Posudek bakalářské práce

Matematicko-fyzikální fakulta Univerzity Karlovy

Autor práce	Artem Bakhtin
Název práce	A tool for querying multi-model data
Rok odevzdání	2023
Studijní program	Informatika
Specializace	Databáze a web

Autor posudku	Jáchym Bártík		Oponent
Pracoviště	Katedra softwarového inženýrství		

K celé práci

	lepší	OK	horší	nevyhovuje
Obtížnost zadání	X			
Splnění zadání		X		
Rozsah práce *... textová i implementační část, zohlednění náročnosti*		X		

Cílem bakalářské práce je navrhnout a naprogramovat grafický dotazovací jazyk pro multi-modelové databáze. Přitom se navazuje na dřívější snahu vedoucího práce a jeho kolegů o popis multi-modelových dat pomocí teorie kategorií. Jelikož tyto snahy vedly na vytvoření modelu, který si lze zjednodušeně představit jako orientovaný multigraf, tato bakalářská práce začíná analýzou současných dotazovacích jazyků nad grafovými databázemi a rozborem nástrojů, které vizualizují výsledky dotazů v těchto jazycích.

Bohužel, tato část nemá konkrétní závěr. Dále, některé popisované nástroje podporují vizualizaci výsledků, ale o žádném není zmíněno, že by umožňoval sestavit dotaz pomocí grafického rozhraní. *Vzhledem k tomu, že právě vizuální nástroj pro sestavení dotazu je podstatou této práce, bych se chtěl autora zeptat, jestli prozkoumal i některé další nástroje, které tuto funkcionalitu mají.*

Zbytek práce je věnován návrhu a následné implementaci VQL (Visual Query Language). Ten podporuje základní databázové operace (projekci, selekci, řazení a další) a překládá se do jazyka Cypher, který je možné spustit nad Neo4j databází. Jazyk je pouze podmnožinou jazyka Cypher, což je samozřejmě v pořádku vzhledem k rozsahu jazyka Cypher. Jako nejzásadnější omezení se mi jeví nemožnost procházet přes více než jednu hranu, tzn. vždy můžeme pracovat pouze s kořenovým vrcholem a jeho přímými sousedy. *Rád bych se autora zeptal, jaké jsou důvody této limitace, a zda už přemýšlel na způsoby, jak by se daly překonat.*

Textová část práce

	lepší	OK	horší	nevyhovuje
Formální úprava *... jazyková úroveň, typografická úroveň, citace*			X	
Struktura textu *... kontext, cíle, analýza, návrh, vyhodnocení, úroveň detailu*		X		
Analýza		X		
Vývojová dokumentace	X			
Uživatelská dokumentace		X		

Kvalitu práce mírně sráží časté gramatické chyby a překlepy. To samozřejmě není nic, co by se nedalo rychle opravit pomocí Grammarly nebo podobného nástroje, ale celkově to přispívá k horší čitelnosti. Podobně je na tom typografie, kde bych zmínil, že různá textová prostředí zvýrazní kód mnohem lépe, než uvozovky či vložení kódu jako obrázku.

Analýza je celkem podrobná, nicméně jí, jak jsem zmiňoval výše, chybí konkrétní závěr. V programátorské dokumentaci je popsáno vše důležité, takže tam nevidím problém. K uživatelské dokumentaci musím zmínit, že návod na instalaci by rozhodně neměl začínat větou ve významu *otevřete si konkrétní IDE*. Konkrétní instrukce v textu naštěstí také jsou, ale musel jsem je hledat (což zase souvisí s typografickou výtkou výše).

Implementační část práce

	lepší	OK	horší	nevyhovuje
Kvalita návrhu *... architektura, struktury a algoritmy, použité technologie*		X		
Kvalita zpracování *... jmenné konvence, formátování, komentáře, testování*	X			
Stabilita implementace	X			

Při implementaci aplikace musel autor použít celou řadu technologií. Backend je napsán v Javě, konkrétně ve frameworku Spring Boot. Přitom následuje doporučenou architekturu po sobě následujících vrstev (databázové, perzistenční, servisní a prezenční). Na druhou stranu, téměř veškerá logika (tj. parsování grafu, který představuje dotaz v jazyce VQL) je pak umístěna ve třídě `Parser` na servisní vrstvě. Z hlediska další rozšiřitelnosti aplikace bych se spíš zamyslel nad tím, jak tuto logiku rozdělit – například aby parsování různých struktur jazyka VQL bylo řešeno v jim odpovídajících třídách.

Frontend je naprogramován v Javascriptu, konkrétně v Reactu. Podobně jako u backendu, i tady kód kopíruje doporučenou architekturu, nicméně většina logiky je opět na jednom místě, konkrétně v jednom Reduxovém slice, a to včetně věcí, které by bylo lepší řešit lokálně. Nepovažoval bych to ale za zásadní nedostatek, v současném řešení to problém není a později bude stačit drobný refactoring.

Kód jako takový je čitelný, konzistentní a přiměřeně okomentovaný. Klíčové vlastnosti jazyka VQL jsou otestovány (na backendu). Aplikace funguje podle popisu v textu práce.

Celkové hodnocení spíše lepší Velmi dobře

Práci navrhuji na zvláštní ocenění Ne

Datum 20.06.2023 Podpis