УДК 004.9

ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ИНТЕРАКТИВНОГО ВЕБ-СЕРВИСА ГЕОЛОГИЧЕСКОГО МУЗЕЯ С ИНТЕГРАЦИЕЙ НЕЙРОСЕТЕВОЙ КЛАССИФИКАЦИИ И ЖЕСТОВОГО УПРАВЛЕНИЯ

Руппель М.В., студент гр. ПИб-221, IV курс,
Филатов К.А., студент гр. ПИб-221, IV курс
Научный руководитель: Тайлакова А.А., к.н., доцент
Кузбасский государственный технический университет
имени Т.Ф. Горбачева, г. Кемерово

Современная парадигма цифровизации музейного пространства диктует необходимость создания не просто информационных порталов, а полноценных интерактивных образовательных сред. Целью данной работы является проектирование и разработка «с нуля» официального веб-представительства геологического музея. В отличие от использования готовых шаблонных решений, проект подразумевает индивидуальную разработку пользовательского интерфейса (UI/UX), развертывание и настройку системы управления контентом WordPress, а также внедрение двух ресурсоемких интеллектуальных функциональностей.

Ключевыми инновационными модулями системы выступают автоматическая классификация минералов (по 15 категориям) на основе предварительно обученной нейросети и интерактивная галерея экспонатов, управляемая бесконтактно - жестами рук пользователя через веб-камеру.

Прямая интеграция подобных разнородных технологий в монолитную архитектуру сопряжена с техническими противоречиями. Экосистема PHP, на которой базируется WordPress, не предназначена для эффективного исполнения (инференса) нейросетей, а выполнение тяжелых моделей машинного зрения непосредственно в браузере клиента приводит к недопустимой деградации производительности. В связи с этим возникает научно-практическая задача проектирования распределенной системы, обеспечивающей баланс между вычислительной нагрузкой, скоростью отклика и визуальной эстетикой [1].

На начальном этапе был спроектирован уникальный UI/UX-дизайн, отражающий специфику геологического музея и отвечающий современным стандартам эргономики (рис. 1). На основе созданных макетов была осуществлена верстка адаптивного интерфейса и разработана кастомная (индивидуальная) тема для CMS WordPress.

Рис. 1. Главная страница сайта.

Выбор WordPress в качестве базового ядра системы обусловлен необходимостью предоставления сотрудникам музея удобного инструмента для управления контентом (добавление новых экспонатов, публикация новостей, редактирование описаний). Развертывание платформы включало настройку базы данных MySQL, конфигурацию веб-сервера и интеграцию разработанной визуальной оболочки с бэкендом на PHP. Таким образом, WordPress взял на себя роль фронтенда и системы управления базовой текстово-графической информацией, сформировав надежный фундамент для подключения сложных надстроек [2] [3].

При проектировании логики работы с нейросетевой моделью (весом 166 МБ), обученной для классификации 15 видов минералов, критически важным стал выбор места ее исполнения.

Рассматривался вариант использования библиотеки TensorFlow.js для классификации непосредственно в браузере клиента. Однако необходимость загрузки файла объемом 166 МБ при каждом уникальном посещении сайта требует значительной пропускной способности сети, приводит к длительному ожиданию и способна вызвать критические сбои («падения» вкладки) на мобильных устройствах с ограниченным объемом оперативной памяти [5].

В качестве оптимального инженерного решения была выбрана гибридная микросервисная архитектура, разделяющая ответственность между узлами:

1. Базовый интерфейс и управление контентом остаются в юрисдикции разработанной темы WordPress.
2. Логика распознавания жестов реализуется на стороне клиента (браузер) с использованием легковесных JS-решений.
3. Нейросетевая классификация выделяется в независимый backend-сервис на языке Python, с которым основной сайт общается посредством API.

Разработанный программный комплекс функционирует как совокупность взаимодействующих слоев:

Слой пользовательского интерфейса (WordPress/JS): реализует визуальную оболочку. Интерактивная галерея построена на базе библиотеки Swiper.js (с использованием модулей Lightbox) и поддерживает плавное масштабирование (zoom) изображений экспонатов.

Слой распознавания жестов (MediaPipe): интегрирован непосредственно в клиентский JavaScript-код. Захват видеопотока с веб-камеры и детекция ключевых точек кисти (Landmarks) происходят в браузере. Вычисленные векторы движения транслируются в API библиотеки Swiper (например, свайп влево/вправо для переключения слайдов). Обработка «на лету» без отправки видеопотока на сервер гарантирует нулевую сетевую задержку и абсолютную приватность биометрических данных пользователя, не нагружая серверные мощности [7].

Слой машинного обучения (FastAPI, Python): реализует RESTful API для приема запросов на классификацию. Модель загружается в оперативную память сервера однократно при инициализации сервиса. Это исключает длительные задержки на чтение 166 МБ с диска при каждом новом запросе, позволяя серверу асинхронно и быстро обслуживать множество одновременных подключений [4].

Особое внимание в проекте уделено защите серверной инфраструктуры Python от перегрузок (DDoS-эффекта), которые могут возникнуть при массовой загрузке неоптимизированных фотографий минералов со смартфонов (размер которых достигает 10–20 МБ). Разработан двухконтурный алгоритм контроля и сжатия:

1. Клиентская оптимизация (Front-end): при выборе пользователем фотографии скрипт на стороне браузера (с использованием Canvas API) производит предварительную обработку. Изображение масштабируется до целевого разрешения, требуемого нейросети (224×224 пикселя), и конвертируется в формат JPEG с коэффициентом качества ~80%. В результате размер передаваемого Payload сокращается с нескольких мегабайт до десятков килобайт.
2. Серверная валидация (Back-end): API на базе FastAPI принимает запрос в формате multipart/form-data. Производится fallback-проверка (повторный контроль размера, формата и целостности файла). Только после успешной валидации тензор передается в модель, которая возвращает JSON-объект, содержащий предсказанный класс минерала и коэффициент достоверности (вероятность).

Тестирование разработанной с нуля архитектуры показало высокую эффективность выбранного подхода. Созданный дизайн корректно отображается на всех типах устройств, а WordPress обеспечивает удобное управление контентом. Вынос логики MediaPipe на сторону клиента полностью избавил веб-сервер от вычислений, связанных с компьютерным зрением в реальном времени. FastAPI-сервис продемонстрировал устойчивость к множественным конкурентным запросам благодаря предварительной

загрузке модели в RAM и радикальному уменьшению веса входящих изображений на этапе клиентского препроцессинга [6].

В перспективе распределенная природа системы позволяет легко масштабировать вычислительные мощности: Python-сервис может быть обернут в Docker-контейнер и реплицирован независимо от основного сайта на WordPress, если посещаемость музея значительно возрастет.

Разработанный программный комплекс представляет собой комплексное инженерное решение задачи создания современной цифровой среды для геологического музея. Выполнение полного цикла работ - от индивидуального UI/UX-дизайна и развертывания CMS WordPress до внедрения сложных алгоритмов искусственного интеллекта - позволило создать гибкий и масштабируемый продукт. Разделение бизнес-логики на независимые микросервисы (делегирование контента PHP, распознавания жестов клиентскому JavaScript, а тяжелых нейросетевых вычислений отдельному Python-сервису) обеспечило синергетический эффект. Достигнута стабильная, безопасная и высокопроизводительная работа ресурсоемких функций без ущерба для скорости загрузки и отзывчивости основного сайта. Предложенная архитектура может служить эталоном для разработки современных музейных веб-платформ с применением ML-технологий.

Список литературы

1. WordPress. Официальный сайт : [сайт]. URL: https://wordpress.org (дата обращения: 28.03.2026).
2. MySQL. Документация : [сайт]. URL: https://dev.mysql.com/doc/ (дата обращения: 28.03.2026).
3. PHP: Hypertext Preprocessor. Официальный сайт : [сайт]. URL: https://www.php.net (дата обращения: 28.03.2026).
4. FastAPI. Документация : [сайт]. URL: https://fastapi.tiangolo.com (дата обращения: 28.03.2026).
5. TensorFlow.js. Документация : [сайт]. URL: https://www.tensorflow.org/js (дата обращения: 28.03.2026).
6. MediaPipe. Руководство разработчика : [сайт]. URL: https://developers.google.com/mediapipe (дата обращения: 28.03.2026).
7. Swiper.js. Документация : [сайт]. URL: https://swiperjs.com (дата обращения: 28.03.2026).