МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

ОТЧЕТ

по учебной вычислительной (ознакомительной) практике

Ткаченко Радомира Владимировича

студента 1 курса, 4 группы

специальность «Информатика»

Руководитель практики:

старший преподаватель

О.М. Кондратьева

Минск, 2025

# ОГЛАВЛЕНИЕ

[Введение 3](#_heading=h.djmh2oyiztuj)

[Глава 1 Теоретические основы 4](#_heading=h.v9f1vcrvy7bb)

[1.1 Общая характеристика курса «Контроль версий с помощью Git» 4](#_heading=h.szahf6dpli6r)

[1.1.1 Обзор возможностей 4](#_heading=h.7c2xjm4kl63j)

[1.1.2 Модули курса 4](#_heading=h.cgb94tirmv41)

[1.2 Общая характеристика курса «Python» 4](#_heading=h.qzz4cp55csmi)

[1.2.1 Модули курса 4](#_heading=h.lv49pgi7l6eq)

[1.3 Выводы к главе 1 5](#_heading=h.oumdkvyv5xtb)

[Глава 2 Индивидуальный проект «Название» 6](#_heading=h.oortp3t4m67m)

[2.1 Этап предпроизводства разработки проекта «Название» 6](#_heading=h.qim84dfyu6zf)

[2.2 Этап производства разработки проекта «Название» 6](#_heading=h.ma8l7ok6jk7k)

[2.3 Интерфейс приложения 6](#_heading=h.49fh8wr2al6o)

[2.4 Архитектура проекта 6](#_heading=h.27ywkfh5zeyz)

[2.5 Реализация бизнес-логики приложения 7](#_heading=h.8p30hlz0e1x2)

[2.6 Выводы к главе 2 8](#_heading=h.3ldiw5ckrczi)

[Заключение 9](#_heading=h.646wc6f4r2cz)

[Список использованных источников 10](#_heading=h.51cfi7uoacv7)

# Введение

Целью учебной вычислительной (ознакомительной) практики является закрепление полученных знаний за первый год обучения на факультете прикладной математики и информатики посредством выполнения специальных учебных заданий, участия в работе над командным проектом по разработке приложения для интервального повторения «Emphizor», изучение языка программирования Python, отработка и закрепление навыков работы с системой контроля версий — Git.

Избранным направлением работы стало приложение для интервального повторения с поддержкой искусственного интеллекта. Актуальность данного приложения подтверждается многочисленными исследованиями в области психологии, например исследования университетов Северной Каролины и Оксфорда [1; 2], результаты которых подтверждают превосходство такого метода повторения над традиционными. Кроме того, в отличие от существующих решений, «Emphizor» является более доступным и гибким средством для интервального повторения за счет некоммерческой направленности продукта.

Emphizor представляет собой программу для интервального повторения. Графический интерфейс создан при помощи библиотеки PySide6[3]. Для реализации повторения в проекте использован алгоритм FSRS[4], реализованный в библиотеке Py-FSRS[5]. Для сохранения данных пользователя в облако были избраны следующие технологии: Supabase[6], PostgreSQL[7]. Ключевой особенностью рассматриваемого приложения является доступность использования искусственного интеллекта, выбранного самим пользователем.

На период практики были поставлены следующие задачи: Изучение инструментов для совместной разработки проектов и контроля версий(Git), подготовка и настройка программных сред и средств тестирования для прохождения тренинга по выбранному направлению, знакомство с IT компанией «LeverX», возможностями обучения и повышения квалификации, знакомство с разными направлениями профессиональной подготовки IT специалистов, выполнение стандартных заданий, выполнение заданий повышенного уровня, разработка приложения Emphizor, оформление отчёта, получение опыта работы в команде, развитие «soft-skills».

Во время УП 10.07.2025 была проведена экскурсия в IT компанию «LeverX»[8]. На этой встрече опытные спикеры из этой компании делились своим опытом работы, рассказывали про свой карьерный рост. Большое внимание с их стороны было уделено умению следить за новыми технологиями и адаптироваться к ним. Рассказчики рассказали также про набор используемых ими технологий, так, был упомянут язык программирования ABAP[9]. Кроме этого был сделан некоторый акцент на способах попасть в данную компанию, спикеры упоминали два способа – через их собственный бесплатный курс на позицию джуниор-разработчика, а также через вакансию на места мидл-разработчиков и выше. В ходе встречи выяснилось, что компания занимается аутсорсингом, на данном рынке они выступают в качестве исполнителей исполнителей.

# Глава 1 Теоретические основы

## 1.1 Общая характеристика курса «Контроль версий с помощью Git»

Для изучения системы контроля версий **«**Git[10]**»** был использован курс от компании EPAM[11], под названием «Version Control with Git[11]». Данный курс предоставляет обширные теоретические знания подкрепленные примерами, а также позволяет закреплять полученные знания на практике с помощью тематических тестов и практических заданий. Информация в курсе представлена как в текстовом, так и в видео формате.

## 1.1.1 Обзор возможностей

Курс предоставляет возможность изучения следующих модулей: «настройка Git», «базовые операции», «внутреннее устройство git», «отмена действий», «ветки и merge», «stash, tags, remote»

Помимо теории в курсе также предоставлен раздел для практики, в котором присутствует выбор уровней сложности, а именно пользователю предлагаются варианты: «I can Win», «Bring it On», «Hurt me plenty», «Hardcore», «Nightmare!»

### 1.1.2 Модули курса

#### 1.1.2.1 Настройка Git

В данном модуле учащийся сразу получает некоторое количество вводной информации, касающейся основных концепций выбранной системы контроля версий, её достоинства и недостатки. После пользователю показывают как установить и настроить Git.

#### 1.1.2.2 Базовые операции

Этот модуль посвящён основам работы с Git: тому как создавать и клонировать репозиторий, как «подтянуть» данные из удаленного репозитория к себе на машину. Так, последнее действие можно сделать при помощи команды «git pull».

#### 1.1.2.3. Внутреннее устройство git

Модуль повествует о структуре папок, о значении названий тех или иных файлов, о скрытых папках

#### 1.1.2.4 Отмена действий

В этом модуле затрагивается один из ключевых аспектов системы Git. А именно, про то как в данной системе контроля версий отменять свои действия. Так, стоит отметить следующую задачу – «Вы сделали коммит и забыли добавить пару изменений. Задача - добавить данные в уже созданный коммит. Укажите все решения.», у неё есть два ответа. Первый – «git commit --amend» и второй – «git gui» + «amend last commit». Стоит отметить, что информация этого модуля не раз выручала при работе над проектом.

#### 1.1.2.5 Ветки и merge

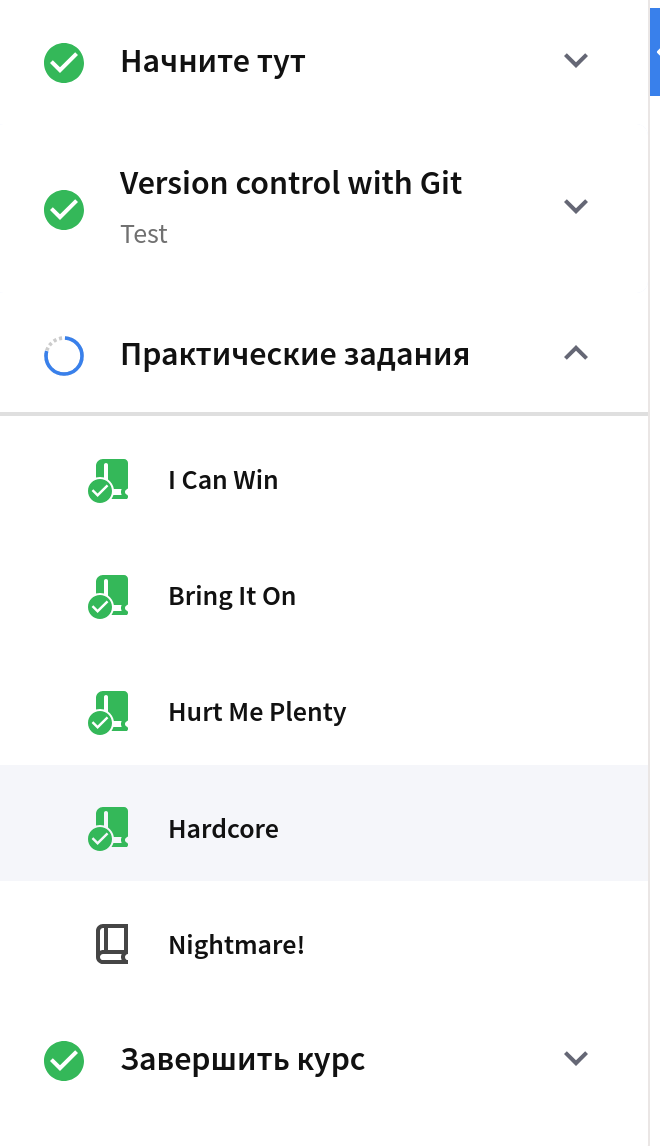
Этот модуль рассматривает такие понятия как:cherry pick, rebase. Что позволяет не только понять механизм того как это делать с технической точки зрения, но и дает понимание того как и когда его надо использовать. Знания этого модуля были одними из наиболее часто применяемых.

#### 1.1.2.6 Stash, tags, remote

Модуль содержит информацию о одноименных командах системы Git, кроме этого стоит отметить «Branch strategy».

### 1.1.3 Итоги

Продемонстрировать итоги изучения курса по системе Git можно наглядно следующим изображением



Представлена статистика по курсу по Git

## 1.2 Общая характеристика курса «Python»

Для изучения языка программирования Python был предложен хендбук[13] от компании Яндекс, содержащий теоретическую информацию, а также большое количество разнообразных задач для закрепления полученных знаний.

### 1.2.1 Модули курса

Весь курс делится на следующие темы: введение, базовые конструкции Python, Коллекции и работа с памятью, функции и их особенности в Python, ООП, библиотеки для получения и обработки данных. Каждая из этих тем в свою очередь разбивается на подтемы, которые подробнее будут рассмотрены далее.

#### 1.2.1.1 Введение

Данный раздел предоставляет вводную информацию необходимую для работы с автоматической системой проверки, установки и настройки среды разработки.

#### 1.2.1.2 Базовые конструкции

Данный модуль подразделяется на следующие подразделы: ввод и вывод данных, условный оператор, циклы, вложенные циклы.

Из задач этого модуля следует отметить задачу «Игра в угадайку» из раздела про циклы. Она является крайне интересной ведь является интерактивной задачей на алгоритм двоичного поиска. По условию загадано число, от 1 до 1000 включительно, требуется получая ответы вида: «Больше», «Меньше», «Угадал!» отгадать загаданное системой проверки число не более чем за 10 попыток. Ниже приложено решение автора

left = 1

r = 1000

ans = False

last = ''

count = 0

while left + 1 < r:

mid = (left + r) // 2

print(mid)

count += 1

res = input()

last = res

if (res == 'Угадал!'):

ans = True

break

if (res == 'Больше'):

left = mid + 1

if (res == 'Меньше'):

r = mid - 1

if ans is False:

if last == 'Больше':

print(r)

else:

print(left)

print(count)

#### 1.2.1.3 Коллекции и работа с памятью

Этот модуль дает пользователю информацию о встроенных коллекциях, их наиболее часто используемых методах и особенностях их реализаций. Модуль включает в себя: «строки, кортежи, списки», «множества, словари», «списочные выражения, «встроенные возможности для работы с коллекциями», «потоковый ввод\вывод» . Так, например, можно заметить существенное отличие от C++, а именно – реализация set и dict не чезе красно-чёрное дерево, а через хеш-таблицы.

Из этих задач наиболее интересными были задачи на списочные выражения, потому что в них достаточно интересно думать о том как сделать всё в одну строчку. Например: «преобразовать список чисел в строку вида x1 - x2 -... » и вот как она была решена:

(''.join((' - ' + str(char) for char in sorted(list(set(numbers)))))).lstrip(' - ')

#### 1.2.1.4 Функции и их особенности в Python

В этом модуле речь идет об особенностях работы функций в Python, которых достаточно много. Больше внимание уделяется таким важным нюансам как: именованные и позиционные аргументы, значения по умолчанию, управление размерами стека рекурсии, декораторы.

Модуль подразделяется на: «функции, области видимости, передача параметров в функции», «позиционные и именованные аргументы, лямбда функции», «рекурсия, декораторы, генераторы.

Отметим задачу, являющуюся хорошей иллюстрацией особенностей функций в Python бы следующая задача:

«Напишите функцию grow, принимающую, неопределенное количество позиционных и именованных параметров.

Для каждого именованного параметра, требуется найти все позиционные, кратные длине его названия, а затем увеличить их на его значение.  
Результатом функции должен, являться кортеж преобразованных позиционных параметров».

И соответственно решение:

def grow(\*args, \*\*named\_args):

lst = [x for x in args]

for key in named\_args:

for i in range(len(lst)):

if args[i] % len(key) == 0:

lst[i] += named\_args[key]

return tuple(lst)

#### 1.2.1.5 ООП

Этот модуль посвящен объектно-ориентированному программированию. Подразделяется же модуль на: «объектная модель Python», «волшебные методы, переопределение методов, наследование», «модель исключений Python, try, except, else, finally». Крайне увлекательными задачами этого модуля являются задачи на обработку исключений, ведь они предлагают необычный формат задач, где надо что-то сломать. Пример – «Давайте немного поиграем в «багоюзеров».

Вашему решению будет предоставлена функция func, которая принимает два позиционных параметра и производит с ними некую **математическую** операцию.

Предложите вызов функции, который гарантированно породит ошибку внутри функции». И решение – func({'': ''}, {'': ''})

#### 1.2.1.6. Библиотеки для получения и обработки данных

Это последний модуль курса, содержащий следующие темы: math и numpy, pandas, request. Модуль представляет большой интерес для изучения ведь предлагает поработать с достаточно необычными библиотеками. Стоит привести два показательных примера.

Пример 1. «Вася пришёл на образовательный семинар и обнаружил, что зрителей на мероприятии — N, а количество мест — M.  
Помогите Васе определить вероятность того, что он попадёт на семинар»

from math import comb

n, m = map(int, input().split())

print(comb(n - 1, m - 1), comb(n, m))

А без библиотек пришлось бы или считать факториалы, либо пред просчитывать биномиальные коэффициенты(например через дп).

Пример 2. «Напишите программу, находящую наибольшие общие делители для всех переданных в стандартный поток последовательностей чисел.»

from sys import stdin

import math

res = 0

for line in stdin:

lst = list(map(int, line.split()))

res = 0

for x in lst:

res = math.gcd(res, x)

print(res)

Эта задача запомнилась так как сочетает удобство потокового ввода и реализованных функций в библиотеках. Хоть алгоритм Евклида и сам по себе не очень сложный и маленький, но это всё равно было очень приятной находкой.

#### 1.2.2 Итоги

Итоги изучения данного курса можно продемонстрировать следующим изображением: 

На данном изображении представлена статистика изучения курса по Python

## 1.3 Выводы к главе 1

- Git позволяет распределить нагрузку на разработчиков

- Git это «Семь раз отмерь один отрежь»

- Python – лаконичный и красивый язык, позволяющий быстро реализовать свои задумки

- В Python много синтаксического сахара, который надо запомнить

- В Python достаточно много значительных отличий от C++

- Если в приложении важна скорость, то лучше рассмотреть другой язык программирования

- Python не всегда сильно уступает C++, так как некоторые его библиотеки написаны на более производительных языках

- теория гораздо лучше закрепляется на практике

# Глава 2 Индивидуальный проект «Emphizor»

В данной главе представлен разработанный проект (при наличии).

Описать самостоятельно реализованный проект (проекты): назначение, фрагменты кода с комментариями, копии экранов.

Укажите состав команды, репозиторий проекта, этапы разработки проекта.

Состав команды: Ткаченко Радомир, Лукьянович Никита

## 2.1 Этап предпроизводства разработки проекта «Emphizor»

### 2.1.1 Концепция и Идея

Приложение аналог Anki [14] и Remnote [15] для интервального повторения, с использованием библиотеки py-fsrs [5].Целевая аудитория приложения - студенты, школьники, обучающиеся.

Исследование рынка: Изучили основные концепции существующих аналогов(Remnote, Anki), провели опрос целевой аудитории и выяснили, что важным фактором является высокая стоимость использования существующих аналогов.

Цели: Создание приложения дешевого, удобного для пользователей приложения

Творческая: Попробовать себя в дизайне архитектуры и интерфейса.

Техническая: Использовать pyside6, supabase, PostgreSQL.

### 2.1.2 Документ Концепции

#### 2.1.2.1 Краткое описание

оконное приложение для интервального повторения, созданное наподобие существующих конкурентов, но предоставляющее пользователю выбор того, каким ИИ он желает пользоваться.

Весь функционал по созданию карт которого умещался бы на одном экране. создание карт же предусматривается как самим пользователем, так и с помощью искусственного интеллекта (генерация ответов по заданному вопросу). Карты в свою очередь должны группироваться по так называемым тегам, у одной карты может быть несколько тегов. Теги карты говорят о том, какие темы охватывает её вопрос. Для практики с картами мы предусматриваем два варианта: классический(сразу возникает вопрос и кнопка по которой можно узнать ответ и оценить своё знание темы) и Concept Connect режим(в котором на экране появляется 9 плиток, каждая из которых содержит либо вопрос, либо ответ.

Целью пользователя является совмещение плиток с соответствующими вопросами и ответами. Также в наборе есть 1 лишняя плитка, чтобы не возникло ситуации, когда в конце остаётся лишь 2 плитки, которые соответствуют друг другу).

Также мы планируем добавить в наш проект кастомизацию интерфейса, а именно пользовательский выбор цветовой темы приложения. Пользователь бы выбирал 2 цвета, которые бы и составляли все градиенты отображаемые в приложении. Всю информацию о пользователе, такую как:имя, пароль, адрес электронной почты, его теги, карты, ревью его ответов приложение будет хранить в облачном хранилище.

#### *2.1.2.2 Назначение, целевая аудитория*

наш продукт предназначен для оптимизации процесса повторения, изучения нового материала на основе современных технологий, учитывающих недавние исследования в области психологии. Предназначается же он для студентов, школьников, в целом всех людей которые занимаются саморазвитием.

#### *2.1.2.3 Основные концепции*

мэтчинг подходящих плиток в режиме Concept Connect вместе с ведением рейтинга ответа учитывающего время и правильность, квиз с отзывом в режиме Practice, выбор тегов реализован через нажатие на кнопки с нужными названиями. Учет правильности ответов пользователя, его знания карт и связь этого с частотой их появления реализована через алгоритм FSRS(Free Spaced Repetition Shcheduler)

#### *2.1.2.4 Уникальные особенности*

пользовательский выбор ИИ, кастомизация интерфейса, Concept Connect режим тренировки.

#### *2.1.2.5 Примеры похожих приложений:*

Сейчас на рынке существуют следующие аналоги: Anki, Remnote

### 2.1.3 Разработка Дизайн-Документа (ADD)

#### *2.1.3.1 Опыт использования*

создание тегов для карточек, создание карточек, группировка карт по тегам

кнопки меню, (возможно горячие клавиши)

Концепции:

Карточки: карточка — единица «знаний» в нашем приложении заключающая в себе вопрос и ответ.

Теги: карточки имеют теги, указывающие на то к какой теме они относятся.

Использование ИИ: Для удобства и экономии времени при создании карточек.

Сюжет: Вам надо выучить что-то и вы нашли наше приложение!

Элементы интерфейса: Главный экран, список тегов, экран создания карт

UI/UX: Чистый, минималистичный интерфейс , понятные переходы.

Стиль: минималистичный

Технический Дизайн: QT Designer, PySide6.

#### *2.1.3.2 Прототипирование*

Основной задачей прототипирования на данном этапе является разработка чернового варианта интерфейса, дизайна, наполнение его функционалом и далее выход к mvp и сбор обратной связи от заинтересованных людей через Telegram.

#### *2.1.3.3 Планирование:*

В ресурсы нашей команды входят: 2 студента, идея

## 2.2 Этап производства разработки проекта «Emphizor»

**2.2 Второй этап разработки приложения Emphizor – «Производство (Production)»  
  
Цель: Создание всех элементов интерфейса в соответствии с дизайн-документом и планом производства.  
  
2.2.1. Разработка дизайна интерфейса (Art Development)**

#### 2.2.1.1. Создание макета интерфейса

Создание разметки расположения основных элементов интерфейса

#### 2.2.1.2 Наполнение макета

В соответствии с ранее разработанной разметкой расставить элементы интерфейса по своим местам

#### 2.2.1.3 Настройка стилей

Задачей этого этапа является подбор и установка стилей ранее добавленных элементов интерфейса при помощи CSS

#### 2.2.1.4 Анимация

### Данный этап предполагает добавление анимаций при взаимодействии пользователя с интерфейсом. Например: анимации нажатия кнопок, анимации переходов между меню. 2.2.2 Программирование (Programming) 2.2.2.1. Реализация концепций (Conception Implementation)

#### По задумке данный этап должен в себя включать наполнение ранее созданного интерфейса функционалом для ответа приложения на действия пользователя. Также должны быть реализованы игровые режимы тренировки: Practice, Concept Connect. Помимо этого должен быть реализован пользовательский выбор цветовой темы.

#### 2.2.2.2 Разработка Искусственного Интеллекта (AI Development)

Целью этого периода разработки является разработка функционала связанного с взаимодействием пользователя с выбранным им искусственным интеллектом. А именно реализация автозаполнения поля ответа карты при заданном пользователем вопросе.

#### 2.2.2.3 Добавление SFX

Задачей этого пункта устанавливается поиск подходящих звуковых эффектов для приложения с их последующей интеграцией в продукт.

### 2.2.3 Оптимизация (Optimization)

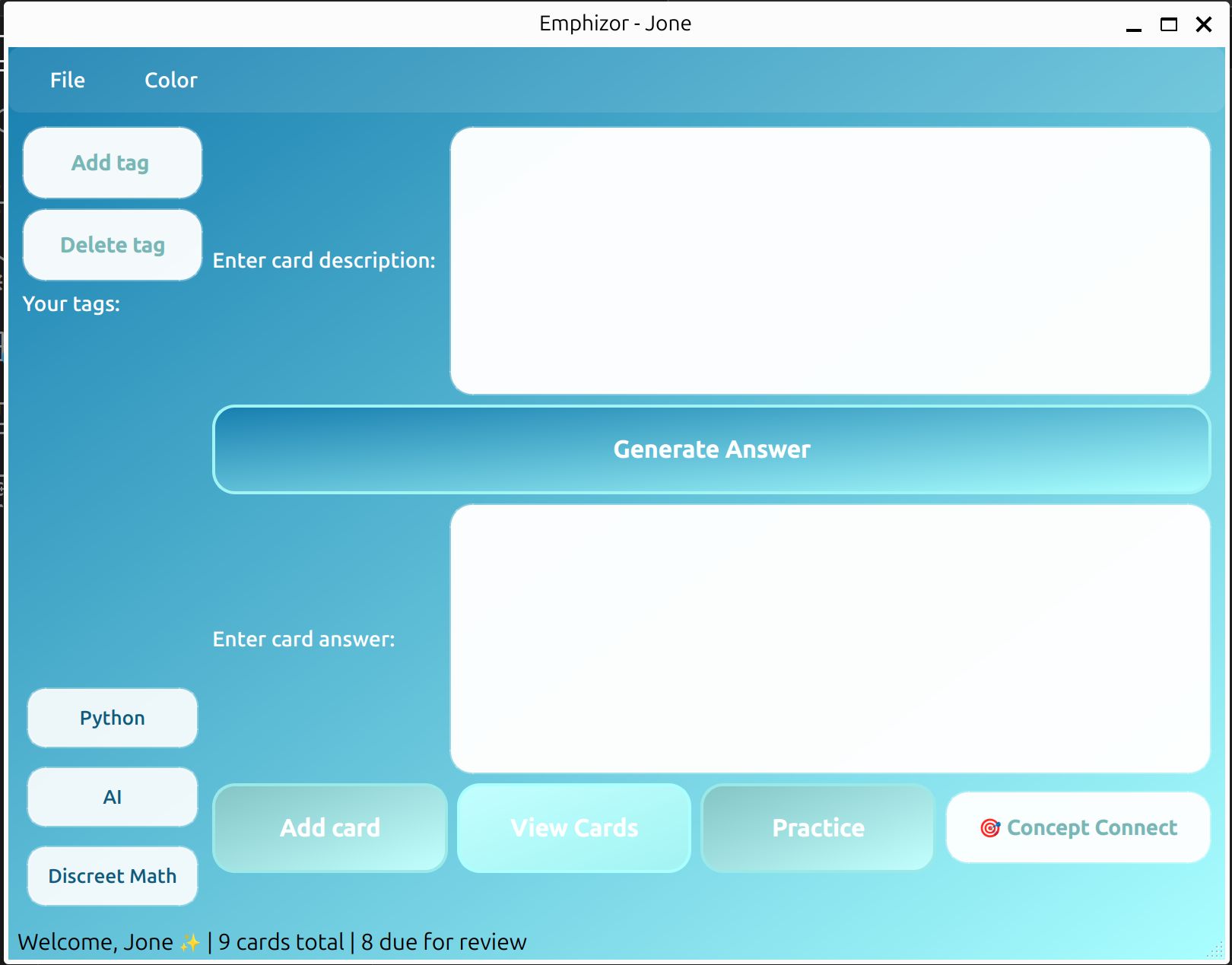
### После окончания работ по этапу программирования, планируется оптимизация существующих процессов и механик для улучшения быстродействия продукта на пользовательских устройствах. Особенно большое внимание на этом этапе планируется уделить потреблению приложением оперативной памяти.

### 2.2.4 План работ по этапу «Программирование»

Для демонстрации планируемых работ по программированию в удобном виде приводится следующая таблица:

| **Даты** | **Планируемая деятельность** |
| --- | --- |
| 02.07.2025 | Программирование кнопок создания тегов, разработка механизма аутентификации пользователя |
| 04.07.2025 | Программирование кнопок создания карт, синхронизация локальных действий пользователя с облачным хранилищем |
| 05.07.2025 | Поиск и исправление багов для выхода к mvp |
| 07.07.2025 | Программирования пользовательского выбора цветовой темы приложения |
| 08.07.2025 | Программирование кнопок удаления карт и тегов |
| 09.07.2025 | Поиск багов и их исправление |
| 10.07.2025 | Оптимизация потребления оперативной памяти путём поиска и реализации более эффективных структур данных |

### 2.2.3. Дизайн Интерфйеса

По нынешней задумке, интерфейс приложения должен состоять из: поля ввода вопроса карты, поля ввода ответа карты, кнопки добавления карты, кнопки добавления тега, списка существующих тегов, кнопки генерации ответа искусственным интеллектом, кнопки для запуска режима Practice, кнопки для запуска режима Concept Connect. По предварительному черновика располагаться они должны следующим образом: 



На данном изображении изображен предварительный вариант компоновки элементов интерфейса

### 2.2.4 Разработка бэкенд части приложения

-Аутенфикация, облачное хранение информации о пользователе

### 2.2.5 Тестирование ( Testing)

Тестирование на предмет наличия багов, таких как вход в учетную запись несуществующего пользователя, отсутствие добавленных карт и т.п.

### 2.2.6 Создание Звуковых Эффектов (SFX Creation)

Создание звуковых эффектов при помощи Audacity, а также поиск уже существующих звуков в публичных сборниках.

### 2.2.7 Интеграция Звука в Приложение (Audio Integration)

Данный этап предполагает наложение созданных и найденных звуковых эффектов на действия пользователя, такие как: нажатие на кнопки, печать текста, перемещение плиток.

### 2.2.8 Интеграция и Тестирование (Integration & Testing)

#### 2.2.8.1 Альфа-Тестирование (Alpha Testing)

- Тестирование приложения внутри команды разработчиков.  
- Обнаружение и исправление ошибок.

#### 2.2.8.2. Багфиксинг (Bug Fixing)

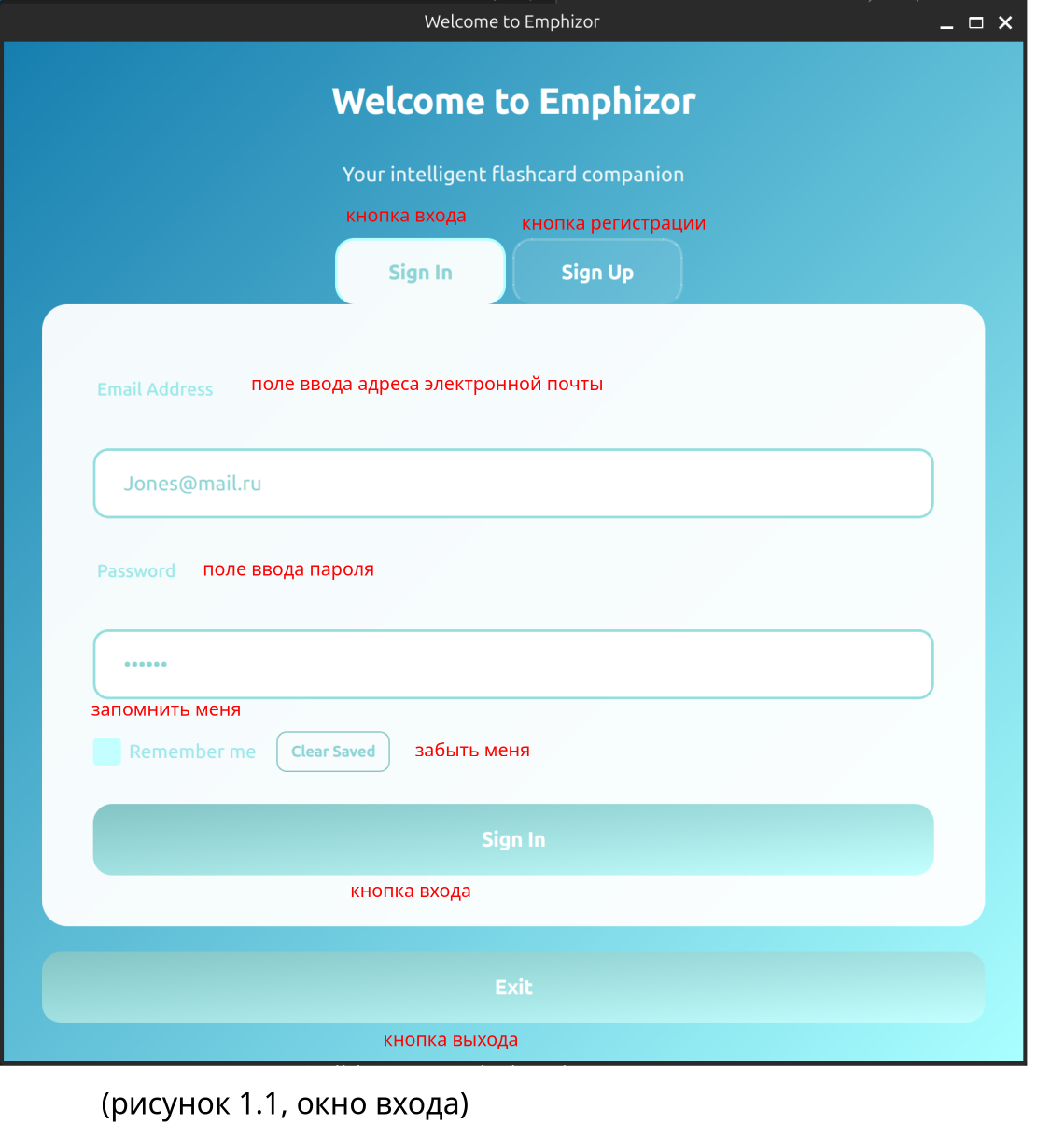
- Исправление всех обнаруженных ошибок и багов.

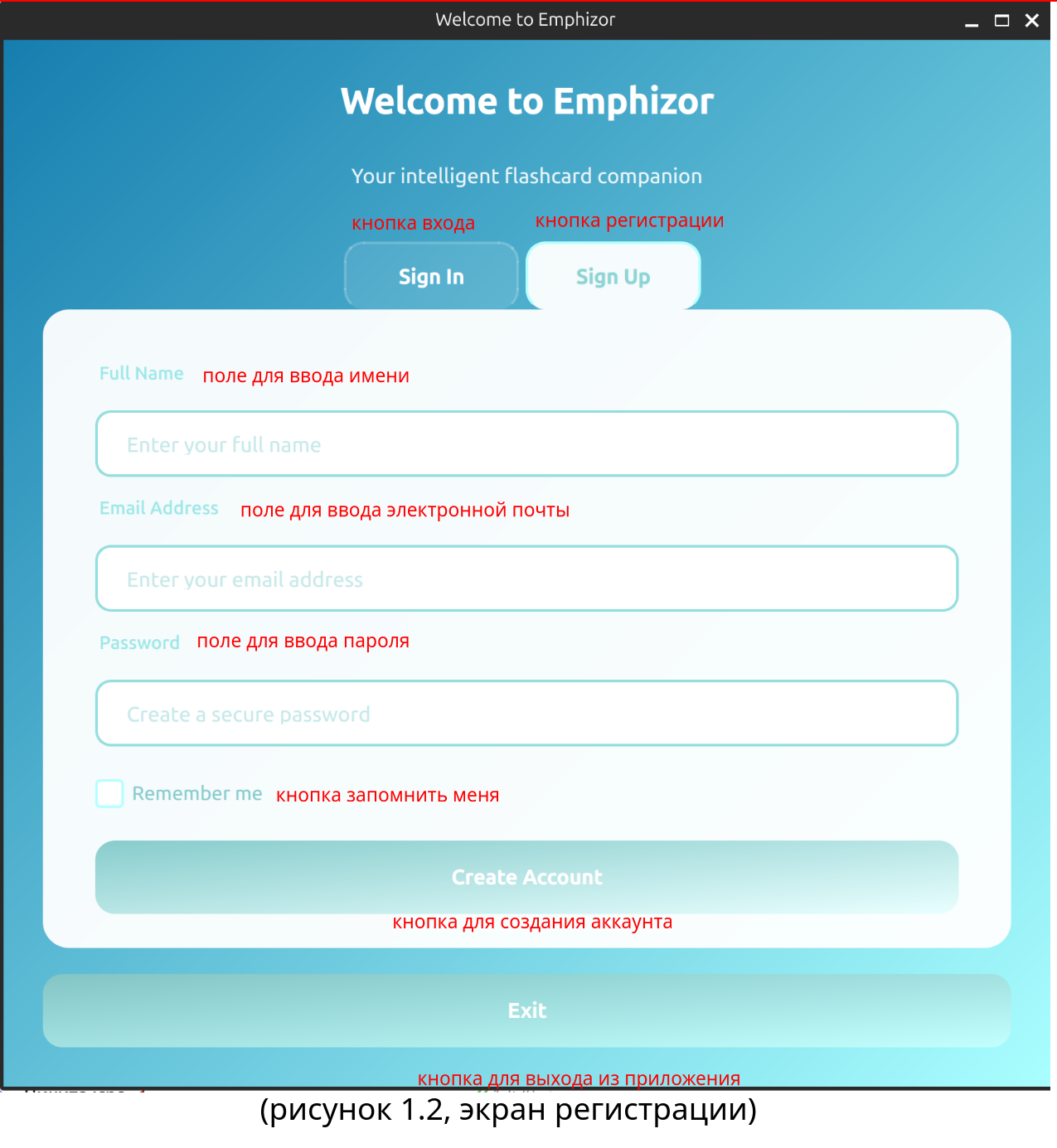
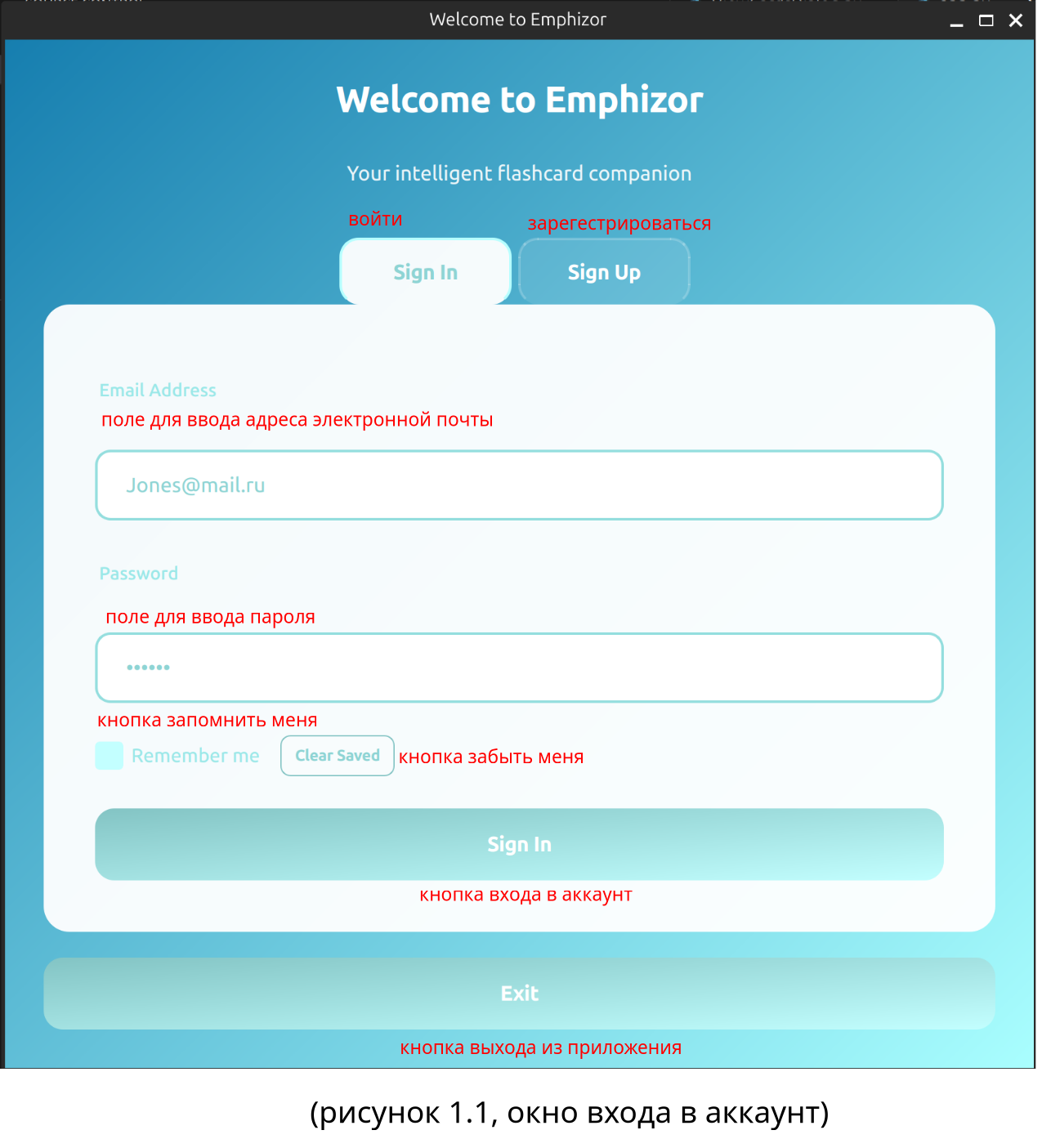
#### 2.2.8.3. Оптимизация (Optimization)

- Оптимизация производительности приложения на основе результатов тестирования.

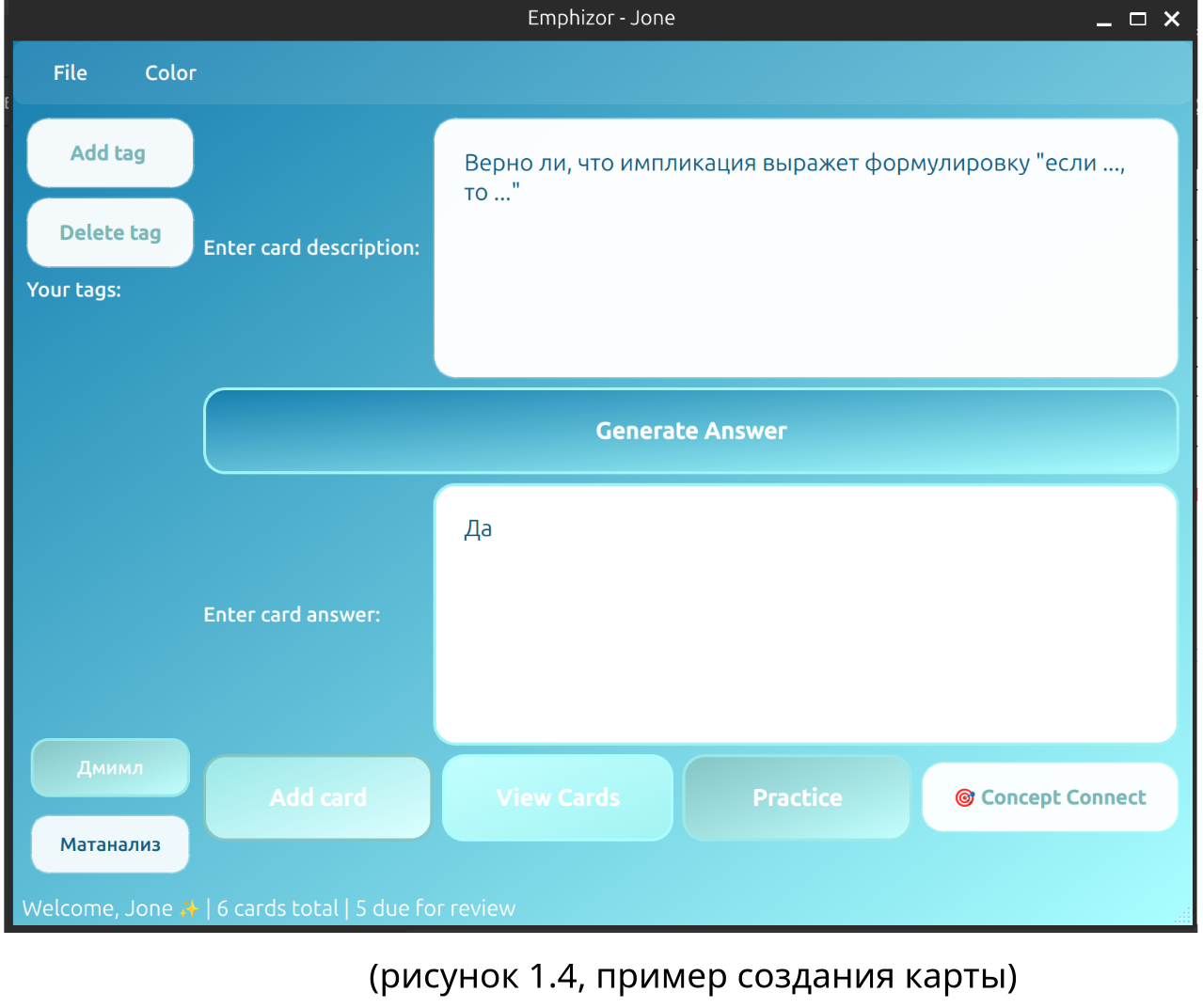
## 2.3 Интерфейс приложения

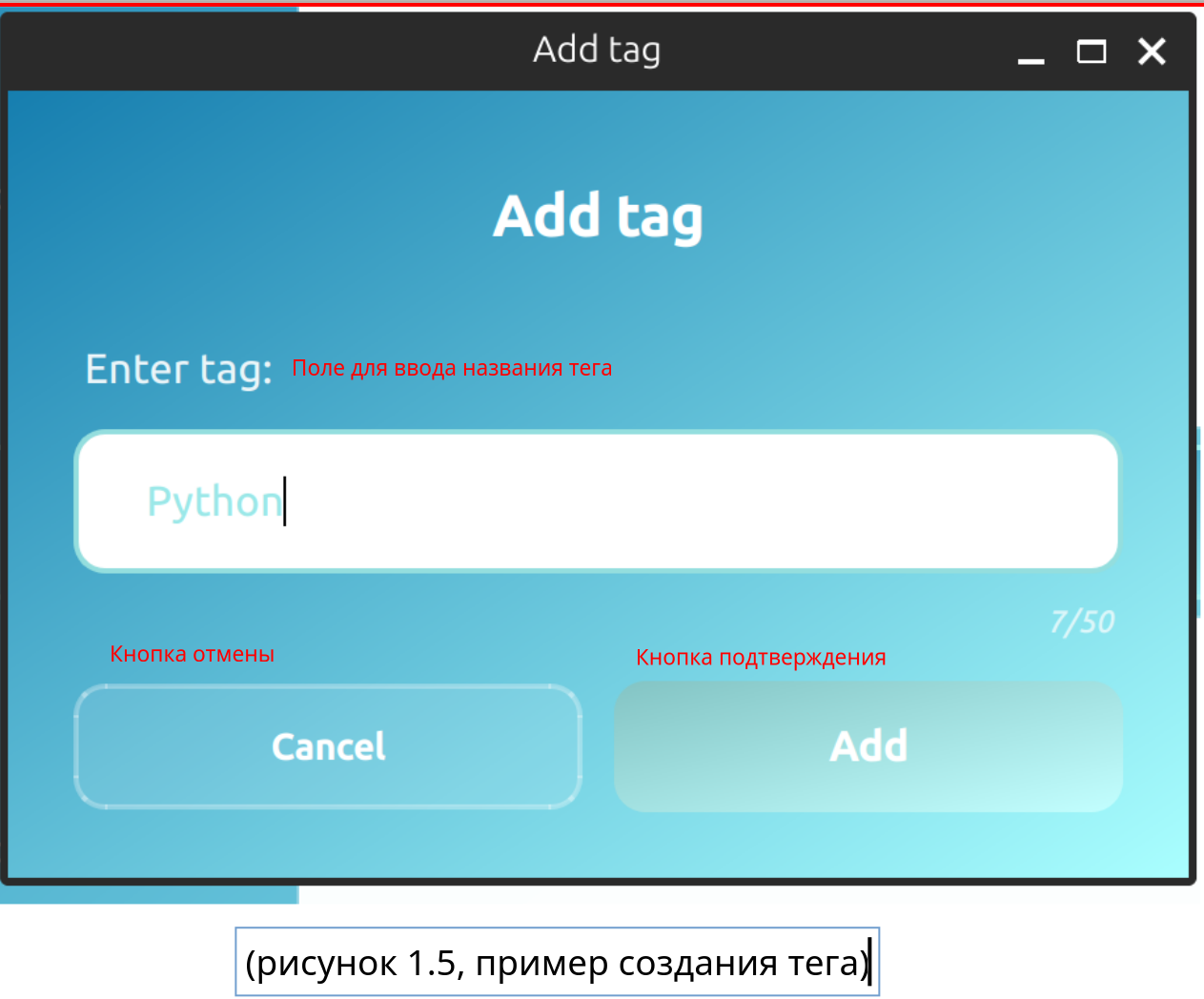
При запуске приложения вас встречает следующее окно

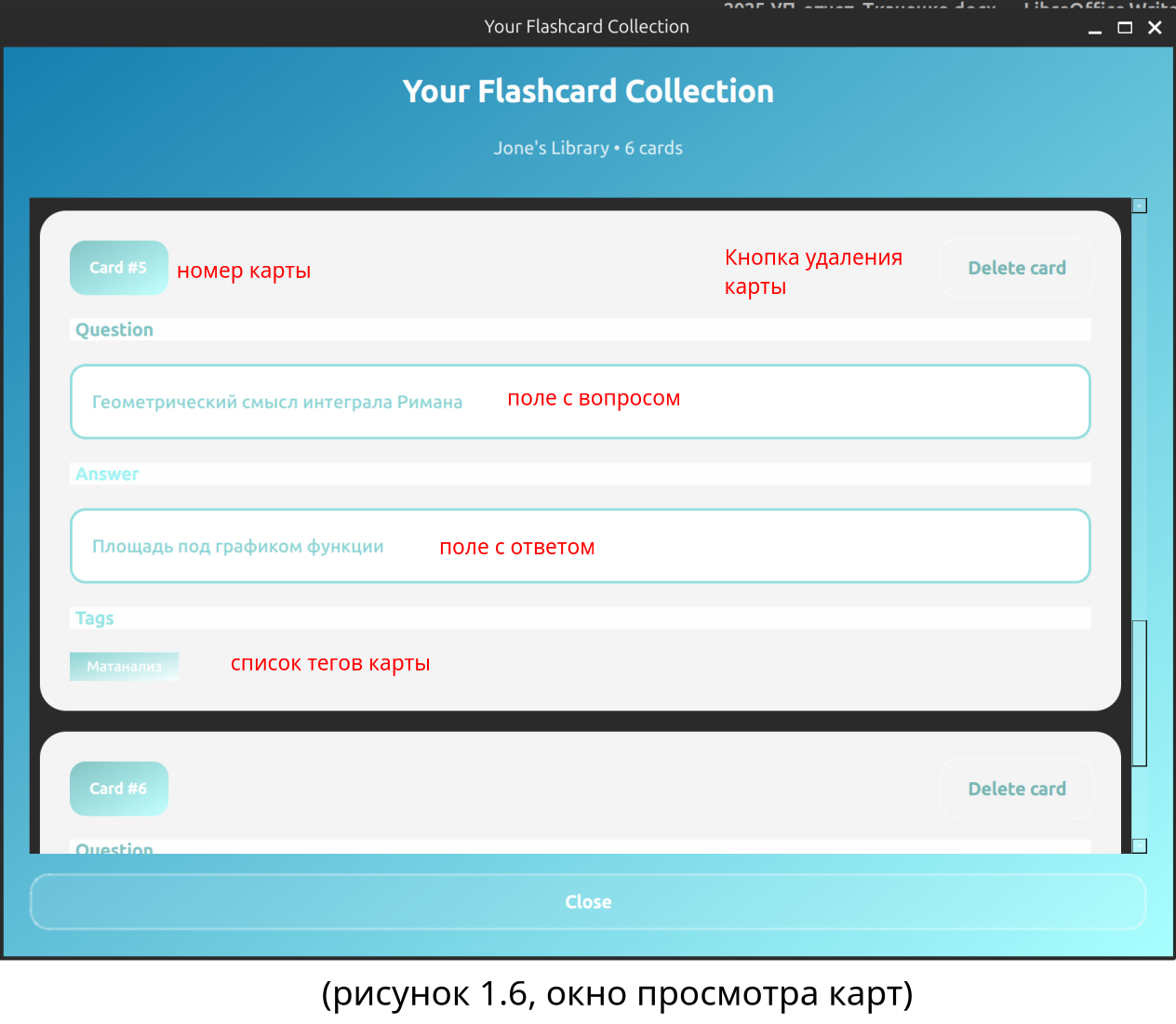


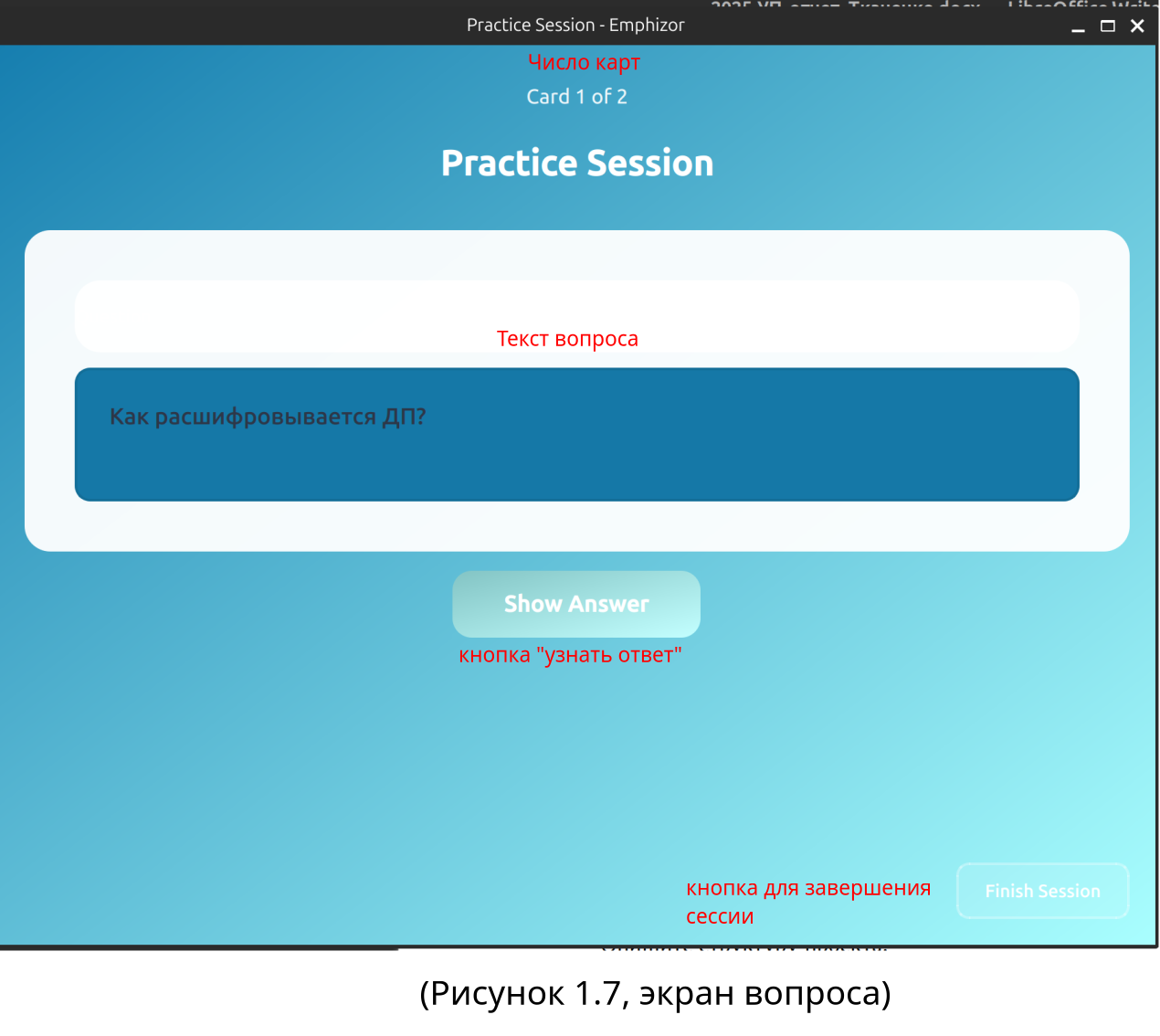


Теперь покажем, как создать карту. Для этого в списке тегов, нужно выбрать те, которые относятся к карте, которую вы хотите создать. Далее необходимо вписать информацию в о наполнении карты и ответе, в соответствующие поля и нажать кнопку «Add card»

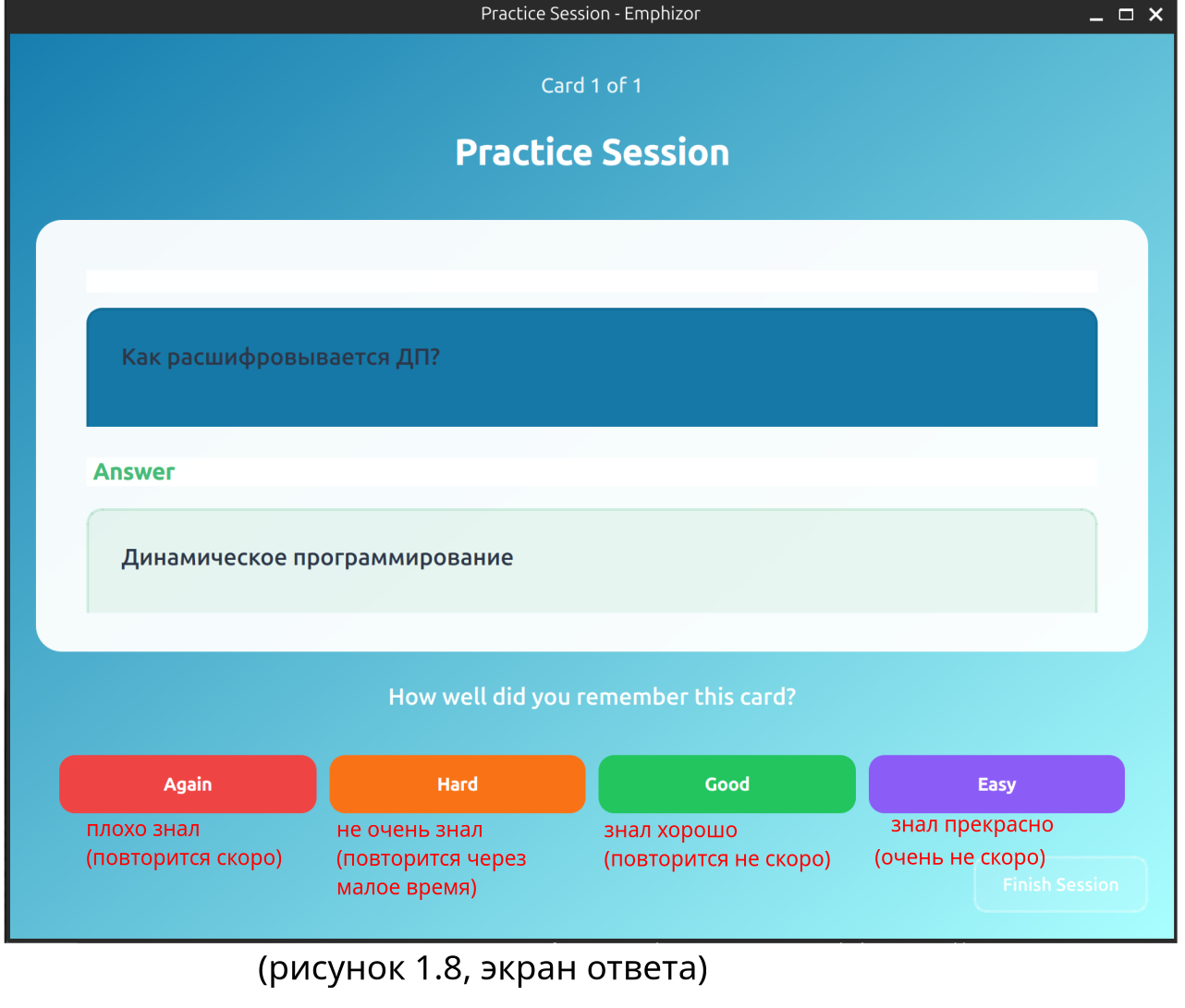
Теперь покажем, как создать тег. Для этого нажмите на кнопку «Add tag» и во всплывающем окне напишите название вашего тега и нажмите «Add». 

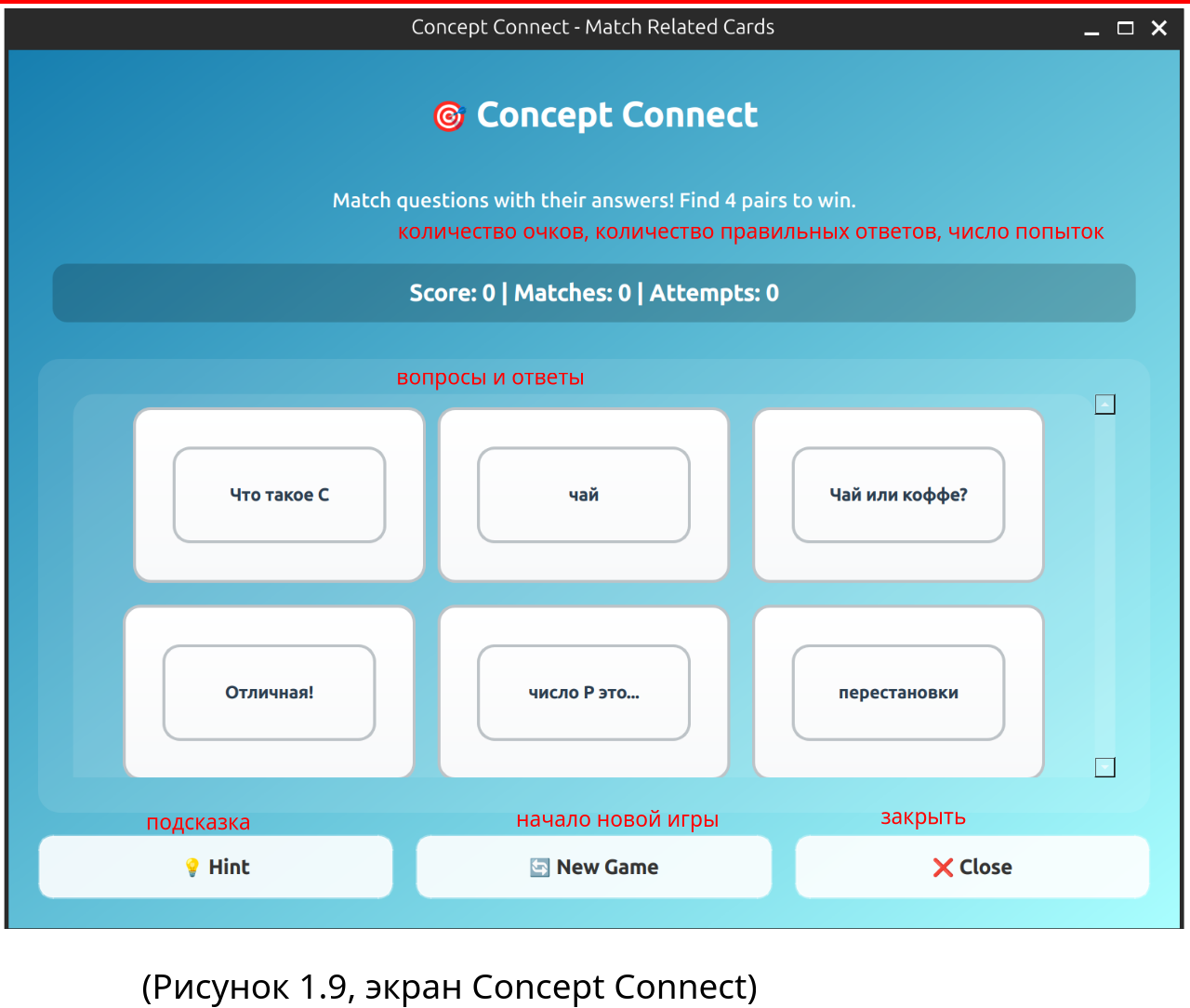
Для просмотра карт, в рабочей области нажмите на кнопку «View cards». Перед вами появится окно содержащее информацию о ваших картах, где вы также можете удалить лишние.

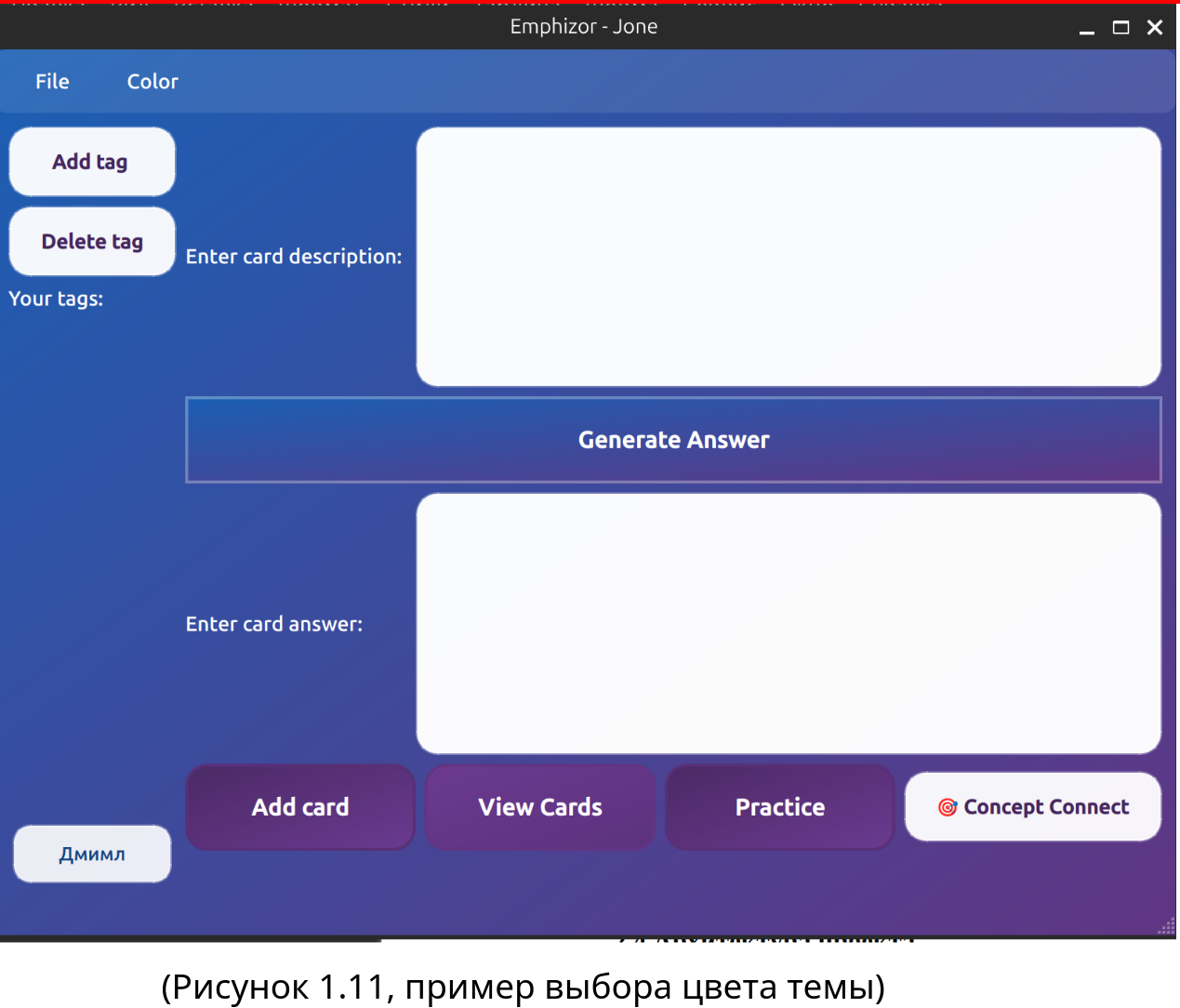
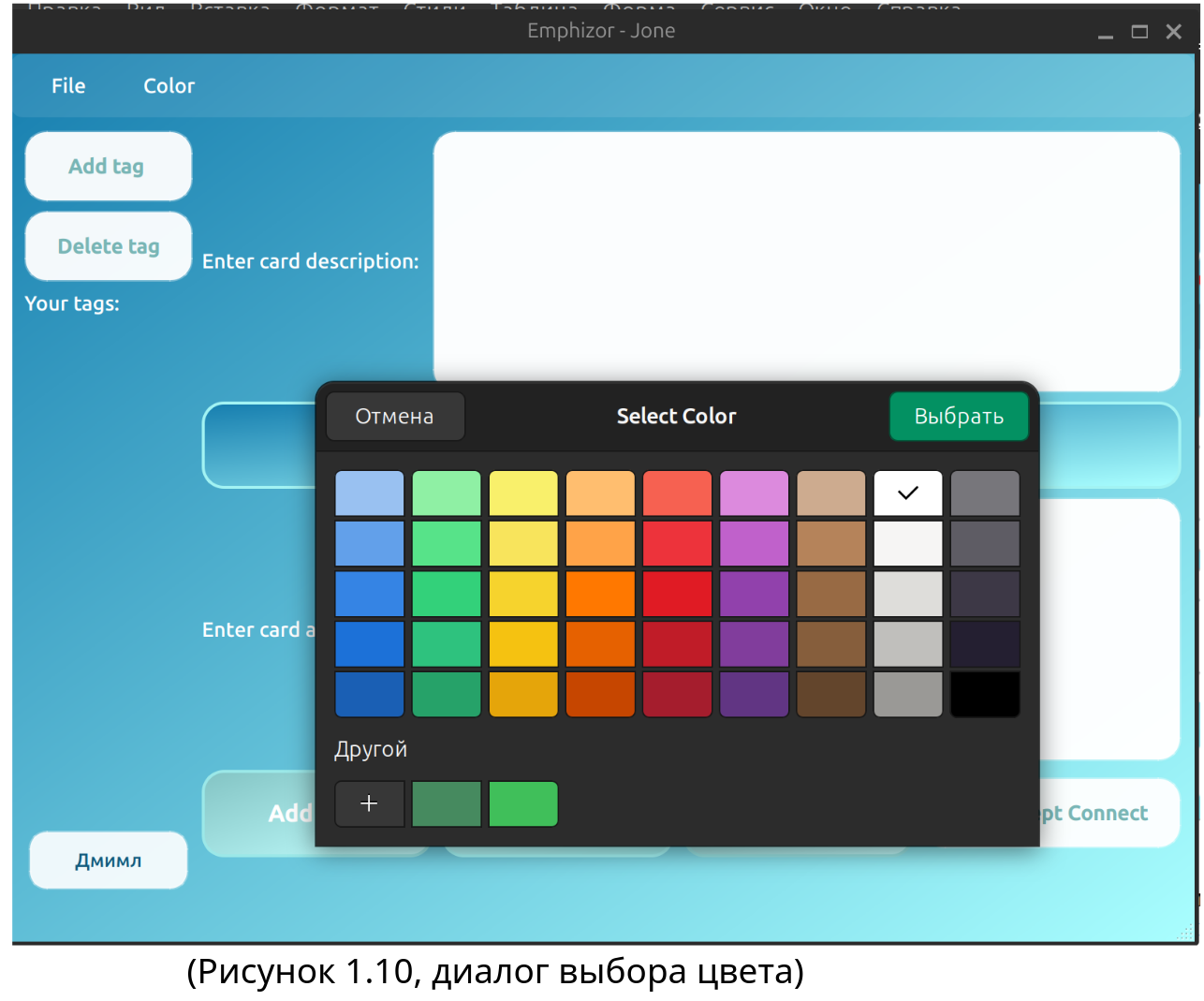
Теперь давайте отработаем созданные карты. Для этого существует два режима. Начнем с классического. Для этого нажмите на кнопки тегов, которые вы бы хотели повторить, далее нажмите на кнопку «Practice». Перед вами появится окошко, с вопросом, написанным на одной из ваших карт.



Если далее вы нажмете на «Finish session», то практика завершится. Если же вы нажмете на «Show Answer», то программа покажет вам ответ и предложит вам оценить ваше знание вопроса, от вашего ответа зависит то как часто эта карта будет далее появляться.

Теперь попробуем второй режим тренировки, для этого нажмите в рабочей области кнопку «Concept Connect» 

Также приложение позволяет менять цветовую гамму, для этого в рабочей области выберите раздел «Color», затем выберите первый или второй цвет градиента вы хотите поменять. Откроется диалог выбора цвета.

Вот, например, что получится если выбрать сочетание синего и фиолетового. Таким образом вам были представлены основные детали интерфейса программы. 

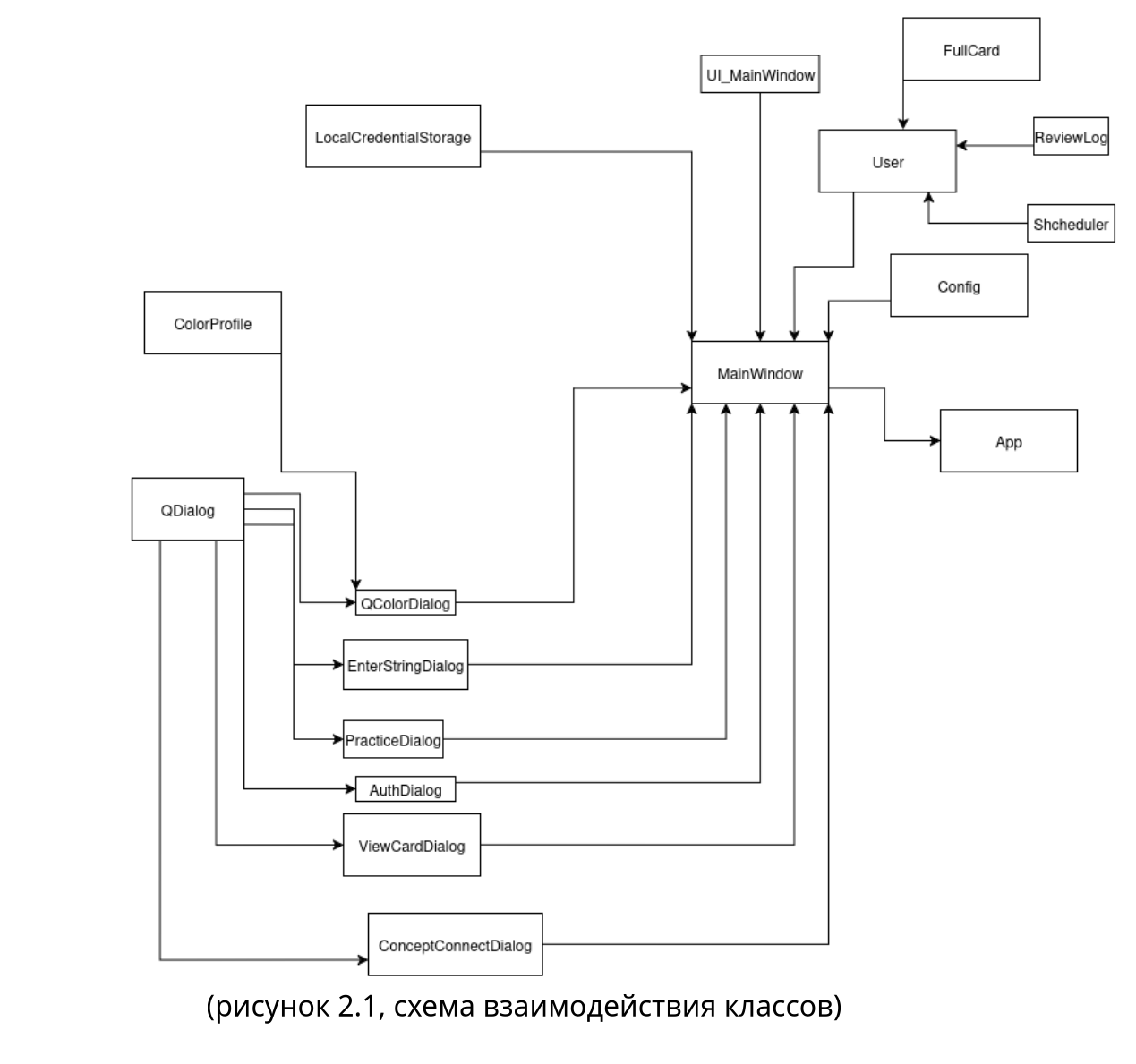
## 2.4 Архитектура проекта

В нашем проекте есть основной класс — MainWindow, на котором находятся все виджеты, кнопки и на котором располагается основная рабочая область программы.

Есть вспомогательные классы диалогов, которые отвечают за отображение и функционирование соответствующих им диалогов, например: AuthDialog, PracticeDialog, ViewCardsDialog, EnterStringDialog, ConceptConnectDialog.

Также существуют вспомогательные классы, предназначенные для хранения некоторого набора «однородных» данных, например ColorProfile, User, FullCard, Config.

Кроме них используются классы из py-fsrs, такие как: Shcheduler, ReviewLog. Более подробно схема их взаимодействия представлена на схеме ниже:



Стрелка от одного класса к другому обозначает «задействован в». Так например, отсюда видно, что класс ReviewLog задействован в User, а User в App(последняя связь происходит через MainWindow)

Также ниже приводится сравнительная таблица Emphizor с аналогами.

|  | Emphizor | Anki | Remnote |
| --- | --- | --- | --- |
| модель распространения | полностью бесплатное приложение | полностью бесплатное приложение | условно бесплатное, использование ИИ и хранилище ограничено для бесплатных пользователей |
| способы тренировки | классический, Concept Connect | классический | классический |
| ИИ возможности | есть поддержка ИИ, пользовательский выбор модели | нет нативной поддержки ИИ | есть поддержка ИИ |
| доступные форматы | десктопное приложение | десктопное приложение, мобильное приложение | веб-приложение, мобильное приложение, десктопное приложение |
| Коммерческая направленность | некоммерческое | некоммерческое | коммерческое |



## 2.5 Реализация бизнес-логики приложения

В приложении я разрабатывал общую архитектуру а также, программную часть, отвечающую за смену цвета интерфейса, дизайн, отображение и внешний вид кнопок, отрисовку интерфейса, создание и добавление тегов, удаление тегов и карт, поиск и фикс багов и т. д.

Пример метода класса, отвечающего за удаление тега:

def delete\_tag\_button\_clicked(self):

to\_delete\_names = self.get\_selected\_tags()

for button in self.tag\_buttons[:]:

text = button.text().strip()

if text in to\_delete\_names:

idx = self.ui.verticalLayout.indexOf(button)

if idx != -1:

item = self.ui.verticalLayout.takeAt(idx)

w = item.widget()

if w:

w.setParent(None)

w.deleteLater()

self.tag\_buttons.remove(button)

for card in self.user.full\_cards:

card.tags = card.tags - to\_delete\_names

self.tags -= to\_delete\_names

Метод, отвечающий за удаление карт:

def delete\_card(self, card\_to\_delete):

self.user.full\_cards.remove(card\_to\_delete)

while self.cards\_layout.count():

item = self.cards\_layout.takeAt(0)

if item.widget():

item.widget().deleteLater()

if self.user.full\_cards:

for i, full\_card in enumerate(self.user.full\_cards):

card\_widget = self.create\_card\_widget(full\_card, i + 1)

self.cards\_layout.addWidget(card\_widget)

else:

no\_cards\_widget = QFrame()

no\_cards\_widget.setStyleSheet("""

QFrame {

background: rgba(255, 255, 255, 0.1);

border: 2px dashed rgba(255, 255, 255, 0.3);

border-radius: 20px;

padding: 30px;

margin: 15px;

}

""")

no\_cards\_layout = QVBoxLayout(no\_cards\_widget)

no\_cards\_icon = QLabel("📚")

no\_cards\_icon.setAlignment(Qt.AlignmentFlag.AlignCenter)

no\_cards\_icon.setStyleSheet("font-size: 48px; margin-bottom: 10px;")

no\_cards\_layout.addWidget(no\_cards\_icon)

no\_cards\_label = QLabel("No cards found")

no\_cards\_label.setAlignment(Qt.AlignmentFlag.AlignCenter)

no\_cards\_label.setStyleSheet("""

color: white;

font-size: 18px;

font-weight: 600;

margin-bottom: 5px;

""")

no\_cards\_layout.addWidget(no\_cards\_label)

no\_cards\_hint = QLabel("Create some flashcards to get started!")

no\_cards\_hint.setAlignment(Qt.AlignmentFlag.AlignCenter)

no\_cards\_hint.setStyleSheet("""

color: rgba(255, 255, 255, 0.7);

font-size: 14px;

font-style: italic;

""")

no\_cards\_layout.addWidget(no\_cards\_hint)

self.cards\_layout.addWidget(no\_cards\_widget)

self.cards\_layout.addStretch()

subtitle\_labels = [self.header\_layout.itemAt(i).widget() for i in range(self.header\_layout.count())]

for label in subtitle\_labels:

if isinstance(label, QLabel) and f"{self.user.name}'s Library" in label.text():

label.setText(f"{self.user.name}'s Library • {len(self.user.full\_cards)} cards")

break

## 

## 2.6 Выводы к главе 2

Резюмируя всё вышеизложенное, можно сказать сделать следующие выводы: этап проектирования архитектуры является одним из ключевых в разработке и позволяет сэкономить много времени в дальнейшем, несмотря на то что не всегда получается следовать плану к этому надо стремиться, soft-skills не менее важны чем hard-skills, ведь работая в команде можно успеть гораздо больше, не надо всё разрабатывать с нуля, ведь есть больше количество open source библиотек в которых уже реализовано большое количество функций, язык Python действительно позволяет очень быстро проверять свои задумки, для правильной совместной работы надо заранее согласовывать то, каким образом будет происходить работа с Git, особенно Branch strategy, разработка дизайна приложения оказалась не такой простой, как казалось.

# Заключение

В результате индивидуальной работы было решено 190 задач по Python, пройден курс по Git, разработано приложение для интервального повторения, получен опыт работы в команде и реальный опыт использования системы контроля версий Git, также получены знания в области CSS, а также изучено большое количество смежных к теме проекта тем, таких как: кривая забывания, алгоритм FSRS, PySide6.

Важной частью всей проделанной работы был тайм-менеджмент, халатное отношение к нему в первые дни разработки привела к большим переработкам и усталости, снижению эффективности рабочего процесса.

Курс по Python был полностью прочитан, первые 5 параграфов были решены на 100%, последующие модули решались на 50%. Общий процент решенных задач составил 67%, то есть приблизительно две трети от общего числа задач.

Курс по Git был полностью прочитан, были выполнены все тесты на максимальный балл. Практические задания были выполнены не на всех уровнях сложности, а именно было принято решение не решать задания из пункта «Nightmare!»

Было написано большое количество отчётов. При их написании было допущено много ошибок, связанных с экономией времени. В результате на их исправление ушло еще больше времени, чем если бы они сразу были сделаны должным образом. Выбранная форма ведения отчётов помогла выработать привычку следить за временем выполнения рабочих задач.

Были получены навыки работы с UML диаграммами, стоит сказать о неэффективной трате времени на попытки построения диаграммы в обычных графических редакторах.

В индивидуальном проекте был получен опыт распределения задач. Была .разработана архитектура приложения, разработан дизайн приложения, его интерфейс, реализованы core механики приложения, реализована персонализация интерфейса, была проделана работа над стилями при помощи CSS.

Также были получены знания о рабочем процессе в реальных IT компаниях благодаря посещению компании «LeverX».

Были установлены цели на дальнейшее развитие проекта: добавление новых интерактивных режимов обучения, усовершенствование дизайна и пользовательского опыта, расширение возможностей пользователя для кастомизации приложения, улучшение архитектуры, работа над качеством кода, добавление в нее некоторой модульности для облегчения процесса модификации программы. Последние пункты были выделены особенно важными в связи с тем, что репозиторий планируется сделать публичным со всем исходным кодом.

# Список использованных источников

1. Caple, C. (1996). «The Effects of Spaced Practice and Spaced Review on Recall and Retention Using Computer Assisted Instruction». Dissertation for the degree of Doctor of Education, North Carolina State University.
2. Greene R. L. (2008). Repetition and spacing effects. In Roediger H. L. III (Ed.), Learning and memory: A comprehensive reference. Vol. 2: Cognitive psychology of memory (pp. 65–78). Oxford: Elsevier.
3. <https://doc.qt.io/qtforpython-6/>
4. <https://dl.acm.org/doi/10.1145/3534678.3539081?cid=99660547150>
5. <https://github.com/open-spaced-repetition/py-fsrs>
6. <https://supabase.com/>
7. <https://www.postgresql.org/>
8. <https://leverx.com/ru/>
9. <https://developers.sap.com/topics/abap-platform..html>
10. <https://git-scm.com/>
11. <https://www.epam.com/cis>
12. <https://learn.epam.com/detailsPage?id=601f195a-d408-4439-a16d-0630ed2a412e>
13. <https://education.yandex.ru/handbook/python>
14. <https://apps.ankiweb.net/>
15. <https://www.remnote.com/>