

## Содержание

Введение .....	6
1. Анализ предметной области. ....	8
1.1 Описание предметной области. ....	8
1.2 Обзор аналогов разрабатываемой системы.....	9
1.3 Прототипирование.....	16
2. Проектирование Веб-приложения.....	18
2.1 Разработка структуры приложения .....	25
2.2 Проектирование взаимодействий между элементами системы.....	27
2.3 Входные и выходные данные .....	30
3. Разработка Веб-приложения.....	31
3.1 Основные сведения о платформе для разработки .....	31
3.3 Реализация информационной составляющей .....	33
3.4 Описание логической структуры Базы данных .....	34
3.5 Реализация пользовательского интерфейса .....	41
4.Тестирование .....	48
ЗАКЛЮЧЕНИЕ .....	62
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ .....	63
ПРИЛОЖЕНИЕ А. ФРАГМЕНТЫ ИСХОДНОГО КОДА .....	64

## **Введение**

Современная IT-сфера охватывает огромное количество областей, и каждая эта область заинтересована в качественных специалистах. Но для того чтобы стать этим самым специалистом необходимо получить большое количество навыков и освоить соответствующие технические средства. Однако даже поиск информации о навыках не является ключевым вопросом в становлении специалистом. Для начала необходимо выбрать саму область, в которой человек хочет развиваться и узнавать новое.

Для того чтобы выбрать интересующую сферу деятельности абитуриенты стараются выбирать направления которые соответствуют их интересам, но зачастую бывает так, что ВУЗы дают большое количество базовых навыков в различных областях деятельности. Таким образом даже после окончания обучения у некоторых студентов могут возникнуть трудности с определением будущей деятельности.

Помимо студентов, с проблемами освоения новой профессии, и необходимости поиска информации о навыках, которые современные работодатели требуют от своих сотрудников, с аналогичными трудностями сталкиваются люди, которые хотят сменить профессию, или углубиться в уже освоенную, улучшив свои навыки.

Разработанное в дипломном проекте веб-приложение должно помочь всем людям, которые сталкиваются с вышеописанными проблемами.

Веб-приложение - клиент-серверное приложение, в котором клиент взаимодействует с веб-сервером при помощи браузера. Такая структура позволяет гибко подстраиваться под современные требования пользователя к интерфейсу и функционалу, при этом от самого пользователя ничего не требуется кроме браузера, для просмотра веб-страниц.

Поскольку на сегодняшний день разработка веб-приложений отличается стремительным развитием, данному направлению посвящаются многие исследования и публикации[1].

Создание веб-приложения состоит из нескольких этапов.

Основными из этих этапов являются:

1. Определение целей и задач проекта;
2. Разработка структуры сайта;
3. Создание макетов;
4. HTML-верстка;
5. Программирование;
6. Запуск и сопровождение.

Из этих этапов следует трудоемкость разработки в связи взаимосвязи всех этапов друг с другом. Однако гибкость веб-приложений и удобства для пользователей, которые они представляют, перечеркивает все трудности, которые могут возникнуть в процессе разработки.

## **1. Анализ предметной области.**

### **1.1 Описание предметной области.**

Многие студенты во время обучения или после его завершения задаются вопросом, в какой области они хотят дальше развиваться. На текущий момент сфера компьютерных технологий получила большое развитие в различных областях, и в каждую из этих областей требуется свой набор знаний и навыков, чтобы стать специалистом. Веб-приложение будет нацелено на помощь студентам в определение дальнейших целей и стеков разработки, которые они могут изучить.

Для изучения стеков разработки, студенты смогут ознакомиться с рекомендованной литературой, которая должна будет, как дать базовые знания работы с определенной технологии, так и ознакомить с более углубленной информацией.

Веб-приложение должно также дать информацию касательно самих стеков разработки: их сферы применения, профессии в которых стек используется, общее их направление в сфере компьютерных технологий.

Помимо готовых стеков разработки дана информация касательно конкретных “Hard skills” и “Soft skills” имеющих важное место в сфере компьютерных технологий: описание языков программирования, их области применения, литература, касающаяся описываемых навыков, описана популярность и востребованной конкретных навыков в различных сферах деятельности. Также будет представлена статистика, в которой будет видна востребованность различных навыков работодателями взятая с веб-сайтов с интернет-рекрутмента в IT-сфере.

Реализована возможность общения между участниками сайта в формате форума, а также дополнительная возможность общения между участниками сайта в виде чата используя сторонние мессенджеры.

В веб-приложении реализована система взаимодействия с пользователем, в виде различных опросах и сохранения действий пользователя и интересующих его тем.

## 1.2 Обзор аналогов разрабатываемой системы.

В интернете можно найти сервисы, которые частично реализуют возможности разработанной системы в помощи выбора направлений подготовки, однако их функционал редко направлен на студентов, а в основном это либо сайты со статьями касательно стеков разработки, или конкретных “hard skills” и “soft skills” или же сайты интернет-рекрутинга на которых пользователь может посмотреть востребованные сейчас профессии и требования к ним. Рассмотрим примеры таких систем:

### 1.Сайт Habr.ru

На рисунке 1 представлен сайт [habr.ru/hub/programming/](https://habr.com/ru/hub/programming/)[2].

На сайте habr пользователь может, как писать, так и выкладывать статьи касательно различных областей в различных сферах.

К плюсам сайта можно отнести рейтинговую систему оценивания статей по которым пользователь может отличить полезную статью от ненужной.

Однако сайт имеет не четкую направленность на сферу компьютерных технологий и не подразумевает какую-то прямую помощь в выборе направления для работы.

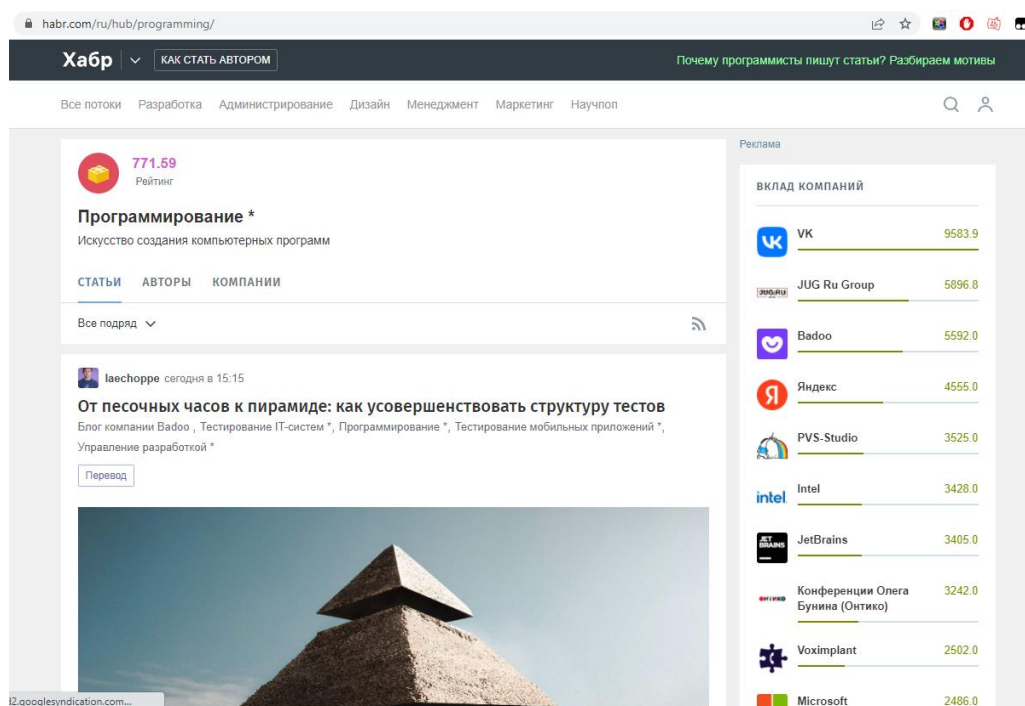


Рисунок 1–Сайт [habr.ru/hub/programming](https://habr.com/ru/hub/programming/)

## 2. Сайт headhunter.ru

На рисунке 2 представлена главная страница одно из самых популярных сайтов интернет-рекрутинга headhunter.ru [3].

Как уже понятно сайт изначально предназначен для поиска вакансий или же наоборот открытия вакансий для приема на работу. Он не предполагает прямой помощи в выборе направления для изучения и подходит лишь для анализа навыков и требований, которые работодатель может предъявить своему будущему работнику

К плюсам системы можно отнести то, что пользователь может ознакомиться с вакансиями, которые открыты в различных городах требующих различный уровень подготовки, и в дальнейшем, на основе этих данных сможет определиться с вещами которые ему стоит изучить, чтобы заняться работой на интересующих вакансиях

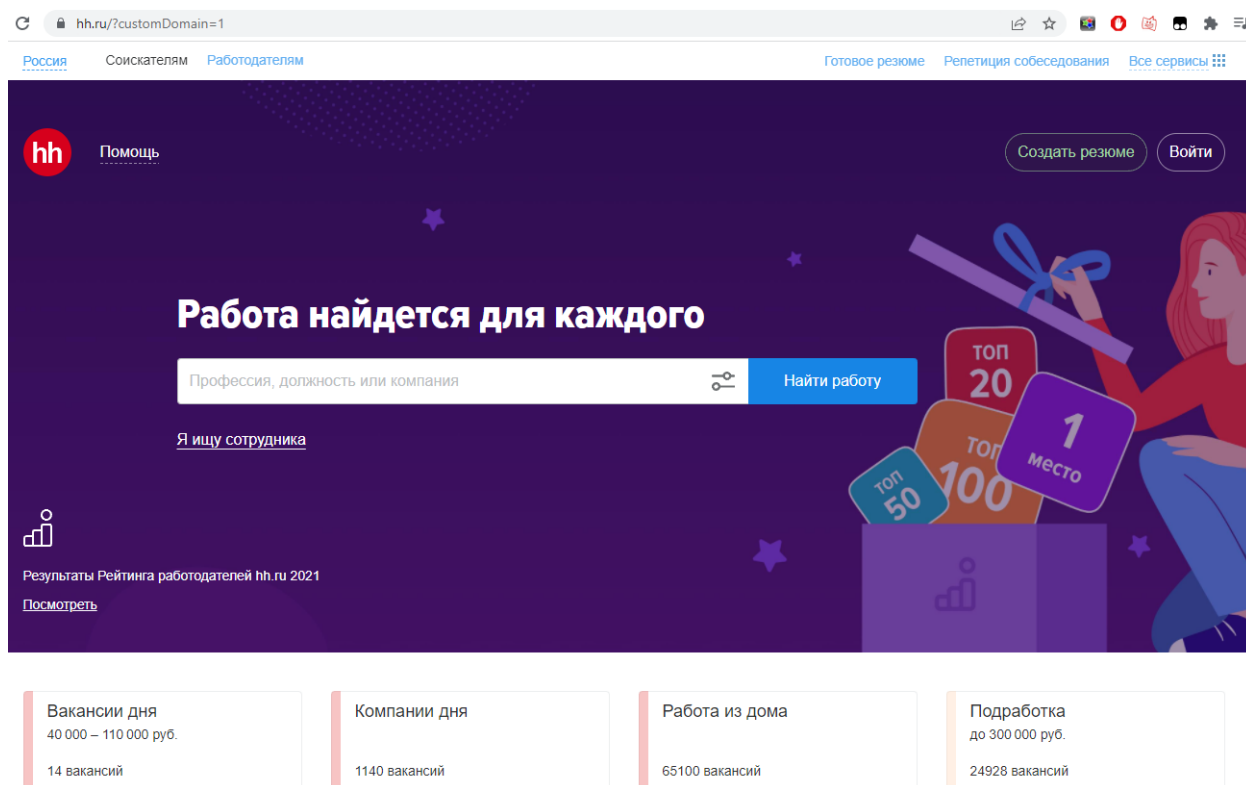


Рисунок 2– headhunter.ru

## 3. Форум itfy.org

На рисунке 3 представлена главная страница форума itfy.org[4]. Форум предполагает создание различных тем от IT-специалистов на разную

тематику. То есть пользователь может сам задавать различные вопросы таким же пользователем, которые могут разбираться в интересующей его теме. Или же может наоборот просматривать темы, в которых люди описывают их рабочий опыт или дают советы касательно различных областей информационных технологий.

Однако форумная направленность сайта подразумевает взаимодействие пользователей путем общения в специальных темах, то есть на интересующий вопрос можно попросту не получить ответ в виду отсутствия интереса к стартовой теме форума.

Но вместе с этим, потенциально можно получить гораздо больше полезной информации от людей которые уже прошли путь от выбора стека разработки до становления специалистами в определенной области.

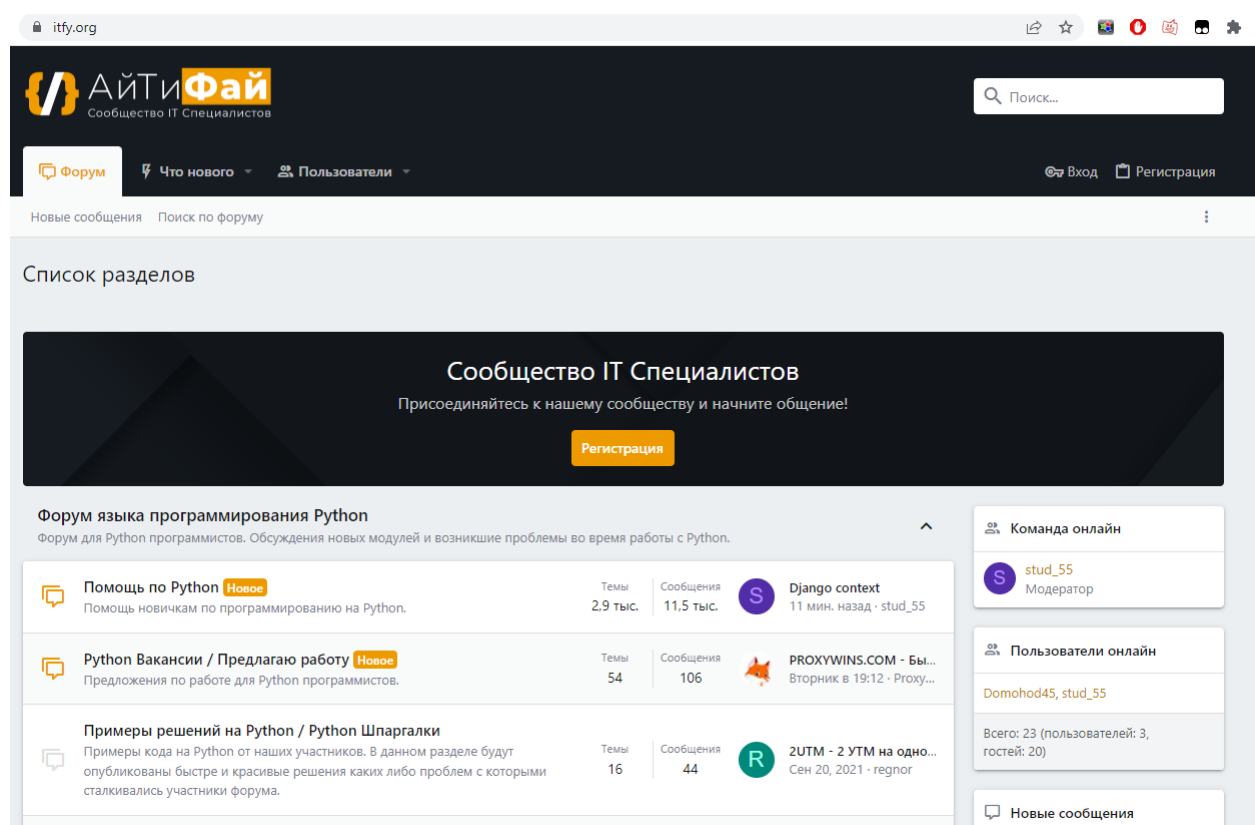


Рисунок 3–Форум itfy.org

#### 4. Сайт vuzopedia.ru.

На рисунке 5 предсавлена главная страница сайта vuzopedia[5].Сайт предназначен для помощи абитуриентам в выборе направлений подготовки в вуз на основании баллов ЕГЭ.

На сайте есть удобные калькуляторы ЕГЭ, по которым пользователь может посмотреть различные направления ВУЗов в которые абитуриент может поступить. Так же существует обратный вариант, в котором будущие студенты могут на основе интересующих их программ подготовки лучше определиться с предметами, которые необходимо сдать на ЕГЭ.

Тем не менее, сайт предназначен исключительно для абитуриентов, и не учитывает различия учебных планов, у разных вузов которые имеют одинаковые направления подготовки. Так же не очевидно, какие конкретные технические навыки, и в каком объеме получит абитуриент после выпуска из ВУЗа.

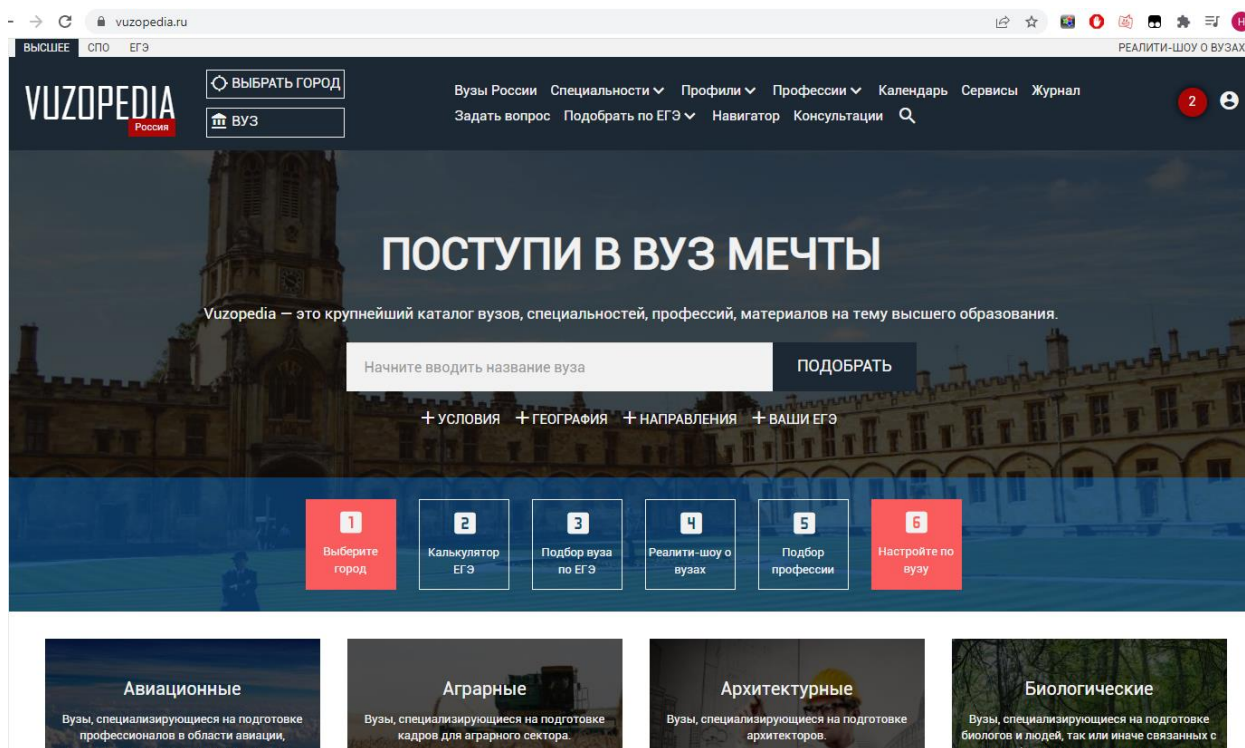


Рисунок 4– vuzopedia.ru

##### 5. Сайт opensource.com

Зарубежный сайт на английском языке [6]. На котором пользователь может посмотреть описание различных языков программирования и их сравнения в различных областях применения.

Основная тематика сайта соответствует разработанной системе, пользователь может в различных статьях, разбитых на соответствующие тематики посмотреть из чего строятся современные востребованные в IT-



сфере профессии. Сами статьи четко разграничены по смысловому содержанию и относятся к своим темам, например DevOps разработка, программирование на различных языках и т.д.

Однако основной язык сайта английский, и основные “гайды” касательно различных профессий весьма малы по объему предлагаемой литературы, а самих этих “гайдов” не так много, и затрагивают буквально 3-4 современных профессии.

Каждая система имеет свои достоинства и недостатки, лучше всего это представлено в таблице 1.

Таблица 1 – Аналоги разрабатываемой системы.

Критерии	Habr.ru	headhunter.ru	itfy.org	vuzopedia.ru	opensource.com	Разработанная система
Наличие информации касательно конкретных стеков разработки	+	-	+	-	+	+
Доступность веб-версии приложения	+	+	+	+	+	+

Продолжение таблицы 1

Критерии	Habr.ru	headhunte r.ru	itfy.org	vuzopedi a.ru	opensour ce.com	Разрабо танная система
Возможност ь проанализи ровать современну ю область интересов в ИТ-сфере	+	+	-	-	+	+
Готовые статьи содержащие информаци ю о интересую щих работодател я навыках	+	-	-	-	+	+

Продолжение таблицы 1

Критерии	Habr.ru	headhunte r.ru	itfy.org	vuzopedi a.ru	opensour ce.com	Разрабо танная система
Возможно сть общения между пользоват елями касательн о интересу ющих тем	+	-	-	-	+	+
Доступно сть сайта на русском языке	+	+	+	+	-	+

Таблица наглядно демонстрирует различия в существующих аналогов разрабатываемого веб-приложения. Итоговый продукт подразумевает некое слияние вышеназванных сайтов и его направленность на помощь студентов в определение их будущего направления для изучения, чтобы стать специалистом в области, которую они также смогут выбрать при помощи статей на сайте.

**Предлагаемый способ решения задачи и его преимущества по сравнению с известными аналогами**

- Приложение должно быть доступно на русском языке
- Доступна возможность общения между пользователями сайта

- Сайт представляет готовые статьи касающиеся готовых стеков разработки или интересующих работодателя навыков которые затем, позволяют четко проанализировать современный рынок IT профессий
- Возможность оповещения клиентов посредством сообщений.

По сравнению с аналогами система должна получиться наглядной в использовании, доступной на русском языке. Дающей четкое представление о современном IT рынке

### 1.3 Прототипирование

Разрабатываемое веб-приложение предназначено для помощи студентам в выборе курсов изучения, и описывает готовые стеки разработки и навыки которые необходимы работодателям в современной IT-сфере

Веб-приложение - это клиент-серверное приложение, состоящее из 2 частей:

1)Серверная часть должна иметь следующие возможности.

- Получать запросы от клиентов и интерпретировать их назначение;
- Выполнять обработку операций согласно логике запроса;
- Просмотр интересующих пользователю статей;
- Взаимодействовать с базами данных для хранения информации и обеспечения доступа к ней;
- Отправлять ответ клиенту на его запрос с указанием статуса и результата запроса.

Серверная часть веб-приложения состоит из нескольких компонентов, обычно называемых уровнями или слоями приложения.

Наиболее распространенные слои веб-приложения:

**Уровень представления.** Этот уровень относится к пользовательскому интерфейсу приложения. Пользовательский интерфейс веб-приложения включает код HTML, CSS и JavaScript, который запускается в браузере для отображения веб-страниц, составляющих интерфейс приложения, и

управления их динамическим поведением при взаимодействии пользователя со страницей.

**Прикладной уровень.** Этот слой обрабатывает логику, которая обеспечивает функционирование веб-приложения. Тут выполняются любые операции и алгоритмы, обеспечивающие работу бизнес-функций приложения. Например, если пользователь пытается запланировать возврат товара в приложении для покупок, бизнес-логика, скорее всего, проверит, что крайний срок возврата не прошел, запросит у пользователя желаемый метод возврата и обновит статус заказа. Этот тип бизнес-логики может обрабатываться на стороне клиента и на стороне сервера, но обычно обрабатывается на стороне сервера, поскольку он тесно связан с динамическими данными приложения.

**Уровень данных.** Этот уровень хранит, организует и управляет доступом к данным приложения с использованием базы данных. Реализуется в коде на стороне сервера.

Серверная часть приложения реализована на базе сервера Apache, взаимодействие с которым реализовано при помощи phpMyAdmin, и написана с использованием языков php и javascript. Базой данных выбрана MySQLServer.

2) Клиентская часть веб-приложения (Front-end) – интерактивная часть программы, исполняемая в веб-браузере на компьютере, смартфоне или планшете пользователя. Она реализует пользовательский интерфейс веб-приложения и загружается на устройства в виде динамических веб-страниц. Веб-приложения запускаются на любых устройствах и операционных системах, где есть интернет-браузеры. Для разработки Front-end частей веб-приложений привлекают разработчиков, знакомых с технологиями React, Vue.js, Angular, JavaScript, HTML, CSS, Ajax и другими.

В случае разработанной системы были использованы технологии HTML, CSS, JavaScript, Ajax, DOM.

## 2. Проектирование Веб-приложения

В веб-приложении разработана система регистрации и авторизации пользователей, данные которых, заносятся в базу данных.

Разработанное веб-приложение имеет 3 вида пользователей:

- 1) Администратор. Данный режим пользователя назначается разработчиком. Помимо всех функций доступных обычным пользователям, он так же может заниматься администрированием форума, удаляя некорректные сообщения.
- 2) Пользователь. Данный тип пользователя проходит регистрацию на сайте, а затем входит при помощи своего логина и пароля. Он может писать сообщения на форуме и создавать там новые темы, а так же добавлять интересующие его статьи в список понравившихся. Так же этот тип пользователя участвует в реализации алгоритма релевантных статей, которые формируются путем опросов и общего интереса к статьям.
- 3) Гость. Вид пользователя, который не прошел регистрацию на сайте. Данная категория может просматривать весь список статей на сайте и просматривать сообщения на форуме, однако писать туда они не могут. Статьи выводятся в произвольном формате, не основываясь на интересах пользователя.

Пароли пользователей шифруются при помощи алгоритма sha1 с добавлением случайной переменной-токеном. При авторизации пользователя в целях защиты, введенный им пароль складывается с хранящимся в базе данных токеном и дешифруется применением к нему sha1. Если при дешифровке полученный пароль равен хэшу хранящемуся в базе данных то пользователь авторизуется. Все данные пользователей защищены т.к. все данные лежат на стороне сервера с базой данных, и все запросы разработаны с защитой от sql инъекций.

Для каждого вида пользователя разработано единое веб-приложение с интуитивно-понятным интерфейсом, пользоваться которым, можно не устанавливая какое-либо дополнительное ПО, необходим только браузер.

Как и все сайты или интернет сервисы, веб-приложение будет реализовано на основе Клиент-серверной архитектуры. В такой архитектуре, присутствует 3 главных компонента: клиент, сервер, база данных.

С клиентом взаимодействует пользователь путем интерфейса. Вся логика запросов, программный код, и обработка запросов происходит на сервере и не нагружает устройства пользователей. На базе данных хранятся данные касательно всей системы, по которым можно делать быстрые выборки нужной информации, кроме того все данные надежно сохранены и не будут потеряны при перезагрузке системы[7]. Взаимодействие между компонентами устроено таким образом, что пользователь сначала посылает запрос серверу, затем на сервере этот запрос обрабатывается и получает данные из базы данных, и затем сервер возвращает нужные пользователю данные. Таким образом, можно понять, что клиент-серверная архитектура имеет более сложную структуру по сравнению с архитектурой обычных, статических сайтов, они предлагают пользователю активное взаимодействие путем генерации HTML страниц которые формируются в зависимости от запросов пользователя.

Для того чтобы приложение выполняло это самое активное взаимодействие оно имеет следующие характеристики:

- интуитивно понятный интерфейс, доступный пользователю;
- авторизация и регистрация пользователей;
- просмотр интересующих пользователю статей;
- взаимодействие между сайтом и пользователем путем различных опросов или сохранения интересующих пользователю тем ;
- Вывод статистики касательно популярных в современной IT-сфере направлений разработки и требуемых работодателям навыкам.

Наглядно увидеть суть работы клиент-серверной архитектуры можно на рисунке 5

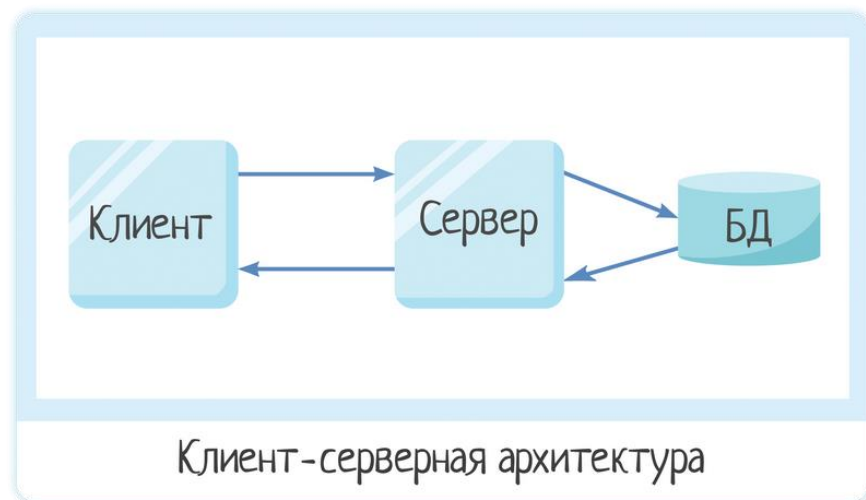


Рисунок 5 – Протокол взаимодействия пользователя и приложения

Для полного понимания функциональных требований приведена расширенная диаграмма вариантов использования (прецедентов), которая представлена на рисунке 6.

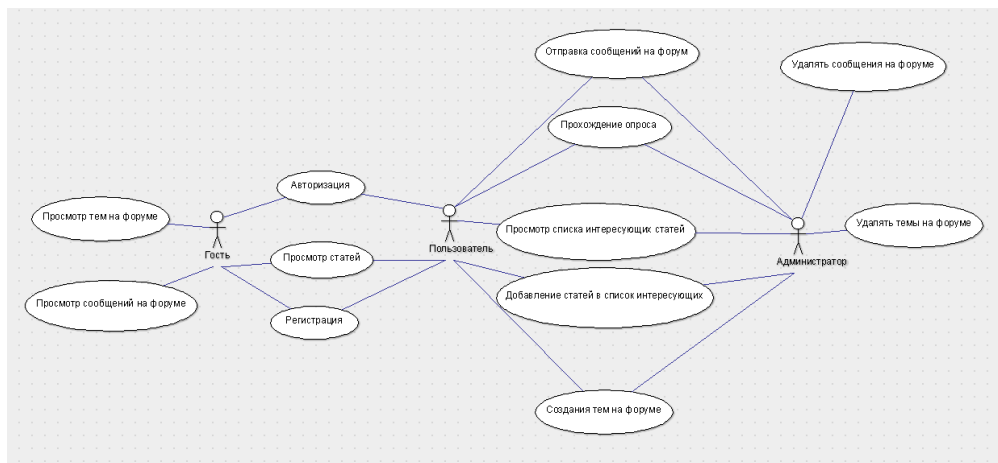


Рисунок 6 – Диаграмма прецедентов

Актерами разрабатываемой информационной системы являются:

- Пользователь.
- Администратор.
- Гость.

Диаграмма прецедентов – это диаграмма вариантов использования в UML, которая позволяет описать типичные взаимодействия пользователей



системы с самой системой и предоставит описание работы ее функций. Прецедент — вариант использования или возможность моделируемой системы (часть её функциональности). С помощью прецедентов можно описать поведения системы с точки зрения пользователя, с такой степенью детализации, что разработчики могут легко спроектировать систему изнутри. Можно отметить что, нотация диаграммы вариантов использования настолько проста, что даже неподготовленный пользователь может понять их значение и помочь уточнить некоторые моменты.

Пользователю будут доступны следующие возможности:

- Просмотр статей
- Регистрация
- Добавление статей в список понравившихся
- Прохождение опроса с целью выявления интересующей тематики
- Создание тем на форуме
- Авторизация
- Отправка сообщений на форум
- Просмотр списка статей составленного на основе интересов

Описание вариантов использования для «Пользователь» представлено в таблице 2.

Таблица 2. – Описание вариантов использования для Пользователя

Действующее лицо	Цель	Описание вариантов использования
Пользователь	Авторизация	Пользователь, после ввода логина и пароля может авторизоваться в системе, если он уже зарегистрирован в системе
Пользователь	Регистрация	Пользователь может зарегистрироваться под новым логином например для отдельного списка понравившихся статей

Продолжение таблицы 2

Действующее лицо	Цель	Описание вариантов использования
Пользователь	Просмотр статей	Пользователь может список статей на сайте, искать нужные статьи при помощи поиска по ключевым словам
Пользователь	Добавление статей в список понравившихся	Пользователь может нажать на кнопку “Понравилось ” и затем статья добавиться в список понравившихся статей который пользователь может посмотреть на личной странице
Пользователь	Прохождения опроса с целью выявления интересующей тематики	Пользователь может пройти на сайте опрос из результатов которого будет сформирован список релевантных для пользователя материалов
Пользователь	Создание тем на форуме	Пользователь может создавать на странице со списком тем форума новую тему
Пользователь	Отправка сообщений на форум	Клиент может отправить сообщение в конкретную тему на форуме
Пользователь	Просмотр списка статей составленного на основе интересов	После прохождения опроса для выявления интересов, статьи на сайте будут показываться на основе полученных в результате опроса данных

Администратору доступны все возможности обычного пользователя, но кроме них он может:

- Удалять сообщения на форуме
- Удалять темы на форуме

Описание вариантов использования для «Администратор» представлено в таблице 3.

Таблица 3. – Описание вариантов использования для Администратора

Действующее лицо	Цель	Описание вариантов использования
Администратор	Удалять сообщения на форуме	Администратор может модерировать сообщения на форуме, удаляя некорректные
Администратор	Удалять темы на форуме	Администратор может модерировать список тем форума, удаляя некорректные

Гость может пользоваться ограниченным функционалом обычного пользователя, однако ему доступны только:

- Просмотр статей
- Просмотр тем на форуме
- Просмотр сообщений на форуме
- Регистрация
- Авторизация

Описание вариантов использования для «Гость» представлено в таблице 4.

Таблица 4. - Описание вариантов использования для Гостя

Действующее лицо	Цель	Описание вариантов использования
Гость	Просмотр статей	Пользователь может список статей на сайте, искать нужные статьи при помощи поиска по ключевым словам
Гость	Просмотр тем на форуме	Гость может просматривать темы на форуме, без возможности создания новых
Гость	Просмотр сообщений на форуме	Гость может просматривать сообщения на форуме, без возможности отправки новых
Гость	Регистрация	Пользователь может зарегистрироваться на сайте получив все возможности Пользователя
Гость	Авторизация	После регистрации Гость может зайти на сайт под своим логином и получить возможности Пользователя

Исходя из таблиц 2-4, видно, что гостевой режим пользователя дает доступ лишь к базовым функциям приложения. Для полноценного использования системы пользователям необходимо зарегистрироваться. Однако функционал класса Администратор можно получить лишь от разработчика системы.

## 2.1 Разработка структуры приложения

Структура клиент-серверного приложения состоит из 3 частей.

Структура разработанного веб-приложения и работа его частей представлена на рисунке 7.

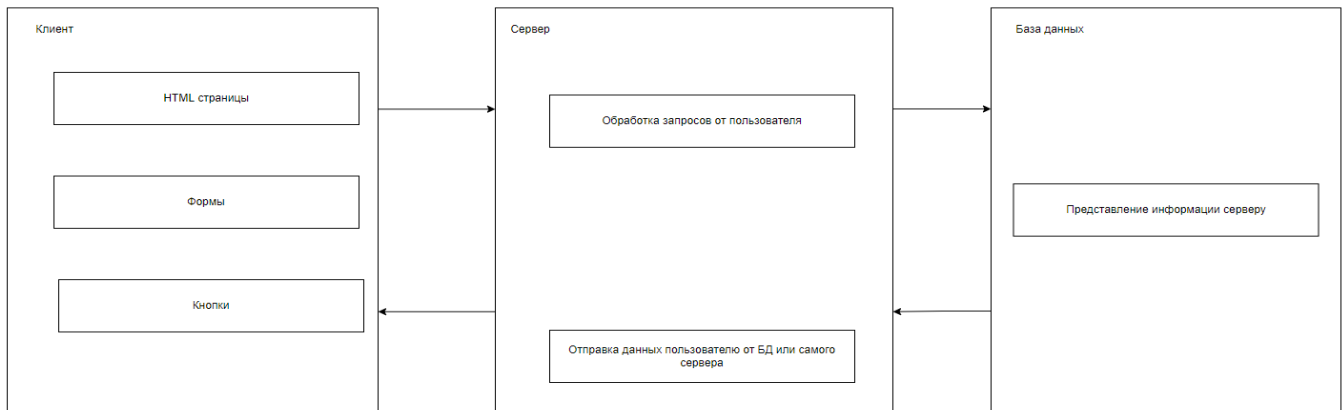


Рисунок 7 – Структурная схема разработанного веб-приложения

Рассмотрим подробнее каждый компонент.

Подсистемы клиента описаны ниже:

- Пользователь взаимодействует с приложением на HTML страницах.
- Взаимодействие с сервером происходит путем заполнения форм и нажатия на кнопки
- В приложении предусмотрено 8 основных типов страниц:
  1. Главная. Начальная страница, на которой расположены все статьи на сайте и переходы на остальные страницы
  2. Регистрация. Страница для регистрации пользователей
  3. Авторизация. Страница для авторизации пользователей
  4. Статьи. Полный список страниц на сайте
  5. Поиск. Страница для поиска интересных статей
  6. Статья. Страница для отображения статей или опросов
  7. Форум. Страница для отображения тем форума.
  8. Тема форума. Страница для отображения сообщений касательно темы форума.

Архитектура веб-приложения разработана на основе паттерна MVC

Впервые паттерн MVC появился в языке SmallTalk в конце семидесятых. Собственно задача была хорошо всем знакомая, надо было придумать архитектурное решение, которое позволяло бы манипулировать графическими представлениями данных некоего приложения, таким образом, чтобы изменение Представления этих данных не влияло на бизнес-логику и данные (Модель) приложения, а так же, чтобы была возможность иметь несколько Представлений для одной Модели. Таким решением и стал паттерн MVC, идея которого родилась в недрах Xerox PARK, и получила свое первое публичное упоминание в документации к SmallTalk'80. В классическом варианте, MVC состоит из трех частей, которые и дали ему название.[8]Общую схему паттерна можно посмотреть на рисунке 9.

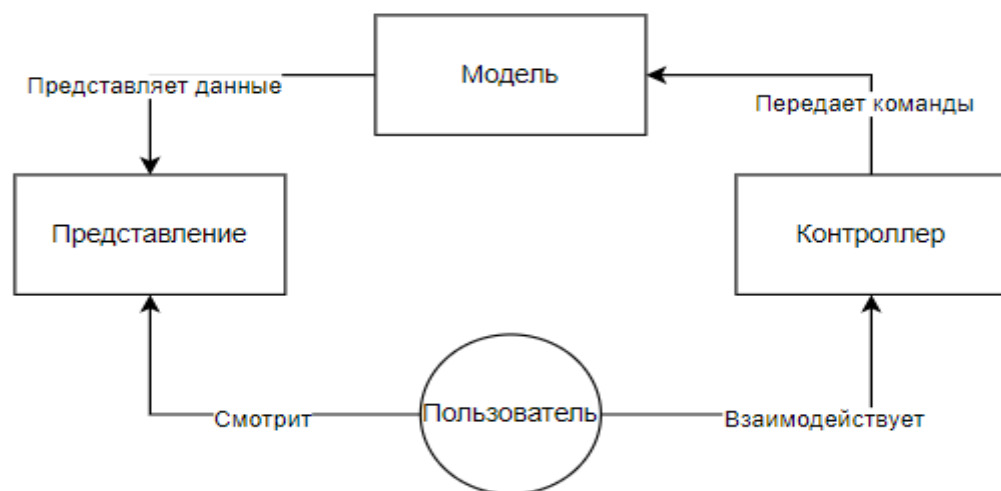


Рисунок 8- Общая схема MVC паттерна

В разработанном проекте компоненты паттерна представлены следующими элементами

**Модель:** Серверная часть Web-приложения.(базы данных,скрипты)

**Представление:** Пользовательский интерфейс Web приложения. (запросы, формы,кнопки)

**Контроллер:** Клиентская часть web-приложения.(Веб-браузер)

Схему паттерна MVC конкретно для разработанного веб-приложения можно посмотреть на рисунке 9.

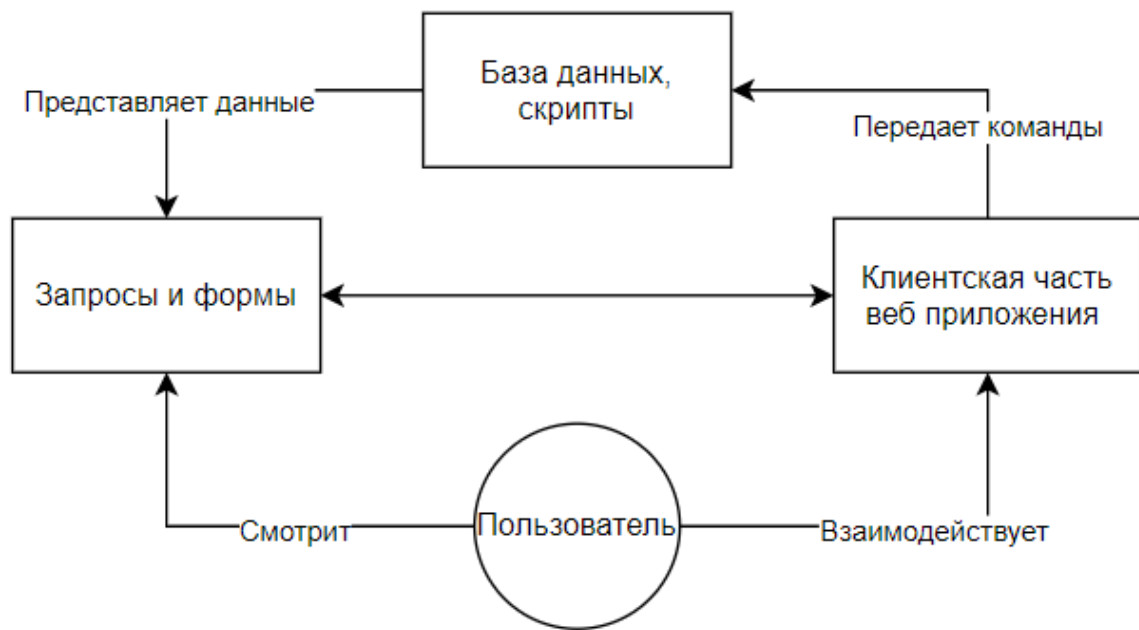


Рисунок 9. Схема MVC паттерна для разработанного веб-приложения

## 2.2 Проектирование взаимодействий между элементами системы

Для разработки курсового проекта были созданы диаграммы последовательностей, показывающие цикл работы определенных функций системы.

Диаграмма последовательности используется в основном для отображения взаимодействий между объектами в том порядке, в котором эти взаимодействия происходят. Подобно диаграмме классов, разработчики обычно думают, что диаграммы последовательности предназначены исключительно для них. Тем не менее, бизнес-персонал организации может найти диаграммы последовательности полезными для информирования о том, как бизнес работает в настоящее время, показывая, как взаимодействуют различные бизнес-объекты. Помимо документирования текущих дел организации, диаграмма последовательности операций бизнес-уровня может использоваться в качестве документа с требованиями для передачи требований к будущей реализации системы. На этапе требований проекта аналитики могут вывести варианты использования на следующий уровень, предоставив более формальный уровень уточнения. Когда это происходит,

варианты использования часто преобразуются в одну или несколько диаграмм последовательности [9].

Для разрабатываемой системы в рамках ВКР, построена диаграмма последовательности для прецедента «Регистрация». На ней представлено взаимодействие с объектами HTML страницы, такими как: кнопки и формы которые затем отправляются в базу данных.

Диаграмма последовательности для прецедента «Регистрация» представлена на рисунке 10.

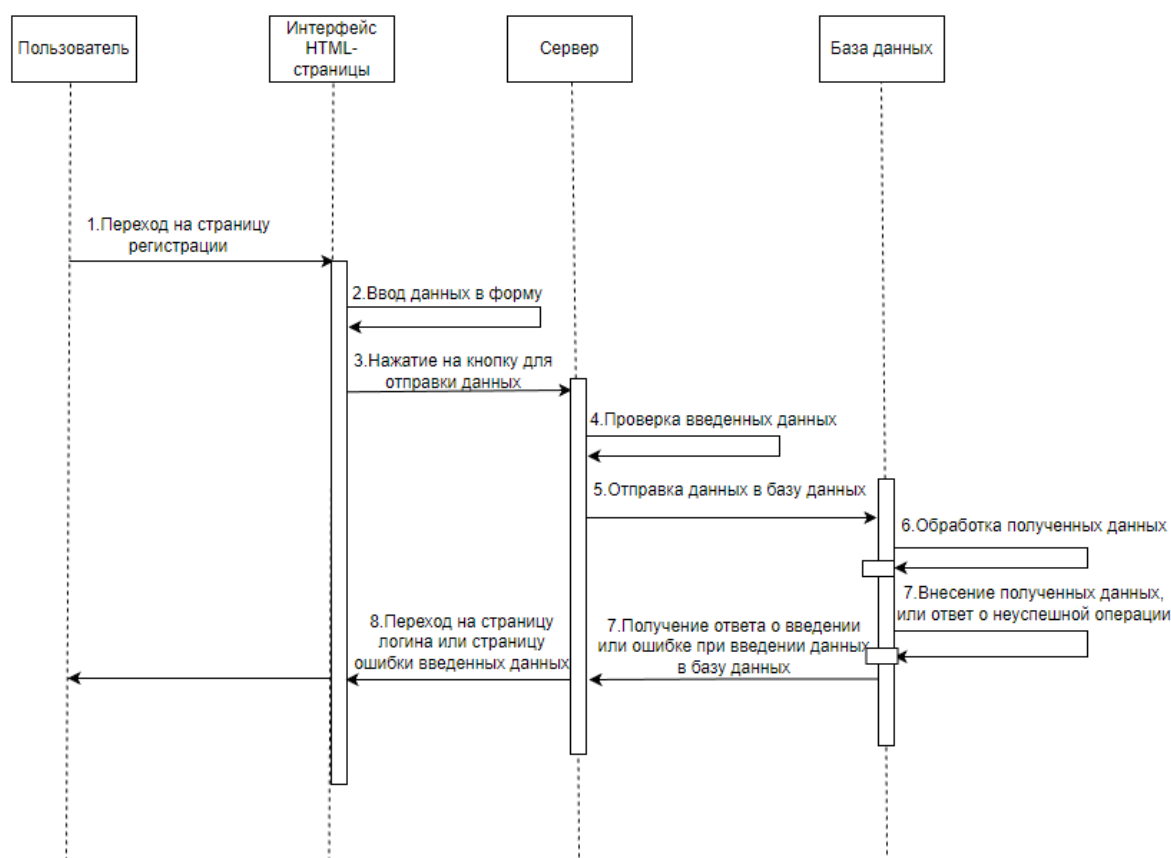


Рисунок 10 – Диаграмма последовательностей для Регистрации

«Пользователь» при заходе на сайт открывает главную HTML страницу. После клика на кнопку регистрации открывается HTML страница с формой для регистрации. После ввода данных в форму и нажатия на кнопку данные отправляются на сервер, происходит проверка корректности введенных данных, и проверка существования логина в Базе данных. Пароль пользователя шифруется при помощи алгоритма sha1 с добавлением к нему



случайного значения, полученных в результате шифрования хэш-код заносится в базу данных вместе с случайным значением который называется токен . После проверки данных они заносятся в Базу данных и пользователь переходит на страницу Авторизации.

Диаграмма последовательности для прецедента «Авторизация» представлена на рисунке 11.

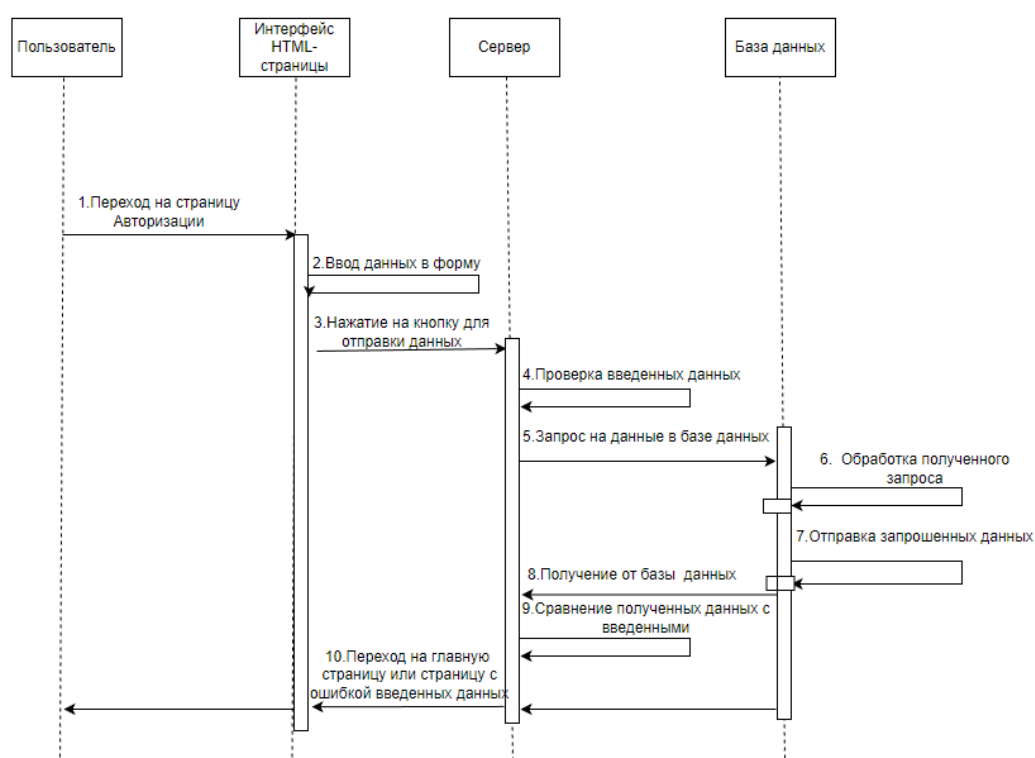


Рисунок 11 – Диаграмма последовательностей для Авторизации

Пользователь с главной страницы нажимает на кнопку перехода, на страницу авторизации. Вводит данные в форму и нажимает кнопку авторизации. Сервер проверяет введенные данные и отправляет запрос в базу данных о логине и пароле пользователя. Сервер шифрует введенный пользователем пароль с добавлением к нему токена из базы данных с хранящимся в БД хэш-кодом. После проверки данных пользователь либо авторизируется и попадет на главную страницу, либо на страницу с ошибкой при авторизации.

## 2.3 Входные и выходные данные

Разрабатываемая автоматизированная система имеет входную и выходную информацию. В качестве входных данных будут: данные пользователя, сообщения на форуме, результаты опроса. В качестве выходных: статьи, сообщения на форуме, список интересующих и понравившихся пользователю статей.

В данной главе были выделены основные пользователи информационной системы сервисного центра: Пользователь, Гость и Администратор. Определено то, что веб-приложение состоит из 3 компонентов: клиента, сервера, базы данных. В основе архитектуры лежит MVC паттерн. Определены основные прецеденты для каждого пользователя. Были составлены диаграммы последовательности для прецедентов «Авторизация», «Регистрация».

### 3. Разработка Веб-приложения

#### 3.1 Основные сведения о платформе для разработки

Для разработки веб-приложения администратора была использована единая среда разработки PhpStorm. PhpStorm разработана компанией JetBrains, и позволяет удобно писать код на PHP, Java и HTML. Кроме вышеописанных языков имеется полноценная sql консоль, которая позволяет создавать запросы к используемой разработчиком базе данных. Функционал среды разработки постоянно дополняется разработчиками, а также может быть дополнен устанавливаемыми плагинами. PhpStorm, как и все продукты JetBrains распространяется по лицензии. В зависимости от использования продукта покупать лицензии может быть на коммерческой основе, или же предоставляться бесплатно, например, для образовательных целей.

В качестве веб-сервера выбран Apache, за счет простоты использования и надежности. Кроме того, Apache позволяет свободное использование и распространение.

Основное назначение веб-сервера - это обработка запросов от клиента и выдавая им корректные данные. Веб-сервером называют как программное обеспечение, выполняющее функции веб-сервера, так и непосредственно компьютер, на котором это программное обеспечение работает.

База данных хранит большие объемы данных в различных формах. В базах данных может храниться информация разного типа: фотографии, тестовая информация и т.д.. Однако, для доступа к этой информации необходимо ПО позволяющее обращаться к этим данным, добавлять новые или изменять старые. В рамках данного ВКР была использована система управления базами данных (СУБД) MySQL. MySQL является реляционной СУБД то есть данные представляются в виде таблиц объединяемых путем различных типов связей: один к одному, один ко многим, много ко многим. структурированный набор данных. Сами таблицы состоят из строк и столбцов. Строки представляют собой отдельные объекты БД, столбцы

представляют собой атрибуты (признаки, характеристики, параметры) объектов которые затем будет иметь строки.

RНРMyAdmin веб-приложение с открытым кодом, написанное на языке RНР и представляющее собой веб-интерфейс для администрирования СУБД MySQL. Приложение просто в использовании и представляет весь необходимый базовый функционал для взаимодействия с БД. Можно использовать прямо в браузере.

### 3.2 Основные сведения о языках программирования

HTML-стандартизированный язык 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. HTML является простым в освоении и имеющий гибкое средство CSS для настройки стилей каждого тега, что удобно в создании приятного интерфейса

RНР – это широко используемый язык сценариев общего назначения с открытым исходным кодом. Говоря проще, RНР это язык программирования, специально разработанный для написания web-приложений (сценариев), исполняющихся на веб-сервере. Аббревиатура RНР означает “Hypertext Preprocessor (Препроцессор Гипертекста)”. Синтаксис языка берет начало от C, Java и Perl. RНР достаточно прост для изучения. Преимуществом RНР является предоставление веб-разработчикам возможности быстрого создания динамически генерируемых web-страниц.

JAVA – строго-типизированный язык общего назначения. Java - один из самых важных и широко применяемых языков программирования в мире на протяжении многих лет. В отличие от некоторых других языков программирования, влияние Java не только не уменьшилось со временем, а, наоборот, возросло. С момента первого выпуска он выдвинулся на передний край программирования приложений для Интернета. И каждая последующая

версия лишь укрепляла эту позицию. Ныне Java по-прежнему остается первым и одним из лучших языком для разработки веб-ориентированных приложений [10].

AJAX — Технология проектирования интерактивных пользовательских веб-приложений. Общий смысл состоит в том, что информация пользователю из базы данных выводится без обновления страницы. С помощью этой технологии например формируются элементы поиска или бесконечных лент.

DOM — Программный интерфейс для HTML и XML документов. DOM представляет собой объектную модель. Объекты в этой модели расположены в виде дерева объектов DOM. Где элементами (тэги) и текстовые строки документов являются узлами дерева. DOM может взаимодействовать с JavaScript например представляя ссылки на необходимые ему тэги в конкретных местах HTML страницы.

### 3.3 Реализация информационной составляющей

База данных в веб-приложении используется MySQL, то есть представляет собой таблицы и связи между ними. База данных diploma содержит информацию о пользователя, статьях. На основе таблиц из базы данных реализован форум. Хранится информация о типах интересов пользователя на основе которой реализован алгоритм релевантных статей.

Таблицы базы данных serviscenter:

- article – таблица содержащая статьи .
- articleImg – дополнительная таблица в которой хранятся ссылка на заглавные изображения статей.
- articletags – дополнительная таблица в которой описана основная тематика статьи. Таблица необходима для реализации алгоритма релевантности
- articletype – дополнительная таблица в которой описаны основные типы статей
- forumtheme – Таблица в которой хранятся основные темы на форуме

- likedpost – Список понравившихся пользователям статей
- message – Таблица необходимая для реализация форума. Содержит сообщения которые пользователи отправляют в соответствующие темы
- tags – Таблица хранящая основные тематики статей. Таблица необходима для реализации алгоритма релевантности
- users – Таблица хранящая информацию о пользователях
- usertags – дополнительная таблица в хранятся интересующие пользователя тематики статей.
- usertype – Таблица с описание типов пользователя

Общую схему связей в базе данных можно увидеть на рисунке 12

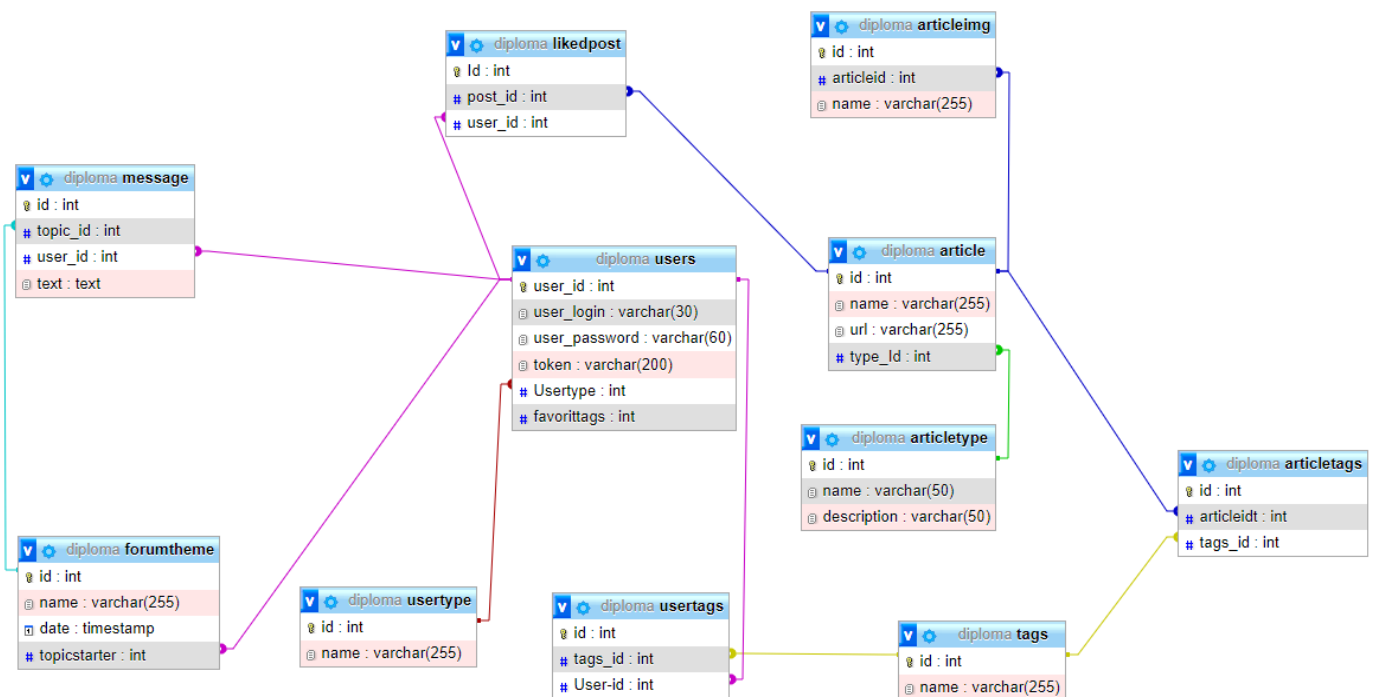


Рисунок 12-Схема связей в базе данных

### 3.4 Описание логической структуры Базы данных

В реляционных базах данных информация представлена в виде столбцов, которые являются атрибутами, и строками которые являются единичным экземпляром таблицы. В таблицах могут отсутствовать строки, но 1 столбец обязательно должен быть. Для реализации связей в СУБД

реализована система первичных и внешних ключей. Первичные ключи это поля которые однозначно идентифицирующие запись в БД. В разработанной ВКР первичные ключи всегда обозначают уникальный номер строки(её ID). Внешний ключ служит для обеспечения ссылочной ценности со связанной таблицей. В общем случае внешний ключ должен обозначать уникальный номер (то есть первичный ключ) строки в таблице, с которой осуществляется связь.

Описание всех таблиц с их атрибутами и ключами можно увидеть в таблицах 5-16.

Таблица 5-Атрибуты таблицы article

№	Название	Тип	Дополнительная информация	Описание
1	id	int	Автоувеличение	Первичный ключ
2	name	Varchar(255)		Название стааи
3	url	Varchar(255)		Ссылка на статью
4	type_Id	int		Вторичный ключ Для связи с типом статьи

Таблица 6-Атрибуты таблицы articleimg

№	Название	Тип	Дополнительная информация	Описание
1	Id	int	Автоувеличение	Первичный ключ
2	articleid	int		Вторичный ключ Для связи с статьей
3	name	Varchar(255)		Ссылка на изображение

Таблица 7-Атрибуты таблицы articletags

№	Название	Тип	Дополнительная информация	Описание
1	id	int	Автоувеличение	Первичный ключ
2	articleid	int		Вторичный ключ для связи с статьей
3	Tags_id	int		Вторичный ключ для связи с категорией статьи

Таблица 8-Атрибуты таблицы articletype

№	Название	Тип	Дополнительная информация	Описание
1	id	int	Автоувеличение	Первичный ключ
2	name	Varchar(50)		Название типа статьи
3	description	Varchar(50)		Описание типа статьи

Таблица 9-Атрибуты таблицы forumtheme

№	Название	Тип	Дополнительная информация	Описание
1	id	int	Автоувеличение	Первичный ключ
2	name	Varchar(255)		Название темы
3	date	Timestamp		Дата создания темы
4	topicstarter	int		Вторичный ключ для связи с пользователем создавшим тему



Таблица 10-Атрибуты таблицы likedpost

№	Название	Тип	Дополнительная информация	Описание
1	Id	int	Автоувеличение	Первичный ключ
2	post_id	int		Вторичный ключ для связи с статьей
3	user_id	int		Вторичный ключ для связи с пользователем

Таблица 11-Атрибуты таблицы message

№	Название	Тип	Дополнительная информация	Описание
1	id	int	Автоувеличение	Первичный ключ
2	topic_id	int		Вторичный ключ для связи с темой форума
3	user_id	int		Вторичный ключ для связи с пользователем отправляющим сообщение
4	text	text		Текст сообщения

Таблица 12-Атрибуты таблицы tags

№	Название	Тип	Дополнительная информация	Описание
1	id	int	Автоувеличение	Первичный ключ
2	name	Varchar(255)		Название категории статей

Таблица 13-Атрибуты таблицы users

№	Название	Тип	Дополнительная информация	Описание
1	user-id	int	Автоувеличение	Первичный ключ
2	user-login	Varchar(30)		Логин пользователя
3	user-password	Varchar(60)		Пароль пользователя в зашифрованном виде
4	token	Varchar(60)		Значение добавляемое к паролю пользователя при его шифровании и дешифровании
5	Usertype	int		Вторичный ключ для связи с типом пользователя

Таблица 14-Атрибуты таблицы usertags

№	Название	Тип	Дополнительная информация	Описание
1	id	int	Автоувеличение	Первичный ключ
2	tags_id	int		Вторичный ключ для связи с категориями статей
3	User_id	int		Вторичный ключ для связи с таблицей пользователей

Таблица 15-Атрибуты таблицы usertype

№	Название	Тип	Дополнительная информация	Описание
1	id	int	Автоувеличение	Первичный ключ
2	name	Varchar(255)		Название типа пользователей

Таким образом, на основании приведенных выше таблиц, видно, что структура базы данных покрывает все нужные веб-приложения и обеспечивает сохранность данных, а связи между таблицами обеспечивают взаимосвязь между информацией и путем запросов будут представлены пользователю в необходимом виде.

Кроме логики самой базы данных, веб-приложение активно использует запросы, которые позволяют вносить новые данные в таблицы, изменять данные, или просто просматривать существующую информацию.

В разработанной ВКР используются 2 основных вида запросов: на выборку и вставку.

Запрос на выборку позволяет получить нужную информацию в заранее форматированном формате. Для примера рассмотрим запрос, который формирует список сообщений на форуме в определенной теме. Текст запроса приведен на рисунке 13.

```
select message.text as текст, forumtheme.name as тема, users.user_login as имя from message, forumtheme, users
where (message.user_id=users.user_id) and (message.topic_id=forumtheme.id) and (topic_id=$ThreadID)
```

Рисунок 13-Текст запроса на выборку сообщений в нужной теме  
форума

Как видно из тела запроса, используется 3 таблицы: message, forumtheme, users. Для более удобной выборки мы заранее выбрали поля которые нам будут отображаться и как будут именоваться столбцы в новой выборке. Для организации связей между таблицами мы в части когда, начиная со слова where указываем, какие поля имеют связи, и в последней скобке кода указываем интересующую нас тему вместо переменной \$ThreadID. Результат выполнения запроса при по теме имеющий 1 порядковый номер можно увидеть на рисунке 14.

	текст	тема	имя
1	Посоветуйте библиотеки JavaScript для веб-дизайна	Библиотеки JavaScript	54321
2	React, jQuery	Библиотеки JavaScript	denchik

Рисунок 14-Результаты выполнения запроса

Второй используемый тип запросов это вставка. Данным запросом можно занести в таблицу необходимую нам информацию. Текст запроса на отправку нового сообщения в тему форума можно увидеть на рисунке 15.

```
INSERT INTO message(topic_id,user_id,text) VALUES (?, ?, ?)
```

Рисунок 15-Запрос на вставку.

В теле запроса мы указываем в какую таблицу мы хотим вставить наши данные, затем указываем столбцы которые будут заполнены и затем пишем сами данные которые затем окажутся в таблице.

Полный текст всех запросов с скриптами, в которых они используется представлен в приложении Б.

### 3.5 Реализация пользовательского интерфейса

На веб-страницах реализация визуального компонента осуществляется при помощи стандартизированного языка разметки HTML. HTML – это язык гипертекстовой разметки. Он позволяет пользователю создавать и организовывать веб-страницы и приложения в разделы, абзацы, заголовки, ссылки и блоки.

Для формирования приятного интерфейса HTML может взаимодействовать со стилями CSS которые позволяют, настроить внешний вид каждого элемента в теге. CSS также имеет возможность создавать классы стилей которые реализуют механизмы наследования и могут использоваться в разные моменты жизненного тега.

Пользовательский интерфейс web-страницы «Главная» представлен на рисунке 16.

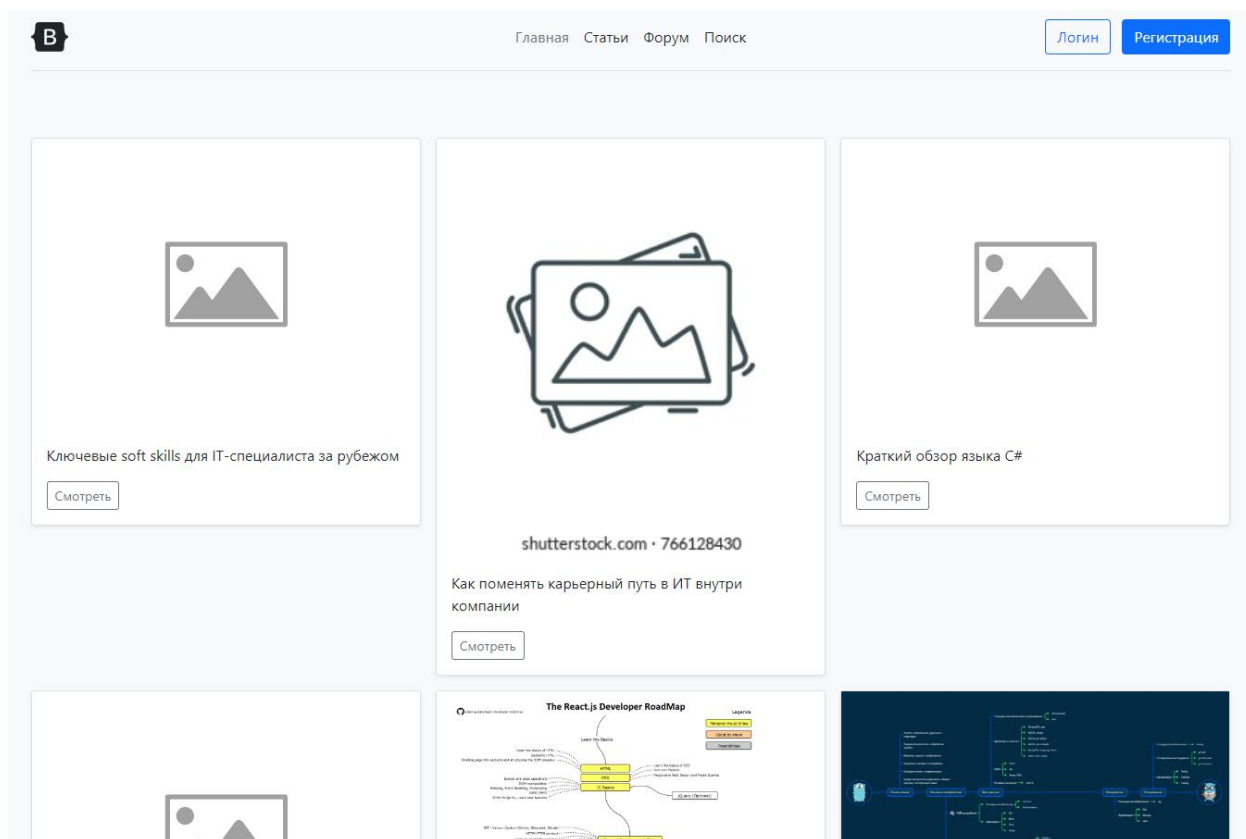


Рисунок 16 – Страница «Главная»

С главной страницы пользователь может сразу открывать интересующие его статьи

Внешний вид всех статей мало отличается друг от друга функционалом, на странице всех статей имеет информация о названии и текст касательно поставленного вопроса. Для примерного внешнего вида всех статей рассмотрим статью «Краткий обзор языка C#».

Внешний вид страницы «Краткий обзор языка C#» представлен на рисунке 17

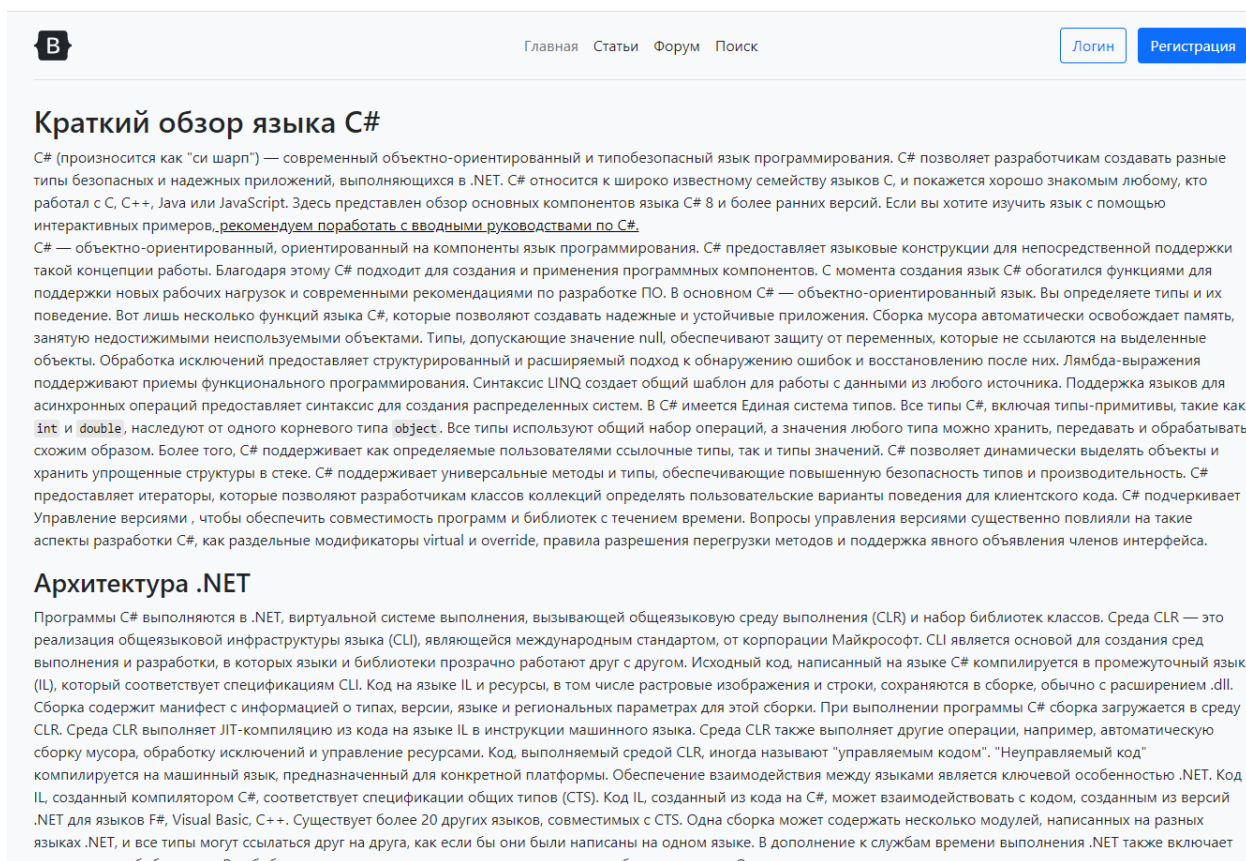


Рисунок 16-Внешний вид статьи

Однако некоторые статьи имеют другую структуру интерфейса и могут представлять собой какие-либо картинки с планами изучения определенной области IT-технологий. Примером такой статьи может быть «Roadmap.React» Roadmap в среде программистов можно назвать краткий план по изучению определенной отрасли разработки в котором описаны аспекты которые можно изучить касательно этой самой отрасли. Внешний вид страницы «Roadmap.React» представлен на рисунке 17

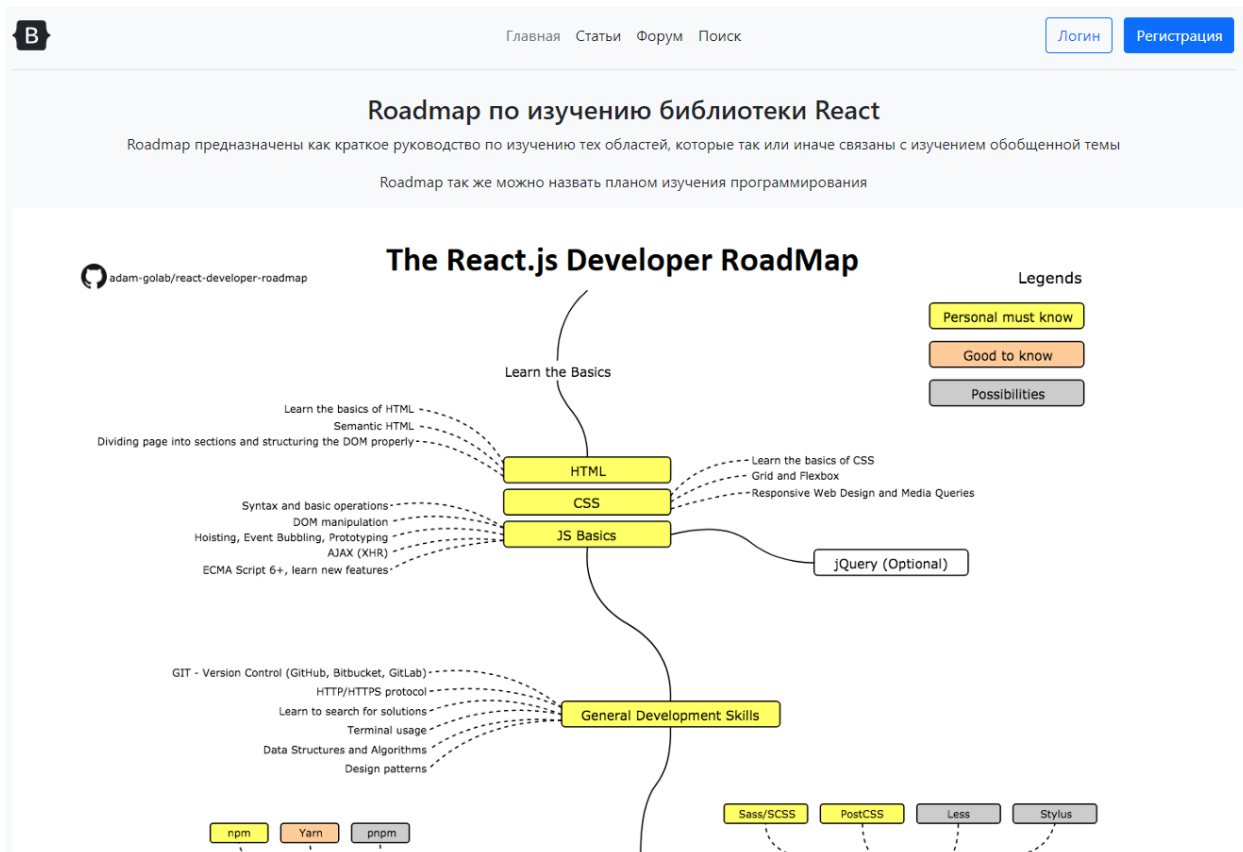


Рисунок 17-Внешний вид «Roadmap.React»

Пользователи с главной и всех второстепенных страниц могут попасть на список всех статей, нажав на кнопку статьи. Внешний вид страницы «Статьи» представлен на рисунке 18

Главная Статьи Форум Поиск

Логин Регистрация

Название	Тип
<a href="#">Ключевые soft skills для IT-специалиста за рубежом</a>	Статья
<a href="#">Как поменять карьерный путь в ИТ внутри компании</a>	Статья
<a href="#">Краткий обзор языка C#</a>	Статья
<a href="#">Освоить веб-дизайн с нуля: что читать и где учиться?</a>	Статья
<a href="#">Roadmap.React</a>	Рoadmapa
<a href="#">Roadmap.Go</a>	Рoadmapa
<a href="#">Тест.Подходит ли вам язык Python для изучения?</a>	Тест

Рисунок 18-Внешний вид страницы «Статьи»

Пользователям также доступен поиск интересующих их статей по ключевым словам или категориям. Внешний вид на рисунке 19

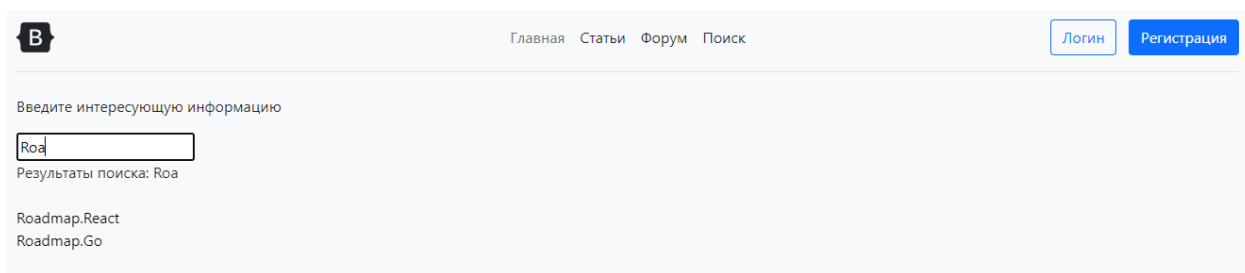


Рисунок 19-Страница поиска

Реализована система регистрации которая открывает пользователям новые возможности, такие как создание и отправка сообщений на форумы, доступ к отображению релевантных статей, добавление статей в список понравившихся, возможность пройти опрос касательно интересов пользователя. Внешний вид страницы регистрации представлен на рисунке 20

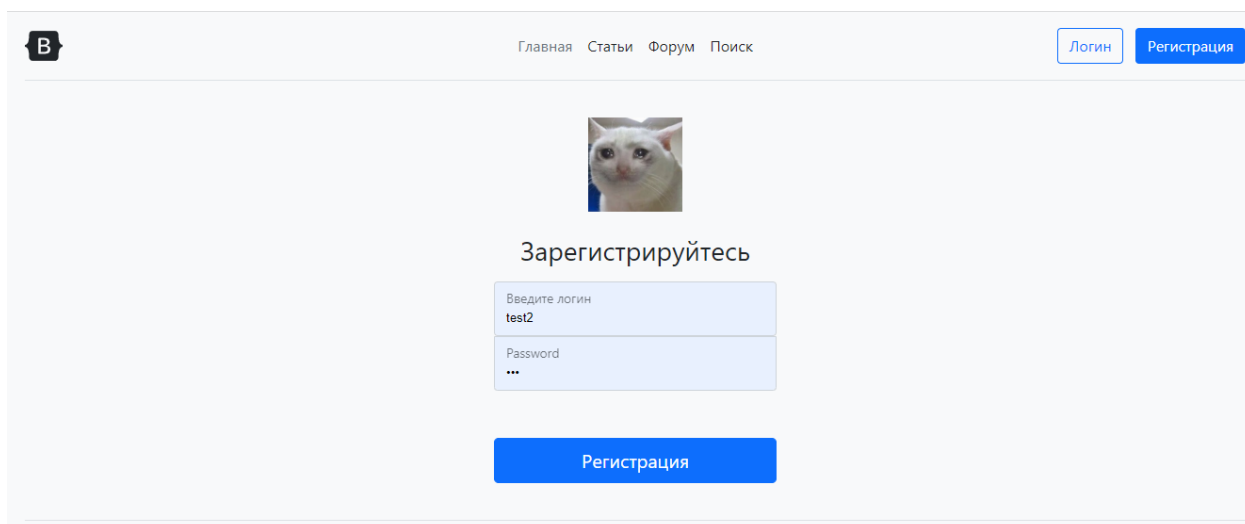


Рисунок 20 – Страница «Регистрация»

Страница предназначена для регистрации нового пользователя в базе данных. После регистрации в случае отсутствия ошибок, пользователь попадет на страницу авторизации. Её внешний вид можно увидеть на рисунке 21.



Рисунок 21-Страница «Авторизации»

После прохождения авторизации пользователь может создавать темы на форуме и писать в них сообщения. Внешний вид страниц со списком тем форума и сообщениями на рисунках 22-23.

Название	Автор	Дата
<a href="#">Библиотеки JavaScript</a>	54321	2022-05-24 20:42:40
<a href="#">sql</a>	oleg	2022-05-25 00:43:37
<a href="#">c#</a>	aboba	2022-05-25 13:51:19
<a href="#">Чат на React</a>	aboba	2022-05-25 13:51:52

Рисунок 22.-Список тем форума

Сообщение	Автор
Посоветуйте библиотеки JavaScript для веб-дизайна	54321
React.jQuery	denchik

Рисунок 23.-Сообщения на теме форума

Если пользователь нажмет на свой логин, он попадет на страницу понравившихся статей. Внешний вид Страницы с кнопкой понравилось и список понравившихся статей на рисунках 24-25.

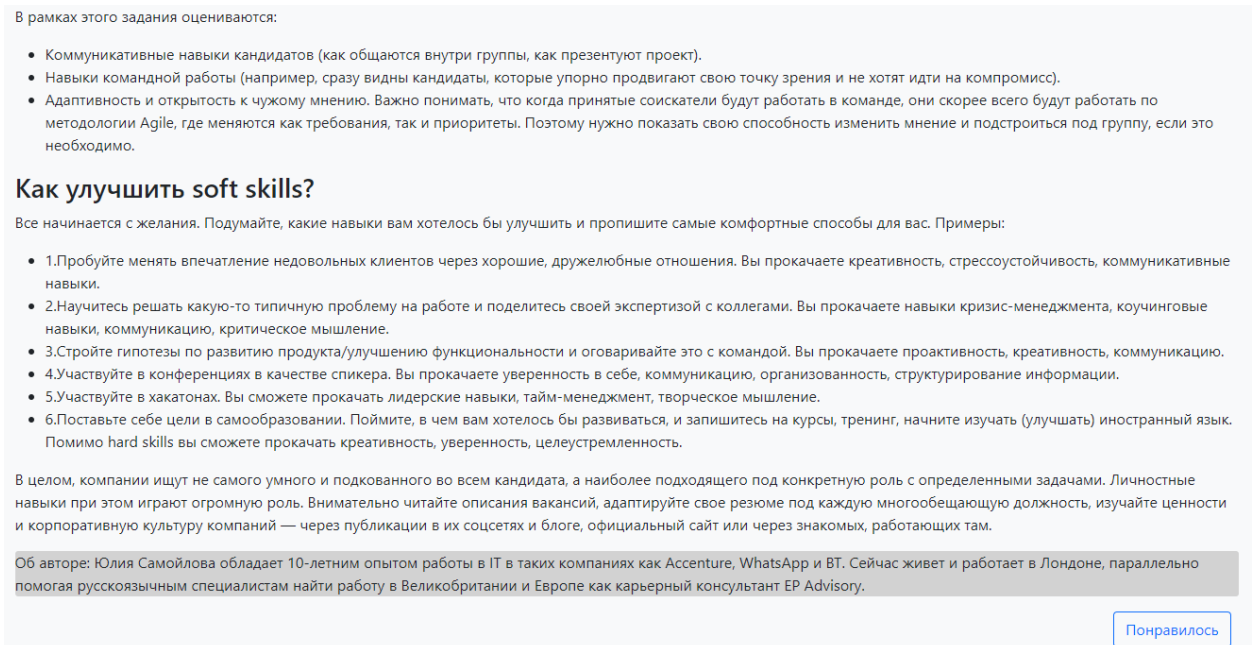


Рисунок 24.-Внешний вид страницы с кнопкой «Понравилось»

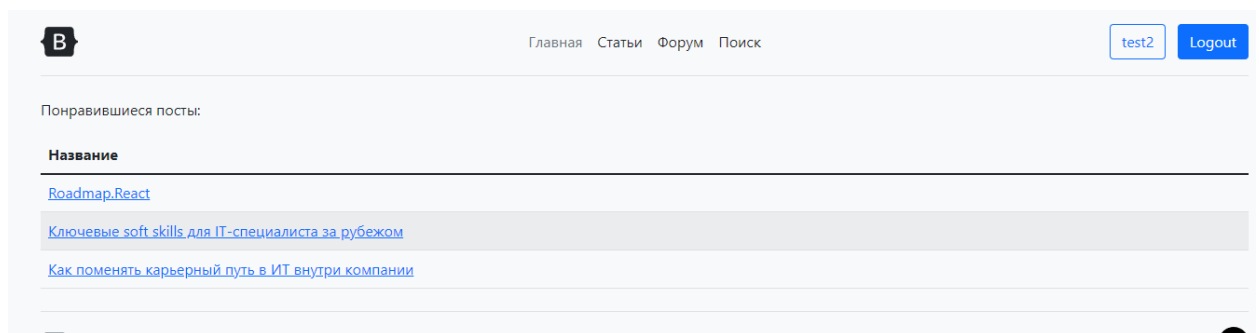


Рисунок 25.-Внешний вид списка понравившихся статей.

Кроме того, авторизированные пользователи могут пройти опрос с целью выявления ключевых интересующих отраслей. Опрос составлен, с целью выявить, одну из 10 наиболее важных для пользователя тематики, которые будут формировать список релевантных статей. Внешний вид страницы с опросом представлен на рисунке 26.

B

[Главная](#)
[Статьи](#)
[Форум](#)
[Поиск](#)

test2

Logout

•

•

•

Ответы

0-1 балл. Не подходит.

2-3 балла. Можно рассмотреть этот язык.

4-5 балла. Язык подходит к вашим предпочтениям.

Python

nix2000nikita@gmail.com

(без совместного доступа)

Сменить аккаунт

Вы уже знакомы с другими ЯП?

1 балл

Да

Нет

Вас волнует синтаксис языка или тонкости распределения памяти?

1 балл

Да

Нет

Рисунок 26.-Опрос на интересующие тематики

После прохождения опроса на главной странице меняется порядок вывода статей, основываясь на полученных данных. Внешний вид обновленной главной страницы для пользователя интересующегося С# представлен на рисунке 27.

Рисунок 27.-Внешний вид главной страницы после опроса

47

#### 4.Тестирование

Тестирование программного обеспечения - это тестирование поведения программы на соответствие ожиданиям, выполняемое с помощью специального набора тестов, выбранных особым образом.

Разработка и тестирование веб-приложения для выбора курсов студентами было разработано и испытано на персональном компьютере со следующими характеристиками:

- тактовая частота процессора – 3000 МГц;
- оперативная память – 12 Гбайт;
- операционная система Windows 10.

Веб-приложение использовалось при помощи браузера GoogleChrome

Объектом испытаний является прототип веб-приложения выбора курсов для студентов.

Целью испытаний является проверка правильности функционирования системы и определение соответствия продукта техническому заданию. Проверка корректного отображения web страниц. Исправность работы всех скриптов. Проверка работоспособности разработанной системы проводилась с использованием разработанных тестовых примеров.

##### 4.1 Функциональное тестирование системы

Тестовые примеры в таблицах 16–28 следует использовать для проверки правильности работы программы.

В качестве метода тестирования использовался метод «черного ящика», который проверяет работоспособность через пользовательский интерфейс и сравнивает ожидаемые результаты с полученными.

Список тестовых примеров был разработан на основе функций .

Тестовые примеры в таблицах 5–14 следует использовать для проверки правильности работы программы.

Таблица 16 – Тестовый пример 1

Входные данные	Ожидаемый результат	Полученный результат
Данные логина и пароля пользователя для регистрации	После введения данных нового пользователя и нажатия кнопки «Зарегистрироваться» будет выполнен переход на страницу авторизации	После введения данных нового пользователя и нажатия кнопки «Зарегистрироваться» был выполнен переход на страницу авторизации

Результат выполнения первого тестового примера продемонстрирован на рисунках 28-29.

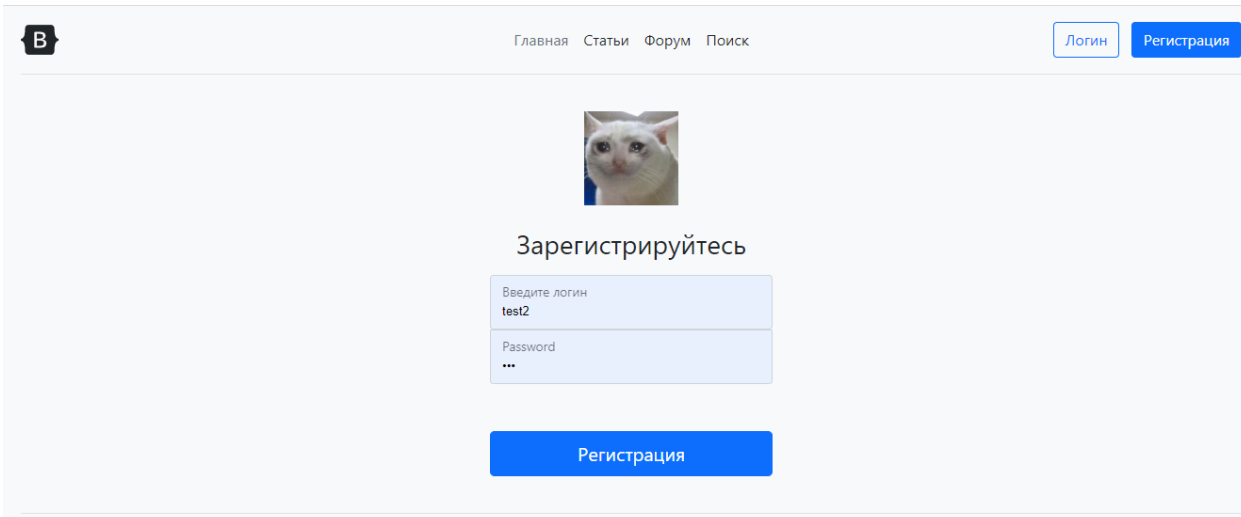


Рисунок 28 – Тестовый пример 1.1

На рисунке 28 пользователь вводит в форму личные данные на форме регистрации, после нажатия кнопки «Регистрация» выполняется переход на страницу авторизации, если пользователя с таким логином не существует в базе данных, в противном случае будет выведено сообщение об ошибке.

Рисунок 29 – Тестовый пример 1.2

При верном введении данных в форму система открывает форму авторизации для нового пользователя.

Таблица 17 – Тестовый пример 2

Входные данные	Ожидаемый результат	Полученный результат
Ввод уже существующего логина при регистрации	Переход на страницу ошибки введенного логина	После введения некорректных данных пользователь перенаправлен на страницу с ошибкой

Результат выполнения второго тестового примера продемонстрирован на рисунках 30-31.

Рисунок 30 — Тестовый пример 2.1

На рисунке 30 пользователь вводит в форму уже существующий в базе данных логин и. При проверке логина из базы данных пользователь будет перенаправлен на страницу с ошибкой о введенном логине .

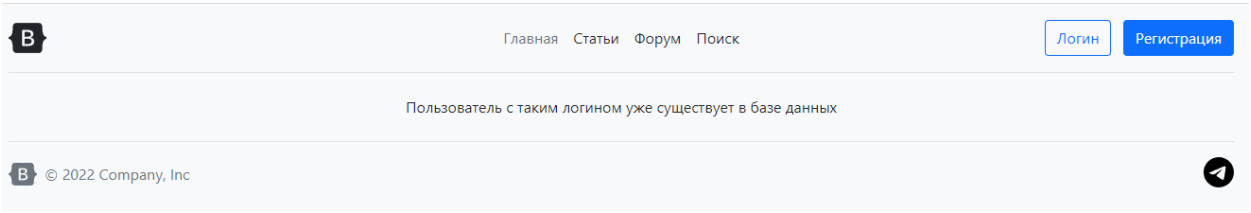


Рисунок 31 — Тестовый пример 2.2

Таблица 18 – Тестовый пример 3

Входные данные	Ожидаемый результат	Полученный результат
На странице авторизации ввести логин и пароль пользователя	После ввода логина и пароля пользователь будет перенаправлен на главную страницу, в правом верхнем углу страницы будет написан его логин.	После введения логина и пароля пользователь перешел на главную страницу. В правом верхнем углу написан его логин

Результат выполнения третьего тестового примера продемонстрирован на рисунках 32-33.

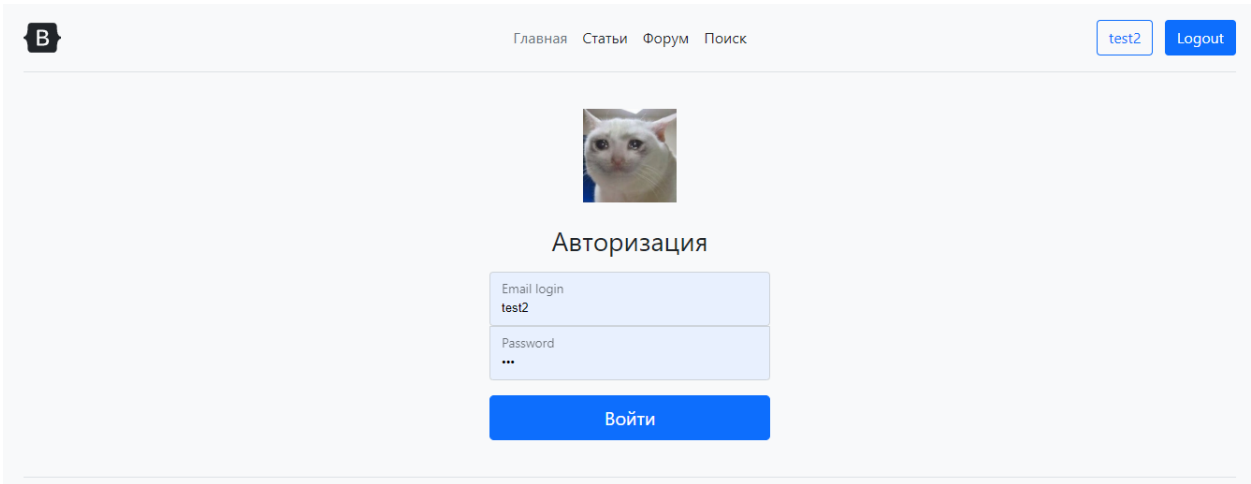


Рисунок 32 — Тестовый пример 3.1

На рисунке 32 пользователь вводит в форму логин и пароль от которые хранятся в БД. После нажатия на кнопку «Войти» пользователь будет перенаправлен на главную страницу.

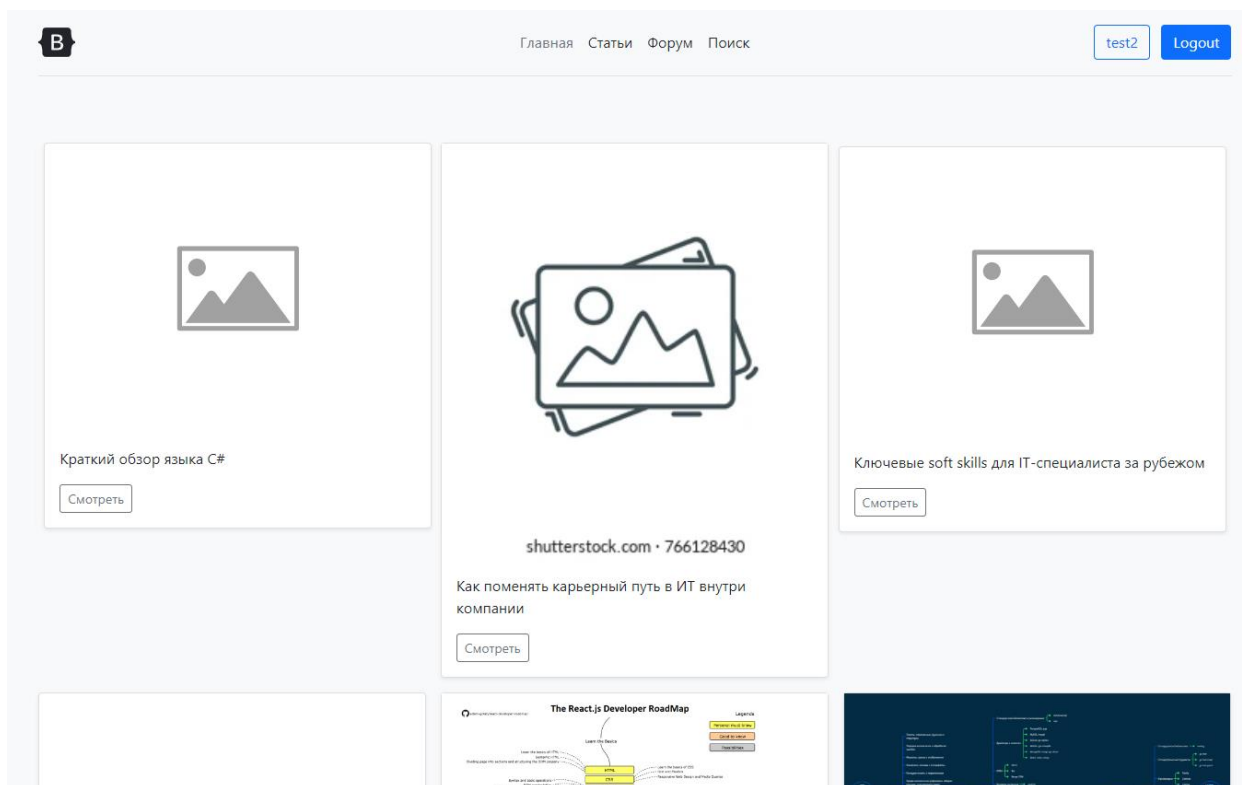


Рисунок 33 — Тестовый пример 3.2

Таблица 19 – Тестовый пример 4

Входные данные	Ожидаемый результат	Полученный результат
Ввести неверные данные пользователя	Переход на страницу с оповещением об ошибке ввода логина или пароля	Пользователь перешел на страницу с оповещением об ошибке

Результат выполнения четвертого тестового примера продемонстрирован на рисунках 34-35.



Рисунок 34 — Тестовый пример 4.1

На рисунке 34 пользователь вводит неверный пароль. После нажатия на кнопку входа его переносит на страницу с сообщением об ошибке.

Рисунок 35 — Тестовый пример 4.2

Таблица 20 – Тестовый пример 5

Входные данные	Ожидаемый результат	Полученный результат
Пользователь добавляет статью в список понравившихся	После нажатия на кнопку «Понравилось»  Список понравившихся статей у пользователя пополняется нужной статьей	Список статей пользователя увеличился

Результат выполнения пятого тестового примера продемонстрирован на рисунках 36-38.

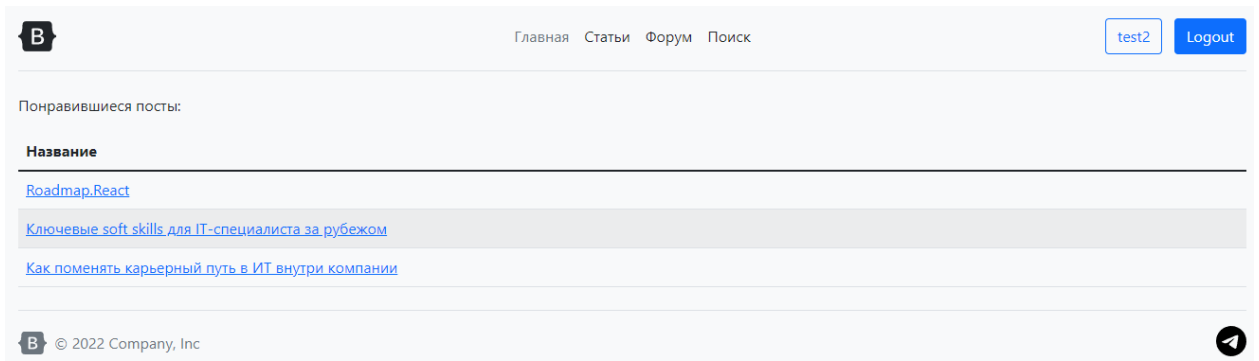


Рисунок 36 — Тестовый пример 5.1

На рисунке 36 можно увидеть список пользователя до нажатия на кнопку «Понравилось» у интересующей статьи. Внешний вид кнопки «Понравилось» на рисунке 37

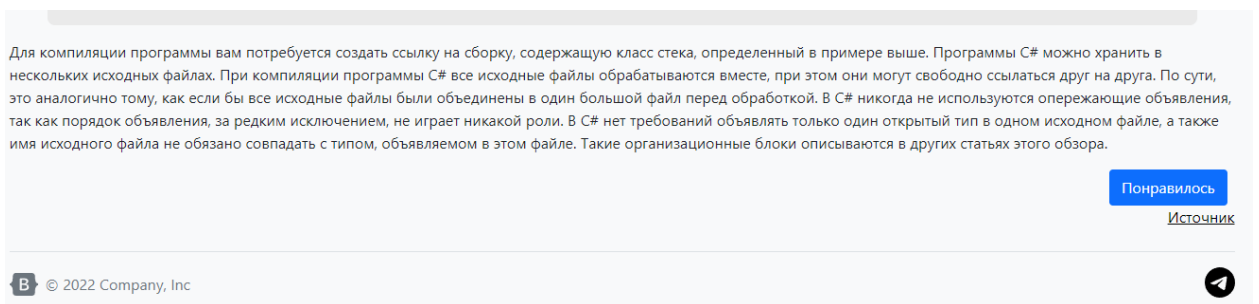


Рисунок 37 — Тестовый пример 5.2

Пользователь на статье «Краткий обзор языка C#» Нажимает на кнопку «Понравилось».

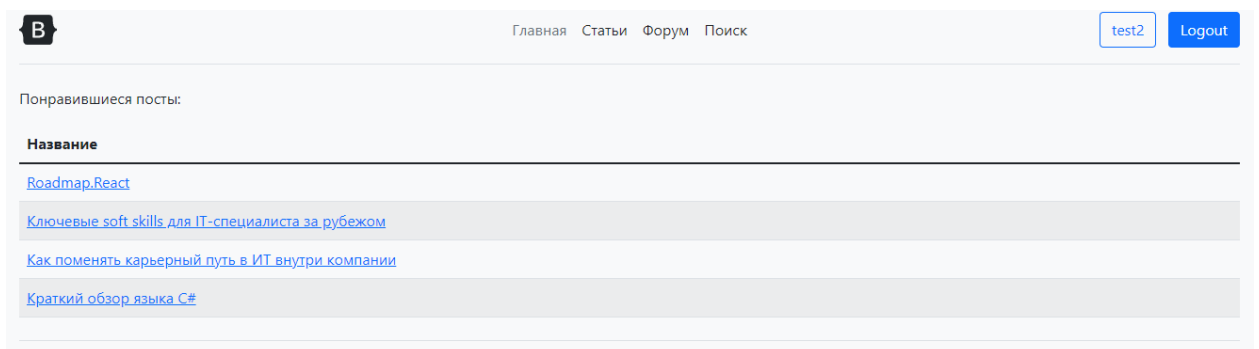


Рисунок 38 — Тестовый пример 5.3

На рисунке 38 список статей пользователя пополнился дополнительной статьей.

Таблица 21 – Тестовый пример 6

Входные данные	Ожидаемый результат	Полученный результат
Пользователь нажимает «Понравилось» на статью которая уже есть в списке	Переход пользователя на страницу с указанием что статья уже есть в списке	Пользователь перемешен на страницу с оповещением о повторном добавлении статьи

Результат выполнения шестого тестового примера продемонстрирован на рисунках 39.

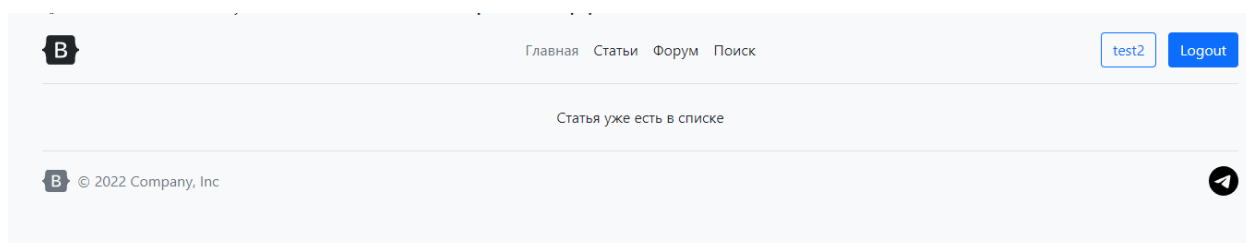


Рисунок 39 — Тестовый пример 6

Пользователь повторно пытался добавить статью «Краткий обзор языка C#» в список понравившихся, в результате пользователь был перемешен на страницу с оповещением об ошибке.

Таблица 22 – Тестовый пример 7

Входные данные	Ожидаемый результат	Полученный результат
Пользователь вводит в поиск часть заголовка интересующей его статьи	На странице появляются результаты поиска с гиперссылками на нужные статьи	На странице появились результаты поиска с гиперссылками на нужные статьи

Результат выполнения седьмого тестового примера продемонстрирован на рисунках 40-41.

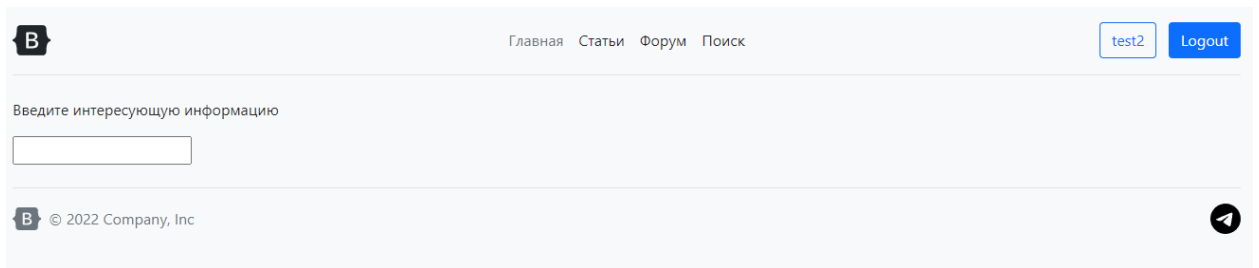


Рисунок 40 — Тестовый пример 7.1

На рисунке 40 пользователь открывает страницу с поиском и начинает вводить ключевые фразы из названия статей

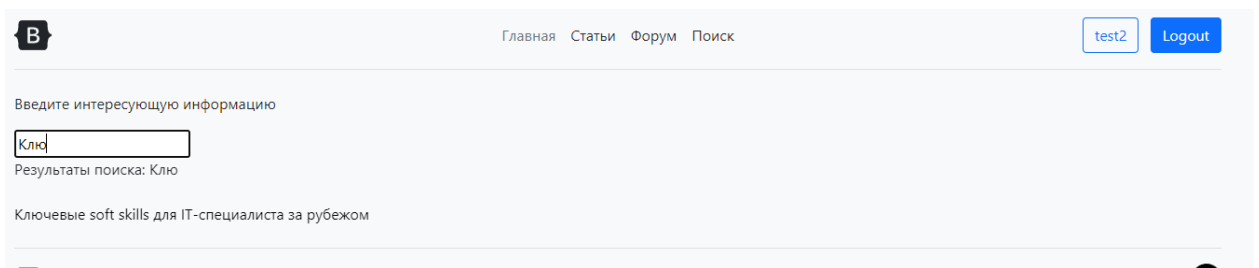


Рисунок 41 — Тестовый пример 7.2

Результаты пользователя выводятся в виде списка ниже поиска с гиперссылками на нужные статьи

Таблица 23 – Тестовый пример 8

Входные данные	Ожидаемый результат	Полученный результат
Пользователь вводит в форму название темы и нажимает на кнопку создать	Новая тема создана на форуме	Новая тема создана на форуме

Результат выполнения восьмого тестового примера продемонстрирован на рисунках 42-43.

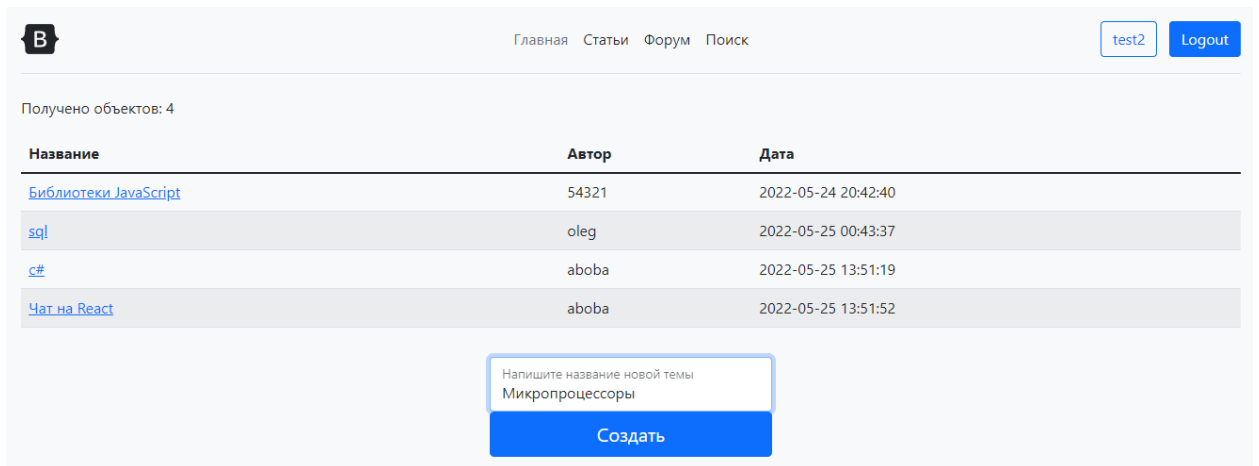


Рисунок 42 — Тестовый пример 8.1

На рисунке 42 пользователь вводит в форму название темы и нажимает на кнопку создать. Страница пользователя обновляется и появляется новая тема на форуме

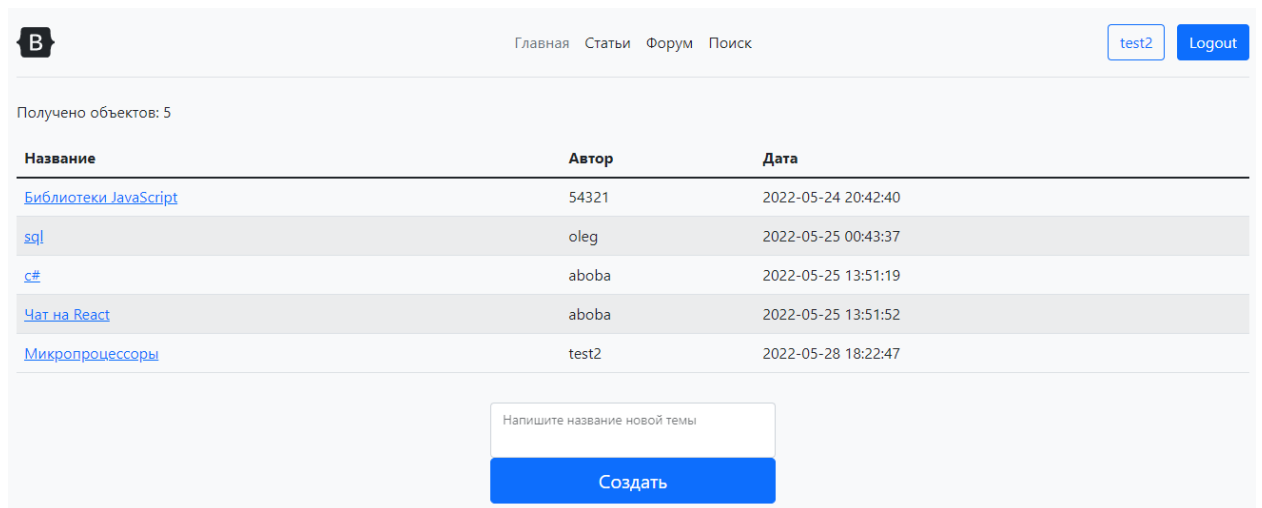


Рисунок 43 — Тестовый пример 8.2

Таблица 24 – Тестовый пример 9

Входные данные	Ожидаемый результат	Полученный результат
Пользователь вводит сообщение в нужную тему форума и нажимает на кнопку отправить	Появляется новое сообщение на форуме	Появилось новое сообщение на форуме

Результат выполнения девятого тестового примера продемонстрирован на рисунках 44-45.

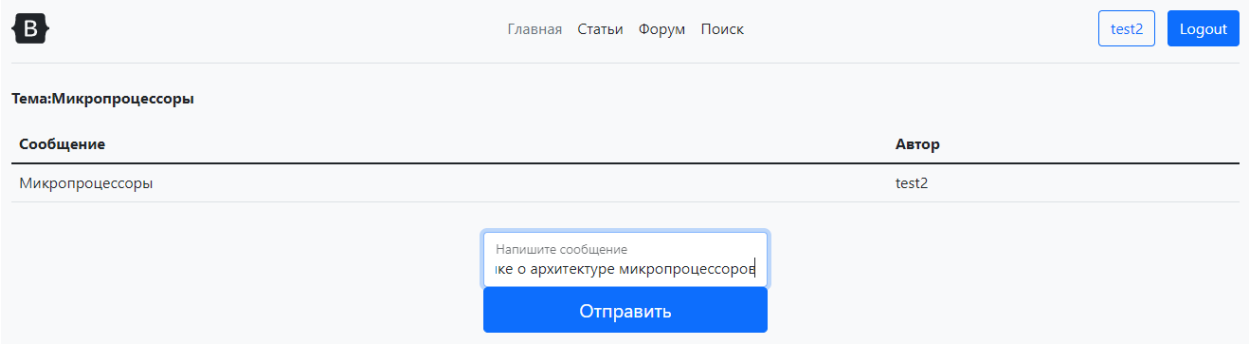


Рисунок 44 — Тестовый пример 9.1

На рисунке 44 пользователь вводит в форму сообщение, которое он хочет отправить и нажимает на кнопку. Сообщение появляется на теме форума.

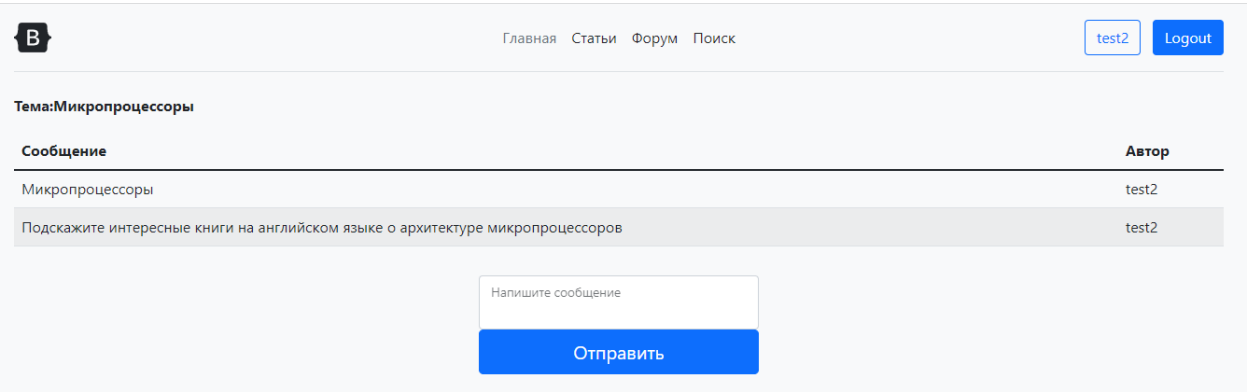


Рисунок 45 — Тестовый пример 9.2

Таблица 25 – Тестовый пример 10

Входные данные	Ожидаемый результат	Полученный результат
Гостевой аккаунт пользователя не может создавать темы на форуме	На странице с отображением тем форуме не будет формы для заполнения	На странице с темами форума отсутствует форма для заполнения

Результат выполнения десятого тестового примера продемонстрирован на рисунке 46.

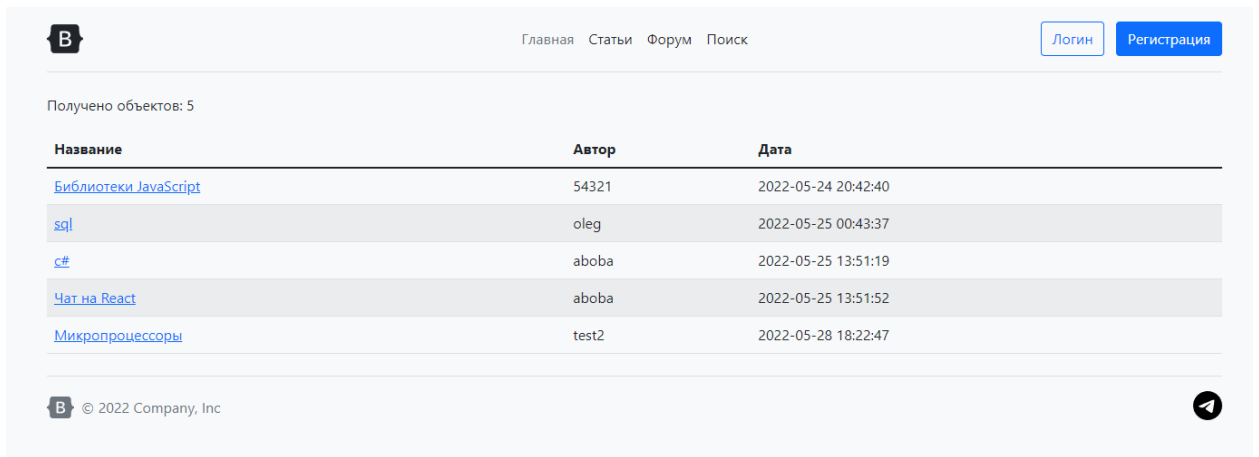


Рисунок 46 — Тестовый пример 10

Не авторизированный пользователь не имеет возможности создавать темы на форуме

Таблица 26– Тестовый пример 11

Входные данные	Ожидаемый результат	Полученный результат
Гостевой аккаунт пользователя смотрит на страницу с сообщения на форуме	На странице будет отсутствовать форма ввода сообщения	На странице отсутствует форма ввода сообщения

Результат выполнения одиннадцатого тестового примера продемонстрирован на рисунке 47.

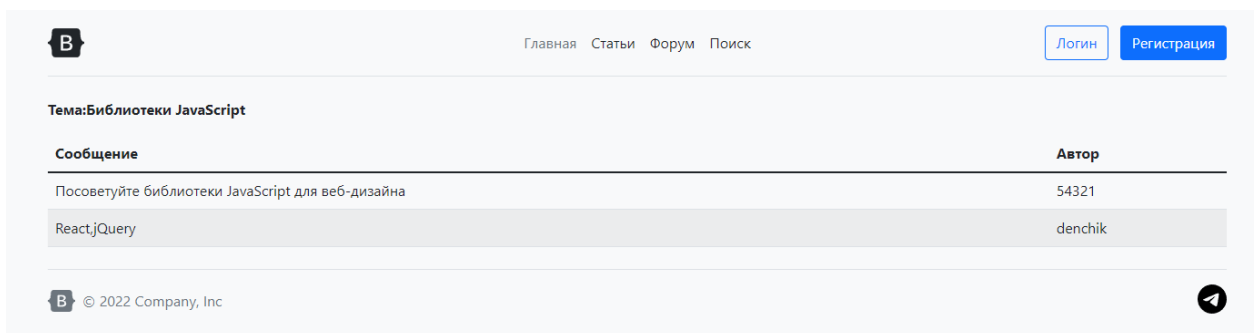


Рисунок 47 — Тестовый пример 11

На странице сообщений по теме форума отсутствует форма для ввода сообщений

Таблица 27-Тестовый пример 12

Входные данные	Ожидаемый результат	Полученный результат
Пользователь типа администратор имеет возможность удалять некорректные темы форума	На странице с темами форума будет кнопка удаления нужной темы	На странице около темы форума присутствует кнопка удаления темы

Результаты тестирования двенадцатого тестового примера продемонстрированы на рисунке 48

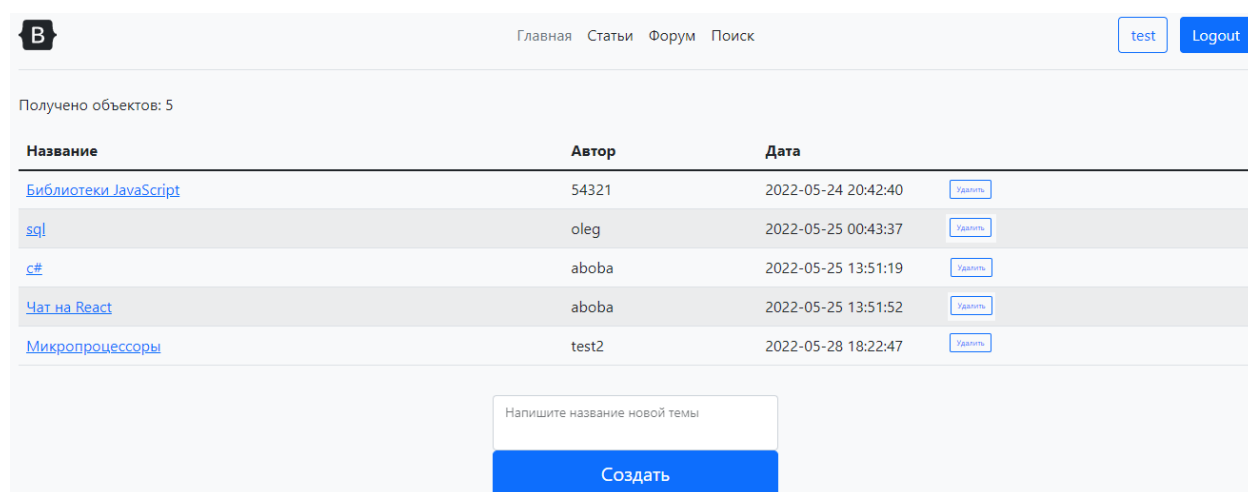


Рисунок 48 – Тестовый пример 12

Таблица 28-Тестовый пример 13

Входные данные	Ожидаемый результат	Полученный результат
Пользователь типа администратор имеет возможность удалять некорректные сообщения на форуме	На странице с сообщениями будет кнопка удаления сообщения	На странице около сообщения присутствует кнопка удаления

Результаты выполнения тринадцатого тестового примера продемонстрированы на рисунке 48



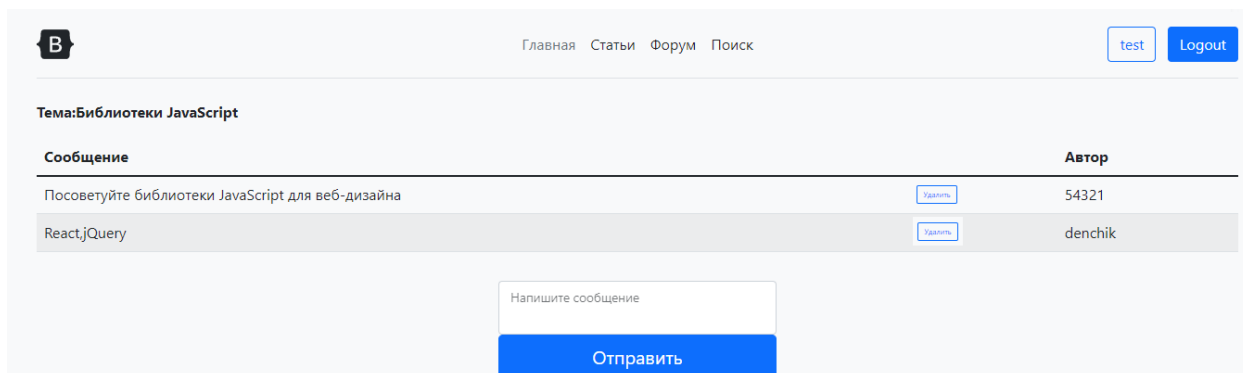


Рисунок 48-Тестовый пример 13

В результате тестирования можно сделать вывод о том, что все функции, которые отражены в техническом задании были реализованы и корректно выполняют свои задачи

## **ЗАКЛЮЧЕНИЕ**

В ходе выполнения выпускной квалификационной работы был разработан прототип веб-приложения выбора курсов для студентов. Данный продукт помогает определиться с областью деятельности которую человек захочет подробнее изучить. В процессе были исследованы аналоги разработанной системы, прочитана соответствующая литература для проектировки и разработки отдельных компонентов веб-приложения, для обеспечения работы всех технических требований.

Был выполнен этап проектирования, на котором были выбраны среды разработки и необходимое ПО, спроектирована и разработана База данных для обеспечения необходимых требований к ВКР.

Разработка программы осуществлена с помощью среды разработки PHPStorm, целесообразность использования которой была предварительно проанализирована и обоснована, работа с базой данной осуществлялась при помощи PHPMyAdmin преимущества которой, были описаны в тексте пояснительной записки. Остальные используемые программные продукты также были тщательно отобраны и выбраны наиболее удобные для использования в рамках ВКР.

Выпускная квалификационная работы была выполнена точно соответствии с поставленным техническим заданием и с соблюдением календарного графика выполнения бакалаврской работы.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Бекирова, Э.А, Халилова, З.Э. Основные этапы создания Web-приложений [научная статья] / Э.А. Бекирова,З.Э. Халилова// ИНФОРМАЦИОННО-КОМПЬЮТЕРНЫЕ ТЕХНОЛОГИИ В ЭКОНОМИКЕ, ОБРАЗОВАНИИ И СОЦИАЛЬНОЙ СФЕРЕ. – 2019. — № 1. – С. 84-91.
2. Habr [Электронный ресурс]. URL: <https://habr.com/ru/all/>(дата обращения:27.05.2022)
3. HeadHunter [Электронный ресурс]. URL: <https://hh.ru/> (дата обращения:27.05.2022)
4. Itfy [Электронный ресурс]. URL: <https://itfy.org/> (дата обращения: 27.05.2022)
5. Vuzopedia [Электронный ресурс]. URL: <https://vuzopedia.ru/> (дата обращения: 27.05.2022)
6. Opensource [Электронный ресурс]. URL: <https://opensource.com/> (дата обращения: 27.05.2022)
7. Клиент-серверная архитектура в картинках [Электронный ресурс]-Режим доступа: <https://habr.com/ru/post/495698/>. (дата обращения: 27.05.2022)
8. Model-View-Controller в .Net — [Электронный ресурс] – Model-View-Presenter и сопутствующие паттерны. — Режим доступа: <http://rsdn.org/article/patterns/modelviewpresenter.xml> (дата обращения: 27.05.2022)
9. Explore the UML sequence diagram [Электронный ресурс] — The diagram's purpose – Режим доступа: <https://developer.ibm.com/articles/the-sequence-diagram> (дата обращения:27.05.2022)
10. Шилдт, Г. Java.Полное руководство. Десятое издание[Книга] — Диалектика-Вильямс: — Отдельное издание,2 018. — 1488 С.

## **ПРИЛОЖЕНИЕ А. ФРАГМЕНТЫ ИСХОДНОГО КОДА**

### Код файла addthread.php

```
//Скрипт предназначен для добавления новой темы форума
<?php
    session_start();//Создание сессии. Сессия-набор временных данных
    которые хранятся на сервере.
    $mysqli= new mysqli("localhost", "root", "", "diploma");//Подключение
    базы данных
    if(isset($_POST['submit']))//Проверка наличия данных в форме
    {
        $ts=$_SESSION['userid'];//Переменные взятые из формы
        $name = $_POST['text'];
        echo $name;
        $stmt          =          $mysqli->prepare('INSERT          INTO
forumtheme(name,topicstarter) VALUES (?,?)');//Подготовленный SQL запрос
на вставку
        $stmt->bind_param('si',$name,$ts);
        $stmt->execute();
        header("forumtheme.php");
    }
?>
```

### Код файла articlelist.php

//Скрипт предназначен для формирования списка всех статей.  
Используется HTML для формирования интерфейса сайта

```
<table class="table table-striped">
    <thead>
    <tr>
        <th scope="col">Название</th>
        <th scope="col">Тип</th>
    </tr>
</thead>
```

```

<tr>//HTML код для формирования интерфейса

    <?php
        $sql = "SELECT article.name as имя,article.url,articletype.name
FROM article,articletype  where article.type_Id=articletype.id";//sql запрос на
выборку статей

        if($result = $mysqli->query($sql)){
            $rowCount = $result->num_rows;

            foreach($result as $row){ $entry=$row["url"];//Проход по всем
строкам полученным из запроса

                echo "<tr>";
                echo  "<td><a      '  href=\""$entry\"">" .  $row["имя"]  .
"</a></td>";

                echo "<td>" . $row["name"] . "</td>";
                echo "</tr>";//обрисовка статей в нужном формате
            }
            echo "</table>";
            $result->free();
        } else{
            echo «Ошибка: « . $mysqli->error;
        }//
    ?>
</tr>

</table>

<?php include("footer.php");?>
</body>
</html>

```

### **Код like.php**

//Необходим для добавления страниц в список понравившихся

```

<?php
$mysqli = new mysqli("localhost", "root", "", "diploma");

```

```

if(isset($_POST['submit'])) {
    $article = $_POST['articleid'];
    $ts = $_SESSION['userid'];
    $sql = "SELECT article.id as Айди FROM article WHERE
url='$article'";

    if ($result = $mysqli->query($sql)) {
        $rowCount = $result->num_rows;
        foreach ($result as $row) {
            echo "<td>" . $row["Айди"] . "</td>";
            $articleid = $row["Айди"];
            break;
        }
        echo $articleid;
    }

    $chek="SELECT user_id,post_id FROM likedpost WHERE
(user_id='$ts') and (post_id='$articleid')";//SQL запрос на поиск уже
существующих строк

    if ($result = $mysqli->query($chek)) {
        $rowCount = $result->num_rows;
        if($rowCount>0)//проверка наличия существующих строк
        {
            include "header.php";
            echo '<div class="text-center">';
            echo $err[] = «Статья уже есть в списке»;
            echo '</div>';
            include "footer.php";
        }
        else{
            $sql = "SELECT article.id as Айди FROM article WHERE
url='$article'";

```

```

        if ($result = $mysqli->query($sql)) {
            $rowCount = $result->num_rows;
            foreach ($result as $row) {
                echo "<td>" . $row["Айди"] . "</td>";
                $articleid = $row["Айди"];
                break;
            }
            echo $articleid;
        }
        $stmt = $mysqli->prepare('INSERT INTO likedpost(post_id,
user_id) VALUES (?,?) ');
        $stmt->bind_param('ii', $articleid, $ts);
        $stmt->execute();
        header("Location: $article");
    }
}
}

```

?>

### **Код файла likestuff.php**

//Формирует кнопку которая позволяет добавлять статьи в список понравившихся

```

<?php
    $mysqli= new mysqli("localhost", "root", "", "diploma");// Right at the top
of your script
    $url = $_SERVER['REQUEST_URI'];
    $url = explode('?', $url);
    $url = $url[0];
    $url=explode('/', $url);
    $url=$url[3];

```



```

?>
<?php if(!isset($_COOKIE['user_login'])){ ?>
<?php }
else {?>
    <main >
        <form method="post" action="like.php">
            <div class="">
                <input type="hidden" name="articleid" value="<?php echo $url;
            </div>
                <input          class="btn          btn-outline-primary          me-2"
name="submit" type="submit" value="Удалить">
            </form>
        </main>
    <?php } ?>

```

### **Код файла login.php**

```

//Проводит авторизацию пользователя
<?php
    $mysqli= new mysqli("localhost", "root", "", "diploma");// Right at the top
of your script
    $url = $_SERVER['REQUEST_URI'];
    $url = explode('?', $url);
    $url = $url[0];
    $url=explode('/', $url);
    $url=$url[3];
    ?>
    <?php  if(!isset($_COOKIE['user_login'])){ ?>//проверка  пройденной
авторизации пользователя
    <?php }
    else {?>
        <main >

```

```

        <form method="post" action="like.php">
            <div class="">
                <input type="hidden" name="articleid" value="<?php echo $url;
?>">

            </div>
            <input class="btn btn-outline-primary me-2"
name="submit" type="submit" value="Удалить">
        </form>
    </main>
<?php } ?>
</html>

```

### Код файла **logout.php**

//Выводит пользователя из акаунта удаляя его куки

```

<?php
session_start();
// Удаляем куки
unset($_SESSION['userid']);
setcookie('token','');
setcookie('user_login','');
header('«Location:index.php»');
?

```

### Код файла **register.php**

//Регистрирует пользователя на сайте

```

<?php
$mysqli= new mysqli("localhost", "root", "", "diploma");
if(isset($_POST['submit']))
{
    $query = mysqli_query($mysqli, "SELECT user_id FROM users
WHERE user_login='".mysqli_real_escape_string($mysqli,
$_POST['login'])."'");//SQL запрос на поиск заданных строк
    if(mysqli_num_rows($query) > 0)//проверка наличия заданных строк

```

```

{ include "header.php";
    echo '<div class="text-center" >';
    echo $err[] = «Пользователь с таким логином уже существует в базе
данных»;
    echo '</div>';
    include "footer.php";
}
else{
$login = $_POST['login'];
$password = $_POST['password'];
$rnd = print('%08x%08x%08x%08x',rand(),rand(),rand(),rand());
$pwdhash = sha1($password. $rnd);
echo "Рнд: $rnd,Логин:$login,Пароль: $password,Новый пароль:
$pwdhash<br/>";
$stmt = $mysqli->prepare('INSERT INTO users(user_login,
user_password, token) VALUES (?,?,:)');
$stmt->bind_param('sss',$login,$pwdhash,$rnd);
$stmt->execute();
    header("Location: loginpage.php");
}
}
?>

```

### **Код файла searchpage.php**

//Использует технологию AJAX для поиска статей по ключевым словам

```

<script>

```

function ajaxRequestupload(url)//функция для получения текста из  
форма и отправки текста в функцию поиска слов.

```

{
    var text = document.getElementById("search_text").value;
    var httpRequest =new XMLHttpRequest();

```

```

    httpRequest.onreadystatechange = function()
    {
        alertResponse(this);
    }
    httpRequest.open('Get', "upload.php?a=" + text, true);
    httpRequest.send(null);
}
function alertResponse(httpRequest)
{
    if(httpRequest.readyState ==4)
    {
        if(httpRequest.status == 200)
        {
            var responseDiv = document.getElementById('ajaxDiv');
            responseDiv.innerHTML = httpRequest.responseText;
        }
        else
        {
            alert('проблема с получением ответа от сервера');
        }
    }
}
function ajaxrequestshow(url)
{
    var text = document.getElementById("search_text").value
    var xmlhttp = new XMLHttpRequest();
    xmlhttp.onreadystatechange = function()
    {
        alertResponse(this);
    };
}

```

```

        xmlhttp.open("GET", "show.php?a=" + text, true);
        xmlhttp.send(null);
    }
</script>

```

### **Код файла sendmessage.php**

//Скрип отправляет сообщение на форум.

```

<?php
session_start();

$mysqli= new mysqli("localhost", "root", "", "diploma");
if(isset($_POST['submit']))
{
    $ts=$_SESSION['userid'];
    $ThreadId=$_POST['threadid'];
    $name = $_POST['text'];
    $stmt = $mysqli->prepare('INSERT INTO message(topic_id,user_id,text)
VALUES (?, ?, ?)');
    $stmt->bind_param('iis', $ThreadId, $ts, $name);
    $stmt->execute();
    header("Location: thread.php?id=".$ThreadId);
}
?>

```

### **Код файла show.php**

```

while($data =$show_table -> fetch_object())
{
    $mystring = "$data->name";
    $findme  = $q;
    $pos = mb_stripos($mystring, $findme); //Сравнение полученной из
формы строки с существующей в БД
    if ($pos === false)
    {

```

```

    }
    else
    {
        echo "<a class='shinesearch' ' href=\"\$data->url\">" . \$data->name .
"</a><br>";
    }
}
?>

```

### Код файла thread.php

//Скрипт формирует список сообщений отправленных в нужную тему  
форума

```

<?php

$mysqli= new mysqli("localhost", "root", "", "diploma");// Right at the top
of your script

?>

<?php include("header.php");?>

<?php
if (isset($_GET['id'])){
    $ThreadID=$_GET['id'];

    $sql = "select message.text as текст,forumtheme.name as
тема,users.user_login as имя from message,forumtheme,users
where (message.user_id=users.user_id) and
(message.topic_id=forumtheme.id) and (topic_id=$ThreadID)";

    if($result = $mysqli->query($sql)){
        $rowCount = $result->num_rows;
        foreach($result as $row){
            echo "<tr>";
            echo "<td>" . $row["текст"] . "</td>";
            echo "<td>" . $row["имя"] . "</td>";

```

```

        echo "</tr>";
    }
    foreach($result as $row) {$name2=$row["тема"];
        echo "<p><strong>Тема:$name2</strong></p>";break;
    }
    echo "</table>";
    $result->free();
} else{
    echo "Ошибка: " . $mysqli->error;
} }// количество полученных строк
?>
</tr>
</table>
</div>
</div>
<?php if(!isset($_COOKIE['user_login'])){ ?>
<?php }
else {?>
<main class="form-signin">
    <form method="post" action="sendmessage.php">
        <div class="form-floating">
            <input type="text" name="text" class="form-control" >
            <label for="floatingInput">Напишите сообщение </label>
            <input type="hidden" value="<?php echo $ThreadID; ?>"
name="threadid">
        </div>
        <input name="submit" class="w-100 btn btn-lg btn-primary"
type="submit" value="Отправить">
    </form>
</main>

```

```

</div>

<?php } ?>

<?php include("footer.php");?>

</body>

</html>

```

### **Код файла userpage.php**

//Формирует страницу со списком понравившихся пользователю статей

```

<?php include("header.php");?>

    <?php
        $userid=$_SESSION['userid'];
        $sql = "select article.name as Название,article.url as url from
likedpost,users,article
        where likedpost.post_id=article.id and likedpost.user_id=users.user_id and
users.user_id='$userid'";

        if($result = $mysqli->query($sql)){
            $rowCount = $result->num_rows;
            echo "<p>Понравившиеся посты:</p>";
            foreach($result as $row){
                $entry=$row["url"];
                echo "<tr>";
                echo " <td><a    ' href=\"\$entry\">\" . $row[\"Название\"] .
\"</a></td>";

                echo "</tr>";
            }
            echo "</table>";
            $result->free();
        } else{
            echo "Ошибка: " . $mysqli->error;

```