Název práce: Webové šachy pomocí Socket.IO technologie
Řešitelka: Bc. Ondřej Synek
Vedoucí práce: Ing. Zdeněk Šilar, Ph.D.

V rámci práce vytvořila řešitel webovou aplikaci pro dva a více hráčů, pracující v reálném čase. Vytvořil poměrně rozsáhlý software, skládající se ze dvou základních částí, serverové, běžící v prostředí technologie node.js a klientské, běžící v prohlížeči jednotlivých hráčů. Veškerý software byl implementován v jazyce Javascript, respective jeho novější nadstavbě ES6.

Výsledná šachová aplikace má vyzrálý, vyladěný grafický design, figurky se renderují v prohlížeči ve 3D pomocí knihovny babylon.js. Výhodou použitého řešení je to, že serverová i klientská část může sdílet funkcionalitu, implementovanou v javascriptových modulech. Pro zobrazení UI je využita moderní knihovna react.js. Serverová část používá objektovou databázi MongoDB pro uložení stavu šachové partie a přihlášených uživatelů (hráčů). Pro real-time komunikaci mezi klientem a serverem, nezbytnou pro správné fungování a rychlé odezvy při šachové partii, je vhodně použitá technologie webových socketů, s využitím knihovny Socket.io.

Cíl práce, tj. navrhnout a implementovat weboovu aplikaci pracující v reálném čase, která představuje elektronickou podobu deskové šachové hry, byl splněn.

Práce je psaná srozumitelně a je vhodně strukturovaná do kapitol. Zbytečně dlouhé mě však přijdou kapitoly o pravidlech hry šachy. Jsou všeobecně dobře známé, pro samotnou práci nejsou nijak důležité, a jsou dobře popsané v jiných zdrojích. Pouhý odkaz na zdroj, kde jsou pravidla šachu vysvětlená, by zcela postačoval. Navíc je popis povolených pohybů šachových figur částečně zduplikován v kapitole o implementaci chování šachovnice.

Delší bych uvítal kapitolu o testování aplikace a použitých metodách pro otestování.

Teoretický popis použitých technologií, jako je React, ES6, Babel, Flux aj., svědčí o dobrém porozumění řešitele dané problematice a je vhodnou součástí diplomové práce. Tento popis potom velmi pomáhá pro porozumění dalšího textu v kapitole o návrhu a implementaci. Informace byly nejenom podávány, ale i uváděny a vysvětlovány ve vzájemných souvislostech (např. nejprve představení technologie Flux, poté Redux, a pak vysvětlení, v čem je Redux lepší).

Co se týče formální úpravy, obsahuje řadu pravopisných chyb a překlepů, jejichž množství je nad mírou, která považuji za akceptovatelnou u tohoto typu práce – např. „animace přišli“, „znevýhodňit“, „označí“ jsou do očí bijící.

I přes tyto drobné nedostatky textové části, zásadní pro posouzení kvality diplomové práce, jejímž výstupem je software, je pro mne kvalita návrhu a vlastní implementace, se kterou jsem velmi spokojen.

<u>Závěr:</u>
Práci doporučuji k obhajobě a hodnotím kvalifikačním stupněm výborně. Celkově oceňuji ambiciózní rozsah a množství vynaložené práce, při níž bylo nutné vyřešit řadu programovacích i návrhových úloh, real-time komunikaci pomocí socketů a pospojování řady různorodých technologií a přístupů. Celkově práce přinesla přinesla nadstandartní výsledek,

za nějž by se nemusel stydět ani zkušený programátor. K tomu řešiteli i vedoucímu práce srdečně gratuluji.

Otázky k dovysvětlení při obhajobě:

- Uvádíte, že jste aplikaci řádně otestoval na různých OS a prohlížečích. Jakou metodiku testování jste použil (jak jste při testování postupoval)?
- Bude technologie webových socketů fungovat i hráčům, kteří sdílejí s jinými uživateli IP adresu poskytovatele?
- Šachové figurky jste modeloval sám?
- Váš projekt by se dle mého názoru vhodně doplňoval s projektem KinectChess, co si o tom myslíte? KinectChess totiž neumožní v současné podobě hrát nevidomým hráčům spolu přes Internet, pouze proti počítači.

Pár připomínek k vlastní práci:

- U zavedených anglických termínu nepoužívejte české překlady, působí až humorně (např. Listener místo Posluchač)
- Statické metody getMoves pro jednotlivé typy figurek bych doporučil přesunout do samostatných tříd, tj. Rock.getMoves(), Knight.getMoves() atp.
- Není vhodné směšovat v kódu/návrhu aplikace české a anglické názvy. Vždy používejte pouze anglické názvy, nikdy nevíte, kdo se k vám v budoucnosti přidá do týmu nebo bude ke kódu přistupovat, angličtině rozumí většina (např. v práci máte setFigurka, ale mělo by být setPiece)
- Příště prosím o tisknutí oboustraně, šetří to lesy. Tiskl jsem takhle i dizertační práci a nebyl s tím žádný problém.

Ing. Pavel Jetensky, PhD

6.9. 2017