1. Объект испытаний

1.1 Наименование
Рекомендательная система с защитой от сетевых атак в финансовом секторе.

1.2 Область применения

- Банковские рекомендации (кредиты, инвестиции, страхование).
- Логирование трафика в JSON для анализа.

1.3 Обозначение - recommendation_system
Используемые технологии:

- Nginx (прокси, логирование).
- Flask (бэкенд рекомендаций).
- MySQL (хранение данных).
- Fail2ban

2. Цель испытаний

2.1 Основная цель
Проверка:

- Работоспособности системы (запуск, взаимодействие между контейнерами).
- Соответствия требованиям ТЗ (HTTPS, JSON-логи, защита от атак).
- Надежности (работа под нагрузкой, восстановление после сбоев).
- Безопасности (предотвращение SQL-инъекций, XSS, DDoS).

2.2 Дополнительные цели

- Проверка возможности расширения (интеграция WAF, Redis, мониторинг).
- Подтверждение корректного логирования и хранения данных.

2.3 Задачи испытаний

- Проверка функциональности:
    - Работа Nginx как прокси.
    - Поиск рекомендаций через **?q=**.
    - Хранение данных в MySQL.
- Тестирование безопасности:
    - Защита от DDoS, SQL-инъекций, XSS.
    - Логирование подозрительных действий.

3. Требования к программе

3.1 Функциональные требования

- Nginx: прокси, логирование в JSON, шифрование TLS 1.2+.

- Flask: безопасный поиск через ORM (SQLAlchemy), генерация HTML.
- MySQL: хранение данных, инициализация через **init.sql**.
- Fail2ban: блокировка IP после 50+ запросов за 60 секунд.

3.2 Требования к надежности

- Docker Compose: автоматическое восстановление контейнеров (**restart: unless-stopped**).
- Изоляция: контейнеры работают в сети **secure-network**.
- Логирование: все данные доступны для анализа в **access.log**.

3.3 Требования к безопасности

- HTTPS: шифрование данных через Nginx.
- Защита от SQL-инъекций: использование SQLAlchemy (ORM).
- Защита от XSS: экранирование входных данных.
- Fail2ban: блокировка IP при частых запросах.

3.4 Требования к расширяемости

- Возможность добавления WAF (ModSecurity).
- Интеграция с Redis для кэширования.
- Поддержка мониторинга (Prometheus, Grafana)

4. Требования к программной документации

4.1 Соответствие ГОСТ

- Пояснительная записка (ГОСТ 19.404-79): описание архитектуры, целей, задач.
- Руководство программиста (ГОСТ 19.504-79): структура проекта, сборка, тестирование.
- Описание программы (ГОСТ 19.402-78): логическая структура, технические средства, вызов программы.
- Текст программы (ГОСТ 19.401-78): исходный код, Dockerfile, **docker-compose.yml**.

4.2 Полнота и структура

- Все документы должны быть оформлены по ГОСТ:

4.3 Доступность

- Документация должна быть в одном репозитории (GitHub)

5. Состав и порядок испытаний

1. Предварительная проверка:
    - Структура проекта: docker-compose.yml, Dockerfile, app.py, nginx.conf, init.sql.

- Документация: пояснительная записка, руководство программиста, описание программы.

2. Функциональное тестирование:
    - Запуск: docker-compose up -d.
    - Проверка работы: curl -k https://localhost.
    - Поиск: curl -k "https://localhost/search?q=Card".
3. Тестирование безопасности:
    - DDoS: for i in {1..100}; do curl -k "https://localhost"; done.
    - SQL-инъекции: curl -k "https://localhost/search?q=1' OR '1'='1".
    - XSS: curl -k "https://localhost/search?q=<script>alert(1)</script>".
    - Логирование: cat ./nginx-proxy/logs/access.log.
4. Проверка документации:
    - Соответствие ГОСТ 19.404-79, 19.504-77, 19.402-78, 19.401-78.
    - Полный список документов: README, пояснительная записка, руководство программиста, описание программы, текст программы, презентация

6. Методы испытаний

6.1 Инструменты

- Docker Engine 24.0+ → сборка и запуск контейнеров.
- Docker Compose 2.0+ → оркестрация.
- curl → тестирование запросов.
- MySQL CLI → проверка данных.
- fail2ban-regex → анализ логов.
- iptables → проверка блокировки IP)

6.2 Подходы к тестированию

Запуск системы - **docker-compose up -d**,**docker ps**

Работа Nginx - **curl -k https://localhost**,**docker logs proxy**

Поиск рекомендаций - curl -k "https://localhost/search?q=Card"

DDoS-защита - for i in {1..100}; do curl -k "https://localhost"; done

SQL-инъекции - curl -k "https://localhost/search?q=1' OR '1'='1"

XSS-атаки - curl -k https://localhost/search?q=<script>alert(1)</script>

Логирование - cat ./nginx-proxy/logs/access.log

Блокировка IP - docker exec -it fail2ban fail2ban-client status nginx

6.3 Методы анализа результатов

- Визуальный осмотр: проверка наличия файлов, структуры проекта, README.
- Функциональное тестирование: **curl**, **docker ps**, **docker logs**.
- Безопасность: анализ логов, проверка блокировки IP, тестирование атак.
- Документация: проверка соответствия ГОСТ, полноты описания