|  |  |  |
| --- | --- | --- |
|  |  | УТВЕРЖДАЮ |
|  |  | Руководитель образовательной программы  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сергеева Е. Г.  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.  М.П. |

**ТЕХНИЧЕСКИЙ ПРОЕКТ**

«Telegram-бот для просмотра информации о рыночных активах»

|  |  |  |
| --- | --- | --- |
| СОГЛАСОВАНО |  | СОГЛАСОВАНО |
| Колледж ВятГУ  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Юдинцев Б. С.  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.  М.П. |  | Руководитель УП  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Долженкова М. Л.  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

2024

**Содержание**

[1. Концепция предлагаемого решения 1](#_Toc182835048)

[2. Пользовательский интерфейс 1](#_Toc182835049)

[3. Передача и хранение данных 8](#_Toc182835050)

[3.1. Описание основных сущностей 8](#_Toc182835051)

[3.2. Построение логической ER-диаграммы 10](#_Toc182835052)

[3.3. Построение физической ER-диаграммы 10](#_Toc182835053)

[3.4. Построение диаграммы потоков данных 11](#_Toc182835054)

[4. Бизнес-процессы системы 12](#_Toc182835055)

[5. Разработка алгоритма решения задачи 14](#_Toc182835056)

[5.1. Алгоритм регистрации пользователя и инициализации настроек 14](#_Toc182835057)

[5.2. Алгоритм взаимодействия с пользователем 15](#_Toc182835058)

[5.3. Алгоритм просмотра информации об активах 16](#_Toc182835059)

[5.4. Алгоритм управления избранными активами 17](#_Toc182835060)

[5.5. Алгоритм настройки уведомлений и оповещений 18](#_Toc182835061)

[5.6. Алгоритм отправки уведомлений 19](#_Toc182835062)

[5.7. Алгоритм настройки параметров бота 20](#_Toc182835063)

[5.8. Алгоритм обработки пользовательского ввода и ошибок 21](#_Toc182835064)

[6. Выбранная архитектура системы 22](#_Toc182835065)

[6.1. Общая архитектура 22](#_Toc182835066)

[6.2. Технологический стек 23](#_Toc182835067)

[6.3. Структура программы 23](#_Toc182835068)

# **Концепция предлагаемого решения**

Проект представляет собой Telegram-бот, который предоставляет пользователю актуальную информацию о рыночных активах (акциях, криптовалютах, валютных парах). Основные возможности бота включают:

* Получение текущих котировок активов.
* Просмотр исторических данных и графиков изменений цен.
* Управление списком избранных активов.
* Настройка уведомлений по различным условиям, таким как изменение цен или достижение заданного порога.
* Простой и удобный интерфейс на платформе Telegram с использованием текстовых команд и встроенных клавиатур.

# **Пользовательский интерфейс**

Пользовательский интерфейс реализуется через платформу Telegram с использованием текстовых сообщений и встроенных клавиатур (Inline Keyboards).

Макеты интерфейсов:

Главное меню является стартовой точкой взаимодействия с ботом. Оно содержит четыре основные кнопки, каждая из которых ведет к определенному функционалу:

* Информация об активах: предоставляет доступ к данным об акциях, валютных парах и криптовалютах.
* Избранные активы: позволяет управлять списком избранных активов.
* Настройки бота: открывает настройки бота, включая параметры уведомлений, часовой пояс и базовую валюту.
* Справка по командам и помощь: предоставляет информацию о возможностях бота и доступных командах.

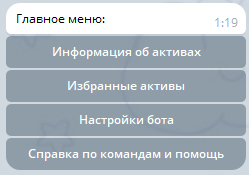


Рисунок 1 – Главное меню

Экран выбора типа актива. Содержит три основные категории:

* Акции: Данные о компаниях и их рыночных котировках.
* Валюты: Информация о курсах валютных пар.
* Криптовалюты: Данные о стоимости популярных криптовалют.
* На экране также доступна кнопка "Назад" для возврата в главное меню.

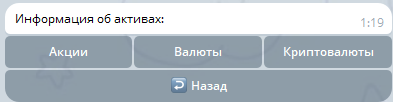


Рисунок 2 – Информация об активе

Экран выбора типа избранных активов. Экран позволяет:

* Просматривать информацию об избранных активах.
* Управлять списком избранного (например, удалять активы). Кнопка "Назад" возвращает пользователя в главное меню.

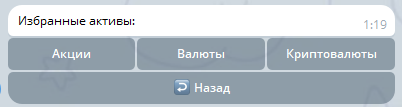


Рисунок 3 – Избранные активы

Экран для изменения параметров взаимодействия с ботом. Включает:

* + Настройку часового пояса.
  + Выбор базовой валюты.
  + Частоту уведомлений.

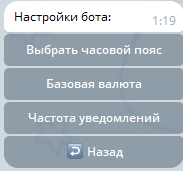


Рисунок 4 – Настройки бота

Экран помощи предоставляет пользователю справочную информацию о возможностях бота.  
Включает описание доступных команд и функций, таких как:

* Просмотр информации об активах (акции, валюты, криптовалюты).
* Управление избранным.
* Настройки (часовой пояс, базовая валюта, уведомления).
* Возможность получения графиков и аналитики.
* Кнопка "Назад" позволяет вернуться в главное меню.

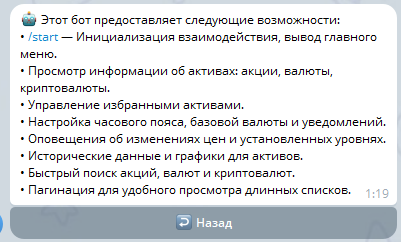


Рисунок 5 – Помощь

Экран акций отображает список доступных акций для просмотра. Пользователь может:

* Выбрать интересующую акцию из списка для получения подробной информации.
* Использовать кнопку "Поиск" для нахождения нужной акции.
* Переключаться между страницами акций, если список длинный.
* Кнопка "Назад" возвращает пользователя к разделу "Информация об активах".

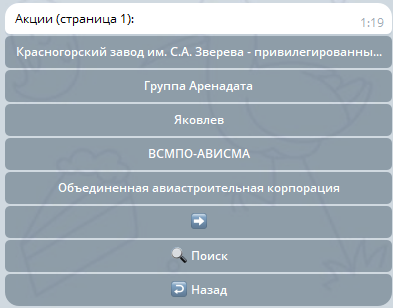


Рисунок 6 – Акции

Экран валют позволяет выбрать валютные пары для анализа. Основные функции:

* Выбор двух валют для сравнения курса.
* Подсветка базовой валюты пользователя (например, RUB).
* Подтверждение выбора через кнопку "Получить информацию".
* Кнопка "Назад" возвращает в меню "Информация об активах".

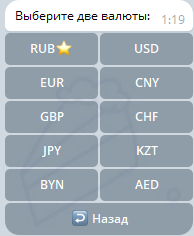


Рисунок 7 – Валюты

Экран криптовалют представляет список популярных криптовалют, доступных для анализа.

* Каждая кнопка криптовалюты ведет к детальной информации о выбранном активе.
* Кнопка "Назад" возвращает в раздел "Информация об активах".
* Список криптовалют разделен по строкам для удобной навигации.



Рисунок 8 – Криптовалюты

Экран сводки об активе предоставляет детализированную информацию о выбранном активе.

* Данные: текущая цена, минимальная и максимальная цена за день, рыночная капитализация.
* Управление активом:
  + График: кнопка для отображения графика изменения цены.
  + Уведомления: настройка уведомлений по изменению цены.
  + Избранное: возможность добавить/убрать актив в/из избранного.
  + Обновление информации: кнопка для получения актуальных данных об активе.

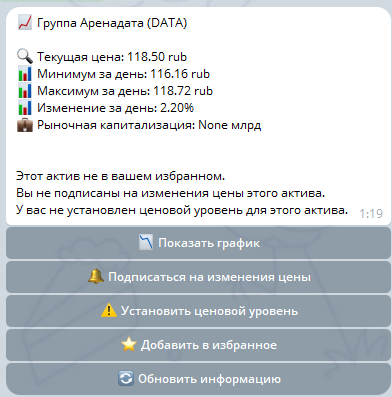


Рисунок 9 – Сводка об активе

Экран избранных активов отображает список активов, которые пользователь добавил в избранное.

* Навигация:
  + Возможность открыть подробную информацию о каждом активе.
  + Кнопка "Назад" возвращает к предыдущему экрану.

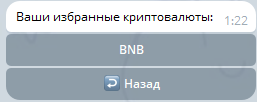


Рисунок 10 – Пример избранных активов

Экран выбора часового пояса отображает доступные часовые пояса, от -12 до +14 часов.

* Отметка текущего часового пояса: активный часовой пояс помечен галочкой.
* Навигация:
  + Кнопка "Назад" возвращает в настройки бота.



Рисунок 11 – Выбор часового пояса

Экран выбора базовой валюты, используемой для отображения данных.

* Особенности:
  + Список включает наиболее популярные валюты (RUB, USD, EUR, и т.д.).
  + Текущая базовая валюта выделена галочкой.
* Навигация:
  + Кнопка "Назад" возвращает в настройки.



Рисунок 12 – Выбор базовой валюты

Экран выбора частоты уведомлений позволяет настроить периодичность уведомлений.

* Функциональность:
  + Возможен выбор интервала между уведомлениями (1 минута, 5 минут, 10 минут и т.д.).
  + Текущая частота уведомлений помечена галочкой.
* Навигация:
  + Кнопка "Назад" возвращает в настройки.



Рисунок 13 – Выбор частоты уведомлений

Экран выбора периода для графика предназначен для настройки временного интервала, на основе которого строится график актива.

* Функциональность:
  + Доступны следующие таймфреймы: "За 1 день", "За 1 неделю", "За 1 месяц", "За 1 год".
* Навигация:
  + Кнопка "Назад" возвращает к предыдущему экрану.

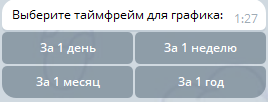


Рисунок 13 – Выбор периода для графика

# **Передача и хранение данных**

В ходе уточнения данных были выполнены следующие задачи, показанные в Таблице 2.

Таблица 1 – Задачи для уточнения структуры данных.

|  |  |  |  |
| --- | --- | --- | --- |
| **№ задачи** | **Название** | **Содержание** | **Результат** |
| 1 | Определение атрибутов сущностей | Определение и описание атрибутов для каждой сущности, таких как типы данных и ограничения | Таблица с атрибутами сущностей и их описанием |
| 2 | Представление атрибутов сущностей | Создание таблицы с атрибутами для каждой сущности | Готовая таблица с атрибутами сущностей |
| 3 | Построение диаграммы потоков данных | Определить, как будут передвигаться данные в программе; построить DFD диаграмму | Готовая диаграмма потоков данных (DFD) |

* 1. **Описание основных сущностей**

1. User (Пользователь)

Эта сущность описывает пользователей системы, взаимодействующих с Telegram-ботом.

Таблица 2 – Атрибуты таблицы User:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Название** | **Тип** | **Уникальность** | **Обязательность** | **Описание** |
| Id | Integer | Да | Да | Уникальный идентификатор |
| Username | String | Нет | Нет | Имя пользователя в Telegram |
| settings | JSON | Нет | Нет | Настройки пользователя |

Пример структуры JSON для поля settings:

{

 "timezone": "UTC+3",

 "base\_currency": "USD",

 "notification\_frequency": "10min",

 "favorites\_assets": [

  {"asset\_type": "stock", "ticker": "AAPL"},

  {"asset\_type": "crypto", "ticker": "BTC"}

],

 "notification\_price\_change": {

   "stock": ["AAPL", "GOOGL"],

   "crypto": ["BTC", "ETH"]

},

 "notification\_price\_level": {

   "stock": {

     "AAPL": {"type": "value", "value": 150.00, "current\_price": 148.50}

  },

   "crypto": {

     "BTC": {"type": "percent", "value": 5, "current\_price": 45000}

  }

}

}

​Пояснение структуры JSON:

* + timezone: Часовой пояс пользователя.
  + base\_currency: Валюта по умолчанию для отображения цен.
  + notification\_frequency: Частота отправки уведомлений.
  + favorites\_assets: Список избранных активов пользователя.
  + notification\_price\_change: Активы, для которых настроены уведомления об изменении цен.
  + notification\_price\_level: Настройки пороговых уведомлений для активов.

​

* 1. **Построение логической ER-диаграммы**

Сущности и связи:

* Пользователи (users): содержат идентификатор, имя пользователя и настройки.
* Связь: Один пользователь может иметь только один набор настроек.

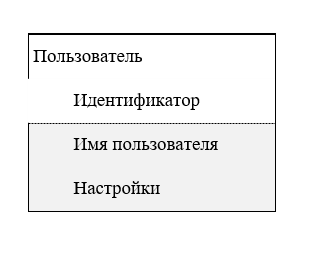


Рисунок 14 – Логическая диаграмма

* 1. **Построение физической ER-диаграммы**

Таблица users:

* Поля:
  + id: Уникальный идентификатор (INTEGER).
  + username: Имя пользователя (VARCHAR).
  + settings: JSON с настройками пользователя.

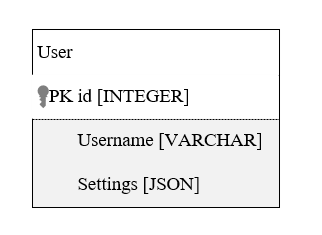


Рисунок 15 – Физическая диаграмма

* 1. **Построение диаграммы потоков данных**

Центральный элемент: Telegram-бот.

Потоки:

* Взаимодействие с пользователем (запросы/ответы).
* Работа с данными (запрос/обновление).
* Настройки уведомлений.
* Обработка данных.

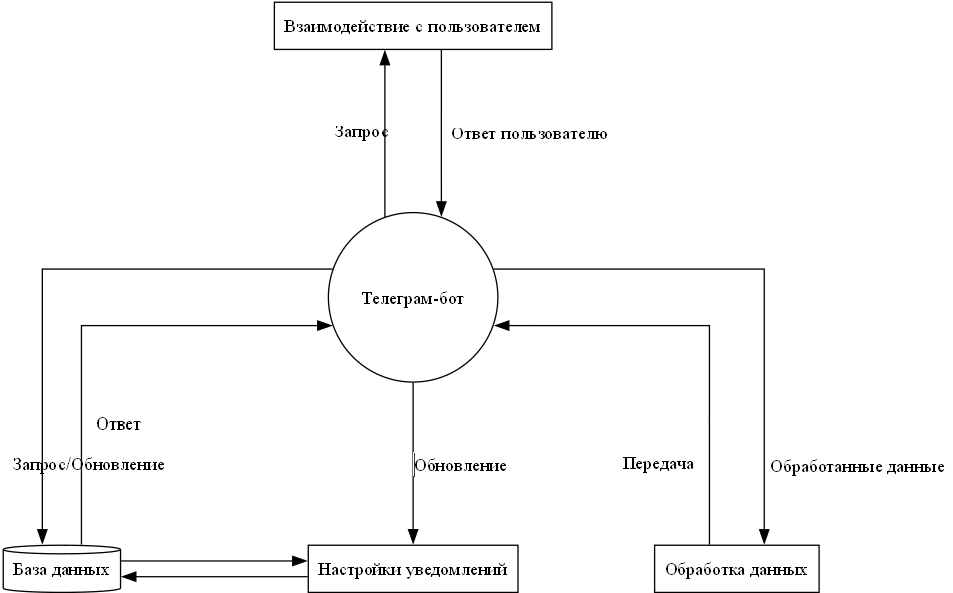
****

Рисунок 16 – Диаграмма потоков данных

# **Бизнес-процессы системы**

На рисунке 14 представлена контекстная диаграмма IDEF0.

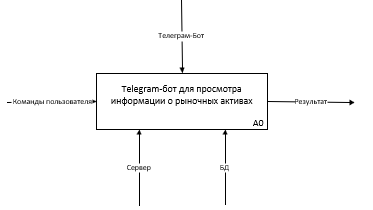


Рисунок 17 – диаграмма IDEF0

Основные бизнес-процессы:

1. Регистрация пользователя и инициализация настроек: при первом взаимодействии с ботом пользователь добавляется в базу данных, и для него создаются базовые настройки.
2. Просмотр информации об активах:
   * + Пользователь выбирает тип актива (акции, валюты, криптовалюты).
     + Бот предоставляет список доступных активов или поиск по ним.
     + При выборе актива отображается текущая цена, исторические данные и другая информация.
3. Управление избранными активами:
   * + Добавление активов в избранное.
     + Удаление активов из избранного.
     + Просмотр списка избранных активов.
4. Настройка уведомлений и оповещений:
   * + Подписка на изменения цен конкретных активов.
     + Установка ценовых уровней для оповещений.
     + Настройка частоты уведомлений.
5. Настройка параметров бота:
   * + Изменение часового пояса.
     + Выбор базовой валюты.
     + Настройка частоты уведомлений.
6. Обработка пользовательского ввода и ошибок:
   * + Валидация вводимых данных.
     + Предоставление пользователю соответствующих сообщений об ошибках или подтверждениях.

На рисунке 15 показан результат декомпозиции деятельности бота.

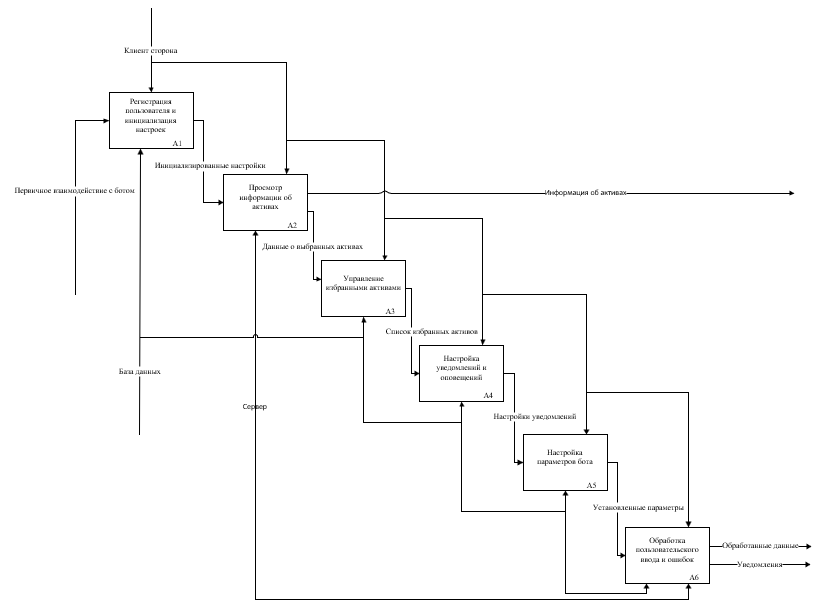


Рисунок 18 – результат декомпозиции деятельности бота для системы Telegram-бота о рыночных активах

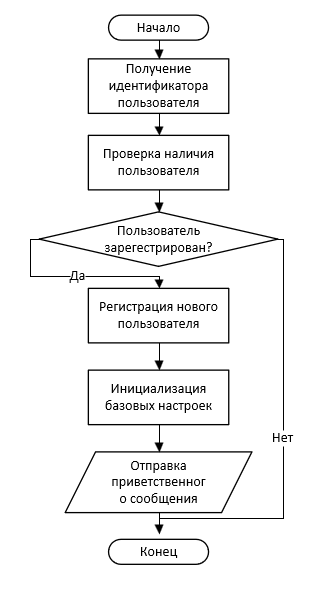
# **Разработка алгоритма решения задачи**

## **5.1. Алгоритм регистрации пользователя и инициализации настроек**

**Описание:** Регистрация нового пользователя при первом взаимодействии с ботом и инициализация его базовых настроек.

**Шаги:**

1. **Получение идентификатора пользователя:**
   * Бот получает уникальный идентификатор пользователя из сообщения.
2. **Проверка наличия пользователя:**
   * Бот проверяет, зарегистрирован ли пользователь в системе.
3. **Создание записи при отсутствии пользователя:**
   * Если пользователь не зарегистрирован:
     + Бот регистрирует нового пользователя.
     + Инициализирует базовые настройки для пользователя.
4. **Отправка приветственного сообщения:**
   * Бот отправляет пользователю сообщение с подтверждением регистрации и инструкциями по дальнейшему использованию.
5. **Завершение процесса регистрации:**
   * Алгоритм завершается.



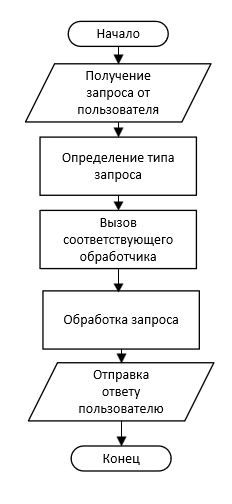
**Рисунок 19 – Алгоритм регистрации пользователя**

## **5.2. Алгоритм взаимодействия с пользователем**

**Описание:** Обработка запросов пользователя, определение типа запроса и вызов соответствующих обработчиков.

**Шаги:**

1. **Получение запроса от пользователя:**
   * Пользователь отправляет команду или нажимает кнопку в Telegram-боте.
2. **Определение типа запроса:**
   * Бот анализирует команду или действие пользователя для определения категории запроса (например, просмотр активов, управление избранным, настройка уведомлений и т.д.).
3. **Вызов соответствующего обработчика:**
   * На основе типа запроса бот выбирает и вызывает соответствующий обработчик.
4. **Обработка запроса:**
   * Обработчик выполняет необходимые действия:
     + Получает необходимые данные из источников.
     + Обновляет настройки пользователя при необходимости.
     + Формирует ответ для пользователя.
5. **Отправка ответа пользователю:**
   * Бот отправляет сформированный ответ через Telegram.
6. **Завершение процесса взаимодействия:**
   * Алгоритм завершается.



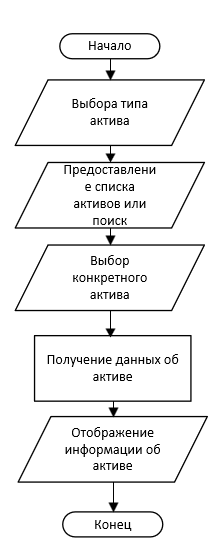
**Рисунок 20 – Алгоритм взаимодействия с пользователем**

## **5.3. Алгоритм просмотра информации об активах**

**Описание:** позволяет пользователю просматривать информацию о различных типах активов, включая текущие цены и исторические данные.

**Шаги:**

1. **Выбор типа актива:**
   * Пользователь выбирает тип актива (акции, валюты, криптовалюты) через интерфейс бота.
2. **Предоставление списка доступных активов:**
   * Бот отображает список доступных активов данного типа или предоставляет возможность поиска по ним.
3. **Выбор конкретного актива:**
   * Пользователь выбирает интересующий его актив из списка.
4. **Получение данных об активе:**
   * Бот получает текущую цену, исторические данные и другую информацию о выбранном активе.
5. **Форматирование и отображение информации:**
   * Отформатированные данные отправляются пользователю.
6. **Завершение процесса просмотра:**
   * Алгоритм завершается.



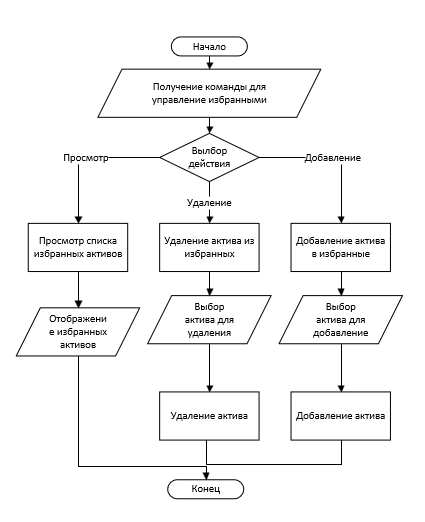
**Рисунок 21 – Алгоритм получения данных о задаче**

## **5.4. Алгоритм управления избранными активами**

**Описание:** позволяет пользователю добавлять или удалять активы из списка избранных и просматривать этот список.

**Шаги:**

1. **Получение команды управления избранным:**
   * Пользователь выбирает опцию управления избранными активами (добавить, удалить, просмотреть).
2. **Добавление актива в избранное:**
   * Если пользователь выбирает добавление:
     + Пользователь выбирает актив из списка.
     + Бот добавляет актив в список избранных для пользователя.
     + Отправляет подтверждающее сообщение пользователю.
3. **Удаление актива из избранного:**
   * Если пользователь выбирает удаление:
     + Пользователь выбирает актив из списка избранных.
     + Бот удаляет актив из списка избранных пользователя.
     + Отправляет подтверждающее сообщение пользователю.
4. **Просмотр списка избранных активов:**
   * Если пользователь выбирает просмотр:
     + Бот извлекает список избранных активов пользователя.
     + Отображает список активов пользователю.
5. **Завершение процесса управления избранными активами:**
   * Алгоритм завершается.



**Рисунок 22 –** Алгоритм управления избранными активами

## **5.5. Алгоритм настройки уведомлений и оповещений**

**Описание:** позволяет пользователю настраивать уведомления о изменениях цен активов, устанавливать ценовые уровни и частоту уведомлений.

**Шаги:**

1. **Получение запроса на настройку уведомлений:**
   * Пользователь выбирает опцию настройки уведомлений в боте.
2. **Определение параметров уведомлений:**
   * Бот запрашивает у пользователя:
     + Тип уведомления (например, изменение цены актива).
     + Значение и условия (например, превышение определенной цены).
     + Частоту уведомлений (например, раз в 10 минут).
3. **Сохранение настроек:**
   * Бот сохраняет параметры уведомлений для пользователя.
4. **Отправка подтверждения пользователю:**
   * Бот отправляет пользователю сообщение с подтверждением сохранения настроек.
5. **Завершение процесса настройки уведомлений:**
   * Алгоритм завершается.



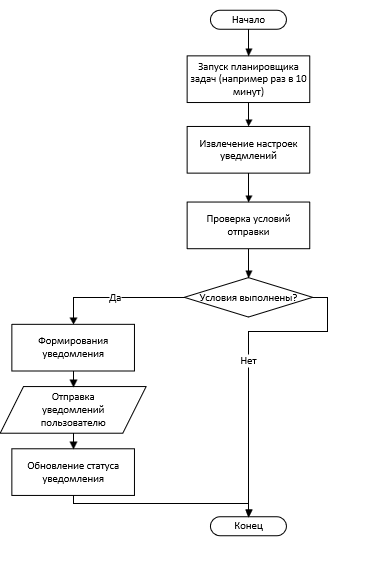
**Рисунок 23 – Алгоритм настройки уведомлений**

## **5.6. Алгоритм отправки уведомлений**

**Описание:** автоматически отправляет пользователю уведомления на основе настроек, хранящихся в системе.

**Шаги:**

1. **Запуск планировщика задач:**
   * Периодически (например, раз в 10 минут) запускается задача отправки уведомлений.
2. **Извлечение настроек уведомлений для каждого пользователя:**
   * Для каждого пользователя бот получает соответствующие настройки уведомлений.
3. **Проверка условий для отправки уведомлений:**
   * Для каждого уведомления проверяются текущие данные активов и соответствие условий.
4. **Формирование и отправка уведомления:**
   * Если условия выполнены:
     + Бот формирует уведомление с необходимой информацией.
     + Отправляет уведомление пользователю через Telegram.
     + Обновляет статус уведомления, если требуется (например, если уведомление одноразовое).
5. **Завершение процесса отправки уведомлений:**
   * Алгоритм завершается.



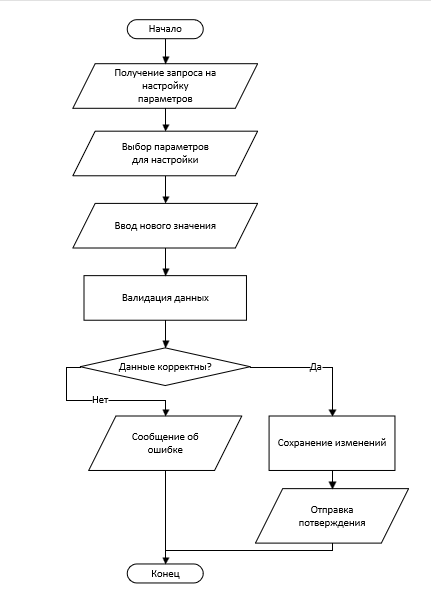
**Рисунок 23 – Алгоритм отправки уведомлений**

## **5.7. Алгоритм настройки параметров бота**

**Описание:** позволяет пользователю изменять параметры работы бота, такие как часовой пояс, базовая валюта и частота уведомлений.

**Шаги:**

1. **Получение запроса на настройку параметров:**
   * Пользователь выбирает опцию настройки параметров бота в интерфейсе.
2. **Выбор параметра для изменения:**
   * Бот предоставляет список доступных параметров (часовой пояс, базовая валюта, частота уведомлений).
3. **Ввод нового значения параметра:**
   * Пользователь вводит или выбирает новое значение для выбранного параметра.
4. **Валидация введенных данных:**
   * Бот проверяет корректность введенного значения (например, существование часового пояса, допустимость валюты).
5. **Сохранение изменений:**
   * Бот обновляет соответствующие настройки для пользователя.
6. **Отправка подтверждения пользователю:**
   * Бот уведомляет пользователя об успешном изменении параметра.
7. **Завершение процесса настройки параметров бота:**
   * Алгоритм завершается.



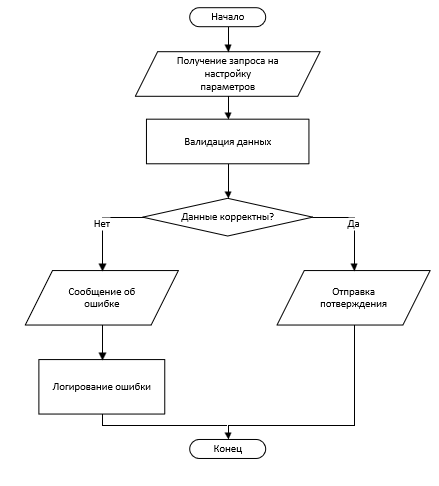
**ы**

## **5.8. Алгоритм обработки пользовательского ввода и ошибок**

**Описание:** обеспечивает валидацию вводимых пользователем данных и предоставляет соответствующие сообщения об ошибках или подтверждениях.

**Шаги:**

1. **Получение пользовательского ввода:**
   * Пользователь вводит команду, выбирает опцию или вводит данные через интерфейс бота.
2. **Валидация вводимых данных:**
   * Проверка корректности формата данных (например, числовые значения для ценовых уровней).
   * Проверка наличия необходимых параметров.
3. **Обработка ошибок:**
   * Если данные некорректны:
     + Бот отправляет пользователю сообщение об ошибке с указанием причины и инструкциями по исправлению.
4. **Подтверждение успешного ввода:**
   * Если данные корректны:
     + Бот подтверждает успешное выполнение действия (например, сохранение настроек).
5. **Логирование ошибок (опционально):**
   * Сохранение информации об ошибках для дальнейшего анализа и улучшения системы.
6. **Завершение процесса обработки ввода:**
   * Алгоритм завершается.



**Рисунок 25 – Алгоритм обработки пользовательского ввода и ошибок**

# **Выбранная архитектура системы**

## **6.1. Общая архитектура**

* Система построена на основе клиент-серверной архитектуры с использованием асинхронных технологий для обеспечения высокой производительности и масштабируемости. Архитектура системы состоит из следующих компонентов:
* Клиентская часть:  
  Интерфейс пользователя реализован через Telegram-бот, который предоставляет доступ к функциональности системы.
* Серверная часть:  
  Серверная логика реализована на Python с использованием FastAPI для REST API и Aiogram для взаимодействия с пользователями через Telegram.
* База данных:  
  Хранит информацию о пользователях, их настройках и избранных активах. Для небольших нагрузок используется SQLite, для более масштабных задач — PostgreSQL.
* Внешние API:  
  Интеграция с внешними сервисами для получения данных об активах (например, Yahoo Finance, Investpy, Tinkoff).

Архитектура приложения представлена на рисунке 10.

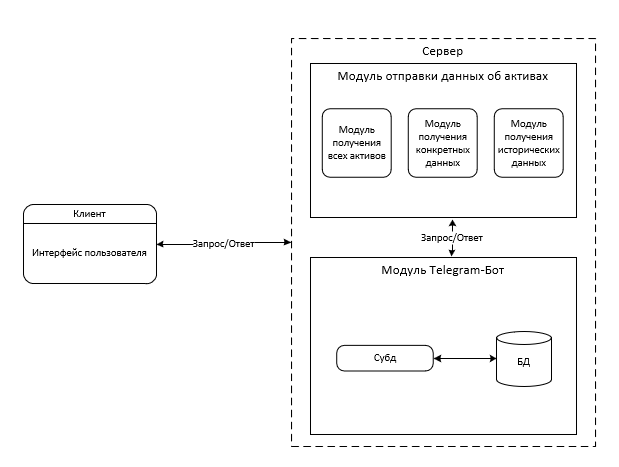


Рисунок 26 – Архитектура приложения

## **6.2. Технологический стек**

* Язык программирования: Python 3.9+.
* Фреймворки и библиотеки:
  + Aiogram: Асинхронный фреймворк для Telegram-ботов.
  + FastAPI: Для реализации REST API.
  + SQLAlchemy Async: Для асинхронного взаимодействия с базой данных.
  + Asyncio: Для реализации асинхронных задач.
  + Httpx: Асинхронный HTTP-клиент для работы с внешними API.
* База данных: SQLite для локального хранения данных или PostgreSQL для более сложных сценариев.
* Среда выполнения: Сервер или облачный сервис с поддержкой Python и необходимых библиотек.

## **6.3. Структура программы**

* Программа имеет четко структурированную организацию, основанную на принципах модульности и разделения обязанностей:
* Корневой каталог:
  + Документация проекта: техническое задание, руководство программиста, аналитическая записка и т.д.
  + Файл requirements.txt для управления зависимостями.
* Модуль app:
  + main.py: Основной файл для запуска серверного приложения.
  + Подкаталоги:
    - src/endpoints: Эндпоинты FastAPI для обработки запросов (получение информации об активах, графиков и т.д.).
    - src/services: Модули для работы с внешними API (Investpy, Yahoo Finance, Tinkoff).
    - src/\_\_init\_\_.py: Инициализация структуры проекта.
* Модуль bots:
  + main.py: Файл для запуска Telegram-бота.
  + Подкаталоги:
    - handlers: Логика обработки команд, состояний и сообщений.
    - keyboards: Генерация клавиатур для взаимодействия с пользователем.
    - data: Скрипты для взаимодействия с базой данных.
    - other: Утилиты для построения графиков, отправки уведомлений и работы с данными.
    - utils: Вспомогательные модули, такие как обработчики ошибок и middlewares.