|  |  |  |
| --- | --- | --- |
|  |  | УТВЕРЖДАЮ |
|  |  | Руководитель образовательной программы  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сергеева Е. Г.  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.  М.П. |

**ТЕХНИЧЕСКИЙ ПРОЕКТ**

на разработку

«Система автоматизированного тестирования безопасности веб-приложений»

|  |  |  |
| --- | --- | --- |
| СОГЛАСОВАНО |  | СОГЛАСОВАНО |
| Колледж ВятГУ  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Вершинина Н. Д.  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.  М.П. |  | Руководитель УП  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Долженкова М. Л.  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

2025

СОДЕРЖАНИЕ

[1. Функциональная структура 3](#_Toc211445631)

[1.1. Функциональное моделирование в виде диаграммы IDEF0 3](#_Toc211445632)

[1.2. Диаграмма потоков данных 6](#_Toc211445633)

[2. Спецификация процессов 7](#_Toc211445634)

[2.1. Диаграмма вариантов использования 7](#_Toc211445635)

[2.2. Диаграмма последовательности 8](#_Toc211445636)

[2.3. Диаграмма состояний 9](#_Toc211445637)

[2.4. Диаграмма деятельности 10](#_Toc211445638)

[3. Структура хранимой информации 11](#_Toc211445639)

[4. Архитектурно-структурное решение 12](#_Toc211445640)

[5. Особенности взаимодействия с пользователем и макеты интерфейса 15](#_Toc211445641)

[5.1. Общие принципы взаимодействия 15](#_Toc211445642)

[5.2. Особенности взаимодействия пользователя с системой 15](#_Toc211445643)

[5.3. Макеты интерфейса 17](#_Toc211445644)

1. Функциональная структура
   1. Функциональное моделирование в виде диаграммы IDEF0

Функциональная модель системы представлена в виде диаграммы IDEF0 (уровень A-0), отражающей основные бизнес-процессы, задействованные при автоматизированном тестировании веб-приложений.

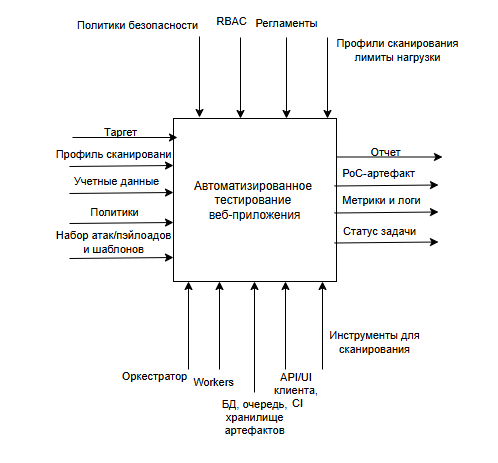


Рисунок 1 – IDEF0 0-го уровня

Система «Автоматизированное тестирование веб-приложения» состоит из входов:

* таргеты;
* профиль сканирования;
* учетные данные;
* политик;
* набор атак/пэйлоадов и шаблонов.

Выходы:

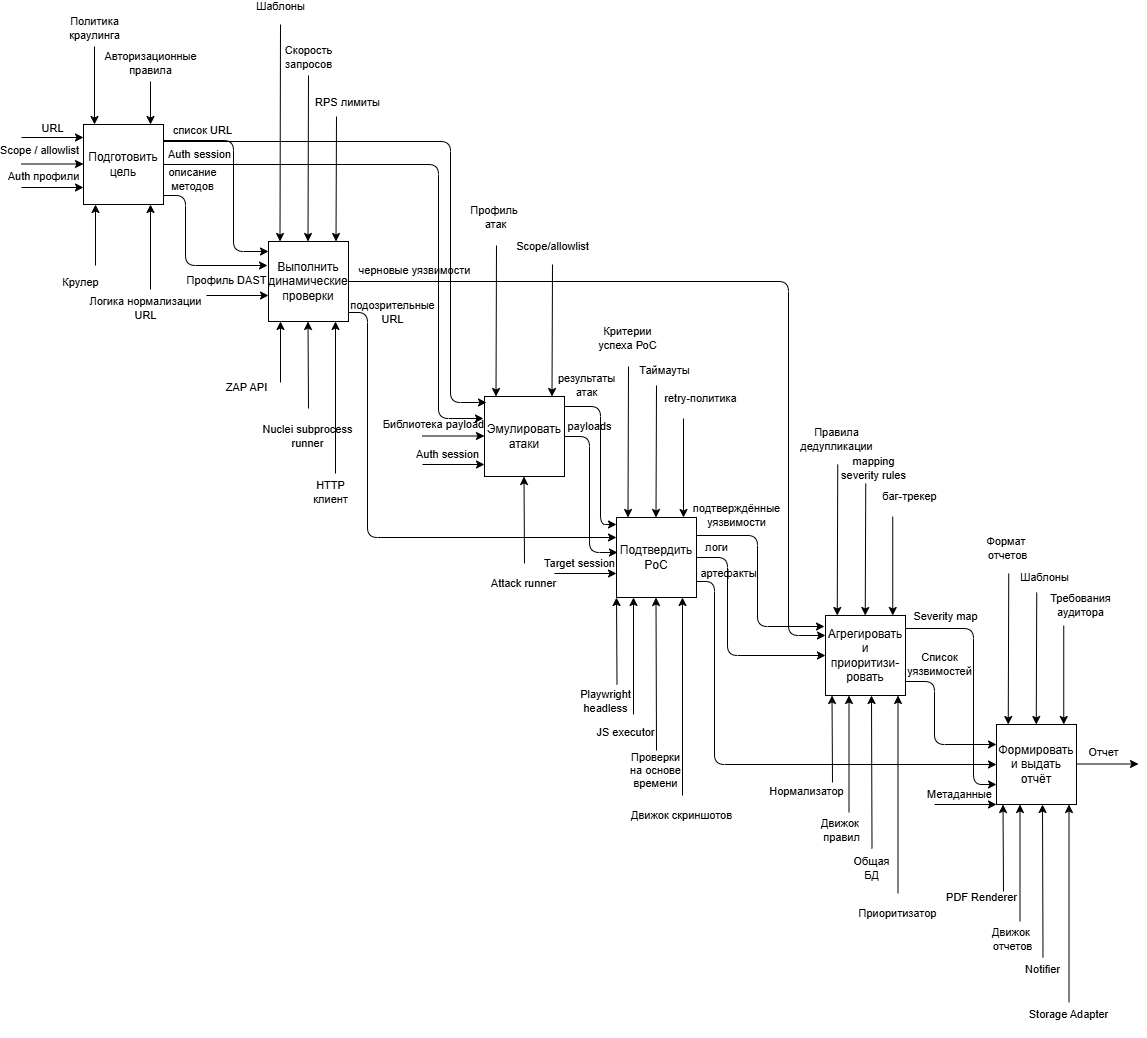
* отчет;
* РоС-артефакт;
* метрики и логи;
* статус задачи.

Управления:

* политики безопасности;
* RBAC;
* регламенты;
* профили сканирования лимиты нагрузки.

Механизмы:

* оркестратор;
* workers;
* БД, очередь, хранилище артефактов;
* API/UI клиента и CI;
* инструменты для сканирования.

Рисунок 2 – Декомпозиция IDEF0

Декомпозиция состоит из 6 подпроцессов:

* подготовить цель;
* выполнить динамическое проверки;
* эмулировать атаки;
* подтвердить РоС;
* агрегировать и приоритизировать;
* сформировать и выдать отчет.

Каждый подпроцесс показывает входы, выходы, управления и механизмы.

* 1. Диаграмма потоков данных

Диаграмма потоков данных (DFD), представленная на рисунке 3, демонстрирует движение информации между основными компонентами системы в процессе выполнения сканирования.

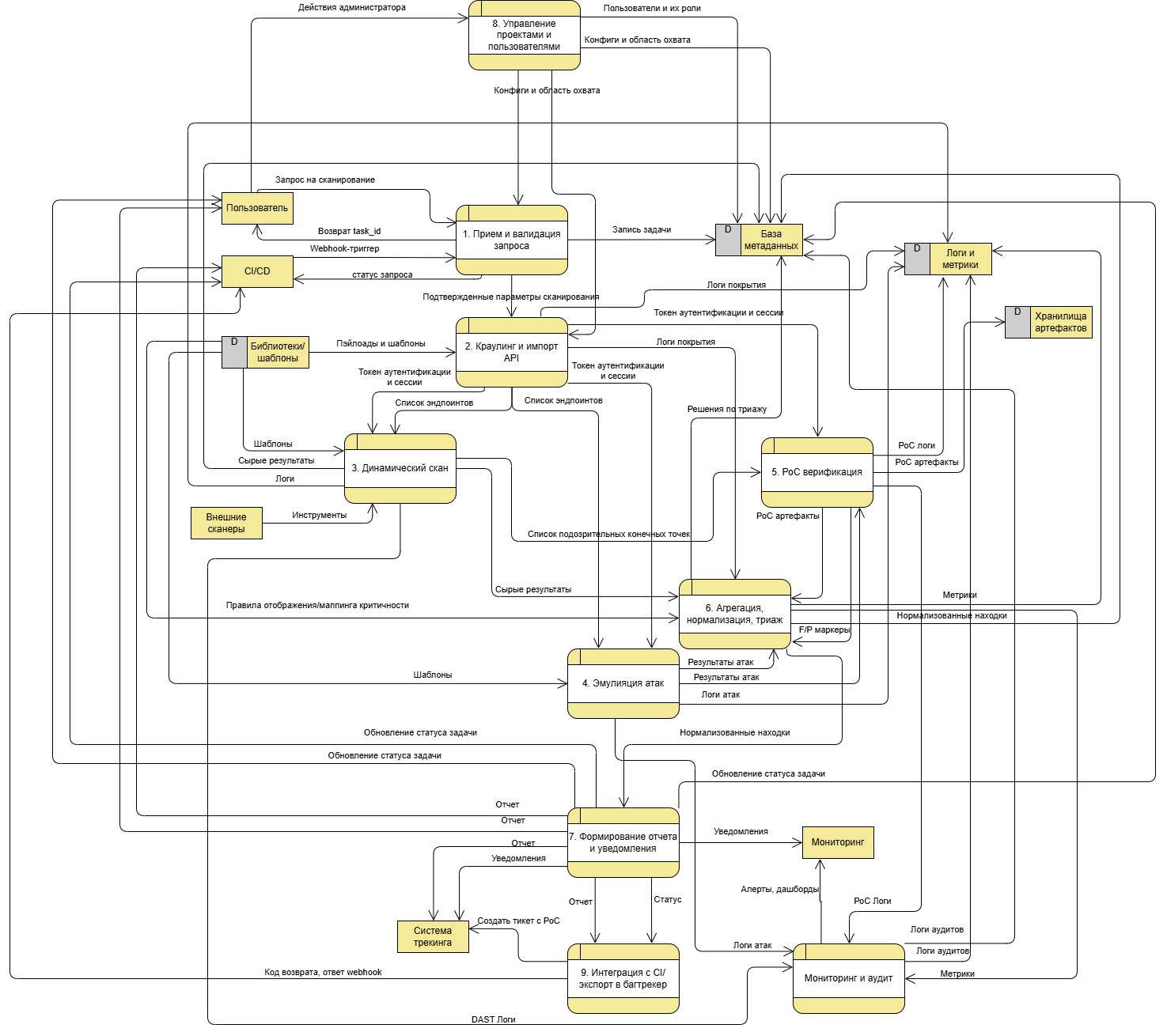


Рисунок 3 – Диаграмма DFD

1. Спецификация процессов
   1. Диаграмма вариантов использования

Диаграмма вариантов использования (рисунок 4) отражает взаимодействие системы с различными типами пользователей. Она демонстрирует, какие действия доступны каждому участнику и каким образом осуществляется доступ к функциональности.

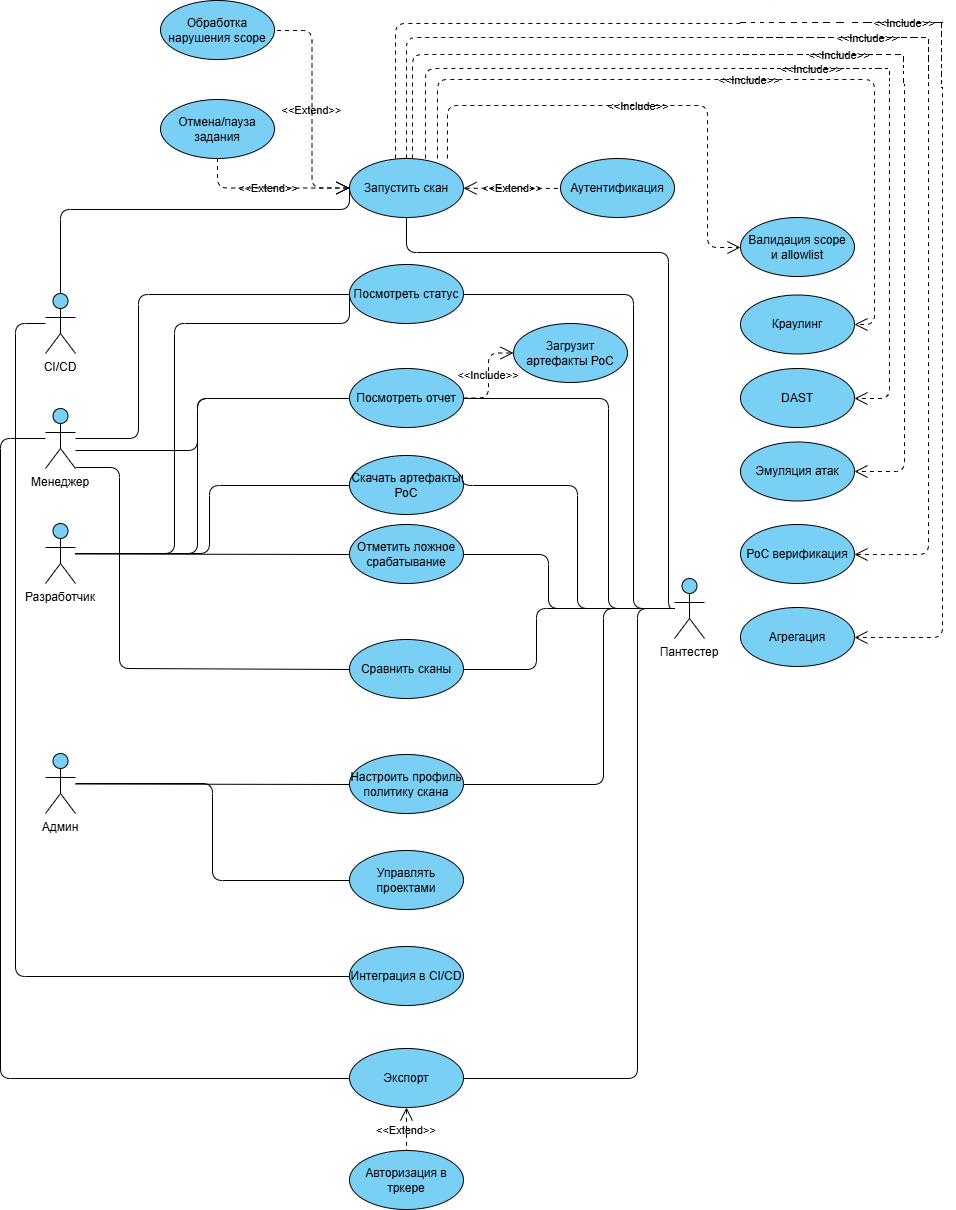


Рисунок 4 – Диаграмма Use Case

* 1. Диаграмма последовательности

Диаграмма последовательности (рисунок 5) показывает детализированный сценарий взаимодействия между подсистемами на этапе запуска и выполнения сканирования.



Рисунок 5 – Диаграмма последовательности

* 1. Диаграмма состояний

Диаграмма состояний (рисунок 6) отображает жизненный цикл задачи сканирования.

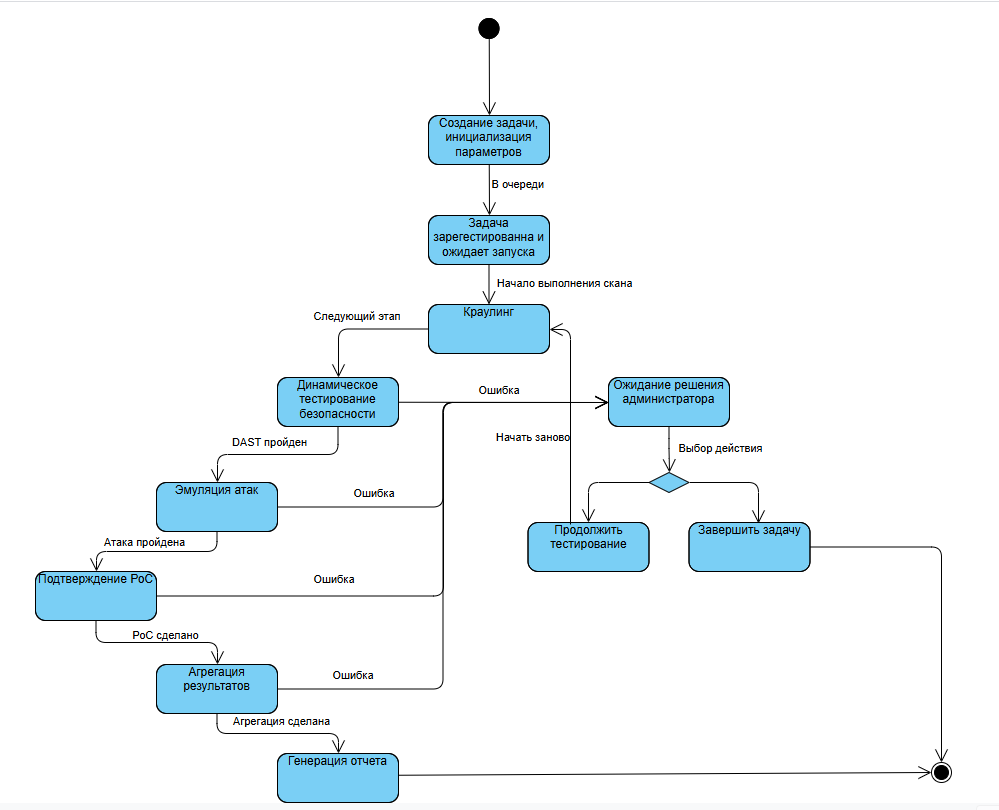


Рисунок 6 – Диаграмма состояний

* 1. Диаграмма деятельности

Диаграмма деятельности (рисунок 7) описывает алгоритмическую последовательность действий при запуске сканирования.

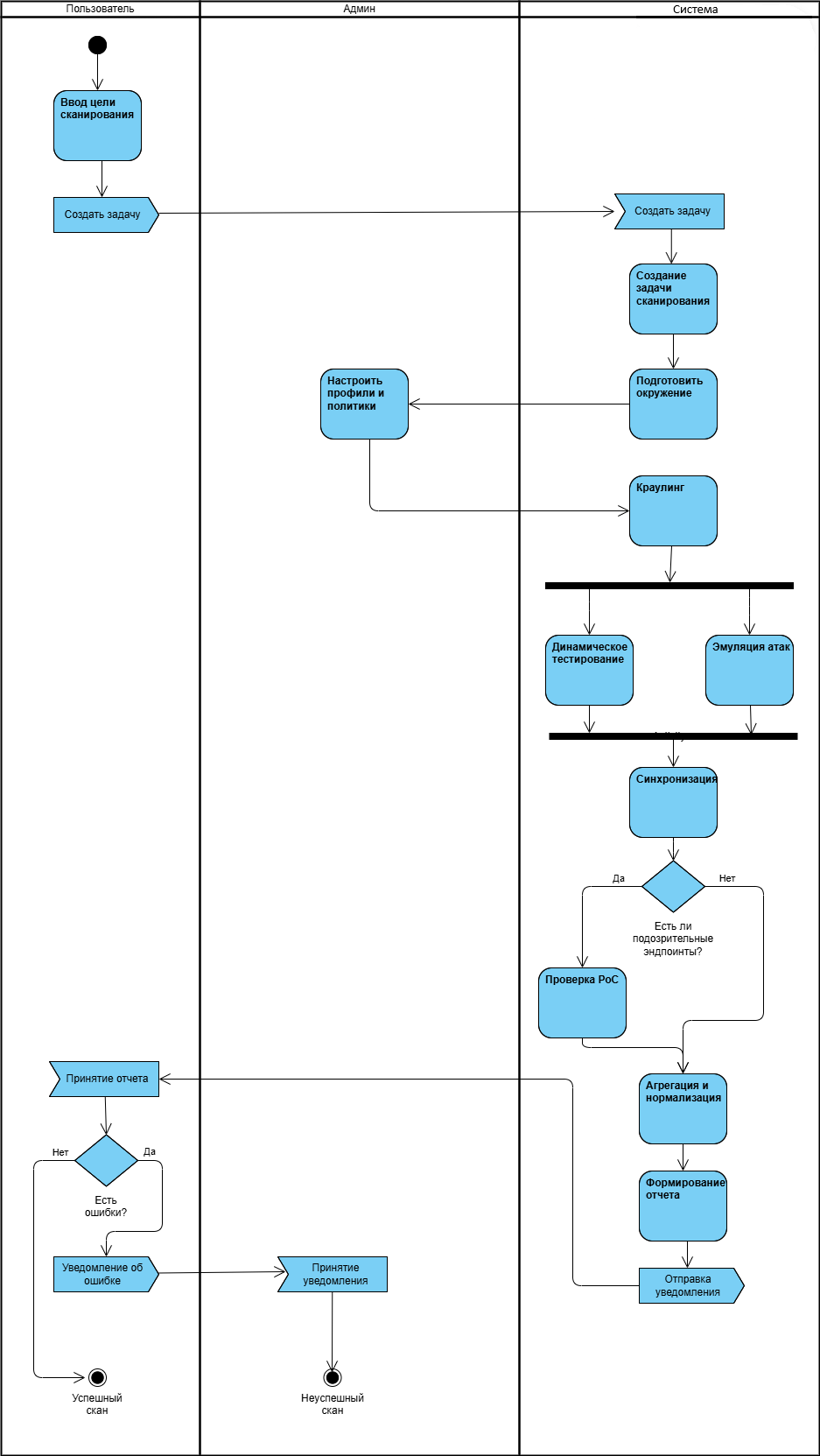


Рисунок 7 – Диаграмма деятельности

1. Структура хранимой информации

ER-диаграмма (рисунок 8) отражает основные сущности системы и связи между ними. На рисунке сформирована основа для проектирования базы данных и описано, какие объекты подлежат хранению, как они взаимосвязаны и какие атрибуты являются ключевыми.

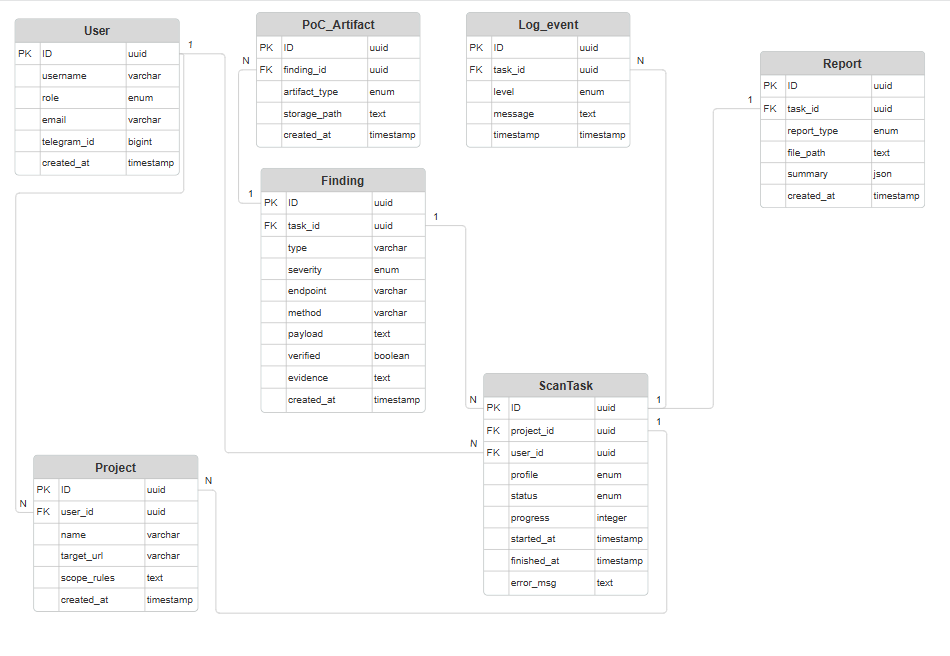


Рисунок 8 – ER диаграмма

1. Архитектурно-структурное решение

Система строится по принципу модульной (микросервисной) архитектуры с разделением на независимые компоненты, взаимодействующие по REST API и очередям сообщений.

Такой подход обеспечивает:

* устойчивость к сбоям;
* возможность горизонтального масштабирования;
* независимое обновление отдельных модулей (краулинг, DAST, эмуляция атак, отчётность).

Таблица 1 – основные компоненты системы.

|  |  |  |
| --- | --- | --- |
| Подсистема | Назначение | Реализация |
| API-шлюз | Прием запросов от пользователей и CI/CD, проверка прав, валидация входных данных | Python FastAPI + JWT аутентификация |
| Оркестратор задач | Управление жизненным циклом задач | Celery + Redis |
| Воркеры анализа | Асинхронное выполнение различных этапов сканирования:  краулинг;  DAST;  эмуляция атак;  РоС-проверка. | Python сервисы, объединённые очередью Celery |
| Подсистема агрегации | Объединение результатов, фильтрация ложных срабатываний, приоритизация по серьезности | Отдельный worker + SQLAlchemy |
| Подсистема отчетности | Генерация отчетов и передача уведомлений | FastAPI + Jinja2 + wkhtmltopdf |
| Хранилища данных | PostgreSQL – метаданные, задачи, результаты  MinlO – артефакты и отчеты  Redis – очередь и кэш | Вынесены как отдельные сервисы |
| Подсистема мониторинга и логирования | Сбор логов, метрик, статусов воркеров и API | Prometheus + Grafana + Loki |
| Интерфейс пользователя | Веб-панель для запуска сканов и просмотра отчетов | React |

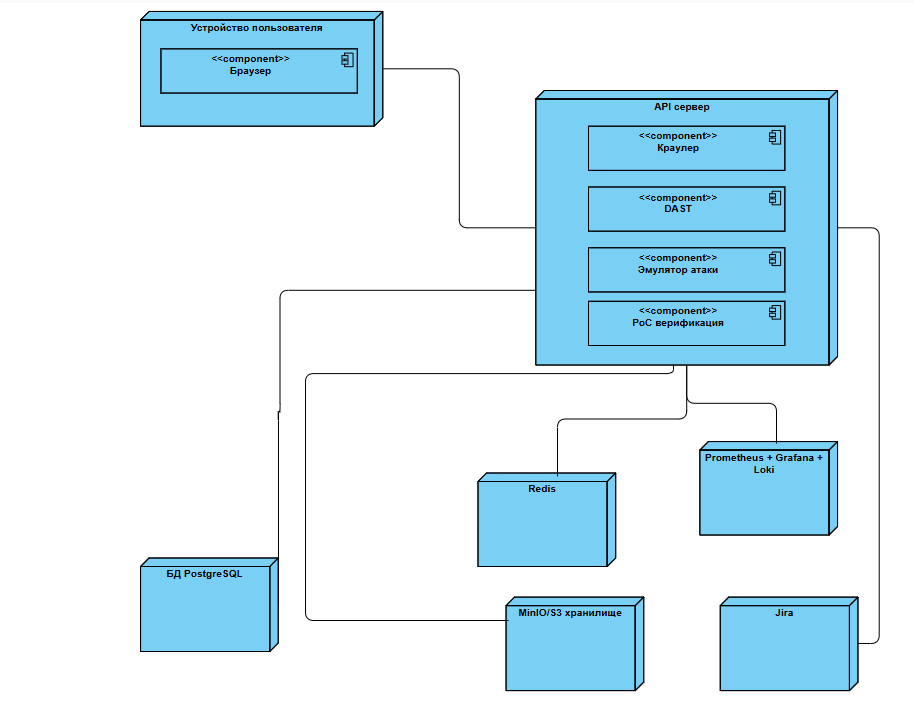


Рисунок 9 – Графическое представление

**Сервис-ориентированная архитектура (SOA)** с элементами **микросервисного подхода**:

* каждый модуль может работать автономно;
* связь между ними через REST или очередь сообщений (RabbitMQ / Redis);
* отчётность и хранение изолированы в отдельных узлах;
* безопасность обеспечивается API-ключами и HTTPS-взаимодействием.

1. Особенности взаимодействия с пользователем и макеты интерфейса
   1. Общие принципы взаимодействия
2. Простота и наглядность:

Система предоставляет интуитивно понятный интерфейс, не требующий специальных знаний в кибербезопасности.

1. Роль-ориентированный подход:

Доступные функции зависят от роли пользователя:

Пентестер / Тестировщик — запускает сканирование, анализирует уязвимости, просматривает отчёты.

Администратор — управляет пользователями, профилями, ограничениями scope.

Разработчик / Аналитик — получает результаты и использует отчёты для устранения уязвимостей.

CI/CD — вызывает API для автоматических сканов при релизах.

1. Формы взаимодействия:

Веб-панель (браузерная версия, desktop-friendly).

1. Типы интерфейса:

Графический веб-интерфейс — управление задачами, просмотр отчётов, статистика.

* 1. Особенности взаимодействия пользователя с системой

Таблица 2 – особенности взаимодействия пользователя с системой

|  |  |  |
| --- | --- | --- |
| Этап | Действия пользователя | Реакция системы |
| Авторизация | Ввод логина и пароля | Проверка прав и ролей, переход в личный кабинет |
| Создание задачи сканирования | Заполнение формы: ERL, профиль, описание проекта | Создается запись задачи, отображается статус «В очереди» |
| Отслеживание статуса | Пользователь видит список задач и их состояние | Автоматическое обновление прогресса |
| Просмотр отчета | Клик по завершенной задаче -> открытие подробного отчета | Отображается сводка уязвимостей, список найденных проблем |
| Подтверждение или отклонение FP | Нажимает кнопку «Ложное срабатывание» | Система обновляет метку в БД |
| Загрузка артефактов РоС | Кнопка «Скачать артефакт» | Передача файла из хранилища |
| Уведомления | Получает оповещение о завершении сканов, ошибках, нарушении области охвата | Email-уведомления |

* 1. Макеты интерфейса

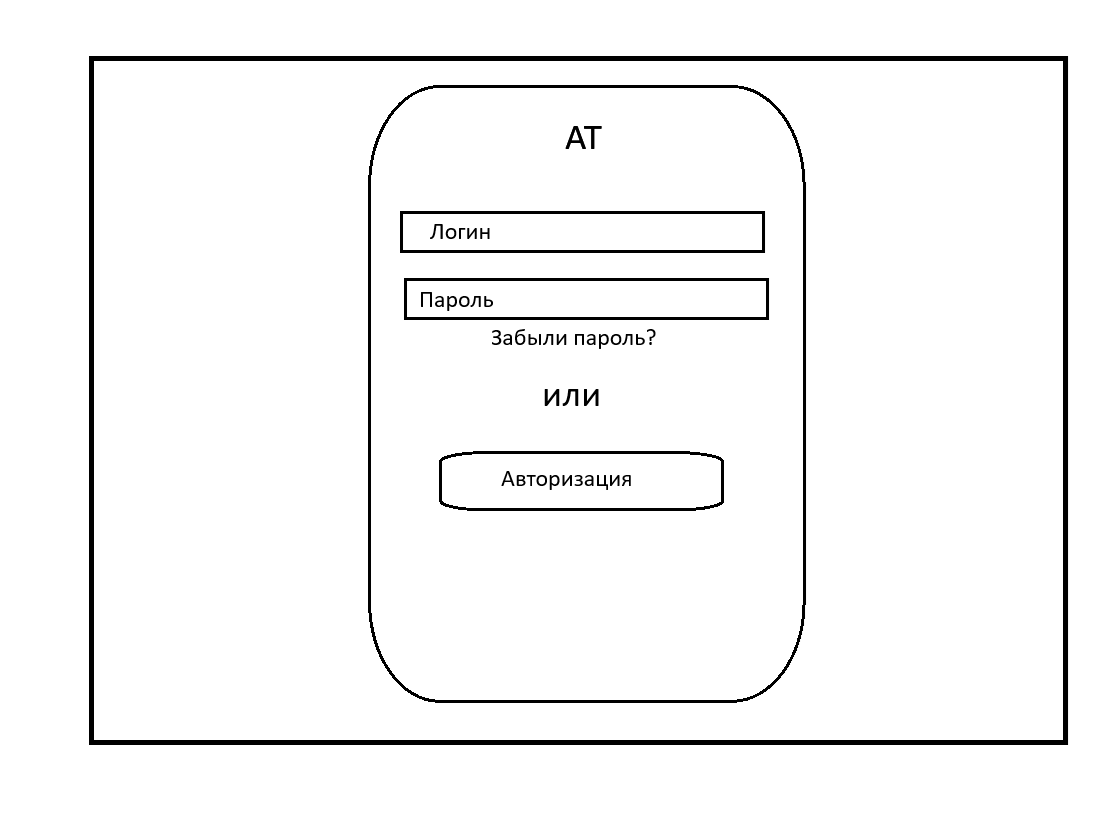


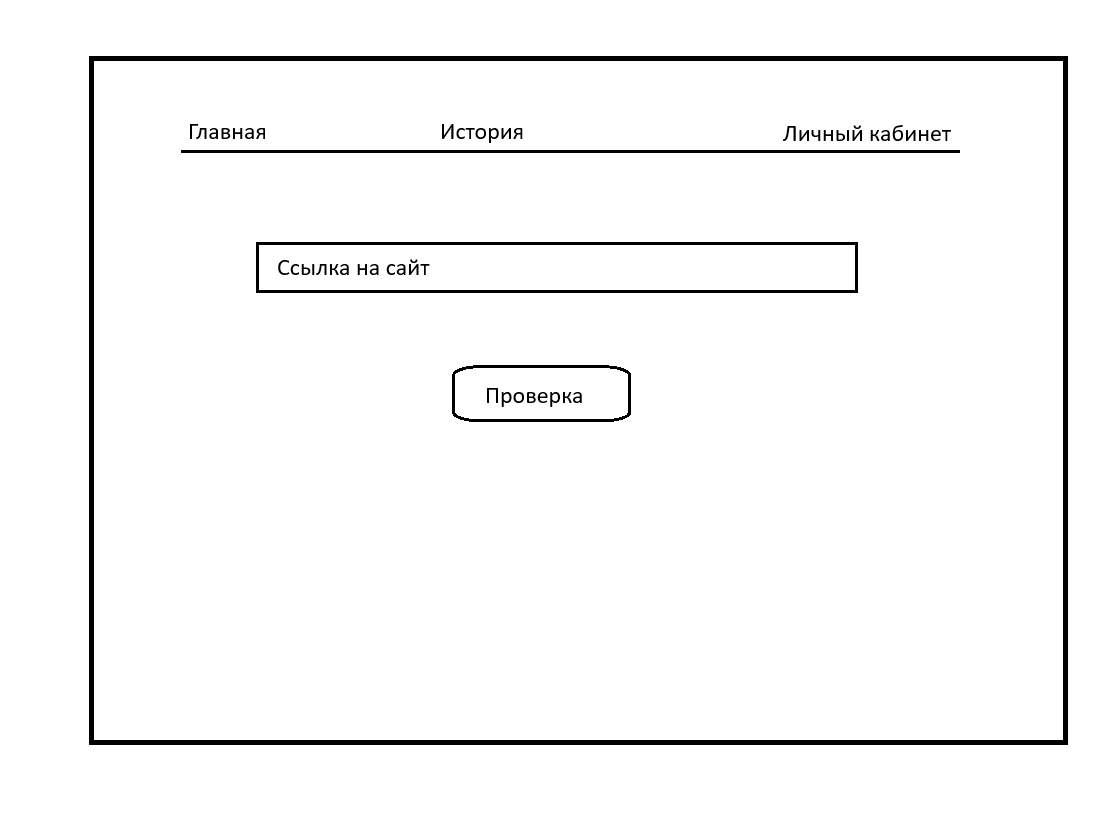
Рисунок 10 – Прототип окна входа и регистрации

Рисунок 11 – Прототип окна запуска сканирования

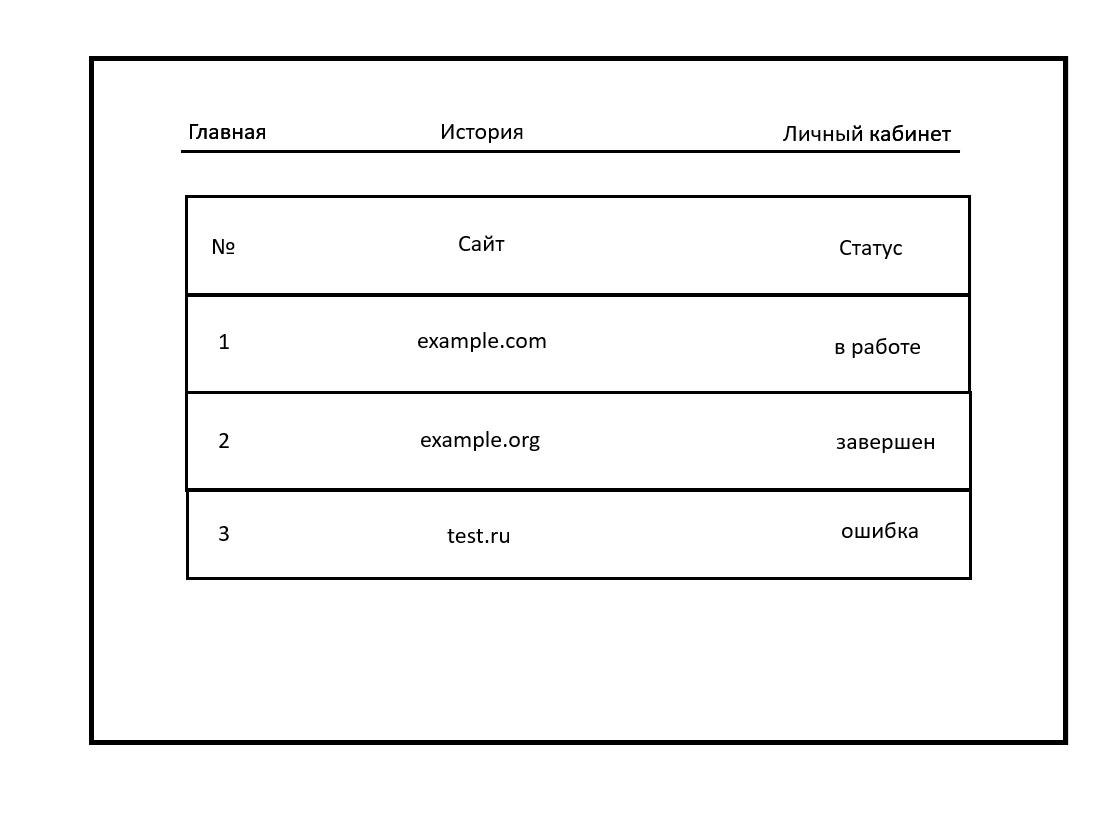


Рисунок 12 – Прототип окна истории

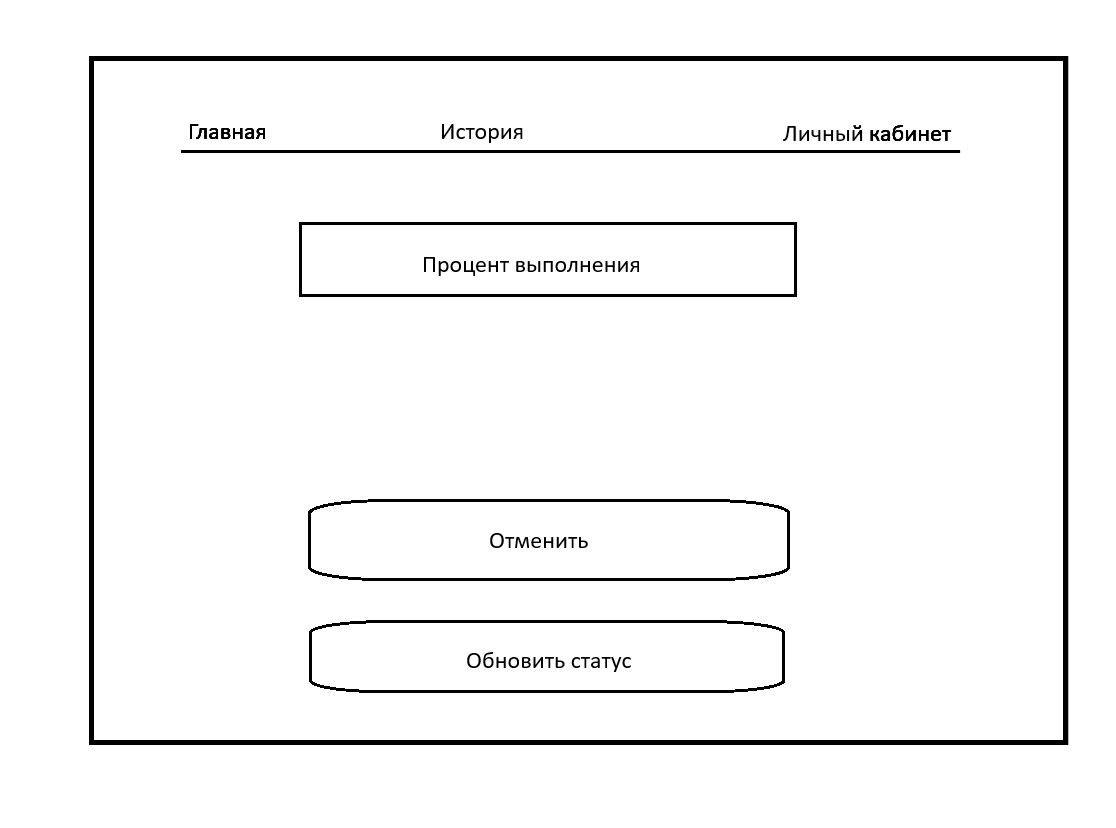


Рисунок 13 – Прототип окна процента выполнения

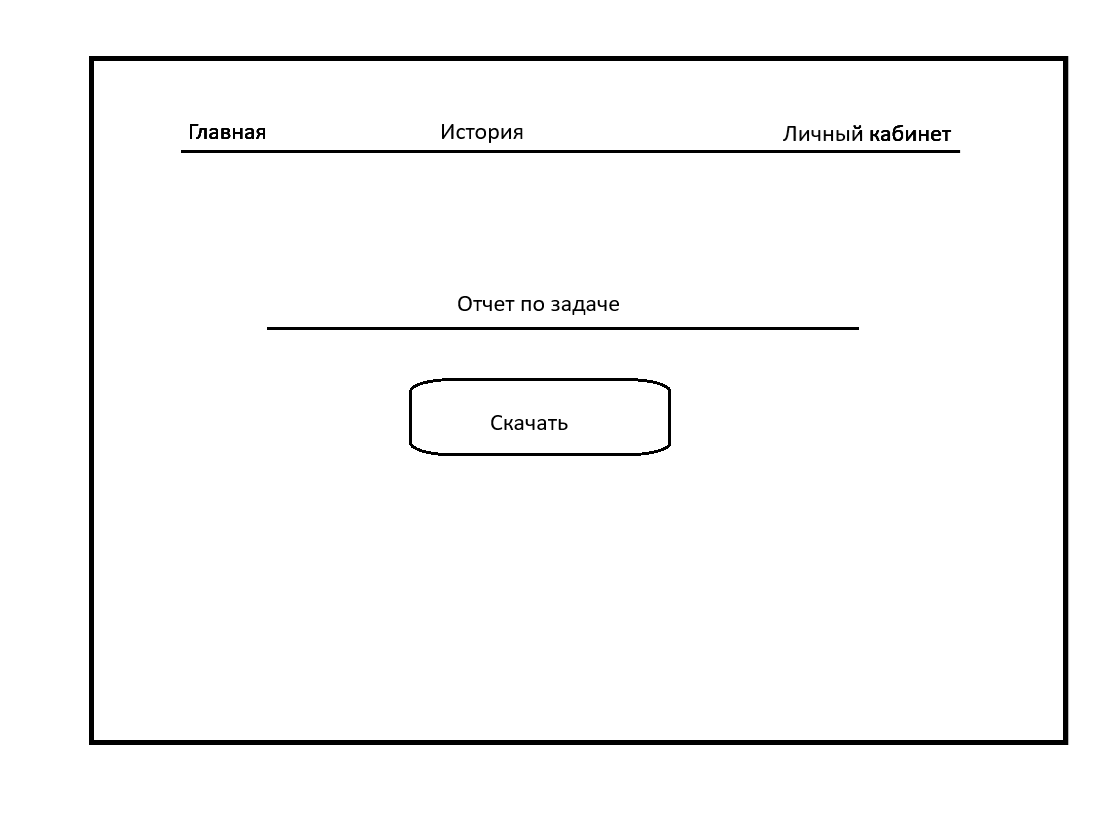


Рисунок 14 – Прототип окна отчета

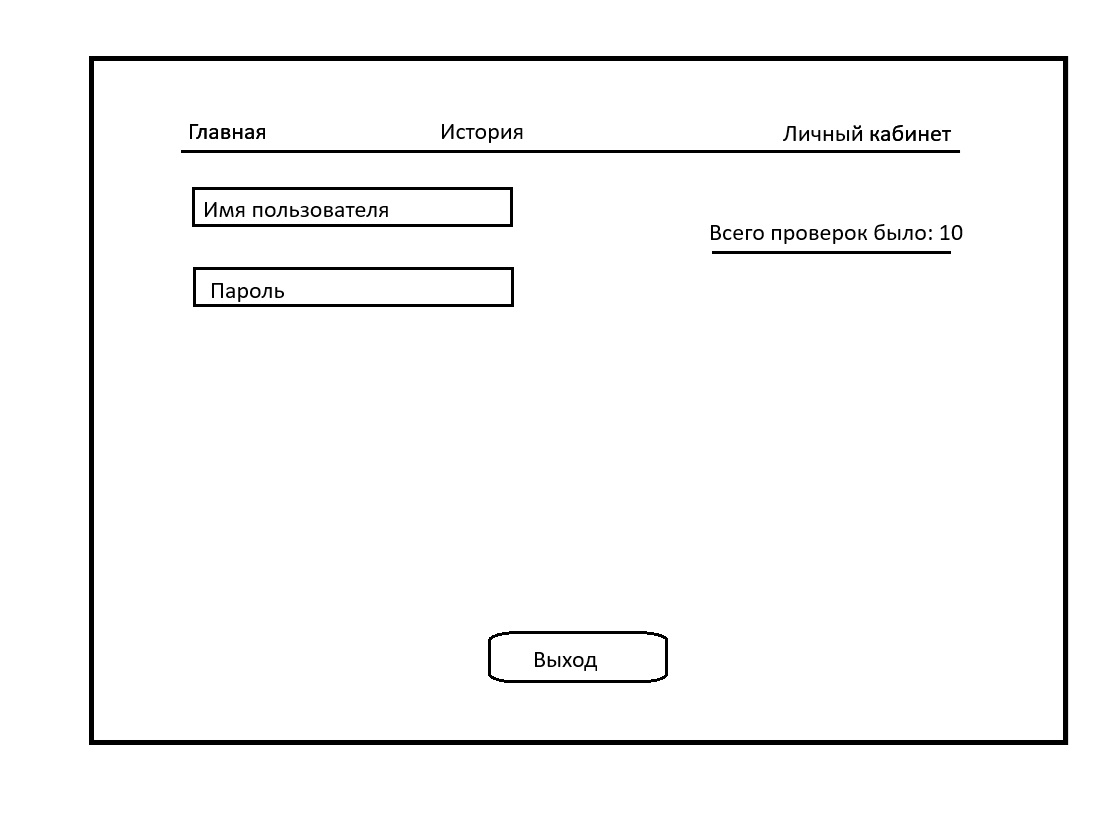


Рисунок 15 – Прототип окна личного кабинета