Introduction
Touchstone Institute's PRO Portal plays a pivotal role in processing applications for Practice Ready Ontario. One of its key features is an automatic flagging system that helps identify potential eligibility concerns early in the process. As part of your internship assessment, you are tasked with evaluating and improving this system.

Objective
Demonstrate your understanding of software development practices and frontend/backend skills by proposing and prototyping improvements to the automatic flagging system within the PRO Portal. This project simulates real work you might do as a Summer Student at Touchstone Institute. While we expect a functioning prototype, we understand the time constraints. Focus on demonstrating your approach to the problem, your technical skills, and your understanding of Agile methodologies rather than building a complete production-ready system.

Part 1 – Software Development Lifecycle & SCRUM (Written)
Prepare a short (1–2 page) written submission responding to the following:
1.SCRUM Roles & Events:

- Briefly describe how you would contribute to a SCRUM-based team as a student intern.
- What SCRUM ceremonies would you participate in, and what would be your input?

Briefly describe how you would contribute to a SCRUM-based team as a student intern.

SCRUM Roles & Events

My Role as a Student Intern

Supporting the Product Owner (PO): I'd help refine user stories by researching eligibility rules for Practice Ready Ontario (PRO) applicants and summarizing findings.

Assisting the Development Team: I'd pair-program on frontend flag-display components, write unit tests for backend rules, and document configuration options.

Contributing to Quality & Documentation: I'd draft acceptance criteria, help maintain the Definition of Done, and keep Confluence pages up to date with design decisions.

Ceremonies & My Input

Ceremony	***Frequency***	***My Participation & Input***
Sprint Planning	Every 2 weeks	Estimate flagging-rule stories, propose technical tasks (e.g., API endpoints, UI mocks).
Daily Scrum	Daily (15 min)	Report progress on assigned tasks ("Yesterday I…" / "Today I'll…" blockers: "Need clarification on X-rule.").
Backlog Refinement	Mid-sprint	Review upcoming stories, ask clarifying questions about edge cases (e.g., foreign credentials).
Sprint Review	End of sprint	Demo new prototype of the flagging UI, solicit feedback from stakeholders.
Sprint Retrospective	End of sprint	Share what worked (e.g., pair-programming) and suggest improvements (e.g., more automated test coverage).

What SCRUM ceremonies would you participate in, and what would be your input?

SDLC Planning for the Automatic Flagging System

Phase	***Key Activities & Deliverables***
Requirements	***Gather & Analyze:*** Interview subject-matter experts to list all eligibility criteria. ***User Stories:*** "As an intake officer, I want the system to flag missing credentials so I can follow up immediately." ***Acceptance Criteria:*** Define pass/fail conditions for each rule (e.g., "flag if years of experience < 2").
Design	***High-Level Architecture:*** Decide on microservice vs. monolith; choose tech stack (e.g., React for UI, Node.js/Express for API). ***Data Model & Flow:*** Draft ER diagram for applicant data; sequence diagram for flagging logic. ***UI/UX Mockups*:** Sketch how flags appear in the PRO Portal, including severity levels and "more info" links.
Implementation	***Back-end Development:*** Build a Rule Engine module that reads JSON-configurable rules and evaluates each application record. ***Front-end Development:*** Create React components to fetch and display flags, with inline tooltips. • **Configuration Interface:** Prototype a simple admin page to add/edit flag rules without code changes.
Testing	***Unit Tests:*** Cover each rule in isolation (e.g., Jest for JS).

	Integration Tests: Simulate end-to-end flagging on sample applications. ***User Acceptance Testing (UAT)*:** Have intake officers run through "happy path" and "edge case" scenarios; collect feedback.
Deployment	***CI/CD Pipeline:*** Configure GitHub Actions to run tests, build Docker images, and deploy to a staging environment. ***Staging Validation:*** Smoke-test in staging; load-test with synthetic data to ensure performance. ***Production Rollout:*** Deploy to production during low-traffic window; monitor key metrics.
Maintenance	***Monitoring & Alerts:*** Set up dashboards (e.g., Grafana) for rule-evaluation errors and response times. ***Bug Fixes & Enhancements:*** Triage issues in JIRA; iterate on flag criteria based on reviewer feedback. ***Continuous Improvement:*** Schedule periodic rule-review sessions with the PO to retire outdated flags and add new ones.

Agile Alignment

Iterative Delivery: Deliver a minimal viable flagging prototype in the first sprint, then incrementally add complexity (e.g., batch processing, severity levels).

Continuous Feedback: Engage stakeholders at each Sprint Review to ensure the system meets evolving needs and regulatory updates.

Transparency & Adaptation: Keep an up-to-date burndown chart and backlog; adjust priorities based on incoming change requests or edge-case discoveries.