Development Plan
SFWRENG 4G06A

Team #25, RapidCare
Pranav Kalsi
Gurleen Rahi
Inreet Kaur
Moamen Ahmed

September 24, 2024

Table 1: Revision History

Date	**Developer(s)**	**Change**
24–09–2024	Inreet	Introduction
24–09–2024	Inreet	Confidential Information
24–09–2024	Inreet	IP to Protect
24–09–2024	Gurleen	Copyright License
24–09–2024	Gurleen	Team meeting Plan
24–09–2024	Gurleen	Team Communication Plan
24–09–2024	Inreet	Team member Roles
24–09–2024	Inreet, Pranav	Workflow Plan
24–09–2024	Inreet	Project Decomposition and Scheduling
24–09–2024	Pranav	Proof of Concept Demonstration Plan
24–09–2024	Pranav	Expected Technology
24–09–2024	Gurleen	Coding Standards
24–09–2024	All	Reflection
24–09–2024	All	Team Charter
16–03–2025	Gurleen Rahi	Copyright License Update + Expected Technology Update

This document will provide details about various essential aspects of our development plan. It covers information about critical areas such as confidential information, IP, and copyright considerations ensuring compliance with ethical standards and regulatory requirements. In addition to this, we will outline the team meeting plan and team communication plan, and team member roles. We will include a workflow plan that describes how we plan to use Git and GitHub projects and project decompositions and scheduling. Moreover, we have included a proof-of-concept demonstration plan along with the technology stack and coding standards we wish to use for the development of our project. Finally, we will reflect on our learning from this exercise and include a team charter that defines our goals, expectations, and decision-making strategy.

1 Confidential Information?

Our project would not use any confidential data and information from the industry. We plan to use publicly available data in the development of this application, ensuring there is no breach of confidentiality.

2 IP to Protect

There is no IP to protect this project at the current stage. All work during the development phase will be original and no existing patents or proprietary technologies are involved.

3 Copyright License

The team has chosen a GNU GPL License for this project. This means that the copyright holder allows others to use, copy, modify, and distribute the software under some conditions. The license file in our Github repository has been updated with the same.

Link – License File

4 Team Meeting Plan

The team will have in-person as well as virtual meetings (on Microsoft Teams) to discuss different stages, and milestones, and track progress throughout the duration of the project. We will meet at least once every week to discuss the current stage of the project as well as update ourselves on other team members' work. Meetings with the industry advisor will depend on their availability. However, the mode of meeting will be in-person. We create an issue for team meetings on the GitHub project – capstone metrics where we list the agenda of the meeting, meeting minutes and action items, i.e., the next steps.

5 Team Communication Plan

The team members will communicate with each other using Microsoft Teams for meetings. Before every team meeting, an issue is created in the GitHub repository which contains the attendance, meeting agenda, meeting minutes and action items. The issue is closed once the meeting is done. Team members can review the contents of the issue for future reference. The team will use Microsoft Teams chat for general communication. In addition to this, the team will provide feedback and review comments for code and documentation-specific changes using issues through GitHub projects. Once a member has committed their changes at least two other team members will review the changes and add feedback to the linked issues. The team member will then raise a pull request with the feedback incorporated. The PR is finally reviewed by another team member and merged into the main branch.

6 Team Member Roles

The team will assign specific roles for the project after the preliminary elicitation and design thinking process is complete. Currently, the team would have the following distribution of administrative roles.

- **Project Leader** – Pranav Kalsi
  - Prepare agenda for team meetings, ensuring all relevant topics are covered and goals are clear
  - Assign issues on the Kanban board, tracking the progress of each task and redistributing work if needed
  - Act as a liaison between the group and instructional team
- **Notetaker** – Gurleen Rahi
  - Record meeting minutes and attendance during team meetings
  - Assist in updating the Kanban board based on meeting outcomes
- **Communication Liaison** – Moamen Ahmed
  - Organize team meetings and send reminders to all team members
  - Schedule meetings with external stakeholders
- **Administrator** – Inreet Kaur
  - Submit deliverables on the Avenue and ensure they meet the requirements
  - Maintain project documentation and files

The team will remain flexible for the following roles, with responsibilities subject to change based on the needs of each deliverable and the final project:

- **Meeting Chair** – This role will rotate between team members. The chair will organize and lead meetings, ensuring that the agenda is followed, and discussions stay focused.
- **Reviewer** – The team plans to distribute each milestone evenly among the team members. At least two other team members will review each team member's work. The reviewer will update issues on GitHub with appropriate comments.
- **Subject researcher** – During a deliverable, team members may be assigned to gather information about the industry, their processes, and information about the technology used in development. This role will shift based on individual expertise.

By adopting a flexible approach to the following roles, we want to ensure that everyone has the opportunity to lead. Regular check-ins during the team meetings will allow the team to reassess the roles and redistribute tasks as needed.

7 Workflow Plan

We will utilize Git as our version control throughout this project. Here is a detailed breakdown of how we collaborate using different features on GitHub.

- **Branches** – We will create different branches related to each task or issue.
  - Main branch – The main branch will have a stable production-ready code.
  - Feature branches – All features will be assigned their own branch. Once tested it will be merged into the main branch.
  - Documentation branches – Each team member will create their branch when working on a particular documentation task.
- **Pull requests** – Once a feature is ready for production or some documentation is ready, team members will create a pull request to merge them into the main branch. Each commit should include a brief description of the changes and related issue numbers of the Kanban board.
- **Issue tracking** – The team will utilize the Kanban board under projects for issue tracking. The team will use the provided templates for the appropriate issues and create its own where needed.

The team will be using the following projects:

- **Metrics** – This project will be used to record metrics such as attended lectures and team meetings. It has two types of issues, lecture meetings and team meetings and uses the appropriate templates provided. Each team member adds their own attendance and updates team meeting notes as per their assigned roles.

- **Documentation** – This project is used to create issues for each assigned task related to project documentation. Each issue will be reviewed by at least two other team members. Each issue uses a customized documentation template and is classified based on different milestones of project documentation.

- **Development workflow** – This project will be used to record issues for different features, feature enhancements, bug fixes, and other needs of the project. The issues will be divided into the above listed categories and will use customized templates for each.

CI/CD will be critical for project development. We will be using **GitHub Actions** which is a CI/CD automation tool provided by GitHub. It will act as an automated DevOps tool to streamline our development pipeline.

GitHub Actions will allow us to achieve Continuous Integration and Continuous Delivery through the following:

- **Continuous Integration** – On the continuous integration time GitHub Actions provides automated build and testing workflows directly within the GitHub repository. Upon each code commit or pull request, it will trigger a linting service that will ensure that the code maintains code standards that are outlined in the section below. This helps to identify any possible errors rightaway and resolves any integration issues leading to faster development iterations and also improves the code quality. Additionally, continuous integration has automated test cases that run automatically to ensure that the code passes all the checks.

- **Continuous Deployment** – Continuous Deployment is not required in this project due to absence of multiple environment which is needed for continuous delivery.

By implementing a CI/CD tool (GitHub Actions) we can ensure that code isn't riddled with errors and automate a lot of tasks which will increase productivity. Having GitHub Actions also will reduce the risk of human error in the project and will automate many areas of DevOps.

8 Project Decomposition and Scheduling

The team will utilize Kanban Board Projects for issue tracking with four different stages of an issue: 'To Do', 'In Progress', 'Review', and 'Done'. The team will create different issues for features, bugs, documentation, team meetings, lecture meetings and other things as needed using appropriate templates. The team will create templates for features, bugs and documentation. Issues under different projects will be classified into different milestones with appropriate deadlines. This will help us to track our progress for major milestones in the project. Furthermore, we will divide the Kanban board issues into sprints.

Therefore, we can reduce the risk of the overall project. Sprints will ensure that the risk of the project is lowered as we will be working in an agile fashion.

Project Links:

- Metrics
- Documentation
- Development workflow

Here is a general process that our team will follow to work on each issue. The team leader will create different issues under the 'To do' section and assign them to different team members as discussed during team meetings. Each team member creates their branch and will pull the latest changes from the main branch. Once the team member has made their changes they can be pushed to GitHub and the related issue can be moved under the 'Review' tab. The reviewers will provide appropriate comments and suggested changes and move the issue to the 'In progress' section if needed. At least two other team members will review each issue. Once the final changes are approved, a PR request will be raised to merge the changes into the main branch and the issue will now be moved to the 'Done' section.

Each task within the three projects will be scheduled according to the deadlines outlined in the course outline. The documentation project will have a separate milestone for each deliverable mentioned in the course outline along with the appropriate deadlines. The metrics project will track all the team meetings, lecture attendance, and other metrics required throughout the course. We will create an issue for each lecture and each team meeting with the required details in the template. The development project will be classified into different milestones throughout this course. Some stages include initial feature planning, Phase 1 which includes developing core functionalities, including voice recording and chart filling, and Phase 2 which includes developing additional features like the analytical dashboard, triage integration etc. Each phase will have separate issues for feature development, bug fixes, enhancement, and final reviews.

9 Proof of Concept Demonstration Plan

The main risk of the project is related to the reduction of documentation overhead. This means the biggest risk is really if this project can make a significant reduction to overhead time and truly reduce patient wait times. Again, through elicitation and our domain experts, we will create an effective set of requirements to reduce the risk of the project and build something that is needed.

Through our development plan, the coding standards, and the agile approach the development risk is made to be lower than a herculean approach. Therefore due to our development approach and using an agile workflow the project's risk will be reduced as features will be created over time.

Looking at implementation details there are a few risks that come up, if these risks can be addressed or have a clearer roadmap that will make us more confident in the project.

- **Speech Input** – A hospital or a clinic can be a loud place, in the event audio input is taken we need to ensure that it is clean and clear. This would mean essentially blocking outside noise.
- **Data Privacy** – This application will hold a lot of patient data so creating a store that is secure and making sure standard data security practice is applied is a must.
- **User Acceptance** – This will require further elicitation with our supervisor. This would help us to gather data on what critical needs of healthcare professionals such that critical features are present.

10 Expected Technology

In terms of expected technologies, we need to expect will again change will implementation details and architectures. Below is a starting point for a microservices-based architecture.

- Programming languages/Frameworks and Data management:
  - ReactJS – This will be used for the front end of the application.
  - Python (Flask) – This will be used to write backend services.
  - Firebase – All data will be managed through a Firebase realtime database as it provides the database along with authentication functionality.
- Libraries Key frameworks:
  - LangGraph – LangGraph will be critical in building multi-agent applications with LLMs by managing interactions between different agents. By representing workflows as directed graphs, LangGraph allows for precise control over the flow of information and execution, enabling complex reasoning and problem-solving. Agents, acting as nodes in the graph, process information and perform tasks, guided by dynamically generated prompts that can incorporate knowledge retrieved from vector stores.
  - OpenAI – OpenAI will be critical in interacting with the nodes in LangGraph and fine tuning them to provide faithful output.
  - Redux – Redux will be critical in managing frontend states.
  - OAuth – OAuth will be critical in microservices communication to validate the services. This will especially come in handy in a specific middleware layer is implemented.

  - Jest – Jest will be critical to use in unit testing for frontend development to validate the services.
  - Unittest – Unittest framework will be critical in writing and running the unit tests for validating backend services.

- Specific linter tool – Linters are useful for catching errors early in the development process. This also ensures the code that is written adheres to the standard which will mitigate code smells and enforce a specific style. The linter we will use is Super-linter, which is packaged into a Docker container which is called by GitHub Actions. This also makes sure that the required coding standard is upheld.
- Specific unit testing framework – For frontend testing, we can use Jest to interact with the front and leverage frameworks like Unittest for backend unit testing.
- Investigation of code coverage measuring tools – language-specific code coverage tools will be used such as Coverage.py for Python to assess coverage metrics and which cases are tested for.
- Continuous integration – As stated above GitHub Actions will be used to automate a lot of CI/CD tasks. Please refer to the CI/CD section above.
- Tools we will be using include Git, GitHub, and GitHub projects for development. This will mean using version control as be the workflow plan above, and planning features and sprints based on GitHub projects.

11 Coding Standard

Our team has decided to move forward with Pylint coding standard for Python. Pylint analyzes the code without actually running it. It checks for errors, enforces a coding standard and can also make suggestions about how the code can be refactored [1]. By using this coding style, we will ensure that our code is clean and understandable and consistent throughout the project. For TypeScript- For TypeScript, we are using ESLint and standard js coding standards. It analyzes the code to find any potential errors and enforces coding standards along with indentifying problematic patterns [1].

These coding standards can be added in CI/CD by integrating their components in the code. By making sure that the code's consistency and quality is not compromised whenever it is pushed or merged, it will automate the process by following the guidelines. CI/CD tool such as GitHub Actions will help ensure that the code is readable and optimized.

12 References

- [1] Coding Standards for Python and TypeScript

Appendix — Reflection

1. **Why is it important to create a development plan prior to starting the project?**

   A development plan helps to lay out the goals and objectives of the project. It also helps to identify any potential risks that will give us an opportunity to pre-plan the strategies to avoid them. Furthermore, it is a great tool to track the project's progress. Not only can we identify the risks, but we can also move towards creating a roadmap such that the project's overall risk is reduced. Through defining a CI/CD strategy we are also ensuring that the quality of code we deliver is high.

2. **In your opinion, what are the advantages and disadvantages of using CI/CD?**

   Advantages:

   - Tests will be automated such that the code during integration will always be tested.
   - Automated deployment will ensure that all code deployed follows the same code smell checks, and developers will get quick feedback if their code is not up to standard.
   - Due to a lot of DevOps tasks being automated that time can be transferred into development and improve productivity.

   Disadvantages

   - Setting up a CI/CD tool takes a lot of time and can pose a huge initial cost overhead.
   - Requires tests to cover a variety of cases and be robust and that will act as a report card for each feature.

3. **What disagreements did your group have in this deliverable, if any, and how did you resolve them?**

   In terms of disagreements in specific, we didn't have any but the process of creating this deliverable helped us align our visions for the project, user pain points, and challenges. This made the development plan very clear and emphasized how significant the problem is to society. It highlighted how our approach will have a significant positive impact on the healthcare department and improve the actual world.

Appendix — Team Charter

External Goals

- Our primary goal is to develop a solution that addresses a genuine problem in the healthcare industry, ultimately creating a positive impact on society.
- We strive to deepen our understanding of different technologies, such as speech recognition, LLM, NLP, and cloud-based solutions, to enhance our technical skillset.
- We aim to enhance our soft skills such as teamwork, communication, and project management skills by collaborating effectively and managing project timelines efficiently.
- We aim to develop a solution that exceeds expectations and get recognition at the Capstone EXPO.

Attendance

Expectations

Each team member is required to attend weekly team meetings as well as meetings with the supervisor, TAs and other stakeholders. If for any reason a team member cannot attend a specific meeting, they should notify the team well in advance. This would be at least 24 hours prior to the team meeting if they cannot attend and 1 hour prior to the team meeting if they are running late.

Acceptable Excuse

Acceptable excuses

- Family emergency
- Health issues
- Academic Conflict (lectures or exams)
- Any technical difficulties

Unacceptable excuses

- Workload from other courses
- Lack of preparation
- Any other non-urgent issues

In Case of Emergency

If a team member cannot attend a team meeting and finish their part before the promised deadline due to any acceptable excuses, they should notify the team in advance through the Microsoft team Channel or Microsoft Teams Chat. In case, a team member is unable to complete their assigned work, they should provide other team members with an update on their current progress and upload any necessary material on the Microsoft Teams.

Accountability and Teamwork

Quality

Each team member should complete all assigned tasks before the team meeting. Members should also review the agenda for the meeting posted on the GitHub Project Issue and prepare accordingly. For deliverables, each team member should review and read all interdependent parts and should make sure that their work is coherent and does not conflict with the overall continuity of the document. Each team member should check grammar and formatting issues in their work before making a commit or merging into the main branch. Each team member should be willing to incorporate feedback provided by other team members or should be open to discussion if they believe otherwise.

Attitude

All team members should treat each other with respect and should maintain a cooperative attitude. If a team member fails to meet expectations, the issue will be addressed through a team meeting. The team members should provide constructive feedback when adding any comments for Issues on GitHub projects. Any conflict within the team should be resolved professionally in a team meeting. The issue will be discussed through open discussion, allowing each team member involved to express their viewpoint. If unsuccessful, the team will contact the TA or Professor for resolution. We are committed to maintain an inclusive environment where everyone feels comfortable without fear of criticism.

Stay on Track

To keep the team on track, the team will organize weekly team meetings to report and review the project's progress. The team will adopt the agile methodology and divide the project into different milestones according to the deadlines. Each team member will be assigned a specific task with clear deadlines using GitHub projects. Each team member will adhere to the guidelines for the quality of team members' preparation for team meetings and the quality of the deliverables. If a team member fails to meet the expectations, the issue will be discussed through a team meeting and will be escalated to the TA or professor if needed. // The team will use the GitHub project – capstone metrics to track attendance for lecture attendance and team meetings. The team will be using

the templates provided to create issues for each meeting. // Team members who consistently meet or exceed expectations will be acknowledged during meetings. Members who miss targets without acceptable excuses will be responsible for bringing snacks or coffee to the next meeting. If a team member is consistently not meeting the expectations the issue will be escalated to the professor.

Team Building

For fun time, we will go to Software events like software and computing industry night which is a great way to hang out and explore. We will gather as a team to play Table Tennis and Basketball on campus which is a great way to know each other outside of academics. This will get us to know more about each other's strengths and weaknesses, which will aid us in our team project.

Decision Making

All decisions in the team will be made through consensus. Each team member would be given the opportunity to present their viewpoints and concerns. If consensus cannot be reached the team will contact the stakeholders or instructional staff for further clarification to ultimately reach a decision. All disagreements in the team will be addressed through team meetings. The team will engage in respectful discussions allowing all team members to present their points. If the issue is still not resolved, the issue will be escalated to the TA or professor.