- # Sprint 2 Report (Oct 7th – Nov 5th)
-
- ## YouTube Link of Sprint 2 Video
- https://youtu.be/CtGpTAU0grk
-
- ---
-
- ## What's New (User Facing)
-
- - Migrated all front-end/back-end functionality from JavaScript/HTML to Python.
- - Added CSV import and export functionality for applicant data.
- - Implemented an analytical dashboard for visualizing applicant scoring and regional trends.
- - Enhanced document handling to flag suspicious submissions for manual review.
- - Added AI-driven Financial Fraud Detection module that screens uploaded financial documents for tampering, inconsistencies, or digital edits using OCR and GPT-based analysis.
- - Added AI-driven Transcript Fraud Detection module to analyze academic records for authenticity, GPA consistency, and image anomalies.
-
- ---
-
- ## Work Summary (Developer Facing)
-
- This sprint focused on converting the project from a hybrid JavaScript/HTML prototype to a fully Python-based system, improving maintainability and scalability. We rewrote data processing, scoring algorithms, and data handling modules, while maintaining existing functionality. The team also implemented CSV import/export features and built an analytical dashboard to visualize applicant scores and regional trends. Challenges included mapping front-end logic into Python and ensuring smooth data flows between modules.
- We also focused on integrating AI-powered document validation features into the International Rating Algorithm system. Developing two major fraud detection subsystems, one for financial documents and another for academic transcripts, that automatically classify submissions as Low, Medium, or High risk using structured forensic analysis. Both detectors use OCR extraction, error level analysis (ELA), copy-move detection, and GPT-5-based reasoning to flag inconsistencies. These help support the goal of automating applicant screening.
- During testing, we noticed a clear pattern: real documents usually scored in the Low Medium range, while fake ones landed in the Medium High range. This matched our goal of cautious detection, catching suspicious cases without unfairly flagging valid ones. We’ll keep adjusting the thresholds to stay balanced as the model improves.
-
- ---
-
- ## Unfinished Work
-

- - Integration of the analytical dashboard with a web-based interface (planned for Sprint 3).
- - Automating email notifications for flagged documents.
- - UI integration for fraud detection (able to upload a file and see the results on screen, and export info)
- - Fine-tune transcript and financial models for even better separation between low, medium, and high cases for a more confident case.
-
- ---
-
- ## Completed Issues / User Stories
-
- - Issue #1 – Migrated JS/HTML scoring algorithm to Python
- - Issue #2 – Added CSV import/export functionality
- - Issue #3 – Built analytical dashboard for applicant trends
-
- ---
-
- ## Incomplete Issues / User Stories
-
- - Issue #4 – Web integration for dashboard – Needs front-end connection to Python backend.
- - Issue #5 – Automated email notifications – Pending configuration for flagged documents.
- - [Teammate Incomplete Issue 1] – [Reason not completed]
- - [Teammate Incomplete Issue 2] – [Reason not completed]
-
- ---
-
- ## Code Files for Review
-
- - `main.py`
- - `data_processing.py`
- - `dashboard.py`
- - `financial_fraud_detector.py`
- - `transcript_fraud_detector.py`
- - `eval_results.py`
-
- ---
-
- ## Retrospective Summary
-
- ### What Went Well
-
- - Successful migration from JS/HTML to Python.
- - Smooth implementation of CSV import/export.
- - Analytical dashboard provides clear insights.

- - Integrated the financial fraud detection system into the existing workflow without breaking prior functionality.
- - Developed a new transcript fraud detection module that can accurately identify suspicious academic records.
- - Achieved consistent detection patterns across both authentic and fraudulent datasets with clear confidence scoring.
- - Improved data organization by separating financial and transcript results into distinct batch reports for better clarity.
- - Maintained compatibility between new AI-driven features and existing data processing code.
-
- - [Teammate Point 2]
-
- ### What We'd Like to Improve
-
- - Reduce manual configuration for new datasets.
- - Improve dashboard interactivity and usability.
- - Increase automation in the workflow.
- - Make the AI results more consistent between runs (reduce small random differences).
- - Fine-tune how the system decides between low, medium, and high risk levels for better accuracy.
- - Improve runtime speed when analyzing large multi-page PDFs.
-
- ### Changes Planned for Next Sprint
-
- - Integrate the dashboard with a web interface (Flask/FastAPI).
- - Implement automated notifications for flagged applications.
- - Optimize scoring algorithm performance for larger datasets.
- - Improve consistency of AI fraud detection results so the same document gives the same outcome each run.
- - Adjust confidence thresholds to better separate low, medium, and high risk documents.
- - Add a summary report showing how many documents were flagged at each risk level
- - Expand testing on both financial and transcript datasets to validate detection accuracy
- - Add plain-english explanations for each fraud detection result that describes why the document was marked how it was. (for easier understanding)
-