Промежуточный отчет по программному проекту

1. Основные планы и задачи разработчика Зобов А.А.

1.1 Краткое описание проекта:

«Веб-приложение для комплексной автоматизации процессов поискового продвижения веб-сайтов» («SEO Мастер») — это высокоавтоматизированная, микросервисная платформа, предназначенная для комплексного управления и оптимизации SEO-продвижения веб-сайтов. Платформа фокусируется на бизнес-ориентированной аналитике (FF-Score, E-E-A-T Score) и внедряет механизм Human-in-the-Loop (HITL).

Название проекта:

Веб-приложение для комплексной автоматизации процессов поискового продвижения веб-сайтов

Цель работы (Зобов А.А.):

Разработка ключевых микросервисов, отвечающих за управление бизнес-логикой (Management Service), обеспечение безопасности (API Gateway), реализацию пользовательского интерфейса (Frontend) и механизмов внедрения изменений (Client API Gateway, адаптеры WordPress и Tilda).

Краткое описание задач:

1. Разработка и реализация **Management Service** (Оркестратор SEO Robot, HITL-логика, приоритизация задач).
2. Разработка и реализация **Frontend (SPA)** (Дашборд FF-Score, HITL-UI с DiffViewer, экспорт CSV-отчетов).
3. Разработка и реализация **Client API Gateway** (Безопасное внедрение изменений, HMAC-SHA256).
4. Разработка адаптеров для **WordPress** и **Tilda**.
5. Обеспечение асинхронного взаимодействия через Celery/RabbitMQ.

1.2 Планы и этапы выполнения проекта (Специфично для Разработчика 1)

Микросервис / Компонент	Ключевые модули	Технологии	Статус/Сроки
Management Service	Оркестратор (`orchestrator.py`), HITL-логика, Приоритизация	Python, FastAPI, PostgreSQL	В процессе
Frontend (SPA)	Дашборд FF-Score, HITL-UI, DiffViewer, CSV-экспорт	React, Redux Toolkit, Tailwind CSS	В процессе
Client API Gateway	HMAC-Auth, PATCH-эндпоинты, Логирование	Python, FastAPI	В процессе
Адаптеры	Плагин для WordPress, адаптер для Tilda	PHP, JavaScript	В процессе
Общие задачи	Событийная шина (HITLApproved, TaskCreated)	Celery / RabbitMQ	В процессе

2. Используемый технологический стек и его обоснование

2.1 Перечень используемых технологий

Технология/Инструмент	Назначение	Обоснование
Python (FastAPI)	Backend-логика Management Service и API Gateway	Высокая производительность, встроенная поддержка асинхронности, отличная документация (OpenAPI)
React / Redux Toolkit	Frontend-интерфейс (SPA)	Компонентный подход, эффективное управление состоянием для сложных интерфейсов (HITL-UI)
PostgreSQL	Хранилище задач, HITL-одобрений и логов	Надежность, транзакционность, поддержка JSONB
Celery / RabbitMQ	Асинхронное выполнение оркестрации и деплоя	Необходимость отделения тяжелых операций от основного API-потока
Tailwind CSS	Стилизация интерфейса	Скорость разработки адаптивного и современного дизайна
PHP	Разработка плагина для WordPress	Стандартный язык для экосистемы WordPress

2.2 Обоснование выбранного технологического стека

FastAPI выбран за его скорость и автоматическую генерацию схем данных, что критично для интеграции микросервисов. **React** в связке с **Redux Toolkit** позволяет реализовать сложный динамический интерфейс HITL-контроля с сохранением состояния.

PostgreSQL обеспечивает целостность данных при хранении критических SEO-задач и истории их одобрения. Использование **Celery** позволяет масштабировать обработку задач оркестрации независимо от нагрузки на интерфейс.

Выбор **PHP** для WordPress-плагина обусловлен необходимостью нативной интеграции в ядро CMS клиента для максимально безопасного и производительного внедрения изменений.

Примечание по безопасности:

В production-окружении для управления секретами HMAC-ключей рекомендуется:

- Использование HashiCorp Vault или AWS Secrets Manager
- Автоматическая ротация ключей каждые 90 дней
- Аудит логов доступа к секретам

3. Критерии оценивания работы (Зобов А.А.)

Критерий	Описание
Функциональность Management Service	Корректная оркестрация задач, точность приоритизации по Impact x Effort, надежное хранение HITL-записей.
Качество Frontend-интерфейса	Удобство HITL-UI, корректная работа DiffViewer, адаптивность и скорость загрузки.
Безопасность Client API Gateway	Надежность HMAC-авторизации, корректность PATCH-операций, полнота логирования изменений.
Качество адаптеров	Стабильность плагина WordPress, корректная интеграция с Tilda, отсутствие негативного влияния на производительность сайта клиента.
Соблюдение архитектуры	Корректная реализация событийной модели и взаимодействие с другими микросервисами.