M08 Updated Project Plan

Kyle Ingersoll, Vanya Bays, Dylan Howard

5-12-2024

SDEV265

# **Introduction**

* 1. The objectives of this project are to adapt Kyle Ingersoll’s CSCI101 Final Project program, written in C++ for command line usage, to Python using a graphical user interface powered by Tkinter. The project must be done by May 10th, 2024. It must be entirely written in Python with no C++ code, and it must be written using a modular approach; it must have straightforward navigation through the GUI, it must have four windows, two images, five labels, and five buttons, utilize input validation on every input variable and box, have validation testing and documentation of source code, and have a ReadMe file with installation instructions. We weren’t able to do input validation at all, due to it causing various widgets to disappear and not having the time to solve the bug.

# **Project Organization**

* 1. Team Members and Roles:
     1. Kyle Ingersoll: Chief Programmer, Chief Rubric Checker
        1. Ensures that the program both meets rubric requirements and is the primary person responsible for coding the program
     2. Vanya Bays: Editor and writer
        1. Ensures that the written assignments are written and edited for quality and accuracy.
     3. Dylan Matthew Howard: Assistant writer, etc
        1. Will help write on the writing assignments, and help with any other duties that may arise.

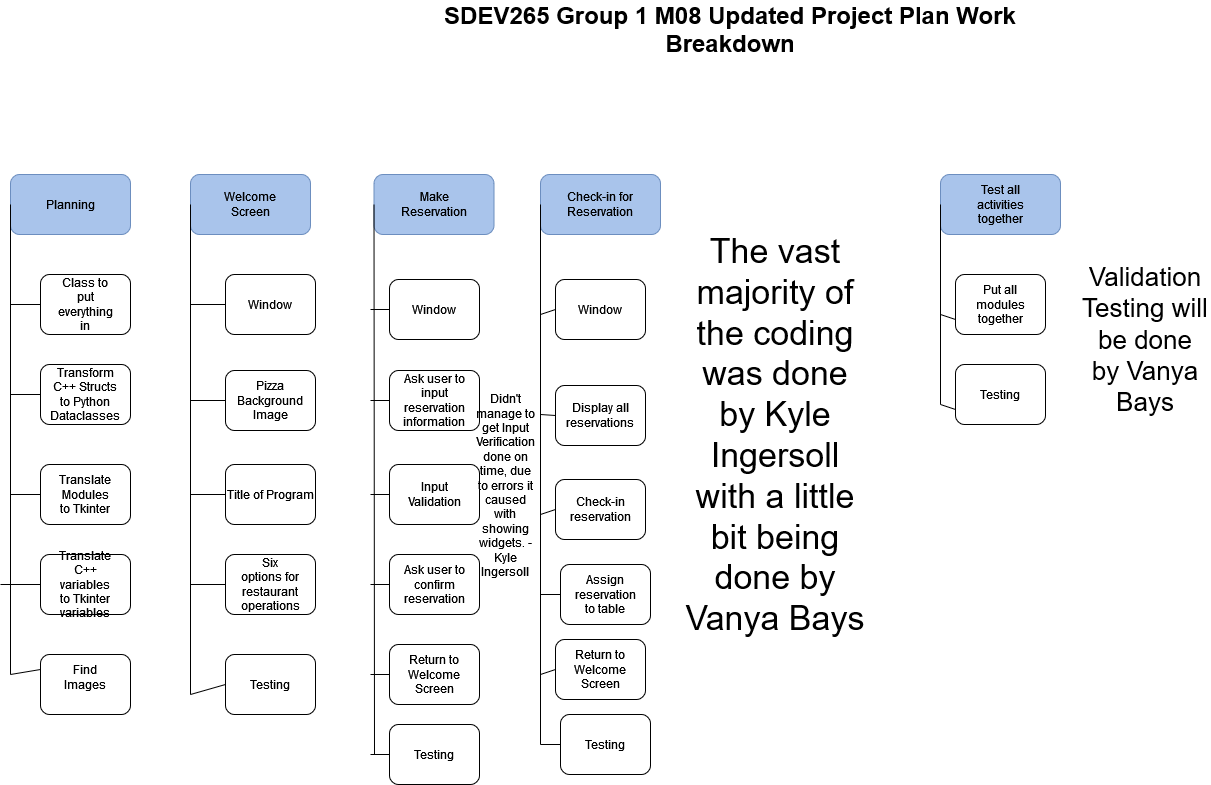
# **Risk Analysis**

* 1. **Risk Identification**
  2. **Cost Risk**
  3. During the project, some required platforms may require a subscription to be fully used. This is an issue as the team would prefer not to spend money if it could be avoided.
  4. Probability: Low
  5. Impact: Medium
  6. Risk Planning:
  7. To mitigate cost risks, research free software that offers what is required of the project. The best free software will be used in place of paid software.
  8. **Schedule Risk**
  9. During the project, there may be delays. Delays can occur from poor planning and poor communication.
  10. Probability: Low
  11. Impact: High
  12. Risk Planning:
  13. To mitigate schedule risks, the team will develop a project schedule that clearly outlines when something needs to be finished. When the project falls behind schedule, action will be taken to correct the issue.
  14. **Performance Risk**
  15. The project might not be adequately adapted to Python, and some parts of C++ might not translate well in Python. There may be bugs related to the differences between languages.
  16. Probability: Medium
  17. Impact: Low
  18. Risk Planning:
  19. To mitigate the performance risk, the team will implement input validation to ensure the correct type of input is performed.
  20. **Operational Risk**
  21. The project will need to be shared with the team to confirm whether the code works or not. There will be some issues with converting C++ code into Python.
  22. Probability: Medium
  23. Impact: Medium
  24. Risk Planning:
  25. To mitigate the risk, the project will be shared using platforms like GitHub, and documents will be shared using platforms like Google Drive.
  26. **Technology Risk**
  27. The project will be stored on GitHub, and the team will use Google Drive to share documents. There is a chance that the data stored on the cloud will be hacked or deleted.
  28. Probability: Low
  29. Impact: High
  30. Risk Planning:
  31. To mitigate the risk, backups of the project will be stored on a local system. This will ensure a usable version of the code is always available.
  32. **Communication Risk**
  33. Team members will need help to meet with the group.
  34. Probability: Medium
  35. Impact: low
  36. Risk Planning:
  37. If a team member cannot meet with the group, either reschedule the meeting or explain what happened during the last meeting to the absent party and ask for their opinion.
  38. **Scope Creep Risk**
  39. The project may end up having more changes than initially expected. When rewriting an application in C++ to Python, some changes must be made.
  40. Probability: Medium
  41. Impact: High
  42. Risk Planning:
  43. To mitigate Scope Creep Risk, the team will have a clear scope to ensure it is understood by everyone. If the scope is changed, it will be communicated to everyone as soon as the change is made.
  44. **Skills Resource Risk**
  45. The project is C++ code being rewritten in Python code. Part of the original code may not be able to be translated into Python very well. The project may require more advanced knowledge of Python and Tkinter.
  46. Probability: High
  47. Impact: High
  48. Risk Planning:
  49. To mitigate the skill resource risk, the team will search online to learn how to correctly rewrite C++ code into Python. If a concept in C++ does not apply to Python, then the closest substitute will be used.
  50. There was some Scheduling Risk, and Scope Creep Risk, since our original project plan was way too large in scope to do in the time allotted, so we had to cut it down to just doing the Make Reservation and Check-In Reservation functionalities to the program. I also had to simplify all the maximum number of people who could sit in the table numbers to 10 people for convenience, and I looped back to Make Reservation Screen instead of 3 other potential screens for each input box when the Cancel Option in Confirm Reservation Screen was pressed. Furthermore, I wasn’t able to create Input Verification at all due to it making widgets not show up, and not having the time to debug it.

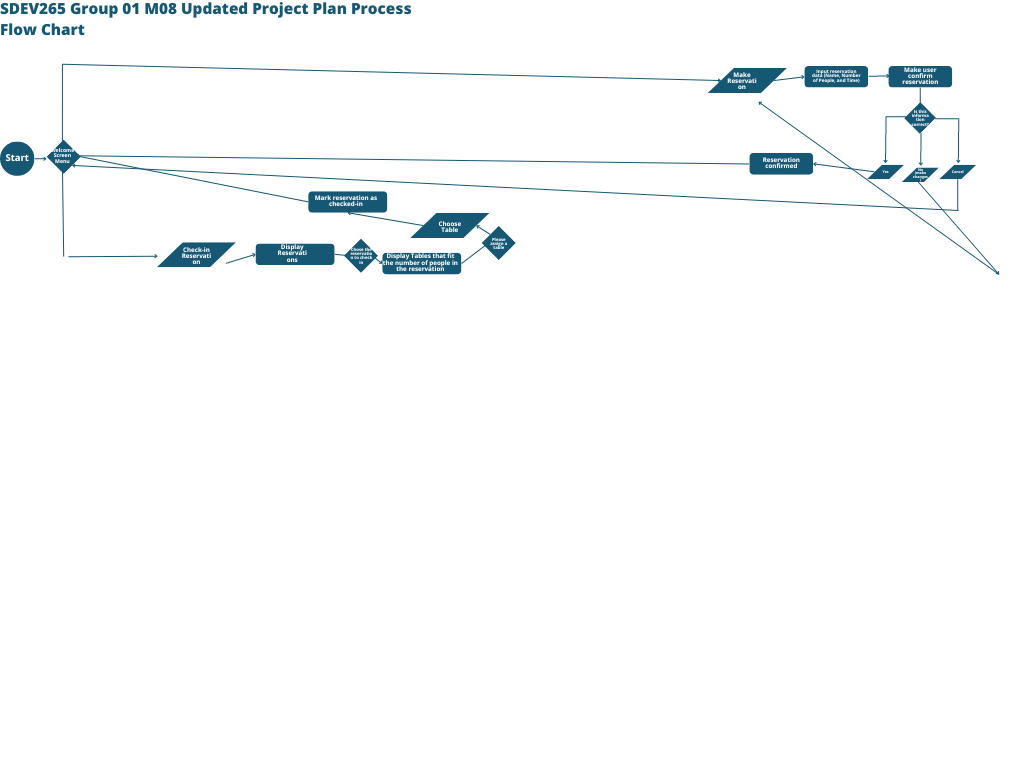
# **Hardware and Software Requirements**

* 1. Hardware requirements:
     1. 8 GB of RAM and a relatively recent processor (with at least the processing power of i3 11th generation) with at least 20 GB of storage (hard drive or SSD) free.
     2. PC
     3. Monitor
     4. Keyboard and Mouse
  2. Software requirements:
     1. Have at least Python 3.0 installed on your system; the more recent, the better
     2. Operating system can be any operating system that supports Python 3.0 (i.e. Windows, Mac, Linux).
     3. Install Tkinter, dataclasses, and PIL as a module using PIP

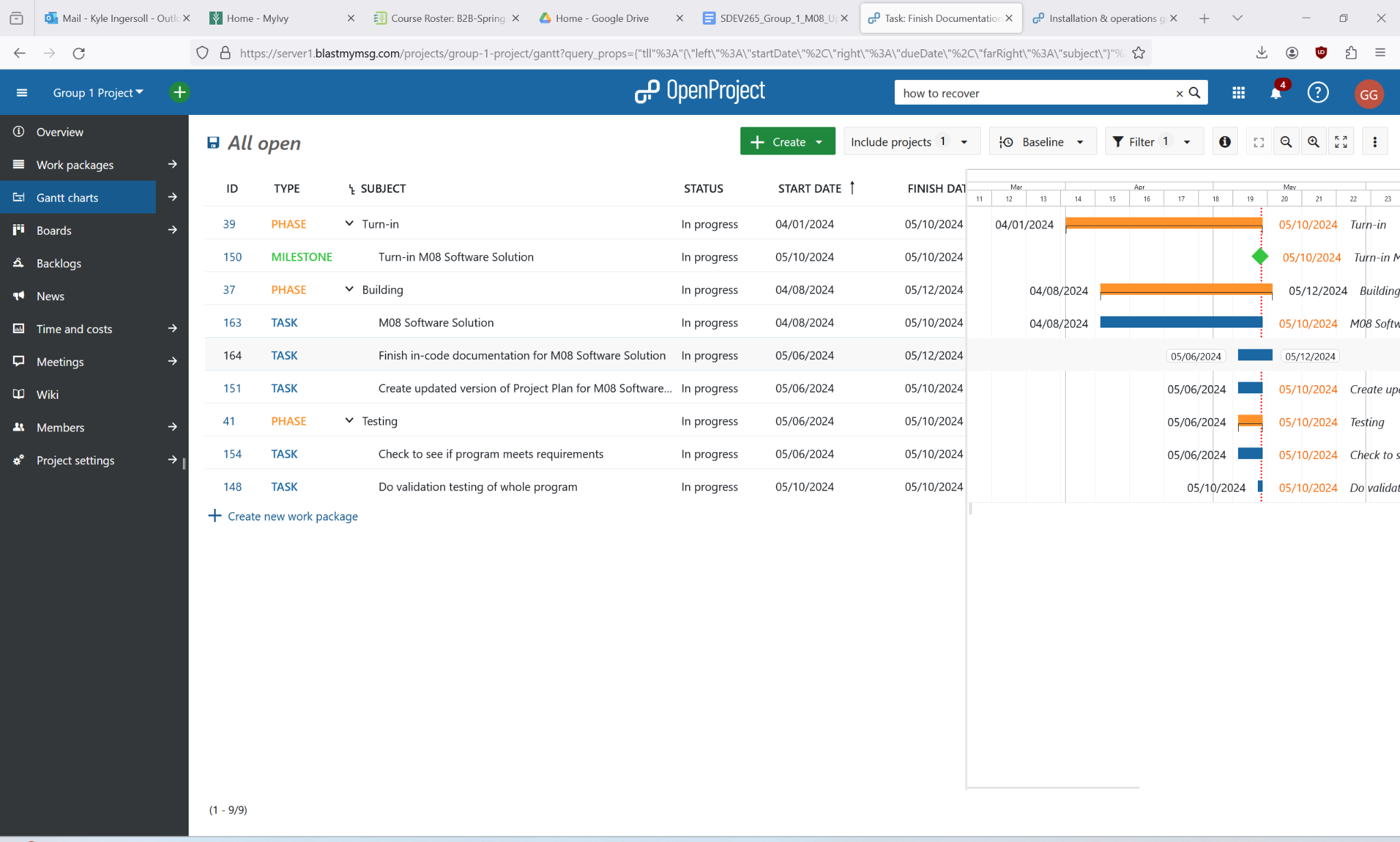
# **Work Breakdown**

* 1. 

# **Process Flow Diagrams**

* 1. 
  2. https://www.canva.com/design/DAGBHd7rc9k/cqUu2ReGRsGJ3-V83HEgIg/edit?utm\_content=DAGBHd7rc9k&utm\_campaign=designshare&utm\_medium=link2&utm\_source=sharebutton

# **Project Schedule**

* 1. 

# **Monitoring and reporting mechanisms**

* 1. The tools that were chosen to help with communication, collaboration, and scheduling are Discord, Zoom, Github, and OpenProject. Discord is used for out of class meetings and messaging, a constant communication line basically. Zoom is where we hold our weekly meetings in our breakout rooms, where we strategize about how we will tackle our assignments. Github is where the software project will be hosted, and it is where we will collaborate to create the project. OpenProject will be used for scheduling the tasks that we will do as well as to keep track of our progress on our projects and assignments. We didn’t really actually use OpenProject to keep track of our assignments that much, partially since we had Zoom, Canvas, and Discord to work together and keep track of our assignments, and also because we just didn’t bother to use it after creating the Project Schedule.