

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение
высшего образования «Самарский национальный исследовательский
университет имени академика С.П. Королева (Самарский университет)»

Институт _____ информатики и кибернетики _____

Кафедра _____ программных систем _____

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

_____ к курсовой работе по дисциплине «Программная инженерия»
_____ по теме «Клавиатурный тренажер с функциями администратора»

Обучающийся _____ К.С. Лапин

Обучающийся _____ А.И. Петров

Обучающийся _____ С.А. Марковский

Руководитель _____ Л.С. Зеленко

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение
высшего образования «Самарский национальный исследовательский
университет имени академика С.П. Королева (Самарский университет)»

Институт _____ информатики и кибернетики _____

Кафедра _____ программных систем _____

ЗАДАНИЕ

на курсовую работу по дисциплине
«Программная инженерия»
обучающимся в группе № 6403-020302D
К.С. Лапину
А.И. Петрову
С.А. Марковскому

- 1 Тема проекта: «Клавиатурный тренажер с функциями администратора» _____
- 2 Исходные данные к проекту: см. приложение к заданию _____
- 3 Перечень вопросов, подлежащих разработке:
 - 3.1 Произвести анализ предметной области: изучить основные принципы составления упражнений для клавиатурного тренажера _____
 - 3.2 Выполнить обзор существующих систем-аналогов _____
 - 3.3 Разработать информационно-логический проект системы по методологии UML _____
 - 3.4 Разработать и реализовать программное и информационное обеспечение, провести его тестирование и отладку _____.
 - 3.5 Оформить документацию курсовой работы _____
 - 3.6 Подготовить презентацию по разработанной системе _____
- 4 Перечень графических разработок:
 - 4.1 Структурная схема системы _____
 - 4.2 Канонические диаграммы UML _____
 - 4.3 Схемы основных алгоритмов _____

5 Календарный план выполнения работ

№ п/ п	Содержание работы по этапам	Объем этапа в % к общему объему проекта	Срок окончания	Фактическое выполнение
1	Оформление технического задания и его утверждение	5	20.09.2024	
2	Описание и анализ предметной области	10	27.09.2024	
3	Проектирование системы	40	13.12.2024	
3.1	Разработка структурной схемы системы	5	11.10.2024	
3.2	Разработка функциональной спецификации системы и прототипа интерфейса пользователя	10	25.10.2024	
3.3	Разработка информационно-логического проекта системы и его предъявление руководителю	25	13.12.2024	
4	Реализация проекта, разработка контрольных примеров. Предъявление реализации руководителю	40	13.12.2024	
5	Корректировка проекта и оформление документации проекта. Защита проекта с представлением презентации.	5	27.12.2024	

Задание принял
к исполнению

_____ К.С. Лапин
 _____ А.И. Петров
 _____ С.А. Марковский

ПРИЛОЖЕНИЕ

к заданию на курсовую работу
обучающимся в группе № 6403-020302D

К.С. Лапину

А.И. Петрову

С.А. Марковскому

Тема проекта: «Клавиатурный тренажер с функциями администратора»

Исходные данные к проекту:

1 Характеристика объекта автоматизации:

- 1) объект автоматизации: клавиатурный тренажер;
- 2) виды автоматизируемой деятельности:
 - процесс авторизации/регистрации пользователей;
 - процесс ручного составления упражнения;
 - процесс автоматического составления упражнения;
 - процесс определения уровня подготовки обучаемого;
 - процесс настройки уровня сложности;
 - процесс выполнения упражнения;
 - процесс визуализации выполнения упражнения;
 - процесс сбора и сохранения статистики;
 - процесс визуализации статистики;
- 3) количество ролей пользователей – 2;
- 4) минимальная длина логина – 4 символа;
- 5) максимальная длина логина – 10 символов;
- 6) минимальная длина пароля – 4 символа;
- 7) максимальная длина пароля – 10 символов;
- 8) минимальное количество уровней сложности – 3;
- 9) максимальное количество уровней сложности – 12;
- 10) количество зон клавиатуры – 9;
- 11) минимальное количество знаков в упражнении – 20;
- 12) максимальное количество знаков в упражнении – 80;

- 13) минимальное время нажатия клавиши – 0.5 с;
 - 14) максимальное время нажатия клавиши – 1.5 с;
 - 15) минимальное количество символов в названии упражнения – 1;
 - 16) максимальное количество символов в названии упражнения – 10;
 - 17) количество способов отображения статистики – 3;
 - 18) максимальное количество упражнений на уровне сложности: 20;
 - 19) минимальное количество ошибок – 0;
 - 20) максимальное количество ошибок – 10% (с округлением вниз);
 - 21) количество способов создания упражнения – 2.
- 2 Требования к информационному обеспечению:
- 1) информационное обеспечение разрабатывается на основе следующих источников:
 - клавиатурный тренажер [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/Keyboard_trainer (дата обращения: 16.09.2024);
 - Гладкий А. А. Самоучитель слепой печати. Учимся быстро набирать тексты на компьютере. М.: Директмедиа Паблишинг, 2019. 116 с;
 - 2) структура базы данных (БД) разрабатывается на основании следующих сведений:
 - о пользователе (логин, пароль, статус, статистика);
 - об упражнении (название, уровень сложности, количество знаков, текст);
 - об уровне сложности (название, минимальное количество знаков, максимальное количество знаков, время между нажатиями на клавишу, количество допустимых ошибок);
 - статистика (пользователь, упражнение, дата выполнения, затраченное на выполнение время, количество ошибок, средняя скорость набора);
 - 3) должна быть обеспечена целостность базы данных и защита от

несанкционированного доступа.

3 Требования к техническому обеспечению:

3.1 Требования к техническому обеспечению серверной части:

- 1) тип ЭВМ – IBM PC совместимый;
- 2) объем ОЗУ – не менее 2 Гб;
- 3) объем свободного пространства на внешнем диске – не менее 50 Гб;
- 4) наличие подключения к сети Интернет;
- 5) манипулятор – мышь;
- 6) технические характеристики определяются в процессе выполнения проекта.

3.2 Требования к техническому обеспечению клиентской части:

- 1) тип ЭВМ – IBM PC совместимый;
- 1) монитор с разрешающей способностью не ниже 800 x 600;
- 2) манипулятор – мышь, клавиатура;
- 3) технические характеристики определяются в процессе выполнения проекта.

4 Требования к программному обеспечению:

4.1 Требования к программному обеспечению серверной части:

- 1) тип операционной системы – Windows 7 и выше;
- 2) СУБД – MySQL 8.

4.2 Требования к программному обеспечению клиентской части:

- 1) тип операционной системы – Windows 7 и выше;
- 2) браузер – Google Chrome 86.0.4240.183 (64-битный) и выше, Firefox 83.0 (64-битный) и выше.

4.3 Требования к программному обеспечению рабочего места разработчика:

- 1) тип операционной системы – Windows 7 и выше;
- 2) язык программирования – Python 3.12, TypeScript 5.0;
- 3) фреймворк – Flask 3.0.3, Vue 3.5.6;
- 4) среда программирования – VSCode 1.93, PyCharm 2024 community

edition;

5) СУБД – MySQL 8;

6) среда проектирования – StarUML 5.2.0.

5 Общие требования к проектируемой системе:

5.1 Функции, реализуемые системой:

1) функции системы:

- аутентификация пользователя в системе, настройка интерфейса пользователя на заданную роль;
- оценка выполнение тестового задания для определения уровня сложности для обучаемого;
- формирование списка упражнений по заданному уровню сложности;
- контроль длины упражнения и допустимых символов при ручном вводе;
- контроль введенных символов при выполнении упражнения;
- генерация текста упражнения по заданным настройкам;
- генерация длины упражнения для автоматического создания;
- визуализация процесса выполнения упражнения;
- визуализация статистики по заданному критерию;
- контроль выполнения упражнения и сообщение о результатах;
- сбор статистики и ее сохранение;
- выдача справочной информации о системе;

2) функции администратора:

- авторизация пользователя в системе (ввод логина и пароля);
- создание нового уровня сложности;
- настройка уровня сложности:
 - 1) название (номер);
 - 2) задание выбора зон клавиатуры;
 - 3) задание минимальной длины упражнения;

- 4) задание максимальной длины упражнения;
- 5) задание допустимого времени нажатия на клавишу;
- 6) задание максимально допустимое количество ошибок;
- создание/редактирование упражнения:
 - 1) выбор уровня сложности;
 - 2) выбор способа создания упражнения;
 - 3) ввод текста;
 - 4) задание длины текста упражнения для автоматической генерации;
- сохранение упражнения в БД;
- загрузка упражнения из БД;
- работа со статистикой:
 - 1) просмотр статистики по пользователям;
 - 2) просмотр статистики по упражнениям;
 - 3) просмотр статистики по уровням;
- работа с пользователями:
 - 1) добавление пользователей;
 - 2) блокировка пользователей;
- просмотр справочной информации;
- 3) функции игрока:
 - регистрация пользователя в системе (ввод логина, пароля);
 - авторизация пользователя в системе (ввод логина, пароля);
 - выполнение тестового задания;
 - выбор уровня сложности упражнения;
 - выбор упражнения из списка;
 - просмотр статистики:
 - 1) собственной статистики;
 - 2) средней скорости набора;
 - просмотр справочной информации.

5.2 Технические требования к системе:

- 1) режим работы – диалоговый;
- 2) система должна удовлетворять санитарным правилам и нормам СанПин 2.2.2./2.4.2198-07;
- 3) условия работы средств вычислительной техники (содержание вредных веществ, пыли и подвижность воздуха) должны соответствовать ГОСТ 12.1.005, 12.01.007;
- 4) температура окружающего воздуха – 15-35°C;
- 5) влажность воздуха – 45-75%.

Руководитель

проекта _____ Л.С. Зеленко

Задание принял

к исполнению

_____ К.С. Лапин

_____ А.И. Петров

_____ С.А. Марковский

РЕФЕРАТ

Пояснительная записка 40 с, 14 рисунков, 5 таблиц¹, 12 источников, 2 приложения.

Графическая часть: ??? слайдов презентации PowerPoint.

ДЕРЕВО ПОИСКА, ГЕНЕРАТОР КРОССВОРДОВ, ГОЛОВОЛОМКА, СЛОВАРЬ ТЕРМИНОВ, ВАРИАНТ ОТОБРАЖЕНИЯ, РАЗГАДЫВАНИЕ

Объектом автоматизации является линейный кроссворд.

Во время курсового проектирования разработаны алгоритмы и соответствующая им программа, позволяющая выполнять автоматическую генерацию линейного кроссворда по заданной теме. Задания (понятие и его расшифровка) хранятся в текстовом файле и могут дополняться вручную внутри программы, при этом ограничений на длину словаря не существует. Тема кроссворда выбирается пользователем в соответствии с содержанием словаря заданий. Программа позволяет сформировать кроссворд, учитывая ограничения на параметры. В системе имеется возможность сохранения кроссвордов в файл с целью последующего их разгадывания.

Программа написана на языке C# в среде Visual Studio 2015 и функционирует под управлением операционной системы Windows 7 и выше. Доступ к данным осуществляется с помощью СУБД PostgreSQL 10.

¹ Количество страниц, рисунков, таблиц указывается с учетом приложений

СОДЕРЖАНИЕ

Введение.....	13
1 Описание и анализ предметной области.....	15
1.1 Описание предметной области	15
1.2 Описание систем-аналогов.....	22
1.2.1 Название системы-аналога 1 Ошибка! Закладка не определена.	
1.2.2 Название системы-аналога 2	23
1.3 Диаграмма объектов предметной области Ошибка! Закладка не определена.	
1.4 Постановка задачи	28
2 Проектирование системы	33
2.1 Выбор и обоснование архитектуры системы	33
2.2 Структурная схема системы	45
2.3 Разработка спецификации требований	47
2.3.1 Функциональная спецификация	48
2.3.2 Перечень исключительных ситуаций	49
2.4 Разработка прототипа интерфейса пользователя системы	53
2.5 Разработка информационно-логического проекта системы.....	54
2.5.3 Язык UML	56
2.5.4 Диаграмма вариантов использования	56
2.5.5 Сценарии	56
2.5.6 Диаграмма классов.....	58
2.5.7 Диаграмма состояний	58
2.5.8 Диаграмма деятельности	58
2.5.9 Диаграмма последовательности	59
2.6 Логическая модель данных (при необходимости).....	60
2.7 Выбор и обоснование алгоритмов обработки данных /Разработка и описание алгоритмов обработки данных	61
2.8 Выбор и обоснование комплекса программных средств.....	63
2.8.1 Выбор языка программирования	63
2.8.2 Выбор операционной системы	64

2.8.3	Выбор среды программирования	64
2.8.4	Выбор системы управления базами данных (при необходимости).....	64
3	Реализация системы	65
3.1	Разработка и описание интерфейса пользователя	65
3.2	Диаграммы реализации	66
3.2.1	Диаграмма компонентов	66
3.2.2	Диаграмма развертывания.....	67
3.2.3	Диаграмма классов.....	67
3.3	Физическая модель данных (при необходимости)	68
3.4	Выбор и обоснование комплекса технических средств.....	69
3.4.1	Расчет объема занимаемой памяти.....	69
3.4.2	Минимальные требования, предъявляемые к системе.....	71
	Заключение	72
	Список использованных источников	73
	Приложение А Руководство пользователя	76
A.1	Назначение системы	76
A.2	Условия работы системы	76
A.3	Установка системы	76
A.4	Работа с системой	77
A.4.1	Работа с системой в режиме администратора (если необходимо). ..	77
A.4.2	Работа с системой в режиме пользователя	77
	Приложение Б Листинг модулей программы.....	78

ВВЕДЕНИЕ

Одним из основных критериев умения человека работать на компьютере является его способность быстро и правильно набирать тексты. Поэтому с каждым днем растет число пользователей компьютера, желающих самостоятельно и в короткие сроки научиться быстрому и качественному набору текстов. Секретари, референты, журналисты, копирайтеры, веб-разработчики, менеджеры, студенты и школьники – вот далеко не полный перечень лиц, для которых умение владеть навыками «слепой» печати является необходимым.

«Слепой» метод набора был разработан Франком Эдгаром Макгуррином, стенографистом суда из Солт-Лейк-Сити. 25 июля 1888 года Макгуррин, будучи единственным известным человеком, в то время использующим данный метод, одержал решающую победу над Луисом Тробом, который пользовался восьми пальцевым зрячим методом печати. Это было первым соревнованием по печатанию, проводилось оно в Цинциннати [1].

Навыки «слепой» печати становятся все более ценными в современном мире, где скорость и точность ввода информации играют ключевую роль.

В начале 20-го века появились механические клавиатурные тренажеры, которые использовали систему с «закрепленными» клавишами, чтобы научить пользователей правильно позиционировать пальцы [2].

С развитием компьютеров появились электронные клавиатурные тренажеры, которые могли отслеживать скорость и точность печати, а также предлагать упражнения и тесты.

Сегодня клавиатурные тренажеры стали неотъемлемой частью обучения грамотной работе на компьютере, предоставляя пользователям доступные инструменты для развития навыков печати и повышения эффективности работы.

Во время курсового проектирования необходимо разработать клавиатурный тренажер для тренировки слепой печати, с помощью которого

администратор сможет создавать упражнения, настраивать различные уровни сложности, а обучающийся сможет выполнять эти упражнения.

Разработка системы будет производиться по технологии быстрой разработки приложений Rapid Application Development (RAD), которая поддерживается методологией структурного проектирования и включает элементы объектно-ориентированного проектирования и анализа предметной области [3].

При проектировании системы будут использоваться методология Object-Oriented Analysis/Design (ООАП), в основу которой положена объектно-ориентированная методология представления предметной области в виде объектов, являющихся экземплярами соответствующих классов, и язык моделирования UML (Unified Modeling Language), который является стандартным инструментом для разработки «чертежей» программного обеспечения [4].

1 Описание и анализ предметной области

Под предметной областью (application domain) принято понимать ту часть реального мира, которая имеет существенное значение или непосредственное отношение к процессу функционирования программы. Другими словами, предметная область включает в себя только те объекты и взаимосвязи между ними, которые необходимы для описания требований и условий решения некоторой задачи [5].

1.1 Описание предметной области

1.1.1 Основные понятия и определения

Клавиатура – устройство, позволяющее осуществлять ввод информации в электронно-вычислительную машину (ЭВМ) и управлять ею. Все клавиатуры унифицированы, у всех – одинаковый набор клавиш, как на печатных машинках, а также имеются дополнительные клавиши управления движением курсора. Часто в них устанавливаются ещё и малые клавиатуры – цифровые блоки, содержащие цифровые клавиши и кнопки осуществления математических операций [6]. Пример компьютерной клавиатуры приведен на рисунке 1.



Рисунок 1 – Пример компьютерной клавиатуры

«Слепой» метод печати – способ печати на клавиатуре, при котором пользователь не смотрит на клавиши, а использует мышечную память и сенсорные ощущения.

Клавиатурный тренажёр – вид компьютерных программ или онлайн-сервисов, предназначенных для обучения набору на компьютерной клавиатуре. Обычно целями тренажёров являются научить слепому методу печати [7]:

- в частности, задействовать для набора все десять пальцев рук;
- увеличить скорость набора;
- уменьшить количество опечаток;
- улучшить ритмичность набора (что позволяет уменьшить усталость при наборе).

Под скоростью печати подразумевают количество символов, введенных в минуту (символов в минуту, С/М). А под точностью печати подразумевают долю правильных символов в напечатанном тексте.

Клавиатурные тренажеры отличаются по формату, набору функций и способу доступа. Выбирая тренажер, важно учесть свой уровень подготовки, цели обучения, доступ к компьютеру и предпочтения. Пример того, как может выглядеть клавиатурный тренажер приведен на рисунке 2.

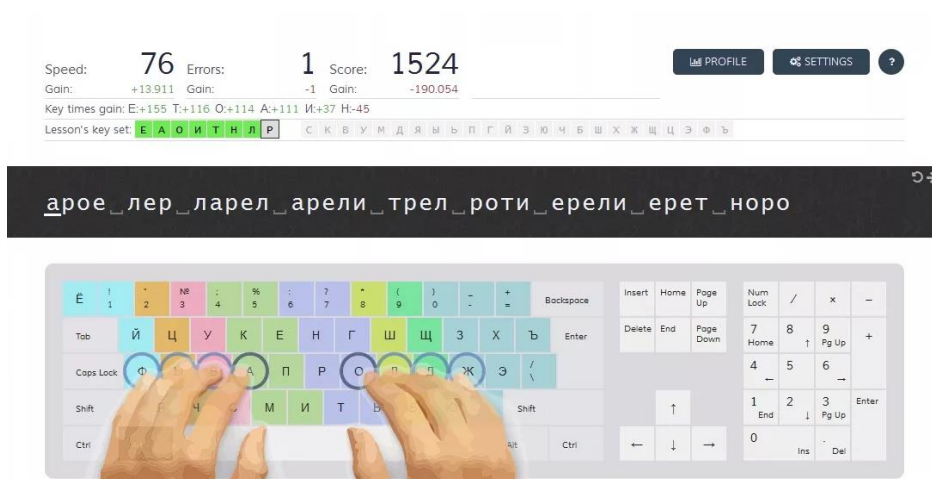


Рисунок 2 – Пример клавиатурного тренажера

1.1.2 Классификация типов клавиатур

Клавиатуры можно классифицировать по различным признакам, в частности [8]:

- 1) по типу подключения:
- проводные. Подключение к компьютеру осуществляется через

кабель;

- беспроводные. Подключение к компьютеру осуществляется по беспроводному каналу (Bluetooth, радиочастотный);

2) по типу раскладки:

- QWERTY. Стандартная англоязычная раскладка, которая используется в большинстве стран мира;

- ЙЦУКЕН. Стандартная русскоязычная раскладка;

- DVORAK. Раскладка, которая разработана специально для повышения скорости печати;

- AZERTY. Раскладка, которая используется в некоторых европейских странах, например, во Франции, Бельгии, Швейцарии;

3) по размеру:

- полные клавиатуры. Содержат все стандартные клавиши, в том числе и цифровую панель, находящуюся справа клавиатуры;

- компактные клавиатуры. Не содержат цифровую панель, могут иметь уменьшенный размер клавиш;

- мини-клавиатуры. Такие клавиатуры являются очень компактными, часто используются с ноутбуками или мобильными устройствами;

4) по функциональности:

- механические клавиатуры. В них используются специальные механические переключатели, которые обеспечивают тактильную отдачу, удобство набора текста и более долговечную работу;

- мембранные клавиатуры. В них используются мембранные переключатели, которые дешевле, но не так долговечны, как механические;

- ножничные клавиатуры. Они работают с помощью двух пластиковых лезвий, которые соединены посредством шарнирного механизма, похожего на ножницы. Используются в ноутбуках, обеспечивают более компактный дизайн и удобный набор текста;

– сенсорные клавиатуры. Такие клавиатуры используют сенсорную панель, которая имитирует клавиши, часто используются в мобильных устройствах;

5) по форме:

– прямоугольная форма. Стандартная форма, используемая в большинстве клавиатур;

– эргономичные клавиатуры. Имеют искривленную форму, чтобы снизить нагрузку на кисти и запястья;

– сплит-клавиатуры. Клавиатура разделена на две части, которые располагаются под углом друг к другу, чтобы руки находились в более естественном положении.

1.1.3 Описание клавиатурных раскладок

QWERTY – наиболее популярная в настоящее время латинская раскладка клавиатуры, используемая для английского языка. На её основе создано большинство раскладок для языков, использующих латиницу. Название произошло от 6 левых символов верхнего ряда [9]. Пример раскладки QWERTY приведен на рисунке 3.

~	!	@	#	\$	%	^	&	*	()	-	=	Backspace
Tab	Q	W	E	R	T	Y	U	I	O	P	{	}	\
Caps Lock	A	S	D	F	G	H	J	K	L	:	"	;	Enter
Shift	Z	X	C	V	B	N	M	<	>	?	/	Shift	
Ctrl	Win	Alt							Alt	Win	Menu	Ctrl	

Рисунок 3 – Пример раскладки QWERTY

ЙЦУКЕН – основная русскоязычная раскладка клавиатуры компьютеров и пишущих машинок. Название произошло от 6 левых знаков верхнего ряда раскладки. Пробраз раскладки появился в конце XIX века, в середине 1950-х годов раскладка стала похожа на современную. Некоторые изменения происходили в 1990-е годы, связано это было с развитием и

повсеместным распространением вычислительной техники [10]. Пример раскладки ЙЦУКЕН приведен на рисунке 4.



Рисунок 4 – Пример раскладки ЙЦУКЕН

Клавиатура с раскладкой Дворака была разработана доктором Августом Дворак в 1930-х гг. для повышения скорости, эффективности печати и снижения утомления пальцев. Все гласные расположены слева в основном ряду, а наиболее распространенные согласные расположены справа. Так как наиболее употребимые буквы располагаются в основном ряду, а менее распространенные прямо над ними, вы можете сильно облегчить процесс печатания. 70% букв этого абзаца расположены в основном ряду раскладки Дворак, остальные 15% ниже и другие 15% выше. В QWERTY лишь 30% букв в основном ряду [11]. Пример раскладки Дворака приведен на рисунке 5.



Рисунок 5 – Пример раскладки Дворака

Еще один пример похожей на QWERTY раскладки. Этот вариант используется большинством франкоязычных стран, хотя Франция и Бельгия имеют свои собственные национальные вариации и AZERTY не всем

приходится по вкусу. Q меняется с А и W с Z в верхнем ряду. Точка с запятой заменена на клавишу М. Пример раскладки AZERTY приведен на рисунке 6 [12].

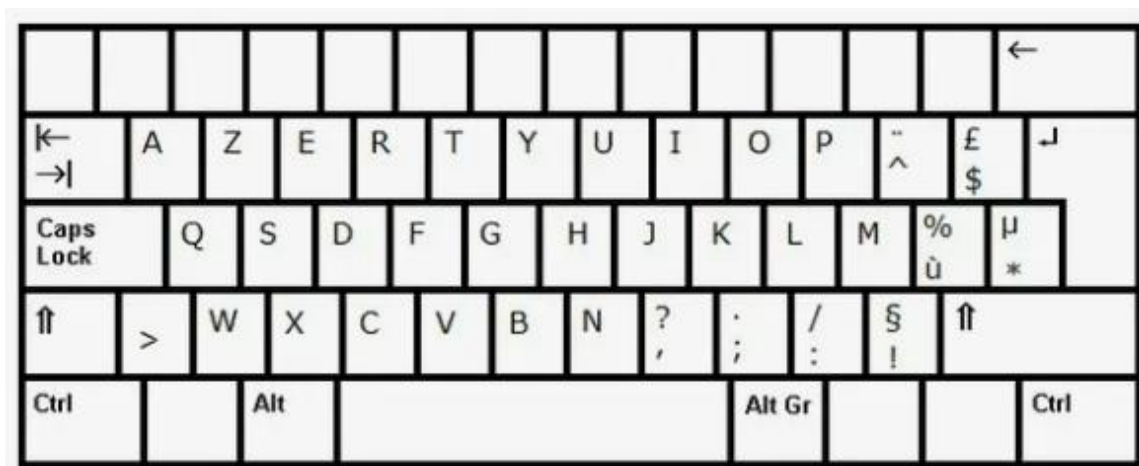


Рисунок 6 – Пример раскладки AZERTY

1.1.4 Клавиатурные зоны

Клавиатурные зоны – это важная концепция в обучении слепому методу печати. Они представляют собой разделение клавиш на группы, которые отвечают за определенные пальцы. Понимание зон и назначение пальцев для каждой из них – ключевой момент в овладении слепым методом печати.

Важно понимать, что каждому пальцу отводится своя позиция. Чтобы запомнить, каким пальцем, что нажимать, надо мысленно разбить клавиатуру на зоны. Основным «полем битвы» при освоении слепого метода является область алфавитно-цифровой клавиатуры. Мысленно эту область разбивают на две части: для левой и правой руки. Каждый палец ответственен за определенную группу клавиш [8].

Итак, левая рука:

- указательный палец отвечает за буквы «А», «Е», «И», «К», «М», «П» и цифры «4» и «5»;
- среднему пальцу соответствуют буквы «У», «В», «С» и цифра «3»;
- безымянный палец отвечает за буквы «Ц», «Ы», «Ч» и цифру «2»;
- мизинцу приходятся «Ф», «Я», «Й», цифра «1» и буква «Ё».

Правая рука отвечает за:

- указательный палец – буквы «О», «Р», «Н», «Г», «Т», «Ь», цифры «6» и «7»
- средний палец – «Ш», «Л», «Б», цифра «8»;
- безымянный – «Щ», «Д», «Ю», цифра «9»;
- мизинец – «З», «Ж», «Э», «Х», «Ъ», цифра «0», знаки «-», «=», «\».

Клавиша «Пробел» нажимается большими пальцами.

Такое расположение букв на клавиатуре закономерно. Те буквы, которые используются чаще всего в текстах, сосредоточены в центре, и за них отвечают указательные пальцы – самые подвижные и натренированные. Редкие буквы и клавиши, находящиеся рядом с буквенной клавиатурой, «достались» мизинцам.

Так, левым мизинцем нажимают клавиши «Tab», «Caps Lock», левые клавиши «Shift», «Ctrl», «Alt». Мизинцем правой руки – правые клавиши «Alt», «Ctrl», «Shift», а также «Enter», «Backspace».

Пример расположения зон клавиатуры приведен на рисунке 7.

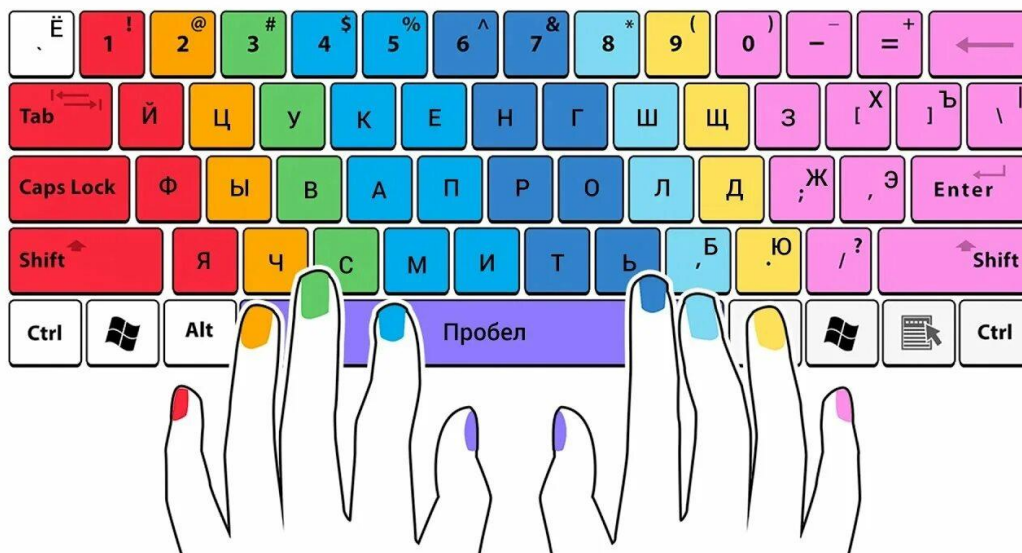


Рисунок 7 – Пример расположения зон на клавиатуре

1.1.5 Как устроены клавиатурные тренажеры

Клавиатурный тренажер – это не просто набор упражнений, а сложная программа, которая использует различные механизмы для обучения и

отслеживания прогресса пользователя. Внутри него хранится база данных, содержащая информацию о расположении клавиш, правильных движениях пальцев, словарях слов и фраз для упражнений, и других необходимых данных. Движок обучения, используя алгоритмы, строит индивидуальный план обучения и выбирает упражнения, учитывая уровень подготовки пользователя. Система упражнений предлагает разнообразные задачи: от обучающих, показывающих расположение клавиш и правильные движения пальцев, до упражнений на скорость и точность, а также игры, делающие обучение более интересным и мотивирующим, и тесты для проверки уровня знаний и навыков.

В состав клавиатурного тренажера обычно встраивается система для отслеживания прогресса, которая анализирует результаты пользователя, фиксирует достижения и предоставляет статистику, а удобный интерфейс обеспечивает взаимодействие с программой, настройку параметров, отслеживание прогресса и выбор упражнений.

Некоторые тренажеры используют адаптивный алгоритм, который изменяет сложность упражнений в зависимости от прогресса пользователя. Они сохраняют историю тренировок, отображают графики прогресса и дают рекомендации по улучшению результатов. Интеграция с другими приложениями позволяет использовать тренажер в текстовых редакторах, например, чтобы сразу проверять скорость печати в реальном времени. Это делает их идеальным инструментом для изучения нового навыка и повышения продуктивности.

1.2 Описание систем-аналогов

Мир клавиатурных тренажеров разнообразен и предлагает широкий выбор решений для обучения слепому методу печати. Каждый тренажер обладает уникальными особенностями и функционалом, что позволяет пользователю выбрать наиболее подходящий вариант, учитывая свои

потребности и цели. Давайте рассмотрим несколько популярных систем, чтобы лучше понять их преимущества и недостатки.

1.1.1 Stamina-online

Stamina-online – уникальный клавиатурный тренажер, который предлагает два подхода к обучению слепому методу печати [13]. Помимо традиционного расположения пальцев на клавишах ASDF, он позволяет освоить альтернативный метод, где пальцы располагаются на нижнем ряду клавиатуры SDFV. Разработчик утверждает, что такое положение кистей рук снижает усталость.

Тренажер предлагает практические занятия, включающие уроки для запоминания расположения клавиш и работу с текстами и фразами, которые могут быть предоставлены как программой, так и загружены из внешнего файла. Виртуальная клавиатура, интерактивная и удобная, не только подсвечивает нужные буквы, но и визуально показывает область работы каждого пальца.

Stamina-online выделяется своим легким и юморным стилем в разделах помощи и FAQ. Полезная информация разбавляется анекдотами и интересными логическими задачами, делая обучение более занимательным. На рисунке 8 приведен пример экранной формы программы Stamina-online.



Рисунок 8 – Главный экран программы «Stamina-online»

К достоинствам данной системы относятся:

- бесплатный доступ;
- понятный интерфейс;
- наличие уровней сложности;

- отображение используемых пальцев;
- наличие теории;
- наличие русской раскладки;
- возможность тренироваться на языках программирования;
- есть счет времени;
- визуализация клавиатуры.

К недостаткам системы относятся:

- большая нагрузка на руки с первых занятий;
- большое количество рекламы;
- отсутствие визуализации статистики.

1.2.6 TypingClub

TypingClub – это бесплатный онлайн-тренажер, ориентированный на обучение слепому методу печати, который использует игровую механику для повышения мотивации и заинтересованности у пользователя [14]. Он предлагает широкий выбор упражнений, включая игры, которые стимулируют развитие скорости и точности печати в интерактивном формате. Например, игра «TypeRacer» представляет собой гонку на скорость, где пользователи соревнуются в наборе текста, что может служить дополнительным стимулом к улучшению результатов.

TypingClub предлагает разные уровни сложности, что позволяет адаптировать обучение к уровню подготовки пользователя. Для повышения мотивации в тренажере используются персонажи, которые поддерживают пользователей на пути к мастерству слепого метода печати. Тренажер также отслеживает прогресс пользователя и предоставляет детальную статистику о скорости, точности и других показателях, что позволяет анализировать динамику обучения. Кроме того, TypingClub предлагает специализированные курсы, которые помогают отработать навыки печати в конкретных областях, например, для программистов, секретарей или журналистов. Пример экранной формы приложения TypingClub приведен на рисунке 9.

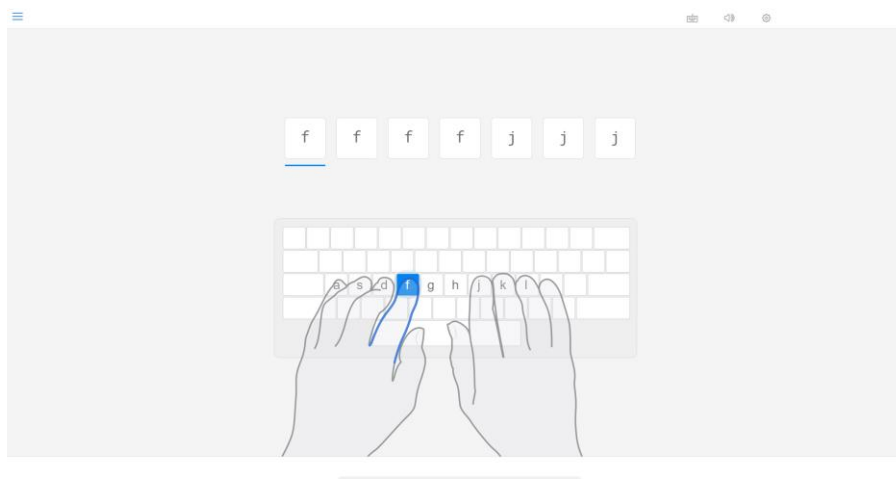


Рисунок 9 – Экранная форма программы «TypingClub»

К достоинствам данной системы относятся:

- приложение бесплатное;
- реализовывает игровой формат;
- наличие разнообразия упражнений;
- наличие различных уровней сложности;
- отображение используемых пальцев;
- наличие статистики и отслеживания прогресса.

К недостаткам системы относятся:

- наличие рекламы;
- зависимость от интернета;
- отсутствие счета времени;
- отсутствие русской раскладки.

На основании анализа возможностей систем-аналогов были сформулированы требования к разрабатываемой системе (см. таблицу 1).

1.3 Диаграмма объектов предметной области

В объектном или объектно-ориентированном подходе в первую очередь выделяется множество основных объектов системы и впоследствии определяется множество операций над объектами. Такой подход базируется на абстрактных типах, и решение задачи выражается в терминах выделенных объектов [15].

Таблица 1 – Сравнительные характеристики систем-аналогов

Название системы Название показателя	Система «Stamina-online»	Система «TypingClub»	Разрабатываемая система
Бесплатный доступ	+	+	+
Удобство использования	+	+	+
Уровень сложности	+	+	+
Визуализация клавиатуры	+	+	+
Визуализация статистики	–	+	+
Иностранные языки	+	+	–
Наличие русской раскладки	+	–	+
Наличие рекламы	+	+	–
Счёт времени	+	–	+

Объектно-ориентированный анализ и проектирование (ООАП, Object-Oriented Analysis/Design) - технология разработки программных систем, в основу которых положена объектно-ориентированная методология представления предметной области в виде объектов, являющихся экземплярами соответствующих классов [4].

На рисунке 10 приведена диаграмма объектов предметной области.

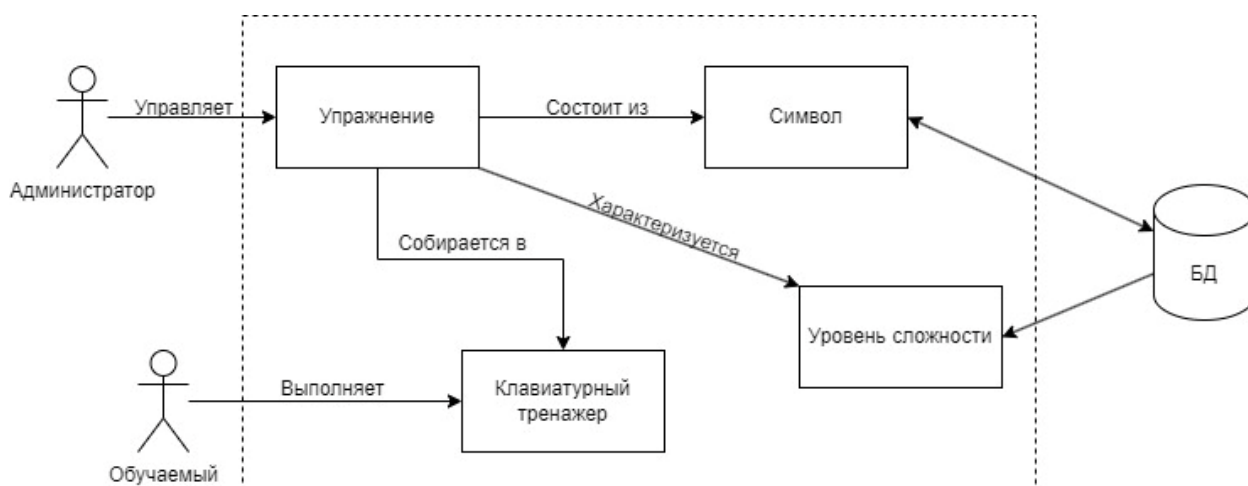


Рисунок 10 – Диаграмма объектов предметной области

Упражнение – это объект, представляющий собой отдельный блок обучения в клавиатурном тренажере. Оно содержит набор символов, который пользователь должен напечатать, и может иметь различные параметры, влияющие на процесс обучения. Каждое упражнение характеризуется уровнем сложности и состоит из символов, текст упражнения ограничен некоторым количеством знаков.

Символы – объект представляет собой набор символов, используемых в упражнении клавиатурного тренажера. В него входят как отдельные буквы, цифры и знаки препинания, так и комбинации символов. Символы группируются по расположению зон на клавиатуре.

Уровень сложности – это объект, представляющий собой набор параметров, определяющих сложность упражнения в клавиатурном тренажере. Он устанавливает определенные ограничения и требования к упражнению, делая обучение более сложным или более простым.

Все объекты будут храниться в БД.

Администратор – это человек, отвечающий за управление системой тренажера. Он обладает широкими полномочиями, которые позволяют ему добавлять и редактировать упражнения, создавать новых пользователей, изменять их права доступа, а также анализировать статистику и отслеживать прогресс обучения всех пользователей.

Обучаемый – это человек, который использует тренажер для тренировки своих навыков печати. Он выбирает упражнения, отслеживает свой прогресс, анализирует статистику и стремится повысить свою скорость и точность печати.

1.4 Постановка задачи

Во время курсового проектирования необходимо разработать клавиатурный тренажер для тренировки слепой печати, с помощью которого администратор сможет создавать упражнения, настраивать различные уровни сложности, а обучающийся сможет выполнять эти упражнения. Система должна быть реализована в виде веб-приложения.

В системе должно быть реализовано две роли пользователей: администратор и обучающийся, поэтому для того, чтобы работать с системой, при первом входе в систему обучающиеся должны пройти процедуру регистрации: ввести логин и пароль (длина логина должна быть от 4 до 10 символов, длина пароля – от 4 до 10 символов). При повторном входе, любой пользователь должен авторизоваться в системе и затем, после аутентификации, система должна настроить интерфейс пользователя на заданную роль.

1.4.1 Режим администратора

Администратор обладает широким спектром полномочий, позволяющим ему эффективно управлять функционированием системы. Он сможет добавлять пользователей и выполнять их блокировку.

Он сможет добавлять и настраивать уровни сложности упражнений, в системе должно быть реализовано от 3 до 12 уровней сложности. При настройке уровня сложности администратор должен задать минимальную (от 20 символов) и максимальную (до 80 символов) длину упражнения, выбрать зоны клавиатуры, которые будут задействованы в упражнении (9 зон), а также устанавливать время ограничения на нажатие клавиш (от 0,5 до 1,5 секунды), максимальное количество ошибок (10% от длины текста).

Кроме того, администратор будет иметь возможность создавать и редактировать упражнения: предварительно выбрать уровень сложности, задавать название (от 1 до 10 символов), способ создания упражнения (ручной, автоматический) и задать длину упражнения (для автоматического создания). В ручном режиме администратор должен сам ввести текст упражнения.

Система будет проверять, правильно ли составляет упражнение администратор в соответствие с теми параметрами, которые он выбрал. При ручном создании упражнения система проверит символы зон клавиатуры, количество введенных символов.

Администратор сможет просматривать статистику по пользователям, упражнениям и по уровням сложности, это позволит ему проанализировать то, как обучаемые справляются с упражнениями различной степени сложности.

Все изменения, которые делает администратор сохраняются в БД. Все данные, необходимые для функционирования системы, должны храниться в БД, ее структура определяется на основании следующих сведений:

- о пользователе (логин, пароль, статус, статистика);
- об упражнении (название, уровень сложности, количество знаков, текст);
- об уровне сложности (название, минимальное количество знаков, максимальное количество знаков, время между нажатиями на клавишу, количество допустимых ошибок);

– статистика (пользователь, упражнение, дата выполнения, затраченное на выполнение время, количество ошибок, средняя скорость набора).

1.4.2 Режим обучаемого

Обучаемый после авторизации в системе сможет выполнить тестовое задание для определения уровня своей подготовки пользователя и выбора оптимального уровня сложности.

Обучаемый сможет выбирать доступные уровни сложности и выполнять соответствующие уровню сложности упражнения, которые формирует система.

При выполнении упражнения система должна контролировать правильность набора текста и время нажатия клавиш, в случае превышения допустимого количества ошибок или превышении времени нажатия клавиши система должна остановить выполнение упражнения. Все параметры набора должны сохраняться в статистике.

После завершения упражнения пользователь получает возможность ознакомиться с собственной статистикой в своем профиле. Система должна отображать статистику в удобном для обучаемого формате.

В системе должна быть реализована выдача справочной информации о возможностях системы.

Таким образом, в системе должны быть реализованы следующие функции:

1) функции системы:

- аутентификация пользователя в системе, настройка интерфейса пользователя на заданную роль;
- оценка выполнения тестового задания для определения уровня сложности для обучаемого;
- формирование списка упражнений по заданному уровню сложности;

- контроль длины упражнения и допустимых символов при ручном вводе;

- контроль введенных символов при выполнении упражнения;
- генерация текста упражнения по заданным настройкам;
- генерация длины упражнения для автоматического создания;
- визуализация процесса выполнения упражнения;
- визуализация статистики по заданному критерию;
- контроль выполнения упражнения и сообщение о результатах;
- сбор статистики и ее сохранение;
- выдача справочной информации о системе;

2) функции администратора:

- авторизация пользователя в системе (ввод логина и пароля);
- создание нового уровня сложности;
- настройка уровня сложности:
 - 1) название (номер);
 - 2) задание выбора зон клавиатуры;
 - 3) задание минимальной длины упражнения;
 - 4) задание максимальной длины упражнения;
 - 5) задание допустимого времени нажатия на клавишу;
 - 6) задание максимально допустимое количество ошибок;
- создание/редактирование упражнения:
 - 1) выбор уровня сложности;
 - 2) выбор способа создания упражнения;
 - 3) ввод текста;
 - 4) задание длины текста упражнения для автоматической генерации;
- сохранение упражнения в БД;
- загрузка упражнения из БД;
- работа со статистикой:

- 1) просмотр статистики по пользователям;
- 2) просмотр статистики по упражнениям;
- 3) просмотр статистики по уровням;
- работа с пользователями:
 - 1) добавление пользователей;
 - 2) блокировка пользователей;
- просмотр справочной информации;
- 3) функции игрока:
 - регистрация пользователя в системе (ввод логина, пароля);
 - авторизация пользователя в системе (ввод логина, пароля);
 - выполнение тестового задания;
 - выбор уровня сложности упражнения;
 - выбор упражнения из списка;
 - просмотр статистики:
 - 1) собственной статистики;
 - 2) средней скорости набора;
 - просмотр справочной информации.

2 Проектирование системы

На данном этапе происходит выбор архитектуры, система по функциональному признаку разделяется на основные подсистемы, между ними указываются информационные связи и/или связи по управлению, описывается основное назначение подсистем.

2.1 Выбор и обоснование архитектуры системы

Архитектура программного обеспечения (ПО) включает в себе ряд важных решений об организации программной системы, среди которых выбор структурных элементов и их интерфейсов, составляющих и объединяющих систему в единое целое; поведение, обеспечиваемое совместной работой этих элементов; организацию этих структурных и поведенческих элементов в более крупные подсистемы, а также архитектурный стиль, которого придерживается данная организация. Выбор архитектуры ПО также касается функциональности, удобства использования, устойчивости, производительности, повторного использования, понятности, экономических и технологических ограничений, эстетического восприятия и поиска компромиссов [16].

2.1.1 Виды архитектур

Архитектура – это основа любого проекта. Могут применяться различные виды архитектур в зависимости от поставленной задачи, размеров проекта и его специфики. Рассмотрим основные виды архитектур.

Централизованная архитектура

При использовании этой технологии база данных, система управления базой данных (СУБД) и прикладная программа располагаются на одном компьютере [17]. Для такого способа организации не требуется поддержки сети и все сводится к автономной работе. Все приложения, работающие в такой архитектуре, полностью находятся в основной памяти хост-ЭВМ.

Терминалы являются лишь устройствами ввода-вывода и таким образом в минимальной степени поддерживают интерфейс пользователя. Структура централизованной архитектуры представлена на рисунке 11.

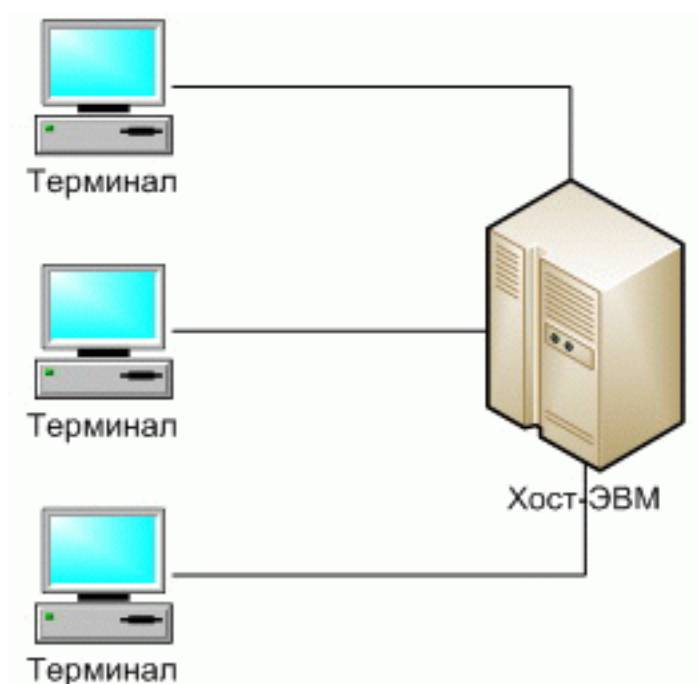


Рисунок 11 – Структура централизованной архитектуры

Достоинствами данной архитектурой являются:

- централизация ресурсов и оборудования облегчает обслуживание и эксплуатацию вычислительной системы;
- отсутствует необходимость администрирования рабочих мест пользователей.

Недостатками данной архитектуры являются:

- пользователя полностью зависит от администратора хост-ЭВМ;
- пользователь не может настроить рабочую среду под свои потребности – все используемое программное обеспечение является коллективным.

Архитектура «файл-сервер»

Эта архитектура баз данных с сетевым доступом предполагает назначение одного из компьютеров сети в качестве выделенного сервера, на котором будут храниться файлы базы данных [17]. В соответствии с

запросами пользователей файлы с файл-сервера передаются на рабочие станции пользователей, где и осуществляется основная часть обработки данных. Центральный сервер выполняет в основном только роль хранилища файлов, не участвуя в обработке самих данных. Структура архитектуры «файл-сервер» представлена на рисунке 12.

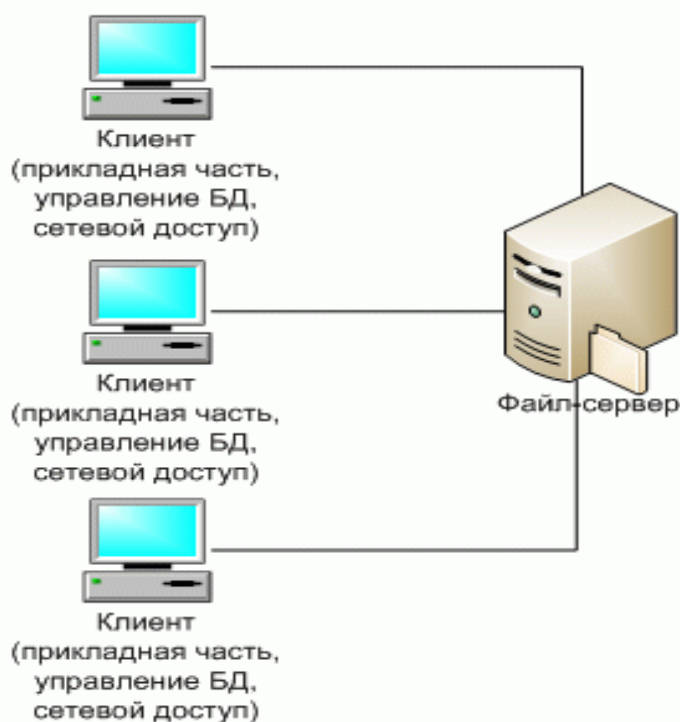


Рисунок 12 – Архитектура «файл-сервер»

Достоинствами данной архитектурой являются:

- многопользовательский режим работы с данными;
- высокая скорость и простота разработки.

Недостатками данной архитектуры являются:

- проблемы многопользовательской работы с данными: последовательный доступ, отсутствие гарантии целостности;
- низкая производительность, ненадежность системы.

Микросервисная архитектура

Микросервисная архитектура – это подход к созданию приложения, подразумевающий отказ от единой, монолитной структуры. То есть вместо того, чтобы исполнять все ограниченные контексты приложения на сервере с

помощью внутрипроцессных взаимодействий, мы используем несколько небольших приложений, каждое из которых соответствует какому-то ограниченному контексту. Причём эти приложения работают на разных серверах и взаимодействуют друг с другом по сети, например посредством Hyper Text Transfer Protocol (HTTP) [18]. Пример микросервисной архитектуры приведен на рисунке 13.

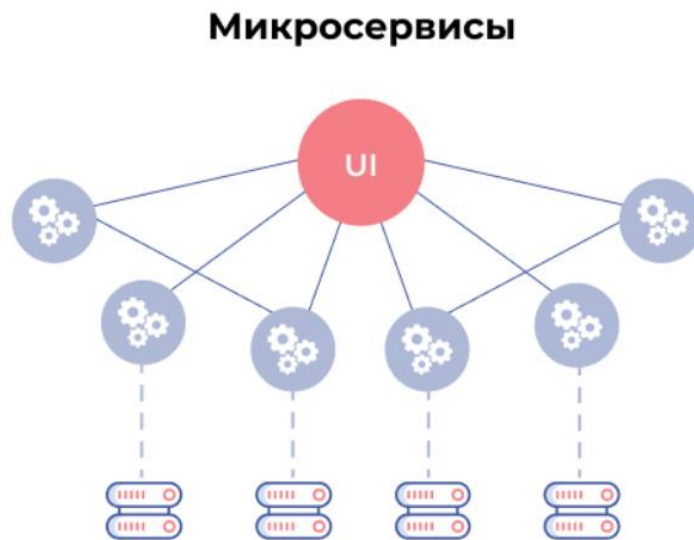


Рисунок 13 – Пример микросервисной архитектуры

Микросервисы можно легко развертывать, обновлять и масштабировать, причем делая это независимо друг от друга. Они обычно представляют собой небольшие автономные единицы, которые используют протоколы связи. Обычно микросервисы разрабатываются независимо друг от друга, что обеспечивает гибкость и масштабируемость.

Преимущества микросервисной архитектуры:

- микросервисы можно легко масштабировать, что позволяет легко обрабатывать увеличивающуюся нагрузку на приложение;
- независимое развертывание и модульность приложения упрощает внесение изменений и добавление нового функционала;
- если один микросервис выходит из строя, другие микросервисы могут продолжать работать, а это в свою очередь повышает надежность приложения;

- микросервисы проще обслуживать и обновлять по отдельности;
- микросервисная архитектура позволяет командам разработчиков работать независимо друг от друга, что ускоряет разработку приложения.

Недостатки микросервисной архитектуры:

- разработка и управление распределенной системой микросервисов может быть сложнее, чем при использовании монолитной архитектуры;
- взаимодействие между микросервисами может привести к снижению производительности по сравнению с монолитными приложениями;
- обмен сообщениями между микросервисами может добавить дополнительную задержку и сложность.

2.1.2 Клиент-серверные архитектуры

Клиент-серверная архитектура представляет собой модель проектирования программного обеспечения, которая разделяет приложение на две основные части: клиент и сервер. Клиент запрашивает информацию или услуги у сервера, а сервер обрабатывает запросы и отправляет ответ клиенту [19]. Клиент серверные приложения состоят из двух компонент – клиента и сервера. Пример клиент-серверной архитектуры приведен на рисунке 14.

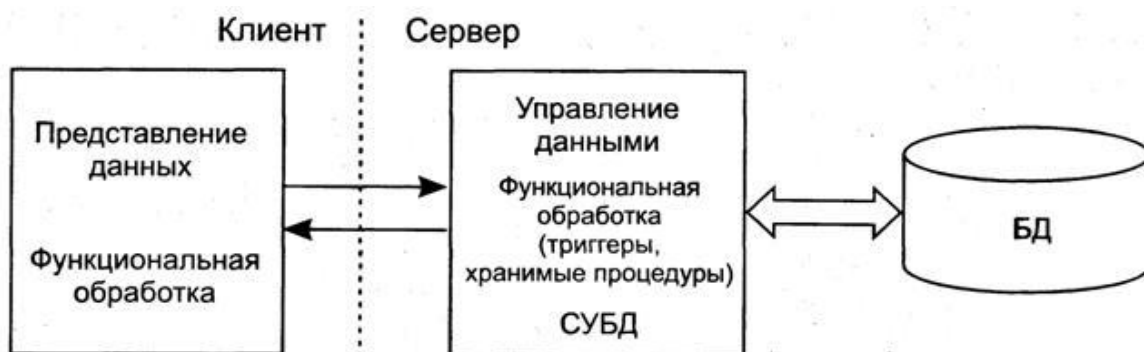


Рисунок 14 – Пример клиент-серверной архитектуры

В качестве клиента выступает компьютер, мобильное устройство или другое устройство, которое может делать запросы к серверу. Клиент обычно отвечает за отображение пользовательского интерфейса, сбор данных ввода.

В качестве сервера выступает часть, которая хранит, управляет данными и бизнес-логикой приложения. Кроме того, на сервере происходит обработка запросов от клиента и отправка ответов.

Преимущества клиент-серверной архитектуры:

- все данные и бизнес-логика хранятся и управляются на сервере, что упрощает обслуживание и обеспечивает безопасность приложения;
- сервер можно масштабировать для обработки увеличивающейся нагрузки путем добавления дополнительных мощностей;
- клиенты могут получить доступ к приложению с любого устройства, которое подключено к сети интернет, это в свою очередь обеспечивает высокую доступность приложения;
- приложение можно легко обновлять и обслуживать на сервере, без необходимости обновлять каждый клиент;
- централизованное хранение данных на сервере упрощает реализацию мер безопасности и предотвращение несанкционированного доступа к данным.

Недостатки клиент-серверной архитектуры:

- клиенты зависят от надежного сетевого подключения для доступа к приложению;
- если сеть медленная или перегруженная, то это может повлиять на производительность приложения;
- разработка и развертывание клиент-серверных приложений может быть более сложной процедурой, чем использование одноуровневых приложений.

Самыми используемыми являются следующие типы клиент-серверных архитектур [20]:

Одноуровневая архитектура «клиент-сервер» (1-Tier) – такая, где все прикладные программы рассредоточены по рабочим станциям, которые обращаются к общему серверу баз данных или к общему файловому серверу. Никаких прикладных программ сервер при этом не исполняет, только

предоставляет данные. Пример одноуровневой архитектуры приведен на рисунке 15.

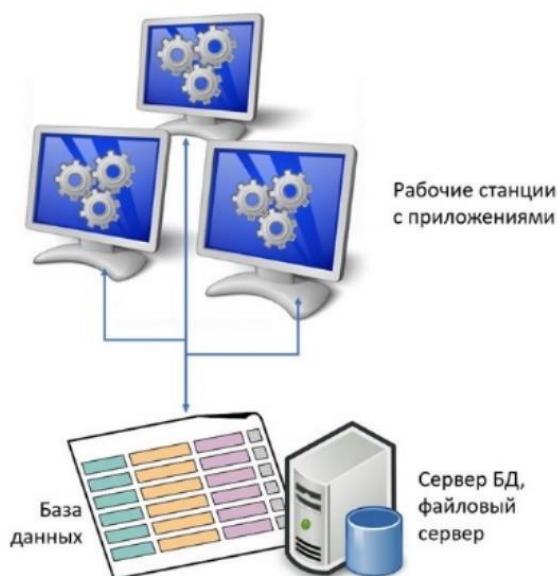


Рисунок 15 – Одноуровневая архитектура

К двухуровневой архитектуре «клиент-сервер» следует относить такую, в которой прикладные программы сосредоточены на сервере приложений (Application Server), например, сервере 1С или сервере CRM, а в рабочих станциях находятся программы-клиенты, которые предоставляют для пользователей интерфейс для работы с приложениями на общем сервере. Пример двухуровневой архитектуры приведен на рисунке 16.



Рисунок 16 – Двухуровневая архитектура

В трёхуровневой архитектуре сервер баз данных, файловый сервер и другие представляют собой отдельный уровень, результаты работы которого

использует сервер приложений. Логика данных и бизнес-логика находятся в сервере приложений. Все обращения клиентов к базе данных происходят через промежуточное программное обеспечение (middleware), которое находится на сервере приложений. Вследствие этого, повышается гибкость работы и производительность. Пример трехуровневой архитектуры приведен на рисунке 17.



Рисунок 17 – Трёхуровневая архитектура

В отдельный класс архитектуры «клиент-сервер» можно вынести многоуровневую архитектуру, в которой несколько серверов приложений используют результаты работы друг друга, а также данные от различных серверов баз данных, файловых серверов и других видов серверов. По сути, предыдущий вариант, трёхуровневая архитектура – не более, чем частный случай многоуровневой архитектуры. Пример многоуровневой архитектуры приведен на рисунке 18.

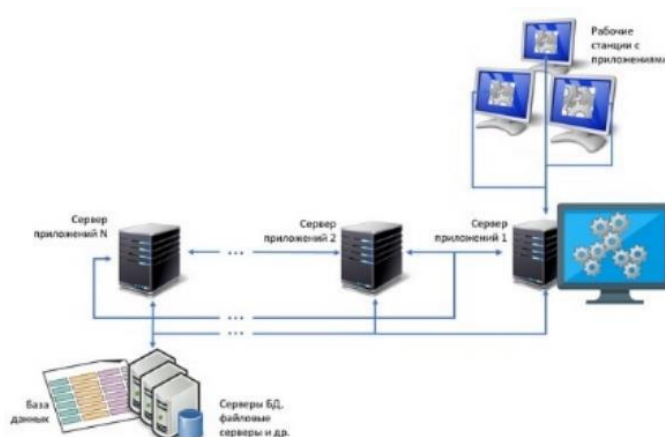


Рисунок 18 – Многоуровневая архитектура

2.1.3 Протоколы обмена данными

Сетевой протокол – это набор правил, определяющий принципы взаимодействия устройств в сети. Чтобы отправка и получение информации прошли успешно, все устройства-участники процесса должны принимать условия протокола и следовать им. В сети их поддержка встраивается или в аппаратную часть (в «железо»), или в программную часть (в код системы), или и туда, и туда [21].

Есть много различных протоколов, которые могут использоваться в клиент-серверных приложениях, в том числе [22]:

- HTTP: широко используемый протокол для передачи данных в сети, в особенности для веб-приложений;
- Hyper Text Transfer Protocol Secure (HTTPS): защищенная версия HTTP, которая использует шифрование для защиты данных при передаче;
- File Transfer Protocol (FTP): протокол, используемый для передачи файлов между двумя компьютерами;
- Secure Shell (SSH): защищенный протокол, используемый для удаленного подключения к компьютеру и выполнения команд.

Протокол HTTP предполагает использование клиент-серверной структуры передачи данных. Клиентское приложение формирует запрос и отправляет его на сервер, после чего серверное программное обеспечение обрабатывает данный запрос, формирует ответ и передает его обратно клиенту. После этого клиентское приложение может продолжить отправлять другие запросы, которые будут обработаны аналогичным образом [23].

Задача, которая традиционно решается с помощью протокола HTTP, заключается в обмене данными между пользовательским приложением, осуществляющим доступ к веб-ресурсам (обычно это веб-браузер) и веб-сервером. На данный момент именно благодаря протоколу HTTP обеспечивается работа Всемирной паутины.

Application programming interface (API) многих программных продуктов также подразумевает использование HTTP для передачи данных — сами данные при этом могут иметь любой формат, например, eXtensible Markup Language (XML) или JavaScript Object Notation (JSON).

Как правило, передача данных по протоколу HTTP осуществляется через Transmission Control Protocol и Internet Protocol (TCP/IP) соединения. Серверное программное обеспечение при этом обычно использует TCP-порт 80 (и, если порт не указан явно, то обычно клиентское программное обеспечение по умолчанию использует именно 80-й порт для открываемых HTTP-соединений), хотя может использовать и любой другой.

2.1.4 Типы клиентов

«Толстый» клиент представляет собой клиентское приложение, которое содержит значительную часть логики и функциональности непосредственно на стороне пользователя. Этот тип клиента активно взаимодействует с сервером, но при этом обладает значительной автономностью [24]. Пример технологии обработки данных с использованием «толстого» клиента приведен на рисунке 19.

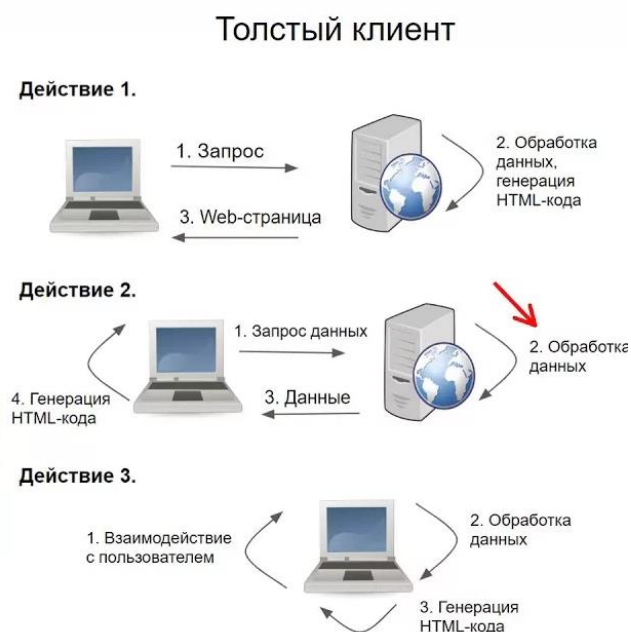


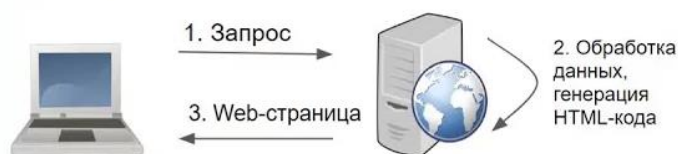
Рисунок 19 – Пример «толстого» клиента

«Толстый» клиент загружает все необходимые ресурсы (включая интерфейс и логику) на устройство пользователя при запуске приложения. Затем большая часть обработки данных и логики приложения выполняется непосредственно на клиентской стороне. Примерами «толстых» клиентов могут служить настольные приложения, написанные на языках программирования типа Java, Python или C++.

«Тонкий» клиент – это клиентское приложение, которое минимизирует логику и функциональность на стороне пользователя, делегируя большинство задач серверу. Он зависит от сервера для предоставления большей части функциональности [24]. Пример технологии обработки данных с использованием «тонкого» клиента приведен на рисунке 20.

Тонкий клиент

Действие 1.



Действие 2.



Рисунок 20 – Пример тонкого клиента

«Тонкий» клиент требует минимальной установки на устройстве пользователя. Он осуществляет связь с сервером, который выполняет основные вычисления и предоставляет пользователю всю необходимую информацию. Примерами «тонких» клиентов могут служить веб-приложения и мобильные приложения.

В таблице 2 приведено краткое сравнение преимуществ и недостатков двух технологий обработки данных.

Таблица 2 – Сравнение характеристик технологий обработки данных

Технология Характеристика	Толстый клиент	Тонкий клиент
Обработка данных	На устройстве пользователя	На сервере
Хранение данных	На устройстве пользователя	На сервере
Производительность	Высокая	Более низкая
Работа в автономном режиме	Да	Нет
Безопасность	Более высокая	Менее безопасен
Нагрузка на устройство	Высокая	Низкая
Обслуживание и обновление	Сложное	Простое
Совместимость	Может быть несовместим	Совместим с разными устройствами

2.1.5 Выводы

Клавиатурный тренажер будет реализован в виде веб-приложения.

Веб-приложение – клиент-серверное приложение, в котором клиент взаимодействует с веб-сервером при помощи браузера. Логика веб-приложения распределена между сервером и клиентом, хранение данных осуществляется преимущественно на сервере, обмен информацией происходит по сети. Одним из преимуществ такого подхода является тот факт, что клиенты не зависят от конкретной операционной системы пользователя, поэтому веб-приложения являются межплатформенными службами [25].

В качестве архитектуры будет выбрана двухуровневая клиент-серверная архитектура, потому что она легко масштабируема, пользователю обычно удобно работать в такой среде, а также легко конфигурировать и модифицировать такое приложение. В приложении будет использован «тонкий» клиент, потому что основные вычисления и хранение данных осуществляется именно на сервере, а задача клиента заключается в сборе

данных, чтобы отправить их серверу и их отображении. Клиент и сервер будут обмениваться данными через протокол HTTP стека TCP/IP.

2.2 Структурная схема системы

Система (греч. «составленное из частей», «соединение» от «соединяю») – множество элементов, находящихся в отношениях и связях друг с другом, которое образует определённую целостность, единство [26].

Как следует из определения, отличительным (главным свойством) системы является её целостность: комплекс объектов, рассматриваемых в качестве системы, должен обладать общими свойствами и поведением.

Очевидно, необходимо рассматривать и связи системы с внешней средой. В самом общем случае понятие «система» характеризуется [27]:

- наличием множества элементов;
- наличием связей между ними;
- целостным характером данного устройства или процесса.

Структурная схема – это схема, определяющая основные функциональные части изделия, их назначение и взаимосвязи [28].

Сущность структурного подхода к разработке системы заключается в её декомпозиции (разбиении) на автоматизируемые функции: система разбивается на функциональные подсистемы, которые в свою очередь делятся на подфункции, подразделяемые на задачи и так далее. Процесс разбиения продолжается вплоть до конкретных процедур. При этом автоматизируемая система сохраняет целостное представление, в котором все составляющие компоненты взаимосвязаны [29].

На рисунке 21 приведена структурная схема разрабатываемой системы, разделяется клиентскую и серверную часть. Взаимодействие между ними осуществляется по протоколу HTTP стека TCP/IP.

В состав клиентской части входит:

- 1) подсистема «Администратор», которая включает в себя:

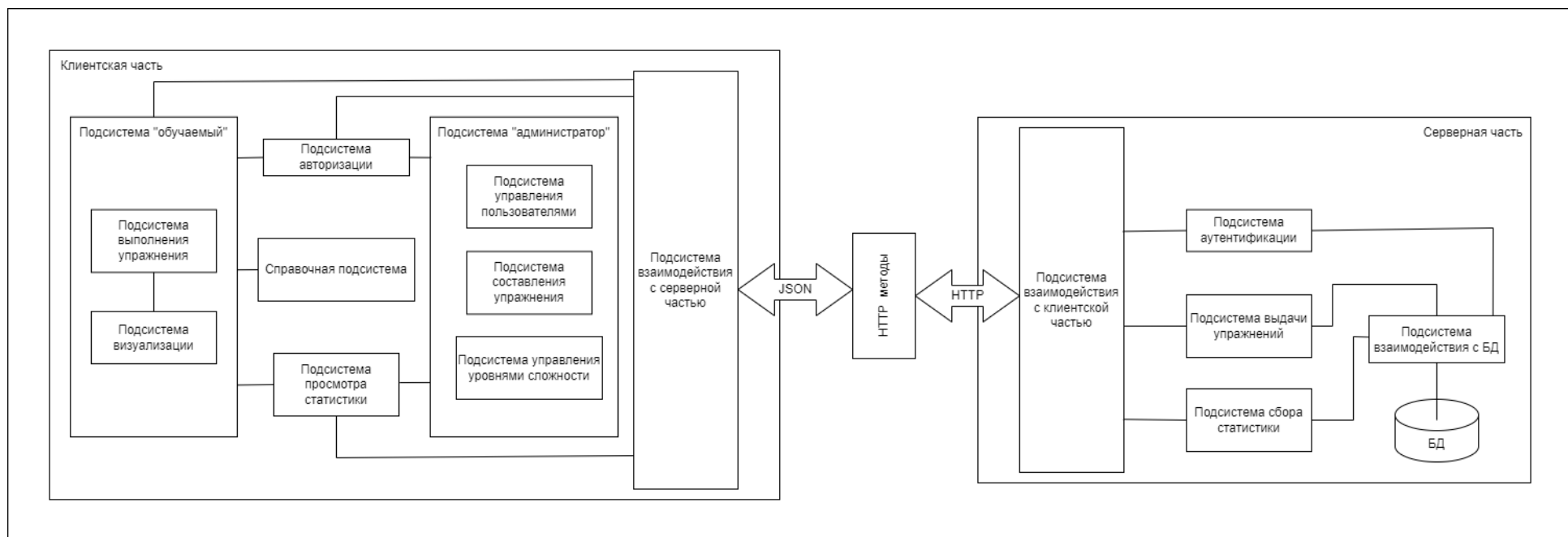


Рисунок 21 – Структурная схема системы

- подсистему управления пользователями, которая обеспечивает добавление, блокировку и редактирование пользователей;

- подсистему составления упражнения, которая отвечает за создание, редактирование и сохранение упражнений;

- подсистема управления уровнями сложности обеспечивает создание, изменение, удаление уровней сложности;

2) подсистема «Обучаемый», которая включает в себя:

- подсистему выполнения упражнения отвечает за запись ввода, проверку правильности выполнения упражнения;

- подсистему визуализации отвечает за показ текста упражнения, показ результата и визуализацию процесса выполнения упражнения;

3) подсистема авторизации проверяет корректность ввода данных, валидирует логин и пароль, и подтверждает доступ пользователя к системе;

4) подсистема просмотра статистики отображает собранную статистику в удобном для пользователя виде;

5) справочная подсистема предоставляет пользователям информацию о функциях и возможностях веб-приложения;

6) подсистема взаимодействия с сервером, которая осуществляет установку соединения с сервером, формирование и отправку запросов.

В состав серверной части входит:

1) подсистема взаимодействия с БД, обеспечивающая доступ ко всем другим подсистемам, которые нуждаются в работе с базой данных;

2) подсистема аутентификации проверяет корректность ввода данных от клиента, валидирует логин и пароль, и подтверждает доступ пользователя к системе;

3) подсистема выдачи упражнения отвечает за выбор или генерацию и отправку упражнения;

4) подсистема сбора статистики собирает, обрабатывает и хранит статистику о выполнении упражнений;

5) подсистема взаимодействия с базой данных, которая представляет собой СУБД, позволяющую управлять БД;

6) база данных, которая осуществляет хранение данных о выполненных обучаемым упражнениях.

2.3 Разработка спецификации требований

Разработка спецификации программного обеспечения является одним из фундаментальных процессов технологии разработки ПО. Этот процесс анализа, формирования, документирования и проверки функциональных возможностей и ограничений системы называется «разработка требований» (спецификация требований). Он является критическим этапом в создании всех видов программных систем, что обусловлено тем, что ошибки, допущенные на этой стадии, ведут к возникновению серьёзных проблем на этапах проектирования и разработки [30].

Требования – это свойства, которыми должно обладать ПО для адекватного определения функций, условий и ограничений выполнения ПО, а также объёмов данных, технического обеспечения и среды функционирования [31].

2.3.1 Функциональная спецификация

Функциональные требования задают «что» система должна делать; нефункциональные – с соблюдением «каких условий» (например, скорость отклика при выполнении заданной операции). При разработке этих требований в первую очередь необходимо учитывать потребности пользователя (заказчика). Пользовательские требования (User Requirements) – описывают цели/задачи пользователей системы, которые должны достигаться/выполняться пользователями при помощи создаваемой программной системы. Часто пользовательские требования представляют в виде сценариев (вариантов использования) Use Case [32]. Функциональная спецификация системы приведена в таблице 3.

2.3.2 Перечень исключительных ситуаций

Исключительная ситуация – это ситуация, при которой система не может выполнить возложенных на нее функций или которая может привести к денормализации работы системы.

В таблице 4 приведен перечень исключительных ситуаций для разрабатываемой системы и описаны реакции системы на их возникновение.

Таблица 3 – Перечень функций, выполняемых системой

Название подсистемы	Название функции	Информационная среда			
		Входные данные		Выходные данные	
		Назначение (наименование)	Тип, ограничения	Назначение (наименование)	Тип, ограничения
1	2	3	4	5	6
Справочная	Выдать сведения о разработчиках	Сведения о разработчиках системы (ФИО, номер группы)	Текст	Визуальное отображение информации	–
	Выдать сведения о системе	Файл справки	Текстовый (*.HTML)		
				Код ошибки	Целое
Управления уровнями сложности	Сохранить уровень сложности	Указание названия уровня сложности	Строка	Уровень сложности	Объект «Уровень сложности»
		Зоны клавиатуры	Массив целых чисел		
		Минимальная длина упражнения	Целое		
		Максимальная длина упражнения	Целое		
		Максимально допустимое количество ошибок	Целое		

Продолжение таблицы 3

1	2	3	4	5	6
Управления уровнями сложности		Допустимое время нажатия на клавиатуру	Вещественное	Код ошибки	Целое
	Выбрать уровень сложности	Название уровня сложности	Строка	Уровень сложности	Строка
	Задать максимальное количество знаков	Допустимая длина	Целое	Максимальная длина упражнения	Целое
	Задать минимальное количество знаков	Допустимая длина	Целое	Минимальная длина упражнения	Целое
	Задать количество ошибок	Допустимое количество ошибок	Целое	Максимальное количество ошибок	Целое
	Выбрать зоны клавиатуры	Список доступных зон	Массив целых чисел	Список активированных зон	Массив целых чисел
	Задать время нажатия на клавиатуру	Допустимое время	Вещественное	Время	Вещественное

Продолжение таблицы 3

1	2	3	4	5	6
Составление упражнения	Сохранить упражнения	Название уровня сложности	Строка	Упражнение	Объект «Упражнение»
		Текст упражнения	Строка	Код ошибки	Целое
	Задать длину упражнения	Допустимая длина	Целое	Длина	Целое
	Сформировать список упражнений	Название уровня сложности	Строка	Список упражнений	Массив объектов «Упражнение»
	Ввести текст упражнения	Набор допустимых символов	Массив символов	Текст упражнения	Строка
		Допустимая длина	Целое		
	Сгенерировать текст	Набор допустимых символов	Массив символов	Текст упражнения	Строка
		Допустимая длина	Целое		

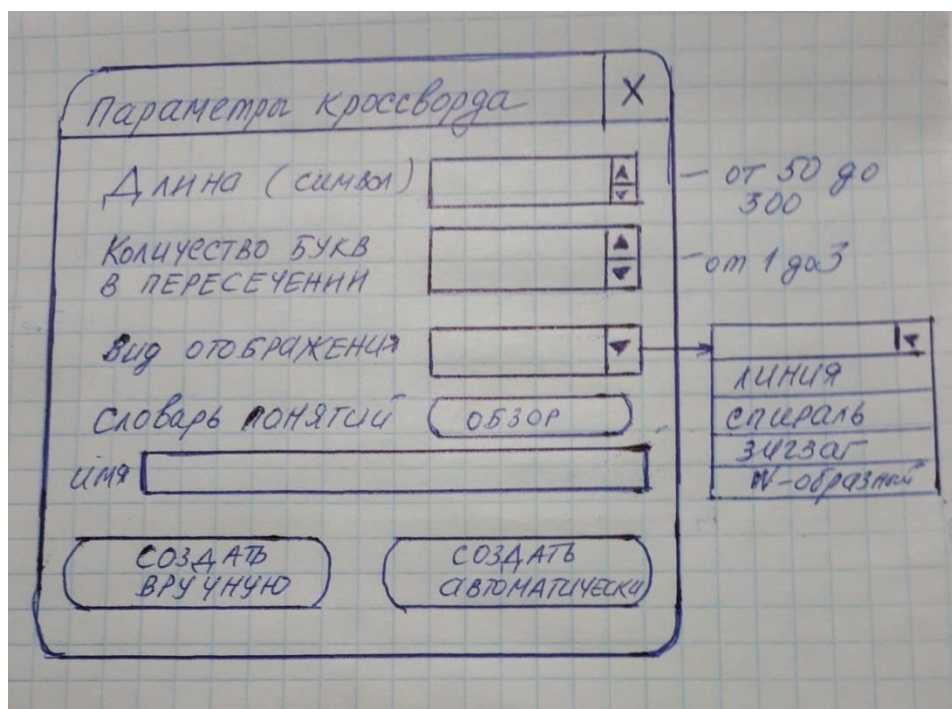
Таблица 4 – Перечень исключительных ситуаций

Название подсистемы	Название исключительной ситуации	Реакция системы
1 Справочная	1.1 Не возможно открыть файл справки	Выдача сообщения «Файл справки поврежден»
	1.2 Не возможно найти файл справки	Выдача сообщения «Отсутствует файл справки»
2 Файловая	2.1 Попытка открытия файла с несобственным форматом	Выдача сообщения «Файл поврежден или недопустимого формата»
...

2.4 Разработка прототипа интерфейса пользователя системы

Дать определение интерфейса, отметить основные особенности разработки интерфейса.

Здесь должны быть разработаны прототипы **всех** основных форм приложения с описанием привязанной к ней функциональности, например:



Пример.

На рисунке xxx приведен прототип экранной формы начальной настройки приложения. Здесь пользователь должен выбрать язык программирования, на котором написан алгоритм, категорию (поиск или сортировка) и нажать кнопку «Далее» для перехода к следующему экрану (форме).

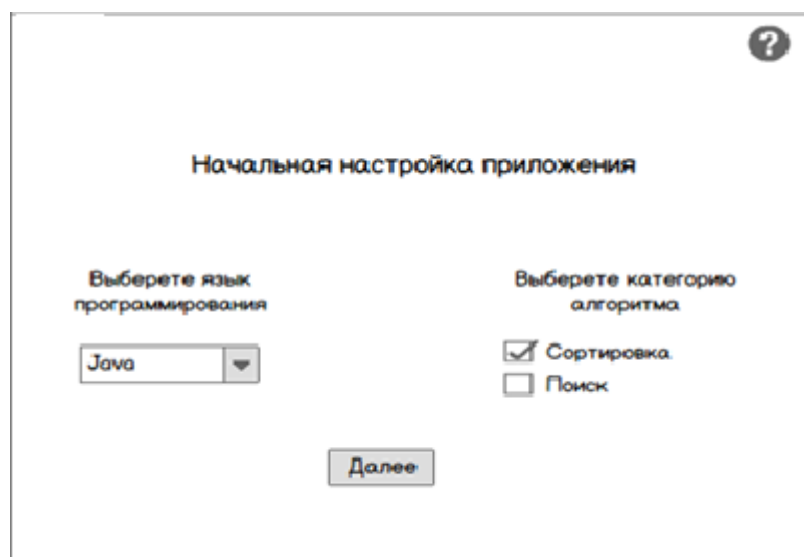


Рисунок xxx – Прототип экранной формы начальной настройки приложения

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На Рисунок **XXX** приведена навигационная модель разрабатываемого приложения.

2.5 Разработка информационно-логического проекта системы

Вводные слова Вводные слова Вводные слова Вводные слова Вводные
слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные
слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные
слова.

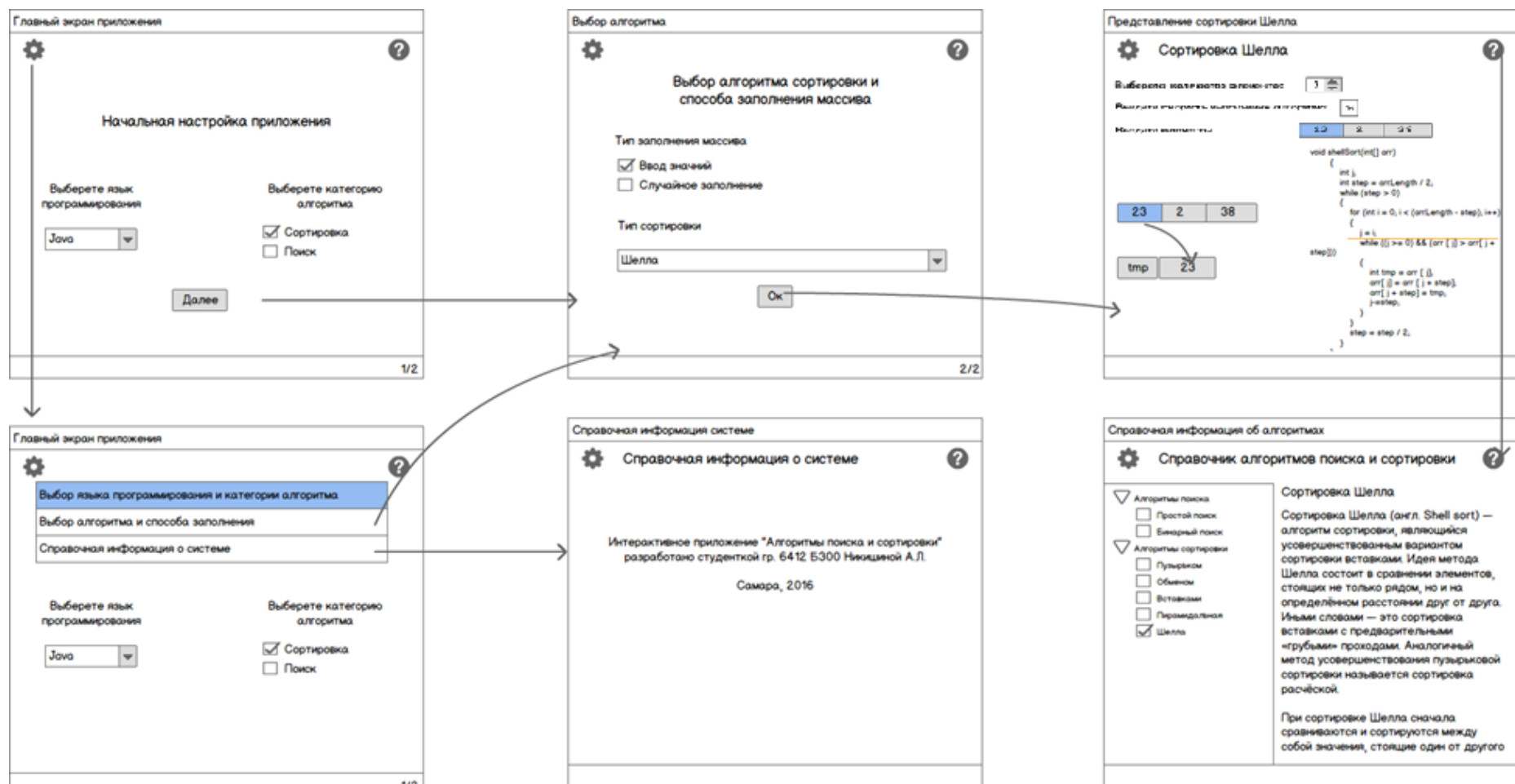


Рисунок XXX – Навигационная модель приложения

2.5.3 ЯЗЫК UML

Для специфирования (построения точных, недвусмысленных и полных моделей) системы и ее документирования используется унифицированный язык моделирования UML.

Вводные слова Вводные слова Вводные слова Вводные слова Вводные
слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные
слова Вводные слова Вводные слова Вводные слова Вводные слова Вводные
слова.

2.5.4 Диаграмма вариантов использования

Диаграмма вариантов использования представляет собой наиболее общую концептуальную модель сложной системы, которая является исходной для построения всех остальных диаграмм. На ней изображаются отношения между актерами и вариантами использования.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке XXX приведена диаграмма вариантов использования (пользователя). Здесь должно быть описание диаграммы.

2.5.5 Сценарии

Сценарий (scenario) - определенная последовательность действий, которая описывает действия актеров и поведение моделируемой системы в форме обычного текста [27].

В контексте языка UML сценарий используется для дополнительной иллюстрации взаимодействия актеров и вариантов использования.

Рассмотрим несколько сценариев.

Сценарии определяются преподавателем.

Рисунок XXX – Диаграмма вариантов использования системы

2.5.6 Диаграмма классов

Диаграммы классов – это наиболее часто используемый тип диаграмм, которые создаются при моделировании объектно-ориентированных систем, они показывают набор классов, интерфейсов и коопераций, а также их связи. На практике диаграммы классов применяют для моделирования статического представления системы, они служат основой для целой группы взаимосвязанных диаграмм – диаграмм компонентов и диаграмм размещения [XXX].

На рисунке XX приведена диаграмма классов системы (этап проектирования). В таблице XX приведено описание классов.

Таблица XX – Описание классов системы

Название класса	Назначение
1	2

2.5.7 Диаграмма состояний

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке XXX приведена диаграмма состояний системы. Здесь должно быть описание диаграммы (диаграмм).

2.5.8 Диаграмма деятельности

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке XXX приведена диаграмма деятельности системы. **Здесь должно быть описание диаграммы (диаграмм).**

2.5.9 Диаграмма последовательности

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке XXX приведена диаграммы последовательности системы для варианта использования «???». **Диаграммы построены на основании сценариев, приведенных в п.2.4.3.**

2.6 Логическая модель данных (при необходимости)

Логическая информационная модель – модель данных, в которой учитывается способ логического хранения данных в памяти ЭВМ. При построении модели базы данных (БД) используются следующие понятия.

Сущность – объект предметной области, который можно отличить от других понятий по некоторым признакам. Сущность состоит из множества своих экземпляров. Каждая сущность обладает свойствами – атрибутами [Ошибка! Источник ссылки не найден.].

Атрибут – определенное свойство сущности. Именно набор атрибутов, в общем случае уникальный для каждой сущности, позволяет выделить ее среди других объектов и назвать уникальным именем.

Атрибут или набор атрибутов, используемый для идентификации экземпляра сущности, называется ключом сущности. В случае если для идентификации экземпляра используется один атрибут, ключ называется простым; в противном случае ключ составной. Каждый экземпляр сущности однозначно определяется ключом [Ошибка! Источник ссылки не найден.Ошибка! Источник ссылки не найден.].

Логическая модель БД разрабатываемой системы приведена на рисунке XXX.

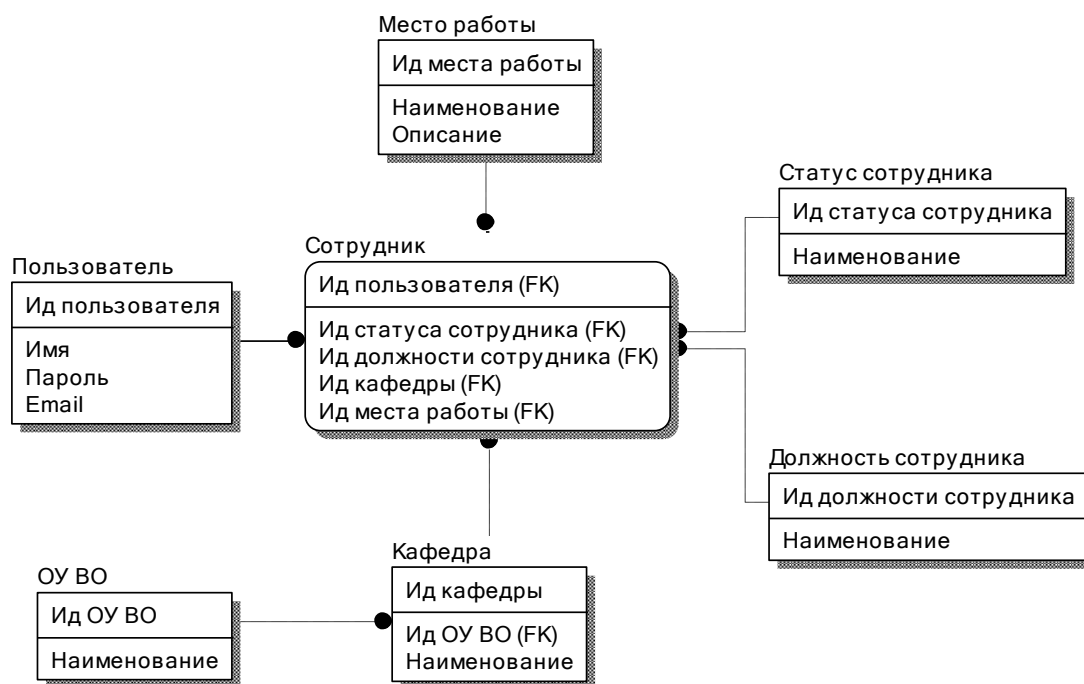


Рисунок 1 – Логическая модель данных

Описание объектов рассматриваемой предметной области, которые хранятся в базе данных, приведено в таблицах 2-???

Таблица 2 – Сущность «Пользователь»

Идентификатор	Тип данных	Описание
Ид пользователя	Целый	Уникальный идентификатор пользователя
Имя	Символьный[30]	Имя, используемое при идентификации пользователя и его взаимодействии с системой
Пароль	Символьный[10]	Пароль пользователя, преобразованный в закодированную строку
Email	Символьный[50]	Электронная почта, указанная пользователем при регистрации

2.7 Выбор и обоснование алгоритмов обработки данных /Разработка и описание алгоритмов обработки данных

Вводные слова про необходимость разработки алгоритмов.

На рисунке XXX приведена схема алгоритма обработки элементов массива. Здесь должно быть краткое описание алгоритма.

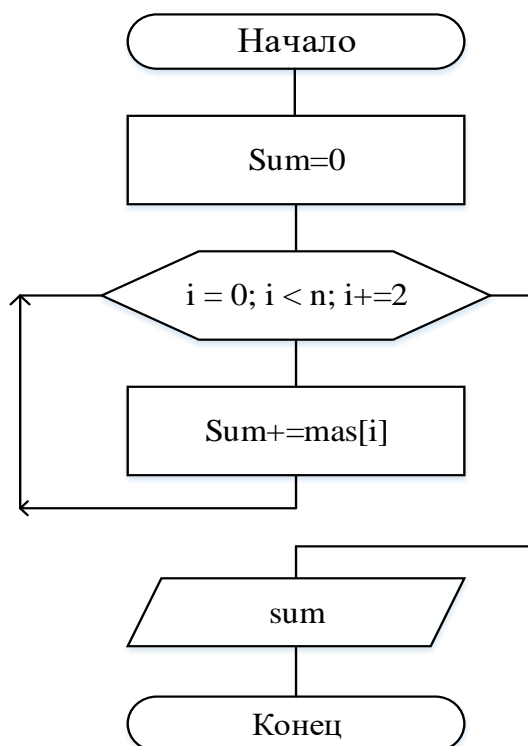


Рисунок XXX – Схема алгоритма обработки элементов массива

На рисунке XXXX приведена схема алгоритма вычисления исходного выражения.

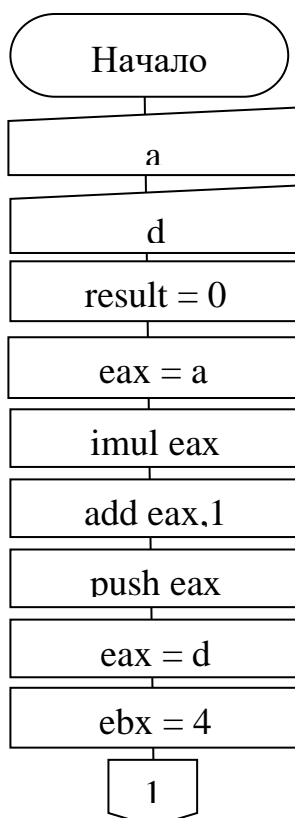


Рисунок XXX – Схема алгоритма вычисления исходного выражения (начало)

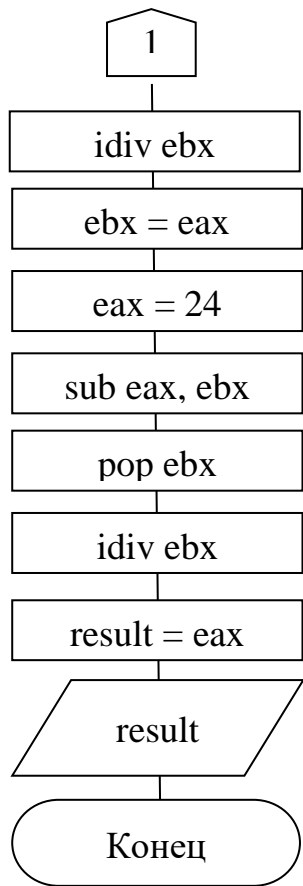


Рисунок XXXX– Схема алгоритма вычисления исходного выражения
(окончание)

2.8 Выбор и обоснование комплекса программных средств

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

1.1.1 Выбор языка программирования

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

1.1.2 Выбор среды программирования

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

2.8.10 Выбор операционной системы

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

2.8.11 Выбор системы управления базами данных (при необходимости)

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

3 Реализация системы

3.1 Разработка и описание интерфейса пользователя

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунках приведены примеры того, как нужно оформить сведения о разработчиках.

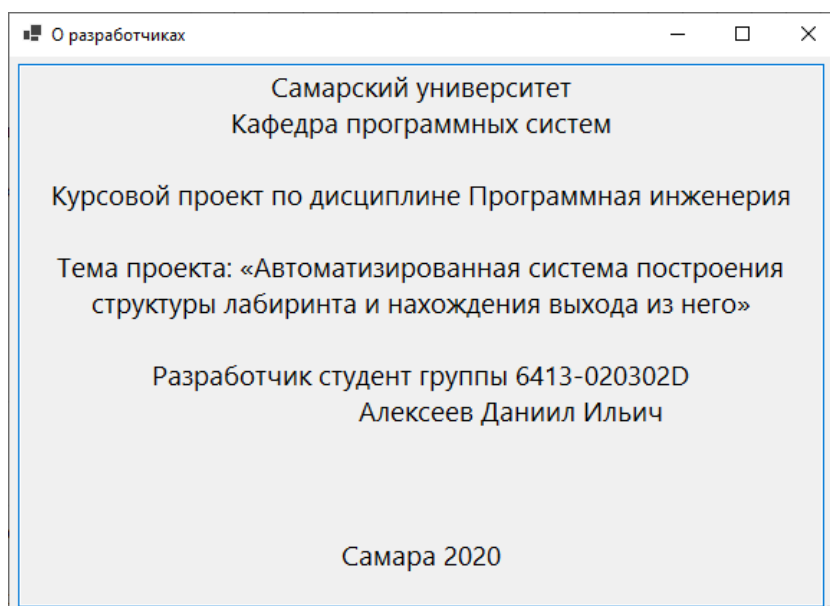


Рисунок XXX – Сведения о разработчиках

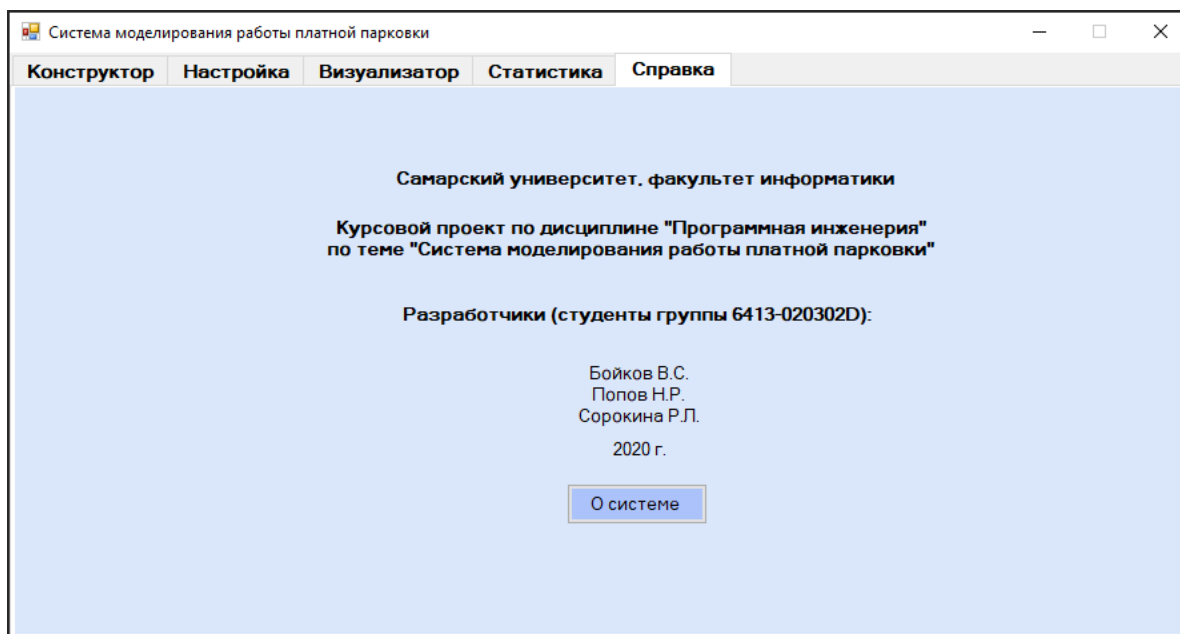


Рисунок XXX – Сведения о разработчиках

3.2 Диаграммы реализации

Диаграммы реализации предназначены для отображения состава компилируемых и выполняемых модулей системы, а также связей между ними. Диаграммы реализации разделяются на два конкретных вида: диаграммы компонентов (component diagrams) и диаграммы развертывания (deployment diagrams) [XXX].

3.2.1 Диаграмма компонентов

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке XXX приведена диаграмма компонентов, их описание приведено в таблице XXX.

Рисунок XXX – Диаграмма компонентов системы

Таблица XXX – Описание компонентов системы

Название компонента	Назначение компонента	Подсистема

3.2.2 Диаграмма развертывания

Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст
Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст Текст.

На рисунке XXX приведена диаграмма развертывания системы. **Здесь должно быть описание** тех компонентов, которые развернуты на узлах ЭВМ.

Рисунок XXX – Диаграмма развертывания системы

3.2.3 Диаграмма классов

В соответствии со спецификацией, приведенной в п. 2.5.6, и с учетом выбранного языка программирования (см. п. 2.8.1) разработана диаграмма классов системы (этап реализации), приведенная на рисунке XXX.

Рисунок XXX – Описание классов системы (этап реализации)

3.3 Физическая модель данных (при необходимости)

Физическое проектирование является последним этапом проектирования базы данных, при выполнении которого принимается решение о способах реализации разрабатываемой базы данных. Во время логического проектирования была определена логическая структура базы данных (которая описывает отношения и ограничения в рассматриваемой прикладной области).

Физическая модель базы данных содержит все детали, необходимые конкретной СУБД для создания базы: наименования таблиц и столбцов, типы данных, определения первичных и внешних ключей [??].

На рисунке **Ошибка! Источник ссылки не найден.** представлена физическая модель данных системы.

Рисунок XXX – Физическая модель данных системы

В таблицах ??-?? приведено описание сущностей БД. Первичные ключи выделены жирным шрифтом, а внешние – курсивом.

Таблица XXX – Сущность « User »

Имя поля	Имя атрибута	Тип	Размер (байт)
user Id	uniqueidentifier	int	4
Name	Имя пользователя	varchar(30)	30
Password	Пароль	varchar(10)	10
e-mail	Адрес электронной почты	varchar(50)	50

3.4 Выбор и обоснование комплекса технических средств

1.1.1 Расчет объема занимаемой памяти

Расчет объема внешней памяти

Для расчета необходимого объема свободной внешней памяти, необходимой для функционирования системы, воспользуемся следующей формулой:

$$V_{\text{ЖД}} = V_{\text{ОС}} + V_{\text{ПР}} + V_{\text{СПО}} + V_{\text{БД}} + V_{\text{справки}},$$

где $V_{\text{ОС}}$ – объем памяти, занимаемый операционной системой (операционная система Windows 7 Professional 64 бит с пакетом обновлений SP1, $V_{\text{ОС}} = 20$ Гб);

$V_{\text{ПР}}$ – объем памяти, занимаемый непосредственно файлами приложения ($V_{\text{ПР}} = 2$ Мб);

$V_{\text{СПО}}$ – объем памяти, занимаемый сопутствующим программным обеспечением (библиотеки cryptopp.dll, simplexlsx.dll, sqlite3.dll, sqlitcpp.dll, Qt Framework 5.11.1, Internet Explorer 9; дадим оценку сверху $V_{\text{СПО}}$ в 3 Гб);

$V_{\text{БД}}$ – объем памяти, занимаемый базой данных (всеми таблицами) при ее максимальном заполнении. Расчет этой составляющей приведен в таблице XXX ($V_{\text{БД}} = \text{???? байт} = \text{??? Кб} = \text{??? Мб} = \text{??? Гб}$).

$V_{\text{справки}}$ – объем памяти, необходимый для хранения файла справки ($V_{\text{справки}} = 0,8$ Мб).

Таким образом, суммарный объем внешней памяти составит:

$$V_{\text{ЖД}} = 20 \text{ Гб} + 2 \text{ Мб} + 3 \text{ Гб} + \text{??? Мб} + 1 \text{ Мб} \sim \text{??? Гб}.$$

Таблица 1 – Расчет объема внешней памяти, необходимой для хранения БД

Таблица	Размер записи (байт)	Максимум записей	Всего (байт)
Пользователь	94	10	940
Сотрудник		30	
Статус сотрудника		10	
Должность сотрудника		10	
Место работы		10	
Кафедра		10	
ОУ ВО		10	
Итого			

Расчет объема ОЗУ

Для расчета необходимого объема ОЗУ воспользуемся следующей формулой:

$$V_{\text{ОЗУ}} = V_{\text{ОС}} + V_{\text{ПР}} + V_{\text{БД}} + V_{\text{браузера}},$$

где $V_{\text{ОС}}$ – ОЗУ, занимаемое операционной системой (2 Гб);

$V_{\text{ПР}}$ – ОЗУ, которое займет само приложение (не превысит 80 Мб);

$V_{\text{БД}}$ – объем данных из базы, который может быть одновременно загружен в оперативную память (дадим ему оценку сверху в 10 Мб).

$V_{\text{браузера}}$ – ОЗУ, занимаемое браузером (оценим его сверху значением в 100 Мб).

Суммарные объемы ОЗУ составит:

$$V_{\text{ОЗУ}} = 2 \text{ Гб} + 80 \text{ Мб} + 10 \text{ Мб} + 100 \text{ Мб} \sim 2.2 \text{ Гб}.$$

Таким образом, 2.2 Гб оперативной памяти можно считать минимально необходимым для функционирования системы.

3.4.4 Минимальные требования, предъявляемые к системе

Для корректного функционирования системы необходимо:

- тип ЭВМ: x86-64 совместимый;
- объем ОЗУ – не менее 3 Гб;
- объем свободного дискового пространства – не менее ??? Гб;
- клавиатура или иное устройство ввода;
- мышь или иное манипулирующее устройство;
- процессор – Intel Pentium не менее 1,5 ГГц;
- дисплей с разрешением не менее 1024 × 768 пикселей;
- операционная система Windows 7 и выше;
- браузер Internet Explorer 9 и выше;
- Qt framework 5.11 и выше.

ЗАКЛЮЧЕНИЕ

В процессе выполнения курсового проекта была разработана автоматизированная система ..., позволяющая

В первом разделе приведены основные понятия предметной области, описаны характеристики систем-аналогов, приведен их сравнительный анализ. На основе проведенного анализа выполнена объектная декомпозиция, отраженная в диаграмме объектов, и сформулирована постановка задачи.

Во втором разделе ...

В третьем разделе ...

Разработанная система может использоваться ...

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Книги

Целиком

1 Буч Г., Рамбо Д., Якобсон А. Язык UML. Руководство пользователя. Изд. 2-е. М.: ДМК Пресс, 2006. 546 с.

Если нужно указать номера конкретных страниц

2 Буч Г., Рамбо Д., Якобсон А. Язык UML. Руководство пользователя. Изд. 2-е. М.: ДМК Пресс, 2006. С. 21.

Если повторная ссылка на тот же документ

3 Буч Г., Рамбо Д., Якобсон А. Язык UML ... С. 31.

Если больше 3 авторов

4 Нестационарная аэродинамика баллистического полета/ Липницкий Ю.М. и [др.]. М.: Физматлит, 2003. 176 с.

Журналы

5 Зеленко Л.С., Шумская Е.А. Комплекс программ для работы с учебным контентом в дистанционных обучающих системах// Известия СНЦ РАН. 2015. №2 (5). Т. 17. С. 992-1003.

Руководящие материалы и ГОСТы

6 РД 34.20.571. Методические указания по расчету показателей готовности к работе электростанции и энергосистем. Введ. 1976-10-22. М., 1976. 25 с.

7 ГОСТ Р 7.0.4-2006. Издания. Выходные сведения. Общие требования и правила оформления. М., 2006. II. 43 с. (Система стандартов по информ., библиот. и изд. делу).

Методические указания или учебные пособия

8 Зеленко Л.С. Методические указания к лабораторному практикуму по дисциплине «Программная инженерия». Самара: СГАУ, 2012. 67 с.

Электронные ресурсы

9 Российская гидроэнергетика [Электронный ресурс] // Русгидро: [сайт]. URL: <http://www.rushydro.ru/industry/russianhydropower/> (дата обращения: 20.12.2024).

10 Гидроэлектростанция (гидроэлектрическая станция, ГЭС) // Энциклопедический словарь юного техника М.: Издательство «Педагогика», 1987 [Электронный ресурс] // Библиотекарь.Ру: электрон. библ. 2006-2024. URL: <http://www.bibliotekar.ru/enc-Tehnika/58.htm> (дата обращения: 20.12.2024).

11 Субботин А.С. Основы гидротехники [Электронный ресурс]. URL: <http://www.cawater-info.net/bk/dam-safety/files/subbotin.pdf> (дата обращения: 03.02.2024).

12 Филиальная структура компании [Электронный ресурс] // Системный оператор Единой энергетической системы: [сайт]. [2009-2017]. URL: <http://so-ups.ru/index.php?id=about> (дата обращения: 20.12.2024).

13 Автоматизированные системы управления технологическими процессами гидроэлектростанции [Электронный ресурс] // Микроника. Инжиниринговый центр: [сайт]. [1999-2016]. URL: <http://mikronika-energo.ru/products/asutp/ges-asu-tp/> (дата обращения: 24.12.2024).

14 Автоматизированная система управления производственными процессами [Электронный ресурс] // MEScontrol: [сайт]. [2003-2017]. URL: <http://mescontrol.ru/articles/systems> (дата обращения: 02.04.2024).

15 Пушкинов А.Ю. Введение в системы управления базами данных: учеб. пособие [Электронный ресурс] // CITForum: электрон. библиотека. 1997-2017. URL: <https://citforum.ru/database/dblearn/dblearn06.shtml> (дата обращения: 20.12.2024).

16 Пользовательский интерфейс [Электронный ресурс] // Википедия: электрон. энциклопедия. 2001-2017. URL: https://ru.wikipedia.org/wiki/Пользовательский_интерфейс (дата обращения: 17.03.2024).

Если необходимо указать системные требования для доступа к документу (наличие специального ПО), то

17 Белова С.В. Язык UML. Диаграмма вариантов использования. Систем. требования: PowerPoint. URL: nkse.ru/component/k2/item/download/7_754f5a247edc6ec6be78218f187338a5.html (дата обращения: 17.10.2024).

Сборники научных трудов или трудов конференций

18 Философия культуры и философия науки: проблемы и гипотезы: межвуз. сб. науч. тр./ Саратов. гос. ун-т; [под ред. С.Ф. Мартыновича]. Саратов: изд-во Сарат. ун-та, 1999. 199 с.

19 Акимова А.Е., Трешников А.А., Зеленко Л.С. Информационная среда ГЭС. Подсистема расчета показателей эффективности работы оборудования // Перспективные информационные технологии (ПИТ-2017): сб. науч. тр. межд. научно-техн. конф.; [под ред. С.А. Прохорова]. Самара: Изд-во СНЦ РАН, 2017. С. 41-44.

Если электронное издание

20 Акимова А.Е., Трешников А.А., Зеленко Л.С. Подсистема расчета показателей эффективности работы оборудования // Математика. Компьютер. Образование: труды XXIV межд. конф., 23-28 января 2017 г., г. Пущино. URL: <http://www.mce.su/rus/presentations/p283063/> (дата обращения: 02.03.2017).

ПРИЛОЖЕНИЕ А

Руководство пользователя

А.1 Назначение системы

Приводится краткое описание возможностей системы.

А.2 Условия работы системы

Пример.

Для корректной работы системы необходимо наличие соответствующих программных и аппаратных средств.

1 Требования к техническому обеспечению:

- ЭВМ типа IBM PC;
- процессор типа x86 или x64 тактовой частоты 1400 МГц и выше;
- ...

2 Требования к программному обеспечению:

- Windows 7 Professional 64 бит с пакетом обновлений SP1 и выше;
- установленная платформа .Net версии 4.0 и выше;
- установленная СУБД

А.3 Установка системы

Пример.

Система поставляется в **виде zip-архива**. Данный файл необходимо распаковать в любую директорию на жестком диске. Запускаемым файлом системы является файл **xxx.exe**.²

² Если необходимы дополнительные ресурсы для обеспечения работоспособности системы, то все для них также должны быть перечислены условия установки. *Если установка нестандартная, то она должна быть подробно описана (в объеме, достаточном для понимания пользователя).*

А.4 Работа с системой

А.4.1 Работа с системой в режиме администратора (если необходимо)

Вход в систему (авторизация)

...

А.4.2 Работа с системой в режиме пользователя

Вход в систему (авторизация)

Вход в систему (регистрация)

Настройка параметров кроссворда

ПРИЛОЖЕНИЕ Б

Листинг модулей программы

7-10 страниц исходного кода шрифт Times New Roman 10 пт 1
интервал