**Вступление**

Данная исследовательская работа посвящена безопасности Django framework.

Django – это большой веб-фраеморк написанный на Python, реализующий паттерн MVC, в который встроено все необходимое для постороения большинства веб-приложений. Django позволяет удешевить разработку за счет встроенных механизмов работы с базой данных, рендеринга HTML-шаблонов, роутинга, а также большого количества библиотек, как специфичных для Django, так и из богатой экосистемы Python.

**Цели**

Целью данной работы является исследование Django framework с точки зрения защиты инфромации. Приступая к данной работе я заранее понимаю, что найти реальные уязвимости является достаточно сложной и практически нереальной задачей. Причиной тому является зрелость данной технологии: django существует с 2005-го года, обладает большим сообществом разработчиков, используется компаниями IT-гигантами. С учетом данного факта я ставлю перед собой следующие цели:

* Исследовать структуру Django: выделить основные компоненты и опеределить их зону отвественности.
* Исследовать механизмы авторизации, аутентификации и идентификация спецефичные для веб-приложений, реализовать их на практике с помощью Django и на практике объяснить их достоинства и недостатки.
* Исследовать встроенные механизмы защиты и предпринять попытки обойти их, как в стандартных ситуациях, так и в ситуациях, когда разработчиком допущены ошибки, создающие уязвимости.

Более глобальной целью является выделение и обоснование знаний, которые помогут Django-разрабочикам создавать безопасные веб-приложения.

**Структура работы**

Данная работа будет разбита на 3 главы, соответственно отвечающие за каждую сформулированную цель.

Глава 1. Изучение структуры Django. Как framework работает, из каких слоев состоит, какие задачи решает каждый слой.

Глава 2. Практическая реализация различных способов авторизации и аутентификации (base authorization, session authorization, token authrization, oauth 1.0, oauth 2.0) и демонстрация на практике достоинств и недостатков каждого способа.

Глава 3. Изучение встроенных механизмов защиты от распостраненных видов атак: csrf, clickjacing, ddos, sql инъекции, подбор пароля, намеренное допущение ошибок, которые позволят данные атаки провести успешно и релизация защиты от них.

Дополнительно в главы будут встраиваться кейсы, которые будут нумероваться отдельно. Кейс – это конфигурция Django-проекта и последовательность действий, которые позволят либо получить неправомерный доступ к ресурсу поставляемому приложением, либо помешать правомерному доступу.