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 05.02.2024 по 10.06.2024
(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 № 6301-020302D А.А. Алёнушка

Руководитель практики,
доцент кафедры программных систем О.А. Гордеева

Дата сдачи 10.06.2024
Дата защиты 10.06.2024

Оценка _____

Самара 2024

ОГЛАВЛЕНИЕ

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 6301-020302D

Направлен на практику приказом по университету от 24.01.2024 г. № 62-ПР на кафедру программных систем
(наименование профильной организации или структурного подразделения университета)

Тема: «Разработка …»

Планируемые результаты освоения образовательной программы (компетенции/индикаторы)	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 использовать их в профессиональной деятельности. **ОПК-1.1.** Использует основные положения и концепции в области математических и естественных наук, Базовые теории и истории основного, теории	Провести анализ и обзор существующих методов интервального повторения, применяемых для запоминания учебного материала. Изучить и описать алгоритмы, используемые в других успешных решениях, и адаптировать	Проведен анализ существующих методов интервального повторения и их применения для эффективного запоминания. Описаны алгоритмы, подходящие

коммуникации; знает основную терминологию. **ОПК-1.2.** Осуществляет первичный сбор и анализ материала, интерпретирует различные математические объекты. **ОПК-1.3.** Применяет опыт решения стандартных математических задач в профессиональной деятельности.	их для реализации в приложении. Ознакомиться со стандартом оформления текстовых учебныхдокументов Самарского университета. Оформить отчет по результатам прохождения практики в строгом соответствии со стандартом оформления текстовых учебных документов.	для реализации в веб-приложении, и проведена их адаптация под требования проекта. Подготовлен отчет по практике.
ОПК-2. Способен применять компьютерные/суперкомпьютерные методы, современное программное обеспечение, в том числе отечественного происхождения, для решения задач профессиональной деятельности. **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. **ОПК-2.2.** Анализирует код на типовых языках программирования, может составлять программы. **ОПК-2.3.** Применяет опыт решения задач анализа, интеграции различных типов	Составить список функций, необходимых для реализации веб-приложения флэш-карточек, включая создание карточек, организацию коллекций, интеграцию с Vim-редактором, поддержку Markdown и KaTeX.	Составлен список основных функций приложения, включая создание и редактирование карточек с использованием Markdown/KaTeX и Vim-режима.

программного обеспечения, анализа типов коммуникаций.		
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. **ОПК-3.1.** Понимает методы теории алгоритмов, методы системного и прикладного программирования, основные положения и концепции в области математических, информационных и имитационных моделей. **ОПК-3.2.** Соотносит знания в области программирования, интерпретацию прочитанного, определяет и создает информационные ресурсы глобальных сетей, образовательного контента, средств тестирования систем. **ОПК-3.3.** Имеет практический опыт применения разработки программного обеспечения.	Определить архитектуру приложения, включающую фронтенд и бэкенд. Описать обоснование выбранной архитектуры, исходя из требований к масштабируемости и удобству взаимодействия с пользователем. Описать интеграцию компонентов приложения для обеспечения целостности данных и эффективности работы.	Определена архитектура приложения с разделением на фронтенд и бэкенд. Обоснован выбор архитектуры, учитывая требования к масштабируемости и удобству использования. Подготовлено описание взаимодействия компонентов, чтобы обеспечить целостность данных и эффективность их обработки.
ОПК-4. Способен участвовать в разработке технической документации программных продуктов и комплексов с	Оформить описание архитектуры приложения на уровне компонент и взаимодействий между	Подготовлено описание архитектуры приложения,

использованием стандартов, норм и правил, а также в управлении проектами создания информационных систем на стадиях жизненного цикла. **ОПК-4.1.** Использует принципы сбора и анализа информации, создания информационных систем на стадиях жизненного цикла. **ОПК-4.2.** Осуществляет управление проектами информационных систем. **ОПК-4.3.** Демонстрирует практический опыт анализа и интерпретации информационных систем.	ними. Создать техническое описание модели данных на концептуальном уровне, включая основные сущности.	включающее взаимодействие между фронтенд и бэкенд компонентами. Описана модель данных на концептуальном уровне, включая основные сущности
ОПК-5. Способен инсталлировать и сопровождать программное обеспечение информационных систем и баз данных, в том числе отечественного происхождения, с учетом информационной безопасности. **ОПК-5.1.** Понимает методику установки и администрирования информационных систем и баз данных. Знаком с содержанием Единого реестра российских программ. **ОПК-5.2.** Реализует техническое сопровождение информационных систем и баз данных. **ОПК-5.3.** Использует практические навыки установки и инсталляции программных комплексов, применения основ сетевых технологий.	Описать модель данных приложения на концептуальном уровне, включая сущности, их атрибуты и взаимосвязи между ними. Разработать ER-диаграмму для визуализации структуры данных, которая будет использоваться для хранения информации о пользователях, карточках и прогрессе. Обосновать выбор реляционной СУБД и основных методов обеспечения целостности данных.	Разработана и описана концептуальная модель данных приложения, включая основные сущности и их взаимосвязи. Создана ER-диаграмма для визуализации структуры данных, которые будут храниться в базе данных. Обоснован выбор реляционной СУБД для обеспечения целостности данных и эффективности

		работы системы.
ОПК-6. Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности **ОПК-6.1.** Понимает основные положения, концепции и современные методы обработки и хранения данных. **ОПК-6.2.** Осуществляет первичный сбор и анализ данных для организации информационных процессов **ОПК-6.3.** Обладает практическим опытом применения современных информационных технологий для решения задач профессиональной деятельности.	Изучить и описать основные концепции хранения данных в приложении по флэш-карточкам, включая использование СУБД для хранения коллекций карточек, пользователей и истории повторений. Определить способы интеграции модели данных с фронтендом и серверной частью для поддержки функции интервального повторения.	Изучены основные подходы к хранению данных, включая использование реляционной СУБД для организации хранения карточек, коллекций и информации о пользователях. Подготовлено описание интеграции модели данных с фронтендом и серверной частью, чтобы поддерживать алгоритм интервального повторения.

Дата выдачи задания 05.02.2024.

Срок представления на кафедру отчета о практике 10.06.2024.

Руководитель практики,
Доцент кафедры программных систем _____ О.А. Гордеева
(подпись)

Задание принял к исполнению
обучающийся группы № 6301-020302D _____ А.А. Алёнушка
(подпись)

ВВЕДЕНИЕ

Эффективное запоминание и изучение большого объема информации является одной из ключевых задач для студентов, специалистов технических направлений и всех, кто стремится к непрерывному самообразованию. В последние годы флэш-карточки стали популярным инструментом для этой цели, поскольку позволяют быстро организовать информацию и использовать интервальное повторение — научно обоснованный метод, обеспечивающий сохранение знаний в долговременной памяти [1]. Однако традиционные приложения для флэш-карточек часто не предоставляют удобных инструментов для форматирования содержания карточек или интеграции с существующими рабочими процессами, особенно для тех, кто привык работать с текстом с помощью инструментов, подобных Vim.

Целью данной работы является разработка веб-приложения для создания и изучения флэш-карточек, которое сочетает удобство использования с мощными возможностями форматирования и редактирования. Предлагаемое приложение будет включать следующие ключевые функции: поддержку синтаксиса Markdown для форматирования текста, интеграцию с KaTeX для отображения математических выражений, а также использование режима Vim для редактирования карточек, что значительно улучшит удобство работы для пользователей, знакомых с этим текстовым редактором. Кроме того, приложение будет использовать алгоритмы интервального повторения, чтобы оптимизировать процесс запоминания.

Основные этапы достижения цели включают изучение методов и алгоритмов интервального повторения, проектирование и разработку архитектуры приложения, создание концептуальной модели данных, а также определение и описание функциональных возможностей системы. В процессе выполнения работы будут применены современные технологии, такие как React для фронтенда и Java/Spring для серверной части, а также подходы, обеспечивающие удобное и надежное взаимодействие между компонентами системы.

Разработка данного приложения направлена на создание инструмента, который позволит пользователям эффективно организовать учебный процесс, интегрируя мощные возможности для создания, редактирования и повторения учебных материалов, делая процесс обучения более структурированным и приятным.

Во время практики необходимо решить следующие задачи:

- выделить основную цель работы;
- рассмотреть основные методы решения;
- выделить основные функции и роли программы;
- выделить основные части программы;
- оформить логическую модель программы;
- подготовить и оформить письменный отчет по практике.

1. Формулировка цели работы и основные этапы достижения цели

1.1 Формулировка цели работы

Целью работы является разработка веб-приложения для флэш-карточек, поддерживающего интервальное повторение, которое позволяет пользователям эффективно запоминать изучаемый материал. Приложение также будет включать возможности удобного редактирования карточек с использованием Markdown и KaTeX для форматирования текстов и математических выражений, а также Vim-режим для опытных пользователей, привыкших к работе в этом редакторе.

Основная цель приложения заключается в создании инструмента для улучшения процесса обучения путем применения научно обоснованных методов интервального повторения, которые способствуют лучшему удержанию информации в долговременной памяти. Это особенно важно для студентов и специалистов, которые стремятся усвоить большое количество информации за ограниченное время.

1.2 Основные этапы достижения цели

1.2.1 Изучение методов и алгоритмов интервального повторения

Интервальное повторение — это методика, которая помогает улучшить запоминание путем повторения информации через стратегически важные интервалы времени. Основой данной методики является кривая забывания Эббингауза, которая демонстрирует, как быстро информация забывается с течением времени. Применяя интервальное повторение, система напоминает пользователю о необходимости повторения материала до того, как он будет забыт. Существует несколько популярных моделей, которые используются для реализации интервального повторения.

Система Лейтнера

Одной из самых простых и широко используемых моделей является система Лейтнера. Эта методика предлагает деление карточек на несколько уровней или

ящиков, каждый из которых соответствует определенному интервалу повторения. Все карточки начинаются с первого ящика (например, 1 день), и если пользователь правильно отвечает на вопрос, карточка перемещается в следующий ящик с увеличенным интервалом (например, 2 дня, 4 дня и так далее). Если пользователь делает ошибку, карточка возвращается в первый ящик. Преимущество системы Лейтнера заключается в ее простоте, однако она не всегда адаптируется под индивидуальные потребности пользователя [2, с. 36].

Алгоритм SuperMemo (SM-2)

Более продвинутой моделью является алгоритм SuperMemo. Он основан на расчете коэффициента легкости (E-Factor), который определяет, насколько сложно пользователю запомнить тот или иной элемент. Каждый элемент начинается с начального значения E-Factor, равного 2.5. Формула для расчета интервалов выглядит следующим образом:

$$I(1) = 1, \quad I(2) = 6, \quad I(n > 2) = I(n-1) \times E_F$$

где $I(n)$ — интервал повторения после n-го повторения (в днях), а E-Factor регулирует скорость увеличения интервалов [2, с. 37].

После каждого ответа качество реакции оценивается по шкале от 0 до 5:

- 5 - идеальный ответ;
- 4 - правильный ответ с небольшой задержкой;
- 3 - правильный ответ после значительной задержки;
- 2 - неверный ответ, но правильный был легко восстановлен;
- 1 - неверный ответ, правильный ответ вспомнился с трудом;
- 0 - ответ полностью забыт.

После каждого повторения новый E-Factor рассчитывается по формуле:

$$E_F{}' = E_F + (0.1 - (5-q) \times (0.08 + (5-q) \times 0.02))$$

где q — это качество ответа. Если результат меньше 1.3, значение E-Factor фиксируется на 1.3 [2, с. 38].

Anki

Алгоритм Anki также основан на модели SM-2, но с рядом улучшений. В Anki добавлены функции для более гибкого управления интервалами, в том

числе возможность регулирования длины начальных этапов обучения. Также Anki предоставляет пользователю четыре варианта для оценки своего ответа, где только одна оценка для ошибочного ответа, что позволяет более точно оценивать легкость карточки [2, с. 38].

HLR (High Learning Rate)

HLR (Half-Life Regression) — это модель интервального повторения, которая применяется для обучения и оценки длительности сохранения информации в памяти. Основной характеристикой HLR является расчет периода полураспада информации, который определяет, как долго знание останется в памяти учащегося до того, как начнется его забывание. Для определения периода полураспада слова в памяти используется следующая формула:

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

где:

- h — это период полураспада (в днях),
- Θ — весовые коэффициенты регрессионной модели,
- x — набор переменных, которые обобщают историю обучения слова [2, с. 40].

Функция потерь для оптимизации модели HLR вычисляется по следующей формуле:

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

где:

- p - вероятность правильного припоминания предмета;
- Δ - функция времени задержки с момента последнего выполнения задания;
- α - параметр, регулирующий относительную важность периода полураспада в общей функции потерь;
- λ - параметр регуляризации для предотвращения переизбытка информации.

Эта модель сочетает методы машинного обучения с психолингвистическими подходами для оценки того, как долго ученик запомнит конкретный

материал, и позволяет прогнозировать моменты, когда следует повторить материал для поддержания знаний в долговременной памяти [2, с. 41].

Таким образом, для нашего приложения была выбрана модель SuperMemo (SM-2) за ее адаптивность и точность в управлении процессом повторения.

1.2.2 Описание списка функций разрабатываемого приложения

Следующим этапом является определение функциональных возможностей приложения. Основные функции включают:

- создание и редактирование карточек с использованием Markdown и KaTeX, что позволит пользователям форматировать текст и вводить математические выражения;
- поддержка режима Vim для удобного редактирования карточек;
- организация карточек в коллекции и реализация поиска по ним;
- реализация интервального повторения на основе алгоритма SuperMemo для эффективного управления процессом обучения.

1.2.3 Разработка архитектуры приложения

На основе анализа требований и доступных технологий была определена архитектура приложения, включающая следующие компоненты:

- фронтенд (React) — для реализации интерактивного интерфейса пользователя;
- бэкенд (Java/Spring) — для обработки данных, реализации логики приложения и взаимодействия с базой данных;
- база данных (PostgreSQL) — для хранения информации о пользователях, карточках, коллекциях и истории повторений.

Архитектура разделена на независимые компоненты, что позволяет улучшить масштабируемость и гибкость приложения, а также обеспечивает возможность дальнейшего развития и обновления отдельных частей системы.

1.2.4 Описание модели данных на концептуальном уровне

Была разработана концептуальная модель данных, включающая основные сущности:

- пользователь: информация о пользователях, включая учетные данные и прогресс обучения;
- карточка: учебные карточки с текстом, поддержкой Markdown и KaTeX;
- коллекция карточек: логическое объединение карточек для организации учебного материала;
- история повторений: информация о времени последнего повторения и успешности изучения, необходимая для реализации алгоритма интервального повторения.

2. Описание списка функций разрабатываемого программного приложения

Веб-приложение для флэш-карточек предназначено для поддержки эффективного процесса обучения и включает ряд функций, которые позволяют пользователям удобно создавать, организовывать и изучать учебные материалы. Основные функции приложения направлены на удовлетворение потребностей студентов, специалистов технических направлений и всех, кто активно занимается самообразованием.

2.1 Создание и редактирование флэш-карточек

Одной из ключевых функций приложения является создание и редактирование флэш-карточек с использованием современных инструментов форматирования. Данная функция обеспечивает:

- поддержка Markdown. Markdown является простым и удобным языком разметки, который позволяет пользователям легко форматировать текст. Пользователи могут добавлять заголовки, списки, ссылки и выделение текста для создания структурированных и понятных карточек;
- поддержка KaTeX. Для студентов технических и математических направлений крайне важно иметь возможность добавлять сложные математические выражения. KaTeX предоставляет возможность использовать формат LaTeX для ввода формул и математических символов. Это позволяет создавать карточки, подходящие для подготовки по математике, физике, программированию и другим техническим дисциплинам;
- редактирование в стиле Vim. Приложение предлагает режим редактирования, схожий с текстовым редактором Vim, что особенно удобно для пользователей, привыкших к этому инструменту. Включение Vim-режима позволяет быстро редактировать текст, используя знакомые команды, что существенно ускоряет процесс создания и изменения карточек.

2.2 Организация карточек

Для повышения эффективности учебного процесса важно, чтобы пользователи могли легко организовывать и структурировать свои материалы. Для этого приложение предоставляет следующие возможности:

– создание коллекций карточек. Карточки могут быть сгруппированы в коллекции для упрощения их организации. Например, можно создать коллекцию по теме “JavaScript”, которая будет содержать все карточки, связанные с этим языком программирования;
– поиск и фильтрация карточек. Реализована возможность быстрого поиска по карточкам и коллекциям. Пользователи могут искать карточки по ключевым словам или фильтровать их по коллекциям, чтобы быстрее находить нужную информацию. Это особенно полезно при большом объеме учебных материалов.

2.3 Интервальное повторение

Одной из главных целей приложения является помощь пользователям в эффективном запоминании материала. Для этого используется алгоритм интервального повторения SuperMemo.

Алгоритм SuperMemo (SM-2) адаптируется под индивидуальные особенности пользователя. Приложение отслеживает успешность повторений каждой карточки и на основании этого планирует, когда следующая проверка материала должна быть проведена. Это позволяет оптимально распределить повторения, чтобы информация удерживалась в долговременной памяти.

2.4 Режим обучения и повторения

Приложение предлагает несколько режимов обучения, чтобы пользователи могли адаптировать процесс под свои потребности.

Режим свободного повторения.

Пользователь может вручную выбирать карточки для повторения, изучая материал в удобном для него порядке. Это полезно, если требуется быстро освежить знания по конкретной теме.

Режим интервального повторения.

В этом режиме приложение автоматически определяет, какие карточки необходимо повторить, основываясь на алгоритме SuperMemo. Карточки, которые пользователю даются сложнее, будут повторяться чаще, а те, которые уже выучены, — реже.

Отслеживание прогресса.

Пользователи смогут видеть, сколько карточек выучено, сколько нуждается в повторении, и отслеживать свой прогресс по каждой коллекции. Это способствует лучшему пониманию текущего уровня знаний и помогает планировать дальнейшее обучение.

2.5 Управление пользователями и учетными записями

Для организации персонализированного процесса обучения приложение включает в себя:

- регистрация и авторизация пользователей. Возможность создавать учетные записи, входить в систему и управлять своими данными. Это позволяет каждому пользователю иметь собственный учебный прогресс, коллекции карточек и историю повторений;
- защита данных и восстановление доступа. Приложение поддерживает функции восстановления доступа в случае потери пароля и защиту данных пользователей с использованием современных технологий, таких как JWT для аутентификации.

Таким образом, приложение предлагает функционал, который позволяет не только удобно создавать и редактировать учебные карточки, но и эффективно организовывать и планировать процесс обучения, делая его более целенаправленным и результативным.

3. Описание архитектуры разрабатываемого приложения и ее обоснование

Архитектура разрабатываемого веб-приложения для флэш-карточек основана на принципах масштабируемости, модульности и надежности, что обеспечивает удобство как для пользователя, так и для разработчиков. Приложение построено по клиент-серверной архитектуре и включает фронтенд, бэкенд, и базу данных, которые взаимодействуют между собой с помощью REST API.

3.1 Компонентная структура приложения

Приложение разделено на три основных компонента: фронтенд, бэкенд, и база данных.

3.1.1 Фронтенд (Client-side)

Фронтенд отвечает за взаимодействие с пользователем и обеспечивает удобный интерфейс для работы с приложением. Для реализации фронтенд части используется React, современный JavaScript-фреймворк, который позволяет создавать динамичные и интерактивные пользовательские интерфейсы. Перечислим основные компоненты фронтенда.

Редактор карточек.

Поддержка Markdown и KaTeX для создания и редактирования карточек с текстом и математическими выражениями. Редактор также поддерживает Vim-режим для удобства работы с текстом.

Интерфейс для управления коллекциями.

Функционал для создания, редактирования и удаления коллекций карточек. Также реализован поиск и фильтрация карточек для удобной навигации по учебному материалу.

Интервальное повторение.

Интерфейс для запуска и управления процессом интервального повторения, который взаимодействует с серверной частью для получения данных о карточках, требующих повторения.

React обеспечивает высокую производительность и гибкость, что позволяет легко расширять и модифицировать интерфейс без потери производительности.

3.1.2 Бэкенд (Server-side)

Бэкенд отвечает за бизнес-логику и взаимодействие с базой данных. Он реализован с использованием фреймворка Spring (Java), который обеспечивает надежность и масштабируемость серверной части. Основные задачи бэкенда:

- REST API. Обеспечивает взаимодействие между фронтендом и сервером. API используется для управления пользователями, карточками, коллекциями, а также для реализации алгоритма интервального повторения;
- алгоритм интервального повторения. На стороне бэкенда реализован алгоритм SuperMemo (SM-2), который рассчитывает интервалы повторений для каждой карточки в зависимости от прогресса пользователя;
- аутентификация и авторизация. Бэкенд управляет процессом аутентификации и авторизации пользователей, используя JWT (JSON Web Token) для безопасного управления сессиями;
- интеграция с базой данных. Бэкенд взаимодействует с базой данных через ORM (Object Relational Mapping) с использованием Hibernate, что упрощает работу с базой данных и обеспечивает целостность данных.

3.1.3 База данных (Database)

Для хранения данных используется PostgreSQL, мощная реляционная база данных, которая обеспечивает высокую надежность и поддержку транзакций. Основные сущности, хранимые в базе данных:

- пользователи. Учетные записи пользователей с информацией о логине, пароле и роли (например, студент или преподаватель);
- карточки. Учебные карточки, которые создаются пользователями. Карточки могут содержать текст, математические выражения и метаданные (например, дата создания, дата последнего повторения);

– коллекции карточек. Логическое объединение карточек по темам или учебным материалам, что позволяет пользователю организовать свои учебные материалы;
– история повторений. Данные о времени последнего повторения каждой карточки и оценка знания пользователем (необходимы для расчета алгоритма интервального повторения).

PostgreSQL была выбрана из-за ее соответствия требованиям безопасности, производительности и гибкости в управлении данными.

3.2 Взаимодействие компонентов

Компоненты приложения взаимодействуют друг с другом через REST API, что обеспечивает четкое разделение ответственности и модульность. Вот как осуществляется взаимодействие:

1 пользователь через фронтенд создает, редактирует или удаляет карточки и коллекции. Эти действия отправляются в виде HTTP-запросов на бэкенд через API;
2 бэкенд получает запрос, обрабатывает его, взаимодействуя с базой данных, и возвращает данные или подтверждение успешной операции обратно во фронтенд;
3 алгоритм интервального повторения работает на сервере, периодически вычисляя, какие карточки пользователю нужно повторить. Эти данные передаются на фронтенд для отображения пользователю;
4 база данных хранит все данные приложения и обеспечивает их целостность и доступность для бэкенда через ORM-интерфейс Hibernate.

3.3 Обоснование выбранной архитектуры

Выбор данной архитектуры был основан на следующих критериях:

– масштабируемость. Разделение на фронтенд и бэкенд позволяет масштабировать каждую часть системы независимо друг от друга. Например, можно

легко увеличить количество серверов для обработки запросов на стороне бэкенда, если нагрузка возрастет;

- надежность и поддержка транзакций. PostgreSQL обеспечивает поддержку транзакций и защиту данных, что критично для приложения, в котором важно сохранять историю повторений и прогресс обучения пользователей;
- гибкость и модульность. Использование React на фронтенде и Spring на бэкенде обеспечивает возможность легко расширять и модифицировать отдельные компоненты приложения. Например, можно добавлять новые функции или изменять логику работы алгоритма интервального повторения без необходимости вносить изменения в другие части системы;
- безопасность. Использование JWT для авторизации обеспечивает надежную защиту пользовательских данных, в том числе предотвращение несанкционированного доступа;
- удобство разработки. Интеграция с ORM Hibernate упрощает взаимодействие с базой данных и обеспечивает автоматическую миграцию данных, что ускоряет разработку и улучшает контроль над данными.

Таким образом, данная архитектура обеспечивает устойчивую, масштабируемую и легко расширяемую систему для работы с флэш-карточками, реализующую все необходимые функции для удобного и эффективного учебного процесса.

4. Описание модели данных приложения на концептуальном уровне

Концептуальная модель данных веб-приложения для флэш-карточек организована вокруг ключевых сущностей, которые описывают информацию о пользователях, карточках, коллекциях и процессе интервального повторения. Модель данных обеспечивает поддержку функционала приложения, включая создание и редактирование карточек, хранение информации о прогрессе пользователя и организацию коллекций учебных материалов.

4.1 Основные сущности

4.1.1 Пользователь

Сущность «Пользователь» представляет зарегистрированного пользователя приложения. Каждый пользователь имеет уникальные учетные данные, которые позволяют ему авторизоваться в системе и управлять своими карточками и коллекциями. Основные атрибуты сущности:

- `id` - уникальный идентификатор пользователя;
- `username` - имя пользователя, которое используется для входа в систему;
- `email` - электронная почта пользователя, которая также может использоваться для восстановления пароля;
- `password` - зашифрованный пароль пользователя для аутентификации;
- `role` - роль пользователя (например, «студент» или «администратор»), которая определяет уровень доступа к функциям приложения.

4.1.2 Карточка

Сущность «Карточка» представляет собой учебную единицу, созданную пользователем. Каждая карточка может содержать текст с поддержкой Markdown и математические выражения, отформатированные с помощью KaTeX. Основные атрибуты сущности:

- `id` - уникальный идентификатор карточки;

– `question` - текст вопроса или темы карточки, который отображается пользователю;
– `markdown_content` - текстовое содержание карточки с поддержкой Markdown для форматирования;
– `katex_content` - математические выражения, представленные с использованием KaTeX;
– `created_at` - дата и время создания карточки;
– `updated_at` - дата и время последнего изменения карточки.

4.1.3 Коллекция карточек

Сущность «Коллекция карточек» представляет собой логическое объединение карточек, созданных пользователем для структурирования и организации учебного материала. Пользователь может создавать несколько коллекций для разных предметов или тем. Основные атрибуты сущности:
– `id` - уникальный идентификатор коллекции;
– `name` - название коллекции;
– `description` - краткое описание коллекции для пояснения ее содержания;
– `created_at` - дата создания коллекции;
– `updated_at` - дата последнего изменения коллекции.

4.1.4 История повторений

Сущность «История повторений» хранит информацию о времени последнего повторения каждой карточки и успешность вспоминания пользователем ответа на вопрос. Эти данные используются для расчета алгоритма интервального повторения и определения, когда карточку нужно повторить. Основные атрибуты сущности:
– `id` - уникальный идентификатор записи истории повторений;

– `card_id` - идентификатор карточки, для которой хранится история повторений;
– `user_id` - идентификатор пользователя, которому принадлежит карточка и данные о повторении;
– `last_reviewed_at` - дата и время последнего повторения карточки;
– `success_rate` - оценка успешности повторения (например, 0 - неудачное повторение, 1 - успешное), которая используется для расчета интервалов повторения в будущем.

4.2 Взаимосвязи между сущностями

Модель данных организована таким образом, что каждая сущность взаимосвязана с другими, обеспечивая целостность и эффективность работы приложения. Вот как основные сущности взаимодействуют друг с другом:

– пользователь может создавать несколько коллекций карточек. В каждой коллекции может содержаться множество карточек;
– карточка принадлежит одной коллекции, но может быть связана с несколькими записями в истории повторений, которые хранят данные о прогрессе разных пользователей;
– история повторений связана с конкретной карточкой и определяет для каждого пользователя, когда и как успешно карточка была повторена.

4.3 Описание данных для интервального повторения

Для реализации алгоритма интервального повторения необходимо хранить данные о времени последнего повторения и успехе предыдущих повторений. Эти данные хранятся в сущности История повторений и включают следующие параметры:

– дата последнего повторения используется для расчета интервала до следующего повторения;

– оценка успешности повторения хранится в виде коэффициента успешности (от 0 до 1), который определяет, насколько успешно пользователь ответил на вопрос. Эта оценка влияет на скорость увеличения интервалов повторений.

На основании этих данных серверная часть приложения, используя алгоритм SuperMemo (SM-2), рассчитывает, когда карточку нужно повторить, и отправляет пользователю на фронтенд.

На рисунке 1 представлена схемы базы данных со всеми перечисленными сущностями и связями.

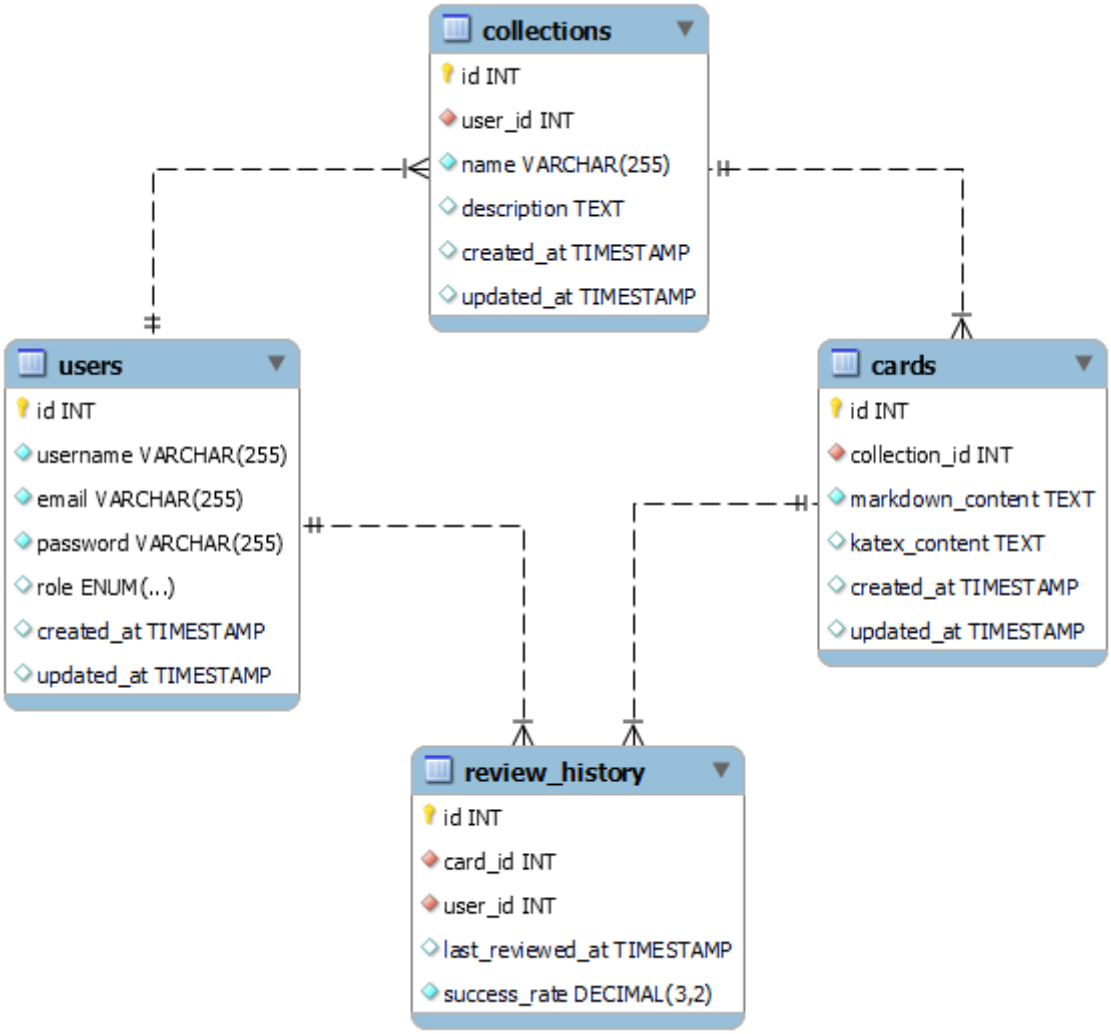

Рисунок 1 – схема концептуальной модели данных

ЗАКЛЮЧЕНИЕ

В ходе выполнения производственной практики (научно-исследовательской работы) были решены следующие задачи:

- выделена основная цель работы;
- рассмотрены основные методы решения;
- выделены основные функции и роли программы;
- выделены основные части программы;
- оформлена логическая модель программы;
- 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 Интервальные повторения [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/Интервальные_повторения

2 Мульдт Валерия Дмитриевна, Максимова Татьяна Геннадьевна МОДЕЛИ И МЕТОДЫ УПРАВЛЕНИЯ ПРОЦЕССАМИ ПРОФЕССИОНАЛЬНОЙ ПОДГОТОВКИ ПЕРСОНАЛА НА ОСНОВЕ ИНТЕРВАЛЬНОГО ОБУЧЕНИЯ // ЭПИ. 2022. №1. URL: https://cyberleninka.ru/article/n/modeli-i-metody-upravleniya-protsessami-professionalnoy-podgotovki-personala-na-osnove-intervalnogo-obucheniya

3 Методология проектирования программных средств [Электронный ресурс]. URL: https://analytics.infozone.pro/methodology-design-software/ (дата обращения: 02.06.2024).

4 Буч Г. Язык UML. Руководство пользователя: Пер. с англ. / Г. Буч,Д. Рамбо, Б. Джекобсон. М.: ДМК-Пресс, 2001. 432 с.

5 IEEE Xplore Digital Library [Электронный ресурс]. URL: https://ieeexplore.ieee.org/ (дата обращения: 02.06.2024).

6 ACM Digital Library [Электронный ресурс]. URL: https://dl.acm.org/ (дата обращения: 02.06.2024).