Youth Safety Signal Application

Krissy Brown

INFO-C450 Systems Design

Midterm Demo

10/24/2025

Version: v0.3

Table of Contents

Proposal- Youth Safety Signal
INFO-C450
Krissy Brown
09/08/2025

Problem Statement

Parents, schools, and youth mentors often learn about online risks after harm happens. Warnings (bullying group chats, risky trends, grooming slang, fake accounts, self-harm challenges) are scattered across texts and social posts. Adults currently do not have one dedicated, simple, trusted platform to see local signals explained in plain English. This leads to slow response, repeated incidents, and anxiety for families.

Users' needs and challenges:

- Parents and guardians want quick, clear alerts without jargon.
- School staff and PTAs need a safe channel to share verified notices with families.
- Community leaders and coaches want to flag concerns without exposing minors' personal data.

Proposed Solution

A small web app where verified adults share and read local online-safety alerts:

- Safety Feed of short, structured posts (platform, category, city/state, risk level) with a plain-English summary and tips.
- Filters by platform (for example, TikTok, Discord), category (bullying, scams, grooming), location, and risk level.
- Rising list that highlights fast-moving issues using time-based upvotes.
- Glossary that explains youth slang and code words with safety tips.
- Report button and a simple moderation queue, plus verified badges for schools and PTAs.
- Optional weekly email of top local risks.

Business value and benefits:

- Faster awareness that can reduce incidents, shared language for families and schools.
- Low cost to run; easy to pilot with one school and expand later.
- Trust built with verified roles and no collection of minors' personal data.

User Experience

- A parent signs up with email, picks their city or school, and sees today's local alerts.
- They filter by platform and read a short summary with clear steps, guidance for conversations, and links to help.
- A counselor posts a new alert using a short form; it goes to a moderation queue and then appears with a verified school badge.
- Users can upvote important alerts, save them, and opt into a Friday email recap.

Data sources and access:

- Community submissions from verified adults (parents, staff, coaches).

- Public sources such as school newsletters, PTA emails, police and safety bulletins, and app safety centers.
- Public social posts that describe trends; links are referenced, not copied, and minors' content is not stored.

Objectives

- Identify and surface emerging local risks within 24–48 hours.
- Deliver clear, action-oriented tips that non-technical adults understand.
- Reach initial adoption of 50 or more parents and three or more school or club partners.
- Maintain privacy-first rules; no minors' personal data stored.

System Requirements

- Accounts and roles: email-verified users; optional school or club affiliation; admins can badge verified contributors.
- Content: Post fields include title, summary, platform, category, city or state, risk level, and links. Glossary terms have definitions and tips.
- Safety: report and flag queue, audit log, clear community rules, and rate limiting.
- Discovery: feed, filters, rising list, search, and weekly digest.

What users can do:

- Post a local safety alert using a short form.
- Browse recent / rising alerts, filter by platform, location, or risk level.
- Upvote, save, and report posts; read glossary definitions.
- Subscribe or unsubscribe to a weekly email.

Typical Customers:

- Parents & Guardians seeking up-to-date awareness.
- Schools, Counselors, PTAs monitoring threats and educating families.
- Youth Serving organizations and schools

Project Plan

- **Software:** Front end with HTML and CSS plus small JavaScript and HTMX; back end with Python and Flask; database with SQLite through SQLAlchemy; email via SendGrid free tier.
- **Hardware:** Developer laptop and one low-cost cloud instance or a university server.
- **Network:** HTTPS on host; normal broadband is sufficient.

Dev Stack

- **OS & Tools:** Windows PC, VS Code**,** Git/GitHub
- **Language/Runtime:** Python 3.12

- **Web Framework:** Flask 3 + Jinja2 templates
- **UI:** HTML5, CSS3, minimal JS, HTMX
- **Database:** SQLite 3 via SQLAlchemy + Alembic (migrations)
- **Auth & Security**: Flask-Login, form validation, basic rate limiting
- **Email/Notifications:** SendGrid (free tier)
- **Search & Ranking:** simple keyword search; time-decay "Rising"
- **Config:** python-dotenv
- **Testing & Quality:** pytest
- **Deploy:** Render (auto-deploy from GitHub) with gunicorn

Estimated Costs

- Domain: $15/year
- Hosting: $0 to $7/month (*student VPS*)
- Email (*SendGrid*): free tier or $15/month if volume grows.
- Database with SQLite: $0.
    - **Total cost:** $15–$37/month

Development Plan by week

- Weeks 1–2: Set up repository, authentication, and basic pages; draft content rules.
- Weeks 3–4: Post model and create form; Safety Feed list and detail; filters for platform and risk.
- Weeks 5–6: Glossary browse and search; upvotes; Rising ranking.
- Weeks 7–8: Report button; moderation queue; midterm demo.
- Weeks 9–10: Location tags and verified school or club badges.
- Weeks 11–12: Weekly email digest; small dashboard for Top 5 this week.
- Weeks 13–14: Usability test with five to seven adults; fix high-impact issues.
- Weeks 15–16: Security pass, bug fixes, polish; final demo.

Business Value & Stakeholder Rationale

- Clear value with safety alerts families can use.
- Low risk with privacy-first design and minimal data collected.
- Practical rollout that starts small, measures results, and expands as needed.

Stakeholders

- **Parents & Guardians:** Concerned with youth safety, contribute and receive alerts.
- **Trusted Contributors:** Verified users who provide reliable reports and insights.
- **Moderators:** Ensure data quality, review signals, verify reports.
- **Admins**: Manage system roles, glossary, risk levels, and users.
- **Community Organizations:** May use system data for outreach or interventions.
- **Researchers:** Analyze trends in youth safety threats.
- **Development Team:** Builds and maintains the platform.
- **Sponsors/Funding Partners:** Fund the development and care about outcomes.

Actors and Their Goals

Primary Actors:

- **Parent**
  - Submit or upvote safety signals.
  - Receive weekly digests of nearby safety threats.
- **Trusted Contributor**
  - Report credible safety signals.
  - Request verification for full contribution rights.
- **Moderator**
  - Review and moderate reported safety signals.
  - Verify or reject signal validity.
- **Admin**
  - Manage accounts, glossary, tags, and risk levels.
  - Oversee verification and platform maintenance.

Secondary Actors:

- **System**
  - Assigns roles, manages signal workflow, distributes alerts.
  - Enforces rules on risk levels, glossary term linking, and account roles.

Use Cases

Parent:

- **Receive Weekly Digest (2 pts)**
  The parent receives a weekly email summarizing the most upvoted or relevant youth safety signals in their region.
- **Report a Signal (2 pts)**
  The parent submits basic info about an observed safety issue without full details.
- **Request Verification (2 pts)**
  The parent submits personal or organizational credentials for trusted contributor status.
- **Submit Safety Signal (2 pts)**
  A detailed form is submitted including platform, tags, location, and risk level.
- **Upvote a Signal (2 pts)**
  The parent upvotes signals they believe are credible or important.
- **View Safety Feed (2 pts)**
  The parent browses and filters through public safety signals by region or category.

(Total: 12 pts)

Trusted Contributor:

- **Submit Safety Signal (2 pts)**
  Similar to the parent, but with verified user privileges.
- **View Safety Feed (2 pts)**
  Access feed to stay informed and engage with community signals.
- **Upvote a Signal (2 pts)**
  Boost visibility of important alerts.

(Total: 6 pts)

Moderator:

- **Moderate Safety Signal (4 pts)**
  Review incoming safety signals and either approve, edit, or reject them with notes.
- **View Safety Feed (2 pts)**
  See a live feed of signals, especially those requiring review.
- **Filter and Search Alerts (2 pts)**
  Use filters to identify trends or concerning patterns by tag, location, or risk level.
- **Upvote a Signal (2 pts)**
  Use voting power to help prioritize critical signals.

(Total: 10 pts)

Admin:

- **Manage Accounts and Roles (4 pts)**
  Add, update, or remove user accounts and change role assignments.
- **Manage Tags & Risk Levels (2 pts)**
  Update system-wide tags and calibrate risk level definitions.
- **Glossary Management (2 pts)**
  Define or edit glossary terms used in tagging or moderation.
- **Filter and Search Alerts (2 pts)**
  Similar to moderators, used to analyze trends and review flagged signals.
- **View Safety Feed (2 pts)**
  For oversight or escalation purposes.

(Total: 12 pts)

System:

- **Send Weekly Digest (2 pts)**
  The system automatically generates and sends digest emails.
- **Assign Tags Automatically (2 pts)**
  Based on content analysis or matching terms.

Plan of Work Status

Task	Status	Notes
Set up GitHub & Repo	✅ Done	Private repo created with README and dev branch.
Install Flask & tools	✅ Done	Flask 3, SQLAlchemy, Alembic, HTMX integration; SendGrid SDK installed.
Basic Auth + UI pages	🔄 In Progress	Routes drafted; Flask-Login wired; CSRF enabled; HTML templates drafted.
Verification Request + Admin Queue	🔄 In Progress	Simple profile action + admin list; store notes and reviewer in audit log.
Alert (Signal) model + form	🔄 In Progress	Will start with mock data before connecting to DB.
Safety Feed + Filters	⏳ Not Started	Display posts and allow filtering by risk/platform.
Glossary + Slang Index	⏳ Week 5–6	Basic glossary page with searchable terms.
Rising Alerts + Votes	⏳ Week 7–8	Optional: time-based upvote logic.
Report Button + Mod Queue	⏳ Week 8	Moderation logic and approval system.
Location Tags + Badges	⏳ Week 9–10	Display verified school or club badge.
Weekly Email Digest	⏳ Week 11–12	SendGrid weekly summary email.
Usability Test & Polish	⏳ Week 13–14	Feedback from 5–7 testers; fix key issues.
Final QA & Demo	⏳ Week 15–16	Security check, polish, and final walkthrough.

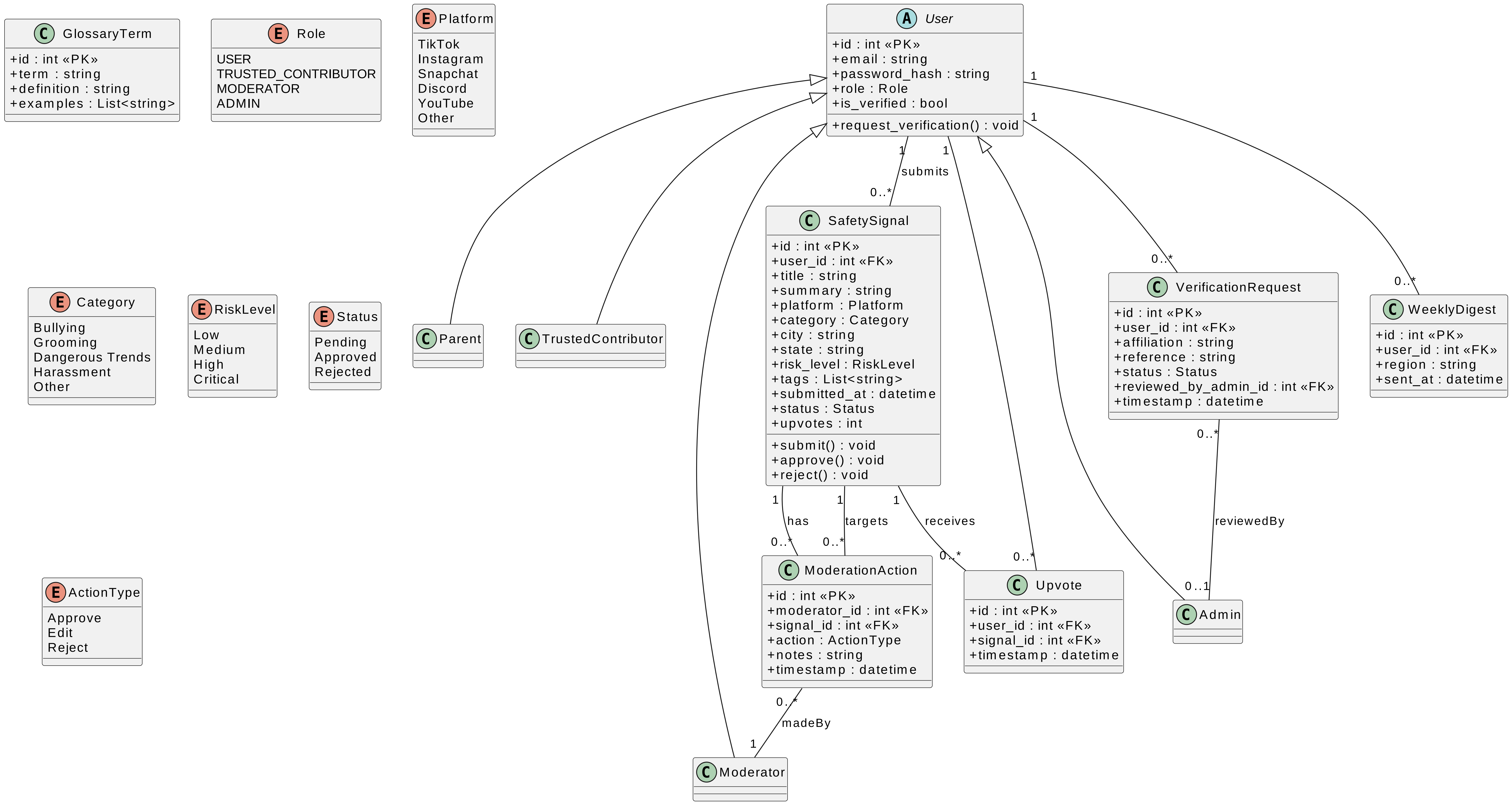
- **Update Status Automatically (2 pts)**
  E.g., automatically change status to "Pending Review" after submission.

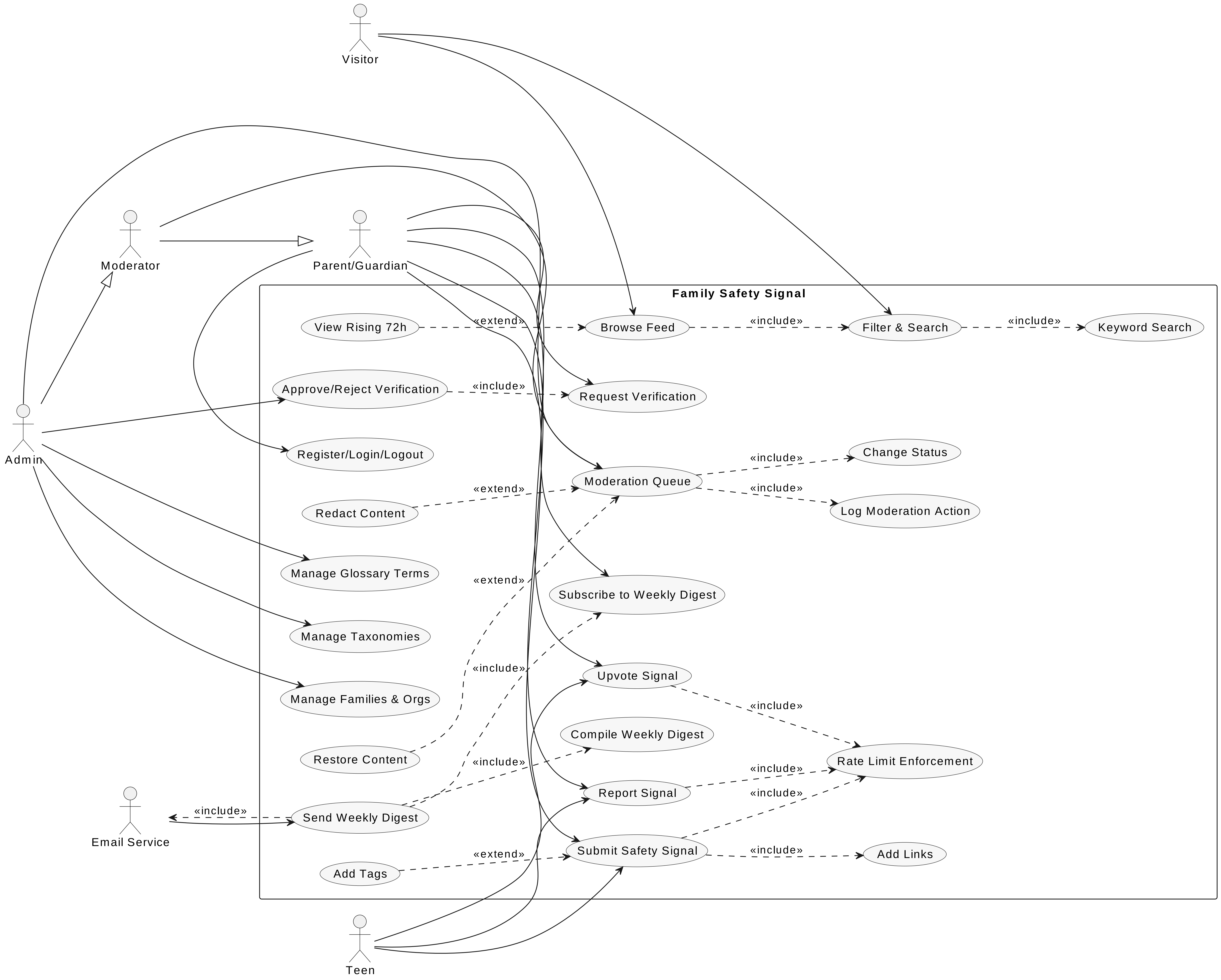
- **Notify Moderators (2 pts)**
  Triggers notifications to moderators for review queues.

(Total: 8 pts)

Actor	Points	Days of Work
Parent	12 pts	6 days
Trusted Contributor	6 pts	3 days
Moderator	10 pts	5 days
Admin	12 pts	6 days
System	8 pts	4 days
Total	**48 pts**	**24 days**

Youth Safety Signal - Class Diagram




Visitor
Moderator
Parent/Guardian
Admin
Email Service
Teen
Family Safety Signal
View Rising 72h
«extend»
Browse Feed
«include»
Filter & Search
«include»
Keyword Search
Approve/Reject Verification
«include»
Request Verification
Register/Login/Logout
Change Status
«include»
Moderation Queue
«include»
Log Moderation Action
Redact Content
«extend»
Manage Glossary Terms
«extend»
Subscribe to Weekly Digest
Manage Taxonomies
«include»
Upvote Signal
Manage Families & Orgs
«include»
Compile Weekly Digest
Restore Content
Rate Limit Enforcement
«include»
«include»
Report Signal
«include»
Send Weekly Digest
«include»
Submit Safety Signal
«include»
Add Links
Add Tags
«extend»

Customer Problem Statement & System Requirements
INFO-C450: Systems Design
Krissy Brown
Fall 2025

Youth Safety Signal

Customer Problem Statement

As a parent, I feel like I'm constantly behind when it comes to knowing what dangers my kids might be facing online. Dangerous TikTok trends, bullying in secret group chats, or even signs of predatory grooming often surface only after someone gets hurt. Social media posts often show up late or buried and there's no single trusted place to turn to when something feels off.

What I need is a safe, organized system where fellow parents, school staff, and community leaders can share early warnings in real time. If a new app is being misused, or a threat is circulating, I want to be aware of it before it impacts my family. I want to see what's trending in my school district, and I want it in plain language I can understand.

Glossary of Terms

Term	Definition
Safety Signal	A report about a potential online risk submitted by a verified user.
Risk Level	The severity rating of a Safety Signal: Low, Medium, High, Critical.
Trusted Contributor	A verified adult (teacher, counselor, PTA leader) with privileges to submit alerts, with visible badge icon. *Manual Verification in MVP*
Safety Feed	The main chronological list of approved Safety Signals with filters (s and search.
Rising	A highlight of fast-moving, recently upvoted issues in the last 72 hours.
Red Flag Term	A keyword/phrase/code work indicating a known online threat or risky behavior that links to explanation and glossary tips
Region Tag	A simple location label (City, State and/or School or Club) for localizing alerts.
Moderation Queue	Where new or reported posts are reviewed by a moderator before publication
Weekly Digest *(optional)*	A simple email with top alerts for a user's city/state.

Functional Requirements

No.	Priority Weight	Description
REQ-1	High	**Accounts & Roles:** Email/password auth with roles user, moderator, admin. Admin can toggle a verified contributor flag; users can Request Verification from profile (flows to admin queue).
REQ-2	High	**Submit Safety Signal:** Form fields like title, summary, platform (enum), category (enum), city/state (text), risk level (enum), links (0–3), optional tags.
REQ-3	High	**Moderation Queue:** New posts start Pending and require moderator approval. Moderators can edit/approve/reject and actions are logged.
REQ-4	High	**Safety Feed + Filters + Search:** List approved posts; add filters by platform, category, risk, and city/state; simple keyword search over title/summary.
REQ-5	High	**Report/Flag:** Authenticated users can report a post (reasons: inaccurate, sensitive, spam, other); Item moves to moderation.
REQ-6	High	**Privacy & Safety Rules:** No minors' personal data stored; visible community rules; redaction tools in moderation UI.
REQ-7	Medium	**Rate Limiting:** Basic per-account/IP rate limits for posting, upvoting, and reporting.
REQ-8	Medium	**Upvotes**: One upvote per user per post; count shown; used for ranking.
REQ-9	Medium	**Rising List:** Start with "most upvoted in last 72 hours" (time-windowed ranking).
REQ-10	Medium	**Glossary:** Browse/search red-flag terms; admins can add/edit terms and tips.
REQ-11	Low	**Verification Requests:** Users can submit a verification request with affiliation/reference; admins can approve/reject with notes; actions audited
REQ-12	Low	**Weekly Digests** *(optional):* Opt-in email with top local risks; respects user's city/state and categories.

Nonfunctional Requirements

Type	Priority	Description
Functionality	High	The system must support posting, viewing, filtering, and moderating Safety Signals without failures.

Usability	High	Interface must be intuitive and mobile-first to suit busy parents and educators.
Reliability	Medium	The system should handle 50+ concurrent users without downtime.
Performance	Medium	Feed pages must load in under 2 seconds on 4G or broadband.
Supportability	Low	Must support simple admin updates to glossary, tags, and risk levels.

User Interface Requirements

No.	Priority	Description
UI-1	High	Safety Feed view with filter bar (region, category, risk level)
UI-2	High	New Safety Signal form with fields: Title, Risk Level, Description, Tags, Region
UI-3	Medium	Glossary page explaining terms and tagging system
UI-4	Medium	Profile view for managing user preferences, digests, and past posts
UI-5	Low	Report export page for authorized school personnel

Plan of Work

Task	Status	Notes
Set up GitHub & Repo	✅ Done	Private repo created with README and dev branch.
Install Flask & tools	🔄 In Progress	Flask 3, SQLAlchemy, Alembic, HTMX integration; SendGrid SDK installed.
Basic Auth + UI pages	🔄 In Progress	Routes drafted; templates stubbed; Flask-Login wired; CSRF enabled.
Verification Request + Admin Queue	⏳ Not Started	Simple profile action + admin list; store notes and reviewer in audit log.

Task	Status	Notes
Alert (Signal) model + form	⏳ Not Started	Will start with mock data before connecting to DB.
Safety Feed + Filters	⏳ Not Started	Display posts and allow filtering by risk/platform.
Glossary + Slang Index	⏳ Week 5–6	Basic glossary page with searchable terms.
Rising Alerts + Votes	⏳ Week 7–8	Optional: time-based upvote logic.
Report Button + Mod Queue	⏳ Week 8	Moderation logic and approval system.
Location Tags + Badges	⏳ Week 9–10	Display verified school or club badge.
Weekly Email Digest	⏳ Week 11–12	SendGrid weekly summary email.
Usability Test & Polish	⏳ Week 13–14	Feedback from 5–7 testers; fix key issues.
Final QA & Demo	⏳ Week 15–16	Security check, polish, and final walkthrough.

Project Plan

Week	Dates (2025)	Deliverable / Task	Status	Notes / Risk & Mitigation
Week 1–2	Sept 1–14	Defined project concept (“Parent Safety Signal”) and conducted early user need exploration for parental awareness of online youth risks.	✅ Completed	Initial brainstorming completed with clear user pain points identified.
Week 3–4	Sept 15–28	Re-scoped and renamed the system to “Youth Safety Signal,” refining focus to a community-centered early-warning network. Drafted problem statement and key objectives.	✅ Completed	Instructor feedback incorporated to improve feasibility for course scope.
Week 5	Sept 29–Oct 5	Developed system documentation: functional/non-functional requirements, use cases, UML diagrams, and stakeholder roles.	✅ Completed	Strong foundational alignment between requirements and planned features.
Week 6	Oct 6–13	Created system flow diagrams and low-fidelity UI wireframes for core use cases	✅ Completed	UX emphasis on simplicity, trust, and clarity for parental users.
Week 7	Oct 14–20	Configured Flask environment (Python 3.14), established. venv, and organized full GitHub repository with structured folders (1_code–5_documentation).	✅ Completed	Environment verified locally; version control successfully integrated.
Week 8	Oct 21–27	Implemented functional mock prototype: Dashboard, submit form, and Moderation queue using in-memory data for midterm demonstration.	✅ Completed	Midterm deliverables tested locally and prepared for grading.
Week 9–10	Oct 28–Nov 10	Initiated MySQL environment setup and SQL Alchemy migration planning. Submitted IT access request for MySQL instance; ticket closed	⚠️ **Delayed**	Due to pending instructor response, database integration postponed. Prototype maintained via local mock data to preserve continuity. Mitigation: will migrate to

		without instructor verification.		MySQL manually post-midterm using local containerization
Week 11	Nov 11–17	Implement role-based authentication (Admin, Moderator, Parent) and update Flask routes for access control.	Planned	Will implement using Flask-Login after MySQL integration is complete
Week 12	Nov 18–24	Build Glossary of Safety Terms with search function and add filtering options on dashboard.	Planned	UX and accessibility refinement phase; moderate development workload.
Week 13	Nov 25–Dec 1	Conduct usability test with small sample group and incorporate improvements from feedback.	Planned	Risk: Limited participant time; mitigation via short, guided prototype demos.
Week 14–15 (Final)	Dec 2–15	Finalize MySQL database migration, complete all documentation, and record final project demo for submission.	Upcoming	Ensure synchronization between data layer and user interface before final handoff.

Summary

Despite external access delays related to MySQL setup, project development has remained on track through adaptive use of mock data and GitHub versioning. The core structures of the project are fully operational. Database migration and authentication logic are prioritized for November to ensure a functional and maintainable final system.

Change Log

Version	Date	Description of Change	Author
v0.1	September 2025	Initial project proposal drafted concept “Parent Safety Signal” defined; identified problem of fragmented online safety reporting for parents and schools. Requested MySQL Access.	Krissy Brown
v0.2	October 2025	Re-scoped project into “Youth Safety Signal” with focus on simplified demo prototype for midterm, finalized system architecture and core flow	Krissy Brown
v0.2.1	October 2025	Developed system requirements (functional & non-functional); added early use-case table, UML diagrams, and initial UI sketches.	Krissy Brown
v0.3	October 2025	Flask development environment configured (Python 3.14 + Flask 3.0); created repository structure with modular folders (1_code through 5_documentation). Implemented working mock prototype with templates for Dashboard, Submit, and Moderation pages using in-memory data.	Krissy Brown
v0.3.1	October 2025	Added favicon, README enhancements, launch configuration (F5 run support), and revised HTML + CSS interface with crimson/white color theme.	Krissy Brown
v0.3.2	October 24 2025	Created documentation folder and midterm deliverables (requirements, brochure, slides). Verified run instructions and finalized demo setup.	Krissy Brown
Planned v0.4	November 2025	Add SQL Alchemy models and migrate from SQLite (mock) to MySQL. Implement role-based authentication for Admin / Moderator / User.	Krissy Brown
Planned v0.5 (Final)	December 2025	Add glossary search, weekly digest email, usability testing, and finalize all diagrams and documentation.	Krissy Brown

References

1. Flask Documentation – *The Pallets Projects* (2025).
   https://flask.palletsprojects.com/
2. SQLAlchemy ORM Documentation – *SQL Alchemy Project* (2025).
   https://docs.sqlalchemy.org/
      i. *Note: Still requires MySQL approval from instructor to UITS*
3. Python 3.14 Standard Library – *Python Software Foundation* (2025).
   https://docs.python.org/3/library/
4. Bootstrap 5 Framework – *getbootstrap.com* (accessed 2025).
   https://getbootstrap.com/
5. Indiana University Luddy School of Informatics, Computing, and Engineering.
   INFO-C450 Systems Design Course Resources and Modules (Fall 2025).
6. Brown, Krissy (2025). *Youth Safety Signal GitHub Repository Documentation.*
   https://github.com/thekrissybrown/YouthSafetySignal_Midterm_KrissyBrown
7. Visual Studio Code Documentation – *Microsoft* (2025).
   https://code.visualstudio.com/docs
8. SendGrid Python SDK – *Twilio* (Planned Feature Reference).
   https://github.com/sendgrid/sendgrid-python