Техническая документация

Сервис для автоматизации приёма и выдачи инструментов авиаинженерам на базе компьютерного зрения.

1. Общая информация

Сервис автоматизирует процессы выдачи и приёма инструментов у авиаинженеров.

В основе – компьютерное зрение: сверка эталонных изображений инструмента с фото, сделанными при возврате.

Ключевые задачи:

- учёт и контроль наличия инструментов;
- автоматическая сверка возвратов по фотографиям;
- снижение ошибок человеческого фактора;
- ведение истории перемещений.

Технологический стек: Flask (Python), SQLAlchemy, OpenCV, Jinja2, HTML5, JS, SQLite/PostgreSQL.

2. Архитектура системы

Система разделена на уровни:

1) Интерфейс пользователя (UI) – HTML/Jinja2, формы, загрузка фото.

2) Серверное приложение (Flask) – маршрутизация, бизнес-логика, работа с ORM.

3) Модуль CV (tool_vision.py) – сегментация, выделение признаков, сравнение с эталонами.

4) Хранилище – база данных SQL и файловая система (/uploads).

3. Основные модули

- app.py – запуск Flask, маршруты, модели ORM (ToolType, ToolItem, Request и др.).
- tool_vision.py – сегментация, дескрипторы (Hu-моменты, HSV-гистограммы, ORB), сравнение.
- cfg.py – настройки для запуска приложения
- HTML-шаблоны – UI для управления типами, инструментами, заявками.

4. Модель данных

Основные сущности:

- ToolType: id, name, photo.
- ToolItem: id, inv, type_id, photo.
- ToolItemRefPhoto: id, item_id, filename.
- Request: id, code, status, created_at, closed_at.
- RequestToolItem: request_id, item_id.
- RequestPhoto: request_id, filename.

- ToolMovement: item_id, request_id, action, ts.

5. Основные сценарии работы

1) Добавление типа – ввод названия, фото, экземпляров, эталонные фото.

2) Создание заявки – выбор инструментов.

3) Закрытие заявки – загрузка фото возврата, сверка CV, сохранение результата.

4) История – лог выдач и возвратов (ToolMovement).

6. Алгоритмы компьютерного зрения

- Сегментация: Otsu, адаптивный порог, Canny, морфология.
- Дескрипторы: форма (контуры, Hu-моменты), цвет (HSV), ключевые точки (ORB).
- Сравнение: matchShapes, сравнение площади, цветовые гистограммы, ORB BFMatcher.
- Итог: интегральный скоринг с весами из конфигурации.

7. Сильные стороны решения

- Автоматическая сверка по фото.
- Возможность проверки целостности наборов из мелких предметов
- Возможность масштабирования в короткие сроки.
- Гибкая модель данных.
- Возможность ручного решения по результатам сверки.
- Модульная архитектура.

8. Возможные направления развития

- Умные пороги. Автоподстройка порога под тип инструмента/условия, учёт «отрыва» и качества кадра.
- Добавить компактную ML детекцию основных типов инструментов для ускорения работы.
- Качество и контроль + безопасность. Логи, генерация актов расхождений и фотоприложений, журнал действий.
- Внедрение возможности расследования по итогам недостающих инструментов.

9. Требования к окружению

- Python 3.10+.
- Flask, SQLAlchemy, OpenCV, scikit-image, NumPy, SciPy.
- БД: SQLite или PostgreSQL.
- ОС: Linux/Windows.