**Имена:**  Божидара Петрова Пачилова **фн:**  *62172***Начална година:** *2017* **Програма:** бакалавър, (СИ) **Курс: 3**   
**Тема: 42 - Редактор и генератор на учебна програма във ФМИ**

**Дата: 2020-07-04 Предмет: w14prj\_SI \_final имейл: bozhidara.p@abv.bg**

**преподавател:** доц. д-р Милен Петров

# ТЕМА: Редактор и генератор на учебна програма във ФМИ – FMI curriculums

## Условие

Редактор и генератор на учебна програма във ФМИ и съставяне на граф на зависимостите между дисциплините. (например: https://intranet.fmi.uni-sofia.bg/index.php/s/fI5QKFCi1xSQKoQ)

- съдържа списък от елементи около 3 таблици + няколко списъка + различни параметри на плана;

-да се добави и таблица за зависимости (от една страна този предмет от кои предмети зависи, и от друга - кои предмети зависят от този - и да се генерира визуализация).

След проведени консултации с доц. д-р Милен Петров по отношение на заданието извлякохме и следните допълнителни **функционални изисквания**:

1. Информацията за дисциплина се налива като ръчно изготвен JSON файл, чиито полета са по наша преценка. (За краткост няма да бъдат изброявани тук).
2. Информацията за дисциплина, под формата на JSON се вмъква (import) от html текстово поле (text area).
3. Информацията за дисциплина се редактира аналогично на вмъкването – нисък приоритет.
4. Да бъде налична функционалност за търсене.
5. Изтегляне (export) на информацията за дисциплина (JSON формат).
6. Списък с учебни програми.
7. Стилизирана визуализация на учебен план/дисциплина в различни режими на преглед:
   1. кратък – най-основна информация;
   2. подробен (+теми, конспект, библиография);
   3. служебен (включващ произволна мета информация като дата на приемане, размер на файл и т.н.);
   4. връзка с други дисциплини – кои зависят от дадена дисциплина и от кои същата зависи. Визуализация в табличен вид или по възможност като граф.
8. Authentication & Authorization
9. Контрол на достъпа до различните изгледи в зависимост от типа потребител:
   1. Обикновен потребител
   2. Администратор

## 2. Въведение

Проектът представлява малка система, която приема, съхранява и визуализира информация за дисциплини и учебни планове във ФМИ. Тя предлага една алтернатива на сивите PDF файлове, чрез които в момента се представят дисциплините във факултета.

## 3. Теория

От значителна важност в една по-сложна от конзолна програма система е структурата и организираността на кода. Един от най-известните шаблони на проектирането за уеб-базирани системи е **MVC (Model-View-Controller)**. Проектът FMI curriculums е изграден върху ръчно изградена рамка на PHP, действаща като MVC шаблон. Идеята за този “custom framework” е заимствана от източник [[1].](#Източник1) (Описание на структурата на кода [– т. 8](#Точка8))

Моделът е клас, представляващ дадена същност в системата. Комуникира с контролера и си взаимодейства с базата данни. За постигане на по-високо ниво на абстракция за всеки модел се създава инстанция на класът **libraries/Database.php** класът, където са формулирани най-базовите **PDO** заявки към базата данни. Моделите в системата са класовете **Discipline, Curriculum** и **User** в папка **app/models**. Именно това са основните абстракции в проекта.

Изгледите (**app/views**) представляват интерфейса на системата. Получават (динамична) информация от контролера.

Контролерът получава информация от **url** адреса, формите, изгледите. Обработва заявки и получава информация/изпраща от/към модела. Предава информация към изгледите. Всеки модел има съответстващ контролер, разширяващ класа **libraries/Controller.php,** състоящ се от два публични метода, зареждащи съответния модел и изглед.

## 4. Използвани технологии

Проектът е реализиран като уеб система с **HTML, CSS, PHP**, **MySQL** и **JavaScript** . Не са използвани външни библиотеки.

Основна роля играе файловият формат **JSON**. От **JSON** файл се прочита цялата информация за дисциплини и учебни планове, обработва се от съответния контролер и бива запазвана в базата данни.

За осъществяване на различните режими на преглед за дисциплина е приложено асинхронно зареждане без презареждане на страницата посредством **AJAX.**

**XAMPP** e използваният за разработка и тестване локален сървър.

## 5. Инсталация и настройки

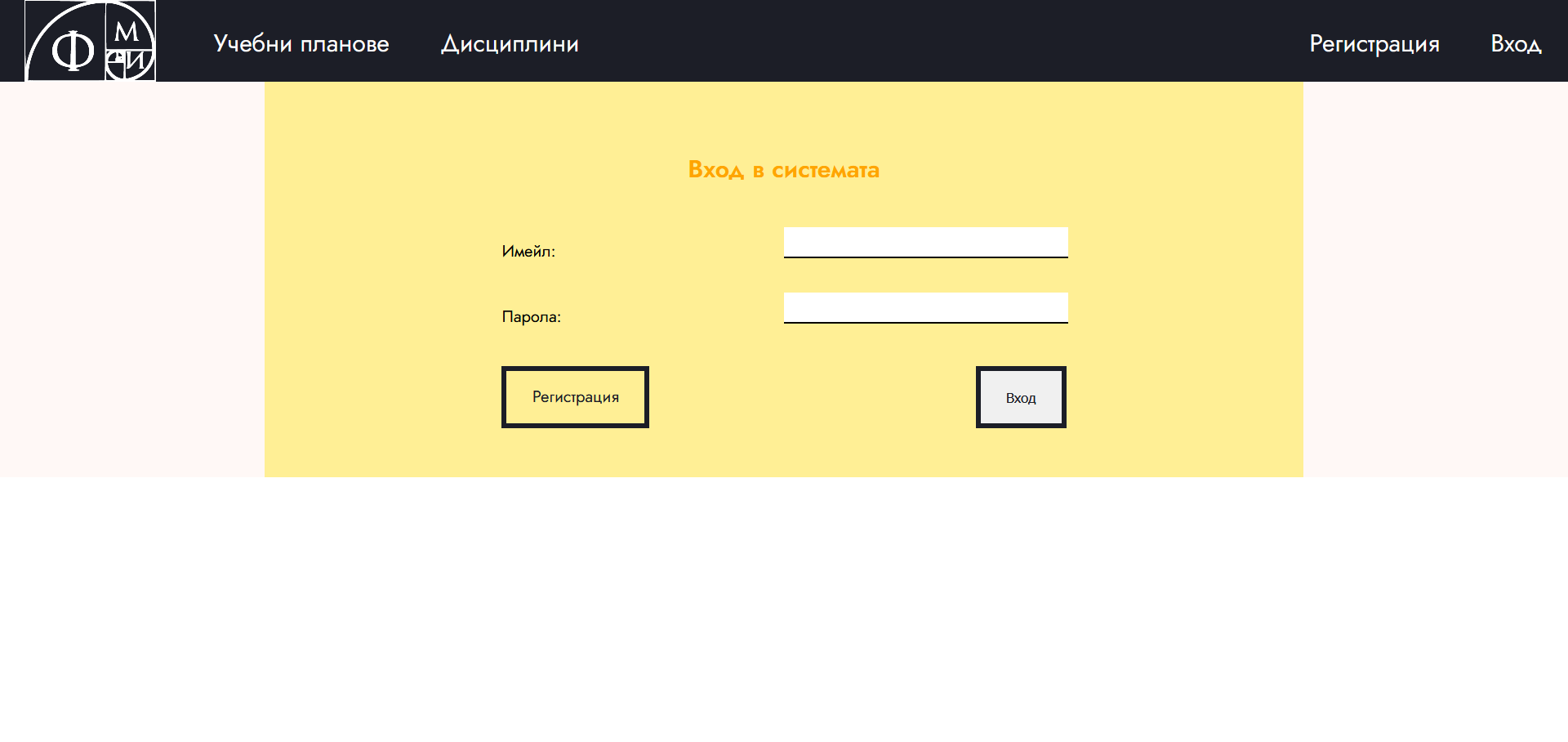
Проектът може да бъде пуснат на всяка машина, която има инсталиран **XAMPP** (Apache + MariaDB + PHP). За целта папката на проекта **FMIcurriculums/** се копира или премества в **xampp/htdocs**.

За допълнителна конфигурация може да се редактира файла **app/config/config.php**. Могат да бъдат настроени *host, user, database name, database password, път на проектната папка на машината, url* адрес, име на системата и версия.

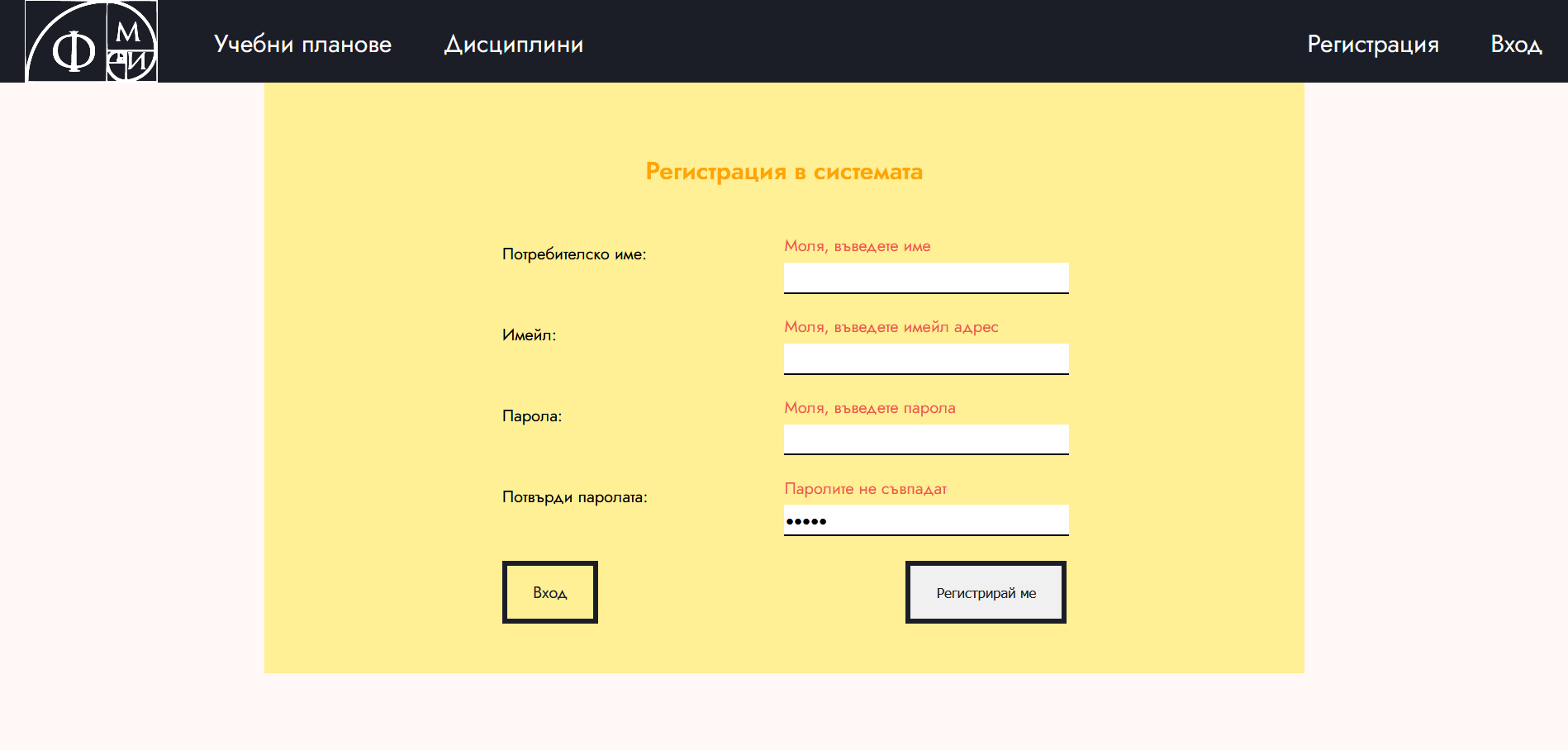
За създаване на базата данни и наливане на данни в системата се изпълнява скрипта **app/config/sql-script.sql** (виж т. 7) през интерфейса **phpMyAdmin**.

## 6. Кратко ръководство на потребителя

1. Регистрация и вход

Страниците за регистрация и вход съдържат стандартни форми. Достъпни са през съответните бутони в навигационното меню. Данните се валидират и при грешка се изписват необходимите насочващи съобщения.

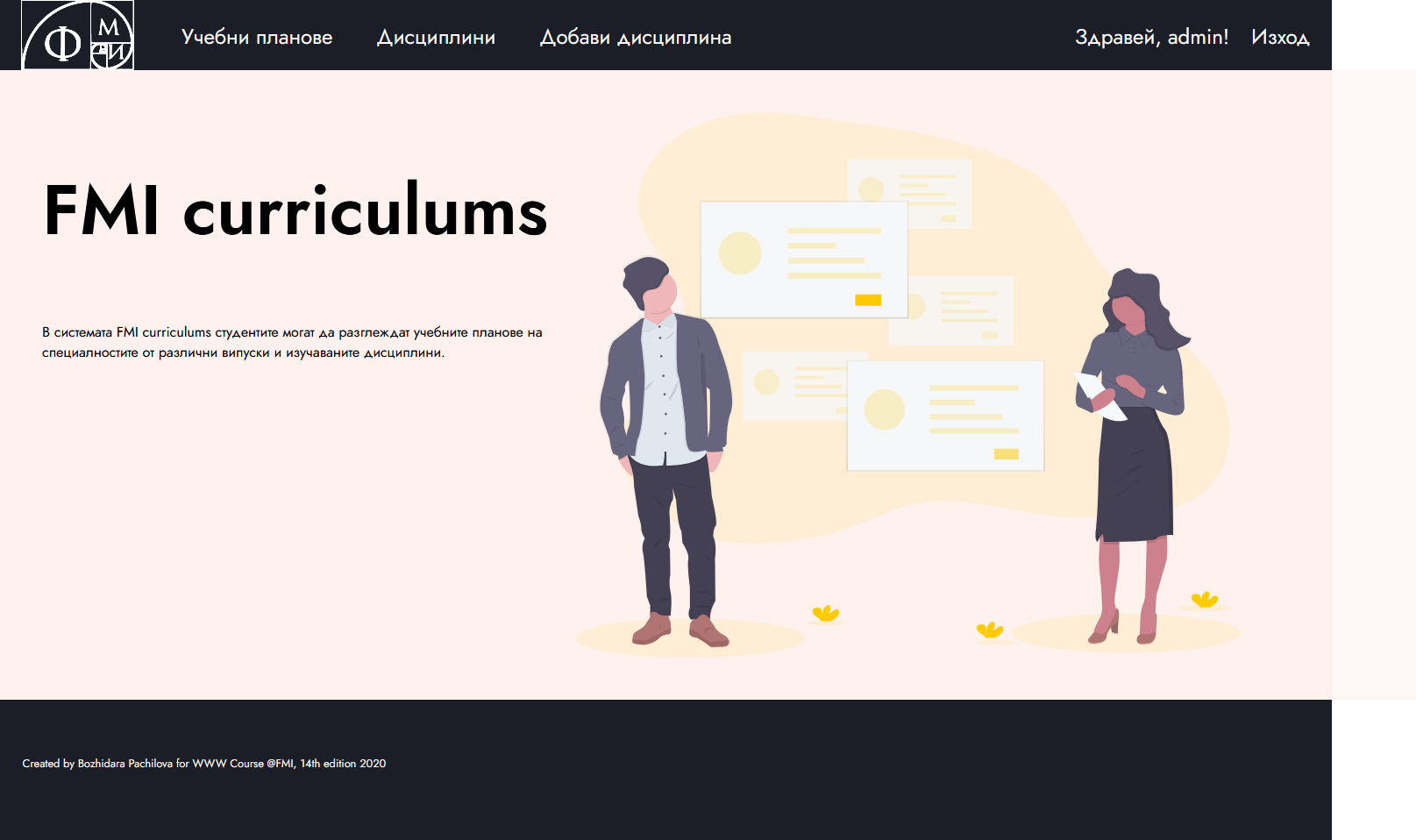
Фиг. 1 – Екран за вход



Фиг. 2 – Екран за регистрация със съобщения за грешки

1. Заглавна страница

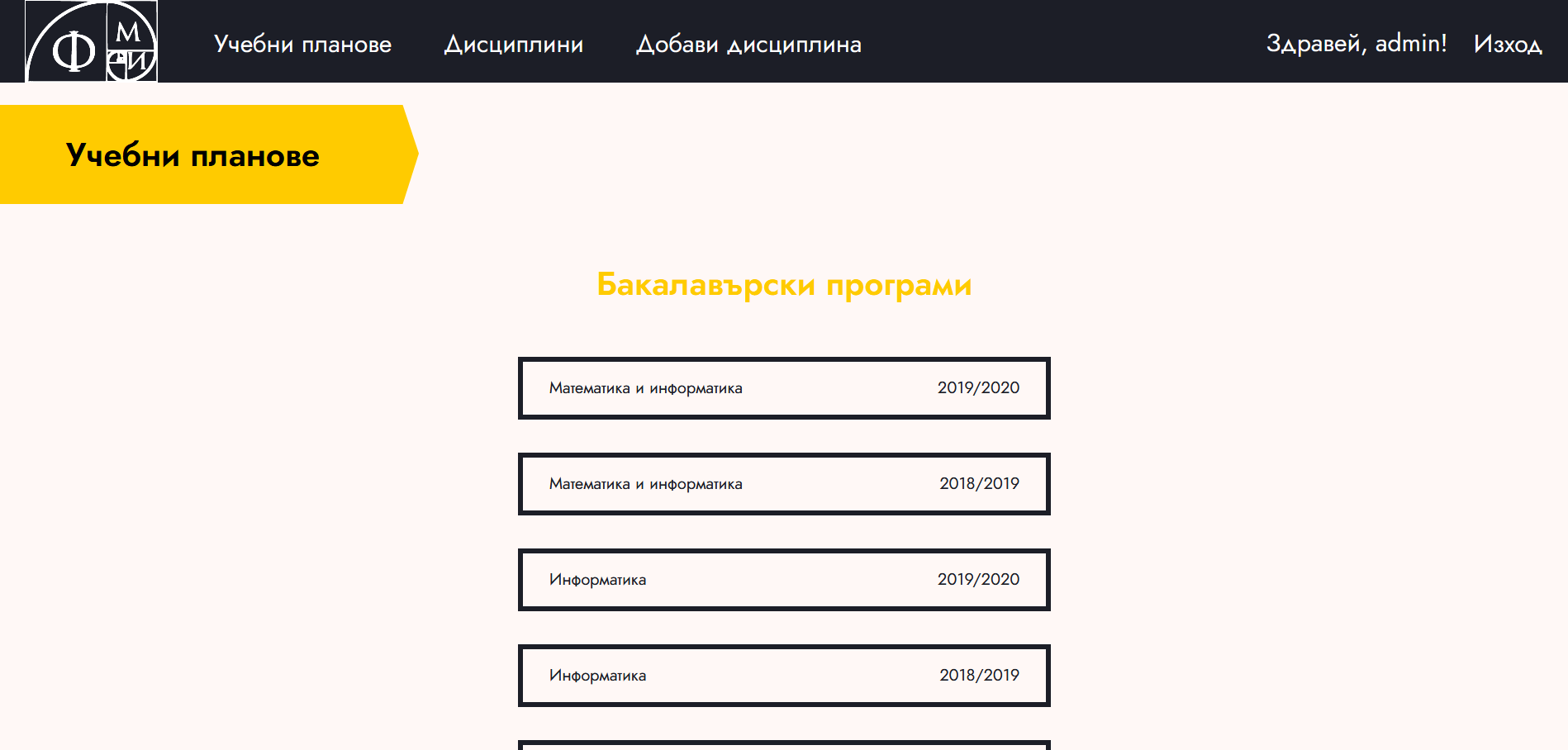
Това е представителната страница (landing page). Когато потребителят е влязъл в акаунта си, в горния десен ъгъл се изписва „Здравей, *потребителско име*“!. В навигационното меню са налични връзките към страниците с учебни планове и дисциплини. Когато потребителят е администратор, има и допълнителна връзка към страницата за добавяне на дисциплина, която само администратор може да достъпи.



Фиг. 3 – Заглавна страница

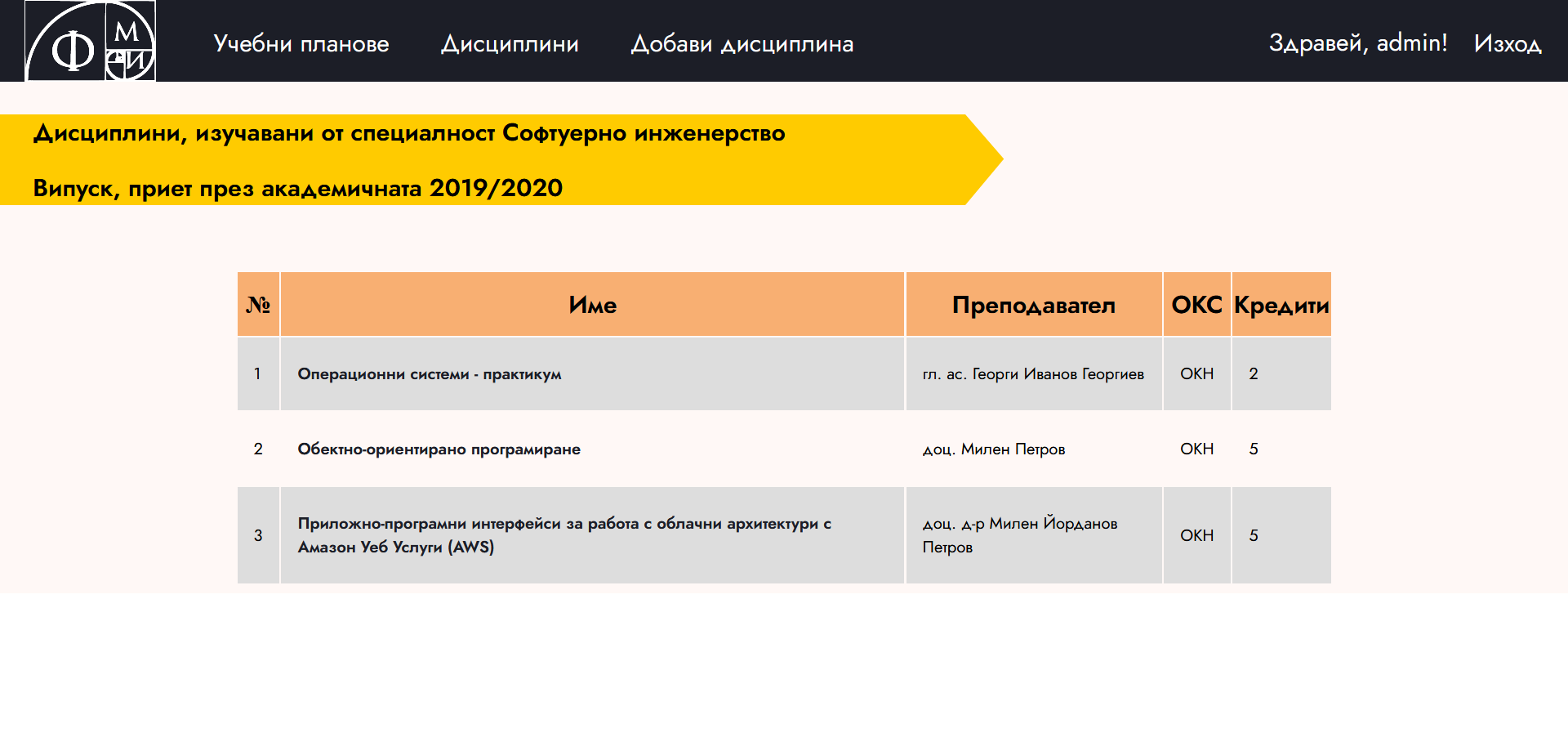
1. Списък с учебни планове по специалности

На тази страница са налични връзки към учебните планове на различните специалности и техните випуски съответно. Разделени са като бакалавърски и магистърски програми.

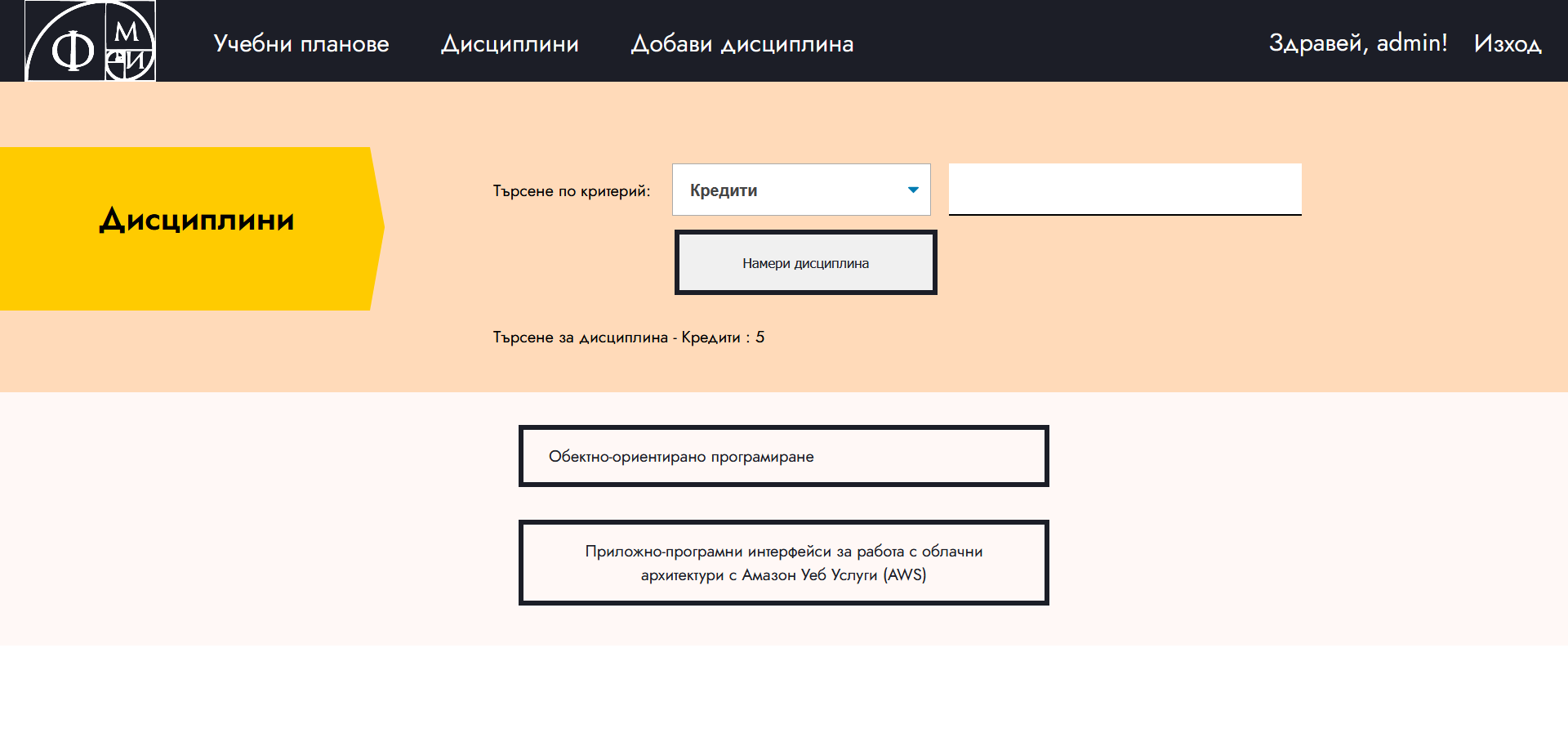


Фиг. 4 – Екран със списък на учебните планове по специалности и години

За всеки учебен план се отваря списък (таблица) с изучаваните дисциплини.



Фиг. 5 – Екран за преглед на дисциплините в учебен план

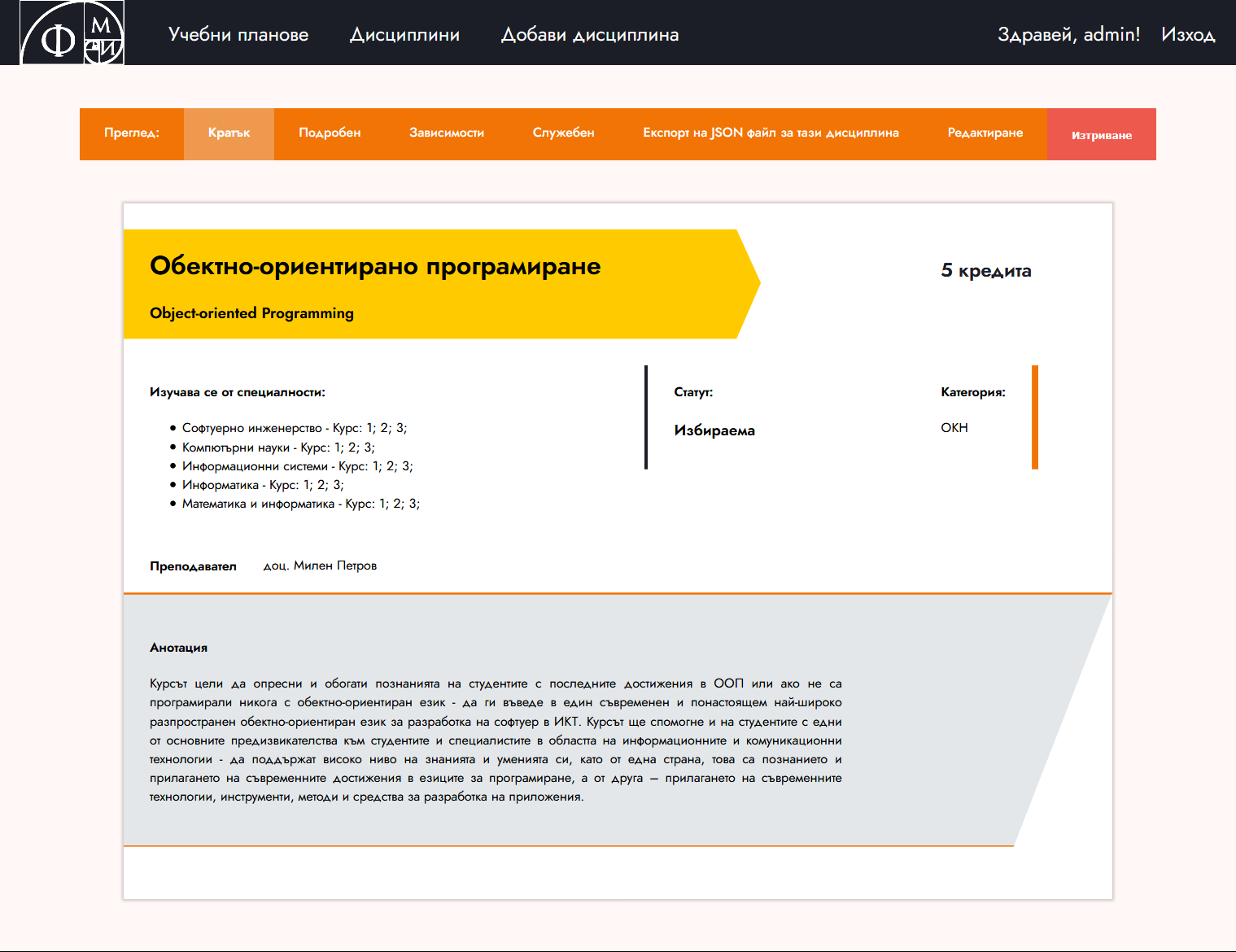
1. Дисциплини
   1. Панелът за дисциплините представя списък от всички налични дисциплини, както и панел за търсене по различни критерии. На екранната снимка е осъществено търсене по филтър „Кредити“ = 5. Заявката за търсене се изписва в панела, а резултатите изпълняват зададеното условие.

Фиг. 6 – Екран за преглед и търсене на дисциплини

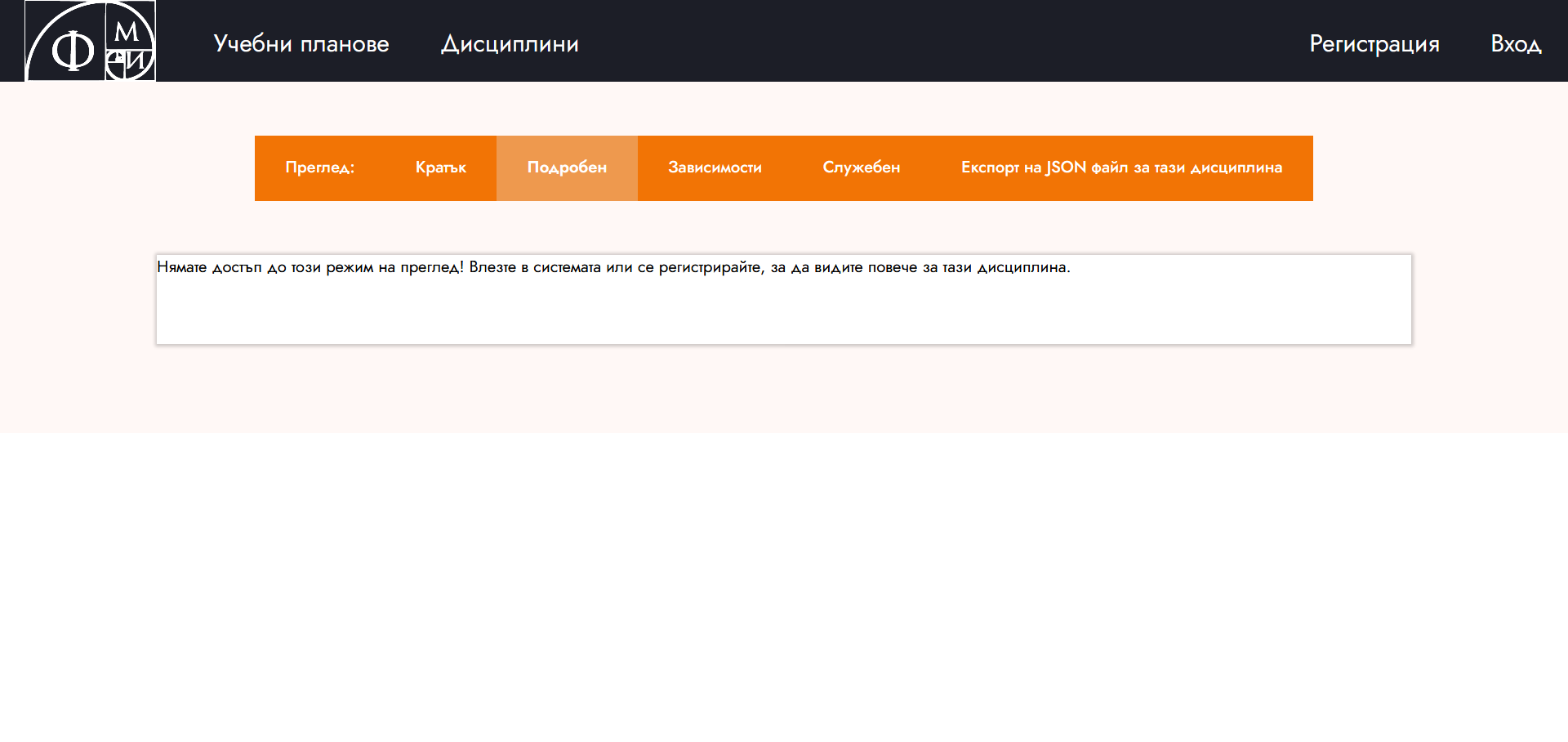
* 1. Преглед

На екранната снимка е показан краткият режим на преглед на дадена дисциплина. Режимите са „Кратък“, „Подробен“, „Зависимости“ и „Служебен“. В режим „Зависимости“ се визуализира таблица с връзки към зависещите от тази дисциплина други дисциплини и обратното – от кои предходни дисциплини зависи настоящата.

В оранжевото меню са достъпни също функционалностите за експорт на JSON файл за тази дисциплина, редактирането й и изтриването й от системата. Последните две са достъпни само за администратори.

Фиг. 7 – Екран с кратък режим на преглед на дисциплина

За невлязъл потребител режимите „Подробен“ и „Зависимости“ не са достъпни. Режимът „Служебен“ е достъпен само за администратори.



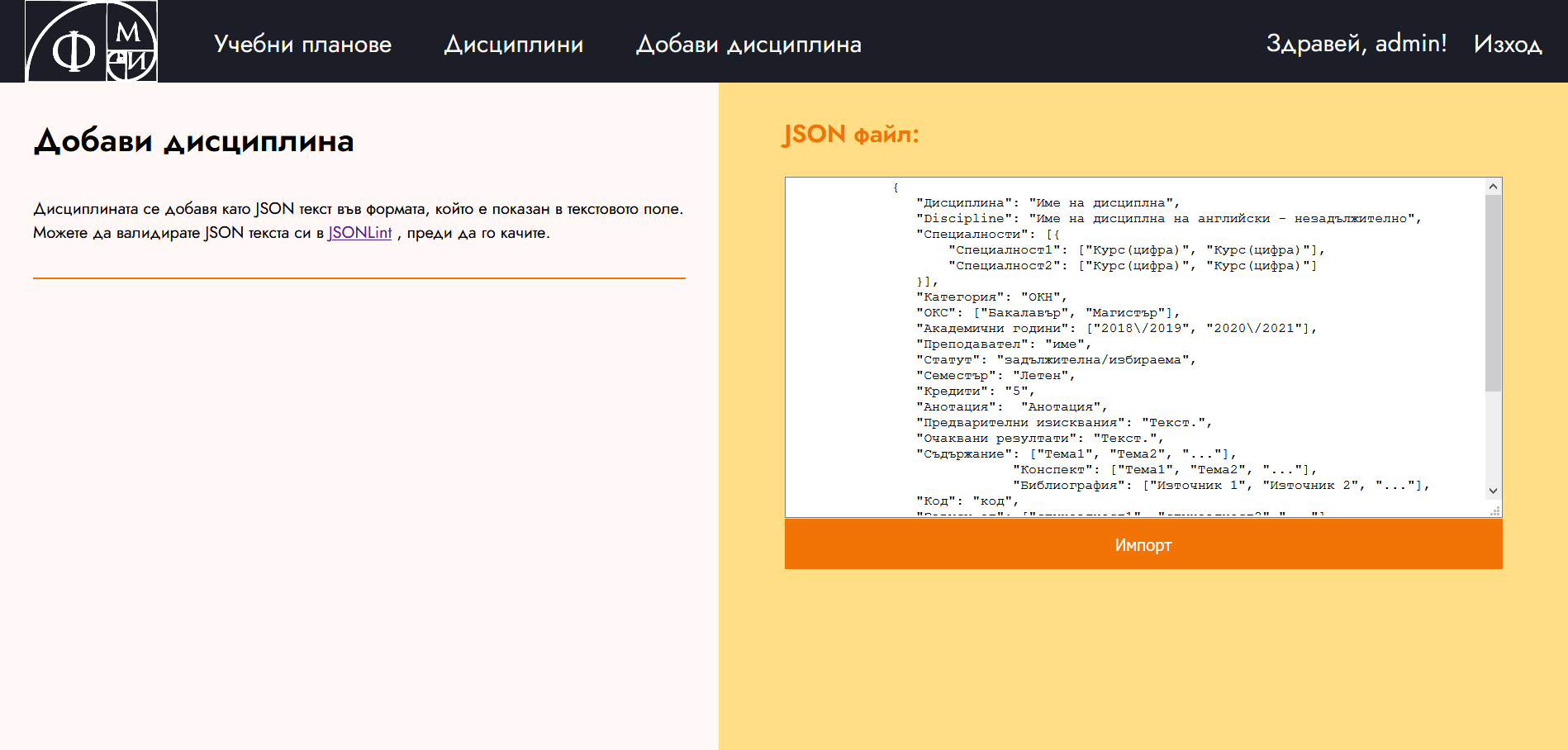
Фиг. 8 – Екран, показващ съобщение за липса на достъп за неавтентикиран потребител

* 1. Добавяне на дисциплина

Добавянето на дисциплина се случва като се постави съдържанието на JSON файл в текстовото поле. По подразбиране в полето се визуализира JSON с фиктивни данни за улеснение на потребителя. По този начин, той би могъл директно да замества с конкретни стойности, спазвайки JSON синтаксиса.

*Ако потребителят желае в някой от по-дългите текстове като „Анотация“ или „Очаквани резултати“, например, да се визуализира нов ред, то трябва в JSON файла да използва стринга* **\\n**

Страницата за редактиране на дисциплина е аналогична, но в текстовото поле се визуализира самият JSON файл за дадената дисциплина, чиито стойности на полетата могат да се променят.

Фиг. 9 – Екран за импорт на дисциплина

## 7. Примерни данни

Скриптът, създаващ базата данни се намира в **app/config/sql-script.sql.** Чрез този скрипт автоматично се наливат данни за 3 дисциплини в системата като за тях предварително съществуват и JSON файлове в папката **public/JSONS**. (По принцип JSON фаловете се запазват при импорт през системата).

За тестване на импортването на дисциплина e предоставен JSON файла **testImportJSON.json** в **app/config/,** който следва да се import-не през системата.

По default в системата има двама потребители – администратор и един обикновен потребител:

Потребителско име: **admin**

Имейл адрес: [**admin@fmi.uni-sofia.bg**](mailto:admin@fmi.uni-sofia.bg)

Парола: **123456** (в БД се съхранява хеширана).

Потребителско име: **regular\_user**

Имейл адрес: [**regular\_user@abv.bg**](mailto:regular_user@abv.bg)

Парола: **123456** (в БД се съхранява хеширана).

Администратор може да се добавя през базата данни, в таблицата **users**, като му се зададе стойност на полето role – „admin“.

## 8. Описание на програмния код

**app/ -** съдържа основната „сървърна“ логика и изгледите;

**config/**

**config.php –** файл с конфигурационни данни. От него могат да бъдат настроени *host, user, database name, database password, url* адреси, име на системата и версия.

**sql-script.sql**

**testImportJSON.json**

**controllers/ -** папка на контролерите. Съдържа **Curriculums.php, Disciplines.php, Users.php** и **DefaultController.php**;

**helpers/ -** папка с помощен php код

**session\_helper.php –** стартира потребителската сесия; код за т.нар. „флаш съобщения“ при грешка/успех при вход или регистрация; метод за проверка дали потребителят е влязъл в акаунт;

**url\_helper.php -** съдържа единствена функция *redirect($page),* задаваща location header;

**libraries/ -** папка, съдържаща php класове, които представляват най-ниското ниво на абстракция в системата – т.нар. „ядро“ и базовите класове за контролер и връзка с базата данни, които ще бъдат разширявани.

**Controller.php –** базовият клас се състои от два публични метода, зареждащи съответния за разширяващия контролер модел и изглед;

**Core.php -** обработва информацията, постъпваща от url адреса, чийто формат е **/controller/method/params** – кой контролер да бъде извикан, кой негов метод и с какви параметри;

**Database.php –** тук са формулирани най-базовите **PDO** заявки към базата данни;

**models/ -** моделите дефинират основните заявки към БД за различните същности (абстракции) в системата. Както вече бе споменато в **т. 3**, те са именно:

**Curriculum.php, Discipline.php, User.php**

**views/ -** изгледи

**default/** : **index.php**

**curriculums/** : **index.php, details.php**

**disciplines/** : **index.php, download.php, edit.phpq import.php, visualise.php;**

**inc/: footer.php, header.php, navbar.php**

**users/: login.php, register.php**

**filesrequire.php –** тук за удобство се „require-ват“ основните файлове *config.php, url\_helper.php* и др., за да се избегне повтаряемост.

**public/ -** съдържа частите от кода, които по принцип няма проблем да бъдат публични. Папката JSONS не попада точно в тази категория, но за удобство при разработката бе поставена тук.

**css/**

**img/**

**js/** : **loadReview.js, search.js**

**JSONS/ -** тук се съхраняват импортнатите JSON файлове; необходимо е да се съхраняват поради възможността да се експортват;

**index.php –** помощен файл, “стартира” системата;

## 9. Приноси на студента, ограничения и възможности за бъдещо разширение

Проектът е разработен самостоятелно и поради това притежава до известна степен ограничени функционалности, но удовлетворява посочените базови изисквания.

Някои възможности за последващо развитие са:

* При предварителното проучване бе обсъдена идея за интеграция със системата „FileMeUp“. На този етап работа в тази насока не е извършена, тъй като авторът на проекта не е запознат с интерфейсите на „FileMeUp“;
* Към момента импортването на JSON файл е през текстово поле. В бъдеще би могло да се качва (upload) JSON файл от машината – през тази или друга система (напр. „FileMeUp“);
* Допълнителна работа по интерфейса. Да се направи responsive (чрез media queries, изцяло друг мобилен дизайн и т.н.);
* Да се оптимизира CSS-а като цяло - да се изведат CSS глобални променливи, основните класове да бъдат по-детайлно обособени и преизползвани. По този начин може да се въведе „конфигурация“ на визуализацията на дисциплините, която да се задава от потребителя, например.
* Да се добавят нови данни (полета) към необходимата информация за дисциплина – изисква се добавяне на нови колони в БД и структурата на примерния JSON файл.
* Зависимостите между дисциплини да се визуализират като графи. За целта е необходимо запознаването с наличните за целта библиотеки – [списък с такива](https://neo4j.com/developer/tools-graph-visualization/). Проблемът с тях е, че изискват различни и специфични формати и като цяло подходът би бил „първо се избира библиотека, после се реализира структурата на данните“.

## 10. Какво научих

В проекта е приложена на практика теорията и всички технологии, изучавани в курса по “Web технологии”.

Допълнителен опит бе проучването за изискванията, което представляваше своеобразен мини-„бизнес анализ“ – бе проведена кореспонденция с преподавателя и последващи две онлайн консултации.

Затвърдих знанията си относно JSON файловия формат и неговия синтаксис, както и за същността и структурата на MVC, но без готова работна рамка. С SQL и съставянето на схема на БД вече се бях сблъсквала, но го приложих още веднъж.

За пръв път работя с AJAX асинхронни заявки, нещо, което не бях включвала в друг проект досега. Приложих и наученото относно създаването на потребителски сесии.

## 11. Използвани източници

[1] Онлайн курс в Udemy – “Object Oriented PHP & MVC”, автор Brad Traversy, последно обновен през ноември 2017 г.

<https://www.udemy.com/course/object-oriented-php-mvc/>

[2] Страница в MDN web docs – “Working with JSON”, последно обновена на 21.02.2020 г.

<https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Objects/JSON>

[3] Валидатор на JSON файлове, уеб инструмент

<https://jsonlint.com/>

### 12. Редакции след защитата

* В Sql-script.sql вече е нужно да се промени името на създаваната база само на 2 места (create database.. and use..).
* Функционалността за експорт на JSON файл вече е видима само за администратор.
* В config.php, беше възникнал проблем с конфигурацията за пътя на проектната папка, който се използва и в кода. От:

define('APPROOT', dirname(dirname(\_\_FILE\_\_)));

което би трябвало да работи, тъй като взима пътя две папки по назад от директорията на текущия файл (config.php), т.е. config->app

Редакцията се състои в това да се постави ръчно пътя на проектната папка и да се конкатенира с '\app':

define('APPROOT', 'C:\xampp\htdocs\FMIcurriculums' . '\app');

Предал (подпис): ………………………….

*/Ф.Н. 62172, Божидара Петрова Пачилова, СИ, 3 курс, 1 група/*

Приел (подпис): ………………………….

/доц. *Милен Петров*/