CS462 Senior Software Engineering Project II

Team Charter & Working Agreement

Date: 2026/02/15 Team ID: CS.019 Team Name: Natural-Language-Processing-of-Financial-Disclos ures	Team Members: ● Norman O'Brien (obrienno@oregonstate.edu) ● Trinity Paulson (paulsotr@oregonstate.edu) ● Rithik Reddy (nibbarar@oregonstate.edu) ● Hsun-Yu Kuo (kuohsu@oregonstate.edu)

Change Log

Date	Version	Sections Changed	Summary of Changes	Reason / Driver	Author(s)
2025-09-30	V1.0	All sections	Initial creation	Start of project	Team
2026-02-15	V 1.1	Team Roles & Responsibilities	Updated individual team roles & responsibilities, and decision-making structure.	Alignment with current workload distribution and project progress.	Team
2026-02-15	V 1.1	Definition of Done & Quality Gates	Expanded the Definition of Done to include enforced linting, formatting, static analysis, CI checks, and documentation requirements with explicit CI workflow references.	Alignment with existing CI pipeline	Team
2026-02-15	V1.1	Communication and Meetings	Updated weekly meeting time to 4:30 PM on Tuesdays.	Adjusted the schedule to better align with our project partner's availability this term.	Team

1. Roles and Decision Making

Specify team member roles (including leadership/rotation) and responsibilities. Define your decision-making model (e.g., consent, majority, unanimity) for different types of choices.

Hsun-Yu Kuo:

- **Role:** WRDS Data Access Lead / Meeting Coordinator
- **Primary Responsibilities:**
  - Manage WRDS data extraction and preprocessing.
  - Ensure the dataset we used for BERTopic is cleaned and organized.
  - Coordinate team meetings and communication.
  - Help with code debugging and support feature development.
- **Rotation plan**: Depending on team needs. May or may not rotate.
- **Contact info**: kuohsu@oregonstate.edu
- **Technical Decisions** — Majority Vote:

  I first attempted to reach consent through team discussion.

- **Minor Decisions** — Consent:

I will just let the assigned team member make the decision as long as no one has strong objections.

- **Major Project Changes** — Unanimity:

  I think the decisions that affect scope, deadlines, or core functionality require unanimous agreement from all team members.

- **Conflict Resolution** — Consent, Escalation if Needed:

  I attempt to resolve disagreements through discussion and consent. If unresolved, I consult our instructor or project partner.

Rithik Reddy

- **Role**: Head of Testing & Developer of BertTopic
- **Primary Responsibilities:**
  - Write and execute automated and manual test cases for BERTopic pipeline components.
  - Ensure all code changes adhere to the team's Definition of Done (DoD), including testing, linting, and doc updates.
  - Help catch bugs, edge cases, and regressions early during development, specifically for data preprocessing pipelines, topic modeling pipelines, and evaluation pipelines.
  - Write production code pertaining to BERTopic modeling, parameter selection, and results interpretation in addition to test cases.
  - Work with the Technical Lead to keep CI pipelines, automated test suites, and modeling validation checks stable.
- **Rotation Plan:**
  - Role rotates approximately every two months or earlier based on team needs and workload distribution.
- **Contact Information:** nibbarar@oregonstate.edu
- **Technical Decisions** - Majority Vote I make suggestions for technical decisions and discuss them with the group in order to gain agreement. If agreement cannot be reached after a reasonable amount of discussion, a majority decision will be used and noted on the issue or pull request.
- **Minor Decisions** - Consent-based. If a decision is low-risk or part of the defined process, the responsible team member can move forward with the decision after notifying the team and receiving no strong opposition.
- **Major Project Changes** - Consensus Decisions that change project scope, timing, or major features require consensus from all team members.
- **Conflict Resolution** - Consent with Mediation. If involved in a conflict, I will attempt to discuss the situation with the people involved in order to gain consent from all parties. If consent cannot be reached, I will escalate the concern to my instructor/project partner.

Norman O’Brien

- **Role:** Documentation Lead / Head of Architecture / Tester
- **Primary Responsibilities:**
  - Manage proper use of documentation
  - Ensure project design specifications are met
  - Be a second line of testing after Rithik
  - Write code for Gemini AI API calls

- Write code for document translation

- **Rotation Plan:**
  - Open to rotating depending on the needs of the team
- **Contact Information:** obrienno@oregonstate.edu, +12069724350
- **Technical Decisions -** Majority Vote; suggestions can be made for big decisions and the team meets and takes everyone's suggestions into account before making final technical decisions
- **Minor Decisions -** Consent; low risk decisions can be discussed briefly and verbally confirmed by multiple team members.
- **Major Project Changes -** A consensus is required for any large changes to project scope, timing, or major features
- **Conflict Resolution -** Consent with Mediation; professional discussion can be had about conflicts and ideally a consensus is made, but if not, the instructor and/or project partner can be notified

Trinity Paulson

- **Role:** Project Manager/Developer
- **Primary Responsibilities:**
  - Organize meetings
  - Compare design plans to implementation
  - Development of proper output
  - Take meeting notes.
- **Rotation Plan:**
  - Rotation depending on the needs of the team
- **Contact Information:** paulsotr@oregonstate.edu
- **Technical Decisions -** Majority Vote
- **Minor Decisions -** Consent
- **Major Project Changes -** A consensus
- **Conflict Resolution -** Consent with Mediation

2. Communication and Meetings

List your meeting cadence (times/days), communication channels, and core tools. Link to your meeting agenda/notes template.

Weekly Team Meeting: Sundays/Tuesdays 2:00–4:00 PM PT via Discord, to review weekly accomplishments, identify blockers, and outline next steps for the following week.

Mentor Check-ins: Every Tuesday at 4:30 PM via Zoom to update the stakeholder on weekly progress, discuss future plans, and confirm approval of recent changes or gather feedback on new requirements.

Async Updates: Discord posts short daily progress updates.

Response Expectation: 6h for messages, 24h for pull-request reviews.

Core Tools:

- GitHub (repo, issues, Actions CI/CD)

- Discord (team communication)
- Google Docs / Drive (draft documents)
- Github Projects (sprint tracking & task board)

3. Definition of Done (DoD) and Quality Gates

Define your quality bar. List required checks (tests, code review, security/lint/static analysis, docs updates) and reference the exact CI/CD gates that enforce them. Point to your CI configuration, linter/formatter config, etc.

Code Review: All changes must be submitted through a pull request and reviewed by at least one other team member before merging (feature/* → PR → ≥1 approval → merge).

Testing & Continuous Integration: Code must pass all automated tests and required CI checks before being merged into main.

Linting: Code must pass all lint checks before merge. Linting is configured in Link_check.yaml and enforced through the CI workflow.

Static Analysis: Static type or code analysis must pass before merge. This includes type checking or other static validation tools configured in the project.

Security Checks: Dependency vulnerability scanning and security checks must pass before merging. Any high-severity vulnerabilities must be resolved before approval.

Documentation Updates: Team design documents must be updated to reflect any structural or functional changes, and any new functionality must be documented in the project's GitHub README with appropriate setup or usage instructions.

CI/CD Enforcement: All Definition of Done requirements are enforced through the project's CI workflow (main_build.yml), which runs automated tests, linting, formatting, and static analysis checks and blocks merges if any checks fail.

4. Conflict Resolution & Accountability

Write a restorative, stepwise plan for resolving conflict. Define how you will use evidence (PRs, reviews, attendance) to hold each other accountable.

Triggers (one of these)

- Missed deadlines or incomplete assigned tasks
- Lack of communication or absence from scheduled meetings
- Code merged without required testing or review
- Repeated unresponsiveness to team updates or feedback

Stepwise Restorative Actions

- **Step 1 - Private Conversation:**
  Within 3 days of the issue, the affected members will have a respectful, private discussion to understand the root cause and reset expectations. The goal is support and clarification, not blame.

- **Step 2 - Team Discussion and Adjustment:**
  If the issue continues, it will be addressed during a team meeting. Responsibilities may be clarified or redistributed, and concrete action steps with deadlines will be agreed upon.

- **Step 3 - Formal Documentation and Escalation:**
  If behavior persists after one week, the concern will be formally documented and shared with the instructor or project partner for guidance. Responsibilities may be reassigned if necessary to protect project progress.

Timelines

- Issues are addressed within 3 days of occurrence.
- Follow-up progress is reviewed at the next weekly meeting.
- If unresolved after one week, escalation or reassignment may occur.

Objective Evidence for Accountability

We use transparent project artifacts to ensure fairness and accountability:

- Pull Requests (PRs): Track feature completion, review participation, and testing compliance.
- Code Commits: Show contribution frequency and development activity.
- PR Reviews: Demonstrate engagement in code quality and collaboration.
- Meeting Logs & Attendance Records: Track participation and ensure consistent communication.
- Issue Tracker / Project Board: Monitor assigned tasks and completion status.

5. Accessibility & Inclusion

Describe meeting norms (time zones, note-taking, turn-taking), how to request accommodations, and how you will surface and address barriers.

Norms:

- Meetings are pacific standard time
- We have a shared document specifically for meeting notes that anyone can add to
- Meetings with project partner we take turns based on the order he asks for updates
- TA meeting turns are guided by the TA

Accommodations:

- If desired, accommodations can be directly discussed with the team, the project partner, and instructors if need be

Barriers:

- If someone can not attend they can give advance notice and notes will be shared with them.
- If someone can not attend an in-person meeting they can give advance notice and an online version will be offered.
- Any technical or scheduling issues that may require a change in multiple meeting plans will be brought up in advance and new plans will be discussed.

6. Risk Management

Identify key risks to the team process and define an escalation path for issues.

Risks and Escalation Path: If a teammate identifies a potential risk, they should post a message on Discord for the team to review

Risk	Early Warning Sign	Escalation Path	Timeline	Evidence
Missing deadlines	Incomplete tasks or delayed PRs before the sprint review	Private reminder →Discuss in team meeting → Escalate to mentor if repeated	Address within 2 days of the missed due date	GitHub issues, commit timestamps
Communication issues	No message replies or absence from meetings for more than 48 hours	Direct message → Team discussion → Escalate to instructor if no response after 1 week	Check in within 2–3 days	Discord chat logs, attendance records
Repeated CI failures	Consecutive failed runs in GitHub Actions on the main branch	Developer investigates → Team debugging → Escalate to TA if persists after multiple failed runs	Within 2 days of the first failure	GitHub Actions logs
Code failure	Tests fail or integration breaks after the merge	Assign debugging pair → Review during next team meeting → Whole team tries to fix code → Escalate to TA /Stakeholder if unresolved	Within 1 day	Test results, PR review comments
Data issues	WRDS filings missing, incomplete,	Notify team → Stakeholder → OSU	Immediately upon detection	Error logs, dataset validation script

	or API fetch errors	WRDS manager		
AI API issues	Timeout errors, invalid responses, or API quota exceeded	Retry and debug → Report to TA/Stakeholder→ look for an alternate model or increase in funding	Same day as the issue	API logs, error reports