Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Mikroframework na bázi komponent Symfony pro publikování vědeckých projektů

Plzeň 2019 Jan Čarnogurský

Místo této strany bude zadání práce.

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 2. května 2019

Jan Čarnogurský

Poděkování

Rád bych touto cestou poděkoval Ing. Pavlu Heroutovi, Ph.D. za odborné vedení mé práce, podnětné připomínky a korekci textu.

Abstract

This bachelor thesis deals with the implementation of microframework, which is based on Symfony components for publishing scientific projects. The aim of this work is to create a web application based on microframework. The theoretical part deals with possible ways of creating websites with comparison of these solutions and various other approaches concerning web technologies. Subsequently, the proposal of the web application concept, which contains a detailed design and implementation. Attention is also dedicated to application testing

Abstrakt

Tato bakalářská práce se zabývá realizací mikroframeworku na bázi Symfony komponent pro publikování vědeckých projektů. Cílem práce je vytvořit webovou aplikaci, která bude vycházet z mikroframeworku. Teoretická část se zabývá možnými způsoby tvorby webových stránek se srovnáním těchto řešení a různými dalšími přístupy týkající se webových technologií. Následně navazuje návrh koncepce webové aplikace, který obsahuje detailní návrh, a jeho implementaci. Pozornost je věnována i samotnému testování aplikace.

Obsah

1 Úvod

Ve školním roce 2017/18 byla zadána diplomová práce s názvem „Aplikace s možností injekce chyb pro ověřování kvality testů“ [8], která byla úspěšně obhájena. Cílem diplomové práce bylo vytvořit přiměřeně složitou softwarovou aplikaci se (semi)realistickým použitím, která bude sloužit jako SUT (System Under Test) pro ověřování správnosti a kvality nově vyvíjených testovacích metod. Práce bude součástí rozsáhlejšího projektu nazvaného TbUIS (Testbed University Information System) a proto bude nezbytné, aby výsledky projektu byly veřejně dostupné. Vznikl tedy požadavek na vytvoření webových stránek projektu. Protože je však celý projekt zasazen do určitého existujícího kontextu webových stránek katedry a výzkumné skupiny, je nutné se tomuto kontextu přizpůsobit. Další omezující požadavky vyplývají z bezpečnostních hledisek, které jsou na webové stránky kladeny, protože budou nasazeny na webový server katedry KIV.

Další skutečností, kterou je nutné vzít u úvahu, je to, že bude prezentován výsledek vědeckého projektu, což není úplně typická webová aplikace. Proto bude třeba při návrhu účel vytvářené webové aplikace tomuto přizpůsobit.

V současné době existuje celá řada používaných technologií pro tvorbu webových stránek a je nezbytné zvolit takovou technologii, která bude v souladu s výše uvedenými skutečnostmi. Samozřejmě tato technologie musí umožňovat další rozvoj webových stránek.

Cílem této bakalářské práce je zjistit existující kontext webových stránek používaných na KIV, dále se seznámit se způsoby publikace vědeckých projektů, provést rešerši používaných technologií pro tvorbu webových stránek, a na základě zjištěných faktů vytvořit webovou aplikaci, jejímž účelem bude zveřejnění projektu TbUIS.

2 Technologie pro tvorbu webových stránek

2.1 HTML

HTML (Hyper Text Markup Language) je standardní značkovácí jazyk pro tvorbu webových stránek, který vychází ze značkovacího jazyka SGML. Jazyk HTML vytvořil Tim Berners-Lee roku 1991 jako součást WWW (World Wide Web), který měl umožnit vědcům sdílet jejich výsledky výzkumů s ostatními vědci po celém světě. Jazyk popisuje strukturu webu pomocí značek neboli tagů, kde každý má svůj význam.

HTML se aktuálně nachází ve verzi HTML5, která přináší množství nových vlastností. Mezi novinky patří například podpora videa, audia a dalších ovládacích modulů. Mezi jednu z nejzajímavějších novinek v HTML5 patří takzvaný canvas, který slouží jako kontejner pro dynamické vykreslování grafikých prvků. Mezi novými tagy najdeme naříklad tag pro definování patičky a hlavičky stránky nebo určité sekce [9].

2.2 CSS

CSS (Cascading Style Sheets) neboli kaskádové styly popisují pravidla, jak a kde se mají HTML elementy zobrazit na stránce. Hlavní myšlenkou CSS je oddělit strukturu HTML dokumentu od jeho vzhledu.

První verze CSS 1 vznikla roku 1996 a podporovala práci s textem, tabulkami, obrázky, barvami textů a pozadí, mezery slov, písmen a řádků, okraje, polstrování, polohování a jedinečnou identifikaci a klasifikaci skupiny atributů. V roce 1998 byla vydána verze CSS 2, která přinesla z-index, typy médií, absolutní, relativní a pevné umístění a podporu pro postižené. V roce 2011 byla vydána současná verze CSS 3, se kterou přišly animace, gradienty, 3D transformace a mnohé další funkce [2].

Existují tři možnosti zápisu kaskádových stylů do kódu. Přímo do HTML elementu, do elementu `head` a do externího souboru, přičemž poslední varianta je nejvhodnější.

2.3 PHP

PHP (PHP: Hypertext Preprocessor) je interpretovaný programovací jazyk, určený především pro programování dynamických webových stránek. PHP scripty jsou prováděny na straně serveru, kde je uživateli přenášen jejich výsledek, který je vložen do HTML obsahu.

PHP bylo napsáno v jazyce C a jeho počátky sahají do roku 1994, kde jeho autorem byl Rasmus Lerdorf, který ho původně používal ke sledování jeho online životopisu a dalších souvisejících osobních informací. Proto původní název byl „Personal Home Page Tools“ neboli PHP Tools. V průběhu času Rasmus implementoval do PHP nové funkce a v roce 1995 se rozhodl zveřejnit své zdrojové kódy, což umožnilo vývojářům jejich použití. Jak čas plynul, Rasmus neustále rozšiřoval a několikrát dokonce přepsal od základu zdrojové kódy, kdy nakonec roku 1996 představil PHP/FI a později tato verze dostala označení 2.0 [13].

Mezi aktuální PHP verze, které jsou podporované, patří verze 7.1, 7.2 a 7.3. Mezi ukončené verze potom patří verze 7.0, 5.6 a starší [15]. Důvod, proč chybí PHP 6 je ten, že v minulosti existoval jako experimentální projekt, ale nikdy nebyl vydán. Aby nedocházelo k nejasnostem, rozhodlo se, že nové PHP ponese označení PHP 7.

Verze PHP 5.x fungují na skriptovacím enginu Zend Engine II. Tento skriptovací engin se stará o převod PHP na HTML, které je odesláno uživateli. Nové verze PHP 7.x fungují na zcela novém enginu s označením „Next Generation“. Jedná se o refaktorovaný Zend Engine s dobře optimalizovaným využitím paměti. Díky novému enginu je PHP 7.x až dvakrát rychlejší než PHP 5.x. Další z větších novinek v PHP 7 je, že je možné deklarovat návratový typ metody. PHP 7 oproti PHP 5 nově podporuje 64-bit. Mezi další novinky v PHP 7 patří například nové operátory, skalární typy, anonymní třídy a úprava zpracování chyb [14].

S vývojem nových verzí PHP vznikají funkce, které často jsou žádoucí i ve starších verzích PHP. Proto vznikají takzvané polyfilly. Polyfill je plugin (kus kódu), který obsahuje novou funkčnost z nové verze a umožnuje její podporu ve starších verzích. Příkladem může být například funkce `random_bytes()`, která se nachází v PHP od verze 7 [12]. Pokud projekt obsahuje nižší verzi PHP, nemůže tuto funkci využívat. Existují potom tedy dvě varianty. První variantou je si požadovanou funkci naimplementovat, nebo si stáhnout `polyfill`, který požadovanou funkcionalitu poskytuje.

2.4 JavaScript

JavaScript neboli zkráceně JS je multiplatformní, objektově orientovaný, interpretovaný jazyk, dobře známý jako skriptovací jazyk pro webové stránky. JS je posílán ze serveru na stranu klienta, kde je vkládán přímo do HTML a vykonáván webovým prohlížečem. Pomocí JS je možné například reagovat na události, aniž by musel být kód odeslán na server. JS se stará se o interaktivitu s uživatelem.

JS byl vytvořen roku 1995 Brendanem Eichem pro firmu Netspace. Účelem bylo pomoci vývojářům s vylepšením webových stránek. Jednalo se například o animování drop-down menu a validaci vstupních hodnot formulářů. Než JS dostal své jméno, jeho původní název byl Mocha, později byl změněn na LiveScript a poté dostal své dnešní jméno JavaScript [11].

2.5 Bootstrap

Bootstrap je front-end framework pro tvorbu responzivního designu, který také obsahuje šablony založené na HTML a CSS pro formuláře, typografii, tlačítka, tabulky, a další vstupní komponenty.

Framework využívá principu mřížky, kdy stránku rozdělí do takzvaného gridu o maximálně dvanácti sloupcích. Následně je možné elementu na stránce přiřadit číslo, které reprezentuje jeho velikost na stránce v poměru s ostatními. Bootstrap před každým načtením stránky zjistí velikost zařízení, na kterém stránku zobrazujeme a podle definovaných hodnot zmenší nebo zvětší element na stránce v poměru k ostatním elementům.

Bootstrap byl vyvinut Markem Ottem a Jacobem Thortonem ve společnosti Twitter pod názvem Twitter Blueprint. Původně měl sloužit jako interní nástroj, ale v roce 2011 se jej rozhodli publikovat jako open-source projekt na GitHubu, kde je dnes projektem číslo jedna [1].

2.6 Šablonovací systém

Jedná se o API, které nahrazuje, modifikuje nebo vyhodnocuje vložené značky v HTML dokumentu, a vytváří tak výstupní soubor. V modelu MVC pracuje šablonovací systém na úrovni pohledu, kde odděluje logiku aplikace od zobrazení dat. Použití šablonovacího systému je velice výhodné, protože je možné používat stejnou šablonu opakovaně, pouze s tím rozdílem, že se do šablony posílají jiná data k zobrazení. Mezi další důvody,

proč je vhodné použít šablonovací systém je, že řeší z části bezpečnost. Například útok XSS je ošetřen automatickým escapováním vstupních dat. Mezi jedny z nejpoužívanějších šablonovacích systémů patří Twig a Smarty.

2.6.1 Smarty

Smarty je jeden z nejstarších šablonovacích systémů používaných v PHP. Systém napsal Monte Ohrt a Andrei Zmieski v roce 2002. Systém spadá pod licenci LGPL, která umožňuje jeho volné šíření. Aktuálně se Smarty nachází ve verzi Smarty 3, kde oproti starší verzi je základ systému celý přepsán, a proto není kompatibilní se staršími verzemi PHP. Pro používání Smarty 3 je nutné mít nainstalované minimálně PHP 5 [19]. Jednou z hlavních novinek ve verzi Smarty 3 je dědičnost [18].

2.6.2 Twig

Twig [24] je moderní šablonovací systém pro PHP vytvořený Fabienem Potencierem, který je také znám jako tvůrce Symfony Frameworku. Twig spadá pod BSD licenci, která umožňuje volné šíření licencovaného obsahu za dodržení několika podmínek. Twig je využíván několika frameworky, a to například Symfony a Node.js.

Mezi hlavní klíčové funkce Twigu patří:

- Rychlost – Twig kompiluje šablony až po prostý optimalizovaný kód PHP. Režijní náklady ve srovnání se standardním PHP jsou sníženy na minimum.
- Bezpečnost – Jednou z funkcí, jak zvyšuje Twig bezpečnost, je HTML escapovaní, které brání před útokem XSS. Druhou je takzvaný sandbox mód, který vyhodnocuje nedůvěryhodné části šablony. Ten je výhodný v částech webu, kde uživatel může měnit vzhled šablony.
- Rozšiřitelnost – Možnost vytvořit si nové funkce, filtry a tagy pro šablony.
- Dědičnost – Díky této schopnosti není potřeba kopírovat rodičovskou šablonu do každé jiné šablony, ale umožňuje vytvořit jeden univerzální layout, do kterého se nadefinují bloky, které se v potomcích nahrazují.

Twig aktuálně nabízí dvě verze, první verzí je 1.x a druhou 2.x. Verze 2.x nabízí nové funkcionality oproti verzi 1.x. Pro svůj běh požaduje minimálně verzi PHP 7.0.

2.7 Composer

Composer je balíčkovací systém pro správu závislostí v PHP. Dal by se přirovnat k příkazu `apt` v operačním systému Linux. Rozdíl spočívá v tom, že Composer se stará o balíčky v jednom daném projektu. Composer umožňuje snadno doinstalovat nové knihovny nebo upgradovat současné knihovny. Jedna z velkých výhod při vývoji projektu je, že umožňuje deklarovat knihovny, na kterých je projekt závislý. To při vývoji v týmu ulehčí správu a aktualizaci knihoven, protože nejste nuceni s projektem přesouvat i knihovny, ale stačí jen soubor `composer.json` nebo kombinace s `composer.lock`.

Soubor `composer.json` obsahuje závislosti projektu a používá se pro aktualizaci závislostí. U aktualizací je možné nastavit kritérium, pomocí kterého se bude projekt aktualizovat. Například pro aktualizaci pouze verze 3.4 se použije kritérium 3.4.*, tím bude zajištěno, že se nestáhne novější verze, ale jen aktualizace v dané verzi. Díky této schopnosti je možné držet závislosti vždy aktuální.

`Composer.lock` určuje aktuální verze závislostí. Používá se pro instalaci specifických verzí do projektu.

Pro instalaci závislostí pomocí Composeru existují dvě možnosti. První možností je příkaz `composer update`, který stáhne nejnovější možné verze závislostí na základě kritérií, a zaktualizuje soubor composer.lock. Druhým příkazem je příkaz `composer install`. Tento příkaz vyhledá v projektu soubor `composer.lock` a nainstaluje verze definované v tomto souboru. Pokud soubor `composer.lock` neexistuje, je spuštěn příkaz `composer update`.

Díky Composeru je možné pohodlně řešit bezpečnostní aktualizace, kde pomocí nadefinovaných kritérií lze snadno aktualizovat software třetích stran pomocí příkazu `composer update`.

Composer byl vydán roku 2013 [6] vývojáři Nilsnem Adermannem a Jordinem Boggianoem. Tvorba Composeru byla silně inspirována `npm`, který slouží jako balíčkovací systém pro Node.js a balíčkovacím systémem `bundler`, který používá Ruby [4].

2.8 Další technologie

2.8.1 AJAX

AJAX (Asynchronous JavaScript And XML) je způsob, jak se serverem komunikovat, bez nutností obnovení stránky. Pomocí AJAXu je možné zvýšit iterakci s uživatelem. Při běžné synchronní komunikaci je uživatel po odeslání formuláře přesměrován na stránku s výsledkem. AJAX umožnuje tento formulář odeslat asynchroně na pozadí a zareagovat na odpověď bez přesměrování.

2.8.2 DataTables

DataTables je JavaScriptová knihovna pro přidání interakce do tabulek. Přidává funkce jako vyhledávání, stránkování a řazení. Některé funkcionality knihovny je možné modifikovat. Lze například nastavit text u názvu prvků nebo skrýt určité prvky. Inicializace se provádí pomocí JavaScriptu, kdy nad klasickou tabulkou je zavolána metoda `DataTable()`, která zaktivuje knihovnu. Knihovnu je vhodné použít na místech, kde se spravují data a zmíněné funkce se využijí. Inicializace DataTables nad tabulkou a nastavení textu u tlačítek se provede pomocí následujícího scriptu.

```
<script>
        $(document).ready( function () {
            $('#table_id').DataTable();
    } );

    $('#table_id').dataTable( {
            "language": {
                "search": "Vyhledavani:",
                "paginate": {
                    "first":      "Prvni",
                    "last":       "Posledni",
                    "next":       "Dalsi",
                    "previous":   "Predchozi"
                },
                "infoEmpty":      "",
                "info":           ""
            }
        } );
</script>
```

2.8.3 CKEditor

Je typ WYSIWYG editoru, kterým je možné vylepšit vstup vstupního pole `textarea`. CKEditor patří mezi nejoblíbenější WYSIWYG editory, a to především kvůli jeho jednoduchosti a přehledné dokumentaci [3].

WYSIWYG je zkratka „What you see is what you get“ v překladu „co vidíš, to dostaneš“. Jedná se o editor, který funguje podobně jako Word nebo OpenOffice. Pomocí editoru je tedy i neprogramátor schopný naformátovat text a nemusí znát HMTL tagy, protože se o to editor stará sám na pozadí. Editor nám tedy umožňuje naformátovat si text podle běžných požadavků.

Pro zobrazení dat z WYSIWYG editoru pomocí šablonovacího nástroje je nutné, aby proměnná, která obsahuje data z editoru, byla vypisována bez escapování znaků. Pokud by data byla escapována, stalo by se, že by se v šabloně zobrazil čistý kód. V šablonovacím nástroji Twig se automatické escapování u proměnné vypne pomocí makra `|raw`.

2.8.4 jQuery

Je JavaScriptová knihovna, která definuje metody pro lehčí, rychlejší přístup k prvkům a jejich modifikací, reakci na události, a AJAXové volání. Knihovna je plně podporována všemi moderními prohlížeči.

Existují další podobné knihovny jQuery, a to například `jQuery UI`, která definuje spoustu uživatelských rozhraní pro interaktivitu s uživatelem a různé grafické prvky (widgety). Obsahuje interakce jako *drag and drop*, pole prvků, u kterých se pomocí myši dá měnit pořadí, dále obsahuje prvky jako záložky (Tabs), menu, výběr datumu (datePicker) a mnohé další.

3 Prostředí pro vývoj webových aplikací

V současné době existuje několik možných nástrojů pro vytvoření prostředí, které umožní vývoj webových aplikací na osobním počítači. Níže jsou popsány dva nejznámější nástroje.

3.1 XAMPP

XAMPP je multiplatformní softwarový balík, který v základu obsahuje technologie pro vytvoření webového serveru. Balík obsahuje Apache, MariaDB, PHP a Perl. K tomuto softwaru je možné si doinstalovat další možné moduly, například WordPress, Moodle, TestLink, a další.

XAMPP byl vyvinut roku 2003 [20] skupinou Apache Friends. Význam písmen, ze kterých je složeno XAMPP je následující: písmeno X odkazuje na to, že je XAMPP multiplatformní (cross-platform), dále ostatní písmena ukazují na první písmeno názvu softwaru, který balík obsahuje. Jsou to Apache, MariaDB, PHP, Perl.

Použití tohoto řešení pro prostředí se hodí tam, kde vyvíjíme menší projekt. Výhodou tohoto řešení je, že se XAMPP postará o všechen potřebný software pro vytvoření webového serveru. Jako nevýhodu by se mohlo například zmínit, že vývojář může pracovat na více projektech zároveň. Tím pádem se může stát, že v různých projektech budou různé verze softwaru. Tím může vzniknout problém s kompatibilitou. Následující nástroj řeší tento problém.

3.2 Docker

Docker je multiplatformní nástroj umožňující vývojáři vývoj aplikace v sandboxu zvaném kontejner. Hlavním benefitem Dockeru je, že umožňuje uživateli zabalit aplikaci s jejími všemi závislostmi, které potřebuje, do standardizované jednotky pro vývoj softwaru (kontejneru). Tato jednotka je dále přenositelná a ulehčuje vývoj. Není nutné řešit, zda vývojář má na svém zařízení nainstalované všechny technologie, které potřebuje pro vývoj, ale stačí pouze spustit tento kontejner, a ten se postará o vytvoření virtuálního prostředí, které bude obsahovat všechny závislosti pro spuštění

aplikace. Důležitým souborem při vytváření kontejneru je soubor zvaný `Dockerfile`, ve kterém se definují závislosti pro daný projekt.

Docker vznikl roce 2014 u společnosti Docker, Inc. a od té doby nabírá na popularitě. Původně byl vyvinut pro operační systém Linux, nyní je ale plně podporován i na macOS a Windows [10].

Toto řešení je vhodné použít jak při vývoji menšího projektu, tak i pro projekt většího rozměru, kde vývojový tým je složen z mnoha členů. Díky Dockeru jsme schopni vytvořit flexibilní vývojové prostředí, které lze rychle modifikovat a hlavně rychle přenášet. Nevýhodou je nutnost znát alespoň základní principy fungování Dockeru.

4 Tvorba webových stránek

V dnešní době existuje několik možností, jak vytvořit webové stránky. Od možnosti čistého HTML, přes PHP s využitím předpřipravených komponent, nebo přes použití různých frameworků typu `Nette`, `Symfony`, až po redakční publikační systém, například WordPress. Všechny tyto možnosti mají svoje výhody, nevýhody a omezení. V následujících sekcích budou tyto typy popsány.

4.1 Čisté HTML

Jedná se o nejjednoduší možnost pro tvorbu webových stránek, která existuje už od počátku WWW. Použití čistého HTML nese omezení v tom, že jsme schopni vytvořit pouze statický obsah webových stránek, proto tuto volbu je vhodné použít u webů, které často nemění svůj obsah. Typicky informační stránky.

4.2 Čisté PHP

Výhodou použití čistého PHP je oproti předešlé variantě v tom, že jejichž prostřednictvím je možné vytvořit dynamický obsah webových stránek. Což znamená, že během jejich používání je možné měnit jejich obsah. Čímž je tato volba vhodná při vývoji složitějších webových aplikací. Samotnou nevýhodou čistého PHP je, že je nutné napsat značné množství kódu, který už někdy někdo napsal. Druhou nevýhodou je, že je potřebné hodně věcí zvážit, jako například strukturu projektu, organizaci zdrojového kódu, bezpečnost a logiku aplikace.

4.3 PHP framework

PHP frameworky zjednodušují vývoj webových aplikací napsaných v PHP tím, že poskytují základní strukturu projektu, předimplementované metody, které usnadňují a urychlují vývoj oproti čistému PHP. Dále obsahují základní ochranu před webovými útoky, napomáhají dobré udržitelnosti kódu, a mnohé další funkce. Jako nevýhodu je nutno zmínit pomalejší dobu odezvy oproti čistému PHP, a jeho náročnost v tom

smyslu, že kompletní framework obsahuje řadu funkcí, které se při vývoji menšího projektu nevyužijí, a tím zbytečně zvětšují aplikaci.

Mezi nejznámější PHP frameworky ve světě patří například `Laravel` nebo `Symfony`. V české komunitě je potom hodně rozšířený framework `Nette`, který zde vznikl.

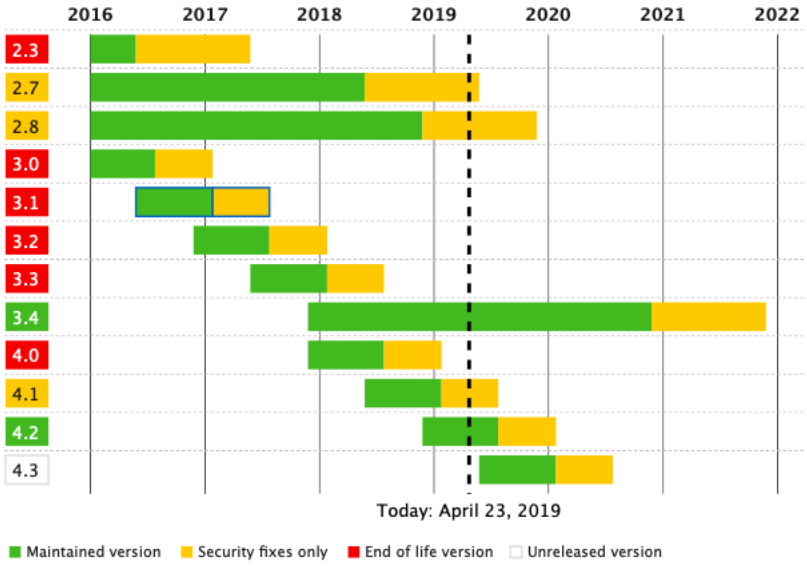
4.4 Symfony komponenty

Symfony komponenty je skupina oddělených a znovupoužitelných knihoven pro tvorbu webových aplikací. Komponenty jsou podmnožinou Symfony frameworku, které dokáží fungovat samostatně. Jedna z výhod komponent je, že předimplementují metody, které usnadní a zrychlí vývoj webové aplikace. Další výhodou je, že není nutné použít celý framework, ale jen části, které jsou potřeba. Kompletní frameworky jako `Nette` nebo `Symfony` se skládají z velké sady knihoven, které se při vývoji menší webové aplikace nevyužijí.

Komponenty nám například zajišťují objektově orientovanou vrstvu pro HTTP specifikaci, vytváří strukturovaný proces pro převod dotazu na odpověd, umožňují efektivnější práci s routováním, infrastrukturu zabezpečení, různé nástroje pro práci s konfiguračními soubory, a mnohé další funkce. Vyjmenované funkce byly jen špičkou ledovce existujících schopností, které je možné získat díky Symfony komponentám. Detailní popis všech komponent lze nalézt v dokumentaci [22] na oficiálních stránkách Symfony frameworku.

Aktuálně existují dvě verze Symfony komponent 3.4 a 4.2, které mají aktivní údržbu a tři verze, které řeší jen bezpečnostní rizika. Nejnovější verzí Symfony komponent je verze 4.2, která byla vydána v listopadu 2018 a pro své fungování potřebuje minimálně verzi PHP 7.1.3, nebo vyšší. Úplný konec podpory pro verzi 4.2 je udáván na leden 2020. Další verze, která má zajištěnou aktivní podporu je verze 3.4. Tato verze byla vydána v listopadu 2017 a konec podpory je v listopadu 2021. Pro fungování této verze je nutné mít nainstalovou verzi PHP 5.5.9, nebo vyšší [23]. Detailní přehled údržby verzí je zobrazen na obrázku 4.1.

Protože byly Symfony komponenty explicitně zmíněny v zadání, nebudou v této části dále popisovány jednotlivě, ale budou popsány až v analýze, tj. v kapitole 7.2.1.



Obrázek 4.1: Plán podpory jednotlivých verzí [23]

4.5 WordPress

Je open-source redakční publikační systém napsaný v PHP a MySQL. Wordpress umožňuje vytvoření široké škály webů, od blogů, až ke komerčním webům. Jednou z jeho dominantních vlastností je, že jeho obsluha je jednoduchá. Pro vytvoření webu nemusí uživatel umět ani programovat. Veškerá tvorba webu se děje v administraci WordPressu, kde si uživatel přes již hotový systém jen přidává a modifikuje jeho obsah. Všechno se děje v uživatelsky přívětivém prostředí. Pokud uživatel požaduje od webu nějakou novou vlastnost, může se podívat do nabídky pluginů, popřípadě si ji naprogramovat sám. Vzhled webu se řeší pomocí šablon. Šablony se dají stáhnout na internetu a většinou už implementují responzivní design. Nevýhodou je, že pokud uživatel chce provádět nějaké změny v kódu, musí mít znalosti PHP. Pokud chce udělat grafickou modifikaci, musí mít znalosti CSS.

Z důvodu velké rozšířenosti vzniká problém s bezpečností. Existují specializované weby, které po zadání verze WordPressu zobrazí možnosti, jak lze daný web napadnout. To je jeden z důvodů, proč je použití WordPressu nebezpečné.

4.6 Závěr

Jelikož se u projektu TbUIS bude jednat o dynamické webové stránky, zůstávají v úvahu všechny možnosti, kromě čistého HTML, které tuto možnost neumožňuje. Z důvodu omezení (viz kapitola 7.1.4), nemůže být použit ani framework, protože by se mohlo stát, že by nějaké nastavení serveru mohlo vést k nefukčnosti nějaké části frameworku. Dále kvůli velkému důrazu na bezpečnost není vhodné použití WordPressu. Proto zbyly poslední dvě možnosti, a to čisté PHP a Symfony komponenty. Protože Symfony komponentám nic nebrání, budou použity v kombinaci s čistým PHP.

5 Bezpečnostní rizika webových aplikací

V následujících sekcích jsou stručně popsány základní typy webových útoků, včetně metodiky, jak se před těmito útoky bránit.

5.1 Cross Site Scripting

Cross Site Scripting, dále jen XSS je typ útoku, který spočívá ve vložení škodlivého kódu, který když se zobrazí, tak se spustí. Typickým příkladem může být neošetřený formulář v diskuzi pro vkládání komentářů, protože po vložení komentáře se zobrazí všem ostatním účastníkům diskuze. Klasicky se může jednat o vyskakovací okno s reklamou, v horším případě o odcizení session. Proti XSS se dá bránit pomocí escapování znaků. Toto escapovaní za nás řeší šablonovací systém, samozřejmě pokud jej používáme.

5.2 Cross Site Request Forgery

Cross Site Request Forgery, dále jen CSRF je útok, který podvrhne požadavek mezi různými stránkami. Před útokem CSRF musí útočník vědět, na co chce útočit. Popis útoku: Uživatel se přihlásí do internetového bankovnictví a neodhlásí se. Mezi tím uživatel navštíví útočníkovu stránku a zobrazí si obrázek. V momentě, kdy uživatel klikne na obrázek, spustí se na pozadí akce, která například simuluje odeslání formuláře pro nastavení trvalého příkazu v internetovém bankovnictví. Proti CSRF je možno se bránit například pomocí skryté hodnoty, která je vložena do formuláře, kterou server generuje.

5.3 SQL injection

SQL injection je typ útoku zaměřující se na databázi. Útok spočívá v úpravě SQL dotazu v útočníkův prospěch. Pomocí útoku je útočník například schopný v lepším případě získat všechna data z databáze, nebo v horším případě všechna data vymazat. Proti útoku se dá bránit tím, že se například použije nějaké rozšíření pro práci s databází. V PHP je možné například použít rozšíření PDO. Dotaz se potom vytvoří tak, že se do něj

nevkládají přímo proměnné, ale vytvoří se takzvaný parametrizovaný dotaz. Ten se vytváří tak, že místo hodnot se vkládají do SQL dotazu zástupné znaky. Následně se dotaz a hodnoty předají do PDO odděleně, a ten se postará o jejich bezpečné dosazení (`prepare()`) a následnou exekuci (`execute()`) dotazu.

6 Architektura webových aplikací

Architektura webových aplikací nám udává strukturu projektu. Architektura nám rozděluje vývoj do několika nezávislých částí, které se dají vyvíjet odděleně. Nejen že se zpřehlední projekt, ale zároveň se zrychlí vývoj projektu, protože je možné vyvíjet paralelně. To znamená, že kodér může pracovat na vzhledu stránky, zatímco programátor programuje logiku aplikace. Tyto části si jen mezi sebou vyměňují data, takže je jen potřeba definovat proměnné. Protože části vyvíjíme odděleně, změna v jedné části nezpůsobí nefunkčnost v jiné. Níže jsou popsány dva architektonické vzory pro vývoj webových aplikací.

6.1 Model-View-Controller

Model-View-Controller dále jen MVC je návrhový vzor pro vývoj webových aplikací. Jeho smyslem je oddělit pohled, data a logiku aplikace do tří nezávislých částí, kde změna v jedné části nemá vliv na funkčnosti části jiné.

Část `Model` reprezentuje data, která jsou přenášena mezi částí `View` a `Controller`, nebo jakákoliv jiná data týkající se obchodní logiky. Tato část se například stará o komunikaci s databází. `Model` má zároveň schopnost poslat data do `View`.

`View` představuje veškerou logiku uživatelského rozhraní. Do této komponenty patří například šablonovací systém. View má zároveň možnost vyměňovat si data s částí `Model`.

`Controller` zpracovává požadavky, a stará se o logiku aplikace. Jeden `Controller` může obsahovat více `View`. Tyto `View` si mění dle potřeby [16].

Typickým příkladem může být požadavek na zobrazení dat o uživateli, který je uložen v databázi. Postup zpracování je takový, že `Controller` přijme požadavek na zobrazení dat o konkrétním uživateli, řekne `Modelu`, aby mu našel data o uživateli a ta mu vrátil. Model získá data pomocí dotazu do databáze a ta vrátí `Controlleru`. `Controller` následně tato data vezme a předá je `View` k zobrazení.

6.2 Model-View-Presenter

Model-View-Presenter zkráceně MVP je velmi podobný MVC. Jako MVC odděluje aplikaci do tří různých částí s tím rozdílem, že místo `Controlleru` je `Presenter`, a mírně se mění i chování ostatních komponent.

Rozdíly mezi MVC a MVP je takový, že část `Presenter` stojí mezi `View`, a `Modelem` a stará se o veškerou komunikaci, která mezi nimi putuje. To znamená, že `Model` v MVP nemůže jako `Model` v MVC posílat data přímo do `View`. Další rozdíl je ten, že `Presenter` má právě jedno `View`. To jsou dva hlavní rozdíly mezi těmito architekturami.

7 Vývoj mikroframeworku

7.1 Analýza potřeb

7.1.1 O projektu TbUIS

TbUIS, je výzkumný projekt, zaměřený na ověřování kvality nově vyvíjených testovacích metod. Hlavním smyslem projektu TbUIS je poskytnout nezbytný software se (semi)realistickým použitím, nad jakým bude možné psát testy, tedy slouží jako SUT (System Under Test).

Projekt TbUIS se skládá ze dvou částí. První částí je UIS a druhou Error-Sedder.

UIS

UIS je webová aplikace, která simuluje chování reálného univerzitního informačního systému, fungujícího jako SUT. V této aplikaci existují tři typy uživatelů, kteří se mohou přihlašovat do aplikace a používat její funkce. První typ uživatele reprezentuje studenta univerzity, druhým typem uživatele je učitel a posledním typem uživatele je nepřihlášený uživatel. Aplikace UIS by měla být na první pohled pochopitelná každému vysokoškolsky vzdělanému uživateli. To byl jeden z důvodů volby této domény.

Aplikace se skládá z několika dílčích entit, které mohou mít volitelně korektní (bezchybnou), nebo naopak injektovanou chybovou funkčnost. Cílem je aplikaci důsledně otestovat a najít všechny zanesené chyby [8].

Error-Seeder

Error-Seeder je samostatná aplikace, která slouží jako správce poruchových entit a má schopnost zanášet do aplikace UIS chyby. Princip spočívá v možnosti výběru entit, z kterých se aplikace UIS skládá. Tím je možné vytvořit aplikaci UIS z korektních entit, nebo chybně fungujících entit. Díky tomu je možné v krátkém čase vytvořit několik různých verzí UIS. Takto vytvořené verze je možné použít k ověření kvality testovacích metod a pro trénovací účely testerů.

Výstupem aplikace Error-Seeder je WAR soubor, který slouží pro distribuci aplikace UIS koncovým uživatelům. Tento soubor je následně možné nasadit na aplikační server (například `Apache Tomcat`) [8].

7.1.2 Požadované funkce

Ze zadání vyplývá, že je nutné, aby aplikace splňovala následující požadavky:

- Schopnost měnit dynamicky strukturu a obsah webových stránek.
- Diskuzní fórum.
- Důraz na bezpečnost a udržitelnost kódu.
- Responzivní design.
- Modifikace pomocí šablon.
- Možnost stahování souborů.

7.1.3 Porovnání vědeckých projektů

V této analýze byly porovnány projekty STILL [21], JUTS [17] a jednotlivé stránky výzkumných skupin na katedře KIV [7]. Hledal se společný průnik těchto vědeckých projektů, a zároveň se analyzovala chybějící funkčnost, kterou je nutné doplnit. Závěrem této analýzy vyšlo, že potřeby pro projekt TbUIS jsou následující:

- Informace o projektu.
- Podrobný popis částí projektu.
- Výsledky experimentů.
- Stahování programů a dat.
- Odkazy na články.
- Diskuze.

7.1.4 Omezující faktory

Webová aplikace bude uložena na serveru katedry KIV, a z toho vyplývají různá omezení. Mezi nejzásadnější patří aktuální instalovaná verze PHP, která se na serveru nachází ve verzi 5.6. Mezi další omezení patří instalace Apache, kde bude nutné si dát pozor na jeho nastavení, a brát ho v potaz při návrhu mikroframeworku.

Omezující faktor po implementační stránce je ten, že se v budoucnu plánuje realizace podobných projektů jako TbUIS, proto je nutné udělat aplikaci generickou, aby ji bylo možné modifikovat pro potřeby dalších projektů.

7.2 Návrh mikroframeworku

7.2.1 Symfony komponenty

Pro tvorbu mikroframeworku budou použity níže popsané Symfony komponenty. Důvod pro vybrání následujících komponent je ten, že zajišťují vytvoření a funkčnost mikroframeworku, nebo se jedná o nástroje pro ulehčení vývoje.

HttpKernel

Komponenta HttpKernel poskytuje strukturovaný proces pro převod požadavku (requestu) na odpověď (response) s využitím komponenty `EventDispatcher`. Tato komponenta vytváří jádro celého mikroframeworku, a proto se jedná o nejdůležitější část celého mikroframeworku.

EventDispatcher

EventDispatcher implementuje návrhový vzor Prostředník (Mediator) a Pozorovatel (Observer). Komponenta umožňuje zaregistrování posluchačů (Listener) na specifické události. Typickým použitím je zaregistrování routovacího listeneru, který čeká na přijetí requestu. Po přijetí requestu se spustí funkce nadefinovaná v listeneru, která v tomto případě vykonnává převod requestu na příslušnou akci kontroleru. Úplný seznam událostí, na které je možné zaregistrovat události, je uveden v dokumentaci komponent [22].

HttpFoundation

Pomocí této komponenty je možné přistupovat k defaultním globálním PHP proměným a funkcím pomocí objektově orientované vrstvy, kterou komponenta HttpFoundation vytváří. Výhodou této komponenty je, že umožní psát lépe strukturovaný a lépe testovatelný kód. Viz ukázka:

```php
<?php
/* Ziskani hodnoty bez HttpFoundation */
if (isset($_GET['jmeno'])) {
        $jmeno1 = $_GET['jmeno'];
} else {
        $jmeno1 = 'default';
}

/* Ziskani hodnoty s pouzim HttpFoundation */
$request = Request::createFromGlobals();

$jmeno2 = $request->get('jmeno', 'default');
?>
```

Routing

Komponenta umožňuje namapovat URL adresu na specifickou akci v kontroleru. Tím se zajistí, že po zadání adresy do adresního řádku se spustí příslušné vykonání akce. Routování se skládá ze tří důležitých částí. První částí je `RouteCollection`, která obsahuje kolekci rout. Druhou částí je `RequestContext`, který obsahuje informace o aktuálním requestu, a poslední částí je `UrlMatcher`, který provádí mapování requestu na konkrétní routu. Routování je možné zaregistrovat do `EventDispatcheru` jako listener, a tím zajistit komplexnost mikroframeworku.

DependencyInjection

Komponenta DependencyInjection poskytuje standardizovaný způsob, jak vložit závislost do konstruktoru tříd. Důležitou komponentou je kontejner, který specifikuje, a drží všechny dostupné služby v aplikaci. Výhodou této komponenty je to, že zjednodušuje proces vytváření objektů a jejich předávání, kdy závislosti budou automaticky vloženy. Díky této vlastnosti se vytvoří volná vazba mezi částmi aplikace.

Form

Je nástroj pro tvorbu formulářů, který znatelně ulehčí a urychlí tvorbu formulářů. Komponenta umožňuje předat formulář do šablony, kde je vykreslen pomocí nadefinovaných maker viz 7.2.2. Formuláře nabízí širokou škálu různých typů vstupních polí, jako jsou například obyčejné textové pole, vstup pro heslo, rozbalovací nabídku, paletu barev, a další.

Yaml

Nástroj pro načítání YAML (tj. konfiguračních) souborů a jejich následný převod na PHP pole. Protože konfigurační soubory jsou uloženy ve formátu YAML, je tento nástroj velmi vhodný.

Config

Nástroj pro načítání konfiguračních souborů ke své činnosti využívá zmíněnou YAML komponentu.

7.2.2 Další technologie

Twig extensions

Jako šablonovací systém byl zvolen Twig, protože Symfony obsahuje balíček pro práci s tímto šablonovacím systémem, který se jmenuje Twig extensions. Díky balíčku je možné přidat do Twigu různá rozšíření. Například vykreslení formulářů, kdy se specifikuje šablona, jakým stylem se má formulář vykreslit, nebo routovací rozšíření, které vytvoří nové makro `path()`, které umí z názvu routy vygenerovat URL adresu.

7.3 Struktura budoucí apikace

7.3.1 Veřejná část

Veřejná část bude přístupná všem typům uživatelů. Proto bude kladen velký důraz na její vzhled a bezpečnost.

Vzhled

Webová aplikace bude obsahovat menu s odkazy na Homepage, forum, přihlášení, registraci, a na dynamické stránky, které se definují v administraci.

Jako šablona bude použita „Law: Free HTML5 Bootstrap Template for Law Firm Websites“ [5], která implementuje responzivní design a je uživatelsky přívětivá. Šablona spadá pod licenci `Creative Commonst Attribution 3.0`.

Fórum

Z analýzy řešeného problému vyplynulo, že se neočekává žádný složitý diskuzní systém, který by zahrnoval lajkování, a další podobné vlastnosti

známé z rozsáhlých diskuzních fór. Účel tohoto diskuzního fóra je pouze předat relevantní informaci, popřípadě požadavek, a na něj stručně reagovat.

Fórum bude členěno do témat, kde každé téma bude mít své vlákno. Témata budou uložena v tabulce, ve které bude fungovat stránkování, a budou seřezena sestupně podle poslední změny v tématu.

Příspěvky do fóra budou moci přidávat uživatelé na základě nastavení v administraci.

7.3.2 Neveřejná část

Bude se jednat o administaci webové aplikace. Do této části webu bude mít přístup jen uživatel typu `admin` nebo `copywriter`, kde se bude lišit jejich oprávnění. Uživatel typu `admin` bude moci vytvářet a editovat obsah dynamických webových stránek, spravovat uživatele, a měnit nastavení webové aplikace. Uživatel typu `copywriter` bude smět pouze editovat obsah existujících dynamických webových stránek.

Jako šablona bude použita šablona „Glance Design Dashboard Bootstrap Responsive Web Template“ [26]. Šablona implementuje responzivní design a je přehledná. Šablona spadá pod licenci `Creative Commons Attribution 3.0 Unported`.

Správa stránek

Bude obsahivat tabulku s přehledem stránek, která bude implementovat knihovnu DataTables, pro lepší správu dat. Dále budou existovat možnosti pro vytváření a editaci jednotlivých stránek. Bude existovat několik typů webových stránek, kde každá se bude chovat trochu jinak, viz níže.

Typy stránek

Ačkoli se bude každý typ stránky od jiných trochu lišit, budou mít stránky společné vlastnosti. Mezi společné vlastnosti bude patřit:

- Název v menu.
- Název v URL.
- Nastavení typu stránky.
- Jestli je stránka viditelná na veřejné části.

Dynamická stránka Jedná se o nejzákladnější stránku, kterou bude možné vytvořit. Vytvoření nebo editace bude obsahovat základní vlastnosti a WYSIWYG editor.

Homepage Bude klasická dynamická stránka s tím rozdílem, že tato stránka nepůjde odstranit. Jedná se o domovskou stránku webové aplikace.

Stránka s tabulkou Stránka, která bude sloužit především pro publikování výsledků nebo pro stažení dat. Tabulka bude dynamická, takže bude možné nastavit volitelný počet sloupců s jejich nadpisy. Záznam v prvním sloupci bude zároveň sloužit jako odkaz, takže bude možné na tento sloupec dát odkaz na stažení souboru nebo odkaz na jinou webovou stránku. Jednotlivým buňkám bude možné nastavit barvu jejich pozadí.

Počet sloupců se bude nastavovat pomocí speciální akce, která bude existovat u tohoto typu stránky. Tato speciální akce bude obsahovat dva formuláře, jeden pro nastavení počtu sloupců, a druhý pro definování jejich nadpisů.

Vkládání záznamů do tabulky se bude provádět pomocí další speciální akce u tohoto typu stránky. Ta administrátora přesměruje na správu záznamů. V této správě bude administrátor přidávat nové záznamy nebo editovat existující. Při editaci nebo vytváření záznamu se bude dynamicky generovat formulář, který bude obsahovat takový počet vstupů, jako je počet sloupců v tabulce. Ke každému vstupu bude navíc existovat vstup pro barvu pozadí buňky. Barva bude vybírána pomocí paletky barev (`colorPickeru`).

Fórum Představuje stránku pro zobrazení diskuzního fóra. Tato stránka existuje právě jednou a bude možné u ní nastavit pouze název záložky v menu a její viditelnost. Tuto stránku nebude možné smazat, ale pouze skrýt.

Správa uživatelů

Bude se jednat o tabulku s přehledem uživatelů. Tabulka bude implementovat knihovnu DataTables pro efektivnější práci s uživateli. Díky knihovně bude možné lépe spravovat přehled uživatelů.

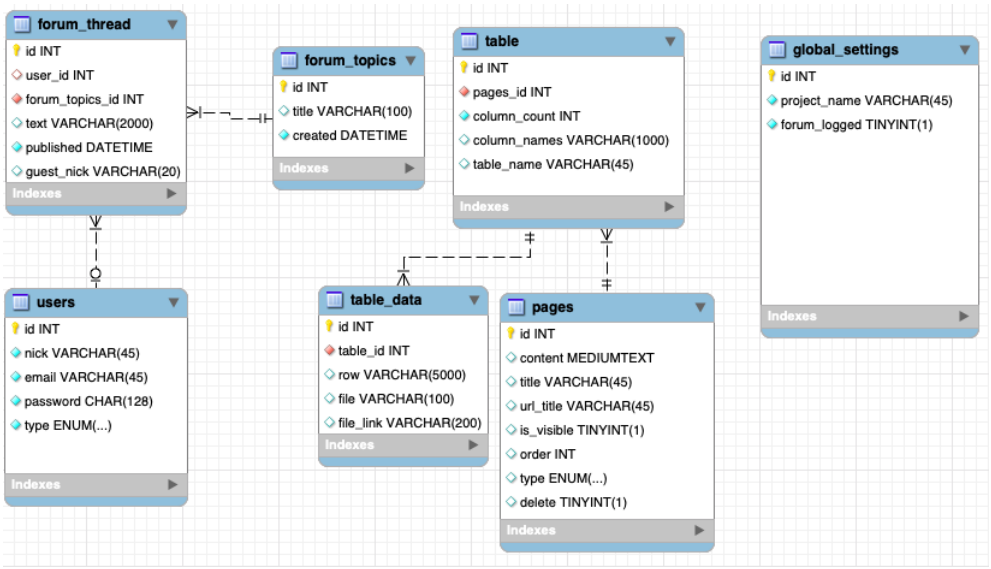
Nastavení

Zde se bude měnit nastavení projektu. Nastavení bude obsahovat formulář, ve kterém bude možné nastavit název projektu. Ten se bude

zobrazovat ve veřejné části. Dále zde bude možné navolit nastavení fóra pro přidávání příspěvků.

7.3.3 Návrh databáze

Z předchozích kapitol návrhu mikroframeworku vyplývá, že bude nutné mít pod webovou aplikací databázi pro ukládání dat. Proto byla navržena následující struktura databáze, viz 7.4.9.

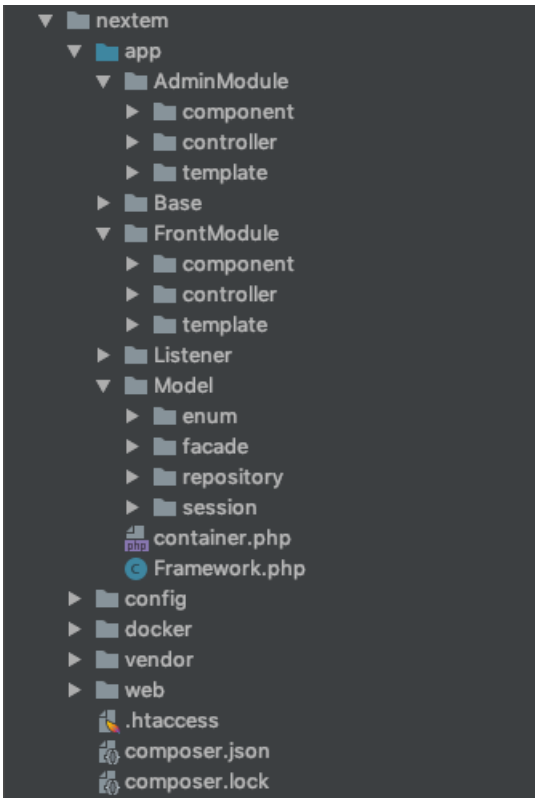


Obrázek 7.1: Návrh databázové struktury

7.4 Implementace

7.4.1 Struktura aplikace

Struktura webové aplikace je znázorněna na následujícím obrázku. Struktura je členěna do několika částí a dodržuje architekturu MVC.



Obrázek 7.2: Struktura webové aplikace

AdminModule

Zde jsou uloženy zdrojové kódy pro administrátorskou část webové aplikace.

component Komponenty administrátorské části. Jejím obsahem jsou formuláře použité v této části.

controller Kontrolery administrátorské části.

template Obsahuje hlavní šablonu pro administrátorskou část a podšablony pro jednotlivé akce v kontrolerech.

Base

Společné třídy, které jsou společné jak pro `AdminModule`, tak pro `FrontModule`. Jedná se především o společnou funkčnost, kterou třídy v modulech následně dědí.

FrontModule

Zde jsou uloženy zdrojové kódy pro veřejnou část webové aplikace. Obsahuje stejnou strukturu jako `AdminModule`, viz 7.4.1.

Listener

Listenery webové aplikace.

Model

Obsahuje třídy pro práci s daty. Tyto třídy jsou dále rozděleny do složek podle jejich působnosti.

- **enum – Obsahuje výčtové typy webové aplikace.**
- **facade – Obsahuje třídy s návrhovým vzorem fasáda.**
- **repository – Třídy pro práci s databází.**
- **session – Třídy pro práci se session.**

config

Konfigurační soubory pro definování parametrů, definování závislostí tříd a routování akcí.

docker

Konfigurační a inicializační soubory pro vytvoření Docker kontejneru pro vývoj webové aplikace na lokálním zařízení.

vendor

Třídy a knihovny třetích stran nainstalovaných pomocí Composeru.

web

Soubory, které jsou přístupné veřejnosti. Jde především o CSS, JS soubory a soubor `index.php`

Bez složky

- `container.php` – Obsahuje konfiguraci mikroframeworku, viz 7.4.2.
- `Framework.php` – Třída, která představuje mikroframework.
- `composer.json a composer.lock` – Viz 2.7.

7.4.2 Mikroframework

Mikroframework je vytvářen v souboru `index.php` z kontejneru závislostí, který je nadefinovaný ve třídě `container.php` a pomocí konfiguračních souborů ve složce `config`, které kontejner závislostí načítá.

Konfigurační soubory

Všechny konfigurační soubory jsou typu YAML (.yml). Webová aplikace obsahuje celkem sedm konfiguračních souborů. Níže jsou jednotlivé konfigurační soubory popsány.

parameters.yml Inicializace parametrů pro webovou aplikaci. V tomto souboru jsou například uloženy proměnné pro připojení se do databáze.

routes.yml Konfigurační soubor pro definice routování. V tomto konfiguračním souboru se mapuje adresa na příslušnou akci v kontroleru. Tento soubor je používán routovací komponentou. Ukázka:

```
homepage.front:
  path: /
  defaults: {_controller: 'Controller::defaultAction'}
```

controllers_admin.yml V tomto souboru jsou konfigurovány kontrolery v `AdminModule`. U jednotlivých kontrolerů se definuje označení, jejich rodičovská třída (pokud třída nějakou dědí), třída kontroleru, argumenty, které se mají předat do konstruktoru, a případně další konfigurace. Viz příklad:

```
App\AdminModule\Controller\LoginController:
  class: App\AdminModule\Controller\LoginController
  parent: App\AdminModule\Controller\BaseSecureController
  arguments: ['@App\Model\Repository\UsersRepository']
```

controllers_front.yml Viz předchozí odstavec s tím rozdílem, že v tomto konfiguračním souboru jsou nadefinovány kontrolery pro `FrontModule`.

models.yml Konfigurační soubor pro konfiguraci tříd ve složce `Model`. Používá se zde stejná konfigurace jako v konfiguračním souboru pro kontrolery, jen s odlišnými třídami.

components.yml Konfigurační soubor pro komponenty. Stejná logika jako pro předešlé konfigurační soubory.

services.yml Importuje do sebe předešlé konfigurační soubory. Důvodem, proč tento soubor existuje je to, že je nutné zmíněné konfigurační soubory načíst do kontejneru závislostí, viz 7.4.2, proto je vhodné načíst jen jeden soubor, ne každý zvlášť.

Kontejner závislostí

V kontejneru závislostí jsou registrovány, tzn. definovány služby a vytvářeny závislosti mezi nimi. Kontejner stojí na Symfony komponentě `DependencyInjection`. V kontejneru se nejdříve vytvoří instance třídy `DependencyInjection\ContainerBuilder`, následně jsou do ní registrovány objekty metodou `register()`. Metoda obsahuje dva parametry. Prvním parametrem je označení registrovaného objektu (ID), který musí být jedinečný, a je typu `string`. Druhým parametrem je třída objektu. Registrované službě lze předat parametry do konstruktoru metodou `setArguments()`. V následující ukázce je znázorněna registrace služby `Framework` s předanými dříve registrovanými objekty, které jsou odkázány pomocí ID.

```
$containerBuilder
          ->register('framework', Framework::class)
    ->setArguments([
        new Reference('dispatcher'),
        new Reference('controller_resolver'),
        new Reference('request_stack'),
        new Reference('argument_resolver'),
    ]);
```

V kontejneru jsou postupně registrovány následující služby:

- `Konfigurační soubory`
- `Routing\RequestContext`
- `Routing\Matcher\UrlMatcher`
- `Routing\Generator\UrlGenerator`
- `HttpFoundation\RequestStack`
- `HttpFoundation\Request`
- `HttpKernel\Controller\ContainerControllerResolver`
- `HttpKernel\Controller\ArgumentResolver`
- `HttpKernel\EventListener\RouterListener`
- `HttpKernel\EventListener\ResponseListener`
- `App\Listener\ExceptionListener`
- `EventDispatcher\EventDispatcher`
- `App\Framework`
- `App\Model\session\SessionModel`

Důvodem, proč byl tento soubor vytvořen, je to, že umožňuje flexibilní a přehledné definování služeb pro budoucí správu projektu.

Reakce na události

Reakce na události jsou implementavány s použitím Symfony komponenty `EventDispatcher`. V kontejneru závislostí jsou registrovány listenery pro reakci na routování (`RouterListener`), přípravu response (`ResponseListener`), kontrolu přístupu do administrace (`AdminAccessListener`) a reakci na chybu (`ExceptionListener`). Následně jsou tyto `listery` přidány při registrování služby `EventDispatcheru` v kontejneru závislostí. Listenery jsou přidány metodou `addMethodToCall()`. Ta volá metodu ve třídě, která byla definována v metodě `register()`, tzn. v tomto případě třídu `EventDispatcher`.

Jednotlivé třídy listenerů implementují interface `EventSubscriberInterface`, který obsahuje metodu

`getSubscribedEvents()`, která mapuje událost na metodu v listeneru. V následující ukázce je znázorněno mapování události, kterou vyvolala chyba, na metodu `onKernelException`.

```
public static function getSubscribedEvents() {
  return [KernelEvents::EXCEPTION => 'onKernelException'];
}
```

7.4.3 Base třídy

Následující třídy implementují funkčnosti, které jsou společné pro oba moduly, tj. `FrontModule` a `AdminModule`.

BaseObject

Obsahuje metody a proměnné, které využije každá třída. V této třídě je definována proměnná pro přístup do kontejneru závislostí. Díky ní jsou třídy, které tuto třídu dědí, schopny přistupovat do kontejneru závislostí. Dále je zde implementovaný přístup do session, přesměrování a vyskakovací zprávy (Flash message).

BaseController

Obsahuje společnou metodu pro kontrolery. Je zde nakonfigurován Twig pro vykreslování šablon. Třída dědí třídu `BaseObject`.

BaseForm

Konfigurace továrny pro formuláře a pomocné metody pro práci s nimi. Třída zároveň obsahuje abstraktní metodu `createComponentForm()`, ve které potomci definují tělo formuláře. V konstruktoru této třídy je volána metoda `createComponentForm()`. Díky tomu je realizováno automatické vygenerování formuláře, po vytvoření objektu třídy. Třída dědí třídu `BaseObject`.

7.4.4 Vytvoření a zpracování formulářů

Pro každý formulář je vytvořena samostatná třída, která je umístěna podle modulu, kde je použita ve složce `component\form`. Pro vytváření formulářů je použita Symfony komponenta `Form`. Třídy jednotlivých formulářů nejčastěji obsahují konstruktor, metodu pro nastavení výchozích hodnot, validaci formuláře, uložení hodnot do databáze, a metodu

`createComponentForm()`. V akci, ve které požadujeme formulář, je nutné vytvořit novou instanci požadovaného formuláře.

Všechny vytvořené formuláře dědí třídu `BaseForm`, viz 7.4.3. Metoda `createComponentForm()` v jednotlivých třídách nejprve získá továrnu pro vytvoření formuláře a zavolá nad ní metodu `createBuilder()` pro získání builderu formuláře. Formulář je získán zavoláním metody `getForm()` nad builderem. Do formuláře jsou následně přidávány prvky metodou `add()`.

Vykreslení formuláře je řešeno pomocí rozšíření `FormExtension`, které bylo přidáno do konfigurace Twigu. Rozšíření přidává makra do Twigu, které vykreslí formulář dle předimplementovaných šablon. Jako výchozí vzhled pro formulář byla zvolena šablona `bootstrap_3_layout.html.twig`. Získání pohledu formuláře pro jeho předání do šablony se provede metodou `createView()`. V šabloně je formulář vykreslen pomocí makra:

```
{{ form_start(form) }}
{{ form_widget(form) }}
{{ form_end(form) }}
```

Kontrola, zda byl formulář odeslán, se provádí v kontroleru dané akce. Akce obsahuje podmínku, která kontroluje, jestli formulář je odeslaný, a zároveň jestli je validní. Pokud je podmínka splněna, je formulář zpracováván. Výsledek je zobrazen ve flash message.

Celkové vytvoření formuláře je tedy následující. Jako první je vytvořena třída, která reprezentuje nový formulář. Třída dědí `BaseForm` a definuje potřebné metody. V akci, ve které se formulář zobrazuje, je vytvořena nová instance formuláře, která je následně předána do šablony. V šabloně, kde se formulář zobrazuje, je vloženo makro pro jeho zobrazení. V momentě, když je validní formulář odeslán, je splněna podmínka pro jeho zpracování. Formulář se zpracuje a uživatel je vyrozuměn o výsledku zpracování.

7.4.5 Vytváření a editace stránek

Správa stránek se nachází v administrátorské části v záložce „Pages Overwiev“. V aplikaci lze vytvořit dva typy stránek. Prvním typem je „Dynamic“, druhým typem je „Table“. V aplikaci dále existují další dva typy, a to „Homepage“, který představuje domovskou stánku, a typ „Forum“, který představuje diskuzi. Pro tyto dva poslední typy jsou vytvořeny stránky defaultně a nedají se smazat, ale jen skrýt.

Při vytváření stránek je první krok pro všechny typy společný. Je zde použit formulář `PageForm`, který obsahuje vstupní textové pole pro název, URL adresu a dvě výběrová pole (`selectbox`) pro typ stránky, a zda je stránka viditelná pro veřejnost. U formuláře se kontroluje, jestli je URL

adresa unikátní. Pokud ano, je uživatel přesměrován do dalšího kroku, který se liší podle typu stránky. V opačném případě je uživatel informován pomocí flash message.

Homepage a dynamická stránka

Nastavení pro tyto dva typy stránek je totožné. Je zde použit formulář `PageForm`, jako v předešlém kroku s tím rozdílem, že je navíc zobrazeno textové pole typu `textarea`, které implementuje CKEditor.

Data jsou uložena v tabulce `pages`.

Tabulka

Nastavení pro tento typ stránky je složitější, než v předchozím typu, protože dopředu není zcela jasný počet a název sloupců. Proto bylo nutné naimplementovat takový formulář, který bude možné měnit dynamicky. Z tohoto důvodu bylo nastavení pro tento typ stránky rozděleno do dalších tří formulářů.

Prvním formulářem je `TableSettings`. V tomto formuláři je nastavován název tabulky a počet sloupců, které tabulka obsahuje.

Druhám formulářem je `TableColumnsForm`. Počet prvků v tomto formuláři se mění dynamicky na základě počtu sloupců nadefinovaných v předešlém formuláři. Tento formulář slouží pro nastavení názvu sloupců v tabulce.

Poslední formulář `TableDataForm` slouží pro vkládání řádků neboli záznamů do tabulky. Vstupní pole formuláře se mění na základě počtu sloupců v tabulce. Ke každému vstupnímu prvku náleží paleta barev (colorPicker) pro výběr barvy pozadí buňky. ColorPicker je vytvořen pomocí typu `ColorType`, který byl nadefinován při vytváření vstupního pole ve formuláři. Tento formulář nabízí navíc možnost nadefinovat soubor, který bude možné stáhnout po kliknutí na odkaz v prvním sloupci. Soubor je možné nahrát na server, nebo se na něj odkázat.

Data jsou ukládána do databáze. Při ukládání vznikl problém, jak dynamická data uchovávat. Problém byl vyřešen tak, že do databáze se ukládá JSON, který obsahuje pole hodnot z formuláře `TableDataForm`. V procesu zpracování se nejdříve odstraní z pole data, která nejsou dynamická. V tomto případě je to soubor nebo odkaz ke stažení. Ten je uložen do databáze samostatně. Dynamická část je následně převedena do formátu JSON pomocí PHP nativní metody `json_encode()`. Při získávání hodnot zpět je pole získáno metodou `json_decode()`. Stejným způsobem jsou uložena data s názvem sloupců.

7.4.6 Pořadí stránek v menu

Správa pořadí stránek se nachází v administraci v záložce „Pages order“. Funkcionalita je implementována pomocí jQuery UI, které nabízí množství prvků pro iterakci s uživatelem. Pro tuto funkci byl implementován prvek Sortable, který funguje na principu listu s prvky, které je možné vzájemně přehazovat.

Prvek Sortable je použit v šabloně `manage.html.twig`, kterou akce kontroleru používá. Prvky listu jsou předány společně s ostatními proměnnými z akce kontroleru a následně vykresleny s třídou `list-group-item`. Prvky listu jsou tvořeny viditelnými stránkami. Viz ukázka implementace listu:

```
<ul id="sortable">
    {% for page in pages %}
        <li class="list-group-item" id={{ page.id }}>
                {{ page.title }}
        </li>
    {% endfor %}
</ul>
```

V šabloně dále existuje script pro obsluhu uložení pořadí do databáze. Skript se zavolá po stisku tlačítka „Save“. Po zavolání scriptu, jsou prvky listu načteny a odeslány k uložení do akce kontroleru `pageManageSaveAction()` pomocí technologie `AJAX`. Metoda k jednotlivým stránkám uloží jejich pořadí. Pořadí stránek je uloženo v databázi v tabulce `pages` ve sloupci `order`.

Při zobrazování záložek jsou načteny všechny viditelné stránky z databáze, seřazené vzestupně pomocí proměnné `order`.

7.4.7 Registrace a přihlášení uživatelů

Vytvoření nového účtu je možné ve veřejné části (`RegisterForm`) a v administraci (`UserForm`). Při vytváření nového účtu je nutné ve formuláři zadat emailovou adresu, která bude zároveň sloužit pro přihlášení, přezdívku a dvakrát stejné heslo. V administraci je potom možné nastavit ještě navíc typ uživatele (`Admin`, `Copywriter`, `Normal`). Po odeslání formuláře je zkontrolováno, zda je email jedinečný. Jedinečný musí být, protože se jedná o přihlašovací údaj, a nemůže být shodný s jiným účtem. Zároveň se musí obě dvě hesla shodovat, což je prevence překlepu. Heslo je hashováno pomocí nativní PHP funkce `password_hash()` se zvoleným algoritmem `PASSWORD_BCRYPT`, který představuje silný hashovací

algoritmus.

Přihlášení je realizováno pomocí emailu a uživatelem zvoleného hesla. Formulář pro přihlášení se zobrazuje po klikutí na „Sign In“ v menu, nebo při přihlašování do administrace. Po zadání emailu a hesla se zkontroluje, zda pod zvoleným emailem existuje v databázi uživatel. Pokud ano, je kontrolováno heslo. Pokud kontrola skončí neúspěchem, je uživatel uvědoměn. Heslo je kontrolováno pomocí PHP nativní funkce `password_verify()`.

7.4.8 Práce se šablonami

Každý modul má svou hlavní šablonu, ve které jsou nadefinované bloky, které umožní potomkům přepisovat jejich obsah. Pro administraci je implementována jako hlavní šablona „Glance Design Dashboard Bootstrap Responsive Web Template“ [5] a pro veřejnou část „Law: Free HTML5 Bootstrap Template for Law Firm Websites“ [26]. Šablony jsou umístěny ve svých modulech v souboru `layout.html.twig` ve složce `template`.

Jednotlivé šablony akcí jsou umístěny v souboru pojmenovaném podle jména akce, ve složce, která je pojmenována podle názvu kontroleru a umístěna ve složce `template`.

Příklad: Soubor uložený v `FrontModule\template\Page\default.html.twig` odpovídá akci `default`, která je v kontroleru `PageController`, který je uloženy v modulu `FrontModule`.

7.4.9 Databáze

Databáze je MySQL. Pro obsluhu databáze jsou vytvořeny třídy, tzv. „repositories“, které tvoří abstrakní vrstvu nad databází.

Přehled entit

Kompletní struktura databáze, přehled vazeb a přehled atributů v entitách, je zobrazena na obrázku .

pages Entita pro ukládání hodnot týkajících se všech typů stránek.

users Entita pro ukládání informací o jednotlivých uživatelích.

table Entita pro ukládání informací o tabulce. V tabulce jsou uloženy konfigurační hodnoty tabulky, tzn. stránka, na které se tabulka zobrazuje, její nadpis, počet sloupců a jména sloupců.

table_data Entita pro ukládání hodnot (buněk) v tabulce.

forum_topics Entita, která představuje téma ve fóru.

forum_thread Entita představující příspěvky k danému tématu.

global_settings Entita pro uložení nastavení webové aplikace.

Konfigurace připojení

Konfigurace připojení je nadefinována ve třídě `Connection`, která implementuje interface `ContainerAwareInterface`. Tento interface obsahuje metodu `setContainer()`, kterou je nutno naimplementovat. Metoda slouží pro umožnění vložení kontejneru závislostí do třídy, aby k němu mohla přistupovat. Třída `Connection` obsahuje metodu `connection()`, která navazuje spojení s databází. Konfigurační parametry jsou načteny z kontejneru závislostí metodou `getParametr()`, díky které je možné přistoupit ke konfiguračním proměnným. Proměnné jsou nadefinovány v souboru `parameters.yml`.

BaseRepository

Je to třída, která je postavena nad všemi repository. Definuje základní metody pro přístup do DB bez nutnosti psát celé dotazy v každé metodě. Třída obsahuje proměnnou `tableName`, která je typu `protected`. Proměnná definuje tabulku, nad kterou se budou dotazy provádět. Tuto proměnou je nutné ve všech repository nadefinovat. Třída obsahuje následující metody pro obsluhu DB:

- **findRow()** – Nalezne záznam v tabulce s konkrétním ID.
- **findBy()** – Nalezne záznamy dle zadaných kritérií.
- **fetchAll()** – Vráti všechny řádky v tabulce.
- **insert()** – Vloží záznam do tabulky.
- **update()** – Upraví hodnoty záznamu dle ID.
- **delete()** – Odstraní záznam z tabulky dle ID.

8 Testování aplikace

Po navržení a implementaci webové aplikace bylo nutné ověřit její kvalitu. Bylo nutné otestovat, zda je aplikace responzivní, a také otestovat její funkčnost. V následujících sekcích jsou popsány postupy, kterými byla webová aplikace otestována. Uvedené typy různých prohlížečů a zařízení byly testovány pomocí nástroje BrowserStack, která umožňuje se vzdáleně přihlásit na konkrétní zařízení.

8.1 Ověření funkčnosti v prohlížečích

Pro ověření funkčnosti byly vybrány prohlížeče dle statistiky oblíbenosti „10 nejoblíbenějších webových prohlížečů“ [25], kde od každého prohlížeče byl vybrán jeden představitel. Od Chrome byly testovány dva z důvodu velké oblíbenosti.

- Chrome 72
- Chrome 73
- Safari 12
- IE 11
- Firefox 65

Test probíhal ověřováním vzhledu stránky. Testovaly se různé stránky s různým obsahem a efekty. Dalším testem bylo ověření funkčnosti. Zde byla především zkontrolována funkčnost změny pořadí záložek v menu a zkouška odeslání nějakého formuláře. Testování proběhlo bez chyby.

8.2 Ověření funkčnosti a zobrazení na mobilních zařízeních

Pro ověření funkčnosti na mobilních zařízeních byly náhodně vybrány následující zařízení. U výběru se dbalo na výběr různě velkých zařízení. Předem je nutno zdůraznit, že se nepřepokládá přístup do administrace přes mobilní zařízení, ale jen do veřejné části projektu. Proto bylo testováno jen zobrazení obsahu ve veřejné části aplikace.

- Iphone 5s
- IPhone 6
- IPhone X
- Galaxy S9
- Galaxy note 3

U všech zařízení byl proveden stejný scénář, kde bylo testováno jenotlivé zobrazení všech vytvořených webových stránek. Testovala se především responsibilita všech prvků. Při testování se zjistil problém při zobrazování tabulek. Tento problém byl vyřešem pomocí pomocí třídy `table-responsive`.

8.3 Testovací scénáře

Předchozí testy byly především o kompatibilitě webové aplikace. Tyto testovací scénáře existují pro důkladné ověření funkčnosti webové aplikace. Při testování podle těchto testovacích scénářů se předpokládá informovaný uživatel.

8.3.1 Vytvoření dynamické stránky

- Přejděte do administrace.
- Zvolte záložku „Pages overview“.
- Klikněte na záložku „Add page“.
- Vyplňte formulář následujícími daty. Title => test, Type => Dynamic, Url => náhodné číslo (vyberte si náhodné číslo, toto pole musí být jedinečné), Visible => NO a klikněte na tlačítko „Next“.
- Vložte do vstupního pole Content libovolný text a uložte tlačítkem „Save“, pod formulářem.
- Přejděte do veřejné části a zkontrolujte, že se nezobrazuje vaše nová záložka.
- Jdete znovu do administrace a přejděte do editace stránky.
- Změňte hodnotu Visible na YES.

- Zkontrolujte, že se ve veřejné části zobrazuje nová záložka a po zobrazení obsahuje nadefinovaný text.

8.3.2 Vytvoření tabulky

- Přejděte do administrace.
- Zvolte záložku „Pages overview“.
- Klikněte na záložku „Add page“.
- Vyplňte formulář následujícími daty. Title => test, Type => Table, Url => náhodné číslo (vyberte si náhodné číslo, toto pole musí být jedinečné), Visible => YES a klikněte na tlačítko „Next“.
- Nastavte jméno tabulky na „test“ a počet sloupců dejte „tři“ a klikněte na tlačítko „Save“.
- Vyplňte volitelně názvy sloupců.
- Přejděte do správy záznamů v tabulce.
- Přejděte do přidávání nového záznamu, pomocí tlačítka „Add record“
- Vyplňte volitelně hodnoty pro sloupce. Odkaz na soubor a soubor nechejte prázdné. Formulář odešlete.
- Zkontrolujte ve veřejné části tabulku.

8.3.3 Vytvoření uživatele přes veřejnou část

- Ve veřejné části přejděte do registrace.
- Vyplňte všechny registrační údaje.
- Odešlete formulář.
- Odhlašte se.
- Přihlašte se.

8.3.4 Vytvoření a editace uživatele přes administraci

- Přejděte do administrace pod administrátorem.
- Jděte do správy uživatelů.
- Vyberte možnost „Add user“.
- Vyplňte volitelně formulář, typ uživatele nechte na normal a formulář odešlete.
- Odhlašte se a zkuste se pod účtem přihlásit do administrace (přístup by měl být odepřen).
- Přihlašte se zpět pod administrátorem a jděte do přehledu uživatelů.
- U vámi vytvořeného účtu přejděte do editace.
- Změňte typ uživatele na „admin“ a formulář uložte.
- Odhlašte se a zkuste se přihlásit pod novým účtem (přístup by měl být povolen).

8.3.5 Změna nastavení

- Přejděte do nastavení projektu.
- Změňte název projektu na „test“ a formulář odešlete.
- Zkontrolujte, že se změnil název projektu ve veřejné části.
- Přejděte znovu do nastavení projektu a nastavte, aby nepřihlášený uživatel nemohl přidávat příspěvky.
- Zkontrolujte ve veřejné části.
- Změňte nastavení, aby nepřihlášený uživatel mohl přidávat příspěvky.
- Zkontrolujte ve veřejné části.

9 Závěr

V rámci bakalářské práce byla realizována webová aplikace pro publikování vědeckých projektů, která byla vytvořena s použitím mikroframeworku na bázi Symfony komponent. Nejprve byly prozkoumány existující webové stránky výzkumných skupin pro nalezení společných vlastností, které by měla webová aplikace obsahovat, následně proběhl průzkum dostupných technologií pro tvorbu webových stránek. Na základě tohoto průzkumu proběhla analýza, na jejíž základě byl zvolen návrh webových stránek.

Řešení bylo implementováno pomocí PHP 5.6, protože tato verze je nainstalována na katedrálním serveru. Ovšem v práci je řešena také možná přenositelnost na očekávánou verzi PHP 7.x, což samozřejmě mělo vliv na celkovou komplexnost řešení.

Nad rámec zadání bylo řešeno vytvoření vývojového prostředí pro webovou aplikaci, kdy byla použita moderní technologie Docker. Návod, jak spustit vývojové prostředí v Dockeru, je popsáno v uživatelské příručce.

Samozřejmě pozornost byla též věnována testování celkové webové aplikace, kdy z povahy aplikace testy probíhaly podle scénářů. Byla otestována jednak responsibilita celé aplikace, která byla jedním z klíčových požadavků, a pak i kompatibilita s jednotlivými nejpoužívanějšími prohlížeči. Vlastní testování podle scénářů ověřilo pět způsobů zamýšleného použití aplikace.

V současné době je aplikace plně funkční, je naplněna reálnými daty projektu (tato data nebyla v práci vytvářena, ale byla přejata), a je připravena k nasazení na katedrální server. Tato verze je k dispozici na dočasné adrese http://students.kiv.zcu.cz/~cagy/. K nasazení na katedrální server dojde až po rutinním prověření bezpečnosti aplikace správcem katedrálního serveru.

V rámci práce byly ověřeny některé současné používané technologie, a zejména jejich spojení tak, aby výsledná aplikace tvořila funkční celek. Získané znalosti budou autorem využívány při jeho další práci.

Zadání bylo splněno v celém rozsahu.

Seznam použitých zkratek

TbUIS Testbed University Information System

UIS University Information System

SUT System Under Test

HTML Hyper Text Markup Language

WWW World Wide Web

SGML Standard Generalized Markup Language

CSS Cascading Style Sheets

PHP PHP: Hypertext Preprocessor

JS JavaScript

API Application Programming Interface

MVC Model-View-Controller

MVP Model-View-Presenter

XSS Cross-Site Scripting

LGPL GNU Lesser General Public License

JSON JavaScript Object Notation

AJAX Asynchronous JavaScript and XML

WYSIWYG What You See Is What You Get

HTTP Hypertext Transfer Protocol

CSRF Cross Site Request Forgeries

SQL Structured Query Language

MySQL My Structured Query Language

WAR Web Application Archive

YAML YAML Ain't Markup Language

URL Uniform Resource Locator

DB Databáze

PDO PHP Data Objects

SSH Secure Shell

Literatura a použité zdroje

[1] Bootstrap. *About* [online]. 2019. [cit. 2019/04/05]. Dostupné z: https://getbootstrap.com/docs/4.3/about/overview/.

[2] Bos, B. *A brief history of CSS until 2016* [online]. W3C, 2016. [cit. 2019/04/05]. Dostupné z: https://www.w3.org/Style/CSS20/history.html.

[3] CKEditor. *CKEditor 4 Guides* [online]. 2019. [cit. 2019/04/05]. Dostupné z: https://ckeditor.com/docs/ckeditor4/latest/guide/index.html.

[4] Composer. *Introduction* [online]. 2019. [cit. 2019/04/05]. Dostupné z: https://getcomposer.org/doc/00-intro.md#dependency-management.

[5] FreeHTML5. *Law: Free HTML5 Bootstrap Template for Law Firm Websites* [online]. 2019. [cit. 2019/04/05]. Dostupné z: https://freehtml5.co/law-free-html5-bootstrap-template-for-law-firm-websites/.

[6] github.com. *Software release* [online]. 2019. [cit. 2019/04/05]. Dostupné z: https://github.com/composer/composer/releases/tag/1.0.0-alpha1.

[7] KIV. *Výzkum* [online]. 2016. [cit. 2019/04/05]. Dostupné z: http://www.kiv.zcu.cz/cz/vyzkum/oblasti.html.

[8] Matyáš, J. *Aplikace s možností injekce chyb pro ověřování kvality testu* [online]. Západočeská Univerzita v Plzni, 2018. [cit. 2019/04/05]. Dostupné z: https://dspace5.zcu.cz/bitstream/11025/31808/1/Matyas_Jiri_Diplomova_prace.pdf.

[9] Meloni, J. C. *Sams Teach Yourself HTML, CSS, and JavaScript All in One*. Pearson Education, 2011. ISBN 9780672333323.

[10] Mouat, A. *Using Docker*. O'Reilly Media, Inc, 2016. ISBN 9781491915769.

[11] Peyrott, S. *A Brief History of JavaScript* [online]. 2017. [cit. 2019/04/05]. Dostupné z: https://auth0.com/blog/a-brief-history-of-javascript/.

[12] PHP. *random bytes* [online]. 2019. [cit. 2019/04/05]. Dostupné z: https://secure.php.net/manual/en/function.random-bytes.php.

[13] PHP. *History of PHP* [online]. 2019. [cit. 2019/04/05]. Dostupné z: https://www.php.net/manual/en/history.php.php.

[14] PHP. *New features* [online]. 2019. [cit. 2019/04/05]. Přehled novinek ve verzi PHP 7. Dostupné z: https://www.php.net/manual/en/migration70.new-features.php.

[15] PHP. *Supported Versions* [online]. 2019. [cit. 2019/04/05]. Přehled aktuálních verzí PHP. Dostupné z: https://www.php.net/supported-versions.php.

[16] Pitt, C. *Pro PHP MVC*. Apress, 2012. ISBN 9781430241645.

[17] Potužák, T. *Java Urban Traffic Simulator* [online]. 2006. [cit. 2019/04/05]. Dostupné z: http://www.juts.zcu.cz.

[18] smarty. *All About Smarty* [online]. 2019. [cit. 2019/04/05]. Dostupné z: https://www.smarty.net/about_smarty.

[19] smarty. *Smarty 3 Overview* [online]. 2019. [cit. 2019/04/05]. Dostupné z: https://www.smarty.net/v3_overview.

[20] Sourceforge. *Apache distribution for windows* [online]. 2019. [cit. 2019/04/05]. Dostupné z: https://sourceforge.net/projects/xampp/files/XAMPP%20Windows/.

[21] STILL. *SOFTWARE TESTING INTELLIGENT LAB* [online]. 2019. [cit. 2019/04/05]. Dostupné z: http://still.felk.cvut.cz.

[22] Symfony. *Symfony Components* [online]. 2019. [cit. 2019/04/05]. Dostupné z: https://symfony.com/components.

[23] Symfony. *Symfony Roadmap* [online]. 2019. [cit. 2019/04/05]. Dostupné z: https://symfony.com/roadmap.

[24] Twig. *Introduction* [online]. 2019. [cit. 2019/04/05]. Dostupné z: https://twig.symfony.com/doc/2.x/intro.html.

[25] W3Counter. *Web Browser Market Share* [online]. 2019. [cit. 2019/04/05]. Dostupné z: https://www.w3counter.com/globalstats.php.

[26] W3Layouts. *Glance Design Dashboard Bootstrap Responsive Web Template* [online]. 2019. [cit. 2019/04/05]. Dostupné z: https://w3layouts.com/glance-design-dashboard-bootstrap-responsive-web-template/.

A Uživatelská příručka

Prerekvizity

Pro bezchybný běh webové aplikace je nutné, aby na webovém serveru byly technologie Apache minimálně verze 2, PHP minimálně verze 5.6 a MySQL verze 5.6.30, nebo vyšší.

Spuštění vývojového prostředí

Webová aplikace byla vyvíjena v prostředí Docker. Proto je nutné mít tuto technologii nainstalovanou na svém zařízení. Pro spuštění Dockeru je nutné otevřít terminál ve složce `docker`, která se nachází v projektu webové aplikace, a zadat příkaz `docker-compose up --build`. Tímto se vytvoří a spustí kontejner, který bude obsahovat Apache, PHP, MariaDB a phpMyAdmin. První spuštění kontejneru, trvá déle. Kontejner se zastaví příkazem `docker-compose down`.

Výchozí přihlašovací jméno do phpMyAdmin a `MariaDB` je „root“ s heslem „root“. Adresa pro napojení aplikace do databáze je `host.docker.internal` na portu 3306.

Při prvním spuštění, je nutné nastavit parametr `web.url` v `parameters.yml` na lokální adresu a naimportovat databázi. Postup importu je popsán níže.

Nasazení na webový server

Pro nasazení aplikace na webový server je nutné přesunout složky `app`, `config` a `web` na webový server. Dále pokud server neobsahuje Composer, je nutné spustit příkaz `composer update` na lokálním zařízení ve složce s projektem, který stáhne knihovny třetích stran a uloží je do složky `vendor`, kterou je následně také nutné nahrát na webový server. Pokud server obsahuje Composer, je se složkami navíc přesunut soubor `composer.json`. Následně je na webovém serveru spuštěn příkaz `composer update`, který se postará o stáhnutí knihoven třetích stran.

Přesné umístění složek na serveru je různé podle nastavení Apache, na kterém je webová aplikace spuštěna. Pro nasazení na katedrální server KIV je nutné, nahrát soubory do složky `public-kiv` a přejmenovat složku `web`

na `public_html`. Složka je přejmenována, protože složka `public_html` je nastavená jako `DocumentRoot`.

Po nasazení webové aplikace je nutné nastavit parametr `web.url` v souboru `parameters.yml` na adresu serveru.

Při prvním nasazení na webový server je nutné naimportovat databázi. Postup je popsán v následující sekci.

Import databáze

Struktura databáze je uložena v souboru `databaze.sql`. Postup importu je následující:

- Přejděte do nastavení phpMyAdmin.
- Vytvořte databázi.
- Přejděte do záložky „import“.
- Vyberte soubor `databaze.sql` a proveďte import.

Po naimportování struktury databáze je nutné nadefinovat její připojení v aplikaci. Konfigurační parametry se nastavují ve složce `config` v souboru `parameters.yml`.

Obsluha webové aplikace

Veřejná část

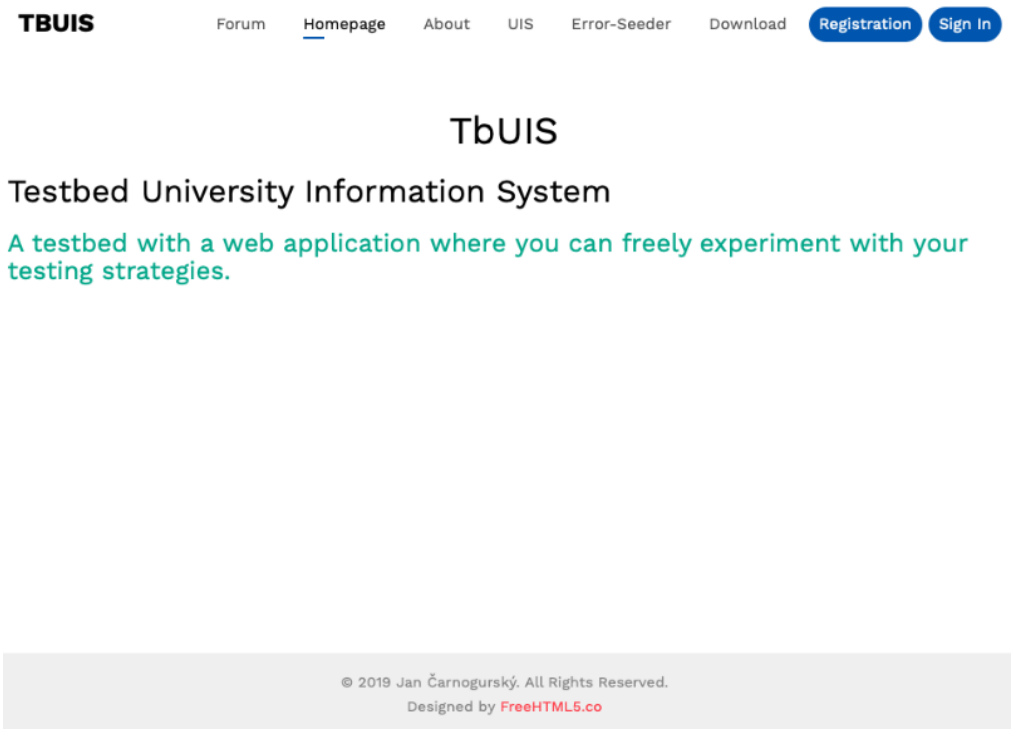
Jedná se o část webové aplikace, do které mají přístup všichni uživatelé a slouží pro předání informací o vědeckém projektu. Obsah veřejné části je možné modifikovat v administraci, viz dále. Na obrázku A.1 je zobrazena ukázka veřejné části.

Registrace

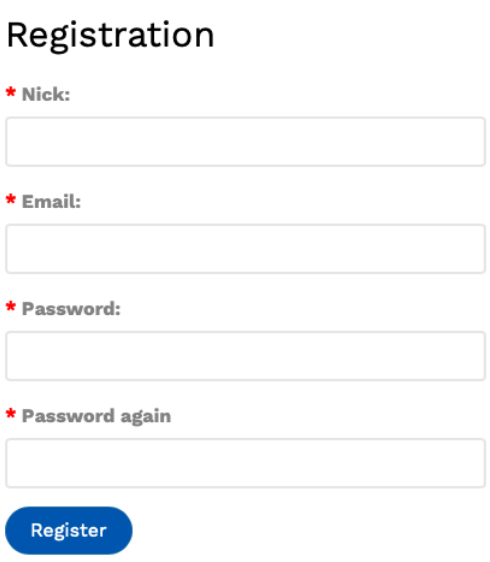
Registrace do aplikace je možná v záložce „Register“, ve veřejné části aplikace. Ukázka registračního formuláře je zobrazena na obrázku A.13.

Přihlášení

Přihlášení do aplikace je možné v záložce „Sing In“, ve veřejné části aplikace. Uživatel se přihlašuje pomocí emailu a hesla. Formulář je zobrazen na obrázku A.3



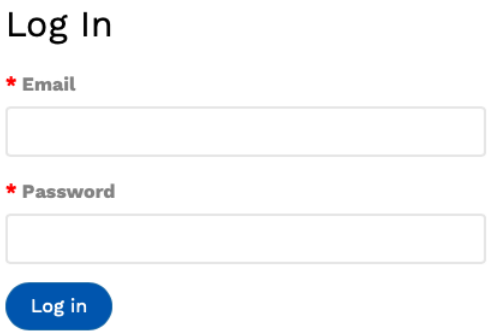
Obrázek A.1: Veřejná část aplikace



Obrázek A.2: Ukázka registračního formuláře

Diskuzní fórum

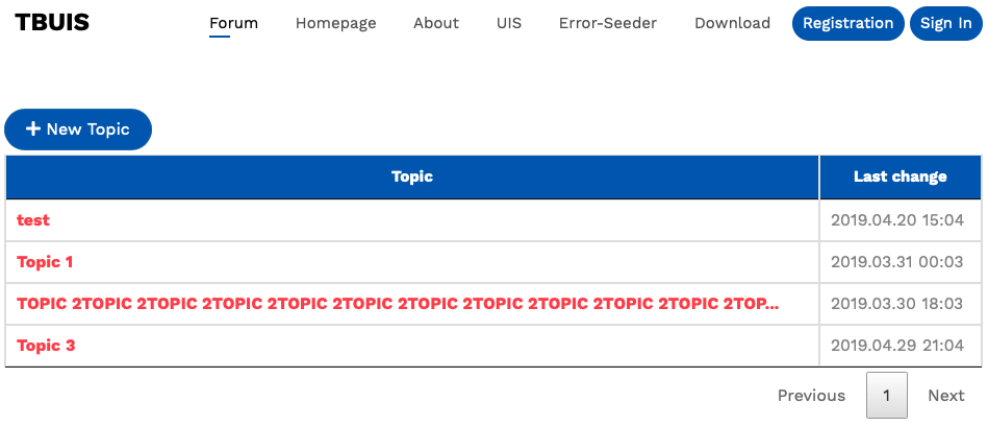
Diskuzní fórum se nachází pod záložkou „Forum“. V diskuzním fóru je možné zakládat nová témata pomocí tlačítka „New topic“ nebo přispívat



Obrázek A.3: Ukázka registračního formuláře

do diskuze pomocí vstupního pole, které se nachází pod diskuzí.

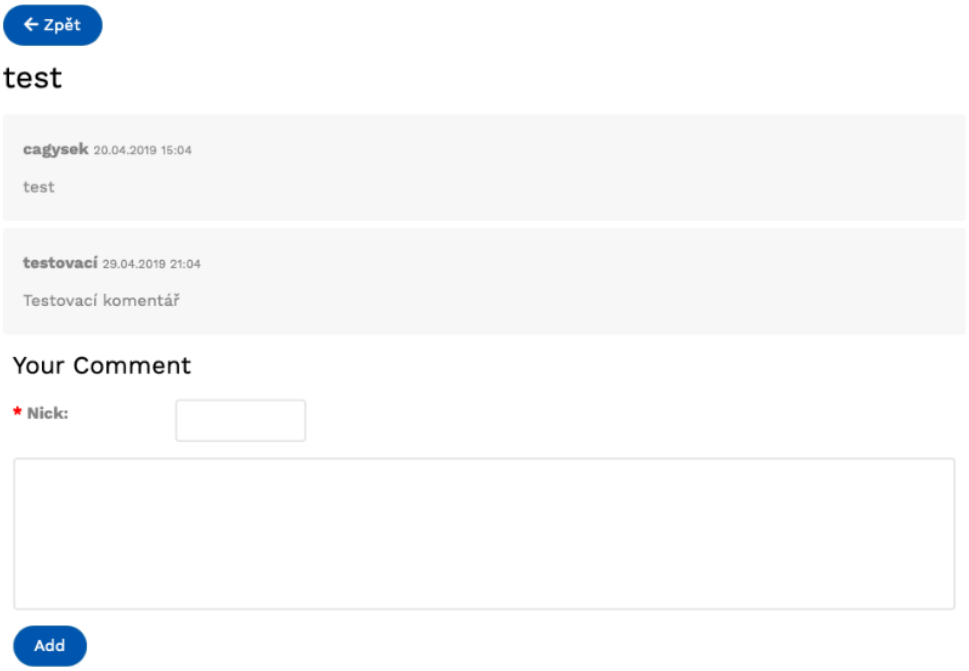
Seznam témat je seřezen podle poslední změny v tématu od nejnovějších po nejstarší. Diskuze v tématu je formou jednoho vlákna, které je seřezeno od nejstarších po nejnovější příspěvky.



Obrázek A.4: Diskuzní fórum

Přidávat nová témata a reagovat v diskuzi mohou uživatelé, na základě nastavení webové aplikace v administraci. Jednou z možností je, že přidávat příspěvky a reagovat v tématech mohou všichni přihlášení i nepřihlášení uživatelé. Druhou možností je, že mohou přidávat příspěvky a reagovat v tématech jen přihlášení uživatelé a nepříhlášení uživatelé mohou jen fórum prohlížet.

Mazání témat a příspěvků může pouze uživatel typu `Admin`.



Obrázek A.5: Vlákno fóra

Administrace

Část webové aplikace, do které mají přístup jen uživatelé typu `Admin` a `CopyWritter`. Tato část obsahuje veškerou správu projektu. Pro přístup do administrace je nutné přejít na adresu `http://<jmeno_domeny>/admin`. V případě, že uživatel není přihlášen, nebo nemá dostatečná oprávnění, je přesměrován na přihlašovací stránku administrace.

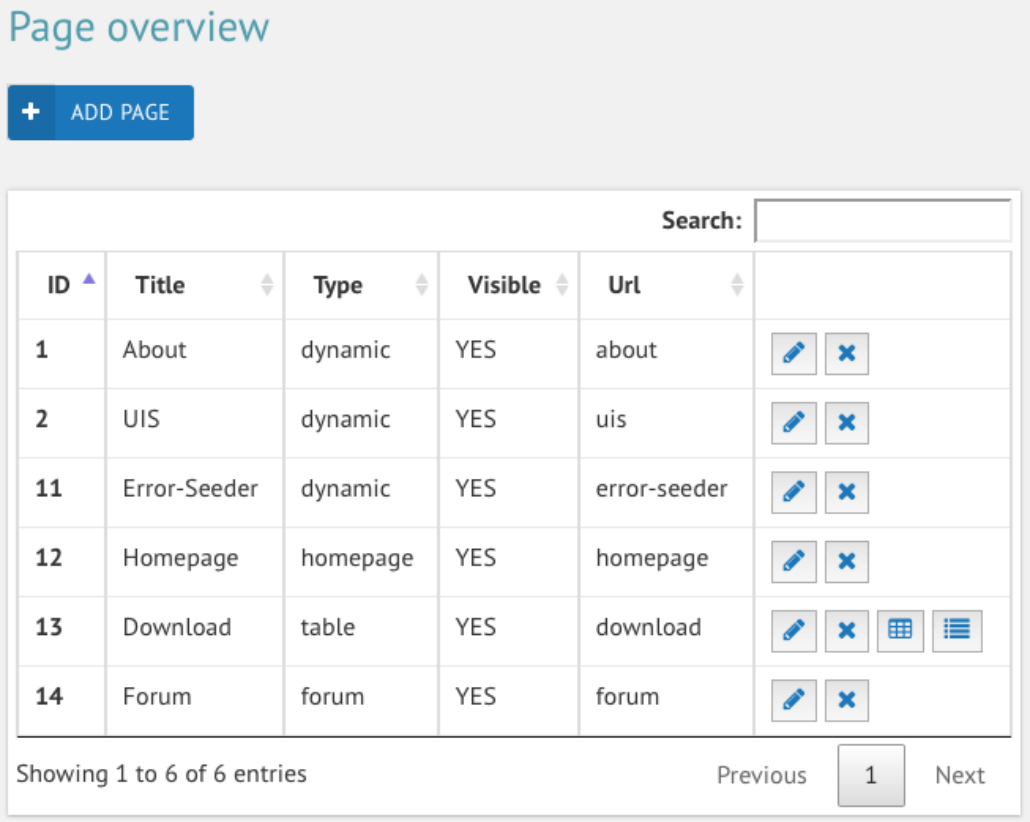
Pro prvotní přístup do administrace je připraven účet „admin@admin.cz“ s heslem „admin“.

Přehled stránek

Přehled stránek se nachází v záložce „Pages overview“ a obsahuje tabulku s přehledem již existujících stránek. Novou stránku je možné přidat tlačítkem „Add page“.

Význam ikon u řádkových akcí:

- Tužka – Editace webové stránky.
- Křížek – Odstranění webové stránky.
- Tabulka – Přechod do nastavení tabulky.



Obrázek A.6: Přehled webových stránek

- List – Přechod do správy záznamů v tabulce.

Dynamické stránky

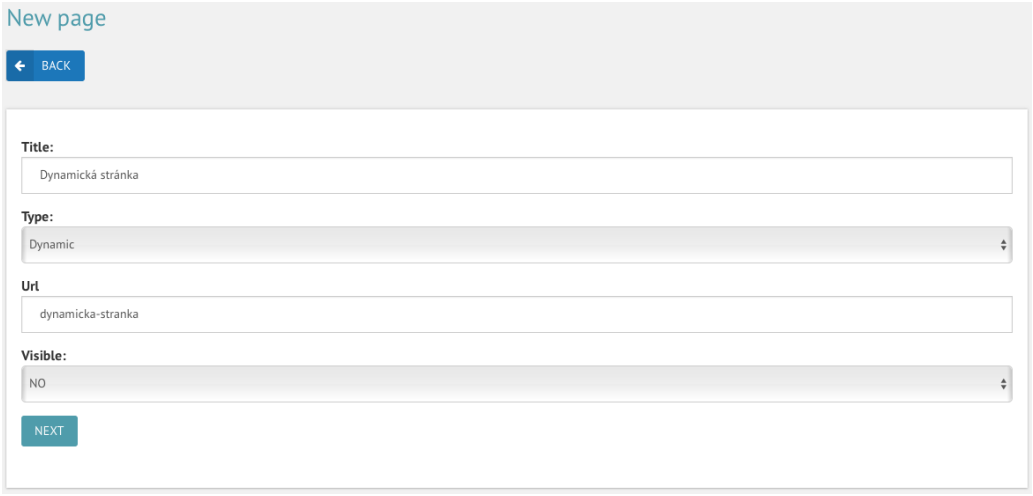
Dynamická stránka je typ stránky, která slouží pro publikování informací, jako jsou například informace o projektu nebo představení členů vědeckého týmu.

Dynamická stránka se vytvoří pomocí tlačítka „Add page“ v přehledu stránek. Po stisku je uživatel přesměrován na formulář pro vytvoření nové stránky. Ve formuláři je důležité, aby hodnota `Type` byla nastavena na hodnotu „Dynamic“.

Po odeslání formuláře tlačítkem „Next“ je zobrazeno navíc ve formuláři vstupní pole „Content“, ve kterém se definuje obsah webové stránky.

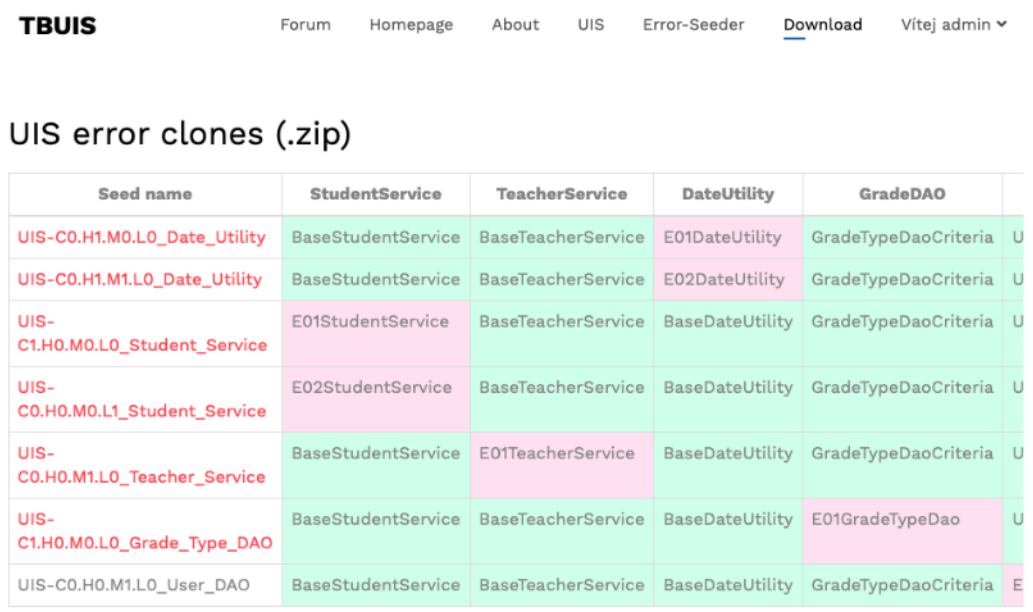
Stránky s tabulkou

Stránka s tabulkou slouží například pro zobrazení výsledků měření, nebo jako prostředek pro distribuci různých verzí softwaru. U tabulky lze



Obrázek A.7: Vytvoření dynamické stránky

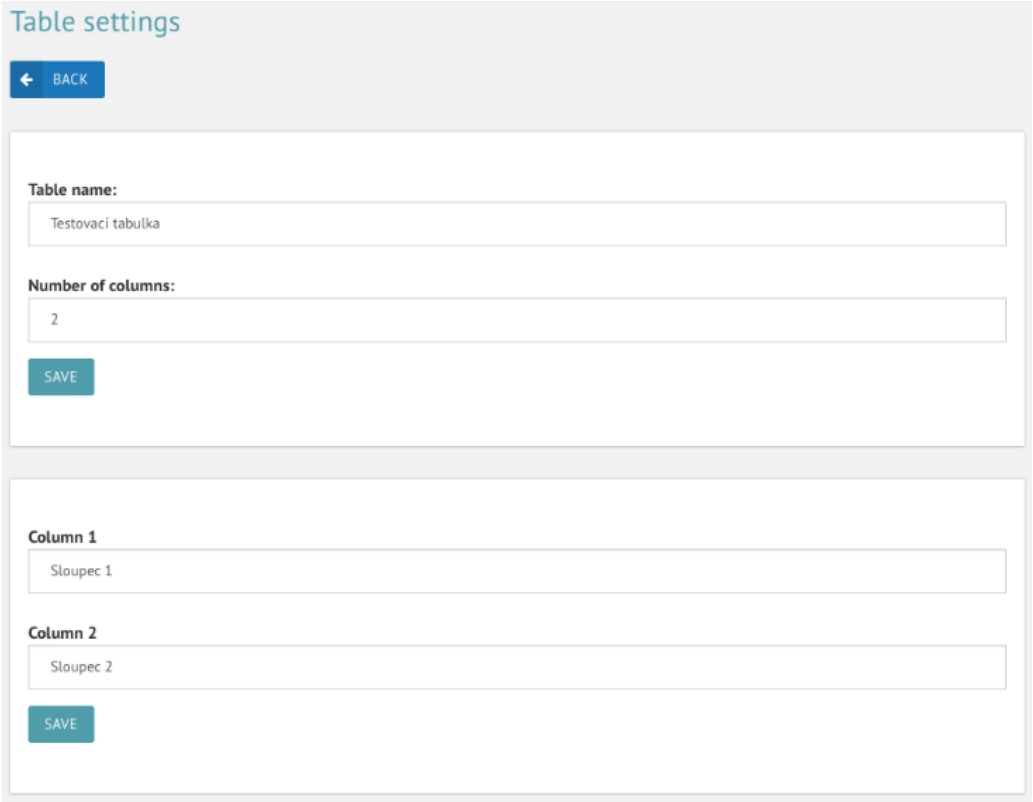
nadefinovat volitelný počet sloupců s jejich nadpisy. Následně lze do tabulky přidávat záznamy, u kterých lze zvolit jejich pozadí. U jednotlivých záznamů je možné nadefinovat odkaz na soubor, který bude umístěn na první sloupci v záznamu.



Obrázek A.8: Ukázka stránky s tabulkou

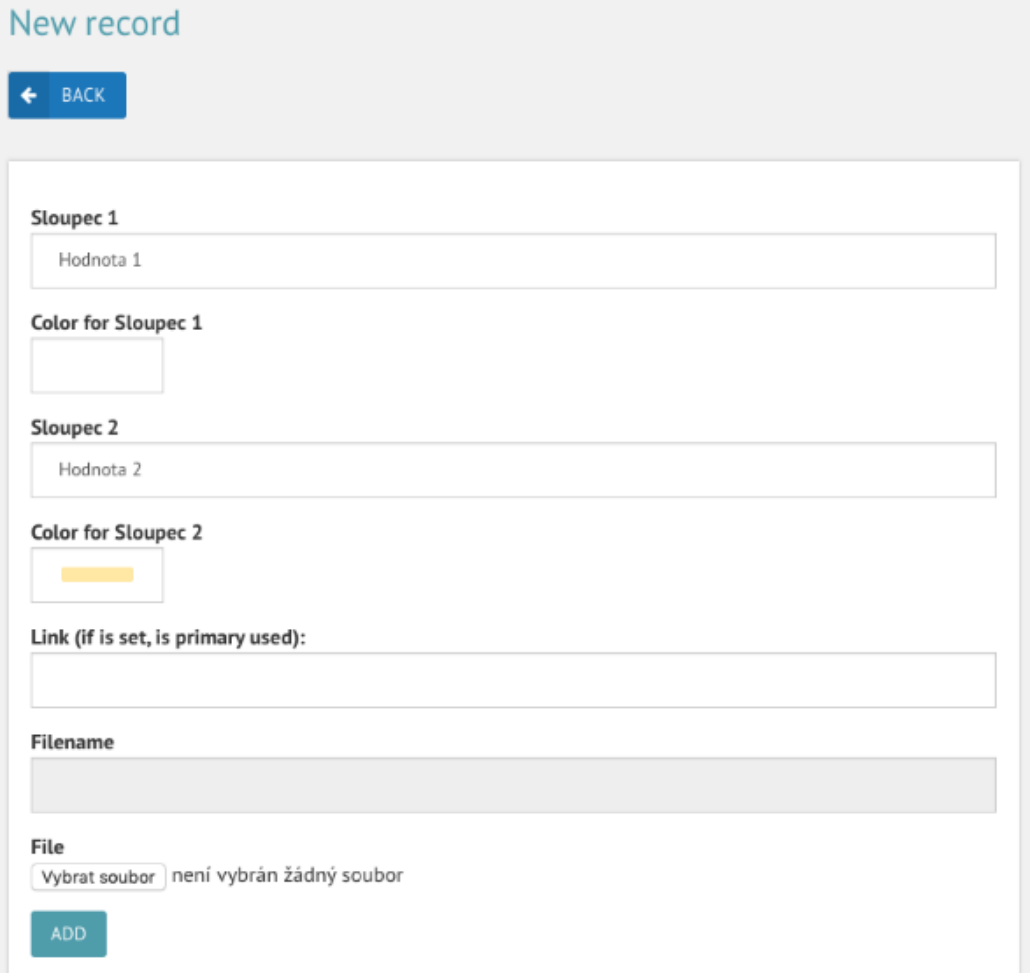
Stránky s tabulkou se jako dynamické stránky vytvářejí v přehledu stránek pomocí tlačítka „Add page“ s tím rozdílem, že jako typ stránky se vybere hodnota „Table“. Po přechodu do dalšího kroku tlačítkem „Next“ je uživatel přesměrován do nastavení tabulky. Toto nastavení obsahuje dva

formuláře, kdy jeden slouží pro specifikaci počtu sloupců a druhý pro nastavení jejich nadpisů.



Obrázek A.9: Nastavení tabulky

Pro vkládání záznamů do tabulky je nutné se vrátit do přehledu stránek, a pomocí řádkové akce přejít do správy záznamů tabulky (ikona listu). Nové záznamy se vkládají tlačítkem „Add record“. Jednotlivé záznamy v tabulce lze volitelně editovat nebo mazat.



Obrázek A.10: Formulář pro vložení záznamu do tabulky

Testovací tabulka

Sloupec 1	Sloupec 2
Hodnota 1	Hodnota 2

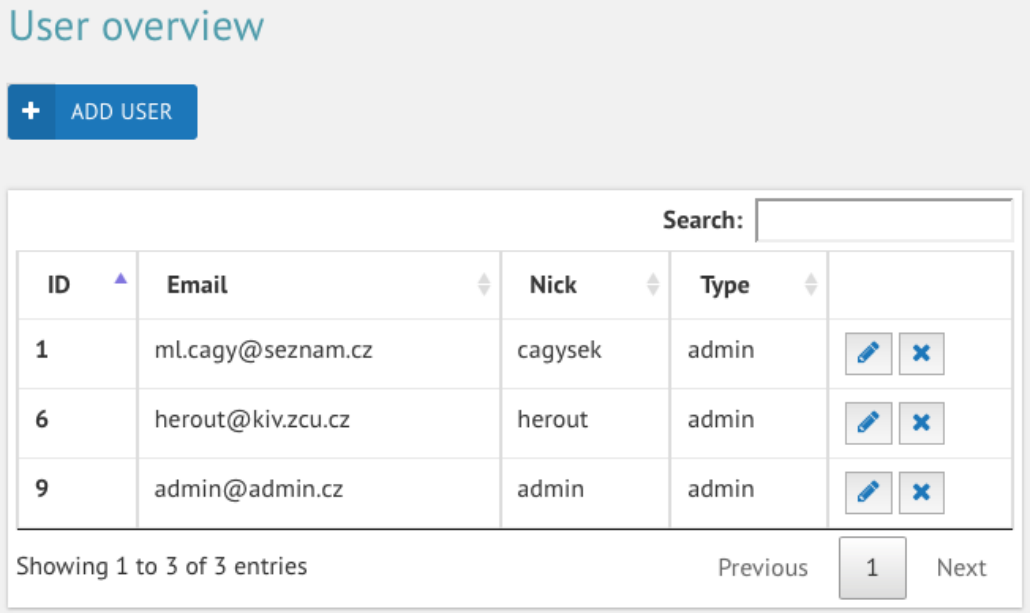
Obrázek A.11: Ukázka výsledného vzhledu tabulky

Správa uživatelů

Správa uživatelů se nachází v záložce „Users overview“. Je zde možné přidat, odebrat nebo editovat uživatele. Nový uživatel je přidán pomocí tlačítka „Add new“, které se nachází nad tabulkou s uživateli. Po stisku tlačítka je aplikace přesměrována na registrační formulář.

Editace a smazání uživatele se provede pomocí řádkové akce, která se

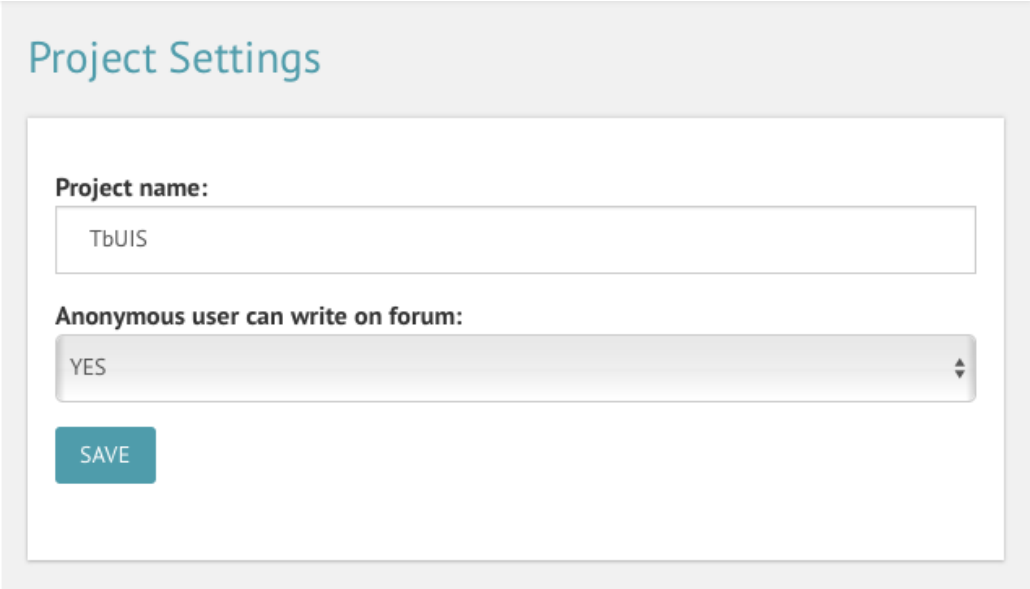
nachází v tabulce u každého záznamu. Přechod do editace je pomocí řádkové akce s ikonou tužky. Smazání uživatele se provede pomocí řádkové akce s ikonou křížku.



Obrázek A.12: Správa uživatelů

Nastavení projektu

Nastavení se nachází v záložce „Settings“. V nastavení je možné nastavit jméno projektu, a oprávnění na přidávání příspěvků neregistrovaných uživatelů do fóra.



Obrázek A.13: Nastavení projektu