

Střední průmyslová škola elektrotechniky a informatiky,
Ostrava

19. SPSEI BookMarket

Maturitní práce

Autor práce: Peter Butora

Vedoucí práce: Mgr. Antonín Kačerovský

Třída: I4C

Školní rok: 2022/2023

Zadání maturitní práce

Jméno a příjmení žáka: Peter Butora
Třída: I4C
Školní rok: 2022/2023
Obor vzdělání: 18-20-M/01 Informační technologie
Vedoucí maturitní práce: Mgr. Antonín Kačerovský

Téma: 19. SPSEI BookMarket - online burza učebnic

Zadání maturitní práce

Navrhněte a realizujte webovou aplikaci, která nabídne žákům školy online prostředí, v němž budou moci nabízet a poptávat knihy – zejména učebnice – a další učební materiály formou klasického prodeje (za prodávajícím stanovenou cenu), nebo formou dražby (aukce).

Výsledkem maturitní práce bude příslušná webová aplikace a písemná dokumentace k vytvořené aplikaci.

Způsob zpracování a pokyny k obsahu a rozsahu maturitní práce:

Webová aplikace bude nabízet jednotlivým rolím uživatelů následující možnosti:

- Nepřihlášený uživatel
 - Prohlížet si nabídky, které vytvořili přihlášení uživatelé
 - Vytvořit si uživatelský účet (registrovat se)
- Přihlášený uživatel
 - Upravovat údaje vlastního uživatelského účtu
 - Přidat si nabídku mezi oblíbené, odebrat nabídku z oblíbených
 - Komunikovat s ostatními uživateli přes interní chat
 - Přihazovat do aukcí
 - Zveřejnit novou nabídku
 - Zahájit novou aukci
- Administrátor
 - Spravovat údaje
 - knih (učebnic) – název, autor, ISBN, popis, běžná prodejní cena, (v případě učebnice, pro který ročník a obor je určena a zda je povinná nebo doporučená)
 - nabídek – předmět nabídky (kniha, učební materiál), cena, autor nabídky (nabízející), stav nabídky, kupující
 - aukcí – předmět aukce (kniha, učební materiál), výchozí cena, autor aukce (nabízející), stav aukce, kupující
 - tříd
 - uživatelů

Webová aplikace bude postavena na jazycích a technologiích: HTML, CSS, JavaScript, PHP a MySQL. Zdrojové kódy budou validní dle použitých standardů jednotlivých jazyků.

Uživatelské rozhraní webové aplikace bude splňovat následující požadavky:

- originální vzhled
- responzivní zobrazení
- přehledná a intuitivní struktura
- dodržení zásad přístupnosti webu

Písemná dokumentace, v rozsahu 8-12 normostran (normostrana = 1 800 znaků, tedy asi 30 řádků na stránku a 60 úhozů na řádek), bude zpracována dle požadavků stanovených v oficiálním dokumentu (*Pokyny pro zpracování maturitních prací 2022/2023*) umístěném na webových stránkách školy.

Povinné části písemné dokumentace:

- Titulní strana, Anotace, Prohlášení, Obsah, Úvod, Závěr, Zdroje (dle výše uvedených *Pokynů*)
- Postup tvorby webové aplikace
- Schéma a popis databáze
- Popis struktury aplikace, vysvětlení činnosti nejdůležitějších funkčních prvků (včetně popisu kódu)
- Popis uživatelského rozhraní aplikace (HTML elementy klíčových částí, použité CSS šablony)
- Manuál pro uživatele (popis a ovládání webové aplikace) /může být ve formě přílohy/
- Pokyny pro instalaci a konfiguraci aplikace /může být ve formě přílohy/

Vytvořte archiv s názvem *Třída_Příjmení_ČísloPráce.zip*, který bude obsahovat soubory aplikace a veškerou dokumentaci v elektronické formě a odevzdejte ho přes aplikaci Google Classroom. Tištěná verze práce bude odevzdána dle epidemiologické situace.

Archiv (*Třída_Příjmení_ČísloPráce.zip*) bude obsahovat:

- zdrojové kódy vytvořené webové aplikace,
- základní databázový import (*.sql), pokud je nutný ke zprovoznění aplikace,
- pokyny pro instalaci a konfiguraci aplikace (*.pdf/*.html/*.txt),
- manuál pro uživatele (*.pdf/*.html/*.txt) a
- elektronickou verzi písemné dokumentace (*.docx/*.odt + *.pdf).

Kritéria hodnocení maturitní práce

Hodnocení obsahu vedoucím maturitní práce

20 bodů – Obsahová stránka práce – stanovení cílů zadání, úroveň a úplnost praktického zpracování (výrobek, aplikace, program, ...), splnění požadavků zadání, postup práce, výzkum, vývoj, metodika, úroveň zpracování, odborná terminologie, práce s literaturou, naplněnost cílů, analýza dat, formulace závěru.

10 bodů – Formální stránka práce – úroveň zpracování dokumentace (úplnost, logické členění, uspořádání), rozsah práce, formální a grafická stránka práce, dodržování typografických pravidel a respektování norem, citace, ...

10 bodů – průběžné hodnocení – aktivita studenta, tvůrčí přístup, schopnost komunikace, soustavnost práce.

Hodnocení obsahu oponentem maturitní práce

20 bodů – Obsahová stránka práce – stanovení cílů zadání, úroveň a úplnost praktického zpracování (výrobek, aplikace, program, ...), splnění požadavků zadání, postup práce, výzkum, vývoj, metodika, úroveň zpracování, odborná terminologie, práce s literaturou, naplněnost cílů, analýza dat, formulace závěru.

10 bodů – Formální stránka práce – úroveň zpracování dokumentace (úplnost, logické členění, uspořádání), rozsah práce, formální a grafická stránka práce, dodržování typografických pravidel a respektování norem, citace, ...

Hodnocení obhajoby vedoucím maturitní práce

15 bodů – Obsahová část prezentace – vysvětlení cíle, jeho realizace a zdůvodnění postupu, vlastní prezentace práce, formální zpracování prezentace, orientace v problematice, věcné a jasné odpovědi při diskuzi.

Hodnocení obhajoby oponentem maturitní práce

15 bodů – Obsahová část prezentace – vysvětlení cíle, jeho realizace a zdůvodnění postupu, vlastní prezentace práce, formální zpracování prezentace, orientace v problematice, věcné a jasné odpovědi při diskuzi.

U obhajoby práce musí žák získat minimálně 10 bodů, jinak je práce hodnocena známkou „nedostatečný“.

Prokáže-li se, že žák v práci použil text jiného autora a tento neoznačil jako citaci (práci, nebo její část opsal), bude práce považována za plagiát, bude posuzována jako neobhajitelná a hodnocena známkou „nedostatečný“.

Celkové hodnocení maturitní práce se stanoví na základě výsledných bodů

100–85	výborný
84–67	chvalitebný
66–49	dobrý
48–34	dostatečný
33–0	nedostatečný

Délka obhajoby maturitní práce před zkušební maturitní komisí je 15 minut.

U prací vypracovaných týmově

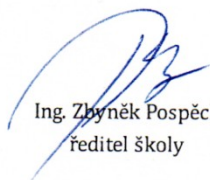
- každý člen týmu má přesně konkretizovaný svůj podíl na daném úkolu
- každý člen týmu se musí podílet na práci ve všech sledovaných parametrech, míra podílu může být různá.

Počet vyhotovení maturitní práce: 1 vyhotovení

Termín zadání maturitní práce: 21. prosinec 2022

Termín odevzdání maturitní práce: 3. duben 2023

Ostrava 8. prosinec 2022



Ing. Zbyněk Pospěch
ředitel školy

Čestné prohlášení autora práce

Prohlašuji, že předložená práce je mým původním dílem, které jsem vypracoval samostatně. Veškerou literaturu a další zdroje, z nichž jsem při zpracování čerpal, v práci řádně cituji a jsou uvedeny v seznamu použité literatury a zdrojů informací.

V Ostravě 2. 4. 2023

podpis:

Anotace

Cílem práce je navrhnout a realizovat webovou aplikaci pro nákup, dražbu a prodej knih (učebnic) a učebních materiálů.

Obsa

h

1.	Úvod.....	12
2.	Postup zpracování maturitní práce.....	13
3.	Databáze.....	14
3.1	Tabulka: api_keys.....	14
3.2	Tabulka: auctions.....	15
3.3	Tabulka: banned_ips.....	15
3.4	Tabulka: books.....	15
3.5	Tabulka: categories.....	15
3.6	Tabulka: chats.....	15
3.7	Tabulka: classes.....	16
3.8	Tabulka: class_room.....	16
3.9	Tabulka: majors.....	16
3.10	Tabulka: messages.....	16
3.11	Tabulka: notebooks.....	16
3.12	Tabulka: Notifications.....	16
3.13	Tabulka: nabídky.....	17
3.14	Tabulka: tokens.....	17
3.15	Tabulka: users.....	17
4.	Jádro aplikace.....	18
4.1	Database.php.....	18
4.2	FileRules.php.....	18
4.3	InputRules.php.....	18
4.4	Filter.php.....	18
4.5	FormGenerator.php.....	18
4.6	HelperFunctions.php.....	20
4.7	Mail.php.....	21
4.8	Pagination.php.....	21
4.9	QueryResult.php.....	22
4.10	Router.php.....	23
4.11	Validator.php.....	24
5.	Modely.....	25
5.1	Auction.php.....	26
5.2	Offer.php.....	26
6.	Kontroléry (+ pohledy).....	30
6.1	AccountController.php.....	30

6.1.1	Metoda my_account.....	30
6.1.2	Metoda tab_my_offers.....	31
6.1.3	Metoda tab_my_auctions.....	32
6.2	AdminController.php.....	32
6.2.1	Metoda user_maintenance.....	33
6.2.2	Metoda offer_maintenance.....	33
6.2.3	Metoda auction_maintenance.....	33
6.2.4	Metoda class_maintenance.....	33
6.2.5	Metoda book_maintenance.....	33
6.2.6	Metoda notebook_maintenance.....	34
6.2.7	Metoda cr_maintenance.....	34
6.2.8	Metoda banned_ip_maintenance.....	34
6.2.9	Metoda api_key_maintenance.....	34
6.2.10	Metody s příponou _edit.....	34
6.3	AjaxController.php.....	35
6.4	ApiController.php.....	35
6.4.1	Metoda offers (URL: /api/nabídky).....	35
6.4.2	Metoda classes (URL: /api/tridy).....	36
6.5	AuctionController.php.....	36
6.5.1	Metoda rise_price.....	36
6.5.2	Metoda can_user_bid.....	37
6.6	AuthController.php.....	37
6.6.1	Metoda login.....	37
6.7	BaseController.php.....	38
6.7.1	Metoda render.....	38
6.7.2	Metoda get_notifications.....	38
6.8	ChatController.php.....	39
6.8.1	Metoda create_new_chat.....	39
6.9	CronController.php.....	39
6.10	ErrorController.php.....	39
6.11	OfferController.php.....	40
6.12	PageController.php.....	41
6.13	WishlistController.php.....	41
6.13.1	Metoda add_or_delete.....	41
7.	Server běžící na protokolu WebSocket.....	41
7.1	ChannelServer.php.....	42

7.2	SocketIO Server.php.....	43
8.	Knihovny.....	44
8.1	Správa knihoven.....	44
8.2	Seznam PHP knihoven.....	44
8.3	Seznam JS knihoven.....	44
8.4	Seznam CSS knihoven.....	45
8.5	Šablony.....	45
9.	Návod.....	46
9.1	Instalace / inicializace aplikace pro Linux.....	46
9.2	Používání aplikace pro uživatele.....	49
9.2.1	Registrace.....	49
9.2.2	Zveřejnění nabídky.....	49
9.2.3	Nakupování nabídek.....	50
9.2.4	Zapojení v aukcích.....	51
9.2.5	Oblíbené nabídky.....	51
9.2.6	Správa účtu.....	51
9.3	Používání aplikace pro administrátora.....	52
9.3.1	Správa nabídek.....	52
9.3.2	Správa aukcí.....	53
9.3.3	Správa uživatelů.....	53
10.	Závěr.....	54
11.	Použité zdroje.....	55

Seznam obrázků

Obrázek 1 Schéma databáze.....	15
Obrázek 2 Metoda "join_sql"	19
Obrázek 3 Metoda "prefill_form_by_id" - zjištění primárních klíčů tabulek.....	20
Obrázek 4 Metoda "prefill_form_by_id" - vytvoření výsledného SQL dotazu.....	20
Obrázek 5 Metoda "prefill_form_by_id" - přednastavení „checkboxu“ a „selectu“	21
Obrázek 6 Metoda "getClientIp"	21
Obrázek 7 Třída "Pagination"	22
Obrázek 8 Třída "QueryResult"	23
Obrázek 9 Třída "Router"	24
Obrázek 10 Cesty ("routes").....	24
Obrázek 11 Ukázka zápisu validačních pravidel.....	25
Obrázek 12 Spuštění validace.....	25
Obrázek 13 Ukázka metody „run“	25
Obrázek 14 Ukázka modelu „BannedIp“ k databázové tabulce „banned_ips“	26
Obrázek 15 Ukázka metody „get_all_with_filters“	27
Obrázek 16 Metoda „get_won_auctions_from_user“	27
Obrázek 17 Metoda „get_current_state“	28
Obrázek 18 Atribut „allowed_filters“ třídy „Offer.php“	28
Obrázek 19 Metoda „url_has_fitler“	28
Obrázek 20 Třída „Offer.php“	29
Obrázek 21 Metoda „read_filters“	30
Obrázek 22 Metoda „my_account“	31
Obrázek 23 Pohled „my_account.php“	32
Obrázek 24 Metoda „tab_my_offers“	32
Obrázek 25 Záložka „Moje nabídky“	32
Obrázek 26 Metoda „tab_my_won_auctions“	33
Obrázek 27 Pohled „dashboard.php“	33
Obrázek 28 Metoda „user_maintenance“	34
Obrázek 29 Pohled „user_maintenance.php“	34
Obrázek 30 Metody s příponou „_edit“	35
Obrázek 31 Ukázka vygenerovaného formuláře.....	36
Obrázek 32 Metoda „process_list“	36
Obrázek 33 Ukázka URL adresy.....	36
Obrázek 34 Metoda „offers“	37
Obrázek 35 Metoda „rise_price“	37
Obrázek 36 Metoda „can_user_bid“	38
Obrázek 37 Atributy třídy „AuthController“	38
Obrázek 38 Metoda „login“	38
Obrázek 39 Pohled „login.php“	39
Obrázek 40 Metoda „render“	39
Obrázek 41 Metoda „get_notifications“	39
Obrázek 42 Metoda „create_new_chat“	40

Obrázek 43 Metoda „send_message“	40
Obrázek 44 Ukázka chybové hlášky.....	41
Obrázek 45 Validace na základě kategorie nabídky.....	41
Obrázek 46 Validace pro nabídky typu aukce.....	41
Obrázek 47 Validace pro nabídky s pevnou cenou.....	42
Obrázek 48 Metoda „add_or_delete“	42
Obrázek 49 Skript „ChannelServer “	43
Obrázek 50 Skript „SocketIOServer “	44
Obrázek 51 „Composer.json“	45
Obrázek 52 Stránka přihlášení.....	50
Obrázek 53 Zveřejnění nabídky.....	50
Obrázek 54 Kontaktování skrze „chat“	51
Obrázek 55 Kontaktování skrze formuláře pro „chat“	51
Obrázek 56 Kontaktování skrze emailu.....	51
Obrázek 57 Ukázka příhozu do aukce.....	52
Obrázek 58 Zpráva při výhře aukce.....	52
Obrázek 59 Oblíbené nabídky.....	52
Obrázek 60 Patička webu.....	53
Obrázek 61 Správa nabídek.....	53
Obrázek 62 Zobrazení obrázků.....	54
Obrázek 63 Správa aukcí.....	54
Obrázek 64 Správa uživatelů.....	54

1. Úvod

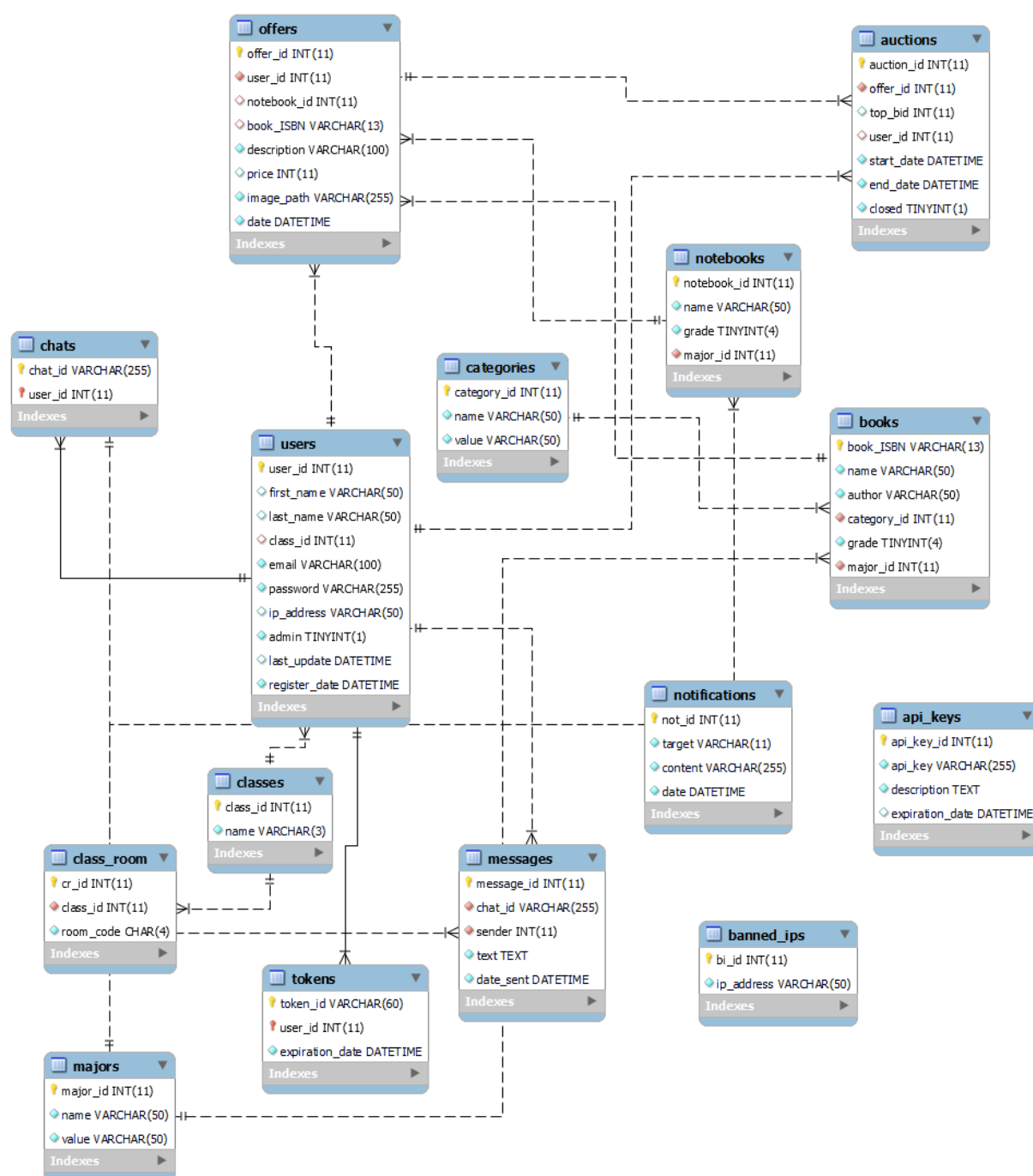
Cílem mé práce bylo vytvořit webovou aplikaci pro prodej učebnic, sešitů a jiných učebních materiálů. Mimo prodej má aplikace disponovat také možností dražení těchto předmětů. Uživatel aplikace bude mít možnost si vytvořit účet, čímž se mu otevře možnost si svůj účet upravit, zveřejnit nabídku, zapojovat se do aukcí a taky kontaktovat ostatní prodejce skrze *chat*. Součástí aplikace bude administrátorské rozhraní dostupné pro uživatele s administrátorskými oprávněními. Zde bude správa zveřejněných nabídek, aukcí a dalších potřebných objektů dle zadání maturitní práce. Toto téma jsem si zvolil, jelikož vím, že ne každý může mít čas se osobně účastnit burz učebnic ve škole. Očekávám, že tato aplikace může přinést změnu v pořádání školní burzy.

2. Postup zpracování maturitní práce

Má práce na tomto projektu začala tím, že jsem si promyslel požadovanou funkcionalitu aplikace a na základě ní vytvořil schéma databáze. Poté jsem začal programovat jádro aplikace v jazyce *PHP*. Na jádro aplikace jsem použil známou a běžně používanou architekturu *MVC* a dále jsem se inspiroval běžnými frameworky jako např: *Nette*, *Laravel*, *CodeIgniter*. Po vytvoření jádra aplikace jsem se zaměřil na funkcionalitu. To znamená, že jsem vytvořil příslušné modely a kontroléry. Pohledy jsem zatím po vzhledové stránce neřešil a sloužily pouze pro zobrazení dat. Jakmile byla funkcionalita hotová, začal jsem se stylováním a designem. V dalším kroku jsem vytvořil administrátorské rozhraní pro správu. A nakonec jsem napojil externí knihovnu pro práci s protokolem *WebSocket* a zajistil interní komunikaci a dražbu učebních materiálů v reálném čase.

3. Databáze

Pro svoji aplikaci jsem zvolil relační databázový systém *MySQL*. Vytvořil jsem potřebné tabulky a model jsem vygeneroval pomocí programu *MySQL Workbench*.



Obrázek 1 Schéma databáze

3.1 Tabulka: api_keys

api_key_id: Jedinečný identifikátor každého klíče *API*

api_key: Řetězec znaků, který slouží jako klíč *API*

description: Stručný popis účelu klíče *API*

expiration_date: Nepovinné pole, které určuje datum vypršení platnosti klíče *API*

3.2 Tabulka: auctions

auction_id: Jedinečný identifikátor každé aukce

offer_id: Identifikátor nabídky, se kterou je aukce spojena

top_bid: Nejvyšší nabídka podaná v aukci

user_id: Identifikátor uživatele, který v aukci učinil nejvyšší nabídku

start_date: Datum a čas zahájení aukce

end_date: Datum a čas ukončení aukce

closed: označuje, zda byla aukce ukončena, nebo ne

3.3 Tabulka: banned_ips

bi_id: Jedinečný identifikátor každé zakázané *IP adresy*

ip_address: *IP adresa*, která byla zakázána

3.4 Tabulka: books

book_ISBN: Jedinečné ISBN každé knihy

name: Název knihy

autor: Jméno autora

category_id: Identifikátor kategorie, do které kniha patří

grade: Identifikátor třídy pro kterou je kniha určena

major_id: Identifikátor oboru, ke kterému se kniha vztahuje

3.5 Tabulka: categories

category_id: Jedinečný identifikátor každé kategorie

name: Název kategorie

value: Hodnota spojená s kategorií

3.6 Tabulka: chats

chat_id: Jedinečný identifikátor každého *chatu*

user_id: Identifikátor uživatele, který se účastní *chatu*

3.7 Tabulka: classes

class_id: Jedinečný identifikátor každé třídy

name: Název třídy

3.8 Tabulka: class_room

cr_id: Jedinečný identifikátor pro každou třídu

class_id: Identifikátor třídy, ke které je místnost přiřazena

room_code: Kód učebny

3.9 Tabulka: majors

major_id: Jedinečný identifikátor každého oboru

name: Název oboru

value: Hodnota spojená s oborem

3.10 Tabulka: messages

message_id: Jedinečný identifikátor každé zprávy

chat_id: Identifikátor chatu, ke kterému zpráva patří

sender: Identifikátor uživatele, který zprávu odeslal

text: Obsah zprávy

date_sent: Datum a čas odeslání zprávy

3.11 Tabulka: notebooks

notebook_id: Jedinečný identifikátor každého zápisníku

name: Název zápisníku

grade: Ročník pro který je sešit určený

major_id: Identifikátor oboru, pro který je sešit určený

3.12 Tabulka: Notifications

not_id: Jedinečný identifikátor každého oznámení

target: Pro koho oznámení bude (identifikátor uživatele)

obsah: Obsah oznámení

date: Datum a čas vytvoření oznámení

3.13 Tabulka: nabídky

offer_id: Jedinečný identifikátor každé nabídky

user_id: Identifikátor uživatele, který nabídku zveřejnil

notebook_id: Identifikátor sešitu spojeného s nabídkou

book_ISBN: ISBN knihy přidružené k nabídce

description: Popis nabídky

price: Cena nabídky

image_path: Cesta k obrázkům nabízené položky

date: Datum a čas zveřejnění nabídky

3.14 Tabulka: tokens

token_id: Jedinečný identifikátor tokenu

user_id: ID uživatele spojeného s tokenem

expiration_date: Datum a čas vypršení platnosti tokenu

3.15 Tabulka: users

user_id: Jedinečný identifikátor uživatele

first_name: Křestní jméno uživatele

last_name: Příjmení uživatele

class_id: ID třídy, do které uživatel patří

e-mail: E-mailová adresa uživatele

password: Heslo uživatele

ip_address: IP adresa uživatele

admin: Označuje, zda je uživatel administrátorem, nebo ne

last_update: Datum a čas poslední aktualizace uživatele

register_date: Datum a čas, kdy byl uživatel zaregistrován

4. Jádru aplikace

Celá aplikace je napsána pomocí často používané architektury MVC. Zdrojové kódy jádra aplikace nalezneme v adresáři */src/Core*. Dále bude následovat popis funkcí jednotlivých tříd.

4.1 Database.php

Tato třída je k zajištění spojení s *MySQL* databází. Mimo jiné taky obsahuje metodu k poslání zabezpečeného předpřipraveného dotazu na databázi. Zajímavostí je ale metoda *join_sql*, která umí spojit více podmínek dotazu do jedné (odzávkuje je a spojí klíčovým slovem *AND*, takže všechny podmínky musí platit zároveň)

```
/**
 * Join SQL parts (array) to one SQL statement (string)
 * @param array $sql
 * @return string joined sql
 */
public function join_sql($sql)
{
    if(!empty($sql))
    {
        return implode("", (array_map(function($value, $key, $count) {
            $s = " ";
            if($key != 0 && $key != $count-1) $s .= "AND ";
            $s .= "(";
            return $s.$value."";
        }, $sql, array_keys($sql), [count($sql)])));
    }
    return "";
}
```

Obrázek 2 Metoda "join_sql"

4.2 FileRules.php

Pomocná třída obsahující metody určené k validaci souborů.

4.3 InputRules.php

Pomocná třída obsahující metody určené k validaci vstupů od uživatele.

4.4 Filter.php

Pomocná třída s metodami pro zkontrolování, jestli má uživatel k dané části aplikace přístup.

4.5 FormGenerator.php

Třída, která umí vygenerovat formulář na základě vybraných databázových tabulek. Napsal jsem ji proto, abych si ulehčil vytváření formulářů v administrátorském rozhraní. Třída umí nejen formulář vygenerovat, ale také předvyplnit. Samozřejmostí je taky napojení hodnot z tabulek cizích klíčů. A další zmínění hodná funkce je vygenerování validačních pravidel na základě extrémů (*maxlength* atd.) atributů v tabulkách. Pro funkčnost této třídy je potřeba mít správně nastavené omezení v tabulce.

Předvyplnění formuláře funguje tak, že v *SELECTU* zvolíme všechny sloupce ze všech dostupných tabulek. Následně první tabulku napojíme klíčovým slovem *FROM* a všechny ostatní napojíme *LEFT JOINEM*. Abychom mohli napojit tabulky *LEFT JOINEM* musíme znát název atributu s primárním klíčem. K tomu použijeme *MySQL* příkaz *DESCRIBE*. Stačí si tedy pro každou tabulku z parametru konstruktoru zjistit primární klíč a výsledný příkaz uložit do pole a později toto pole spojit do jednoho řetězce.

```
public function prefill_form_by_id($record_id)
{
    $tables = $this->tables;
    $left_joins = [];
    $select = $tables[0].".*";
    if(count($tables) > 1)
    {
        $select = join(".*, ", $tables).".*";

        $tables_without_first = $tables;
        array_shift($tables_without_first);

        foreach($tables_without_first as $table)
        {
            $ts = $this->db->query("DESCRIBE " . $table)->getResultArray();
            $primary_key = array_filter($ts, function($value) {
                return $value['Key'] == "PRI";
            })[0]['Field'];

            $left_joins[] = "LEFT JOIN ".$table." USING('".$primary_key."')";
        }
    }
}
```

Obrázek 3 Metoda "prefill_form_by_id" - zjištění primárních klíčů tabulek

```
$table_structure_0 = $this->db->query("DESCRIBE " . $tables[0])->getResultArray();
$primary_keys = array_filter($table_structure_0, function($value) {
    return $value['Key'] == "PRI";
});

$sql = "SELECT ".$select."
FROM ".$tables[0].
      ".join(" . $left_joins).
      " WHERE ".$tables[0].".$primary_keys[0]['Field'] = '".$record_id."'";

$db_record = $this->db->query($sql)->getRowArray();
```

Obrázek 4 Metoda "prefill_form_by_id" - vytvoření výsledného SQL dotazu

Když už máme odpovídající záznam, včetně hodnot v napojených tabulkách, můžeme začít procházet formulář a políčkům se stejným jménem jako sloupec databáze můžeme přednastavit hodnotu. Předvyplnění formuláře nastavuje jenom *checkboxy* a *selecty* (nastavení hodnot tagu *input* a *textarea* se objevuje jinde, protože jsem hodnotu u těchto tagů bral jako jednu z vlastností)

```
foreach($this->fields as $field)
{
    $input_name = $field["attributes"]["name"];

    if(in_array($input_name, array_keys($db_record)))
    {
        $this->fields[$input_name]["attributes"]["value"] = $db_record[$input_name];
        if($field["tag"] == "select")
        {
            // Get select values only
            $select_values = array_column($this->fields[$input_name]["select_options"], "value");
            // Find option index from select, with same value as stored in db
            // If db value not NULL
            if(!empty($db_record[$input_name]))
            {
                $index = array_search($db_record[$input_name], $select_values);
                $this->fields[$input_name]["select_options"][$index]['selected'] = true;
            }
        }
        else if($field["attributes"]["type"] == "checkbox")
        {
            // Set checkbox to checked if value stored in db equals to 1
            if($db_record[$input_name] == 1)
                $this->fields[$input_name]["attributes"]["checked"] = "checked";
        }
    }
}
```

Obrázek 5 Metoda "prefill_form_by_id" - přednastavení „checkboxu“ a „selectu“

4.6 HelperFunctions.php

Obsahuje pouze pomocné metody, které s jinými třídami nesouvisí. Např. zobrazení uložení/získání zprávy do *SESSION*, nebo dalším příkladem je pomocná metoda na získání uživatelské *ip adresy*.

```
// https://stackoverflow.com/questions/3003145/how-to-get-the-client-ip-address-in-php
public static function getClientIp()
{
    if (isset($_SERVER['HTTP_CLIENT_IP']))
        $ip = $_SERVER['HTTP_CLIENT_IP'];
    else if(isset($_SERVER['HTTP_X_FORWARDED_FOR']))
        $ip = $_SERVER['HTTP_X_FORWARDED_FOR'];
    else if(isset($_SERVER['HTTP_X_FORWARDED']))
        $ip = $_SERVER['HTTP_X_FORWARDED'];
    else if(isset($_SERVER['HTTP_FORWARDED_FOR']))
        $ip = $_SERVER['HTTP_FORWARDED_FOR'];
    else if(isset($_SERVER['HTTP_FORWARDED']))
        $ip = $_SERVER['HTTP_FORWARDED'];
    else if(isset($_SERVER['REMOTE_ADDR']))
        $ip = $_SERVER['REMOTE_ADDR'];
    else
        $ip = 'UNKNOWN';
    return $ip;
}
```

Obrázek 6 Metoda "getClientIp"

4.7 Mail.php

Nadstavba pro klasické *PHP* odesílání emailů funkce mail.

4.8 Pagination.php

Třída pro rozložení záznamů z databáze na počet určených stránek pro přehlednost. Informace o tom, na jaké stránce se právě nacházíme předáváme pomocí parametru (jehož název se mimo jiné dá libovolně zvolit) v *URL adrese*. Výsledkem je skupina odkazů zobrazených jako čísla vedle sebe, podle toho, na jakou stranu odkazují.

```
public function render()
{
    $result = '';

    for($i = 0; $i < $this->page_count; $i++)
    {
        $result .= '<a href="' . $this->generate_url($i) . '" class="me-2 text-decoration-';
    }

    return $result;
}

public function get_limit_a()
{
    return $this->get_current_page() * $this->get_items_per_page();
}

public function get_limit_b()
{
    return $this->get_items_per_page();
}

public function set_items_per_page($items_per_page)
{
    $this->items_per_page = $items_per_page;
    $this->page_count = intval(ceil(($this->items_count / $this->items_per_page)));
    if($this->page_count == 0) $this->page_count = 1;
}

public function get_items_per_page()
{
    return $this->items_per_page;
}
```

Obrázek 7 Třída "Pagination"

4.9 QueryResult.php

Rozšiřující třída pro přidání metod nad *mysqli_result* objektem. Například jsem vytvořil metody pro převedení výsledku do pole, vícerozměrného asociativního pole, nebo metodu pro spočítání počtu záznamu.

```
public function getResultArray()
{
    if($this->mysqli_result)
    {
        $tmp = $this->mysqli_result->fetch_all(MYSQLI_ASSOC);
        $this->escape($tmp);
        return $tmp;
    }
    return false;
}

public function getRowArray()
{
    if($this->mysqli_result)
    {
        $result_array = $this->getResultArray();
        if(isset($result_array[0]))
            return $result_array[0];
    }
    return false;
}

public function countAll()
{
    if($this->mysqli_result)
        return count($this->getResultArray());
    return false;
}
```

Obrázek 8 Třída "QueryResult"

4.10 Router.php

Třída pracuje jako směrovač. Propojuje požadavky zadané uživatelem s daným kontrolérem a jeho metodou. Zároveň používá dříve zmíněnou třídu *Filter* pro kontrolu, jestli má uživatel právo danou stránku navštívit. Cesty (*route*y) určují, jaká URL adresa spustí, jaký kontrolér. Cesty jsou zapsány v souboru *routes.php*.

```
/*
 * Routes
 */
public function route($url)
{
    $require = "Error:page_not_found";

    if(isset($this->routes) && !empty($this->routes))
    {
        if(array_key_exists($url, $this->routes))
        {
            $require = $this->routes[$url];
        }
    }

    $this->call_controller_method($require);
}

public function call_controller_method($string)
{
    $namespace = "SpseiMarketplace\\Controllers\\";

    $arr = explode(":", $string);
    $controller = $namespace . $arr[0] . "Controller";
    $method = $arr[1];

    $co = new $controller();
    $co->$method();
}
```

Obrázek 9 Třída "Router"

```
/*
 * ROUTES
 */

$routes = [
    "" => "Page:home",
    "/" => "Page:home",
    "/domu" => "Page:home",
    "/3d" => "Page:offers_3d",
    "/zpravy" => "Chat:index",
    "/send-message" => "Chat:send_message",
    "/create-new-chat" => "Chat:create_new_chat",
    // ADMIN
    "/admin" => "Admin:dashboard",
    "/admin/panel" => "Admin:dashboard",
    "/admin/get-auctions" => "Admin:get_auctions",
```

Obrázek 10 Cesty ("routes")

4.11 Validator.php

Pro urychlení kontroly správnosti vstupu od uživatele je právě tato třída. Stačí specifikovat název vstupu ve formuláři a pravidla jaké chceme na tento vstup uplatnit. Zápis se provádí v následujícím formátu:

```
$this->validator->addMultipleRules([
    'category' => 'required|is_not_unique[categories.category_id]',
    'price_type' => 'required|in_list[pevna, aukce]',
    'description' => 'required|min_length[3]|max_length[100]',
    'photo*' => 'is_image|max_size[1024]|min_count[1]|max_count[4]'
]);
```

Obrázek 11 Ukázka zápisu validačních pravidel

Zápis validačních pravidel (nikoliv kód) je inspirován moderním frameworkem *CodeIgniter 4*.

```
if ($this->validator->run())
```

Obrázek 12 Spuštění validace

Metoda „run“ poté projde všechny vstupy a ověří na nich všechny přiřazená pravidla a to tak, že se z dříve zmíněné třídy *InputRules/FileRules* zavolá metoda se stejnojmenným názvem jako pravidlo. Po kompletním zkontrolování metoda navrátí *true*, pokud vše prošlo pravidly, jinak vrátí *false*.

```
public function run()
{
    $parsed_rules = $this->parseRules();
    $validation_successfull = true;

    if(isset($parsed_rules))
    {
        // Loop through all rules
        foreach($parsed_rules as $key => &$rule)
        {
            $input_name = $rule['input_name'];
            $input_value = null;
            if(isset($_POST[trim($rule['input_name'], '*')]))
            {
                $input_value = $_POST[trim($rule['input_name'], '*')];
                $rule_name = $rule['rule'];
                $rule_value = null;

                if(preg_match('/(.*?)\[([*])\]/', $rule['rule'], $match))
                {
                    // Parse data from rule string
                    $rule_name = $match[1];
                    $rule_value = $match[2];
                }
            }
        }
    }
}
```

Obrázek 13 Ukázka metody „run“

5. Modely

Modely v aplikaci zajišťují práci s přiřazenou databázovou tabulkou. Zpravidla platí, že pro každou databázovou tabulku je vytvořený 1 model pro práci s ní. Každý model téměř vždy obsahuje metody pro mazání, přidání, aktualizování a získání záznamů. Jelikož se modely zásadně moc neliší, budu dále popisovat jen ty zajímavější.

```
<?php
namespace SpseiMarketplace\Models;

class BannedIp extends BaseModel
{
    public function get_all()
    {
        return $this->db->query("SELECT *
                                FROM `banned_ips`")->getResultArray();
    }

    public function get_by_id($bi_id)
    {
        return $this->db->query("SELECT *
                                FROM `banned_ips`
                                WHERE bi_id = ?",
                                [$bi_id])->getRowArray();
    }

    public function delete_by_id($bi_id)
    {
        $this->db->query("DELETE FROM `banned_ips`
                        WHERE `bi_id` = ?",
                        [$bi_id]);
    }
}
```

Obrázek 14 Ukázka modelu „BannedIp“ k databázové tabulce „banned_ips“

Seznam modelů:

1. *Api.php*
2. *Auction.php*
3. *BannedIp.php*
4. *BaseModel.php*
5. *Book.php*
6. *Category.php*
7. *Chat.php*
8. *ClassRoom.php*
9. *Major.php*
10. *Message.php*
11. *Notebook.php*
12. *Notification.php*
13. *Offer.php*
14. *SchoolClass.php*
15. *User.php*

5.1 Auction.php

Model nad databázovou tabulkou *auctions*.

Metoda *get_all_with_filters* vrátí všechny záznamy které splňují uplatněné filtry. Filtrovat lze na základě hledaného textu (text je poté hledán ve všech sloupcích), nebo podle statusu aukce (zda aukce běží, skončila nebo ještě nezačala).

```
public function get_all_with_filters($start, $length, $search, $status)
{
    $base_sql = "SELECT *
                FROM `auctions`";
    $sql_where = [];

    if(isset($search) && !empty($search))
    {
        $sql_where[] = "`offer_id` LIKE '%$search%'
                    OR `top_bid` LIKE '%$search%'
                    OR `start_date` LIKE '%$search%'
                    OR `end_date` LIKE '%$search%'";
    }
}
```

Obrázek 15 Ukázka metody „get_all_with_filters“

Dále máme metodu *get_won_auctions_from_user*, která navrátí všechny aukce, který daný uživatel vyhrál.

```
public function get_won_auctions_from_user($user_id)
{
    return $this->db->query("SELECT *
                        FROM `auctions`
                        WHERE CURRENT_TIMESTAMP() > `end_date` AND user_id = ?",
                        [$user_id])->getResultArray();
}
```

Obrázek 16 Metoda „get_won_auctions_from_user“

Poslední metoda, která stojí za zmínku je metoda *get_current_state*, která navrátí aktuální status aukce.

```
public function get_current_state($auction_id)
{
    return $this->db->query("SELECT `a`.`top_bid` AS 'top_bid', `u`.`user_id` AS 'user_id',
                        FROM `auctions` `a`
                        LEFT JOIN `users` `u` ON `u`.`user_id` = `a`.`user_id`
                        WHERE `auction_id` = ?",
                        [$auction_id])->getRowArray();
}
```

Obrázek 17 Metoda „get_current_state“

5.2 Offer.php

Model nad databázovou tabulkou *offers*.

Tento model je celkem obsáhlý a řeší mimo CRUD taky filtrování a stránkování nabídek.

Atribut *allowed_filters* říká které sloupce tabulky *offers* lze filtrovat.

```

private $allowed_filters = [
    'search',
    'price',
    'category',
    'price_type',
    'major',
    'grade',
];

```

Obrázek 18 Atribut „allowed_filters“ třídy „Offer.php“

Metoda *url_has_filter* navrácí hodnotu *true* / *false* podle toho, jestli *URL adresa* obsahuje jakýkoliv z povolených parametrů k filtrování.

```

private function url_has_filter()
{
    if(isset($_GET))
    {
        foreach($_GET as $par)
        {
            if(in_array(array_search($par, $_GET), $this->allowed_filters))
                return true;
        }
    }
    return false;
}

```

Obrázek 19 Metoda „url_has_fitler“

Následují metody s předponou *set*, které zahrnou do výsledného dotazu podmínku týkající se daného parametru. Například metoda *set_categories* zahrne do výsledného dotazu podmínku, ať je kategorie v množině kategorií co jsme zadali do parametru metody.

```

public function set_search($search)
{
    $this->sql[] = "`b`.`book_ISBN` LIKE '%" . $search . "%' OR `b`.`name` LIKE '%" . $search . "%' OR `b`.`author` LI
}

public function set_price($price_type, $price)
{
    switch($price_type)
    {
        case "aukce":
            $this->sql[] = "`a`.`auction_id` IS NOT NULL";
            break;

        case "pevna":
            if(isset($price))
            {
                $price = explode(" ", $price);
                $price_min = is_numeric($price[0]) ? $price[0] : 1;
                $price_max = is_numeric($price[1]) ? $price[1] : MAX_OFFER_PRICE;
                $this->sql[] = "`price` >= ".$price_min." AND `price` <= ".$price_max;
            }
            break;

        case "zdarma":
            $this->sql[] = "`price` = 0 AND `a`.`auction_id` IS NULL";
            break;

        case "vse":
        default:
            break;
    }
}

public function set_categories($categories)
{
    $categories = implode(",", (array_map(function($value) {
        return "`$value`";
    }, $categories)));
    $sql = "`cat`.`value` IN (". $categories . ")";

    if(in_array("'sesity'", explode(",", $categories)))
        $sql .= " OR `cat`.`value` IS NULL";

    $this->sql[] = $sql;
}

public function set_major($major)
{
    if($major != 3) // All majors
        $this->sql[] = "`b`.`major_id` = ".$major." OR `nb`.`major_id` = ".$major;
}

public function set_grade($grade)
{
    if($grade != 0) // All grades
        $this->sql[] = "`b`.`grade` = ".$grade." OR `nb`.`grade` = ".$grade;
}

```

Obrázek 20 Třída „Offer.php“

Tyto metody s příponou *set* poté využíváme v metodě *read_filters*. Její funkce spočívá v přečtení parametrů z URL *adresy* a zavolání příslušných metod. Nakonec nastaví atribut *filters* na řetězec, kterým je spojení všech nastavených podmínek.

```
public function read_filters()
{
    // Filter out empty fields
    $_GET = array_filter($_GET);
    // Product filtering
    if($this->url_has_filter())
    {
        // Set search
        if(isset($_GET['search']) && !empty($_GET['search']))
        {
            $this->set_search($_GET['search']);
        }

        // Set price
        $price_type = (isset($_GET['price_type']) && !empty($_GET['price_type'])) ? $_GET['price_type'] : "vse";
        $price = (isset($_GET['price']) && !empty($_GET['price'])) ? $_GET['price'] : null;
        $this->set_price($price_type, $price);

        // Set categories
        if(isset($_GET['category']) && !empty($_GET['category']))
            $this->set_categories($_GET['category']);

        // Set major
        if(isset($_GET['major']) && !empty($_GET['major']))
            $this->set_major($_GET['major']);

        // Set grade
        if(isset($_GET['grade']) && !empty($_GET['grade']))
            $this->set_grade($_GET['grade']);

        // Prepare sql
        $this->filters = $this->db->join_sql($this->sql);
    }
}
```

Obrázek 21 Metoda „*read_filters*“

6. Kontroléry (+ pohledy)

Kontroléry používáme pro předání dat z modelů a pohledů. V pohledu poté graficky vyobrazíme požadovaná data.

6.1 AccountController.php

Tento kontrolér řeší zobrazení a správu účtu od přihlášeného uživatele. V konstruktoru této třídy probíhá získání dat o uživateli, abychom tuto akci v každé metodě (pro záložky) nemuseli provádět zvlášť a tím se v kódu opakovat.

6.1.1 Metoda my_account

Pokud jsou na server poslána data, proběhne aktualizace uživatelského profilu. Pokud je v URL adrese uveden parametr *delete* jehož hodnotou je identifikační číslo nabídky, proběhne vymazání této nabídky (pokud nabídka uživateli patří).

```
public function my_account()
{
    $account = $this->users_model->get_by_id($SESSION['user_data']['user_id']);

    if($_POST)
    {
        if(date("Y-m-d", strtotime($account['last_update'])) < date("Y-m-d", time()))
        {
            $this->validator->addMultipleRules([
                'first_name' => 'permit_empty|min_length[2]|max_length[50]',
                'last_name' => 'permit_empty|min_length[2]|max_length[50]',
                'class' => 'permit_empty|is_not_unique[classes.class_id]',
            ]);
            if ($this->validator->run())
            {
                $this->db->query("UPDATE `users` SET `first_name` = ?, `last_name` = ?, `class_id` = ?, `last_update` = ? WHERE `user_id` = ?", [$POST['first_name'], $_POST['last_name'], $POST['class_id'], time()]);
                $account = $this->db->query("SELECT * FROM `users` WHERE `user_id` = ?", [$SESSION['user_data']['user_id']])->getArray();
                HelperFunctions::setAlert("success-profile", "Váš profil byl aktualizován");
            }
        }
        else
        {
            HelperFunctions::setAlert("error-profile", "Profil můžete aktualizovat maximálně 1x denně");
        }
    }

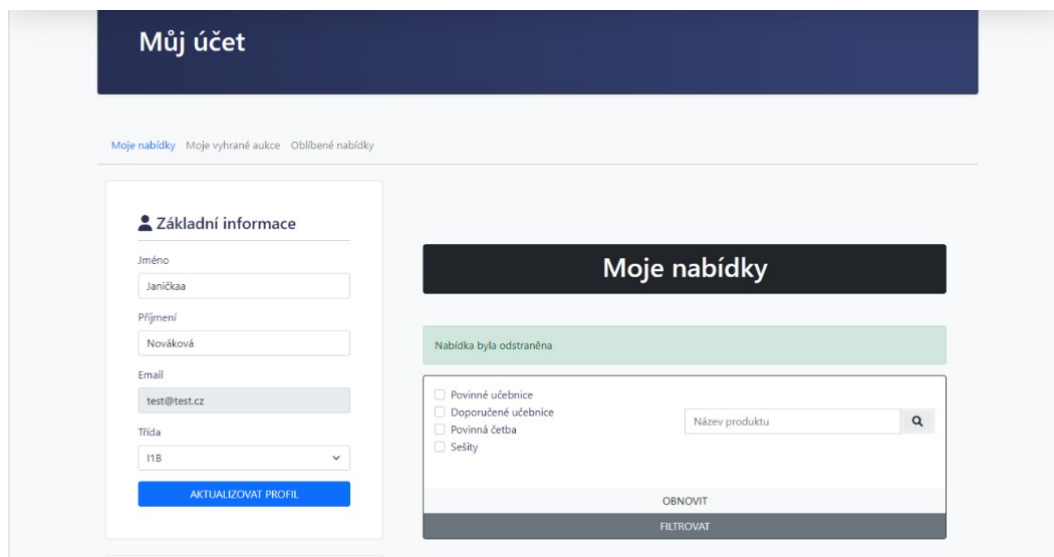
    $account = $this->users_model->get_by_id($SESSION['user_data']['user_id']);

    if(isset($_GET['delete']))
    {
        if($this->offers_model->is_mine($SESSION['user_data']['user_id'], $_GET['delete']))
        {
            $this->offers_model->delete($SESSION['user_data']['user_id'], $_GET['delete']);
            HelperFunctions::setAlert("success-offer", "Nabídka byla odstraněna");
        }
    }

    $this->render("views/templates/header.php");
    $this->render("views/account/my_account.php", $this->account_data);
    $this->render("views/templates/footer.php");
}
```

Obrázek 22 Metoda „my_account“

A nakonec tato metoda zobrazí pohled *my_account.php* s daty přihlášeného uživatele.



Obrázek 23 Pohled „my_account.php“

6.1.2 Metoda tab_my_offers

Zobrazení nabídek přihlášeného uživatele v záložce „Moje nabídky“

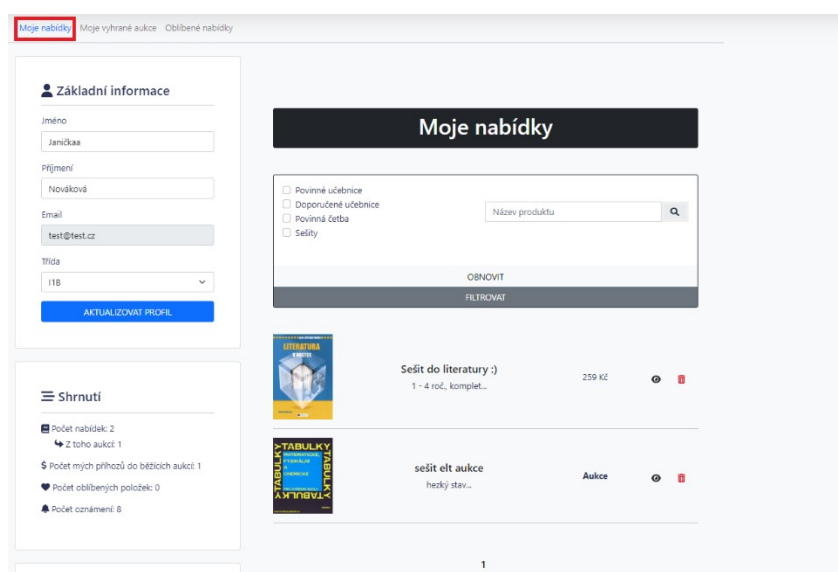
```
public function tab_my_offers()
{
    // Read filters
    $this->offers_model->read_filters();
    $tmp_offer_count = $this->offers_model->get_count_from_user($_SESSION['user_data']['user_id']);

    // My offers pagination (5 offers per page)
    $this->account_data['my_offers_pagination'] = new Pagination($tmp_offer_count, "muj-ucet", "po");
    $this->account_data['my_offers_pagination']->set_items_per_page(5);
    $this->offers_model->set_limit($this->account_data['my_offers_pagination']->get_limit_a(), $this->account_data['my_offers_pagination']->get_limit_b());

    // Get offers for my account
    $this->account_data['offers'] = $this->offers_model->get_from_user($_SESSION['user_data']['user_id'], "date", "DESC");

    $this->render("views/account/my_offers.php", $this->account_data);
}
```

Obrázek 24 Metoda „tab_my_offers“



Obrázek 25 Záložka „Moje nabídky“

6.1.3 Metoda `tab_my_auctions`

Zobrazení aukcí, které uživatel vyhrál v záložce „Moje vyhrané aukce“

```
public function tab_my_won_auctions()
{
    // Get won auctions for my account
    $won_auctions = $this->auctions_model->get_won_auctions_from_user($SESSION['user_data']['user_id']);
    $this->account_data['won_auctions'] = [];
    foreach($won_auctions as $won_auction)
    {
        // Lazy for SQL join :D (Sorry for performance impact)
        $this->account_data['won_auctions'][] = array_merge($won_auction, $this->offers_model->get_by_id($won_auction['offer_id']));
    }

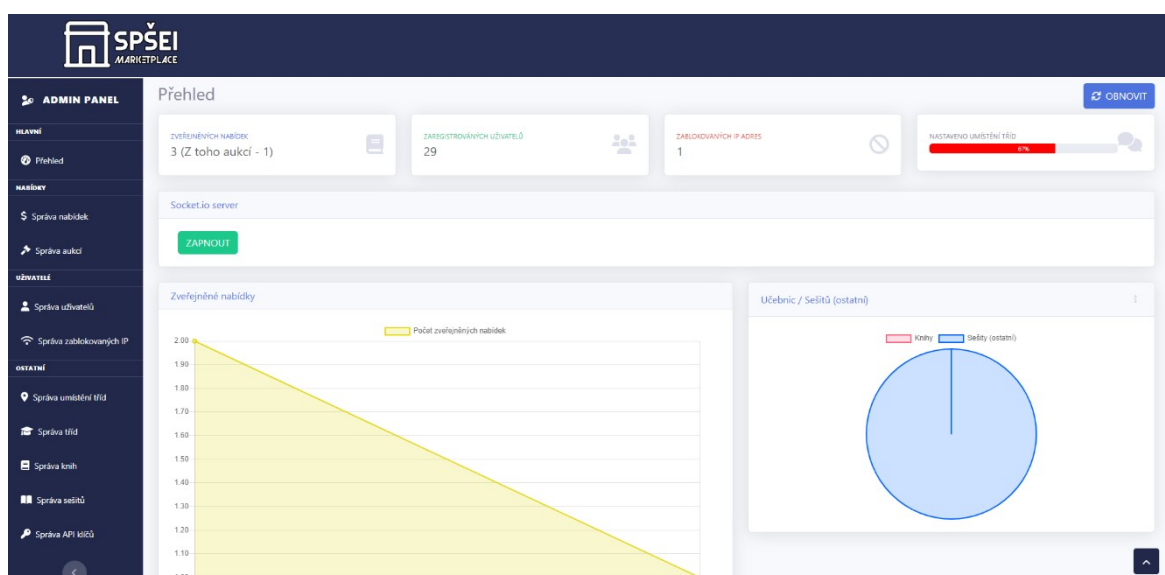
    // Won auctions pagination (5 offers per page)
    $this->account_data['my_won_auctions_pagination'] = new Pagination(count($this->account_data['won_auctions']), "muj-ucet", "pa");
    $this->account_data['my_won_auctions_pagination']->set_items_per_page(5);
    $this->offers_model->set_limit($this->account_data['my_won_auctions_pagination']->get_limit_a(), $this->account_data['my_won_auctions_pagination']->get_limit_b());

    $this->render("views/account/my_won_auctions.php", $this->account_data);
}
```

Obrázek 26 Metoda „`tab_my_won_auctions`“

6.2 AdminController.php

Tento kontrolér řídí administrátorské rozhraní a všechny jeho příslušné části. V konstruktoru této třídy proběhne získání dat pro pohled *dashboard.php*, který je všeobecným přehledem (shrnutím) nad dosavadními daty. Například počet zaregistrovaných uživatelů apod. Viz obrázek níže.



Obrázek 27 Pohled „*dashboard.php*“

6.2.1 Metoda user_maintenance

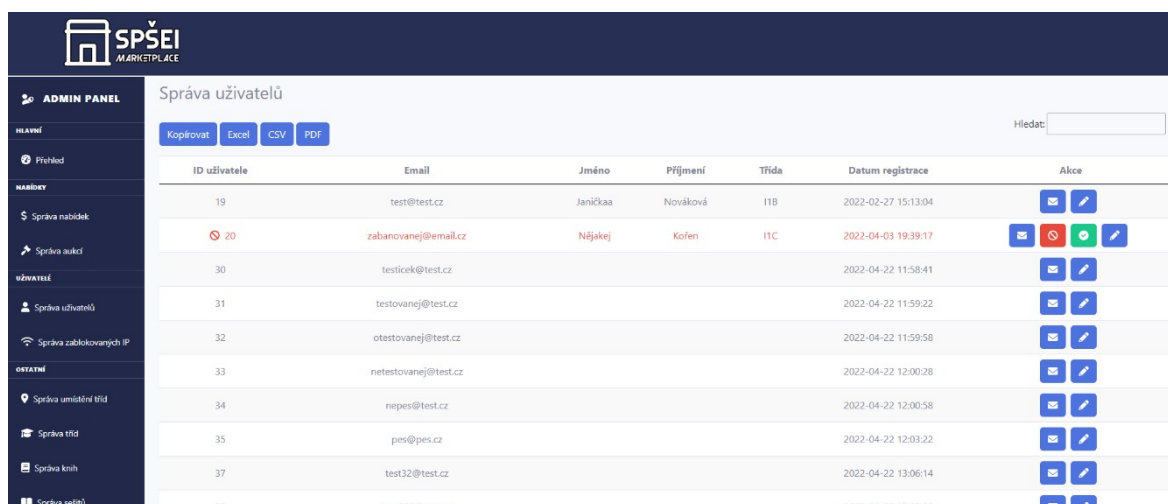
Zajišťuje správu nad uživateli. Podle parametru umí zablokovat / odblokovat uživatele. Nelze však zablokovat sám sebe.

```
public function user_maintenance()
{
    if(isset($_GET['ban']) && $_GET['ban'] != "null" && $_GET['ban'] != $_SESSION['user_data']['ip_address'])
    {
        $this->users_model->ban_ip($_GET['ban']);
        unset($_GET['ban']);
    }
    else if(isset($_GET['unban']) && $_GET['unban'] != "null" && $_GET['unban'] != $_SESSION['user_data']['ip_address'])
    {
        $this->users_model->unban_ip($_GET['unban']);
        unset($_GET['unban']);
    }

    $data['users'] = $this->users_model->get_all();

    $this->render("views/templates/admin/header.php");
    $this->render("views/admin/user_maintenance.php", $data);
    $this->render("views/templates/admin/footer.php");
}
```

Obrázek 28 Metoda „user_maintenance“



ID uživatele	Email	Jméno	Příjmení	Třída	Datum registrace	Akce
19	test@test.cz	Janička	Nováková	11B	2022-02-27 15:13:04	[edit] [delete]
20	zabanovanej@email.cz	Nějaký	Kořen	11C	2022-04-03 19:39:17	[edit] [delete] [ban] [unban]
30	testick@test.cz				2022-04-22 11:58:41	[edit] [delete]
31	testovanej@test.cz				2022-04-22 11:59:22	[edit] [delete]
32	otestovanej@test.cz				2022-04-22 11:59:58	[edit] [delete]
33	netestovanej@test.cz				2022-04-22 12:00:28	[edit] [delete]
34	nepes@test.cz				2022-04-22 12:00:58	[edit] [delete]
35	pes@pes.cz				2022-04-22 12:03:22	[edit] [delete]
37	test32@test.cz				2022-04-22 13:06:14	[edit] [delete]
38	test327@test.cz				2022-04-22 13:09:08	[edit] [delete]

Obrázek 29 Pohled „user_maintenance.php“

6.2.2 Metoda offer_maintenance

Správa nabídek

6.2.3 Metoda auction_maintenance

Správa aukcí

6.2.4 Metoda class_maintenance

Správa tříd (ve škole)

6.2.5 Metoda book_maintenance

Správa knížek

6.2.6 Metoda notebook_maintenance

Správa sešitů

6.2.7 Metoda cr_maintenance

Správa umístění tříd (jaká třída je v jaké učebně, např. I4C = B306)

6.2.8 Metoda banned_ip_maintenance

Správa zablokovaných *ip adres*.

6.2.9 Metoda api_key_maintenance

Správa *API klíčů* pro přístup k soukromému *API*.

6.2.10 Metody s příponou _edit

Ve třídě *AdminController.php* máme hodně metod s příponou *_edit*. Všechny metody pracují v podstatě stejně. Vygenerují formulář pomocí dříve zmíněné třídy *FormGenerator.php*, předvyplní ho a po jeho odeslání aktualizují záznam v databázové tabulce.

```
public function book_edit()
{
    if(!isset($_GET['id']) || empty($_GET['id']) || empty($this->books_model->get_by_id($_GET['id'])))
    {
        header('Location: /polozka-neexistuje');
        return;
    }

    $id = $_GET['id'];

    $form_generator = new FormGenerator("/admin/upravit-knihu?id=".$id, "POST", ["books"], "Editace knihy (ID: ".$id.")", [], [
        "class" => "mb-5"
    ]);
    $form_generator->set_fields_from_tables([
        "books" => ["book_ISBN", "name", "author", "category_id", "grade", "major_id"],
    ]);

    if($_POST)
    {
        $this->validator->addMultipleRules($form_generator->get_validation_rules());

        if ($this->validator->run())
        {
            $this->books_model->update($_POST, $id);
            // In case we updated book_ISBN:
            $parameters = $_GET;
            $parameters['id'] = $_POST['book_ISBN'];
            header("Location: /admin/upravit-knihu?".http_build_query($parameters));
            die;
        }
        else
        {
            $data['errors'] = $this->validator->getErrors();
        }
    }

    // Prefill form after form submission so we get new data
    $form_generator->prefill_form_by_id($id);

    $data['back_btn'] = "<a href='/admin/sprava-knih' class='my-1 btn btn-primary text-uppercase'><i class='fa-solid fa-arrow-left'></i> Zpět</a>";
    $data['edit_form'] = $form_generator->get_html_result();

    $this->render("views/templates/admin/header.php");
    $this->render("views/templates/admin/edit.php", $data);
    $this->render("views/templates/admin/footer.php");
}
```

Obrázek 30 Metody s příponou „_edit“

Obrázek 31 Ukázka vygenerovaného formuláře

6.3 AjaxController.php

Obsahem této třídy jsou metody, které jsou volány skrze *ajax* a navrací odpověď ve formátu *JSON*.

Např. metoda *process_list* vrátí procesy běžící nad databází.

```
public function process_list()
{
    if(!Filter::is_admin()) die;
    echo json_encode($this->db->query("SHOW FULL PROCESSLIST")->getResultArray());
}
```

Obrázek 32 Metoda „process_list“

6.4 ApiController.php

Třída pro zajištění spojení na *API* pro externí aplikace. Když by někdo v budoucnu chtěl využít data z burzy, bude mít pro to možnost. V současnosti jsou dostupné 2 *endpointy*.

Aby *api* fungovalo, je potřeba v *URL adrese* specifikovat parametr *key* s klíčem který je možno vygenerovat v administrátorském rozhraní.

Obrázek 33 Ukázka URL adresy

6.4.1 Metoda offers (URL: /api/nabídky)

Seskupí nabídky podle *uživatelského id* a navrátí v *JSON* formátu.


```

public function offers()
{
    $result = [];
    $offers = $this->db->query("SELECT cr.room_code AS 'room_code', o.*, cat.value AS 'category', u.user_id AS 'u_user_id', u.first_name AS 'u_first_name', u.last_name AS
                                FROM 'offers' `o`
                                LEFT JOIN 'books' `b` ON `o`.`book_ISBN` = `b`.`book_ISBN`
                                INNER JOIN 'users' `u` ON `u`.`user_id` = `o`.`user_id`
                                LEFT JOIN 'classes' `c` ON `c`.`class_id` = `u`.`class_id`
                                LEFT JOIN 'class_room' `cr` ON `cr`.`class_id` = `u`.`class_id`
                                LEFT JOIN 'auctions' `a` ON `a`.`offer_id` = `o`.`offer_id`
                                LEFT JOIN 'categories' `cat` ON `b`.`category_id` = `cat`.`category_id`
                                WHERE a.auction_id IS NULL AND `c`.`name` = ?", [$GET['trida']])->getResultArray();

    foreach($offers as $offer)
    {
        if(isset($result[$offer["user_id"]]))
        {
            array_push($result[$offer["user_id"]], $offer);
        }
        else
        {
            $result[$offer["user_id"]] = [
                $offer
            ];
        }
    }

    echo json_encode($result);
}

```

Obrázek 34 Metoda „offers”

6.4.2 Metoda classes (URL: /api/tridy)

Navrátí názvy tříd společně s kódem místností, ve které se nacházejí.

6.5 AuctionController.php

Kontrolér pro řízení aukcí.

6.5.1 Metoda rise_price

Za předpokladu že aukce s poslaným id existuje a že uživatel nabídnul více peněz než poslední nabízející, se nejvyšší nabídka této aukce aktualizuje, a jako dosavadní výherce se uloží právě tento uživatel. Taktéž uložíme do *session* informace o čase, kdy uživatel nabídku provedl. To proto, abychom předešli hromadnému nabízení a přetěžování serveru. Nakonec se informace o změně se pošlou na *WebSocket* server.

```

public function rise_price()
{
    if($_POST && Filter::is_ajax_request())
    {
        $this->validator->addMultipleRules([
            'auction_id' => 'is_not_unique[auctions.auction_id]',
            'new_price' => 'required|is_number|greater_than[0]|less_than[10001]',
        ]);
        if($this->validator->run() && $this->can_user_bid() && !$this->auction_model->is_mine($_POST['auction_id'], $_SESSION['user_data']['user_id']))
        {
            if($_POST['new_price'] > $this->auction_model->get_current_state($_POST['auction_id'])['top_bid'])
            {
                $this->auction_model->rise_price($_POST['auction_id'], $_POST['new_price'], $_SESSION['user_data']['user_id']);
                $_SESSION['auction']['last_bid_time'] = time();

                $data = [
                    "event" => "auction_price_rised",
                    "auction_id" => $_POST['auction_id'],
                    "new_price" => $_POST['new_price'],
                    "user_id" => $_SESSION['user_data']['user_id'],
                ];

                $this->emitter->emit('auction_change', json_encode($data));
            }
        }
    }
}

```

Obrázek 35 Metoda „rise_price”

6.5.2 Metoda `can_user_bid`

Pomocná metoda pro zjištění, zdali uživatel může provést další nabídku. To zjistíme tak že doba od jeho poslední nabídky je větší než povolený interval pro nabízení (v sekundách). Tento interval lze nastavit v konfiguračním souboru *config.php* a je definovaný jako konstanta *AUCTION_BID_DELAY*.

```
private function can_user_bid()
{
    return (!isset($_SESSION['auction']['last_bid_time']) || (time() - $_SESSION['auction']['last_bid_time'] >= AUCTION_BID_DELAY));
}
```

Obrázek 36 Metoda „*can_user_bid*”

6.6 AuthController.php

Řízení přihlášení, registrace a odhlášení uživatele.

Pro uchování hesel v databázi používáme hashovací algoritmus *CRYPT_BLOWFISH* který je definovaný konstantou *PASSWORD_BCRYPT*.

```
class AuthController extends BaseController
{
    private $hashing_algorithm = PASSWORD_BCRYPT;
    private $validator;
    private $user_model;
```

Obrázek 37 Atributy třídy „*AuthController*”

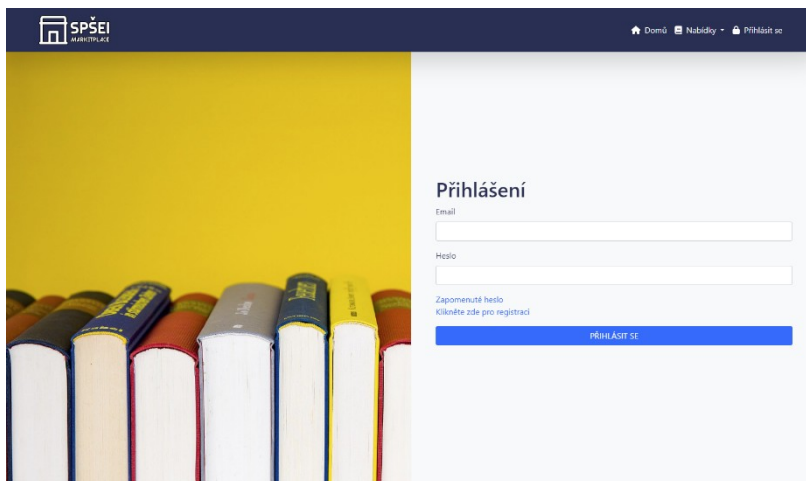
6.6.1 Metoda `login`

Přihlášení uživatele probíhá tak, že se vybere odpovídající uživatel z databáze podle emailu a pokud takový uživatel existuje a jeho heslo je shodné s heslem odeslaným ve formuláři, proběhne přihlášení uživatele. Za přihlášeného uživatele považujeme takového uživatele, který má uložená data v globální proměnné *\$_SESSION['user_data']*.

```
public function login()
{
    if ($_POST)
    {
        $this->validator->addMultipleRules([
            'email' => 'required|max_length[100]|is_not_unique[users.email]|is_valid_email',
            'password' => 'required|max_length[255]',
        ]);
        if ($this->validator->run())
        {
            $email = trim($_POST['email']);
            $password = $_POST['password'];
            $user_info = $this->user_model->get_by_email($email);

            // If user with posted email exists and posted password matches the stored user password
            if(isset($user_info) && password_verify($password, $user_info['password']))
            {
                $_SESSION['user_data'] = $user_info;
                header("Location: /domu");
                die;
            }
        }
    }
}
```

Obrázek 38 Metoda „*login*”



Obrázek 39 Pohled „login.php“

6.7 BaseController.php

Tato třída je děděná každým kontrolérem.

6.7.1 Metoda render

Rozbalení dat pro pohled a připojení pohledu.

```
protected function render($path, $data = null)
{
    $data['notifications'] = $this->get_notifications();

    extract($data);

    require_once($path);
}
```

Obrázek 40 Metoda „render“

6.7.2 Metoda get_notifications

Tato metoda načte a navrátí všechny oznámení pro přihlášeného uživatele. Jelikož se nachází v základním kontroléru, budou tyto oznámení aktualizovány při každém načtení jakékoliv stránky. To pro tyto účely stačí.

```
private function get_notifications()
{
    $notifications_model = new Notification();

    if(isset($_SESSION['user_data']['user_id']))
    {
        return $notifications_model->get_for_user($_SESSION['user_data']['user_id']);
    }

    return [];
}
```

Obrázek 41 Metoda „get_notifications“

6.8 ChatController.php

Řízení komunikace v interním *chatu*.

6.8.1 Metoda `create_new_chat`

Založení nového *chatu* probíhá tak, že se prvně zjistí, jestli daný *chat* pro uživatele již neexistuje. Pokud ne tak se vygeneruje *unikátní id* pro nový *chat*. Potom se vloží nový záznam do tabulky s *chaty* pro oba uživatele (jak pro přihlášeného, který spustil založení *chatu*, tak pro cíleného uživatele, s kterým *chat* bude).

```
if(!($chat_id = $this->chat_model->get_chat_id($_SESSION['user_data']['user_id'], $_POST['user_id'])))
{
    $chat_id = uniqid("", true);

    // Associate new chat with me and target user :D
    $this->chat_model->post([
        "chat_id" => $chat_id,
        "user_id" => $_SESSION['user_data']['user_id'],
    ]);

    $this->chat_model->post([
        "chat_id" => $chat_id,
        "user_id" => $_POST['user_id'],
    ]);
}
```

Obrázek 42 Metoda „`create_new_chat`“

Před posláním zprávy se zkontroluje, jestli *chat* patří mě. Pokud ano tak se nová zpráva odešle na *WebSocket* server a taky se uloží do databáze.

```
if($this->chat_model->is_chat_mine($chat_id, $_SESSION['user_data']['user_id']))
{
    $data = [
        "chat_id" => $chat_id,
        "sender" => $_SESSION['user_data']['user_id'],
        "text" => $_POST['message'],
        "date_sent" => date("Y-m-d H:i:s", time()),
    ];

    $this->emitter->emit('message_sent', json_encode($data));

    $this->message_model->post($data);
}
```

Obrázek 43 Metoda „`send_message`“

6.9 CronController.php

Tato třída obsahuje metody, které jsou spouštěny plánovačem úloh pro *Linux*. Jedná se například o smazání starých nabídek, nebo uzavření aukcí, které skončily.

6.10 ErrorController.php

Jak napovídá název třídy, tato třída řeší zobrazení chybových hlášek.

404 Chyba

Litujeme, stránka nebyla nalezena

Obrázek 44 Ukázka chybové hlášky

6.11 OfferController.php

Tento kontrolér řídí proces zobrazování a nahrávání nabídek.

Dalo by se říct, že proces nahrání nabídky je poměrně složitý. To proto, že na každou kategorii nabídky jsou různá validační pravidla.

Například pro kategorii č.3 (knihy) musí existovat odpovídající záznam knihy v databázi, ale pro kategorii č.4 (sešity) musí sešit být pro ročníky v rozsahu 1-4 a pro existující obory (ELT / IT).

```
switch(intval($_POST['category']))
{
    case 1:
    case 2:
    case 3:
        $this->validator->addMultipleRules([
            'name' => 'required|is_not_unique[books.book_isbn]',
        ]);
        break;

    case 4:
        $this->validator->addMultipleRules([
            'name' => 'required|min_length[3]|max_length[50]',
            // Grade 0 = All grades
            'grade' => 'required|in_list[0,1,2,3,4]',
            'major' => 'required|is_not_unique[majors.major_id]',
        ]);
        break;
}
```

Obrázek 45 Validace na základě kategorie nabídky

Nejsložitější je validace pro typ aukce. Musí se totiž zkontrolovat odeslaný datum a čas, tak aby aukce mohla být spuštěna. Například aukce nemůže běžet v minulosti, a taky nemůže trvat 1 sekundu.

```
case 'aukce':
    $this->validator->addMultipleRules([
        'price' => 'permit_empty',
        'start_date' => 'required|is_valid_datetime|datetime_greather_than['.(time() + 3600 - 1).']',
        'end_date' => 'required|is_valid_datetime|datetime_greather_than[start_date,'.(86400 - 1).']',
    ]);
    break;
```

Obrázek 46 Validace pro nabídky typu aukce

Pro pevnou cenu to je jednoduché. Stačí zkontrolovat, aby uživatel nezadal cenu příliš vysokou (např. 1 milión Kč). Nastavení, jakou maximální cenu nabídka může mít nalezneme opět v souboru *config.php*.

```

case 'pevna':
    $this->validator->addMultipleRules([
        'price' => 'required|is_number|less_than['.(MAX_OFFER_PRICE+1).']',
    ]);
    $price = $_POST['price'];
    break;

```

Obrázek 47 Validace pro nabídky s pevnou cenou

6.12 PageController.php

Obsahuje metody pro zobrazení jednoduchých statických stránek bez funkčnosti. Příkladem je domovská stránka.

6.13 WishlistController.php

Třída pro řízení seznamu oblíbených nabídek. Zajišťuje přidávání / mazání ze seznamu oblíbených. Velkou nevýhodou je, že oblíbené nabídky ukládáme do `$_SESSION`, a tak po odhlášení se vymažou. Na druhou stranu cílem seznamu oblíbených nabídek bylo uložit si nabídky jen dočasně a po výběru všech potřebných učebních materiálů se hned domluvit na jejich nákupu. Nejde tedy o dlouhodobé uchování položek (počítá se také s tím, že položky uveřejněné na burze půjdou na odbyt rychle).

6.13.1 Metoda `add_or_delete`

Tato metoda zkontroluje, jestli nabídka vůbec existuje, poté jestli nabídka je cizí (nemůžu si přidat do oblíbených svoji nabídku) a nakonec se nabídka do seznamu oblíbených položek přidá (pokud tam ještě není) anebo odebere (pokud tam už byla).

```

public function add_or_delete()
{
    if($_POST && Filter::is_ajax_request())
    {
        $offer_id = $_POST['offer_id'];

        $this->validator->addMultipleRules([
            'offer_id' => 'is_not_unique[offers.offer_id]',
        ]);
        if($this->validator->run() && !$this->offers_model->is_mine($_SESSION['user_data']['user_id'], $_POST['offer_id']))
        {
            if(in_array($offer_id, $_SESSION['wishlist']))
                $this->delete($offer_id);
            else
                $this->add($offer_id);
        } // If offer doesn't exist anymore -> delete it from wishlist
        else
        {
            $this->delete($offer_id);
        }
    }
}

```

Obrázek 48 Metoda „`add_or_delete`”

7. Server běžící na protokolu WebSocket

V adresáři `/bin` se nachází dva skripty (dvě třídy). První je skript *ChannelServer.php* a druhý je skript *SocketIOServer.php*. Oba skripty obsahují třídy, které jsou tvořeny dalšími třídami externích knihoven *Ratchet* a *phpsocket.io*. Pro zajištění funkčnosti serveru je třeba spustit oba tyto skripty zároveň. Pro ulehčení tohoto úkonu se v tomto adresáři nachází také soubor *start_server.bat*, který stačí spustit, nechat otevřený a server nám bude na pozadí běžet.

7.1 ChannelServer.php

```
<?php

namespace SpseiMarketplace\Core;

// This has to be required because this is fired via CMD :D
// Vendor autoloading classes
require realpath(' ../vendor/autoload.php');
require realpath(' ../config.php');

use Channel\Server;
use Workerman\Worker;

class ChannelServer
{
    private $ip;
    private $server;

    public function __construct($ip)
    {
        $this->ip = $ip;
        $this->server = new Server($this->ip);
    }

    public function run()
    {
        Worker::runAll();
    }
}

// When this file is executed from CMD with php command -> start socket server
$server = new ChannelServer(SITE_IP);
$server->run();
```

Obrázek 49 Skript „ChannelServer”

7.2 SocketIOServer.php

```
<?php

namespace SpseiMarketplace\Core;

// This has to be required because this is fired via CMD :D
// Vendor autoloading classes
require realpath('../vendor/autoload.php');
require realpath('../config.php');

use Workerman\Worker;
use PHPSocketIO\SocketIO;
use Emitter;

class SocketIOServer
{
    private $port;
    private $socket_io;

    public function __construct($port)
    {
        $this->port = $port;
        $this->socket_io = new SocketIO($port);
    }

    public function run()
    {
        $io = $this->socket_io;

        $io->on('workerStart', function () use ($io) {
            $io->adapter('\PHPSocketIO\ChannelAdapter');
        });
        $io->on('connection', function ($socket) use ($io) {
            echo "new connection\n";
            // Broadcast all information about new client to all connections (for debugging)
            $io->emit('broadcast', [
                'id' => $socket->id,
                'rooms' => $socket->rooms,
                'request' => $socket->request,
                'handshake' => $socket->handshake
            ]);
        });

        Worker::runAll();
    }
}

// When this file is executed from CMD with php command -> start socket server
$server = new SocketIOServer(WEB_SOCKETS_PORT);
$server->run();
```

Obrázek 50 Skript „SocketIOServer“

8. Knihovny

8.1 Správa knihoven

V moji aplikaci pro správu *PHP* knihoven používám *Composer*. Tento jednoduchý nástroj zajišťuje stažení knihoven v případě, že neexistují. Taký umí připojené knihovny aktualizovat.

```
{
  "name": "p0bt/spseimarketplace",
  "description": "Spšei Marketplace - School Project",
  "license": "MIT",
  "autoload": {
    "psr-4": {
      "SpseiMarketplace\\": "src/"
    }
  },
  "require": {
    "cboden/ratchet": "^0.4.4",
    "ratchet/pawl": "^0.4.1",
    "workerman/phpsocket.io-emitter": "^1.0",
    "workerman/phpsocket.io": "^1.1"
  },
  "config": {
    "platform-check": false
  }
}
```

Obrázek 51 „Composer.json“

8.2 Seznam PHP knihoven

- *workerman/phpsocket.io*
<https://packagist.org/packages/workerman/phpsocket.io>
- *workerman/phpsocket.io-emitter*
<https://packagist.org/packages/workerman/phpsocket.io-emitter>
- *cboden/ratchet*
<https://packagist.org/packages/cboden/ratchet>
- *ratchet/pawl*
<https://packagist.org/packages/ratchet/pawl>

8.3 Seznam JS knihoven

- *Aos.js*
<https://michalsnik.github.io/aos/>
- *Chart.js v2.8.0*
<https://www.chartjs.org>
- *Confetti.js*
<https://confettijs.org/>
- *Datatables*
<https://datatables.net/download>
- *Dropzone*
<https://www.dropzone.dev/>

- *jQuery v3.6.0*
<https://jquery.com/download/>
- *jQuery UI – v1.13.1*
<http://jqueryui.com>
- *Natural.js*
<https://www.npmjs.com/package/natural>
- *SweetAlert2*
<https://sweetalert2.github.io/>
- *Swiper 8.1.6*
<https://swiperjs.com>
- *Tilt.js*
<https://gijsroge.github.io/tilt.js/>

8.4 Seznam CSS knihoven

- *Animate.css 4.1.1*
<https://animate.style/>
- *Bootstrap v5.0.2*
<https://getbootstrap.com/>
- *Font Awesome Free 6.0.0*
<https://fontawesome.com>

8.5 Šablony

- *SB Admin 2*
<https://startbootstrap.com/theme/sb-admin-2>

9. Návod

9.1 Instalace / inicializace aplikace pro Linux

Pokud jste s instalací obeznámení, můžete rovnou přeskočit na „Speciální kroky instalace.“

1. Stáhneme *.zip* *archív* a přesuneme ho na plochu
2. Extrahujeme *.zip* *archív*

```
sudo apt install unzip  
  
unzip spsei_bookmarket.zip
```

3. Nainstalujeme webový server *apache2*

```
apt-get install apache2
```

4. Nainstalujeme *PHP* (jelikož v aplikaci potřebujeme verzi *PHP* 8 a výše, musíme přidat další repositář do seznamu zdrojů)

```
sudo apt install software-properties-common  
  
sudo add-apt-repository ppa:ondrej/php  
  
sudo apt update  
  
sudo apt install php8.0 libapache2-mod-php8.0  
  
sudo systemctl restart apache2
```

5. Nainstalujeme *MySQL*

```
apt-get install default-mysql-server
```

6. Nainstalujeme *phpmyadmin* rozhraní

```
apt-get install phpmyadmin
```

7. Nyní je potřeba pozměnit konfiguraci *apache2*

```
nano /etc/apache2/apache2.conf
```

A připojte tento řádek na konec souboru, pro připojení rozhraní *phpmyadmin*:

```
Include /etc/phpmyadmin/apache.conf
```

Dále je ještě potřeba povolit přepisování, protože používáme soubory *.htaccess*.

Nahrad'te:

```
<Directory /var/www/>  
  
    Options Indexes FollowSymLinks  
  
    AllowOverride None
```

```

        Require all granted

</Directory>

Tímto:

<Directory /var/www/>

    Options Indexes FollowSymLinks

    AllowOverride All

    Require all granted

</Directory>

```

8. Nyní musíme vytvořit nového uživatele *phpmyadmin* s heslem *phpmyadmin* (heslo změňte na bezpečné), protože se nemůžeme přihlásit do *phpmyadmin* jako *root* od novějších verzí

```

CREATE USER 'phpmyadmin'@'localhost' IDENTIFIED BY
'phpmyadmin';

GRANT ALL PRIVILEGES ON *.* TO 'phpmyadmin'@'localhost' WITH
GRANT OPTION;

FLUSH PRIVILEGES;

```

9. Restartujeme *apache2*, otevřeme prohlížeč, přihlásíme se do *phpmyadmin*

```
systemctl restart apache2
```

10. Vytvoříme novou databázi *spsei_marketplace* a importujeme do ní přiložený *SQL skript*
11. Přesuneme se do adresáře */var/www/html* a smažeme soubor *index.html*

```

cd /var/www/html

rm index.html

```

12. Přesuneme se opět na plochu a zkopírujeme obsah složky *spsei_bookmarket* do adresáře */var/www/html*

```
cp -a ./spsei_bookmarket/. /var/www/html
```

Speciální kroky instalace

13. Nastavíme oprávnění adresáři */var/www/html*

```
sudo chmod -R 755 /var/www/html
```

14. Otevřeme soubor */var/www/html/config.php* a nastavíme hodnoty konstant (intuitivně) podle potřeby

```
nano /var/www/html/config.php
```

Pokud konfiguruje server pro produkci, je důležité od komentovat tento řádek:

```
//error_reporting(0)
```

Taktéž je důležité změnit konstanty pro nastavení databáze, a použít místo uživatele *root* námi vytvořeného uživatele

15. Přidáme skriptu pro *WebSocket server* veškerá oprávnění

```
chmod 777 /var/www/html/bin/start_linux.sh
```

16. Otevřeme nový terminál a přejdeme do adresáře */var/www/html/bin* a spustíme *WebSocket server* na pozadí

```
nohup start_linux.sh &
```

17. Posledním krokem je nastavení plánovaných úloh. Je potřeba spouštět tyto *URL adresy* v těchto intervalech:

"*/cron/delete-old-offers*" = Každý měsíc

"*/cron/close-old-auctions*" = Každých 5 minut

"*/cron/delete-old-tokens*" = Každý měsíc

```
sudo apt install curl
```

```
0 0 0 * * /usr/bin/curl --silent http://DOMÉNA/cron/delete-old-offers &>/dev/null
```

```
*/5 * * * * /usr/bin/curl -silent http://DOMÉNA/cron/close-old-auctions &>/dev/null
```

```
0 0 0 * * /usr/bin/curl --silent http://DOMÉNA/cron/delete-old-tokens &>/dev/null
```

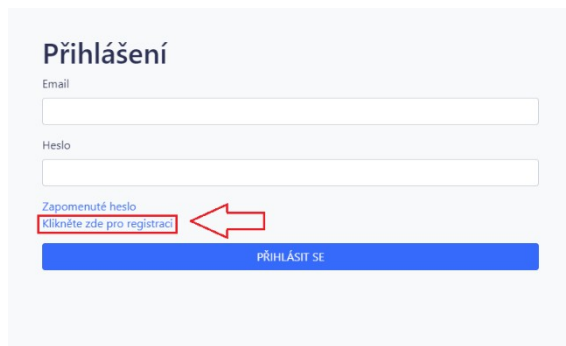
18. A nakonec restartujeme službu *apache2* a otestujeme, že vše funguje

```
systemctl restart apache2
```

9.2 Používání aplikace pro uživatele

9.2.1 Registrace

Pro začátek prodávání nebo nakupování na burze učebnic, je třeba si vytvořit uživatelský účet. To je možné na stránce přihlášení a kliknutí na odkaz „Klikněte zde pro registraci“

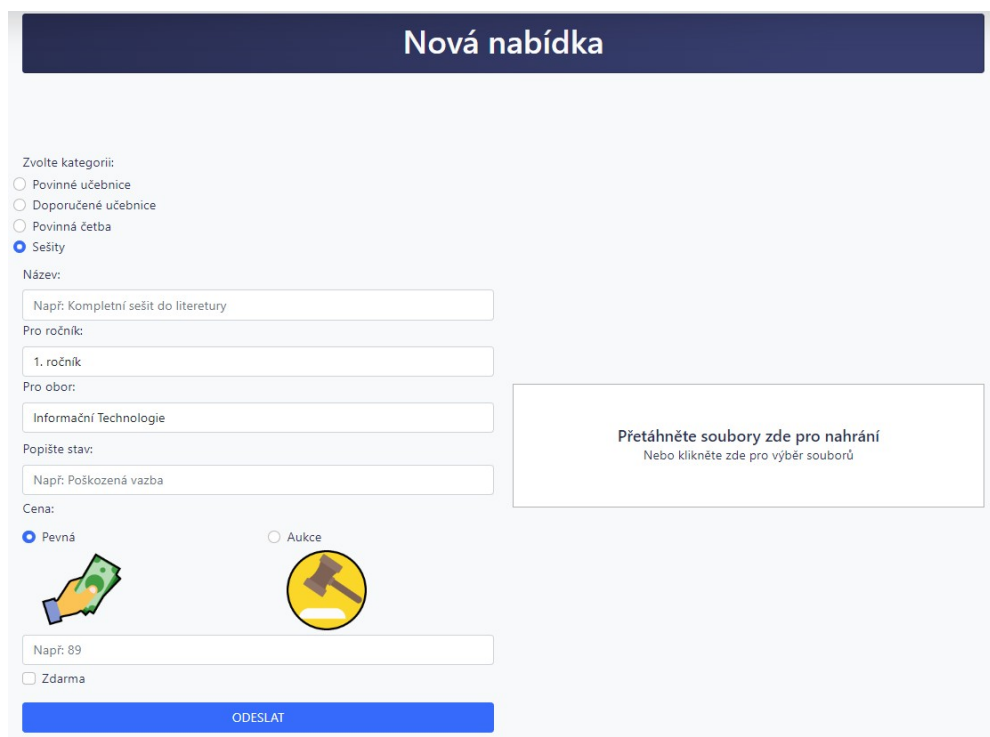


The screenshot shows a login form titled "Přihlášení". It has two input fields: "Email" and "Heslo". Below the "Heslo" field, there is a link "Zapomenuté heslo" and a red box containing the text "Klikněte zde pro registraci". A red arrow points from this box to the right. At the bottom of the form is a blue button labeled "PŘIHLÁSIT SE".

Obrázek 52 Stránka přihlášení

9.2.2 Zveřejnění nabídky

Zveřejnění nové nabídky je celkem jednoduché a intuitivní. Stačí vyplnit všechna políčka, a nezapomenou nahrát obrázek (obrázky), které jsou povinné. Po kliknutí na tlačítko „odeslat,“ vás aplikace upozorní na případné chyby.



The screenshot shows a form titled "Nová nabídka". It contains several sections: "Zvolte kategorii:" with radio buttons for "Povinné učebnice", "Doporučené učebnice", "Povinná četba", and "Sešity" (selected); "Název:" with a text input field containing "Např: Kompletní sešit do literatury"; "Pro ročník:" with a text input field containing "1. ročník"; "Pro obor:" with a text input field containing "Informační Technologie"; "Popište stav:" with a text input field containing "Např: Poškozená vazba"; "Cena:" with radio buttons for "Pevná" (selected) and "Aukce", and a text input field containing "Např: 89"; and a checkbox for "Zdarma". There are also icons for a hand holding money and a gavel. At the bottom is a blue button labeled "ODESLAT". On the right side, there is a box with the text "Přetáhněte soubory zde pro nahrání" and "Nebo klikněte zde pro výběr souborů".

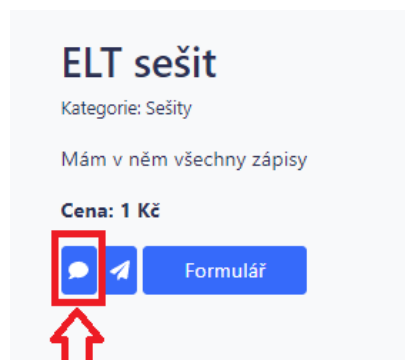
Obrázek 53 Zveřejnění nabídky

Po zveřejnění nabídky si můžete nabídku prohlédnout na stránce „Můj účet -> Moje Nabídky.“ Zájemci vás budou moci kontaktovat přes interní chat.

9.2.3 Nakupování nabídek

Pokud máte zájem o některou z uvedených nabídek, po zobrazení detailu nabídky máte možnost kontaktovat prodejce těmito způsoby:

- a) Můžete kliknout na obrázek *chatu* a napsat prodejci vlastní zprávu



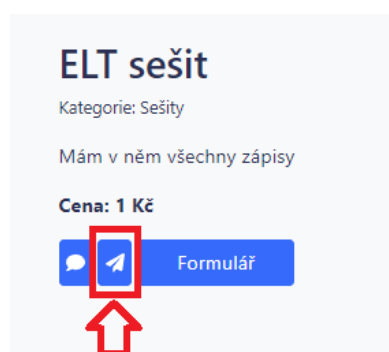
Obrázek 54 Kontaktování skrze „chat“

- b) Můžete použít předpřipravenou zprávu ve formuláři, pro odeslání zprávy prodejci do interního *chatu*

The image shows a contact form titled 'Kontaktní formulář'. It has a text area labeled 'Zpráva' containing the pre-filled text: 'Dobrý den, mám zájem o Váš inzerát č.73 [ELT sešit]. Je stále platný? Děkuji za odpověď.' Below the text area is a blue button labeled 'Odeslat'.

Obrázek 55 Kontaktování skrze formuláře pro „chat“

- c) Můžete prodejce kontaktovat přes email



Obrázek 56 Kontaktování skrze emailu

9.2.4 Zapojení v aukcích

Pro zapojení do aukce stačí vyplnit políčko s částkou v korunách a kliknutím na tlačítko „+“.



Obrázek 57 Ukázka příhozu do aukce

Pokud váš příhoz nikdo nepřekoná, přijde vám i prodejci po skončení aukce tato zpráva do chatu:

Aukce č. 19 skončila. Prodejce: test@test.cz, výherce: test1234@test.cz. Domluvte se prosím na předání a zaplacení předmětu.

Obrázek 58 Zpráva při výhře aukce

9.2.5 Oblíbené nabídky

Na stránce „Můj účet -> Oblíbené nabídky“ naleznete nabídky, které jste přidali do oblíbených. Umožňuje to poté rychlejší kontaktování prodejců skrz tlačítko. Po kliknutí se zobrazí okénko s předepsanou zprávou pro rychlé odeslání.



Náhled	Název	Cena / Info	Akce
	Matematika I (Jiřina Petáková)	Aukce již skončila	
	Sešit do literatury :)	259 Kč	  

Obrázek 59 Oblíbené nabídky

9.2.6 Správa účtu

Spravovat nastavení svého účtu můžete na stránce „Můj účet.“ Můžete si nastavit jméno, příjmení a třídu. Taký si můžete změnit heslo. Pro správu svých nabídek přejděte na „Můj účet -> Moje nabídky.“

9.3 Používání aplikace pro administrátora

Administrátor může určovat další administrátory. Výchozí účet pro administrátora je zvolen takto:

Email: [XXXXXX](#)

Heslo: XXXXXX

Je nutné hned po uvedení aplikace do provozu změnit heslo na nějaké bezpečné.

Do administrátorského rozhraní se dostaneme přes odkaz v patičce webu.



Obrázek 60 Patička webu

Stránka v administrátorském rozhraní je hodně, a proto popíšu jen speciální funkce, které jsou jedinečné pro danou stránku.

9.3.1 Správa nabídek

Správa nabídek								
Kopírovat Excel CSV PDF			Hledat: <input type="text"/>					
ID nabídky	Přidáno uživatelem	Název	Popis	Kategorie	Cena / Typ	Adresář s obrázky	Datum zveřejnění	Akce
73	pes1234@pes.cz	ELT sešit	Mám v něm všechny zápisy	Sešity	1 Kč	 	2022-12-06 16:26:39	 
75	test@test.cz	Sešit do literatury :)	1 - 4 roč., komplet	Sešity	259 Kč		2022-12-20 13:01:04	 
77	test@test.cz	das	aasdsadads	Sešity	121 Kč		2022-12-20 14:08:38	 
78	test@test.cz	Matematika I (Jiřina Petáková)	Nová	Doporučené učebnice	Aukce		2023-02-08 13:28:20	 
Zobrazuji 1 až 4 z celkem 4 záznamů							Předchozí	1 Další

Obrázek 61 Správa nabídek

1) Lze zobrazit obrázky nahrané k nabídce



Obrázek 62 Zobrazení obrázků

2) Na každé stránce lze najít ikonu, jak je vidět na obrázku, jedná se o formulář s editací.

9.3.2 Správa aukcí

Správa aukcí

☒ Všechny aukce **1**
☐ Budoucí aukce
☐ Běžící aukce
☐ Ukončené aukce

Kopírovat Excel CSV PDF
 Hledat:

ID aukce	Nabídka 2	Nejvyšší příhoz	Aktuální vyhrávající	Začátek aukce	Konec aukce	Akce 3 4 5
19		553234323 Kč	ID: 19	2023-02-07 15:27:00	2023-02-08 13:51:44	

Zobrazuji 0 až 0 z 0 záznamů (filtrováno z celkem NaN záznamů)

Předchozí **1** 2 3 4 5 ... NaN Další

Obrázek 63 Správa aukcí

- 1) Výběr statusu aukce
- 2) Zobrazení nabídky přiřazené k aukci
- 3) Ukončení aukce
- 4) Upravení samotné aukce
- 5) Upravení aukce včetně nabídky

9.3.3 Správa uživatelů

Správa uživatelů

Kopírovat Excel CSV PDF
 Hledat:

ID uživatele	Email	Jméno	Příjmení	Třída	Datum registrace	Akce 1 2
0	SYSTEM	SYSTEM	SYSTEM		2023-02-06 14:23:07	
19	test@test.cz	Janička	Nováková	I1B	2022-02-27 15:13:04	
20	zabanovanej@email.cz	Nějakaj	Kořen	I1C	2022-04-03 19:39:17	

Obrázek 64 Správa uživatelů

- 1) Zablokování uživatele
- 2) Odblokování uživatele

10. Závěr

Závěrem můžu říct, že mě maturitní práce bavila. Aplikace možná najde i praktické využití přímo ve škole. Ovládnutí aplikace je intuitivní a případně s ovládnutím pomůže návod. Při vytváření webové aplikace jsem se naučil pracovat s některými knihovnami co se týče *Javascriptu* a taky jsem si osvojil více samotný programovací jazyk *PHP*. Když se ohlédnu zpět, možná bych pro aplikaci nyní zvolil jiné technologie. Například pro samotnou burzu na *frontendu* by mohlo být vhodnější použití knihovny *React* jelikož je uzpůsobená na aplikace s často měněným obsahem. Taky bych pro *backend* použil již kompletní *frameworkové* řešení jako třeba *Laravel*.

11. Použité zdroje

- 1) Password hashing: Use bcrypt instead of Argon2. In: Reddit [online]. 2020 [cit. 2023-03-13]. Dostupné z: https://www.reddit.com/r/PHP/comments/gjhr4/password_hashing_use_bcrypt_instead_of_argon2
- 2) Php shell_exec with realtime updating. In: Stackoverflow [online]. 2019 [cit. 2023-03-13]. Dostupné z: <https://stackoverflow.com/questions/8370628/php-shell-exec-with-realtime-updating>
- 3) How to get the client IP address in PHP. In: Stackoverflow [online]. 2011 [cit. 2023-03-13]. Dostupné z: <https://stackoverflow.com/questions/3003145/how-to-get-the-client-ip-address-in-php>
- 4) JQuery .load() with fadeIn effect. In: Stackoverflow [online]. 2015 [cit. 2023-03-13]. Dostupné z: <https://stackoverflow.com/questions/9337220/jquery-load-with-fadein-effect>
- 5) How to load all the images from one of my folder into my web page, using JQuery/Javascript. In: Stackoverflow [online]. 2022 [cit. 2023-03-13]. Dostupné z: <https://stackoverflow.com/questions/18480550/how-to-load-all-the-images-from-one-of-my-folder-into-my-web-page-using-jquery>
- 6) Scroll to bottom of Div on page load (jQuery). In: Stackoverflow [online]. 2013 [cit. 2023-03-13]. Dostupné z: <https://stackoverflow.com/questions/10503606/scroll-to-bottom-of-div-on-page-load-jquery>
- 7) Scroll to bottom of div?. In: Stackoverflow [online]. 2009 [cit. 2023-03-13]. Dostupné z: <https://stackoverflow.com/questions/270612/scroll-to-bottom-of-div>
- 8) Capturing text between square brackets in PHP. In: Stackoverflow [online]. 2013 [cit. 2023-03-13]. Dostupné z: <https://stackoverflow.com/questions/10104473/capturing-text-between-square-brackets-in-php>
- 9) Validation: Rules for General Use. In: Codeigniter4 [online]. 2023 [cit. 2023-03-13]. Dostupné z: <https://codeigniter4.github.io/userguide/libraries/validation.html#rules-for-general-use>

- 10) MVC - Jednoduchý redakční systém v PHP objektově - Online kurz: Lekce 3 - Směrovač (router). In: Itnetwork [online]. 2023 [cit. 2023-03-13]. Dostupné z: <https://www.itnetwork.cz/php/mvc>
- 11) Creating Your First Application. In: Socketo [online]. 2023 [cit. 2023-03-13]. Dostupné z: <http://socketo.me/docs/hello-world>
- 12) Step-by-step guide. In: Chartjs [online]. 2023 [cit. 2023-03-13]. Dostupné z: <https://www.chartjs.org/docs/latest/getting-started/usage.html>
- 13) PHP Mysql Chat Application with WebSocket. In: Youtube [online]. 2023 [cit. 2023-03-13]. Dostupné z: <https://youtube.com/playlist?list=PLxl69kCRkiI0U4rM9RA1VBah5tfU26-Fp>
- 14) Array_filter. In: Php.net [online]. 2023 [cit. 2023-03-13]. Dostupné z: <https://www.php.net/manual/en/function.array-filter.php>
- 15) Array_map. In: Php.net [online]. 2023 [cit. 2023-03-13]. Dostupné z: <https://www.php.net/manual/en/function.array-map.php>
- 16) Popen. In: Php.net [online]. 2023 [cit. 2023-03-13]. Dostupné z: <https://www.php.net/manual/en/function.popen.php>
- 17) Realpath. In: Php.net [online]. 2023 [cit. 2023-03-13]. Dostupné z: <https://www.php.net/manual/en/function.realpath.php>
- 18) Shell_exec. In: Php.net [online]. 2023 [cit. 2023-03-13]. Dostupné z: <https://www.php.net/manual/en/function.shell-exec.php>
- 19) Array_column. In: Php.net [online]. 2023 [cit. 2023-03-13]. Dostupné z: <https://www.php.net/manual/en/function.array-column.php>
- 20) Array_values. In: Php.net [online]. 2023 [cit. 2023-03-13]. Dostupné z: <https://www.php.net/manual/en/function.array-values.php>

- 21) How to run a background process by PHP in windows?. In: Stackoverflow [online]. 2021 [cit. 2023-03-13]. Dostupné z: <https://stackoverflow.com/questions/66364984/how-to-run-a-background-process-by-php-in-windows>
- 22) Which is best array_search or in_array?. In: Stackoverflow [online]. 2011 [cit. 2023-03-13]. Dostupné z: <https://stackoverflow.com/questions/4518404/which-is-best-array-search-or-in-array>
- 23) How to use PHP in_array with associative array?. In: Stackoverflow [online]. 2014 [cit. 2023-03-13]. Dostupné z: <https://stackoverflow.com/questions/21809116/how-to-use-php-in-array-with-associative-array>
- 24) How can I reset the dropzone in this code?. In: Stackoverflow [online]. 2014 [cit. 2023-03-13]. Dostupné z: <https://stackoverflow.com/questions/21702526/how-can-i-reset-the-dropzone-in-this-code/45247184#45247184>
- 25) Scrolling Overflowed DIVs with JavaScript. In: Stackoverflow [online]. 2009 [cit. 2023-03-13]. Dostupné z: <https://stackoverflow.com/questions/13362/scrolling-overflowed-divs-with-javascript>
- 26) Plain JavaScript - ScrollIntoView inside Div. In: Stackoverflow [online]. 2018 [cit. 2023-03-13]. Dostupné z: <https://stackoverflow.com/questions/45408920/plain-javascript-scrollintoview-inside-div>
- 27) Get First Option Text on Button Click JQuery. In: Stackoverflow [online]. 2022 [cit. 2023-03-13]. Dostupné z: <https://stackoverflow.com/questions/67146941/get-first-option-text-on-button-click-jquery>
- 28) JQuery Datatables Search only one column. In: Stackoverflow [online]. 2016 [cit. 2023-03-13]. Dostupné z: <https://stackoverflow.com/questions/33135047/jquery-datatables-search-only-one-column>
- 29) How To Unzip Files On Linux. In: Ezyzip [online]. 2023 [cit. 2023-03-13]. Dostupné z: <https://www.ezyzip.com/how-to-unzip-files-linux.html>
- 30) I4C OPS cvičení 22/23. In: Google Classroom [online]. 2023 [cit. 2023-03-13]. Dostupné z: <https://classroom.google.com/u/1/c/NTI0OTkyMDkyMTE2>

- 31) Difference between nohup, disown and &. In: StackExchange [online]. 2011 [cit. 2023-03-13]. Dostupné z: <https://unix.stackexchange.com/questions/3886/difference-between-nohup-disown-and>
- 32) [SOLVED] mysql-server (VPS). In: Debian User Forums [online]. 2022 [cit. 2023-03-13]. Dostupné z: <https://forums.debian.net/viewtopic.php?t=151504>
- 33) Phpmyadmin "Not Found" after install on Apache, Ubuntu. In: Stackoverflow [online]. 2015 [cit. 2023-03-13]. Dostupné z: <https://stackoverflow.com/questions/26891721/phpmyadmin-not-found-after-install-on-apache-ubuntu>
- 34) Cannot enter phpmyadmin as root (MySQL 5.7). In: Askubuntu [online]. 2017 [cit. 2023-03-13]. Dostupné z: <https://askubuntu.com/questions/763336/cannot-enter-phpmyadmin-as-root-mysql-5-7/763359#763359>
- 35) How To Install the Apache Web Server on Debian 11. In: DigitalOcean [online]. 2022 [cit. 2023-03-13]. Dostupné z: <https://www.digitalocean.com/community/tutorials/how-to-install-the-apache-web-server-on-debian-11>
- 36) Debugging Apache on Debian, where are the error logs?. In: SuperUser [online]. 2011 [cit. 2023-03-13]. Dostupné z: <https://superuser.com/questions/173833/debugging-apache-on-debian-where-are-the-error-logs>
- 37) 404 Not Found The requested URL was not found on this server. In: Stackoverflow [online]. 2014 [cit. 2023-03-13]. Dostupné z: <https://stackoverflow.com/questions/18853066/404-not-found-the-requested-url-was-not-found-on-this-server>
- 38) How to Install PHP 8 on Ubuntu 20.04. In: Linuxize [online]. 2020 [cit. 2023-03-13]. Dostupné z: <https://linuxize.com/post/how-to-install-php-8-on-ubuntu-20-04/>
- 39) How to run URL from cronjob at specific time in Linux. In: DevArticles.In [online]. 2022 [cit. 2023-03-13]. Dostupné z: <https://devarticles.in/how-to-run-url-from-cronjob-at-specific-time-in-linux/>
- 40) How do I use curl in a cron job? In: Serverfault [online]. 2012 [cit. 2023-03-13]. Dostupné z: <https://serverfault.com/questions/299287/how-do-i-use-curl-in-a-cron-job>

Knihovny

<https://packagist.org/packages/workerman/phpsocket.io>

<https://packagist.org/packages/workerman/phpsocket.io-emitter>

<https://packagist.org/packages/choden/ratchet>

<https://packagist.org/packages/ratchet/pawl>

<https://michalsnik.github.io/aos/>

<https://www.chartjs.org>

<https://confettijs.org/>

<https://datatables.net/download>

<https://www.dropzone.dev/>

<https://jquery.com/download/>

<http://jqueryui.com>

<https://www.npmjs.com/package/natural>

<https://sweetalert2.github.io/>

<https://swiperjs.com>

<https://gijsroge.github.io/tilt.js/>

<https://animate.style/>

<https://getbootstrap.com/>

<https://fontawesome.com>

<https://startbootstrap.com/theme/sb-admin-2>