**Имена:**  Георги Светлозаров Стефанов **фн:** *81319***Начална година:** *2015*  **Програма:** бакалавър, (КН) **Курс: 4**   
**Тема: 23.1 Генератор на тест за самопроверка по зададени въпроси към реферат и статистика  
Дата: 2020-01-26 Предмет: Web технологии (w15ed) имейл: geosteffanov@gmail.com**

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

# ТЕМА: 23.1 Генериране на тест за самопроверка по зададени въпроси към реферат и статистика

## Условие

Да се създаде система, която позволява правенето на тест от даден потребител, генерирайки тест за самопроверка по зададени въпроси към реферат. Системата трябва да поддържа и статистика за направените тестове от даден потребител, за отговорените въпроси – съответно правилно или грешно, както и техния брой. Системата да предоставя възможност за добавяне към базата от данни на нова тема, както и нов въпрос по съответна тема. При започването на тест потребителят трябва да има възможност да избере темата, за която да се генерират въпросите за теста, който ще прави, както и да избере типа на тези въпроси за теста, което ще рече да избере дали иска да прави тест ‘за предварителни знания (pre-question)’, ‘по време на презентацията (poll-question)’ или ‘след презентацията (test-your-knowledge)’. Позволява се вход в системата, като за целта всеки потребител въвежда факултетен номер, чрез който бива вписан в системата и използва съответните необходими и предоставени му функционалности.

## Въведение

Вземайки предвид характера на идеята и разработката на системата, може да кажем, че има немалко системи с аналог на нашата. Това, към което тя се цели като функционалност, която да предостави, обаче, е строго ориентирано към генериране на тестове, свързани с рефератите, създавани от студенти в курса Web технологии, воден от доц д-р Милен Петров във ФМИ на СУ „Св. Климент Охридски”. В основата е предоставянето на възможност на даден потребител да проследява как другите потребители отговарят на въпросите, които той е задал към реферата си. Именно за кой въпрос колко верни отговори има, колко са грешните, колко общо пъти е отговорен, имайки предвид и типа на въпроса, който е задал. Това се прави с цел подобряване на самите въпроси, които студентите в курса пишат по темата си, това е една обратна връзка за тях, която би им помогнала да променят, обновят или добавят различни въпроси. Другият момент опира до самопроверката на потребителите, които използват функционалността да правят тип тест по дадена тема. В някакъв обхват това би повишило знанията, които студентите придобиват от запознаването с рефератите на останалите, защото, правейки теста, биха имали по-голяма мотивация да запомнят и проучат определени въпроси. Системата позволява ролята на администратор, за когото има предоставен администраторски панел, чрез който той получава информация и под формата на статистика за това какви тестове са правени и с какви резултати. Има достъп до всички останали функционалности.

## Теория

Използваната архитектура следва до известна степен архитектурния шаблон MVC (Model-view-controller), където се предполага да има обособени съответните компоненти за front-end частта, работата с клиентските заявки и организирането на потребителския интерфейс. В директория api са обособени логиките по отделните елементи – topic, question, test, history и users. Като връзката с базата данни е отделена в utils.php, където чрез определените функции се позволява изпълнението на писане, четене и промяна на записите в базата. Предимството на този модел е по-голямата независимост на отделните части, която се постига, както и по-високата в определени граници, изменяемост на системата.

## Използвани технологии

При разработката на проекта са ползвани **HTML**, **CSS**, **Javascript** и **PHP**. Също така е използвана и библиотека, написана на Javascript, Papa Parse, чрез която да се прочитат CSV файловете, за изпълняване на функционалността import csv за добавяне на въпроси от файл в базата от данни с въпроси. Потребителският интерфейс позволява потребителят да оперира със системата чрез достъпни и разбираеми за обикновен потребител менюта, бутони и визуализации. PHP е използван за сървърната част на системата, основната бизнес логика е реализирана чрез него. Сървърната част предоставя REST api. Използва се за връзка с базата данни, писане и четене от нея, както и промяна. Валидацията на потребителския вход също се случва чрез PHP. Потребителските заявки се изпращат към сървъра асинхронно чрез Javascript. Визуализирането на обратната връзка с потребителя след изпълнение на изпратена от него заявка, се реализира отново чрез Javascript и HTML. За съответните избрани стилове е използван CSS. Използва се приложението XAMPP, което предоставя и осъществява достъп до Apache сървър и позволява правенето на заявки към базата от данни чрез езика MySQL.

* Сървър – **Microsoft Windows 7, x64**
* Сървърен език – **PHP 8**
* Уеб сървър – **Apache 2.4.46**
* Сървър за База данни – **MariaDB 10.4.17**
* **phpMyAdmin – 5.0.4**

## Инсталация и настройки

Предварително е необходимо да бъде инсталиран XAMPP. Когато решим да пуснем даден проект, какъвто проект е предметът на настоящата документация, трябва всички файлове, с които разполагаме(или цялата папка на проект), да бъдат поставени в  C:\xampp\htdocs\ и, когато искаме да заредим в браузъра някоя от страниците, вместо  C:\xampp\htdocs\ пишем localhost. Трябва чрез контролния панел на XAMPP да се стартират модулите Apache и MySQL като се натиснат бутоните start срещу тях.

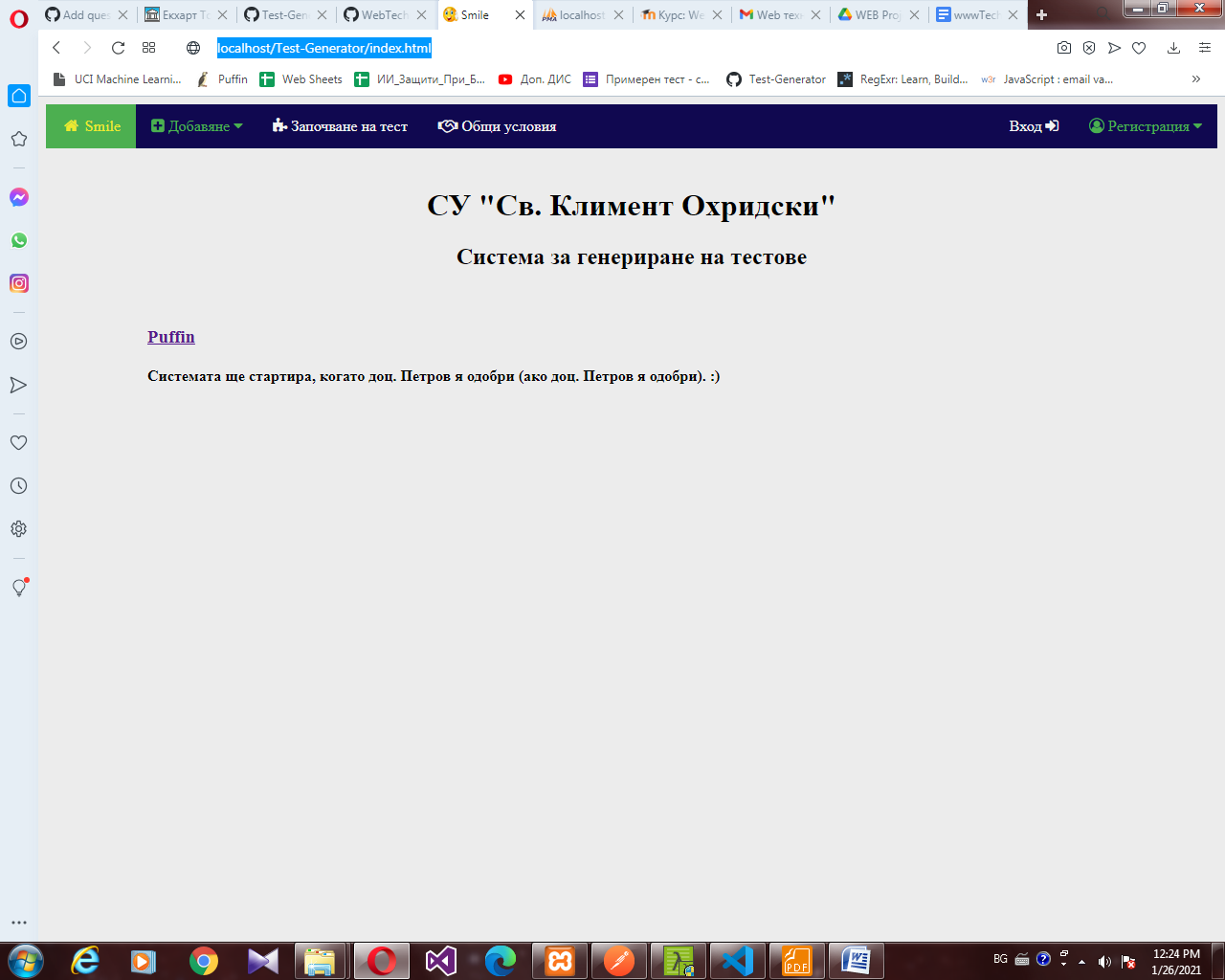
Пример: Ако искаме да достъпим страницата за започване на тест, пишем в браузъра

http://localhost/Test-Generator/pages/Test.html

Другият начин е да отворим началната страница на системата на адрес:

http://localhost/Test-Generator/index.html

и оттам чрез потребителския интерфейс да навигираме до съответните места, които искаме да посетим.



**Фиг.1 Екранна снимка - url за достъп до системата**

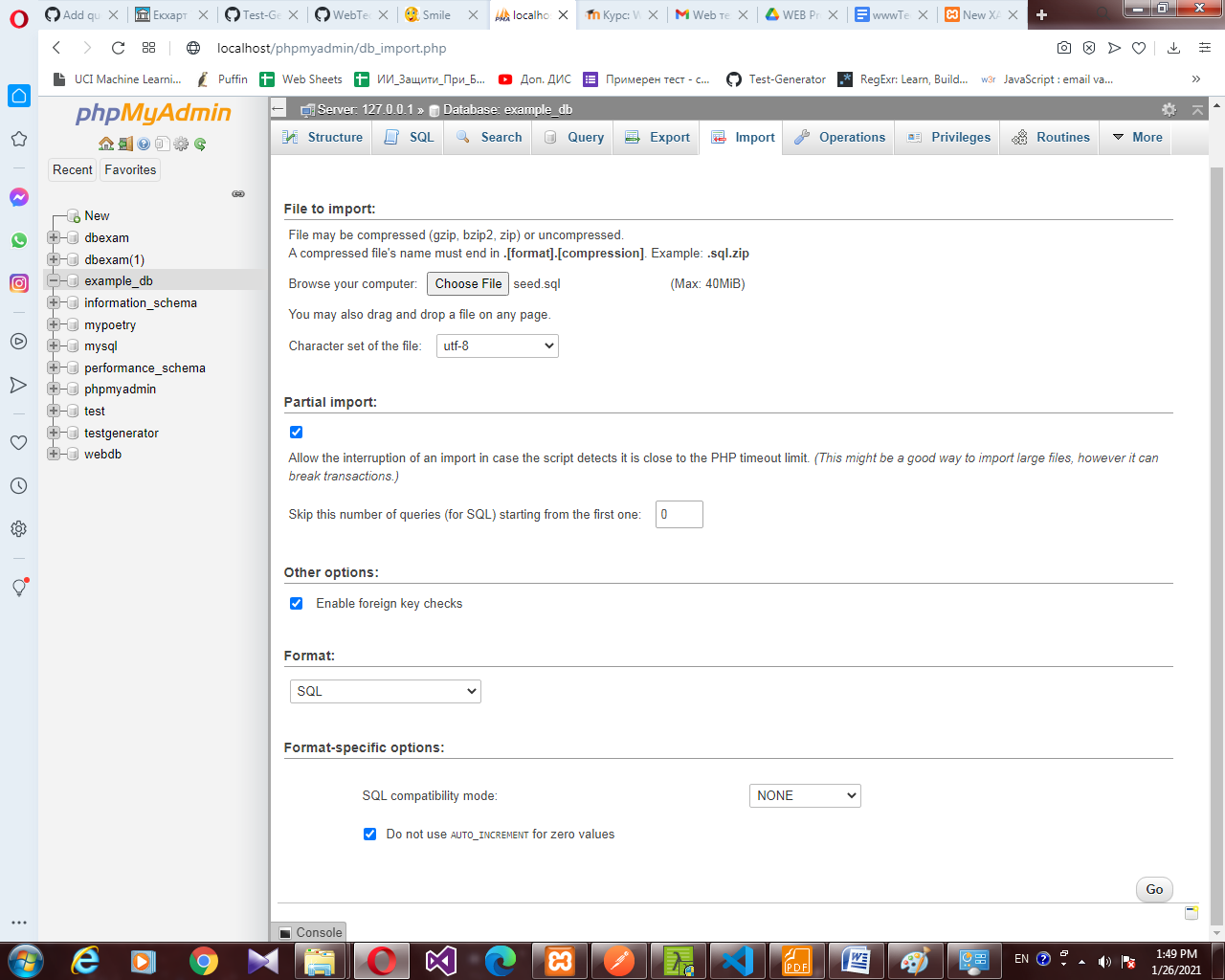
## Кратко ръководство на потребителя

В началото потребителят стартира **XAMPP (C:\xampp\ xampp-control.exe)** и натиска **start** бутоните срещу модулите **Apache** и **MySQL**. След това достъпва през браузъра следния адрес:

<http://localhost/phpmyadmin/>

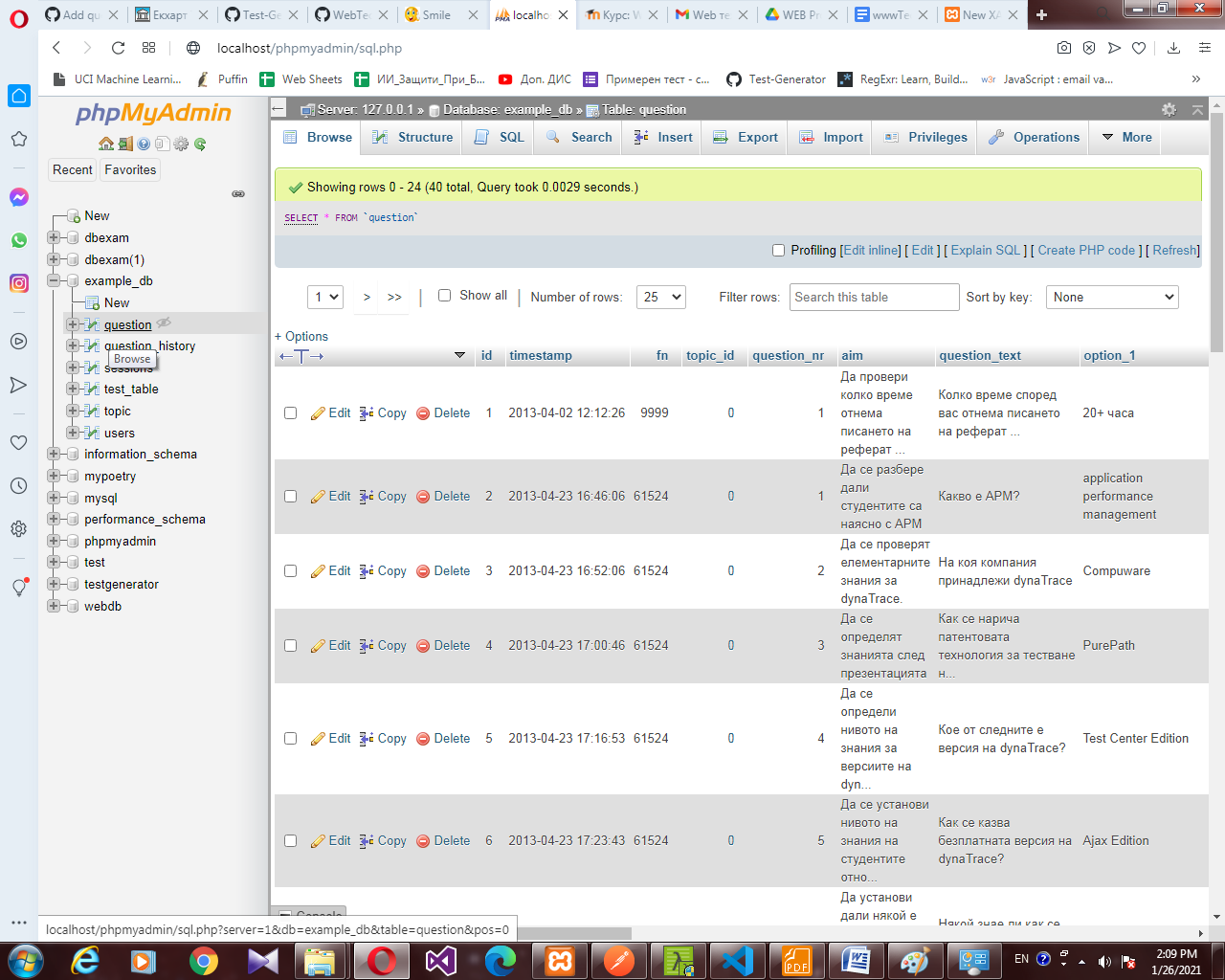
Тук с помощта на **phpMyAdmin** потребителят ще създаде собствената база от данни. Какви са стъпките за това?

В горния ляв ъгъл натиска опцията **New**, с която ще създаде празна база данни за начало. След което в двете поленца за въвеждане(едно за въвеждане и едно за избиране) пише съответно името на базата, в този случай пише **testgenerator**  и избира от падащото меню опцията **utf8mb4\_general\_ci**. След това натиска бутона **Create**. Зарежда му се страничката с новосъздадената база данни, която за момента е празна. От менюто с изредените отгоре опции избира **Import**. След което натиска **Choose file** и навигира до желания файл (в нашия случай това е файлът **seed.sql**), пътят е следният: **C:\xampp\htdocs\Test-Generator\database\_seeds\seed.sql**. След това натиска бутона **Go** най-долу вдясно.



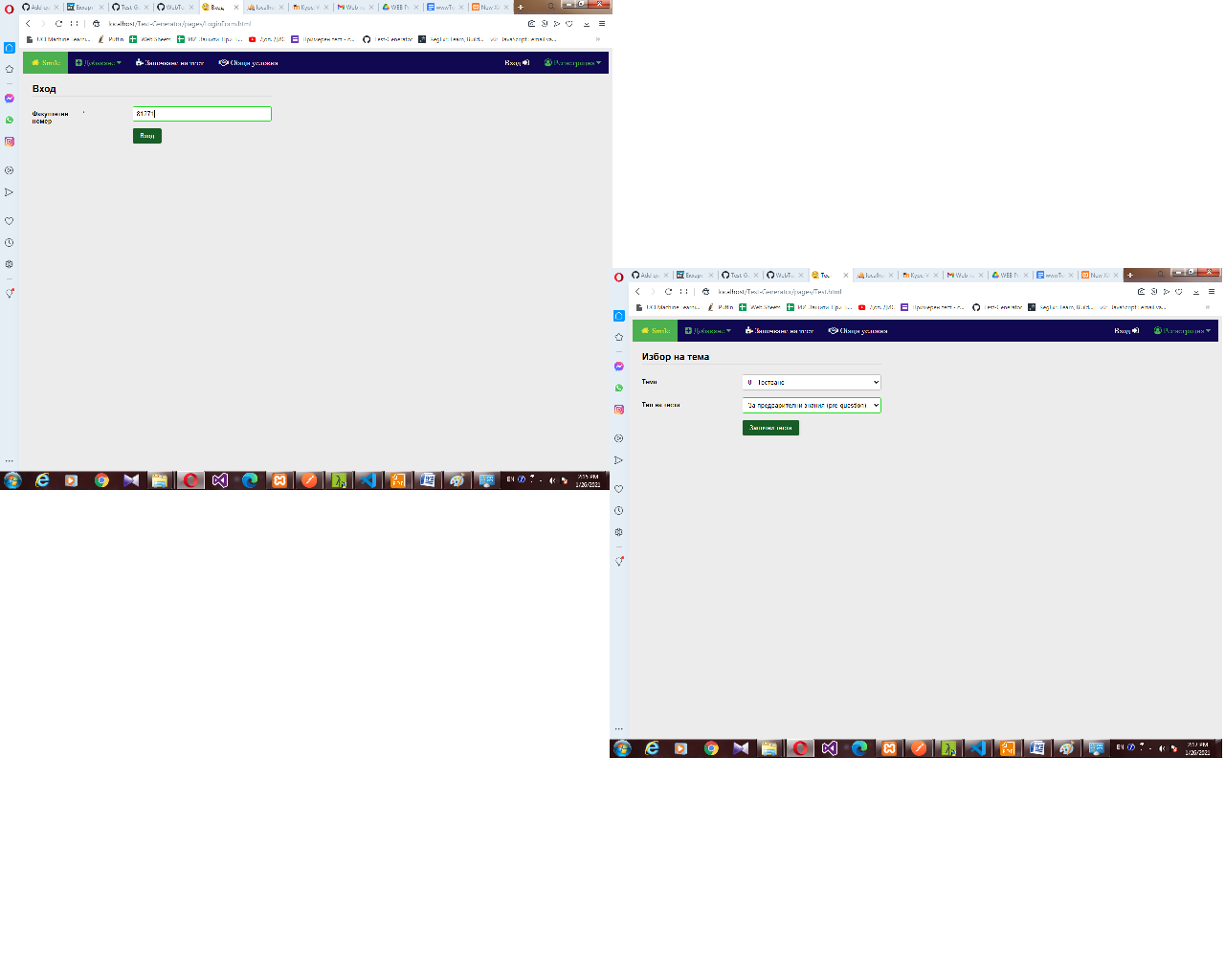
**Фиг.2 Екранна снимка – работа с phpMyAdmin – import на sql фйл**

След изпълнение на тази заявка отива в менюто отляво и натиска върху името на създадената база – **testgenerator**. Като подмодули може да види отделните таблици (релации), които са се създали при зареждането на файла. Те са **question**, **question**\_history, **sessions**, **test\_table**, **topic**, **users, test\_history**. При натискането върху някоя от тях може да види цялата информация за записите вътре, може да добавя нови, да редактира и да трие такива, както и други опции, позволени от phpMyAdmin.



**Фиг.3 Работа с phpMyAdmin – достъп до таблиците на базата**

Сега вече е ред потребителят да напише в браузъра адреса: **localhost/Test-Generator/index.html**. Отваря се **началната страница**, откъдето да отиде на **Вход** и да влезе със своя факултетен номер. От падащото меню **Добавяне** може да избере съответно да добави **Тема** или **Въпрос**, в **Общи условия** може да прочете и да се запознае с общите условия, касаещи работата му в системата и в **Започване на тест** може да избере тема и тип на въпросите по темата, които да бъдат генерирани, и да започне теста.



**Фиг. 4 Работа със системата за генериране на тестове – въвеждане на входни данни**

## Примерни данни

За **Добавяне**:

* На **тема**:
* Тема111
* 111
* Допълнителна информация за Тема111
* На въпрос:
* Тема56
* Да провери дали потребителят се е научил да работи със системата
* За предвартиелни знания (pre-question)
* 4
* Откъде се добавя въпрос?
* От стр.1
* От стр.2
* От стр.3
* От стр.4
* Опция1
* Отлично!
* Лошо!
* Виж документацията на проекта
* 81271

За **import csv**:

Натиска се **Choose file**, избира се : **C:\xampp\htdocs\Test-Generator\database\_seeds\test\_questions\_for\_import.csv**. И след това се натиска върху **import csv** бутона, така съответните записи се вкарват в базата данни. Това са примерни данни за таблицата на въпросите – **question**.

За **Вход**:

* 81271

Към момента няма добавена функционалност Регистрация на потребител. Има HTML формички, които потребителят може да попълни. По-нататъшно развиване на тази функционалност изисква промяна на логиката и на сървъра и към момента не е реализирано.

Общи условия е страница, на която се представят забележки и правила за използване на функционалността. Могат да бъдат променяни като се добавят, променят или изтрият съответните елементи в **C:\xampp\htdocs\Test-Generator\pages\Conditions.html.**

## 8. Описание на програмния код

В папката **database\_seeds** се намира файлът **seed.sql**, който се използва за създаване на таблицитие в базата данни testgenerator. Трите csv файла в същата папка съдържат примерни данни за добавяне в таблицата на въпросите – question.

В папката **config** се намира файлът **config.ini**, който съдържа конфигурацията за връзката с базата данни, а именно host, name, user и password.

В папката **src** се намира файлът **utils.php**, където са функциите за изпълнение на SQL заявки към базата, за обработка на входа от дадена заявка, както и за управление на проверката за логнат или не потребител.

В папката **api** имаме подпаките **admin**, **history**, **question**, **test**, **topic** и **users**. Всяко от тях съдържа един или няколко php файла за работа със съответните елементи, имената описват функционалността.

В папката **scripts** се намират **javascript файловете**, чрез които се взимат **данните, които потребителят е въвел** в съответната форма или страница и чрез определените функции се изпращат асинхорнно заявки към сървъра, където съответните php файлове се грижат за обработката на изпратените данни. Отново чрез тези javascript файлове се връща и **отговор на потребителя** за резултата от заявкатаи при необходимост се предоставя нещо изискано от него.

Файлът **navigation\_bar.js** съдържа javasacript функция, чрез която се генерира HTML кодът на **навигационния бар**, който на свой ред се визуализира на всяка страница в системата и позволява връзките от една страница към останалите.

Файлът **papaparse.js** **е библиотеката Papa Parse**, с чиято помощ четем от **csv файловете**.

В папката **pages** се съдържат **HTML файловете на страниците в системата**, които потребителят достъпва.

Стиловото им оформление се задава чрез **CSS код**, който изпълва файловете, намиращи се в папката **styles**, съответно за стила на самите страници и за стила на оформление на формите, които се попълват.

**В папката на проекта** се намира и файлът **index.html**, който е главната страница в системата.

**Кодът на проекта** се намира в следното **github repository** и е достъпен **на адрес:**

[**https://github.com/Dimitko/Test-Generator**](https://github.com/Dimitko/Test-Generator)

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

Участие в писането на кода по сървърната част – функционалностите, касаещи test (generate, submit), администратор контролен панел, части от функциите за връзка с базата данни в utils.php, config.ini. Във front-end частта: Изпращане на заявки чрез javascript – test.js и части от admin.js, export на csv файлове с всички данни в сървъра. Статистики и визуализацията им във front-endа.

Postman request тестове.

## 10. Какво научих

Основи на PHP, основи на Javascript и Async Javascript концепции (fetch API, Promises, etc.), комуникация с база данни, затвърдих силно разбирането си връзката между Front-end и Back-end сървъри. Затвърдих знанията си върху HTML и CSS.

## 11. Използвани източници

[1]ngadzeva/ WebTechnologies-labs-KN20

<https://github.com/ngadzheva/WebTechnologies-labs-KN20>

[2]Javascript-Upload-File-Stackoverflow

https://stackoverflow.com/questions/5587973/javascript-upload-file [3]PHP Manual

<https://www.php.net/manual/en/>

[4]PHP Date&Time Function With Example

<https://www.guru99.com/php-date-functions.html#:~:text=A%20timestamp%20is%20a%20numeric,ini%20file>.

[5]Курс Web технологии – зимен семестър 2020/2021

<https://learn.fmi.uni-sofia.bg/course/view.php?id=6487>

## 12. Списък с фигурите

[1] **Фиг.1 Екранна снимка - url за достъп до системата**

**[2] Фиг.2 Екранна снимка – работа с phpMyAdmin – import на sql фйл**

**[3] Фиг.3 Работа с phpMyAdmin – достъп до таблиците на базата**

**[4] Фиг. 4 Работа със системата за генериране на тестове – въвеждане на входни данни**

Предал (подпис): ………………………….

/*фн, имена, спец., група*/

Приел (подпис): ………………………….

/доц. *Милен Петров*/