1. **Введение**
   1. Назначение

“Телеграм-бот” – органайзер ссылок и заметок.

* 1. Обзор системы

Данный проект является телеграм ботом, главной функциональностью является организация и сортировок текстовой информации в полуавтоматичском режиме.

* 1. Значения и акронимы

Пользователь – человек, который использует продукт.

Бот - бот, а также интернет-бот, www-бот и тому подобное — специальная программа, выполняющая автоматически и/или по заданному расписанию какие-либо действия через интерфейсы, предназначенные для людей.

1. **Предполагаемый дизайн**
   1. Предположения

При открытии чата с ботом появляется информация о нем и предложение начать диалог. При отправке сообщения боту, пользователь получит текстовый с прикрепленной к нему клавиатурой с несколькими вариантами действий. В свою очередь, выбор определенного ответа, может создать еще одну клавиатуру с новыми действиями.

* 1. Ограничения

Нельзя создать свою папку.

* 1. Системная среда

Бот работает совместно с “Telegram bot API”, для запуска бота необходимо иметь аккаунт в мессенджере “телеграм”.

* 1. Риски

В случае отказа серверов “telegram” и/или хостинга/облака, где код бота удаленно исполняется, бот перестанет принимать от пользователя запросы.

1. **Архитектура**
   1. Для написания данного проекта используется парадигма функционального программирования. А именно используются высокоуровневые абстракции, которые скрывают большое количество подробностей таких рутинных операций. За счет этого код получается короче, и, как следствие, гарантирует меньшее количество ошибок, которые могут быть допущены.
   2. Стратегии и решения

На архитектуру повлияли следующие решения

и стратегии:

1. В коде используются lambda функции
2. Интерфейс взаимодействуюет с пользователем через функции высшего порядка и декораторы
3. **Дизайн низкого уровня**
   1. Описание  
      Основные компоненты – клавиатуры, main\_keyboard, note\_keyboard, changeType2Keyboard и т.д.

Компоненты взаимодействуют между собой посредством функций высшего порядка.

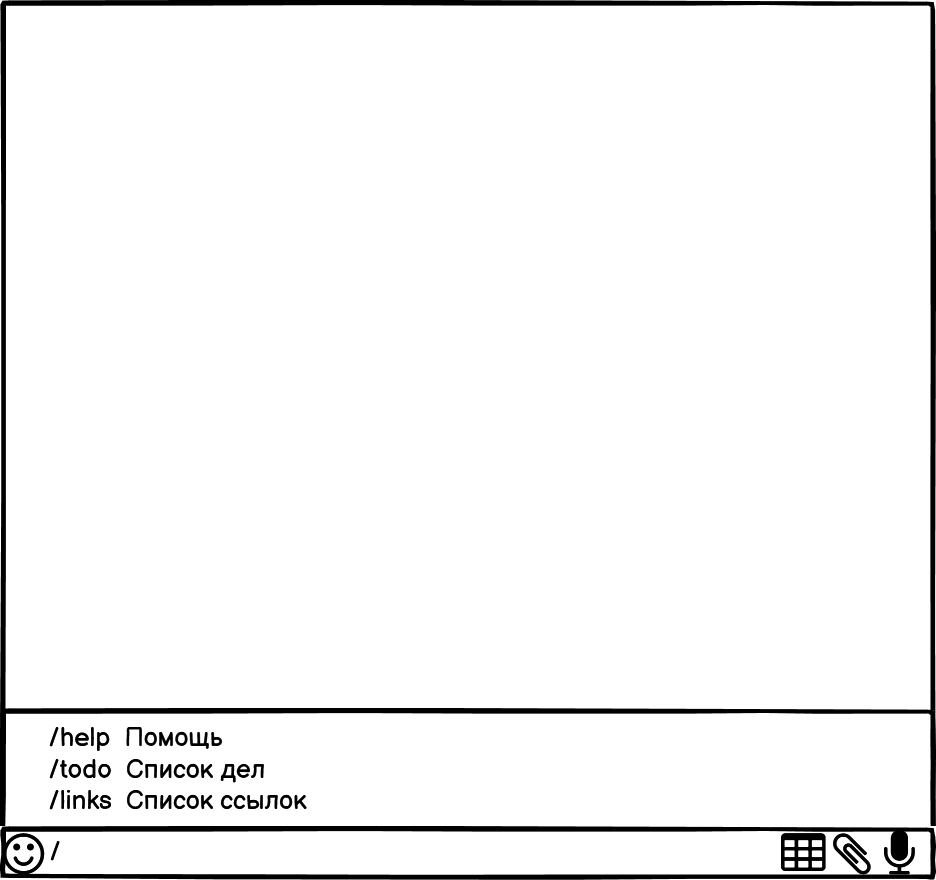
1. **Структура хранения данных**

5.1 Описание

Для хранения данных была выбрана СУБД SQLite3 . СУБД SQLite3 не использует парадигму [клиент-сервер](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B8%D0%B5%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80), то есть [движок](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B4%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%85%D1%80%D0%B0%D0%BD%D0%B5%D0%BD%D0%B8%D1%8F) SQLite не является отдельно работающим процессом, с которым взаимодействует программа, а представляет собой [библиотеку](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), с которой программа компонуется, и движок становится составной частью программы. Таким образом, в качестве протокола обмена используются вызовы функций ([API](https://ru.wikipedia.org/wiki/API)) библиотеки SQLite. Такой подход уменьшает накладные расходы, время отклика и упрощает программу. SQLite хранит всю базу данных (включая определения, таблицы, индексы и данные) в единственном стандартном файле на том компьютере, на котором исполняется программа.

1. **Дизайн высокого уровня**
   1. Варианты диалога с ботом

6.2.1 Вызов команд



6.2.2 Ответы с клавиатурами

