Development Plan
SFWRENG 4G06 - Capstone Design Process

Team 17, DomainX

Awurama Nyarko
Haniye Hamidizadeh
Fei Xie
Ghena Hatoum

Table 1: Revision History

Date	Developer(s)	Change
September 17, 2025	Awurama Nyarko, Fei Xie, Ghena Hatoum, Haniye Hamidizadeh	Created first draft of document
November 17, 2025	Fei Xie	Implemented TA feedback issues

This document outlines the development plan for the project. Covering details on Intellectual Property, team roles and expectations, workflow plan, and project scheduling. With additional sections that include the expected technologies and coding standards. This document will help ensure all members are aware of the expectations on the team, and help provide a roadmap for the project.

1 Confidential Information

This project will be open-source, no confidential information to protect.

2 IP to Protect

This software and associated documentation files for this project are protected by copyright as dictated by the MIT License.

3 Copyright License

This project use's the MIT License, located in the LICENSE file.

4 Team Meeting Plan

Meeting minutes and attendance will be taken for all meetings as their related issue in the project's GitHub. Each member is responsible for verifying their own attendance in the related Github issue.

4.1 Lectures

Members are expected to attend every **lecture** for Software Engineering Students **in-person** (maximum 3/week).

4.2 Tutorials

Members are expected to attend all TA-lead tutorials. For tutorials without a pre-designated lesson plan, the team will decide what to do, such as having a check-in, working session, and etc.

4.3 Team Meetings

Weekly team meetings are held every **Monday 10:30-11:20pm virtually**. All members are expected to attend, and it is up to the responsibility of the individual to inform the rest of the team why they cannot make the weekly meeting, and provide their updates asynchronously. Meetings will be structured as follows:

1. Each member will give a short recap of the work they've done since the previous meeting
2. Each member has the opportunity to bring up any blockers, concerns, or issues they are facing
3. Group will discuss and determine action items for the project that are to be done by the next meeting, and assign them to member(s).
4. Meeting minutes will be taken and recorded as a corresponding Github Issue.

4.4 Supervisor and Domain Expert Meetings

Weekly supervisors meetings are held every **Thursday 2:30-3:30pm in-person**, unless otherwise stated. Meeting topics must be prepared beforehand. One team member will be designated as the **meeting chair** for the meeting. They are responsible for coordinating the meeting between the supervisor/domain expert and the rest of the team, including sending the meeting invite on Microsoft Calendar.

5 Team Communication Plan

- **Discord:** Main way of communication between team members.
- **Microsoft Teams:** Secondary way of communication between team members, used for online meetings.
- **GitHub:** Tracking of the project process and meeting minutes. GitHub issues will be created for all meetings (Lecture, Peer Review, Supervisor Meeting, etc.) and code tasks.
- **Email:** Communication with the supervisor and domain expert, with all team members cc-ed.

6 Team Member Roles

- **DevOps Lead:** Responsible for maintaining the GitHub repository to properly track the development process. Updating and maintaining the GitHub Project Board, and ensuring issues are properly assigned the correct milestone and project.
- **Team Leader:** Oversees the overall process of the whole project. Ensures both subteam are on track. Creates and assigns issues, delegates work. Coordinates meetings between team members and all stakeholders.

- **Research Lead:** Main lead for the research subteam. Responsible for coordinating and identifying work required for the research paper portion of the project.
- **Tech Lead:** Main lead for the tool subteam. Responsible for coordinating and identifying work required for the tool creation of the project.
- **Note Taker:** In charge of creating issues for all meeting types. Takes meeting minutes for the team during meetings.
- **Meeting Chair:** Chair for meetings involving stakeholders. Responsible for creating the agenda before each meeting, leading the meeting and guiding discussion topics.
- **Peer Reviewer:** Reviews artifacts for other team and provides feedback.

Role	**Assigned To**
DevOps Lead	Fei
Team Leader	Fei
Peer Reviewer	Fei
Research Lead	Awurama
Tech Lead	Haniye and Ghena
Note Taker	Ghena and Fei *(rotating every few weeks with Haniye and Awurama)*
Meeting Chair	Fei and Ghena *(may rotate to Haniye or Awurama if needed)*

7 Workflow Plan

The main branch of the project should be protected. Meaning it should not be directly committed into, all changes must be merged in via a pull request. Following are the general steps of what development should look like:

- Create a new **Project Work** issue, ensure the following are assigned:
  - Assignees: Who is working on the project, multiple members can be assigned.
  - Label: Assign the appropriate label for the issue:
    - *feature:* Adding new functionality.
    - *refactor:* Updating existing code without altering functionality.
    - *bug:* Error found in code.
    - *documentation:* Updating documents, no code changes.
  - Milestone: Assign to the related project deliverable.
  - Projects: This should be automatically set to DomainX, if the Project Work issue template was used.
    - If not, add the project label to the issue and set the project to DomainX if not done automatically by the workflow action.

- Create a branch from the issue, branch name should include the issue number.
  - By default, the issue branch should be created from the epic branch, usually related to the milestone.
  - For bug type issues, the branch can be created directly from the main branch.
  - Branch naming format: **{Issue #}**-name-of-branch
- Commit changes with descriptive names.
- Add tests:
  - **Unit tests** must be created for feature, refactoring, and bug issues.
  - **Integration tests** must be created for feature issues, optional for refactoring and bug.
  - A **coverage tool**, such as `coverage.py`, will be used to track which parts of the code are tested and to identify any untested code paths.
- Create pull request
  - All GitHub Actions need to pass
  - At least one approval from a teammate
  - When merging, make sure to do **squash and merge**
- Once the epic branch contains the related issues, the epic branch can be merged into main branch.

8 Project Decomposition and Scheduling

Project Scheduling

We will use GitHub Issues to organize both meeting notes and project development tasks. The team's GitHub Project, **DomainX**, will track non-meeting issues such as feature development, bug fixes, and documentation updates. Each deliverable will be broken down into smaller tasks, assigned to team members, and linked to the relevant milestone. This setup will help us keep track of progress and make sure deadlines are met.

The overall schedule follows the course timeline:

Deliverable	Week
Team Formed, Project Selected	Week 03
Problem Statement, POC and Dev Plan	Week 04
SRS + HA Revision 0	Week 06
V&V Plan Revision 0	Week 08
Design Document Revision -1	Week 10
Proof of Concept Demonstration	Week 11–12
Proof of Concept Team Contribution	Week 11–12
Design Document Revision 0	Week 16
Revision 0 Demonstration	Week 18–19
V&V Report Revision 0	Week 22
Final Demonstration Revision 1	Week 24
EXPO Demonstration	Week 26
Final Documentation Revision 1	Week 26

9 Proof of Concept Demonstration Plan

As with most projects, there are certain risks that could affect how successful ours will be. Below are the main risks we have identified and plan to address:

Data Access & API Limitations

One of the risks in our project is accessing the correct data through APIs. We may face challenges finding an API that provides the information we need in the way we require. In addition, the API we choose could be unreliable or return data in a format that is hard to process, such as poorly structured JSON or XML. There may also be restrictions, such as requiring special keys or authentication, and many APIs limit how many requests can be made within a given time. Any of these issues could make it difficult to collect enough usable data for our tool.

Integration Risk

Another risk is that even if each component works individually, combining them into one complete workflow could cause unexpected issues. For instance, data collected from the API might not store properly in the database, or the way the data is structured in the database may not connect smoothly with the visualization tool. These kinds of problems could lead to delays and require extra adjustments during development.

Lack of open-source packages or hard to find source codes

As we are non-domain experts for the domain of Neural Network libraries, it's hard to gauge the available packages accurately. We may come to find that only a small subset of library/packages commonly used are open-sourced or have accessible source code. This could hinder data gathering as we work with our supervisor and domain expert to find alternatives or obtain the appropriate licensing.

Difficulty communicating with the Domain Expert
Due to the busy schedule of both the team and the domain expert, it may be hard to coordinate and plan meetings. This could result in longer than desired breaks between meetings to catch up with the domain expert. With the risk of a small misaligned spiraling larger between the meetings.

Data Management & Database Integration
Databases are widely used, but there are still risks when applying them to our specific case. One challenge is designing a schema that fits the data we collect from APIs, since the structure may not align well with a relational database. There is also the possibility of data integrity issues, such as duplicate or missing values, which could lead to an unreliable dataset. In addition, if the amount of data grows larger than expected, the database may struggle with performance and make queries too slow for our application.

Visualization & Graphing
Another risk is related to how we present the final results. If the data cannot be shown in a clear and effective way, the value of the tool is greatly reduced. The graphing library we choose might not fully support our data structure or may not integrate well with the rest of the system. There is also the chance that creating complex graphs with larger datasets could affect performance and make the tool slow to use. Finally, the library we select may not provide the types of charts or visualizations that best highlight the comparisons we need to show.

To address these risks, our proof of concept will focus on showing that the main components of the system can work together in practice.

For the risk of data access and API limitations, we will build a small script that connects to the intended API endpoint, displays the raw response, and then parses the data into a structured format such as a Python dictionary. The parsed data will then be inserted into a database and used to generate a simple visualization, confirming that the entire pipeline from retrieving external data to processing, storing, and displaying it functions as expected.

To address the risk of library packages, we will coordinate with the domain expert as early as possible. In addition to scheduling meeting at minimum a week in advance, allowing for adequate time for both the team and the domain expert to determine our availability.
Finally, for the risks around database and graphing, we will demonstrate that parsed data can be stored in a suitable schema, retrieved through simple queries, and displayed in a basic graph using our chosen visualization tool. If we find during the PoC that the relational database struggles with large datasets or that the schema design limits flexibility, we will consider exploring a NoSQL option such as MongoDB as a backup to reduce these risks.

10 Expected Technology

Our project will be a web-based application with a React frontend to handle interaction and show graphs, Django backend to process data and manage requests, and a MySQL database to store the results. We plan to explore deploying the application on a cloud platform such as Amazon Web Services (AWS) or Google Cloud later in the project to make collaboration and access easier for the team and supervisors.

We plan to use Requests for making API calls and Pandas to organize and clean the data before saving it on the backend. NumPy might also be used for simple calculations. On the frontend, we expect to use a library like Chart.js to build the graphs and comparisons.

We will use Git and GitHub for version control. Our workflow will be based on creating GitHub Issues for tasks, linking the related work to those issues, and developing each feature or fix on a separate branch. Once the work is ready, it will be merged into the main codebase through a pull request. We will also use GitHub Projects to track tasks and progress.

For testing, we plan to start with simple manual tests to confirm that each part of the system works as expected and later explore automated testing as the project grows.

Most of the important logic for evaluating the libraries will be written by us, while tools and libraries will be used as a help.

For the backend, we'll follow Python's official style guide, PEP 8, to keep our code clean and consistent.

To help automate adherence to PEP 8 and maintain high code quality, we will use the following tools:

- **Black** – an automatic code formatter to ensure consistent style across all files.
- **isort** – to automatically sort and organize import statements.
- **Flake8** – a lightweight linter for detecting syntax errors and style violations.
- **Pylint** – a feature-rich linter for deeper static analysis and code quality checks.

On the frontend, we'll use React with TypeScript. For this, we'll rely on the React TypeScript Cheatsheet for common patterns and Google's TypeScript Style Guide.

Appendix — Reflection

1. Why is it important to create a development plan prior to starting the project?
   Ghena: It makes sure everyone is on the same page, knowing what their roles and responsibilities as well as holding them accountable. It also gives clear guidelines of how everything should be done and organized making it easier to track changes and improves the overall collaboration.
   Fei: To consolidate the team's expectation before the project starts, allowing for everyone to start at the same level of understanding of what the expectations are. This also allows for better accountability, having a pre-determined development plan beforehand implies that everyone is aware of the expectations before the project even starts.
   Awurama: It ensures that roles, deliverables, and timelines are clearly defined from the beginning. This prevents misunderstandings and helps the team stay aligned on expectations.
   Haniye: Having a development plan is important because it makes sure we've thought through every part of the project. It helps us set deadlines, check if the project is actually doable, and plan for any risks with a response in mind. It also gives us a chance to see which parts should stay, change, or be dropped. It's also easier to know how to handle client meetings and decide what requests we can accept or which ones we should say no to.

2. In your opinion, what are the advantages and disadvantages of using CI/CD?
   Ghena: It gives results faster as well as detect bugs earlier. However, a disadvantage is that there might be a learning curve, which, in those cases, would be better to allow the developer to learn a system and then implement.
   Fei: As projects get bigger with bigger moving components, having a well documented CI/CD process reduces the paranoia involved when a user releases new features. Thus, CI/CD allows for faster iterations of the design/development process, and potentially reducing downtime. Some obvious disadvantage is that due to the faster development cycle, developers can feel more stressed, constantly having to work sprint by sprint. Another disadvantage is the initial setup phase, where moving process and change how people are used to working can take some time, which can slow down the development cycle during the initial stages.
   Awurama: CI/CD improves consistency and reduces integration errors by automating builds and tests. However, it can introduce overhead if the setup is complex or if the team has limited experience maintaining pipelines.
   Haniye: The biggest advantage of CI/CD is that it keeps us on track with what the client needs. Since we test and confirm features at each step, we know we'll have a working project by the end. It also helps us stick to

deadlines and avoid last minute rushing. It lets us build the final solution little by little, so we can improve along the way. But on the downside, deadlines can feel tighter and more stressful. Clients might also change their minds after seeing part of the work done, or they might ask for more features. That means we'd have to adjust the old part while still working on the next step, which can put a lot of pressure on the team.

3. What disagreements did your group have in this deliverable, if any, and how did you resolve them?
   Ghena: N/A
   Fei: N/A
   Awurama: Our group did not have major disagreements, but we had different preferences for tool usage and formatting. We resolved these by reviewing the course checklists together and aligning on a standard approach.
   Haniye: N/A

Appendix — Team Charter

External Goals

Our team expects at a minimum to get a good grade for capstone, aiming for A+. Additionally, to write and publish a research paper with our name on it, and hopefully have it cited within academia. And the tool itself will hopefully be used by future students working on other State Of Practice assessments for future domains.

Attendance

Expectations

Our team expects full commitment to all scheduled meetings, with everyone arriving on time and staying for the entire duration. It is up to the responsibility of the individual to identify if they cannot make a meeting, and inform the team beforehand. If it is an important meeting, it is the responsibility of the individual to suggest an alternative time that all members can attend. Missing meetings frequently without prior notice will be addressed by the team and escalated to the supervisor if necessary.

Acceptable Excuse

Acceptable excuses include unforeseen emergencies, personal illness, family matters, or other significant personal obligations. Unacceptable excuses include vague, or last-minute reasons such as forgetting and conflicting non-essential plans.

In Case of Emergency

In the event of an emergency that prevents a team member from attending a meeting or completing their assigned work for a deliverable, the team member must inform the team as soon as possible through the team's designated communication channel (Microsoft Teams). This will allow for adjustments to be made, such as redistributing tasks or rescheduling the meeting if necessary. For deliverables, if the emergency impacts a deadline, the team member should notify both the team and the professor promptly to ensure that any necessary arrangements are made without affecting the team's progress/grades.

Accountability and Teamwork

Quality

Our team has the following expectations regarding the quality of preparation for meetings and the deliverables brought to the team:

- **Meeting Preparation**:

  - Team members are expected to arrive at meetings fully prepared, having reviewed relevant materials and the agenda for the meeting.
  - Each member should come ready to discuss their progress, share insights, and address any challenges they are facing.
  - Members should ensure that their updates are clear and concise, allowing meetings to stay focused and productive.
  - For online meetings, members should also turn on their camera's to promote team cohesion.

- **Deliverables Quality**:
  - All deliverables must meet the team's agreed-upon standards, demonstrating a high level of accuracy, thoroughness, and attention to detail.
  - Each deliverable should be carefully reviewed by each member before submission to avoid any errors or incomplete work.
  - Deliverables must align with the project's requirements and deadlines, ensuring they are both functional and meet the expected quality criteria.

- **Accountability and Feedback**:
  - Team members are responsible for completing their work to a high standard, communicating any issues early if they need assistance or more time.
  - Feedback on deliverables should be welcomed by all members, and revisions should be made within 7 days to improve the overall quality of the team's output.

Attitude

Our team has established the following **expectations** for team members' contributions, interactions, and cooperation to ensure a productive and respectful working environment:

- **Respectful Communication**: All team members are expected to listen to each other's ideas and provide constructive feedback. Communication should remain respectful, even in cases of disagreement.

- **Open Collaboration**: Each member is encouraged to share their ideas openly. Everyone should be willing to collaborate and help each other achieve team goals.

- **Accountability**: Team members are responsible for completing their tasks by the agreed-upon deadlines. If a member is struggling, they are expected to ask for help or communicate early.

- **Positive Attitude**: Maintaining a positive attitude, especially in challenging moments, is essential for team morale. Each member should encourage and support their teammates.
- **Commitment to Quality**: Every team member is expected to contribute to the project with their best effort, ensuring that the final product reflects high standards of quality.

We adopt the following **code of conduct** to guide behavior and interaction among team members:

- **Inclusivity**: Our team values diversity and is committed to creating an inclusive environment where everyone feels welcome and valued, regardless of background, experience, or opinion.
- **Professionalism**: Members will engage professionally, refraining from any inappropriate or offensive language or behavior. This applies to both in-person and online interactions.
- **Collaboration and Feedback**: We encourage constructive feedback and expect team members to accept and provide feedback in a way that helps everyone grow. Criticism should be focused on the work, not the individual.
- **No Tolerance for Harassment**: Harassment of any kind will not be tolerated. Any issues will be reported immediately and addressed in a structured manner.

To manage conflicts or disagreements that may arise during the project, we have a **conflict resolution plan** in place:

1. **Address the Issue Directly**: If a conflict arises, the involved members should first try to resolve the issue directly through a respectful discussion.
2. **Mediation by a Neutral Member**: If the conflict cannot be resolved, the team will appoint a neutral team member to act as a mediator to facilitate a discussion and find common ground.
3. **Escalation to Instructor/TA**: In the event that the conflict cannot be resolved within the team, the issue will be escalated to the instructor or TA for further guidance and resolution.
4. **Follow-Up and Monitoring**: After resolving the conflict, the team will continue to monitor the situation to ensure that the issue does not resurface and that team dynamics remain positive.

By adhering to these expectations, the code of conduct, and our conflict resolution plan, we aim to maintain a positive, collaborative, and respectful team environment.

Stay on Track

To keep our team on track, we will implement the following methods:

1. **Regular Check-ins and Progress Updates**: We will hold *weekly meetings* where each member will provide an update on their tasks and progress and any concerns or troubles they faced. These updates will help us identify issues early and adjust accordingly to stay on schedule.
2. **Performance Metrics**: We will track the following key metrics:
    - *Attendance* at meetings and check-ins will be documented through Issues on GitHub.
    - *Commits to the repository*, ensuring steady contributions.
    - *Task completion rates*, ensuring deadlines are met.
3. **Rewards for High Performers**: To encourage good performance, we will recognize and celebrate team members who meet or exceed expectations (completing more than the agreed upon work for the week, taking charge in organizing, planning, leading discussions in order to further the project process). Informal rewards may include public recognition during meetings and can choose the next activity for the team bonding activity.
4. **Managing Under performance**: If a team member's performance is below expectations (not completing the same amount of work as every other team member for more than 3 weeks):
    - We will start with a *team conversation* to understand any obstacles and offer support.
    - If under performance continues, consequences may include *more tasks* for milestone or in severe cases, a meeting with the TA or instructor.
5. **Consequences for Not Contributing**: If a team member does not contribute their fair share:
    - They may be assigned additional *tasks* to balance the workload.
    - In serious cases, the issue will be brought up to the TA or instructor.
6. **Incentives for Meeting Targets Early**: Members who consistently meet or exceed their targets will be rewarded with more desirable tasks as per their wants, such as leadership roles in key project components, helping to build their leadership experience. They will get first pick on tasks for the next team milestone.

Team Building

For team building, we will have a bi-weekly hangout session, where the team can get together to grab food, drinks or other activities. As decided by the team.

Decision Making

In our group, our primary way of making decisions will be through consensus. We believe that it is important to include everyone in the decision-making process, so it can lead to better outcomes and strong group work. In certain situations where consensus cannot be reached, the group will take a vote and each member will have equal say and the decision will be based on the majority rule. We will make sure all group members had a chance to voice their opinions before making the final decision through consensus or a vote.

To Handle disagreements: The team will address each disagreement directly and respectfully.

1. Allow all team members to express their concerns and opinions without interruption, ensuring everyone feels heard.
2. Keep the focus of the discussion on the topic at hand rather than personal feelings.
3. When necessary, we may appoint a neutral party to facilitate the discussion and help guide it to a resolution.
4. If a resolution is not found or the disagreement persists after the resolution is found, we will aim to revisit our project goals and objectives to ensure that our decisions align with our common purpose.

By following these strategies, we aim to maintain a collaborative and positive team environment while effectively managing decisions and conflicts.