

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

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

Институт _____ информатики и кибернетики _____

Кафедра _____ программных систем _____

ОТЧЕТ ПО ПРАКТИКЕ

Вид практики _____ производственная _____
(учебная, производственная)

Тип практики _____ научно-исследовательская работа _____
(в соответствии с ОПОП ВО)

Сроки прохождения практики: с 01.10.2024 по 26.12.2024
(в соответствии с календарным учебным графиком)

по направлению подготовки 02.03.02
«Фундаментальная информатика и информационные технологии
(уровень бакалавриата)
направленность (профиль) «Информационные технологии»

Обучающийся группы № 6401-020302D _____ А.А. Алёнушка

Руководитель практики,
Доцент кафедры программных систем,
к.т.н, доцент _____ О.А. Гордеева

Дата сдачи 26.12.2024
Дата защиты 26.12.2024

Оценка _____

Самара 2024

СОДЕРЖАНИЕ

Задания по практике для выполнения определенных видов работ, связанных с будущей профессиональной деятельностью (сбор и анализ данных и материалов, проведение исследований)	3
ВВЕДЕНИЕ	8
1 Описание средств реализации	10
1.1 Описание операционной системы	10
1.2 Описание языка программирования	10
1.3 Описание среды разработки	11
1.4 Описание используемых библиотек	12
2 Описание проекта разрабатываемого приложения.....	13
2.1 The Super Memo	14
2.2 Алгоритм Anki.....	16
2.3 Система Duolingo	17
2.4 База данных	18
2.5 Диаграмма use case	19
3 Описание экранных форм разработанного программного приложения	21
3.1 Начало работы: регистрация и вход.....	21
3.2 Работа с коллекциями карточек.....	22
3.3 Добавление и редактирование карточек.....	23
3.4 Редактирование коллекций и поиск карточек.....	23
3.5 Повторение карточек.....	24
ЗАКЛЮЧЕНИЕ.....	26
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	27

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ

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

Институт _____ информатики и кибернетики _____

Кафедра _____ программных систем _____

Задания по практике для выполнения определенных видов работ, связанных с
будущей профессиональной деятельностью
(сбор и анализ данных и материалов, проведение исследований)

Обучающемуся Алёнушка Александру Александровичу
группы 6401-020302D

Направлен на практику приказом по университету от 27.09.2024 г. №451-ПР
на _____ кафедру программных систем _____

Тема: Разработка веб-приложения для обучения по системе Лейтнера с
реализацией алгоритма интервального повторения

Планируемые результаты освоения образовательной программы (компетенции)	Выполнение определенных видов работ, связанных с будущей профессиональной деятельностью (сбор и анализ данных и материалов, проведение исследований)	Результаты практики
ОПК-1. Способен применять фундаментальные знания, полученные в области математических и (или) естественных наук, и	Исследовать современные подходы и технологии создания веб-приложений с использованием	Обзор существующих библиотек и программных средств для

<p>использовать их в профессиональной деятельности.</p> <p>ОПК-1.1. Использует основные положения и концепции в области математических и естественных наук, Базовые теории и истории основного, теории коммуникации; знает основную терминологию.</p> <p>ОПК-1.2. Осуществляет первичный сбор и анализ материала, интерпретирует различные математические объекты.</p> <p>ОПК-1.3. Применяет опыт решения стандартных математических задач в профессиональной деятельности.</p>	<p>универсальных программных средств. Исследовать существующие алгоритмы интервального повторения.</p>	<p>разработки веб-приложений и алгоритмов интервального повторения.</p>
<p>ОПК-2. Способен применять компьютерные/суперкомпьютерные методы, современное программное обеспечение, в том числе отечественного происхождения, для решения задач профессиональной деятельности.</p> <p>ОПК-2.1. Использует основные положения и концепции в области программирования, архитектуры языков программирования, теории коммуникации, знает основную терминологию, знаком с содержанием Единого Реестра Российских программ.</p> <p>ОПК-2.2. Анализирует код на типовых языках программирования, может составлять программы.</p> <p>ОПК-2.3. Применяет опыт решения задач анализа,</p>	<p>Изучить возможности и основные подходы Spring Boot для создания веб-приложений, библиотеки для работы с миграциями Liquibase, библиотеки для работы с запросами к базе данных MyBatis.</p>	<p>Изучены и освоены следующие программные средства: Spring Boot, Liquibase, MyBatis.</p>

интеграции различных типов программного обеспечения, анализа типов коммуникаций.		
<p>ОПК-3. Способен к разработке алгоритмических и программных решений в области системного и прикладного программирования, математических, информационных и имитационных моделей, созданию информационных ресурсов глобальных сетей, образовательного контента, прикладных баз данных, тестов и средств тестирования систем и средств на соответствие стандартам и исходным требованиям.</p> <p>ОПК-3.1. Понимает методы теории алгоритмов, методы системного и прикладного программирования, основные положения и концепции в области математических, информационных и имитационных моделей.</p> <p>ОПК-3.2. Соотносит знания в области программирования, интерпретацию прочитанного, определяет и создает информационные ресурсы глобальных сетей, образовательного контента, средств тестирования систем.</p> <p>ОПК-3.3. Имеет практический опыт применения разработки программного обеспечения.</p>	<p>Анализ современных подходов разработки, разворачивания и доставки веб-приложений. Изучить алгоритмы интервального повторения.</p>	<p>Проанализированы современные подходы разработки, разворачивания и доставки веб-приложений. Изучены алгоритмы интервального повторения.</p>
<p>ОПК-4. Способен участвовать в разработке технической документации программных продуктов и комплексов с использованием стандартов, норм и правил, а также в</p>	<p>Написание письменного отчета по выполненной работе.</p>	<p>Отчет по результатам выполненных задач</p>

<p>управлении проектами создания информационных систем на стадиях жизненного цикла.</p> <p>ОПК-4.1. Использует принципы сбора и анализа информации, создания информационных систем на стадиях жизненного цикла.</p> <p>ОПК-4.2. Осуществляет управление проектами информационных систем.</p> <p>ОПК-4.3. Демонстрирует практический опыт анализа и интерпретации информационных систем.</p>		
<p>ОПК-5. Способен устанавливать и сопровождать программное обеспечение информационных систем и баз данных, в том числе отечественного происхождения, с учетом информационной безопасности.</p> <p>ОПК-5.1. Понимает методику установки и администрирования информационных систем и баз данных. Знаком с содержанием Единого реестра российских программ.</p> <p>ОПК-5.2. Реализует техническое сопровождение информационных систем и баз данных.</p> <p>ОПК-5.3. Использует практические навыки установки и инсталляции программных комплексов, применения основ сетевых технологий.</p>	<p>Настроить окружение разработки, развернуть базу данных PostgreSQL, настроить систему контроля версий и автоматизировать процесс сборки и развертывания приложения с использованием Docker.</p>	<p>Настроено локальное окружение разработки, установлена и сконфигурирована PostgreSQL БД, настроен Git репозиторий. Создан и протестирован процесс автоматического развертывания приложения через Docker.</p>
<p>ОПК-6. Способен понимать принципы работы современных</p>	<p>Реализовать современный пользовательский</p>	<p>Разработан адаптивный пользовательский</p>

<p>информационных технологий и использовать их для решения задач профессиональной деятельности</p> <p>ОПК-6.1. Понимает основные положения, концепции и современные методы обработки и хранения данных.</p> <p>ОПК-6.2. Осуществляет первичный сбор и анализ данных для организации информационных процессов</p> <p>ОПК-6.3. Обладает практическим опытом применения современных информационных технологий для решения задач профессиональной деятельности.</p>	<p>интерфейс для работы с карточками, интеграцию с бэкендом и систему хранения данных пользователя.</p>	<p>интерфейс, реализована интеграция фронтенда с бэкендом через REST API, внедрена система аутентификации и авторизации пользователей.</p>
--	---	--

Дата выдачи задания 01.10.2024.

Срок представления на кафедру отчета о практике 26.12.2024.

Руководитель практики,
Доцент кафедры программных систем,
к.т.н, доцент _____

О.А. Гордеева

(подпись)

Задание принял к исполнению
обучающийся группы № 6401-020302D _____

А.А. Алёнушка

(подпись)

ВВЕДЕНИЕ

Современные технологии обучения и методики запоминания информации играют ключевую роль в образовательном процессе, помогая людям эффективно усваивать новые знания в различных областях — от изучения иностранных языков до освоения профессиональных навыков. Особое место среди этих методик занимает система интервального повторения Лейтнера, которая основана на принципе повторения с оптимальными временными интервалами и показала свою эффективность в многочисленных исследованиях. С развитием цифровых технологий традиционные методики обучения трансформируются в современные образовательные инструменты, делая процесс обучения более доступным, удобным и эффективным.

В современных условиях особую актуальность приобретает создание веб-приложений, переносящих классические образовательные методики в цифровую форму. Система Лейтнера, изначально разработанная для работы с физическими карточками, прекрасно адаптируется к цифровому формату, позволяя автоматизировать процесс управления интервалами повторения и отслеживания прогресса обучения. Такое веб-приложение может найти широкое применение в образовательных учреждениях, языковых школах и среди самостоятельно обучающихся пользователей, предоставляя им эффективный инструмент для запоминания информации.

Во время практики необходимо решить следующие задачи:

- исследовать современные подходы к разработке веб-приложений с использованием Spring Boot и сопутствующих технологий;
- изучить принципы работы системы интервального повторения Лейтнера;
- освоить инструменты разработки, включая Docker, PostgreSQL и MyBatis;
- реализовать удобный пользовательский интерфейс для работы с карточками;
- изучить алгоритмы интервального повторения;

- подготовить и оформить письменный отчет по выполненной работе.

1 Описание средств реализации

В настоящее время существует огромное количество программных продуктов, позволяющих в эффективно и качественно разработать программный комплекс для различных предметных областей. Для правильного и обоснованного выбора во внимание принимались различные критерии для оценки качества программного продукта.

1.1 Описание операционной системы

В качестве операционной системы (ОС) для клиентской части выбрана Windows 10 – операционная система, выпущенная компанией Microsoft в 2015 году [1]. Среди ключевых преимуществ данной ОС можно выделить:

- интуитивно понятный пользовательский интерфейс, упрощающий процесс установки и администрирования программного обеспечения;
- высокий уровень безопасности благодаря регулярным обновлениям и встроенному антивирусу Windows Defender;
- широкая совместимость с различным программным обеспечением, так как большинство разработчиков в первую очередь выпускают свои продукты именно для Windows 10.

1.2 Описание языка программирования

Для реализации клиентской части системы выбран язык программирования JavaScript с использованием библиотеки React.

JavaScript — это интерпретируемый язык программирования, широко применяемый для разработки интерактивных веб-приложений [2]. Он обеспечивает динамическое взаимодействие с пользователем, обновление контента без перезагрузки страницы и реализацию сложной клиентской логики. React – это JavaScript-библиотека, ориентированная на создание пользовательских интерфейсов путем компоновки независимых и повторно используемых UI-элементов [3]. Выбор JavaScript и React обусловлен их распространенностью в веб-разработке, высокой производительностью, обширной экосистемой инструментов и библиотек, а также способностью создавать динамичные и отзывчивые пользовательские интерфейсы.

Для реализации серверной части системы был выбран язык программирования Java. Java – объектно-ориентированный, компилируемый язык программирования, характеризующийся надежностью, платформенной независимостью и широким применением в серверной разработке. В качестве фреймворка для серверной части приложения выбран Spring Framework, предоставляющий обширный набор инструментов и компонентов, упрощающих разработку Java-приложений. Spring Framework способствует созданию масштабируемых, поддерживаемых и тестируемых систем. Обоснованием выбора Java и Spring Framework служат их надежность, производительность, зрелость экосистемы, а также способность создавать отказоустойчивую и эффективную серверную часть (backend) для реализации алгоритмов интервального повторения и обработки данных.

1.3 Описание среды разработки

В качестве интегрированной среды разработки (IDE) была выбрана IntelliJ IDEA. IntelliJ IDEA — это мощная IDE, разработанная JetBrains, предоставляющая широкий спектр инструментов для разработки программного обеспечения на различных языках, включая Java, JavaScript и многие другие. IntelliJ IDEA предлагает следующую:

- умное автодополнение кода (IntelliSense) — IDE анализирует контекст кода и предлагает релевантные варианты автодополнения, включая имена переменных, методов, классов и ключевые слова, что ускоряет процесс разработки и снижает вероятность ошибок;
- встроенный отладчик — позволяет пошагово выполнять код, отслеживать значения переменных и выявлять источники ошибок, что существенно упрощает процесс отладки;
- поддержка рефакторинга — IntelliJ IDEA предоставляет богатый набор инструментов для рефакторинга кода, позволяющих безопасно и эффективно изменять структуру проекта, улучшая его читаемость и поддерживаемость;

– интеграция с системами контроля версий — IDE бесшовно интегрируется с популярными системами контроля версий, такими как Git, облегчая совместную работу над проектом и управление изменениями в коде;

– поддержка широкого спектра технологий и фреймворков — IntelliJ IDEA предоставляет встроенную поддержку для Java, Spring Framework, JavaScript, React и других технологий, используемых в данном проекте, что упрощает настройку окружения и разработку.

1.4 Описание используемых библиотек

Для управления миграциями базы данных в проекте используется Liquibase. Liquibase — это инструмент управления версиями базы данных с открытым исходным кодом, который позволяет отслеживать, управлять и применять изменения схемы базы данных. Использование Liquibase обеспечивает согласованность и воспроизводимость структуры базы данных в различных средах разработки и развертывания.

Взаимодействие с базой данных реализовано с помощью MyBatis. MyBatis — это инструмент для работы с базой данных, отличающийся простотой и эффективностью для Java, который упрощает отображение SQL-запросов на объекты Java и обратно. MyBatis обеспечивает гибкость в написании SQL-запросов и позволяет оптимизировать производительность запросов к базе данных.

Для преобразования сущностей Java в объекты передачи данных (DTO) и обратно используется MapStruct. MapStruct — это генератор кода, который создает эффективный и типобезопасный код для маппинга объектов. Применение MapStruct уменьшает объем шаблонного кода и повышает производительность приложения.

На клиентской стороне для поддержки синтаксиса Markdown в карточках используется Markdown-it. Markdown-it — это быстрый и совместимый парсер Markdown, написанный на JavaScript. Он позволяет отображать форматированный текст Markdown в пользовательском интерфейсе веб-приложения.

В качестве основы веб-приложения используется Spring Boot. Spring Boot — это фреймворк, упрощающий разработку и развертывание приложений на основе Spring. Он предоставляет автоматическую настройку, встроенный веб-сервер и множество других функций, которые ускоряют процесс разработки и повышают производительность приложения.

Для разработки пользовательского интерфейса используется библиотека React. React — это JavaScript-библиотека, предназначенная для создания динамических и интерактивных пользовательских интерфейсов. Она основана на компонентном подходе, что позволяет разбивать интерфейс на независимые и повторно используемые блоки, упрощая разработку и поддержку сложных UI. React эффективно обновляет и перерисовывает только измененные части интерфейса, обеспечивая высокую производительность и отзывчивость приложения.

2 Описание проекта разрабатываемого приложения

Для разработки веб-приложения, основанного на системе Лейтнера с алгоритмом интервального повторения, была выбрана архитектура, которая включает в себя клиент-серверный подход, слоистую структуру и использование контейнеров. Такой выбор обусловлен рядом факторов, которые обеспечивают гибкость, масштабируемость и удобство в развертывании приложения.

Клиент-серверная архитектура — приложение разделено на два компонента: фронтенд, который отвечает за интерфейс пользователя, и бэкенд, обрабатывающий запросы и выполняющий бизнес-логику. Это позволяет изолировать логику и интерфейс, что улучшает безопасность и производительность системы.

Слоистая архитектура на бэкэнде включает три уровня:

- контроллеры — отвечают за обработку входящих HTTP-запросов и передачу их в соответствующие сервисы;
- сервисы — содержат бизнес-логику приложения, осуществляют обработку данных и взаимодействуют с репозиториями;

– репозитории — отвечают за доступ к данным, взаимодействие с базой данных и выполнение CRUD-операций.

Такой подход позволяет легко тестировать и изменять отдельные слои без влияния на другие компоненты системы, а также упрощает поддержание чистоты кода.

Использование контейнеров (Docker) позволяет обеспечить изоляцию всех компонентов приложения, упростить развертывание и поддержку различных сред, а также повысить гибкость и масштабируемость системы.

Эта архитектура является оптимальной для создания приложения, которое будет поддерживать сложную бизнес-логику (алгоритм интервального повторения), легко масштабироваться и интегрироваться с другими сервисами.

Архитектура приложения описана, можно перейти к рассмотрению алгоритмов, которые лежат в основе системы Лейтнера и других методов интервального повторения. Эти алгоритмы играют ключевую роль в оптимизации процесса обучения, позволяя пользователю эффективно запоминать информацию с учетом принципов распределенного повторения.

2.1 The Super Memo

Алгоритмы интервального повторения Super Memo являются одними из самых известных в мире. Эти алгоритмы постоянно совершенствуются и оптимизируются с 1982 года, когда д-р Петр Вожняк из Польши разработал свой первый алгоритм Super Memo [4].

Преимущество Super Memo перед более простыми алгоритмами (системой Лейтнера) в том, что Super Memo включает в себя способность обеспечивать точную оценку сложности карточек, и способность алгоритмов адаптироваться к учащемуся, основываясь на индивидуальности его мозга и памяти.

Последним выпущенным алгоритмом является SM-18, 2019 год. Однако данный алгоритм требует лицензирования, и получить к нему доступ можно только путем покупки лицензии. Поэтому рассмотрим алгоритм SM-2, который находится в открытом доступе.

Последовательность действий алгоритма SM-2, который используется в электронном варианте метода Super Memo и предполагает расчет коэффициентов простоты для отдельных заданий, следующая:

- разделить знания на мельчайшие элементы;
- со всеми предметами связать E-Factor, равный 2.5;
- повторять информацию, используя следующий расчет интервалов:
 - $I(1) := 1$
 - $I(2) := 6$
 - for $n > 2$: $I(n) = I(n - 1) * EF$,

где $I(n)$ – межповторный интервал после n -го повторения (в днях), EF (E-Factor) – коэффициент легкости, отражающий легкость запоминания и удержания в памяти данного элемента.

Если результат представляет собой дробь, то необходимо округлить его до ближайшего целого числа.

– сразу после ответа оценить качество реакции на повторение по шкале от 0 до 5:

- 5 – идеальный ответ;
- 4 – правильный ответ после небольшого колебания;
- 3 – правильный ответ после длительного колебания;
- 2 – неверный ответ, где правильный казался легким;
- 1 – неверный ответ, правильный вспомнился;
- 0 – ответ полностью забыт.

– после каждого повторения необходимо пересчитывать E-Factor недавно повторенного элемента по формуле:

$$EF' = EF + (0.1 - (5 - q) * (0.08 + (5 - q) * 0.02))$$

где EF' – новое значение E-Factor, EF – старое значение E-Factor, q – качество ответа по пятибалльной шкале.

Если EF меньше, чем 1.3, значит присвоить EF значение 1.3.

- если качество реакции на ответ было ниже трех, то повторение задания начинать с самого начала без изменения E-Factor, как если бы приходилось отвечать на вопрос заново;

- после каждого захода на повторение в текущий день повторить еще раз все пункты, получившие менее четырех баллов реакции. И продолжать повторения до тех пор, пока все эти элементы не наберут не менее четырех.

2.2 Алгоритм Anki

Anki – одна из самых популярных в мире систем интервального повторения на основе карточек. Алгоритм Anki, основанный на оригинальном алгоритме SM-2 Петра Вожняка, представляет собой систему с открытым исходным кодом, которую могут бесплатно использовать все учащиеся.

Так как Anki основан на SM-2, то рассмотрим основные отличия двух алгоритмов [5]:

- SM-2 описывает исходный интервал в один день, последующий – в шесть дней. С Anki появляется возможность полного контроля над длительностью начальных шагов обучения. Метод Anki понимает, что может возникнуть необходимость увидеть новую информацию несколько раз, прежде чем она будет запомнена;

- Anki использует четыре варианта ответа на вопрос. Таким образом, существует только один вариант с ошибкой. Благодаря этому в дальнейшем можно отрегулировать легкость вопроса, просто изменив положительные ответы;

- ответы на вопросы позже запланированного времени будут учитываться при расчете следующего интервала, так как Anki понимает понятие «поздно», что как будто говорит о том, что вы опоздали с изучением материала;

- как и в SM-2, отказ от ответа в Anki по умолчанию сбрасывает интервал вопроса. Но в Anki вместо полного сброса пользователь может выбрать уменьшение интервала;

– легкость запоминания увеличивает коэффициент простоты (E-Factor) и также добавляет дополнительный «бонус» к текущему расчету интервала повторения.

2.3 Система Duolingo

Half-life Regression (HLR) – новая модель практики интервального повторения, используемая в приложениях по обучению иностранным языкам. HLR сочетает психолингвистическую теорию с современными методами машинного обучения, косвенно оценивая «период полураспада» слова или понимания в долговременной памяти учащегося [6].

В разработке данной модели использовались методы работы с «большими данными», такими как логистическая регрессия, и с использованием экспоненциальной функции кривой забывания Эббингауза.

Период полураспада слова в памяти рассчитывается с помощью следующей формулы:

$$h = 2^{\Theta \cdot x}$$

где Θ – вес регрессионной модели, X – набор переменных, которые обобщают историю обучения слова.

HLR включает в себя поиск «лучшего» веса модели для Θ путем минимизации «функции потерь» l на каждом занятии для каждого ученика:

$$l((p, \Delta, x); \Theta) = \left(p - 2^{-\frac{\Delta}{2^{\Theta \cdot x}}}\right)^2 + \alpha \left(-\frac{\Delta}{\log_2(p)} - 2^{\Theta \cdot x}\right)^2 + \lambda \|\Theta\|_2^2$$

где:

- p – вероятность правильного припоминания предмета;
- Δ – функция времени задержки с момента последнего выполнения вопроса;
- α – параметр для контроля относительной важности периода полураспада в общей целевой функции тренировки;
- λ – параметр, контролирующий срок регуляризации и помогающий предотвратить переизбыток информации.

Например, на Рисунке 1 показано, как может выглядеть кривая забывания HLR для определенного слова:

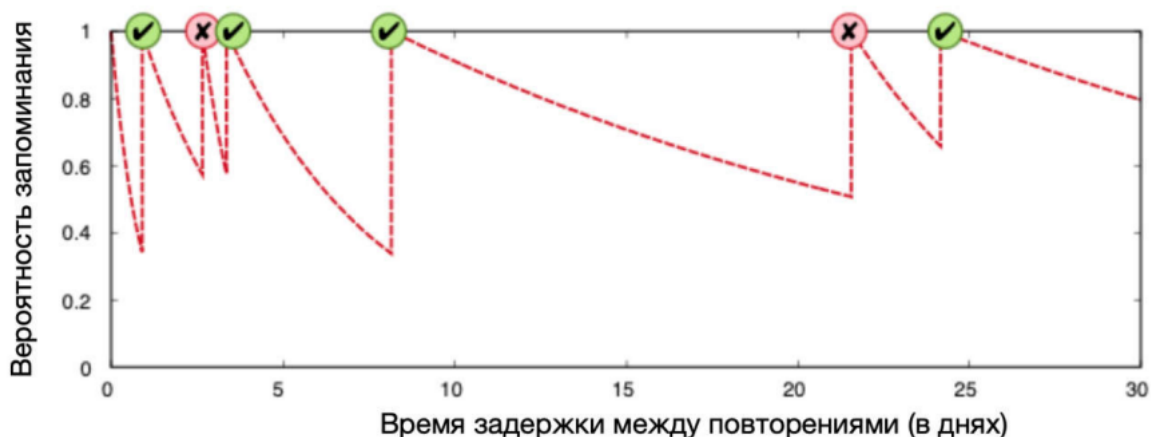


Рисунок 1 – История запоминания слов учащегося и кривая забывания, предсказанная HLR

Каждый раз, когда ученик правильно отвечает (зеленые галочки), h увеличивается, и память начинает ухудшаться медленнее (пунктирная линия). Но каждый раз, когда возникает ошибка (красные крестики), h уменьшается, и необходимо будет повторять материал раньше времени.

2.4 База данных

Для первоначальной версии приложения была спроектирована база данных со следующими сущностями:

- users (id, email, password, created_at)
- cards (id, question, answer, collection_id, created_at)
- collections (id, name, user_id, created_at)
- repetitions (id, card_id, user_id, repetition_date, interval, easiness_factor, grade, next_repetition_date)

Рассмотрим подробнее сущность repetitions. Она предназначена для реализации алгоритма SM2. Рассмотрим ее поля и их назначения:

- id – идентификационный номер для каждой записи по повторению;
- card_id – идентификационный номер карточки, которую повторял пользователь;
- user_id – идентификационный номер пользователя;

- repetition_date – дата, когда происходило повторение;
- interval – рассчитанный в днях интервал, через который данная карточка должна быть повторена;
- easiness_factor – коэффициент для расчета интервала, который отражает насколько хорошо пользователь запоминает эту карточку;
- grade – оценка пользователем своего ответа по 4 бальной шкале;
- next_repetition_date – дата с которой данная карточка должна выводиться в списке для повторения.

На рисунке 2 изображена ER диаграмма базы данных:

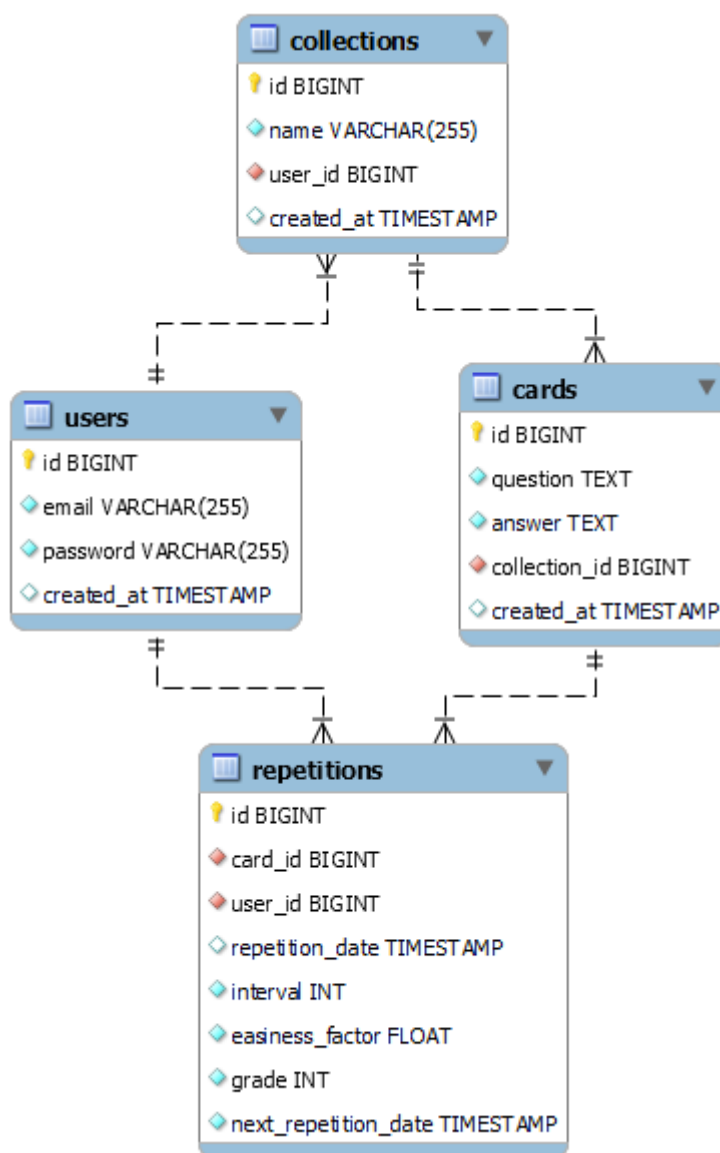


Рисунок 2 – ER диаграмма базы данных

2.5 Диаграмма use case

Для иллюстрации функциональных требований к системе на концептуальном уровне используется диаграмма вариантов использования, позволяющая наглядно отразить ключевые сценарии взаимодействия пользователей и компонентов приложения.

Диаграмма вариантов использования (use case diagram) — это поведенческая диаграмма в языке UML, отображающая отношения между акторами (пользователями или внешними системами) и прецедентами (вариантами использования), что позволяет описать функциональные требования системы на концептуальном уровне [7]. Такая диаграмма показывает, какие сервисы предоставляет система и каким образом они используются внешними сущностями, не вдаваясь в детализацию внутренней реализации [8].

На рисунке 3 представлена диаграмма прецедентов (use case), демонстрирующая виды взаимодействия акторов с Java-приложением и определяющая основные варианты использования.

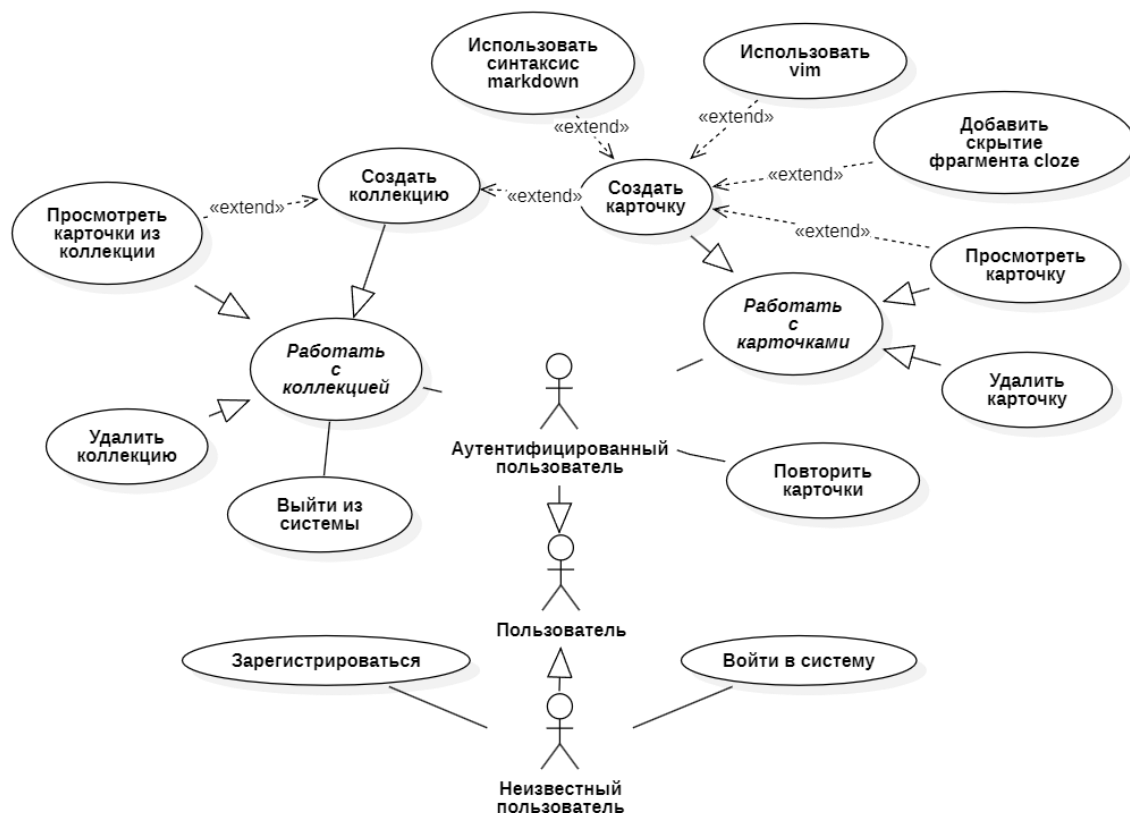


Рисунок 3 – Диаграмма вариантов использования

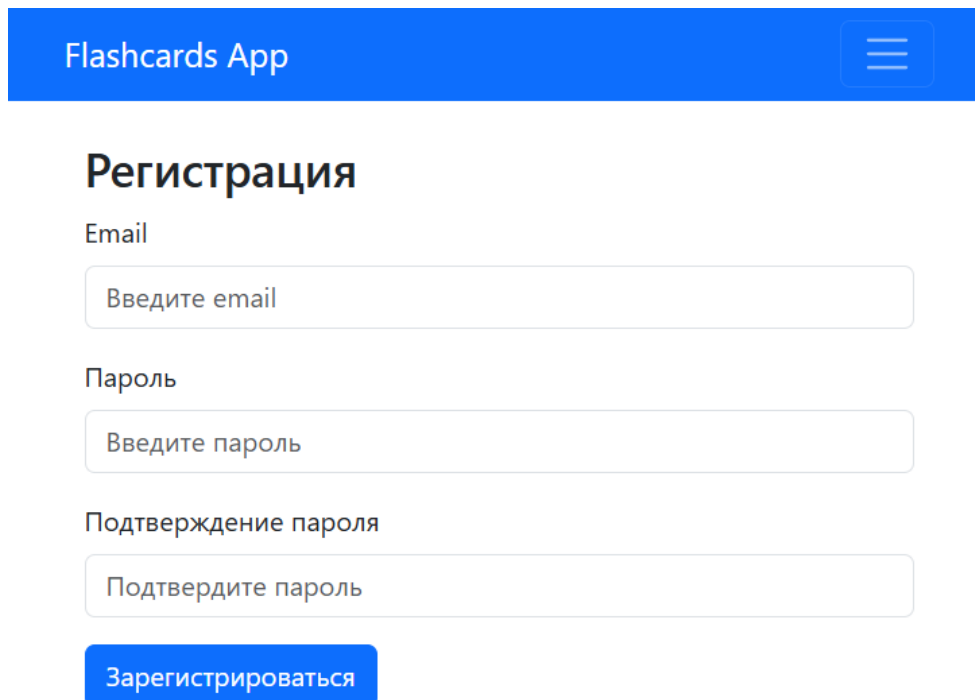
3 Описание экранных форм разработанного программного приложения

3.1 Начало работы: регистрация и вход

Для работы с веб-приложением пользователь должен пройти процедуру регистрации, указав email, пароль и подтверждение пароля. Требования к регистрации:

- email должен соответствовать формату электронной почты (проверка через регулярное выражение: $^{\wedge}\backslash S+@ \backslash S+ \backslash . \backslash S+ \$$);
- пароль должен содержать не менее 6 символов;
- при успешной регистрации отображается сообщение: «Пользователь успешно зарегистрирован!».

На рисунке 4 приведена форма регистрации.



Flashcards App

Регистрация

Email

Пароль

Подтверждение пароля

Зарегистрироваться

Рисунок 4 – Страница регистрации

После успешной регистрации можно выполнить вход в систему указав логин и пароль. На рисунке 5 приведена форма для входа.

Вход

Email

Пароль

Войти

Рисунок 5 – Страница входа

3.2 Работа с коллекциями карточек

После входа пользователь попадает на главную страницу, где отображается список всех его коллекций карточек. Здесь отображаются следующие данные для каждой коллекции:

- название коллекции;
- количество всех карточек;
- количество новых;
- количество карточек в стадии «обучения»;
- количество карточек; готовых к повторению.

На рисунке 6 изображена главная страница.

Список коллекций пользователя


Коллекция	Всего	Новые	Learning	К повтор.	Действия
Java	0	0	0	0	+ 



Рисунок 6 – Главная страница

Также на этой странице реализованы следующие функции:

- создание новой коллекции;
- удаление существующих коллекций;
- переход к добавлению карточек в коллекцию;
- запуск режима повторения для выбранной коллекции.

3.3 Добавление и редактирование карточек

Выбрав коллекцию, пользователь может добавить в неё новые карточки.

Редактор поддерживает:

- ввод текста карточки с использованием Markdown-разметки;
- выделение фрагментов для последующего скрытия («cloze»-режим) —

позволяет создавать карточки с пропусками, которые надо вспомнить при повторении;

- предпросмотр карточки в реальном времени;
- включение Vim-режима для продвинутых пользователей.

На рисунке 7 изображена страница для создания карточек.

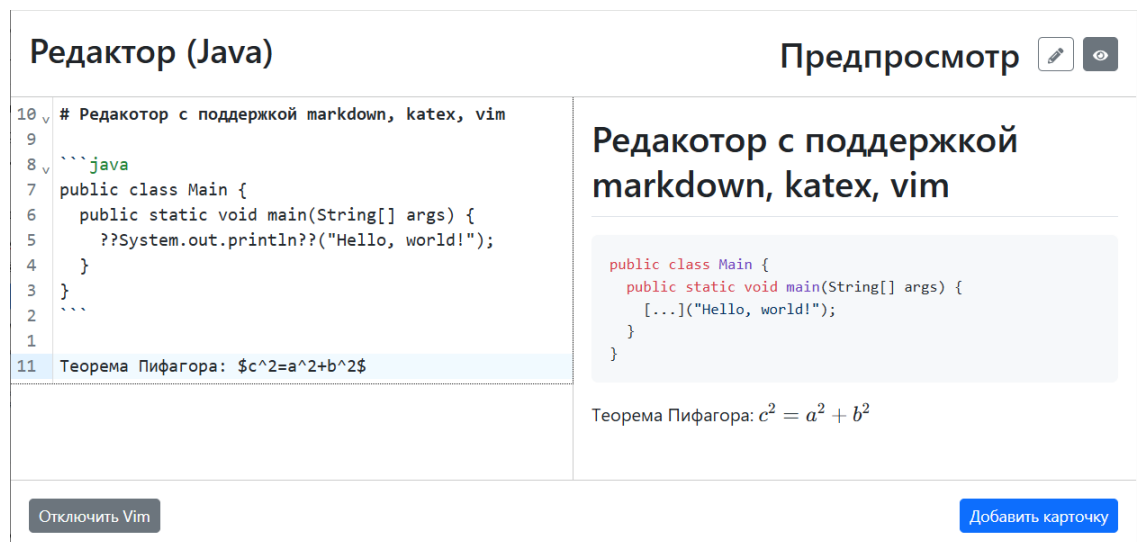


Рисунок 7 – Страница создания карточек

После ввода текста карточки пользователь сохраняет её в коллекции нажатием кнопки «Добавить карточку».

3.4 Редактирование коллекций и поиск карточек

В разделе редактирования коллекций пользователь может:

- просматривать и фильтровать все свои коллекции;

- просматривать список всех карточек в выбранной коллекции;
- выполнять сортировку и поиск по содержимому карточек;
- просматривать содержимое отдельной карточки в Markdown-формате.

На рисунке 8 представлена страница редактирования коллекций.

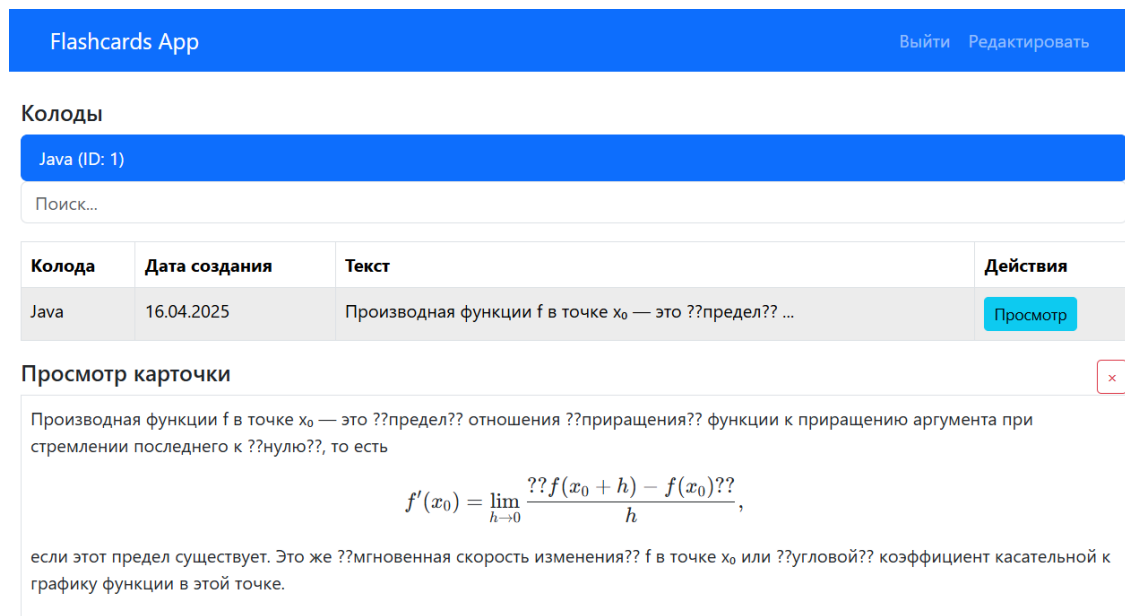


Рисунок 8 – Страница редактирования коллекций

3.5 Повторение карточек

Основная функция системы — проведение интервального повторения по алгоритму Лейтнера. При запуске повторения:

- пользователю последовательно показываются карточки из выбранной коллекции;
- карточки автоматически сортируются по приоритету показа: сначала новые; затем карточки, находящиеся в стадии обучения, и далее — карточки на повторение,
- для cloze-карточек часть информации скрыта и открывается по нажатию Tab.

После просмотра карточки пользователь выбирает оценку ответа (кнопки: «Again»; «Hard», «Good», «Easy»), что влияет на дату следующего показа по алгоритму интервального повторения.

На рисунке 9 изображена страница режима повторения.

Производная функции f в точке x_0 — это [...] отношения [...] функции к приращению аргумента при стремлении последнего к [...], то есть

$$f'(x_0) = \lim_{h \rightarrow 0} \frac{[\dots]}{h},$$

если этот предел существует. Это же [...] f в точке x_0 или [...] коэффициент касательной к графику функции в этой точке.

<1 мин

AGAIN

<6 мин

HARD

<10 мин

GOOD

2 дн

EASY

0 + 1 + 0

Пропустить

Рисунок 9 – Страница режима повторения

ЗАКЛЮЧЕНИЕ

В результате выполнения производственной практики (научно-исследовательской работы):

- исследованы современные подходы к разработке веб-приложений с использованием Spring Boot и сопутствующих технологий;
- изучены принципы работы системы интервального повторения Лейтнера;
- освоены инструменты разработки, включая Docker, PostgreSQL и MyBatis;
- реализован удобный пользовательский интерфейс для работы с карточками;
- изучены алгоритмы интервального повторения;
- подготовлен и оформлен письменный отчет по практике.

Таким образом, в процессе выполнения научно-исследовательской работы были освоены все необходимые индикаторы (ОПК-1.1, ОПК-1.2, ОПК-1.3, ОПК-2.1, ОПК-2.2, ОПК-2.3, ОПК-3.1, ОПК-3.2, ОПК-3.3, ОПК-4.1, ОПК-4.2, ОПК-4.3, ОПК-5.1, ОПК-5.2, ОПК-5.3, ОПК-6.1, ОПК-6.2, ОПК-6.3) компетенций (ОПК-1, ОПК-2, ОПК-3, ОПК-4, ОПК-5, ОПК-6).

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- 1 What Is Windows 10? [Электронный ресурс]. URL: <https://www.lifewire.com/windows-10-2626217> (дата обращения: 12.12.2024).
- 2 JavaScript [Электронный ресурс]. URL: <https://en.wikipedia.org/wiki/Javascript> (дата обращения: 12.12.2024).
- 3 React (software) [Электронный ресурс]. URL: https://en.wikipedia.org/wiki/React_%28software%29 (дата обращения: 12.12.2024).
- 4 Woźniak P. A. Optimization of Learning : магистерская диссертация, Познанский технологический университет, 1990 [Электронный ресурс]. URL: <https://super-memory.com/english/ol.htm> (дата обращения: 12.12.2024).
- 5 What Spaced Repetition Algorithm Does Anki Use? [Электронный ресурс]. Информационный портал «Frequently Asked Questions». URL: <https://faqs.ankiweb.net/what-spaced-repetition-algorithm.html> (дата обращения: 12.12.2024).
- 6 Settles B., Meeder B. A Trainable Spaced Repetition Model for Language Learning [Электронный ресурс]. URL: <https://aclanthology.org/P16-1174.pdf> (дата обращения: 12.12.2024).
- 7 Диаграмма прецедентов [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/Диаграмма_прецедентов (дата обращения: 12.12.2024).
- 8 Использование диаграммы вариантов использования UML при проектировании программного обеспечения [Электронный ресурс]. URL: <https://habr.com/ru/articles/566218/> (дата обращения: 12.12.2024).