**Computing Project Proposal**

**Name:** Samuel Young

**Project Title:** SIMP = Sam Image Manipulation Program (Name open to change)

**Aim:**

* Allow user to create images
* Allow user to edit these images with a variety of tools such as brush, fill and shapes
* Allow the user to save the images to be loaded later
* Allow users to edit pre-existing images on their computer.

**Stakeholders:**

* Art students
* Those who create images on computer
* Those who want to make simple image edits

**Computational Suitability:**

* Pointer allows for precise on-screen locations to be entered quickly and intuitively
* Undo feature allows mistakes to be quickly erased
* Other computational features such as fill and gradient which would be impractical using traditional hardware
* File-saving feature allows for quick duplication of files, archiving and electronic sharing

**Requirements**

**Hardware**

* Mouse or other pointer device to enter cords and progress through menus
* **Not** a keyboard (should be fully accessible only mouse pointer) as to accommodate those who wish to draw with only a pointing device
* Monitor or other display device
* Computer
* Secondary drive to store saved images

**Software**

* Windows operating system to run the application
* File system to store saved files

**User Requirements**

From discussing with potential client I established these desired features:

* **Image editing**
  + RGB Colour Picker
  + Layer System
  + Resizable image editor
* **File system**
  + Saving file containing layer information
  + Importing and exporting popular image formats
* **Brushes & Other tools**
  + Brushes of varying sizes
  + Shape creation tool
  + Fill tool

**Limitations**

* Will only be supported on Windows operating systems, requiring use of a VM to run on other operating systems. This is because it will be written in C# in the Microsoft .NET application framework.
* Will only edit **Bitmap** images. There will not be vector support, this is to simplify design by focusing on only one image methodology.
* Will not have in-built touchscreen support, as I cannot test touchscreen devices.

**Testing Strategy**

* Alpha testing
  + Done internally by me
  + All major tests should be documented
  + As many tests as possible should be planned before implementation
* Beta testing
  + Done with potential clients
  + Done regularly, new build released weekly or fortnightly
  + Feedback taken and implemented into next build

**Key Milestones**

* GUI Design finalised
* Single-layer editing completed
* Layer system completed
* Importing files completed
* Exporting files completed
* Final testing & release

**Barriers to success**

* Creating a bespoke file system may be difficult, it will be likely saved in binary and so difficult to read and debug.
* Large edited images may result in large files. Rudimentary compression may be needed to help reduce file size.
* Other tools such as ‘Fill’ may be potentially time and CPU consuming if every square in selection is individually filled. Efficient algorithms will need to be researched.
* Layer manipulation may prove tricky to store each one separately and display in correct order.
* Time may be easily spread too thinly over small features (such as small brushes), taking focus away from larger tasks. Any changes to initial design should be well justified.

**Knowledge required**

* C#
* .NET Form creation knowledge
* System.Drawing class knowledge
* File handling knowledge
* Proprietary file format creation knowledge