ПРОЕКТ

**персонального приложения ERP**

**Выполнил студент:**

**Ким Олег Герасимович**

**(поток «**Python-разработчик»  **от 24.05.2024)**

**2024**

**СОДЕРЖАНИЕ**

[1. Введение 3](#_Toc185643761)

[1.1 Обоснование выбора темы 3](#_Toc185643762)

[1.2 Определение цели 3](#_Toc185643763)

[2. Методы и подходы к разработке 3](#_Toc185643764)

[3. Обзор проекта 4](#_Toc185643765)

[4. Состав и структура приложения 6](#_Toc185643766)

[4.1 Используемые библиотеки 6](#_Toc185643767)

[4.2 Файловая структура проекта 8](#_Toc185643768)

[5. Анализ результатов 8](#_Toc185643769)

[6. Заключение 8](#_Toc185643770)

# Введение

Дипломная работа по разработке проекта приложения планирования ресурсов на языке Python.

## Обоснование выбора темы

Со временем в жизни увеличивается количество задач и процессов, их сложность и динамичность, а в наше время быстро растет и объем поступающей информации. Поэтому разработка приложения системы планирования ресурсов, важнейшим из которых является время, - важный шаг для повышения эффективности и оптимизации жизненных процессов, и борьбы с многозадачностью. Автоматизация управленческих функций и интеграция всех процессов в единую информационную систему становится необходимостью.

## Определение цели

Основной целью разработки приложения является автоматизация жизненных процессов: управление временем, собственной логистикой, финансами и прочими ресурсами. Приложение должно интегрировать данные из различных сфер жизнедеятельности в единую базу, обеспечивая их актуальность, доступность и целостность. Это дает возможность улучшить управление временем, повысить надежность принятия решений и снизить затраты на выполнение рутинных задач.

# Методы и подходы к разработке

Разработка проводилась в OS Windows 10 в среде Visual Studio Code, Python версии 3.11.9.

Методика основана на компоновке максимального количества навыков полученных в результате обучения. В связи с ограниченным временем в данной работе прорабатываются основные информационные потоки структуры приложения с эскизом интерфейса.

# Обзор проекта

Приложение включает в себя следующие элементы:

- основной процесс (main): запускает потоки телеграмм-бота и web api, выполняет обмен данными с потоком телеграмм-бота и базой данных;

- поток телеграмм-бота: организует интерфейс пользователя в телеграмм-чате. В качестве макета - вывод списка запланированных событий для пользователя с возможностями создания и редактирования расписания.

- поток web api: предназначен для организации интерфейса пользователя на web сайте. В качестве макета – работа с базой данных с помощью основных функций: создания, чтения, модификации и удаления данных.

Точка подключения – телеграмм-бот. При подключении бота пользователь отправляет команду /start или произвольное сообщение. Приложение определяет данные пользователя и регистрирует его.

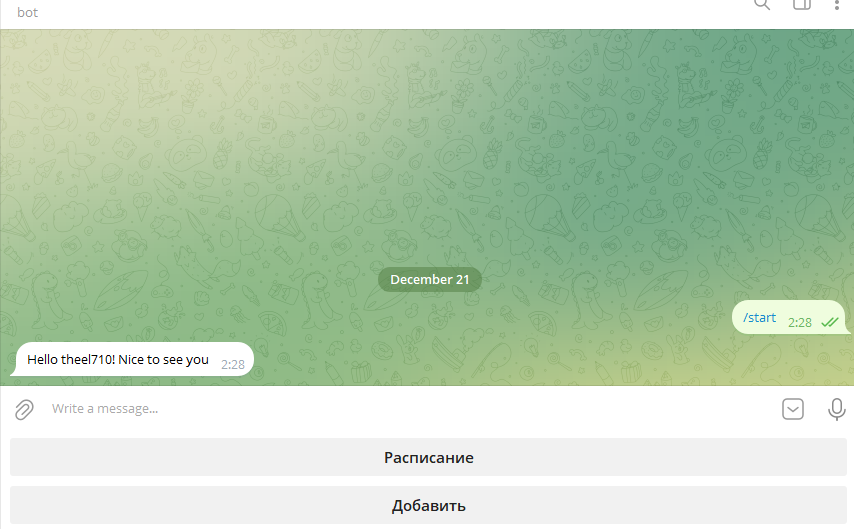


Рис.1 Подключение в телеграмм-боте.

Приложение ищет пользователя в оперативном списке класса TelegramUser(), и если пользователь новый создает новый объект класса TelegramUser(). Данные пользователя отправляются в основной процесс, а в качестве ответа возвращается список событий данного пользователя из базы данных

После этого пользователь может посмотреть свое расписание, нажав соответствующую кнопку меню.

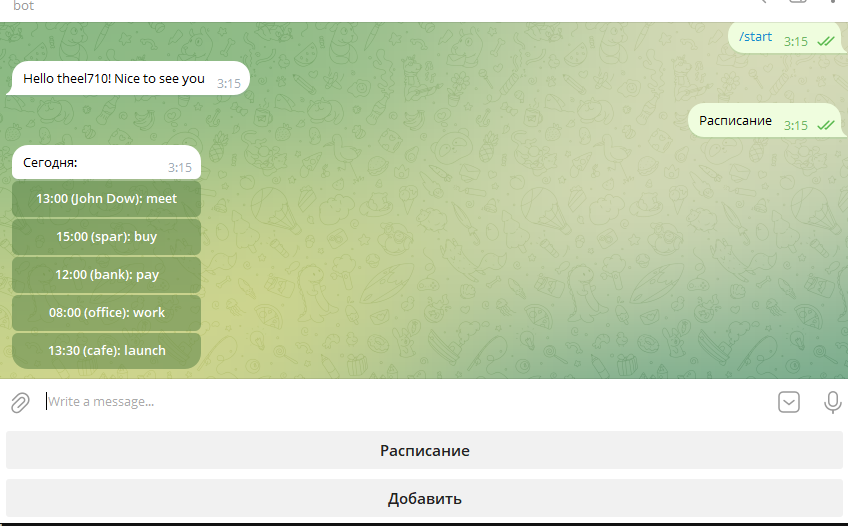


Рис.2 Вывод расписания.

При выборе события из меню появляется окно с операциями с эти событием.

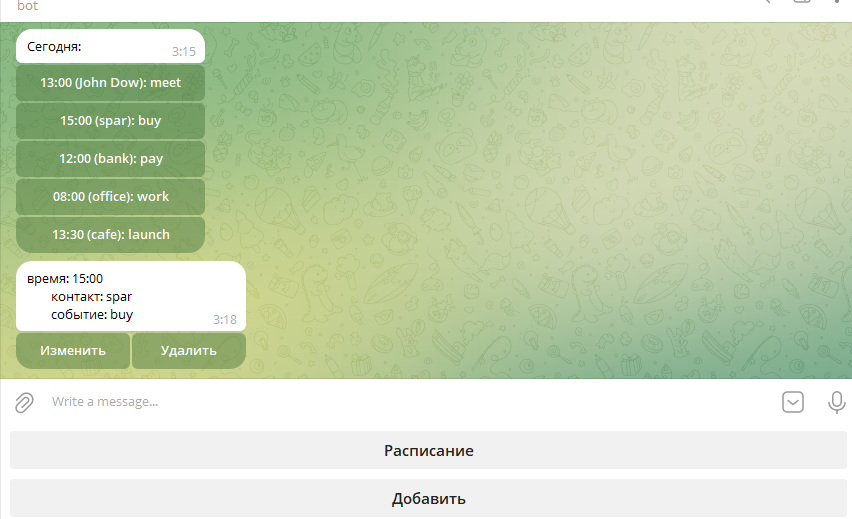


Рис.3 Работа с сообщением.

# Состав и структура приложения

## Используемые библиотеки

|  |  |
| --- | --- |
| Пакет | Версия |
| aiofiles  aiogram  aiohappyeyeballs  aiohttp  aiosignal  alembic  annotated-types  anyio  async-timeout  attrs  Babel  certifi  charset-normalizer  click  colorama  fastapi  frozenlist  greenlet  h11  idna  magic-filter  Mako  MarkupSafe  multidict  pip  propcache  pydantic  pydantic\_core  python-slugify  pytz  setuptools  sniffio  SQLAlchemy  starlette  text-unidecode  typing\_extensions  uvicorn  yarl | 24.1.0  2.25.1  2.4.4  3.8.6  1.3.1  1.14.0  0.7.0  4.7.0  4.0.3  24.2.0  2.9.1  2024.8.30  3.4.0  8.1.7  0.4.6  0.115.6  1.5.0  3.1.1  0.14.0  3.10  1.0.12  1.3.8  3.0.2  6.1.0  24.3.1  0.2.1  2.9.2  2.23.4  8.0.4  2024.2  65.5.0  1.3.1  2.0.36  0.41.3  1.3  4.12.2  0.34.0  1.18.3 |

## Файловая структура проекта

|  |  |
| --- | --- |
| Каталог | Назначение |
| p-Diploma | корневой каталог проекта |
| * data | файлы классов пользователей |
| * database | файлы взаимодействия с базой данных |
| * migrasions | структура миграций alembic |
| * telegram | файлы телеграмм-бота |
| * tests | каталог Unit тестов |
| * web | файлы Web api |
| alembic.ini  main.py  requirements.txt  Udatabase.db | * файл конфигурации базы данных * основной файл приложения * список используемых библиотек * файл базы данных |

# Анализ результатов

Язык Python и разработанные на его базе библиотеки предоставляют достаточный потенциал для разработки приложений данной направленности.

# Заключение

На мой взгляд приложения типа календарей и напоминаний не получают широкого распространения и такого частого использования как мессенджеры, поскольку отражают только поверхностную часть нашей жизнедеятельности. А работа с ними в основном представлена в ручном режиме.

Автоматизация мессенджеров, обработка больших объемов информации и наконец, возможности технологий AI открывают новые возможности в области информационной автоматизации жизненных процессов человека.

Технические требования

Текст письменной работы пишется на стандартном листе бумаги формата

А4 (в электронной форме).

Поляоставляются по всем четырем сторонам печатного листа: левое поле – 30 мм,правое – 10 мм, верхнее и нижнее – 20 мм.

Шрифт Times New Romanразмером 14, межстрочный интервал – 1,5. Диплом должен содержать введение, основную часть с разработкой и заключение. В образце содержания пример написания. Можете продолжать просто писать в этом файле и все. Скрины работы тоже приветствуются, так проверяющий понимает, что вы действительно сами писали и проект рабочий. Не забываем подписывать скрины в формате:

Рис. 1 Страница авторизации

А далее сам скрин

Если будут вопросы – пишите!

Удачи вам, у вас все получится!)