**Имена:**  Божидар Димитров Димов (**фн:**  *81112*), Христина Красимирова Тодорова(**фн:**  *81179*) **Начална година:** *2017* **Програма:** бакалавър, (КН) **Курс: 4**   
**Тема: Система за КСК  
Дата: 2018-01-28 Предмет: WEB технологии**

**имейл:** [**bojko958@abv.bg**](mailto:bojko958@abv.bg)**,** [**hrist\_todorova@abv.bg**](mailto:hrist_todorova@abv.bg)

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

# ТЕМА: Система за КСК (тема номер 43)

## 1. Условие

Да се създаде анимирана КСК с държавна поръчка и платено обучение.

## 2. Въведение и теория

Използвахме официалния сайт на СУ Климент Охридски, за да получим информация за квотите и балообразуването на всяка специалност. Алгоритъмът за класиране, който измислихме, е следният: при добавяне на нов кандидат:

1. Взимаме всички студенти, които не са все още класирани от базата(това включва и човека, които току-що сме въвели)

2. Ако има такива – стъпка 3, ако не – край на класирането.

3. За всеки студент:

3.1 Взимаме текущото му най-високо желание(за нов кандадат това е първо желание). Ако сме стигнали вече до последното и не сме се класирали – отпадаме от класацията.

3.2 Ако в съответната специалност има места – студента се класира там, продължаваме със следващия студент, иначе - не правим нищо. Ако няма места, но имаме бал, по-висок от този на последния класирал се студент в тази специалност -> изместваме го, а другият човек се класифицира като некласирал се.

4. След като преминем през този списък проверяване отново 1. (за да класираме изместените от някой друг или некласиралите се на текущо желание студенти по време на точка 3 – проверяваме следващото им желание)

## 4. Използвани технологии

За front-end сме използвали VueJS 2.5 с изцяло правен на ръка CSS. За таблицата с класирането експериментирахме с flex-box. Content-ът на страницата центрирахме по средата чрез auto-margin. Приложението е single page, тоест различните изгледи се зареждат динамично. За целта направихме собствен loader. Недостатъкът е че джаваскриптът не се зарежда динамично, но това може да се подобри чрез webpack конфигурация. Компонентите са енкапсулирани в анонимни обекти. Динамичното обновяване на данните става чрез polling през 5сек. Класирането се пази в таблица, за да не се изчислява всеки път когато поискаме обновяване

Back-end-ът е на PHP5. За достъп до базата данни използваме mysqli. Имаме скриптове за генериране на базата от php, но сме добавили и createAll.sql което е експорт на базата. Приложението има логин и данните се пазят в $\_SESSION.Имаме два конфигурационни файла – един за front-end и един за back-end(по-долу прилагаме структурата на директориите).

## 5. Инсталация и настройки

Изпълняваме

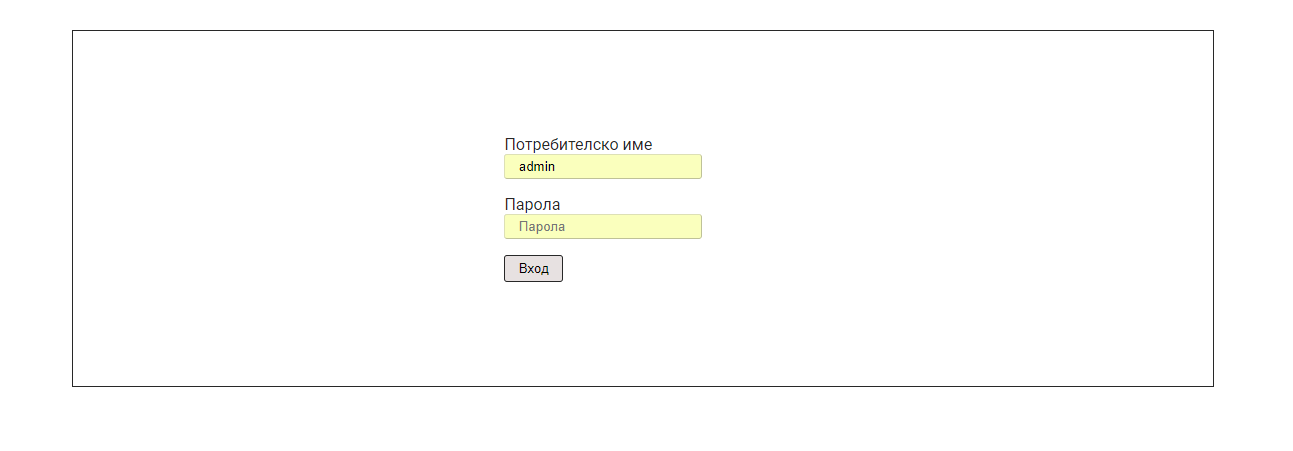
CREATE DATABASE `uni\_ranking` /\*!40100 DEFAULT CHARACTER SET utf8 \*/;

grant all privileges on uni\_ranking.\* to admin@localhost identified by '!QAZ2wsx' with grant option;

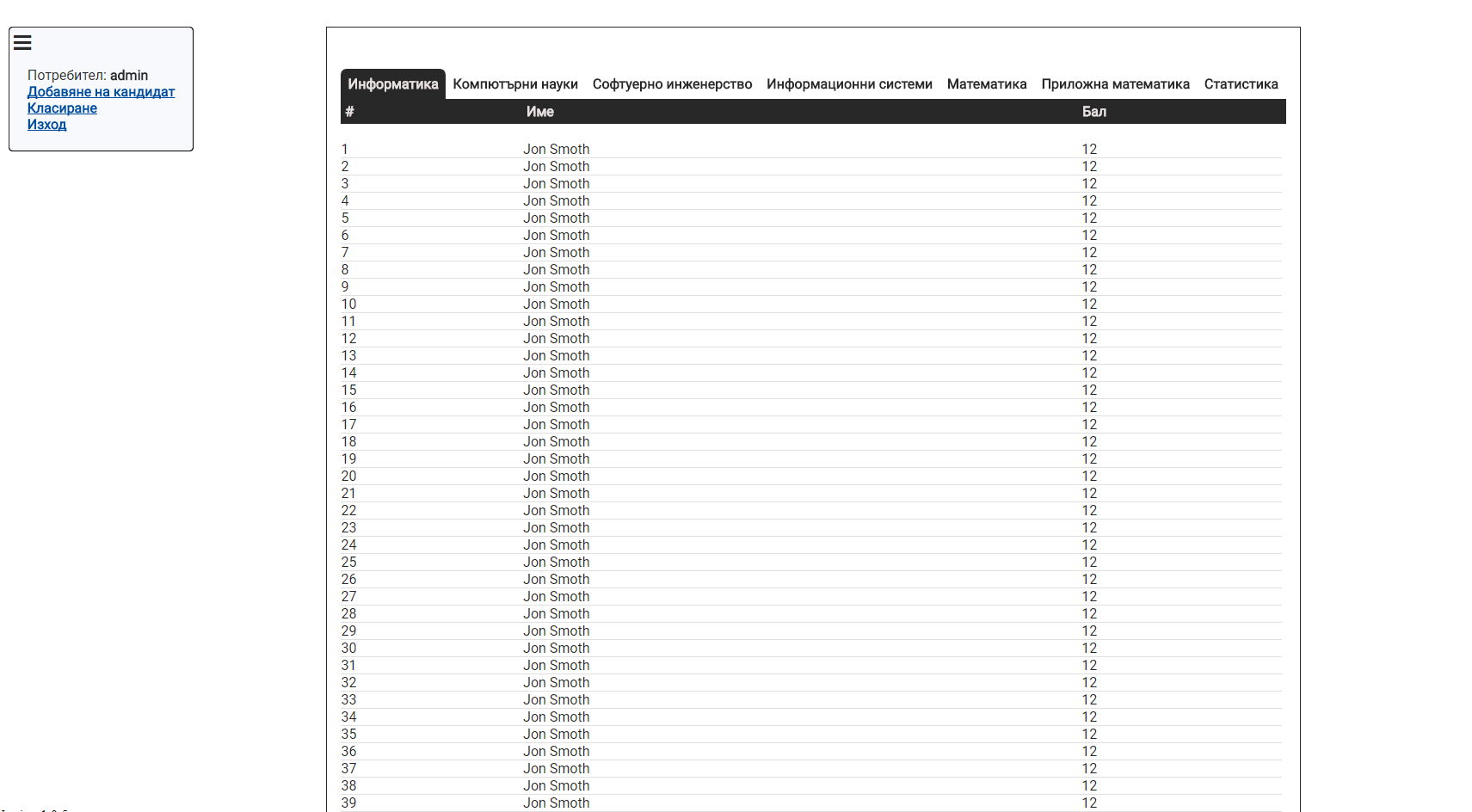
Конфигурацията на базата е в resources/config.php

След това в браузъра навигираме до resources/database/createdb.php. Този файл ще създаде необходимите таблици и изгледи.

## 6. Кратко ръководство на потребителя

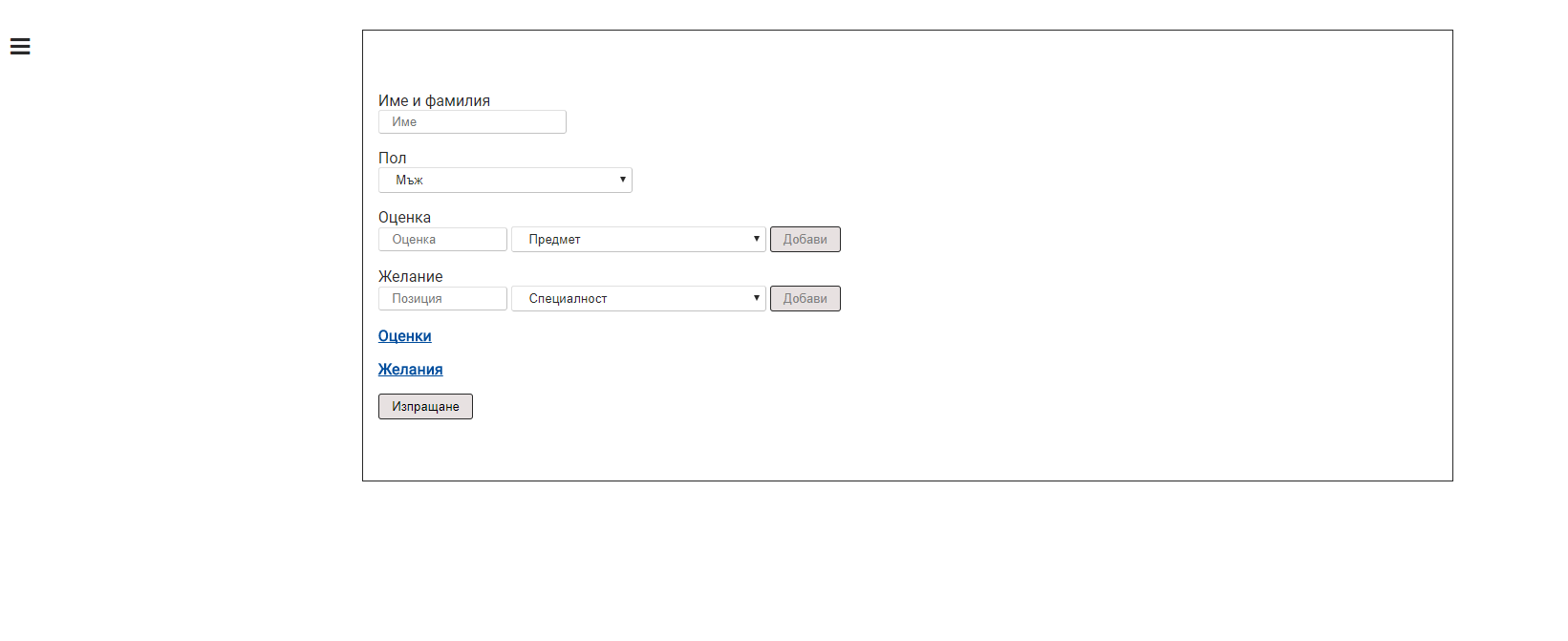


Екран за вход в системата



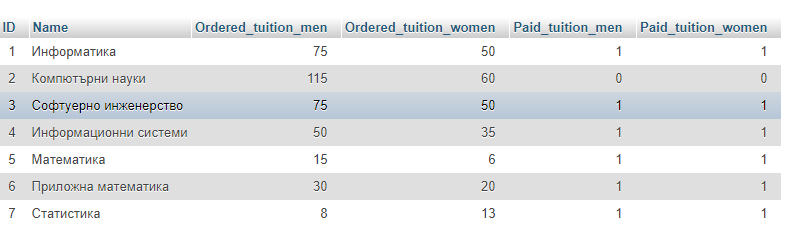
Меню и примерна класация

В черен цвят са кандидат-студентите, класирани за държавна поръчка, а във синьо са тези с платено обучение.

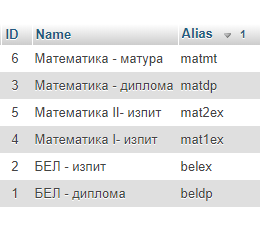


Форма за регистриране на кандидат-студент

## 7. Примерни данни

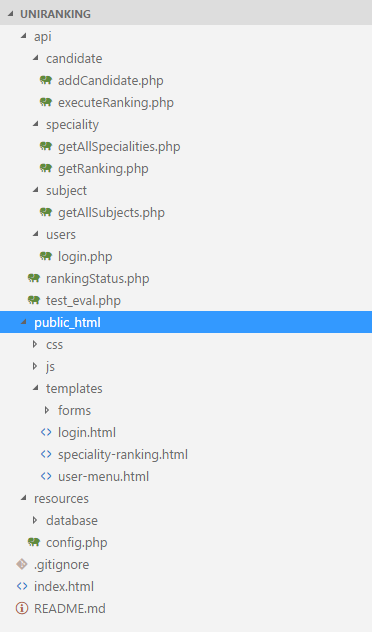


Speciality – таблицата за специалностите по които се кандидатства



Subjects – това са предметите по които се изискват оценки

## 8. Описание на програмния код

Има 3 главни папки – api, public\_html и resources.

В api се намира кодът, който приема api-request-ите и кодът, който извършва класацията на кандидат студентите.

В public\_html се намират html, css и javascript файловете, които са front-end-ът на приложението.

В resources има php файлове, в които се грижат за създаването и конфигурирането на базата от данни.

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

Приложението може да бъде разширено по много начини. В момента то поддържа само класиране на бакалаври – трябва да се добяват и магистри за по-пълен вид. Може да се добавят различни роли на потребителите като например админи и само такива които могат да виждат класирането. Може да се добавят форми за регистрация и за добавяне и промяна на параметрите на всяка специалност. Динамичното обновяване може да се реализира чрез сокети което би го направило в пъти по-оптимално. Бекендът може да се смени на друга платформа(NodeJS или C# например :) ). Стиловете могат да се прегледат и пооправят или да се пренапишат на SASS. Може да се имплементира repository pattern за работа с базата. Също така се разчита, че потребителят въвежда верни данни. За евентуална поправка трябва админ да влезе в базата и да промени данните – може да има форма за редактиране на кандидатура.

## 10. Какво научих

Божидар Димитров Димов - Научих се да работя донякъде с VueJS както и измислих собствен html loader. Подобрих CSS уменията си и работата си в екип

Христина Красимирова Тодорова - Задълбочих познанията си по PHP. Упражних в истински проект наученото по Бази от данни. Научих се да работя с HTTP заявки.

## 11. Използвани източници

[1] https://www.uni-sofia.bg – Балообразуване на специалностите и квоти

Предал: ………………………….

/*фн, имена, спец., група*/

Приел: ………………………….

/доц. *Милен Петров*/