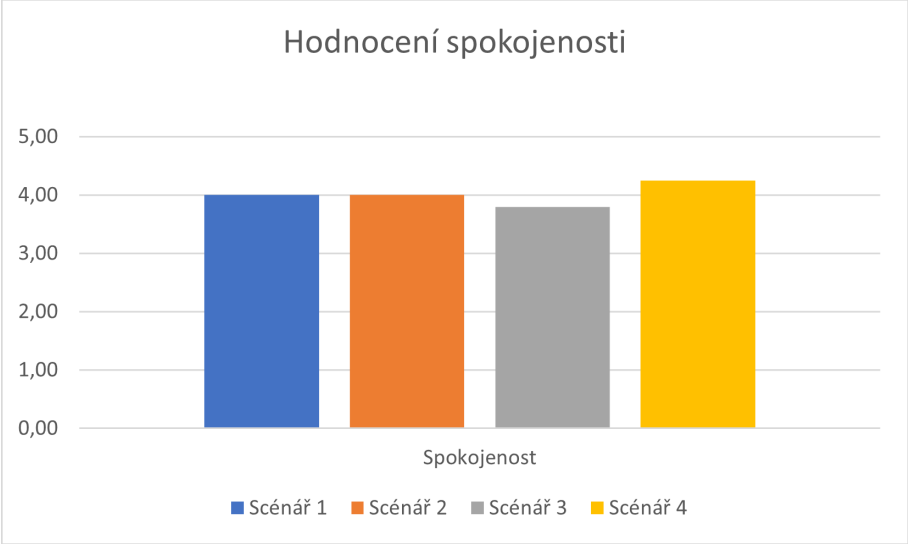
Uživatelské testování považuji za úspěšné, protože ačkoliv se objevilo několik výtek, neobjevil jsem žádný zásadní problém kromě jedné kritické chyby, kterou jsem opravil. Testeři chválili koherentní uživatelské rozhraní a celkový design aplikace, což je pozitivní zpětná vazba, která ukazuje, že moje aplikace je uživatelsky přívětivá a vizuálně přitažlivá.

Testeři nebyli schopni posoudit přínos aplikace na předmět, vzhledem k tomu, že se nejednalo o cílovou skupinu aplikace. To by na jednu stranu mohlo být vnímáno jako špatná metodika, na druhou stranu vývoj a požadavky byly průběžně konzultovány s garantem předmětu, tudíž by to v konečném důsledku neměl být problém.

Shrnutí výsledků testování naleznete na grafech 9.2 a 9.3.



Obrázek 9.2. Testování - Graf úspěšnosti



Obrázek 9.3. Testování - Graf spokojenosti

Kapitola 10
Projektové vyhodnocení

V této kapitole se zaměřím na vyhodnocení diplomové práce z pohledu projektového řízení. Je důležité si uvědomit, že realizace diplomové práce je v podstatě projekt, který má stanovené cíle, časový rámec a omezené zdroje.

Cílem tohoto vyhodnocení je analyzovat, jak byl tento konkrétní projekt realizován, a identifikovat silné stránky, případně oblasti, kde by bylo možné provést zlepšení pro budoucí projekty. Zařazením této kapitoly do diplomové práce chci zdůraznit důležitost správného projektového řízení, které je klíčovým faktorem pro úspěch jakéhokoli projektu.

10.1 Popis projektu

Primárním cílem projektu bylo vyvinout jednoduší způsob testování znalostí notace BPMN ve výuce předmětu Procesní řízení. Ačkoli existuje mnoho nástrojů, které využívají notaci BPMN, jak jsem popsal v kapitole 4, doposud žádný není svou povahou určen pro výukové účely.

Na celém projektu jsem úzce spolupracoval s vedoucím práce p. Náplavou, který je zároveň garantem předmětu Procesní řízení. Spolupráce byla velmi prospěšná, protože mi umožnila navrhnout systém přesně na míru potřebám předmětu. Výsledný systém tak nalezne praktické uplatnění a přinese řadu benefitů. Vyučujícím šetří čas díky funkcím automatického vyhodnocení a celkové centralizaci, studentům pak poskytne velmi dobrou a rychlou zpětnou vazbu a prostor pro trénování.

Automatické vyhodnocení bylo jedním z nejdůležitějších požadavků a zároveň se jedná o nejkomplexnější součást systému. Zpočátku nebylo jasné, zda-li je něco takového vůbec proveditelné a jednalo se tak o hlavní oblast výzkumné části.

10.2 Průběh projektu

Projekt jsem zahájil diskuzí o předmětu Projektové řízení, který jsem měl i sám možnost dříve absolvovat. Snažil jsem se pochopit, jakým způsobem je testování prováděno nyní a postupně si začal vytvářet představu a potenciálních případech užití.

Po počáteční analýze a výběru technologií jsem se vzhledem k omezeným časovým kapacitám z důvodu další výuky v 5. semestru rozhodl přistupovat k vývoji způsobem rapid prototypingu. Ten spočívá v rychlém a iterativním vytváření a ověřování konceptů přímo na vyvíjené aplikaci a jejich konzultaci se stakeholderem, v mém případě

s vedoucím práce. Tak jsem byl každou konzultaci schopen předvést další novou část funkcionalit, zkonzultovat ji a případně upravit. Agilní metodiky jsou dnes trendem a retrospektivně vzato považuji svůj přístup za správný a přínosný.

Důležitým milníkem bylo odevzdání výzkumného projektu na konci 5. semestru, ve kterém jsem se rozhodl doručit „minimum viable product“, tedy produkt s nejmenší možnou funkcionalitou, který je však plně použitelný. Konkrétně se jednalo o aplikaci s funkční autentizací, tvorbou zadání, testů a jejich vyplňování a odevzdání. Klíčovou chybějící funkcionalitou bylo tolik skloňované automatické vyhodnocení.

Ta byla vyvíjena až v předposlední fázi stejnou metodikou. Ze zadání vyplynula povinnost vytvořit detekci nejčastějších modelovacích anti-patternů, kterou jsem rozšířil o algoritmus srovnání s referenčním řešení.

Během projektu se objevil požadavek na maximálně jednoduché zapnutí a opětovné vypnutí systému. To vedlo k potřebě minimalizovat závislosti na externích systémech, což výrazně omezilo možnost využití služeb třetích stran, jako je například autorizační server a vše muselo být integrováno v rámci aplikace.

10.3 Zhodnocení

Celkově musím projekt zhodnotit jako úspěšný. V relativně rozumném časovém úseku se podařilo doručit produkt za využití moderních technologií a postupů. Výsledný produkt je plně funkční s moderním a elegantním uživatelským rozhraní. Plánované ostré nasazení bude probíhat další semestr výuky Projektového řízení.

10.3.1 Časový rámec

V rámci časového rámce projektu bylo původně plánováno dokončení softwarové části a textu diplomové práce k 1. květnu 2023. Během realizace projektu jsem čelil několika výzvám a nakonec jsem nabral zhruba týdenní zpoždění oproti plánovanému termínu. Přestože jsem nesplnil původní časový rámec, považujeme tento výsledek za úspěch, protože zpoždění bylo minimální a neovlivnilo kvalitu výstupů projektu.

V průběhu projektu jsem definoval následující klíčové milníky:

Sběr požadavků a návrh aplikace: Prvním milníkem bylo dohodnout se s vedoucím práce na popisu problému a jeho možném řešení. Tento úkol zahrnoval navržení podoby systému i technického řešení. Na tento krok byl vyčleněn jeden měsíc v rámci výzkumného projektu. Tato fáze byla úspěšně splněna včas.

Nejživotaschopnější produkt (MVP): Druhým milníkem bylo vytvoření nejživotaschopnějšího produktu, který byl odevzdán jako součást výzkumného projektu na konci pátého semestru. MVP zahrnovalo text v rozsahu 20 stran a systém ve stavu, kdy splňoval všechny základní požadavky s výjimkou automatického vyhodnocení.

Systém ve finální podobě: Třetím milníkem bylo mít systém téměř kompletně hotový do 1. dubna s cílem již jen ladit detaily a psát finální verzi textu diplomové práce. I tento úkol se podařilo úspěšně naplnit včas.

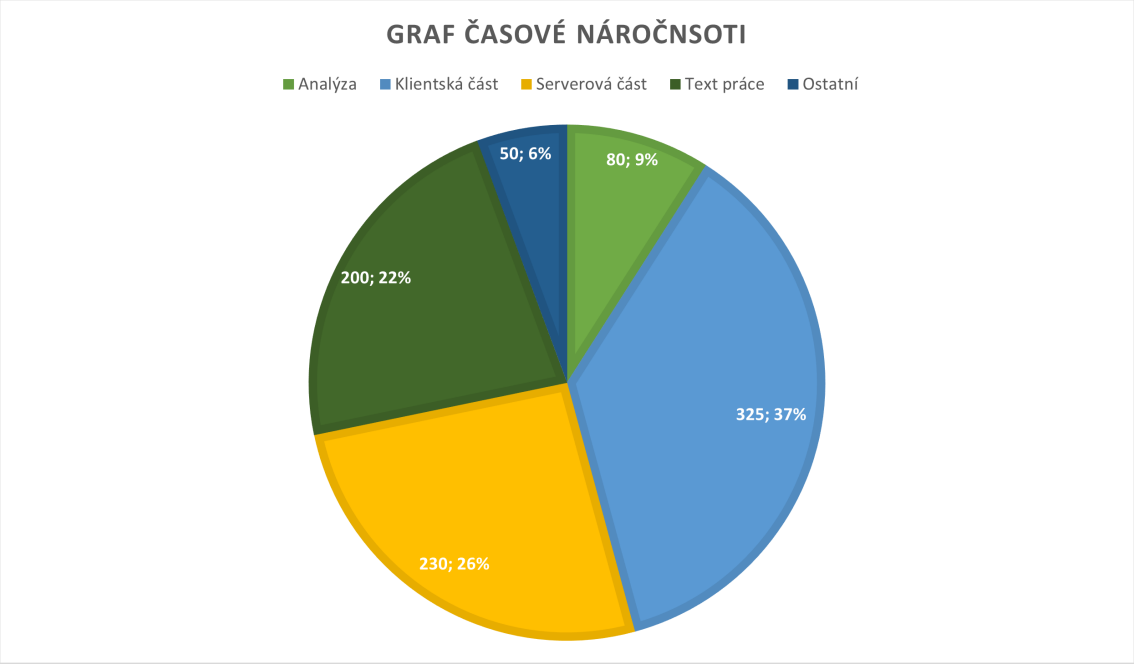
Tyto milníky ukazují, že projekt byl dobře naplánován a řízen, což umožnilo úspěšně dosáhnout klíčových cílů v daných termínech.

10.3.2 Časová náročnost

V tabulce 10.1 a na obrázku 10.1 naleznete odhadovaný čas strávený na jednotlivých částech diplomové práce. Je důležité zdůraznit, že tento čas nebyl měřen přesně, ale jde o aproximaci. Tabulka nám poskytuje přehled o tom, kolik času jsem věnoval jednotlivým segmentům práce, jako jsou rešerše, psaní, vývoj a další. Tento odhad může být užitečný pro lepší orientaci v procesu tvorby diplomové práce. Na práci jsem pracoval průběžně zhruba 34 týdnů v průměru 25 hodin týdně.

Část	Aktivita	Počet hodin
Sběr a modelování požadavků	Analýza	80
Serverová část	Rešerše a testování použitých technologií	30
Serverová část	Vývoj	200
Klientská část	Rešerše a testování použitých technologií	50
Klientská část	Vývoj	260
bpmn-js	Vývoj	15
Testování	Integrační testování	10
Testování	Uživatelské testování	20
Text práce	Psaní	200
Instalační příručka	Příprava	20
Celkem		**885**

Tabulka 10.1. Přehled časové náročnosti.



Obrázek 10.1. Graf časové náročnosti

Kapitola 11
Závěr

V rámci této diplomové práce jsem se zaměřil na vývoj testovacího systému pro notaci BPMN (Business Process Model and Notation). Po důkladném zkoumání a analýze výsledků mohu konstatovat, že práci považuji za úspěšnou.

V této závěrečné kapitole se zaměřím na zhodnocení dosažených cílů diplomové práce, přínosy, které práce přináší, možnosti zlepšení a nakonec na nabyté znalosti během jejího zpracování.

11.1 Vyhodnocení cílů

Diplomová práce si kladla tři hlavní cíle:

1. Analýza současného postupu výuky notace BPMN v předmětu Procesní řízení

2. Návrh a vývoj systému pro snadné testování znalostí modelování v notaci BPMN

3. Funkce automatického vyhodnocení odevzdaných diagramů

Prvním cílem byla analýza současného postupu výuky notace BPMN v předmětu Procesní řízení. Během analýzy současného přístupu jsem zjistil, že použitá metoda testování je neintuitivní a časově velmi náročná pro lektora. Tento zjištěný nedostatek poukazuje na potřebu zefektivnění procesu testování a vyhodnocování.

Druhým cílem bylo navržení a vývoj systému pro snadné testování znalostí modelování v notaci BPMN. Na základě zjištěných poznatků z analýzy současného stavu a diskuzí s vedoucím práce byl navržen a vyvinut nový systém, který je přizpůsoben potřebám předmětu. Tento systém výrazně zjednodušuje proces testování a hodnocení znalostí studentů v oblasti modelování BPMN.

Třetím cílem byla implementace funkce automatického vyhodnocení odevzdaných diagramů. V rámci vývoje nového testovacího systému byla úspěšně implementována tato funkce, která umožňuje s relativně velkou jistotou určit shodu s referenčním řešením. V případě, že automatické vyhodnocení selže, obsahuje systém nástroje, které lektorovi značně usnadní vyhodnocení.

Na základě úspěšného splnění všech tří cílů lze konstatovat, že diplomová práce přinesla významné vylepšení v oblasti testování a hodnocení znalostí modelování v notaci BPMN. Blíže se přínosům věnuje následující kapitola 11.2.

11.2 Přínosy diplomové práce

Diplomová práce přináší řadu výhod pro výuku předmětu Procesní řízení a zároveň představuje pilotní verzi systému pro výuku BPMN notace. Tyto přínosy lze shrnout do následujících klíčových bodů:

Zlepšení výuky předmětu Procesní řízení: Výsledný systém zjednodušuje a zlepšuje výuku tohoto předmětu tím, že poskytuje studentům tréninkové prostředí, ve kterém se mohou lépe seznámit s praktickými aspekty procesního řízení, což jim pomáhá lépe pochopit a zvládnout látku. Toto prostředí rovněž umožňuje studentům intuitivnější způsob skládání zkoušek.

Úspora času a snížení zátěže pro učitele: Systém učitelům usnadňuje a urychluje vyhodnocení výsledků zkoušek. Díky tomuto systému mohou učitelé snadno sledovat pokrok svých studentů a identifikovat oblasti, kde je třeba zaměřit pozornost a zlepšit výuku. Tímto způsobem je snížena administrativní zátěž a učitelé mohou věnovat více času studentům a jejich individuálním potřebám.

Pilotní verze systému pro výuku BPMN notace: Diplomová práce představuje první systém, který je speciálně navržen pro výuku BPMN notace.

Podpora rozvoje dovedností studentů: Díky tréninkovému prostředí se studenti mohou efektivněji učit BPMN notaci a rozvíjet své dovednosti v oblasti procesního modelování.

11.3 Prostor pro zlepšení

Ačkoli projekt splňuje všechny případy užití prostor pro zlepšení vidím obligatorně v systému pro automatické vyhodnocení. V rámci vývoje jsem implementoval pouze anti-patterny, kterých se studenti na základě výzkumu dopouští nejčastěji. Z logiky věci však takových anti-patternů existuje podstatně více a jsou mnohem obtížněji detekovatelné.

Jedním z takových je například problém uváznutí (deadlock), který může nastat při špatném použití bran. Příkladem můžem být exkluzivní brána, z níž obě větve vedou logicky do brány paralelní. K detekci tohoto chování by bylo nutné simulovat toky v diagramu.

Rovněž jsem přesvědčen, že systém pro referenční srovnání má potenciál být více spolehlivým vylepšením současného srovnání izomorfismů. Současný stav je však to nejlepší, co jsem byl schopný vytvořit. Nepochybuji však, že s využitím pokročilých znalostí teorie grafů, by mohl být ještě lepší.

11.4 Získané znalosti

Za nejcennější osobní přínos považuji praktickou implementaci znalostí teorie grafů, kterou jsem absolvoval v několika předmětech na fakultě. Díky znalostem z předmětu

Pokročilá algoritmizace jsem si uvědomil, že námi řešený problém v podobě automatického vyhodnocení je jen pokročilejší implementací problému hledání izomorfismů. Díky svým zkušenostem jsem byl také schopen dohledat knihovnu pro práci s grafy a vyhnul jsem se tak potřebě tyto velmi komplexní grafové algoritmy implementovat sám.

Cennou zkušeností taky byla možnost vyzkoušet si vývoj systému přes celý jeho vývojový cyklus, od sběru požadavků a návrhu přes implementaci až po psaní instalační příručky. Byl jsem schopen použít všechny své dovednosti jakožto full-stack vývojář a dosáhl kýžené dodávky.

Příloha A
Testovací scénáře

Tato příloha obsahuje scénáře pro uživatelské testování.

Scénáře jsou záměrně psané vágně, jelikož tato technika zároveň ověřuje, zda je uživatelské rozhraní snadno ovladatelné a intuitivní. Tím, že se scénáře nezabývají přílišnými detaily, poskytují uživatelům prostor pro prozkoumání a objevování funkcí systému. To umožňuje designérům a vývojářům pozorovat, jak si uživatelé s aplikací nebo webovou stránkou poradí bez předchozího podrobného návodu. Opačným přístupem jsou detailní scénáře, kdy jsou kroky a instrukce pečlivě popsány, aby bylo jasné, jak má uživatel postupovat. Tyto scénáře mohou být užitečné při testování konkrétních funkcí a jejich splnění konkrétních úkolů.

Vzhledem k provázanosti datového modelu jsou tyto scénáře na sobě závislé a je nutné je vykonávat sekvenčně. Pro průchod testovacími scénáři je zapotřebí disponovat tajným kódem pro registraci učitelského účtu.

A.1 Scénář 1 - Registrace učitele

Prvním krokem je zřízení účtu s oprávněním učitele. Ten může vykonávat veškerou správcovskou činnost v rámci aplikace. Registrace probíhá prostřednictvím tajného kódu, který se nastavuje během instalace.

Běžte na registrační stránku a zadejte tajný kód. Vyplňte uživatelská data z tabulky A.1 a proveďte registraci.

Atribut	Hodnota
Jméno	Igor
Příjmení	Hnízdo
Email	hnizdig@cvut.cz
Uživatelské jméno	hnizdig
Heslo	gamajun

Tabulka A.1. Scénář 1 - Testovací uživatel - učitel.

Ověřte úspěšnost registraci přihlášením do systému.

A.2 Scénář 2 - Vytvoření třídy a registrace studenta

Začíná nový semestr a je potřeba připravit se na výuku. Prvním krokem je založení třídy, která seskupuje uživatele. Po přihlášení do systému jako učitel ve správci tříd založte novou. Pojmenujte ji *Testovací třída* a jako kód pozvánky nastavte *test2023*.

Nyní je na čase ověřit, že se studenti budou schopni zaregistrovat. Odhlaste se ze systému a zaregistrujte testovacího uživatele. Jako kód pozvánky použijte *test2023*, údaje uživatele naleznete v tabulce A.2.

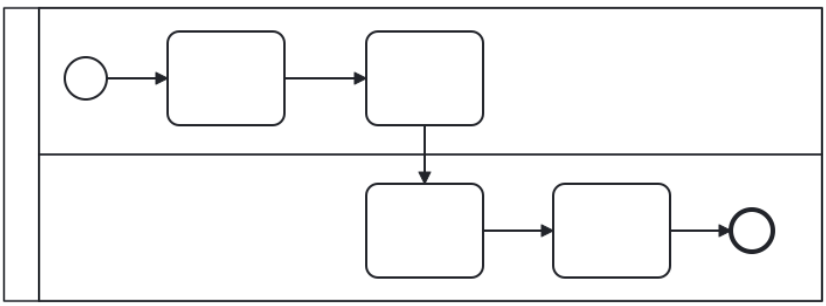
Atribut	Hodnota
Jméno	Eduard
Příjmení	Souček
Email	souceed@cvut.cz
Uživatelské jméno	souceed
Heslo	gamajun

Tabulka A.2. Scénář 2 - Testovací uživatel - student.

Úspěšnost registrace ověřte přihlášením se studentským účtem.

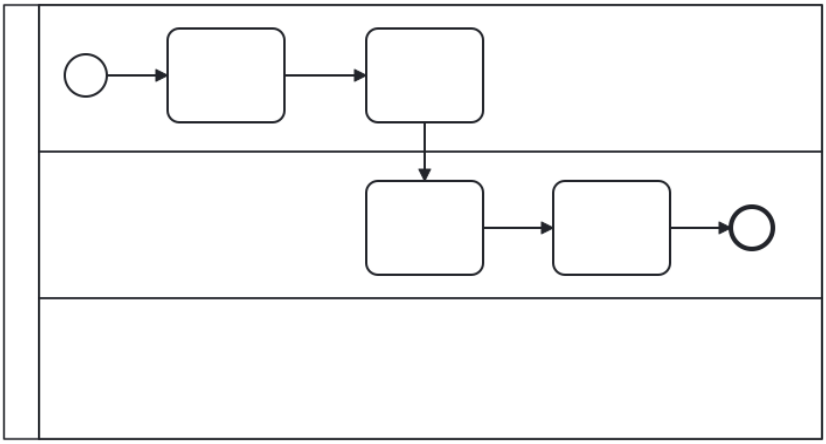
A.3 Scénář 3 - Vytvoření zadání a Sandbox

Zadání jsou základním stavebním prvkem zkoušek i sandboxu. Vstupte do sekce **Správce zadání** a vytvořte nové. Pojmenujte jej *Testovací zadání* a uveďte libovolný popis. V záložce referenční řešení překreslete diagram, který se nachází na obrázku A.1. Nakonec na kartě Nastavení přepněte zadání jako sandboxové a uložte.



Obrázek A.1. Testovací zadání

Nyní přejděte do sekce **Sandbox**. Mělo by zde na vás čekat nově vytvořené zadání, které máte za úkol vyzkoušet. Vypracujte diagram z obrázku A.2. Svůj pokus pak nechte vyhodnotit. Jestliže vše funguje jak má, celý výsledek validátoru by měl být zelený, vyjma počtu participantů u referenčního srovnání. V sekci **Sandbox** pak ještě zkontrolujte, že v historii přibyl absolvovaný pokus.



Obrázek A.2. Sandbox diagram

Jako poslední krok zadání v sekci **Správce zadání** aktualizujte tak, aby nebylo sandboxové.

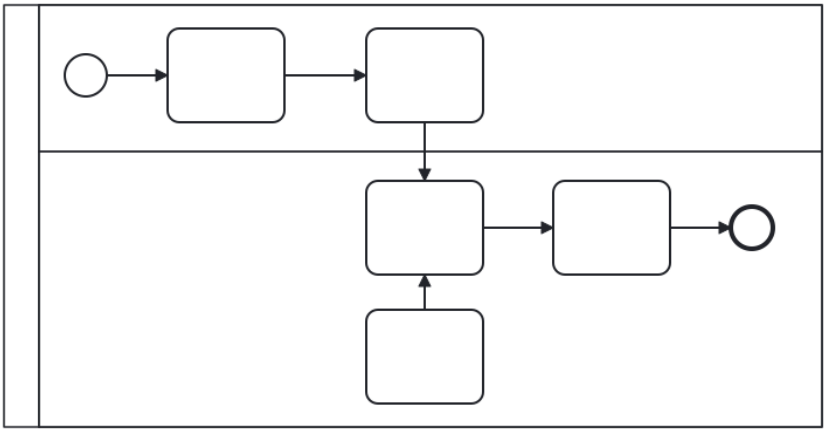
A.4 Scénář 4 - Zkoušky

Nyní je vhodná doba naplánovat zkoušku. Přejděte do **Správce zkoušek** a vytvořte novou. Nastavte parametry podle tabulky A.3 a uložte. Tímto je zkouška naplánovaná a měla by být dostupná.

Atribut	Hodnota
Název	Testovací zkouška
Platné od	(dnešní datum) mínus 1 den : 00:00
Platné do	(dnešní datum) plus 1 den : 00:00
Časový limit	30
Přidělená zadání	Testovací zadání
Přidělená třída	Testovací třída

Tabulka A.3. Scénář 4 - Testovací zkouška.

Vaším úkolem je pokusit se zkoušku vypracovat, přihlaste se proto testovacím uživatelem ze Scénáře 2. V sekci **Zkoušky** by se měla objevit námi vytvořená *Testovací zkouška*. Zkoušku tedy zahajte a v editoru diagramů překreslete diagram A.3 a odevzdejte. Automatické vyhodnocení by mělo hlásit neshodu v izomorfismu a porušení pravidla zavěšení. Zkouška by se měla dostat do stavu *odevzdáno*.



Obrázek A.3. Zkouškový diagram

Zkouškový pokus je potřeba ohodnotit. Znovu se tak přihlaste jako učitel a ve **Správci zkoušek** otevřete přehled odevzdání Testovací zkoušky a následně námi odevzdané řešení testovacího uživatele. To libovolně ohodnoťte a uložte. Stav zkoušky by se měl dostat do stavu *ohodnoceno*.

Naposledy se přihlaste jako testovací uživatel a v sekci odevzdaných zkoušek zkontrolujete, zda byla zkouška správně ohodnocena.

A.5 Scénář 5 - Objevování

V tomto testovacím scénáři vás vyzývám, abyste volně prozkoumávali aplikaci a objevovali různé funkce, na které narazíte. Zvláštní pozornost věnujte možnostem automatického vyhodnocení. Scénář neposkytuje konkrétní kroky, které musíte provést, ale spíše vám umožní vytvořit si vlastní zkušenost při průzkumu aplikace.

Příloha B
Nasazení a instalační příručka

Následující kapitola se zabývá nasazením aplikace v produkčním prostředí.

Produkční a vývojové prostředí jsou dva různé typy prostředí používané v průběhu vývoje a nasazení softwarových aplikací. Vývojové prostředí je určeno pro tvorbu, testování a ladění aplikace, zatímco produkční prostředí je zaměřeno na běh finální verze aplikace, která je přístupná uživatelům. V obou prostředích mohou být nastavení a konfigurace odlišné, aby vyhovovaly jejich specifickým potřebám a požadavkům.

B.1 Komponenty a komunikae

Tři komponenty jsou nutné pro nasazení systému: klient, server a Postgres databáze. Server a databáze spolu komunikují přímo, zatímco server a klient si navzájem vyměňují data jak přímo, tak prostřednictvím prohlížeče. Je důležité zohlednit tuto komunikační strukturu při nasazení, jelikož může způsobit řadu komplikací.

B.2 Prostředí

Systém je v produkčním prostředí doporučeno nasazovat pomocí Dockeru, což je platforma založená na virtualizační technologii kontejnerů. Docker umožňuje automatizaci nasazení, zjednodušení konfigurace a řízení aplikací v kontejnerech. Kontejnery izolují aplikace a jejich závislosti do samostatných prostředí, čímž zajišťují konzistenci mezi různými vývojovými a provozními prostředími. Na rozdíl od tradiční virtualizace, která emuluje celý operační systém, kontejnery sdílejí zdroje a jádro hostitelského systému, což vede k menšímu zatížení a vyššímu výkonu.

B.2.1 Proměnné prostředí

Systém je konfigurován prostřednictvím proměnných prostředí (environment variables), což je flexibilní a bezpečný způsob nastavení různých aspektů aplikace. Proměnné prostředí umožňují definovat klíčové hodnoty, jako jsou přístupové údaje, adresy serverů nebo jiné konfigurační informace, aniž by tyto údaje byly uloženy přímo v kódu aplikace. Díky tomu je možné snadno upravit nastavení systému bez nutnosti úprav kódu, což usnadňuje správu a zabezpečení aplikace.

Proměnné prostředí klientské aplikace:

- **OAUTH2_CLIENT_ID** - Client Id OAuth2 autorizačního serveru

- **OAUTH2_CLIENT_SECRET** - Secret OAuth2 authorizačního serveru
- **OAUTH2_PROVIDER_URL** - URL pro OAuth2 autorizační server
- **NEXTAUTH_URL** - URL na které poběží klientská aplikace
- **NEXTAUTH_SECRET** - Secret klientské aplikace
- **NEXT_PUBLIC_GAMAJUN_API_URL** - Veřejná URL Gamajun serveru

Hodnoty proměnných **OAUTH2_PROVIDER_URL** a **NEXT_PUBLIC_GAMAJUN_API_URL** budou obvykle shodné, což odpovídá URL adrese, na které Gamajun server běží. Nicméně protože OAuth autorizační server je volán nejen z prohlížeče, ale také na pozadí klientem, je vhodnější mít tyto proměnné rozdělené.

- **POSTGRES_HOST** - Adresa Postgres serveru (jdbc:postgresql://xxxxx)
- **POSTGRES_USER** - Uživatelské jméno pro přístup do databáze
- **POSTGRES_PASSWORD** - Heslo pro přístup do databáze
- **ADMIN_CODE** - Tajný kód pro registraci učitelských účtů
- **OAUTH2_CLIENT_ID** - Client Id OAuth2 autorizačního serveru
- **OAUTH2_CLIENT_SECRET** - Secret OAuth2 authorizačního serveru
- **CLIENT_URL** - Veřejná URL klientské aplikace

B.3 Instalace

Pro snadné nasazení jsem vytvořil instalační bash skript, který slouží k usnadnění procesu instalace a s tím spojených nastavení proměnných prostředí. Po úspěšném nastavení skript automaticky zkompiluje a spustí požadované Docker instance. Server poběží na portu 8080 a klient na portu 3000, databáze bude dostupná pouze mezi kontejnery.

Skript je navržen pro použití na linuxovém serveru a předpokládá, že uživatel má k dispozici veřejnou IP adresu nebo doménu a má nainstalovaný Docker. Instalace je interaktivní, což znamená, že během jeho provádění komunikuje s uživatelem a žádá o potřebné informace pro úspěšné nasazení aplikace.

B.3.1 Postup

1. Stáhněte repositář včetně submodulů příkazem *git clone –recurse-submodules https://github.com/pan-sveta/gamajun-platform*

2. Povolte spuštění scriptu příkazem *chmod +x install.sh*

3. Spusťte script příkazem *./install.sh*

4. Řiďte se pokyny ve scriptu

Příloha C
Dotazník k uživatelskému testování

C.1 Dotazník k uživatelskému testování

Gamajun - Dotazník k uživatelskému testování

https://gamajun.vercel.app/

Kód pro registraci administrátora: tajnykod

* Povinné

Scénář 1

V tomto oddíle odpovídejte na otázky týkající se scénáře 1. Vzhledem k návaznosti scénářů v případě neúspěchu testování ukončete a odešlete formulář. Děkuji!

1. Setkali jste se s nějakým problémem?
   Každý detail se počítá!

2. Podařilo se Vám scénář 1 úspěšně splnit? *

   ◯ Ano

   ◯ Ne

3. Jak hodnotíte jednoduchost a přehlednost navigace v aplikaci v tomto scénáři? *

Scénář 2

V tomto oddíle odpovídejte na otázky týkající se scénáře 2. Vzhledem k návaznosti scénářů v případě neúspěchu testování ukončete a odešlete formulář. Děkuji!

4. Setkali jste se s nějakým problémem?

Každý detail se počítá!

5. Podařilo se Vám scénář 2 úspěšně splnit? *

- ◯ Ano
- ◯ Ne

6. Jak hodnotíte jednoduchost a přehlednost navigace v aplikaci v tomto scénáři? *

😄 😄 😄 😄 😄

Scénář 3

V tomto oddíle odpovídejte na otázky týkající se scénáře 3. Vzhledem k návaznosti scénářů v případě neúspěchu testování ukončete a odešlete formulář. Děkuji!

7. Setkali jste se s nějakým problémem?

Každý detail se počítá!

8. Podařilo se Vám scénář 3 úspěšně splnit? *

- ◯ Ano
- ◯ Ne

9. Jak hodnotíte jednoduchost a přehlednost navigace v aplikaci v tomto scénáři? *

😄 😄 😄 😄 😄

Scénář 4

V tomto oddíle odpovídejte na otázky týkající se scénáře 4. Vzhledem k návaznosti scénářů v případě neúspěchu testování ukončete a odešlete formulář. Děkuji!

10. Setkali jste se s nějakým problémem?

Každý detail se počítá!

11. Podařilo se Vám scénář 4 úspěšně splnit? *

- ◯ Ano
- ◯ Ne

12. Jak hodnotíte jednoduchost a přehlednost navigace v aplikaci v tomto scénáři? *

😄 😄 😄 😄 😄

Závěr

Děkujeme za testování naší aplikace! Nakonec bychom Vám rádi položili několik dobrovolných otázek pro lepší pochopení.

13. Je v aplikaci nějaký opakovaný vzorec, který považujete za rušivý?

Každý detail se počítá!

Literatura

[1] Václav Řepa. *Podnikové procesy: procesní řízení a modelování*. 2., aktualiz. a rozš. vyd vyd.. Grada, 2007. ISBN 9788024722528.

[2] Mark von Rosing, Henrik von Scheel a August-Wilhelm Scheer. *The Complete Business Process Handbook*. Elsevier, 2015. ISBN 9780128004722 .

[3] Kanbanize. *What Is Plan-Do-Check-Act (PDCA) Cycle?* `https://kanbanize.com/lean-management/improvement/what-is-pdca-cycle`. Accessed: March 11, 2023.

[4] Michael Cousins. *As-is To-be model*. `https://blog.triaster.co.uk/blog/as-is-to-be-essential-business-model-process-improvement`. Accessed: December 15, 2022.

[5] *The Process Orchestration Handbook*. `https://camunda.com/process-orchestration/`. Accessed: December 12, 2022.

[6] Kees Hee, Natalia Sidorova a Jan Martijn Van der Werf. Business Process Modeling Using Petri Nets. 2013, 7480 DOI 10.1007/978-3-642-38143-0_4.

[7] Grady Booch, James Rumbaugh a Ivar Jacobson. *Unified Modeling Language User Guide*. Addison Wesley, 1998. ISBN 0-201-57168-4.

[8] Gregor Engels, Alexander Forster, Reiko Heckel a Sebastian Thone. *Process Modeling Using UML*. In: *Process-Aware Information Systems*. 2005.

[9] Wikimedia Commons. *File:BPM 300dpi.png — Wikimedia Commons, the free media repository*. 2020. `https://commons.wikimedia.org/w/index.php?title=File:BPM_300dpi.png&oldid=461949109`. [Online; accessed 14-Dec-2022.

[10] *What is Event-Driven Process Chain (EPC)?* `https://online.visual-paradigm.com/knowledge/business-design-tools/what-is-epc-diagram/`. Accessed: December 8, 2022.

[11] Marcelo Bernardino Araujo a Rodrigo Franco Gonçalves. *Selecting a Notation to Modeling Business Process: A Systematic Literature Review of Technics and Tools*. In: *Advances in Production Management Systems. Initiatives for a Sustainable World*. Cham: Springer International Publishing, 2016. 198–205. ISBN 978-3-319-51133-7.

[12] *The History of BPMN*. `https://bpmn.gitbook.io/bpmn-guide/what-is-bpmn/the-history-of-bpmn`. Accessed: December 27, 2022.

[13] BizzDesign. *Diagram types and data for BPMN modeling*. Website. unknown. `https://support.bizzdesign.com/display/knowledge/Diagram+types+and+data+for+BPMN+modeling##DiagramtypesanddataforBPMNmodeling--1283184360Process`. Accessed: December 26, 2022.

[14] *Business Process Model and Notation*. OMG, prosinec 2010. http://www.omg.org/spec/BPMN/2.0. Accessed: January 2, 2023.

[15] bpmn.io. *BPMN-JS - The BPMN 2.0 diagram toolkit*. Website. unknown. https://bpmn.io/toolkit/bpmn-js/. Accessed: March 23, 2023.

[16] JGraph Ltd. *Diagrams.net*. Website. unknown. https://www.diagrams.net/. Accessed: March 22, 2023.

[17] *Camunda Platform*. https://camunda.com/platform/. Accessed: December 11, 2022.

[18] *Bizagi Platform Overview*. https://www.bizagi.com/en/platform. Accessed: December 11, 2022.

[19] IBM. *IBM Business Process Manager V7.5 replaces Lombardi Teamworks and WebSphere Lombardi Edition*. Website. unknown. https://www.ibm.com/support/pages/ibm-business-process-manager-v75-replaces-lombardi-teamworks-and-websphere-lombardi-edition. Accessed: April 18, 2023.

[20] IBM. *Business process overview*. Website. unknown. https://www.ibm.com/docs/en/bpm/8.5.5?topic=manager-business-process-overview. Accessed: March 25, 2023.

[21] Zakhutskyi Viacheslav. *Simulátor modelování procesů v notaci BPMN*. 2020.

[22] Seliverstov Aleksandr. *Aplikace pro ověřování dodržování standardů notace BPMN*. 2020.

[23] GISELE SANTOS. *Alkonost and the Gamayun, the mythical beings of Slavic folklore*. Website. 2015. https://www.ancient-origins.net/myths-legends/alkonost-and-gamayun-mythical-beings-slavic-folklore-004076. Accessed: April 18, 2023.

[24] Tomislav Rozman a Gregor Polančič. *Analysis of most common process modelling mistakes in BPMN process models.* In: 2008. 233–246.

[25] The Editors of EncyclopaediaBritannica. *client-server architecture*. Encyclopedia Britannica. 2023. https://www.britannica.com/technology/client-server-architecture. Accessed: March 20, 2023.

[26] Baeldung. *Layered Architecture in C*. Website. unknown. https://www.baeldung.com/cs/layered-architecture. Accessed: March 19, 2023.

[27] Spring Framework Contributors. *Spring Framework Reference Documentation*. https://docs.spring.io/spring-framework/docs/6.0.7/reference/html/overview.html. 2023. Accessed on March 21, 2023.

[28] Spring. *Spring Framework MVC*. https://docs.spring.io/spring-framework/docs/3.2.x/spring-framework-reference/html/mvc.html. 2013. Accessed: March 25, 2023.

[29] Spring. *Spring Data*. https://spring.io/projects/spring-data. 2022. Accessed: March 25, 2023.

[30] Spring. *Spring Security*. https://spring.io/projects/spring-security. 2022. Accessed: March 25, 2023.

[31] Spring. *Spring GraphQL*. https://spring.io/projects/spring-graphql. 2022. Accessed: March 25, 2023.

[32] Spring. *Spring Authorization Server*. https://spring.io/projects/spring-authorization-server. 2022. Accessed: March 25, 2023.

[33] Spring. *Spring Security OAuth2 Resource Server*. https://docs.spring.io/spring-security/reference/servlet/oauth2/resource-server/index.html. 2022. Accessed: March 26, 2023.

[34] Camunda. *Camunda BPM Platform*. https://github.com/camunda/camunda-bpm-platform/tree/master/model-api/bpmn-model. 2022. Accessed: April 10, 2023.

[35] JGrapht Contributors. *JGrapht – Graphs in Java*. 2023. https://jgrapht.org/. Accessed: March 15, 2023.

[36] React. *React*. 2022. Accessed: April 10, 2023.

[37] Next.js. *Next.js*. https://nextjs.org/. 2022. Accessed: March 27, 2023.

[38] NextAuth.js. *NextAuth.js*. https://next-auth.js.org/. 2022. Accessed: March 27, 2023.

[39] Mantine. *Mantine*. https://mantine.dev/. 2022. Accessed: March 28, 2023.

[40] Apollo GraphQL. *Apollo GraphQL*. https://www.apollographql.com/docs/react/. 2022. Accessed: March 27, 2023.

[41] TypeScript. *TypeScript*. https://www.typescriptlang.org/. 2022. Accessed: March 28, 2023.

[42] bpmn.io. *bpmn-js*. https://bpmn.io/toolkit/bpmn-js/. 2022. Accessed: April 02, 2023.

[43] Facebook Engineering. *GraphQL: A data query language*. https://engineering.fb.com/2015/09/14/core-data/graphql-a-data-query-language/. 2015. Accessed: April 02, 2023.

[44] GraphQL. *Learn GraphQL*. https://graphql.org/learn/. 2022. Accessed: April 03, 2023.

[45] HowToGraphQL. *GraphQL vs REST: Overview*. https://www.howtographql.com/basics/1-graphql-is-the-better-rest/. 2022. Accessed: April 02, 2023.

[46] GraphQL. *Queries and Mutations*. https://graphql.org/learn/queries/. 2022. Accessed: April 03, 2023.

[47] *OAuth 2.0*. https://oauth.net/2/. Accessed: April 29, 2023.

[48] OpenID. *OpenID*. https://openid.net/connect/. 2023. Accessed: April 29, 2023.

[49] IEEE. IEEE Standard for Software Unit Testing. *ANSI/IEEE Std 1008-1987*. 1986, 1–28. DOI 10.1109/IEEESTD.1986.81001.

[50] CircleCI. *How to Test Software Part I: Mocking, Stubbing, and Contract Testing*. https://circleci.com/blog/how-to-test-software-part-i-mocking-stubbing-and-contract-testing/. 2022. Accessed: April 07, 2023.

[51] Usability.gov. *Usability Testing*. https://www.usability.gov/how-to-and-tools/methods/usability-testing.html. 2022. Accessed: April 07, 2023.