Техническое задание (ТЗ) Название проекта: Telegram-бот «**Расписание АИТ**» Исполнитель: Сиваков Николай, студент 3-го курса Цель: Разработка Telegram-бота для предоставления расписания занятий студентов АИТ в удобной форме.

****1. Общие положения:****

* ****Наименование программы:**** Telegram-бот «**Расписание АИТ**».
* ****Область применения:**** быстрый доступ к расписанию занятий для студентов и преподавателей АИТ.
* ****Основание для разработки:**** Учебное задание.

****2. Функциональные требования:****

* ****Получение расписания:****
  + ****Полное расписание (на текущую неделю):**** Команда /расписание. Возвращает расписание на текущую неделю.
  + ****Расписание на конкретный день недели:**** команда /расписание [день недели] (например, /расписание понедельник). Возвращает расписание на указанный день недели.
  + ****Расписание конкретной группы:**** Команда /группа [номер группы] (например, /группа 101). Возвращает расписание указанной группы на текущую дату.
  + ****Расписание на завтра:**** Команда /завтра. Возвращает расписание на следующий день.
  + ****Расписание на следующую неделю:**** Команда /расписание следующая неделя или /расписание nextweek.
* ****Обработка ошибок:**** Бот должен корректно обрабатывать ошибки:
  + Неверные команды.
  + Несуществующие группы.
  + Некорректный ввод данных.
  + Ошибки при обращении к внешнему ресурсу (сайт с расписанием).

****3. Нефункциональные требования:****

* ****Надежность:**** Бот должен корректно обрабатывать различные входные данные и ситуации.
* ****Производительность:**** Ответ на запросы не должен превышать 5 секунд.
* ****Совместимость:**** Бот должен быть совместим с последними версиями Telegram API.
* ****Пользовательский интерфейс:**** Четкое и понятное представление расписания в чате. (Например, в формате: “Понедельник:
  + 8:00 — 9:00 Математика (преп. Иванов)”)

****4. Архитектура (выбор подхода):****

* ****Взаимодействие:**** Бот будет взаимодействовать с внешним сервисом (<https://aitanapa.ru/>). Важно использовать запросы к API (если оно доступно) или хорошо структурированный способ парсинга данных с сайта.
* ****Обработка данных:**** данные, полученные с внешнего ресурса, будут обработаны и представлены пользователю в удобном формате.
* ****Хранение данных:**** бот не должен хранить данные в своей базе данных. Вместо этого он должен получать их в режиме реального времени из внешнего источника.

****5. Используемые технологии:****

* ****Язык программирования:**** Python.
* ****Библиотеки:****
  + python-telegram-bot: для работы с Telegram API.
  + requests: для отправки запросов на внешний ресурс.
  + [Библиотека для парсинга HTML/XML (BeautifulSoup, lxml или другая подходящая)]: для извлечения данных из HTML-страницы расписания (если нет API).
  + [Библиотека для работы с PDF (например, pymupdf, reportlab или другая, если используется PDF-файл)]: если расписание находится в PDF-документе.

****6. Ожидаемые результаты:****

* Разработанный бот должен предоставлять расписание в удобном виде.
* Бот должен корректно обрабатывать запросы пользователей.
* Бот должен быть устойчивым к возможным изменениям структуры данных на сайте с расписанием.

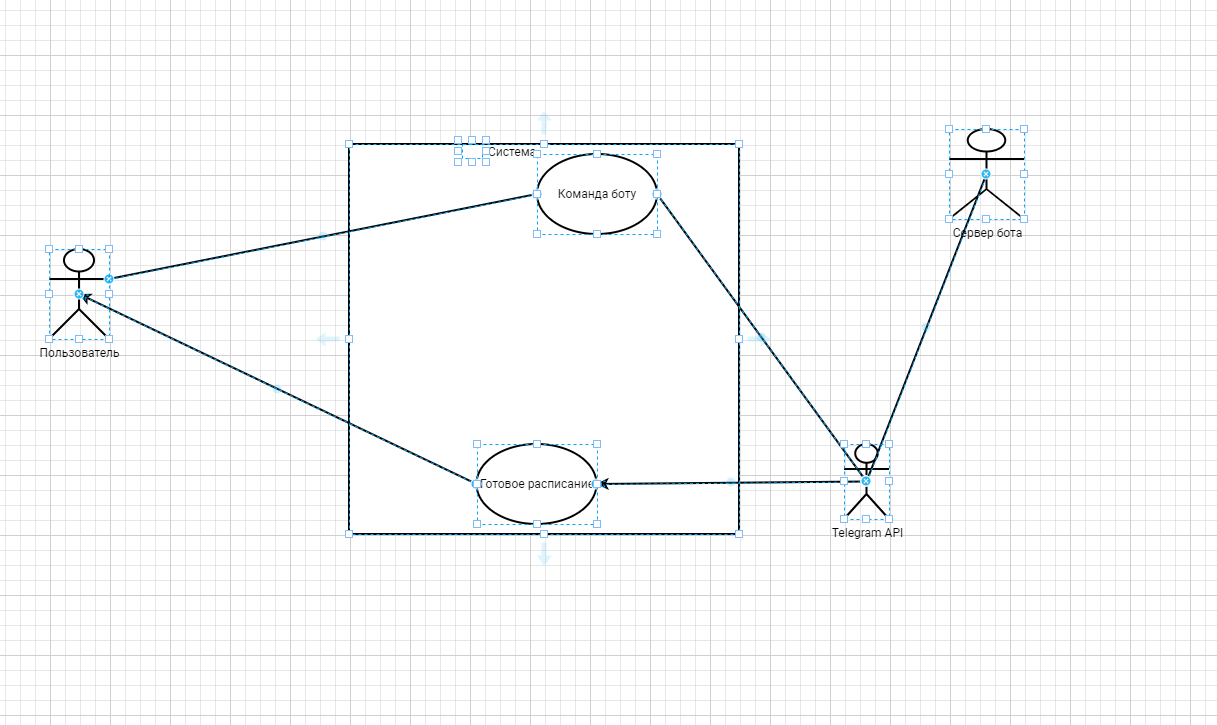
#### ***Шаг 2. Диаграмма UML взаимодействия клиента с сервером***

**Описание диаграммы:**

* **Акторы**:
  + Пользователь (студент, преподаватель).
  + Telegram API (посредник).
  + Сервер https://aitanapa.ru/ (хранение данных расписания).
* **Процесс взаимодействия**:
  1. Пользователь отправляет команду через Telegram-бота.
  2. Бот обращается к серверу Telegram API.
  3. Telegram API перенаправляет запрос серверу https://aitanapa.ru для получения расписания.
  4. https://aitanapa.ru возвращает данные о расписании.
  5. Telegram-бот формирует ответ и отправляет его пользователю через Telegram API.

**Диаграмма UML:**

**Создана диаграмма вариантов использования отображающая взаимодействие между пользователем, Telegram API и сервером бота:**



#### *****Шаг 3: Алгоритм работы*****

1. ***Запуск****:*
   * Пользователь вводит команду /start — бот отправляет приветственное сообщение.
2. ***Получение запроса****:*
   * Пользователь выбирает команду (например, /расписание или /группа N).
   * Бот проверяет корректность команды.
3. ***Обработка запроса****:*
   * Если запрос корректен, бот формирует запрос к серверу https://aitanapa.ru для получения данных расписания.
   * Если запрос некорректен, бот отправляет сообщение об ошибке.
4. ***Ответ пользователю****:*
   * Сервер возвращает данные, бот форматирует их и отправляет пользователю.
5. *****Завершение****:***
   * Бот ожидает следующего запроса.

Вот сам код 