












# Trust Indicators Project Handover Document (S1 2025 -> S2 2025)

## 1. Project Overview

-  **Project Title:** Trust Indicator
-  **Objective:** To develop an Australian-themed image gallery website that helps users better understand and assess the authenticity of uploaded images by displaying their metadata and providing visual cues. This website will be an open-source project, allowing users to upload their own images while adhering to Creative Commons (CC) licensing requirements. By collecting image metadata and displaying credibility signals, the website will assist users in identifying and discerning genuine images. Additionally, website members can contribute background information, such as photographer statements, to provide more contextual details. The project aims to offer consumers a reliable resource to enhance their ability to judge image authenticity. It also provides a platform for photographers and creators to showcase their work and offer additional information, thereby improving public understanding and awareness of images. By being an open-source project, the website encourages other developers to contribute and make improvements, fostering the project's continuous development and enhancement.
-  **Expected Deliverables:**
  - Live project website
  - Technical documentation
  - Source code - GitHub repository link
-  **Landing page:** <https://sites.google.com/view/trustindicators>
-  **Acknowledgements:** We sincerely thank the previous team for their valuable contributions to the project. Their hard work and dedication have laid a solid foundation for us. For detailed information about the previous team's work, please refer to the materials provided at <https://github.com/Trust-Indicator/Trust-Indicator>.

## 2. Codebase & Setup

-  **Repository:** The core code is located in the <https://github.com/bridgeL/Trust-indicator> directory.
-  **Tech Stack:**

Category	Details
 <b>Backend</b>	Python 3.8+, Flask, SQLAlchemy
 <b>Frontend</b>	HTML, CSS, JavaScript (See <code>templates</code> and <code>static</code> directories for specific frameworks and libraries)
 <b>Database</b>	SQLite (Managed via <code>database.py</code> and <code>MyDatabase.db</code> file)
 <b>External APIs</b>	Alibaba Cloud AIGC detection, ImgBB (for temporary image hosting to get URLs)

• 🔑 **Key Libraries/Dependencies:**

Library/Dependency	Description
Pillow (PIL)	For image processing and EXIF data extraction (See <code>ExifExtractor</code> directory and <code>routes/upload.py</code> )
Werkzeug	WSGI utility library, Flask dependency
Flask Extensions	Flask-Login (user authentication), Flask-SQLAlchemy (database operations), Flask-Mail (mail service), Flask-Session (session management) (See <code>extension.py</code> and <code>app.py</code> )
alibabacloud_green20220302	Alibaba Cloud Content Moderation SDK (See <code>aigc_detector.py</code> )
requests	For sending HTTP requests (See <code>aigc_detector.py</code> )
PyJWT	For generating and validating tokens (See <code>routes/login.py</code> )
requirements.txt	See for the complete list of dependencies

• 🚀 **Setup Instructions:**









- 🏠 **Local Development Environment Setup:**
  - a. 1 Clone repository: `git clone [Repository URL] ~/Trust-Indicator`
  - b. 2 Navigate to project directory: `cd ~/Trust-Indicator`
  - c. 3 Create and activate Python virtual environment:
    - ▶ `python -m venv .env`
    - ▶ `source .env/bin/activate` (Linux/macOS) or `.env\Scripts\activate` (Windows)
  - d. 4 Install dependencies: ▶ `pip install -r requirements.txt`
  - e. 5 🔒 **Environment Variables Setup: ! Important Note:** The AIGC Detector module requires Alibaba Cloud Access Key ID, Access Key Secret, and ImgBB API Key. These can be set as operating system environment variables, in the Python script, or using a `.env` file with the `python-dotenv` library. **Do not hardcode API keys and access credentials into the source code, and do not commit files containing this sensitive information (e.g., `.env` ) to the version control system.**


Credential Type	Variable Names	Notes
☁️ Alibaba Cloud Credentials	ALIBABA_CLOUD_ACCESS_KEY_ID , ALIBABA_CLOUD_ACCESS_KEY_SECRET	Required for AIGC Detector module
🖼️ ImgBB API Key	IMGBB_API_KEY	Required for AIGC Detector module
✉️ Email Service Credentials (Gmail)	MAIL_USERNAME , MAIL_PASSWORD	See <code>extension.py</code>

- f. 6 📁 **Database Initialization:** On the first run, if `instance/MyDatabase.db` does not exist, the system will create it automatically. (See `database.py` )
- g. 7 🏃 **Run application:** ▶ `python app.py`
- ☁️ **Server Deployment:** For detailed deployment instructions, please refer to `deploy.md` . Key steps include:












- a. Server access and user account setup.
- b. Environment setup (clone repository, virtual environment, dependency installation).
- c. Using `screen` for persistent application running.
- d. Caddy reverse proxy configuration.

## ✨ 3. Features & Current State





-  **Completed Features (Based on S1 2025 Delivery Plan):**
  -  **User System:**
    - User registration, login, logout. (See `routes/login.py` )
    - Password change, password reset (via email verification code). (See `routes/login.py` )
    - User profile viewing and avatar modification. (See `routes/user_profile.py` )
  -  **Image Upload and Processing:**
    - User image upload (JPEG format only). (See `routes/upload.py` )
    - Image storage as binary data and thumbnail generation. (See `create_thumbnail` in `routes/upload.py` )
    - **EXIF Metadata Extractor:** Extracts EXIF metadata from images, including device information, shooting parameters, GPS location, etc. (See `ExifExtractor` directory and `routes/upload.py` )
    - **AIGC Detector:** Integrates Alibaba Cloud API to detect if an image is AI-generated. (See `aigc_detector.py` and `routes/aigc_detector.py` )
      - If API credentials are not provided or API call fails, it will automatically fall back to simulated results. (See `AIGC_DETECTOR_README.md` and `aigc_detector.py` )
    - Users can tag images (Original, AIGC, Manipulation) and add descriptions. (See `routes/upload.py` )
    - Users can set image visibility (public/private). (See `routes/upload.py` )
  -  **Image Browse and Details:**
    - Image gallery page, supporting filtering and sorting by upload time (ascending/descending), tags, and keyword search. (See `routes/gallery.py` )
    - Image detail page, displaying the image, detailed metadata, AIGC detection probability, user tags, and description. (See `routes/image_detail.py` )
  -  **Credibility Indicators and Report:**
    - **Trust Profile:** Users can set personalized credibility preferences, such as acceptable thresholds for AI-generated content, metadata integrity requirements, etc. (See `routes/trust_profile.py` )
    - **Trust Report:** On the image detail and analysis pages, a brief credibility assessment is generated based on the user's Trust Profile and the image's detection results (AIGC probability, etc.). (See `static/js/imagetail.js` and `WorkspaceTrustReport` related logic in `routes/image_detail.py` )  
(Note: Original text here was `WorkspaceTrustReport`; based on context and other code logic, it is presumed to be `fetchTrustReport`)
    - The analysis page displays image analysis results, including charts for AIGC probability, original probability, manipulation probability, and a comprehensive credibility score based on the user's Trust Profile. (See `templates/html/analysis.html` and `static/js/analysis.js` )
  -  **Favorites Management:** Users can add images to or remove them from their favorites. (See `routes/image_detail.py` )
  -  **Feedback System:** Users can submit feedback (Bugs, Questions, Comments), and the system will send a confirmation email. (See `routes/feedback.py` )
  -  **Other Pages:**
    - Home Page ( `routes/home.py` )
    - What We Do Page ( `routes/whatwedo.py` )

-  **In Progress / Known Issues:**
  - According to 24-S2 Trust indicators delivery plan.pdf , most core features have been validated.
  - AIGC detection API dependency risk: Service unavailability or exceeding quotas may lead to detection feature interruption. Mitigation strategy is to implement local fallback or cache recent results.
  - API key leakage risk: May lead to security vulnerabilities or service suspension. Mitigation strategy is to properly store sensitive keys and rotate them periodically.
  - The frontend may have some minor UI/UX issues that require further testing and refinement.
  - Some error handling and edge cases may need enhancement.

## 4. Documentation

Icon	Document File	Description
	README.md (Project Root)	Project Overview
	AIGC_DETECTOR_README.md	AIGC Detection Module Guide
	deploy.md	Deployment Guide
	docs-backend.md	Backend API Documentation
	docs-database.md	Database Documentation (includes data models and relationships)
	docs-page.md	Page Structure / Website Page Summary
	docs-request-type.md	Request Type Documentation (describes interaction of different request types between frontend and backend)
	requirements.txt	Project Dependencies
	Part 2 of this document & deploy.md	Local Installation Guide
	AIGC_DETECTOR_README.md , aigc_detector.py	Alibaba Cloud AIGC Integration Guide
	This Handover Document	This Handover Document

## 5. Maintenance and Future Development Tips

-  **API Key Management:** Ensure the security of all API keys (Alibaba Cloud, ImgBB, email) and do not leak them. The next team will need to apply for their own keys and update them in the environment variables.
-  **Dependency Updates:** Regularly check libraries in requirements.txt for security updates or important version iterations.
-  **Code Readability and Comments:** Continuously maintaining and adding comments is very helpful for new members to understand the code.
-  **Frontend Performance:** For pages with many images, like the image gallery, consider further optimizing frontend performance, e.g., advanced image lazy loading strategies, virtual lists, etc.

- 💡 **Extensibility of Metadata Extraction:** `ExifExtractor` currently focuses on common EXIF tags. If support for more types of metadata (like XMP, IPTC) or more complex extraction logic is needed in the future, this part of the code can be extended.
- 💡 **Testing:**
  - 🔧 It is recommended to write unit tests for core modules (e.g., AIGC detection, metadata extraction, user authentication, database operations).
  - 🔧 Conduct integration tests for API interfaces.
- 💡 **Asynchronous Tasks:** For time-consuming operations, consider using task queues like Celery to make them asynchronous, improving user experience and avoiding request timeouts.

## 📞 6. Contacts

Role	Name	Semester	Contact (Email, etc.)
💡 <b>Team Leader</b>	Juliang Xiao	S1 2025	u7757949@anu.edu.au
💡 <b>Web Developer</b>	Xinyang Li	S1 2025	u7760022@anu.edu.au
🏆 <b>Frontend Engineer</b>	Yushan Zhang	S1 2025	u7759158@anu.edu.au
🏆 <b>Backend Engineer</b>	Kun Gong	S1 2025	u7628201@anu.edu.au
💡 <b>Project Admin</b>	Chu Zhang	S1 2025	u7770023@anu.edu.au

We hope these Notes and Tips will help the next team to smoothly take over and advance the Trust Indicators project!