Автор: Шарманжинов Николай Павлович  
  
**Оглавление дипломной работы:**

Оглавление

[1. Введение 1](#_Toc187429881)

[2. Используемые фреймворки. 2](#_Toc187429882)

[3. Описание используемых фреймворков 2](#_Toc187429883)

[4. Преимущества и недостатки применяемых фреймворков 3](#_Toc187429884)

[5. Реализация 5](#_Toc187429885)

[6. Описание приложения 5](#_Toc187429886)

[7. Заключение 7](#_Toc187429887)

# **Введение**

**Данное Веб-приложение предназначено для получения информации по ИНН юридического лица о наличии в базах, свободно доступных в Интернет, без необходимости захода на их сайты и поиска, что экономит трудозатраты, позволяет получить все в одном окне. Поиск может осуществляться как непосредственно через веб-форму, так и через телеграмм бота. Мониторинг осуществляется по следующим базам:**

**- Государственный реестр ломбардов;**

**- Жилищные накопительные кооперативы;**

**- Государственный реестр сельскохозяйственных кредитных потребительских кооперативов;**

**- Государственный реестр микрофинансовых организаций;**

**- Государственный реестр кредитных потребительских кооперативов;**

**- Перечень микрофинансовых организаций;  
- ПАО «СПБ Биржа»;**

**- Единая информационная система в сфере закупок;**

**- ЕГРЮЛ.**

# **Используемые фреймворки.**

Проект на Django - <https://github.com/except134/Python/tree/master/Diplom/MonitoringULDjango>

Проект на FastAPI - <https://github.com/except134/Python/tree/master/Diplom/MonitoringULFastAPI>

Проект на Flask - <https://github.com/except134/Python/tree/master/Diplom/MonitoringULFlask>

Проект на Bottle - <https://github.com/except134/Python/tree/master/Diplom/MonitoringULBottle>

# **Описание используемых фреймворков**

**Django** - это свободный фреймворк, используемый для разработки веб-приложений. Работает на языке Python. Django использует для выполнения необходимого спектра задач шаблон проектирования MVC.

Это бесплатная библиотека. Она позволяет создавать быстрые и безопасные сайты/онлайн-программы на языке программирования Python. Этот инструмент был создан в 2005 году разработчиками Андрианом Головати и Саймоном Уиллисоном. Библиотека была названа в честь французского гитариста Джанго Рейнхарда.

Первая версия рассматриваемого инструмента официально была опубликована в 2005 году. С тех пор Django активно развивается и совершенствуется. Ежегодно обновление фреймворка осуществляется 1-2 раза. С 2008 года поддержкой этого проекта занимается компания Django Software Foundation (DSF).

**FastAPI** - это современный и высокопроизводительный веб-фреймворк для создания API на языке программирования Python, который был запущен в 2018 году. Он разрабатывался как простой и удобный инструмент для ускорения разработки и уменьшения количества шаблонного кода.

**Flask** - фреймворк для создания веб-приложений на языке программирования Python, использующий набор инструментов Werkzeug, а также шаблонизатор Jinja2. Относится к категории так называемых микрофреймворков — минималистичных каркасов веб-приложений, сознательно предоставляющих лишь самые базовые возможности. Создатель и основной автор — австрийский программист Армин Ронахер, начал работу над проектом в 2010 году.

**Bottle** - это мини-фреймворк для Python, позволяющий писать веб-приложения с высокой скоростью. Вот только слово «мини» добавляет ограничения, например, здесь нет быстрого способа создать административную панель. Если нужна работа с БД, то ее надо подключать отдельно.

# **Преимущества и недостатки применяемых фреймворков**

**Django**

**Преимущества:**

- позволяет создавать красивые и интерактивные страницы, которые пользователи могут видеть в своих браузерах;

- облегчает работу с базой данных, где хранится информация о веб-сайте, такая как список товаров или информация о пользователях;

- помогает обрабатывать запросы от пользователей и отправлять им нужную информацию;

-  имеет мощный административный интерфейс, который позволяет администраторам управлять содержимым веб-сайта без необходимости знать программирование.

**Недостатки:**

- имеет свой собственный синтаксис и структуру, которые могут потребовать времени для изучения;

- проект часто превращается в странную субстанцию из папок в папках.

**FastAPI**

**Преимущества:**

- скорость работы;

- нет какой-либо утвержденной архитектуры, что дает волю вашей фантазии и различным подходам разработки;

- использует ASGI-сервера по умолчанию, когда в том же Django нужно заниматься конфигурацией приложения с WSGI на ASGI, что занимает достаточно времени;

- валидация данных через Pydantic.

**Недостатки:**

- дался сложнее в изучении, чем Django, Flask и Bottle;

- нет встроенного ORM;

- требует от разработчиков большей ответственности за безопасность, поскольку он не предоставляет встроенной системы аутентификации и авторизации.

- необходимость сторонних библиотек.

**Flask**

**Преимущества:**

- легкий в изучении;

- предоставляет минимальный набор инструментов и позволяет разработчикам свободно расширять функциональность с помощью сторонних библиотек. Это дает большую свободу в проектировании и реализации приложений;

- имеет простой и понятный API, который легко изучить и использовать.

**Недостатки:**

- нет встроенного ORM;

- требует от разработчиков большей ответственности за безопасность, поскольку он не предоставляет встроенной системы аутентификации и авторизации;

- необходимость сторонних библиотек.

**Bottle**

**Преимущества:**

- легкий в изучении (дался легче всех в изучении);

- предоставляет минимальный набор инструментов и позволяет разработчикам свободно расширять функциональность с помощью сторонних библиотек. Это дает большую свободу в проектировании и реализации приложений;

- имеет простой и понятный API, который легко изучить и использовать;

- минимальные зависимости, не требует установки дополнительных библиотек или модулей, что делает его легко устанавливаемым и использованием;

- благодаря своей легковесности, обладает высокой производительностью, что делает его хорошим выбором для создания быстрых и отзывчивых веб-приложений.

**Недостатки:**

- нет встроенного ORM;

- требует от разработчиков большей ответственности за безопасность, поскольку он не предоставляет встроенной системы аутентификации и авторизации;

- необходимость сторонних библиотек.

# **Реализация**

Bottle - получилось реализовать проект в полном объёме. (1 место по скорости реализации).

Flask - получилось реализовать проект в полном объёме. (2 место по скорости реализации).

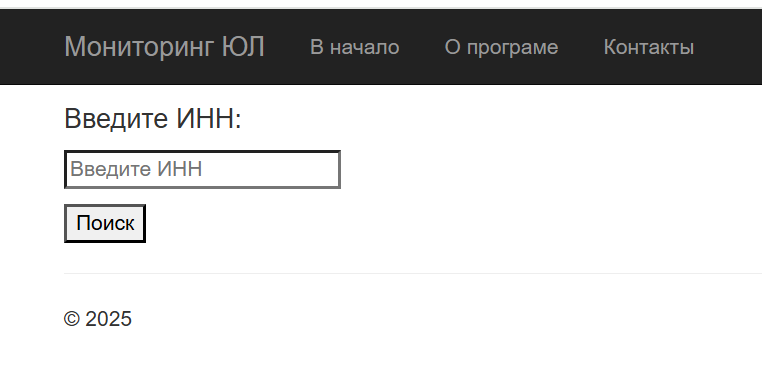
Django - получилось реализовать проект в полном объёме. (3 место по скорости реализации).

FastAPI - получилось реализовать проект в полном объёме. (4 место по скорости реализации).

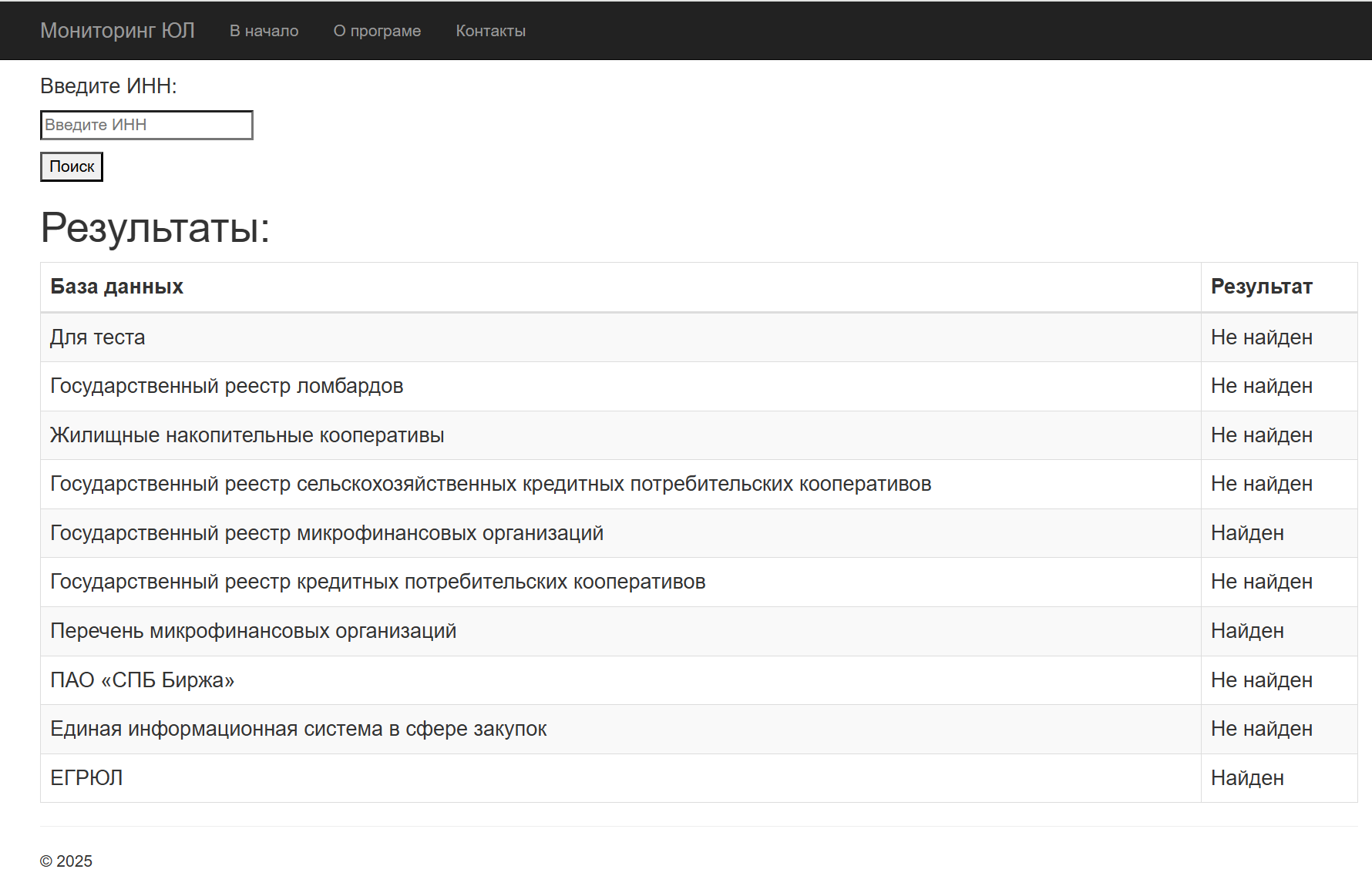
# **Описание приложения**

**Получение данных через Веб-интерфейс.**

Основной целью было добиться абсолютно одинакового внешнего вида приложения с одинаковым функционалом с возможностью получать данные также через телеграмм. Также целью было создать максимально простой, минималистичный интерфейс, не перегруженный лишними элементами.

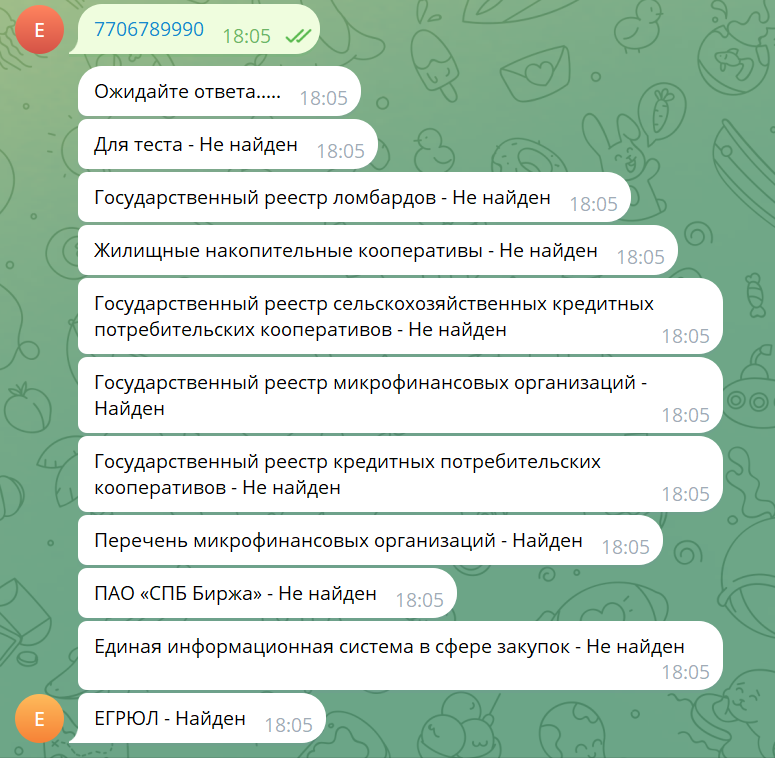


При вводе ИНН, введенный номер передается на сторону сервера, который начинает работу по сбору данных с ресурсов. Данное действие требует ожидания несколько секунд. После чего появляется результат:



**Получение данных через Телеграмм.**

Ссылка на Телеграмм бота: <https://t.me/ExceptTestBot>

Получении данных осуществляется простым написанием боту ИНН, после чего необходимо подождать несколько секунд результат:

# **Заключение**

**Разработав четыре приложения, могу сделать следующие выводы:**

**- для небольших приложений мой выбор однозначно упадет на Bottle или Flask;**

**- для больших комплексных проектов – Django;**

**- FastAPI по собственной воле скорее всего использовать не буду. Только если того будет требовать работодатель.**