

# Development Plan

## Software Engineering

Team 5, GradSight  
Zahin Hossain  
Hammad Pathan  
Henushan Balachandran  
Wajdan Faheem  
Willie Pai

[Put your introductory blurb here. Often the blurb is a brief roadmap of what is contained in the report. —SS]

[Additional information on the development plan can be found in the lecture slides. —SS]

This document outlines the development plan for the GradSight project, created by Team 5 as part of the Software Engineering program. The goal of this project is to deliver a functional application that modernizes the way graduation composites are displayed and managed within the Faculty of Engineering. The plan includes key details such as meeting schedules, communication strategies, team roles, workflow, project risks, and the expected technology stack. Additionally, we cover intellectual property considerations, coding standards, and our proof-of-concept demonstration plan to mitigate project risks.

## 1 Confidential Information?

[State whether your project has confidential information from industry, or not. If there is confidential information, point to the agreement you have in place. —SS]

[For most teams this section will just state that there is no confidential information to protect. —SS]

There is no confidential information to protect for our project.

## 2 IP to Protect

[State whether there is IP to protect. If there is, point to the agreement. All students who are working on a project that requires an IP agreement are also required to sign the “Intellectual Property Guide Acknowledgement.” —SS]

We will be using digital photos provided to use by an external company LifeTouch, which will hold the intellectual property rights to these images. However, at this stage, we do not have a formal IP agreement in place with LifeTouch regarding the use of the photos. We understand the need and are currently working to secure one before proceeding.

## 3 Copyright License

[What copyright license is your team adopting. Point to the license in your repo. —SS]

Our team is adopting an Apache 2.0 license. We believe out of the options presented to us this suited our project the best.

<https://github.com/WajdanF/team5?tab=Apache-2.0-1-ov-file#>

## 4 Team Meeting Plan

[How often will you meet? where? —SS]

[If the meeting is a physical location (not virtual), out of an abundance of caution for safety reasons you shouldn't put the location online —SS]

[How often will you meet with your industry advisor? when? where? —SS]

[Will meetings be virtual? At least some meetings should likely be in-person. —SS]

[How will the meetings be structured? There should be a chair for all meetings. There should be an agenda for all meetings. —SS]

**Team Meetings:** We'll meet twice a week to stay on top of things and ensure we're all aligned with a mix of in-person and virtual meetings, depending on what works best for everyone.

- **Virtual Meetings:** We'll use Microsoft Teams to make it easy to join from anywhere.
- **In-Person Meetings:** We'll meet on campus at Thode library.

**Meetings with our Industry Advisor (Meggie MacDougall):** We'll meet with Meggie to get feedback and ensure we're on the right track. The majority of these meetings will be held online over Teams.

### 2. How will the meetings be structured?

**Chair:** We'll rotate who leads the meeting so everyone gets a chance to run things. The person leading will ensure we stick to the subject at hand.

**Agenda:** Whoever is in charge of the meeting will send out an agenda the day before, covering:

- Progress updates on tasks
- Upcoming tasks and objectives
- Challenges we're facing
- Feedback or advice from Meggie (if applicable)
- Deciding on next steps and assigning tasks

### 3. In-person vs Virtual Meetings

Most meetings will likely be virtual for convenience, but we'll get together in person when needed, such as for big milestones or hands-on work with hardware.

## 5 Team Communication Plan

[Issues on GitHub should be part of your communication plan. —SS]

## 1. Primary Communication Channels

**Microsoft Teams:** Our main platform for day-to-day communication as it is convenient for quick discussions, file sharing, and scheduling meetings.

**Email:** Used for formal communication with the industry advisor (Meggie MacDougall) and other external stakeholders.

## 2. Task Management and Issue Tracking

**GitHub:** We'll manage tasks and track issues using GitHub. Each feature, bug, or task will be logged as an issue, with clear labels to indicate priority and type (e.g., enhancement, bug, question).

**Pull Requests (PRs):** All code changes will go through a pull request process, where at least one team member will review the changes before merging. For large PRs, we'll discuss as a team and ensure everyone is on board before merging it.

## 3. Meeting Notes and Agendas

**Google Docs:** Agendas for each meeting will be prepared in advance and shared through Google Docs. Meeting notes will be documented in the same document and shared with the team post-meeting to keep everyone informed.

## 4. Industry Advisor Communication

**Updates:** We will provide updates to Meggie, summarizing progress and any critical issues via Microsoft Teams or email.

## 5. Project Documentation

**GitHub Wiki:** We'll maintain a project wiki on GitHub that will include all project-related documentation such as technical specifications, design documents, and user guides.

## 6. Escalation Process

**Team Leads:** If an issue needs immediate attention, we'll escalate it to the team lead for that task area (rotating chair). If necessary, it will be brought up during meetings for group resolution.

# 6 Team Member Roles

[You should identify the types of roles you anticipate, like notetaker, leader, meeting chair, reviewer. Assigning specific people to those roles is not necessary at this stage. In a student team the role of the individuals will likely change throughout the year. —SS]

### **Willie: Communications Lead**

Manages communication with external stakeholders, ensuring progress updates and feedback are exchanged regularly.

### **Wajdan: Leader**

Oversees the overall progress of the project and ensures deadlines are met, while coordinating team efforts.

### **Zahin: Meeting Chair**

Responsible for creating meeting agendas, leading discussions, and ensuring meetings stay focused and productive.

### **Hammad: Reviewer**

Reviews code and project changes, ensuring they meet quality standards before being merged into the main project.

### **Henushan: Notetaker**

Takes detailed notes during meetings, capturing key decisions, action items, and follow-up tasks for the team.

## **7 Workflow Plan**

- How will you be using git, including branches, pull request, etc.?
- How will you be managing issues, including template issues, issue classification, etc.?
- Use of CI/CD

### **Git Usage**

We will be using Git to implement version control for the project. The workflow will involve creating **feature branches** for specific key components of the project. This approach will significantly reduce merge conflicts when multiple team members are working on different parts of the service.

The **master branch** will be used for production-ready, working code, while the **develop branch** will contain all features that are ready to be merged into the master branch. This setup follows a typical agile development process.

Pull requests (PRs) will occur between branches to ensure smooth code management and to facilitate reviews that guarantee quality code.

## Issue Management

We will use a **Feature Request Template** when creating pull requests for new features added to the project, which will include the following:

- **Title:** Descriptive title of the feature.
- **Description:** Explanation of the functionality and why it's needed.
- **Priority:** High/Medium/Low.

Labels can be added throughout the templates to identify and tag specific filters such as:

- **Type of entity:** bug, report, documentation.
- **Priority:** High/Medium/Low.
- **Status:** In Progress/Done.

A Kanban board with tickets will be used to track features and tasks. This can be done through GitHub's project board or potentially external tools like Jira for task management.

## CI/CD Pipeline

We will add a **staging environment** in the **develop branch** to ensure that the code pushed to the develop branch is thoroughly tested.

Automated deployment to the **master branch** will be configured from the develop branch. Once all tests pass, deployment to the master branch will be rolled out.

Docker will be used to create separate environments (different from develop and production) to run tests in a controlled setup.

## 8 Project Decomposition and Scheduling

- How will you be using GitHub projects?
- Include a link to your GitHub project:

[How will the project be scheduled? This is the big picture schedule, not details. You will need to reproduce information that is in the course outline for deadlines. —SS]

### How will you be using GitHub projects?

- We can use GitHub projects to manage planning and tracking work.
- Integrate issues and pull requests into the spreadsheet.

- Can create filters, views, and groupings to visualize work and allocated tasks.
- More information about metadata on the given issues and tasks can be decided later (complexity level, estimated task points, etc).
- **Include a link to your GitHub project:** <https://github.com/WajdanF/team5>

## Project Scheduling

- The project will be scheduled based on a sequential order.
- Large sections of the project and deliverables will be done in this order.
- The work within each section can be broken down to allocate to all members of the group.

## 9 Proof of Concept Demonstration Plan

What is the main risk, or risks, for the success of your project? What will you demonstrate during your proof of concept demonstration to convince yourself that you will be able to overcome this risk?

The main risks for the success of the project are:

- **Hardware Requirements:** The app is designed to run on a touchscreen display, which could be a hardware risk. To mitigate this risk, we can do testing on any touchscreen device, such as a smartphone or tablet, to validate the functionality without the need to test it on the final hardware.
  - The consequence of this risk being unmitigated would be the need to arrange earlier access to the final touchscreen display. In this case, we would need to coordinate with the Faculty of Engineering to secure the hardware for testing.
- **Converting Composites into Digital Form:** We are required to convert all past physical, printed composites into digital form and parse the name and image of each student. This poses a risk due to potential difficulties with image quality, resolution, and variation in composite layouts.
  - To overcome this risk, we could fetch the names of all students from the respective graduating class directly from the faculty. This would allow us to create an accurate, searchable database even if parsing is unreliable.
  - This manual approach, while time-consuming, will ensure that the system is fully functional. If it becomes too time-consuming, we will attempt to collaborate further with LifeTouch to get better-quality composites.

- **Demonstration of POC:** A low-fidelity prototype will be made to showcase the overall structure of the application to showcase how the digital composites will be presented to the user. Additionally, a demonstration will be completed of parsing all students' names from a physical composite, and having it rendered as a list on an application.

## 10 Expected Technology

[What programming language or languages do you expect to use? What external libraries? What frameworks? What technologies. Are there major components of the implementation that you expect you will implement, despite the existence of libraries that provide the required functionality. For projects with machine learning, will you use pre-trained models, or be training your own model? —SS]

[The implementation decisions can, and likely will, change over the course of the project. The initial documentation should be written in an abstract way; it should be agnostic of the implementation choices, unless the implementation choices are project constraints. However, recording our initial thoughts on implementation helps understand the challenge level and feasibility of a project. It may also help with early identification of areas where project members will need to augment their training. —SS]

Topics to discuss include the following:

- Specific programming language
- Specific libraries
- Pre-trained models
- Specific linter tool (if appropriate)
- Specific unit testing framework
- Investigation of code coverage measuring tools
- Specific plans for Continuous Integration (CI), or an explanation that CI is not being done
- Specific performance measuring tools (like Valgrind), if appropriate
- Tools you will likely be using?

[git, GitHub and GitHub projects should be part of your technology. —SS]

- **React.js** → Front-end library
- **TypeScript** → Language we are using for the front-end
- **Tailwind CSS** → Layout/formatting/design library



- **Optical Character Recognition (OCR)** with a pre-trained model → For previous composites
- **Prettier + ESLint** → Code formatting
- **Jest** → Testing library
- **GitHub Actions** → CI/CD
- **GitHub** → Version control
- **GitHub Projects Kanban Board**
- **Microprocessor (e.g., Raspberry Pi)**
- **Chrome**
- **Visual Studio Code**

## 11 Coding Standard

[\[What coding standard will you adopt? —SS\]](#)

The coding standards we will follow are listed below:

- **DRY (Don't Repeat Yourself):** Avoid code duplication by abstracting repeated logic, which allows it to be reused across the project.
- **Keep it Simple:** Write code that's as simple as possible without over-engineering or thinking too far ahead, as pre-optimization can lead development down a rabbit-hole.
- **Consistent Naming Conventions:** Agree on consistent naming conventions for variables, functions, and classes. In this project, we will be using **camelCase** for variables and functions, and **PascalCase** for classes and file names.
- **Proper Commenting and Documentation:** Encourage adding comments where necessary to explain complex logic that isn't intuitive.
- **Version Control Best Practices:** It is best to avoid large commits in case rollbacks need to occur and we need to revert to an older version of the code.

## Appendix — Reflection

[Not required for CAS 741 —SS]

The purpose of reflection questions is to give you a chance to assess your own learning and that of your group as a whole, and to find ways to improve in the future. Reflection is an important part of the learning process. Reflection is also an essential component of a successful software development process.

Reflections are most interesting and useful when they're honest, even if the stories they tell are imperfect. You will be marked based on your depth of thought and analysis, and not based on the content of the reflections themselves. Thus, for full marks we encourage you to answer openly and honestly and to avoid simply writing "what you think the evaluator wants to hear."

Please answer the following questions. Some questions can be answered on the team level, but where appropriate, each team member should write their own response:

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

It is important to create a development plan prior to starting the project because it helps provide a clear outline of the project's goals, tasks, timelines, and deliverables. This allows everyone to understand what we need to do and what steps we are going to take.

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

One of the key advantages of using CI/CD is the fast development and deployment it offers by ensuring that changes are incorporated into the project continuously, as well as catching errors early through automated test cases. Moreover, it further accelerates this process by enabling automated deployments, allowing features and bug fixes to reach users more quickly and efficiently.

When it comes to the disadvantages, a smaller team with limited scope may find that the overhead of setting up and maintaining a CI/CD pipeline outweighs the benefits. In this case, simpler workflows may be more efficient, as the complexity of a full CI/CD setup might be unnecessary.

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

We did not have any disagreements in this deliverable, as we were all clear about every section.

## Appendix — Team Charter

[borrows from University of Portland Team Charter —SS]

### External Goals

[What are your team's external goals for this project? These are not the goals related to the functionality or quality of the project. These are the goals on what the team wishes to achieve with the project. Potential goals are to win a prize at the Capstone EXPO, or to have something to talk about in interviews, or to get an A+, etc. —SS]

- **Provide a solution to the Engineering Faculty:** Our main goal is to deliver a functional digital composite web app that will help the Faculty of Engineering modernize how they display and manage graduation composites. We want this solution to potentially be used long term by the faculty.
- **Learning and Professional Development:** This project will give our team valuable experience to discuss in future job interviews and showcase our skills in web development, machine learning, and secure data management.
- **High Academic Mark:** Our group would like to achieve an A+ in this course with this project to maintain our high CGPA.

### Attendance

#### Expectations

[What are your team's expectations regarding meeting attendance (being on time, leaving early, missing meetings, etc.)? —SS]

Regarding meeting attendance, our team expects everyone to show up on time to all scheduled meetings because punctuality ensures that we make the most of our time. If anyone knows they're going to be late or need to leave early, we ask that they give advance notice so we can plan around it. In the case of missing meetings, the person should let the team know ahead of time unless it is an urgent issue because decisions are made every meeting and missing one can slow down development.

#### Acceptable Excuse

[What constitutes an acceptable excuse for missing a meeting or a deadline? What types of excuses will not be considered acceptable? —SS]

An acceptable excuse for missing a meeting or a deadline is when there is an emergency, but other than that team members should be showing up. In terms of deadline, the only reason that a milestone shouldn't be handed in is if there is missing content.

## In Case of Emergency

[What process will team members follow if they have an emergency and cannot attend a team meeting or complete their individual work promised for a team deliverable? —SS]

In case of a personal emergency, the person should let the team know and take leave right away. The work that was expected from the excused member will be divided up among the remainder of the people and finished accordingly. Also, if the milestone is not due for a little longer and their emergency situation comes to a halt, they are expected to go over the work that they were initially assigned and add to the section.

## Accountability and Teamwork

### Quality

[What are your team's expectations regarding the quality of team members' preparation for team meetings and the quality of the deliverables that members bring to the team? —SS]

- **Meeting Preparation:** Everyone is expected to come prepared for meetings by reviewing the agenda and completing their assigned tasks. This ensures we can make the most of our time and keep things moving smoothly.
- **Quality of Deliverables:** Deliverables, whether code or documents, should meet high standards and accomplish the goal of the task at hand. This means testing code, clear documentation, and sticking to the project's goals. To ensure high-quality work, the team will review each other's work and provide feedback as needed.
- **Teamwork:** We expect open communication and support from everyone. If someone is facing challenges or delays, they should let the team know early, so we can adapt and help out.

### Attitude

[What are your team's expectations regarding team members' ideas, interactions with the team, cooperation, attitudes, and anything else regarding team member contributions? Do you want to introduce a code of conduct? Do you want a conflict resolution plan? Can adopt existing codes of conduct. —SS]

- All team members' ideas will be heard and put to a vote when making a decision.
- Interactions between the team are expected to be respectful, polite, honest, and with only good intentions.
- All team members are expected to come to meetings with a positive attitude.

- All team members will contribute an equal amount of work to the best of their ability.
- Should any team members have any issues, they are expected to communicate with the entire team first.
- If issues arise, all team members are expected to support and comply with the majority vote, which should be the best possible solution for everyone.
- If a team member does not agree with the majority vote, they must take the next step in contacting a TA or the Professor to escalate the matter.

### Stay on Track

[What methods will be used to keep the team on track? How will your team ensure that members contribute as expected to the team and that the team performs as expected? How will your team reward members who do well and manage members whose performance is below expectations? What are the consequences for someone not contributing their fair share? —SS]

[You may wish to use the project management metrics collected for the TA and instructor for this. —SS]

[You can set target metrics for attendance, commits, etc. What are the consequences if someone doesn't hit their targets? Do they need to bring the coffee to the next team meeting? Does the team need to make an appointment with their TA, or the instructor? Are there incentives for reaching targets early? —SS]

## Methods to Keep the Team on Track

- Regular meetings, at least once a week. In these meetings, we will all give a general update on what we are working on, our plans, and if any support is needed.
- We will use Jira to keep track of tickets to help ensure everyone knows what is being worked on at all times so we can manage deadlines.
- Our team will set up personal deadlines for deliverables, aside from the official due dates, so that we can take time to review our final submissions.

## Team Contribution and Performance

- Attendance at meetings will be tracked to see if anyone is falling behind.
- Tasks will have to be completed by their deadlines.
- If the group feels that work completed is not up to our standards, we will conduct a code review with the author and take note of the review.

## Rewards

- For team members who are consistently completing tasks early and supporting the team, they will have the option to take the lead on tasks they find interesting.
- In addition, they will receive virtual high fives from the rest of the team members.

## Consequences for Not Contributing

- The team member will be required to sit with the team to explain why they are missing deadlines or unable to commit to meetings.
- If this cannot be resolved internally, the team will reach out to a TA or the Professor.

## Team Building

[How will you build team cohesion (fun time, group rituals, etc.)? —SS]

## Team Socials

Set aside time for non-work-related activities, like virtual or in-person game nights, trivia quizzes, or even casual meetups at a café. This will help team members connect beyond work.

## Online Team Games

We all have interest in online games such as Valorant and League of Legends. We can use this as a bonding strategy to work as a team, outside of the project!

## Weekly Check-ins

Start meetings with a short, personal "How's everyone doing?" check-in. It helps build empathy and understanding among team members.

## Celebrate Wins

At the end of each week, celebrate milestones, both small and big, and acknowledge each other's contributions. This could be done with fun titles like "MVP of the Week" or "Bug Slayer."

## Decision Making

[How will you make decisions in your group? Consensus? Vote? How will you handle disagreements? —SS]

We should aim for consensus as the primary decision-making process. In consensus, all team members should have a chance to voice their opinions, and the goal is to reach a solution that everyone can support, even if it's not their first choice. Inevitably, everyone will feel heard. We can start by presenting the issue at hand. Then discuss all perspectives and approaches. Express and convey thoughts through conversation. Come to a decision as a group.

For any disagreements within the group, before anything, everyone should have the chance to speak and everyone should simultaneously give their full attention to the speaker. This encourages open and respectful communication. If we are still unable to come to a consensus, we can bring in a mediator, third party, someone who is neutral to help mediate the issue and come to a resolution. Finally, when consensus isn't available, we as a team should make compromises from both positions. This may require more responsibilities and testing, but ensures that we are structured as a team.