

Министерство Образования и Науки Кыргызской Республики
Кыргызский Государственный Технический Университет им. И. Раззакова
Кыргызско-Германский Технический Институт
Кафедра «Телематика»
Дисциплина: Интернет программирование

Пояснительная записка к курсовому проекту

**на тему: Проектирование и разработка Web-сайта для кафедры
«Художественное проектирование изделий»**

Выполнил: Анарбеков А.К.

Группа: Тг(б)-1-19

Проверила: Султангазиева Р.Т.

Бишкек 2023 г.

«УТВЕРЖДАЮ»

Зав.кафедры «Телематика»

Медралиева Б.Н. _____

«__» _____ 2022 г.

Задание

Студент: Анарбеков Аскар

Тема курсового проекта: «Создание сайта для кафедры «ХПИ».

Руководитель курсового проекта: Султангазиева Р.Т

Техническое задание на курсовую работу

- Разработать веб сайт с использованием языка программирования общего назначения с открытым исходным кодом
- создать базу сотрудников
- новости и разработки кафедры
- информация для абитуриентов
- форма связи с администратором
- загрузка веб сайта на хостинг

Дата выдачи задания «__» _____ 2022 г.

Задание выдала: _____ (подпись)

Задание приняла: _____ (подпись)

Аннотация

Целью данной курсовой работы является возможность разработки и создание Web-сайта.

При создании сайта были изучены следующие технологии по созданию Web-сайтов (WampServer, ReactJS, Node.js, Bootsrtap, Visual Studio Code, HTML, CSS, JS). Для реализации проекта был выбран фреймворк «ReactJS».

Результатом данной работы является разработка полноценного сайта для кафедры «Художественное проектирование изделий».

Созданный web-сайт позволит пользователям находить всю необходимую информацию о кафедре и свежие новости.

В первой главе рассматриваем общие термины и представления о веб сайтах.

Во второй главе рассматривалось множество ПО (программное обеспечение), с помощью которых разрабатываются web-сайт. Использовались следующие инструменты такие как редактор VS code, HTML CSS, Java Script, Node Js, библиотека React.

В третье главе более подробно представлена подробная информация о кафедре для пользователей. Описываем каталог проекта, связь с базой данных и рассмотрение пользовательской части.

В заключении приведен общий вывод о проделанной работе и выполнение задач.

Курсовая работа состоит из 25 страниц и 13 рисунков.

Введение	5
Глава 1. Основные понятия Web – сайта.....	6
1.1. Классификация веб сайтов.....	7
1.2. Языки программирования	8
Глава 2. Обзор программных обеспечений для разработки web-сайта	9
2.1. VS code.....	9
2.2. Node Js	10
2.3. JavaScript	11
2.4. HTML CSS	13
Глава 3 Реализация веб ресурса	15
3.1. Описание каталога проекта	15
3.2. Связь с базой данных.....	19
3.3. Пользовательская часть	21
Заключение.....	24
Список литературы	25

Введение

В современную цифровую эпоху наличие хорошо спроектированного и функционального веб-сайта крайне важно для любой организации, включая учебных заведений.

Веб-сайт служит окном в отдел, предоставляя информацию о его услугах, проектах и персонале. Он также выступает в качестве платформы для общения, позволяя посетителям взаимодействовать с отделом и его сотрудниками.

В условиях растущей зависимости от технологий и интернета хорошо разработанный веб-сайт может значительно повысить узнаваемость департамента, доверие к нему и его охват.

В этой курсовой работе будет рассмотрен процесс разработки веб-сайта для кафедры художественного проектирования изделий, включая связанные с этим проблемы и возможности, обзор использованных для разработки ресурсов и инструментов, а также внедрение и оценку конечного продукта. В процессе будут использованы соответствующая литература, тематические исследования и современные инструменты разработки, чтобы предоставить всеобъемлющий обзор разработки веб-сайтов.

Глава 1. Основные понятия Web – сайта

В данной главе рассматриваются общие термины и представления о веб сайтах

Сайт, или **веб-сайт** (от англ. website: web — «паутина, сеть» и site — «место», букв. — «место, сегмент, часть в сети»), также **веб-узел**, — одна или несколько логически связанных между собой веб-страниц; также место расположения контента сервера. Обычно сайт в Интернете представляет собой массив связанных данных, имеющий уникальный адрес и воспринимаемый пользователями как единое целое. Веб-сайты называются так, потому что доступ к ним происходит по протоколу HTTP.

Веб-сайт как система электронных документов (файлов данных и кода) может принадлежать частному лицу или организации и быть доступным в компьютерной сети под общим доменным именем и IP-адресом или локально на одном компьютере.

Веб-сайты могут включать различные мультимедийные элементы, такие как текст, изображения, аудио, видео и интерактивные функции, которые предназначены для привлечения и информирования посетителей. Они также могут включать различные технологии, такие как HTML, CSS, JavaScript и языки сценариев на стороне сервера, для улучшения функциональности и пользовательского опыта веб-сайта.

Веб-сайты могут быть разработаны для различных устройств, включая настольные компьютеры, ноутбуки, планшеты и смартфоны, чтобы обеспечить доступность для широкого круга пользователей. В целом, веб-сайты играют решающую роль в цифровом ландшафте, служа основным средством коммуникации, коммерции и развлечения для миллионов людей по всему миру. [1]

1.1. Классификация веб сайтов

Сайт является набором веб-страниц, объединенных общей тематикой и связанных между собой гиперссылками, единой системой навигации. В зависимости от решаемых задач для создания сайта выбирают тот или иной язык серверных скриптов. Для создания малых и средних интерактивных сайтов целесообразно применить язык сценариев PHP. Конкурентами PHP являются технологии ASP, JSP, Cold Fusion, Perl. Достоинством языка PHP является то, что он является бесплатным, имеет открытые исходные коды и работает почти на всех платформах. В зависимости от технологии создания можно выделить следующие типы сайтов:

1. Статические сайты, содержащие статические HTML или XHTML страницы. Статические веб-страницы – это статические файлы (набор текста, таблиц, рисунков и т.д.), которые создаются с помощью языка разметки HTML (имеют расширение .html или .htm) и хранятся в готовом виде в файловой системе сервера.
2. Динамические сайты, в которых веб-страницы генерируются или формируются (создаются динамически) в процессе исполнения запроса пользователя. Динамические сайты бывают двух типов. В первом типе сайтов, веб-страницы генерируются или формируются из данных хранящихся на сервере в базе данных. Во втором типе сайтов веб-страницы генерируются на стороне клиентского приложения (в браузере).
3. Flash-сайты – это интерактивные приложения, разработанные в среде Macromedia Flash. Основным инструментом разработки flash-программ является векторная графика (интерактивная векторная анимация для Web). Flash придает сайтам динамичность и интерактивность.
4. Комбинированные сайты, в которых используются вышеизложенные технологии создания сайтов. [2]

1.2. Языки программирования

Язык программирования - это формальный язык, предназначенный для передачи инструкций компьютеру или другому вычислительному устройству. Это набор правил и символов, который позволяет программистам писать инструкции, которые компьютер может понять и выполнить.

Языки программирования используются для создания программного обеспечения, веб-сайтов, мобильных приложений, видеоигр и множества других компьютерных программ. Существует множество языков программирования, каждый со своим собственным синтаксисом и правилами. Некоторые популярные языки программирования включают Java, Python, JavaScript, C++, Ruby и Swift.

Когда программисты пишут код на языке программирования, они используют текстовый редактор или интегрированную среду разработки (IDE) для ввода кода. Затем код переводится на машинный язык, который является языком низкого уровня, который компьютер может выполнять непосредственно. Этот процесс обычно выполняется компилятором или интерпретатором. [3]

Глава 2. Обзор программных обеспечений для разработки web-сайта

В период разработки курсового проекта рассматривалось множество ПО (программное обеспечение), с помощью которых разрабатываются web-сайта. Для создания web – сайта использовались следующие инструменты такие как редактор VS code, HTML CSS, Java Script, Node Js, библиотека React.

2.1. VS code

VS Code (сокращение от Visual Studio Code) - это бесплатный редактор кода с открытым исходным кодом, разработанный Microsoft. Он разработан как легкий и настраиваемый редактор для написания кода на различных языках программирования и фреймворках.

VS Code предоставляет ряд функций, помогающих разработчикам быть более продуктивными, включая подсветку синтаксиса, завершение кода, отладку и интеграцию с Git. Он также поддерживает расширения, которые могут быть установлены для добавления дополнительных функций, таких как поддержка определенных языков программирования, инструменты для тестирования и отладки и интеграция с облачными сервисами.

Одной из ключевых сильных сторон VS Code является его способность настраиваться в соответствии с индивидуальными предпочтениями разработчика. Это включает в себя возможность изменять тему редактора, сочетания клавиш и макет пользовательского интерфейса. VS Code также поддерживает использование нескольких интегрированных терминалов и имеет встроенную поддержку популярных инструментов командной строки.

VS Code доступен для Windows, macOS и Linux и стал одним из самых популярных редакторов кода, используемых разработчиками. Его популярность отчасти обусловлена простотой использования, множеством функций и большим сообществом разработчиков, которые вносят свой вклад в его разработку и поддержку. [4]

2.2. Node Js

Node.js это бесплатная кроссплатформенная среда выполнения JavaScript с открытым исходным кодом на стороне сервера. Это позволяет разработчикам создавать и запускать серверные приложения с использованием JavaScript, который традиционно используется для интерфейсной веб-разработки. Node.js использует управляемую событиями неблокирующую модель ввода-вывода, что делает ее эффективной и масштабируемой для создания приложений с интенсивным использованием данных в реальном времени.

Node.js был создан Райаном Далем в 2009 году и с тех пор стал очень популярным среди разработчиков. Он поставляется со встроенным менеджером пакетов под названием npm, который позволяет разработчикам легко устанавливать зависимости для своих Node.js приложения.

Node.js обычно используется для создания веб-приложений, RESTful API, приложений для чата в реальном времени и бессерверных функций, среди прочего. У него большое и активное сообщество разработчиков, и есть много ресурсов, доступных для изучения и использования Node.js .

Node.js предлагает ряд следующих преимуществ:

JavaScript: Node.js позволяет разработчикам писать серверный код с использованием JavaScript, который является тем же языком, что и для интерфейсной веб-разработки. Это облегчает разработчикам переключение между front-end и back-end разработкой, а также облегчает совместное использование кода между front-end и back-end.

Скорость и масштабируемость: Node.js использует неблокирующую модель ввода-вывода, управляемую событиями, что делает ее очень быстрой и эффективной. Это позволяет ему обрабатывать большое количество одновременных подключений, что делает его хорошим выбором для создания приложений с интенсивным использованием данных в режиме реального времени.

Большая экосистема: Node.js имеет большое и активное сообщество разработчиков, что привело к созданию богатой экосистемы модулей и пакетов, которые могут быть легко установлены и использованы в Node.js приложения. Это позволяет разработчикам легко добавлять функциональность в свои приложения без необходимости писать все с нуля.

Кроссплатформенность: Node.js является кроссплатформенным, что означает, что он может работать на нескольких операционных системах, включая Windows, Linux и macOS. Это упрощает разработку и развертывание Node.js приложения на различных платформах.

Легко усваивается: Потому что Node.js использует JavaScript, который является популярным языком для фронтальной веб-разработки, многие разработчики уже знакомы с ним. Это делает его относительно легким для разработчиков в освоении и использовании Node.js .

В целом, Node.js это мощная и гибкая технология, которая обладает многими преимуществами для создания серверных приложений и сервисов.

[5]

2.3. JavaScript

JavaScript - это язык программирования высокого уровня, который в основном используется для создания интерактивных веб-страниц и веб-приложений. Впервые он был представлен в середине 1990-х годов компанией Netscape как способ добавления динамического поведения к веб-страницам.

JavaScript - это клиентский скриптовый язык, что означает, что он выполняется на стороне клиента (т.е. в веб-браузере пользователя), а не на стороне сервера. Это делает его идеальным для создания интерактивных веб-страниц и веб-приложений, которые могут реагировать на действия пользователя в режиме реального времени без необходимости перезагрузки всей веб-страницы.

JavaScript - это универсальный язык, который может использоваться для широкого спектра задач, от создания простых анимаций и проверки форм до создания сложных веб-приложений. Он также широко используется в

серверном программировании, благодаря таким технологиям, как Node.js , который позволяет использовать JavaScript для создания серверных приложений.

В целом, JavaScript является одним из самых популярных и широко используемых языков программирования в мире, и он сыграл ключевую роль в развитии современного Интернета.

JavaScript обладает рядом преимуществ такие как:

Универсальность: JavaScript может использоваться в широком диапазоне контекстов, от интерфейсной веб-разработки до серверного программирования. Его можно использовать для создания интерактивных веб-страниц, веб-приложений, мобильных приложений, настольных приложений и даже игр.

Интерактивность: JavaScript позволяет разработчикам создавать динамические и интерактивные веб-страницы, которые реагируют на действия пользователя в режиме реального времени, без необходимости перезагрузки всей страницы. Это может повысить вовлеченность пользователей и создать лучший пользовательский опыт.

Совместимость с браузером: JavaScript поддерживается всеми основными веб-браузерами, включая Google Chrome, Firefox, Safari и Internet Explorer. Это означает, что разработчики могут писать код, который будет работать во всех основных веб-браузерах.

Большое и активное сообщество разработчиков: JavaScript имеет большое и активное сообщество разработчиков, что означает, что доступно множество ресурсов для обучения и устранения неполадок. Существует также множество сторонних библиотек и фреймворков, которые могут помочь ускорить разработку.

Простота в освоении: JavaScript имеет относительно низкий барьер для входа, что облегчает его изучение новичкам. Его синтаксис похож на другие языки программирования, такие как C++, Java и Python, что позволяет разработчикам легко переключаться между языками. [6]

2.4. HTML CSS

HTML (от англ. *HyperText Markup Language* — «язык гипертекстовой разметки») — стандартизированный язык гипертекстовой разметки документов для просмотра веб-страниц в браузере. Веб-браузеры получают HTML документ от сервера по протоколам HTTP/HTTPS или открывают с локального диска, далее интерпретируют код в интерфейс, который будет отображаться на экране монитора.

Элементы HTML являются строительными блоками HTML страниц. С помощью HTML разные конструкции, изображения и другие объекты, такие как интерактивная веб-форма, могут быть встроены в отображаемую страницу. HTML предоставляет средства для создания заголовков, абзацев, списков, ссылок, цитат и других элементов. Элементы HTML выделяются тегами, записанными с использованием угловых скобок. Такие теги, как `` и `<input />`, напрямую вводят контент на страницу. Другие теги, такие как `<p>`, окружают и оформляют текст внутри себя и могут включать другие теги в качестве подэлементов. Браузеры не отображают HTML-теги, но используют их для интерпретации содержимого страницы. [7]

CSS расшифровывается как каскадные таблицы стилей. Это язык таблиц стилей, используемый для описания представления и верстки веб-страниц, написанных на HTML (языке разметки гипертекста).

CSS позволяет веб-разработчикам отделять представление веб-страницы (например, макет, шрифты, цвета и интервалы) от ее содержимого. Это означает, что разработчики могут писать чистый и организованный HTML-код, который проще поддерживать и обновлять, не беспокоясь о дизайне и верстке страницы.

CSS работает, применяя правила к HTML-элементам, которые определяют, как они должны отображаться на веб-странице. Правила CSS состоят из селектора (который идентифицирует HTML-элемент) и набора

объявлений (которые определяют свойства стиля и макета, которые будут применены к элементу).

CSS можно использовать для оформления практически любого HTML-элемента, от текста и изображений до кнопок и форм. Он также поддерживает передовые методы верстки, такие как flexbox и grid, которые упрощают создание сложных макетов страниц.

В целом, CSS является важным инструментом для современной веб-разработки, и он играет решающую роль в создании визуально привлекательных и удобных для пользователя веб-страниц. [8]

Глава 3 Реализация веб ресурса

Целью данной курсовой работы является создание сайта для кафедры художественного проектирования изделий. Одним из основных требований к сайту является ее несложность и легкость в использовании. А также подробная информация о кафедре для пользователей. Описываем каталог проекта, связь с базой данных и рассматриваем пользовательскую часть.

3.1. Описание каталога проекта

React предполагает, что логика отображения UI и обработки событий должна быть объединена вместе, а не разделена на разные файлы. Вместо этого, React использует слабо связанные единицы, называемые «компоненты», которые содержат и разметку, и логику.

Каталог проекта состоит из следующий папок как показано на рис.3.1.1.

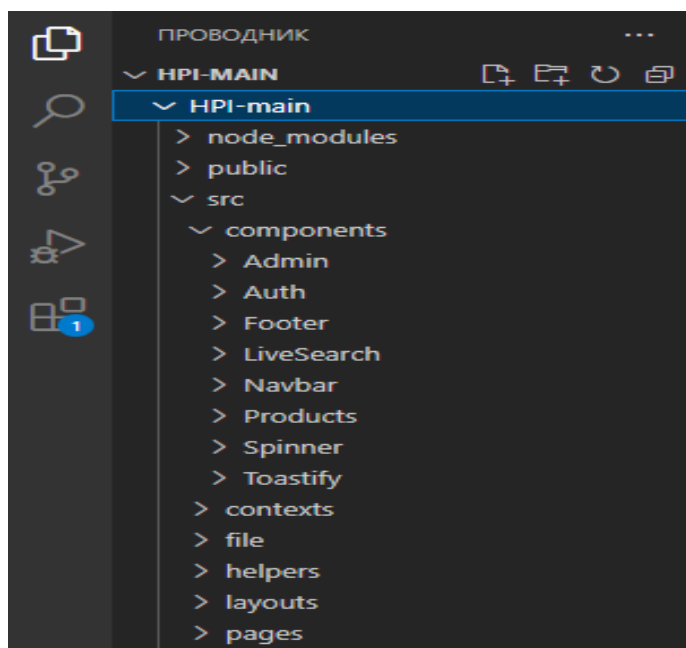


Рис.3.1.1. Каталог проекта

Далее идет описание основных компонентов

На рис.3.1.1. показан компонент админ. Она выполняет следующую функцию:
добавление контента во вкладку новости

```
import { IconButton } from "@mui/material";
import React from "react";
import ArrowBackIosNewIcon from "@mui/icons-material/ArrowBackIosNew";
import { useNavigate } from "react-router-dom";
import Form from "../Form/Form";
import { useProductContext } from "../../contexts/ProductContextProvider";

const Add = () => {
  const navigate = useNavigate();
  const { addProduct } = useProductContext();
  return (
    <div>
      <div style={{ display: "flex" }}>
        <IconButton onClick={() => navigate(-1)}>
          <ArrowBackIosNewIcon fontSize="large" sx={{ color: "black" }} />
        </IconButton>
        <h2>Добавить новость</h2>
      </div>
      <Form saveValues={addProduct} compForEdit={false} />
    </div>
  );
};

export default Add;
```

Рис.3.1.2. Компонент админ

Далее идет компонент AuthForm он выполняет функции авторизации пользователя как показано на рис.3.1.3.


```

4 import { createTheme, ThemeProvider } from "@mui/material/styles";
5 import { Link as AuthLink } from "react-router-dom";
6 import { notify } from "../Toastify/Toastify";
7
8 function Copyright(props) {
9   return (
10     <Typography
11       variant="body2"
12       color="text.secondary"
13       align="center"
14       {...props}
15     >
16       {new Date().getFullYear()}
17       {".."}
18     </Typography>
19   );
20 }
21
22 const theme = createTheme();
23
24 export default function AuthForm({
25   title,
26   btnText,
27   link,
28   linkText,
29   handleSave,
30 }) {
31   const handleSubmit = (event) => {
32     event.preventDefault();
33     const data = new FormData(event.currentTarget);
34     handleSave(data.get("email"), data.get("password"));
35   };
36 }

```

Рис.3.1.3. Компонент AuthForm

Далее идет описание компонента footer. Эта нижняя часть сайта, которая хранит в себе различные ссылки, такие как: контактные номера, новости, фотогалерея, ссылки на социальные сети как показано на рис.3.1.4.

```

4 import InstagramIcon from "@mui/icons-material/Instagram";
5 import TwitterIcon from "@mui/icons-material/Twitter";
6 import FacebookIcon from "@mui/icons-material/Facebook";
7 import YouTubeIcon from "@mui/icons-material/YouTube";
8 const Footer = () => {
9   return (
10     <div className="footer" style={{ paddingBottom: "20px" }}>
11       <div className="footer-two">
12         <div className="first sp">
13           <p>Бакалавр</p>
14           <ul>
15             <li>Учебные проекты</li>
16             <li>Кредиты и система оценки</li>
17           </ul>
18         </div>
19         <div className="second sp">
20           <p>Магистр</p>
21           <ul>
22             <li>Телемедицина</li>
23           </ul>
24         </div>
25         <div className="third sp">
26           <p>Наука </p>
27           <ul>
28             <li>Научные направления</li>
29             <li>Аспирантура</li>
30           </ul>
31         </div>
32         <div className="fourth sp">
33           <p>Абитуриентам</p>
34           <ul>
35             <li>Для поступления в бакалавриат</li>
36           </ul>
37         </div>
38       </div>
39     </div>
40   );
41 }

```

Рис.3.1.4. Компонент footer

На рис.3.1.5. показан навигационный бар. Он содержит в себе основные вкладки, содержащие информацию.

```
HPI-main > src > components > Navbar > Navbar.jsx > ...
17 import './Navbar.css';
18 import { useCart } from '../../contexts/CartContextProvider';
19 import { useAuth } from '../../contexts/AuthContextProvider';
20 import PersonIcon from '@mui/icons-material/Person';
21 import AdminPanelSettingsIcon from '@mui/icons-material/AdminPanelSettings';
22 import { useFav } from '../../contexts/FavContextProvider';
23 import LocalOfferIcon from '@mui/icons-material/LocalOffer';
24 import ImportContactsIcon from '@mui/icons-material/ImportContacts';
25
26 export default function Navbar() {
27   const [anchorEl, setAnchorEl] = React.useState(null);
28   const [mobileMoreAnchorEl, setMobileMoreAnchorEl] = React.useState(null);
29
30   const { currentUser, logOutUser } = useAuth();
31   const { getCartLength, cartLength } = useCart();
32   const { getFavLength, favLength } = useFav();
33
34   React.useEffect(() => {
35     getFavLength();
36   }, []);
37
38   React.useEffect(() => {
39     getCartLength();
40   }, []);
41
42   const isMenuOpen = Boolean(anchorEl);
43   const isMobileMenuOpen = Boolean(mobileMoreAnchorEl);
44
45   const handleProfileMenuOpen = (event) => {
46     setAnchorEl(event.currentTarget);
47   };
48 }
```

Рис.3.1.5. Компонент Navbar

Далее идет Toastify. Он отвечает за получение уведомлений при возникновении каких либо сбоев или ошибок при работе сайта.

```
1 import React from 'react';
2 import { toast, ToastContainer } from 'react-toastify';
3 import 'react-toastify/dist/ReactToastify.css';
4
5 export const notify = (type, message) => {
6   toast(message, {
7     position: 'top-right',
8     autoClose: 5000,
9     hideProgressBar: false,
10    closeOnClick: true,
11    pauseOnHover: true,
12    draggable: true,
13    progress: undefined,
14    type: type,
15    theme: 'colored',
16  });
17 };
18
19 export const notifyError = (err) => {
20   if (err && err?.response) {
21     notify('error', `${err?.response.status} ${err?.response.statusText}`);
22   } else {
23     notify('error', 'Произошла ошибка!');
24   }
25 };
26
27 const Toastify = () => {
28   return (
29     <ToastContainer
30     />
31   );
32 }
```

Рис.3.1.6 Toastify

3.2. Связь с базой данных

База данных «PhpMyAdmin» – Веб-приложение с открытым кодом, написанное на языке PHP и представляющее собой веб-интерфейс для администрирования СУБД MySQL. PhpMyAdmin позволяет через браузер и не только осуществлять администрирование сервера MySQL, запускать команды SQL и просматривать содержимое таблиц и баз данных. Приложение пользуется большой популярностью у веб-разработчиков, так как позволяет управлять СУБД MySQL без непосредственного ввода SQL команд.

На рисунке 3.2.1. файл `init.php`, обеспечивает соединение с базой данных на локальном уровне. Если соединение прошло успешно в браузере получим сообщение «Подключение успешно...» если возникнут какие-нибудь ошибки в таком случае получим сообщение «Connection fail».

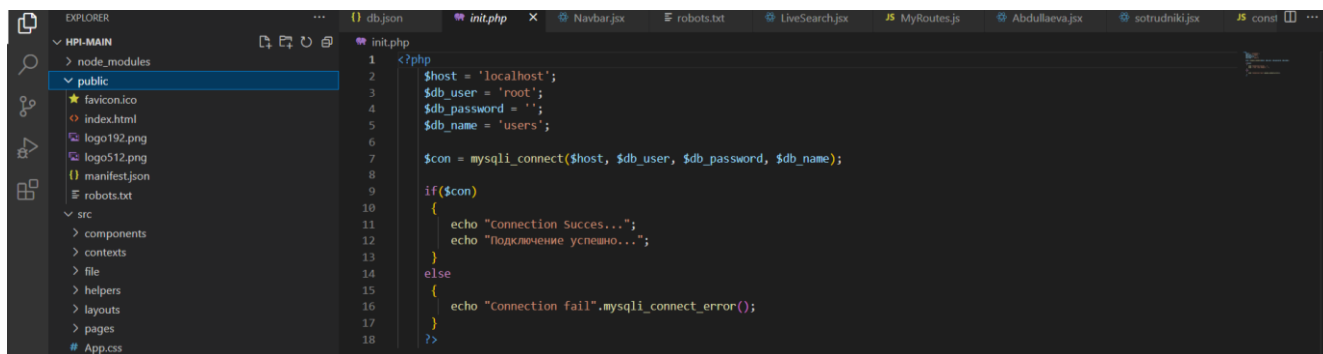


Рис.3.2.1. App – `init.php`

На рисунке 3.2.2. PhpMyAdmin – база данных, представлены элементы, составляющие структуру базы данных:

ID – Идентификаторы– это имена различных объектов, хранящихся в БД. В соответствии со стандартом идентификатор – это уникальная в пределах схемы последовательность из букв, цифр и знаков подчеркивания длиной до 128 символов, начинающаяся с буквы. Допустимы идентификаторы с разделителями (символами, отличными от разрешенных), написанные в кавычках (“a%z”).

Здесь используется тип данных `int` (от англ. `integer` - целое число) один из наиболее часто используемых типов данных для хранения чисел. Служит для представления целых чисел, ограниченного минимальным и максимальным значением, зависящими от выделенной под число памяти.

`Name` – имя пользователя, `Email` – почта пользователя, `Password` – пароль пользователя, `Created_at` – логгер для создания аккаунта пользователя.

Все данные в этих полях указаны в переменной `varchar`, которая в свою очередь представляет собой следующее:

`VARCHAR`-это строковый тип данных переменной длины, поэтому он содержит только те символы, которые вы ему назначаете. Термин `varchar` относится к типу данных поля в базе данных, которое может содержать буквы и цифры. Поля `Varchar` могут быть любого размера до предела, который зависит от баз данных.

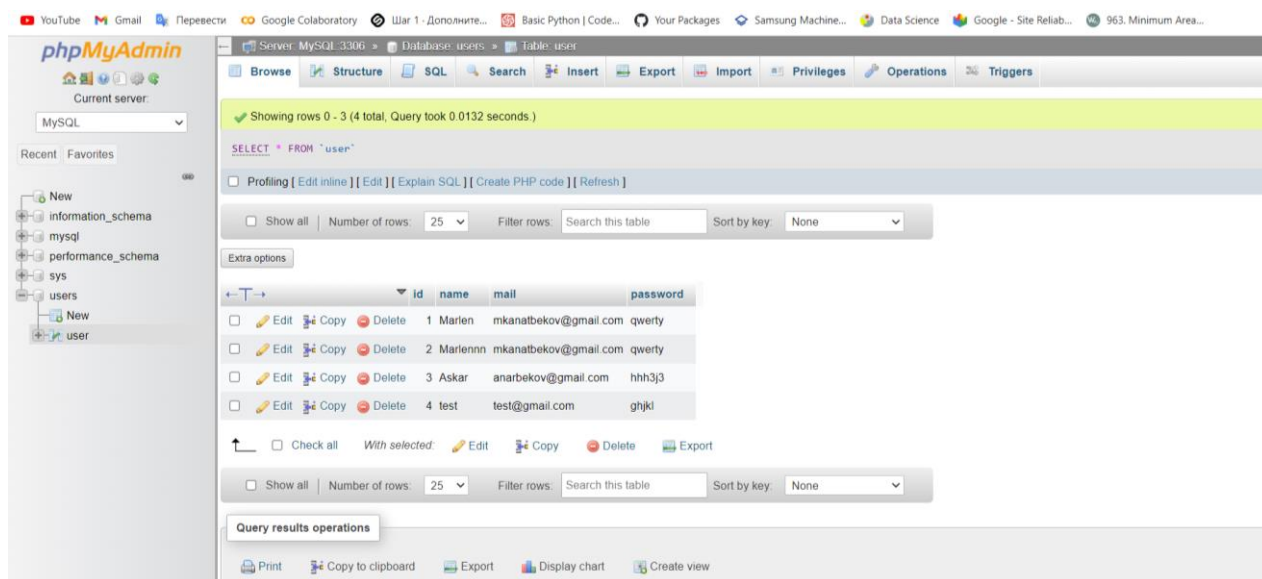


Рис.3.2.2. Структура база данных

На рисунке 3.2.3. Файл `login.php`, в котором проверяется почта и пароль пользователя. Если данные введенные пользователем совпадает с данными которые хранятся в базе данных, то пользователь получает сообщение «Добро пожаловать на страницу кафедры Художественное проектирование изделий».

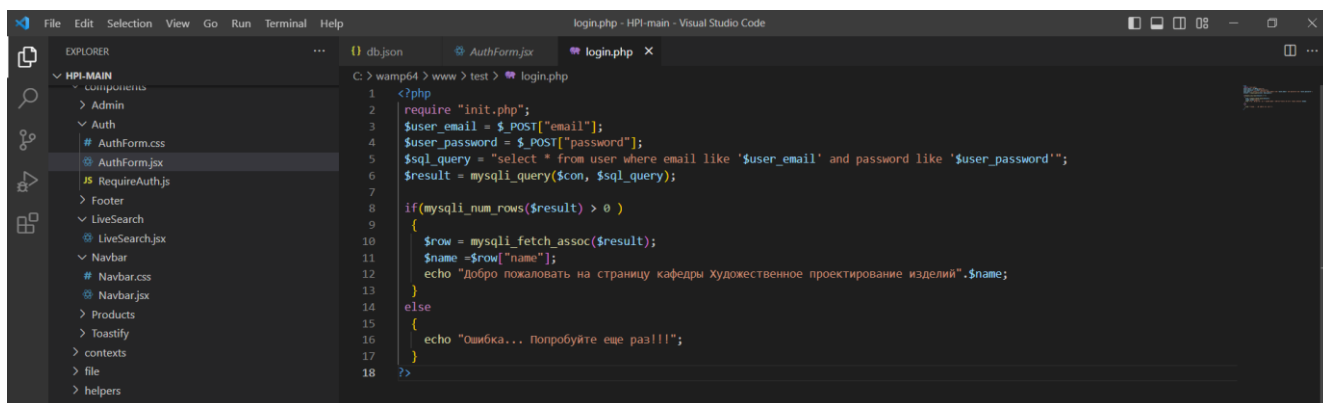


Рис.3.2.3. login.php

На рисунке 3.2.4. Представлен файл register.php, который регистрирует нового пользователя.

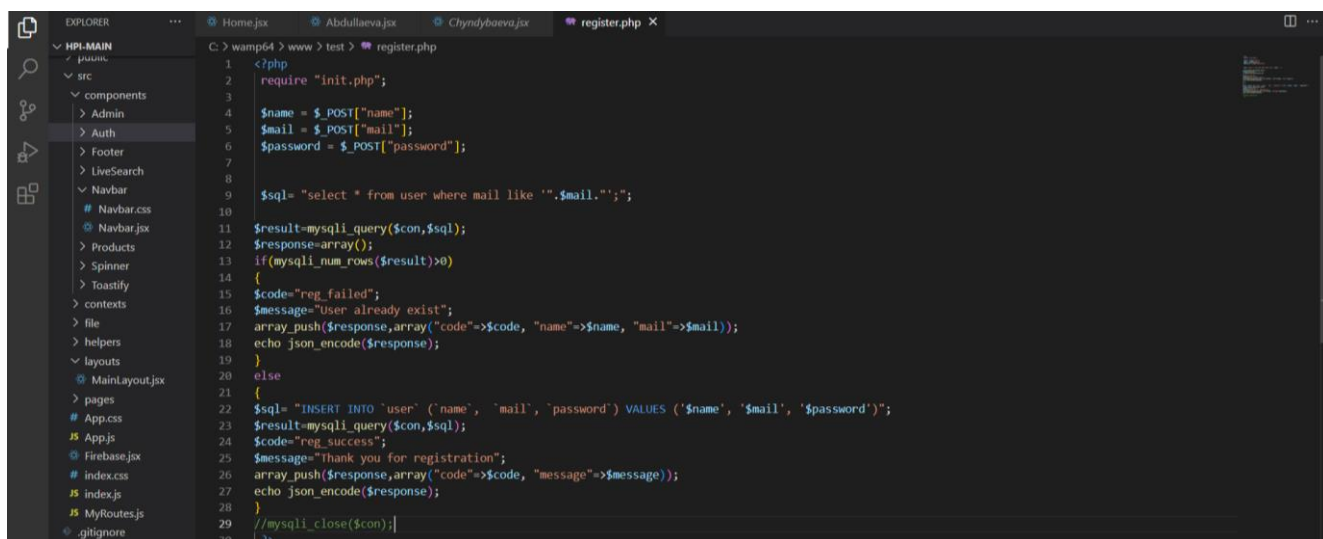


Рис.3.2.4. register.php

На этом этапе была выполнена работа по созданию веб-ресурса, вывод всей проделанной работы описывается в заключении.

3.3. Пользовательская часть

Пользовательская часть сайта позволяет пользователю взаимодействовать с ресурсами сайта а также получать нужную и актуальную информацию

На главной странице пользователи могут получить подробную информацию о кафедре. А также в данной странице располагается шапка сайта, где имеются меню-ссылки, нажав на которые можно перейти в разделы о истории, науке, новости, фотогалерею, партнеров, и сотрудников кафедры как показано на рис.3.3.1.

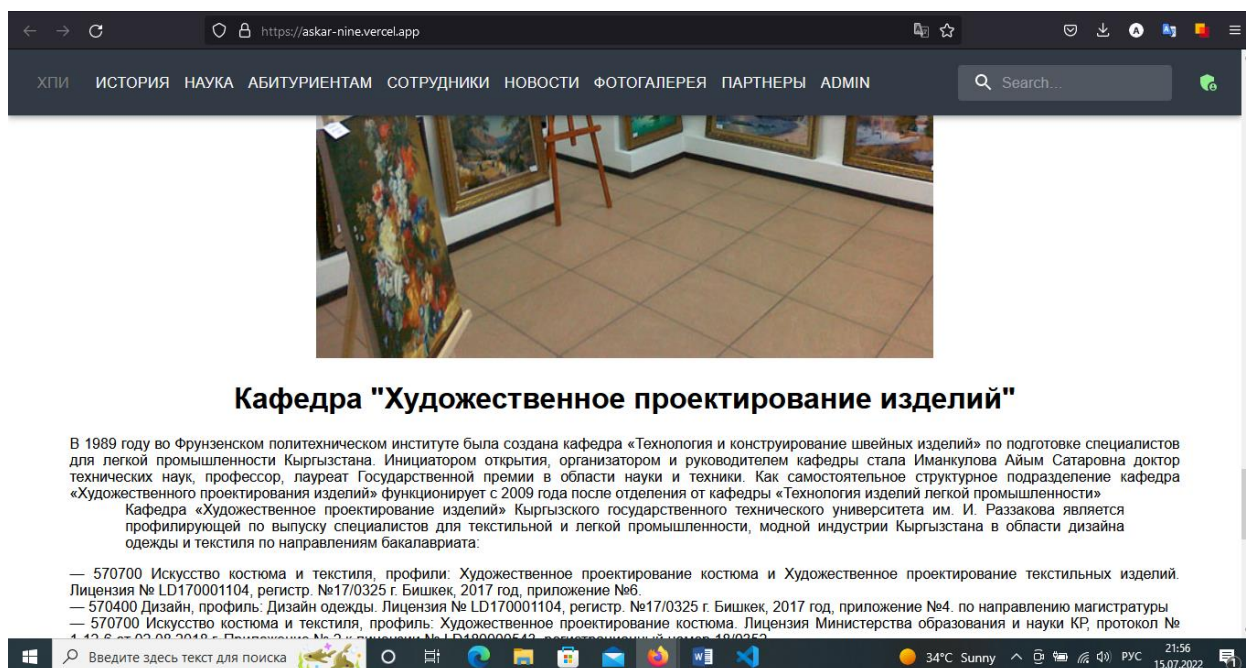


Рис.3.3.1. Главная страница

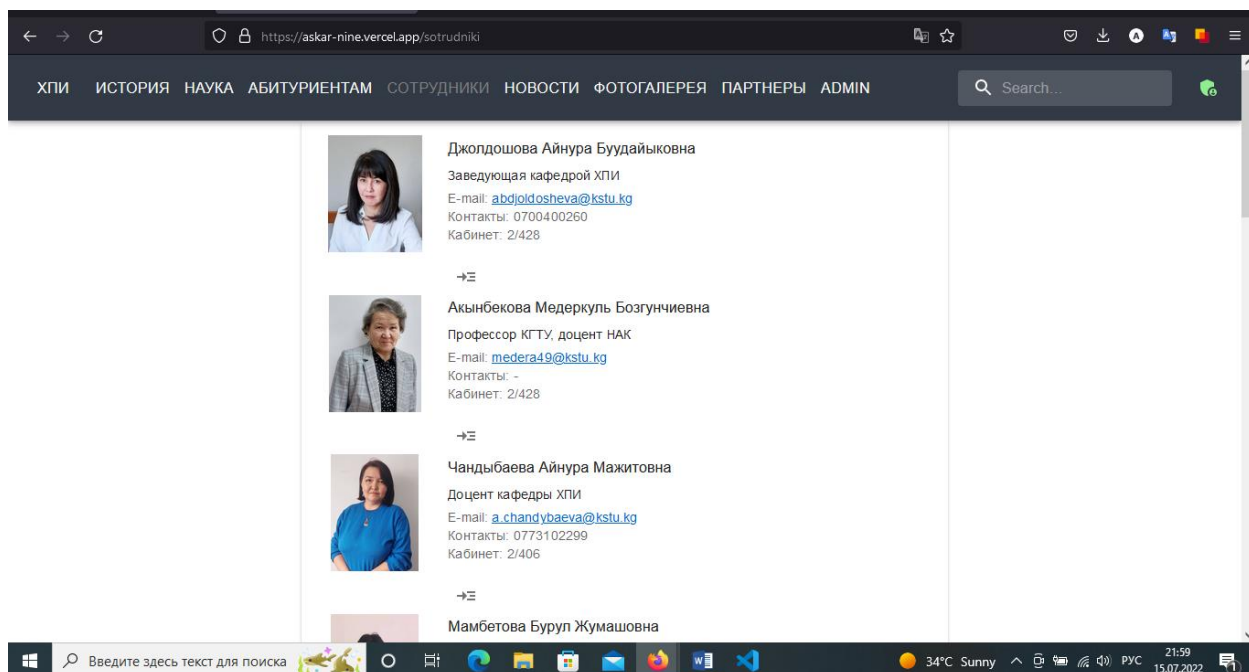


Рис.3.3.2. Окно сотрудников

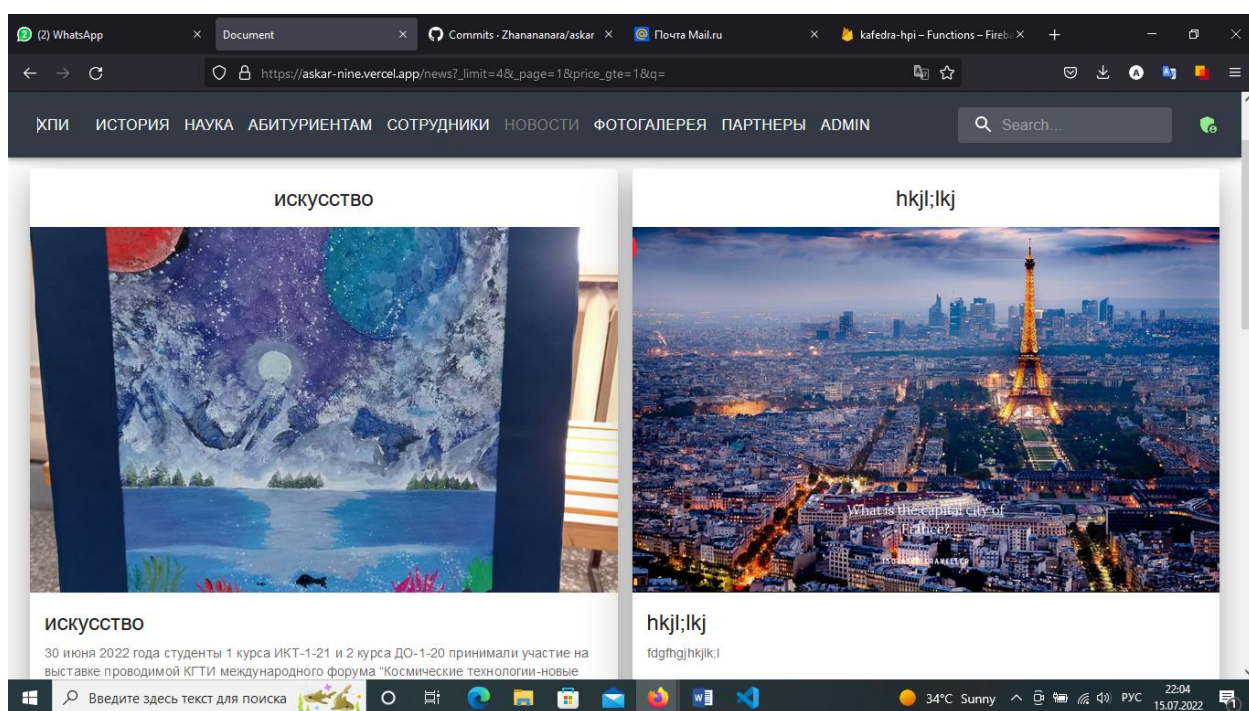


Рис.3.3.3. Окно новостей

Заключение

В результате выполнения курсовой работы были решены следующие поставленные задачи:

- обзор существующих платформ и средств разработки
- описать информационные технологии для создания веб сайта
- реализовать веб сайт на основе изученного материала

В результате работы был разработан информационный сайт для кафедры художественного проектирования изделий. Создана с целью предоставления полной информации о кафедре, а также о ее сотрудниках для любого рода посетителей, в том числе учащихся и абитуриентов.

Вся текстовая информация, предоставленная на сайте, легко управляема, обновляема и удаляема. Так же на локальном уровне используется база данных MySQL (WampServer) для осуществления регистрации, но в перспективе будет использоваться для добавления новостей и фотографий. И дальнейшая загрузка на хостинг.

Список литературы

1. <https://ru.wikipedia.org/wiki/%D0%A1%D0%B0%D0%B9%D1%82>
2. <https://swsu.ru/structura/up/umcpkip/folder1/>
3. <https://swsu.ru/structura/up/umcpkip/folder1/>
4. <https://habr.com/ru/post/490754/>
5. https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%B3%D0%BB%D0%B0%D0%B2%D0%BD%D0%B0%D1%8F_%D1%81%D1%82%D1%80%D0%B0%D0%BD%D0%B8%D1%86%D0%B0
6. <https://ru.wikipedia.org/wiki/>
7. <https://ru.wikipedia.org/wiki/HTML>
8. <https://ru.wikipedia.org/wiki/%D0%A1%D0%B0%D0%B9%D1%82>

ПРИЛОЖЕНИЕ 1. ЛИСТИНГ

```
import React from "react";
import { Route, Routes } from "react-router-dom";
import MainLayout from "./layouts/MainLayout";
import Home from "./pages/Home";
import About from "./pages/About";
import Admin from "./pages/Admin";
import Products from "./pages/Products";
import List from "./components/Admin/List/List";
import Add from "./components/Admin/Add/Add";
import Edit from "./components/Admin/Edit/Edit";
import Login from "./pages/Login";
import Register from "./pages/Register";
import RequireAuth from "./components/Auth/RequireAuth";
import NoMatch from "./pages/NoMatch";
import Nayka from "./pages/nayka.jsx";
import Partners from "./pages/partners.jsx";
// import News from "./pages/news.jsx";
import Sotrudniki from "./pages/sotrudniki.jsx";
import Galery from "./pages/galery";
import Abitur from "./pages/abitur";
import ProdDetail from "./pages/ProdDetail";
import Chyndybaeva from "./pages/Chyndybaeva";
import Joldosheva from "./pages/Joldosheva";
import Akynbekova from "./pages/Akynbekova";
import Halanskaya from "./pages/Halanskaya";
import Asanakunov from "./pages/Asanakunov";
import Abdullaeva from "./pages/Abdullaeva";
import Moldosaeva from "./pages/Moldosaeva";
import Yugai from "./pages/Yugai";
import Mamytova from "./pages/Mamytova";
import Suiutbekova from "./pages/Suiutbekova";
import Mambetova from "./pages/Mambetova";

const MyRoutes = () => {
  return (
    <Routes>
      <Route element={<MainLayout />}>
        <Route path="/" element={<Home />} />
        <Route path="/about" element={<About />} />
        <Route
          path="/admin"
          element={
            <RequireAuth>
              <Admin />
            </RequireAuth>
          }
        />
      </Routes>
    )
  }
}
```

```

    <Route index element={<List />} />
    <Route path="add" element={<Add />} />
    <Route path="edit/:id" element={<Edit />} />
  </Route>
  <Route path="/news" element={<Products />} />
  <Route path="/products/detail/:prodId" element={<ProdDetail />} />
  <Route path="/register" element={<Register />} />
  <Route path="/login" element={<Login />} />
  <Route path="/*" element={<NoMatch />} />
  <Route path="/nayka" element={<Nayka />} />
  <Route path="/abiturientam" element={<Abitur />} />
  <Route path="/sotrudniki" element={<Sotrudniki />} />
  {/* <Route path="/news" element={<News />} /> */}
  <Route path="/partners" element={<Partners />} />
  <Route path="/photogallery" element={<Galery />} />
  <Route
    path="/chandybaeva-ainura-mazhitovna"
    element={<Chyndybaeva />}
  />
  <Route
    path="/dzholdoshova-ainura-buudaiykovna"
    element={<Joldosheva />}
  />
  <Route
    path="/akynbekova-mederkul-bozgunchievnna"
    element={<Akynbekova />}
  />
  <Route path="/mambetova-burul-zhumashevna" element={<Mambetova />} />
  <Route path="/khalanskaja-elena-sergeevna" element={<Halanskaya />} />
  <Route
    path="/asanakunov-zhoomart-shakenovich"
    element={<Asanakunov />}
  />
  <Route path="/abdullaeva-gulnara-mukashevna" element={<Abdullaeva />} />
  <Route
    path="/moldosanova-nurgul-duishshchenbekovna"
    element={<Moldosaeva />}
  />
  <Route path="/yugai-zhanna-kerimbekovna" element={<Yugai />} />
  <Route path="/mamytova-akylai-abdykaparovna" element={<Mamytova />} />
  <Route
    path="/suiutbekova-sabina-maralbekovna"
    element={<Suiutbekova />}
  />
</Route>
</Routes>
);
};

```

```
export default MyRoutes;
```