МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
**«Южно-Уральский государственный университет
(национальный исследовательский университет)»**
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

РАБОТА ПРОВЕРЕНА

Рецензент

Ст. преподаватель кафедры
ММОМ ФГБОУ ВО «ЮУрГГПУ»,
к.ф.-м.н.
_____А.М. Шарафутдинова
“___”_____ 2019 г.

ДОПУСТИТЬ К ЗАЩИТЕ

Заведующий кафедрой,
д.ф.-м.н., профессор
_____ Л.Б. Соколинский
“___”_____ 2019 г.

РАЗРАБОТКА ОБРАЗОВАТЕЛЬНОГО ПОРТАЛА ДЛЯ ОБУЧЕНИЯ ПРОГРАММИРОВАНИЮ

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 02.03.02.2019.115-115.ВКР

Научный руководитель,
к.ф.-м.н., доцент кафедры СП
_____С.А. Иванов

Автор работы,
студент группы КЭ-401
_____А.К. Кузин

Ученый секретарь
(нормоконтролер)
_____ О.Н. Иванова
“___”_____ 2019 г.

Челябинск–2019

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

УТВЕРЖДАЮ
Зав. кафедрой СП
_____ Л.Б. Соколинский
09.02.2019

ЗАДАНИЕ
на выполнение выпускной квалификационной работы бакалавра
студенту группы КЭ-401
Кузину Андрею Константиновичу,
обучающемуся по направлению
02.03.02 «Фундаментальная информатика и информационные технологии»

1. Тема работы (утверждена приказом ректора от «25» апреля 2019 № 899)

Разработка образовательного портала для обучения программированию.

2. Срок сдачи студентом законченной работы: 05.06.2019.

3. Исходные данные к работе

3.1. Р.Никсон. Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5 — «O'Reilly», 2016. — 688 с.
3.2. Д.Дакетт. HTML и CSS. Разработка и дизайн веб-сайтов— ИД «Эксмо», 2017. —480 с.
3.3. Д.Скляр. Изучаем PHP 7. Руководство по созданию интерактивных веб-сайтов— «O'Reilly», 2017. —464с.

4. Перечень подлежащих разработке вопросов

4.1. Произвести обзор аналогов
4.2. Разработать веб-интерфейс портала
4.3. Спроектировать и реализовать базу данных, необходимую для работы портала
4.4. Разработать серверную часть портала

4.5. Провести комплексное тестирование программного продукта

5. Дата выдачи задания: 08.02.2019.

Научный руководитель

к.ф.-м.н., доцент кафедры СП С.А. Иванов

Задание принял к исполнению А.К. Кузин

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

АКТУАЛЬНОСТЬ РАБОТЫ

В настоящее время профессия программиста является высоко востребованной [14], поэтому всё больше людей стремятся освоить эту профессию. Востребованность и высокая заработная плата рождает большой спрос на рынке труда. Кроме того, в последние годы наблюдается рост популярности онлайн-обучения [15], поскольку такой тип обучения позволяет человеку изучать интересный или необходимый ему материал в удобном рабочем ритме.

ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Целью данной работы является разработка образовательного портала для обучения программированию.

Для достижения поставленной цели были поставлены следующие задачи.

1. Провести анализ предметной области и аналогичных проектов.
2. Разработать веб-интерфейс портала.
3. Спроектировать и реализовать базу данных.
4. Реализовать серверную часть веб-портала.
5. Провести тестирование.

СТРУКТУРА И ОБЪЕМ РАБОТЫ

Работа состоит из введения, 4 разделов, заключения и библиографии. Объем работы составляет 42 страницы, объем библиографии – 18 источников, объем приложения – 4 страницы.

В первом разделе проводится обзор предметной области, обзор и сравнительный анализ аналогов разрабатываемого портала. Также в этом разделе выбираются технологии и инструменты для разработки.

Во втором разделе определяются требования к системе, компоненты системы и проводится проектирование основных компонентов. Также в этом разделе представлены диаграмма вариантов использования портала и диаграмма деятельности.

В третьем разделе описывается процесс разработки веб-портала и алгоритма индивидуальной траектории.

В четвертом разделе приведены результаты тестирования системы.

В заключении сделаны выводы о проделанной работе и результатах и рассмотрены дальнейшие планы по развитию и улучшению системы.

В приложении представлены скриншоты веб-страниц портала.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1. Обзор аналогов

С целью определения оптимального содержания и необходимого функционала для веб-портала с обучающими курсами, был проведен обзор интернет-ресурсов, предоставляющих возможность проходить обучающие курсы по программированию. В ходе обзора были рассмотрены следующие интернет-ресурсы:

Ulearn [6]

Этот сайт предоставляет доступ к нескольким обучающим курсам, разработанным СКБ «Контур» совместно с Уральским Федеральным университетом. Курсы разбиты на блоки и включают в себя как практические задания, так и теоретические видео-лекции. К преимуществам данной системы можно отнести удобную навигацию по курсу и отображение прогресса прохождения, которую можно увидеть на рис. 1.

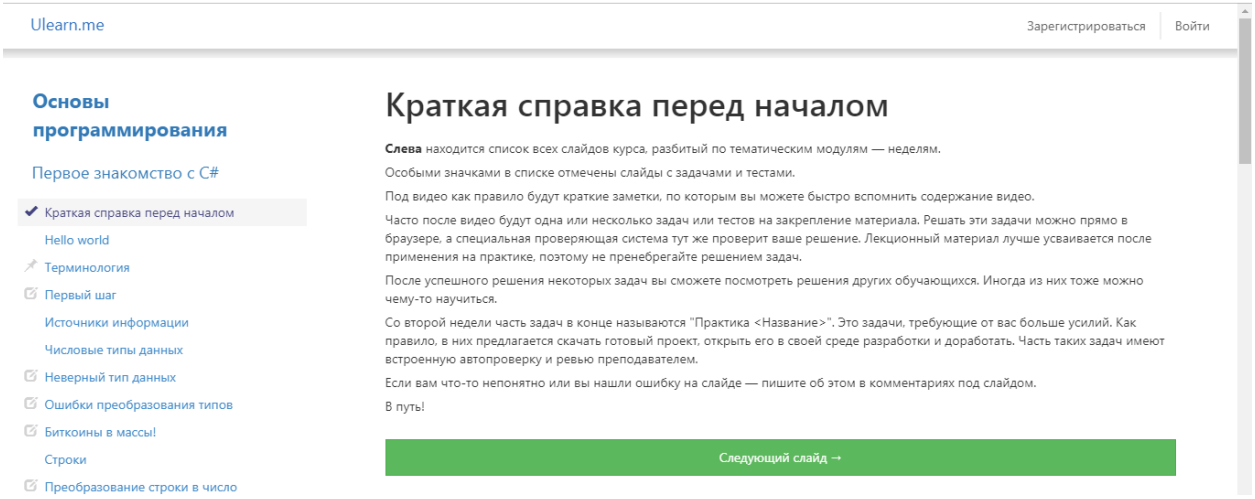

Рис. 1. Пример страницы прохождения курса на Ulearn

Практические задания представлены в двух вариантах: тесты и задания, требующие написания программного кода. Системой выполняется проверка представленного ответа и выносится вердикт о правильности выполнения задания. Кроме того, проверяющая система оценивает не только правильность кода, но и его оформление (соответствие стандартам оформ-

ления программного кода, осмысленные названия переменных и функций и т.п.).

Informatics [10]

Данный сайт представляет собой базу задач и обучающих курсов. Пользователи ресурса делятся на два типа: простые пользователи, которые могут пройти любой открытый курс или решить задачу по определенной теме, и учителя, которые могут создавать новые курсы и соревнования. Для регистрации в роли учителя требуется подтверждение от учебного заведения. Главная страница данного ресурса представлена на рис. 2.

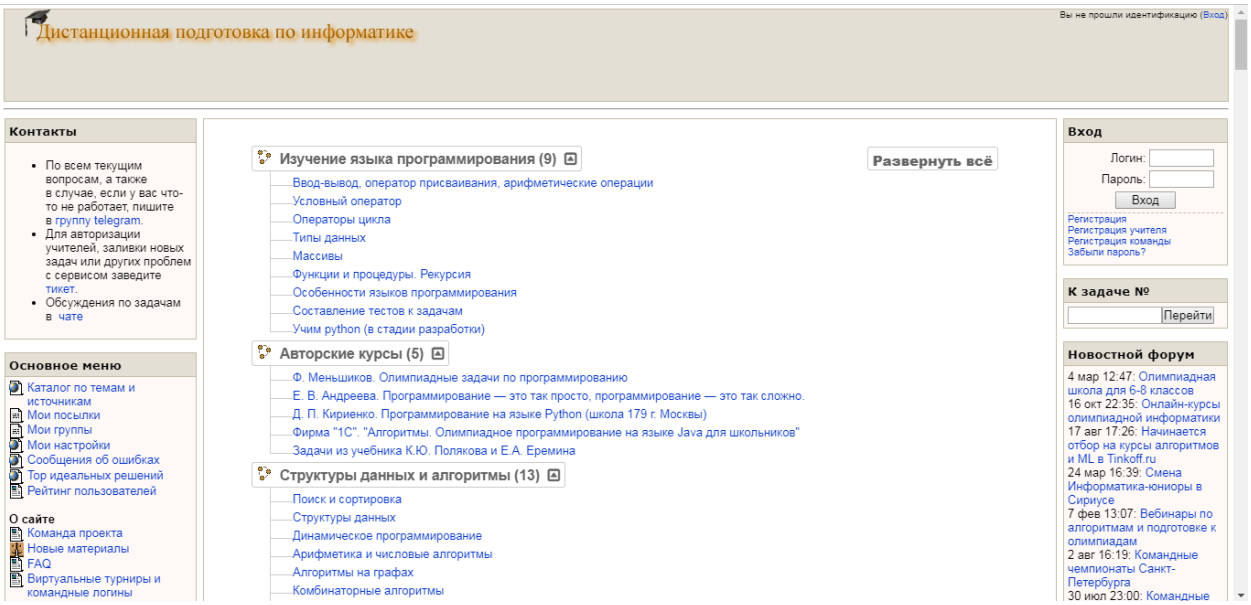

Рис. 2. Главная страница сайта Informatics

Codeforces [3]

Основным предназначением данного ресурса является предоставление возможности прохождения соревнований по программированию (т.н. контестов). Любой пользователь может создать свой контест, используя задачи из публичной базы, и предоставить публичный доступ к нему. Codeforces не предоставляет права создавать или проходить обучающие курсы, однако пользователь вправе выбрать любую задачу по программированию из публичной базы и представить свое. Страница решения задачи представлена на рис. 3.

ЗАДАЧИ ОТОСЛАТЬ СТАТУС ПОЛОЖЕНИЕ ЗАПУСК

A. Куча из камней

ограничение по времени на тест: 1 секунда
ограничение по памяти на тест: 256 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

У Васи была куча, состоящая из нескольких камней. Он n раз либо убирал один камень из кучи, либо добавлял один камень в кучу. Куча до применения любой операции удаления всегда была непустой.

Вам даны n операций, которые сделал Вася. Найдите минимальное количество камней, которое может оказаться в конце в куче, после применения всех операций.

Входные данные
В первой строке находится одно целое число n — количество операций, сделанных Васей ($1 \le n \le 100$).

В следующей строке находится строка s, состоящая из n символов, равных "−" (без кавычек) или "+" (без кавычек). Если Вася убирал камень на i операции, то s_i равно "−" (без кавычек), а если добавлял, то s_i равно "+" (без кавычек).

Выходные данные
Выведите одно целое число, равное минимальному количеству камней, которое могло остаться в куче, после применения Васей n операций.

Примеры

входные данные	Скопировать
3 ---	
выходные данные	Скопировать
0	

Рис. 3. Пример страницы решения задачи на Codeforces

Система фиксирует траекторию прохождения каждой задачи (количество попыток отправить решение, количество пройденных тестов и т.п.). Помимо этого, пользователь может обсудить непонятные ему темы и задать необходимые вопросы на форуме.

Stepic [5]

Данный ресурс предоставляет пользователям доступ к обширной базе онлайн-курсов на различные тематики: программирование, математика, иностранные языки и т.д. Пользователи могут пройти любой бесплатный курс, который может состоять из текстовых теоретических сводок, видеолекций, тестов и практических заданий. На рис. 4 представлен скриншот главной страницы ресурса.

При желании любой пользователь может создать новый курс, наполнив его содержимым по своему усмотрению. Портал предоставляет право выбрать степень доступности курса для других пользователей: платный или бесплатный, публичный или приватный. Кроме того, данный ресурс

дает создателям курсов право указать обязательное последовательное прохождение курса.

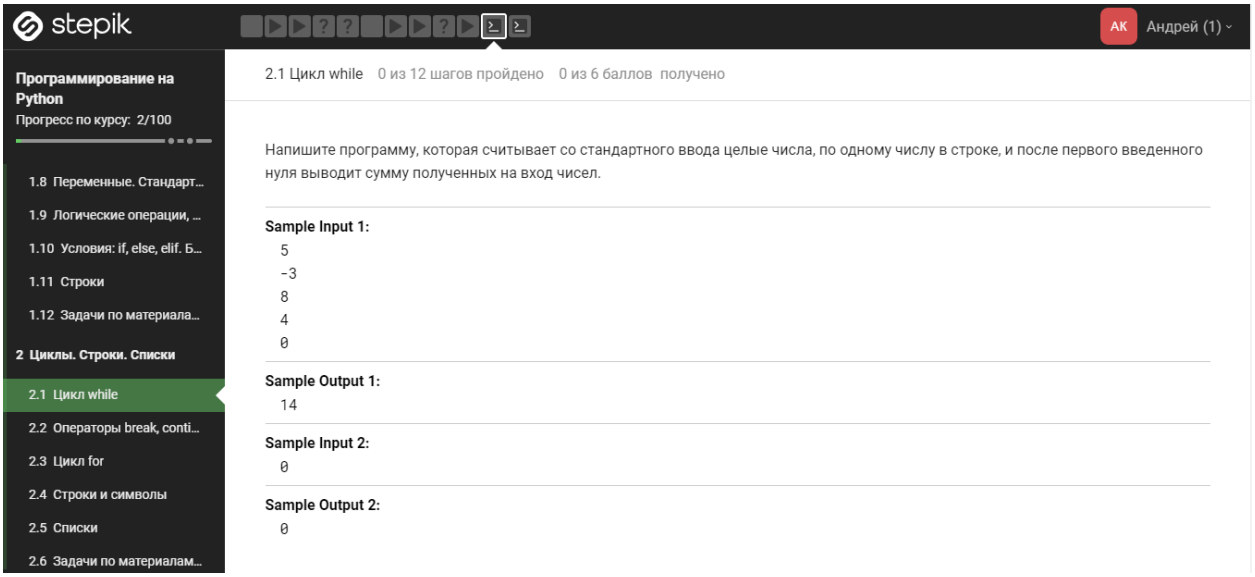

Рис. 4. Пример страницы решения задания на Stepic

1.2. Сравнительный анализ аналогов

В ходе обзора аналогов было выделено несколько ключевых особенностей портала по обучению программированию. В таблице 1 представлен сравнительный анализ по этим критериям.

Табл. 1. Сравнительный анализ аналогов

Аналог / Критерий	*Ulearn*	*Codeforces*	*Informatics*	*Stepic*
Возможность проходить определенные курсы	Есть	Отсутствует	Есть	Есть
Возможность решать отдельные задачи	Отсутствует	Есть	Есть	Отсутствует

Продолжение табл. 1

Аналог / Критерий	*Ulearn*	*Codeforces*	*Informatics*	*Stepic*
Наличие теоретических вставок в курсе	Есть	Отсутствует	Есть	Есть
Последовательное прохождение курса	Отсутствует	Отсутствует	Отсутствует	Есть(опционально)
Право создавать новые курсы	Отсутствует	Отсутствует	Есть	Есть
Удобная навигация по курсу	Есть	Отсутствует	Отсутствует	Есть
User-friendly интерфейс	Есть	Есть	Отсутствует	Есть
Личный кабинет с личным прогрессом пользователя	Отсутствует	Есть	Есть	Есть

Окончание табл. 1

Аналог / Критерий	*Ulearn*	*Codeforces*	*Informatics*	*Stepic*
В каком виде представляется решение задачи	Текст	Файл	Файл	Текст
Проверка усвоения материала	Отсутствует	Отсутствует	Отсутствует	Отсутствует

По результатам сравнительного анализа можно сделать вывод, что типичный портал с обучающими курсами не предусматривает возможности проверить, как хорошо пользователь усвоил материал курса.

1.3. Инструменты и технологии

Для создания веб-портала были выбраны следующие инструменты и технологии:

– Язык разметки HTML для создания структуры веб-страниц.

– Формальный язык CSS для описания стиля веб-страниц. Для достижения кроссбраузерности и уменьшения времени разработки стилей был выбран CSS-фреймворк Bootstrap [11].

– Язык программирования JavaScript для добавления интерактивных элементов на веб-страницы, взаимодействия с пользователем и для обмена данными с сервером.

– Фреймворк AngularJS для языка JavaScript [1]. Выбор этого фреймфорка обусловлен его популярностью, наличием подробной доку-

ментации и облегчением написания программного кода. Кроме того, AngularJS использует схему Model-View-Controller и двустороннее связывание данных на веб-странице, что позволяет упростить работу с динамическими страницами.

– Язык программирования PHP для написания серверной части [17]. Данный язык был выбран из-за наличия опыта работы с ним, наличия обширной коллекции библиотек и развитой поддержки различных баз данных.

– СУБД MySQL для создания базы данных [12]. Выбор этой СУБД обусловлен ее простотой в использовании, масштабируемостью и наличием опыта работы с ней.

Выводы по первому разделу

В данном разделе был проведен обзор и сравнительный анализ аналогов разрабатываемой системы, а также обусловлен выбор инструментов для её создания.

2. ПРОЕКТИРОВАНИЕ

2.1. Требования к системе

После обзора аналогов и их сравнительного анализа были сформулированы требования к разрабатываемой системе.

Функциональные требования

Функциональные требования описывают функциональность программного обеспечения, ожидаемое поведение системы [9]. Были определены следующие функциональные требования:

– система должна предоставлять возможность неавторизованному пользователю посмотреть список доступных к прохождению курсов;

– система должна предоставлять возможность зарегистрироваться в роли ученика, который может проходить любой доступный ему курс, или преподавателя, который может создавать новые курсы;

– система должна предоставлять возможность добавить последовательную навигацию по курсу при создании;

– система должна давать право заполнять курсы как практическими заданиями, так и теоретическими сведениями;

– система должна фиксировать и предоставлять пользователю его траекторию прохождения того или иного курса;

– система должна содержать модуль для проверки усвоения материала пользователем.

Нефункциональные требования

Нефункциональные требования представляют собой описание таких характеристик, как дизайн пользовательского интерфейса продукта, легкость и простота использования, ограничения внешнего вида и структуры продукта. Были определены следующие нефункциональные требования:

– система должна иметь различный интерфейс для разных ролей;

– серверная часть должна быть реализована по принципу RPC;

– веб-интерфейс должен поддерживать адаптивную верстку.

2.2. Диаграмма вариантов использования

Для проектирования приложения был использован язык графического описания UML [7]. Для отображения функционала системы была составлена диаграмма вариантов использования, представленная на рис. 5.

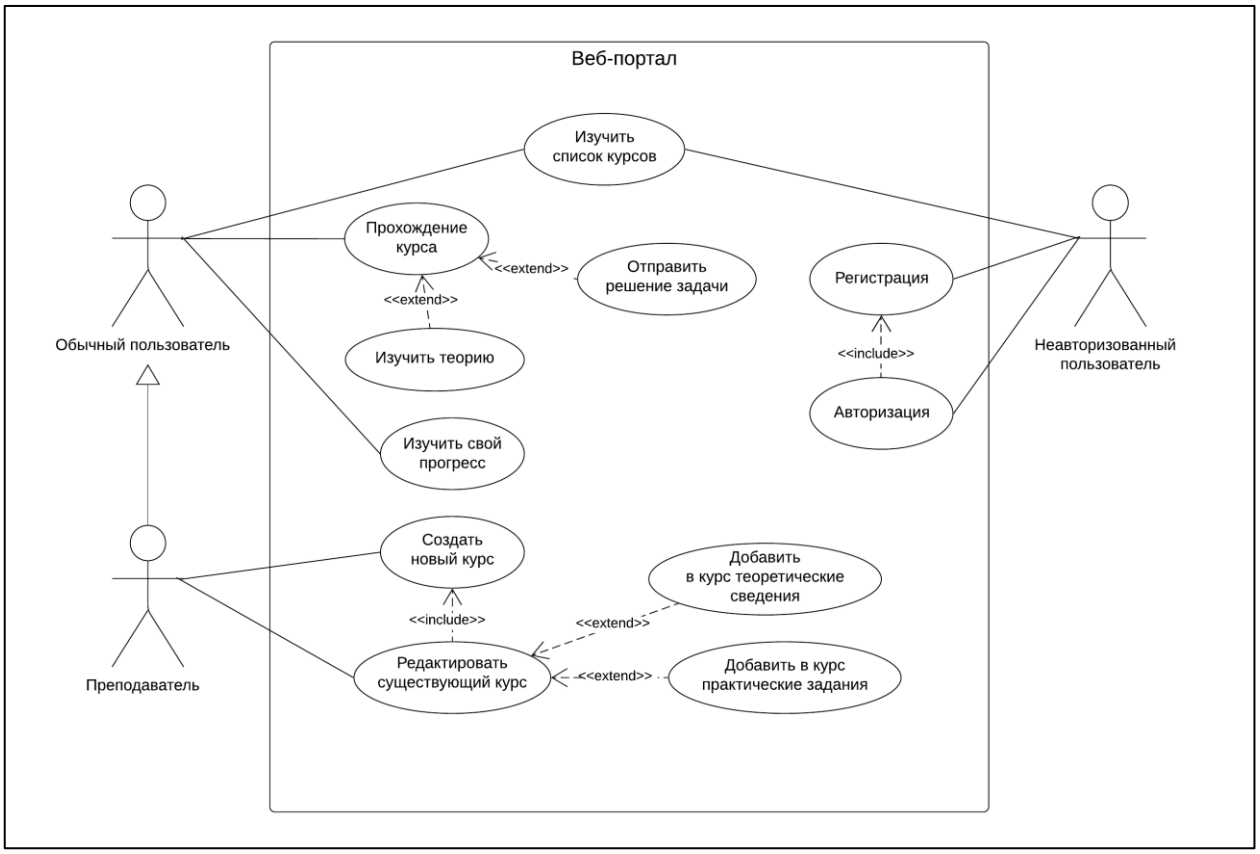

Рис. 5. Диаграмма вариантов использования системы

В данной системы представлены три актера:

– *Неавторизованный пользователь* – любой пользователь, не прошедший процедуру авторизации на сайте. Обладает ограниченным функционалом.

– *Обычный пользователь* – авторизованный пользователь, не зарегистрировавшийся как преподаватель. Обладает базовым функционалом.

– *Преподаватель* – авторизованный пользователь, зарегистрировавшийся в роли преподавателя. Обладает расширенным функционалом *Обычного пользователя*.

Неавторизованный пользователь может:

1) зарегистрироваться на портале. В процессе регистрации можно выбрать опцию «Зарегистрироваться в роли преподавателя» для получения расширенного функционала в системе;

2) авторизоваться, используя данные, указанные при регистрации;

3) изучить список общедоступных курсов.

Обычный пользователь может:

1) пройти любой существующий курс. В процессе прохождения пользователь может как изучать теорию, так и решать практические задания. Навигация по курсу может быть как свободной, так и последовательной, т.е. новые задания разблокируются лишь после решения предыдущих. Решение практических заданий включает в себя отправку файла с исходным кодом, решающим задание, после чего решение проверяется на сервере, а пользователю выводится сообщение о правильности его варианта решения задачи;

2) изучить свой прогресс прохождения того или иного курса в личном кабинете.

Преподаватель может:

1) создать новый курс, после чего сразу же приступить к его редактированию;

2) отредактировать существующий курс. Преподаватель может редактировать лишь созданные им курсы. В процессе редактирования можно добавлять или удалять теоретические и практические блоки, менять порядок блоков, указывать порядок прохождения курса (последовательный или произвольный), добавлять блоки для проверки усвоения материала курса пользователем.

2.3. Компоненты системы

Для описания структуры разрабатываемой системы была разработана диаграмма компонентов, представленная на рис. 6. Разрабатываемая система состоит из двух основных компонентов: веб-портал и база данных.

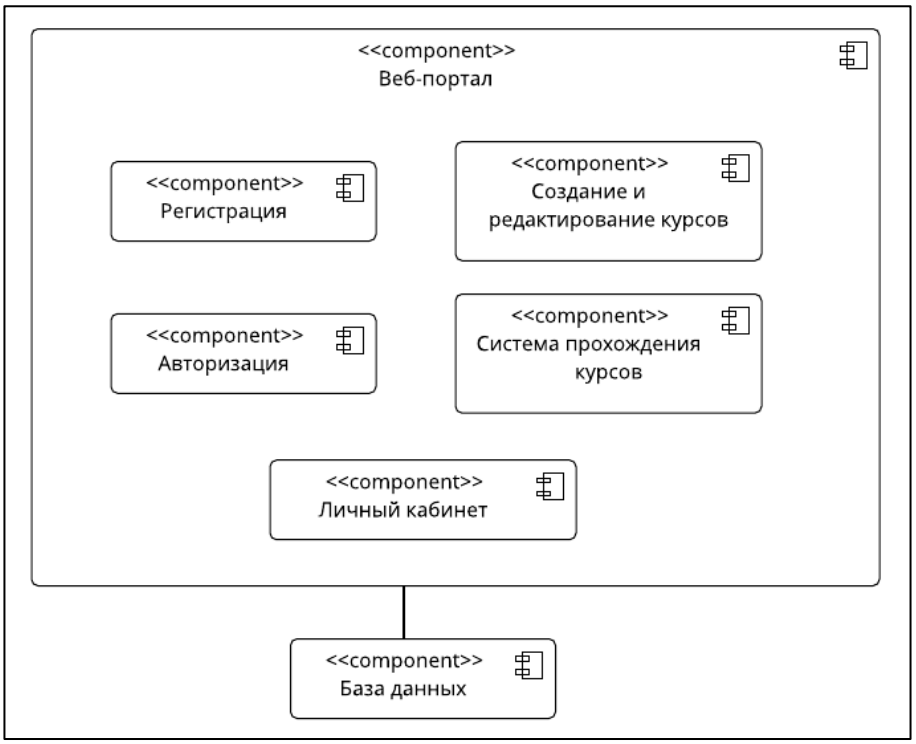

Рис. 6. Диаграмма компонентов системы

Компонент «Веб-портал» включает в себя следующие компоненты.

1. Компонент регистрации. Добавляет новых пользователей в систему. В рамках именно этого компонента определяется доступный пользователю функционал.

2. Компонент авторизации. Авторизует пользователя, давая ему доступ к прохождению или созданию курсов.

3. Компонент личного кабинета. Позволяет отслеживать свой прогресс прохождения курсов и менять контактные данные.

4. Компонент создания и прохождения курсов. В рамках этого компонента происходит создание и наполнение новых курсов или внесение изменений в уже существующие.

5. Компонент системы прохождения курсов. Включает в себя процессы отображения всех доступных курсов и прогресса по курсам. Именно в рамках этого компонента осуществляется проверка правильности решений тех или иных задач в рамках курса и осуществляется проверка усвоения материала пользователем.

2.4. Диаграмма деятельности

На рис. 7 представлена диаграмма деятельности, отображающая процесс прохождения любого курса на портале.

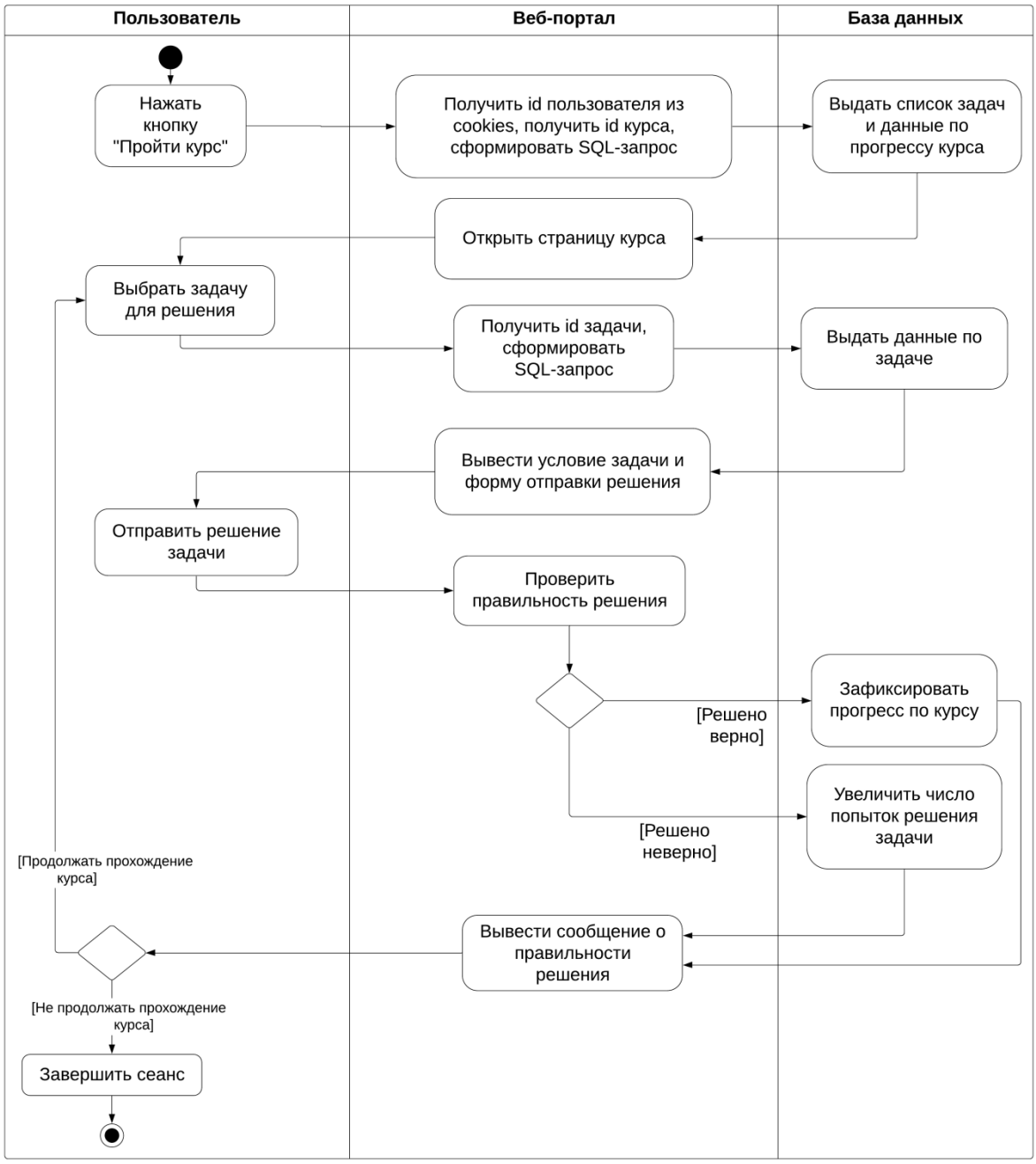

Рис. 7. Диаграмма деятельности процесса прохождения курса

На этой диаграмме элемент «Веб-портал» представляет собой клиентскую и серверную сторону портала, отображенные на рисунке как единый элемент.

2.5. Проектирование базы данных

Для корректной работы веб-портала была спроектирована база данных, схема которой представлена на рис. 8.

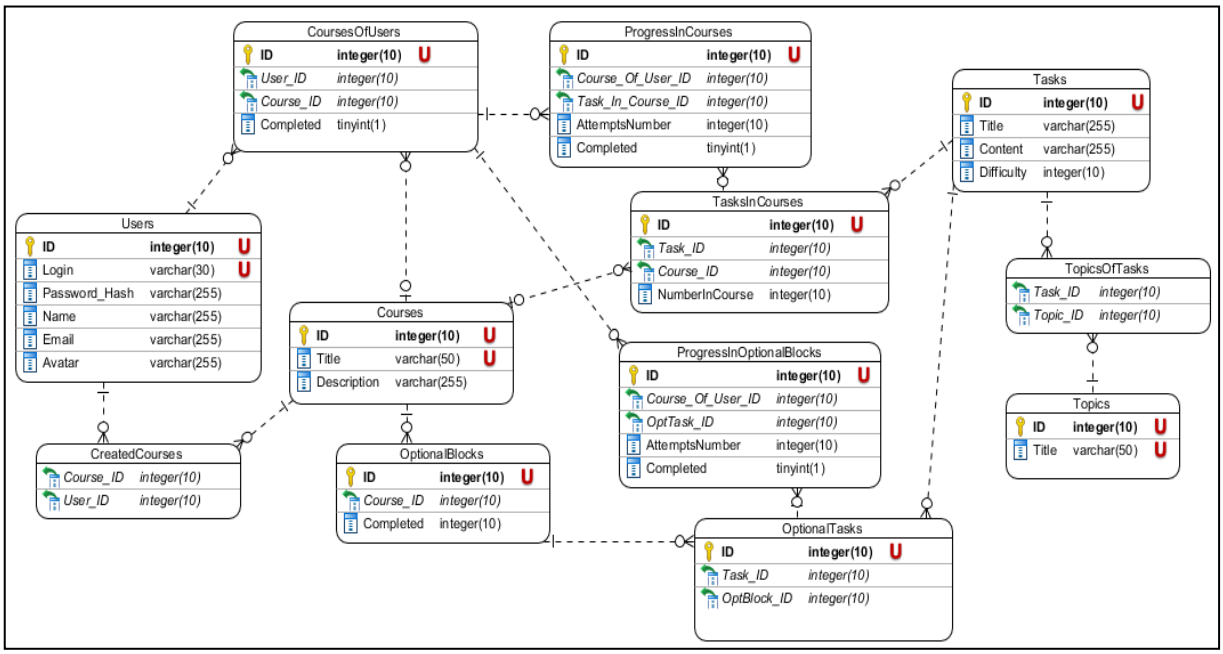

Рис. 8. Схема базы данных

Ниже представлено описание таблиц базы данных.

Users – данные о зарегистрированных пользователях портала.

1. ID – уникальный идентификатор пользователя. Является первичным ключом.
2. Login – уникальный логин для входа в систему.
3. Password_Hash – хэш-функция пароля для аутентификации пользователя.
4. Avatar – ссылка на изображение-аватар пользователя.
5. Name – отображаемое на странице имя.
6. Email – адрес электронной почты для связи.

Courses – список всех курсов, представленных в системе.

1. ID – уникальный идентификатор курса. Является первичным ключом.
2. Title – уникальное название курса.

3. Description – краткое описание курса (назначение, длительность и т.п.).

CoursesOfUsers – содержит информацию о том, какие курсы проходит пользователь.

1. ID – уникальный идентификатор связи Пользователь-Курс. Является первичным ключом.
2. User_ID – внешний ключ к таблице Users: идентификатор пользователя.
3. Course_ID – внешний ключ к таблице Courses: идентификатор курса.
4. Completed – булево значение, означающее, завершено ли прохождение курса или нет.

CreatedCourses – содержит информацию о том, кто создал тот или иной курс.

1. User_ID – внешний ключ к таблице Users: идентификатор пользователя.
2. Course_ID – внешний ключ к таблице Courses: идентификатор курса.

Tasks – список заданий.

1. ID – уникальный идентификатор задания. Является первичным ключом.
2. Title – название задания.
3. Content – содержание задания.
4. Difficulty – сложность задания, определяет его тип (0-теоретическая вставка, 1 и более – задача).

Topics – список тем для задач.

1. ID – уникальный идентификатор темы. Является первичным ключом.
2. Title – название темы.

TopicsOfTasks – содержит информацию о том, какие темы есть у той или иной задачи.

1. Task_ID – внешний ключ к таблице Tasks: идентификатор задания.
2. Topic_ID – внешний ключ к таблице Topics: идентификатор темы.

TasksInCourses – содержит информацию о том, какие задачи находятся в том или ином курсе.

1. ID – уникальный идентификатор связи Задание-Курс. Является первичным ключом.
2. Task_ID – внешний ключ к таблице Tasks: идентификатор задания.
3. Course_ID – внешний ключ к таблице Courses: идентификатор курса.
4. NumberInCourse – номер задания в общей структуре курса.

ProgresssInCourses – содержит информацию о прохождении курса пользователем.

1. ID – уникальный идентификатор записи. Является первичным ключом.
2. Course_Of_User_ID – внешний ключ к таблице CoursesOfUsers: идентифицирует курс и проходящего его пользователя.
3. Task_In_Course_ID – внешний ключ к таблице TasksInCourses: идентифицирует курс и задание в нем.
4. AttemptsNumber – количество попыток решить конкретную задачу.
5. Completed – булево значение, равное 1, если пользователь правильно решил конкретную задачу.

OptionalBlocks – содержит информацию об опциональных блока курса.

1. ID – уникальный идентификатор блока. Является первичным ключом.

2. Course_ID – внешний ключ к таблице Courses: идентификатор курса.

3. NumberInCourse – номер текущего блока в общей структуре курса.

OptionalTasks – содержит информацию о том, какие задачи находятся в том или ином опциональном блоке.

1. ID – уникальный идентификатор связи Задание-Блок. Является первичным ключом.

2. Task_ID – внешний ключ к таблице Tasks: идентификатор задания.

3. OptBlock_ID – внешний ключ к таблице OptionalBlocks: идентификатор блока.

ProgresssInOptionalBlocks – содержит информацию о прохождении курса пользователем.

1. ID – уникальный идентификатор записи. Является первичным ключом.

2. Course_Of_User_ID – внешний ключ к таблице CoursesOfUsers: идентифицирует курс и проходящего его пользователя.

3. OptTask_ID – внешний ключ к таблице OptionalTasks: идентифицирует опциональный блок и задание в нем.

4. AttemptsNumber – количество попыток решить конкретную задачу.

5. Completed – булево значение, равное 1, если пользователь правильно решил конкретную задачу.

2.6. Проектирование серверной части

В качестве основного подхода к архитектуре сервера был выбран метод вызова удаленных процедур (англ. Remote Procedure Call, RPC) [2], суть которого можно описать следующим алгоритмом.

1. Клиент отправляет на сервер через http-запрос JSON-файл, содержащий два поля: название функции и входные параметры для этой функции.

2. Сервер вызывает требуемую функцию, передает в нее полученные параметры и возвращает результат клиенту.

Особенностями данного подхода является тот факт, что все необходимые функции реализованы исключительно на сервере, потому клиент вызывает функции из другого адресного пространства, не зная, как именно они реализованы.

2.7. Проектирование веб-интерфейса

Первым этапом создания веб-сайта является разработка макетов веб-страниц. Составление макета позволяет разработчику определить наиболее важные функциональные элементы, создать удобный и эргономичный интерфейс.

Учитывая вышеописаные принципы, были составлены макеты страниц веб-портала. На рис. 9, 10 представлены макеты основных страниц портала: главной страницы и страницы прохождения курса.

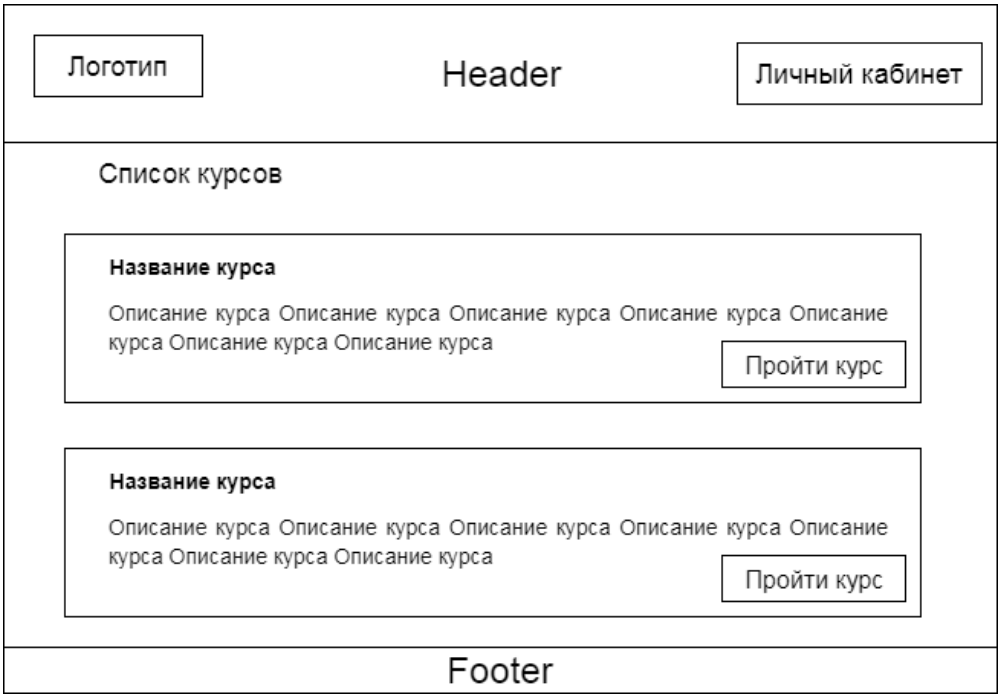

Рис. 9. Макет главной страницы

Рис. 10. Макет страницы прохождения курса.

В результате обзора аналогов и создания своего стиля для веб-сайта были выбраны следующие элементы фирменного стиля портала.

Основные цвета. Были выбраны спокойные пастельные оттенки синего и голубого. Данная цветовая палитра способствуют благоприятному и спокойному времяпрепровождению пользователя на сайте [16]. Оказание успокаивающего воздействия является важной частью для комфортного нахождения пользователя на портале, поскольку процесс обучения требует не активного мозгового штурма, а тщательного и методичного восприятия новой информации.

Основные шрифты. В качестве наборного шрифта был выбран шрифт Montserrat. Этот шрифт подходит для набора больших объемов текста, поскольку хорошо читается в кегле до 16 пунктов. В качестве заголовочного шифра был выбран шрифт Oswald, концентрирующий внимание на набранном тексте, что важно для заголовков. Кроме того, важным фактором при выборе шрифтов были поддержка кириллицы и их популярность, отраженная в рейтинге Google Fonts Analytics [4].

2.8. Проектирование алгоритма индивидуальной траектории

В контексте образовательного портала важную роль играет индивидуальная образовательная траектория, т.е. персональный путь реализации личностного потенциала каждого ученика [18]. Для поддержки индивидуальной траектории предлагается внедрить в разрабатываемый веб-портал модуль, реализующий алгоритм индивидуальной траектории.

Идея заключается в том, чтобы добавить в структуру каждого курса новый тип блока – проверяющий (опциональный) блок, состоящий из практических задач. Особенность данного типа блока в том, что его содержимое не задается при создании курса, а формируется динамически при переходе на него пользователя. Создатель курса может поместить такой блок в любой части курса по своему разумению.

Проверяющий блок становится доступным к прохождению при решении всех задач до него. При открытии данного блока система запускает алгоритм индивидуальной траектории, который анализирует траекторию прохождения курса конкретным пользователем и выбирает для него несколько задач из общей базы для проверки усвоения материала. Траектория прохождения курса состоит из следующих параметров:

- количество попыток решить каждую задачу;
- время, затраченное на решение каждой задачи;
- сложность каждой задачи;
- список тем каждой задачи.

Алгоритм должен выдать список задач только с темами, которые присутствовали в курсе до этого блока, и со сложностью, не превышающей максимальную сложность задач до этого блока. Этот список зависит от прогресса конкретного пользователя, поэтому он является индивидуальным. Наличие такого алгоритма будет являться отличительной особенностью разрабатываемого веб-портала.

На этапе проектирования алгоритм выглядит следующим образом.

1. Входные данные: массив задач **A.**

2. Разделить задачи по степеням сложности: легкие, сложные и средней сложности.

3. Для каждого типа сложности вычислить среднее число попыток решить задачу.

4. Найти задачи, число попыток решить у которых больше среднего по типу сложности значения. Внести темы этих задач в массив **T**.

5. Вычислить максимальную сложность **D** по всем задачам.

6. Выбрать из общей базы несколько задач, удовлетворяющих следующим условиям: сложность задачи **d** <= **D**, темы задачи **t**$\subseteq$**T** и этой задачи нет в **A**.

7. Выходные данные: массив задач **B**, полученные в ходе шага 6.

Данный алгоритм реализует содержательное направление индивидуальной образовательной траекторией [8].

Выводы по второму разделу

В данном разделе были сформулированы требования к разрабатываемому веб-порталу, спроектирован и описан с помощью диаграмм UML необходимый функционал, спроектирован веб-интерфейс и предложен алгоритм индивидуальной траектории для проверки усвоения материала курса пользователем.

3. РЕАЛИЗАЦИЯ

Клиентская и серверная часть веб-портала были реализованы с использованием интегрированной среды разработки JetBrains PhpStorm 2018.1.6. База данных была реализована и запущена на локальном сервере с помощью OpenServer.

3.1. Реализация веб-интерфейса

В соответствии с требованиями к системе и на основе макетов веб-страниц был реализован веб-интерфейс. На рис. 11 представлен скриншот главной страницы портала, представляющей из себя список всех доступных курсов.

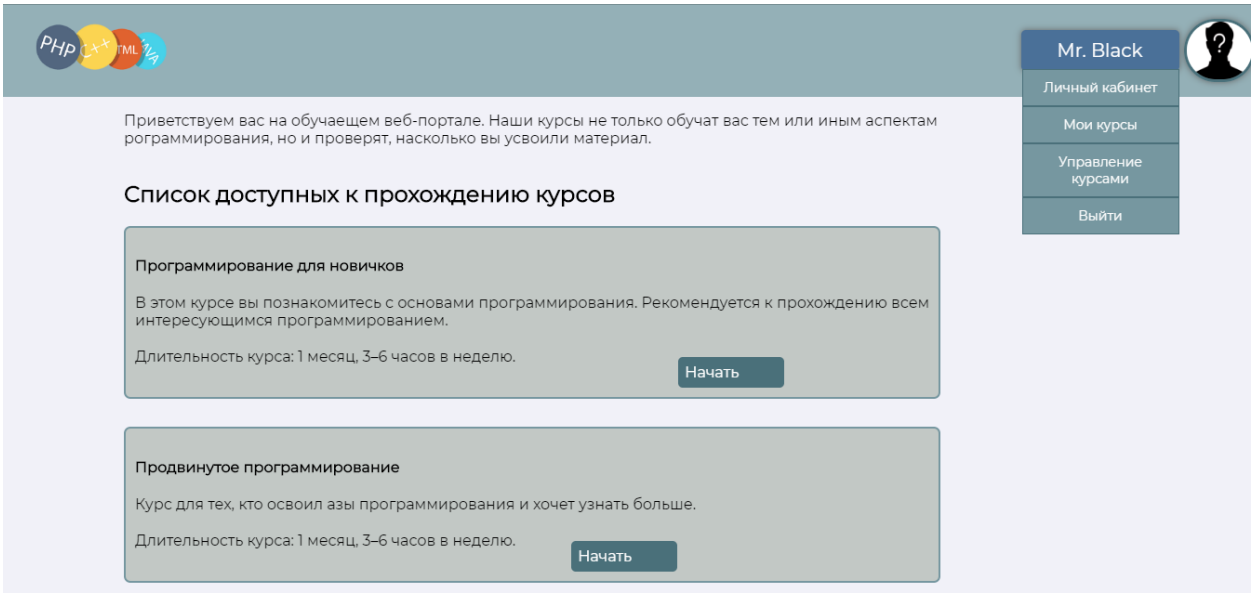

Рис. 11. Скриншот главной страницы портала

Скриншоты, демонстрирующие прочие страницы портала и его функционал, представлены в приложении.

Для предоставления единообразия и удобства веб-интерфейса был разработан шаблон шапки веб-страницы с использованием концепции шаблонов AngularJS. Листинг этого шаблона представлен на рис. 12. При использовании данного шаблона в шапку веб-страницы выводятся контактные данные пользователя: имя и аватар. Для неавторизованного пользователя предусмотрен другой шаблон, выводящий в шапку страницы приглашение авторизоваться.

```
<header>
 <a href="Main Page.html"><img class="logo" src="images/logo.png"></a>
 <div class="personal_area_block">
  <ul class="menu">
   <li><a href=# class="men">{{name}}</a>
    <ul class="submenu">
     <li><a href="Personal Area.html" class="sub">Личный кабинет</a></li>
     <li><a href = # class="sub">Мои курсы</a></li>
     <li      ng-show=show_btn><a      href      =      'ManagingCourses.html'
class="sub">Управление  курсами</a></li>
     <li><a href="php_scripts/log_out.php" class="sub">Выйти</a></li>
    </ul>
   </li>
  </ul>
  <img src="{{image}}" class="avatar">
 </div>
</header>

<div ng-transclude></div>
```

Рис. 12. Листинг шаблона шапки веб-страницы

3.2. Реализация серверной части

Было принято решение обращаться к серверу, используя POST-запросы, отправляя единообразные JSON – файлы для реализации принципа RPC. Пример такого файла приведен на рис. 13.

```
let user_data = {
            "function":'Authorization',
            "params": {
                "login": $scope.login,
                "password": $scope.password
            }
        };
```

Рис. 13. JSON-файл, отправляющийся на сервер при авторизации

Единообразие структуры таких файлов позволяет реализовывать принцип Remote Procedure Call следующим образом:

– поле «function» определяет функцию, вызываемую сервером;

– поле «params» определяет параметры, передаваемые в эту функцию.

Часть RPC-сервера, отвечающая за авторизацию пользователя, приведена на рис. 14.

```
$data = json_decode(file_get_contents('php://input'),true);
$func = $data['function'];
$params = $data['params'];

switch($func){
    case 'Authorization':
        {
          echo Authorization($params['login'],$params['password']);
        }
        break;
  //Here are other conditions}

 function Authorization($login, $password)
{
    $db = new mysqli('localhost', 'root', '', 'GraduationProject', '3307');
    $query = "select  Password_Hash,ID from Users where Login='" . $login .
"'";
    $result = $db->query($query);
    $array = $result->fetch_array(MYSQLI_ASSOC);
    $db->close();
    if ($array == null) {
        return 'no login';
    } else
      {
        $hash = $array['Password_Hash'];
        if (password_verify($password, $hash)) {
            setcookie("id", $array['ID'], time() + 60 * 60 * 24 * 2, '/');
            return 'ok'; }
        else {
            return 'wrong password';
             }
       }
  }

//Other functions
```

Рис. 14. Листинг фрагмента RPC-сервера

3.3. Взаимодействие веб-интерфейса с сервером

Для реализации одинаковых частей веб-страниц использовались шаблоны AngularJS. На рис. 15 представлено содержимое JavaScript – файла, подключаемого к каждой странице портала и отвечающего за вывод одинаковой шапки сайта с указанием данных пользователя из базы данных: аватара и имени.

```
let app = angular.module("app", []);
let template = "templates/HeaderTemplate.html";
if(getCookie('id')===undefined){
    template = 'templates/NonAuthorizedHeaderTemplate.html';
}
app.directive("head", function () {
    return {
        restrict: "A",
        templateUrl: template,
        transclude: true
    }
});
app.controller("headerCtrl", function ($scope, $http) {
    $scope.show_btn = false;
    let user_data = {
        "function":'GetInfo',
        "params": {
            "id": getCookie('id')
        }
};
    $http({
        method: 'POST',
        url: 'Index.php',
        data: user_data,
        headers: {'Content-Type':'application/x-www-form-urlencoded'}
    }).then(function success (response) {
        $scope.name = response.data['Name'];
        $scope.image = response.data['Avatar'];
        if(response.data['IsTeacher'] === '1'){
            $scope.show_btn = true;
        }});});
```

Рис. 15. Листинг функции получения персональных данных пользователя от сервера

Метод вызова удаленных процедур обладает свойством ассиметричности (одна из взаимодействующих сторон является инициатором) и синхронности (выполнение вызывающей процедуры приостанавливается с момента выдачи запроса и возобновляется только после возврата из вызываемой процедуры). Эти особенности позволяют порталу обращаться к серверу без перезагрузки веб-страницы и выводить результат запроса в режиме реального времени. Это необходимо, например, в момент проверки правильности решения задачи. Фрагмент JavaScript – файла, отсылающий код на сервер для проверки, представлен на рис. 16.

```
$scope.CheckCode = function () {
            $scope.show_verdict = false;
            let user_data = {
                "function":'CheckTask',
                "params": {
                    "user_id": getCookie('id'),
                    "course_id":$scope.course_id,
                    "task_id":$scope.task_id,
                    "code":$scope.code
                }};
            $http({
               url: 'Index.php',
               method: 'POST',
               data: user_data,
               headers:{'Content-Type':'application/x-www-form-urlencoded'}
            }).then(function success (response) {
                    if(response.data==='Yes'){
                        $scope.decision='Задача решена верно';
                        $scope.verdict=true;
                        $scope.image="images/right.png";
                    }
                    else{
                        $scope.decision='Задача решена неверно';
                        $scope.verdict=false;
                        $scope.image="images/wrong.png";
                    }
                $scope.show_verdict = true;
                });};
```

Рис. 16. Листинг функции отправки решения задачи на проверку

3.4. Реализация алгоритма индивидуальной траектории

Алгоритм индивидуальной траектории был реализован на языке PHP и интегрирован в серверную часть портала. На рис. 17 представлен листинг реализованного алгоритма.

```
function IndividualAlgorithm($user_id, $course_id){
    $db = GetDB();
    $query = 'call GetThemesForRepeating('.$course_id.','.$user_id.')';
    $result = $db->query($query);
    $array = $result->fetch_all(MYSQLI_ASSOC);
    $task_query = "select tt.Task_ID from Tasks t left join
TasksInCourses tc on tc.Task_ID=t.ID join TopicsOfTasks tt on
tt.Task_ID=t.ID
          where (tc.Course_ID<>".$course_id." or tc.Course_ID is NULL)
and(tt.Topic_ID=".$array[0]['Topic_ID'];
    for ($i=1;$i<count($array);$i++)
        $task_query .= " or tt.Topic_ID=".$array[$i]['Topic_ID'];
    $task_query .=")";
    $db->close();
    $db = GetDB();
    $tasks_result = $db->query($task_query);
    $tasks_array = $tasks_result->fetch_all(MYSQLI_ASSOC);
    $db->close();
    return $tasks_array;
}
```

Рис. 17. Листинг алгоритма индивидуальной траектории

Функция IndividualAlgorithm вызывает хранимую процедуру базы данных GetThemesForRepeating, возвращающую в соответствие со спроектированным ранее алгоритмом темы заданий, которые необходимо повторить пользователю, после чего формирует запрос, возвращающий список задач для проверки пользователя. Листинг определения процедуры ThemesForRepeating приведен на рис. 18.

Сама функция IndividualAlgorithm вызывается в функции MakeOptionalBlock, формирующей состав опционального блока для конкретного пользователя. Данная функция возвращает клиенту ID первой за-

дачи опционального блока для отображения на веб-странице. Листинг функции MakeOptionalBlock представлен на рис. 19.

```
SELECT DISTINCT tt.Topic_ID
FROM ProgressInCourses pc join CoursesOfUsers cu on pc.Course_User_ID=cu.ID
join TasksInCourses tc on tc.ID=pc.Task_In_Course_ID join Tasks t on
tc.Task_ID=t.ID join TopicsOfTasks tt on t.ID=tt.Task_ID
where cu.User_ID=userID and cu.Course_ID=courseID and pc.AttemptsNumber>
(
      select AVG(pc.AttemptsNumber)
    from ProgressInCourses pc join CoursesOfUsers cu on
pc.Course_User_ID=cu.ID
    where cu.User_ID=userID and cu.Course_ID=courseID
)
```

Рис. 18. Определение хранимой функции GetThemesForRepeating

```
function MakeOptionalBlock($opt_block_id,$user_id,$course_id)
{
    $db = GetDB();
    $query = 'select ID,Task_ID from OptionalTasks where
OptBlock_ID='.$opt_block_id;
    $result = $db->query($query);
    $array = $result->fetch_all(MYSQLI_ASSOC);
    if($array!=null)
    {
        //Опциональный блок уже сформирован
        return $array[0]['ID'];
    }
    else
     {
        $tasks_array = IndividualAlgorithm($user_id, $course_id);
        foreach($tasks_array as $task){
            $query ='insert into OptionalTasks(OptBlock_ID,Task_ID,User_ID)
values('.$opt_block_id.','.$task["Task_ID"].','.$user_id.')';
            $db->query($query);
        }
        $db->close();
        return $tasks_array[0]['ID'];
    }
}
```

Рис. 19. Листинг функции MakeOptionalBlock

На рис. 20 представлен фрагмент JavaScript файла, описывающий процесс отправки на сервер заявки о составлении опционального блока для пользователя.

```
let user_data = {
                "function":'MakeOptionalBlock',
                "params": {
                    "user_id": getCookie('id'),
                    "course_id":$scope.course_id,
                    "opt_block_id":$scope.opt_block_id
                }
            };
$http({
        method: 'POST',
        url: 'Index.php',
        data: user_data,
        headers:{'Content-Type':'application/x-www-form-urlencoded'}
  }).then(function success (response) {
         window.location.href = 'CoursePage.html?course=' +
$scope.course_id+'&opt_task=' + response.data + '&opt_block=' +
$scope.opt_block_id;
            });
```

Рис. 20. Листинг отправки на сервер заявки о составлении опционального блока

Выводы по третьему разделу

В данном разделе описан процесс реализации веб-интерфейса, алгоритма индивидуальной траектории и серверной части портала, приведены листинги часто используемых функций.

4. ТЕСТИРОВАНИЕ

4.1. Функциональное тестирование

Функциональное тестирование – это тестирование программного обеспечения в целях проверки реализуемости функциональных требований, т.е. способности программного обеспечения в определенных условиях решать задачи, нужные пользователям. В таблице 2 приведен протокол функционального тестирования веб-портала.

Табл. 2. Протокол функционального тестирования портала

№ п/п	Предусловия	Действия пользователя	Ожидаемый результат	Тест пройден?
1	Пользователь не авторизовался	Перейти на главную страницу	Открывается главная страница, в шапке страницы появляется кнопка «Войти»	Да
2	Пользователь не авторизовался в системе и перешел на главную страницу портала	Нажать кнопку «Пройти курс»	Открывается окно с просьбой авторизоваться	Да
3	Пользователь не авторизовался в системе и перешел на главную страницу портала	Нажать кнопку «Войти»	Осуществляется переход на страницу авторизации	Да
4	Пользователь перешел на страницу регистрации	Ввести уже существующий в системе логин	Появляется сообщение, что такой логин уже существует	Да

Продолжение табл. 2

№ п/п	Предусловия	Действия пользователя	Ожидаемый результат	Тест пройден?
5	Неавторизованный пользователь перешел на страницу регистрации	Ввести необходимые данные, нажать кнопку «Зарегистрироваться»	Новый пользователь вносится в БД, осуществляется переход на главную страницу	Да
6	Неавторизованный пользователь перешел на страницу авторизации	Ввести неверные данные	Появляется соответствующее сообщение, отказ в авторизации	Да
7	Пользователь перешел на страницу авторизации	Ввести верные данные	Осуществляется переход на главную страницу	Да
8	Пользователь находится на главной странице	Выбрать новый курс для прохождения	В БД вносятся соответствующие записи, осуществляется переход на страницу курса	Да
9	Пользователь находится на главной странице	Выбрать курс для продолжения прохождения	Из БД получаются данные о прогрессе курса, открывается страница курса	Да
10	Пользователь выбрал задачу для решения	Отправить решение задачи на сервер	Решение проверяется на сервере и выводится сообщение о правильности решения	Да

Продолжение табл. 2

№ п/п	Предусловия	Действия пользователя	Ожидаемый результат	Тест пройден?
11	Пользователь прошел очередную тему курса	Перейти в блок проверки усвоения материала	На сервере запускается модуль индивидуальной траектории и формируется список задач для проверки усвоения материала	Да
12	Пользователь авторизовался как преподаватель и перешел на страницу создания нового курса	Ввести название и описание курса, нажать кнопку «Перейти к наполнению курса»	В БД вносится запись о новом курсе, осуществляется переход на страницу редактирования только что созданного курса	Да
13	Пользователь авторизовался как преподаватель и перешел на страницу редактирования курса	Добавить новое задание в курс	В БД вносится соответствующие записи, на странице отображается текущий состав курса	Да
14	Пользователь авторизовался	Перейти в личный кабинет	Открывается страница личного кабинета пользователя	Да

4.2. Тестирование верстки

Было проведено тестирование интерфейса пользователя. Интерфейс был протестирован в следующих браузерах:

1) Internet Explorer 11;
2) Mozilla Firefox версии 50.2. и выше;
3) Safari версии 5 и выше;
4) Google Chrome версии 50 и выше;
5) Opera версии 40 и выше;
6) Яндекс.Браузер версии 16.9 и выше;
7) Microsoft Edge версии 38 и выше;
8) Mobile Chrome версии 30 и выше.

По результатам тестирования, проблем с версткой не обнаружено. Пользовательский интерфейс сохраняет идентичность в различных веб-браузерах. Результаты тестирования совпали с ожидаемыми. Тест пройден успешно.

4.3. Usability тестирование

Юзабилити-тестирование (проверка эргономичности) (англ. Usability testing) — исследование, выполняемое с целью определения, удобен ли некоторый искусственный объект (такой как веб-страница, пользовательский интерфейс или устройство) для его предполагаемого применения. Таким образом, проверка эргономичности измеряет эргономичность объекта или системы. Проверка эргономичности сосредоточена на определённом объекте или небольшом наборе объектов, в то время как исследования взаимодействия человек-компьютер в целом - формулируют универсальные принципы.

Проверка эргономичности - метод оценки удобства продукта в использовании, основанный на привлечении пользователей в качестве тестировщиков, испытателей и суммировании полученных от них выводов [13].

В ходе тестирования респондентам было предложено выполнить следующие действия:

- зарегистрироваться на портале в роли преподавателя;
- создать и наполнить новый курс;
- решить несколько задач из вновь созданного курса;
- изучить прогресс курса в личном кабинете.

Все задачи были решены респондентами без каких-либо затруднений. Тест пройден успешно.

Выводы по четвертому разделу

В данном разделе было проведено тестирование верстки и эргономичности разработанного веб-портала и приведен протокол функционального тестирования.

ЗАКЛЮЧЕНИЕ

В рамках данной работы был реализован веб-портал для обучения программированию. В ходе разработки были решены следующие задачи.

1. Провести анализ предметной области и аналогичных проектов.
2. Разработать веб-интерфейс портала.
3. Спроектировать и реализовать базу данных.
4. Реализовать серверную часть веб-портала.
5. Провести тестирование.

В дальнейшем планируется улучшать алгоритм индивидуальной траектории.

ЛИТЕРАТУРА

1. AngularJS. Официальное руководство. [Электронный ресурс] URL: https://docs.angularjs.org/tutorial (дата обращения: 10.04.2019).

2. Arpaci-Dusseau R.H., Arpaci-Dusseau A.C. – Operating Systems: Three Easy Pieces. – «Arpaci-Dusseau Books», 2015. – P.551-556.

3. Codeforces.com. [Электронный ресурс] URL: http://codeforces.com (дата обращения: 20.02.2019).

4. Google Fonts Analytics. [Электронный ресурс] URL: https://fonts.google.com/analytics (дата обращения: 10.02.2019).

5. Stepic – Бесплатные онлайн-курсы. [Электронный ресурс] URL: https://welcome.stepik.org/ru (дата обращения: 30.03.2019).

6. Ulearn.me. Интерактивные онлайн-курсы по программированию. [Электронный ресурс] URL: https://ulearn.me (дата обращения: 20.02.2019).

7. Буч Г., Рамбо Д., Якобсон И. Введение в UML от создателей языка. – М.: ДМК Пресс, 2015. – 496 с.

8. Вдовина С.А., Кунгурова И.М. Сущность и направления реализации индивидуальной образовательной траектории. // Интернет-журнал Науковедение. – 2013. – №. 6 (19) (дата обращения: 30.05.2019).

9. Вигерс К. Разработка требований к программному обеспечению. – М.: Издательско-торговый дом «Русская Редакция», 2004. – 576 с.

10. Дистанционная подготовка по информатике. [Электронный ресурс] URL: https://informatics.msk.ru (дата обращения: 20.02.2019).

11. Документация по Bootstrap. [Электронный ресурс] URL: https://getbootstrap.com/docs/4.3/getting-started/introduction/ (дата обращения: 06.03.2019).

12. Документация по MySQL. [Электронный ресурс] URL: https://dev.mysql.com/doc/ (дата обращения: 14.03.2019).

13. Нильсен Я., Лоранжер Х. Web-дизайн: удобство использования Web-сайтов – М.: «Вильямс», 2007. – 368 с.

14. Общероссийская база вакансий. [Электронный ресурс] URL: https://trudvsem.ru (дата обращения: 02.05.2019).

15. Пеккер П.Л. Востребованность онлайн курсов в России. // Современные информационные технологии и ИТ-образование. – 2016. – Т. 12. – № 4.

16. Роббинс Д. Н. Web-дизайн. Справочник – O'Reilly, 2008. – 816 с.

17. Руководство по PHP на русском языке. [Электронный ресурс] URL: https://www.php.net/manual/ru/ (дата обращения: 24.02.2019).

18. Хуторской А. В. – Методика личностно-ориентированного обучения. Как обучать всех по-разному? – М., 2005. – 383 с.

ПРИЛОЖЕНИЕ

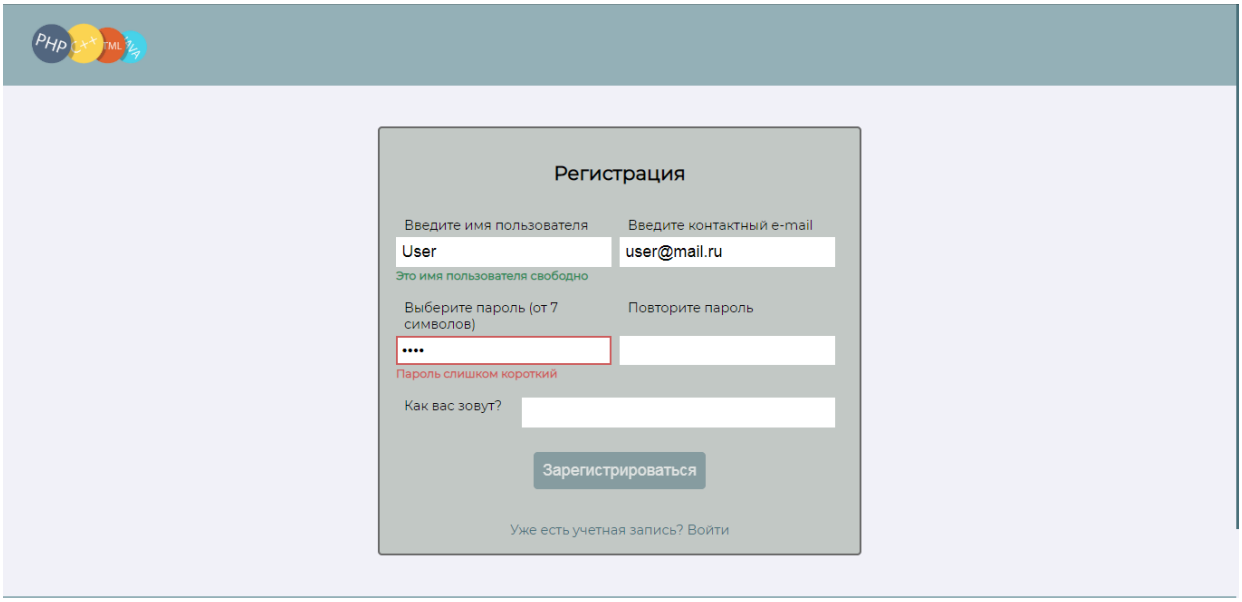

Рис. 1. Страница регистрации

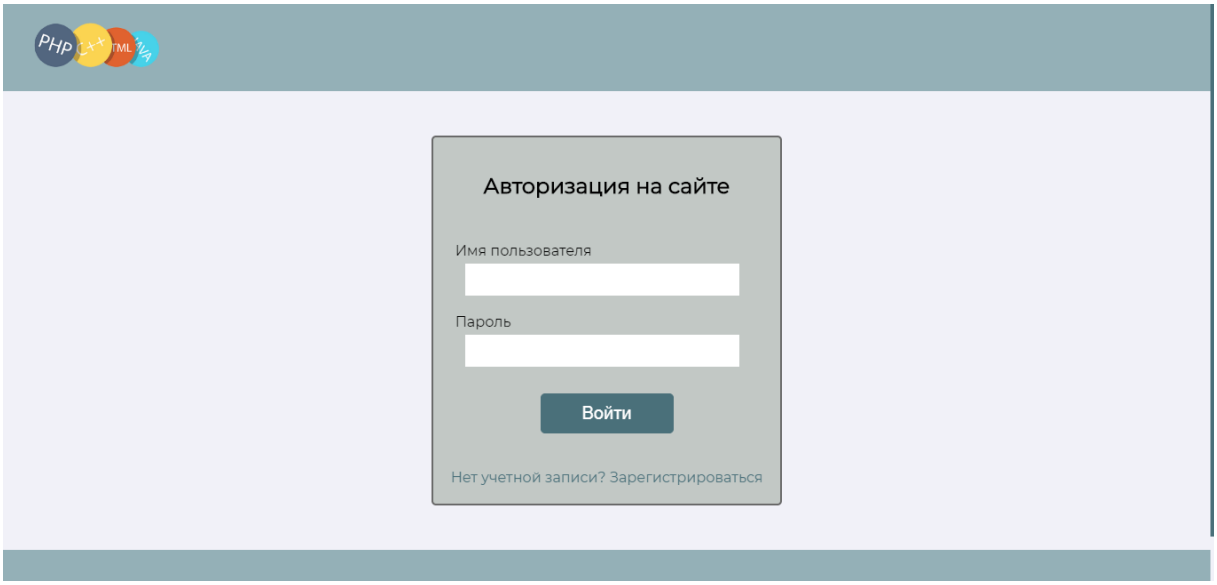

Рис. 2. Страница авторизации

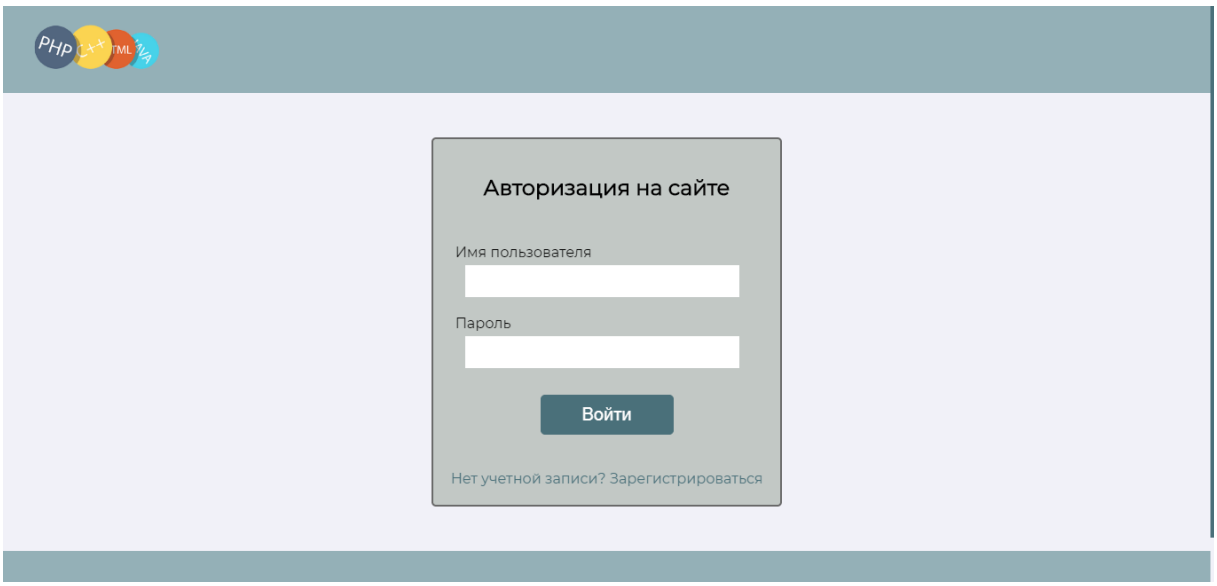

Рис. 3. Главная страница (неавторизованный пользователь)

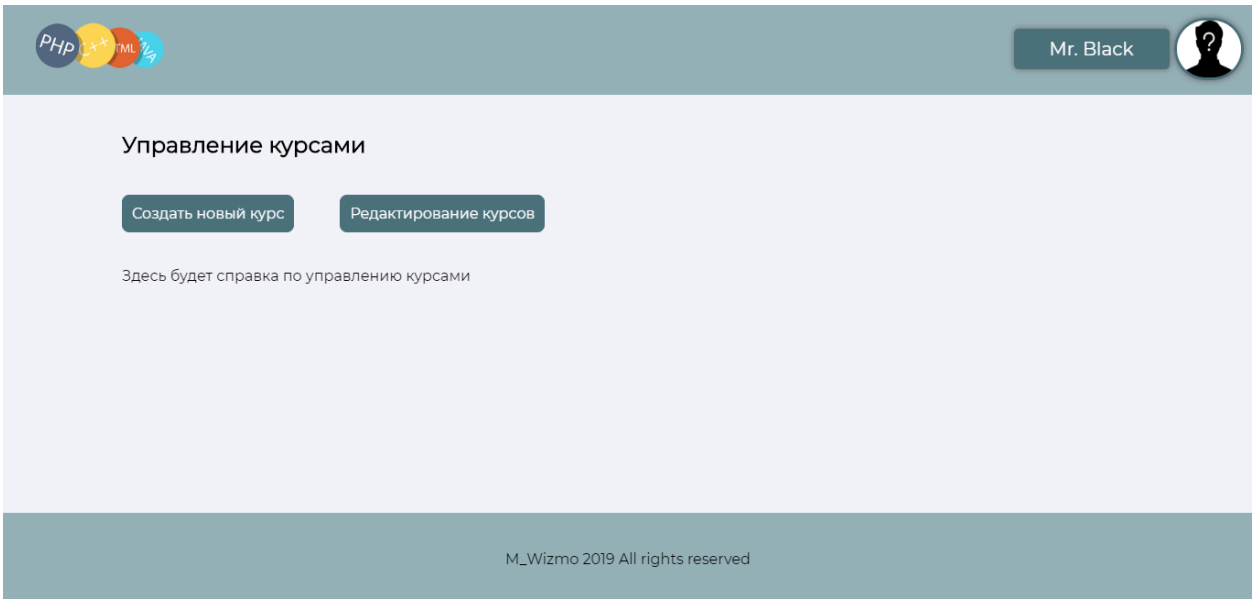

Рис. 4. Страница управления курсами

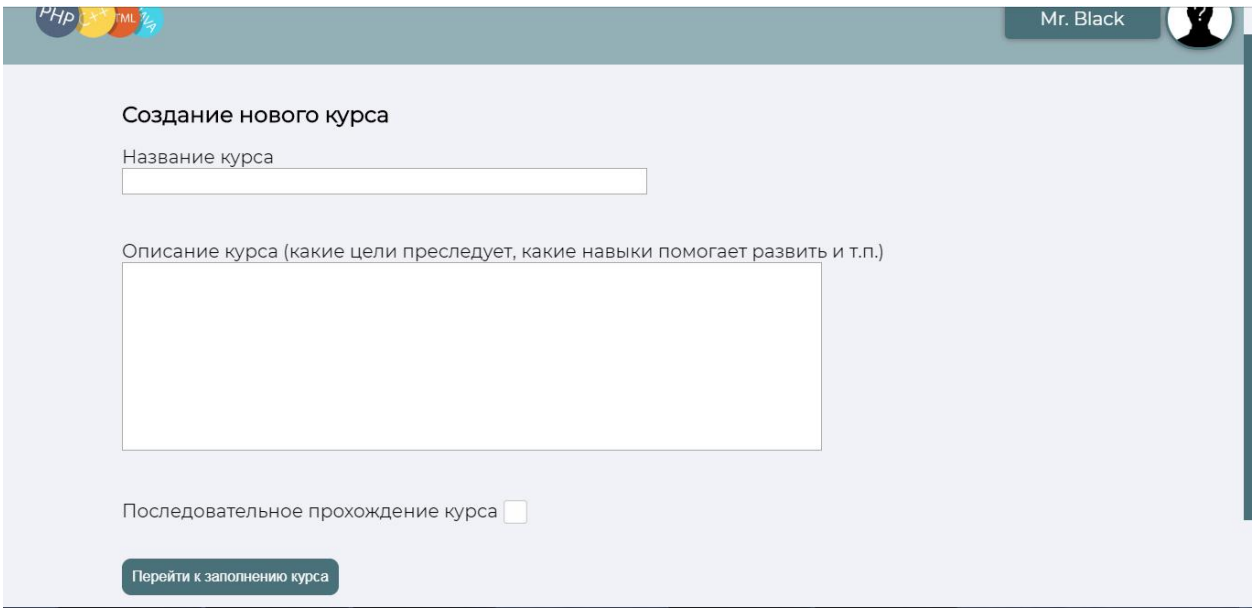

Рис. 5. Страница создания курса

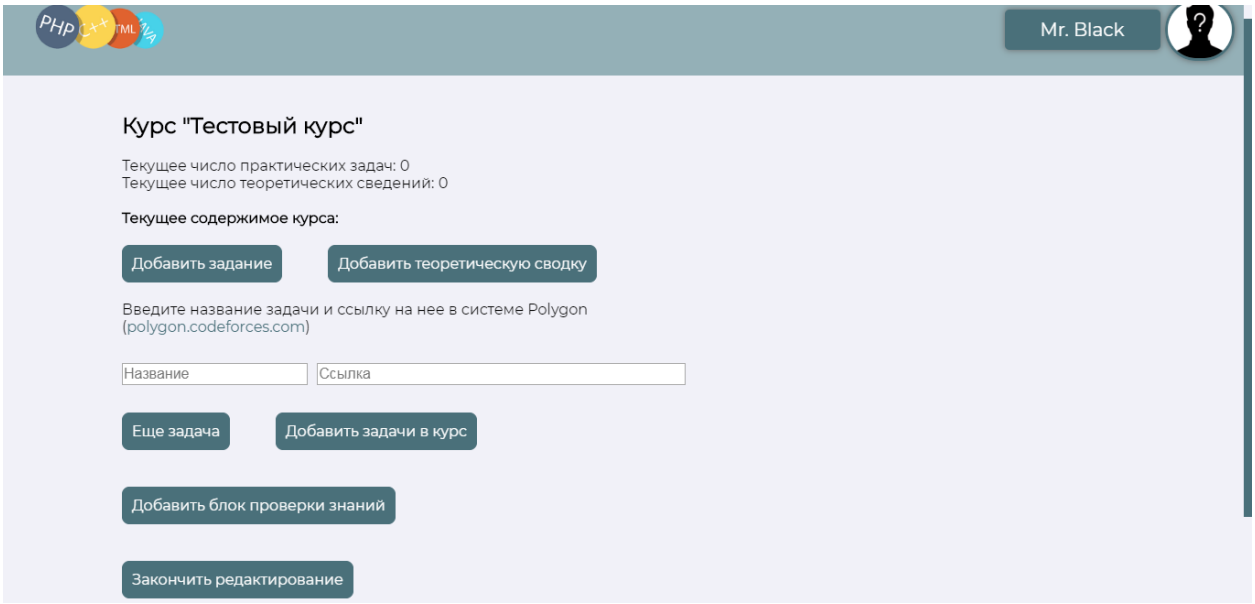

Рис. 6. Страница редактирования курса (пустой курс)

Рис. 7. Страница редактирования курса (наполненный курс)

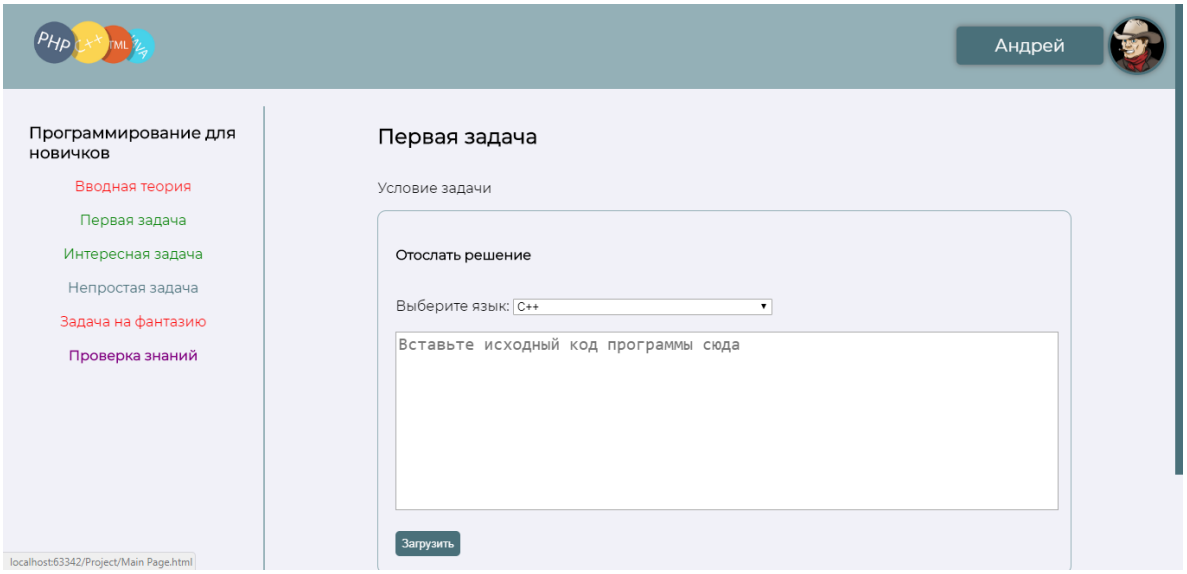

Рис. 8. Страница прохождения курса

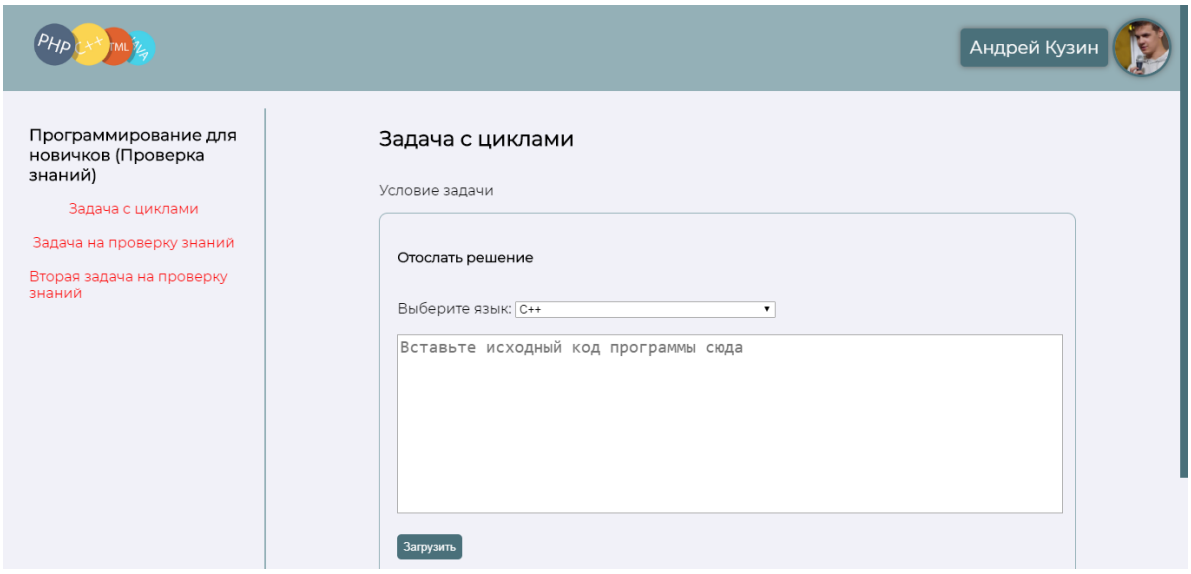

Рис. 9. Результат работы алгоритма индивидуальной траектории