STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

Obor: 18. Informatika

Webový portál BurzaŠkol.Online

Vít Falta

Pardubice 2021

STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

WEBOVÝ PORTÁL BURZAŠKOL.ONLINE

BURZAŠKOL.ONLINE WEB PORTAL

AUTOR	**Vít Falta**
ŠKOLA	**DELTA - Střední škola informatiky a ekonomie, s.r.o.**
KRAJ	**Pardubický**
ŠKOLITEL	**RNDr. Jan Koupil, Ph.D.**
OBOR	**18. Informatika**

Pardubice 2021

Prohlášení

Prohlašuji, že svou práci na téma *Webový portál BurzaŠkol.Online* jsem vypracoval/a samostatně pod vedením RNDr. Jana Koupila, Ph.D. a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Dále prohlašuji, že tištěná i elektronická verze práce SOČ jsou shodné a nemám závažný důvod proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a změně některých zákonů (autorský zákon) v platném znění.

V Pardubichích dne: _____ _____

Vít Falta

Poděkování

Děkuji svému školiteli RNDr. Janu Koupilovi, Ph.D. za obětavou pomoc, podnětné připomínky a nekonečnou trpělivost, kterou mi během práce poskytoval. Dále bych chtěl poděkovat panu Jiřímu Formánkovi za poskytnutí myšlenky projektu, podnětů a propagace projektu.

Na závěr bych chtěl poděkovat pánům Matěji Půhonému a akademickému malíři Danielu Václavíkovi za grafický návrh a pomoc s grafickou stránkou projektu.

Anotace

Cílem této práce je vytvořit online platformu, která umožní nahrazení klasických burz škol v České Republice, pomocí moderních webových technologií v době pandemie Covidu-19. Tato platforma zprostředkuje kontakt mezi středními školami a žáky 9. ročníků základních škol v době, kdy jsou klasické burzy škol z hygienických důvodů nemožné.

Klíčová slova

burza škol; webová stránka; www; online; online platforma

Annotation

The goal of this work is to create an online platform that will allow the replacement of traditional school presentation fairs in the Czech Republic using modern web technologies during the Covid-19 pandemic. This platform mediates contact between secondary schools and pupils in the 9th grade of primary schools at a time when traditional school exchanges are impossible due to hygienic restrictions.

Keywords

school exchange; web page; www; online; online platform

Obsah

Úvod **8**

1 Teoretická část **9**

1.1 Serverová infrastruktura 9

1.1.1 Load Balancery 9

1.1.2 Objektová úložiště 10

1.1.3 Relační databáze 10

1.1.4 Webové servery 11

1.2 Aplikační architektura 12

1.2.1 HTTP 12

1.2.2 PHP 12

1.2.3 PHP Frameworky 12

1.2.4 Webové sokety 13

1.2.5 Datový model 14

1.3 Vývojové prostředí 14

1.3.1 Integrovaná vývojová prostředí (IDE) 15

2 Realizace projektu **16**

2.1 Návrh datového modelu 16

2.2 Vývoj projektu 18

2.2.1 Využité technologie 18

2.2.2 Přípravy na vývoj projektu 20

2.2.3 Vývoj projektu ve frameworku Laravel 21

2.3 Nasazení a provoz projektu 23

2.3.1 Hostingová služba 23

2.3.2 Serverová architektura 25

2.3.3 Postup nasazení aplikace 25

2.3.4 Nasazování aktualizací aplikace 26
2.4 Bezpečnost aplikace 27
2.4.1 Aplikační zabezpečení 27
2.4.2 Zabezpečení infrastruktury 29
2.5 Postup projektu 30
2.5.1 Jedno-serverové nasazení 30
2.5.2 Více-serverové nasazení 31
2.5.3 Problém s pracovními vlákny load-balanceru Nginx . . 31
2.6 Provoz a přínos aplikace 32
2.6.1 Provoz aplikace 32
2.6.2 Zapojení škol 34
2.6.3 Přínos aplikace 34

Závěr **36**

Seznam obrázků **37**

Seznam literatury **44**

Slovník **48**

Úvod

Každý podzim se v České Republice konají tzv. výstavy středních škol, nebo také burzy, či veletrhy vzdělávání (názvy se liší kraj od kraje). Hromadné akce na kterých se střední školy snaží zaujmout co nejvíce žáků 9. tříd základních škol a vylíčit svou školu v nejlepším světle.

Žáci 9. třídy mají možnost si v jeden čas a na jednom místě prohlédnout nabídky jednotlivých škol. Pokud je nějaká škola svými obory zaujme mají možnost získat podrobnější informace od zástupců dané školy.

Vzhledem k pandemické situaci roku 2020 spojené s nemocí COVID-19 nebylo možné tyto burzy uspořádat. Systém *BurzaŠkol.Online* umožnil přesunutí celého náboru studentů do virtuálního prostředí internetu.

Jak to funguje?

BurzaŠkol.Online je internetový portál zprostředkující kontakt mezi žáky devátých tříd a středními školami za pomoci on-line video konferencí (nejčastěji MS Teams[1], Google Meet[2], Zoom[3], ...). Běžnému návštěvníkovi portál nabízí seznam virtuálních burz škol, ve kterém má každá burza přiřazený termín a lokalitu konání.

Střední školy se mohou registrovat k jednotlivým burzám a vložit odkaz na svou on-line konferenci. V den konání burzy se mohou žáci devátých tříd kliknutím na odkaz připojit do on-line konference vybrané školy.

Střední školy mají na portále založený svůj profil s informacemi o škole, seznam oborů a další informace jako informační brožura, či výsledky v soutěžích.

Kapitola 1

Teoretická část

1.1 Serverová infrastruktura

Tato sekce popisuje prvky serverové infrastruktury, které jsou nasazeny v reálném řešení *BurzaŠkol.Online*.

1.1.1 Load Balancery

Load balancer[4] je prvek serverové infrastruktury určený k rozložení příchozích požadavků na skupinu backendových serverů. K rozložení požadavků je využíváno nejrůznějších algoritmů, například Round Robin, Least Connections, či IP Hash. Dělíme je na SW load balancery a HW load balancery.

HW load balancery jsou zpravidla proprietární a nákladná řešení. Vysokého výkonu dosahují využitím dedikovaných HW akcelerátorů. Jedinou možností jejich škálování je zakoupení více fyzických strojů, či zakoupení výkonějších strojů.

SW load balancery jsou distribuovány ve formě uživatelských aplikací spustitelných na běžném hardwaru, či ve virtuálních strojích cloudových providerů. Škálovatelnost je tedy velice triviální. V porovnání s HW load balancery stačí pouze zakoupit on-demand virtuální stroj s odpovídajícím výpočetním výkonem. Tato vlastnost činí z SW load balancerů velice flexibilní řešení, ideální pro agilní cloudové prostředí.

1.1.2 Objektová úložiště

Objektové úložiště je typ úložiště, určený k ukládaní nestrukturovaných dat. Na rozdíl od tradičních způsobů ukládání dat, jakými jsou souborové systémy a bloková úložiště, pracují objektová úložiště[5] s daty jako se samostatnými objekty. Každý objekt obsahuje data samotná, uložená v jejich nativním formátu, a metadata. Metadata obsahují informace jako jsou přístupová práva objektu, unikátní globální identifikátor a další přidružené informace, například formát uložených dat. Díky adresaci za pomocí globálních identifikátorů je možné objekty transparentně přesouvat. Tato vlastnost nám poskytuje nezávislost dat na použitém záznamovém médiu.

Je tedy například možné přesunout málo používané objekty z diskového úložiště na úložiště založené na magnetické pásce. Na druhou stranu ale také můžeme transparentně přesunout často požadované objekty z úložiště do pracovní paměti pro rychlý přístup.

Objektová úložiště dovolují ukládání velkého objemu nestrukturovaných dat. Tato úložiště jsou proto ideální pro uživatelské obrázky, dokumenty a zálohy. Díky své flexibilitě jsou objektová úložiště výrazně levnější než například bloková.

1.1.3 Relační databáze

Relační databáze[6] jsou úložiště strukturovaných dat, založená na tzv. relačním modelu. Relační model organizuje data do *tabulek*, *řádků* — záznamů a *sloupců*. Vzájemné vztahy mezi záznamy jsou reprezentovány *relacemi* — shodnými hodnotami dvou sloupců.

Software implementující relační databáze se nazývá Relational Database Management System (RDBMS). RDBMS dále obsahuje implementaci některého z dotazovacích jazyků, např. SQL, a podporu funkcí jako jsou ACID, triggery a uložené procedury.

SQL

Pro dotazování a správu dat uchovaných v RDBMS se využívá standardizovaný jazyk SQL (Structured Query Language)[7]. Díky němu můžeme data analyzovat a získávat z nich informace. Data můžeme shlukovat, transformovat a zkoumat pomocí nejrůznějších statistických funkcí.

Spravované databáze

Provize a správa databází je velice komplikovaná. Je třeba řešit replikaci, a také automatické škálování. Spravované databáze[8] se snaží tyto problémy řešit poskytováním předpřipravených databázových instancí, u kterých je automaticky řešena provize a škálování. Většina spravovaných databází dále poskytuje velice bezpečné nastavení vyladěné pro maximální bezpečnost a výkon databáze. To nám dovoluje se starat pouze o databázové objekty a uživatelské účty, tedy aspekty které přímo ovlivňují naši aplikaci.

Další výhodou spravovaných databází je poskytování různých metrik a varování, která nám ukazují detailnější informace o stavu a provozu databáze. Spravovaná databáze je například schopná detekovat tabulky bez primárního klíče a zaslat upozornění databázovému administrátorovi, či zobrazovat detailní real-time statistiky o propustnosti databáze.

1.1.4 Webové servery

Webový server (webserver)[9] je klíčový prvek infrastruktury webové aplikace. Hlavní rolí webserveru, jak už název napovídá, je poskytování webových stránek. Moderní webservery podporují širokou škálu protokolů jako je Hyper Text Transport Protocol (HTTP), Web Sockets (WS), či Secure Sockets Layer (SSL). Pro standardní internetovou komunikaci využívají webservery síťové porty 80 pro nešifrovanou komunikaci a 443 pro šifrovanou komunikaci.

Většina webserverů poskytuje obsah dvěma způsoby, staticky a dynamicky. Statický obsah je poskytován ze souborů uložených na webserveru samotném. Dynamický obsah je na druhou stranu generován samostatným

aplikačním serverem, který poté využívá webserver jako komunikační bránu. Některé webservery mají schopnost přímo provádět aplikační kód a separátní server není potřeba. Pro vytváření dynamických stránek můžeme například využít PHP, Perl, či různé CGI skripty. Statický obsah je ideální pro poskytování obsahu jako jsou obrázky, či klientské skripty. Dynamický obsah se na druhou stranu využívá pro obsluhu formulářů, či generování sestav.

1.2 Aplikační architektura

Tato sekce popisuje technologie využité při vývoji aplikace *BurzaŠkol.Online*.

1.2.1 HTTP

HTTP[10] je jeden z klíčových internetových protokolů. Mezi jeho hlavní přednosti patří jeho textová podoba a bezstavovost. Po síti tedy cestují pouze jednoduše stravitelné textové řetězce, převážně čitelné i pro člověka. Díky tomu je oproti ostatním přenosovým protokolům jednoduše použitelný a implementovatelný. Bezstavovost znamená, že jednotlivé HTTP požadavky na sebe nijak nenavazují a jsou plně nezávislé.

1.2.2 PHP

PHP[11] je skriptovací jazyk vytvořený v roce 1994 pro snadný vývoj dynamických a interaktivních webových aplikací. Hlavními přednostmi jazyka PHP je dynamický typový systém, extenzívní standardní knihovna a hluboká integrace s klíčovými webovými technologiemi. Funkcionalita jazyka PHP je dále rozšiřována velkým množstvím knihoven a frameworků. Pro jednoduchou správu těchto knihoven je využíván správce balíčků Composer[12].

1.2.3 PHP Frameworky

Vytváření rozsáhlé a propracované webové aplikace je těžký a dlouho trvající úkol. Z tohoto důvodu se využívají tzv. *frameworky*. Tyto frameworky obsa-

hují široké množství nástrojů pro usnadnění vývoje webových aplikací. Mezi nejčastěji poskytované nástroje patří ORM[13], směrovač[14], či šablonovací knihovna[15]. Dále může obsahovat nástroje na správu uživatelských sezení a různé další utility. Časté jsou i různé terminálové programy určené k ulehčení práce s daným frameworkem, např. pro generování kontrolerů, databázových migrací a správu mezipaměti.

Využitím frameworku můžeme významně zkrátit čas vývoje aplikace, jelikož autoři frameworku za nás učinil většinu architekturálních rozhodnutí a implementovali obslužný kód. Nám tedy zbývá pouze byznysová logika naší aplikace a nemusíme řešit věci jako je ukládání často využívaných dat do mezipaměti, či ověřování uživatelů.

Na druhou stranu využití frameworku sebou nese i jisté nebezpečí. Frameworky se mezi sebou liší, tudíž zkušenosti s jedním frameworkem nemusí být přenositelné na druhý. Často je tomu naopak - při používání frameworku nevhodným způsobem mohou vzniknout velké problémy, především po stránce výkonu aplikace. Dalším velkým nebezpečím mohou být bugy samotného frameworku, jejichž opravou, či obcházením ztrácíme čas určený pro vývoj aplikace samotné.

Laravel

Laravel[16] je PHP framework pro snadné tvoření komplexních webových aplikací. Hlavními výhodami frameworku Laravel je propracovaná, detailní dokumentace a obrovský ekosystém podpůrných knihoven, projektů a služeb, jako např. Laravel Vapor[17] nebo Laravel Nova[18].

1.2.4 Webové sokety

Většina webových aplikací vyžaduje dříve či později nějakou real-time funkcionalitu. Je mnoho způsobů, jak tuto funkcionalitu implementovat. Můžeme využít HTTP Polling[19] nebo Server-sent events (SSE)[20]. Velká slabina obou těchto přístupů je, že i když máme stále aktuální data ze serveru, tak neexistuje způsob, jak tyto metody využít pro odesílání požadavků na server.

Je proto nutné udělat separátní HTTP požadavek, což je velice neefektivní.

Protokol Web Sockets (WS)[21] se tento problém snaží řešit poskytováním plně duplexní komunikace mezi klientským programem a serverem. Jelikož se jedná o protokol cílený na použití ve webových prohlížečích, staví tento protokol nad protokolem HTTP, který používá pro navázání komunikace — handshaking. Využívá i stejné porty jako protokol HTTP, port 80 pro nešifrovanou a port 443 pro šifrovanou komunikaci.

Pusher

Pusher[22] je software stavějící nad WS protokolem poskytující pub-sub komunikaci v reálném čase mezi serverem a klientskými zařízeními. Je vhodný pro aplikace, které nemohou tyto funkcionality poskytovat samy o sobě, např. stránky napsané v PHP. Pro tyto aplikace poskytuje Pusher jednoduché HTTP rozhraní, skrze které je možné zprávy zasílat a docílit tím komunikace v reálném čase.

1.2.5 Datový model

Datový model[23] určuje způsob průtok dat, interakcí mezi daty a uložení dat aplikace. Od datového modelu se odvíjí většina aspektů aplikace, např. způsob fungování nebo rozložení formulářů a přehledů.

Návrh datového modelu je jedna z nejdůležitějších částí vývoje webové aplikace. Pokud pří návrhu uděláme chybu můžeme si vývoj aplikace velice ztížit. Na druhou stranu dobře připravený datový model velice zjednodušuje vývoj aplikace, jelikož může sloužit jako referenční manuál pro její rozložení.

1.3 Vývojové prostředí

Pro vývoj softwaru je zapotřebí specializovaných vývojových prostředí. Vývojová prostředí obsahují interpreter, či kompilátor (dle typu jazyka)[24], textový editor a další pomocný software, např. linter[25] nebo

jazykový server[26]. Hlavním účelem vývojových prostředí je umožnit vývoj softwaru a co nejvíce ho zjednodušit.

Jelikož je příprava vývojových prostředí náročný a zdlouhavý proces, najdeme řadu postupů, jak jej zjednodušit. Existují různé programy, které vývojové prostředí vytvoří a nastaví a snaží se při tom brát v potaz nejběžnější nastavení, která by běžný uživatel mohl potřebovat. Jedná se o programy jako *create-react-app*[27] a *Artisan Console*[28]. Tyto programy nabízejí funkcionality jako zakládání nového projektu, generování šablon pro různé kusy kódu a spouštění projektu.

Hlavní slabinou těchto programů je, že i když pomáhají s přípravou projektu, často nepomáhají se samotným vývojem. Tento problém se snaží řešit tzv. Integrovaná vývojová prostředí (IDE)[29].

1.3.1 Integrovaná vývojová prostředí (IDE)

Hlavním cílem IDE je pokrýt kompletní vývojový cyklus aplikace. Od založení projektu, přes psaní, až po její vydání. IDE proto obsahuje široké množství nástrojů. Na rozdíl od pomocných programů se ale jedná o ucelený balíček. Dále často obsahují editor kódu s inteligentním napovídáním a zvýrazňováním, jedna z hlavních výhod použití IDE. Samozřejmostí jsou také pokročilé možnosti ladění kódu s GUI debuggery nebo navigování pomocí symbolů.

Pokud by nám základní funkcionalita IDE nestačila, můžeme využít dalších zásuvných modulů. Ty jsou poskytovány buďto autorem samotného IDE nebo třetí stranou. Zásuvné moduly poskytují funkcionality jako podporu dalších jazyků, či formátů souborů, rozšíření možností editování textu nebo integraci s dalšími službami.

Kapitola 2

Realizace projektu

2.1 Návrh datového modelu

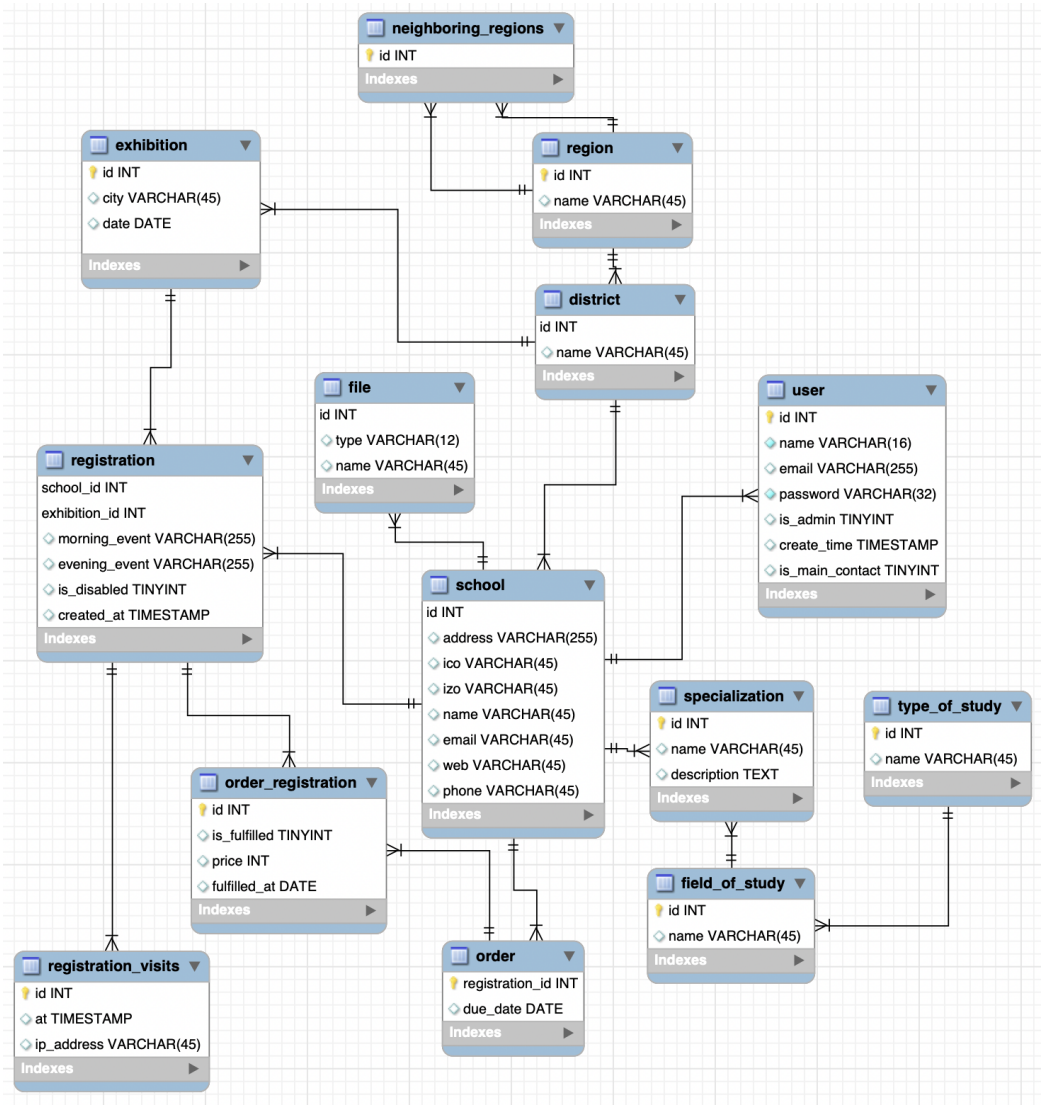
První krok[23] při návrhu datového modelu je vypracování zadání a přesné vymezení datových entit, zatím určujeme pouze název a účel entity. Následně musíme definovat relace mezi daty, tedy popsat, jakým způsobem mezi sebou data interagují.

Druhým krokem je vytyčení atributů datových entit. U atributu je třeba určit název, typ dat, která v něm budou uložena, např. datum. Je také nutné určit zda bude atribut držet unikátní hodnoty, např. emailové adresy a zda je atribut povinný nebo zda může nést prázdnou hodnotu, tzv. *null*[30].

Dalším krokem je adaptace tohoto obecného datového modelu, do modelu specifického pro prostředí, v kterém budou data uložena, např. relační databáze. V tomto kroku dojde k materializaci relací mezi entitami na cizí klíče a intermediate tabulky[31]. Podle atributů vytvořených v druhém kroku vytvoříme příslušné sloupce s odpovídajícími datovými typy a omezeními.

Posledním krokem, který nemusí nutně probíhat v době návrhu, ale může být proveden dodatečně, je určení často využívaných dat a vytvoření tzv. *indexů*[32]. Ty slouží pro optimalizaci databázových dotazů a urychlení vyhledávání dat. *Přílišné indexování může vést k nárůstu velikosti databáze a zpomalení dotazů!*[33]

Datový model *BurzaŠkol.Online* je postaven kolem konceptu vystavovatele, z implementačních důvodů reprezentováno tabulkou *schools*.



Obrázek 2.1: Datový model aplikace *BurzaŠkol.Online* k říjnu roku 2020

Vystavovatelé jsou hlavní cílová skupina portálu. Mají možnost přihlásit se na výstavy a vložit krátký článek o sobě, případně o svých oborech, s podporou formátování za pomoci HTML značek. Portál umožňuje vystavovatelům nahrávat loga, informační brožury a obrázky, aby tak mohli zájemcům poskytnout maximální množství informací. V současné podobě projektu *BurzaŠkol.Online* existují tři typy vystavovatelů: školy, firmy a Úřady práce České Republiky.

Školy jsou hlavním typem vystavovatele. Mohou si vytvářet obory, které dále přiřazují k oborům z číselníku MŠMT. To nám dovoluje poskytovat filtrování škol podle typu studia, zaměření a oborů samotných. Školy jsou také automaticky párovány s výsledky maturitních zkoušek, výsledku soutěží zařazených do programu Excelence MŠMT a k inspekčním zprávám. Hlavní přínos portálu *BurzaŠkol.Online* pro školy je navázání kontaktu s žáky 9. tříd a propagace.

Druhým významným typem vystavovatele jsou firmy. Ty mohou vyjádřit svou podporu školám skrze funkcionalitu spolupráce. Kromě toho mají možnost seznámit zájemce se svou nabídkou pracovních pozic a stipendijních programů. Úřady práce mohou poskytovat nerozhodným žákům poradenské služby a dopomoci jim k vybrání vysněného studijního oboru.

2.2 Vývoj projektu

V této sekci jsou popsány technologie využité při vývoji portálu *BurzaŠkol.Online* a důvody pro jejich výběr, postup vývoje projektu a získané zkušenosti s vývojem i samotným provozem.

2.2.1 Využité technologie

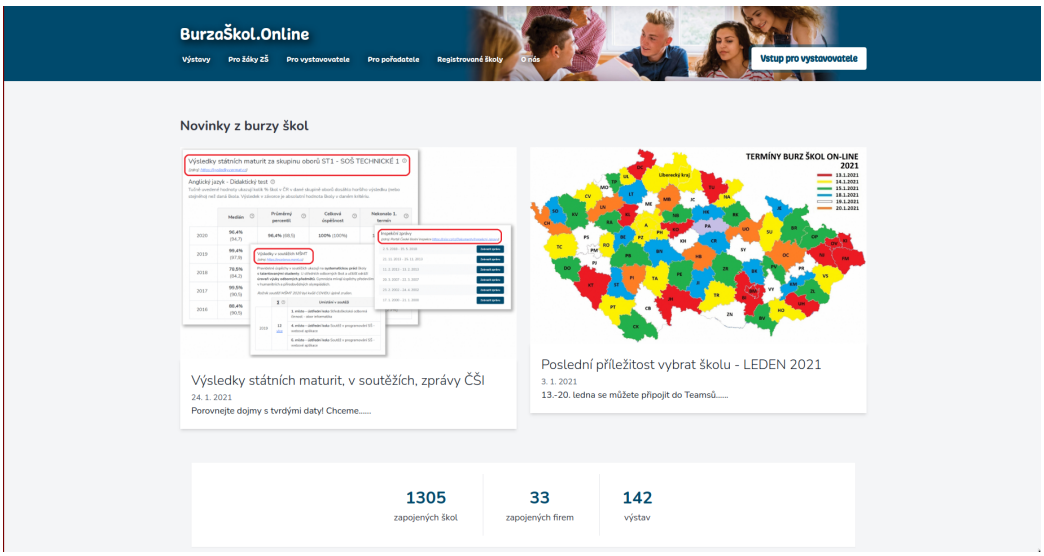
Při vývoji projektu byl využit programovací jazyk PHP s frameworkem Laravel. Pro real-time funkcionality, jako chat a notifikace, byl využit open-source implementace aplikace Pusher, Laravel Websockets[34]. Jako RDBMS jsem využil MySQL[35]. Emailové služby pro aplikaci zajišťuje služba Amazon SES[36].

PHP a Laravel

Jelikož byla na začátku projektu *BurzaŠkol.Online* klíčová rychlost dodání MVP byl vybrán jazyk PHP společně s frameworkem Laravel z důvodu jednoduchosti nasazení, nízkých nákladů na provoz a velké flexibility a rychlosti

vývoje. Dalším velkým plus jsou pomocné utility jako Composer[12] nebo Laravel Artisan Console[28] a rozšíření Laravel Idea[37] pro IDE PhpStorm[38].

Laravel, skrze Laravel Jetstream[39], poskytuje předpřipravené části aplikace jako registrace a ověřování uživatelů, CSS framework Tailwind CSS[40] a JavaScriptový scaffolding Inertia.js[41] nebo Laravel Livewire[42].



Obrázek 2.2: Aplikace *BurzaŠkol.Online*

Pusher

Technologii Pusher byla vybrána z důvodu jednoduché integrace do stávajícího projektu a možnosti pokročilého ladění. Z implementačních důvodů byla využita open-source implementace protokolu Pusher nazvaná „Laravel Websockets“. To nám poskytlo větší kontrolu nad nasazením a škálováním projektu.

MySQL

RDBMS MySQL oproti ostatním relačním databázím vyniká jednoduchostí správy a množstvím poskytovatelů spravovaných databázových instancí[43]. Výše uvedené, a excelentní integrace s jazykem PHP a frameworkem Laravel,

z ní dělá ideální volbu pro vývoj webových aplikací. Velkým přínosem je také široká řada nástrojů jako mysqldump[44] nebo phpMyAdmin[45].

Amazon SES

Amazon Web Services (AWS) poskytuje v rámci služby Simple Email Service (SES) jednoduchou a cenově dostupnou emailovou bránu. Ta nabízí funkcionality jako DKIM podepisování emailů a zobrazování pokročilých statistik jako počet odeslaných emailů, počet „odražených“ emailů[46] a počet stížností na námi zaslanou poštu.

2.2.2 Přípravy na vývoj projektu

Předtím než může započít vývoj SW projektu je zapotřebí se zadavatelem vytvořit zadání. Jedná se o dokument popisující jednotlivé konkrétní funkcionality a procesy aplikace. Ze zadání by měly vycházet veškeré části aplikace jako je datový model a použité technologie. *Zadání by mělo být konkrétní a jednoznačné, pokud existuje více výkladů, může dojít k velkým problémům při vývoji aplikace.*

Dalším krokem je výběr technologií, které budou na vývoj projektu použity. Ze zadání určíme pro jakou platformu bude projektu určen, zda-li se jedná o webovou, desktopovou, mobilní, serverovou nebo kombinovanou aplikaci. Dle toho vybereme konkrétní technologie jako např Laravel[16] pro vývoj aplikace, MySQL[35] pro ukládání dat a Git[47] pro správu zdrojového kódu.

Když jsou zvoleny technologie, můžeme začít vytvářet datový model pro vybraný typ databáze. Dále rozvrhneme jednotlivé obrazovky a postupy, které bude aplikace obsahovat. Je také potřeba vytvořit plán nasazení projektu, tedy jak bude aplikace hostováná, či jaký mail server využijeme.

Když máme všechny přípravy hotovy, můžeme přejít k samotnému programování projektu.

2.2.3 Vývoj projektu ve frameworku Laravel

Prvním krokem při vývoji projektu ve frameworku Laravel, je vytvoření tříd tzv. *modelů*[48] a k nim náležejících *migrací*[49]. Ty vytvoříme dle připraveného datového modelu.

Když jsou modely a migrace úspěšně vytvořeny můžeme přistoupit k vytváření *kontrolerů*[50], *komponent*[51] a definování *cest*[52].

Modely

Modely jsou PHP třídy reprezentující samostatné databázové tabulky, řádky a relace mezi nimi. Modely nám dovolují vytvářet, upravovat a filtrovat záznamy jako by se jednalo o nativní PHP objekty. To velice zjednodušuje práci s daty.

V současné době aplikace *BurzaŠkol.Online* obsahuje 26 modelů, např. *School* pro vystavovatele nebo *Region* pro kraj.

Migrace

Migrace, ve frameworku Laravel, slouží pro kontrolované úpravy databáze. Jejich hlavní úkol je udržet databázi v konzistentním stavu s verzí aplikace. Při nasazení nové verze tedy stačí spustit migrace a databáze je synchronizována s verzí aplikace.

HTTP cesty

HTTP cesty definují jaký kód je spuštěn při HTTP požadavku. Cesta je identifikována pomocí jedné, či více HTTP metod a HTTP cesty. Cesta v sobě může obsahovat parametry, např. id knihy. Cesta může vést na *HTTP přesměrování*, *PHP funkci* nebo *HTTP kontroler*.

Aplikace *BurzaŠkol.Online* definuje 38 unikátních HTTP cest. Velké množství HTTP cest dále přijmá různé parametry, které dále ovlivňují obsah vygenerované stránky.

HTTP kontrolery

HTTP kontrolery, ve frameworku Laravel, jsou PHP třídy obsahující logiku pro zpracování HTTP cest — handlery, PHP metody. Můžeme je rozdělit na standardní kontrolery, *single-action kontrolery*[53] a *resource kontrolery*[54].

Standardní kontrolery obsahují libovolný počet handlerů. Při definování HTTP cesty musíme definovat jak kontroler, tak metodu, která má být zavolána.

Single-action kontrolery obsahují pouze metodu *__invoke*. Jsou vhodné pokud je logika pro nějakou cestu velice komplexní a je lepší ji izolovat. Při definování cesty stačí zmínit pouze název třídy a Laravel metodu automaticky spáruje.

Resource kontrolery mají přesně definovanou strukturu. Obsahují skupinu metod pro vytváření, úpravu a čtení daného zdroje — resource. Cesty se definují skupinově pomocí statické metody *resource*.

Blade komponenty

Všechny aplikace mají za cíl zobrazovat uživateli informace. Webové aplikace využívají k zobrazování informací značkovací jazyk HTML.

U jednoduchý stránek můžeme využít statických HTML souborů, které jsou pro každý požadavek stejné a nemění se. Pro úpravu obsahu stránky je nutný přístup k hostingové službě a znalost HTML.

Moderní webové aplikace vyžadují obsah dynamický, který je závislý na velké řadě faktorů, např. zda-li je uživatel přihlášen, jaká jsou jeho oprávnění, či jaké výstavy se právě konají. Obsah těchto stránek je většinou uživatelsky editovatelný za pomocí grafického WYSIWYG editoru.

Pro generování těchto stránek se používají šablonovací nástroje jako samotné PHP nebo různé pomocné knihovny jako Twig[55] nebo Laravel Blade Views[56]. Takové knihovny nám dovolují psát čisté HTML s přídavkem speciálních příkazů. Tyto příkazy nám dovolují například podmínečné generování nebo možnost vypisovat a formátovat PHP proměnné.

2.3 Nasazení a provoz projektu

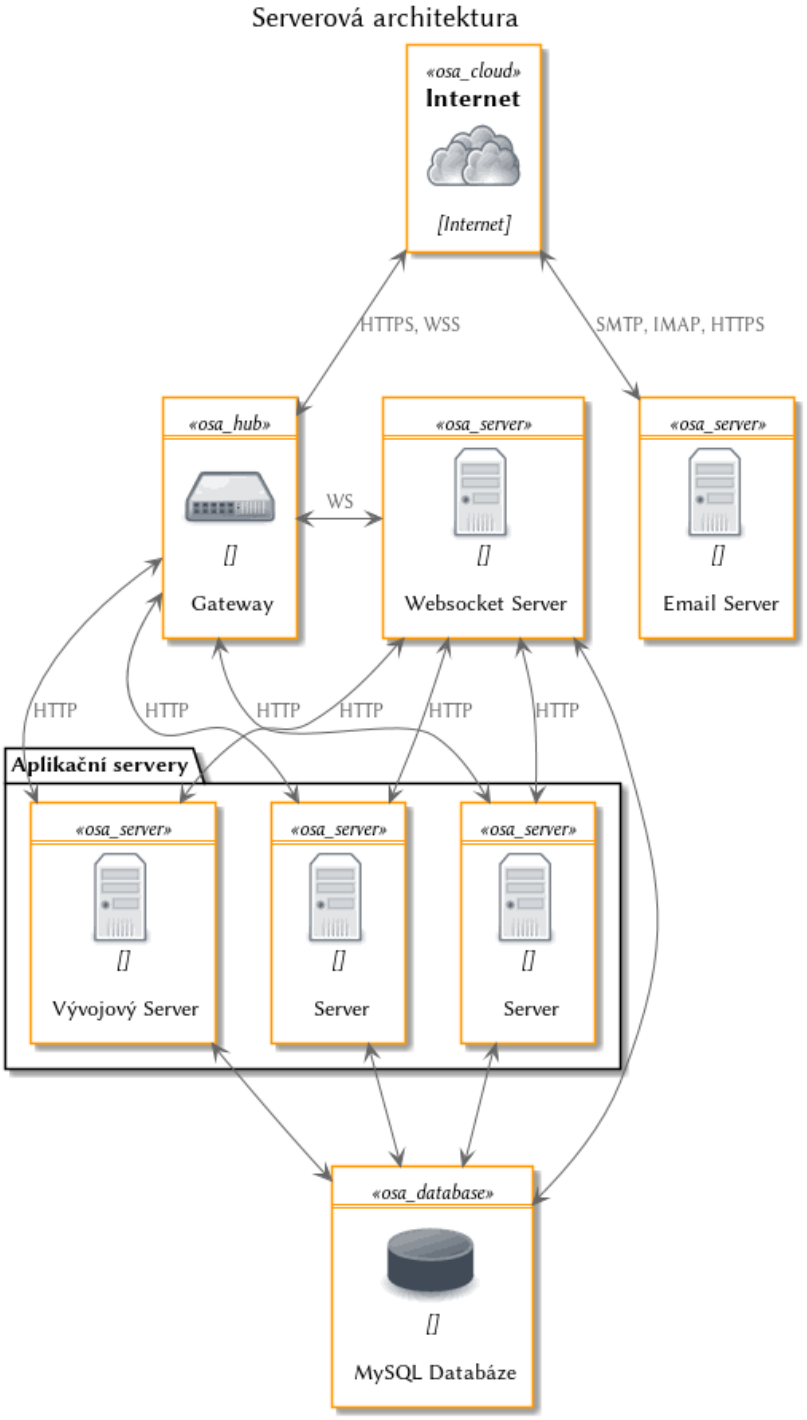
Tato kapitola je věnována nasazení a provozu projektu *BurzaŠkol.Online*. Popisuje volbu hostingové služby a serverovou architekturu. Obsahuje také poznatky a překonané problémy, které v průběhu nasazování projektu vznikli.

2.3.1 Hostingová služba

Při navrhování projektu *BurzaŠkol.Online* byly zvažovány služby Amazon AWS[57], Microsoft Azure[58] a cloudem společnosti Digitalocean[59]. Všechny tři služby poskytují velké množství pokročilých funkcí pro vývoj a provoz webových aplikací.

Nakonec byl pro většinu funkcí zvolen cloud společnosti Digitalocean, jelikož má ze všech tří poskytovatelů nejjednodušší správu a finanční politiku, je přesně uvedeno za co platíte a neexistují žádné skryté limity a kvóty. Mezi další výhody cloudu Digitalocean patří vysoký výkon a velmi příznivé ceny.[60]

Pro další funkcionality jako ukládání souborů a zasílání emailů byly využity služby Amazon S3[61] a Amazon SES[36]. Důvody pro toto rozhodnutí byly bezkonkurenční ceny a integrace s použitými technologiemi, hlavně frameworkem Laravel.



Obrázek 2.3: Serverová architektura projektu *BurzaŠkol.Online*

2.3.2 Serverová architektura

Internet je divokým a nebezpečným místem[62].

Software a Hardware je nespolehlivý, chyba není výjimka ale pravidlo[63].

Podle těchto dvou faktů musíme řídit svůj návrh serverové infrastruktury. Musíme tedy vytvořit serverovou infrastrukturu, která je odolná, bezpečná a škálovatelná.

Z hlediska bezpečnosti můžeme rozdělit servery projektu *BurzaŠkol.Online* na dvě skupiny: *okrajové servery* a *interní servery*.

Okrajové servery mají za úkol komunikaci s Internetem. Jedná se o několik přesně definovaných míst, skrze něž prochází všechna komunikace určená mimo naši infrastrukturu.

Jednou z hlavních výhod tohoto přístupu je možnost využívat uvnitř naší infrastruktury nešifrované protokoly jako HTTP a provádět jejich zašifrování pouze na těchto dedikovaných okrajových serverech. Tyto nešifrované protokoly jsou výkonnější a jednodušší než jejich šifrované varianty[64]. Z důvodu bezpečnosti je nelze použít pro Internetovou komunikaci.

Z pohledu resilience, obrany proti výpadku, jsou nejdůležitějšími opatřeními replikace a load-balancing klíčových částí. Využíváme proto load balancery a spravované databáze, u kterých neznamená výpadek jednoho stroje nedostupnost celé aplikace.

2.3.3 Postup nasazení aplikace

Prvním krokem je provize serveru. Jelikož projekt *BurzaŠkol.Online* obsahuje pouze malý počet serverů, bylo možné všechny servery nakonfigurovat a vytvořit přímo z ovládacího panelu Digitalocean.

Poté co jsou servery vytvořeny přejdeme ke konfiguraci samostatných serverových rolí. Každá serverová role vyžaduje separátní konfiguraci a specifický software, aby mohla svou funkci vykonávat.

Dalším krokem je vytvoření obrazů disků nakonfigurovaných serverů, abych mohl rychle a jednoduše vytvářet další servery těchto rolí. Pro každou serverovou roli byl vytvořen seznam parametrů, které je po vytvoření serveru z obrazu disku nutné upravit. U aplikačních serverů je například nutné upravit IP adresu, na které má webový server naslouchat pro nová připojení.

Posledním krokem je nastavení příslušných DNS záznamů a získání SSL certifikátů.

2.3.4 Nasazování aktualizací aplikace

První přístup nasazování aktualizací, který byl v projektu *BurzaŠkol.Online* využit, byl založený na Github Actions[65]. Github Actions je CI/CD mechanismus založený na spouštění kódu v závislosti na událostech verzovacího systému Git[47]. Využíván byl SSH for Github Actions[66], pro automatické nasazování při commitu do dedikované větve.

Tento přístup má několik problémů. Hlavním problémem je fakt, že pro každý přidaný cílový server je zapotřebí vytvořit novou konfiguraci. Když vysazovací akce selže, je těžké najít příčinu, jelikož výpisy Github Actions jsou nepřehledné a zaměňují STDOUT a STDERR. Posledním velkým problémem tohoto přístupu je, že pro spuštění vysazení je zapotřebí změna v kódu aplikace.

Tyto problémy jsem vyřešil použitím Laravel Envoy[67]. Laravel Envoy využívá konfigurační soubory v jazyce PHP pro provádění dávek akcí na skupině serverů zároveň. Pokud je zapotřebí přidat nový server, stačí přidat jeho IP adresu do seznamu serverů a Laravel Envoy automaticky provede všechny potřebné akce. Další velkou výhodou je absence nutnosti ukládat SSH klíče na serveru třetí strany. Pro zopakování vysazení stačí pouze znovu spustit Laravel Envoy.

2.4 Bezpečnost aplikace

Kapitola zaměřená na zabezpečení projektu *BurzaŠkol.Online* je z důvodu komplexity tématu rozdělena na dvě části. První část pojednává o zabezpečení systému proti útokům vedeným skrze samotnou aplikaci, např. útokům typu *SQL Injection*. Druhá kapitola je zaměřena na zabezpečení infrastruktury proti útokům vedeným z internetu cílících na špatně nakonfigurované služby, např. *Útok na SSH přihlašování serveru*.

2.4.1 Aplikační zabezpečení

Hlavní nebezpečí při provozování webového portálu plyne z práce s uživatelským vstupem. Aplikace musí tento vstup přijímat a operovat nad ním. To může vést k celé řadě útoků. Je tedy důležité, aby uživatel svým vstupem nemohl webovou aplikaci napadnout, zpomalit, ani jinak vyřadit z provozu. Nejdříve se zaměříme na bezpečnost textových polí a útoky s nimi spojenými.

SQL Injection

SQL Injection[68] je typ útoku, při kterém útočník zadá do textového vstupu kód v jazyce SQL. Naše aplikace poté tento kód vezme a přímo ho vloží do řetězce dotazu. Při vykonávání příkazu pak databázový server vykoná tento škodlivý kód, jako by pocházel od samotné aplikace.

Tento útok představuje velké nebezpečí, jelikož může vést od ztráty dat, až k úniku citlivých dat do rukou útočníka.

Aplikace *BurzaŠkol.Online* se proti tomuto typu útoku brání pomocí *předpřipravených dotazů*[69] — rozdělení dotazu na dvě části. První část tvoří příkaz, který popisuje, jak bude daný dotaz proveden. Druhou část tvoří uživatelské vstupy. Tato část obsahuje pouze data a databázový server ji nikdy nespouští. Tím zabraňuje tomu, aby server vykonal uživatelský vstup jako příkaz. *Předpřipravené dotazy*[69] riziko tohoto útoku plně eliminují.

Cross-site skriptování

Cross-site skriptování[70], nebo také XSS, je typ útoku, při kterém je do textového vstupu zadán validní kód spustitelný prohlížečem. Zranitelný webserver poté tento kód vloží přímo do stránky, která je zaslána na klientské zařízení. Klientské zařízení tento škodlivý kód následně vykoná v domnění, že se jedná o kód naší aplikace.

Tento útok představuje pro návštěvníky velké nebezpečí, jelikož může vést ke krádeži uživatelských přihlášení, zobrazování reklam nebo krádeži platebních údajů. Mezi stránkové skriptování[70] může být také využito jako mezikrok pro další, např. phishingové[71], útoky.

Potenciálními místy kde by mohlo u aplikace *BurzaŠkol.Online* k napadení XSS dojít, jsou jednořádková vstupní pole a grafické textové editory. Jednořádková vstupní pole slouží pro jednoduchý uživatelský vstup bez formátování. Zabezpečení je tedy velice jednoduché. Jakýkoliv vstup, který uživatel zadá, je očištěn a veškeré HTML značky jsou odstraněny. U grafických textových editorů však není tato obrana možná, jelikož validním vstupem je i HTML kód. Aplikace tedy daný kód na serveru parsuje a odstraní veškeré závadné HTML značky, jako je například značka *script*, a závadné HTML atributy, jako například *onmove*.

Podvržení požadavků mezi stránkami

Podvržení požadavků mezi stránkami (CSRF) [72] je typ útoku, při kterém cizí stránka obsahuje odkaz, který vede na naši webovou adresu. Většinou se jedná o požadavky typu POST. Tento požadavek má většinou za cíl vykonat škodlivou akci, např. napsat příspěvek bez vědomí uživatele. Při útoku je využíváno faktu, že prohlížeč s každým požadavkem odesílá HTTP cookies, které ověřují uživatele serveru.

Útočník tedy může například vytvořit novou objednávku a tím způsobit finanční újmu. Tento útok představuje pro uživatele velké nebezpečí.

Aplikace *BurzaŠkol.Online* se brání podvržení požadavků tak, že pro každou uživatelskou akci vyžaduje tzv. CSRF Token[72]. Token je unikátní pro

každý požadavek a dovoluje uživateli vykonat přesně jednu akci. Token je vygenerován serverem, přímo do HTML kódu stránky. Útočník k němu proto nemá přístup a není mu dovoleno vykonat žádnou akci.

Útok hrubou silou

Útoky hrubou silou[73] je skupina útoků využívající velkého počtu požadavků za účelem uhodnutí uživatelských přihlašovacích údajů. Tyto požadavky jsou vedeny na přihlašovací formulář aplikace a podle odpovědi serveru určují zda byl útok úspěšný, či nikoliv.

Útok typu brute-force vede k vyzrazení přihlašovacích údajů útočníkovi. Útočník tím získá možnost plné kontroly nad uživatelským účtem.

Proti tomuto útoku je nasazena obrana spočívající v limitaci počtu požadavků na IP adresu. Pokud uživatel udělá za určený časový více požadavků na přihlášení než je povolený limit, aplikace požadavky zahazuje a dále je nezpracovává. Tato ochrana činí útoky typu brute-force velice nepraktické, až nemožné.

2.4.2 Zabezpečení infrastruktury

Dalším důležitým faktorem bezpečnosti, při provozování webového portálu, je zabezpečení serverové infrastruktury. Útoky na serverovou infrastrukturu aplikace mají často větší dopad než zranitelnosti aplikace samotné. Tyto útoky mají často za úkol aplikaci vyřadit z provozu nebo ji zneužít. Jedná se, např. o *slovníkové útoky* nebo *brute-force útoky na SSH přihlašování.*

Útoky na SSH přihlašování

Při útoku na SSH přihlašování serveru se využívá povoleného přihlašování pomocí uživatelských hesel. Útok se snaží najít kombinace uživatelských jmen a hesel, které nejsou dostatečně komplexní a bezpečné. Častým cílem útoku je kořenový uživatel, *root*, který by úspěšnému útočníkovi poskytl kompletní kontrolu nad cílovým strojem. K prolomení hesel se využívají tzv.

slovníkové útoky — soubory velkého množství častých hesel, kterými se útočník snaží zabezpečení prolomit.

Pokud dojde k prolomení SSH přihlášení serveru získá tím útočník plný přístup k serverovému terminálu. Je tedy schopný vykonávat všechny akce, ke kterým má napadený účet oprávnění. Útočník může například zapojit stroj do botnetu[74], zfalšovat nebo zaměnit poskytované stránky, či ukrást platební nebo osobní údaje.

Jednou z hlavních obran, kterou využívají servery projektu *BurzaŠkol.Online*, proti tomuto útoku je zákaz používaní hesel pro vzdálený přístup. K přihlašování se poté využívá tzv. SSH klíčů[75] — certifikátů založených na asymetrické kryptografii. Druhou linií obrany je poté rozřazení jednotlivých úloh pod samostatné uživatelské účty a princip minimálních oprávnění. Toto opatření pak minimalizuje škody způsobené útočníkem na úzkou napadenou oblast.

2.5 Postup projektu

V této sekci je popsáno praktické nasazení projektu *BurzaŠkol.Online*.

2.5.1 Jedno-serverové nasazení

První verze aplikace *BurzaŠkol.Online* využívala pouze jeden server pro hostování všech aplikačních funkcí jako webserver, databázový server a aplikační server. Tento způsob hostování má však mnoho problémů.

Jelikož celá infrastruktura běží na jednom serveru narážíme na velké bezpečnostní riziko vytvářením kritického bodu infrastruktury, jehož nedostupnost nebo porucha znamená výpadek celé aplikace. To také znamená, že není možné aplikaci aktualizovat bez jejího výpadku.

Druhým, i když menším problémem, je omezení možnosti škálování v odpovědi na nárůst požadavků, jelikož pro alokaci větších serverových prostředků je nutné server vypnout. Pro aplikaci *BurzaŠkol.Online* toto nepředstavovalo velký problém, jelikož jazyk PHP je velice efektivní ve využívání

serverových prostředků a provoz aplikace nepřekročil alokované serverové prostředky.

2.5.2 Více-serverové nasazení

Pro vyřešení tohoto problému byla infrastruktura rozšířena na více serverů.

Jako vstupní bod infrastruktury byl vyčleněn server označen *gateway*. Mezi hlavní role tohoto serveru byly určeny load-balancer a vstupní bod protokolů HTTP a WS. To nám dovoluje provádět provizi ssl certifikátů a ssl terminaci pouze v jednom místě naší infrastruktury.

Jelikož *gateway* poskytuje služby load-balanceru mohlo být vytvořeno několik, na sobě nezávislých instancí, aplikace *BurzaŠkol.Online*. Pro provedení tohoto kroku bylo zapotřebí rozčlenit všechny zdroje dat, např. RDBMS nebo ukládání souborů na dedikované servery, jelikož aplikační servery nemohou tyto služby mezi sebou sdílet. Jako RDBMS byla využita spravovaná databáze společnosti Digitalocean s replikací a automatickou správou redundantních serverů. Na ukládání souborů bylo využito objektové úložiště Amazon S3 pro manipulaci se soubory ze všech aplikačních serverů najednou, bez potřeby vlastní infrastruktury.

Rozčlenění aplikačních serverů nám poskytuje větší odolnost proti výpadku samostatných serverů, výpadek serveru neznamená výpadek aplikace, pouze zhoršení dostupnosti. Také nám dovoluje aplikaci škálovat bez potřeby její odstávky.

2.5.3 Problém s pracovními vlákny load-balanceru Nginx

Při prvním veřejné výstavě uspořádané skrze platformu *BurzaŠkol.Online* docházelo při požadavcích na server k výpadkům a překročení časového limitu požadavku. Po kontrole souboru `/var/log/nginx/access.log` bylo zjištěno, že na serveru *gateway* došlo k vyčerpání FD. To vedlo k nemožnosti zpracování požadavků serverem.

Pokud není webový server schopný odpovědět v časovém limitu, zobrazí webový prohlížeč stránku jako nedostupnou, jelikož webové prohlížeče implementují časový limit pro požadavky[76].

Pro opravu této chyby bylo zapotřebí zvýšit počet požadavků zpracovávaných jedním pracovním vláknem a horní limit FD. Tyto změny byly provedeny v souboru `/etc/nginx/nginx.conf`. Zvýšení počtu požadavků, které je schopné jedno pracovní vlákno zpracovat bylo provedeno použitím direktivy:

```
events {
      worker_connections <počet připojení>;
}
```

Pro zvýšení horního limitu FD byla použita direktiva `worker_rlimit_nofile <počet FD>`.

2.6 Provoz a přínos aplikace

2.6.1 Provoz aplikace

Portál byl uveden do ostrého provozu 6 dní od začátku jeho vývoje, konkrétně 27. září 2020. Ze začátku byl provozován v jedno-serverovou architektuře. Od 2. října 2020 jsme převedli portál na robustnější více-serverovou architekturu, abychom po technické stránce snáze zvládli nápor návštěvníků.

V prostředí portálu proběhlo celkem 142 on-line burz ve dvou hlavních vlnách od 7. do 14.12.2020 — hlavní vlna a potom 13. — 20.1.2021, ve kterých se uskutečnilo přibližně 15 tisíc propojení mezi školou a zájemcem.

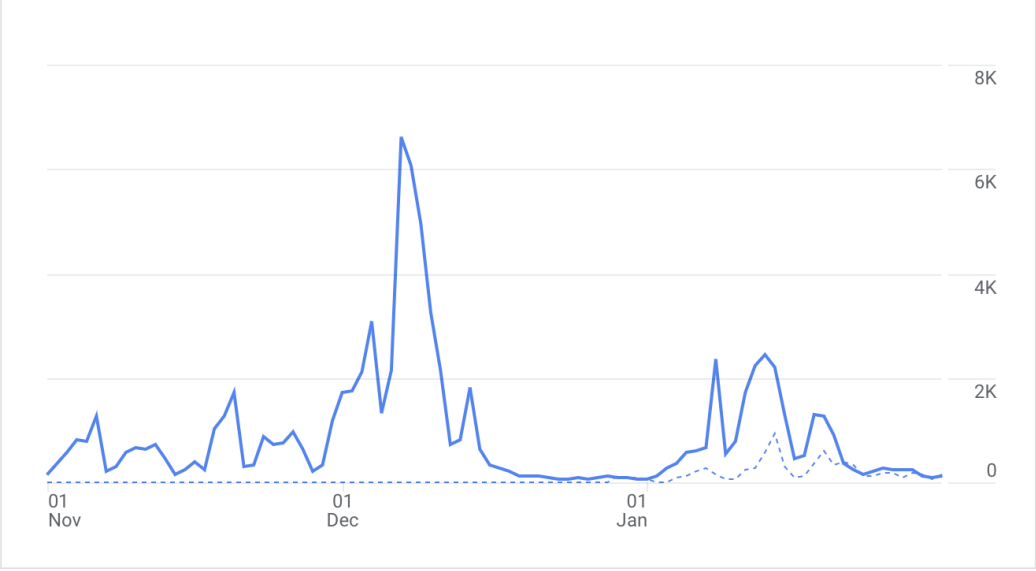
Přes vánoční svátky jsme projekt zásadním způsobem rozšířili. Vytvořili jsme crawlery, které získaly data z veřejně dostupných databází a tato data začali zobrazovat. Konkrétně:

- z rejstříku škol MŠMT jsme dohráli všechny existující střední školy a jejich obory (tak, aby uchazeč hledající na našich burzách viděl i

základní informace o školách, které patří do okresu, ale k burze nejsou zaregistrované)

- z portálu Excelence MŠMT jsme dohráli ke školám všechny účasti a umístění jejich studentů v soutěžích za poslední 4 roky
- z portálu CERMATu jsme dohráli ke všem maturitním oborům škol výsledky u didaktických testů u státních maturit
- z portálu České školní inspekce jsme dohráli všechny inspekční zprávy škol

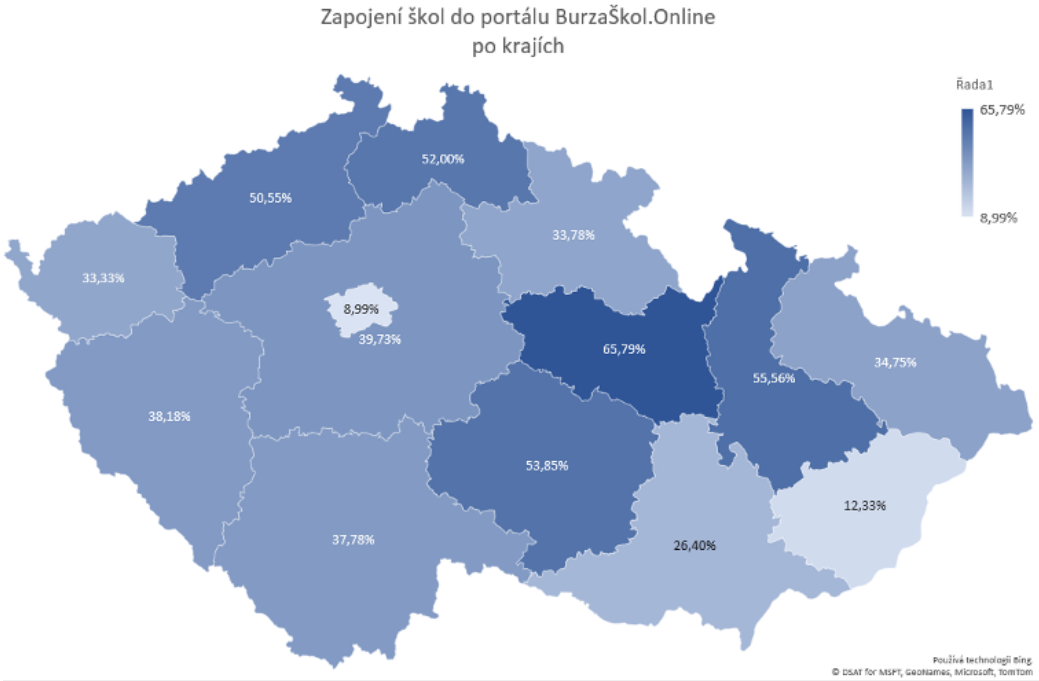
V době největších špiček bylo v jeden okamžik k portálu připojených přes 300 uživatelů, a za necelého půl roku jeho existence se k němu připojilo přes 66 tisíc unikátních uživatelů, kteří si zobrazili přes 686 tisíc stránek. V současné době, kdy skončila pro střední školy „náborová sezóna“, je na serveru provoz již spíše sporadický (řádově nižší stovky zobrazených stránek denně).



Obrázek 2.4: Graf návštěvnosti portálu *BurzaŠkol.Online* od listopadu 2020 do ledna 2021

2.6.2 Zapojení škol

Během prvních 14-ti dní projektu se v Pardubickém kraji, díky podpoře odboru školství, zapojilo více jak 60% středních škol. V ostatních krajích bylo, i přes podporu MŠMT, zapojení škol pouze sporadické. Pokrok ale nastal po zapojení Úřadu práce na celostátní úrovni, kdy jediný dopis informačním a poradenským střediskům přinesl zapojení více než 250 škol z celé České Republiky.



Obrázek 2.5: Zapojení škol v jednotlivých krajích

2.6.3 Přínos aplikace

Přínos aplikace jsme rozdělili do hlavních pěti bodů:

1. Vymysleli jsme nový kanál pro komunikaci mezi uchazeči a středními školami, který v době pandemie alespoň částečně nahradil klasické prezenční výstavy středních škol. Využili jsme možností on-line konferencí, které jsme podpořili internetovým portálem pro vyhledávání škol podle

vhodných kategorií (geografická lokace, zaměření, obory, úroveň vzdělání, ...).

2. Agregovali jsme dosud nevyžité a opomíjené veřejné zdroje dat o školách (výsledky státních maturit, výsledky soutěží pořádaných MŠMT, zprávy České školní inspekce) do jednoho portálu a umožnili jsme porovnávání škol.

3. Dokázali jsme celý portál „oživit“. Aktivně se do něj zapojilo 465 středních škol a učilišť z celé ČR (více než $\frac{1}{3}$). Informovali jsme o něm všechny základní školy v ČR.

4. „Navedli“ jsme střední školy v ČR, aby zorganizovaly své dny otevřených dveří pomocí on-line konferencí.

5. Ve školním roce 2020/21 pomohl náš portál s volbou školy přibližně v 15 tisících případů. Přes portál proběhlo přes 43 tisíc pokusů o online propojení mezi návštěvníky a školou. Podle zkušenosti z naší školy byl poměr mezi těmito pokusy a skutečnými smysluplnými připojeními přibližně 3:1.

Závěr

V rámci práce byl vytvořen webový portál, který umožňuje nahrazení prezenčních Burz škol on-line videokonferencemi v době pandemie koronaviru Covid-19. Webový portál byl vyvinut a nasazen na takových technologiích a s takovou architekturou, abychom mohli pružně reagovat na výrazné kolísání požadavků na výkon odpovídající termínům on-line burz.

Serverová infrastrukturu pro provoz portálu byla navržena a zvolena s ohledem na bezpečnost, stabilitu a předpokládanou návštěvnost.

Veškeré překážky zjištěné při provozu projektu *BurzaŠkol.Online* byly úspěšně odstraněny a všechny stanovené cíle projektu byly úspěšně splněny.

Portál je dostupný na adrese https://burzaskol.online, kód celého projektu je veřejně přístupný a je hostován na adrese https://github.com/GrimirCZ/delta-burza.

V rámci verze portálu 2.0 byl navázán kontakt s portálem AtlasŠkolství.cz z důvodu poskytování lepších služeb a dosažení většího počtu uživatelů.

Seznam obrázků

2.1 Datový model aplikace *BurzaŠkol.Online* k říjnu roku 2020 . . 17
2.2 Aplikace *BurzaŠkol.Online* 19
2.3 Serverová architektura projektu *BurzaŠkol.Online* 24
2.4 Graf návštěvnosti portálu *BurzaŠkol.Online* od listopadu 2020 do ledna 2021 33
2.5 Zapojení škol v jednotlivých krajích 34

Literatura

1. *Microsoft Teams* [online]. [N.d.] [cit. 2021-02-21]. Dostupné z: https://www.microsoft.com/en/microsoft-teams/group-chat-software.
2. *Google Meet* [online]. [N.d.] [cit. 2021-02-21]. Dostupné z: https://meet.google.com/.
3. *Zoom* [online]. [N.d.] [cit. 2021-02-21]. Dostupné z: https://zoom.us/.
4. *What Is Load Balancing?* [Online] [cit. 2021-02-01]. Dostupné z: https://www.nginx.com/resources/glossary/load-balancing/.
5. *What is Cloud Object Storage?* [Online] [cit. 2021-02-01]. Dostupné z: https://aws.amazon.com/what-is-cloud-object-storage/.
6. *Co je relační databáze* [online] [cit. 2021-02-02]. Dostupné z: https://www.oracle.com/cz/database/what-is-a-relational-database/.
7. HELLER, Martin. *What is SQL? The lingua franca of data analysis* [online] [cit. 2021-02-02]. Dostupné z: https://www.infoworld.com/article/3219795/what-is-sql-the-lingua-franca-of-data-analysis.html.
8. DRAKE, Mark. *What is SQL? The lingua franca of data analysis* [online] [cit. 2021-02-03]. Dostupné z: https://www.digitalocean.com/community/tutorials/understanding-managed-databases.
9. *What is a web server?* [Online] [cit. 2021-02-04]. Dostupné z: https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_is_a_web_server.
10. *HTTP* [online] [cit. 2021-02-04]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/HTTP.
11. *PHP* [online]. [N.d.] [cit. 2021-02-04]. Dostupné z: https://www.php.net/.

12. NILS ADERMANN, Jordi Boggiano; CONTRIBUTORS, community. *Composer* [online]. [N.d.]. Ver. 2.0.9 [cit. 2021-02-04]. Dostupné z: https://getcomposer.org/.

13. HOYOS, Mario. *What is an ORM and Why You Should Use it* [online] [cit. 2021-02-04]. Dostupné z: https://blog.bitsrc.io/what-is-an-orm-and-why-you-should-use-it-b2b6f75f5e2a.

14. PAUL, John O. *How to build a basic server side routing system in PHP.* [Online] [cit. 2021-02-04]. Dostupné z: https://medium.com/the-andela-way/how-to-build-a-basic-server-side-routing-system-in-php-e52e613cf241.

15. UNDERWOOD, Paul. *The Ups and Downs of PHP Template Engines* [online] [cit. 2021-02-04]. Dostupné z: https://dzone.com/articles/ups-and-downs-php-template.

16. *Laravel* [online]. [N.d.] [cit. 2021-02-08]. Dostupné z: https://laravel.com/.

17. *Laravel Vapor* [online]. [N.d.] [cit. 2021-02-08]. Dostupné z: https://vapor.laravel.com/.

18. *Laravel Nova* [online]. [N.d.] [cit. 2021-02-08]. Dostupné z: https://nova.laravel.com/.

19. HANSON, Joe. *What is HTTP Long Polling?* [Online] [cit. 2021-02-08]. Dostupné z: https://www.pubnub.com/blog/http-long-polling/.

20. *Using server-sent events* [online] [cit. 2021-02-08]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/API/Server-sent_events/Using_server-sent_events.

21. *The WebSocket API (WebSockets)* [online] [cit. 2021-02-08]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API.

22. *Pusher* [online]. [N.d.] [cit. 2021-02-08]. Dostupné z: https://pusher.com/.

23. *DBMS - Data Models* [online] [cit. 2021-02-09]. Dostupné z: https://www.tutorialspoint.com/dbms/dbms_data_models.htm.

24. *Compiler vs Interpreter* [online] [cit. 2021-02-06]. Dostupné z: https://www.geeksforgeeks.org/compiler-vs-interpreter-2/.

25. *What is a linter and why your team should use it?* [Online] [cit. 2021-02-06]. Dostupné z: https://sourcelevel.io/blog/what-is-a-linter-and-why-your-team-should-use-it.

26. *Language Server Protocol* [online] [cit. 2021-02-06]. Dostupné z: https://microsoft.github.io/language-server-protocol/.

27. EAN PLATTER Jonny Buchanan, Max Stoiber. *Create React App* [online]. [N.d.] [cit. 2021-02-06]. Dostupné z: https://github.com/facebook/create-react-app.

28. *Artisan Console* [online]. [N.d.] [cit. 2021-02-06]. Dostupné z: https://laravel.com/docs/8.x/artisan.

29. *Programming software and the IDE* [online] [cit. 2021-02-06]. Dostupné z: https://www.bbc.co.uk/bitesize/guides/zgmpr82/revision/1.

30. *What is a Database NULL Value?* [Online] [cit. 2021-02-09]. Dostupné z: https://www.essentialsql.com/get-ready-to-learn-sql-server-what-is-a-null-value.

31. BRADLEY, Paul. *KB5772: What is an "intermediate table"?* [Online] [cit. 2021-02-09]. Dostupné z: https://community.microstrategy.com/s/article/KB5772-What-is-an-quot-intermediate-table-quot.

32. *Database Indexes Explained* [online] [cit. 2021-02-09]. Dostupné z: https://www.essentialsql.com/what-is-a-database-index/.

33. OZAR, Brent. *Index Tuning Week: How Many Indexes Are Too Many?* [Online] [cit. 2021-02-09]. Dostupné z: https://www.brentozar.com/archive/2018/10/index-tuning-week-how-many-indexes-are-too-many.

34. MARCEL POCIOT Freek Van der Herten, Community Contributors. *Laravel Websockets* [online]. [N.d.] [cit. 2021-02-11]. Dostupné z: https://github.com/beyondcode/laravel-websockets.

35. *MySQL* [online]. [N.d.] [cit. 2021-02-11]. Dostupné z: https://www.mysql.com/.

36. *Amazon Simple Email Service (SES)* [online]. [N.d.] [cit. 2021-02-12]. Dostupné z: https://aws.amazon.com/ses/.

37. FAIZRAKHMANOV, Adel. *Laravel Idea* [online]. [N.d.] [cit. 2021-02-11]. Dostupné z: https://plugins.jetbrains.com/plugin/13441-laravel-idea.

38. *PhpStorm* [online]. [N.d.] [cit. 2021-02-11]. Dostupné z: https://www.jetbrains.com/phpstorm/.

39. *Laravel Jetstream* [online]. [N.d.] [cit. 2021-02-11]. Dostupné z: https://github.com/laravel/jetstream.

40. *Tailwind CSS* [online]. [N.d.] [cit. 2021-02-11]. Dostupné z: https://tailwindcss.com/.

41. *Inertia.js* [online]. [N.d.] [cit. 2021-02-11]. Dostupné z: https://github.com/inertiajs/inertia.

42. *Laravel Livewire* [online]. [N.d.] [cit. 2021-02-11]. Dostupné z: https://laravel-livewire.com/.

43. JANÍK, David. *Velké srovnání MySQL (MariaDB) a PostgreSQL* [online] [cit. 2021-02-12]. Dostupné z: https://www.vas-hosting.cz/blog-velke-srovnani-mysql-mariadb-a-postgresql.

44. *mysqldump* [online]. [N.d.] [cit. 2021-02-12]. Dostupné z: https://dev.mysql.com/doc/refman/8.0/en/mysqldump.html.

45. *phpMyAdmin* [online]. [N.d.] [cit. 2021-02-12]. Dostupné z: https://www.phpmyadmin.net/.

46. *Soft vs. Hard Bounces* [online] [cit. 2021-02-12]. Dostupné z: https://mailchimp.com/help/soft-vs-hard-bounces/.

47. LINUS TORVALDS, Junio C Hamano. *Git* [online]. [N.d.] [cit. 2021-02-13]. Dostupné z: https://git-scm.com/.

48. *Eloquent: Getting Started* [online] [cit. 2021-02-14]. Dostupné z: https://laravel.com/docs/8.x/eloquent.

49. *Database: Migrations* [online] [cit. 2021-02-14]. Dostupné z: https://laravel.com/docs/8.x/migrations.

50. *Controllers* [online] [cit. 2021-02-14]. Dostupné z: https://laravel.com/docs/8.x/migrations.

51. *Blade Templates* [online] [cit. 2021-02-14]. Dostupné z: https://laravel.com/docs/8.x/blade#components.

52. *Routing* [online] [cit. 2021-02-14]. Dostupné z: https://laravel.com/docs/8.x/routing.

53. *Single Action Controllers* [online] [cit. 2021-02-15]. Dostupné z: https://laravel.com/docs/8.x/controllers#single-action-controller.

54. *Resource Controllers* [online] [cit. 2021-02-15]. Dostupné z: https://laravel.com/docs/8.x/controllers#resource-controllers.

55. *Twig* [online]. [N.d.] [cit. 2021-02-16]. Dostupné z: https://twig.symfony.com/.

56. *Blade Templates* [online] [cit. 2021-02-16]. Dostupné z: https://laravel.com/docs/8.x/blade.

57. *Amazon Web Services* [online] [cit. 2021-02-17]. Dostupné z: https://aws.amazon.com/.

58. *Microsoft Azure: Cloud Computing Services* [online] [cit. 2021-02-17]. Dostupné z: https://azure.microsoft.com/en-us/.

59. *DigitalOcean – The developer cloud* [online] [cit. 2021-02-17]. Dostupné z: https://www.digitalocean.com/.

60. POLOCK, Greg. *DigitalOcean vs AWS: Which Cloud Server is Better?* [Online] [cit. 2021-02-17]. Dostupné z: https://www.upguard.com/blog/digitalocean-vs-aws.

61. *Simple Storage Service (Amazon S3) - Amazon AWS* [online] [cit. 2021-02-17]. Dostupné z: https://aws.amazon.com/s3/.

62. SHATILIN, Ilja. *The dangers of public IPs* [online] [cit. 2021-02-18]. Dostupné z: https://www.kaspersky.com/blog/public-ip-dangers/24745/.

63. CAMERON, Lori. *How Netflix Adopted the New Mindset of Software Failure as the Rule—Not the Exception—and Survived the Great 2015 Amazon Web Services Outage* [online] [cit. 2021-02-18]. Dostupné z: https://www.computer.org/publications/tech-news/research/realizing-software-reliability-in-the-face-of-infrastructure-instability.

64. *HTTP Vs HTTPS: An In-Depth Comparison Of Features And Performance* [online] [cit. 2021-02-18]. Dostupné z: https://www.softwaretestinghelp.com/http-vs-https/.

65. *GitHub Actions* [online]. [N.d.] [cit. 2021-02-20]. Dostupné z: https://github.com/features/actions.

66. WU, Bo-Yi. *SSH for GitHub Actions* [online]. [N.d.] [cit. 2021-02-20]. Dostupné z: https://github.com/appleboy/ssh-action.

67. *Laravel Envoy* [online]. [N.d.] [cit. 2021-02-20]. Dostupné z: https://laravel.com/docs/8.x/envoy.

68. *SQL Injection* [online] [cit. 2020-12-11]. Dostupné z: https://owasp.org/www-community/attacks/SQL_Injection.

69. CORPORATION, Oracle. *MySQL Předpřipravené dotazy* [online] [cit. 2020-12-11]. Dostupné z: https://dev.mysql.com/doc/refman/8.0/en/sql-prepared-statements.html.

70. *Cross-site scripting* [online] [cit. 2020-12-11]. Dostupné z: https://owasp.org/www-community/attacks/xss/.

71. BADGER, ERIC NII SOWAH. *Phishing in depth* [online] [cit. 2020-12-11]. Dostupné z: https://owasp.org/www-chapter-ghana/assets/slides/OWASP_Presentation_FINAL.pdf.

72. PEJŠA, Jan. *Co je Cross-Site Request Forgery a jak se mu bránit* [online] [cit. 2020-12-28]. Dostupné z: https://www.zdrojak.cz/clanky/co-je-cross-site-request-forgery-a-jak-se-branit/.

73. *Brute Force Attack: Definition and Examples* [online] [cit. 2020-12-28]. Dostupné z: https://www.kaspersky.com/resource-center/definitions/brute-force-attack.

74. *What is a DDoS Botnet?* [Online] [cit. 2021-01-27]. Dostupné z: https://www.cloudflare.com/learning/ddos/what-is-a-ddos-botnet/.

75. LARSNOODEN. *SSH/OpenSSH/Keys* [online] [cit. 2021-01-27]. Dostupné z: https://help.ubuntu.com/community/SSH/OpenSSH/Keys.

76. *Chrome, standard timeout?* [Online] [cit. 2021-03-03]. Dostupné z: https://superuser.com/questions/1012176/chrome-standard-timeout.

Slovník

ACID Atomicity, consistency, isolation, durability. 10

AWS Amazon Web Services. 20

Bug Chyby programu zapříčiněné chybou, či nepozorností programátora. 13

CGI Common Gateway Interface. Rozhraní umožňující webovým serverům spouštět terminálové aplikace. 12

CI/CD Continuous integration and continuous delivery. Mechanismus pro automatické testování a nasazování softwaru. 26

Crawler Program určený k stahování dat z webových stránek. 32

CSRF Cross-site request forgery. 28

CSS Cascading Stylesheets. 19

Debugger Programy určené pro ladění programů. 15

DKIM DomainKeys Identified Mail. Systém certifikátů pro ověření autenticity odesílatele emailu. 20

DNS Domain Name System. 26

FD File Descriptor. 31, 32

framework Předpřipravený základ pro tvorbu aplikací. 12, 13, 18, 19, 21–23

GUI Grafické uživatelské rozhraní. 15

HTML Hyper Text Markup Language. 17, 22, 28, 29

HTTP Hyper Text Transport Protocol. 11, 12, 14, 21, 22, 25, 28, 31

HTTP Polling Periodické stahování dat za pomoci separátních HTTP požadavků.. 13

HW Hardware. 9, 25

IDE Integrovaná vývojová prostředí. 15, 19

IP Internet Protocol. 26, 29

IP Hash Algoritmus využívající klientskou IP adresu pro vybrání backendového serveru. 9

Least Connections Algoritmus vybírající server podle nejnižšího počtu právě zpracovávaných požadavků. 9

MVP Minimum Viable Product. Nejmenší možná verze produktu, která je schopná plnit zadaný účel. 18

MŠMT Ministerstvo školství, mládeže a tělovýchovy České republiky. 18, 32–35

on-demand Ve chvili potřeby. Typ škálování, při kterém se daný zdroj automaticky přizpůsobuje objemu uživatelských požadavků. 9

open-source Software s otevřeným zdrojovým kódem. 18, 19

ORM Object-Relational Mapper. Software sloužící pro konverzi dat mezi relačním a objektovým modelem. 13

Perl Rodina dvou vyšších dynamických programovacích jazyků Perl 5 a Perl 6. 12

PHP Rekurzivní název programovacího jazyka PHP: Hypertext Preprocessor. 12–14, 19, 21, 22, 26, 30

pub-sub Publish-subscribe. Způsob předávání zpráv. Zprávy jsou organizovány do kanálů. Klienti se do těchto kanálů mohou zapojit a dostávat příslušné zprávy. 14

RDBMS Relational Database Management System. 10, 11, 18, 19, 31, 47

real-time V reálném čase. Zobrazování, či provádění akcí, v současnou chvíli nebo s minimální odezvou. 11, 13, 18

Round Robin Algoritmus sekvenčně rotující backendové servery ze skupiny dostupných serveru, tj. jeden po druhém. 9

scaffolding Předpřipravená souborová struktura pro vývoj projektu. 19

SES Simple Email Service. 20

směrovač Software rozhodující o tom, jaký kód bude spuštěn v odpovědi na uživatelský požadavek. 13

SQL Structured Query Language. 10, 11, 27, 47

SSE Server-sent events. 13

SSH Secure Shell. 26, 27, 29, 30

SSL Secure Sockets Layer. 11, 26

STDERR Standard Error. 26

STDOUT Standard Output. 26

SW Software. 9, 20, 25

Trigger SQL kód spuštěný v návaznosti ná různé události. 10

Uložená procedura Pojmenovaný SQL kód uložený v RDBMS, který můžeme opakovaně spouštět. 10

webserver Webový server. 11, 12, 30

WS Web Sockets. 11, 14, 31

WYSIWYG What You See Is What You Get. 22

XSS Cross-site scripting. 28

šablonovací knihovna Knihovna ulehčující generování HTML. 13

Příloha 1: Externí oponentský posudek

Posudek práce

Autor práce: Vít Falta

Téma: Webový portál burzaskol.online

Škola: DELTA – Střední škola informatiky a ekonomie, s.r.o.
Ke Kamenci 151, 530 03 Pardubice
Tel.: 466 611 106, www.delta-skola.cz

Posuzovatel: Ing. Radek Burget, Ph.D., FIT, VUT v Brně

Předmět práce

Cílem práce byl návrh, implementace a nasazení webového portálu *burzaskol.online* pro podporu realizace klasických burz škol v České Republice v online prostředí. Portál shromažďuje informace o středních školách a nabízených studijních oborech, umožňuje zveřejnění dalších informačních materiálů a zejména sdílení termínů a webových odkazů k online konferencím, které již následně probíhají pomocí nástrojů třetích stran (např. Microsoft Teams, Google Meet a další). Portál umožňuje i zapojení firem, které mohou podporovat vybrané školy, zveřejňovat pracovní pozice, stipendijní programy a další informace. Cílovou skupinou uživatelů jsou zejména žáci devátých tříd základních škol.

Realizační výstup

Hlavním realizačním výstupem práce je vlastní softwarové řešení portálu vytvořené pomocí serverových a klientských webových technologií. Za další hodnotný výsledek je třeba považovat architekturu nasazení řešení na serverovou infrastrukturu s cílem dosažení dostatečné spolehlivosti a bezpečnosti v reálném provozu.

Vlastní implementace softwarového řešení je řešena poměrně konzervativním způsobem jako monolitická serverová aplikace v jazyce PHP avšak s využitím moderních technologií v podobě aplikačního rámce Laravel, web sockets a dalších technologií. Pro daný účel se jedná o dobrou volbu umožňující efektivní a rychlou implementaci i snadnější správu a nasazení aplikace.

Řešení infrastruktury následně zohledňuje rozložení zátěže na více aplikačních serverů (*load balancing*) a využití spravované (*managed*) databáze a pro zajištění vysoké dostupnosti systému. Při návrhu autor rovněž bral v úvahu bezpečnostní aspekty. Některé detaily zvoleného řešení nejsou z dodané dokumentace zcela zřejmé, nicméně i tak hodnotím navržené řešení jako technicky vyspělé a vhodné pro spolehlivý provoz aplikace v reálných podmínkách.

Dokumentace

Předložená dokumentace je členěna na teoretickou část, která definuje základní pojmy a nejdůležitější současné technologie v oblasti návrhu a implementace webových aplikací, a praktickou část, která se zabývá návrhem, implementací a nasazením vlastního řešení. Obě tyto části jsou převážně dobře strukturované s drobnou výhradou: Teoretická část by zasloužila přehledný úvod, který by objasnil základní přístup k řešení a zdůvodnil, proč jsou dále popisovány právě zvolené technologie. Bez tohoto vysvětlení působí úvodní kapitoly věnované load balancerům a objektovým úložištím poněkud překvapivě a výběr ostatních témat vcelku náhodně. Na druhou stranu část věnovaná vlastní realizaci je velmi srozumitelná a zachycuje vývoj projektu od počátečního návrhu, přes volbu technologií, implementaci a nasazení až po rozbor bezpečnostních aspektů řešení a zhodnocení výsledků dosažených v reálném provozu. Zejména poslední dvě jmenované části hodnotím velmi kladně: Jsou zpracovány detailně a dokazují, že se autor tématu věnoval do hloubky a nespokojil se s pouhou implementací prototypu, což je v praxi jinak bohužel poměrně běžné.

Z faktického hlediska lze v textu ojediněle narazit na drobné nepřesnosti, např. v popisu relačních databází. Některé části řešení jsou bohužel popsány velmi stručně, nejasná zůstává např. role a účel použití *web sockets*, využití objektového úložiště nebo účel a způsob integrace e-mailového serveru, který podle obr. 2.3 není nijak propojen se zbývajícími částmi infrastruktury. Také řešení rozložení zátěže je jen naznačeno a o serveru *nginx* se dozvídáme jen v okrajové zmínce.

Po formální stránce dokumentace působí, jako by byla dokončována poněkud ve spěchu: objevují se zde ojedinělé překlepy, chyby v odkazech (např. název zdroje [8], apod.) Po jazykové stránce je však práce na velmi dobré úrovni a uvedené nedokonalosti nemají vliv na pochopení významu jednotlivých částí.

Závěr

Pan Falta v rámci své práce prostudoval a prakticky využil řadu teoretických postupů a technologií a vytvořil poměrně rozsáhlou aplikaci, kterou posléze pečlivě dovedl až k reálnému nasazení, kde se osvědčila. Přes výše uvedené drobné výhrady k některým částem předložené dokumentace hodnotím proto jeho práci celkově jako velmi nadstandardní.

Práci doporučuji hodnotit stupněm **výborně**.

V Brně dne 30. 3. 2021

Ing. Radek Burget, Ph.D.