Dokumentace semestrální práce:

**Webová aplikace Konference**

**(KIV/WEB, zimní semestr 2017/2018)**

Vypracoval: **Jan Havlíček, A17B0031K**

Obsah

[1. Použité technologie: 3](#_Toc504457853)

[1.1 Platforma 3](#_Toc504457854)

[1.2 GitHub 3](#_Toc504457855)

[1.3 MDBootstrap 3](#_Toc504457856)

[1.4 MySQL, PDO 3](#_Toc504457857)

[1.5 MVC 3](#_Toc504457858)

[1.6 Responzivní design 3](#_Toc504457859)

[2. Adresářová struktura aplikace 4](#_Toc504457860)

[2.1 app 4](#_Toc504457861)

[2.2 articles\_pdf 5](#_Toc504457862)

[2.3 css 5](#_Toc504457863)

[2.4 font 5](#_Toc504457864)

[2.5 img 5](#_Toc504457865)

[2.6 js 5](#_Toc504457866)

[3. Popis architektury aplikace 5](#_Toc504457867)

[3.1 Modely 5](#_Toc504457868)

[3.2 Pohledy 6](#_Toc504457869)

[3.3 Kontroléry 6](#_Toc504457870)

[4. Závěr 7](#_Toc504457871)

## Použité technologie:

### Platforma

Aplikace byla testována a psána na WAMP serveru verze **3.1.0**, PHP verze **7.1.9**, phpMyAdmin **4.7.4,** MySQL **5.1.19** a Apache **2.4.27**. Aplikace nikdy nebyla nasazena (ani to nebyl zámysl) na hosting, ať už veřejně, či nikoliv.

### GitHub

Zdrojové kódy okamžitě při vytvoření projektu byly vloženy na GitHub. Došlo k vytvoření nového účtu a založení repositáře.

Nebylo využito žádného lokálního klienta – práce s GitHubem probíhala přes webového klienta.

### MDBootstrap

Rozhodl jsem se pro využití MDBootstrap (<https://mdbootstrap.com/>). MD = Material Design používaný ve velkém počtu dnešních aplikací. Design vychází z designu Google a mdbootstrap má vcelku přehledně zpracovanou dokumentaci (včetně příkladů).

Bootstrap je použit ve všech .phtml souborech této webové aplikace (včetně ikon).

### MySQL, PDO

Datový model aplikace je realizován pomocí **MySQL 5.1.19** a **phpMyAdmin.** V databázi se nachází 3 tabulky: **user** (uživatel), **articles** (články), **reviews** (recenze). Přístup k nim je zajišťován přes PDO. Byl napsán wrapper pro SQL dotazy, následně modely **Articles**, **Reviews** a **Users**, které parametrizují příslušné dotazy.

Prozatím se nepovedlo zabránit SQL INJECTION pomocí bindingu a ‚?‘ otazníků v SQL dotazech – existuje varianta třídy **Database.php**, kde je daná funkčnost napsána, ale nepodařilo se toto zprovoznit.

### MVC

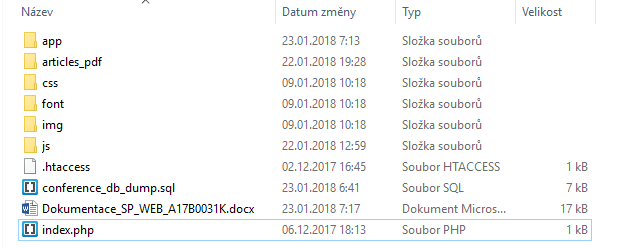
Aplikace je psána ve struktuře **Model-View-Controller**. Pokusil jsem se strukturu implementovat sám. Existuje výchozí **Controller.php** a **RouterController.php**, které se starají o zpracovávání URL adres do „hezkých“, o párování **Controllerů** a **Views**, o zobrazení základního **layout.phtml** na každé page webové aplikace.

Vzhledem k ne úplně zvládnuté (a pochopené) implementaci není možné pracovat s „ošklivými“ URL adresami.

### Responzivní design

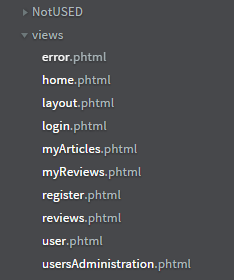
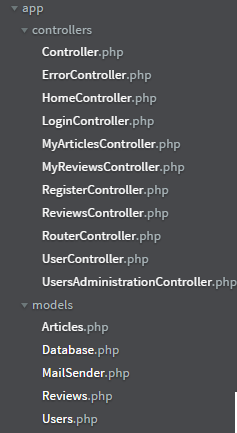
Vzhledem k využití MDBootstrap je zajištěna responzivnost designu.

## Adresářová struktura aplikace



Root adresářové struktury vypadá takto. Obsahuje index.php, .htaccess, dump SQL databáze a tuto dokumentaci.

### app



Adresář app obsahuje podadresáře: **controllers**, **models**, **NotUSED** a **views**. Controllers, models a views obsahují soubory patřících pod jednotlivé vrstvy. Složka **NotUSED** pak obsahuje historické soubory, které se ukázaly jako nepotřebné, či nežádoucí.

### articles\_pdf

Do tohoto adresáře jsou ukládány uživateli (autory) článků nahrané .pdf soubory. Jejich url adresa je poté uskladněna v databázi a návštěvník webové stránky si tyto .pdf může stáhnout.

### css

Tento adresář je určen **.css** souborům. Obsahuje pouze soubory importované společně s mdBootstrap. Nebylo potřeba modifikovat obsažené a obsáhlé stylování.

### font

Obsahuje font **Roboto** patřící k mdBootstrap.

### img

Adresář s obrázky. Původním účelem bylo ukládat uživatelské avatary, případně úvodní obrázky k jednotlivým vkládaným článkům. Momentálně tato složka však obsahuje pouze několik statických obrázků, které jsou součástí html.

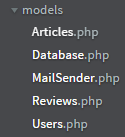
### js

V tomto adresáři jsou uloženy javascripty. Opět se v převážné většině jedná o scripty importované společně s mdBootstrap.

Pokusil jsem se o vlastní použití javascriptu, bohužel na tuto technologii nezbyl dostatek času.

## Popis architektury aplikace

### Modely

Modely tvoří část aplikace, která zajišťuje komunikaci s databází.

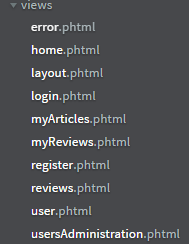
Třída **Database.php** slouží jako wrapper pro skládání základních SQL dotazů. Obsahuje metody pro zjednodušení skládání SQL

Třída **Articles.php** využívá metody třídy Database.php a poskytuje funkčnost týkající se článků, jako např. přidávání, získání všech článků, které byly akceptovány, apod.

Třída **Reviews.php** využívá metody třídy Database.php a poskytuje funkčnost týkající se recenzí článků, jako např. přidávání, úprava, apod.

Třída **Users.php** využívá metody třídy Database.php a poskytuje funkčnost týkající se uživatelů a jejich administrace. Obsahuje metody, jako „registrovat“, „přihlásit“, „getUserId($name)“, apod.

### Pohledy

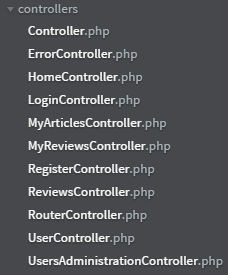
Pohledy obsahují veškeré .phtml soubory použité v aplikaci. Každý pohled má svůj obsluhující kontroler, jehož název je složen z názvu pohledu + „Controller“, s příponou .php.

Jednotlivé pohledy určují, co se bude zobrazovat na té které stránce aplikace.

**Layout.phtml** je základním pohledem. Obsahuje hlavičku i patičku stránky (která je neměnná). Jeho kontrolérem je RouterController.php. Do Layout.phtml je následně zobrazován obsah jednotlivých stránek.

**Error.phtml** je kontrolérem, který se zobrazí ve chvíli, kdy uživatel požaduje URL adresu, která neexistuje. Toto je kontrolováno pomocí MVC, takže je zajištěno, že error.phtml je zobrazen i když např. požadovaný pohled existuje, ale neexistuje jemu odpovídající kontrolér.

### Kontroléry

Kontroléry zpřístupňují logiku modelu a zajišťují její interakci s pohledy. Reagují (zpracovávají) na požadavky od uživatele pomocí php formulářů (není zde použit AJAX).

**Controller.php** je základní kontrolér, od kterého jsou děděny všechny ostatní kontrolery. Obsahuje abstraktní funkci proces, zajišťuje přesměrovávání na správnou URL.

**RouterController.php** se stará o překlad URL adres do „hezkých“.

## Závěr

Tato semestrální práce mi hodně dala. Získal jsem vzhled do procesu vytváření webových aplikací a většinu standardně používaných technologií jsem si vyzkoušel.

Bohužel jsem nedokázal odhadnout časovou náročnost práce a tak jsem si zpočátku nasadil vyšší cíle funkčního webu (dle standardního zadání konference), které mne brzdily a ve výsledku jsem nedokázal uvést do provozu ani některé elementární funkčnosti.

Překvapila mne nejednotnost dokumentace jednotlivých technologií a zároveň jsem až pozdě pochopil, že (dle mého názoru vzhledem k rozšířenosti webových technologií) ne každý pozitivně hodnocený příklad, či tutoriál na internetu platí a je funkční.

Kdybych něco mohl změnit, byl by to přístup k práci. Začal jsem rovnou psát semestrální práci, jednotlivé přednášky a funkce jsem se učil „za pochodu“. Tím pádem v práci zůstal „nepořádek“ a zároveň jsem před začátkem neměl přehled o časové náročnosti. Absolvováním dílčích cvičení a poté použití těchto znalostí v práci by pravděpodobně ušetřilo čas.