МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Новосибирский национальный исследовательский государственный
университет»
(Новосибирский государственный университет, НГУ)
Структурное подразделение Новосибирского государственного университета –
Высший колледж информатики Университета (ВКИ НГУ)
КАФЕДРА ИНФОРМАТИКИ

РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ ОБУЧЕНИЯ СТУДЕНТОВ КИТАЙСКО-РОССИЙСКОГО ИНСТИТУТА ХЭЙЛУНЦЗЯНСКОГО УНИВЕРСИТЕТА ЯЗЫКУ ПРОГРАММИРОВАНИЯ C++

Квалификация программист

Руководитель Голкова Н.В.
инженер ЛИ ВКИ НГУ « 7 » июня 2024 г.

Студент курса Санникова А.С.
107 са1 « 7 » июня 2024 г.

Нормоконтроль Цимбалист Г.Н.
« 7 » июня 2024 г.

Новосибирск
2024

СОДЕРЖАНИЕ

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ, УСЛОВНЫХ ОБОЗНАЧЕНИЙ И ТЕРМИНОВ

Китайско-российский институт (КРИ) – совместный проект Новосибирского Государственного Университета (НГУ) и Хэйлунцзянского университета (ХУ).

Система управления базами данных (СУБД) — это набор инструментов, которые позволяют удобно управлять базами данных: удалять, добавлять, фильтровать и находить элементы, менять их структуру и создавать резервные копии.

Тест по русскому языку как иностранному (ТРКИ) – международный экзамен на определение уровня владения русским языком. В мире также распространены английский перевод названия Test of Russian as a Foreign Language и соответствующая ему аббревиатура TORFL.

Application programming interface (API) – это набор способов и правил, по которым различные программы общаются между собой и обмениваются данными.

Пользовательский интерфейс (UI) – интерфейс, обеспечивающий передачу информации между пользователем и программно-аппаратным компонентами компьютерной системы.

eXtensible Application Markup Language (XAML) – язык разметки, используемый для инициализации объектов в технологиях на платформе .NET, который используется прежде всего для создания пользовательского интерфейса декларативным путем.

Один ко многим (1:M) — тип связи таблиц, когда одной записи главной таблицы можно сопоставить несколько записей подчинённой таблицы.

Model-View-ViewModel (MVVM) – это шаблон проектирования архитектуры приложения, который имеет три основных компонента: модель, представление и модель представления, что позволяют отделить логику программного средства от его визуальной части (представления).

ВВЕДЕНИЕ

На протяжении столетий Россия строила и развивала отношения с Китаем, с которым сейчас имеет тесное всеобъемлющее партнерство. На сегодняшний день заключено более трехсот межправительственных договоров и соглашений, охватывающих почти все области сотрудничества. Одним из важнейших направлений в области гуманитарного сотрудничества является взаимодействие в области образования. Данное сотрудничество имеет договорно-правовую базу, которая основывается на нескольких межправительственных соглашениях и протоколах [1].

Таким образом, в последние десятилетия активно развивалось сотрудничество учебных заведений из разных регионов России и провинций Китая [2]. Благодаря установленным связям было привлечено большое количество китайских студентов. На данный момент вузы России и их зарубежные филиалы насчитывают 32,6 тыс. китайских студентов [3].

Новым толчком для развития отношений в области образования стало функционирование совместных образовательных учреждений (СОУ) и реализации совместных образовательных программ. Первым СОУ стал Китайско-российский институт, открытый в 2011 г. при сотрудничестве Новосибирского Государственного Университета и Хэйлунцзянского университета. Китайско-российский институт ставит своей целью подготовить квалицированных специалистов и обучить студентов русскому языку в рамках выбранной специальности.

В данной ситуации прогресс в развитии цифровых технологий и информатизация образования могли бы существенно помочь достичь целей Китайско-российского института, ведь в современном мире у каждого имеются персональные устройства, которые так же применяются и в процессе обучения. Постоянным же спутником современного человека являются мобильные устройства, которые открывают новые пути для обучения и

совершенствования навыков. Появляется возможность использовать мобильные приложения для расширения, улучшения и углубления знаний.

Целью выпускной квалификационной работы является разработка мобильного приложения для обучения студентов Китайско-российского института Хэйлунцзянского университета языку программирования C++ для ОС Android.

Для достижения поставленной цели необходимо решить следующие задачи:

- произвести постановку и анализ требований к программному продукту;
- произвести анализ предметной области и определить спецификации;
- провести обзор и сравнительный анализ программных сред и средств разработки приложения;
- спроектировать алгоритмы для реализации программного продукта;
- спроектировать пользовательские сценарии и определить операции пользователей;
- спроектировать и реализовать базу данных для программного продукта;
- реализовать мобильное приложение, соответствующее заявленным требованиям;
- произвести тестирование и оптимизацию разработанного приложения.

Способ решения данных задач зависит от выбора рабочей среды и языка программирования. Для разработки программного средства будут использоваться следующие технологии:

- кроссплатформенная среда разработки Xamarin.Forms;
- СУБД SQLite;
- язык программирования C#.

1 ПОСТАНОВКА ЗАДАЧИ ВКР

Для постановки целей, требований и критерий разрабатываемого мобильного приложения было подготовлено Техническое задание (Приложение А), на основе которого были составлены нижеприведённые требования.

1.1 Бизнес-требования

Для отражения целей Заказчика, которые можно достигнуть с помощью мобильного приложения, были составлены следующие бизнес-требования:

- повысить успеваемость студентов по дисциплине «Конструирование языковых программ», чтобы на итоговой аттестации 25% студентов набирали больше 90 баллов, 50% студентов набирали от 80 до 89 баллов, и оставшиеся 25% студентов набирали от 60 до 79 баллов. замотивировав к изучению учебного материала в предпочтительном для самих студентов режиме;
- повысить уровень русского языка студентов до ТРКИ-I, создав возможность для расширения словарного запаса русского языка, необходимого для формирования карьеры студентов.
- увеличить объемы предоставляемой информации и практики изучаемого материала на 20% за счет перехода на новую технологическую платформу и передачи информации через новый канал;
- вдвое уменьшить временные затраты на поиск необходимой информации для изучения дисциплины «Конструирование языковых программ», создав удобную платформу, предоставляющую необходимые материалы.

1.2 Пользовательские требования

Целевой аудиторией являются совершеннолетние граждане Китайской Народной Республики, обучающиеся в Китайско-российском институте дисциплине «Конструирование языковых программ» и русскому языку. Они в свою очередь и являются главными пользователями программного средства.

Основными функциями разрабатываемого мобильного приложения являются:

- Получение необходимого теоретического учебного материала.
- Выполнение практических заданий.
- Самоконтроль успеваемости.
- Расширение словарного запаса русского языка.

Следовательно, с помощью мобильного приложения пользователи должны иметь возможность выполнить следующие задачи:

- Просматривать материалы курса по разделам.
- Искать учебный материал по теме.
- Выполнять задания по разделам курса.
- Получать результат по выполненным заданиям.
- Просматривать китайско-русский словарь.
- Производить поиск слов в словаре на русском языке.
- Производить поиск слов в словаре на китайском языке.
- Сохранять слова в избранное.
- Удалять слова из избранного.
- Просматривать успеваемость по разделам курса.
- Просматривать историю поиска слов в словаре.
- Просматривать историю изучения материалов курса.

1.3 Системные требования

Программное средство имеет клиент-серверную архитектуру и, следовательно, состоит из клиента и сервера.

Клиентская часть представляет собой интерфейс с набором функций, с которым пользователь может взаимодействовать.

Серверная часть представляет собой программно-аппаратное устройство, которая хранит данные, обрабатывает запросы и вычисления.

Таким образом, для мобильного приложения были составлены следующие требования:

- Реализовать клиентскую часть приложения с помощью языка разметки XAML в кроссплатформенной среде разработки Xamarin.Forms.
- Реализовать серверную часть с помощью языка программирования C# в среде разработке Xamarin.Forms.
- Реализовать базу данных серверной части приложения в SQLite.
- Для серверной части необходимо мобильное устройство с версией Android 8.0 и выше.

1.4 Требования к графическому пользовательскому интерфейсу

Пользовательский интерфейс должен быть удобным для пользователей и учитывать их потребности. Программа должна иметь подсказки и указатели на функциональные компоненты приложения. При возникновении ошибки или сбоя, приложение должно выдавать соответствующее информацию, понятное конечному пользователю. У пользователя должна быть возможность продолжить обучение с того места, которое было сохранено в последний раз.

В качестве основного светлого фона используются цвет – #F3F4F4 и #FFFFFF, а темного – #303030 и #1C1C1C;

Для акцентирования внимания пользователя используются следующие цвета:

- в светлом режиме #7766C6 и #46467A;
- в темном #7766C6 и #FFFFFF.

Следующие элементы должны присутствовать на каждой странице:

- панель инструментов в верхней части экрана, содержащая заголовок страницы и кнопки управления текущим экраном;
- главное навигационное меню – панель вкладок в нижней части экрана, позволяющая быстро переключаться между разделами приложения.

Пользовательский интерфейс должен удовлетворять следующим требованиям:

- обеспечивать легкое определение местоположения пользователя в системе;
- обеспечивать минимум усилий и временных затрат при навигации.

В дизайне сайта не должны присутствовать:

- много сливающегося текста;
- цветовые сочетания и графические решения, не заявленные Заказчиком.

1.5 План-график выполнения ВКР

В таблице 1 представлен план разработки программного продукта, определяющий порядок работ, которые необходимо выполнить, ресурсы, обязанности и так далее. Здесь указаны все поставленные задачи и срок их выполнения. Также были выделены количество этапов и общие сроки их выполнения.

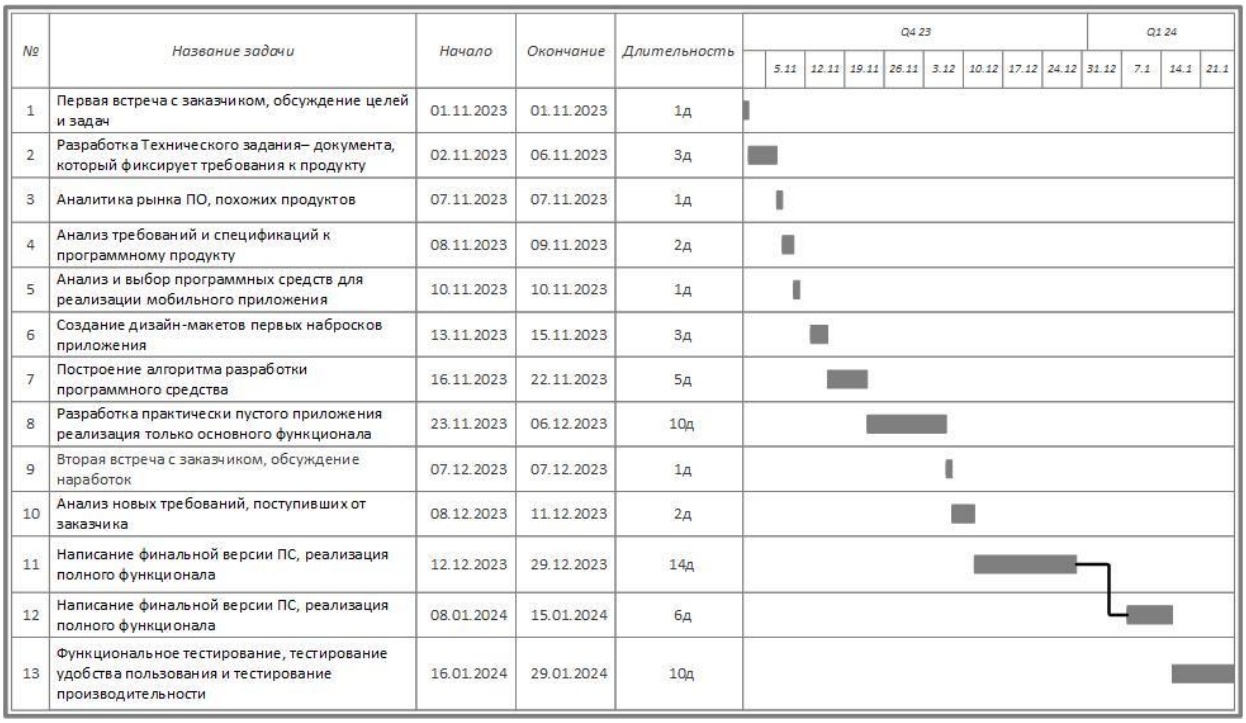
Таблица 1 – План разработки мобильного приложения

№	Задачи	Сроки для задач
1	Первая встреча с заказчиком, обсуждение целей и задач	1 день

Продолжение таблицы 1

№	Задачи	Сроки для задач
2	Разработка Технического задания– документа, который фиксирует требования к продукту	3 дня
3	Аналитика рынка ПО, похожих продуктов	1 день
4	Анализ требований и спецификаций к программному продукту	2 дня
5	Анализ и выбор программных средств для реализации мобильного приложения	1 день
6	Создание дизайн-макетов первых набросков приложения	3 дня
7	Построение алгоритма разработки программного средства	5 дней
8	Разработка практически пустого приложения реализация только основного функционала	10 дней
9	Вторая встреча с заказчиком, обсуждение наработок	1 день
10	Анализ новых требований, поступивших от заказчика	2 дня
11	Написание финальной версии программного средства, реализация полного функционала	20 дней
12	Функциональное тестирование, тестирование удобства пользования и тестирование производительности	10 дней
ИТОГО:		59 дней

На рисунке 1 представлена диаграмма Ганта разработки программного продукта, с помощью которой можно получить общее представление о проекте, обобщающую информацию и иллюстрацию влияния выбора на сроки.



№	Название задачи	Начало	Окончание	Длительность
1	Первая встреча с заказчиком, обсуждение целей и задач	01.11.2023	01.11.2023	1д
2	Разработка Технического задания– документа, который фиксирует требования к продукту	02.11.2023	06.11.2023	3д
3	Аналитика рынка ПО, похожих продуктов	07.11.2023	07.11.2023	1д
4	Анализ требований и спецификаций к программному продукту	08.11.2023	09.11.2023	2д
5	Анализ и выбор программных средств для реализации мобильного приложения	10.11.2023	10.11.2023	1д
6	Создание дизайн-макетов первых набросков приложения	13.11.2023	15.11.2023	3д
7	Построение алгоритма разработки программного средства	16.11.2023	22.11.2023	5д
8	Разработка практически пустого приложения реализация только основного функционала	23.11.2023	06.12.2023	10д
9	Вторая встреча с заказчиком, обсуждение наработок	07.12.2023	07.12.2023	1д
10	Анализ новых требований, поступивших от заказчика	08.12.2023	11.12.2023	2д
11	Написание финальной версии ПС, реализация полного функционала	12.12.2023	29.12.2023	14д
12	Написание финальной версии ПС, реализация полного функционала	08.01.2024	15.01.2024	6д
13	Функциональное тестирование, тестирование удобства пользования и тестирование производительности	16.01.2024	29.01.2024	10д

Рисунок 1 – Диаграмма Ганта проекта

На горизонтальной оси представлена временная шкала, измеряемая в рабочих днях. На вертикальной оси отображается список задач, дата начала и завершения их выполнения. Горизонтальная полоса, расположенная между двумя осями, обозначает задачи, которые необходимо выполнить в период выполнения ВКР. Начало создания программного средства датируется 1 ноября 2023 года, и окончание — 29 января 2024 года.

2 АНАЛИЗ ТРЕБОВАНИЙ И ОПРЕДЕЛЕНИЕ СПЕЦИФИКАЦИЙ

2.1 Описание предметной области задачи ВКР

2.1.1 Информационные объекты предметной области и взаимосвязи между ними

В ходе изучения дисциплины «Конструирование языковых программ» студенты Китайско-российского института осваивают язык программирования C++, а также изучают русский язык в рамках своей будущей профессиональной деятельности.

Сам учебный процесс изучения данной дисциплины делится на блоки и разделы, включающие в себя определенное количество лекций и заданий к ним. В данном случае изучение языка программирования C++ является отдельным разделом изучаемого предмета. Любое изучение раздела делится на блоки – главы, и темы по ним, а по темам уже читаются лекции и выдаются практические задания, которые обязательно оцениваются. Оценка таких практических заданий отражает уровень усвоения материала студентом, а также отслеживает и показывает его успехи в обучении.

Что касается изучения русского языка, то у студентов, возникают трудности, связанные с лексикой [4]. Одной из сложностей для китайских студентов в изучении русского языка как иностранного является использование многозначных слов. Поэтому для правильного понимания смысла слова необходим контекст. Кроме того, студентам следует больше читать, чтобы видеть, понимать и чувствовать слова и их значения.

Таким образом, лучшим способом работы с лексикой является ее запоминание и употребление в речи. При чтении любых публикаций на русском языке китайские студенты имеют возможность запоминать русские фразеологические обороты и выражения, которые они смогут повторять и использовать в устной речи [5].

Отсюда следует, что для обучения студентов языку программирования C++ должны быть соответствующие ресурсы и материалы, разделенные по блокам и темам. Данные материалы включают в себя конспекты лекций и практические задания на русском языке, сопровождающиеся актуальным словарем терминов и их переводом.

Соответственно, можно выделить следующие информационные объекты:

- Раздел дисциплины – язык программирования C++.
- Глава раздела.
- Тема главы.
- Учебный материал по теме главы, включающий в себя конспект лекции и практическое задание.
- Русско-китайский словарь терминов, состоящий из списка русских слов и перевода их значения на китайском.

Вышеуказанные информационные объекты связанны следующим образом:

- В одном разделе дисциплины содержатся множество глав.
- В одной главе может быть множество тем.
- Множество читаемых лекций по множествам темам.
- Множество практических заданий по множествам темам.
- Множество русских терминов, имеющих множество переводов на китайский язык.

Исходя из выделенных информационных объектов и их взаимоотношений, которые изображены на рисунке 2, можно определить требования к ним и логику взаимодействия студента с ними.

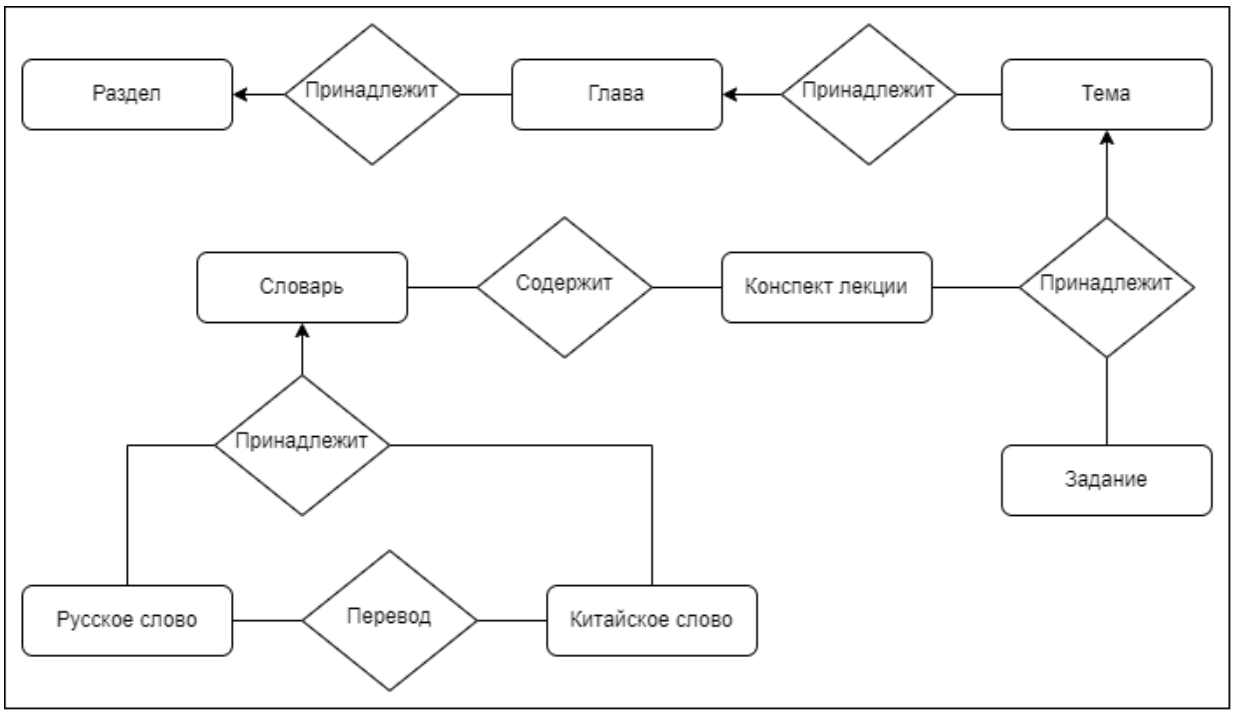


Рисунок 2 – Информационные объекты

2.1.2 Информационные и функциональные потребности пользователей разрабатываемой ПС

Благодаря информационным технологиям на сегодняшний день есть возможность дистанционного обучения на любой доступной технической платформе. Такое обучение ожидает от студентов практически полностью самостоятельное осваивание и усваивание учебного материала. Важную роль играет высокое качество подготовленных учебных материалов, соответствующие требованиям учебного плана и целям освоения дисциплины.

При разработке программного средства для самостоятельного дополнительного обучения дисциплины целью должен стать текстовый элемент с определенной учебной задачей. Например:

- просмотр значения и перевода русских слов;
- описание ожидаемых от студента действий;
- создание визуального дополнения к конспектам лекций;
- создание возможности просмотра значения термина в момент изучения конспекта;
- практические задания с автоматической проверкой.

Использование перечисленных возможностей текстовых материалов позволит студентам направлять усилия на изучение как русского языка, так и на изучение языка программирования C++, самостоятельно скорректировать собственные результаты обучения и развивать практические навыки.

Таким образом, можно выделить информационные требования пользователей:

- просмотр и выбор глав изучаемого раздела;
- просмотр и выбор тем проходимой главы;
- просмотр конспекта лекции по выбранной теме;
- просмотр терминов, используемых в конспекте;
- просмотр и выполнение задания по определенной теме;
- просмотр оценки выполнения практических заданий (успеваемость) по темам и главам в разделе;
- просмотр и поиск значений русских слов и выражений.

В соответствии с разработанными требованиями необходимо спроектировать многостраничный интерфейс мобильного приложения. Главная форма приложения должна содержать такие страницы:

1. Главная.
2. Библиотека.
3. Успеваемость.
4. Избранное.

На странице «Главная» должна отображаться поисковая строка для поиска терминов в словаре. Также тут должны быть выведены последние просмотренные термины и лекция. Здесь выполняется функция просмотра и поиска значений русских слов и выражений, а также конспектов лекций.

Также на странице «Главная» присутствует кнопка «Настройки», отрывающая вкладку с настройками программного средства.

Во вкладке «Библиотека» главной формы пользователь может воспользоваться следующим функциям:

- Просмотр и выбор раздела.

- Просмотр и выбор главы.
- Просмотр конспекта лекции.
- Просмотр используемых терминов в конспекте.
- Просмотр и выполнения задания.

Вкладка «Успеваемость» отображает оценки выполненных практических заданий по темам и главам в выбранном разделе.

Во вкладке «Избранное» отображается список сохраненных терминов и выражений.

2.1.3 Методы работы с информационными объектами предметной области

Для расчета общей оценки в практической части по теме, которая будет отражаться на странице «Успеваемость», используется простая математическая модель оценки уровня знаний [6].

Простейшая модель оценки является самой распространённой и, как очевидно по названию модели, простой. За каждый ответ, выполнение заданий, в практической части в выбранной теме студент будет оцениваться по двухбалльной шкале – правильно или неправильно.

Общая оценка выставляется путем вычисления значения R по следующей формуле (1):

$$R = \frac{\sum_{i=1}^{k} R_i}{n}, \quad (1)$$

где R_i – правильный ответ обучаемого на i-е задание; k – количество выполненных заданий по теме из n предложенных, при этом k <= n.

2.1.4 Обзор существующих программных реализаций решения задачи

В ходе анализа предметной области, были изучены мобильные приложения со схожей тематикой и деятельностью. Рассмотрим некоторые из них:

1. «Выучить C++» – бесплатное мобильное приложение для изучения языка программирования C++, разработанное компанией Coding and Programming. Приложение предлагает учебники по программированию на C++, уроки программирования, готовые примеры программ, вопросы и ответы. На рисунке 3 представлены скриншоты главной страницы и страницы урока.

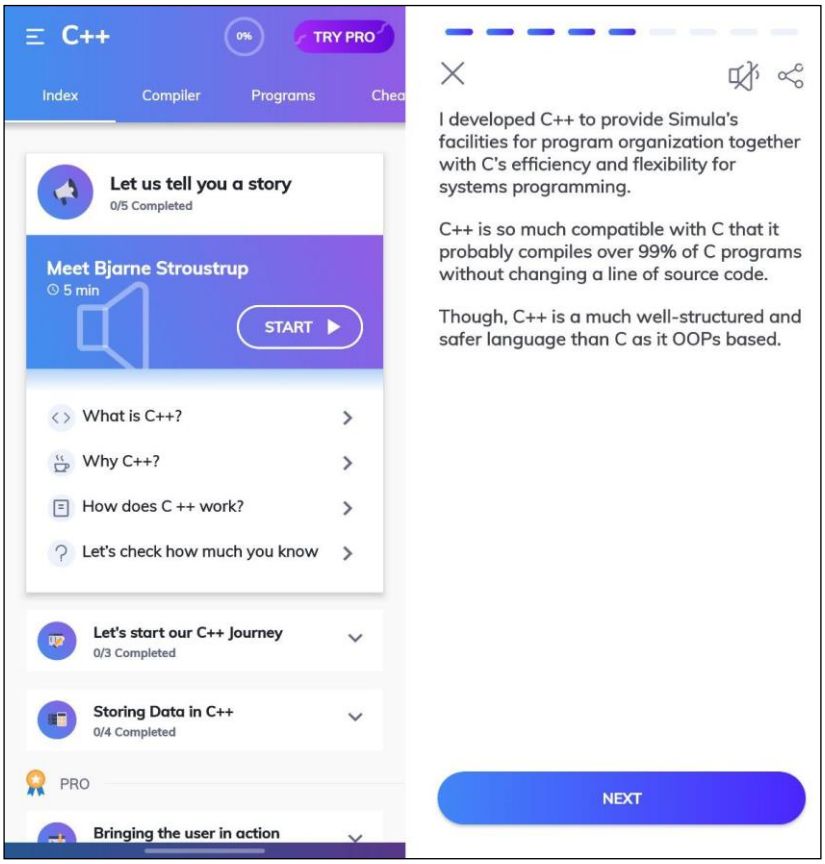


Рисунок 3 – Скриншоты главной страницы и страницы урока в приложении «Выучить C++»

Достоинства приложения:

- Проработанный и интуитивный пользовательский интерфейс.
- Наличие примеров готовых программ.

- Наличие встроенного компилятора.
- Наличие интерактивных уроков.
- Наличие учебного материала, разделенного по категориям.

Недостатки:

- В бесплатной версии приложения доступно ограниченное количество уроков.
- Постоянная реклама, предлагающая купить полную версию приложения.

2. «Learn C++» – бесплатное мобильное приложение для изучения языка программирования C++, выпущенное компанией «Programiz». Программное средство предлагает научиться программировать на C++ с помощью интерактивного редактора кода и викторин. На рисунке 4 представлены скриншоты главной страницы и урока.

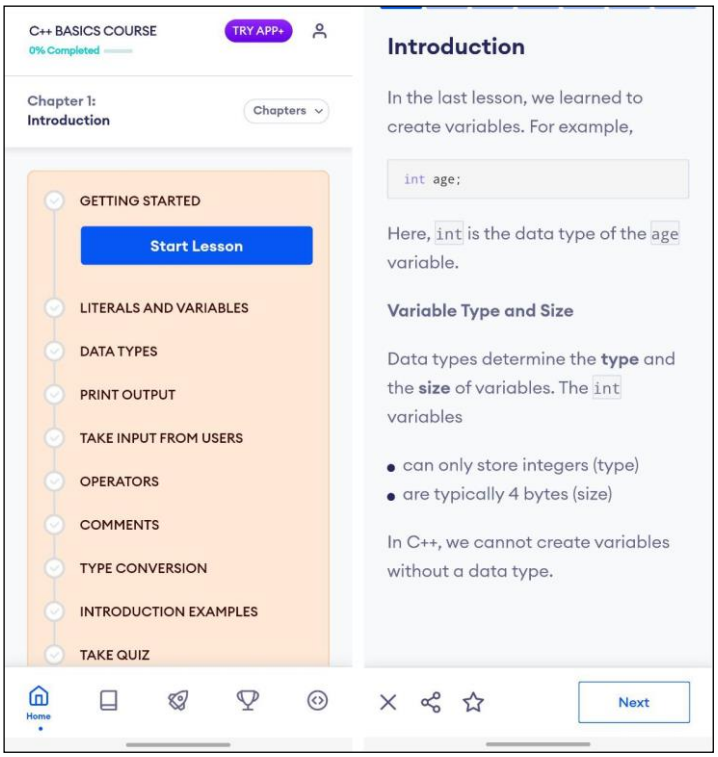


Рисунок 4 – Скриншоты главной страницы и страницы урока в приложении «Learn C++»

Достоинства приложения:

- Интуитивно понятный интерфейс.

- Наличие интерактивных уроков.
- Наличие готовых примеров программ с объяснениями реализации.
- Наличие взаимодействия с пользователем через вывод окон с сообщением.
- Наличие встроенного компилятора с возможностью сохранения кода.
- Наличие рейтинга зарегистрированных пользователей.

Недостатки:

- Ограниченные материалы и ресурсы в бесплатной версии приложения.
- Не видно сроку состояния из-за цветовой гаммы приложения.

3. «Stepik: онлайн курсы» – бесплатное мобильное приложение, разработанное компанией «Stepik». Приложение представляет собой платформу с онлайн-курсами. На рисунке 5 представлены скриншоты главной страницы и страницы обучения приложения.

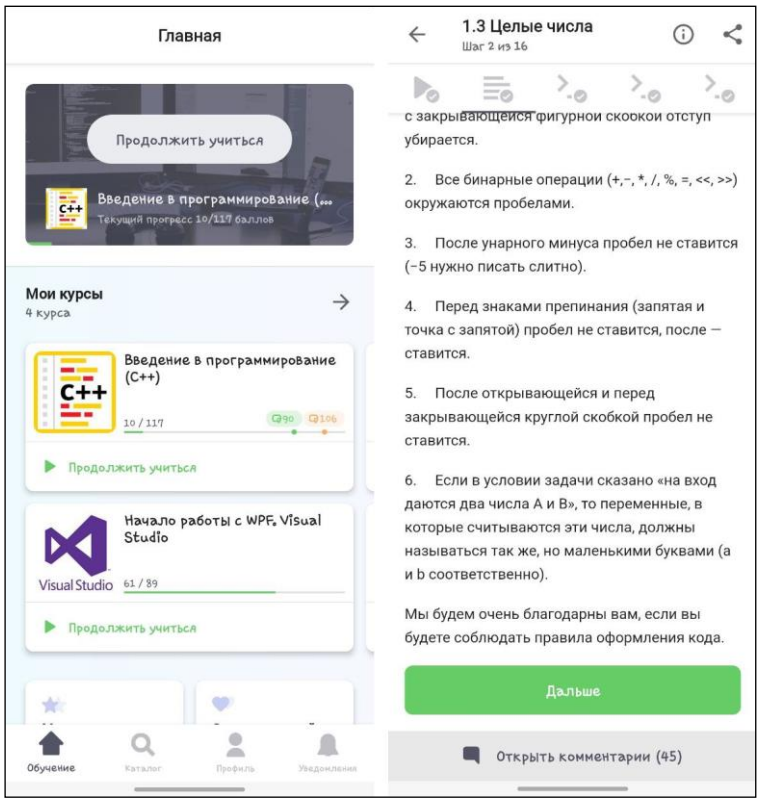


Рисунок 5 – Скриншоты главной страницы и страницы обучения в приложении «Stepik: онлайн курсы»

Достоинства приложения:

- Интуитивно понятный пользовательский интерфейс.
- Наличие большого выбора курсов разных направлений.
- Возможность выбрать язык курсов.
- Возможность отслеживания активности прохождения курсов.

Недостатки:

- Долгая загрузка элементов пользовательского интерфейса.

Выполним сравнение представленных выше мобильных приложений. Данные программные средства будут оцениваться по трех бальной шкале, где: 3-отлично, 2-нормально, 1- плохо. Эта шкала позволит продемонстрировать, в каких областях то или иное приложение имеет преимущества по сравнению с другими, или наоборот уступает им.

Критерии для оценки приложений:

- Дизайн — дизайн должен быть приятным для пользователя и не содержать в себе отвлекающие от восприятия информации элементов, а также шрифт текста должен быть легко читабельным и не сливающимся.
- Пользовательский интерфейс — это удобство пользования приложением: проработанные структура программы и система навигации, наличие функциональных элементов, взаимодействие с пользователем.
- Функциональность — наличие разнообразных функциональных возможностей, например: форма поиска, добавление в избранное, возможность включить и выключить уведомления и т.д.
- Актуальность — учебные материалы в приложении должны содержать в себе актуальную и полную информацию.

В таблице 2 приведены оценки по вышеперечисленным критериям и произведено сравнение программных средств.

Таблица 2 – Сравнение аналогов

Критерии	Выучить C++	Learn C++	Stepik: онлайн курсы	Разрабатываемое мобильное приложение
Дизайн	2/3	3/3	1/3	2/3
Пользовательский интерфейс	2/3	3/3	2/3	3/3
Функциональность	3/3	3/3	3/3	3/3
Актуальность	2/3	2/3	2/3	3/3

Проанализировав данные таблицы 2, можно сделать вывод, что, большинство из представленных приложений имеют проработанную систему навигации и разнообразные функциональные возможности. С другой стороны, программные средства содержат отвлекающие элементы дизайна и ограниченные учебные ресурсы.

Выполненный сравнительный анализ приложений-аналогов и разрабатываемого программного средства позволил выявить достоинства и недостатки аналогов, а также оценить конкурентное преимущество разрабатываемого программного средства.

2.1.5 Концептуальное обоснование разработки

Вышеприведенные приложения-аналоги имеют хорошо подготовленные учебные материалы, не лишены практических заданий и инструментов для их выполнения. Но первые два приложения имеют лишь одну цель – изучить язык программирования C++, в то время как существует потребность студентов КРИ в изучении русского языка в рамках выбранной специальности и, соответственно, в расширении лексического словаря.

К тому же, не смотря на хорошо структурированную и полную информацию в данных приложениях, не все данные и не все задания соответствуют учебному плану прохождения дисциплины «Конструирование

языковых программ». Также стоит отметить, что не все учебные материалы и инструменты доступны в бесплатных версиях рассмотренных аналогов.

Для мотивации к обучению и отслеживания успехов в изучении дисциплины необходима система оценивания, которая присутствовала не во всех указанных аналогах.

Таким образом, необходимо такое мобильное приложение, где есть возможность освоить русский язык в рамках изучаемой дисциплины, что позволило бы студентам расширить профессиональный словарный запас. Такое приложение позволяло бы студентам получать необходимые учебные материалы, а также практиковать и улучшать не только свои профессиональные навыки за счет практических заданий, но и русский язык, который они изучают.

2.2 Классы и характеристики пользователей

Потенциальными пользователями приложения являются студенты. Примерный профиль названной категории пользователей может выглядеть следующим образом (таблица 3).

Таблица 3 – Описание пользовательской роли

Пользователи	Студенты КРИ
Социальные характеристики	Мужчины, женщины Возраст с 18 лет Носители китайского языка Средний уровень владения мобильным устройством
Мотивационно целевая среда	Учебный материалы в высоком разрешении Информативность, наполненность Факты, подтверждённые источниками Расширение русского словарного запаса Высокая мотивация к обучению

Продолжение таблицы 3

Навыки и умения	Должны иметь базовые навыки работы с мобильном устройством Начальный уровень владения русским языком Владение основами синтаксиса C++
Требования к ПО	Удобность Информативность, наполненность Бесплатный доступ Возможность проверки знаний Разрешение материалов в высоком качестве
Задачи пользователя	Просмотр информации Самопроверка знаний
Рабочая среда	Стандартизированное мобильное устройство

2.3 Функциональные требования

2.3.1 Определение функциональных возможностей ПС

Разрабатываемое программное средство должно быть простым в использовании и доступно ограниченному кругу пользователей. В качестве пользователей можно выделить один класс пользователей:

1. Студент – неавторизованный пользователь, обладает правами:

- Просмотр курсов.
- Поиск по теме.
- Просмотр материала курса.
- Просмотр и прохождение заданий курса.
- Просмотр успеваемости.
- Просмотр словаря.
- Поиск значения слова.
- Просмотр значения слова.

- Добавление слова в избранное.
- Удаление слова из избранного.
- Просмотр истории.
- Просмотр и изменения настроек программы.
- Просмотр избранных слов.

Исходя из данных характеристик была разработана UML-диаграмма прецедентов для класса пользователя – студент. На рисунке 6 представлена диаграмма вариантов использования студентом приложения. Рамкой обозначены границы программного средства.

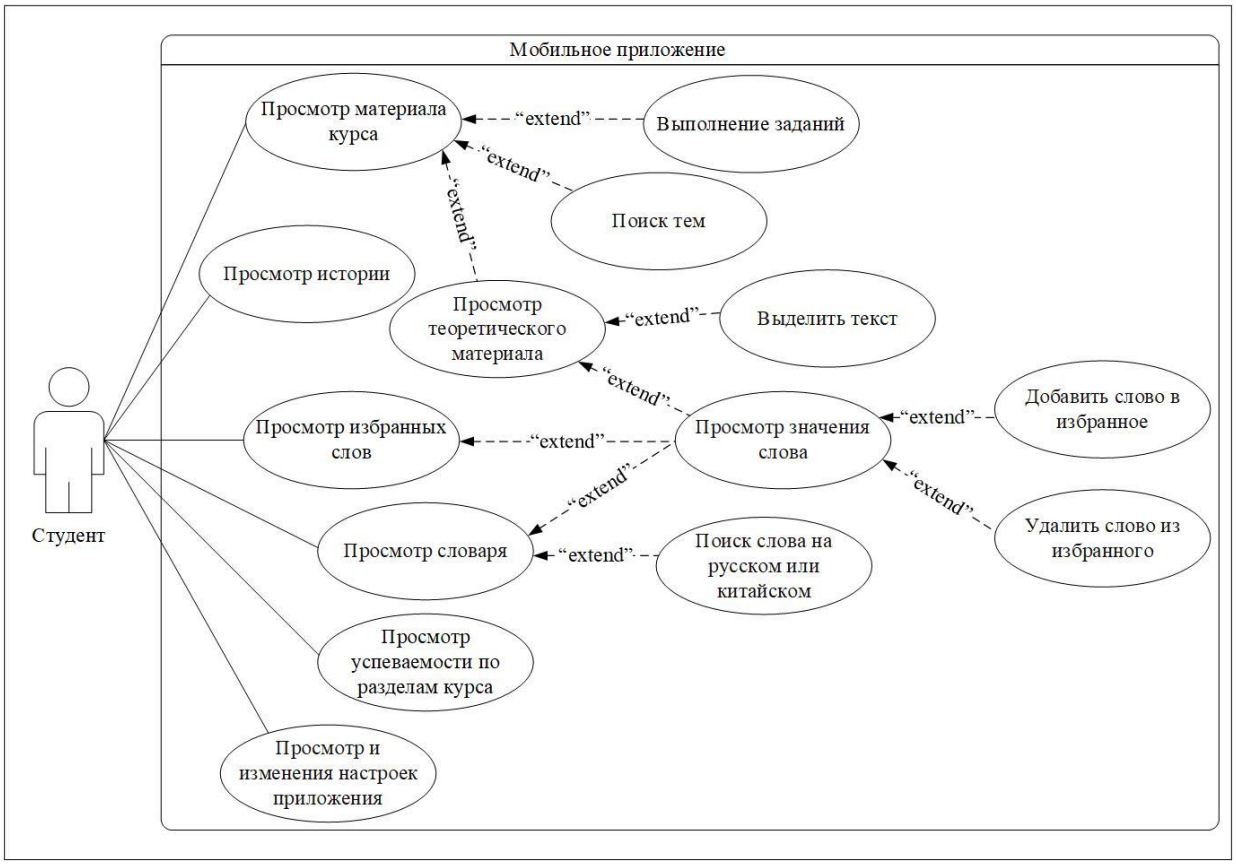


Рисунок 6 – UML-диаграмма для пользователя «Студент»

2.3.2 Описание прецедентов

Выделенная пользовательская роль – «Студент» будет взаимодействовать непосредственно с информационной системой.

Для детализации прецедентов были составлены описания, представленные в таблицах 4-14.

Таблица 4 – Описание прецедента «Просмотр материала курса»

Название прецедента	Просмотр теоретического материала
Исполнитель	Студент
Цель	Изучить теоретический материал по теме
Основной успешный сценарий	1. Студент переходит на страницу «Библиотека», где выбирает раздел, а потом главу. 2. Студент переходит во вкладку с лекциями. 3. Студент выбирает тему в главе. 4. Открывается окно с конспектом теоретического материала.
Ссылки	Отправляется запрос в базу данных

Таблица 5 – Описание прецедента «Просмотр значения слова»

Название прецедента	Просмотр значения слова
Исполнитель	Студент
Цель	Просмотреть значения слова
Основной успешный сценарий	1. На главной странице студент в поисковой строке вводит слово, значение которого хочет найти. 2. На странице отображаются результаты поиска. 3. Студент выбирает среди результатов слово. 4. Открывается страница с значением слова.
Ссылки	В базе данных сохраняется дата просмотра значения слова

Таблица 6 – Описание прецедента «Выполнения задания

Название прецедента	Выполнение задания
Исполнитель	Студент
Цель	Выполнить задание по теме
Основной успешный сценарий	1. Студент переходит на страницу «Библиотека», где выбирает раздел, а потом главу. 2. Студент переходит во вкладку с заданиями. 3. Студент выбирает тему в главе. 4. Открывается окно с заданием. 5. Студент решает задание и сдает его.
Ссылки	1. В базе данных сохраняется результат выполнения задания. 2. Во вкладке «Успеваемость» будет изменена общая оценка по теме.

Таблица 7 – Описание прецедента «Поиск учебного материала по теме»

Название прецедента	Поиск учебного материала по теме
Исполнитель	Студент
Цель	Найти учебный материал по теме
Основной успешный сценарий	1. Студент переходит на страницу «Библиотека». 2. В верхней части страницы в поисковой строке студент вводит название темы. 3. На странице отображаются найденные материалы по запросу.
Ссылки	В базу данных отправляется запрос

Таблица 8 – Описание прецедента «Просмотр избранных слов»

Название прецедента	Просмотр избранных слов
Исполнитель	Студент
Цель	Просмотреть избранные слова
Основной успешный сценарий	1. Студент переходит на страницу «Избранное». 2. На странице отображается список избранных слов.
Ссылки	В базу данных отправляется запрос

Таблица 9 – Описание прецедента «Добавление слова в избранное»

Название прецедента	Добавление слова в избранное
Исполнитель	Студент
Цель	Добавить слово в избранное
Основной успешный сценарий	1. На главной странице студент в поисковой строке вводит слово, значение которого хочет найти. 2. На странице отображаются результаты поиска. 3. Студент выбирает среди результатов слово. 4. Открывается страница с значением слова. 5. На странице нажать на кнопку добавления слова в избранное.
Ссылки	1. В базу данных отправляется запрос. 2. Во вкладке «Избранное» в список слов добавляется новое слово.

Таблица 10 – Описание прецедента «Удаление слова из избранного»

Название прецедента	Удаление слова из избранного
Исполнитель	Студент
Цель	Удалить слово в избранное
Основной успешный сценарий	1. Студент переходит на страницу «Избранное». 2. На странице отображается список слов. 3. Студент выбирает слово. 4. Открывается страница со словом. 5. На странице нажать на кнопку удаления слова из избранного.
Ссылки	1. В базу данных отправляется запрос. 2. Во вкладке «Избранное» из списка слов удаляется выбранное слово.

Таблица 11 – Описание прецедента «Поиск значения слова»

Название прецедента	Поиск значения слова
Исполнитель	Студент
Цель	Найти значение введенного слова
Основной успешный сценарий	1. На главной странице студент в поисковой строке вводит слово, значение которого хочет найти. 2. На странице отображаются результаты поиска.
Ссылки	В базу данных отправляется запрос

Таблица 12 – Описание прецедента «Выделение текста»

Название прецедента	Выделение текста
Исполнитель	Студент
Цель	Выделить текст в учебном материале
Основной успешный сценарий	1. Студент переходит на страницу «Библиотека», где выбирает раздел, а потом главу. 2. Студент переходит во вкладку с лекциями, выбирает тему в главе. 3. Открывается окно с теоретическим материалом. 4. Студент выделяет текст.
Ссылки	В базу данных отправляется запрос.

Таблица 13 – Описание прецедента «Просмотр успеваемости»

Название прецедента	Просмотр успеваемости
Исполнитель	Студент
Цель	Посмотреть успеваемость
Основной успешный сценарий	Студент переходит на страницу «Успеваемость», где отображается вся успеваемость по разделам
Ссылки	В базу данных отправляется запрос

Таблица 14 – Описание прецедента «Просмотр и изменения настроек приложения»

Название прецедента	Просмотр и изменения настроек приложения
Исполнитель	Студент
Цель	Просмотреть и изменить настройки
Основной успешный сценарий	1. Студент переходит на страницу «Настройки» 2. В списке настроек студент выбирает параметр и изменяет его
Ссылки	В базу данных отправляется запрос

2.4 Нефункциональные требования

2.4.1 Требования к программному обеспечению

1. Приложение должно быть разработано на языке C# под платформу Android.
2. Приложение должно функционировать на операционной системе Android с версией не ниже 8.0.
3. Изображение должно адаптироваться под все виды разрешений экрана.
4. В приложении должна быть только вертикальная ориентация.
5. При условии нормального режима функционирования приложение доступно пользователям круглосуточно. Нормальный режим функционирования приложения 365 дней в году 7 дней в неделю 24 часа в сутки режим 24x7x365.
6. Среднее время отклика на действия пользователей при использовании приложения не более 0,5 сек.
7. Приложение должно быть на русском языке с китайским словарем терминов.

2.4.2 Требования к аппаратному обеспечению

1. Мобильное средство должно официально поддерживаться производителями ОС Android.
2. Оперативная память 8 GB и более.
3. Процессор с частотой 2200 МГц и более
4. Разрешение экрана должно быть от 480*800 пикселей до 2048*2732 пикселей.

1.4.3 Требования к надёжности

1. Приложение не должно постоянно требовать подключения к сети Интернет.

2. Приложение должно обеспечивать целостность и непротиворечивость хранимых данных при любых действиях конечных пользователей.

3. Приложение должно обеспечивать корректную обработку ситуаций, вызванных неверными действиями пользователей, неверным форматом или недопустимыми значениями входных данных. В указанных случаях приложение должно выдавать пользователю соответствующие сообщения об ошибках.

4. Сообщения об ошибках не должны содержать техническую информацию и должны предлагать пользователям чёткий алгоритм дальнейших действий.

5. Приложение должно сохранять работоспособность и обеспечивать восстановление своих функций при возникновении следующих внештатных ситуаций:

- при сбоях в системе аппаратной части, приводящих к перезагрузке ОС, восстановление работы подсистемы должно происходить после перезапуска ОС и запуска прикладного программного обеспечения;
- при ошибках в работе аппаратных средств восстановление функций приложения возлагается на платформу;
- при ошибках, связанных с программным обеспечением (ОС и драйверы устройств), восстановление работоспособности возлагается на ОС.

6. В приложении должны быть предусмотрены средства для организации резервного копирования и обеспечения восстановления работоспособности в случае программно-аппаратных сбоев. Должны быть предусмотрены меры по регулярному сохранению файлов и баз данных.

3 ВЫБОР ПРОГРАММНЫХ СРЕД И СРЕДСТВ РАЗРАБОТКИ

Для дальнейшего развития проекта и увеличения охвата пользовательской аудитории было решено разработать кроссплатформенное приложение, которое позволит создать один раз нативное приложение и предоставить его код в совместное использование для разных операционных систем. В процессе выбора платформы для разработки были изучены программные среды и средства, позволяющие реализовать мобильное приложение.

3.1 Сравнительный анализ имеющихся возможностей по выбору средств разработки

Определить выбор программных средств разработки можно определив среду для создания программного средства.

Среди платформ для разработки кроссплатформенных приложений можно выделить следующие:

1. Flutter – это кроссплатформенная среда разработки для создания мобильных приложений под операционные системы Android и iOS, а также веб-приложений и десктопных приложений под ОС Windows, macOS и Linux с использованием языка программирования Dart, разработанный и развиваемый корпорацией Google.

Достоинства:

- Высокая скорость приложения, разработанных на платформе Flutter, и надежность.
- Язык Dart оптимизирован и создан для разработки пользовательского интерфейса.
- Большое сообщество.
- Есть обширная и четкая документация и шаблоны проектирования.

Недостатки:

- Язык Dart больше нигде не используется, кроме платформы Flutter.
- Не постоянная спецификация API, которая может изменится в любой момент.
- Не работают некоторые нативные вещи, проблема с которыми решается костылями.
- Сложная архитектура из-за проблемы большого расхода памяти виджетами.

2. React Native – это кроссплатформенный фреймворк для создания мобильных и настольных приложений под различные операционные среды на основе JavaScript и TypeScript, созданный Facebook, Inc.

Достоинства:

- Приложения на основе React Native будут работать на всех платформах.
- Простота, легкость и удобство разработки.
- Приложения, разработанные на фреймворке React Native, по поведению и внешнему виду близки к нативным.

Недостатки:

- Отсутствуют некоторые компоненты.
- Не подходит для проектов с тяжелой и сложной графикой, анимацией.
- Нет визуального редактора интерфейса. Интерфейс прописывается в коде с помощью JSX-разметки.

3. Native Script – это фреймворк для кроссплатформенной разработки приложений на платформах Android и iOS с использованием языка JavaScript или TypeScript, разрабатываемый компанией Telerik.

Достоинства:

- Бесплатный открытый исходный код.
- Возможность интеграции любой библиотеки JavaScript.
- Обширная библиотека плагинов NativeScript

- Подробная документация.

Недостатки:

- Сложное создание пользовательского интерфейса.
- Проверенные плагины отсутствуют.
- Усложненное получение доступа к оборудованию устройства и другим специфическим функциям.
- Некоторые компоненты пользовательского интерфейса платные.

4. Xamarin.Forms – это фреймворк для кроссплатформенной разработки мобильных приложений под операционные среды iOS, Android, Windows Phone с использованием языка C#, принадлежащей компании Microsoft.

Достоинства:

- Универсальный набор технологий для разработки приложений.
- Производительность максимально близка к той, что у нативных приложений.
- Хорошая совместимость с устройствами.
- Оптимальные условия тестирования.
- Технологии с открытым исходным кодом и полный инструментарий.
- Язык C# – один из пяти самых популярных языков программирования.

Недостатки:

- Ограниченная экосистема.
- Ограниченный доступ к библиотекам.
- Не подходит для приложений с высокопроизводительной графикой.
- Вес приложений, разработанных на Xamarin.Forms, больше, чем у нативных приложений.
- Некоторые функции и интеграции не предоставляются.

Для оптимального выбора программной среды разработки выполним сравнение в таблице 15 вышеперечисленных фреймворков по четырем показателям.

Таблица 15 – Сравнение кроссплатформенных сред разработки

Критерии	Flutter	React Native	NativeScript	Xamarin. Forms
Язык программирования	Dart	JavaScript + React	JavaScript, TypeScript	C#
Платформы	Android, iOS, Google, Fuchsia, Web, Desktop	Android, iOS, UWP	Android, iOS	Android, iOS, UWP
Инструмент для проектирования UI	Есть виджеты, похожие на нативные компоненты	Есть готовые нативные компонент	Использует нативные средства рендеринга каждой платформы и язык, основанный на XML	Создается с помощью языка разметки XAML
Производительность	Очень высокая	Высокая, близкая к нативной	Высокая; уменьшается при запуске приложений	Высокая, близкая к нативной

Исходя из данных таблицы можно сделать вывод, что самым оптимальным и удовлетворительным фреймворком для разработки мобильного приложения в рамках выпускной квалификационной работы является Xamarin.Forms.

3.2 Характеристика выбранных программных сред и средств

3.2.1 Xamarin.Forms

Как отмечалось в предыдущем пункте, Xamarin.Forms – фреймворк для кроссплатформенной разработки мобильных приложений под операционные среды iOS, Android и Windows Phone.

Создание пользовательского интерфейса в Xamarin.Forms происходит с помощью языка разметки XAML, а программной части с помощью языка C#.

Дополнительно можно отметить следующие характеристики фреймворка:

- Общий API, осуществляющий отрисовку собственных элементов управления на разных платформах, позволяющие использовать функции, доступные только на определенной платформе, без создания пользовательских отрисовщиков или эффектов.
- Имеет готовые решения для различных функций, характерных для определенных платформ.
- Выборочное применение пользовательских отрисовщиков к пользовательскому интерфейсу, что обеспечивает единообразный внешний вид и поведение в iOS и Android.

3.2.2 Язык разметки XAML

XAML – расширяемый язык разметки для приложений — основанный на XML язык разметки для декларативного программирования.

С помощью XAML упрощается разработка пользовательского интерфейса для приложений платформы .NET Framework. В декларативной XAML-разметке есть возможность создавать, а затем отделять видимые элементы пользовательского интерфейса от логики их выполнения, используя файлы кода программной части, присоединенные к разметке с помощью определений разделяемых классов.

Преимущества:

- Простота и удобство реализации пользовательского интерфейса.
- Не нужно адаптировать код под разные разрешения.
- Возможность использования стилей, в которых можно менять свойства в зависимости от состояния элемента и указывать обработчики событий.
- Можно использовать шаблоны.
- Можно использовать привязку данных.
- Компиляция XAML в нативный код для любых платформ.

Недостатки:

- При реализации интерфейса получается огромный код.
- Сложность использования стилей.
- Трудности в использовании привязки.

3.2.3 Язык программирования C#

C# — объектно-ориентированный язык программирования, разработан как язык разработки приложений для платформы Microsoft .NET Framework и .NET Core.

C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, переменные, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML.

Преимущества:

- Объектно-ориентированный язык, который позволяет создавать многофункциональные приложения.
- Типы данных имеют фиксированный размер, что повышает «мобильность» языка и упрощает программирование.
- Автоматическая «сборка мусора».

- Большое количество «синтаксического «сахара» — специальных конструкций, разработанных для понимания и написания кода.
- Низкий порог вхождения.
- Наличие большого количества библиотек и шаблонов.
- Большое сообщество.

Недостатки C#:

- Приоритетная ориентированность на платформу Windows.
- Очень легко дизассемблируется, что позволяет любому желающему получить код программы.
- .NET использует концепцию JIT-компиляции. Это означает, что программа будет скомпилирована в машинные коды по мере необходимости прямо во время работы приложения.
- Слабое взаимодействие с аппаратной частью.
- C# не является повсеместно распространенным языком.

3.2.4 СУБД SQLite

Для реализации базы данных на сервере было принято решение использовать SQLite – быструю и компактную встраиваемую одно файловую СУБД на языке C, не имеющую сервера и позволяющую хранить всю базу локально на одном устройстве.

Преимущества:

- Высокая скорость работы.
- Хранение данных в одном файле, который находится на том же устройстве, что и программа.
- Простота решений и легкость администрирования.
- Надежность и минимальный риском непредсказуемого поведения.
- Нулевая конфигурация.
- Маленький размер полностью сконфигурированной базы данных со всеми настройками.

- Поддерживает большинство функций стандарта SQL2 и имеет ряд собственных.
- Доступность.
- Кроссплатформенность.
- Автономность. Для работы SQLite не нужны сторонние библиотеки или службы.

Недостатки:

- Ограниченная поддержка типов данных.
- Отсутствие хранимых процедур.
- Ограничения в применении. Без сервера возможности СУБД меньше.
- Ограничена многопоточность — единовременное выполнение нескольких процессов.
- Отсутствие бесплатной техподдержки.
- Отсутствие встроенной поддержки Unicode — популярный стандарт кодирования символов.

4 АЛГОРИТМ РЕШЕНИЯ ПОСТАВЛЕННОЙ ЗАДАЧИ

4.1 Этапы реализации ПС

Разработка мобильного приложения происходит в несколько этапов. С помощью пошаговой реализации программного средства можно определить пути решения актуальных задач и наблюдать за происходящими изменениями, что позволяет снизить риски возникновения проблем. Основные этапы разработки программного средства выглядят следующим образом:

1. Выделение требований и их анализ.
2. Проектирование. На этом этапе происходит моделирование теоретической основы.

В рамках данного этапа осуществляется:

- высокоуровневое проектирование пользовательского интерфейса, включающего в себя разработку функциональных блоков программы, функциональностью отдельных экранных форм, их взаимосвязью, проектированием навигационной системы продукта;
- низкоуровневое проектирование, заключающихся в разработке прототипов отдельных экранных форм;
- разработка базы данных.

3. Реализация мобильного приложения.
4. Тестирование и отладка, позволяющие ликвидировать огрехи программирования и добиться полнофункциональной работы разработанной программы.

4.2 Пользовательский интерфейс ПС

4.2.1 Взаимодействие пользователей с ПС

Для обеспечения возможности комфортного взаимодействия пользователя с программным средством необходимо разработать простой и понятный пользовательский интерфейс.

С этой целью была сформирована основная логика мобильного приложения на основании анализа предъявленных требований и спецификаций, а также характеристиках основных пользователей и рассмотренных всевозможных вариантов их поведения при взаимодействии с программным средством.

На рисунке 7 изображена диаграмма активности – схема движения пользователя и сценарии, которые могут возникнуть при взаимодействии с программой. Данная схема описывает предполагаемые действия и поведение потребителя и работу самой системы, а также ее состояние, что необходимо для исключения ошибок на ранних этапах работы над приложением.

При визуализации диаграммы активности были выполнены следующие действия:

- выявлены действующие лица;
- учтены точки входа и выхода пользователя;
- определен и прописаны основные точки принятия решений;
- обозначены переходы и связи между актерами.

В дальнейшем схема диаграммы активности поможет:

- разработать и реализовать понятный и удобный пользовательский интерфейс;
- проверить соответствие готового интерфейса, заявленным требованиям и целям.

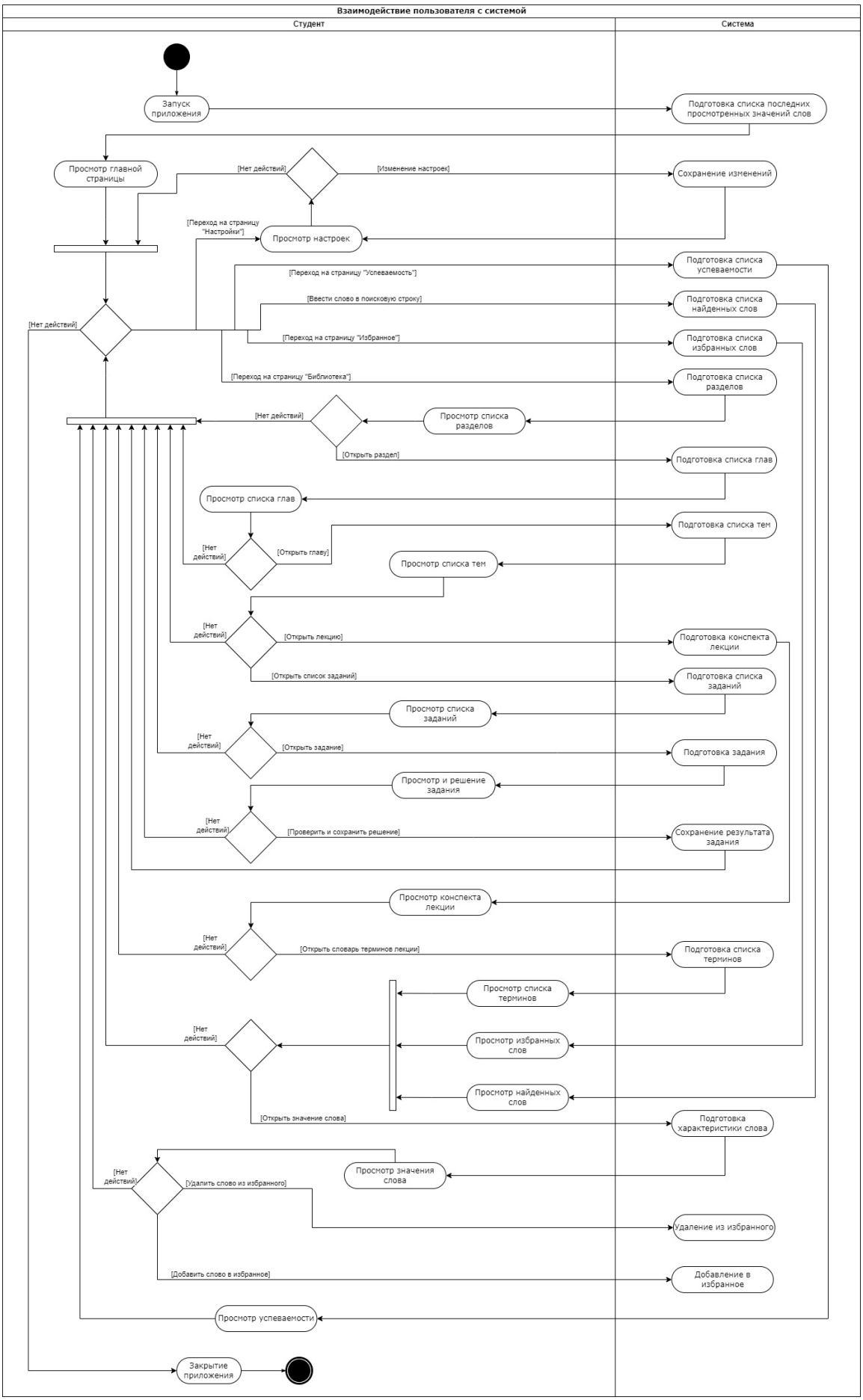


Рисунок 7 – Диаграмма активности

4.2.2 Проектирование пользовательских сценариев

Для описания функций и конкретизации ожидаемого пользователем результатов были составлены пользовательские сценарии, которые помогут рассмотреть варианты действий, выполняемые потребителями. Описанные сценарии позволяют декомпозировать разработку мобильного приложения на функциональные блоки, которые необходимо разработать, оценить их сложность и временные затраты.

Примеры пользовательских сценариев для роли «Студент»:

1) Я, как студент, хочу иметь возможность просмотреть значения терминов на русском языке, которые используются в конспекте лекции, чтобы не искать их перевод на китайский язык в словаре, тем самым сократив время, отведенное на изучение материала.

2) Я, как студент, хочу иметь возможность сохранять значение русских слов в избранное, чтобы потом вернуться к ним для более детального изучения.

3) Я, как студент, хочу иметь возможность выделения текста в конспектах лекций, чтобы сконцентрировать свое внимание и понимание информации.

4) Я, как студент, хочу иметь возможность выполнения практических заданий, чтобы проверить свое усвоение теоретического материала.

5) Я, как студент, хочу иметь возможность просмотра избранных слов, чтобы повторить их.

6) Я, как студент, хочу иметь возможность поиска и вывода перевода русского слова, чтобы не искать самостоятельно в других источниках.

Для пользовательских сценариев 1 и 4 были подготовлены UML-диаграммы последовательностей, изображенные на рисунках 8 и 9.

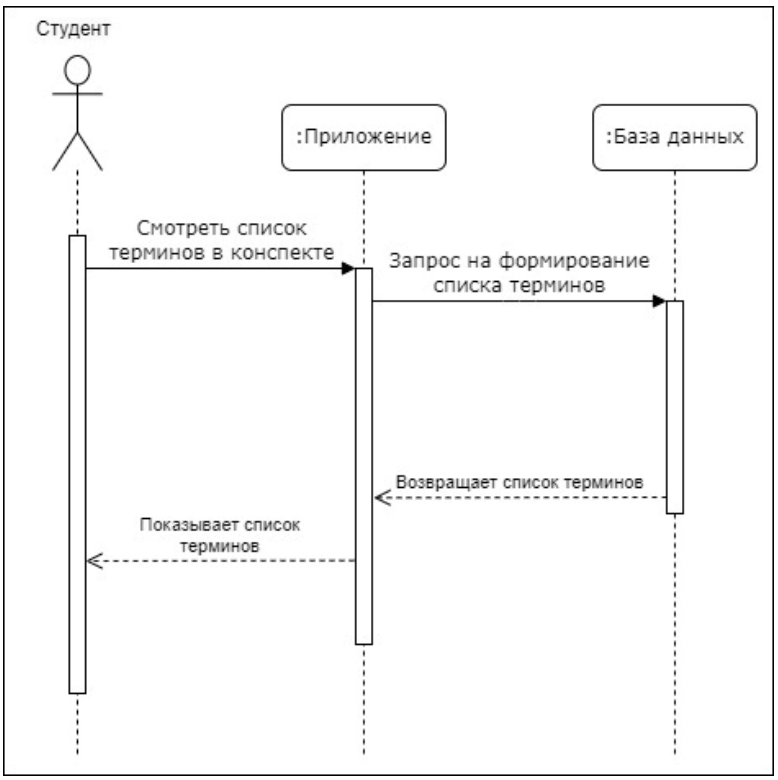


Рисунок 8 – Диаграмма последовательности для просмотра списка терминов в лекции

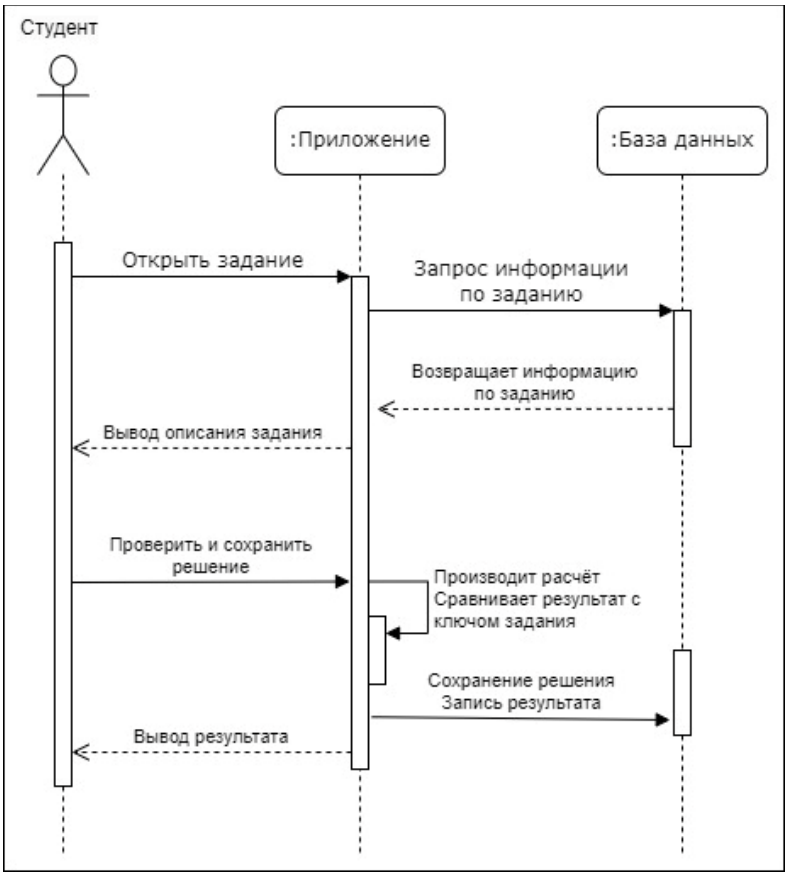


Рисунок 9 – Диаграмма последовательности для решения и проверки задания

4.2.3 Определение операций пользователей

Проанализировав предметную область и выделив профили потенциальных пользователей, можно определить операции пользователей, которые они могут выполнять в рамках используемого ими приложения:

1) открыть список разделов;

2) открыть список глав;

3) открыть список тем;

4) найти тему по названию;

5) открыть лекцию по выбранной теме;

6) открыть список используемых терминов в лекции;

7) открыть упражнение для решения;

8) найти значение русского слова на китайском языке;

9) просмотреть перевод значения русского слова на китайском языке;

10) открыть список слов в избранном;

11) добавить слово в избранное;

12) удалить слово из избранного;

13) просмотреть историю действий;

14) посмотреть успеваемость;

15) открыть список настроек;

16) изменить параметры настроек приложения;

17) вернуться на главную страницу.

4.2.4 Составление функциональных блоков

Определив операции пользователей, можно выделить функциональные блоки исходя из того, с какой информацией он работает.

Таким образом, было выделено четыре функциональных блока соответствующие работе:

- со словарем;
- с учебным материалом;
- с оценками;

- с характеристиками приложения.

В функциональный блок, работающий с характеристиками приложением, входят следующие операции:

15) открыть список настроек;

16) изменить параметры настроек приложения;

17) вернуться на главную страницу.

В функциональный блок, работающий с оценками, входят следующие операции:

1) открыть список разделов;

10) открыть список слов в избранном;

14) посмотреть успеваемость;

17) вернуться на главную страницу.

В функциональный блок, работающий с учебным материалом, входят следующие операции:

1) открыть список разделов;

2) открыть список глав;

3) открыть список тем;

4) найти тему по названию;

5) открыть лекцию по выбранной теме;

6) открыть список используемых терминов в лекции;

7) открыть упражнение для решения;

10) открыть список слов в избранном;

17) вернуться на главную страницу.

В функциональный блок, работающий со словарем, входят следующие операции:

1) открыть список разделов;

8) найти значение русского слова на китайском языке;

9) просмотреть перевод значения русского слова на китайском языке;

10) открыть список слов в избранном;

11) добавить слово в избранное;

12) удалить слово из избранного;

13) просмотреть историю действий;

14) посмотреть успеваемость;

15) открыть список настроек;

17) вернуться на главную страницу.

4.2.5 Проектирование структуры экранов ПС и схемы навигации

После выполнения анализа предметной области, формирования профилей и сценарий действий пользователей, а также определения функциональности приложения была составлена схема навигационной системы, которая изображена на рисунке 10.

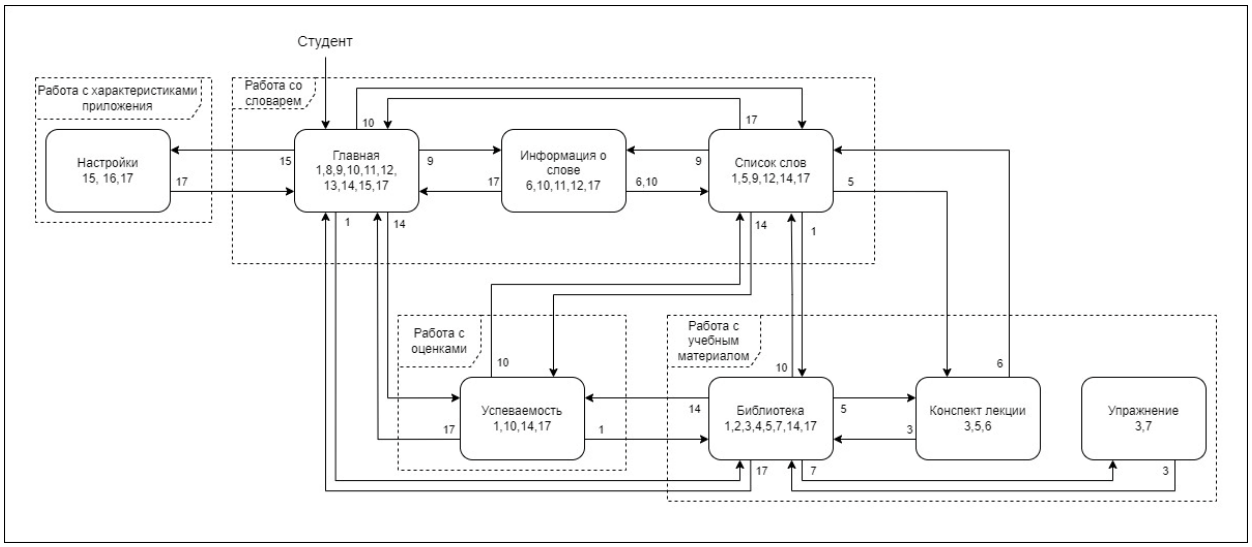


Рисунок 10 – Схема навигации

В данной схеме четыре функциональных блока и восемь экранных форм:

- Главная.
- Избранное.
- Список слов.
- Информация о слове.
- Настройки.
- Успеваемость.

- Библиотека.
- Конспект лекции.
- Упражнение.

Цифрами на рисунке указаны отдельные операции, выполняемые пользователями.

4.2.6 Низкоуровневое проектирование

После прохождения этапа предварительного и высокоуровневого проектирования пользовательского интерфейса в редакторе Figma были подготовлены не детализованные макеты оформления экранов приложения, изображенные на рисунке 11.

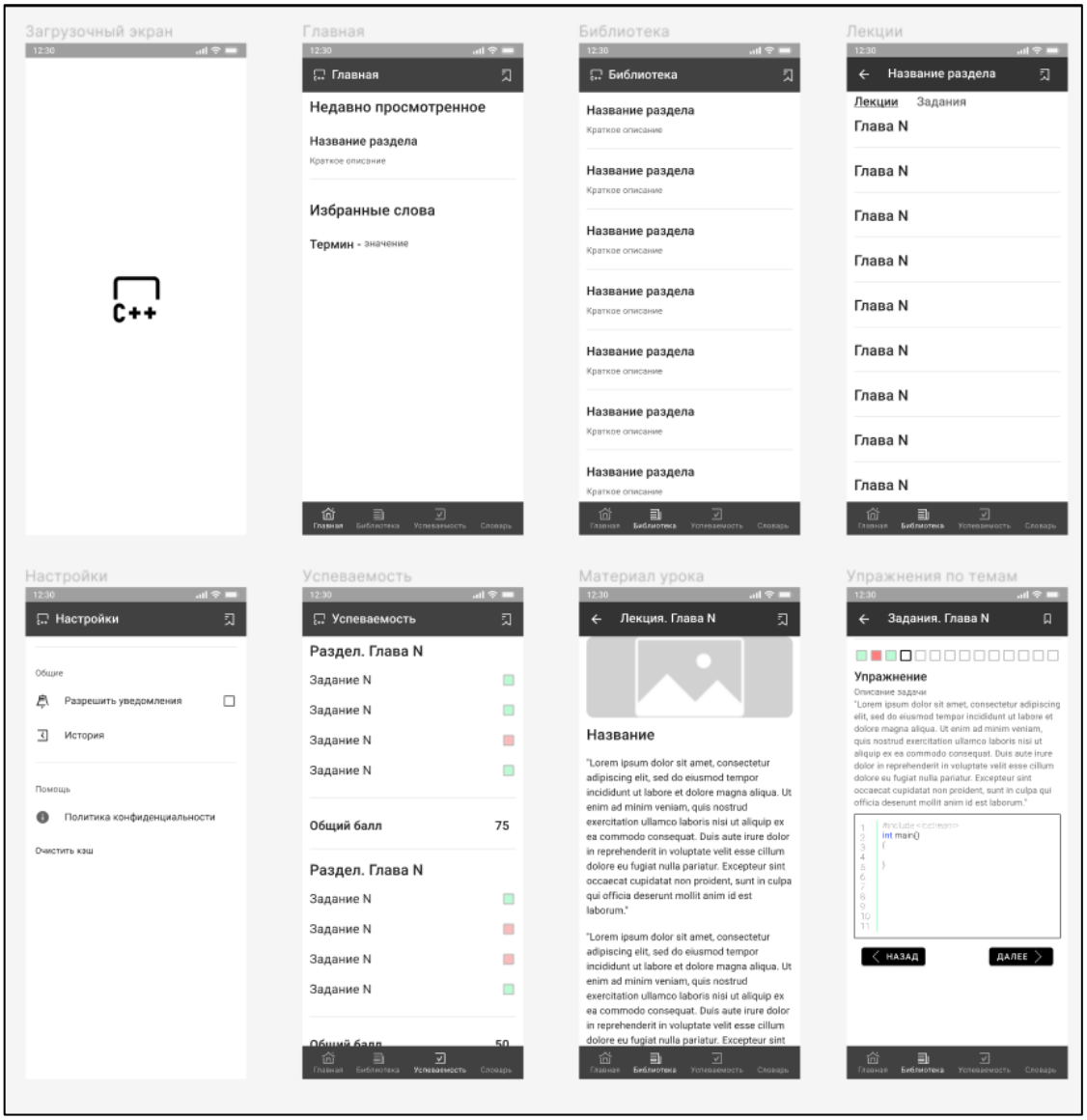


Рисунок 11 – Макеты приложения

На макетах представлены восемь экранных форм:

- Загрузочный экран.
- Главная.
- Библиотека.
- Список лекций.
- Настройки.
- Успеваемость.
- Конспект лекции.
- Упражнение.

Данные макеты предназначены для представления структуры информации и расположения объектов в мобильной программе.

4.3 Входные, выходные и промежуточные данные

Впоследствии рассмотрения предметной области, определения и описания характеристик и сценарий действий пользователей и их операции можно выявить входные и выходные данные программы.

Входные данные – величины, которые задаются пользователем до начала работы алгоритма или определяются динамически во время его работы. Входные данные берутся из определенного набора объектов.

Таким образом, входными данными приложения являются:

- вводимая пользователем текстовая информация, которая будет использоваться при решении задачи;
- вводимый пользователем параметр для поиска информации;
- данные и информация, хранимая в базе данных.

На рисунке 12 изображен пример входных данных, представленные в виде текстовой информации, введенной во время решения задачи.

```
#include <iostream>

int main() {
    std::cout << "Hello World!";
    return 0;
}
```

Рисунок 12 – Пример решения задачи

Выходные данные – это данные, получаемые в результате решения какой-либо задачи или алгоритма, а также в ходе работы с учебными материалами и русско-китайским словарем.

Следовательно, выходными данными являются:

- вывод результата поиска;
- вывод информации о значениях слов;
- графический и текстовый вывод лекционного материала;
- вывод описания задания;
- вывод результата решения задания;
- вывод информации об успеваемости.

На рисунке 13 изображен пример выходных данных, которые представляют собой содержание конспекта лекции «Конструкции циклов», где, помимо текстового содержимого, присутствует визуальный контент в виде схемы.

Другие примеры входных данных представлены в Приложении Б.

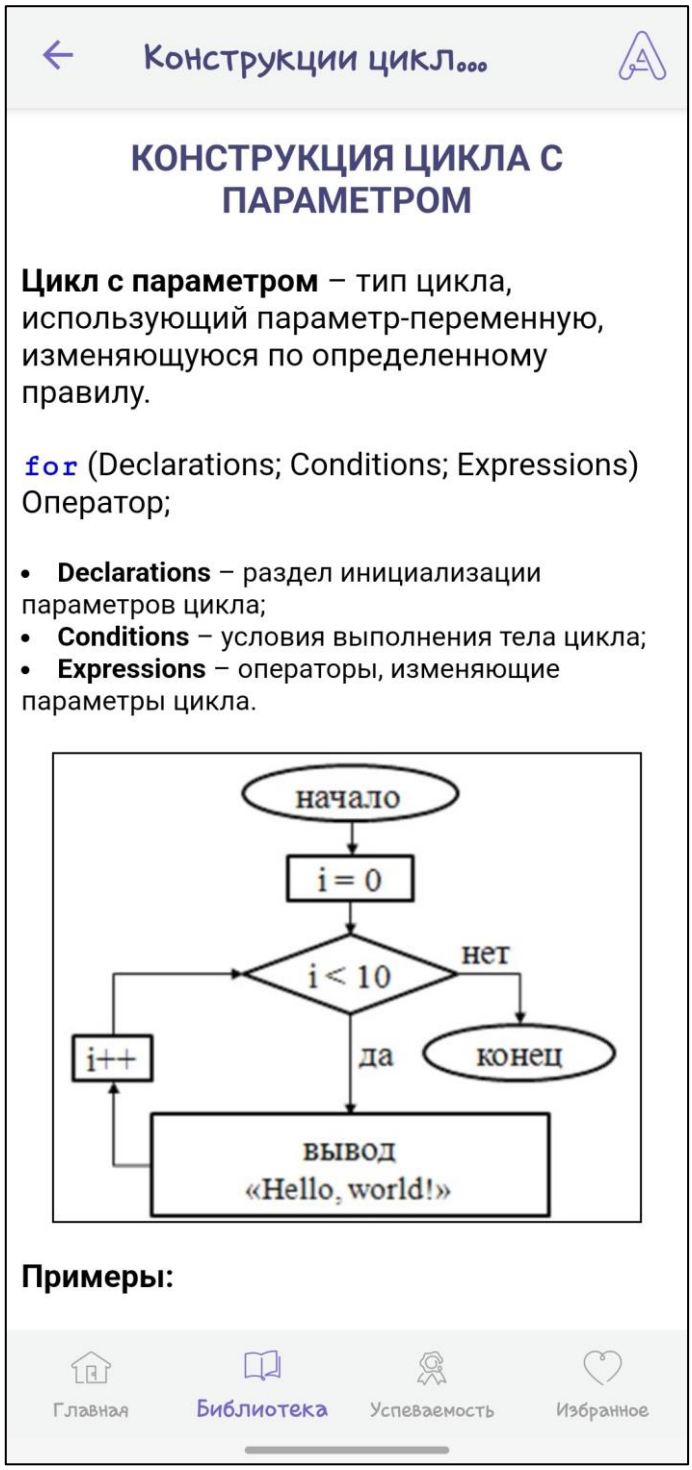


Рисунок 13 – Вывод конспекта лекции

4.4 Разработка базы данных, реализуемой в рамках ПС

Исходя из разработанной структуры, возникает необходимость в хранении данных информационного материала для функционирования приложения. На рисунке 14 приведена схема базы данных приложения.

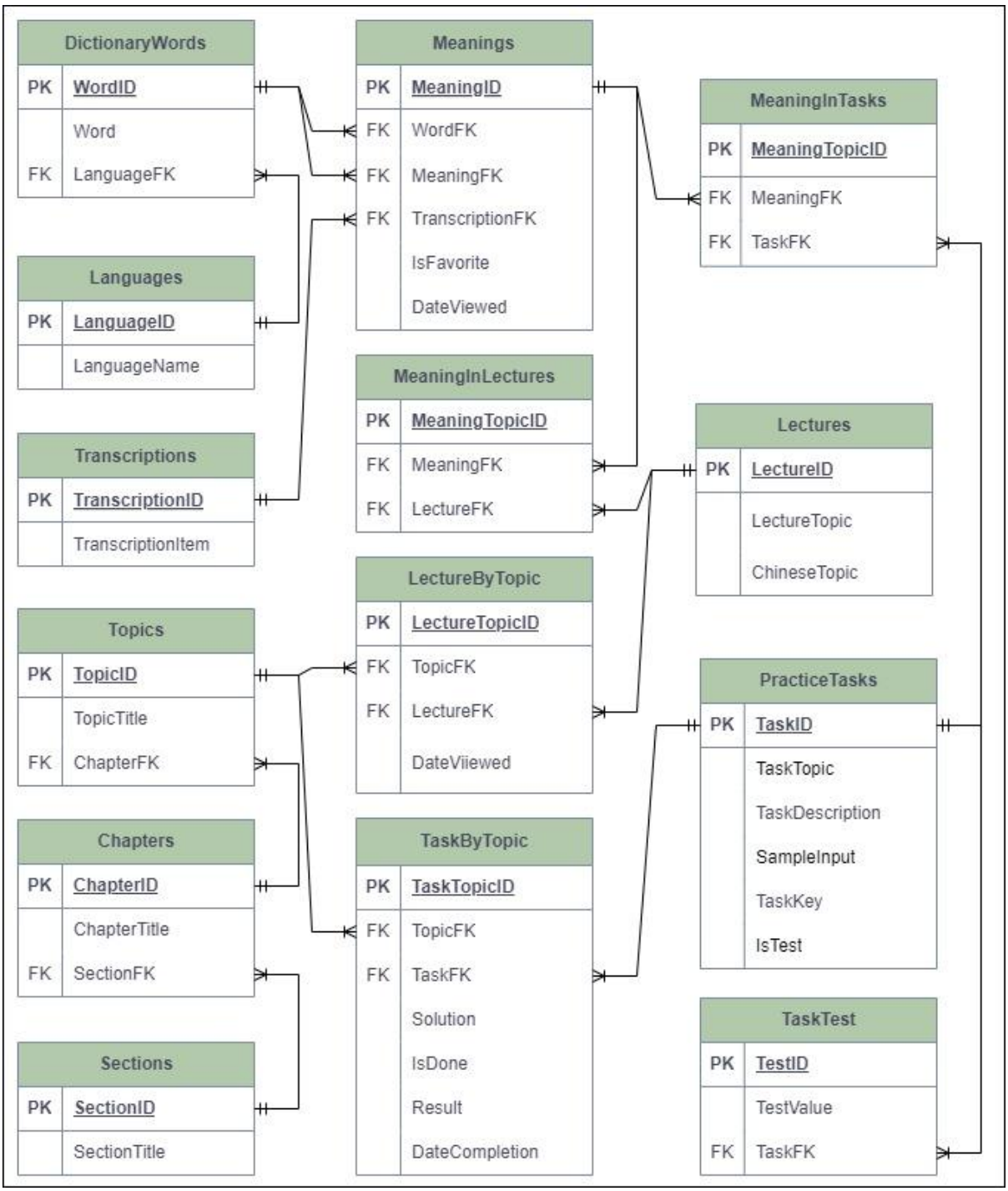


Рисунок 14 – Схема пользовательской базы данных

База данных состоит из четырнадцати таблиц: DictionaryWords, Languages, Transcriptions, Meanings, MeaningInLectures, MeaningInTasks, Sections, Chapters, Topics, Lectures, PracticeTasks, TaskTest, TaskByTopic, LectureByTopic.

Между таблицами на схеме базы данных имеются следующие характеристики связей:

- «Languages» и «DictionaryWords» имеют связь 1:M, т.к. один язык может иметь множество слов.
- «DictionaryWords» и «Meanings» имеют связь 1:M, т.к. одно слово имеет множество значений и одно слово означает множество значений слов.
- «Transcriptions» и «Meanings» имеют связь 1:M, т.к. одна транскрипция имеет множество значений слов.
- «Meanings» и «MeaningInLecture» имеют связь 1:M, т.к. одно значение слова может быть во множестве лекций.
- «Lectures» и «MeaningInLecture» имеют связь 1:M, т.к. в одной лекции содержится множество значений слов.
- «Meanings» и «MeaningInTasks» имеют связь 1:M, т.к. одно значение слова может быть во множестве заданий.
- «PracticeTasks» и «MeaningInTasks» имеют связь 1:M, т.к. в одном задании содержится множество значений слов.
- «Sections» и «Chapters» имеют связь 1:M, т.к. один раздел может иметь множество глав.
- «Chapters» и «Topics» имеют связь 1:M, т.к. одна глава может иметь множество тем.
- «Lectures» и «LectureByTopic» имеют связь 1:M, т.к. в одна лекция может быть во множестве тем.
- «Topics» и «LectureByTopic» имеют связь 1:M, т.к. в одна тема может иметь множество лекций.
- «PracticeTasks» и «TaskByTopic» имеют связь 1:M, т.к. в одно задание может быть во множестве тем.
- «PracticeTasks» и «TaskTest» имеют связь 1:M, т.к. в одно задание может иметь множество вариантов ответов в тесте.
- «Topics» и «TaskByTopic» имеют связь 1:M, т.к. в одна тема может иметь множество заданий.

В таблице 16 представлено описание таблицы «DictionaryWords».

Таблица 16 – Поля таблицы «DictionaryWords»

Название поля	Тип поля	Описание
WordID	INTEGER	Идентификатор
Word	TEXT	Слова и выражения
LanguageFK	INTEGER	Внешний ключ к таблице Languages

Таблица «DictionaryWords» определяет набор или словарь слов, изучение которых возможно в данном приложении.

В таблице 17 представлено описание таблицы «Languages».

Таблица 17 – Поля таблицы «Languages»

Название поля	Тип поля	Описание
LanguageID	INTEGER	Идентификатор
LanguageName	TEXT	Язык

Таблица «Language» представляет собой набор языков. Благодаря данному решению возможно дальнейшее расширение функциональности проекта.

В таблице 18 представлено описание таблицы «Transcriptions».

Таблица 18 – Поля таблицы «Transcriptions»

Название поля	Тип поля	Описание
TranscriptionID	INTEGER	Идентификатор
TranscriptionItem	TEXT	Транскрипция, чтение

Таблица «Transcriptions».» представляет собой набор транскрипций и чтений слов.

В таблице 19 представлено описание таблицы «Sections».

Таблица 19 – Поля таблицы «Sections»

Название поля	Тип поля	Описание
SectionID	INTEGER	Идентификатор
SectionTitle	TEXT	Название раздела

Таблица «Sections» представляет собой список разделов.

В таблице 20 представлено описание таблицы «Chapters».

Таблица 20 – Поля таблицы «Chapters»

Название поля	Тип поля	Описание
ChapterID	INTEGER	Идентификатор
ChapterTitle	TEXT	Название главы
SectionFK	INTEGER	Внешний ключ к таблице Sections

Таблица «Chapters» представляет собой набор глав в разделе.

В таблице 21 представлено описание таблицы «Topics».

Таблица 21 – Поля таблицы «Topics»

Название поля	Тип поля	Описание
TopicID	INTEGER	Идентификатор
TopicTitle	TEXT	Название тем
ChapterFK	INTEGER	Внешний ключ - Chapters

Таблица «Topics» используется для возможности разделение лекционных материалов и практических заданий по темам

В таблице 22 представлено описание таблицы «Meanings».

Таблица 22 – Поля таблицы «Meanings»

Название поля	Тип поля	Описание
MeaningID	INTEGER	Идентификатор
ItemWordFK	INTEGER	Внешний ключ - DictionaryWords
ItemMeaningFK	INTEGER	Внешний ключ - DictionaryWords
TranscriptionFK	INTEGER	Внешний ключ – Transcription
IsFavorite	BOOLEAN	Избранное
DateViewed	TEXT	Дата просмотра

Таблица «Meanings» позволяет хранить информацию о чтении слов и их значений на другом языке.

В таблице 23 представлено описание таблицы «Lectures».

Таблица 23 – Поля таблицы «Lectures»

Название поля	Тип поля	Описание
LectureID	INTEGER	Идентификатор
LectureTopic	TEXT	Заголовок лекции
ChineseTopic	TEXT	Заголовок лекции на китайском

Таблица «Lectures» позволяет хранить названия лекций на русском и на китайском языках.

В таблице 24 представлено описание таблицы «PracticeTasks».

Таблица 24 – Поля таблицы «PracticeTasks»

Название поля	Тип поля	Описание
TaskID	INTEGER	Идентификатор
TaskTopic	TEXT	Заголовок задачи
TaskDescription	TEXT	Описание задачи
SampleInput	Text	Входные данные задачи
TaskKey	TEXT	Ожидаемый результат задачи
IsTest	BOOLEAN	Тестовое задание

Таблица «PracticeTasks» хранит в себе информацию о заданиях.

В таблице 25 представлено описание таблицы «TaskTest».

Таблица 25 – Поля таблицы «TaskTest»

Название поля	Тип поля	Описание
TestID	INTEGER	Идентификатор
TestValue	INTEGER	Вариант ответа
TaskFK	INTEGER	Внешний ключ - PracticeTasks

Таблица «TaskTest» позволяет хранить информацию о вариантах ответов в тестовом задании.

В таблице 26 представлено описание таблицы «LectureByTopic».

Таблица 26 – Поля таблицы «LectureByTopic»

Название поля	Тип поля	Описание
LectureByTopicID	INTEGER	Идентификатор
TopicFK	INTEGER	Внешний ключ - Topics
LectureFK	INTEGER	Внешний ключ - Lectures
DateViewed	TEXT	Дата просмотра

Таблица «LectureByTopic» позволяет хранить информацию о чтении китайских слов и их значений на русском языке в тематической группе.

В таблице 27 представлено описание таблицы «TaskByTopic».

Таблица 27 – Поля таблицы «TaskByTopic»

Название поля	Тип поля	Описание
TaskByTopicID	INTEGER	Идентификатор
TopicFK	INTEGER	Внешний ключ - Topic
TaskFK	INTEGER	Внешний ключ – PracticeTasks
Solution	TEXT	Решение задачи
IsDone	BOOLEAN	Статус выполнение
Result	BOOLEAN	Статус результата
DateCompletion	TEXT	Дата решения задачи

Таблица «TaskByTopic» позволяет хранить информацию о заданиях по теме.

В таблице 28 представлено описание таблицы «MeaningInLecture».

Таблица 28 – Поля таблицы «MeaningInLecture»

Название поля	Тип поля	Описание
MeaningInLectureID	INTEGER	Идентификатор
MeaningFK	INTEGER	Внешний ключ - Meanings
LectureFK	INTEGER	Внешний ключ - Lectures

Таблица «MeaningInLecture» хранит в себе информацию о значении слов, которые присутствуют в конспекте лекции.

В таблице 29 представлено описание таблицы «MeaningInTasks».

Таблица 29 – Поля таблицы «MeaningInLecture»

Название поля	Тип поля	Описание
MeaningInTaskID	INTEGER	Идентификатор
MeaningFK	INTEGER	Внешний ключ - Meanings
TaskFK	INTEGER	Внешний ключ – PracticeTasks

Таблица «MeaningInTasks» хранит в себе информацию о значении слов, которые присутствуют в описании задачи.

Таким образом, разработана схема данных, которая соответствует 3 нормальной форме и обеспечивает необходимую функциональность реализуемой программы.

4.5 Алгоритм реализации вычисления общей оценки

В мобильном приложении имеется возможность выполнения практических заданий для контроля усвоения теоретического материала путём получения результата решения – оценки выполнения задачи:

- задача решена верно;
- задача решена неверно.

В дальнейшем пользователь может просмотреть результаты решения задач и общую оценку по теме на странице «Успеваемость», что поможет ему оценить себя и понять, в каких темах необходимо восполнить пробелы в знаниях.

Для того, чтобы посчитать общую оценку за выполнение практических заданий по теме используется простая математическая модель оценки уровня знаний, описанная в пункте 2.1.3. На основе формулы (1) данной модели был разработан алгоритм расчета общей оценки, изображенный на рисунке 15.

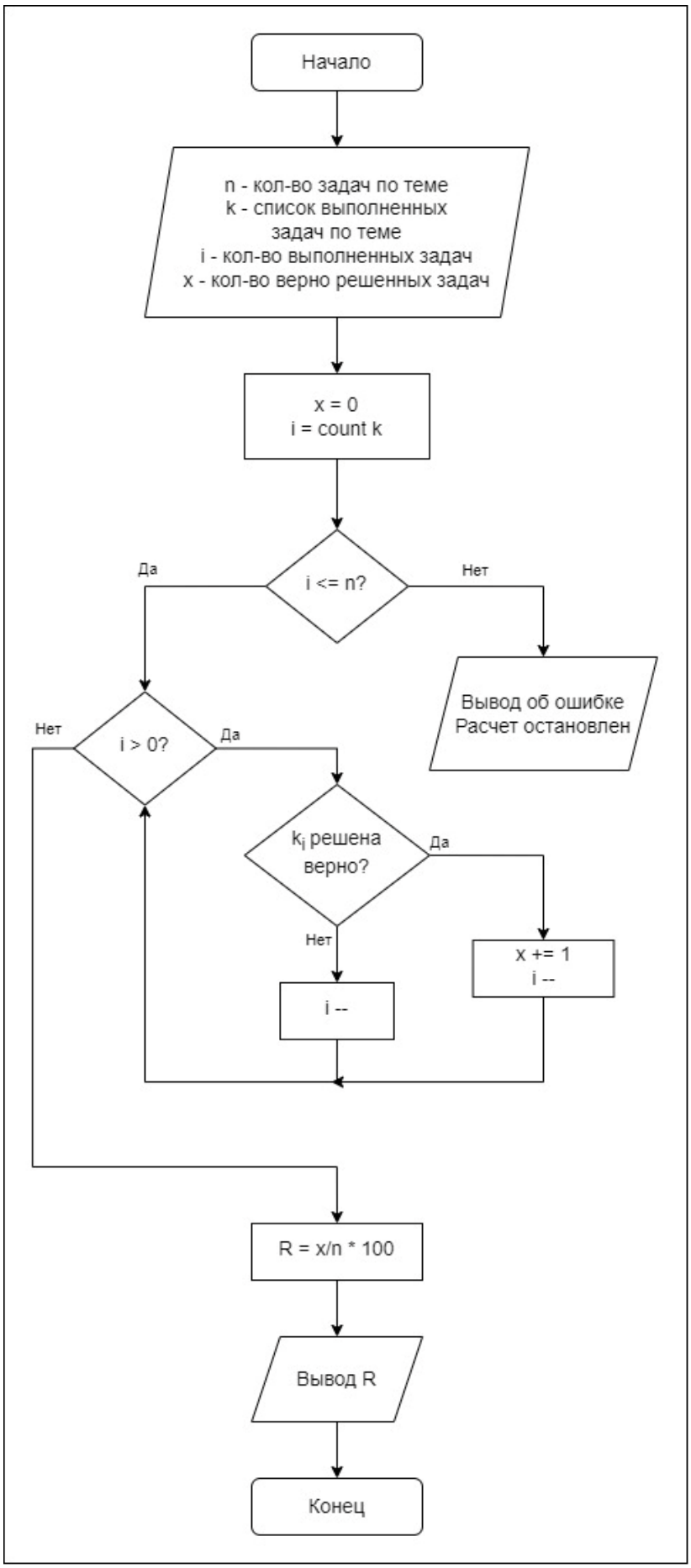


Рисунок 15 – Алгоритм расчета успеваемости

В начале выполнения алгоритма задаются четыре переменных:

- n – количество задач по заданной теме;
- k – список выполненных задач по заданной теме;
- i – количество выполненных задач в списке k;
- x – количество верно решенных задач из списка k.

Далее идет проверка на то, что количество выполненных задач не превышала общее количество заданий, иначе расчет будет невозможно выполнить.

После прохождения проверки идет подсчет количества верно решенных задач из предоставленного списка, пока он не закончится.

В конце рассчитывается общая оценка путем деления количества решенных задач на общее количество заданий в выбранной теме и умножения полученного результата на 100, так как в Китае студенты оцениваются по стобалльной шкале.

4.6 Архитектура и схема функционирования ПС

Было принято решение использовать в качестве архитектуры мобильного приложения шаблон проектирования MVVM, который имеет три основных компонента: модель, представление и модель представления. Данная архитектура позволяет отделить саму логику программного средства от его визуальной части, что позволяет вносить изменение в один компонент, не затрагивая другие, упрощая разработку и тестирование.

Model – компонент модели не содержит никакой логики, связанной с отображением данных и взаимодействием с визуальными элементами управления, а отвечает только за хранение данных приложения и её бизнес-логику.

View – компонент представления определяет внешний вид и структуру пользовательского интерфейса и не содержит бизнес-логику приложения. Данный компонент формируется в XAML и может пользоваться такими

ресурсами, как стили и шрифты. Само представление не обрабатывает события, а только выполняет действия за счет команд.

ViewModel – компонент модели представления отвечает за связывание компонентов представления и модели за счет привязки данных. Модель представления может использовать те же ресурсы, что и представление. Компонент выполняет следующие функции:

- приводит в исполнение команды и свойства, к которым привязаны данные представления;
- сообщает представлению о всех изменениях состояния;
- обеспечивает логику получения данных из модели, которые потом передаются в представление;
- отвечает за логику обновления данных в модели.

Для полного представления работы приложения с данной архитектуры была подготовлена диаграмма компонентов, изображенная на рисунке 16.

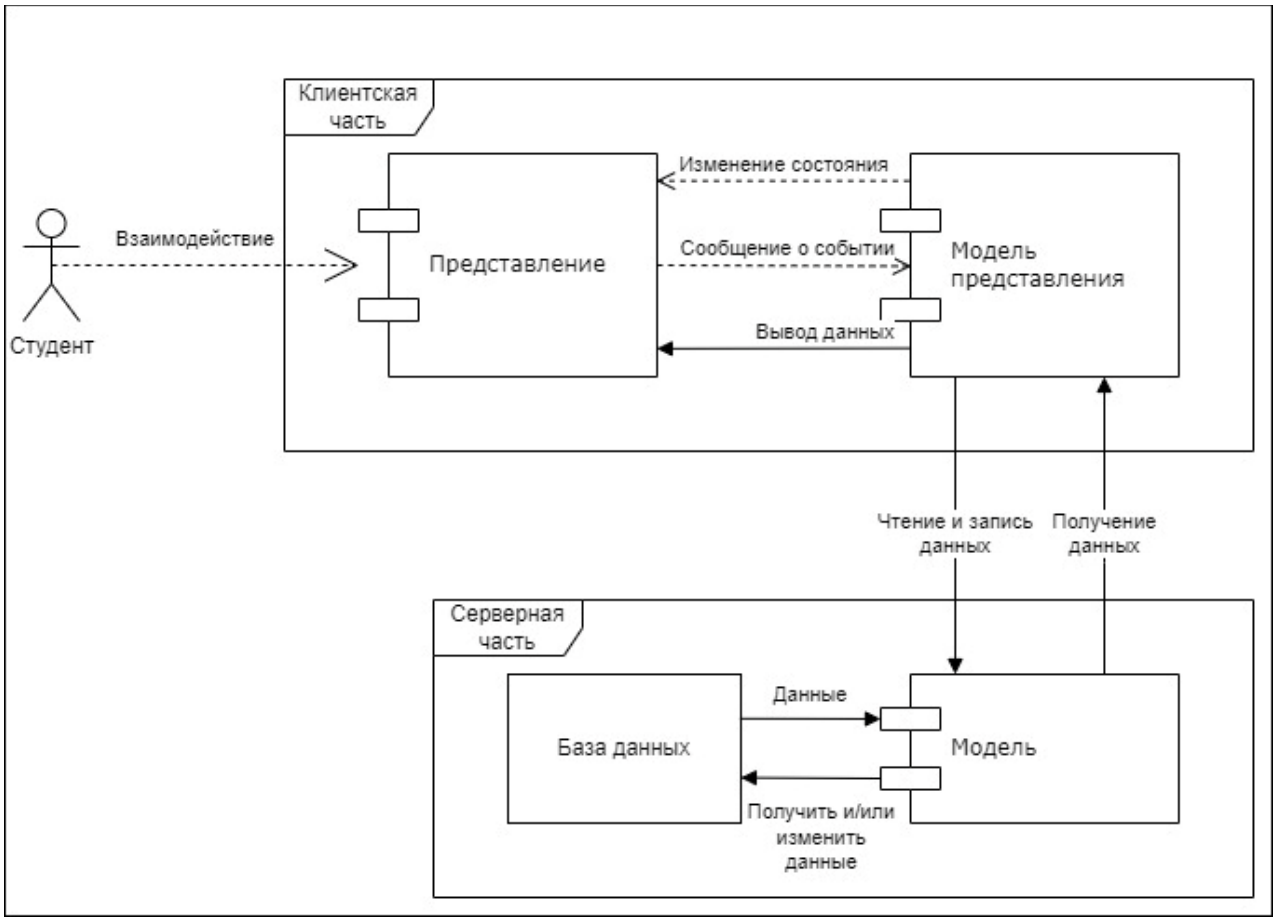


Рисунок 16 – Диаграмма компонентов приложения

В создании приложения основной задачей стоит разработка пользовательских классов, которые входят в состав вышеуказанных компонентов.

App – класс, определяющий логику и ресурсы всего программного средства и с которого начинается выполнение приложения.

Компонент представления содержит следующие классы:

- AppShell – класс визуального представления навигации в приложении.
- MainPage – класс, содержащий визуальное представление для страницы «Главная».
- LibraryTabPage – класс, содержащий визуальное представление для страницы «Библиотека».
- LectureListPage – класс, содержащий визуальное представления для вкладки «Лекции».
- LecturePage – класс, содержащий визуальное представление для страницы «Конспект лекции».
- MeaningInLecturePage – класс, содержащий визуальное представление для страницы с терминами лекции.
- TaskListPage – класс, содержащий визуальное представления для вкладки «Задания».
- TaskPage – класс, содержащий визуальное представление для страницы «Упражнение».
- MeaningInTaskPage – класс, содержащий визуальное представление для страницы с терминами, которые используются в задаче.
- ProgressPage – класс, содержащий визуальное представление для страницы «Успеваемость».
- WordsPage – класс, содержащий визуальное представление для страницы «Список слов».
- WordDetailPage – класс, содержащий визуальное представление для страницы «Информация о слове».

- SettingPage – класс, содержащий визуальное представление для страницы «Настройки».

Следующие классы содержатся в компоненте модели представления:

- BaseViewModel – класс базовой реализации модели представления, от которой наследуются все остальные классы модели представления.
- MainViewModel – класс модели представления для страницы «Главная».
- LectureListViewModel – класс модели представления для вкладки «Лекции».
- LectureViewMoel – класс модели представления для страницы «Конспект лекции».
- MeaningInLectureViewModel – класс модели представления для страницы с терминами лекции.
- TaskListViewModel – класс модели представления для вкладки «Задания».
- TaskViewModel – класс модели представления для страницы «Упражнение».
- MeaningInTaskViewModel – класс модели представления для страницы с терминами, которые используются в задаче.
- ProgressViewModel – класс модели представления для страницы «Успеваемость».
- WordsViewModel – класс модели представления для страницы «Список слов».
- WordDetailViewModel – класс модели представления для страницы «Информация о слове».
- SettingViewModel – класс модели представления для страницы «Настройки».

В компонент модели входят нижеприведенные классы:

- DatabaseRepository – класс репозитория, через который происходят все операции с данными.
- Language – класс объекта «Язык».
- Dictionary – класс объекта «Словарь».
- Transcription – класс объекта «Транскрипция».
- Meaning – класс объекта «Значение».
- MeaningInLecture – класс объекта «Значение в лекции».
- MeaningInTask – класс объекта «Значение в задаче».
- Lecture – класс объекта «Лекция».
- LectureByTopic – класс объекта «Лекция по теме».
- PracticeTask – класс объекта «Задание».
- TaskTest – класс, дополняющий объект «Задание».
- TaskByTopic – класс объекта «Задание по теме».
- Topic – класс объекта «Тема».
- Chapter – класс объекта «Глава».
- Section – класс объекта «Раздел».

Диаграммы классов по компонентам представлены в Приложении В.

5 ТЕСТИРОВАНИЕ И ОПТИМИЗАЦИЯ

5.1 План тестирования

После того, как программное средство было реализовано, необходимо его протестировать на работоспособность перед тем, как вводить приложение в эксплуатацию. Это необходимо для того, чтобы выявить ошибки (баги), недостатки, проблемы и уязвимости приложения.

Следует учитывать, что существует множество инструментов, методов и видов тестирования, а их выполнение делят на ручное и автоматизированное. Исходя из функциональных и нефункциональных требований было решено провести следующие виды тестирования:

- функциональное тестирование – тестирование на соответствие функциональным требованиям;
- тестирование производительности – измерение показателей скорости реакции приложения и потребления ресурсов на внешние воздействия в разных ситуациях;
- тестирование удобство пользования – проверка того, насколько конечному пользователю понятно и удобно использовать продукт;

Как следует из раздела 2.3 Функциональные требования и диаграммы вариантов использования, изображенной на рисунке 6 в вышеуказанном разделе тестирование будет производится по данным пользовательским требованиям (далее ПТ):

- ПТ-1: Установка и удаление приложения.
    - ПТ-1.1: Установка приложения производится через файл приложения в формате APK.
    - ПТ-1.2: Удаление приложения производится посредством долгого нажатия на иконку программы на мобильном устройстве и последующего нажатия на кнопку «Удалить» в открывшемся контекстном меню.

- ПТ-2: Запуск приложения производится посредством нажатия на иконку программы на мобильном устройстве.
- ПТ-3: Работоспособность основного функционала:
  - ПТ-3.1: В процессе работы приложения пользователь имеет возможность перемещаться между основными страницами посредством нажатия на иконки страниц в главном меню, которое всегда располагается в нижней части страницы.
  - ПТ-3.2: В процессе работы программы пользователь имеет возможность вводить данные в поисковую строку на странице «Главная» приложения для поиска слов в русско-китайском словаре или конспекта лекций.
  - ПТ-3.3: В процессе работы приложения пользователь имеет возможность на странице «Главная» просмотреть историю недавно просмотренных слов и тем.
  - ПТ-3.4: В процессе работы приложения пользователь имеет возможность на странице «Библиотека» перейти во вкладку «Лекции», где может выбрать интересующий раздел и тему для просмотра конспекта лекции.
  - ПТ-3.5: В процессе работы приложения пользователь имеет возможность на странице «Библиотека» перейти во вкладку «Задания», где может выбрать интересующий раздел и тему для просмотра заданий и начала их выполнения.
  - ПТ-3.6.: В процессе работы приложения пользователь имеет право при просмотре конспекта лекции перейти на страницу со списком слов и их переводом посредством нажатия на кнопку «Термины» на странице конспекта лекции.
  - ПТ-3.7: В процессе работы приложения пользователь имеет возможность на странице «Избранное» просмотреть список избранных слов.

- ПТ-3.8: В процессе работы приложения пользователь имеет возможность при выполнении ПТ-3.2, ПТ-3.6 и/или ПТ-3.7 нажать на интересующее слово и перейти на страницу со значением слова.
- ПТ-3.9: В процессе работы приложения пользователь имеет возможность при выполнении ПТ-3.7 нажать на кнопку «Сохранить» для того, чтобы сохранить слово в избранное.
- ПТ-3.10: В процессе работы приложения пользователь имеет возможность при выполнении ПТ-3.7 нажать на кнопку «Удалить» для того, чтобы удалить слово из избранного.
- ПТ-3.11: В процессе работы приложения пользователь имеет возможность на странице «Успеваемость» просмотреть свою успеваемость.
- ПТ-3.12: В процессе работы приложения пользователь имеет возможность на странице «Главная» нажать на иконку страницы «Настройки» для открытия и просмотра страницы настроек.

Исходя из раздела 2.4 Нефункциональные требования был выделен следующий атрибут качества (далее АК):

- АК-1: Время запуска приложения не должно превышать 2 секунд.
- АК-2: Среднее время отклика приложения на действия пользователей не должна превышать более 0,5 секунд.

Созданные на основе вышеуказанных требований детальные спецификации (далее ДС) имеют следующий вид:

- ДС-1: Приложение должно функционировать на операционной системе Android с версией не ниже 8.0.
- ДС-2: Окно приложения должно адаптироваться под все виды разрешений экрана.
- ДС-3: В приложении должна быть только вертикальная ориентация.

- ДС-4: Сообщения об ошибках не должны содержать техническую информацию и должны предлагать пользователям чёткий алгоритм дальнейших действий.

Таким образом, области, подвергаемые тестированию, имеют следующий вид:

- ПТ-*: дымовой тест, тест критического пути.
- ДС-*: дымовой тест, тест критического пути.

Области, не подвергаемые тестированию:

- АК-*: заявленная характеристика находится вблизи нижней границы, характерных для разрабатываемого приложения.

Следует отметить, что специфика работы приложения подразумевает исследование вопросов удобства использования и безопасности в процессе тестирования.

Уровни функционального тестирования:

- Дымовой тест выполняется вручную и автоматизировано, т.к. проверяет ключевые функциональности, неработоспособность которых делает бессмысленной саму идею использования приложения.
- Тест критического пути выполняется вручную, т.к. исследует функциональность приложения, используемого пользователем в повседневной деятельности.

Критерии тестирования:

- Приложение считается готовым к приемке при успешном прохождении 100% тест-кейсов уровня дымового тестирования и 90% тест-кейсов уровня критического пути и при условии устранения 100% дефектов критической и высокой важности.
- Переход к тесту критического пути допустим только при успешном прохождении 100% тест-кейсов дымового теста.
- Тестирование считается завершенным при выполнении более 80% запланированных тест-кейсов.

Программные ресурсы:

- Эмулятор Pixel 5 – API 33 1GB Android 13.
- Эмулятор Nexus 6P – API 24 1GB Android 7.

Аппаратные ресурсы:

- Смартфон Samsung Galaxy S20 128GB Android 13.
- Смартфон Honor 8X 128GB Android 10.

Метрики:

- Успешное прохождение тест-кейсов:

$$\boldsymbol{T^{SP}} = \frac{\boldsymbol{T^{Success}}}{\boldsymbol{T^{Total}}} \cdot \mathbf{100}\%, \quad (2)$$

где $\boldsymbol{T^{SP}}$ — процентный показатель успешного прохождения тест-кейсов;

$\boldsymbol{T^{Success}}$ — количество успешно выполненных тест-кейсов;

$\boldsymbol{T^{Total}}$ — общее количество выполненных тест-кейсов.

- Выполнение тест-кейсов:

$$\boldsymbol{T^{E}} = \frac{\boldsymbol{T^{Executed}}}{\boldsymbol{T^{Planned}}} \cdot \mathbf{100}\%, \quad (3)$$

где $\boldsymbol{T^{E}}$ — процентный показатель выполнения тест-кейсов;

$T^{Executed}$ — количество выполненных тест-кейсов;

$T^{Planned}$ — количество тест-кейсов, запланированных к выполнению.

- Покрытие требований тест-кейсами:

$$\boldsymbol{R^{C}} = \frac{\boldsymbol{R^{Covered}}}{\boldsymbol{R^{Total}}} \cdot \mathbf{100}\%, \quad (4)$$

где $\boldsymbol{R^{C}}$ — процентный показатель покрытия требования тест-кейсами;

$R^{Covered}$ — количество покрытых тест-кейсами требований;

$R^{Covered}$ — общее количество требований.

5.2 Результаты тестирования

Для тестирования мобильного приложения были составлены и расписаны 28 тест-кейсов на основе требований из тест-плана, где 16 тест-кейсов дымового тестирования и 12 тест-кейсов тестирования критического пути. Для удобства тест-кейсы были распределены по следующим модулям:

- Установка и удаление приложения.
- Работа со страницами и формами.
- Работа со словарем.
- Работа с лекционным материалом.
- Работа с практическим материалом.

В таблице 30 представлены результаты прохождения тест-кейсов по вышеперечисленным модулям. Подробное описание каждого тест-кейса представлено в Приложении Г.

Таблица 30 – Результаты тест-кейсов

№ Тест-кейса	Название проверки	Результат
Модуль 1: Установка и удаление приложения		
1	Проверка установки мобильного приложения	Пройден
2	Проверка удаления мобильного приложения	Пройден
Модуль 2: Работа со страницами и формами		
3	Проверка запуска мобильного приложения	Пройден
4	Проверка перехода на страницу «Библиотека»	Пройден
5	Проверка перехода на страницу «Успеваемость»	Пройден

Продолжение таблицы 30

№ Тест-кейса	Название проверки	Результат
6	Проверка перехода на страницу «Избранное»	Пройден
7	Проверка перехода на вкладку «Задании»	Пройден
8	Проверка перехода на вкладку «Лекции»	Пройден
9	Проверка перехода на страницу «Настройки»	Пройден
10	Проверка возвращения на вкладку «Лекции»	Пройден
11	Проверка возвращения на вкладку «Задании»	Пройден
Модуль 3: Работа со словарем		
12	Проверка поиска слова на русском языке	Пройден
13	Проверка поиска слова на китайском языке	Пройден
14	Проверка открытия страницы с информацией о слове	Пройден
15	Проверка открытия страницы с информацией о последнем просмотренном слове	Пройден
16	Проверка открытия страницы с информацией о слове из списка терминов лекции	Пройден

Продолжение таблицы 30

№ Тест-кейса	Название проверки	Результат
17	Проверка на добавление слово в избранное	Пройден
18	Проверка на удаление слова из избранного	Пройден
Модуль 4: Работа с лекционным материалом		
19	Проверка возможности просмотра конспекта лекции	Пройден
20	Проверка возможности просмотра терминов лекции	Пройден
21	Проверка поиска лекции на русском языке	Пройден
22	Проверка поиска лекции на китайском языке	Пройден
Модуль 5: Работа с практическим материалом		
23	Проверка возможности просмотра страницы тестового задания	Пройден
24	Проверка возможности просмотра страницы задания с письменным ответом	Пройден
25	Проверка сохранения ответа к тестовому заданию	Пройден
26	Проверка сохранения ответа к письменному заданию	Пройден
27	Проверка сброса ответа к тестовому заданию	Пройден
28	Проверка сброса ответа к письменному заданию	Пройден

С помощью данной таблицы была подготовлена гистограмма, отражающая в процентах успешное прохождение тест-кейсов, которая представлена на рисунке 17.

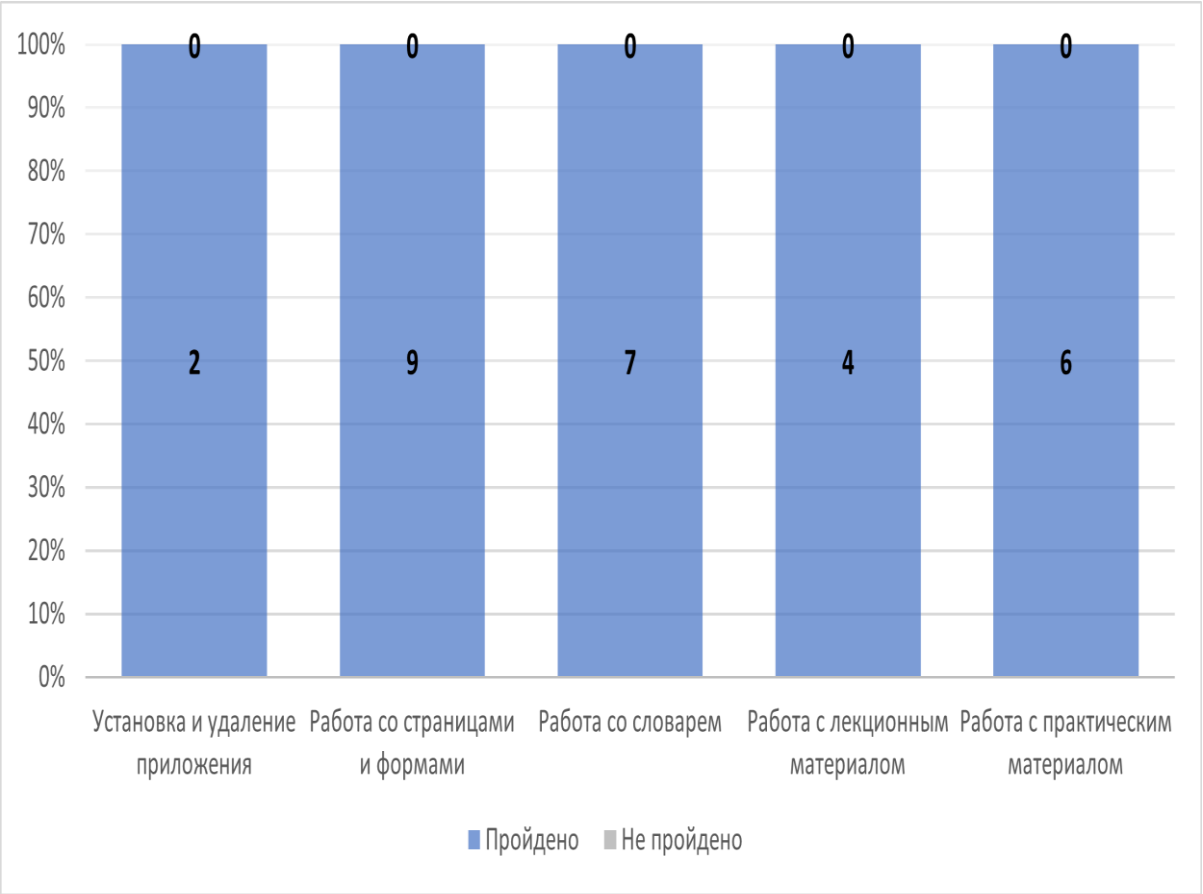


Рисунок 17 – Гистограмма прохождения тест-кейсов

Таким образом, за период 17-29 января были протестированы 28 тест-кейсов, из которых успешно прошло 100% тест-кейсов дымового тестирования и 100% тест-кейсов тестирования критического пути. 100% требований высокой важности реализовано корректно. Метрики качества находятся в зелёной зоне, потому есть все основания считать, что приложение готово к эксплуатации.

6 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

6.1 Установка, обновление, удаление мобильного приложения

Для установки мобильного приложения пользователю необходимо выполнить следующие действия:

1. Скачать приложение в формате APK из предоставленного преподавателем ресурса.
2. Открыть скаченный файл для установки, предварительно разрешив установку приложений из неизвестных источников. Обычно система сама предлагает это сделать при открытии файла, но данную опцию можно настроить самостоятельно.
3. Произвести установку приложения.

Обновление мобильного приложения выполняется через установку новой версии приложения в формате APK. Для обновления приложения будет запрошено подтверждение выполнения этой операции. В случае отказа обновление будет отложено. Также имеет смысл отказаться от обновления, чтобы сделать резервную копию базы данных.

Для удаления приложения необходимо выполнить следующие действия:

1. Выполнить длинное нажатие иконки приложения.
2. В открывшемся контекстном меню выбрать команду «Удалить» и подтвердить действие.

Следует отметить, что после удаления приложения данные информационной базы восстановить невозможно.

6.2 Запуск мобильного приложения

Для запуска программного средства необходимо нажать на иконку приложения, после чего будет открыто его основное окно с главной страницей, как изображено на рисунке 18.

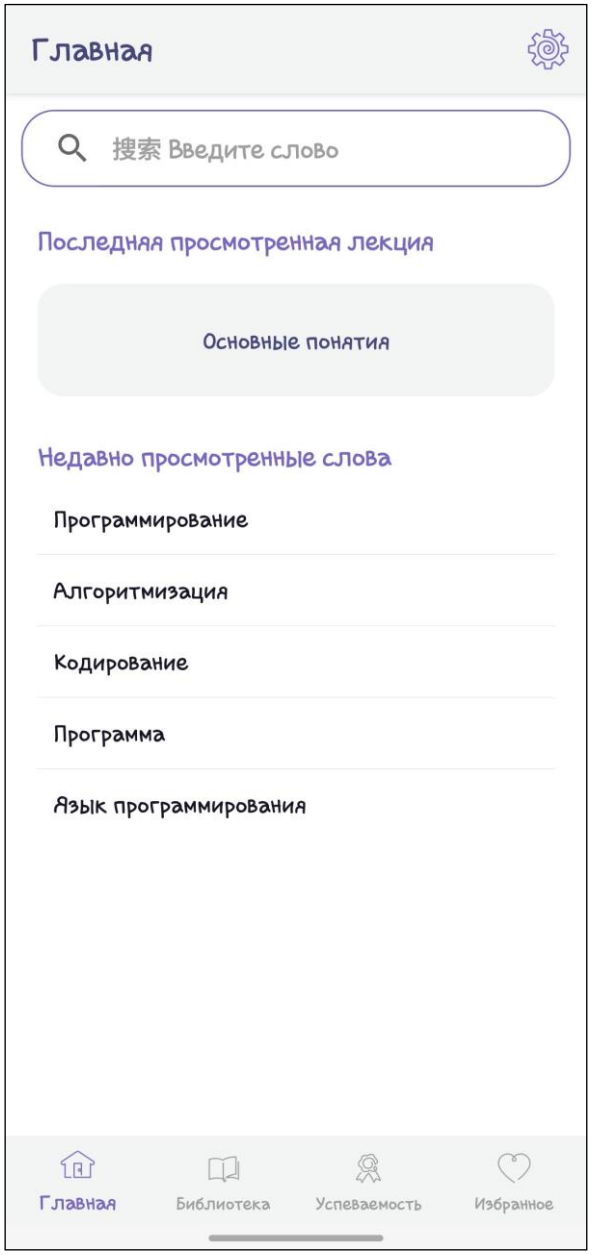


Рисунок 18 – Главная страница приложения

На странице «Главная» присутствуют поисковая строка для поиска слов в словаре или конспекта лекций, последняя просмотренная лекция и пять последних просмотренных слов. В навигационной шапке есть кнопка перехода на страницу «Настройки». На всех страницах приложения главная навигационная панель находится снизу.

6.3 Работа с приложением

При нажатии на главной странице кнопки с изображением шестеренок в навигационной шапке открывается страница настроек приложения, изображенная на рисунке 19.

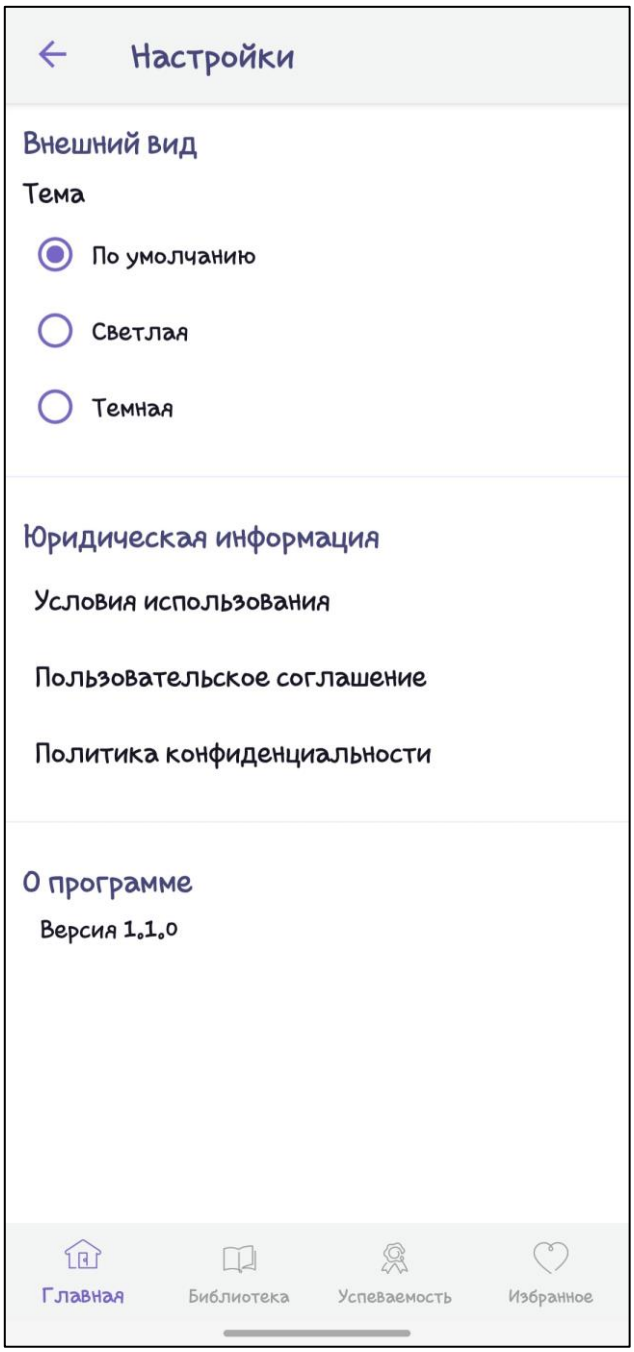


Рисунок 19 – Страница «Настройки»

На данной странице можно изменить цветовое оформление программы, ознакомиться с юридической информацией и информацией о самом

программном продукте. Для выхода на главную страницу необходимо нажать на кнопку «Назад» в навигационной шапке.

Для поиска слов в русско-китайском словаре или конспекта лекции достаточно просто начать вводить слово или иероглиф в строку поиска на странице «Главная», как видно на рисунке 20.

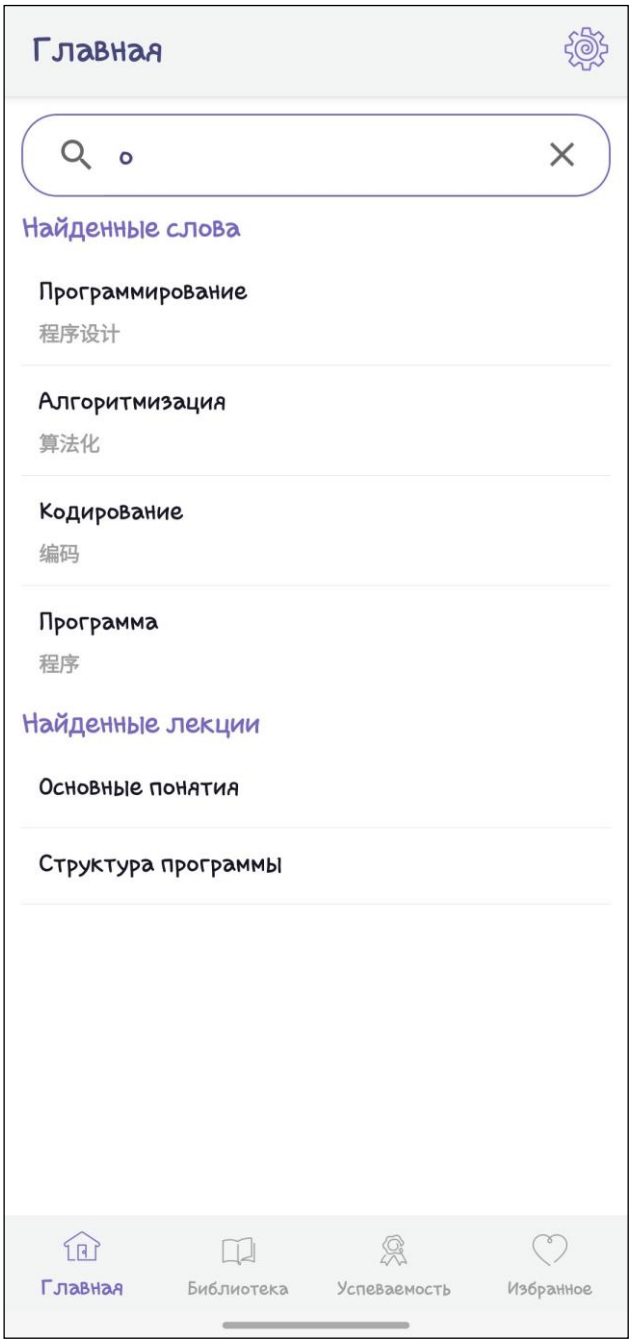


Рисунок 20 – Поиск при помощи ввода данных в поисковую строку

При нажатии на искомое слово из результатов поиска открывается страница с информацией о слове, представленная на рисунке 21. В

зависимости от того, добавлено ли данное слово в избранное или нет, появляется возможность добавить слово в избранное или удалить его из избранного.

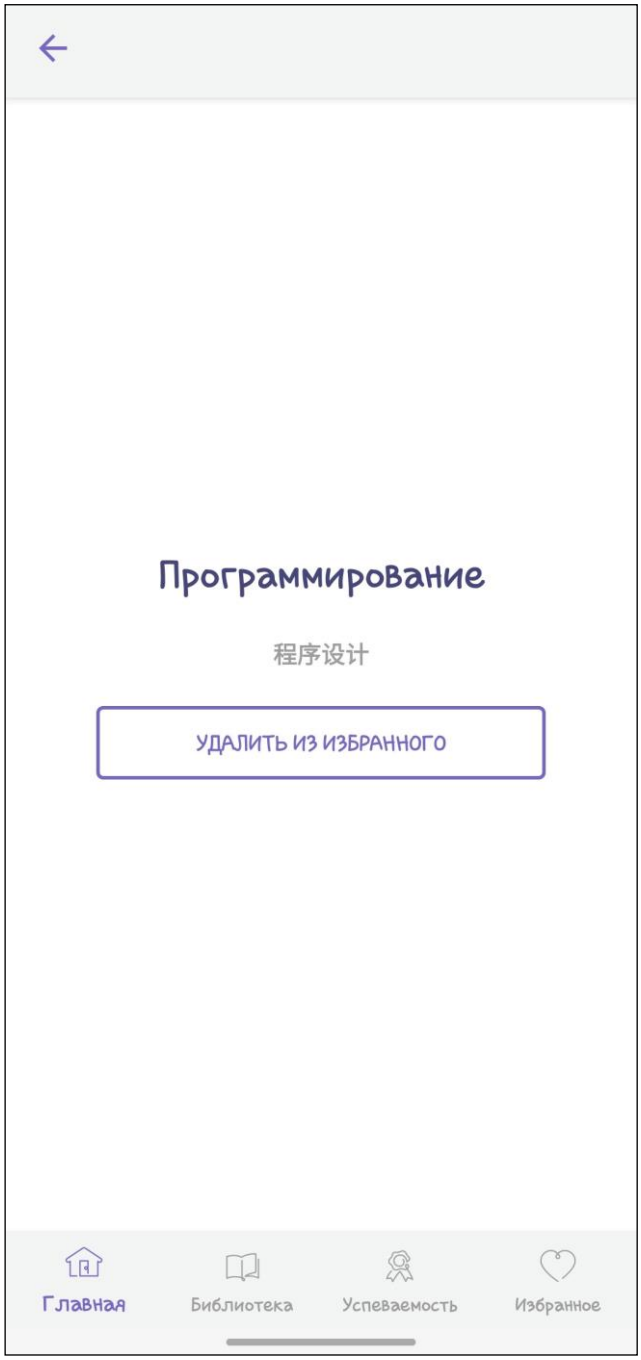


Рисунок 21 – Страница с информацией о слове

Такая же страница открывается при нажатии на слово из списка «Недавно просмотренные слова» на главной странице, списка слов, которые используются в конспекте лекции или в описании задачи, а также из списка, представленного на странице «Избранное», изображённого на рисунке 22.

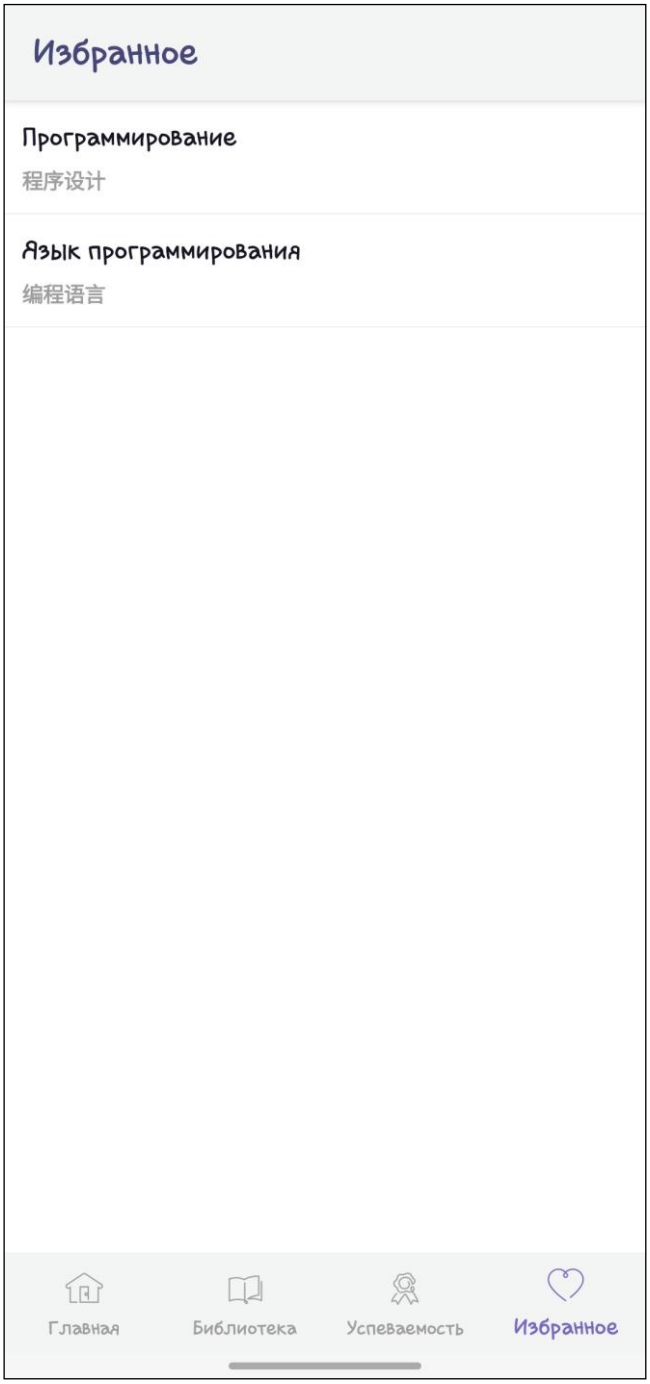


Рисунок 22 – Страница «Избранное»

При нажатии кнопки «Библиотека» в основном навигационном меню открывается страница с открытой вкладкой «Лекции», где представлен список лекций по разделам, и неактивной вкладкой «Задания». Данная страница представлена на рисунке 23.

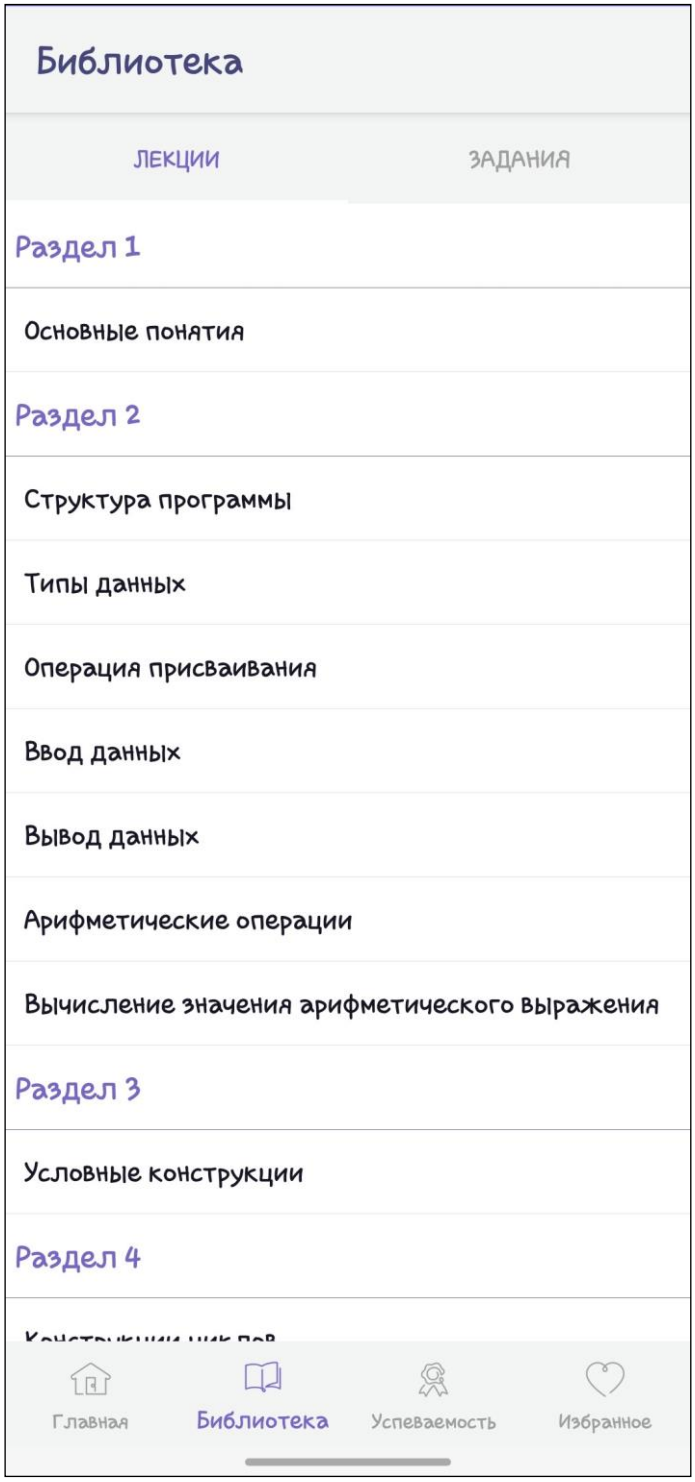


Рисунок 23 – Страница «Библиотека» с вкладкой «Лекции»

На рисунке 24 изображена страница с конспектом выбранной лекции, открывающаяся после нажатия на данную лекцию из списка из рисунка 22. В навигационной шапке также есть кнопка для перехода на страницу со списком слов, используемых в конспекте. Также открыть страницу содержимого

лекции можно с главной страницы, нажав на последнюю просмотренную лекцию или из результатов поиска.

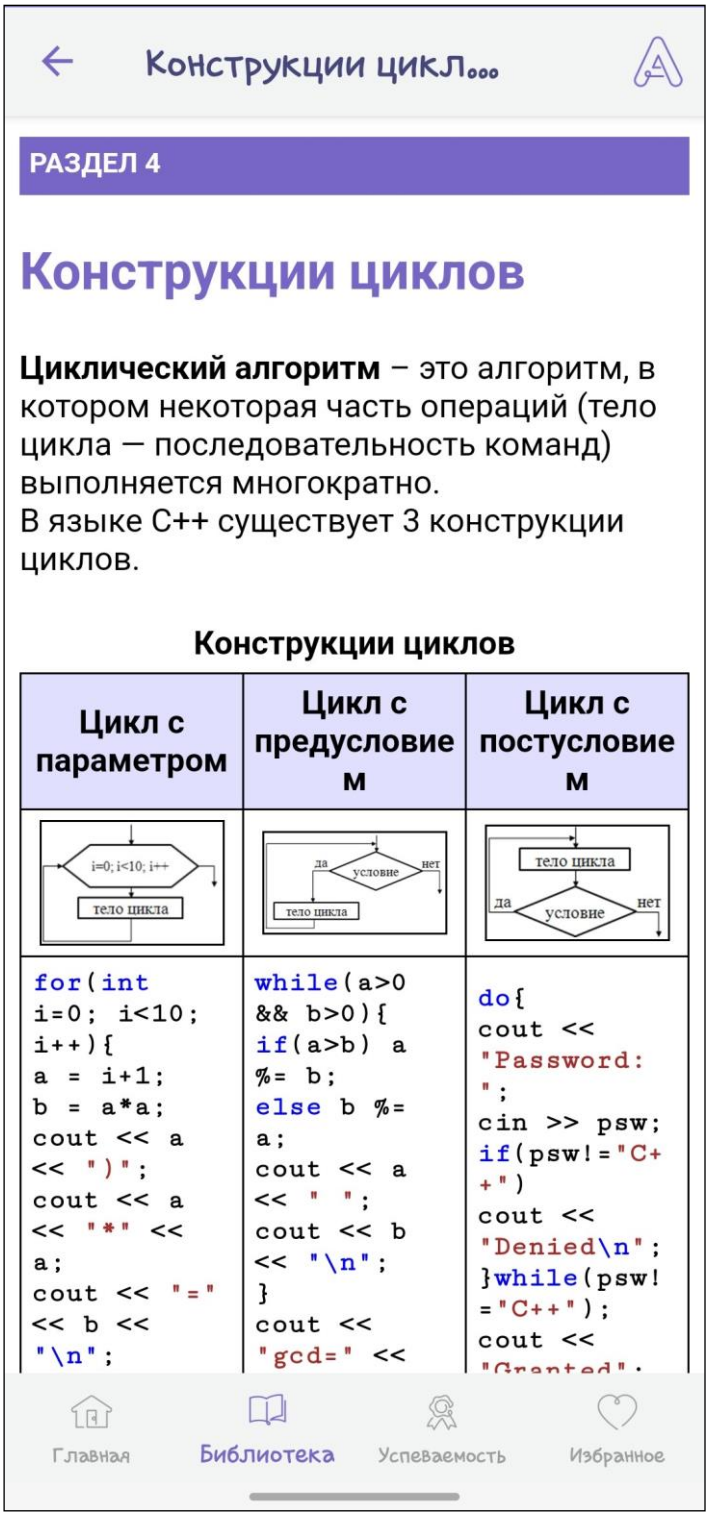


Рисунок 24 – Страница содержимого лекции

Чтобы вернуться к списку лекций на странице «Библиотека» необходимо нажать на кнопку «Назад» в навигационной шапке.

Перейдя на вкладку «Задания» на странице библиотеки пользователь может увидеть список задач, сгруппированных по разделам, как показано на рисунке 25.

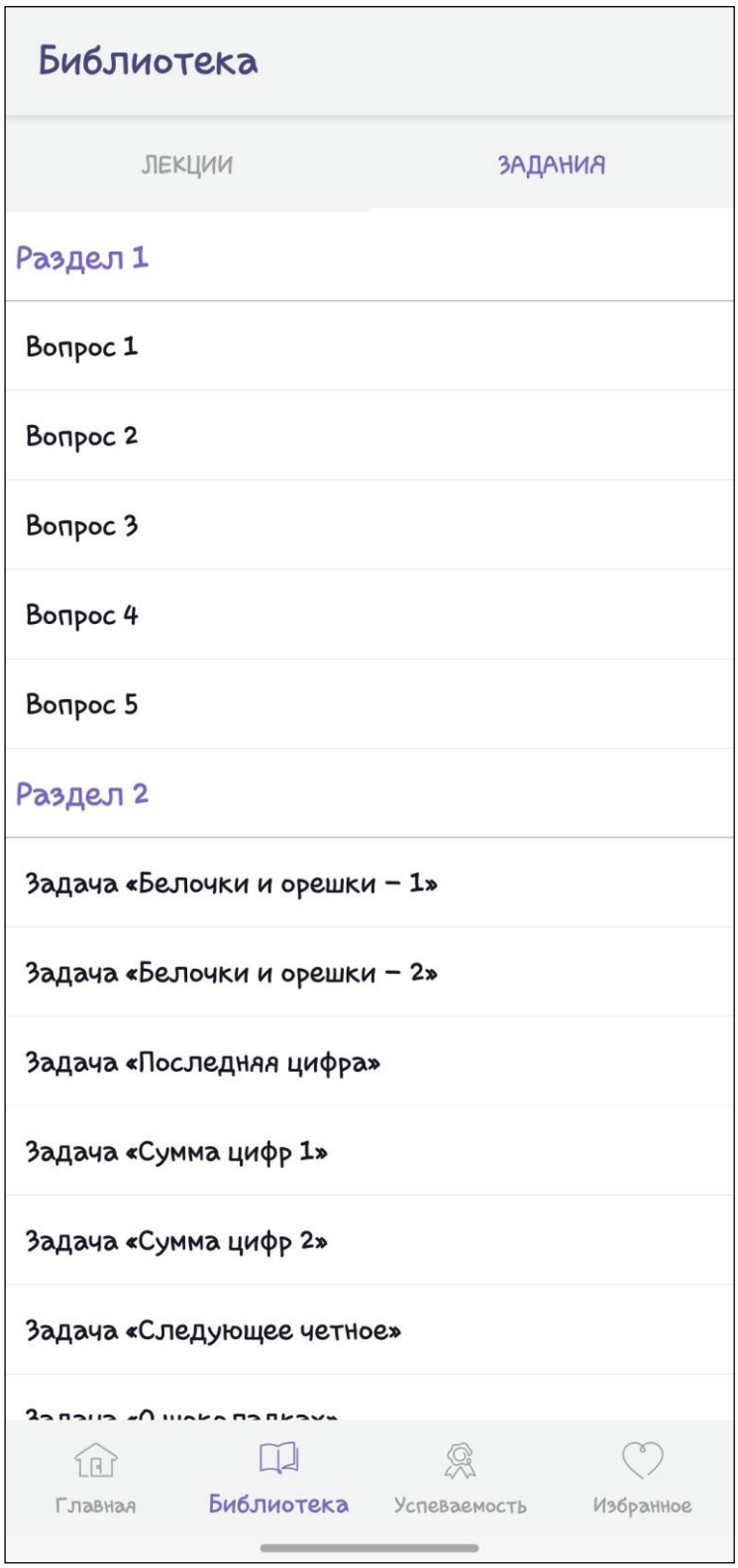


Рисунок 25 – Страница «Библиотека» с вкладкой «Задания»

На рисунке 26 показана страница решенной задачи, которая открывается при её выборе из списка во вкладке «Задания». На данной странице содержится описание задания и формата входных и выходных данных, а также блок для решения задачи, в котором содержится сохраненное решение.

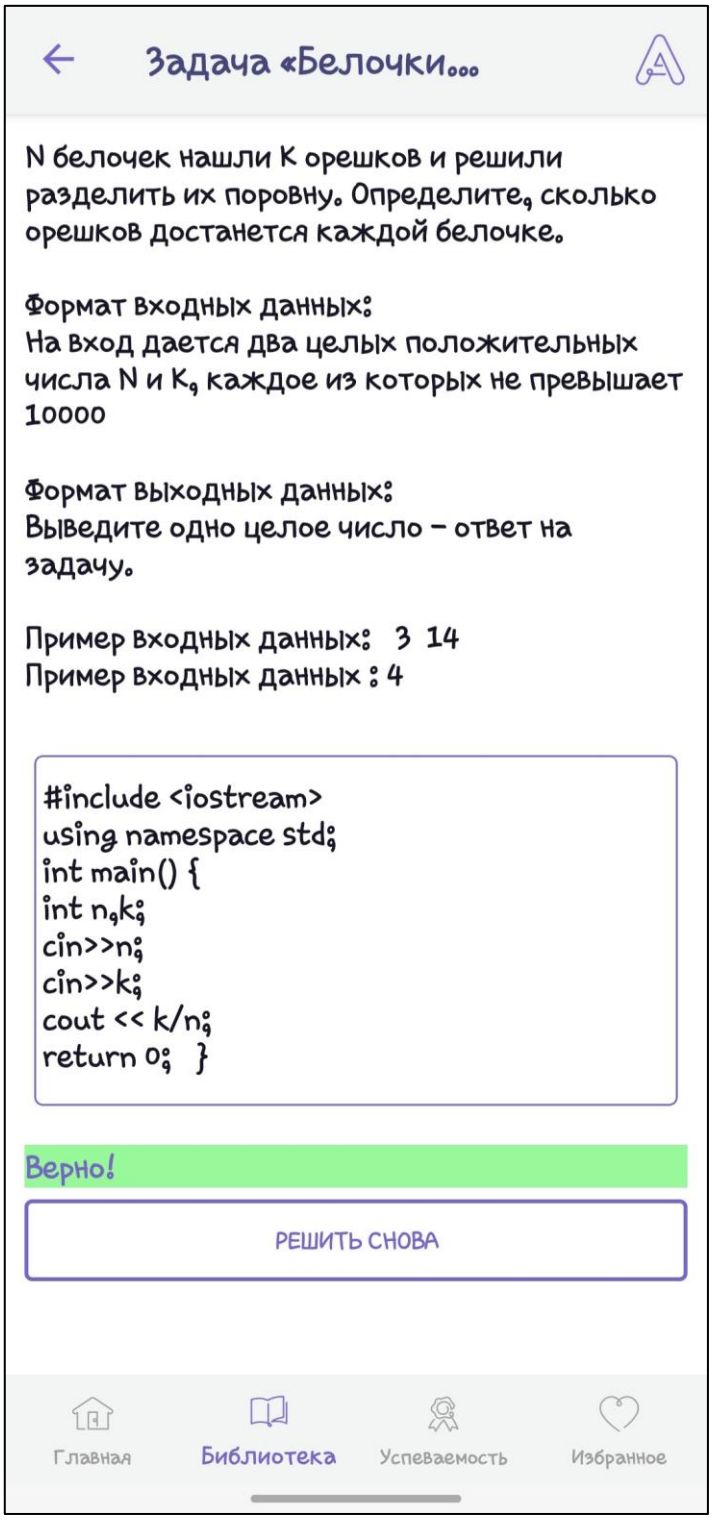


Рисунок 26 – Страница содержимого задания

При переходе на страницу «Успеваемость» пользователь может увидеть свои результаты выполнения задач по разделам и общую оценку за весь раздел, как представлено на рисунке 27.

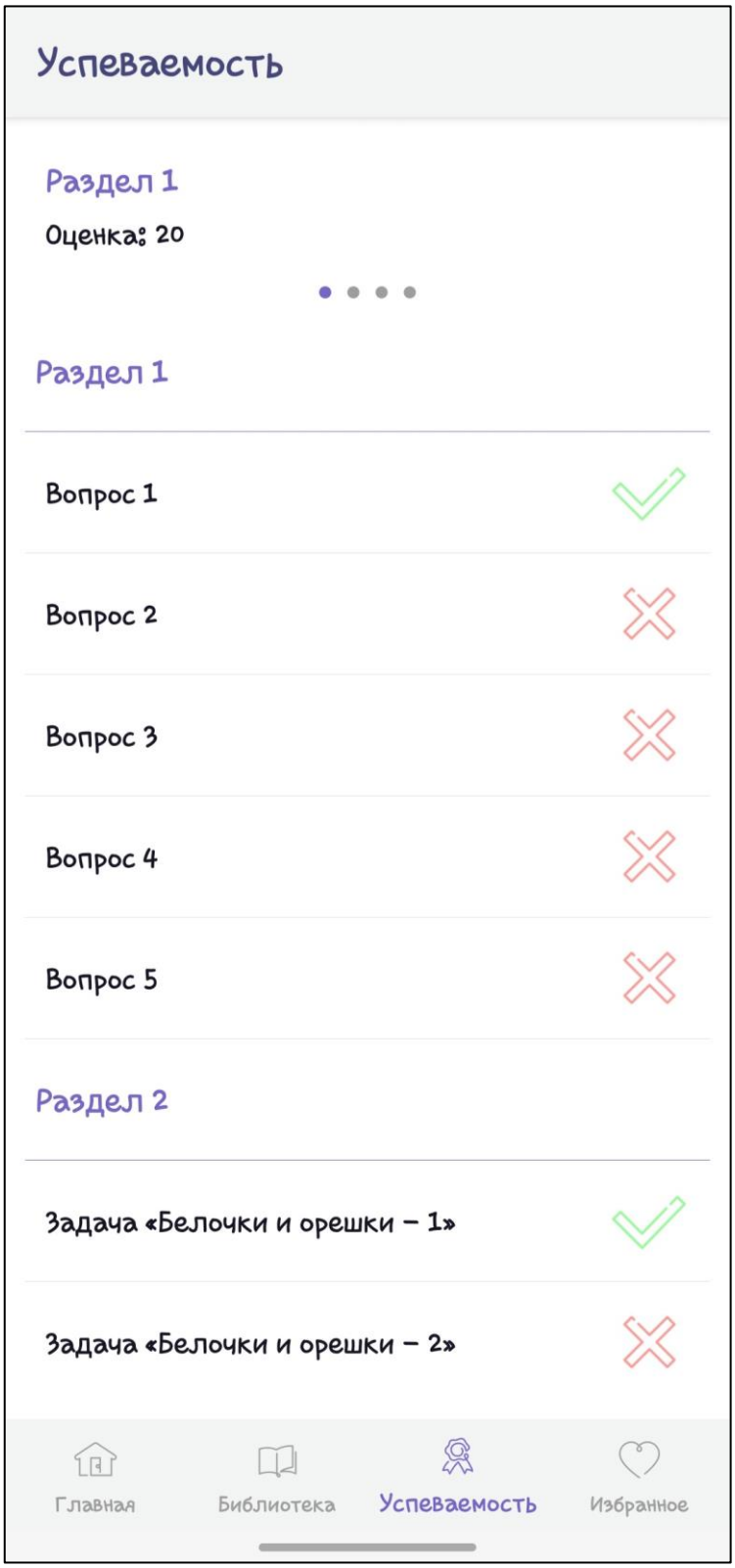


Рисунок 27 – Страница «Успеваемость»

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы было разработано мобильное приложение на ОС Android для изучения языка программирования C++ и русского языка в рамках профессиональной деятельности. Приложение реализовано средствами платформы Xamarin.Forms, языка разметки XAML, языка программирования C# и СУБД SQLite

В результате выполнения данного проекта были решены задачи, поставленные в начале работы:

- произведена постановка и анализ требований к программному продукту;
- произведен анализ предметной области;
- проведен обзор и сравнительный анализ программных сред и средств разработки приложения;
- спроектированы алгоритмы для реализации программного продукта;
- спроектированы пользовательские сценарии и определены операции пользователей;
- спроектирована и реализована база данных;
- реализовано мобильное приложение, соответствующее заявленным требованиям;
- произведено тестирование и оптимизация разработанного приложения.

На основе анализа результатов тестирования можно сделать вывод о том, что разработанное приложение и все входящие в его состав объекты работают корректно.

Таким образом, в результате выполнения выпускной квалификационной работы было разработано мобильное программное средство, отвечающие всем функциональным требованиям.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1 Россия и Китай подписали протокол об увеличении обмена учащимися / Министерство науки высшего образования Российской Федерации [Официальный сайт]. – URL : https://minobrnauki.gov.ru/press-center/news/main/23992/, свободный – Яз. рус. – (Дата обращ. 02.04.2023).

2 Яценко Е.Б. Анализ состояния и перспектив развития сотрудничества России и Китая в области образования в рамках государственных усилий и университетских инициатив / Гуманитарные научные исследования, № 12, 2017 [Электронный ресурс]. – Электрон. ст. – URL : https://human.snauka.ru/2017/12/24720, свободный – Яз. рус. – (Дата обращ. 02.04.2023).

3 Россия и Китай продолжают укреплять сотрудничество в сфере науки и образования / Министерство науки высшего образования Российской Федерации [Официальный сайт]. – URL : https://minobrnauki.gov.ru/press-center/news/mezhdunarodnoe-sotrudnichestvo/53743/, свободный – Яз. рус. – (Дата обращ. 02.04.2023).

4 Основные трудности обучения русскому языку как иностранному китайских студентов / Ян Бай, Е.В. Язовских / Современные направления профессиональной языковой подготовки лингвистов-переводчиков и преподавателей иностранного языка в вузе — Екатеринбург : Издательский Дом «Ажур», 2023. — С. 6-12. (Дата обращ. 10.04.2023).

5 Бай Лицзюань. Русский язык для китайцев: трудности изучения / Экономический рост Республики Беларусь: глобализация, инновационность, устойчивость: материалы XIV Международной научно-практической конференции, Минск, 20 мая 2021 г. – Минск : БГЭУ, 2021. – С. 472-473 (Дата обращ. 10.04.2023).

6 Зайцева, Л. В. Модели и методы адаптивного контроля знаний / Educational Technology & Society, № 4, 2004 – С. 265-277 (Дата обращ. 26.04.2023).

ПРИЛОЖЕНИЯ

Приложение А

Техническое задание на разработку мобильного приложения для обучения студентов Китайско-российского института Хэйлунцзянского университета языку программирования C++

1 Предмет разработки

Предметом разработки является мобильного приложения для обучения студентов Китайско-российского института Хэйлунцзянского университета языку программирования C++.

Назначение приложения:

- предоставление и изучение обучающего материала для изучения языка программирования C++;
- контроль знаний по пройденному материалу;
- изучение русского языка в профессиональной деятельности;
- расширение словарного запаса русского языка.

Цель создания приложения: создать цифровое пространство для самостоятельного изучения языка программирования C++ и русского языка доступное для студентов КРИ.

Конкретные цели проекта заключаются в следующем:

- повысить успеваемость студентов по дисциплине «Конструирование языковых программ», чтобы на итоговой аттестации 25% студентов набирали больше 90 баллов, 50% студентов набирали от 80 до 89 баллов, и оставшиеся 25% студентов набирали от 60 до 79 баллов. замотивировав к изучению учебного материала в предпочтительном для самих студентов режиме;

- повысить уровень русского языка студентов до ТРКИ-I, создав возможность для расширения словарного запаса русского языка, необходимого для формирования карьеры студентов.
- увеличить объемы предоставляемой информации и практики изучаемого материала на 20% за счет перехода на новую технологическую платформу и передачи информации через новый канал;
- вдвое уменьшить временные затраты на поиск необходимой информации для изучения дисциплины «Конструирование языковых программ», создав удобную платформу, предоставляющую необходимые материалы.

Целевая аудитория мобильного приложения: совершеннолетние граждане Китайской Народной Республике (мужчины и женщины), обучающиеся в Китайско-российском институте дисциплине «Конструирование языковых программ».

2 Основные функции мобильного приложения

Основной функцией мобильного приложения является – получать необходимый учебный материал и практику по изучаемой дисциплине, а также расширять словарный запас русского языка, независимо от времени суток и местонахождения.

3 Требования к техническому обеспечению

Мобильные устройства, на которых должен работать программный продукт, должны:

- официально поддерживаться производителями ОС Android;
- обладать разрешением экрана от 480*800 до 2048*2732 пикселей.

4 Минимальные требования к ОС

- Разработанное мобильное приложение должно корректно работать на смартфонах с различной версией операционных систем и размером экрана.
- Минимальная поддерживаемая версия – Android 8.
- Максимальная поддерживаемая версия на момент релиза приложения – Android 14.

5 Требования к программному обеспечению приложения

- Для разработки мобильного приложения под ОС Android должна использоваться кроссплатформенная среда Xamarin.Forms, язык разметки XAML и язык программирования C#.
- Мобильное приложение под ОС Android должно работать на версиях Android 8.0 и выше.
- При условии нормального режима функционирования приложение доступно пользователям круглосуточно. Нормальный режим функционирования приложения 365 дней в году 7 дней в неделю 24 часа в сутки режим 24x7x365.
- Среднее время отклика на действия пользователей при использовании приложения не более 0,5 сек.

6 Требования к локализации приложения

Программное средство должно быть на русском языке, а также иметь отдельный китайско-русский словарь, на который выводят ссылки, для любых слов и терминов, используемых на русском языке.

7 Требования к надежности

Приложение должно обеспечивать целостность и непротиворечивость хранимых данных при любых действиях конечных пользователей.

Приложение должно обеспечивать корректную обработку ситуаций, вызванных неверными действиями пользователей, неверным форматом или недопустимыми значениями входных данных. В указанных случаях приложение должно выдавать пользователю соответствующие сообщения об ошибках.

Сообщения об ошибках не должны содержать техническую информацию и должны предлагать пользователям чёткий алгоритм дальнейших действий.

Приложение должно сохранять работоспособность и обеспечивать восстановление своих функций при возникновении следующих внештатных ситуаций:

- при сбоях в системе аппаратной части, приводящих к перезагрузке ОС, восстановление работы подсистемы должно происходить после перезапуска ОС и запуска прикладного программного обеспечения;
- при ошибках в работе аппаратных средств восстановление функций приложения возлагается на платформу;
- при ошибках, связанных с программным обеспечением (ОС и драйверы устройств), восстановление работоспособности возлагается на ОС.

В приложении должны быть предусмотрены средства для организации резервного копирования и обеспечения восстановления работоспособности в случае программно-аппаратных сбоев. Должны быть предусмотрены меры по регулярному сохранению файлов и баз данных.

8 Требования по стандартизации и унификации

Элементы интерфейса программы должны проектироваться с учётом требований по стандартизации и унификации производителей ОС Мобильных устройств: Google Inc. (Проектирование).

9 Минимальное представление и структура экранов

Все названия разделов платформы, приведенные ниже, являются условными и могут корректироваться по согласованию с Заказчиком в ходе проектирования. Первоначальная структура приложения должна иметь следующий вид:

1. Главная страница:

- Недавно просмотренная лекция.
- Недавно просмотренные термины из словаря.

2. Библиотека:

- Список разделов.
- Название раздела.
- Список глав и тем.

3. Экран одной главы или задачи:

- Содержание главы или задачи.
- Файлы учебных материалов или блок для кода.

4. Экран статистики успеваемости:

- Список успеваемости по разделам и главам.

5. Экран словаря:

- Поисковая строка для поиска значения термина на русском или китайском языках.
- Список сохраненных терминов.

6. Дополнительные экраны:

- Экран термина.
- Экран настроек.

10 Дизайн

Пользовательский интерфейс должен быть дружелюбным и удобным для пользователей. Программа должна иметь подсказки и указатели на функциональные компоненты приложения. При возникновении ошибки или сбоя, приложение должно выдавать соответствующее информацию, понятное

конечному пользователю. У пользователя должна быть возможность продолжить обучение с того места, которое было сохранено в последний раз.

В качестве основного светлого фона используются цвет – #F3F4F4 и #FFFFFF, а темного – #303030 и #1C1C1C;

Для акцентирования внимания пользователя используются следующие цвета:

- в светлом режиме #7766C6 и #46467A;
- в темном #7766C6 и #FFFFFF.

Следующие элементы должны присутствовать на каждой странице:

- панель инструментов в верхней части экрана, содержащая логотип и кнопки управления текущим экраном;
- главное навигационное меню – панель вкладок в нижней части экрана, позволяющая быстро переключаться между разделами приложения.

Пользовательский интерфейс должен удовлетворять следующим требованиям:

- обеспечивать легкое определение местоположения пользователя в системе;
- обеспечивать минимум усилий и временных затрат при навигации.

В дизайне сайта не должны присутствовать:

- много сливающегося текста;
- цветовые сочетания и графические решения, не заявленные Заказчиком.

11 Этапы работы

Проект рассчитан на реализацию в срок не более 4 месяцев с момента утверждения технического задания (работа по задачам может вестись параллельно). В рамках реализации этапов указанных работ необходимо проводить постоянные консультации с Заказчиком для эффективности решения возникших вопросов и задач.

Приложение Б

Тип
Процесс
Вычисление
Целочисленный
Компилятор
Пространство имён
ООП
Описание
Цикл
Директива
类
注释
源代码
編譯器
命名空间
命令行界面
字符
过程理论
实践
算法

Рисунок Б.1 – Примеры входных данных для поиска слов в словаре и лекций

Таблица Б.1 – Примеры входных данных для проверки решения задач

Название задачи	Входные данные
Задача «Белочки и орешки – 1»	3 14
Задача «Последняя цифра»	753
Задача «Сумма цифр 2»	8324
Задача «Наибольшее число»	23 45
Задача «Шахматная ладья»	4 4 7 4
Задача «О существовании треугольника»	3 4 5
Задача «Счастливый билет»	123640
Задача «Знак числа»	-5

Приложение В

Рисунок В.1 – Диаграмма классов компонента модель

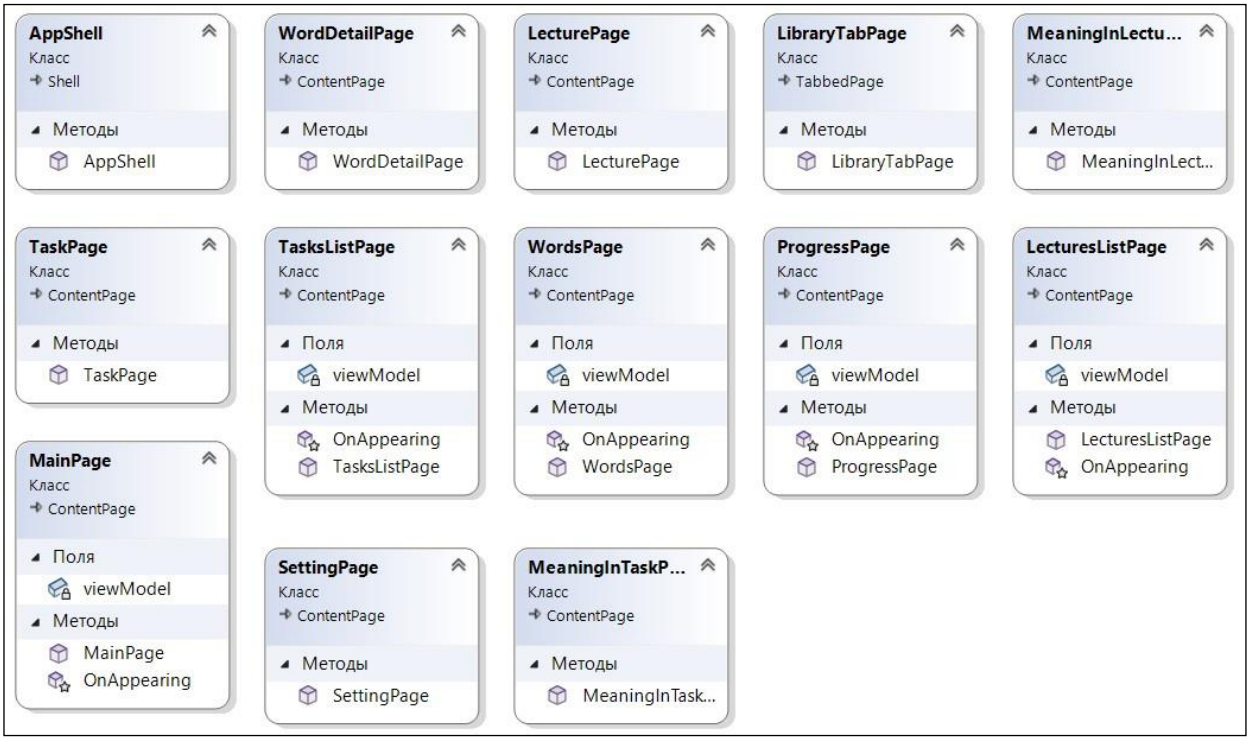


Рисунок В.2 – Диаграмма классов компонента представления

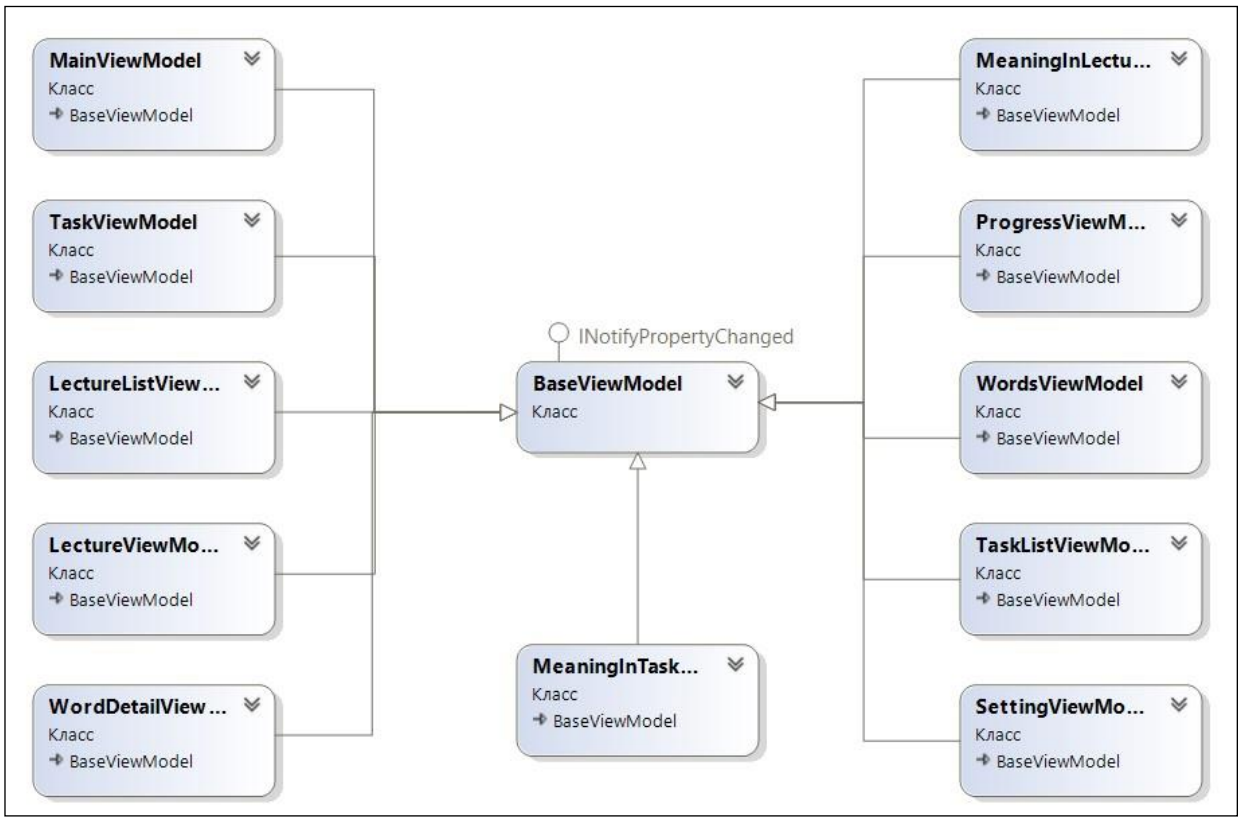


Рисунок В.3 – Диаграмма классов компонента модель представления

Приложение Г

Таблица Г.1 – Тест-кейс № 1 | Проверка установки мобильного приложения

ID: 1	**Приоритет:** Высокий
Требование: ПТ-1.1. Установка приложения.	
Описание тест-кейса: Проверить мобильное приложение на успешную установку.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Предварительно скачать APK файл приложения. 2) Открыть APK файл. 3) Разрешить установку приложения. 4) Подтвердить установку.	Во время установки не должно выскакивать никаких сообщений об ошибке установки. На экране приложений или на главном экране должна появится иконка приложения. В списках приложений должно появиться название приложения.

Таблица Г.2 – Тест-кейс № 2 | Проверка удаления мобильного приложения

ID: 2	**Приоритет:** Высокий
Требование: ПТ-1.2. Удаление приложения.	
Описание тест-кейса: Проверить мобильное приложение на успешное удаление.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) На экране приложений или на главном экране долго нажать на иконку мобильного приложения. 2) В открывшемся контекстном меню нажать «Удалить» и подтвердить действие.	Во время удаления не должно выскакивать никаких сообщений об ошибке установки. На экране приложений или на главном экране иконка приложения должна исчезнуть. В списках приложений не должно числиться название приложения.

Таблица Г.3 – Тест-кейс № 3 | Проверка запуска мобильного приложения

ID: 3	**Приоритет:** Высокий
Требование: ПТ-2. Запуск приложения. ПТ-3.3. Просмотр недавно просмотренных слов и тем.	
Описание тест-кейса: Проверить на успешный запуск мобильного приложения.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
На экране приложений или на главном экране нажать на иконку мобильного приложения.	Приложение должно запуститься и открыть страницу «Главная» в вертикальной ориентации, где присутствуют следующие элементы: • Навигационное поле с наименованием страницы и кнопкой перехода на страницу «Настройки». • Поисковая строка. • Список недавно просмотренных слов и тем. • Главное навигационное меню внизу страницы.

Таблица Г.4 – Тест-кейс № 4 | Проверка перехода на страницу «Библиотека»

ID: 4	**Приоритет:** Высокий
Требование: ПТ-3.1. Навигация между основными страницами. ПТ-3.4. Просмотр вкладки «Лекции».	
Описание тест-кейса: В процессе работы приложения проверить работоспособность перехода с страницы «Главная» на страницу «Библиотека».	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице «Главная» нажать на иконку страницы «Библиотека» в основном меню, находящимся в нижней части страницы.	Открывается страница «Библиотека», где присутствуют следующие элементы: • Навигационное поле с наименованием страницы и кнопкой для поиска. • Активная вкладка «Лекции». • Неактивная вкладка «Задании». • Список разделов во вкладке «Лекции».

Таблица Г.5 – Тест-кейс № 5 | Проверка перехода на страницу «Успеваемость»

ID: 5	**Приоритет:** Высокий
Требование: ПТ-3.1. Навигация между основными страницами. ПТ-3.11. Просмотр успеваемости.	
Описание тест-кейса: В процессе работы приложения проверить работоспособность перехода с страницы «Библиотека» на страницу «Успеваемость».	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице «Главная» нажать на иконку страницы «Библиотека» в основном навигационном меню. 3) На открывшейся странице нажать на иконку страницы «Успеваемость» в основном навигационном меню.	Открывается страница «Успеваемость», где присутствуют следующие элементы: • Навигационное поле с наименованием страницы. • Список с успеваемостью по темам и общая оценка по разделам.

Таблица Г.6 – Тест-кейс № 6 | Проверка перехода на страницу «Избранное»

ID: 6	**Приоритет:** Высокий
Требование: ПТ-3.1. Навигация между основными страницами. ПТ-3.7. Просмотр избранных слов.	
Описание тест-кейса: В процессе работы приложения проверить работоспособность перехода с страницы «Успеваемость» на страницу «Избранное».	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице «Главная» нажать на иконку страницы «Успеваемость» в основном навигационном меню. 3) На открывшейся странице нажать на иконку страницы «Избранное» в основном навигационном меню.	Открывается страница «Избранное», где присутствуют следующие элементы: • Навигационное поле с наименованием страницы. • Список с избранными словами.

Таблица Г.7 – Тест-кейс № 7 | Проверка перехода на вкладку «Задании»

ID: 7	**Приоритет:** Высокий
Требование: ПТ-3.5. Просмотр вкладки «Задании».	
Описание тест-кейса: В процессе работы приложения проверить работоспособность перехода на вкладку «Задании» и просмотр содержимого.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице «Главная» нажать на иконку страницы «Библиотека» в основном меню, находящимся в нижней части страницы. 3) В открывавшейся странице перейти во вкладку «Задании».	Открывается страница «Библиотека» во вкладке «Задании», где присутствуют следующие элементы: • Навигационное поле с наименованием страницы и кнопкой для поиска. • Неактивная вкладка «Лекции». • Активная вкладка «Задании». • Список разделов во вкладке «Задании».

Таблица Г.8 – Тест-кейс № 8 | Проверка перехода на вкладку «Лекции»

ID: 8	**Приоритет:** Высокий
Требование: ПТ-3.4. Просмотр вкладки «Лекции».	
Описание тест-кейса: В процессе работы приложения проверить работоспособность перехода на вкладку «Лекции» и просмотр содержимого.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице «Главная» нажать на иконку страницы «Библиотека» в основном меню, находящимся в нижней части страницы. 3) В открывавшейся странице перейти во вкладку «Задании». 4) Перейти обратно во вкладку «Лекции».	Открывается страница «Библиотека», где присутствуют следующие элементы: • Навигационное поле с наименованием страницы и кнопкой для поиска. • Активная вкладка «Лекции». • Неактивная вкладка «Задании». • Список разделов во вкладке «Лекции».

Таблица Г.9 – Тест-кейс № 9 | Проверка перехода на страницу «Настройки»

ID: 9	**Приоритет:** Средний
Требование: ПТ-3.12. Просмотр настроек приложения.	
Описание тест-кейса: В процессе работы приложения проверить работоспособность перехода на страницу «Настройки» и просмотра содержимого страницы.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице в навигационной панели нажать на иконку страницы «Настройки».	Открывается страница слова «Настройки».

Таблица Г.10 – Тест-кейс № 10 | Проверка возвращения на вкладку «Лекции»

ID: 10	**Приоритет:** Высокий
Требование: ПТ-3.4. Просмотр вкладки «Лекции».	
Описание тест-кейса: В процессе работы приложения проверить работоспособность возврата на вкладку «Лекции» и просмотр содержимого.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице «Главная» нажать на иконку страницы «Библиотека» в основном меню, находящимся в нижней части страницы. 3) На открывшейся странице выбрать «Основные понятия программирования на C++». 4) Перейти обратно во вкладку «Лекции» с помощью кнопки возврата в навигационном меню.	Открывается страница «Библиотека», где присутствуют следующие элементы: • Навигационное поле с наименованием страницы и кнопкой для поиска. • Активная вкладка «Лекции». • Неактивная вкладка «Задании». • Список разделов во вкладке «Лекции».

Таблица Г.11 – Тест-кейс № 11 | Проверка возвращения на вкладку «Задании»

ID: 11	**Приоритет:** Высокий
Требование: ПТ-3.5. Просмотр вкладки «Задании».	
Описание тест-кейса: В процессе работы приложения проверить работоспособность возврата на вкладку «Задании» и просмотр содержимого.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице «Главная» нажать на иконку страницы «Библиотека» в основном меню, находящимся в нижней части страницы. 3) В открывавшейся странице перейти во вкладку «Задании». 4) В данной вкладке открыть первое задание в «Раздел 1». 5) Перейти обратно во вкладку «Задании» с помощью кнопки возврата в навигационном меню.	Открывается страница «Библиотека» во вкладке «Задании», где присутствуют следующие элементы: • Навигационное поле с наименованием страницы и кнопкой для поиска. • Неактивная вкладка «Лекции». • Активная вкладка «Задании». • Список разделов во вкладке «Задании».

Таблица Г.12 – Тест-кейс № 12 | Проверка поиска слова на русском языке

ID: 12	**Приоритет:** Средний
Требование: ПТ-3.2. Поиск слов в русско-китайском словаре.	
Описание тест-кейса: В процессе работы приложения проверить работоспособность поиска русских слов в русско-китайском словаре.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице в поисковой строке ввести «програм».	На странице «Главная» должен появится список слов по результатам поиска.

Таблица Г.13 – Тест-кейс № 13 | Проверка поиска слова на китайском языке

ID: 13	**Приоритет:** Средний
Требование: ПТ-3.2. Поиск слов в русско-китайском словаре.	
Описание тест-кейса: В процессе работы приложения проверить работоспособность поиска китайских слов в русско-китайском словаре.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице в поисковой строке ввести «程序».	На странице «Главная» должен появится список слов по результатам поиска.

Таблица Г.14 – Тест-кейс № 14 | Проверка открытия страницы с информацией о слове

ID: 14	**Приоритет:** Высокий
Требование: ПТ-3.8. Просмотр страницы со значением слова.	
Описание тест-кейса: В процессе работы приложения проверить работоспособность перехода на страницу со значением слова из списка с результатами поиска.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся главной странице в поисковой строке ввести «программирование». 3) Нажать на слово «Программирование» в списке с результатами поиска.	Открывается страница слова «программирование», где присутствуют следующие элементы: • Слово на русском. • Его перевод на китайский. • Кнопка добавление в избранное.

Таблица Г.15 – Тест-кейс № 15 | Проверка открытия страницы с информацией о последнем просмотренном слове

ID: 15	**Приоритет:** Высокий
Требование: ПТ-3.8. Просмотр страницы со значением слова.	
Описание тест-кейса: В процессе работы приложения проверить работоспособность перехода на страницу со значением слова из списка последних просмотренных слов.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся главной странице в разделе «Недавно просмотренные слова» нажать на первое слово в списке.	Открывается страница слова «программирование», где присутствуют следующие элементы: • Слово на русском. • Его перевод на китайский. • Кнопка добавление в избранное.

Таблица Г.16 – Тест-кейс № 16 | Проверка открытия страницы с информацией о слове из списка терминов лекции

ID: 16	**Приоритет:** Высокий
Требование: ПТ-3.8. Просмотр страницы со значением слова.	
Описание тест-кейса: В процессе работы приложения проверить работоспособность перехода на страницу со значением слова из списка терминов лекции.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице «Главная» нажать на иконку страницы «Библиотека» в основном меню, находящимся в нижней части страницы. 3) На открывшейся странице выбрать «Основные понятия программирования на C++». 4) В открывшемся конспекте в навигационной панели нажать на иконку кнопки «Термины». 5) Нажать на слово «Программирование» в списке на открывшейся странице.	Открывается страница слова «программирование», где присутствуют следующие элементы: • Слово на русском. • Его перевод на китайский. • Кнопка добавление в избранное.

Таблица Г.17 – Тест-кейс № 17 | Проверка на добавление слово в избранное

ID: 17	**Приоритет:** Средний
Требование: ПТ-3.9. Добавление слова в избранное.	
Описание тест-кейса: В процессе работы приложения проверить работоспособность добавление слова в избранное.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице в поисковой строке ввести «программирование». 3) Нажать на слово «Программирование» в списке с результатами поиска. 4) В открывшейся странице слова нажать на кнопку добавления в избранное. 5) В основном навигационном меню нажать на иконку страницы «Избранное».	На открывшейся странице «Избранное» в списке избранных слов должно присутствовать слово «программирование».

Таблица Г.18 – Тест-кейс № 18 | Проверка на удаление слова из избранного

ID: 18	**Приоритет:** Средний
Требование: ПТ-3.10. Удаление слова из избранного.	
Описание тест-кейса: В процессе работы приложения проверить работоспособность удаление слова из избранного.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) В основном навигационном меню нажать на иконку страницы «Избранное». 3) Нажать на слово «программирование» в списке избранных слов. 4) В открывшейся странице слова нажать на кнопку удаления из избранного. 5) В навигационной панели нажать на кнопку «Назад».	На открывшейся странице «Избранное» в списке избранных слово «программирование» должно отсутствовать.

Таблица Г.19 – Тест-кейс № 19 | Проверка возможности просмотра конспекта лекции

ID: 19	**Приоритет:** Высокий
Требование: ПТ-3.4. Просмотр конспекта лекции.	
Описание тест-кейса: В процессе работы приложения проверить возможность просмотра конспекта лекции.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице «Главная» нажать на иконку страницы «Библиотека» в основном меню, находящимся в нижней части страницы. 3) На открывшейся странице выбрать «Основные понятия».	Открывается страница с конспектом лекции «Основные понятия».

Таблица Г.20 – Тест-кейс № 20 | Проверка возможности просмотра терминов лекции

ID: 20	**Приоритет:** Средний
Требование: ПТ-3.6. Просмотр терминов в лекции.	
Описание тест-кейса: В процессе работы приложения проверить возможность просмотра терминов лекции.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице «Главная» нажать на иконку страницы «Библиотека» в основном меню, находящимся в нижней части страницы. 3) На открывшейся странице выбрать «Основные понятия». 4) В открывшемся конспекте в навигационной панели нажать на иконку кнопки «Термины».	Открывается страница со списком терминов, связанных с темой «Основные понятия».

Таблица Г.21 – Тест-кейс № 21 | Проверка поиска лекции на русском языке

ID: 21	**Приоритет:** Средний
Требование: ПТ-3.2. Поиск конспекта лекции по названию на русском языке.	
Описание тест-кейса: В процессе работы приложения проверить работоспособность поиска лекции по названию на русском языке.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся главной странице в поисковой строке ввести «основные».	На странице «Главная» должен появится список по результатам поиска, где присутствует лекция «Основные понятия».

Таблица Г.22 – Тест-кейс № 22 | Проверка поиска лекции на китайском языке

ID: 22	**Приоритет:** Средний
Требование: ПТ-3.2. Поиск конспекта лекции по названию на китайском языке.	
Описание тест-кейса: В процессе работы приложения проверить работоспособность поиска лекции по названию на китайском языке.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся главной странице в поисковой строке ввести «程».	На странице «Главная» должен появится список по результатам поиска, где присутствует лекция «Основные понятия».

Таблица Г.23 – Тест-кейс № 23 | Проверка возможности просмотра страницы тестового задания

ID: 23	**Приоритет:** Высокий
Требование: ПТ-3.5. Просмотр страницы задания.	
Описание тест-кейса: В процессе работы приложения проверить возможность просмотра страницы тестового задания	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице «Главная» нажать на иконку страницы «Библиотека» в основном меню, находящимся в нижней части страницы. 3) В открывавшейся странице перейти во вкладку «Задании». 4) В данной вкладке открыть первое задание в «Раздел 1».	Открывается страница с заданием, где присутствуют вопрос к заданию и предложенные варианты ответа, где надо выбрать один вариант и кнопка «Ответить».

Таблица Г.24 – Тест-кейс № 24 | Проверка возможности просмотра страницы задания с письменным ответом

ID: 24	**Приоритет:** Высокий
Требование: ПТ-3.5. Просмотр страницы задания.	
Описание тест-кейса: В процессе работы приложения проверить возможность просмотра страницы задания с письменным ответом	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице «Главная» нажать на иконку страницы «Библиотека» в основном меню, находящимся в нижней части страницы. 3) В открывавшейся странице перейти во вкладку «Задании». 4) В данной вкладке открыть первое задание в «Раздел 2».	Открывается страница с заданием, где присутствуют описание задачи и поле для ввода ответа, а также кнопка «Ответить».

Таблица Г.25 – Тест-кейс № 25 | Проверка сохранения ответа к тестовому заданию

ID: 25	**Приоритет:** Средний
Требование: ПТ-3.5. Просмотр страницы задания.	
Описание тест-кейса: В процессе работы приложения проверить возможность сохранения ответа к тестовому заданию.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице «Главная» нажать на иконку страницы «Библиотека» в основном меню, находящимся в нижней части страницы. 3) В открывавшейся странице перейти во вкладку «Задании». 4) В данной вкладке открыть первое задание в «Раздел 1». 5) Выбрать вариант ответа и нажать на кнопку «Ответить». 6) Вернуться во вкладку «Задании». 7) Снова открыть первое задание в «Разделе 1».	Открывается страница с заданием, где присутствуют вопрос к заданию и предложенные варианты ответа, где выделен выбранный ранее вариант, и кнопка «Решить снова».

Таблица Г.26 – Тест-кейс № 26 | Проверка сохранения ответа к письменному заданию

ID: 26	**Приоритет:** Средний
Требование: ПТ-3.5. Просмотр страницы задания.	
Описание тест-кейса: В процессе работы приложения проверить возможность сохранения ответа к письменному заданию.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице «Главная» нажать на иконку страницы «Библиотека» в основном меню, находящимся в нижней части страницы. 3) В открывавшейся странице перейти во вкладку «Задании». 4) В данной вкладке открыть первое задание в «Раздел 2». 5) Ввести в поле для ввода ответ и нажать на кнопку «Ответить». 6) Вернуться во вкладку «Задании». 7) Снова открыть первое задание в «Разделе 2».	Открывается страница с заданием, где присутствуют вопрос к заданию и текстовое поле, содержащее ранее введенный ответ, и кнопка «Решить снова».

Таблица Г.27 – Тест-кейс № 27 | Проверка сброса ответа к тестовому заданию

ID: 27	**Приоритет:** Средний
Требование: ПТ-3.5. Просмотр страницы задания.	
Описание тест-кейса: В процессе работы приложения проверить возможность сброса ответа к тестовому заданию.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице «Главная» нажать на иконку страницы «Библиотека» в основном меню, находящимся в нижней части страницы. 3) В открывавшейся странице перейти во вкладку «Задании». 4) В данной вкладке открыть первое задание в «Раздел 1». 5) Выбрать вариант ответа и нажать на кнопку «Ответить». 6) Вернуться во вкладку «Задании». 7) Снова открыть первое задание в «Разделе 1». 8) Нажать на кнопку «Решить снова».	Ответ сбрасывается и на странице с заданием присутствуют вопрос к заданию и предложенные варианты ответа, где надо выбрать один вариант и кнопка «Ответить».».

Таблица Г.28 – Тест-кейс № 28 | Проверка сброса ответа к письменному заданию

ID: 28	**Приоритет:** Средний
Требование: ПТ-3.5. Просмотр страницы задания.	
Описание тест-кейса: В процессе работы приложения проверить возможность сброса ответа к письменному заданию.	
История редактирований	
Created on: 16.01.2024 by Санникова А.С.	Новый тест-кейс
Исполнительная часть	
Шаги	***Ожидаемый результат***
1) Запустить приложение. 2) На открывшейся странице «Главная» нажать на иконку страницы «Библиотека» в основном меню, находящимся в нижней части страницы. 3) В открывавшейся странице перейти во вкладку «Задании». 4) В данной вкладке открыть первое задание в «Раздел 2». 5) Ввести в поле для ввода ответ и нажать на кнопку «Ответить». 6) Вернуться во вкладку «Задании». 7) Снова открыть первое задание в «Разделе 2». 8) Нажать на кнопку «Решить снова».	Ответ сбрасывается и на странице с заданием присутствуют описание задачи и поле для ввода ответа, а также кнопка «Ответить».

Приложение Д

- Листинг классов, объекты которого хранятся в базе данных и входящих в компонент модели приложения.

```
using SQLite;
using SQLiteNetExtensions.Attributes;
using System;
using System.Collections.Generic;

namespace LearnCpp.Models
{
    [Table("Languages")]
    public class Language
    {
        [PrimaryKey, AutoIncrement, Column("LanguageID"), NotNull, Unique]
        public int LanguageID { get; set; }

        [NotNull, Unique]
        public string LanguageName { get; set; }

        [OneToMany(CascadeOperations = CascadeOperation.All)]
        public IEnumerable<DictionaryWord> words { get; set; }
    }

    [Table("DictionaryWords")]
    public class DictionaryWord
    {
        [PrimaryKey, AutoIncrement, Column("WordID"), NotNull, Unique]
        public int WordID { get; set; }

        [NotNull, Unique]
        public string Word { get; set; }

        [ForeignKey(typeof(Language)), NotNull]
        public int LanguageFK { get; set; }

        [ManyToOne(CascadeOperations = CascadeOperation.All)]
        public Language language { get; set; }

        [OneToMany(CascadeOperations = CascadeOperation.All)]
        public List<Meaning> meanings { get; set; }
    }

    [Table("Transcriptions")]
    public class Transcription
    {
        [PrimaryKey, AutoIncrement, Column("TranscriptionID"), NotNull, Unique]
        public int TranscriptionID { get; set; }

        [NotNull, Unique]
        public string TranscriptionItem { get; set; }

        [OneToMany(CascadeOperations = CascadeOperation.All)]
        public IEnumerable<Meaning> meanings { get; set; }
    }

    [Table("Meanings")]
    public class Meaning {
        [PrimaryKey, AutoIncrement, Column("MeaningID"), NotNull, Unique]
```

```
        public int MeaningID { get; set; }

        [ForeignKey(typeof(DictionaryWord)), NotNull]
        public int WordFK { get; set; }

        [ForeignKey(typeof(DictionaryWord)), NotNull]
        public int MeaningFK { get; set; }

        [ForeignKey(typeof(Transcription))]
        public int TransctriptionFK { get; set; }

        public bool IsFavorite { get; set; }
        public DateTime DateViewed { get; set; }

        [ManyToOne(nameof(WordFK), CascadeOperations = CascadeOperation.All)]
        public DictionaryWord word { get; set; }

        [ManyToOne(nameof(MeaningFK), CascadeOperations =
CascadeOperation.All)]
        public DictionaryWord meaningWord { get; set; }

        [ManyToOne(CascadeOperations = CascadeOperation.All)]
        public Transcription transcription { get; set; }

        [OneToMany(CascadeOperations = CascadeOperation.All)]
        public List<MeaningInLecture> lectures { get; set; }

        [OneToMany(CascadeOperations = CascadeOperation.All)]
        public List<MeaningInTask> tasks { get; set; }
    }

    [Table("Sections")]
    public class Section
    {
        [PrimaryKey, AutoIncrement, Column("SectionID")]
        public int SectionID { get; set; }

        public string SectionTitle { get; set; }

        [OneToMany(CascadeOperations = CascadeOperation.All)]
        public IEnumerable<Chapter> chapters { get; set; }

    }

    [Table("Chapters")]
    public class Chapter
    {
        [PrimaryKey, AutoIncrement, Column("ChapterID")]
        public int ChapterID { get; set; }
        public string ChapterTitle { get; set;}

        [ForeignKey(typeof(Section))]
        public int SectionFK { get; set; }

        [ManyToOne(CascadeOperations = CascadeOperation.All)]
        public Section section { get; set; }

        [OneToMany(CascadeOperations = CascadeOperation.All)]
        public IEnumerable<Topic> topics { get; set; }

    }
```

```
[Table("Topics")]
public class Topic
{
    [PrimaryKey, AutoIncrement, Column("TopicID")]
    public int TopicID { get; set; }
    public string TopicTitle { get; set; }

    [ForeignKey(typeof(Chapter))]
    public int ChapterFK { get; set; }

    [ManyToOne(CascadeOperations = CascadeOperation.All)]
    public Chapter chapter { get; set; }

    [OneToMany(CascadeOperations = CascadeOperation.All)]
    public IEnumerable<TaskByTopic> tasks { get; set; }

    [OneToMany(CascadeOperations = CascadeOperation.All)]
    public IEnumerable<LectureByTopic> lectures { get; set; }

}

[Table("Lectures")]
public class Lecture
{
    [PrimaryKey, AutoIncrement, Column("LectureID")]
    public int LectureID { get; set; }
    public string LectureTopic { get; set; }
    public string ChineseTopic { get; set; }

    [OneToMany(CascadeOperations = CascadeOperation.All)]
    public IEnumerable<LectureByTopic> lectures { get; set; }

    [OneToMany(CascadeOperations = CascadeOperation.All)]
    public IEnumerable<MeaningInLecture> meanings { get; set; }
}

[Table("PracticeTasks")]
public class PracticeTask
{
    [PrimaryKey, AutoIncrement, Column("TaskID")]
    public int TaskID { get; set; }
    public string TaskTopic { get; set; }
    public string TaskDescription { get; set; }
    public string SampleInput { get; set; }
    public string TaskKey { get; set; }
    public bool IsTest { get; set; }

    [OneToMany(CascadeOperations = CascadeOperation.All)]
    public IEnumerable<TaskByTopic> tasks { get; set; }

    [OneToMany(CascadeOperations = CascadeOperation.All)]
    public IEnumerable<MeaningInTask> meanings { get; set; }
}

[Table("TaskTests")]
public class TaskTest
{
    [PrimaryKey, AutoIncrement, Column("TestID")]
    public int TestID { get; set; }
    public string TestValue{ get; set; }

    [ForeignKey(typeof(PracticeTask))]
```

```
    public int TaskFK { get; set; }

    [ManyToOne(CascadeOperations = CascadeOperation.All)]
    public PracticeTask task { get; set; }
}

[Table("TaskByTopics")]
public class TaskByTopic
{
    [PrimaryKey, AutoIncrement, Column("TaskByTopicID")]
    public int TaskByTopicID { get; set; }

    [ForeignKey(typeof(Topic))]
    public int TopicFK { get; set; }

    [ForeignKey(typeof(PracticeTask))]
    public int TaskFK { get; set; }

    public string Solution { get; set; }
    public bool IsDone { get; set; }
    public bool Result { get; set; }
    public DateTime DateCopmplition { get; set; }

    [ManyToOne(CascadeOperations = CascadeOperation.All)]
    public Topic topic { get; set; }

    [ManyToOne(CascadeOperations = CascadeOperation.All)]
    public PracticeTask task { get; set; }
}

[Table("LectureByTopics")]
public class LectureByTopic
{
    [PrimaryKey, AutoIncrement, Column("LectureByTopicID")]
    public int LectureByTopicID { get; set; }

    [ForeignKey(typeof(Topic))]
    public int TopicFK { get; set; }

    [ForeignKey(typeof(Lecture))]
    public int LectureFK { get; set; }
    public DateTime DateViewed { get; set; }

    [ManyToOne(CascadeOperations = CascadeOperation.All)]
    public Topic topic { get; set; }

    [ManyToOne(CascadeOperations = CascadeOperation.All)]
    public Lecture lecture { get; set; }
}

[Table("MeaningInLectures")]
public class MeaningInLecture
{
    [PrimaryKey, AutoIncrement, Column("MeaningInLectureID")]
    public int MeaningInLectureID { get; set; }

    [ForeignKey(typeof(Meaning))]
    public int MeaningFK { get; set; }

    [ForeignKey(typeof(Lecture))]
    public int LectureFK { get; set; }
```

```
        [ManyToOne(CascadeOperations = CascadeOperation.All)]
        public Meaning meaning { get; set; }

        [ManyToOne(CascadeOperations = CascadeOperation.All)]
        public Lecture lecture { get; set; }
    }

    [Table("MeaningInTasks")]
    public class MeaningInTask
    {
        [PrimaryKey, AutoIncrement, Column("MeaningInTaskID")]
        public int MeaningInTaskID { get; set; }

        [ForeignKey(typeof(Meaning))]
        public int MeaningFK { get; set; }

        [ForeignKey(typeof(PracticeTask))]
        public int TaskFK { get; set; }

        [ManyToOne(CascadeOperations = CascadeOperation.All)]
        public Meaning meaning { get; set; }

        [ManyToOne(CascadeOperations = CascadeOperation.All)]
        public PracticeTask practiceTask { get; set; }
    }
}
```

- Листинг класса репозитория, благодаря которому есть возможность совершать все операции с данными с помощью асинхронных методов, определенные в классе SQLiteAsyncConnection. В конструкторе происходит создание нового подключения к базе данных.

```
using SQLite;
using SQLiteNetExtensionsAsync.Extensions;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace LearnCpp.Models
{
    public class DatabaseRepository
    {
        SQLiteAsyncConnection database;

        public DatabaseRepository(string databasePath)
        {
            database = new SQLiteAsyncConnection(databasePath);
        }

        #region методы для работы с объектом Language
        public async Task CreateLanguage() => await
database.CreateTableAsync<Language>();
        public async Task<IEnumerable<Language>> GetAllLanguages() => await
database.Table<Language>().ToListAsync();
        public async Task<Language> GetLanguage(int id) => await
database.GetAsync<Language>(id);
        public async Task<int> DeleteLanguage(Language item) => await
database.DeleteAsync(item);
        public async Task<int> SaveLanguage(Language item)
        {
            if (item.LanguageID != 0)
```

```
            {
                await database.UpdateAsync(item);
                return item.LanguageID;
            }
            else
            {
                return await database.InsertAsync(item);
            }
        }
        #endregion

        #region методы для работы с объектом DictionaryWord
        public async Task CreateDictionary() => await
database.CreateTableAsync<DictionaryWord>();
        public async Task<IEnumerable<DictionaryWord>> GetAllWords() => await
database.Table<DictionaryWord>().ToListAsync();
        public async Task<DictionaryWord> GetWord(int id) => await
database.GetAsync<DictionaryWord>(id);
        public async Task<int> DeleteWord(DictionaryWord item) => await
database.DeleteAsync(item);
        public async Task<int> SaveWord(DictionaryWord item)
        {
            if (item.WordID != 0)
            {
                await database.UpdateAsync(item);
                return item.WordID;
            }
            else
            {
                return await database.InsertAsync(item);
            }
        }
        #endregion

        #region методы для работы с объектом Transcription
        public async Task CreateTranscription() => await
database.CreateTableAsync<Transcription>();
        public async Task<IEnumerable<Transcription>> GetAllTranscriptions() =>
await database.Table<Transcription>().ToListAsync();
        public async Task<Transcription> GetTranscription(int id) => await
database.GetAsync<Transcription>(id);
        public async Task<int> DeleteTranscriprion(Transcription item) => await
database.DeleteAsync(item);
        public async Task<int> SaveTranscription(Transcription item)
        {
            if (item.TranscriptionID != 0)
            {
                await database.UpdateAsync(item);
                return item.TranscriptionID;
            }
            else
            {
                return await database.InsertAsync(item);
            }
        }
        #endregion

        #region методы для работы с объектом Meaning
        public async Task CreateMeaning() => await
database.CreateTableAsync<Meaning>();
        public async Task<IEnumerable<Meaning>> GetAllMeanings() => await
database.Table<Meaning>().ToListAsync();
```

```
        public async Task<Meaning> GetMeaning(int id) => await
database.GetAsync<Meaning>(id);
        public async Task<int> DeleteMeaning(Meaning item) => await
database.DeleteAsync(item);
        public async Task<int> SaveMeaning(Meaning item)
        {
            if (item.MeaningID != 0)
            {
                await database.UpdateAsync(item);
                return item.MeaningID;
            }
            else
            {
                return await database.InsertAsync(item);
            }
        }
        public async Task<List<Meaning>> GetAllMeaningWithChildren() => await
database.GetAllWithChildrenAsync<Meaning>();
        #endregion

        #region методы для работы с объектом Section
        public async Task CreateSection() => await
database.CreateTableAsync<Section>();
        public async Task<IEnumerable<Section>> GetAllSections() => await
database.Table<Section>().ToListAsync();
        public async Task<Section> GetSection(int id) => await
database.GetAsync<Section>(id);
        public async Task<int> DeleteSection(Section item) => await
database.DeleteAsync(item);
        public async Task<int> SaveSection(Section item)
        {
            if (item.SectionID != 0)
            {
                await database.UpdateAsync(item);
                return item.SectionID;
            }
            else
            {
                return await database.InsertAsync(item);
            }
        }

        #endregion

        #region методы для работы с объектом Chapter
        public async Task CreateChapter() => await
database.CreateTableAsync<Chapter>();
        public async Task<IEnumerable<Chapter>> GetAllChapters() => await
database.Table<Chapter>().ToListAsync();
        public async Task<Chapter> GetChapter(int id) => await
database.GetAsync<Chapter>(id);
        public async Task<int> DeleteChapter(Chapter item) => await
database.DeleteAsync(item);
        public async Task<int> SaveChapter(Chapter item)
        {
            if (item.ChapterID != 0)
            {
                await database.UpdateAsync(item);
                return item.ChapterID;
            }
            else
            {
```

```
                return await database.InsertAsync(item);
            }
        }
        #endregion

        #region методы для работы с объектом Topic
        public async Task CreateTopic() => await
database.CreateTableAsync<Topic>();
        public async Task<IEnumerable<Topic>> GetAllTopics() => await
database.Table<Topic>().ToListAsync();
        public async Task<Topic> GetTopic(int id) => await
database.GetAsync<Topic>(id);
        public async Task<int> DeleteTopic(Topic item) => await
database.DeleteAsync(item);
        public async Task<int> SaveTopic(Topic item)
        {
            if (item.TopicID != 0)
            {
                await database.UpdateAsync(item);
                return item.TopicID;
            }
            else
            {
                return await database.InsertAsync(item);
            }
        }
        #endregion

        #region методы для работы с объектом Lecture
        public async Task CreateLecture() => await
database.CreateTableAsync<Lecture>();
        public async Task<IEnumerable<Lecture>> GetAllLectures() => await
database.Table<Lecture>().ToListAsync();
        public async Task<Lecture> GetLecture(int id) => await
database.GetAsync<Lecture>(id);
        public async Task<int> DeleteLecture(Lecture item) => await
database.DeleteAsync(item);
        public async Task<int> SaveLecture(Lecture item)
        {
            if (item.LectureID != 0)
            {
                await database.UpdateAsync(item);
                return item.LectureID;
            }
            else
            {
                return await database.InsertAsync(item);
            }
        }
        #endregion

        #region методы для работы с объектом PracticeTask
        public async Task CreateTask() => await
database.CreateTableAsync<PracticeTask>();
        public async Task<IEnumerable<PracticeTask>> GetAllTasks() => await
database.Table<PracticeTask>().ToListAsync();
        public async Task<PracticeTask> GetTask(int id) => await
database.GetAsync<PracticeTask>(id);
        public async Task<int> DeleteTask(PracticeTask item) => await
database.DeleteAsync(item);
        public async Task<int> SaveTasks(PracticeTask item)
        {
```

```
            if (item.TaskID != 0)
            {
                await database.UpdateAsync(item);
                return item.TaskID;
            }
            else
            {
                return await database.InsertAsync(item);
            }
        }
        #endregion

        #region методы для работы с объектом TestTask
        public async Task CreateTestTask() => await
database.CreateTableAsync<TaskTest>();
        public async Task<IEnumerable<TaskTest>> GetAllTest() => await
database.Table<TaskTest>().ToListAsync();
        public async Task<TaskTest> GetTest(int id) => await
database.GetAsync<TaskTest>(id);
        public async Task<int> DeleteTest(TaskTest item) => await
database.DeleteAsync(item);
        public async Task<int> SaveTest(TaskTest item)
        {
            if (item.TestID != 0)
            {
                await database.UpdateAsync(item);
                return item.TestID;
            }
            else
            {
                return await database.InsertAsync(item);
            }
        }
        #endregion

        #region методы для работы с объектом TaskByTopic
        public async Task CreateTaskByTopic() => await
database.CreateTableAsync<TaskByTopic>();
        public async Task<IEnumerable<TaskByTopic>> GetAllTasksByTopic() =>
await database.Table<TaskByTopic>().ToListAsync();
        public async Task<TaskByTopic> GetTaskByTopic(int id) => await
database.GetAsync<TaskByTopic>(id);
        public async Task<int> DeleteTaskByTopic(TaskByTopic item) => await
database.DeleteAsync(item);
        public async Task<int> SaveTaskByTopic(TaskByTopic item)
        {
            if (item.TaskByTopicID != 0)
            {
                await database.UpdateAsync(item);
                return item.TaskByTopicID;
            }
            else
            {
                return await database.InsertAsync(item);
            }
        }
        public async Task<List<TaskByTopic>> TaskByTopicWithChildren() => await
database.GetAllWithChildrenAsync<TaskByTopic>();
        #endregion

        #region lectureByTopic
```

```
        public async Task CreateLectureByTopic() => await
database.CreateTableAsync<LectureByTopic>();
        public async Task<IEnumerable<LectureByTopic>> GetAllLecturesByTopic()
=> await database.Table<LectureByTopic>().ToListAsync();
        public async Task<LectureByTopic> GetLecturByTopic(int id) => await
database.GetAsync<LectureByTopic>(id);
        public async Task<int> DeleteLectureByTopic(LectureByTopic item) =>
await database.DeleteAsync(item);
        public async Task<int> SaveLectureByTopic(LectureByTopic item)
        {
            if (item.LectureByTopicID != 0)
            {
                await database.UpdateAsync(item);
                return item.LectureByTopicID;
            }
            else
            {
                return await database.InsertAsync(item);
            }
        }
        public async Task<List<LectureByTopic>> LectureByTopicWithChildren() =>
await database.GetAllWithChildrenAsync<LectureByTopic>();
        #endregion

        #region методы для работы с объектом MeaningInLecture
        public async Task CreateMeaningInLecture() => await
database.CreateTableAsync<MeaningInLecture>();
        public async Task<IEnumerable<MeaningInLecture>>
GetAllMeaningInLecture() => await
database.Table<MeaningInLecture>().ToListAsync();
        public async Task<MeaningInLecture> GetMeaningInLecture(int id) =>
await database.GetAsync<MeaningInLecture>(id);
        public async Task<int> DeleteMeaningInLecture(MeaningInLecture item) =>
await database.DeleteAsync(item);
        public async Task<int> SaveMeaningInLecture(MeaningInLecture item)
        {
            if (item.MeaningInLectureID != 0)
            {
                await database.UpdateAsync(item);
                return item.MeaningInLectureID;
            }
            else
            {
                return await database.InsertAsync(item);
            }
        }
        public async Task<List<MeaningInLecture>>
MeaningInLectureWithChildren() => await
database.GetAllWithChildrenAsync<MeaningInLecture>();
        #endregion

        #region методы для работы с объектом MeaningInTask
        public async Task CreateMeaningInTask() => await
database.CreateTableAsync<MeaningInTask>();
        public async Task<IEnumerable<MeaningInTask>> GetAllMeaningInTask() =>
await database.Table<MeaningInTask>().ToListAsync();
        public async Task<MeaningInTask> GetMeaningInTask(int id) => await
database.GetAsync<MeaningInTask>(id);
        public async Task<int> DeleteMeaningInTask(MeaningInTask item) => await
database.DeleteAsync(item);
        public async Task<int> SaveMeaningInTask(MeaningInTask item)
        {
```

```
            if (item.MeaningInTaskID != 0)
            {
                await database.UpdateAsync(item);
                return item.MeaningInTaskID;
            }
            else
            {
                return await database.InsertAsync(item);
            }
        }
        public async Task<List<MeaningInTask>> MeaningInTaskWithChildren() =>
await database.GetAllWithChildrenAsync<MeaningInTask>();
        #endregion
    }
}
```

- Листинг класса BaseViewModel, реализующий интерфейс INotifyPropertyChanged и от которого наследуются остальные классы модели представления.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Runtime.CompilerServices;

namespace LearnCpp.ViewModels
{
    public class BaseViewModel : INotifyPropertyChanged
    {
        bool isBusy = false;
        public bool IsBusy //определение статуса экземпляра класса
        {
            get { return isBusy; }
            set { SetProperty(ref isBusy, value); }
        }

        string title = string.Empty;
        public string Title // свойство для наименования страниц приложения
        {
            get { return title; }
            set { SetProperty(ref title, value); }
        }

        protected bool SetProperty<T>(ref T backingStore, T value,
            [CallerMemberName] string propertyName = "",
            Action onChanged = null)
        {
            if (EqualityComparer<T>.Default.Equals(backingStore, value))
                return false;

            backingStore = value;
            onChanged?.Invoke();
            OnPropertyChanged(propertyName);
            return true;
        }

        #region реализация INotifyPropertyChanged. Метод OnPropetyChanged
проверяет и сообщает об изменениях.
        public event PropertyChangedEventHandler PropertyChanged;
        protected void OnPropertyChanged([CallerMemberName] string propertyName
= "")
```

```
        {
            var changed = PropertyChanged;
            if (changed == null)
                return;

            changed.Invoke(this, new PropertyChangedEventArgs(propertyName));
        }
        #endregion
    }
}
```

- Листинг класса MainViewModel – модель представления, к которому привязано представление MainPage.

```
using LearnCpp.Models;
using LearnCpp.Views;
using System;
using System.Collections.ObjectModel;
using System.Diagnostics;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Input;
using Xamarin.Forms;

namespace LearnCpp.ViewModels
{
    public class MainViewModel : BaseViewModel
    {
        private Meaning selectedMeaning;
        private Lecture selectedLecture;
        public Meaning SelectedMeaning //выбранное значение
        {
            get => selectedMeaning;
            set
            {
                SetProperty(ref selectedMeaning, value);
                OnMeaningSelected(value);
            }
        }
        public Lecture SelectedLecture //выбранная лекция
        {
            get => selectedLecture;
            set
            {
                SetProperty(ref selectedLecture, value);
                OpenLecture(value);
            }
        }
        public ObservableCollection<Meaning> Meanings { get; } //недавно
просмотренные слова
        public ObservableCollection<Meaning> searchMeaning { get; } //найденные
слова
        public ObservableCollection<Lecture> searchLecture { get; } //найденные
лекции

        private string lectureLast;
        public string LectureLast // последняя просмотренная лекция
        {
            get => lectureLast;
            set => SetProperty(ref lectureLast, value);
        }
```

```
        private int heightSearch;
        private int heightLast;
//свойство для отображения контейнера, содержащего результаты поиска
        public int HeightSearch
        {
            get => heightSearch;
            set => SetProperty(ref heightSearch, value);
        }
//свойство для отображения контейнера, содержащего недавно просмотренные слова
и лекцию
        public int HeightLast
        {
            get => heightLast;
            set => SetProperty(ref heightLast, value);
        }
        private string searchText = string.Empty;
        public string SearchText //параметры поиска
        {
            get => searchText;
            set
            {
                if (searchText != value)
                {
                    searchText = value ?? string.Empty;
                    OnPropertyChanged(nameof(SearchText));

                    if (SearchCommand.CanExecute(null))
                    {
                        SearchCommand.Execute(null);
                    }
                }
            }
        }
        public ICommand LoadSettingsCommand { get; }
        public Command LoadLastInfo {  get; }
        public Command<Meaning> MeaningTapped { get; }
        public Command<Lecture> LectureTapped { get; }
        public ICommand TapLecture { get; }
        private ICommand searchCommand;
        public ICommand SearchCommand
        {
            get
            {
                searchCommand = searchCommand ?? new
Command(ExecuteSearchCommand, CanExecuteSearchCommand);
                return searchCommand;
            }
        }
        public MainViewModel()
        {
            Title = "Главная";
            Meanings = new ObservableCollection<Meaning>();
            searchMeaning = new ObservableCollection<Meaning>();
            searchLecture = new ObservableCollection<Lecture>();
            LoadLastInfo = new Command(async () => await
ExecuteLoadLastInfo());
            LoadSettingsCommand = new Command(OpenSettings);

            MeaningTapped = new Command<Meaning>(OnMeaningSelected);
            LectureTapped = new Command<Lecture>(OnLectureSelected);
            TapLecture = new Command(OpenLecture);
        }
```

```
        private async void OnLectureSelected(Lecture lecture)
        {
            if (lecture == null)
                return;

            await Shell.Current.GoToAsync($"{nameof(LecturePage)}?
{nameof(LectureViewModel.LectureTopic)}={lecture.LectureTopic}");
        }

        private async void OpenSettings(object obj)
        {
            await Shell.Current.GoToAsync(nameof(SettingPage));
        }

        private async void OpenLecture(object obj)
        {
            if (String.IsNullOrEmpty(LectureLast))
                return;

            await Shell.Current.GoToAsync($"{nameof(LecturePage)}?
{nameof(LectureViewModel.LectureTopic)}={LectureLast}");
        }

        private async void OnMeaningSelected(Meaning meaning)
        {
            if (meaning == null)
                return;

            await Shell.Current.GoToAsync($"{nameof(WordDetailPage)}?
{nameof(WordDetailViewModel.MeaningId)}={meaning.MeaningID}");
        }

        private async void ExecuteSearchCommand()
        {
            if (!string.IsNullOrEmpty(SearchText))
            {
                HeightLast = 0;
                HeightSearch = 1000;
                try
                {
                    searchMeaning.Clear();
                    searchLecture.Clear();

                    var words = (await
App.Database.GetAllMeaningWithChildren()).Where(x =>
x.word.Word.ToLower().Contains(SearchText.Trim().ToLower()) ||
x.meaningWord.Word.Contains(SearchText.Trim()));
                    var lectures = (await
App.Database.GetAllLectures()).Where(x =>
x.LectureTopic.ToLower().Contains(SearchText.Trim().ToLower()) ||
x.ChineseTopic.Contains(SearchText.Trim()));

                    foreach (var meaning in words)
                    {
                        searchMeaning.Add(meaning);
                    }

                    foreach (var lecture in lectures)
                    {
                        searchLecture.Add(lecture);
                    }
                }
```

```
                catch (Exception)
                {
                    Debug.WriteLine("Ничего не найдено");
                }
            }
            else
            {
                HeightLast = 400;
                HeightSearch = 0;
            }
        }

        private bool CanExecuteSearchCommand() => true;

        async Task ExecuteLoadLastInfo()
        {
            IsBusy = true;
            try
            {
                Meanings.Clear();
                LectureLast = (await
App.Database.LectureByTopicWithChildren()).OrderBy(x =>
x.DateViewed).Last().lecture.LectureTopic;
                var words = (await
App.Database.GetAllMeaningWithChildren()).OrderByDescending(x =>
x.DateViewed).Take(5);

                foreach (var meaning in words)
                {
                    Meanings.Add(meaning);
                }
            }
            catch (Exception)
            {
                Debug.WriteLine("Не удалось загрузить информацию");
            }
            finally
            {
                IsBusy = false;
            }
        }

        public void OnAppearing()
        {
            IsBusy = true ;
            SelectedMeaning = null;
            HeightLast = 400;
            HeightSearch = 0;
        }
    }
}
```

- Листинг класса представление MainPage, где модель представления MainView связывается со свойством BindingContext.

```
using LearnCpp.ViewModels;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace LearnCpp.Views
```

```
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
      public partial class MainPage : ContentPage
      {
        MainViewModel viewModel;
        public MainPage ()
      {
            InitializeComponent ();
           BindingContext = viewModel = new MainViewModel ();
        }

        protected override void OnAppearing()
        {
            viewModel.OnAppearing();
            base.OnAppearing();
        }
    }
}
```

- Листинг разметки визуального интерфейса класса MainPage, где также происходит привязка методов модели представления MainViewModel.

```
<?xml version="1.0" encoding="utf-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="LearnCpp.Views.MainPage"
             Title="{Binding Title}"
             xmlns:local="clr-namespace:LearnCpp.ViewModels"
             xmlns:model="clr-namespace:LearnCpp.Models">

    <ContentPage.ToolbarItems>
        <ToolbarItem IconImageSource="icon_gear.png" Command="{Binding
LoadSettingsCommand}" x:DataType="local:MainViewModel"/>
    </ContentPage.ToolbarItems>
    <ContentPage.Content>
        <StackLayout Orientation="Vertical" VerticalOptions="StartAndExpand"
Padding="10" x:DataType="local:MainViewModel">
            <Frame Style="{StaticResource boxes}" CornerRadius="20"
WidthRequest="500" MinimumHeightRequest="100">
                <SearchBar Style="{StaticResource searchBox}" Text="{Binding
SearchText}" SearchCommand="{Binding SearchCommand}"/>
            </Frame>
            <!--В данном контейнере списки с результатами поиска-->
            <StackLayout Orientation="Vertical" VerticalOptions="FillAndExpand"
HorizontalOptions="FillAndExpand" HeightRequest="{Binding HeightSearch}">

                <Label Text="Найденные слова" Style="{StaticResource
topicTxt}"/>
                <ListView ItemsSource="{Binding searchMeaning}"
SeparatorVisibility="Default" HasUnevenRows="True">
                    <ListView.ItemTemplate>
                        <DataTemplate>
                            <ViewCell>
                                <StackLayout x:DataType="model:Meaning"
Padding="10">
                                    <Label Text="{Binding word.Word}"
Style="{StaticResource lectureTxt}"/>
                                    <Label Text="{Binding meaningWord.Word}"
Style="{StaticResource normalTxt}" HorizontalOptions="Start"/>
                                    <StackLayout.GestureRecognizers>
```

```
                                    <TapGestureRecognizer
Command="{Binding Source={RelativeSource AncestorType={x:Type
local:MainViewModel}}, Path=MeaningTapped}"

CommandParameter="{Binding .}">
                                    </TapGestureRecognizer>
                                </StackLayout.GestureRecognizers>
                            </StackLayout>
                        </ViewCell>
                    </DataTemplate>
                </ListView.ItemTemplate>
            </ListView>
            <Label Text="Найденные лекции" Style="{StaticResource
topicTxt}"/>
            <ListView ItemsSource="{Binding searchLecture}"
SeparatorVisibility="Default">
                <ListView.ItemTemplate>
                    <DataTemplate>
                        <ViewCell>
                            <StackLayout x:DataType="model:Lecture"
Padding="10">
                                <Label Text="{Binding LectureTopic}"
Style="{StaticResource lectureTxt}"/>
                                <StackLayout.GestureRecognizers>
                                    <TapGestureRecognizer
Command="{Binding Source={RelativeSource AncestorType={x:Type
local:MainViewModel}}, Path=LectureTapped}"

CommandParameter="{Binding .}">
                                    </TapGestureRecognizer>
                                </StackLayout.GestureRecognizers>
                            </StackLayout>
                        </ViewCell>
                    </DataTemplate>
                </ListView.ItemTemplate>
            </ListView>
        </StackLayout>
        <!-- В этом контейнере последние просмотренные слова и лекция-->
        <RefreshView x:DataType="local:MainViewModel" Command="{Binding
LoadLastInfo}" IsRefreshing="{Binding IsBusy, Mode=TwoWay}">
            <StackLayout Orientation="Vertical"
VerticalOptions="FillAndExpand" HorizontalOptions="FillAndExpand"
HeightRequest="{Binding HeightLast}">
                <StackLayout Orientation="Vertical" Padding="10">
                    <Label Text="Последняя просмотренная лекция"
Style="{StaticResource topicTxt}"/>
                    <Button Style="{StaticResource choiceBtn}"
HeightRequest="70"
                        Text="{Binding LectureLast}" Command="{Binding
TapLecture}"
                            CommandParameter="{Binding LectureLast}"/>
                </StackLayout>
                <StackLayout Orientation="Vertical" Padding="10">
                    <Label Text="Недавно просмотренные слова"
Style="{StaticResource topicTxt}"/>
                    <ListView x:Name="recentlyDictionary"
ItemsSource="{Binding Meanings}" >
                        <ListView.ItemTemplate>
                            <DataTemplate>
                                <ViewCell>
                                    <StackLayout x:DataType="model:Meaning"
VerticalOptions="Center" Padding="10">
```

```
                                        <Label Text="{Binding  word.Word}"
Style="{StaticResource lectureTxt}"/>
                                        <StackLayout.GestureRecognizers>
                                            <TapGestureRecognizer
Command="{Binding Source={RelativeSource AncestorType={x:Type
local:MainViewModel}}, Path=MeaningTapped}"
                                        CommandParameter="{Binding .}">
                                            </TapGestureRecognizer>
                                        </StackLayout.GestureRecognizers>
                                    </StackLayout>
                                </ViewCell>
                            </DataTemplate>
                        </ListView.ItemTemplate>
                    </ListView>
                </StackLayout>
            </StackLayout>
        </RefreshView>
    </StackLayout>
</ContentPage.Content>
</ContentPage>
```

- Листинг класса LectureListViewModel – модель представления, к которому привязано представление LectureListPage.

```
using LearnCpp.Models;
using LearnCpp.Views;
using System;
using System.Collections.ObjectModel;
using System.Diagnostics;
using System.Linq;
using Xamarin.Forms;

namespace LearnCpp.ViewModels
{
    public class LectureListViewModel : BaseViewModel
    {
        private Lecture selectedLecture;
        public Lecture SelectedLecture
        {
            get => selectedLecture;
            set
            {
                SetProperty(ref selectedLecture, value);
                OpenLecture(value);
            }
        }

        public ObservableCollection<IGrouping<string, LectureByTopic>>
LectureList { get; }
        public Command<Lecture> TapLecture { get; }

        public LectureListViewModel()
        {
            Title = "Лекции";
            LectureList = new ObservableCollection<IGrouping<string,
LectureByTopic>>();
            LoadLectureListAsync();
            TapLecture = new Command<Lecture>(OpenLecture);
        }

        private async void LoadLectureListAsync()
```

```
        {
            try
            {
                var lectureList = (await
App.Database.LectureByTopicWithChildren()).GroupBy(x => x.topic.TopicTitle);
                foreach (var item in lectureList)
                {
                    LectureList.Add(item);
                }
            }
            catch (Exception)
            {
                Debug.WriteLine("Не удалось загрузить страницу");
            }
        }

        private async void OpenLecture(Lecture lecture)
        {
            if (lecture == null)
                return;

            await Shell.Current.GoToAsync($"{nameof(LecturePage)}?
{nameof(LectureViewModel.LectureTopic)}={lecture.LectureTopic}");
        }

        public void OnAppearing(){
            SelectedLecture = null;
        }
    }
}
```

- Листинг класса LectureViewModel – модель представления, к которому привязано представление LecturePage.

```
using Android.Content.Res;
using LearnCpp.Models;
using LearnCpp.Views;
using System;
using System.Diagnostics;
using System.IO;
using System.Linq;
using System.Windows.Input;
using VersOne.Epub;
using Xamarin.Forms;

namespace LearnCpp.ViewModels
{
    [QueryProperty(nameof(LectureTopic), nameof(LectureTopic))]
    public class LectureViewModel : BaseViewModel
    {
        private string lectureTopic;
        public string LectureTopic{
            get => lectureTopic;
            set{
                lectureTopic = value;
                LoadLectureTopic(value);
            }
        }

        public LectureByTopic lecture = new LectureByTopic();
        public ICommand LoadMeaningsCommand { get; }
        public HtmlWebViewSource viewSource { get; set; }
```

```
        public LectureViewModel()
        {
            viewSource = new HtmlWebViewSource();
            LoadMeaningsCommand = new Command(OpenMeaningsInLecture);
        }

        private async void OpenMeaningsInLecture(object obj) => await
Shell.Current.GoToAsync($"{nameof(MeaningInLecturePage)}?{nameof(MeaningInLectu
reViewModel.LectureFk)}={lecture.LectureFK}");
        public async void LoadLectureTopic(string lectureTopic)
        {
            try
            {
                AssetManager assets = Android.App.Application.Context.Assets;
                Stream stream = assets.Open("LectureCpp.epub");
                EpubBook epubBook = EpubReader.ReadBook(stream);

                foreach (var navigationItem in epubBook.Navigation)
                {
                    foreach(var item in navigationItem.NestedItems)
                    {
                        if (item.Title == lectureTopic)
                        {
                            viewSource.BaseUrl =
DependencyService.Get<IBaseUrl>().Get();
                            viewSource.Html = item.HtmlContentFile.Content;
                            Title = lectureTopic;

                            await App.Database.CreateLectureByTopic();
                            lecture = (await
App.Database.LectureByTopicWithChildren()).Where(x=>x.lecture.LectureTopic ==
lectureTopic).FirstOrDefault();
                            lecture.DateViewed = DateTime.Now;
                            await App.Database.SaveLectureByTopic(lecture);
                        }
                    }
                }
            }
            catch (Exception)
            {
                Debug.WriteLine("Не удалось загрузить лекцию");
            }
        }
        public interface IBaseUrl { string Get(); }
    }
}
```

- Листинг класса WordDetailViewModel – модель представления, к которому привязано представление WordDetailPage.

```
using LearnCpp.Models;
using System;
using System.Diagnostics;
using System.Windows.Input;
using Xamarin.Forms;

namespace LearnCpp.ViewModels
{
    [QueryProperty(nameof(MeaningId), nameof(MeaningId))]
    public class WordDetailViewModel: BaseViewModel
    {
        private int meaningId;
```

```
        private string word;
        private string meaning;
        public Meaning meaningWord = new Meaning();
        private string btnTxt;
        public string BtnTxt {
            get => btnTxt;
            set => SetProperty(ref btnTxt, value);
        }
        public ICommand FavoriteBtn { get => new Command(FavoriteCommand); }
        private async void FavoriteCommand(object obj)
        {
            if (meaningWord.IsFavorite) {
                meaningWord.IsFavorite = false;
                await App.Database.SaveMeaning(meaningWord);
                BtnTxt = "Добавить в избранное";
            }
            else {
                meaningWord.IsFavorite = true;
                await App.Database.SaveMeaning(meaningWord);
                BtnTxt = "Удалить из избранного";
            }
        }
        public string Word {
            get => word;
            set => SetProperty(ref word, value);
        }
        public string WordMeaning
        {
            get => meaning;
            set => SetProperty(ref meaning, value);
        }
        public int MeaningId
        {
            get => meaningId;
            set
            {
                meaningId = value;
                LoadMeaningId(value);
            }
        }
        public async void LoadMeaningId(int meaningId)
        {
            try
            {
                await App.Database.CreateMeaning();
                meaningWord = await App.Database.GetMeaning(meaningId);
                Word = (await App.Database.GetWord(meaningWord.WordFK)).Word;
                WordMeaning = (await
App.Database.GetWord(meaningWord.MeaningFK)).Word;

                meaningWord.DateViewed = DateTime.Now;
                await App.Database.SaveMeaning(meaningWord);
                if (meaningWord.IsFavorite) BtnTxt = "Удалить из избранного";
                else BtnTxt = "Добавить в избранное";
            }
            catch (Exception)
            {
                Debug.WriteLine("Не удалось загрузить страницу");
            }
        }
    }
}
```