Федеральное государственное автономное образовательное учреждение высшего образования

«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет: Факультет информационных технологий

Кафедра «Информационная безопасность»

Направление подготовки/ специальность: 10.03.01 Информационная безопасность

ОТЧЕТ

по проектной практике

Студент: Махров Илья Михайлович Группа: 241-351

Место прохождения практики: Московский Политех, кафедра Информационная безопасность

Отчет принят с оценкой _____ Дата _____

Руководитель практики: Кесель С. А ., к.т.н., доцент кафедры «Информационная безопасность»

Москва 2025

ОГЛАВЛЕНИЕ

ОБЩАЯ ИНФОРМАЦИЯ О ПРОЕКТЕ

Проектная деятельность: Создание автоматизированной системы доставки.

Цель: Разработка эффективной автоматизированной системы для управления процессами доставки в компании-заказчике.

Проект решает ключевые проблемы логистики, повышает эффективность доставки и удовлетворенность клиентов, а также соответствует современным требованиям автоматизации.

Проблематика: Современные курьерские службы сталкиваются с проблемами, связанными с необходимостью повышения эффективности логистики, сокращения времени доставки контактными клиентами и минимизации затрат на систематическую корпоративную доставку. В условиях жесткой конкуренции и растущего спроса на оперативные и надежные услуги, традиционные методы управления доставкой становятся недостаточно гибкими и экономически невыгодными для коммерческих организаций, имеющие дела с государственной тайной, требующие определенных условий транспортировки декларации.

Заказчиком проекта является: АВН Бизнес

Описание задания по проектной практике

Учащиеся были посвящены проектной практике с комплексным заданием, которое касалось формирования навыков работы с современными системами разработки, управления проектами, а также взаимодействия со сторонними организациями. Внутри задания были выделены обязательная (базовая) и опциональная (вариативная) части, каждая из которых предполагала решение определенного количества задач.

Базовая часть

В базовую часть входило начальное проектирование системы контроля версий Git с репозиторием на GitHub или GitVerse по предложенному шаблону. Студентам необходимо было познакомиться или повторить основные команды Git: клонирование репозиториев, создание и фиксация коммитов, отправка изменений на удаленный сервер, работа с ветками. Прогресс работы необходимо было зафиксировать в регулярных промежутках с комментариями.

Важно было также оформить проектную документацию с использованием языка разметки Markdown. Документация кроме пояснительных записок и дневников “отчет” должна была включать записи о ходе и описание работы, накопленные ссылки и другие материалы.

Основной целью начальной части было разработать статический веб-сайт, посвященный проекту по дисциплине "Проектная деятельность". Для веб-сайта могли использоваться базовые языки разметки HTML и CSS, хотя советовалось использовать генератор статических сайтов Hugo для упрощения процесса. Предлагаемая структура сайта включала:

Главную страницу с аннотацией проекта,

Раздел "О проекте",

Раздел "Участники" с описанием вклада каждого студента,

Раздел "Журнал" с минимум тремя записями о ходе работы,

Раздел "Ресурсы" со ссылками на материалы от партнёрской организации

Внимание уделялось разработке сайта — оформление должно было быть не менее 50% оригинальным, как и графические и мультимедийные элементы, которые также должны были быть оригинальные.

Важным аспектом стало взаимодействие с организацией-партнёром: участие в профильных мероприятиях (конференциях, семинарах, мастер-классах, экскурсиях) и организация встреч или стажировок.

Итоговый отчёт о проделанной работе, включая описание полученного опыта и знаний, необходимо было оформить в PDF и DOCX форматах и разместить в репозитории и на сайте.

Вариативная часть

Вариативная часть проектной практики была направлена на углублённую проработку выбранной технической темы в области программирования и информационной безопасности. Студентам предлагалось выбрать одну из тематик для практической реализации, включая такие направления, как разработка собственного интерпретатора, HTTP-сервера, шаблонизатора и т.д.

В рамках проекта была выбрана тема **разработка собственного шаблонизатора на языке Python**. Основной задачей стало создание минималистичной системы шаблонов, поддерживающей подстановку переменных и базовые логические конструкции (if-блоки). Проект включал следующие этапы:

Исследование принципов работы шаблонизаторов и существующих решений (Jinja2)

Реализация собственной логики шаблонизации с нуля,

Разработка и тестирование кода,

Подготовка подробного технического руководства в формате Markdown с пошаговыми инструкциями, примерами использования и комментариями,

Визуализация архитектуры решения с помощью UML-диаграмм, схем и таблиц.

Все результаты исследования и исходный код шаблонизатора были размещены в общем репозитории проекта. Руководство по реализации оформлено в виде отдельного раздела на сайте проекта.

Работа над вариативной частью опиралась на предварительно изученные материалы, в том числе:

фреймворк **MITRE ATT&CK** — для понимания методов злоумышленников,

список **OWASP Top 10** — для анализа распространённых уязвимостей веб-приложений,

Это позволило осмыслить контекст проектной деятельности и связать полученные знания с практической реализацией выбранной темы.

ОПИСАНИЕ ДОСТИГНУТЫХ РЕЗУЛЬТАТОВ ПО ПРОЕКТНОЙ ПРАКТИКЕ

В самом начале практики было выдано общее, включающее в себя все «изучите и опишите главные аспекты матрицы - Mitre Att&ck», “изучите и опишите информацию с сайта OWASP” и “разберите инцидент, произошедший за последний год- полтора, с требованием расписать какие тактики, техники и процедуры были применены злоумышленниками.

В ходе выполнения задания, посвящённого изучению MITRE ATT&CK, OWASP и анализу реального инцидента, были достигнуты значимые результаты, которые позволили улучшить понимание современных угроз информационной безопасности и методов противодействия им. Изучение матрицы MITRE ATT&CK обеспечило систематизацию знаний о тактиках, техниках и процедурах (TTPs), используемых злоумышленниками на различных этапах кибератак. Это позволило не только классифицировать методы атак, такие как lateral movement, credential dumping или execution через вредоносные скрипты, но и научиться прогнозировать возможные векторы угроз в контексте конкретных инфраструктур.

Анализ материалов OWASP, включая актуальную версию OWASP Top 10, позволил получить чёткое понимание наиболее опасных уязвимостей веб-приложений, таких как различные виды инъекций, недостаточная защита данных, а также ошибки конфигурации. Эти знания легли в основу изучения принципов безопасной разработки. Особое внимание было уделено рекомендациям по снижению рисков (mitigations), среди которых — обязательная валидация пользовательского ввода и использование подготовленных выражений (prepared statements) для защиты от SQL-инъекций.

В дополнение к теоретической части был проведён разбор реального инцидента информационной безопасности, произошедшего в 2024–2025 годах. На его основе удалось проследить весь путь злоумышленника, от

начального проникновения до вывода данных. Были определены тактики атаки в соответствии с матрицей MITRE ATT&CK: например, доступ через фишинговые письма (Initial Access), повышение привилегий за счёт уязвимостей ПО (Privilege Escalation) и передача данных через зашифрованные каналы (Exfiltration). Для каждой из стадий были определены конкретные методы и инструменты, включая использование Cobalt Strike для удалённого управления.

Такой подход позволил не только воссоздать картину атаки, но и выделить ключевые этапы, на которых можно было бы своевременно обнаружить или остановить угрозу. Результаты работы оформлены в виде отчёта, который доступен ниже или в каталоге task0 папки task Git-репозитория. Примерное время, затраченное на выполнение этой части, составило около четырёх часов.

Mitre Att&ck

MITRE ATT&CK — это база знаний об известных тактиках, техниках, используемых злоумышленниками при кибератаках. Разрабатывается некоммерческой организацией MITRE и широко используется в информационной безопасности для моделирования угроз, выявления вторжений и обучения.

Mitre Att&ck состоит в виде матрицы где:

Столбцы — это тактики: цели злоумышленников (например, доступ, удержание, выполнение).

Строки — это техники: конкретные способы достижения этих целей.

Каждая техника включает описание, примеры, индикаторы компрометации, средства защиты и сопутствующие инструменты.

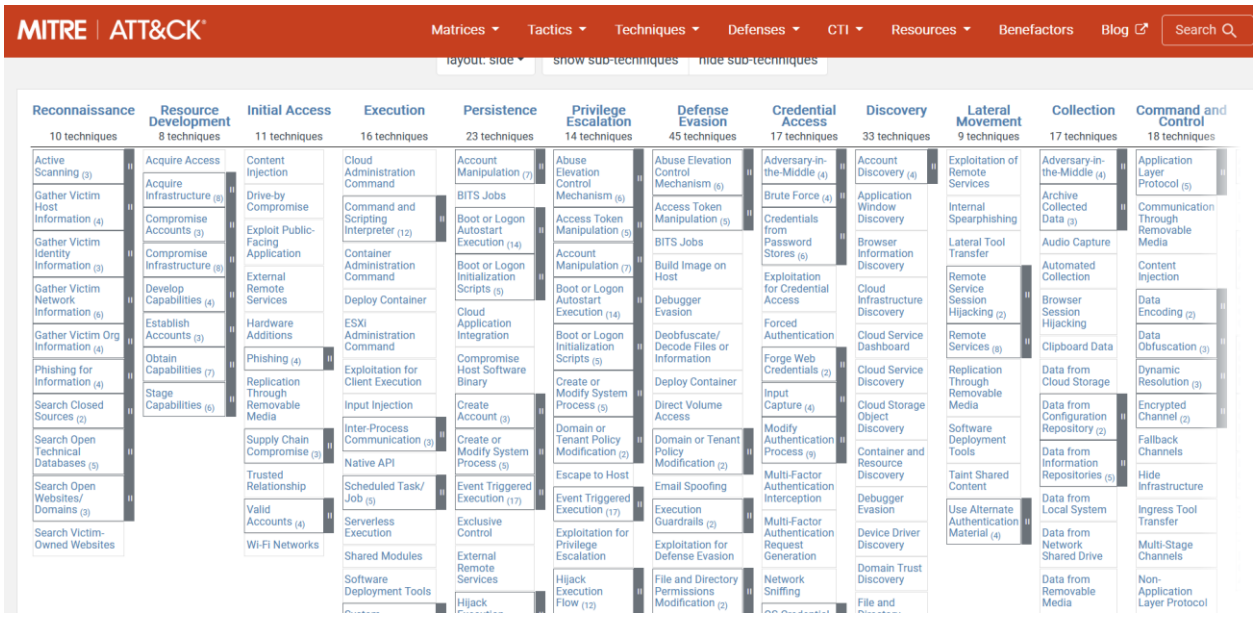

Рис. 1, неполная матрица MITRE ATT&CK

Вот некоторый из ключевых тактик в матрице mitre att&ck:

Тактика	Описание
Initial Access	Получение первоначального доступа (фишинг, эксплойты)
Execution	Выполнение вредоносного кода
Persistence	Удержание доступа после перезагрузки
Privilege Escalation	Повышение уровня привилегий
Defense Evasion	Обход защитных механизмов
Credential Access	Кража учетных данных
Discovery	Разведка внутри системы или сети
Lateral Movement	Перемещение между машинами в сети
Collection	Сбор данных

Exfiltration	Вывод данных за пределы организации
Command and Control	Связь с внешним сервером злоумышленника
Impact	Повреждение, шифрование или уничтожение данных

Виды матриц:

1) Enterprise ATT&CK — для корпоративных ОС: Windows, macOS, Linux.

2) Mobile ATT&CK — для мобильных устройств.

3) ICS ATT&CK — для промышленных систем управления.

Преимущества mitre att&ck:

1) Стандартизированная терминология.

2) Основано на реальных инцидентах.

3) Постоянно обновляется.

4) Поддержка сообщества.

OWASP

OWASP — это некоммерческая международная организация, цель которой — повышение уровня безопасности ПО (программного обеспечения). Проект основан на принципах открытости, доступности и сообщества. Все ресурсы OWASP (документы, инструменты, обучающие материалы) доступны бесплатно.

OWARP TOP 10:

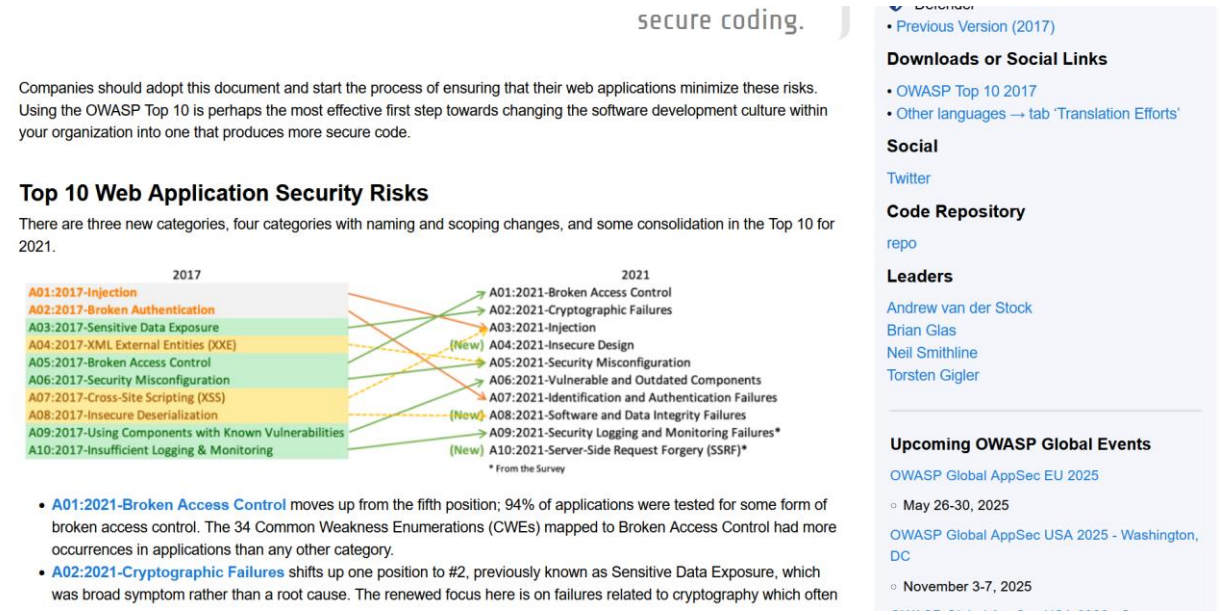

Рис. 2, Отчет OWASP Top-10

A01:2021 – Broken Access Control (Нарушение управления доступом)

Описание: Неправильная настройка прав доступа позволяет злоумышленникам получить доступ к данным или функциям, которым они не должны обладать.

A02:2021 – Cryptographic Failures (Криптографические ошибки)

Описание: Неправильное использование или отсутствие шифрования, уязвимости в конфигурации SSL/TLS, слабые алгоритмы.

A03:2021 – Injection (Инъекции)

Описание: Внедрение вредоносных данных в команды или запросы, обрабатываемые интерпретатором.

A04:2021 – Insecure Design (Небезопасный дизайн)

Описание: Архитектурные слабости, отсутствие модели угроз или безопасности на этапе проектирования.

A05:2021 – Security Misconfiguration (Ошибки конфигурации)

Описание: Небезопасные или стандартные настройки серверов, приложений, СУБД и т.д.

A06:2021 – Vulnerable and Outdated Components (Уязвимые и устаревшие компоненты)

Описание: Использование устаревших библиотек, фреймворков и зависимостей с известными уязвимостями.

A07:2021 – Identification and Authentication Failures (Ошибки аутентификации)

Описание: Недостатки в процессах идентификации пользователей и управления сессиями.

A08:2021 – Software and Data Integrity Failures (Нарушение целостности ПО и данных)

Описание: Отсутствие проверки целостности, доверия к внешним источникам или возможности внедрения вредоносных обновлений.

A09:2021 – Security Logging and Monitoring Failures (Пробелы в введение журнала и мониторинге)

Описание: Отсутствие введения журнала, оповещений и мониторинга событий безопасности.

A10:2021 – Server-Side Request Forgery (SSRF)

Описание: Злоумышленник заставляет сервер отправлять запросы к внутренним ресурсам, к которым клиент не имеет доступа.

Итог:

Риск	Суть	Защита
A01	Контроль доступа	Проверка авторизации
A02	Криптография	Современные алгоритмы
A03	Инъекции	Подготовленные запросы
A04	Проектирование	Моделирование угроз
A05	Конфигурация	Безопасные настройки
A06	Зависимости	Актуальность компонентов
A07	Аутентификация	MFA, безопасные токены

A08	Целостность	Подписи, контроль цепочек
A09	Введение журнала	SIEM, мониторинг
A10	SSRF	Фильтрация и контроль запросов

Другие важные проекты OWASP:

1) OWASP SAMM (Software Assurance Maturity Model)

Модель зрелости процессов безопасности разработки.

Позволяет организациям оценивать и развивать безопасность SDLC.

2) OWASP Web Security Testing Guide

Подробное руководство по тестированию безопасности веб-приложений.

Включает фазы тестирования, чек-листы.

3) OWASP ASVS (Application Security Verification Standard)

Стандарт для проверки уровня безопасности приложений.

Содержит уровни: Level 1 (минимальный), Level 2 (стандарт), Level 3 (высокая защита).

Используется для аудитов и разработки требований безопасности.

4) OWASP Dependency-Check

Инструмент для анализа зависимостей.

Помогает выявлять уязвимости в сторонних библиотеках.

Инцидент

В качестве разобранного инцидента я решил взять инцидент, случившийся 21 февраля 2025 года, связанный с криптовалютной биржей Bybit.

В феврале 2025 года криптовалютная биржа Bybit, базирующаяся в Дубае, подверглась крупнейшей в истории кибератаке, в результате которой было похищено около $1,5 миллиарда в Ethereum (ETH). За атакой стоит северокорейская хакерская группа Lazarus, известная своими высокотехнологичными киберпреступлениями.

1. Initial Access (TA0001) – Первичный доступ

T1195.002 – Supply Chain Compromise: Software Dependencies and Development Tools

Lazarus внедрили вредоносный компонент в систему обновления или библиотеку, используемую в смарт-контракте, связанном с движением средств между кошельками. Это классический вектор через цепочку поставок.

T1078 – Valid Accounts

В некоторых отчетах упоминается возможность использования легитимных учетных данных одного из DevOps-инженеров. Возможно, их украли через фишинг.

2. Execution (TA0002) – Исполнение

T1203 – Exploitation for Client Execution

Код был исполнен в среде обработки транзакций — возможно через вредоносный CI/CD скрипт, запускаемый при подписании транзакции. Это мог быть hook или сторонний плагин.

T1059.006 – Command and Scripting Interpreter: JavaScript

Если был внедрён вредоносный скрипт в смарт-контракт или CI-инфраструктуру, он мог быть написан на JS/TypeScript.

3. Defense Evasion (TA0005) – Уклонение от защиты

T1222.002 – File and Directory Permissions Modification: Linux and Mac File Permissions

Изменение прав доступа к хранилищу ключей или кошельков для сокрытия следов.

4. Credential Access (TA0006) – Доступ к учетным данным

T1555.003 – Credentials from Password Stores

Возможен доступ к хранилищу ключей.

T1110.003 – Brute Force: Password Spraying

Если доступ был получен через административную панель, возможна атака перебором общих паролей.

5. Persistence (TA0003) – Закрепление

T1543.003 – Create or Modify System Process: Windows Service / Linux Daemon

Вредоносный сервис, запускаемая при каждой транзакции или CI-сборке.

T1053.003 – Scheduled Task/Job: Cron

Автоматическая активация вредоносного скрипта в определённый момент — например, при переводе средств между кошельками.

6. Command and Control (TA0011) – Управление

T1071.001 – Application Layer Protocol: Web Protocols

Управляющие сигналы (например, подтверждение момента атаки) могли передаваться через HTTPS-запросы к командному серверу Lazarus.

7. Impact (TA0040) – Влияние/Нарушение

T1496 – Resource Hijacking

В данном случае — перехват и переадресация средств на адрес хакеров.

T1485 – Data Destruction

После перевода ETH — возможная очистка следов и уничтожение улик (журналов, временных файлов).

Lazarus продемонстрировал высокий уровень технической подготовки и тактической координации.

Базовая часть

После завершения общей части проекта была реализована базовая часть, которая охватывала несколько ключевых направлений: настройку системы контроля версий Git, оформление документации в формате Markdown и создание статического веб-сайта. Эти задачи позволили закрепить навыки работы с инструментами разработки, системами управления версиями и подготовки технической документации, а также привести к созданию законченного цифрового продукта — сайта проекта.

Настройка Git и репозитория

Была организована полноценная работа с Git: создан удалённый репозиторий на GitHub, освоены основные команды — клонирование, работа с ветками, коммиты с понятными комментариями, отправка изменений на сервер. Постоянные фиксации прогресса позволили отследить этапы разработки, а ветвление помогло разграничить задачи между участниками. Репозиторий стал основным хранилищем проекта, включающим в себя код, документацию и отчётные материалы.

Время выполнения: ~4 часа

Оформление документации в Markdown

Проектная документация была составлена с применением синтаксиса Markdown, что обеспечило её логичную структуру и удобочитаемость. Были изучены и использованы различные элементы форматирования: заголовки, списки, таблицы, ссылки, изображения. Подготовлены описания проекта, технические характеристики, инструкции и прочие материалы. Вся документация размещена в репозитории для общего доступа и последующего сопровождения.

Время выполнения: ~1 час

Разработка статического веб-сайта

В рамках задания был разработан статический веб-сайт, отражающий содержание проекта. Основными технологиями стали HTML и CSS. Структура сайта включает следующие разделы:

Главная страница с кратким описанием проекта;

Раздел «О проекте» с раскрытием целей и задач;

Страница «Участники» с описанием индивидуального вклада;

«Журнал» с тремя записями, отражающими ключевые этапы разработки;

Раздел «Ресурсы» с полезными ссылками и материалами от партнёров.

Сайт оформлен с учётом требований к уникальности дизайна — добавлены изображения и медиаэлементы. Готовый проект размещён в репозитории.

Время выполнения: ~8 часов

Взаимодействие с организацией-партнёром

В рамках партнёрской части проекта было организовано участие в профессиональных мероприятиях: посещена экскурсия в ООО “НИАС” 2 часа.

Полученные навыки и результаты

Уверенное владение инструментами контроля версий (Git, GitHub), в том числе работа с ветками и pull-request;

Опыт составления технической документации в Markdown-формате;

Практические знания в области фронтенд-разработки: вёрстка, HTML, CSS, размещение сайта и управление исходным кодом.

Результаты выполнения базовой части стали прочной основой для дальнейшей реализации вариативной задачи — разработки собственного

шаблонизатора, а также подтвердили способность эффективно использовать современные ИТ-инструменты и соблюдать сроки выполнения проекта.

Вариативная часть

В рамках выполнения вариативной части проекта была реализована собственная система шаблонизации, позволяющая динамически формировать текстовые отчёты на основе пользовательских данных. Основной задачей стало проектирование и реализация компактного шаблонизатора на Python, поддерживающего как подстановку значений переменных, так и условные конструкции.

Разработка началась с формирования общей структуры шаблонизатора. На начальном этапе была реализована обработка шаблонов с использованием плейсхолдеров {{ ключ }}, заменяемых на соответствующие значения из переданного словаря. Далее функциональность была расширена за счёт поддержки условных блоков {{ if условие }}...{{ endif }}, где наличие или отсутствие текстовых фрагментов определялось логическими значениями в контексте.

Шаблонизатор был протестирован на учебном сценарии, предполагающем ввод имени пользователя и числового результата. В зависимости от значения результата выводилось сообщение о сдаче или несдаче теста, а также выставлялась текстовая оценка ("Отлично", "Хорошо", "Удовлетворительно" или "Неудовлетворительно"). Кроме того, шаблон позволял продемонстрировать подстановку дополнительных данных — тематических тегов, связанных с проектом.

Функциональность шаблонизатора была оформлена в виде класса SimpleTemplateEngine. Итоговый текст формировался методом .render(), в который передавался словарь с данными. Результат отображался в консоли и записывался в файл output.txt. Таким образом, проект позволил продемонстрировать применение регулярных выражений, работу с файловым вводом-выводом, а также реализовать базовые приёмы генерации отчётной документации на основе шаблонов.

Реализация данного шаблонизатора внесена в репозиторий проекта и может служить основой для расширения (например, добавления циклов, вложенных условий и пользовательских фильтров). Выполнение задания заняло 6 часов.

Особое внимание было уделено удобству использования и читаемости кода. Приложены примеры использования. В качестве демонстрации работоспособности был составлен шаблон, выводящий приветствие со вписанным именем, ваш результат + оценка и список тегов.

Проект размещён в репозитории с README-документацией, примерами.

Полученный опыт позволил лучше понять, работы с шаблонами и обработки условий. В дальнейшем проект может быть расширен дополнительными возможностями (напр., поддержкой фильтров, шаблонных блоков или безопасной экранизацией).

(Общее время выполнения — ~60 часов)

ЗАКЛЮЧЕНИЕ

В ходе выполнения проектной практики был реализован комплексный проект, сочетающий исследовательскую и техническую составляющие. Основной акцент работы был сделан на изучение современных технологий защиты веб-приложений. Практическая часть включала глубокий анализ актуальных угроз информационной безопасности через призму методологий MITRE ATT&CK и OWASP Top 10, что позволило систематизировать знания о современных векторах атак и методах защиты. Имело большую ценность разбор реально инцидента, который наглядно показал тактики злоумышленников.

Техническая реализация проекта: Разработка собственного шаблонизатора стала важной частью проектной деятельности, позволившей закрепить навыки работы с текстовыми шаблонами, логикой обработки данных и реализацией элементарной DSL (domain-specific language) внутри Python-программы. Несмотря на простоту реализации, проект продемонстрировал, как даже базовые инструменты могут эффективно решать задачу генерации структурированного контента — от отчётов до уведомлений — с учётом динамически изменяющихся данных.

Важным действием для углубления в сферу информационной безопасности стало участие в экскурсии. Полученные результаты демонстрируют, повышение уровня цифровой грамотности.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. **MITRE ATT&CK®** [Электронный ресурс]. – Режим доступа: https://attack.mitre.org/ (дата обращения: 21.04.2025).
2. **OWASP Foundation** [Электронный ресурс]. – Режим доступа: https://owasp.org/ (дата обращения: 21.04.2025).
3. **GitHub Docs** [Электронный ресурс]. – Режим доступа: https://docs.github.com/ (дата обращения: 21.04.2025).
4. **Markdown Guide** [Электронный ресурс]. – Режим доступа: https://www.markdownguide.org/ (дата обращения: 13.05.2025).
5. **OWASP Top 10:2021** [Электронный ресурс]. – Режим доступа: https://owasp.org/www-project-top-ten/ (дата обращения: 21.04.2025).
6. **Jinja2 шаблонизатор** [Электронный ресурс]. – Режим доступа: https://jinja.palletsprojects.com/ (дата обращения: 13.05.2025).