Table 1: Revision History

Date	Developer(s)	Change
2018-09-27	1.0	Added draft of proof of concept plan
2018-09-28	1.1	Added initial draft of team plan, technology, etc
2018-09-28	1.2	Added git workflow plan
2018-12-03	1.3	Updated team name
2018-12-05	1.4	Removed a lot of excessive git workflow components, as well as other changes

SE 3XA3: Development Plan
Ratava

Team 9, Makiam Group
Aidan McPheilm - mcpheima
Alexie McDonald - mcdona16
Illya Pilipenko - pilipeni

September 28th, 2018

The purpose of this Development Plan is to outline all information required to develop the project. This project details the specifics of the team meetings, team communication, and other team dynamics. In addition to that, it outlines the git workflow plan, the Proof Of Concept plan, and other aspects related to the implementation and technology required. Finally, it outlines the tentative project schedule and the project review (to be filled out later).

1 Team Meeting Plan

- **Team Meeting Day:** Friday
- **Team Meeting Frequency:** Weekly
- **Team Meeting Time:** 7:00pm-8:30pm (1.5 hours in duration)
- **Team Meeting Location:** Mills Library (a pre-booked room for which the location of it will be determined 2 days prior to the meeting)
- **Team Meeting Roles:**
  - **Meeting Minutes Scribe** - Alexie McDonald
  - **Meeting Agenda Consultant** - Aidan McPheilm

2 Team Communication Plan

The following methods of communication are paired with the scenarios where it would be appropriate to communicate such information

- *Group Chat*
  - Mid-priority questions

  - Declarations of intentions to work on a file
  - Other mid-priority communication
- *Email*
  - Transfer of information not pushed to the repository
  - Clarification of prefered merge conflict resolution
  - Other non-urgent communication
- *Text*
  - Late arrivals to lab
  - No reply to communicatons aformentioned
  - Late arrivals to meetings
- *Phone*
  - Urgent questions
  - Emergencies
  - Repeated absences from lab/meetings

3 Team Member Roles

- *Aidan McPhelim*
  - Developer
  - Testing expert
- *Alexie McDonald*
  - Developer
  - Documentation expert
- *Illya Pilipenko*
  - Developer
  - Git expert

4 Git Workflow Plan

4.1 Branch types

The Git repository will consist of the following branches:

- A **master** branch, which will contain the version of the code in production.
- One **personal** branch per team member, which will be the two letters of their initials. These are the branches in which most of the development will happen.

4.2 Pushing and pulling

- The develop branch will be synchronized with the master branch, but will only be merged into it when develop is ready for a release.
- **Personal** branches will be synchronized with either the **master** branch, or a different **personal** branch. They will only be merged into their corresponding branch when they don't contain non- functioning code, to the developer's knowledge.
- Merging ready code into a branch (henceforth called the **target** branch) will work as follows:
  1. Push to your current branch's remote repository to have a back-up of your current code in case merging goes wrong.
  2. Pull from the **target** branch, and make sure there are no merge conflicts. Commit any changes.
     - note that if there were any merge conflicts, in the time it took to fix them, more changes could have been pushed to the **target** branch
  3. Checkout the **target** branch, make sure it's up-to-date with its remote repository, and pull from your development branch.
  4. Commit and push to the **target** branch's remote repository.
     - it is good to notify all developers using the **target** branch (if the target is **master**, then that means all developers) that you are pushing

4.3 Tags and milestones

- Tags will be used for milestones.
- The initial version tags will be "v1.1".
- Milestones will be determined based off the Gantt Chart.
- Additional milestones may be created throughout development.

5 Proof of Concept Demonstration Plan

5.1 Testing

When it comes to testing the project, it will be difficult to do any sort of automation as the output of the program is an image. It is however possible to automatically verify that the output will be the same for the same input, but whether or not that output is meaningful is a different question. We can demonstrate that through simple visual inspection of the output from different inputs, that the output is unique and as expected.

5.2 Scope Size

A major concern when selecting a project was that the scope of Pagan would be too small for a semester worth of work. This was given a lot of thought, and it was concluded that there are numerous features and enhancements (usability increase, readability increase) that can be made to the original program that can fill the expected full project duration time.

6 Technology

6.1 Programming Language

Python 3 will be used throughout the project due to the precence of its robust graphics library, and hash function library.

6.2 IDE

PyCharm will be the main IDE used as it provides a clear UI that shows the file structure of the projects as well as different associations between files.

6.3 Testing Framework

The unittest library will be the testing framework used to perform both the unit testing and integration testing. It has an intuitive suite of testing functions that allow the developers to track the assertions and display what failed and why.

6.4 Program for Document Generation

Doxygen will be used to generate documentation for the files. Doxygen is a good resource for developing documentation through comments within the actual code.

7 Coding Style

The coding standard we will be adopting will be the PEP-8 coding standard for Python3. It dictates certain naming conventions, commenting conventions, and general guidelines for making the program more readable. The documentation for this exists here - https://www.python.org/dev/peps/pep-0008/?

8 Project Schedule

Please refer to the project schedule located in ../ProjectSchedule/3XA3Ganttchart.pdf

9 Project Review