

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Interaktívny pracovný hárok pre výučbu logiky pre informatikov
Bakalárska práca

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Interaktívny pracovný hárok pre výučbu logiky pre informatikov
Bakalárska práca

Študijný program: Aplikovaná informatika

Študijný odbor: Aplikovaná informatika

Školiace pracovisko: Katedra aplikovanej informatiky

Školiteľ: Mgr. Ján Kľuka, PhD.



Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Nikolaj Kniha
Študijný program: aplikovaná informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: aplikovaná informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Interaktívny pracovný hárok pre výučbu logiky pre informatikov
Interactive worksheet for teaching of logic for informatics

Anotácia: Pri výučbe logiky pre aplikovaných informatikov využívame interaktívne webové nástroje, ktoré sú produktom vyučujúcich a niekoľkých bakalárskych prác a slúžia na kontrolu dôkazov alebo prieskum vzťahu prvorádových štruktúr a formúl. Tieto nástroje sú samostatnými webovými aplikáciami na strane klienta. Študenti ich môžu využiť pri vypracovaní zadania, ale zadania aj riešenia sú oddelené statické dokumenty a riešenia sa kontrolujú manuálne. V rámci tejto práce by mala vzniknúť webová aplikácia fungujúca ako interaktívny pracovný hárok spájajúci zadania a existujúce (a potenciálne aj budúce) nástroje na tvorbu riešení. Na ukladanie týchto hárkov pre teoretické cvičenia by sa mali využiť repozitáre verziovaného systému git na portáli github, ktoré v súčasnosti využívame pri praktických cvičeniach.

Cieľ: Implementovať webovú aplikáciu na strane klienta s prípadnou potrebnou serverovou podporou v zmysle anotácie.

Literatúra: Barker-Plummer, D., Barwise, J., Etchemendy, J.: Language, Proof and Logic. Second Edition. CSLI Publications, 2011.
Cifra, M.: Prieskumník sémantiky logiky prvého rádu. Bakalárska práca – Univerzita Komenského, Bratislava, 2018.
Genesereth, M., Kao, E.: Introduction to Logic. Third Edition. Morgan & Claypool, 2017.
Nyitraiová, A.: Educational tools for first order logic. Bakalárska práca – Univerzita Komenského, Bratislava, 2018.
Onódy, Z.: A proof assistant for first-order logic. Bakalárska práca – Univerzita Komenského, Bratislava, 2018.

Kľúčové slová: interaktívny pracovný hárok, výučba logiky, webová aplikácia, verziovací systém git, github

Vedúci: Mgr. Ján Kľuka, PhD.
Konzultant: RNDr. Jozef Šiška, PhD.
Katedra: FMFI.KAI - Katedra aplikovanej informatiky
Vedúci katedry: prof. Ing. Igor Farkaš, Dr.
Dátum zadania: 04.10.2019

Dátum schválenia: 07.10.2019

doc. RNDr. Damas Gruska, PhD.



Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

garant študijného programu

študent

vedúci práce

Pod'akovanie:

Abstrakt:

Abstract:

Obsah

1. Východiská.....	2
1.1. Webová aplikácia.....	2
2. Použité technológie	2
2.1. HTML5, CSS3, JavaScript.....	2
2.2. React.....	3
2.3. Redux	5
2.4. Bootstrap 4	6
2.5. Markdown	6
2.6. jQuery.....	6
2.6. Node.js	7
2.7. Express.js	7
2.8. npm.....	7
2.9. Firebase	7
2.10. Github.....	8
3. Použité bakalárske práce	8
3.1. Prieskumník sémantiky logiky prvého rádu.....	9
3.2. Educational tools for first order logic	9
3.3. A proof assistant for first-order logic.....	10
4. Existujúce riešenia	11
4.1. Jupyter Notebook	11
4.2. MATLAB.....	12
Literatúra.....	14

1. Východiská

1.1. Webová aplikácia

Webová aplikácia je aplikácia, ktorá je uložená na vzdialenom serveri, pomocou internetu poskytnutá klientovi a pomocou webového prehliadača zobrazená užívateľovi, čiže ide o aplikáciu typu klient-server.

Webové aplikácie môžu byť navrhnuté pre rôzne využitia a byť prístupné komukoľvek s pripojením na internet, v prípade firemných aplikácií s pripojením na intranet. Najväčšou výhodou je jednoduchá dostupnosť, pretože klient – webový prehliadač sa nachádza na väčšine moderných zariadení, označuje sa ako tenký klient, pretože nemá informácie o logike samotnej aplikácie.

Webová aplikácia pre svoju správnu funkčnosť potrebuje web server, aplikačný server a databázu. Web server sa stará o žiadosti, ktoré prichádzajú od klienta, aplikačný server tieto žiadosti vykonáva a databáza sa využíva na uchovávanie potrebných dát. [1]

2. Použité technológie

V tejto kapitole budú bližšie vysvetlené technológie, ktoré sú použité pri vytváraní našej webovej aplikácie.

2.1. HTML5, CSS3, JavaScript

HTML5 (HyperText Markup Language) je značkovací jazyk verzie 5 využívaný na tvorbu webových stránok. Dokáže vytvárať dokumenty obsahujúce text, hypertextové odkazy, skripty a ďalšie.

CSS (Cascading Style) je jazyk verzie 3 slúžiaci na popísanie zobrazenia elementov v dokumentoch typu HTML alebo XML. Umožňuje teda meniť vzhľad a rozmiestnenie elementov v dokumente.

JavaScript je multiplatformový a objektový skriptovací jazyk, ktorý umožňuje implementáciu komplexných funkcií prevažne na internetových stránkach. Väčšinou sa vkladá priamo do HTML kódu stránky a o interpretáciu tohto kódu sa zvyčajne stará internetový prehliadač klienta. Slúži hlavne na ovládanie dynamických prvkov stránky (tlačidlá, textové polia, atď.).

2.2. React

React je JavaScriptový framework, ktorý slúži na efektívne vytváranie používateľských rozhraní. Vytvorila a spravuje ju spoločnosť Facebook spolu s komunitou samostatných vývojárov. React sa dá využiť ako základ pre vývoj webovej alebo mobilnej „single-page“ aplikácie. Pretože React sa stará iba o renderovanie dát v DOM, na správne fungovanie celej aplikácie sú potrebné ďalšie knižnice na správu jeho stavu, routing a interakciu s API. Výhodou Reactu je spolupráca s už existujúcim kódom a je jednoduché používať ho s jQuery, Angularom, atď.. [2]

V Reacte máme možnosť vytvárať komponenty, ktoré vyzerajú ako elementy v jazyku html. Výhodou je však znovu použiteľnosť týchto komponentov. Pri programovaní v Reacte sa snažíme o vytváranie menších samostatných komponentov, ktoré budú použité v dynamickej webovej aplikácii.

2.2.1. Komponent

Komponent je väčšinou definovaný ako trieda v ktorej sa nachádza konštruktor a metóda render(). Okrem metódy render() sa v komponente môže nachádzať ľubovoľné množstvo ďalších metód, ktoré sú v ňom využívané.

```
class App extends React.Component {  
    constructor(props)  
    this.state = {  
        item : value  
    }  
    render()  
}
```

Každý komponent môže prijať informácie, ktoré mu boli zadané pri vytváraní pomocou Props, v ktorých sa môžu nachádzať dáta aj funkcie. Funkcia render() je zodpovedná za vykresľovanie požadovaných elementov.

2.2.2. Stav komponentu

Stav komponentu je jednoduchý objekt slúžiaci na ukladanie dát, zvyčajne sa označuje ako state, môže však mať aj iný názov. V prípade že sa dáta v stave zmenia,

funkcia render() znova prekreslí komponent, preto by sa v stave mali nachádzať iba dáta, ktoré priamo ovplyvňujú vykresľovanie komponentu.

Majme komponent App, ktorý zobrazuje číslo uložené v stave a jedno tlačidlo, ktorým toto číslo budeme zvyšovať, po každom zvýšení sa komponent prekreslí aby zobrazil aktuálnu hodnotu.

```
class App extends React.Component {  
  this.state = {  
    number : 0  
  }  
  render(){  
    return (  
      <div>{this.state.number}</div>  
      <Button onClick={handleClick}> + </Button>  
    )  
  }  
}
```

Aby sme mohli zvyšovať hodnotu uloženú v našom stave, musíme si zadať jednoduchú funkciu, ktorá nám bude túto hodnotu meniť. Stav sa nesmie meniť priamo, pre zmenu hodnoty musíme použiť funkciu setState.

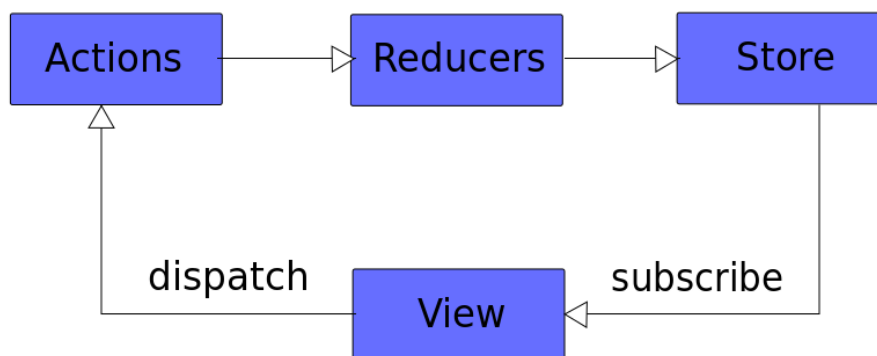
```
Const handleClick = () => {  
  Var x = this.state.number + 1  
  this.setState({number : x})  
}
```

Teraz sa nám po kliknutí na tlačidlo zmení hodnota v stave a to spôsobí prekreslenie a zobrazí sa nám nová, zvýšená hodnota.

Najväčšou výhodou Reactu je ReactDOM, teda virtuálny DOM vytvorený Reactom. Normálny DOM (Document Object Model) v jednoduchosti reprezentuje UI aplikácie a vždy keď nastane zmena, DOM sa aktualizuje aby zobrazil túto zmenu. DOM má stromovú štruktúru, takže zmena jeho údajov je rýchlo, problém je však, že po úprave sa musí prvom aj so všetkými potomkami znova vyrenderovať a to je pomalé. Virtuálny DOM však tento problém vyrieši. Virtuálny DOM je virtuálnou reprezentáciu DOM a pri každej zmene sa najskôr aktualizuje virtuálny DOM, ktorý sa porovná so skutočným DOM a vypočíta najlepší spôsob ako vykonať tieto zmeny aby nezabrali zbytočne veľa času a nemuseli sa prerenderovať aj prvky, ktoré sa nezmenili. [3]

2.3. Redux

Redux je open-source JavaScriptová knižnica slúžiaca na manažment stavu aplikácie spustenej vo webovom prehliadači. Stav aplikácie je objekt, v ktorom sú uchovávané dáta, s ktorými aplikácia pracuje a v akom stave sa celá aplikácia v danom momente nachádza. Všetky dáta sú tak uložené na jednom mieste, v stave, kde je k nim jednoduchý prístup.



Obrázok 2.3.1. – dátový tok v Reduxe

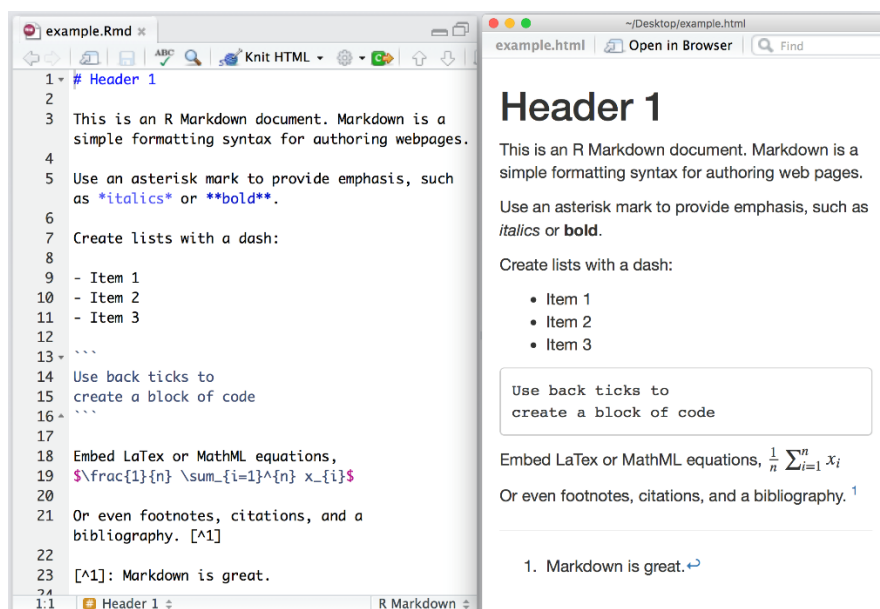
Na obrázku vidíme diagram znázorňujúci tok dát v Reduxe. V Actions sa nachádzajú klasické JavaScriptové objekty, ktoré nesú informácie o tom aká akcia sa vykonala, prípadne ďalšie atribúty a atribút type, ktorý identifikuje vykonanú akciu. Po zavolaní akcie z Reactového komponentu sa zavolá Reducer, čo je funkcia, ktorá prijíma action a state ako argumenty a následne zmení aktuálny stav podľa požadovanej akcie a vráti nový stav, ktorý sa pošle do Store a odtiaľ si ho preberie View čo spôsobí prerenderovanie daného Reactového komponentu podľa požiadaviek. To nám uľahčí spravovanie stavu ako aj oddelenie a sprehl'adenie kódu.

2.4. Bootstrap 4

Bootstrap 4 [4] je open-source CSS framework vytvorený spoločnosťou Twitter, ktorého úlohou je zjednodušiť vytváranie používateľského rozhrania. Obsahuje šablóny ako na samostatné časti užívateľského rozhrania tak aj celé predlohy. Vďaka tomu nie je potrebná veľká znalosť CSS a taktiež sa stará o zobrazovanie pre rôzne zariadenia.

2.5. Markdown

Markdown je odľahčený značkovací jazyk, ktorý slúži na úpravu čistého textu a následné konvertovanie do konkrétneho formátu, v našom prípade do HTML. Často sa používa na formátovanie príspevkov v diskusiách alebo jednoduchých dokumentov. V našom prípade bude použitý na formátovanie zadání úloh pre študentov.



Obrázok 2.5.1. – príklad Markdownu,

zdroj: https://rmarkdown.rstudio.com/authoring_quick_tour.html

2.6. jQuery

jQuery [5] je malá open-source JavaScriptová knižnica, navrhnutá na zjednodušenie prechádzania HTML DOM, spracovanie udalostí, animácie a Ajax. S modulárnym prístupom ku jQuery dovoľuje vytvárať pokročilé webové stránky a aplikácie.

2.6. Node.js

Node.js [6] je open-source multiplatformové prostredie slúžiace na vývoj aplikácií na strane servera v jazyku JavaScript. Slúži na spúšťanie JavaScriptového kódu mimo prehliadača, teda na strane servera. Vďaka tomu vytvára obsah dynamickej webovej stránky ešte pred tým, než sa pošle klientovi.

2.7. Express.js

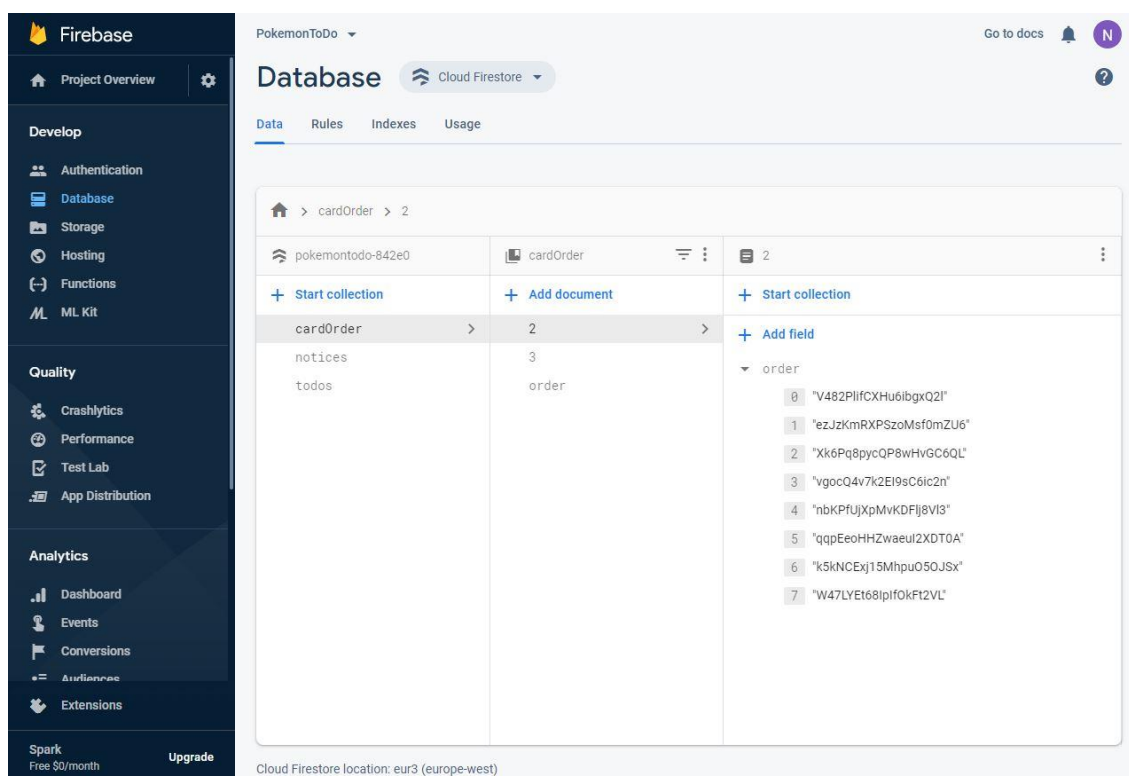
Express.js alebo iba Express [7] je open-source framework pre Node.js. Mení syntax programu, rozširuje funkcionalitu a zjednodušuje veľkú časť celej aplikácie, pretože nie je nutné starať sa o low level protokoly, procesy, atď..

2.8. npm

npm (Node Package Manager) slúži na sťahovanie a manažment balíkov potrebných pre náš projekt. Vďaka veľkej databáze dostupných balíkov je možné rýchlo a efektívne vytvárať a spravovať webové aplikácie.

2.9. Firebase

Firebase [8] je platforma, ktorá je zameraná na podporu procesu vývoja a rastu mobilných a webových aplikácií bez vlastného servera a znalosti programovania serverových častí. Na tejto platforme sa dajú vyvíjať aplikácie rýchlejšie a prehľadnejšie pretože dáta sú okamžite aktualizované. Nie je potrebné zaoberať sa detailmi ako sú dáta ukladané, synchronizované atď., pretože o všetko sa stará Firebase automaticky. Dáta sú v databáze ukladané ako dokumenty typu JSON, no je možné ukladať tam aj súbory iných typov. Firebase sa tiež stará o autentifikáciu používateľov a hostovanie celej webovej aplikácie. Pre vývojára je výhodou aj sledovanie a analýza aplikácie počas jej behu aj výpadku. Pre prácu s dokumentmi mimo webové rozhranie má firebase vytvorené vlastnú knižnicu pre React – react-redux-firebase, vďaka ktorej sa dokumenty v databáze



Obrázok 2.9.1. – Webové užívateľské prostredie Firebase

2.10. Github

Git je verziovací nástroj na zdieľanie zdrojových súborov softvérových projektov. Je založený na myšlienke decentralizovaných rovnocenných repozitárov. [9]

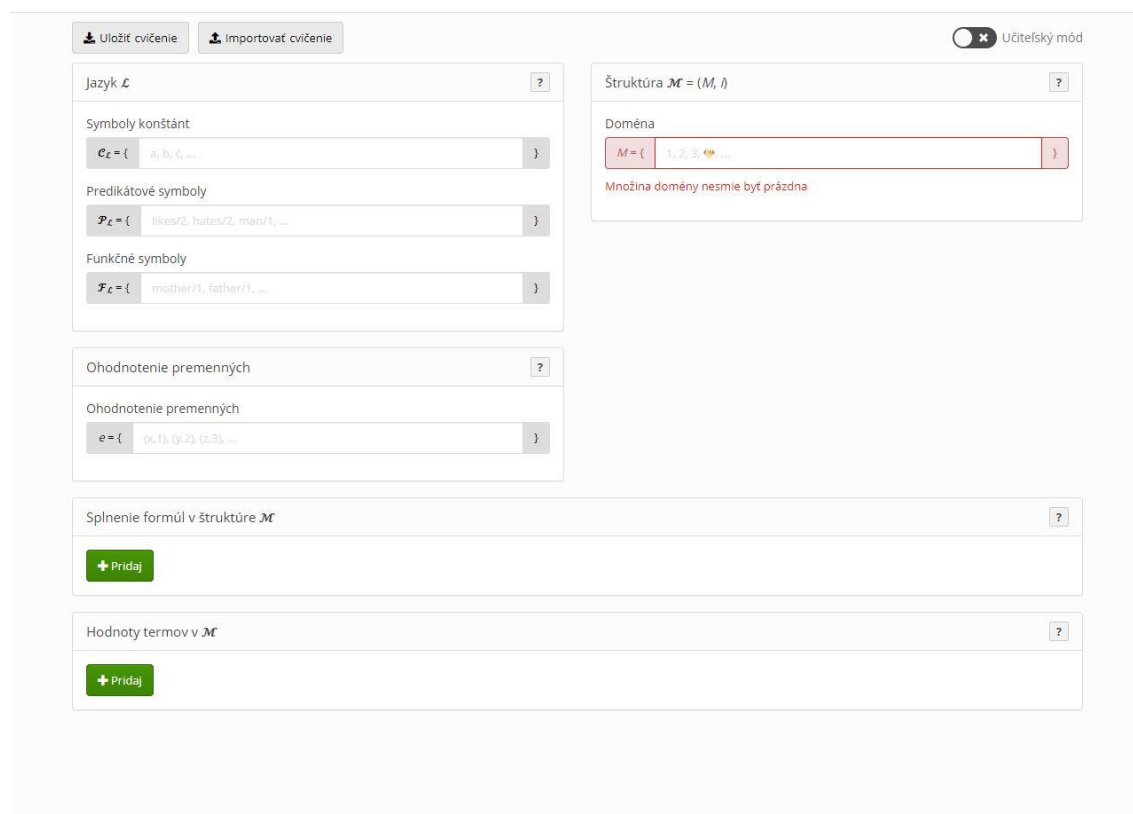
Github je open-source webová služba určená na ukladanie a prístup ku repozitárom. Umožňuje ukladanie zdrojových kódov vo viacerých programovacích jazykoch a sleduje všetky zmeny v týchto súboroch.

3. Použité bakalárske práce

Kapitola obsahujúca informácie o bakalárskych prácach, ktoré sú súčasťou tejto bakalárskej práce. Každá z nich bude v aplikácii vložená ako celok a pretože sú vytvorené tak, aby sa výsledky dali exportovať ako dokument typu JSON, tieto dokumenty budeme ďalej spracovávať aby sa dali zobrazit' vo vyhovujúcom formáte.

3.1. Prieskumník sémantiky logiky prvého rádu

Práca Milana Cifru [10] sa zaoberá tvorbou webovej aplikácie, ktorá ponúka študentom precvičovanie štruktúr logiky prvého rádu. V aplikácii si užívateľ môže definovať ľubovoľnú konečnú štruktúru, jazyk a výrazy (formuly a termy). Výrazy aplikácia vyhodnocuje na základe definovanej štruktúry. Taktiež kontroluje syntax výrazov, čím vynucuje ich správny zápis. Aplikácia môže byť v dvoch módoch, ktoré si používateľ prepína - v študentskom a učiteľskom, ktorý umožňuje zamykanie výrazov a častí definície štruktúry. Vytvorenú štruktúru a výrazy je možné exportovať do textového súboru, a ten sa dá naspäť importovať. Aplikácia je naprogramovaná pomocou knižníc React a Redux.



Obrázok 3.1.1. – Prieskumník sémantiky logiky prvého rádu

3.2. Educational tools for first order logic

Aplikácia vytvorená Alexandrou Nyitraiovou [11] slúži na vytváranie dôkazu podľa metódy analytického tabla. Dôkaz nie je vytváraný automaticky, ale kontroluje každý krok užívateľa hneď po jeho vykonaní, v prípade omylu vypíše pri príslušnej formule chybu, týmto je aplikácia zameraná na edukáciu. Editor podporuje dôkazy v prvorádovej ale aj výrokovej logike. Dôkazy sú zobrazované ako stromy a každý

vrchol je reprezentovaný formulou, tie sa odvodzujú pomocou základných štyroch pravidiel: alfa, beta, gama a delta.

Aplikácia je naprogramovaná v jazyku Elm.

Prettify formulas Print Export as JSON Import from JSON Undo Redo

(1) [1]
 Invalid formula: { row = 1, col = 1, source = "", problem = BadOneOf ([ExpectingKeyword "T",ExpectingKeyword "F"]), context = [] }

+ Add Change Delete Close

This tableau doesn't prove anything.

Problems

- (1) Invalid formula: { row = 1, col = 1, source = "", problem = BadOneOf ([ExpectingKeyword "T",ExpectingKeyword "F"]), context = [] }

Help

Symbols of propositional and first-order logic

Symbols of conjunction	Symbols of disjunction	Symbols of implication	Symbols of negation	Universal quantifier	Existential quantifier
&, /\, ^	, \/, \vee	->, ->	~, ~, ~	\forall , \A, \forallforall, \a	\exists , \E, \existsexists, \e
strictly binary	strictly binary	strictly binary	unary	First order logic term	First order logic term

Important notes

Note	Example
Each of the nodes contains a signed formula, i.e. it must be prefixed by T or F.	T forall x P(x) F exists x forall p (K(x, q) ^ G(p, x))
To enter a premise / assumption (which you want to prove), make it reference itself	(1) T (a -> b) [1] (i.e. "(1) F [1]")
When substituting, choose only such term which does not contain a variable which looks like bound in referenced formula.	wrong example: (1) T forall x exists k P(x,k) [1] (2) T exists k P(k,k) (x-k) [1]
When applying delta rule make sure to use completely new constant, which was not used as free (better bound as well) in a node somewhere above.	wrong example: (1) T L(p) [1] (2) T exists x forall k P(x,k) [2] (3) T forall k P(p,k) (x-p) [2]

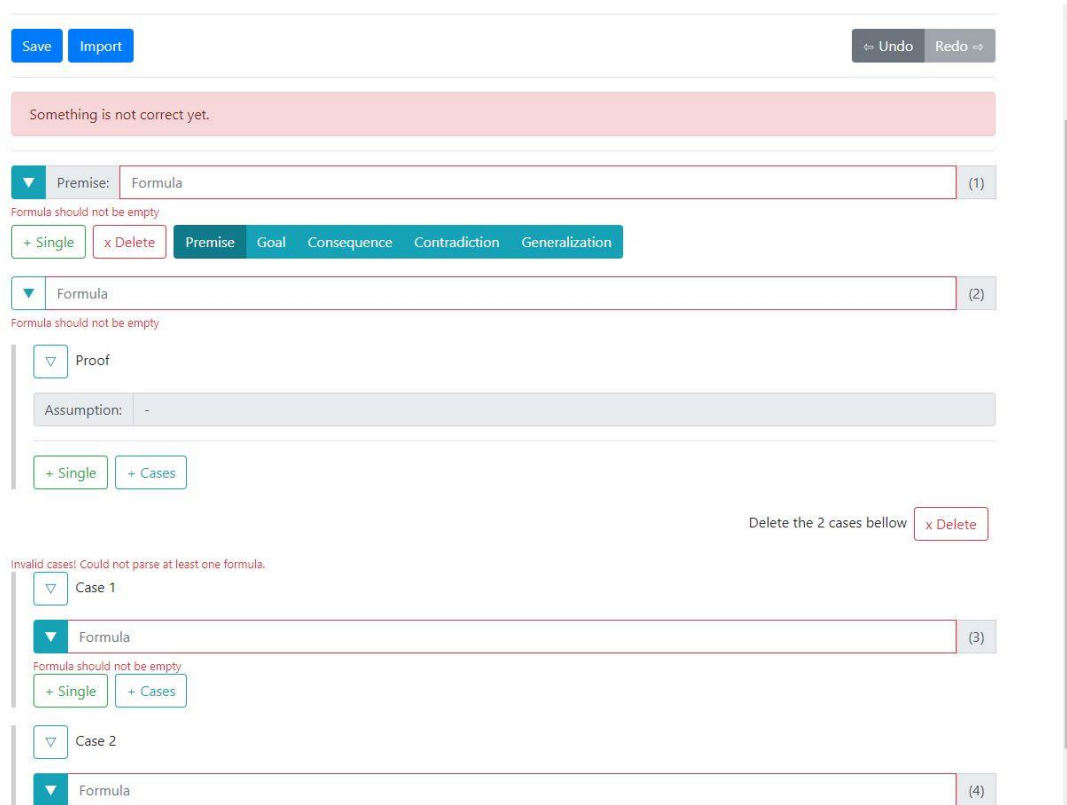
Applying rules

	α -rule					β -rule			γ -rule		δ -rule	
rules	T (A^B)	F (A^B)	F (A -> B)	T ~A	F ~A	F (A^B)	T (A^B)	T (A -> B)	T \forall x P(x)	F \exists x P(x)	F \forall x P(x)	T \exists x P(x)
	T A	F A	T A	F A	T A	F A F B	T A T B	F A T B	T P(x)	F P(x)	F P(x)	T P(x)
	T B	F B	F B									
example	(1) T (a^b) [1] (2) T a [1] (3) T b [1]					(2) T a [1]			(1) T forall x P(x) [1] (2) T P(k) (x-k) [1]		(1) T forall x P(x) [1] (2) T P(k) (x-k) [1]	

Obrázok 3.2.1. - Educational tools for first order logic

3.3. A proof assistant for first-order logic

Aplikácia Zoltána Onódyho [12] funguje ako interaktívny webový dokazovací asistent, ktorý podporuje tri typy dôkazov: priamy dôkaz, dôkaz analýzou prípadov a dôkaz sporom. Keďže je dôkaz založený na reťazení tvrdení, aplikácia overuje či sú nové tvrdenia dôsledkami tých predchádzajúcich a nejakého inferenčného pravidla. Používateľ do aplikácie zadáva sformalizované prvorádové formuly, ktoré sú potom vyhodnocované. Aplikácia je naprogramovaná v jazyku Elm.



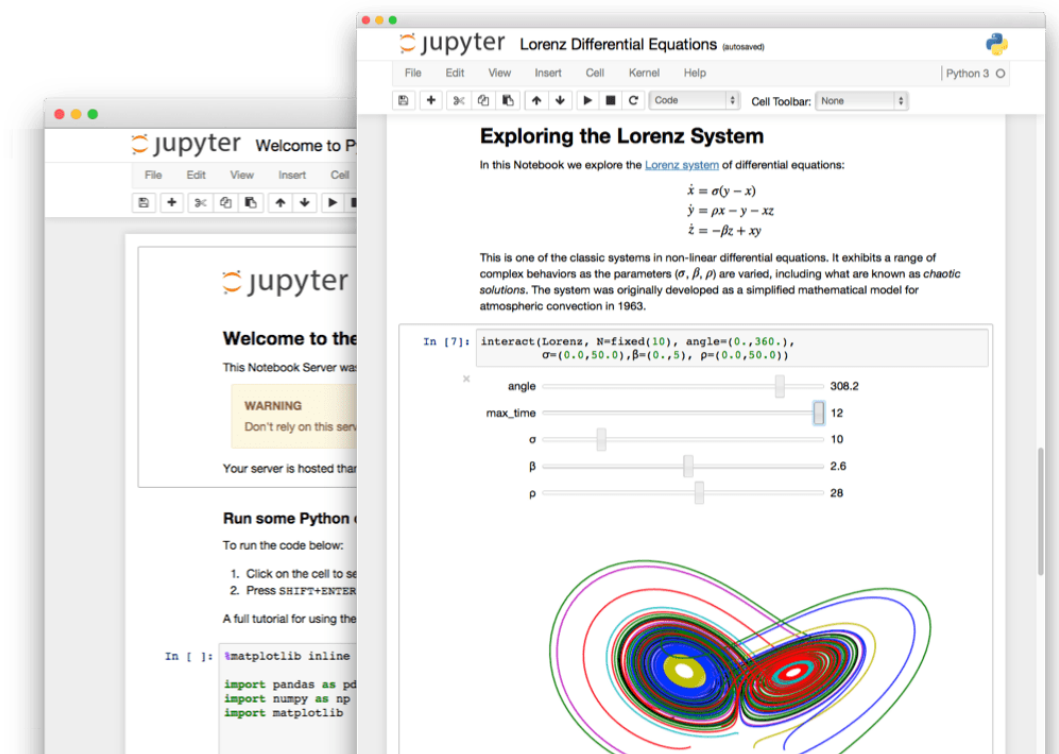
Obrázok 3.3.1. - A proof assistant for first-order logic

4. Existujúce riešenia

V tejto kapitole je popísaný existujúci softvér, nad ktorým sme uvažovali no z nižšie uvedených dôvod sa preň nerozhodli.

4.1. Jupyter Notebook

Jupyter Notebook [13] je interaktívne výpočtové prostredie vo webovom prehliadači, ktoré slúži na vytváranie Jupyter notebook dokumentov. Tieto dokumenty obsahujú JSON dokument a zoradený zoznam input/output políček, ktoré môžu obsahovať kód, text (podporujúci Markdown), matematické výroky, atď.. Dokument tohto typu sa dá konvertovať do ďalších rôznych formátov ako HTML, LaTeX, PDF, Markdown, Python. Vďaka možnosti konvertovať dokument do HTML, je možné dokumenty prezerať aj priamo priamo v prehliadači. Pre programovaciu časť aplikácie je možné pripojiť sa na rôzne jadrá s rôznymi programovacími jazykmi.

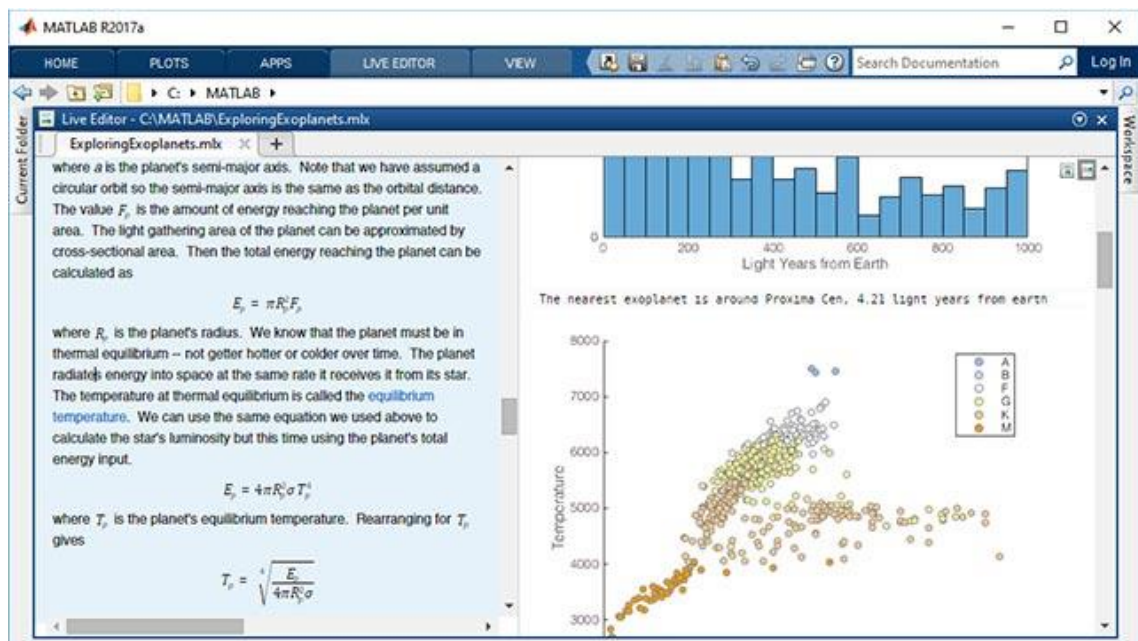


Obrázok 4.1.1. – Jupyter Notebook interface,
zdroj: https://en.wikipedia.org/wiki/Project_Jupyter

Jupyter Notebook teda spĺňa takmer všetky podmienky, ktoré sme mali keďže umožňuje zapisovať ako matematické výroky tak aj programovať v rôznych jazykoch. Pre výuku je však potrebné využívať aj už existujúce edukačné aplikácie, ktoré sú v rôznych programovacích jazykoch – JavaScript a Elm. Jupyter Notebook však nepodporuje priame zakomponovanie týchto aplikácií.

4.2 MATLAB

MATLAB [14] je interaktívne programové prostredie a skriptovací programovací jazyk. Ponúka veľa funkcií ako vykresľovanie 2D a 3D grafov, implementáciu algoritmov, prezentáciu dát, vytváranie používateľských rozhraní a podobne. Názov MATLAB vznikol z skrátením slov Matrix Laboratory, čiže Maticové Laboratórium, pretože kľúčovou dátovou štruktúrou pri výpočtoch sú práve matice.



Obrázok 4.2.1. – Matlab interface,
zdroj: <https://www.mathworks.com/products/matlab.html>

MATLAB je silným nástrojom, ktorý zvládne aj náročné výpočty a simulácie, nič z toho však nepotrebujeme. Keďže cieľom tejto práce je uľahčiť študentom aj vyučujúcim prácu na predmete a riešenie úloh, MATLAB je na tento účel až príliš komplikovaný a keďže na jeho používanie je potrebná licencia, tak aj finančne náročný.

Literatúra

- [1] Wikipedia contributors, „Webová aplikácia,“ [Online]. Available: https://sk.wikipedia.org/wiki/Webov%C3%A1_aplik%C3%A1cia. [Cit. 26. 1. 2020].
- [2] Wikipadia contributors, „React (web framework),“ [Online]. Available: [https://en.wikipedia.org/wiki/React_\(web_framework\)](https://en.wikipedia.org/wiki/React_(web_framework)). [Cit. 26. 1. 2020].
- [3] M. Hamedani, „React Virtual DOM Explained in Simple English,“ 3. 12. 2018. [Online]. Available: <https://programmingwithmosh.com/react/react-virtual-dom-explained/>.
- [4] Bootstrap team, „Bootstrap,“ [Online]. Available: <https://getbootstrap.com/>. [Cit. 22. 1. 2020].
- [5] The jQuery Foundation, „jQuery,“ [Online]. Available: <https://jquery.com/>. [Cit. 22. 1. 2020].
- [6] OpenJS Foundation, „Node.js,“ [Online]. Available: <https://nodejs.org/en/>. [Cit. 24. 1. 2020].
- [7] Node.js Foundation, „Express.js,“ [Online]. Available: <https://expressjs.com/>. [Cit. 26. 1. 2020].
- [8] Google, „Firebase,“ [Online]. Available: <https://firebase.google.com/>. [Cit. 26. 1. 2020].
- [9] P. Petrovič, „Tvorba informačných systémov,“ [Online]. Available: <http://dai.fmph.uniba.sk/courses/tvorbaIS/ex/git/>. [Cit. 25. 1. 2020].
- [10] M. Cifra, *Prieskumník sémantiky logiky prvého rádu*, Bratislava: Univerzita Komenského, 2018.
- [11] A. Nyitraiová, *Educational tools for first order logic*, Bratislava: Univerzita Komenského, 2018.
- [12] Z. Onódy, *A proof assistant for first-order logic*, Bratislava: Univerzita Komenského, 2018.
- [13] Quora, 7. 1. 2019. [Online]. Available: <https://www.quora.com/What-exactly-is-%E2%80%98Jupyter-Notebook%E2%80%99>. [Cit. 20. 1. 2020].

- [14] MathWorks, „MATLAB,“ [Online]. Available:
<https://www.mathworks.com/products/matlab.html>. [Cit. 23. 1. 2020].