Full Stack

Каждая из представленных в резюме работ отражает свою область программирования или web технологий. Тем не менее эта работа — центральная и наиболее показательная так как отражает полный путь создания полноценного сайта и в ней задействован самый широкий диапазон технологий.

Эта работа начиналась как практически полезная. На своей предыдущей работе мне доводилось работать со множеством законодательных актов, постановлений, нормативов, правил. Мне необходимо было всё это выучивать и принимать экзамены у других людей. Поэтому, для пользы всех кому это нужно, было решено создать подобный онлайн тренажёр. И вообще, такой подход соответствует моему подходу к обучению.

Тренажёр имеет браузерную и серверную части. Сервер написан на Node.js. Для обработки запросов используется модуль Express, и несколько других модулей для работы с формам,и базой данных и прочим, полный список смотрите ниже. Для генерации страниц использован шаблонизатор Ejs. Информация хранится в базе данных, под управлением MySQL.

Список используемых модулей и пакетов:

Клиентская часть:

- HTML
- JavaScript
- CSS

Серверная часть:

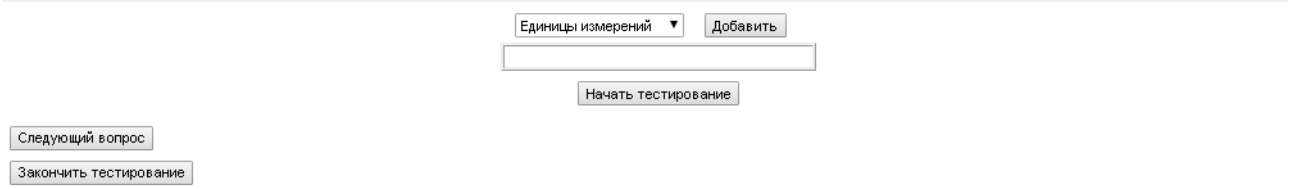
- Node.js
  - Express
  - body-parser
  - mysql
  - cookie-parser
  - ejs
- MySQL

Для работы с базой данных использован инструмент MySQL Workbench 6.3 CE.

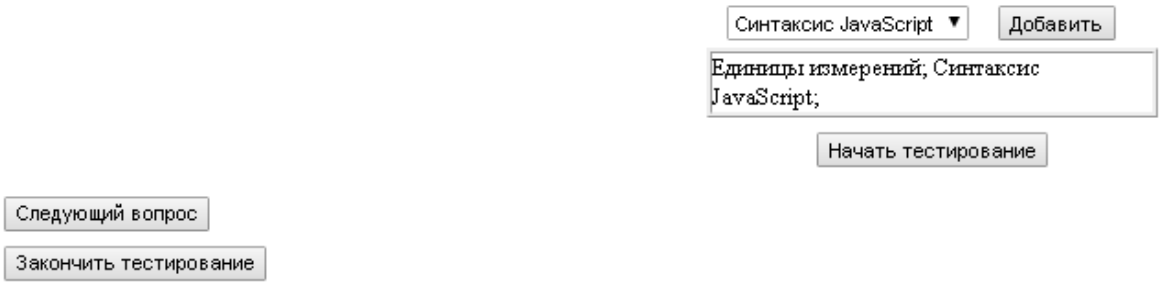
Web интерфейс:

Так окно выглядит при загрузке. В выпадающем списке

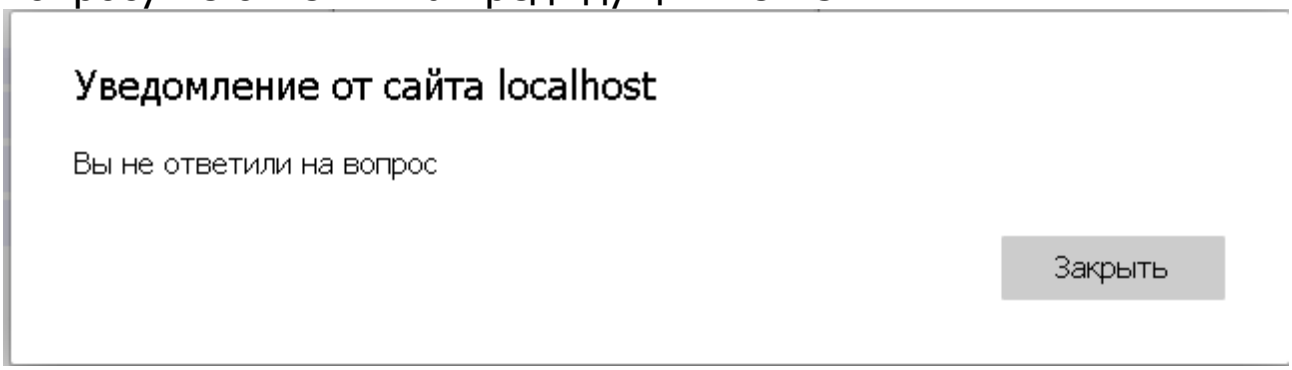
можно выбирать темы и добавлять их в список тем для тестирования.



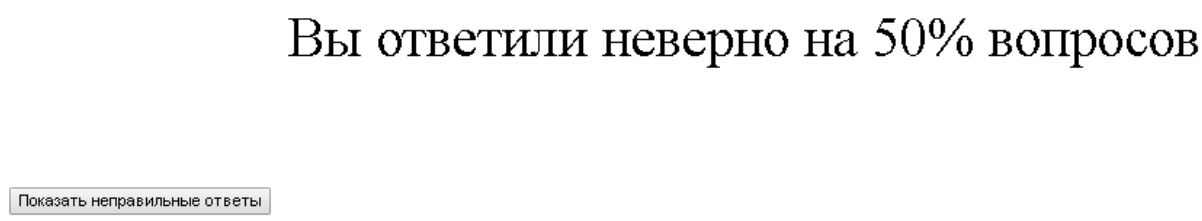
Вид с выбранными темами



По кнопке Начать тестирование переходим к тестированию. Появляются вопросы с вариантами ответов из которых нужно выбрать правильный, переходить к следующему вопросу не ответив на предыдущий нельзя



Закончить тестирование можно в любой момент, текущий ответ засчитан не будет. После окончания тестирования идёт переадресация на окно с результатами тестирования.



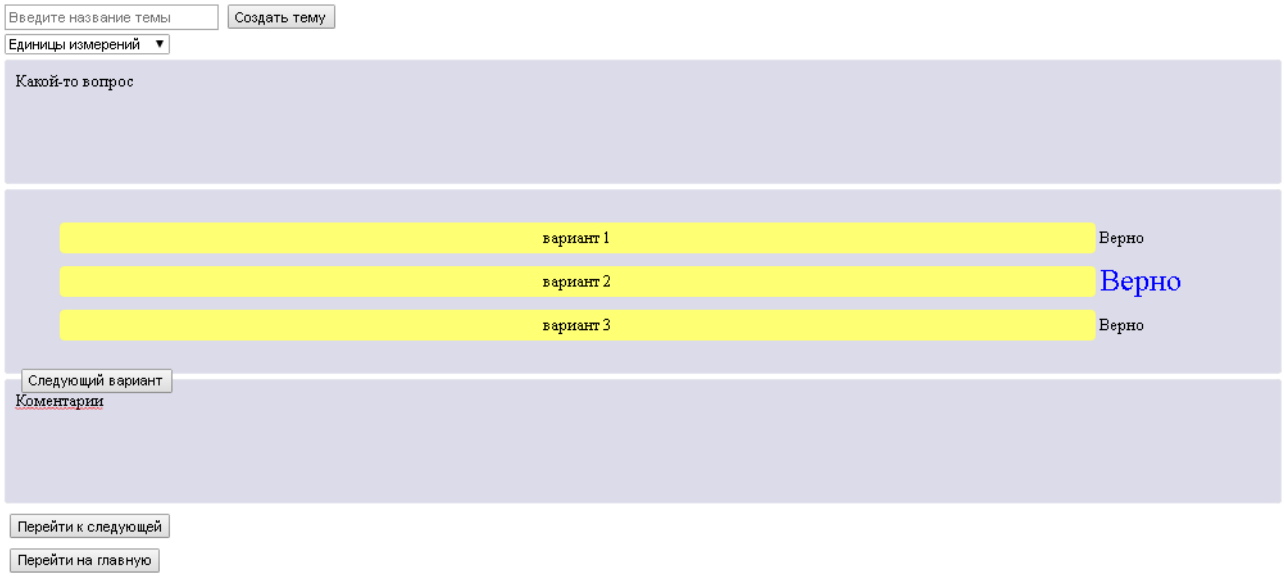
Можно посмотреть вопросы на которые были даны неправильные ответы с правильными вариантами ответов

Ошибки

Сколько сантиметров в метре

100

Есть отдельный интерфейс для создания новых карточек с вопросами и для внесения новых тем.



В первом выпадающем списке можно выбрать уже существующую или внести новую тему. Затем сформулировать вопрос, указать варианты ответов, выбрав правильный.

Код серверной части приведу в сокращении (только пути запросов)

```
var express = require('express');
var app = express();
var body = require('body-parser');
var multer = require('multer');
var upload = multer({dest: 'public/uploads/'});
var DB = require('mysql');
var cookieParser = require('cookie-parser');
var fs = require('fs');

app.use(cookieParser());
app.use(body.json());
app.use(body.urlencoded({extended: true}));
app.use(express.static("public"));

var connection = DB.createConnection({

app.set('view engine', 'ejs');

//блок некоторых служебных переменных глобальной области видимости
var arr_question = [];//массив вопросов по результатам выборки из БД
//var arr_badans = new Array();//массив  номеров неправильных ответов
var actual_question = 'default';//номер текущего вопроса
var actual_bad_answer = 'default';//номер текущего неправильного ответа
//.....................................................................
```

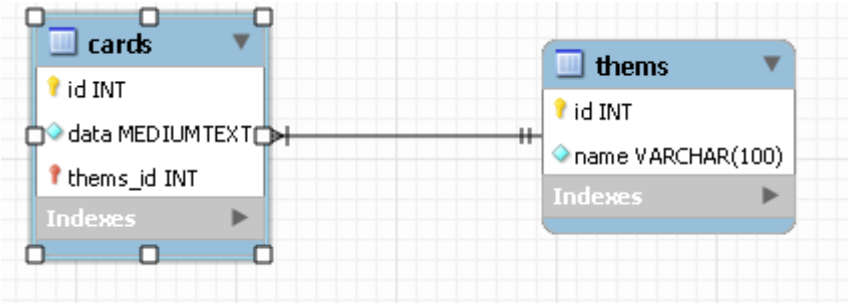
В коде выше подключаются модули и настраивается работа с сервером баз данных MySQL. В коде ниже приведены маршруты обработки запросов, ещё ниже подробно разобран один из запросов.

```
//рендер страницы выбора тем для проверки (главная)
app.get('/', function(req, res){
//страница создания карточки вопроса (для админа и запаролена)
app.get('/entercard', function(req, res){
//формирование массива вопросов
app.post('/', function(req, res){
//добавление карточки в БД
app.post('/entercard', function(req, res){
//добавление новой темы в БД
app.post('/addThems', function(req, res){
    var count_anw = 0;// количество ответов на которыен ответил пользователь
    var arr_bad_anw = [] ;//массив номеров неправильных ответов
 //приём ответа на вопрос и его обработка
 app.post('/klientanswer', function(req, res){
 var ststistica = 0;//переменная которая хранит статистику

 //обработка завершения тестирования
 app.post('/stop', function(req, res){


 app.get('/err answers', function(req, res){
 var server = app.listen(4051, function(){
```

Вот код POST запроса по формированию массива вопросов в соответствии с выбранными темами. Он подробно прокомментирован, поэтому отдельно разбирать я его не буду.

```
app.post('/', function(req, res){
    var str = req.body.klientData.thems;//строка из тем идущих через ;  и пробел
    //разбивка строки на массив
    var arrStr = str.split('; ');
    //если массив не пустой
    if(arrStr.length) {//последний элемент в таком массиве получается строка с одним пробелом, удалим её
    arrStr.pop();
    //формируем строку SQL запроса к БД по выбранным пользователем темам
    var SELECT = 'SELECT data FROM cards INNER JOIN thems ON cards.thems_id = thems.id WHERE thems.name = ' +
           "'" + arrStr[0] + "'";
    var cond = '';//заготовка пустой строки условия
    for(var i = 1; i<=arrStr.length-1; i++){
        cond +=   " OR " + 'thems.name = ' + "'" + arrStr[i] + "'"
    }
    SELECT += cond;
    SELECT += ' ORDER BY RAND();';//добавление в условие выборки случайной последовательности
    connection.query(SELECT, function(error, results, fields){
        if(error){
            console.log('Ошибка связи с базой данных');
            res.send('Что-то пошло не так свяжитесь с администратором сайта')}
        else{
        arr_question = results;
        res.send(results);
    }
} )
    }
    else {res.send('Выберите тему')}
\\
```

Структура базы данных выглядит следующим образом



Резюме

В этом примере показаны все этапы конструирования сайта. От вёрстки до серверной части и работы с базами данных. Дальнейшие доработки этого сайта увеличили бы его масштаб, но не принесли бы ничего принципиально нового. Хотелось бы отметить что node.js в качестве серверного языка, очень эффективен. Наличие огромного количества модулей для любой необходимости делает этот инструмент невероятно эффективным и эластичным (см. пример с прокси сервером). Организация хранения информации в реляционной базе данных (в данном случае MySQL) даёт неограниченные возможности по расширению возможностей сайта.