

Министерство науки и высшего образования Российской Федерации  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ  
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО  
ITMO University

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА  
GRADUATION THESIS

Разработка онлайн-системы для управления отчетами по выявленным уязвимостям в программном обеспечении

Обучающийся / Student Севастьянов Юрий Алексеевич

Факультет/институт/кластер/ Faculty/Institute/Cluster факультет информационных технологий и программирования

Группа/Group М34061

Направление подготовки/ Subject area 09.03.02 Информационные системы и технологии

Образовательная программа / Educational program Программирование и интернет-технологии 2019

Язык реализации ОП / Language of the educational program Русский

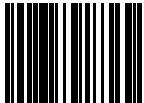
Статус ОП / Status of educational program

Квалификация/ Degree level Бакалавр

Руководитель ВКР/ Thesis supervisor Дядюшкин Александр Александрович, Университет ИТМО, факультет информационных технологий и программирования, программист

Консультант не из ИТМО / Third-party consultant Короткая Елена Васильевна, ООО «СТЦ», Координатор

Обучающийся/Student

Документ подписан	
Севастьянов Юрий Алексеевич	
16.05.2023	

(эл. подпись/ signature)

Севастьянов  
Юрий  
Алексеевич

(Фамилия И.О./ name  
and surname)

Руководитель ВКР/  
Thesis supervisor

Документ подписан	
Дядюшкин Александр Александрович	
16.05.2023	

(эл. подпись/ signature)

Дядюшкин  
Александр  
Александрович

(Фамилия И.О./ name  
and surname)

**Министерство науки и высшего образования Российской Федерации  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ  
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО  
ITMO University**

**АННОТАЦИЯ  
ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ  
SUMMARY OF A GRADUATION THESIS**

**Обучающийся / Student** Севастьянов Юрий Алексеевич  
**Факультет/институт/кластер/ Faculty/Institute/Cluster** факультет информационных технологий и программирования  
**Группа/Group** М34061  
**Направление подготовки/ Subject area** 09.03.02 Информационные системы и технологии  
**Образовательная программа / Educational program** Программирование и интернет-технологии 2019  
**Язык реализации ОП / Language of the educational program** Русский  
**Статус ОП / Status of educational program**  
**Квалификация/ Degree level** Бакалавр  
**Тема ВКР/ Thesis topic** Разработка онлайн-системы для управления отчетами по выявленным уязвимостям в программном обеспечении  
**Руководитель ВКР/ Thesis supervisor** Дядюшкин Александр Александрович, Университет ИТМО, факультет информационных технологий и программирования, программист  
**Консультант не из ИТМО / Third-party consultant** Короткая Елена Васильевна, ООО «СТЦ», Координатор

**ХАРАКТЕРИСТИКА ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ  
DESCRIPTION OF THE GRADUATION THESIS**

**Цель исследования / Research goal**

Разработать онлайн-систему (веб-сайт) для управления отчетами по выявленным уязвимостям в программном обеспечении.

**Задачи, решаемые в ВКР / Research tasks**

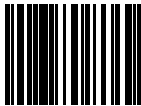
Анализ предметной области, функциональных и нефункциональных требований. Составление списка используемых технологий на основе проведенного анализа. Проектирование системной архитектуры, архитектуры данных и программной архитектуры. Разработатка и тестирование решения.

**Краткая характеристика полученных результатов / Short summary of results/findings**

Спроектировано и реализовано веб-приложение, позволяющее управлять отчетами по выявленным уязвимостям авторизованными пользователями. Сформированы необходимые требования и список технологий. Спроектирована системная архитектура, архитектура данных и программная архитектура. Описано реализованное веб-приложение.

Обучающийся/Student

Документ подписан	
----------------------	--

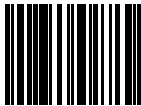
	
Севастьянов Юрий Алексеевич	
16.05.2023	

(эл. подпись/ signature)

Севастьянов  
Юрий  
Алексеевич

(Фамилия И.О./ name  
and surname)

Руководитель ВКР/  
Thesis supervisor

Документ подписан	
Дядюшкин Александр Александрович	
16.05.2023	

(эл. подпись/ signature)

Дядюшкин  
Александр  
Александрович

(Фамилия И.О./ name  
and surname)

**Министерство науки и высшего образования Российской Федерации  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ  
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО  
ITMO University**

**ЗАДАНИЕ НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ /  
OBJECTIVES FOR A GRADUATION THESIS**

**Обучающийся / Student** Севастьянов Юрий Алексеевич  
**Факультет/институт/кластер/ Faculty/Institute/Cluster** факультет информационных технологий и программирования  
**Группа/Group** М34061  
**Направление подготовки/ Subject area** 09.03.02 Информационные системы и технологии  
**Образовательная программа / Educational program** Программирование и интернет-технологии 2019  
**Язык реализации ОП / Language of the educational program** Русский  
**Статус ОП / Status of educational program**  
**Квалификация/ Degree level** Бакалавр  
**Тема ВКР/ Thesis topic** Разработка онлайн-системы для управления отчетами по выявленным уязвимостям в программном обеспечении  
**Руководитель ВКР/ Thesis supervisor** Дядюшкин Александр Александрович, Университет ИТМО, факультет информационных технологий и программирования, программист  
**Консультант не из ИТМО / Third-party consultant** Короткая Елена Васильевна, ООО «СТЦ», Координатор

**Основные вопросы, подлежащие разработке / Key issues to be analyzed**

Выполнить анализ предметной области, функциональных и нефункциональных требований. На основе выполненного анализа составить список используемых технологий. Выполнить проектирование системной архитектуры, архитектуры данных и программной архитектуры. Разработать и протестировать веб-приложение.

**Форма представления материалов ВКР / Format(s) of thesis materials:**

Диаграммы вариантов использования, развертывания, классов, последовательности и ER-диаграмма в нотации UML. Скриншоты интерфейса веб-приложения.

**Дата выдачи задания / Assignment issued on:** 12.02.2023

**Срок представления готовой ВКР / Deadline for final edition of the thesis** 23.05.2023

**Характеристика темы ВКР / Description of thesis subject (topic)**

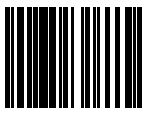
**Название организации-партнера / Name of partner organization:** ООО "СТЦ"

**Тема в области фундаментальных исследований / Subject of fundamental research:** нет / not

**Тема в области прикладных исследований / Subject of applied research:** да / yes

**СОГЛАСОВАНО / AGREED:**

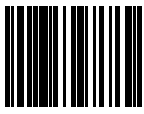
Руководитель ВКР/  
Thesis supervisor

Документ подписан	
Дядюшкин Александр Александрович	
16.05.2023	

(эл. подпись)

Дядюшкин  
Александр  
Александрович

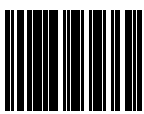
Задание принял к  
исполнению/ Objectives  
assumed BY

Документ подписан	
Севастьянов Юрий Алексеевич	
16.05.2023	

(эл. подпись)

Севастьянов  
Юрий  
Алексеевич

Руководитель ОП/ Head  
of educational program

Документ подписан	
Маятин Александр Владимирович	
19.05.2023	

(эл. подпись)

Маятин  
Александр  
Владимирович

## СОДЕРЖАНИЕ

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ.....	6
ВВЕДЕНИЕ.....	10
ГЛАВА 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ .....	13
1.1 Описание бизнес-процесса .....	13
1.2 Анализ требований .....	16
1.2.1 Функциональные требования .....	16
1.2.2 Нефункциональные требования .....	19
ГЛАВА 2. ПРОЕКТИРОВАНИЕ.....	21
2.1 Использование системы .....	21
2.2 Архитектура данных.....	22
2.2.1 ORM классы.....	22
2.2.2 База данных.....	23
2.3 Выбор технологий.....	26
2.4 Программная архитектура .....	27
2.4.1 Приложение пользователей .....	29
2.4.2 Приложение отчетов.....	35
2.4.3 Приложение ПО .....	42
2.4.4 Приложение сред выполнения.....	45
2.4.5 Основное приложение .....	45
2.5 Системная архитектура .....	46
ГЛАВА 3. РЕАЛИЗАЦИЯ И ТЕСТИРОВАНИЕ.....	50
3.1 Приложение пользователей .....	50
3.2 Приложения ПО и сред выполнения .....	54

3.3 Приложение отчетов.....	60
ЗАКЛЮЧЕНИЕ .....	64
СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ.....	65

## ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

Django – свободный фреймворк для веб-приложений на языке Python, использующий шаблон проектирования MVC. [5]

PostgreSQL – свободная объектно-реляционная система управления базами данных (СУБД). [5]

Python – высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ. [5]

MVC – Model-View-Controller, схема разделения данных приложения и управляющей логики на три отдельных компонента: модель, представление и контроллер. [5]

MVT – Model-View-Template, модификация MVC. [7]

CRUD - акроним, обозначающий четыре базовые функции, используемые при работе с базами данных: создание (англ. create), чтение (read), модификация (update), удаление (delete). [5]

БД – база данных

СУБД – система управления базами данных

ИБ – информационная безопасность

ИТ – информационные технологии.

ПО – программное обеспечение.

СВ – среда выполнения.

Кибератака – это попытки получить несанкционированный доступ к компьютерным системам и украсть, изменить или уничтожить данные. [6]



Программная закладка - скрытно внедрённая в защищенную систему программа, либо намеренно изменённый фрагмент программы, который позволяет злоумышленнику осуществить несанкционированный доступ к ресурсам системы на основе изменения свойств системы защиты. [5]

CVE – база данных общеизвестных уязвимостей информационной безопасности. [5]

CWE – общий перечень дефектов (недостатков) безопасности. [8]

Хэширование – преобразование, производимое хеш-функцией. [5]

BSD – синоним BSD-UNIX — общего названия вариантов UNIX, восходящих к дистрибутивам университета Беркли. [5]

ОС – операционная система

Docker-контейнер – это исполняемый экземпляр образа (image). [9]

Nginx – веб-сервер и почтовый прокси-сервер, работающий на Unix-подобных операционных системах (тестировалась сборка и работа на FreeBSD, OpenBSD, Linux, Solaris, macOS, AIX и HP-UX). [5]

Gunicorn – это WSGI-сервер, созданный для использования в UNIX-системах. [10]

HTTP – протокол прикладного уровня передачи данных, изначально — в виде гипертекстовых документов в формате HTML, в настоящее время используется для передачи произвольных данных. [5]

TCP/IP – сетевая модель передачи данных, представленных в цифровом виде. [5]

WSGI – стандарт взаимодействия между Python-программой, выполняющейся на стороне сервера, и самим веб-сервером, например Apache. [5]

Bootstrap – свободный набор инструментов для создания сайтов и веб-приложений. [5]

Файл входных данных (входной файл) – файл, предназначенный для обработки программой во время ее выполнения.

Файл дампа памяти (дамп-файл) – это моментальный снимок, показывающий выполнявшийся процесс и загруженные для приложения модули в определенный момент времени. [11]

ССЭ МСЭ – сектор стандартизации электросвязи Международного союза электросвязи.

Е.164 – рекомендация ССЭ МСЭ, определяющая общий международный телекоммуникационный план нумерации, используемый в телефонных сетях общего пользования и некоторых других сетях. [5]

Класс представления – программные классы, являющиеся представителями компоненты View в MVT

Шаблонизатор – программное обеспечение, позволяющее использовать шаблоны для генерации конечных документов с помощью декларативного языка разметки. [5]

DTL – Django Template Language

Миксин – это класс, предоставляющий реализации методов для повторного использования дочерними классами. [12]

URL – система унифицированных адресов электронных ресурсов, или единообразный определитель местонахождения ресурса (файла). [5]

JavaScript – мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. [5]

CSS – формальный язык декорирования и описания внешнего вида документа (веб-страницы), написанного с использованием языка разметки (чаще всего HTML или XHTML). [5]

POST, GET – методы запроса, в целях поддерживаемых HTTP протоколом, используемым во Всемирной паутине. [5]

Psycopg – это адаптер базы данных PostgreSQL для Python, то есть одной из его основных задач является автоматическая подстройка типов при составлении запросов и получении ответов между PostgreSQL и Python. [13]

## ВВЕДЕНИЕ

Информационная безопасность играет важную роль в современной сфере ИТ. Почти любой современный человек или организация вынуждены использовать те или иные средства ИТ отрасли. Сложно представить себе современную коммерческую компанию, у которой бы не было хотя бы своего сайта в сети Интернет.

При этом такие последствия кибератак, как утечки данных клиентов, нарушения в работе сервисов и похищение коммерческой тайны могут привести к серьезным негативным последствиям для компаний и организаций, таким как значительные финансовые убытки и ущерб репутации. Так, например, согласно информационному агентству РБК за 2017 год потери от кибератак в России превышали 100 миллиардов рублей.

Согласно данным российской компании Positive Technologies, специализирующейся на разработке решений в сфере информационной безопасности [5], за 2022 год общее количество кибератак в Российской Федерации увеличилось на 21% по сравнению с прошлым годом. В это же время эксперты Positive Technologies ожидают, что в 2023 году число атак, направленных на ИТ-компании, продолжит расти. Также по данным другой международной компании в сфере информационной безопасности Group-IB в 2022 году количество кибератак финансово-мотивированных хакеров увеличилось почти в три раза по сравнению с прошлым годом. [1]

Компания «Ростелеком-Солар» (российский провайдер сервисов и технологий для защиты информационных активов, целевого мониторинга и управления информационной безопасностью, входит в кластер информационной безопасности группы ПАО «Ростелеком» [5]), провела исследование в области информационной безопасности, которое охватывает 280 организаций из разных отраслей, включая госсектор, финансы, энергетику, телеком, медиа, крупный ритейл и не учитывает массовые DDOS-

атаки. Согласно данному исследованию категория «Эксплуатация уязвимостей» заняла 3-е место в рейтинге ИБ-инцидентов по категориям (в процентах) и составляла 8% от общего числа ИБ-инцидентов. [1]

Таким образом за последние годы количество кибератак в Российской Федерации только увеличилось, а эксперты прогнозируют продолжение этого роста и в этом году. В это же время категория «Эксплуатация уязвимостей» занимает 3-е место в рейтинге и является весомой составляющей общего числа кибератак. Поэтому разработка средств, помогающих в исправлении уязвимостей в ПО, крайне важна в современном мире.

В данный момент я являюсь разработчиком программного обеспечения в одном из отделов компании ООО «Специальный Технологический Центр», занимающимся исследованием безопасности программного обеспечения по заказу сторонних организаций. К сожалению, процесс оптимизации рабочего процесса в нашей организации находится на низком уровне и обмен отчетами по найденным уязвимостям происходит в основном в бумажной форме. В свою очередь это мешает оперативному обмену информацией между сотрудниками и может привести к снижению производительности труда и как следствие снижению возможной прибыли компании. В силу всего выше перечисленного наша компания остро нуждается в средствах увеличения производительности сотрудников посредством предоставления им возможности оперативно получать информацию об отчетах по выявленным уязвимостям.

Таковым средством является веб-сайт, предоставляющий сотрудникам компании возможность публиковать и просматривать отчеты, а также редактировать и удалять их при необходимости. Данный веб-сайт позволит сотрудникам получать информацию более своевременно, а также обмениваться ею в более удобном и стандартизированном виде. Данные преимущества позволят увеличить производительность сотрудников и как следствие увеличить прибыль компании.

Таким образом цель данной работы заключается в написании соответствующей системы, представляющей собой данный веб-сайт. Для выполнения данной цели необходимо решить следующие задачи:

1. Анализ предметной области
2. Проектирование системы
3. Реализация спроектированной системы

## **ГЛАВА 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ**

В данной главе будет представлено приблизительное описание бизнес-процесса выявления уязвимостей в программном обеспечении по заказу сторонней организации в нашем отделе. Также в данной главе будет сделан анализ функциональных и нефункциональных требований для разрабатываемой системы.

### **1.1 Описание бизнес-процесса**

При желании воспользоваться нашими услугами заказчик оставляет заявку в бухгалтерии, после чего получает реквизиты оплаты и оплачивает услугу. После оплаты бухгалтерия направляет заявку заказчика в наш отдел, где заявка уже принимается в работу.

Приблизительный план работы над заявкой в нашем отделе состоит из следующих этапов:

1. Анализ требований
2. Составление плана работ
3. Исследование безопасности в соответствии с планом
4. Анализ результатов исследования
5. Отправка результатов анализа заказчику

На этапе исследования безопасности сотрудникам раздается персональный список задач по исследованию который они выполняют. При нахождении сотрудником уязвимости отчет о ней составляется в формате документа (расширение файла .docx или .doc), печатается и передается старшему сотруднику.

В отчете о найденной уязвимости содержится ключевая информация о ней. Примерный список пунктов, содержащихся в данном отчете:

1. Краткое название уязвимости (в котором заключается суть найденной ошибки)
2. Код CWE
3. Описание уязвимости (какие системы затрагивает, какой ущерб системе наносит и другая информация)
4. Какие среды выполнения кода были затронуты (для каких сред удалось экспериментально проверить наличие уязвимости)
5. Оценка опасности (субъективная оценка, насколько большую угрозу несет в себе выявленная уязвимость)
6. Оценка эксплуатабельности (субъективная оценка, насколько сложно воспроизвести найденную ошибку)
7. Способ эксплуатации (каким образом была найдена уязвимость, какой существует способ ее воспроизвести)

По мере поступления отчетов старший сотрудник производит их анализ. После сбора всех присланных отчетов старший сотрудник составляет финальный отчет в формате документа (расширение файла .docx или .doc) и отправляет его заказчику в качестве результатов произведенной работы.

Данный бизнес-процесс проиллюстрирован на рисунке 1.



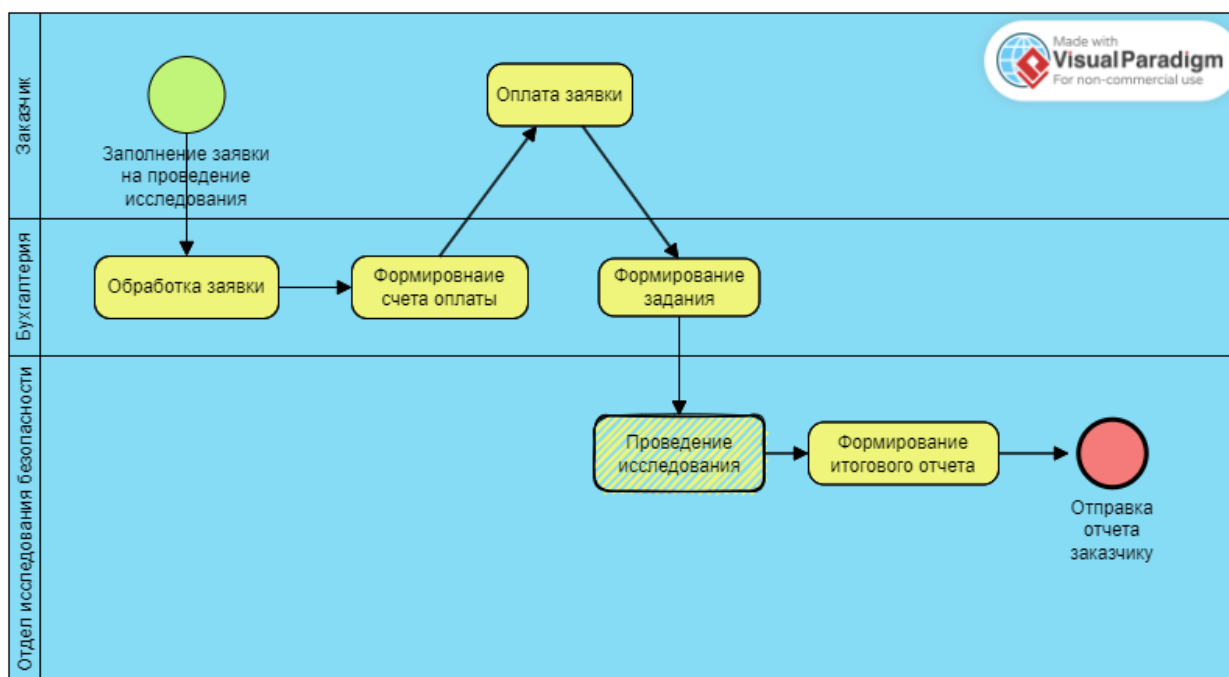


Рисунок 1 – Диаграмма глобального бизнес-процесса

Синим заштрихован отдельный локальный бизнес-процесс, который необходимо улучшить. Данный бизнес-процесс обладает рядом недостатком касающихся производительности труда.

1. Может пройти много времени с момента нахождения уязвимости до получения отчета о ней старшим сотрудником, это время будет потеряно (хотя могло бы быть использовано для ее анализа).
2. Сотрудники не имеют возможности оперативно узнать, какой задачей в данный момент занимаются его коллеги. Может произойти ситуация что два исследователя потратили время и нашли одну и ту же уязвимость через значительное время после друг друга. Разрабатываемая система способна минимизировать такие случаи.
3. Сотруднику может потребоваться помощь в исследовании идей не относящихся к его задаче, однако оперативно сообщить об этом он не может. В таком случае сотрудник мог бы опубликовать отчет в

системе с просьбой о помощи или указанием дальнейших перспективных направлений для исследования.

## **1.2 Анализ требований**

### **1.2.1 Функциональные требования**

Исходя из описанного процесса можно выделить главную сущность системы и ее параметры. Таковой сущностью является отчет о найденной уязвимости. Кратко параметры данной сущности уже были перечислены в разделе «Описание бизнес-процесса»:

1. Название выявленной уязвимости
2. Код CWE
3. Описание уязвимости
4. Список сред выполнения
5. Оценка опасности (целое число от 1 до 10)
6. Оценка эксплуатируемости (целое число от 1 до 10)
7. Способ эксплуатации

К данным пунктам необходимо также добавить программное обеспечение, к которому относится уязвимость, а также пользователя, который является автором отчета. В итоге всего 9 пунктов.

Далее необходимо описать сущности сред выполнения и программного обеспечения.

Сущность среды выполнения представляет собой операционную систему (ОС), на которой была установлена и запущенная программа. Данная сущность должна состоять из:

1. Типа ОС. На данный момент в мире можно выделить 6 основных видов ОС, это: Microsoft Windows, Apple MacOS, Apple iOS, Android, GNU/Linux, BSD.
2. Версии ОС.

Сущность ПО представляет собой программное обеспечение, в котором была найдена уязвимость, для которой составляется отчет. Данная сущность должна содержать в себе следующие характеристики:

1. Категорию. Наш отдел принял решение описать 4 типа ПО:
  - Консольное – запускается, работает и управляется в консоли ОС.
  - Графическое – графическое приложение, для своей работы использует графические элементы.
  - Библиотека. Подразумевается стандартное определение библиотеки для прикладного ПО и операционных систем.
  - Приложение. Рекомендуется использовать в случае если ничего из выше перечисленного не подходит.
2. Название ПО. Необходимо для идентификации разного ПО.
3. Версия ПО. Необходима для идентификации разных версий одного ПО.
4. Опции сборки. В случае если ПО собиралось локально из исходного кода необходимо указать специальные опции, которые были задействованы.

Для поддержки актуальности данных данные три сущности должны иметь реализованный функционал их изменения и удаления со стороны их создателей и администраторов сайта. Любой сотрудник должен иметь возможность просматривать все экземпляры каждой из трех сущностей.

Поскольку исследование нашим отделом безопасности стороннего ПО является коммерческой тайной, то данную систему следует сделать закрытой для использования пользователями, не являющимися сотрудниками отдела.

Данное требование возможно реализовать с помощью системы регистрации и аутентификации пользователей. Данная система должна позволять администратору регистрировать новых пользователей, после чего предоставлять им доступ к системе. У не аутентифицированных пользователей не должно быть возможности пользоваться системой.

Сущность пользователя должна состоять из следующих полей:

1. Фамилия (должна быть возможность написания на кириллице)
2. Имя (должна быть возможность написания на кириллице)
3. Отчество (должна быть возможность написания на кириллице)
4. Номер телефона.
5. Электронная почта.
6. Дата рождения.
7. Имя пользователя (ник).
8. Пароль. В целях безопасности должен храниться в хэшированном виде.

Для сущности отчетов также необходимо реализовать поиск по ключевым параметрам для быстрого и удобного доступа к интересующим отчетам. Ключевыми параметрами отчетов являются:

1. Название
2. Код CWE
3. ПО, к которому относится отчет.

4. Оценка эксплуатабельности. Должна быть возможность указать диапазон.
5. Оценка опасности. Должна быть возможность указать диапазон.

### 1.2.2 Нефункциональные требования

Наш отдел нуждается в данном решении как можно скорее, поэтому желательно использовать технологии, позволяющие быстро разрабатывать подобные системы.

Большинство сотрудников нашего отдела в высокой степени знакомы с языком Python, поскольку он широко используется при написании сценариев поиска уязвимости. Также данное большинство не имеет большого опыта в разработке программного обеспечения. Поэтому в целях поддерживаемости кода необходимо использовать технологии, позволяющие разрабатывать на языке Python, имеющие подробные и понятные документации, пользующиеся широкой поддержкой сообщества программистов по всему миру.

Данное решение должно быть совместимо с уже существующим в организации кластером баз данных PostgreSQL.

Поскольку наш отдел занимается исследованием безопасности данная технология должна позволять разрабатывать достаточно защищенные системы. Взлом разработанной системы может поставить под удар репутацию нашего отдела, а также привести к публикации коммерческой тайны, которой в последствии могут воспользоваться наши конкуренты.

Мы – российская компания, наш документооборот происходит на русском языке, поэтому необходимо в достаточной мере провести русскую локализацию, чтобы носители русского языка могли комфортно пользоваться системой.

Также руководителем отдела была поставлена задача реализовать собственное решение, поскольку, как он пояснил, существующие в данный момент решения не удовлетворяют потребности отдела.

## ГЛАВА 2. ПРОЕКТИРОВАНИЕ

### 2.1 Использование системы

Разрабатываемый веб-сайт должен представлять из себя CRUD систему для управления отчетами по выявленным уязвимостям, которую необходимо совместить с системой авторизации.

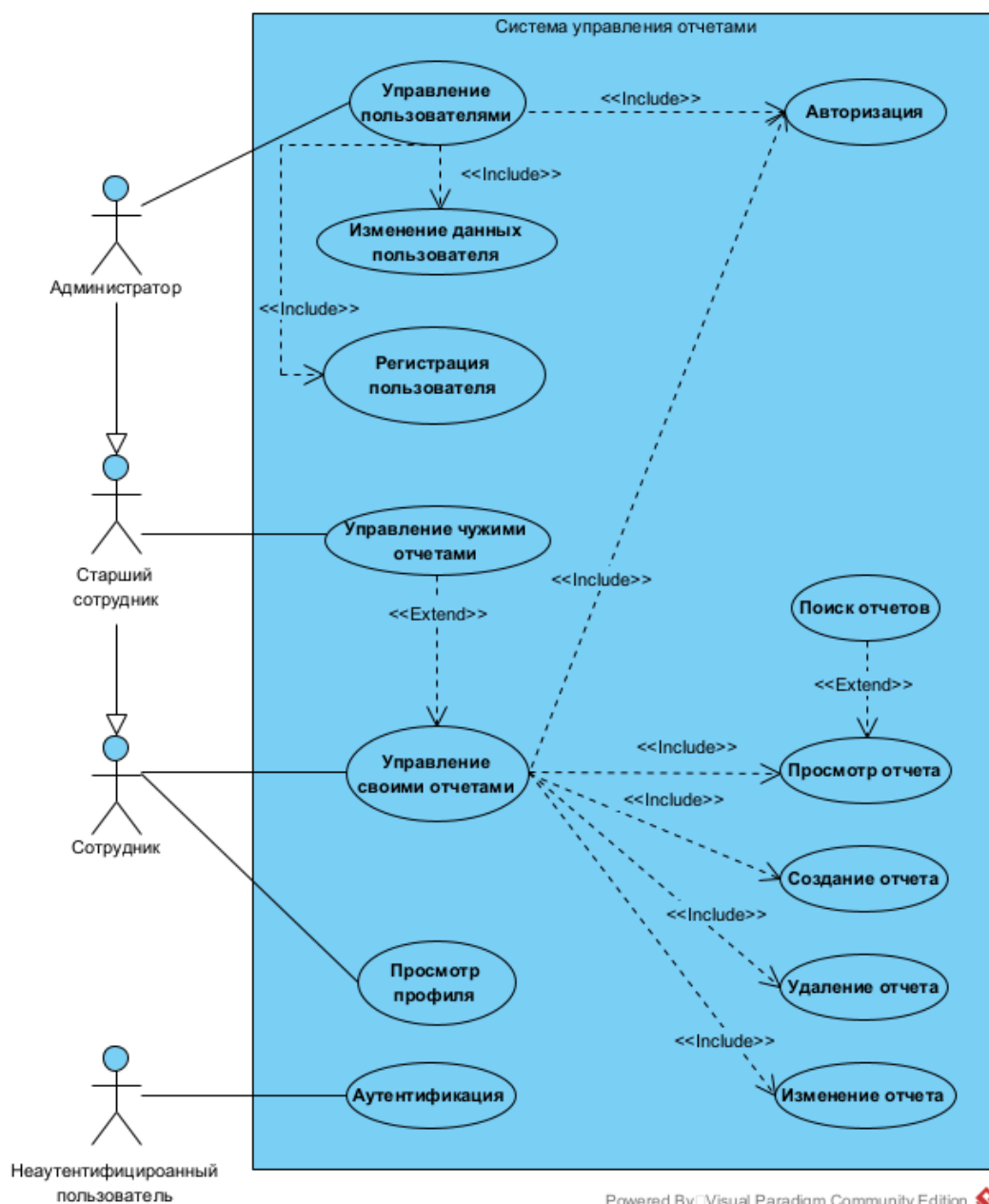


Рисунок 2 – Диаграмма вариантов использования

На данной диаграмме изображены несколько видов пользователей. Обычный пользователь может войти в систему (авторизоваться) и управлять

(создавать, просматривать, изменять, удалять свои, а также просматривать чужие) сущностями (отчетами, ПО, средами выполнения). Также есть администратор, который кроме возможностей обычного пользователя обладает правом управлять чужими отчетами, регистрацией и изменением данных пользователей. Администратор может наделить правом управлять чужими отчетами обычного пользователя по своему усмотрению, без предоставления ему прав управления пользователями.

Внутри системы происходит проверка прав доступа пользователей для предоставления им соответствующей функциональности (авторизация).

## **2.2 Архитектура данных**

### **2.2.1 ORM классы**

Архитектура данных, представленная на рисунке 4 будет автоматически сгенерирована в базе данных при развертывании приложения с помощью ORM классов Django. На диаграмме 4 представлены данные классы.



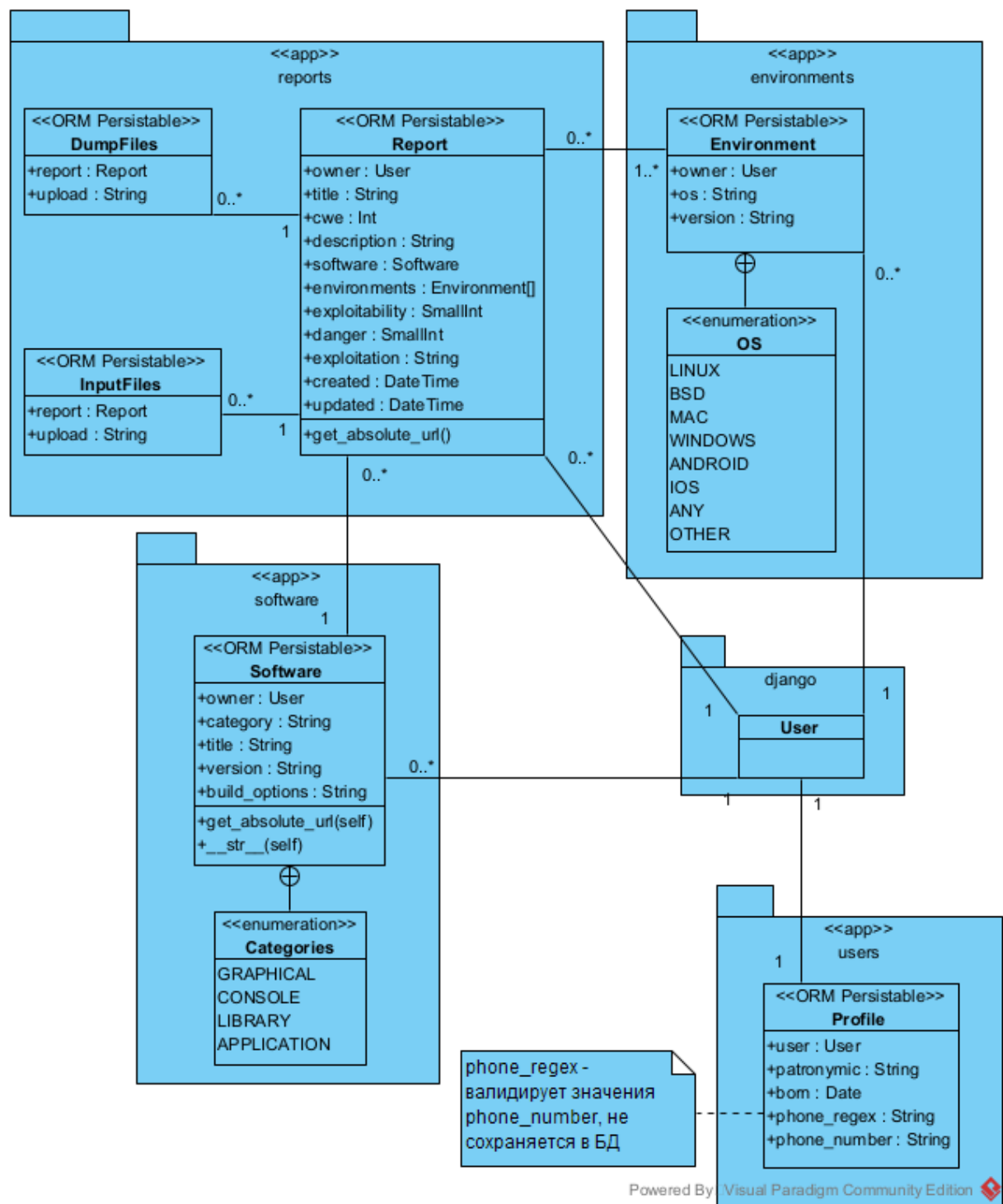


Рисунок 4 – Диаграмма ORM классов

### 2.2.2 База данных

В данном разделе будет приведена ER-диаграмма системы и ее подробное описание.

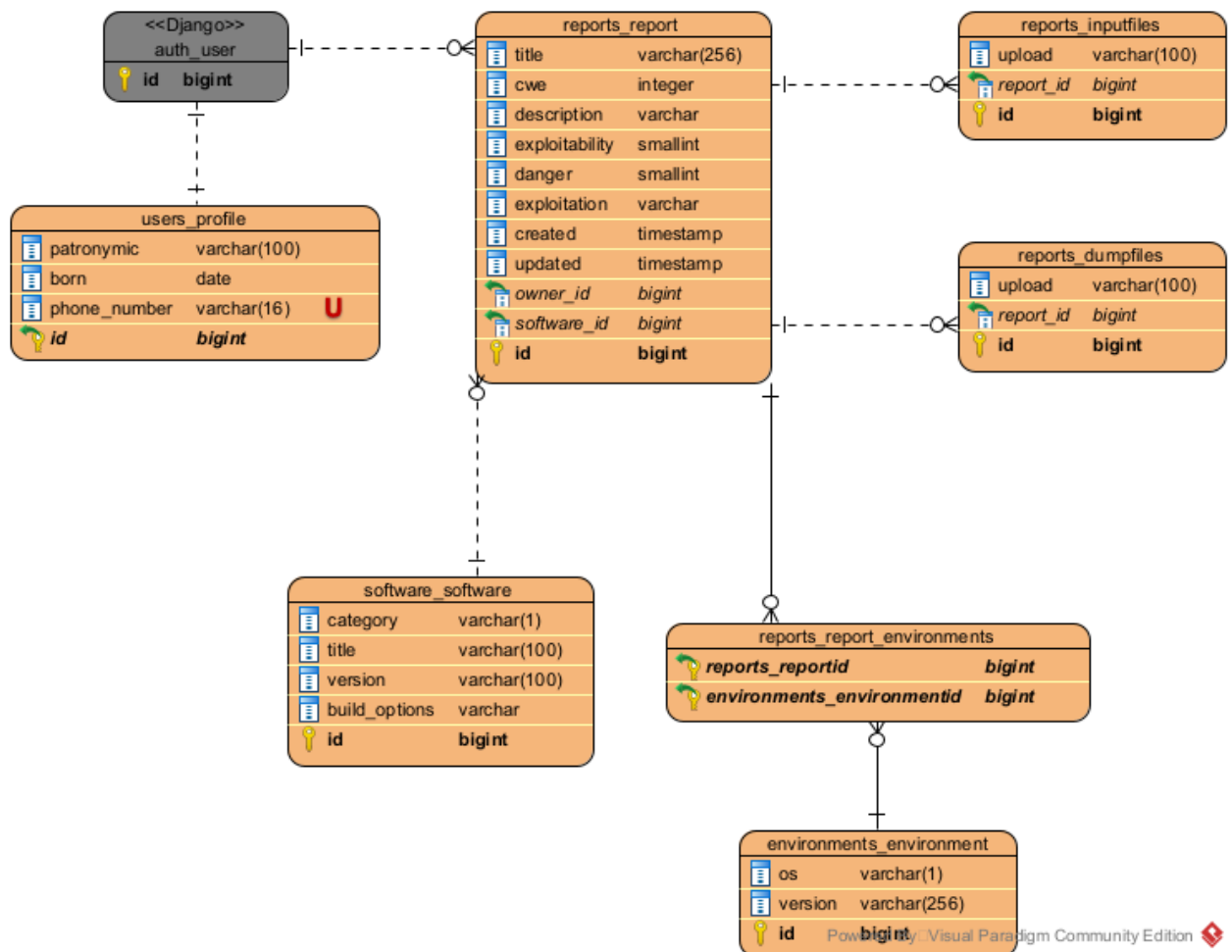


Рисунок 3 – ER-диаграмма всех сущностей системы

Таблица **reports\_report** на данной диаграмме представляет сущность отчета. Ее поля:

1. title – название уязвимости.
2. cwe – номер CWE.
3. description – описание уязвимости.
4. exploitability – оценка эксплуатабельности.
5. danger – оценка опасности.
6. exploitation – способ эксплуатации.
7. created – время создания объекта (заполняется автоматически).
8. updated – время изменения объекта (заполняется автоматически).

9. `owner_id` – внешний ключ, указывающий на создавшего данный объект пользователя.
10. `software_id` – внешний ключ, указывающий на объект ПО, для которого была найдена уязвимость.
11. `id` – первичный ключ объекта, заполняется автоматически как номер последней записи в таблице, инкрементированный на единицу.

Данная сущность имеет связанные сущности **reports\_inputfiles** и **reports\_dumpfiles**. Эти сущности служат таблицами хранения информации о загруженных файлах – входных файлах и дамп-файлах. В данных сущностях поля:

1. `upload` – полный адрес файла в файловой системе.
2. `report_id` – внешний ключ, указывающий на объект отчета.
3. `id` – первичный ключ, заполняется автоматически как номер последней записи в таблице, инкрементированный на единицу.

Сущность **software\_software** представляет собой ПО, для которого осуществляется поиск уязвимости. Данная сущность имеет поля:

1. `category` – категория ПО.
2. `title` – название ПО.
3. `version` – версия ПО.
4. `build_options` – опции сборки ПО.
5. `id` – первичный ключ объекта, заполняется автоматически как номер последней записи в таблице, инкрементированный на единицу.

Сущность **environments\_environment** представляет собой среды выполнения ПО. Данная сущность связана с сущностью отчетов с помощью вспомогательной таблицы для отображения отношения многие-ко-многим. Данная сущность имеет поля:

1. os – тип операционной системы.
2. version – версия операционной системы.
3. id – первичный ключ, заполняется автоматически как номер последней записи в таблице, инкрементированный на единицу.

Сущность **users\_profile** представляет собой расширение полей сущности **auth\_user**, которая является встроенной сущностью Django для управления пользователями. Данная сущность имеет поля:

1. patronymic – отчество сотрудника.
2. born – дата рождения сотрудника.
3. phone – номер телефона сотрудника. Перед занесением в таблицу проверяется на корректность в соответствующем ORM классе Django с помощью регулярного выражения.
4. id – первичный ключ, заполняется автоматически как номер последней записи в таблице, инкрементированный на единицу.

## 2.3 Выбор технологий

Согласно нефункциональным требованиям выбранная технология для создания веб-приложения должна поддерживать высокую скорость разработки, иметь высокую степень безопасности, поддерживать язык программирования Python и быть совместимым с уже существующим кластером баз данных PostgreSQL в организации.

Согласно опросу разработчиков [14], использующих язык программирования Python, проведенному компанией JetBrains, существуют

две наиболее популярных технологии разработки полноценных веб-приложений на языке Python, позволяющих создавать как веб-интерфейсы, так и серверную часть. Данные технологии пользуются широкой поддержкой сообщества разработчиков по всему миру.

Технологии Django и Flask представляет собой фреймворки для создания веб-приложений на языке Python. [5] Основная разница между данными фреймворками заключается в том, что Flask является микрофреймворком предоставляющим лишь базовые возможности по умолчанию, в то время как Django – полноценный фреймворк, который включает в себя многие необходимые компоненты для разработки веб-приложений, такие как ORM, по умолчанию. [15]

В силу своего минимализма Flask не предоставляет по умолчанию такие компоненты, как ORM и панель администрирования ([15], [16], [17] и [20]), а также веб-формы для отображения веб-интерфейсом ([15], [17] и [18]). Также Flask обладает более слабой системой безопасности ([16], [17], [18] и [19]). Одни из ключевых задач в данной работе – авторизация, аутентификация и управление аккаунтами пользователей также не предоставляются по умолчанию Flask, в отличии от Django ([15], [17] и [18]). В целом, согласно многочисленным источникам ([17], [18], [19], [20], [21], [22] и [23]) Flask больше подходит для создания простых статических одностраничных веб-приложений, а Django – для создания сложных функциональных веб-приложений.

Таким образом фреймворк Django является подходящим вариантом для использования в качестве технологии для создания веб-приложения.

## **2.4 Программная архитектура**

Согласно официальному сайту Django:

- Использует язык Python для разработки.

- Обладает высокой скоростью разработки, позволяющей разработчикам быстро создавать из концептов полноценные приложения.
- Обладает высокой степенью защиты. Django имеет встроенные меры безопасности, предотвращающие многие известные ошибки безопасности. Также данный фреймворк предоставляет встроенную защищенную систему менеджмента аккаунтов пользователей.

Для программной реализации был выбран язык Python версии 3 и веб-фреймворк Django версии 4. В качестве СУБД была выбрана PostgreSQL версии 14.

Django использует собственную технологию ORM, благодаря которой будет происходить общение с базой данных PostgreSQL. Данная технология позволяет создавать SQL запросы к базе данных используя язык Python. Также Django по умолчанию использует свою технологию шаблонизации HTML – DTL (Django Template Language). Данный шаблонизатор позволяет использовать специальные теги внутри HTML документа для реализации более сложной логики, позволяющей динамически генерировать HTML страницы. DTL имеет большое количество встроенных тегов и функций, а также позволяет создавать свои теги и функции программисту.

Согласно терминологии веб-фреймворка Django каждый проект на данном фреймворке состоит из множества приложений, которые представляют какую-то функциональность [3]. Разработанная система состоит из 4 основных приложений:

1. Приложение пользователей – отвечает за разработку пользовательской функциональности (например, вход в систему).
2. Приложение отчетов – отвечает за разработку функциональности отчетов (например, их создание и просмотр).

3. Приложение ПО – отвечает за разработку функциональности программного обеспечения, для которого выявляются уязвимости (например, их создание и просмотр).
4. Приложение сред выполнения – отвечает за разработку функциональности сред выполнения, на которых была выявлена уязвимость (например, их создание и просмотр).

Веб-фреймворк Django использует шаблон проектирования MTV (Model-Template-View), являющийся модификацией шаблона MVC (Model-View-Controller). В данном шаблоне проектирования компонент View примерно соответствует компоненту Controller для шаблона MVC, аналогично компонент Template соответствует компоненту View. Также одним из основных принципов Django является DRY (от англ. Don't Repeat Yourself, рус. Не повторяйся). Данный принцип разработки программного обеспечения направлен на снижение повторения информации различного рода в системе. В фреймворке Django данный принцип выражается в том числе наличием таких встроенных классов как миксины, один из которых будет использован в программной реализации системы.

Также важно отметить, что для создания пользовательских интерфейсов активно использовалась технология Bootstrap, помогающая быстро и удобно создавать адаптивные веб-интерфейсы и минимизировать использование технологии CSS.

#### 2.4.1 Приложение пользователей

Приложение пользователей отвечает за работу с пользователями. В данном приложении была разработана ORM модель, дополняющая существующую модель управления пользователями в Django. Была реализована функциональность отображения данной модели во встроенной административной панели Django. Также данное приложение ответственно за контроль аутентификации пользователей для получения прав доступа

использования системы, отображение их данных и выход пользователей из системы.

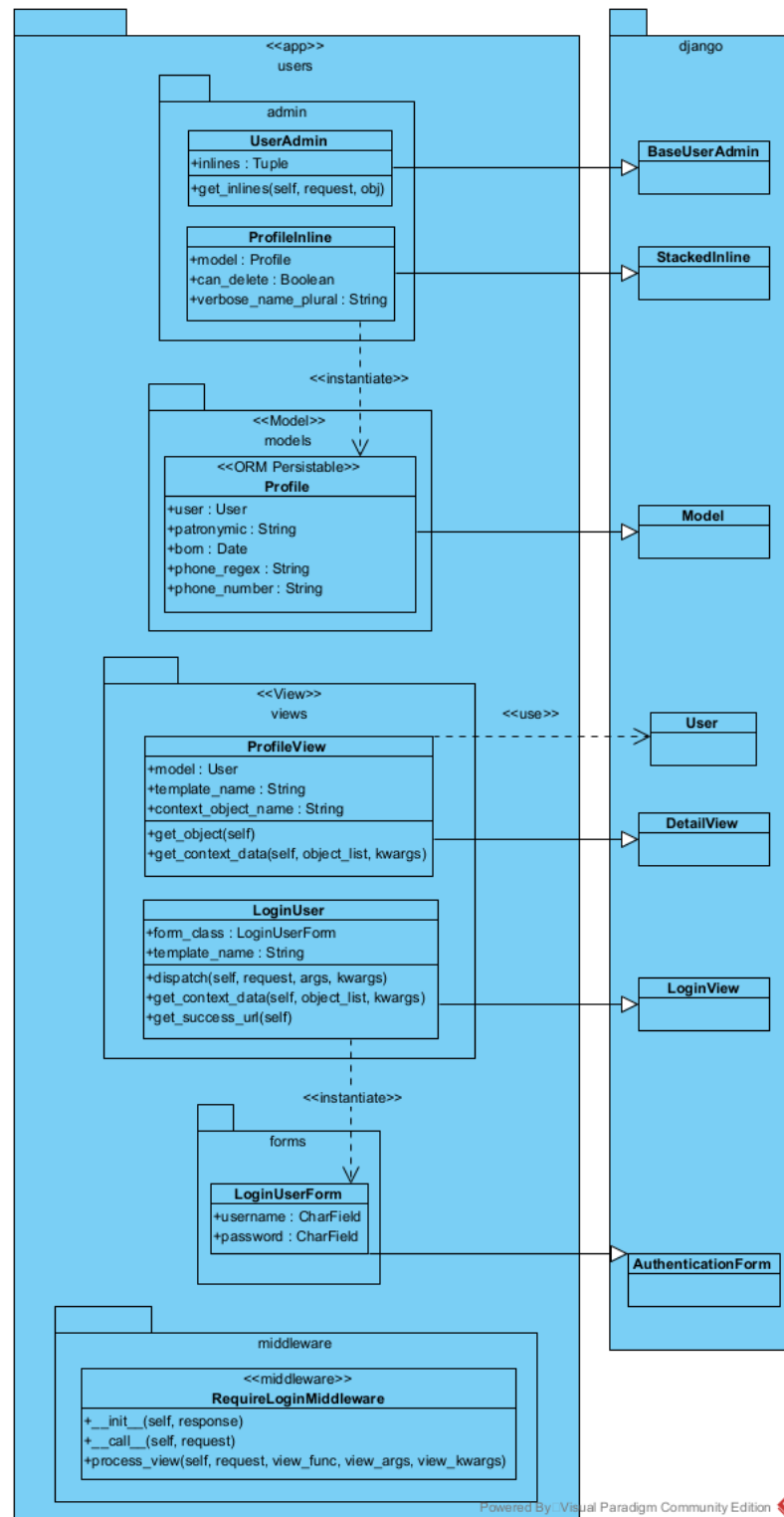


Рисунок 5 – Диаграмма классов приложения пользователей

На данной диаграмме приведены различные классы, составляющие приложение пользователей.



Класс **ProfileInline** отвечает за создание дополнительной формы, состоящей из полей модели **Profile**, добавляемой к основной, встроенной в Django, модели пользователей. Класс **UserAdmin** модифицирует форму управления моделью пользователей в Django, добавляя в нее соответствующую форму **Profile**, тем самым расширяя количество информации, относящееся к каждому пользователю. Данный класс далее будет зарегистрирован, заменив таким образом предыдущую форму управления собой.

Класс **Profile** представляет собой класс ORM, управления соответствующими записями пользовательских профилей в базе данных. Атрибут **phone\_regex** представляет собой регулярное выражение, которое служит для определения корректности введенного номера телефона. Является представителем компонента Model в шаблоне проектирования MVT. Класс Profile наследуется от стандартного класса Django, который необходим для создания собственных ORM классов. Для осуществления связи со встроенным ORM классом Django **User** для управления пользователями данный класс импортируется и подключается в атрибуте **user** класса **Profile**. С помощью модуля **models** происходит создание полей ORM класса. Регулярное выражение переменной **phone\_regex** определяющее корректность номера телефона опирается на рекомендацию ССЭ МСЭ E.164, согласно которой максимально допустимое количество цифр телефонного номера ограничивается 15 цифрами.

Класс **ProfileView** является элементом View в шаблоне проектирования MVT. Данный класс ответственен за отображение страницы профиля пользователя и отображение корректных пользовательских данных на данной странице. При получении запроса от пользователя класс **ProfileView** отправляет ему необходимый элемент компонента Template в шаблоне проектирования MVT. Данный элемент представляет собой файл страницы HTML использующий кроме стандартных тегов дополнительные элементы

DTL (Django Template Language), которые ответственны за динамическую генерацию страниц HTML. Благодаря DTL данные пользователя корректно (внутри необходимых тегов, используя необходимые стили CSS) отображаются на странице его профиля.

Для отображения страницы профиля пользователю была использована HTML страница `profile.html`. В данном файле также была использована технология DTL для динамической генерации HTML страниц. Также для корректировки отображения данной страницы была использована технология CSS. Для использования DTL необходимо указывать специальные теги (фигурные скобки) внутри данного HTML файла (не являются стандартными тегами HTML). С помощью DTL происходит отображение такой информации о пользователе, как его фамилия, имя, отчество, адрес электронной почты и другой.

Класс **LoginUser** также является представителем компонента View в шаблоне проектирования MVT. Данный класс предоставляет пользователю возможность войти в систему и использует для этого шаблон `login.html`, а также использует класс формы **LoginUserForm**, которая отображается в шаблоне с помощью DTL. Данная форма генерируется автоматически, также с помощью DTL на странице могут отображаться ошибки полей и самой формы. Важно отметить необходимость обязательного использования встроенной защиты Django от CSRF атак с помощью использования специального DTL тега `csrf_token`, его отсутствие приведет к ошибке при отправке формы.

Класс **RequireLoginMiddleware** является одним из важнейших в данном приложении, поскольку он ответственен за предоставление доступа к системе зарегистрированным пользователям и отказ в доступе для незарегистрированных пользователей.

В Django существует система связующего программного обеспечения (middleware) и данный класс является представителем такого ПО. Метод `__init__` в данном классе отвечает за первоначальную настройку данного ПО, данный метод вызывается один раз при запуске веб-приложения. Метод `__call__` используется для выполнения логики до и после обработки запроса, вызывается при каждом запросе пользователя. Самый важный метод в данном классе – **`process_view`**, он вызывается прямо перед выполнением компонента View. Django ожидает, что данный метод вернет либо **None** (в таком случае Django перейдет к выполнению компонента View), либо объект **HttpResponse** (в этом случае Django не будет переходить к выполнению компонента View, а вместо этого обработает и вернет пользователю данный объект). В данном методе осуществляется проверка прав доступа пользователя для использования системы. Если пользователь не аутентифицирован – у него нет прав на посещение всех ссылок, кроме специально обозначенных. Исключения составляют аутентифицирующие ссылки, которые собраны в переменной **`NO_AUTH_URLS`**, представляющей собой кортеж регулярных выражений. Переменные конфигурации данного ПО содержатся в настройках проекта.

Пример получения пользователем страницы со своими данными может быть описан следующим образом.

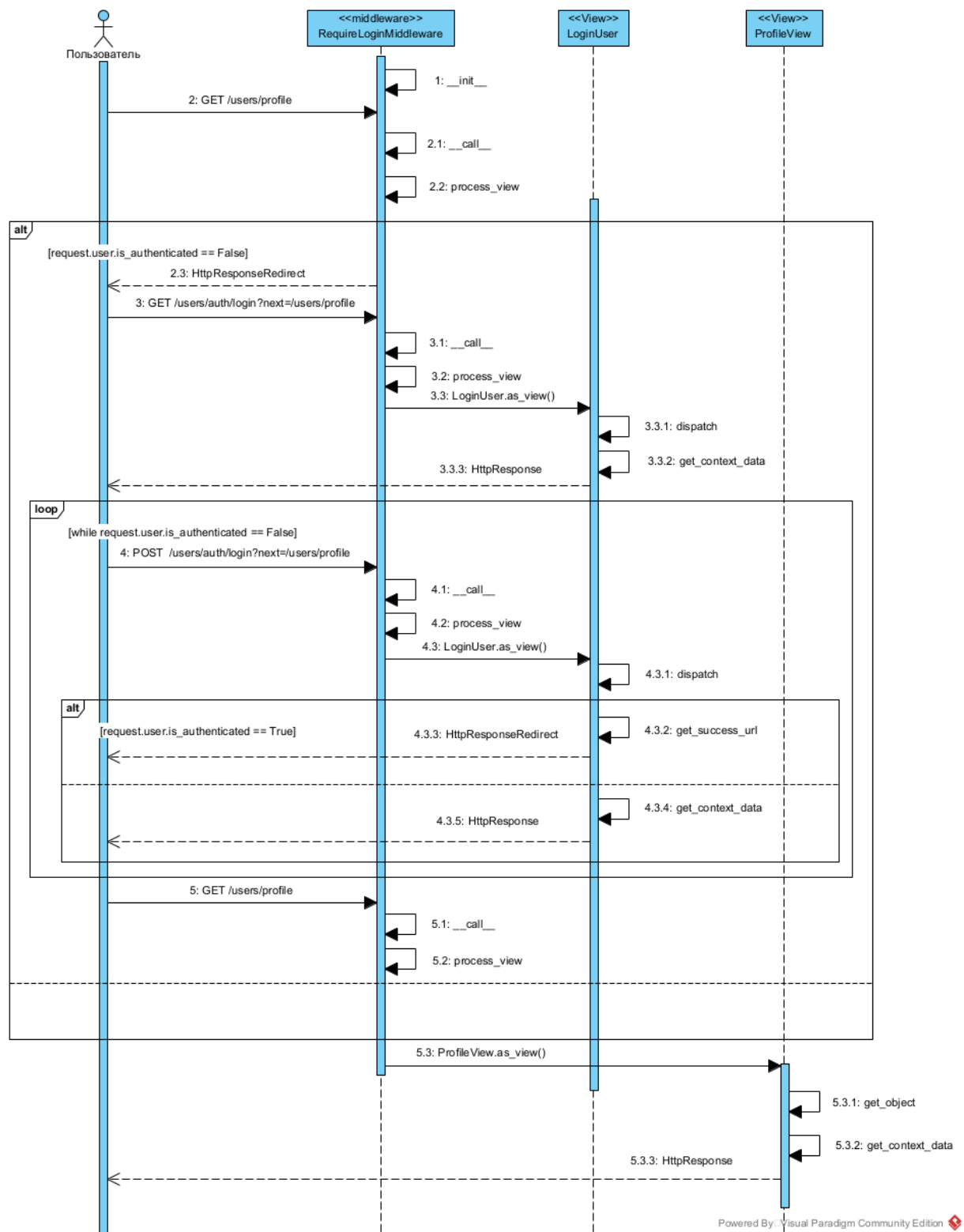


Рисунок 6 – Диаграмма последовательности для получения страницы профиля

На данной диаграмме показан процесс получения страницы профиля пользователя. На данной диаграмме важно отметить, что пользователь будет проходить процесс аутентификации пока он не завершится успешно.

В представленную диаграмму классов не вошли функции представлений – также являются элементами компоненты View.

Функция **logout\_user** ответственна за выход пользователя из системы, а функция **view\_404** переопределяет стандартное поведение Django при возникновении ошибки 404 (то есть страница не была найдена) и производит переадресацию пользователя на страницу его профиля.

#### 2.4.2 Приложение отчетов

Данное приложение предоставляет важнейшую функциональность системы – управление отчетами, а именно функциональность по созданию, просмотру, изменению, удалению отчетов, а также поиску по специальным критериям.

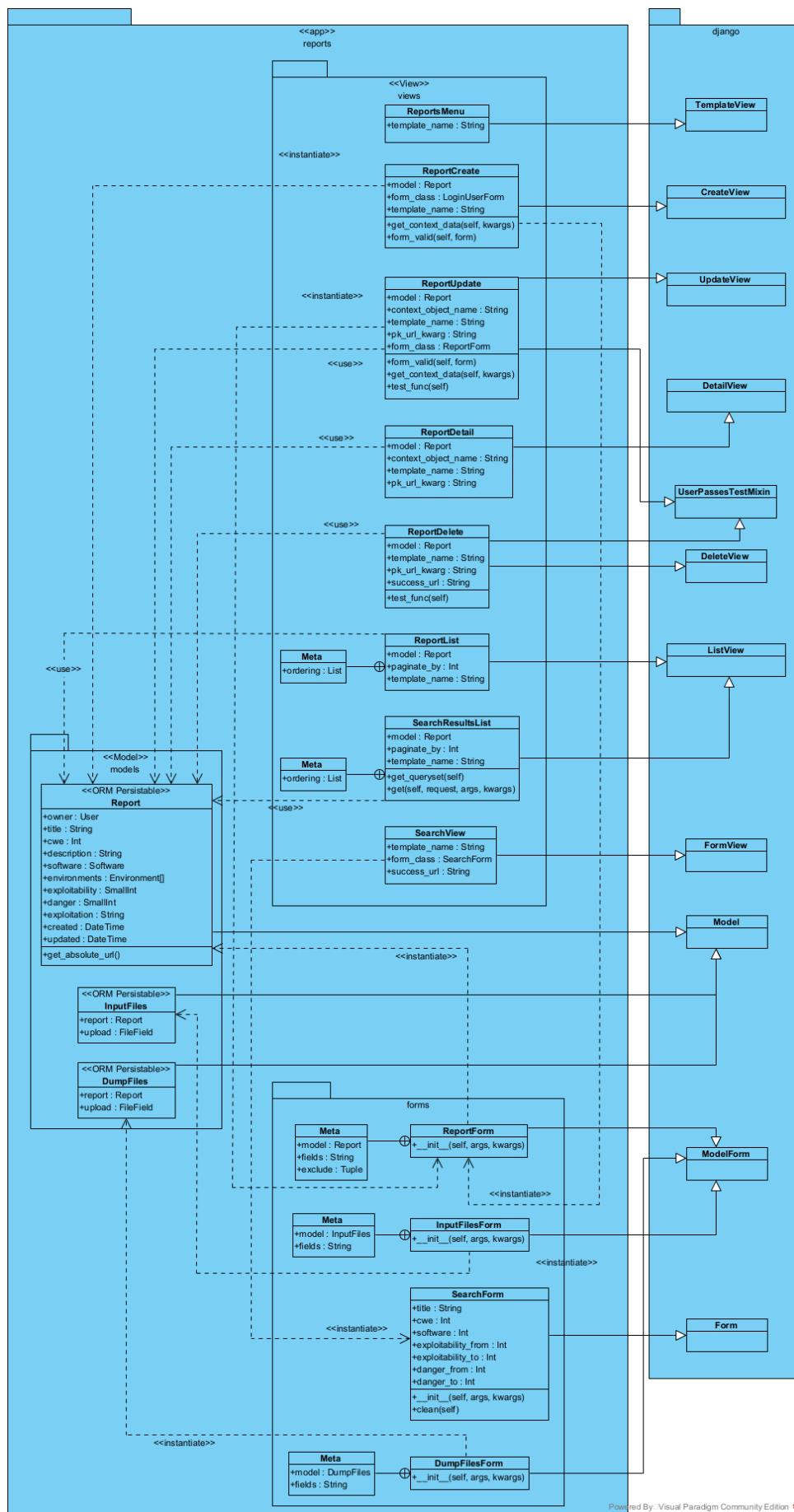


Рисунок 7 – Диаграмма классов приложения отчетов

На рисунке 8 представлена диаграмма классов приложения отчетов. Центральным и важнейшим классом в данном приложении является **Report**, который представляет собой ORM класс отчета. Метод **get\_absolute\_url** возвращает абсолютную URL ссылку на конкретный экземпляр данного класса, он основывается на идентификаторе экземпляра в базе данных (id).

Одним из главных процессов пользования системой является процесс создания и публикации отчета. Создание отчета происходит в классе представления **ReportCreate**. Данный класс использует формы создания отчетов и файлов для генерации соответствующего шаблона HTML. Для управления дополнительными формами отправки файлов был использован статический файл JavaScript. Как уже было упомянуто ранее, при генерации формы в HTML важно указать тег DTL `csrf_token`, данный тег необходимо указывать для каждой формы, отправляющей POST запрос. Пользователь может загрузить файлы входных данных или дампа памяти при создании отчета, данные файлы впоследствии могут быть использованы другими сотрудниками для воспроизведения и исследования уязвимости. Файлы сохраняются в специальной директории, затем адрес сохраненных файлов записывается в базу данных с помощью ORM классов **InputFiles** и **DumpFiles**. Количество файлов не ограничено. Максимальный размер отправляемых файлов ограничен настройками прокси-сервера Nginx.

Для авторизованного пользователя сценарий создания и публикации отчета (при условии наличия необходимых сред выполнения и ПО) выглядит следующим образом:

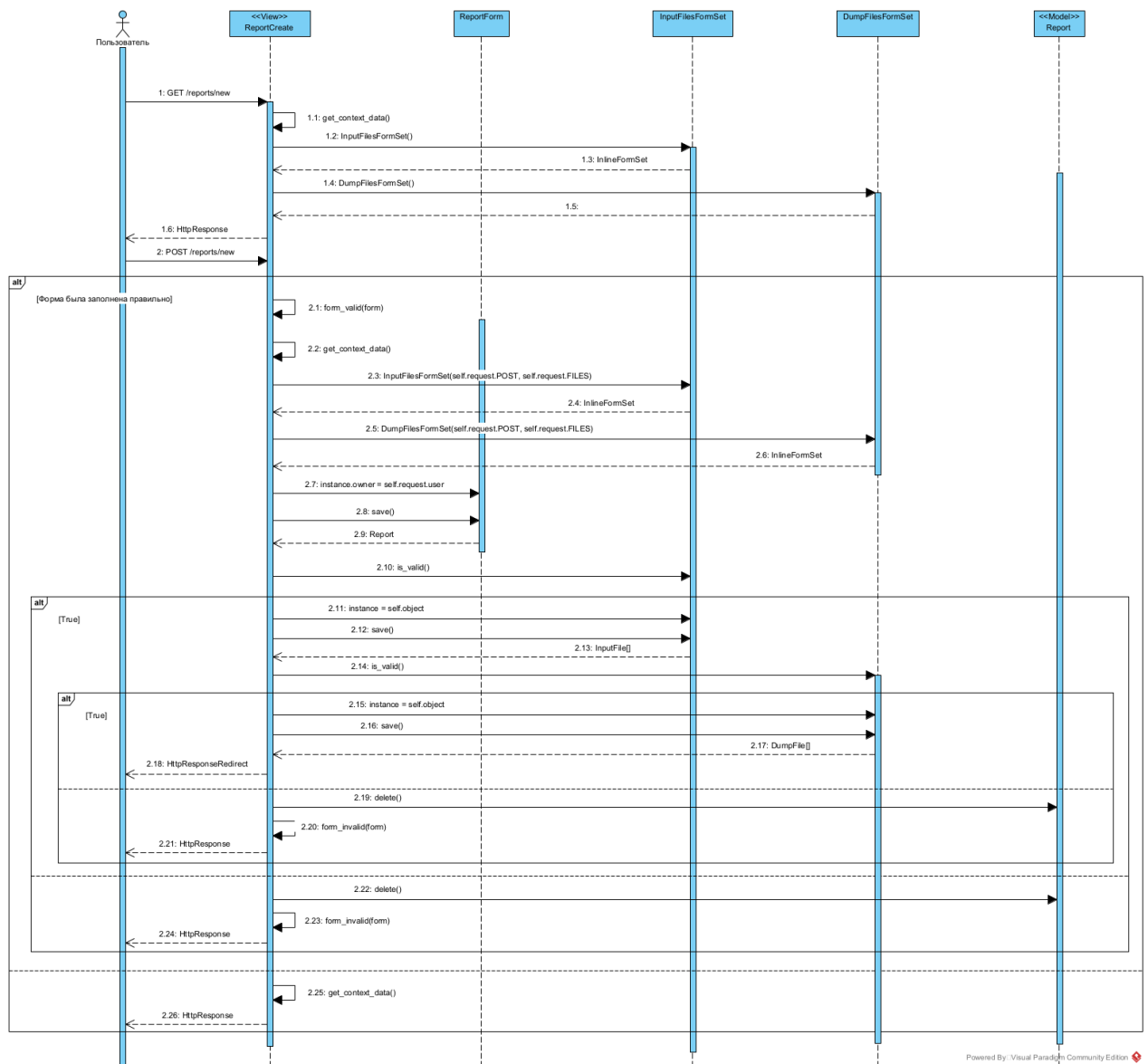


Рисунок 8 – Сценарий создания и публикации отчета пользователем

Главным действующим элементом на данной диаграмме является класс представления **ReportCreate**. При обращении пользователя по определенному URL Django передает соответствующий автоматически созданный объект класса **HttpRequest** данному классу, после обработки данного запроса, класс **ReportCreate** возвращает объект **HttpResponse**, представляющий собой HTTP ответ. Процесс создания отчета состоит из двух этапов – получение и заполнение формы пользователем, после чего отправка этой формы обратно в **ReportCreate**, где она обрабатывается и создается соответствующий объект в базе данных в таблице **reports\_report**. По окончании процесса создания объекта класс **ReportCreate** отправляет пользователю объект



**HttpResponseRedirect**, который перенаправляет его на страницу с детальной информацией о только что созданном объекте. При этом если в процессе обработки формы произойдет ошибка, она отобразится пользователю на странице, где была заполнена форма.

Отображение страницы с детальной информацией об отчете происходит с помощью класса представления **ReportDetail**. Данный класс использует соответствующий шаблон для генерации HTML страницы с информацией о созданном отчете. Данная страница отображается по URL адресу, включающему идентификатор объекта в базе данных.

В случае, если пользователь имеет право на редактирование просматриваемого отчета, то на странице внизу будет отображена небольшая панель управления, включающая две опции: изменить и удалить.

При необходимости внесения правок в существующий отчет он может быть изменен посредством нажатия кнопки «Изменить». Функциональность изменения отчета предоставляется классом представления **ReportUpdate**. Для изменения отчета аналогично предыдущим пунктам нужно перейти по соответствующему URL включающему идентификатор объекта. При обработке GET запроса данный класс использует форму отчета, заполненную данными изменяемого объекта. При отправлении формы изменения объекта сохраняются в базе данных. Все поля объекта могут быть изменены, включая прикрепляемые файлы.

В случае неактуальности или некорректности отчета (например, если уязвимость невозможно воспроизвести) он может быть удален с помощью нажатия кнопки «Удалить» в указанном меню. Удаление отчета происходит с помощью класса представления **ReportDelete**. При нажатии указанной кнопки открывается страница для подтверждения действия. В случае подтверждения объект будет удален. При этом записи о файлах, прикрепленных к удаленному объекту также будут удалены каскадным образом.

Проверка прав пользователя на изменение или удаление объекта происходит с помощью переопределения метода специального класса **UserPassesTestMixin**. Данный класс позволяет сделать проверку прав доступа пользователя перед предоставлением ему функциональности соответствующего класса представления. Разрешение доступа происходит в двух случаях:

- Пользователь является владельцем объекта.
- Пользователь входит в круг лиц, которым разрешено управлять чужими объектами.

Для удобства в данном приложении реализовано простое меню для быстрого доступа к основной функциональности, такой как:

- Создание отчета
- Просмотр всех отчетов
- Поиск отчетов

Просмотр всех отчетов осуществляется с помощью класса представления **ReportList**. Данный класс по запросу возвращает пользователю краткую информацию о содержащихся в базе данных записях отчетов. Краткая информация включает в себя:

- Название отчета
- Код CWE
- Часть описания, включающая в себя 20 первых слов
- Степень опасности
- Степень эксплуатабельности

Данная информация будет отсортирована в порядке убывания времени обновления отчетов, что соответствует их актуальности. При отображении

отчетов используется пагинация – на странице не может быть больше пяти отчетов.

Каждый блок информации об определенном отчете является ссылкой на данный отчет. Нажав на данную ссылку, пользователь переходит на страницу с подробной информацией об отчете.

Поиск отчетов осуществляется с помощью двух классов представления. Класс представления **SearchView** предоставляет форму для поиска отчетов по ключевым критериям, таким как:

- Название. Для заданной строки осуществляет поиск в базе данных для соответствующего поля. Регистр символов строки не имеет значения.
- Код CWE. Для введенного значения осуществляет поиск строгого соответствия для соответствующего поля в базе данных.
- Номер ПО, к которому будут относиться отчеты. Для введенного значения осуществляет поиск строгого соответствия для соответствующего поля в базе данных.
- Эксплуатабельность (от). Нижняя граница эксплуатабельности (включительно).
- Эксплуатабельность (до). Верхняя граница эксплуатабельности (включительно).
- Опасность (от). Нижняя граница опасности (включительно).
- Опасность (до). Верхняя граница опасности (включительно).

Все параметры не являются обязательными. В случае если никакие параметры не были заполнены пользователем возвращается полный список всех отчетов.

За возвращение списка результатов отвечает класс представления **SearchResultsList**. Данный класс получает GET запрос от пользователя с указанием параметров поиска. По данным параметрам осуществляется поиск в базе данных с помощью системы составления запросов ORM класса отчетов. После фильтрации объектов соответствующей таблицы базы данных по всем предоставленным параметрам, данный класс возвращает пользователю результаты поиска в виде списка. Данный список аналогичен списку, предоставляемому классом **ReportList**, пагинация осуществляется по такому же принципу.

### 2.4.3 Приложение ПО

Приложение ПО занимается управлением вспомогательной сущностью программного обеспечения. Данная сущность необходима для создания отчетов, поскольку уязвимость всегда принадлежит конкретному единственному ПО.

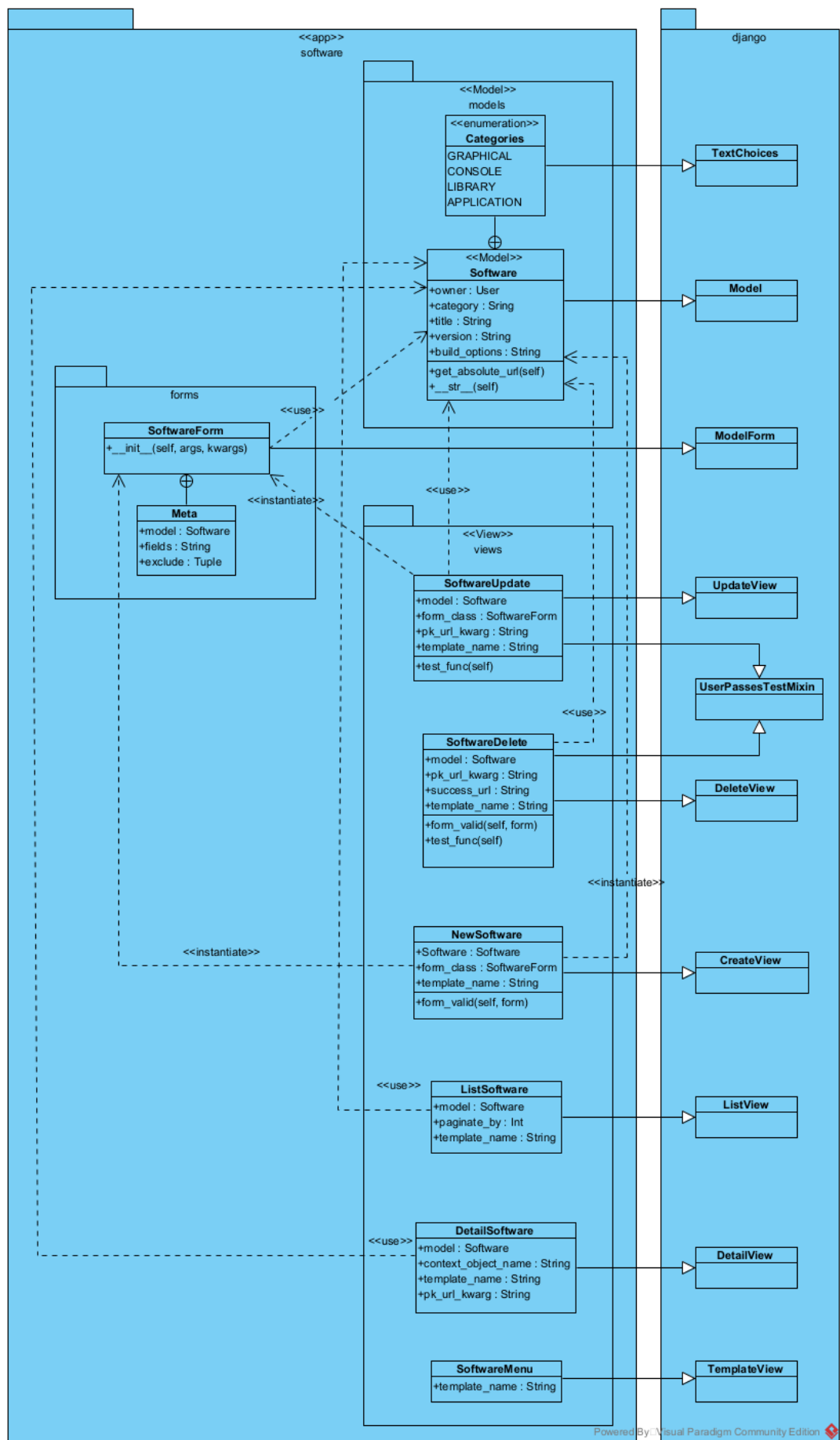


Рисунок 3 – Диаграмма классов приложения ПО

Сущность ПО представлена ORM классом **Software**. Аналогично предыдущему приложению, метод **get\_absolute\_url** возвращает ссылку на данный объект, включающую идентификатор объекта в базе данных. Метод **\_\_str\_\_** осуществляет преобразование объекта ПО в форму символьной строки.

Для создания объекта ПО используется класс представления **NewSoftware**. Принцип его работы аналогичен предыдущему приложению, при получении GET запроса данный класс отправляет пользователю форму создания нового ПО, после заполнения которой пользователь отправляет POST запрос. После обработки данного POST запроса создается соответствующий объект в базе данных.

Класс представления **DetailSoftware** предоставляет полную информацию о ПО, для ее просмотра необходимо перейти по соответствующей ссылке. Аналогично предыдущему приложению внизу страницы с детальной информацией находится меню управления для авторизованных пользователей, которое состоит из двух опций: «Изменить» и «Удалить».

Класс представления **SoftwareUpdate** предоставляет функциональность по изменению объекта ПО. Изменению может подлежать любая информация. Запрос на удаление объекта обрабатывает класс представления **SoftwareDelete**. Данный класс также запрашивает подтверждение перед удалением объекта, однако есть важное отличие. В случае если данный объект ПО уже имеет связанные с ним отчеты удаление будет невозможно и на странице с подтверждением появится соответствующее уведомление об этом.

Для просмотра списка всех объектов ПО необходимо воспользоваться классом представления **ListSoftware**. Данный класс отображает список объектов со всей информацией о них. Пагинация также ограничивает количество объектов на странице пятью.

#### 2.4.4 Приложение сред выполнения

Данное приложение включает в себе функциональность по управлению дополнительной сущностью сред выполнения. Принцип работы данного приложения аналогичен предыдущему приложению, имеются лишь два принципиальных отличия.

Первое отличие заключается в ORM классе сред выполнения. Сущность сред выполнения имеет три соответствующих поля. Второе отличие заключается в связи многие-ко-многим с сущностью отчетов, в отличие от связи один-ко-многим у сущности ПО. Данная связь реализуется автоматически в Django ORM с помощью создания промежуточной таблицы. Данный тип связи был выбран, поскольку некоторое ПО может иметь различную реализацию и поведение на разных платформах, что может обуславливать наличие уязвимостей на одних платформах и отсутствие на других.

#### 2.4.5 Основное приложение

При создании нового проекта Django создает основное приложение внутри него. Данное приложение включает в себя необходимую функциональность и настройки для работы всего проекта.

Настройка проекта происходит в файле `settings.py`. В данном файле происходит настройка директории сохранения медиа файлов, директории хранения статических файлов (файлов CSS и JavaScript), настройка данных директорий обязательна для работы соответствующей функциональности. Есть также возможность настроить локализацию (язык и время, часовой пояс) и режим работы веб-приложения:

- Режим отладки. При возникновении ошибок во время работы приложения пользователю будет отображена полная информация о них.

- Режим полноценной работы. В данном режиме пользователю будут отображены стандартные коды ошибок при их возникновении, подробности будут опущены.

Также можно настроить множество других параметров, используя переменные.

Важно отметить настройку работы с базой данных. Подключение к базе данных PostgreSQL осуществляется с помощью адаптера базы данных `psycopg2`. Данные для подключения к базе данных берутся из переменных окружения (среды). Таким образом их не придется хранить в исходном коде напрямую или заполнять каждый раз при развертывании, а вместо этого хранить напрямую в системе развертывания скрытно от системы управления версиями и использовать при сборке образов Docker.

Кроме базы данных, аналогичным способом (через переменные среды) также создается суперпользователь во время запуска Docker-контейнера приложения Django.

## **2.5 Системная архитектура**

Тестирование работы данного фреймворка будет осуществляться в условиях, приближенных к производственным. Данный фреймворк будет обмениваться данными с базой данных PostgreSQL, использовать WSGI HTTP сервер Gunicorn, обратный HTTP прокси-сервер Nginx.

Стандартная команда запуска сервера Django локально (`python manage.py runserver`) не подходит, так как официальная документация не рекомендует ее использовать в производственной среде, поскольку данный способ запуска не масштабируем и не надежен. В свою очередь Gunicorn является легковесным, производительным и масштабируемым WSGI сервером, предназначенным для использования в производственной среде.



Nginx необходим для оптимизации работы веб-приложения и проксирования данному приложению только запросов касающихся непосредственно его работы (например запрос динамически сгенерированной HTML страницы). В случае же запроса на предоставление статических или медиа файлов его будет обрабатывать Nginx.

Все данные приложения для удобства и универсальности будут находиться внутри Docker-контейнеров. Также для получения стилей будет использоваться Bootstrap – внешний провайдер для быстрого и удобного создания адаптивных веб-интерфейсов.

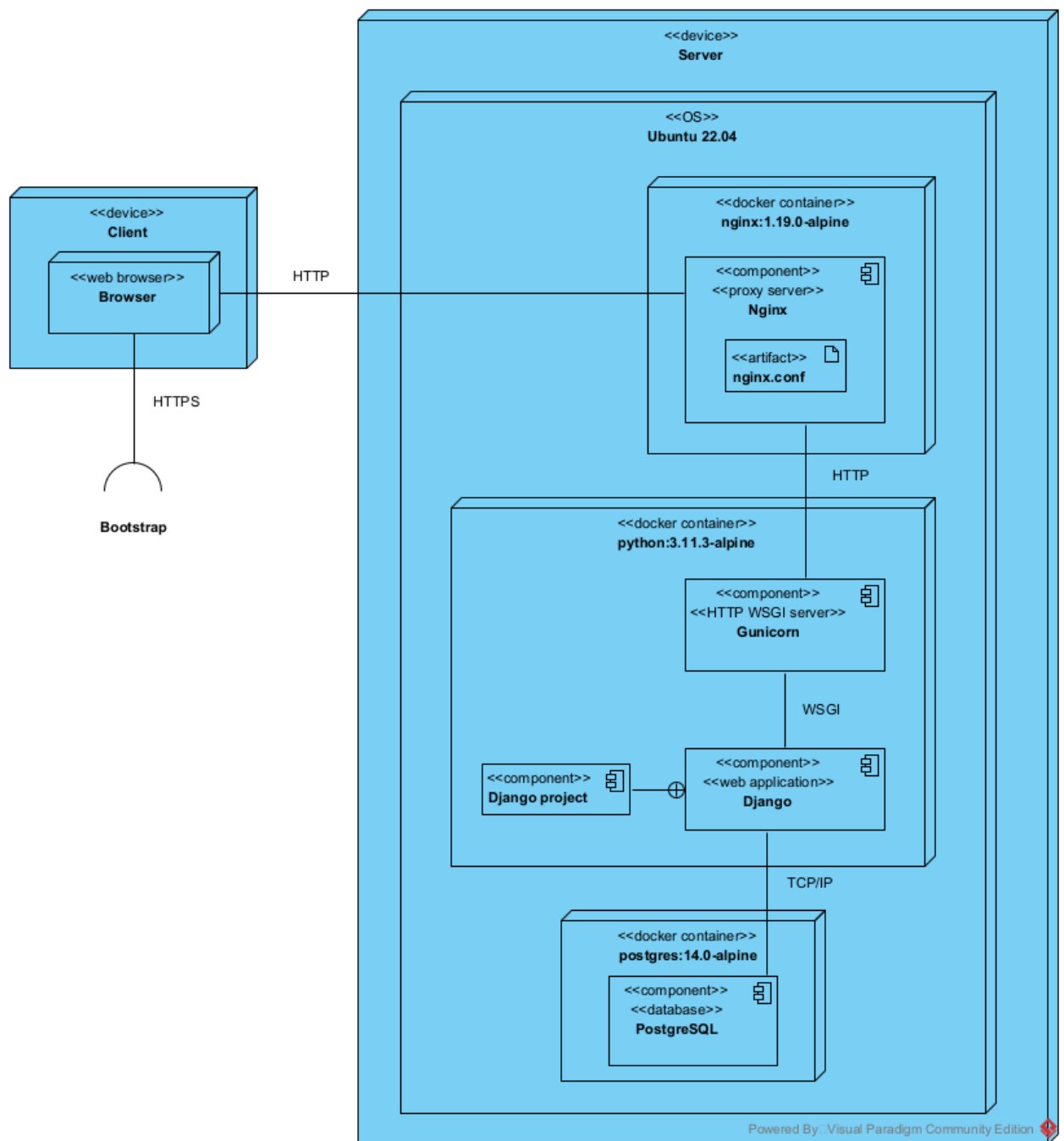


Рисунок 10 – Диаграмма развертывания системы

На данной диаграмме представлены условия тестирования системы, приближенные к производственным. Клиент, используя браузер по протоколу HTTP соединяется с обратным прокси-сервером Nginx. Данный прокси-сервер соединяется по протоколу HTTP с WSGI HTTP сервером Gunicorn, который в свою очередь подключается к приложению Django по протоколу взаимодействия WSGI. В случае, если необходимо получить доступ к данным,

приложение Django обращается по протоколу TCP/IP к базе данных PostgreSQL, находящейся в другом Docker-контейнере.

## ГЛАВА 3. РЕАЛИЗАЦИЯ И ТЕСТИРОВАНИЕ

### 3.1 Приложение пользователей

Главным пользователем системы является ее администратор. Как уже было сказано ранее учетная запись администратора создается во время запуска Docker-контейнера с данными указанными в файле окружения (файл `.env.prod`). Обращаясь по URL адресу `/admin/` можно войти в панель администратора с помощью этих данных.

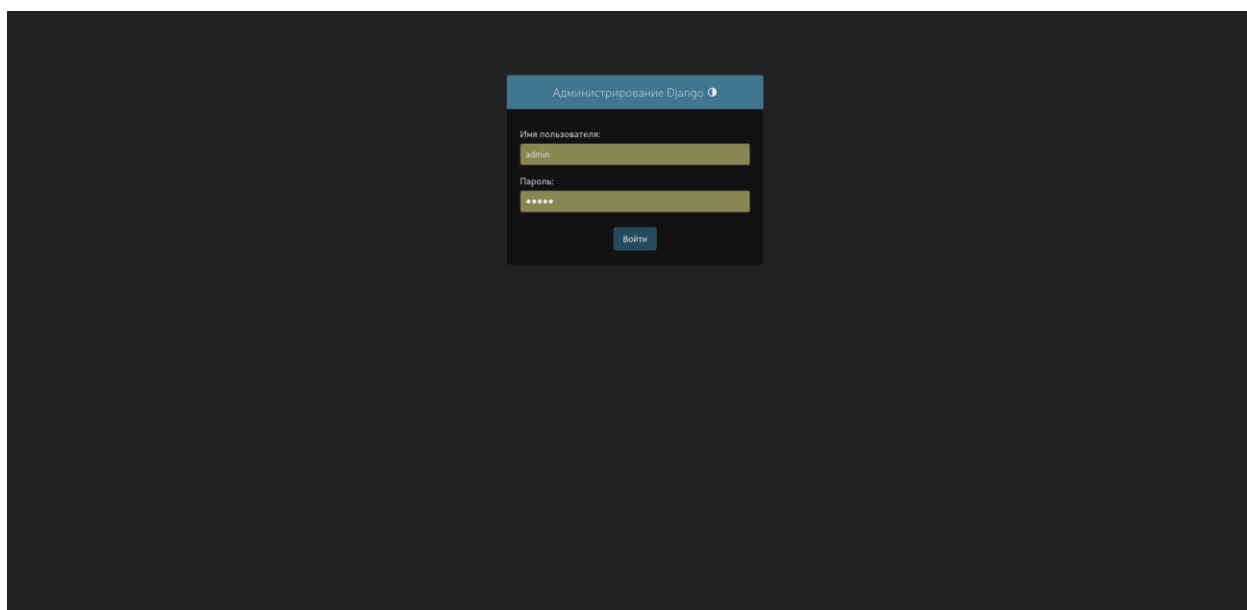


Рисунок 11 – страница входа в панель администрации

Администратор – единственный пользователь, у которого есть право создавать других пользователей. Процесс создания пользователя для администратора состоит из двух этапов – регистрация пользователя и добавления информации о пользователе:

Администрирование Django

Начало > Пользователи и группы > Пользователи > Добавить пользователя

добро пожаловать ADMIN | открыть сайт | изменить пароль | выйти

Начните печатать для фильтрации...

ПОЛЬЗОВАТЕЛИ И ГРУППЫ

Группы + Добавить

Пользователи + Добавить

### Добавить пользователя

Сначала введите имя пользователя и пароль. Затем вы сможете ввести больше информации о пользователе.

Имя пользователя:

Обязательное поле. Не более 150 символов. Только буквы, цифры и символы @, ., +, -, \_.

Пароль:

Пароль не должен быть слишком похож на другую вашу личную информацию.  
Ваш пароль должен содержать как минимум 8 символов.  
Пароль не должен быть слишком простым и распространенным.  
Пароль не может состоять только из цифр.

Подтверждение пароля:

Для подтверждения введите, пожалуйста, пароль еще раз.

Рисунок 12 – Регистрация пользователя

Для добавления информации о пользователе необходимо нажать кнопку «Сохранить и продолжить редактирование».

Администрирование Django

Начало > Пользователи и группы > Пользователи > user1

Изменить пользователь

user1

Имя пользователя: user1

Пароль: алгоритм: pbkdf2\_sha256 итерации: 60000 соль: BvWv\*\*\*\*\* хэш: H[D]\*\*\*\*\*

Персональная информация

Имя: Иван

Фамилия: Иванов

Адрес электронной почты: ivalov@mail.ru

Права доступа

☒ Активный

☐ Статус персона

☐ Статус суперпользователя

Группы: Доступные группы: 0

Выбранная группа: 0

Рисунок 13 – Заполнение информации о пользователе

admin | запись в журнале | Can add log entry  
admin | запись в журнале | Can change log entry  
admin | запись в журнале | Can delete log entry  
admin | запись в журнале | Can view log entry  
auth | группа | Can add group  
auth | группа | Can change group  
auth | группа | Can delete group  
auth | группа | Can view group  
auth | право | Can add permission  
auth | право | Can change permission  
auth | право | Can delete permission

Выбрать все 0

Удалить все

Важные даты

Последний вход: Дата: 14.05.2023 Сегодня | 📅  
Время: 20:28:14 Сейчас | ⌚

Дата регистрации: Дата: 14.05.2023 Сегодня | 📅  
Время: 20:28:14 Сейчас | ⌚

PROFILE

Profile: user1

Отчество: Иванов

Дата рождения: 14.05.2023 Сегодня | 📅

Номер телефона: +7111111111

СОХРАНИТЬ Сохранить и добавить другой объект Сохранить и продолжить редактирование Удалить

Рисунок 14 – Заполнение информации о пользователе

После заполнения необходимо нажать кнопку «Сохранить». Результат:

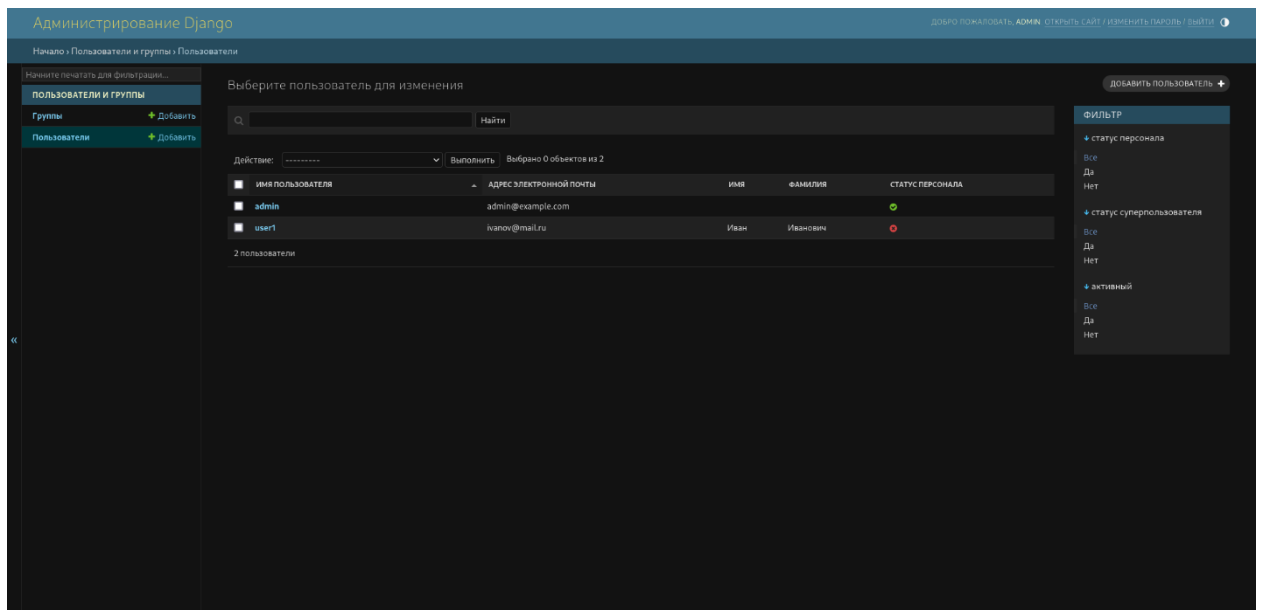


Рисунок 15 – Список пользователей

Пользователь появился в общем списке.

Создадим еще одного пользователя, но с правом управления чужими отчетами:

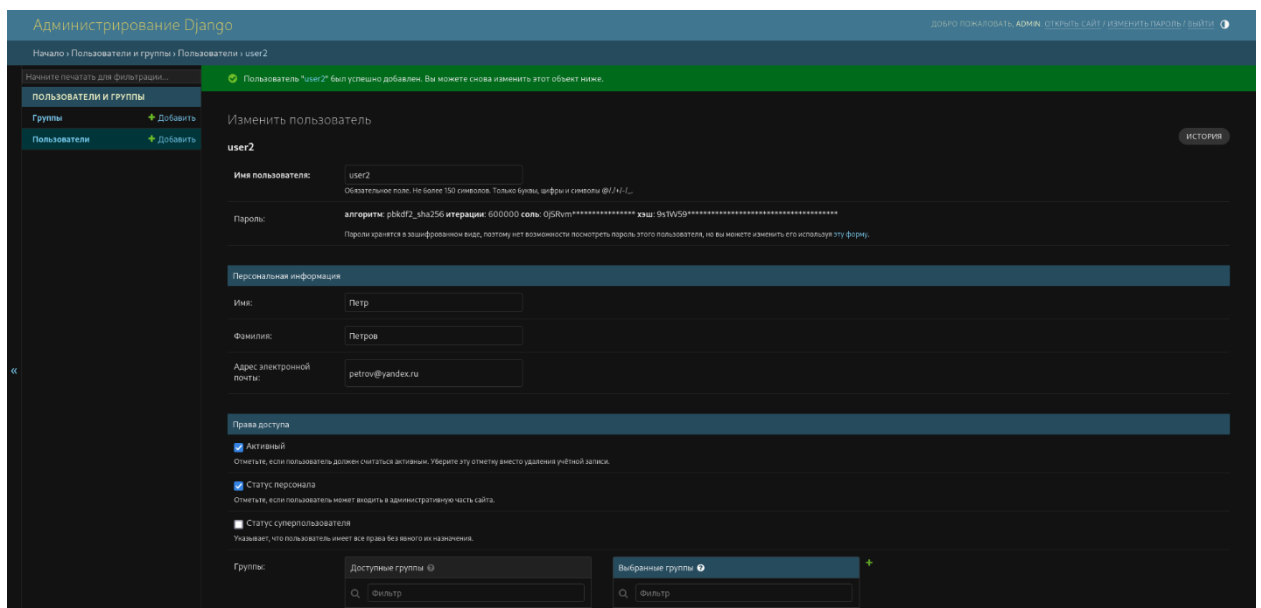


Рисунок 16 – Создание старшего сотрудника

Войдем в систему за Ивана и убедимся, что его данные отображаются верно на странице профиля, также убедимся в работе связующего ПО, перейдя по URL адресам «/» (не является корректным адресом в системе) и

«/reports/new» (корректный, но не доступный неаутентифицированным пользователям):

Вам нужно войти!

Логин:

Пароль:

[Войти](#)

Рисунок 17 – В обоих случаях переадресация на страницу входа в систему

Отчеты ПО Среды исполнения Вы вошли как: user1

Имя	Иван
Фамилия	Иванович
Отчество	Иванович
Email	ivanov@mail.ru
Телефон	+71111111111
Дата рождения	14 мая 2023 г.

[Выйти](#)

Рисунок 18 - Информация отображается корректно

При нажатии кнопки «Выйти» пользователь выходит из системы.

Таким образом функциональность приложения пользователей была реализована в полной мере и ее работа соответствует требованиям.

### 3.2 Приложения ПО и сред выполнения



Создадим за Ивана объект ПО:

Отчеты ПО Среды исполнения Вы вошли как: user1

Категория:

Приложение

Название:

ПО Ивана

Версия:

1.0

Опции сборки:

-a -b --abc 0

Создать

Рисунок 19 – Заполнение формы ПО

Отчеты ПО Среды исполнения Вы вошли как: user1

**ПО N°1**

ПО Ивана	Приложение
Версия: 1.0	
Опции сборки:	-a -b --abc 0

Изменить Удалить

Рисунок 20 – Переадресация на корректную страницу с детальной информацией о созданном ПО

Попробуем изменить это ПО, войдя в систему как Петр:

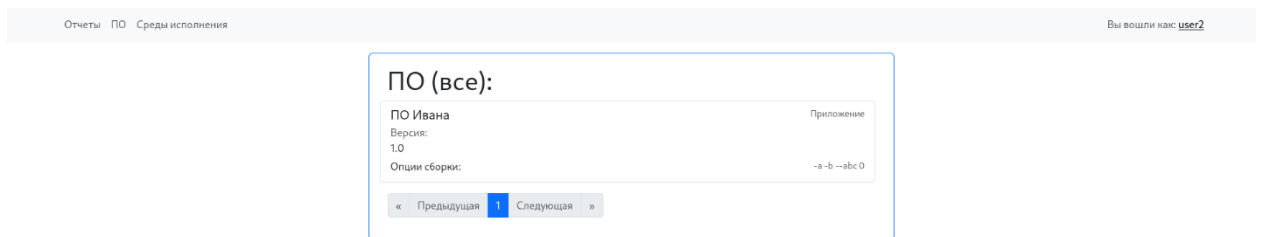


Рисунок 21 – Список всех ПО, который видит Петр

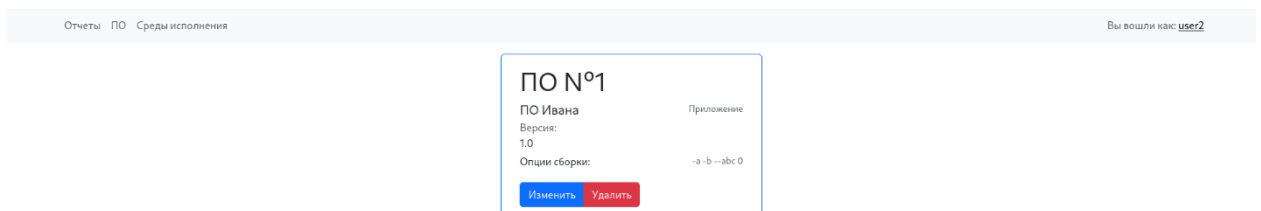


Рисунок 22 – Петру отображается корректное меню управления

Категория:

Приложение

Название:

ПО Ивана (изменено Петром)

Версия:

1.0

Опции сборки:

-a -b --abc 0

Сохранить

Рисунок 23 – Изменение ПО Петром

Нажимаем кнопку сохранить, наблюдаем результат:

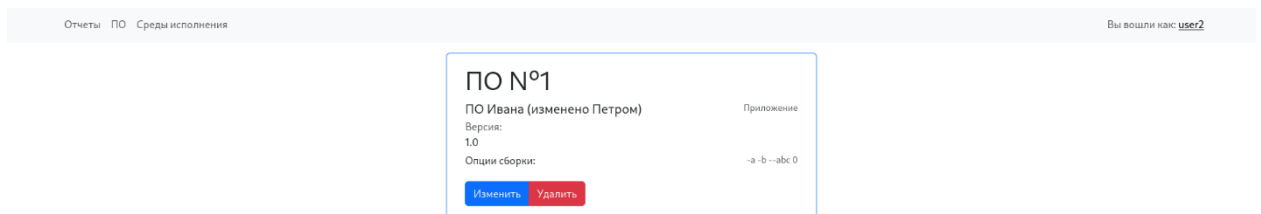


Рисунок 24 – Объект ПО был изменен

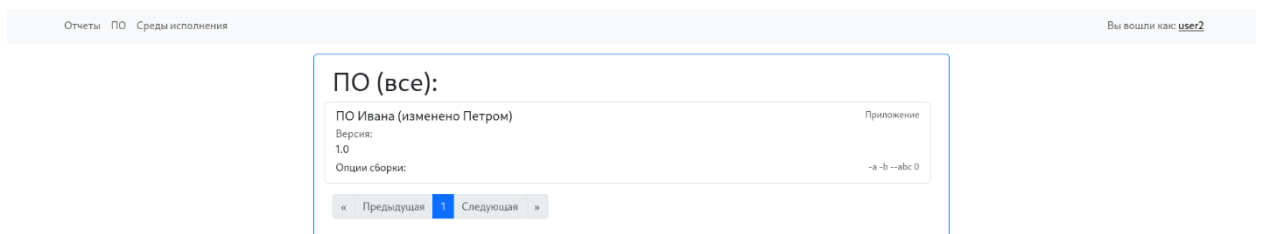


Рисунок 25 – Измененный объект ПО корректно отображается в списке

Аналогично создадим за Петра новую среду выполнения:

Отчеты ПО Среды исполнения

Вы вошли как: user2

ОС:

Windows

Версия:

10 (на компьютере Петра)

Создать

Рисунок 26 – Создание новой среды выполнения

Теперь выходим из системы и заходим обратно за Ивана, чтобы убедиться, что Ивану не хватает прав для изменения созданной среды выполнения:

Отчеты ПО Среды исполнения

Вы вошли как: user1

Среда выполнения №1

Windows

Версия:

10 (на компьютере Петра)

Рисунок 27 – Для Ивана меню управления корректно не отображается

Для достоверности перейдем также по прямой ссылке URL изменения среды выполнения:

Рисунок 28 – Отображается соответствующее сообщение об ошибке

Результатом данного тестирования является заключение о соответствии подсистем ПО и СВ требованиям.

### **3.3 Приложение отчетов**

После создания ПО и СВ можно создать аналогичным образом первый отчет:

Название:  
Первый отчет Ивана

Номер CWE:  
123

Описание:  
Описание отчета

ПО:  
1

Среды выполнения:  
Windows 10 (на компьютере Петра)

Эксплуатируемость:  
10

Опасность:

Рисунок 29 – Форма создания отчета

Отчет №1

Первый отчет Ивана CWE-123

Опасность: 10

Эксплуатируемость: 10

Описание:  
Описание отчета

Способ эксплуатации:  
Нет

ПО:  
[ПО Ивана \(изменено Петром\) 1.0 \(-a -b --abc 0\)](#)

Среды выполнения:  
[Windows 10 \(на компьютере Петра\)](#)

Входные файлы:  
[inputfile1.txt](#)

Дамп-файлы:  
[dumpfile1.bin\\_encoded](#)

Обновлено: 14 мая 2023 г. 21:36  
Создано: 14 мая 2023 г. 21:36

[Изменить](#) [Удалить](#)

Рисунок 30 – Созданный отчет

Создадим аналогичным образом второй отчет за Петра.

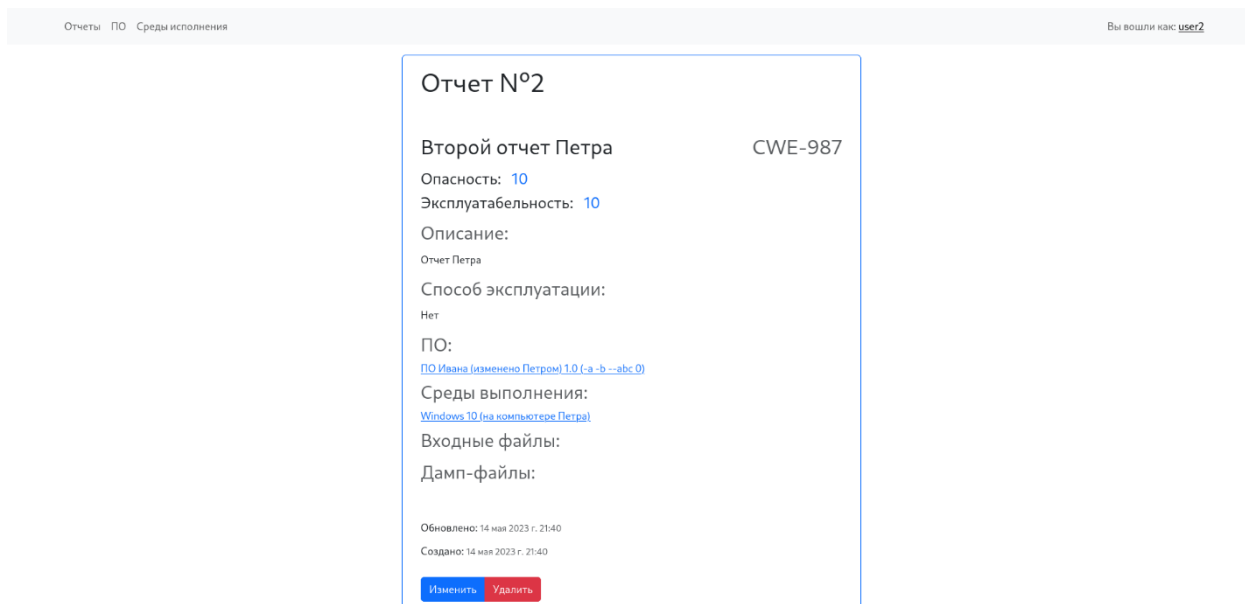


Рисунок 31 – Отчет Петра

Полный список отчетов:

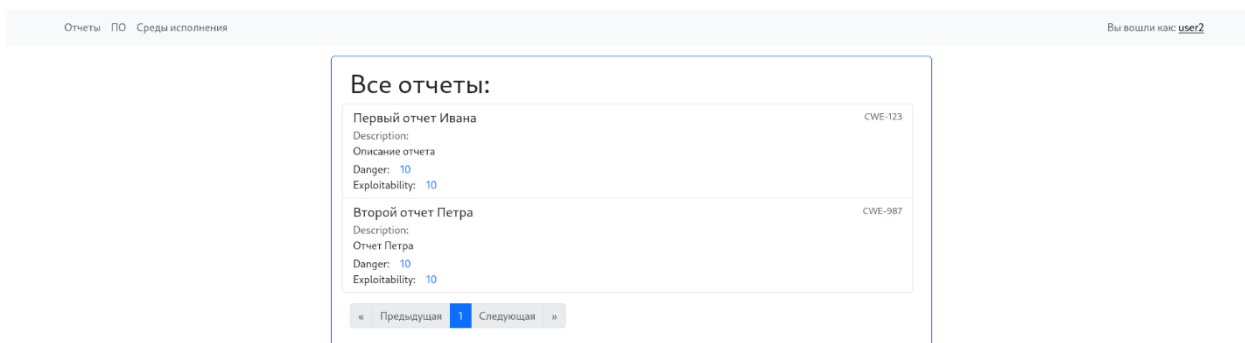


Рисунок 32 – Полный список отчетов

Таким образом возможно создать множество отчетов, поэтому необходимо иметь удобный способ искать необходимые отчеты. Осуществим поиск отчета Ивана:



Отчеты ПО Среды исполнения

Вы вошли как: user2

Название:  
Иван

Номер CVE:

ПО:

Эксплуатабельность (от):  
1

Эксплуатабельность (до):  
10

Опасность (от):  
1

Опасность (до):  
10

Искать

Рисунок 33 – Форма поиска отчетов (заполнены не все поля)

Отчеты ПО Среды исполнения

Вы вошли как: user2

Результаты поиска:

Первый отчет Ивана

CWE-123

Описание:  
Описание отчета  
Опасности: 10  
Эксплуатабельности: 10

« Предыдущая 1 Следующая »

Рисунок 34 – В результате был выдан необходимый отчет

Таким образом функциональность подсистемы отчетов также была реализована в полной мере.

## **ЗАКЛЮЧЕНИЕ**

В результате выполнения данной работы мною была разработана система для оптимизации рабочего процесса нашего отдела. Данная система предоставляет авторизованным пользователям возможности по просмотру, публикации и управлению отчетами. Благодаря разработанной системе сотрудники нашего отдела имеют возможность ускорить свой рабочий процесс и повысить эффективность своего труда. Для разработки данной системы мною был проведен анализ предметной области, спроектировано и реализовано итоговое решение.

## СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Число кибератак в России и мире [Электронный ресурс]. URL - [https://www.tadviser.ru/index.php/Статья:Число\\_кибератак\\_в\\_России\\_и\\_в\\_мире](https://www.tadviser.ru/index.php/Статья:Число_кибератак_в_России_и_в_мире) (дата обращения 12.05.2023)
2. Российские компании за год потеряли более 100 млрд руб. из-за кибератак [Электронный ресурс]. URL - [https://www.rbc.ru/technology\\_and\\_media/19/12/2017/5a38f3749a794710aa15581b](https://www.rbc.ru/technology_and_media/19/12/2017/5a38f3749a794710aa15581b) (дата обращения 12.05.2023)
3. Создание первого приложения [Электронный ресурс]. URL - <https://metanit.com/python/django/1.4.php> (дата обращения 12.05.2023)
4. Django documentation [Электронный ресурс]. URL - <https://docs.djangoproject.com/en/4.2/> (дата обращения 12.05.2023)
5. Википедия – свободная энциклопедия [Электронный ресурс]. URL - <https://ru.wikipedia.org/> (дата обращения 12.05.2023)
6. Что такое кибератака [Электронный ресурс]? URL - <https://www.microsoft.com/ru-ru/security/business/security-101/what-is-a-cyberattack> (дата обращения 12.05.2023)
7. Введение в Django [Электронный ресурс]. URL - <https://metanit.com/python/django/1.1.php> (дата обращения 12.05.2023)
8. CWE Top 25 2021. Что такое, с чем едят и чем полезен при статическом анализе [Электронный ресурс]? URL - <https://habr.com/ru/companies/pvs-studio/articles/580474/> (дата обращения 12.05.2023)
9. Docker: что это и как используется в разработке [Электронный ресурс]. URL - <https://tproger.ru/articles/chto-takoe-docker/> (дата обращения 12.05.2023)

10. Введение в WSGI-серверы: Часть первая [Электронный ресурс]. URL - <https://habr.com/ru/articles/426957/> (дата обращения 12.05.2023)
11. Файлы дампа в отладчике Visual Studio [Электронный ресурс]. URL - <https://learn.microsoft.com/ru-ru/visualstudio/debugger/using-dump-files?view=vs-2022> (дата обращения 12.05.2023)
12. Необычный Python в обычных библиотеках [Электронный ресурс]. URL - <https://habr.com/ru/companies/skillfactory/articles/683744/> (дата обращения 12.05.2023)
13. Как повысить безопасность приложений с помощью фабрик строк в psycorg [Электронный ресурс]. URL - <https://habr.com/ru/companies/ruvds/articles/690582/> (дата обращения 12.05.2023)
14. Python Programming - The State of Developer Ecosystem in 2021 Infographic | JetBrains: Developer Tools for Professionals and Teams [Электронный ресурс]. URL - <https://www.jetbrains.com/lp/devecosystem-2021/python/> (дата обращения 12.05.2023)
15. Compare Django and Flask [Электронный ресурс]. URL - <https://codeahoy.com/compare/django-vs-flask> (дата обращения 12.05.2023)
16. Django vs. Flask: Which Python Framework To Choose in 2023? [Электронный ресурс]. URL - <https://www.trio.dev/blog/django-vs-flask> (дата обращения 12.05.2023)
17. Flask vs Django: Which Python Web Framework to Use in 2023? [Электронный ресурс]. URL - <https://hackr.io/blog/flask-vs-django> (дата обращения 12.05.2023)

- 18.Django vs. Flask in 2023: Which Framework to Choose [Электронный ресурс]. URL - <https://testdriven.io/blog/django-vs-flask/> (дата обращения 12.05.2023)
- 19.Flask vs. Django — a comparison of the Python frameworks [Электронный ресурс]. URL - <https://www.ionos.com/digitalguide/websites/web-development/flask-vs-django/> (дата обращения 12.05.2023)
- 20.Django Vs Flask Vs Node: Which Framework To Select [Электронный ресурс]. URL - <https://www.softwaretestinghelp.com/django-vs-flask-vs-node> (дата обращения 12.05.2023)
- 21.9+ Difference In Flask vs Django: Which Framework Is Best? [Электронный ресурс]. URL - <https://www.calltutors.com/blog/flask-vs-django/> (дата обращения 12.05.2023)
- 22.Flask vs Django: Pirates use Flask, The Navy uses Django [Электронный ресурс]. URL - <https://www.imaginarycloud.com/blog/flask-vs-django> (дата обращения 12.05.2023)
- 23.Flask Versus Django: Which Python Framework Is Right for You? [Электронный ресурс]. URL - <https://mattermost.com/blog/flask-versus-django-which-python-framework-is-right-for-you/> (дата обращения 12.05.2023)