

LAMP - Major Project 2018

Waxy Laser Solutions

Max Wharton-Jones, Shourov Quazi, Jack Jiang

August
2018



Contents

1 Defining the Problem	2
1.1 Client Details	2
1.1.1 Clients	2
1.1.2 Current System	2
1.2 Client Needs Research	3
1.2.1 Interview	3
1.3 Feasibility Study	7
1.3.1 Market feasibility	7
1.3.2 Technical feasibility	7
1.3.3 AutoCAD/Illustrator Interoperability	8
1.3.4 Utility and calibration	8
1.3.5 Laser cutter machine	8
1.3.6 School Systems	9
1.3.7 Financial feasibility	9
1.3.8 Operational feasibility	10
1.3.9 Social and ethical feasibility	11
1.3.10 Conclusion	11
1.3.11 Overall Feasibility	12
1.3.12 Possible Solutions	12
1.4 Rights Research	13
1.4.1 IP rights	15
1.4.2 Contract	15
2 Planning and Designing	18
2.1 Context Data flow diagrams	18
2.2 System Flowchart	20
2.3 IPO Chart	21
2.3.1 Select Template	21
2.3.2 Design Template	21
2.3.3 Queue Job	22
2.3.4 Approve Job	22
2.3.5 Set Up Laser Cutter	23
2.4 Gantt Chart	23
2.5 Data Dictionary	24
2.6 Algorithms	25
2.7 Screen Design Principals	37
2.7.1 Philosophy	37
2.7.2 Storyboard	38
2.7.3 Login Screen	39



2.7.4	Sign Up]	39
2.7.5	Home Screen	40
2.7.6	Create New Design	40
2.7.7	Home Screen	41
2.7.8	Create New Design	41
2.7.9	Create New Job	42
2.7.10	View Jobs	42
2.7.11	View Designs	43
2.7.12	Admin Tools	43
2.7.13	Approve/Reject Templates	44
2.7.14	Dialogs	44

Chapter 1

Defining the Problem

1.1 Client Details¹

1.1.1 Clients

Sydney Boys High School is an academically selective high school conducted by the NSW Department of Education. The school is led by the senior executive team, comprising the Principal, Dr Kim Jaggar, and Deputy Principals Ms Rachel Powell and Mr Robert Dowdell. The executive staff of nine Head Teachers and the twelve School Administration Officers led by Senior Administrative Manager Ms Sharon Kearns, support the senior executive. The school has three main offices - in the Main Building, in the Ken Andrews Library and in McDonald Wing. Finance, purchasing, enrolment and general inquiries are handled in the main building. Secretarial and network services are the responsibility of the McDonald Wing office. The clients for this project will be Ms Dam, Mr Comben and Dr Jaggar. Ms Dam and Mr Comben moderate the use of the laser cutter machine and are both teachers in the Industrial Arts department with Ms Dam being the head teacher. They are seeking a better system regarding the use of the laser cutter, especially with cutting trophies for their respective sports. Dr Jaggar is the principal of SBHS, who will be financing the project. Additional users include all other staff at SBHS and all current students of SBHS, however only Staff that are MIC of sports and other extracurricular activities will have access to the creation of trophies. Other staff and students will have access to the template creator only.

1.1.2 Current System

In the current system, all laser cutting requests are handled by the IA staff over email or in person. Individuals send their design files into to the IA staff or go in person to cut their objects with supervision. Only limited number of students and teachers have access to the system, and each job must be signed off by someone from the IA staff. Before laser cutting, the file must be checked manually by staff to ensure for correctness, and an often trial-and-error approach is used to ensure the correct settings are applied to each line type. Material is then aligned in the laser cutter, and the piece is cut, a process that can take between 5 minutes to several hours, depending on the complexity and size of the job. Only one job can be cut at once. There is no tracking of jobs, relying on email and paper to track jobs in progress and in queue. Currently, they use a program to help generate one type of template, the school trophy, although this program lacks several features, explored in the interview.

¹Client Details by Jack

1.2 Client Needs Research²

1.2.1 Interview

Interview with Mr Comben (MIC, IA Teacher)

Q: How many physical Awards are given out each year

A: The number varies through the years, but for rifle shooting in particular, there are mainly perpetual awards. For these, we would be looking for brass laser cut plaques.

Q: How much area in the budget is there for extra trophies?

A: Keeping in mind of the cost factor, including the cost for a teacher to operate the laser cutter, as well as the material, there is most likely not that much money available for many of the sports. I know rifle shooting has no area in the budget for extra trophies.

Q: How can we make the laser cutting process more efficient?

A: Look for vector-based fonts, this would allow faster cutting. You want faster cutting speeds to minimise the time spent waiting at the cutter. Also, try to make a reference-based system, this way the amount of time spent setting up the cutting process.

Q: What would be some improvements from the previous system that you would recommend?

A: The old system was very good, although due to the time constraint had many flaws. I would recommend an easier GUI. The old GUI was hard to navigate, and I believe there were some areas that were not functional. Also, the system of setting up the laser cutter took up a large bulk of time. If you can find a way to align the cutter easier, that would be good. Also, the workflow from AutoCAD is very unreliable, I would recommend exporting in illustrator

Q: What were some advantages in terms of resources from the previous system?

A: I can't say for sure about the budgetary resources saved, but I can say for sure that in the long term, the money saved would easily pay off the laser cutter. Also, the program has inspired a great deal of the industrial tech classes, we now have year 8's playing around with the potential of the cutter which is great.

Q: Do you have anything you want to see from our program?

A: I would like to see a function that would allow the user to nominate a folder of files that are ready to laser cut, and give the user detailed feedback on the user. Such as, this user has 90% black line in his work, and would take a long time to finish. This would allow us to check the jobs, and let it be easier to pass works.

Interview with Ms Dam. (MIC, Supervisor, Head of Industrial Arts)

Q: What are the costings of using the laser cutter in reference to the program used last year to create the trophies?

A: Last year, the trophies printed was a great success, especially considering the fact that it was the first year to use the laser cutter. The cost was of course the cost of the trophies from its original source. The profit came from the comparison of the cost of a teacher to be printing

²Interview by Shourov

L.A.M.P

Waxy LASER Solutions

Max Wharton-Jones, Shourov Quazi, Jack Jiang



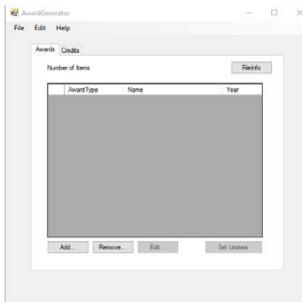
and the cost of printing the trophies elsewhere. The cost of a teacher is around \$400 a day whereas each trophy would cost \$10 elsewhere. Hence, if we could print 40 trophies a day, we would be making a profit, which we easily reached.

Q: What would you want to see from our program?

A: From your program I'd like to see it be a lot less time consuming. I would like to spend less time setting up the cutter and more time watching it cut. This would be done as Mr Comben said, to use a reference point system. Also, I would like to see a cleaner GUI.

Needs

Needs	Objectives
Must store > 50 different records	<ul style="list-style-type: none">• Store > 50 different templates in a database• Store > 50 users, clients• Store fonts and different shapes
An editor to create templates	A drag and drop interface supporting <ul style="list-style-type: none">• text with different fonts• circles• rectangles• lines
Use a variety of different materials	Program must indicate the material and settings to use on the laser cutter OR setting these values automatically before a job.
Be cost effective (man-power, trophy cost)	The program must be efficient in the usage and process of its materials. For example, when engraving school trophies for Sydney Boys High School. Assuming that: <ul style="list-style-type: none">• Each trophy laser cut saves \$10• A working day is 5 hours long.• The cost of a Teacher is \$400 a day Then, the program must be optimised such at LEAST 8 trophies can be cut per hour.
Minimum Manual work to save time	Program must assist the user in aligning the job in the laser cutter. Previously this was done manually with callipers.

Improvement on printing capabilities of the last product	To increase efficiency of the cutter: <ul style="list-style-type: none"> • Reduce Raster lines • Increase Vector Lines • No double lines • Vector-based fonts
Improved GUI usability	Current GUI has many issues, that must be rectified in the solution <ul style="list-style-type: none"> • Hard to navigate • Limited Usage • No tutorial • Some functions don't work like keybindings • Hard to line up 
Export/Import to Illustrator and Autocad	Ability to export/import files compatible with illustrator Illustrator AND/OR AutoCAD or other popular cad programs
Check Illustrator files for efficiency	A process that reads in an Illustrator file and <ul style="list-style-type: none"> • Reads all linetypes in the file • Compiles linetypes in numerical data • Quantitatively assesses linetype data • Provides estimate on print time • Displays linetype data in readable format



Ability to search and sort templates	<p>Ability to attach a number of tags to each template, which can be filtered by the user</p> <ul style="list-style-type: none"> • Template name • Template creator • Date created/Approved • Item ID • Template material • Purpose, e.g. athletics, rifle shooting
Different levels of access	<p>3 levels of access, with each level having all the permissions of the levels below</p> <ul style="list-style-type: none"> • Student (Lowest): can submit templates to be approved then used by teachers/administrators • Teachers: can also submit jobs to administrators for approval, who run the laser cutter. Can also approve templates. • Administrators: have all permissions, can view all information and approve all jobs/templates, to be cut by the laser cutter.

Specifications

The school computers are mostly Dell ThinkCentre machines, with the following specifications

CPU	Intel® Pentium® CPU G3220 @ 3.00 (GHz)
RAM	8192MB (8GB)
Graphics Adapter	Intel® HD Graphics
Operating System	Windows 10 Pro

Therefore, program requirements should include:

	Minimum Requirements	Recommended Requirements
CPU	Dual Core 2 GHz+	Quad Core 3 GHz+
RAM	512 MB	4GB
Operating System (Client)	Windows 7	Windows 7, 8, 10
Dependencies (Client)	NET Framework 4.0	NET Framework 4.0
Operating System (Servier)	Windows 7, MacOSX 4.0, Ubuntu 16.04+	Windows 7+, Ubuntu 16.04+

- We chose to limit the Client Program to the Windows operating system, all the school computers run windows exclusively. However, the server software must be cross platform, as servers are often run from a variety of OS's (Linux, windows etc.)
- The program should run well on school computers, as they meet both the minimum and recommended requirements.

1.3 Feasibility Study ³

1.3.1 Market feasibility

The proposed plan is to design a software product that will use the laser cutting machine to produce physical products. The project involves several industries: software design, industrial production / manufacturing and product design.

The low volume and DIY (Do it yourself) manufacturing market has flourished, due to the higher availability and the diminishing costs associated with manufacturing. Flexible Manufacturing Systems (FMS), like the laser cutter or 3D printer allow for arbitrary objects to be created from computer-aided designer (CAD) files.

Currently, solutions on the marketplace include: 2D CAD programs, which although work with the laser cutter, becomes inconvenient and inefficient when producing a large quantity of objects with variations, and complex templating solutions which require a significant amount of programming and technical expertise. The project involves using a program to aid in the generation and production of files that are sent to the laser cutter. There are few or no off-the-shelf type programs that achieve this, as laser cutting is often a specialised and expensive industrial process. It is often cumbersome for students and teachers to create multiple copies of a single object, which the program seeks to automate Conclusion: LAMP focuses on the niche in the market. There are few commercial programs that help with low volume production, a market that has boomed in the last five years. Since the demand for such a software is increasing and there are few competitors, this program will have a well-defined target market, and is market feasible.

1.3.2 Technical feasibility

There are many technical components that need to be researched and experimented on beforehand to ensure that the project's success. Fortunately, the laser cutting project developed last year can be used as a proof of concept, with many technical barriers solved which can be ported over to the new program (existing software). Designer Technology: LAMP will include a designer that allows editing of templates dynamically inside the program. It will essentially be a 2D only CAD screen, with options for lines, circles, boxes, shapes and text. Text may be in different sizes and fonts. The designer will allow the three different cutting types to be specified for individual elements (vector cut, vector engrave, raster engrave). This will then be saved in a custom format (.spiff file), which contains all the data required to store the required

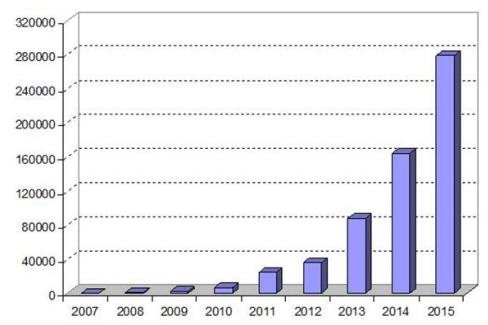


Figure 1.1: Desktop 3D printers sold.

Source: Wohlers Report 2016

³Feasibility Study by Jack



template. Placeholder elements can be specified in the .spiff file, so that the template can be filled in automatically through the program with a list of names, or years.

- Basic 2D CAD interface
 - rendering of elements onto screen
 - optimization to run on lower-powered computers
 - real dimensions (cm/mm)
 - zoom
 - different modes (cutting, engraving)
- custom file type
 - .SPF file contains data on placeholder elements (text that needs to be replaced) and also line/box data
 - read and write .SPF files

1.3.3 AutoCAD/Illustrator Interoperability

The .SPF file will need to be exported to illustrator (.ai) and/or AutocAD (.dxf) files to be used by the laser cutter. The program will also need to be able to take a list of names/years from a document file, using this data to replace the placeholder elements on a template, and layout the template a variable amount of time in the output file in an optimum matter, taking into account the total space in the laser cutter. Manual alignment will also be possible.

- export to vector format (ai/dxf)
 - writing/reading design files
 - generating different vector lines
 - vector fonts
 - vector lines and curves
- read/write document files (.docx, .xlsx, .csv)
- layout of multiple copies of template

1.3.4 Utility and calibration

The program will have error-checking algorithms on .AI files, to ensure it has lines that are compatible with the laser-cutter.

- read/write illustrator files
 - file checking algorithm for illustrator files

1.3.5 Laser cutter machine

The laser cutter machine is a complex piece of hardware, capable of cutting thin material via vector lines, and engraving with both vector and raster modes available. The laser bed or cutting space is 450 x 600mm, which will limit the maximum number of trophies cut at once. It uses a 50 watt, 10µm laser, and the materials it can cut are thin woods and plastics, but it can engrave a variety of materials, including plastics, woods and glass.

L.A.M.P

Waxy LASER Solutions

Max Wharton-Jones, Shourov Quazi, Jack Jiang



-
- Capabilities of the laser cutter
 - engraves plastics, woods and glass
 - cuts thin woods and plastics
 - Size of the laser cutter (450 x 600mm)
 - Safety of operation

1.3.6 School Systems

LAMP stores the approved templates which are available to anyone using the program and the jobs in queue in a server. This may be run locally on the school server, or in a cloud server hosted on SaaS platforms. The files and credentials of users will need to be encrypted, to increase the safety of the system, and some basic measures need to be taken to stop hacking or denial of service (DOS) attempts. This server will come as a separate executable to the client system used by the end user. The client software will need to be able to run on school computers, which mostly have dual-core intel processors with between 8 to 16 gigabytes of ram. Approval from the school's IT staff may be required to install LAMP's client software.

- Server Software (if required)
 - file and credential encryption
 - serve requests to list all approved templates
 - unblocked from school internet
 - needs to be secure and reliable
- Client Software
 - may need administrator permissions to install on school systems (ask IT).
 - will connect to the server through the internet, or the schools internal intranet.
 - optimization to run on school's computers

Conclusion

There are a large number of technical challenges to solve in order for lamp to succeed. Fortunately, several of these have already been addressed in the previous program, and the Industrial Arts staff at school understand the laser cutting system, providing enough information to explain many of the laser-cutting related problems. Over the holidays and throughout the year, research will need to be done on the schools systems, the designer interface and our custom file type. Thanks to the previous program and our teams previous experience with working on the laser cutter, we will be able to focus on these issues instead, reducing the amount of work needed. Overall, the program will be technically feasible.

1.3.7 Financial feasibility

A middle-end laser cutting machine is an expensive instrument, coming around between \$20,000 - \$100,000. There are cheaper alternatives, but they are slower and/or less precise. However, as the school already has bought a laser cutter, the costs are mostly maintenance and teacher time. The original trophy system can be used as a proof of concept, and has proven to be cost effective. Expanding this system to other awards may allow for even more cost saving. Awards for different sports account for a significant savings. Costs can be further reduced



by bulk buying many trophies from overseas. Take for example the previous program, which focused on a particular trophy, the School Trophy. Initially it had cost the school 135\$ per trophy, including raw material and engraving costs. However, a blank trophy can be sourced for 15\$, and engraved on the school's cutter. Given that the hourly rate for teachers is 80\$ per hour, with each school trophy taking approximately 15 minutes of time, and producing a trophy in-house would cost 35\$, or a 75% decrease in cost. LAMP will decrease the time required to setup and cut the trophy, and also allow for other types of trophies to be cut through its templating system, which the old system could not do.

We will require some material and test awards to experiment with the abilities of the laser cutter, which needs to be accounted into our budget. Other costs may include licensing libraries, distribution costs and/or server maintenance. However, even with all these costs factored in, the in-house production of trophies along with other objects will still be cost-effective, saving the school thousands of dollars per annum.

The system will have some server setup and maintenance costs - however, this will be low, as it can be hosted on the school's existing servers or through inexpensive cloud providers. The program will not need much processing power, and will not handle a large amount of data, further reducing the server costs. Other setup costs include install time and storage space for LAMP's client software. A developer may be hired to continue to maintain the program after its release, and to fix any bugs discovered after.

LAMP will require a significant amount of developer time, and falling into the medium range in terms of software, probably costing between \$20,000 - \$60,000, based off several other custom software projects. Developer time costs between 75\$ to 200\$ per hour, and the project overall will take around 200-300 hours. The software will be licensed to the school, and may be licensed to multiple schools and businesses. A fee will be charged per user per year, with business subscriptions including priority tech support and user management features. There will be 3 tiers: individual licence will be between 10-30\$ per year for 1 individual, small business (10-1000 users) for 20-40\$ per year per user, and large businesses negotiated separately. Small and large business are also given a copy of the server software that can be setup on their own servers to serve their organisation's users. This copy will be completely separate from the systems of other businesses, allowing queued jobs and credentials to be kept separately.

Conclusion

The project's main expenses are developer time, and will cost approximately \$40,000. This will be recouped by licensing the software to multiple business and individuals, and charging a yearly fee, which will also help pay the maintenance developers. For the business, LAMP will decrease operational costs by automating parts of the laser cutting process. Therefore, LAMP is financially feasible.

1.3.8 Operational feasibility

Users

Users may include teachers and some select students. Teachers will be able to use the program to quickly and easily submit a set of awards for some students. The process should not be changed significantly: the list of students will be sent to the program instead of emailed to an awards seller. The user interface will need to be reliable, consistent and uncomplicated, especially the template designer, which will allow both students and teachers to create the shapes and text without beforehand knowledge of the intricacies of CAD programs and the laser cutter. Little training will be required to use the system for users - pick a template, give some names, submit

job. This information will be provided by video tutorials, reference manuals and online help. On-call tech support may be available for business users. Users are able to design templates and/or submit jobs, depending on the permissions given to the user by an administrator.

Administrators

Administrators/IA staff will manage and approve request from the users. This will take a significant amount of time, that would otherwise be spent on teaching. To reduce the impact this has, the program will attempt to automate many of the intermediary steps in setting up the laser cutter, and allow for less time calibrating the machine, a process that takes 10-20 minutes each set of trophies. It will also use more efficient processes to reduce the production time. The program will first be tested on only the industrial arts staff, to ensure the time required to process these trophies will not be an overwhelming amount of work. Administrators have the highest level of access to the program, with permissions to approve both submitted jobs and submitted templates, create and manage users and other administrators, reset passwords etc. This will require a significant amount of training, through video tutorials, reference manuals and online help, with tech support available for businesses. An administrator will require experience with the laser cutter, as they must physically set up the material on the laser cutter in accordance to the current job.

Conclusion

For users, the program will require very little training. This means that it will be easy to introduce new users to the system. Administrators will require significant training; however, the industrial arts staff are already accustomed to the laser cutter, easing this process. Support will be given in the form of video tutorials, reference manuals, online support, with on-call technical support for businesses. Overall, this system will not require much training.

1.3.9 Social and ethical feasibility

The program will handle some sensitive data from users - their full name, email, passwords, secret questions and/or contact information will be required by the program in order to operate. This will be stored on the client's computers, and on LAMP's server software if applicable. Security will need to be kept in mind to prevent unauthorized access to this sensitive data, through extra security measures taken on the client's computers, e.g. anti-virus software, correct user privileges, and in the program, e.g. encryption of database, authentication through passwords. Additional server software provides another vector for attackers, and appropriate security measures, like https will need to be used to ensure communications between the program and the server are safe, and to prevent access from unauthorized users. Care will need to be taken to ensure the privacy of the information given to the program by preventing unauthorized access. The program should be as inclusive as possible: this would require the program work on lower-powered machines, have an easy and under stable user interface, and users with disabilities considered. Copyright and intellectual property is another possible issue, with appropriate credit given and/or licenses obtained from code included from another source. The program may reduce the amount of time spent laser cutting, but since laser cutting is a secondary job to teaching at Sydney Boys High School, its effect will be negligible on the workforce.

1.3.10 Conclusion

The privacy of users will need to be considered when developing the solution. Appropriate measures need to be taken to prevent unauthorized access. The software should be inclusive to

all users, by altering the user interface to suit the needs of individuals. Intellectual property rights of others need to be considered in the program.

1.3.11 Overall Feasibility

the issues mentioned with this feasibility study will need to be resolved, but all essential components of the program will be met in the time period set in the Gantt chart

1.3.12 Possible Solutions

Aim: To better use the laser cutting system by simplifying the process required to design and cut objects, and allowing teachers and students limited access into the system.

1. "Upgrade" Approach

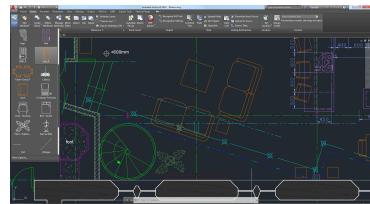
This approach would see the original trophy generation revisited and revamped to fix issues outlined by staff. In addition, new features could be added, like support for different shapes or awards, and improvements could be made to the GUI to increase its useability. This program would only be accessible and usable by IA staff to generate shapes, which can then be rendered by Illustrator a format the laser cutter can use.

- Low cost, complexity and maintenance required
- Code reuse, which decreases development time and cost
- Similar to system already in use, reduceds training required



2. "Designer" Approach

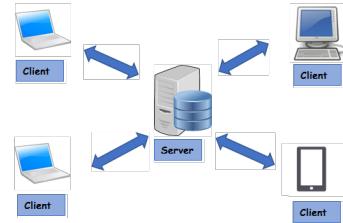
This approach requires a custom piece of software to be created, and is similar to the system already in use in the school. Using a program to generate a known-good file will allow IA staff to skip several steps in the laser cutting process. However, this incurs additional cost, both in development/setup time and cost. Some basic training and documentation will also need to be provided to the industrial arts teachers using the program. This solution does not attempt to track the created files, relying on email or another form of communication to send and receive files.



- Design tool that loads much faster than AutoCAD and uses less resources
- Specialised user interface that contains only the tools required for laser cutting templates
- File checking features to ensure linetypes are correct
- Automates some printing settings to the laser cutter

3. “Server-Client” Approach

This approach uses a server to store templates and jobs that users can edit and use. Using a server allows easier communication between the end-users and the IA staff, but incurs additional cost and setup complexity. Using a server also entails an additional, recurring cost of hosting the server. Security over the net could also be an issue, and care will need to be taken to avoid access by unauthorized individuals over the internet. On the other hand, the server-client approach will allow for better tracking of individual jobs, centralization of users to ensure only the correct people have access to the program through user login and server-side checking of files. This system also may allow users/administrators to use the system from multiple places, as long as there is a working internet connection Features:



- Loads much faster than AutoCAD and uses less resources
- Specialised user interface that contains only the tools required for laser cutting templates
- File checking features to ensure linetypes are correct
- Automates some printing settings to the laser cutter
- Templates on server can be accessed from anywhere with an internet connection
- Tracking of jobs that are in queue or complete
- User and Administrator management

The client has chosen option 3 The client would like to choose Option 3. Option 1 is far too basic and not really an improvement on the current system. Option 2 again is not an improvement on the current system, as we already use a server. Option 3 contains many new features that could help improve the efficiency of laser cutting, despite the extra cost.

1.4 Rights Research ⁴

A software licence determines the use and redistribution of software. It determines how the software can be used by the purchaser of the software, often called the licensee, and may protect the developer legally from damage caused by the software.

⁴Section by Max



There are several types of licences:

- Public domain
- Open source (FOSS) licenses
- Freeware / Shareware
- Proprietary

Software licenses and rights granted in context of the copyright according to Mark Webbink. ^[1] Expanded by freeware and sublicensing.						
Rights granted	Public domain	Permissive FOSS license (e.g. BSD license)	Copyleft FOSS license (e.g. GPL)	Freeware/Shareware/Freemium	Proprietary license	Trade secret
Copyright retained	No	Yes	Yes	Yes	Yes	Yes
Right to perform	Yes	Yes	Yes	Yes	Yes	No
Right to display	Yes	Yes	Yes	Yes	Yes	No
Right to copy	Yes	Yes	Yes	Often	No	No
Right to modify	Yes	Yes	Yes	No	No	No
Right to distribute	Yes	Yes, under same license	Yes, under same license	Often	No	No
Right to sublicense	Yes	Yes	No	No	No	No
Example software	SQLite, ImageJ	Apache web server, ToyBox	Linux kernel, GIMP	Irfanview, Winamp	Windows, Half-Life 2	Server-side World of Warcraft

Open source licenses, like the GNU GPLv3 licence are for collaborative projects, where developers create code, often for free for their own use. Any developer can download and alter the source code of a GPL project, but they must provide the altered source code to end-users for their derivative work, display a notice on the program, crediting the original developers of the source code and license their work under the GPL. Many open source projects use this licence, as it ensures that their work will be credited and improvements to the software carried out will be made free to the public. The MIT licence is another software licence. It is a short, simple licence, that allows the alteration of source code with no other conditions. Both these open source licences disclaim any warranties or responsibilities of the original developer in the quality and usability of the code.

Copyright <2017-2018> <Max Wharton-Jones>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

The MIT Licence

Freeware licences may use ads or donations in order to make a profit. However, it often lacks enterprise support. Shareware uses locked features or a trial period, allowing users to try out the software before committing to a purchase.

Open source licences are unsuitable for our project, as they require the distribution of source code. In addition, there is no need for an open source licence as the project will only be created and maintained by our team. A freeware / Shareware licence is unsuitable, as our program will

L.A.M.P

Waxy LASER Solutions

Max Wharton-Jones, Shourov Quazi, Jack Jiang



mostly target large organisations, who are willing to pay extra in return for support. Therefore, the proprietary licence will be the most suitable. The software will be maintained by our team, allowing for more features and bug fixes when discovered, funded by the licensing fee. The safety, reliability and usability of the program is essential, as it involves the laser cutter, an expensive and potentially dangerous machine.

1.4.1 IP rights

Waxy LASER Solutions retains all intellectual property rights to the software. This is necessary so that the program can be licensed to other businesses, and to allow the program to be maintained by our team in the future.

1.4.2 Contract

L.A.M.P - Terms and conditions - Waxy LASER Solutions

1. Preamble: This Agreement, signed on Dec 6, 2017 (hereinafter: Effective Date) governs the relationship between Sydney Boys High School, a School Entity, (hereinafter: Licensee) and L.A.M.P, a partnership under the laws of whose principal place of School is 556 Cleveland St, Moore Park NSW 2021 (hereinafter: Licensor). This Agreement sets the terms, rights, restrictions and obligations on using L.A.M.P (hereinafter: The Software) created and owned by Licensor, as detailed herein

2. License Grant: Licensor hereby grants Licensee a Personal, non-assignable and non-transferable, commercial, royalty free, non-exclusive license, all with accordance with the terms set forth and other legal restrictions set forth in 3rd party software used while running Software.

1. Limited: Licensee may use Software for the purpose of:
 - (a) Running Software on Licensee's Website[s] and Server[s];
 - (b) Allowing 3rd Parties to run Software on Licensee's Website[s] and Server[s];
 - (c) Publishing Software output to Licensee and 3rd Parties;
 - (d) Distribute verbatim copies of Software's output (including compiled binaries);
2. This license is granted perpetually, as long as it is not materially breached.
3. **Binary Restricted:** Licensee may sublicense Software as a part of a larger work containing more than Software, distributed solely in Object or Binary form under a personal, non-sublicensable, limited license. Such redistribution shall be limited to 1600 codebases.
4. **Non-Assignable and Non-Transferable:** Licensee may not assign or transfer his rights and duties under this license.
5. **Commercial, Royalty Free:** Licensee may use Software for any purpose, including paid-services, without any royalties

3. Term & Termination: The Term of this license shall be until terminated. Licensor may terminate this Agreement, including Licensee's license in the case where Licensee:

1. became insolvent or otherwise entered into any liquidation process; or
2. exported The Software to any jurisdiction where licensor may not enforce his rights under this agreements in; or
3. Licensee was in breach of any of this license's terms and conditions and such breach was not cured, immediately upon notification; or

L.A.M.P

Waxy LASER Solutions

Max Wharton-Jones, Shourov Quazi, Jack Jiang



-
4. Licensee in breach of any of the terms of clause 2 to this license; or
 5. Licensee otherwise entered into any arrangement which caused Licenser to be unable to enforce his rights under this License.
-
- 4. Payment:** In consideration of the License granted under clause 2, Licensee shall pay Licensor a fee, via Credit-Card, PayPal or any other mean which Licensor may deem adequate. Failure to perform payment shall construe as material breach of this Agreement.

5. Upgrades, Updates and Fixes: Licensor may provide Licensee, from time to time, with Upgrades, Updates or Fixes, as detailed herein and according to his sole discretion. Licensee hereby warrants to keep The Software up-to-date and install all relevant updates and fixes, and may, at his sole discretion, purchase upgrades, according to the rates set by Licensor. Licensor shall provide any update or Fix free of charge; however, nothing in this Agreement shall require Licensor to provide Updates or Fixes.

1. Upgrades: for the purpose of this license, an Upgrade shall be a material amendment in The Software, which contains new features and or major performance improvements and shall be marked as a new version number. For example, should Licensee purchase The Software under version 1.X.X, an upgrade shall commence under number 2.0.0.
 2. Updates: for the purpose of this license, an update shall be a minor amendment in The Software, which may contain new features or minor improvements and shall be marked as a new sub-version number. For example, should Licensee purchase The Software under version 1.1.X, an upgrade shall commence under number 1.2.0.
 3. Fix: for the purpose of this license, a fix shall be a minor amendment in The Software, intended to remove bugs or alter minor features which impair the The Software's functionality. A fix shall be marked as a new sub-sub-version number. For example, should Licensee purchase Software under version 1.1.1, a fix shall commence under number 1.1.2.
-

6. Support: Software is provided under an AS-IS basis and without any guarantees of updates or maintenance. Nothing in this Agreement shall require Licensor to provide Licensee with fixes to any bug, failure, mis-performance or other defect in The Software.

1. Bug Notification: Licensee may provide Licensor of details regarding any bug, defect or failure in The Software promptly and with no delay from such event; Licensee shall comply with Licensor's request for information regarding bugs, defects or failures and furnish him with information, screenshots and try to reproduce such bugs, defects or failures.
2. Feature Request: Licensee may request additional features in Software, provided, however, that
 - (a) Licensee shall waive any claim or right in such feature should feature be developed by Licensor;
 - (b) Licensee shall be prohibited from developing the feature, or disclose such feature request, or feature, to any 3rd party directly competing with Licensor or any 3rd party which may be, following the development of such feature, in direct competition with Licensor;
 - (c) Licensee warrants that feature does not infringe any 3rd party patent, trademark, trade-secret or any other intellectual property right; and
 - (d) Licensee developed, envisioned or created the feature solely by himself.

7. Liability: To the extent permitted under Law, The Software is provided under an AS-IS basis. Licensor shall never, and without any limit, be liable for any damage, cost, expense or any other payment incurred by Licensee as a result of Software's actions, failure, bugs and/or any other interaction between The Software and Licensee's end-equipment, computers, other software or any 3rd party, end-equipment, computer or services. Moreover, Licensor shall never be liable for any defect in source code written by Licensee when relying on The Software or using The Software's source code.

8. Warranty:

1. **Intellectual Property:** Licensor hereby warrants that the Software does not violate or infringe any 3rd party claims in regards to intellectual property, patents and/or trademarks and that to the best of its knowledge no legal action has been taken against it for any infringement or violation of any 3rd party intellectual property rights.
2. **No-Warranty:** The Software is provided without any warranty; Licensor hereby disclaims any warranty that The Software shall be error free, without defects or code which may cause damage to Licensee's computers or to Licensee, and that Software shall be functional. Licensee shall be solely liable to any damage, defect or loss incurred as a result of operating software and undertake the risks contained in running The Software on Licensee's Server[s] and Website[s].
3. **Prior Inspection:** Licensee hereby states that he inspected The Software thoroughly and found it satisfactory and adequate to his needs, that it does not interfere with his regular operation and that it does meet the standards and scope of his computer systems and architecture. Licensee found that The Software interacts with his development, website and server environment and that it does not infringe any of End User License Agreement of any software Licensee may use in performing his services. Licensee hereby waives any claims regarding The Software's incompatibility, performance, results and features, and warrants that he inspected the The Software.

9. No Refunds: Licensee warrants that he inspected The Software according to clause 7 and that it is adequate to his needs. Accordingly, as The Software is intangible goods, Licensee shall not be, ever, entitled to any refund, rebate, compensation or restitution for any reason whatsoever, even if The Software contains material flaws.

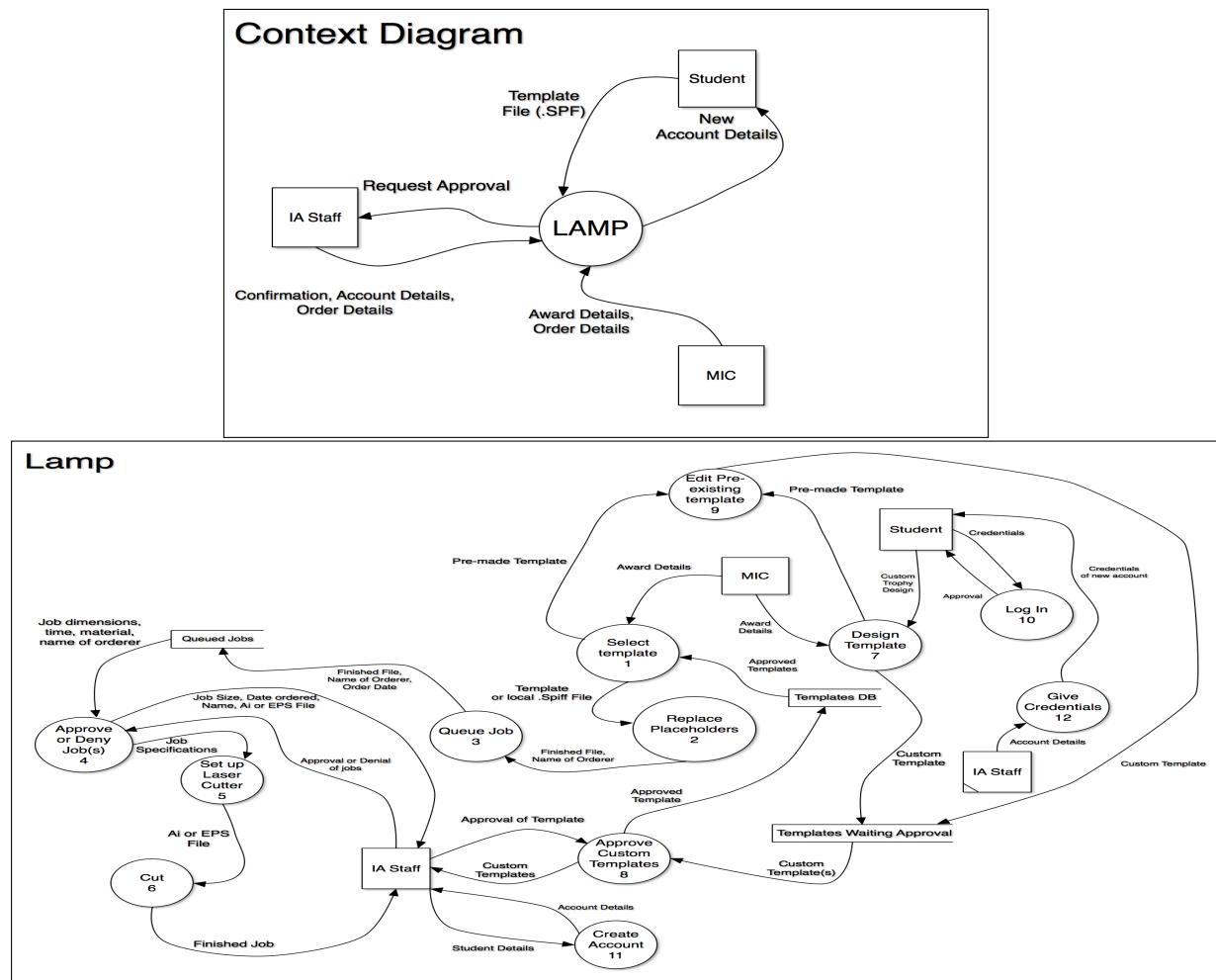
10. Indemnification: Licensee hereby warrants to hold Licenser harmless and indemnify Licenser for any lawsuit brought against it in regards to Licensee's use of The Software in means that violate, breach or otherwise circumvent this license, Licenser's intellectual property rights or Licenser's title in The Software. Licenser shall promptly notify Licensee in case of such legal action and request Licensee's consent prior to any settlement in relation to such lawsuit or claim.

11. Governing Law, Jurisdiction: Licensee hereby agrees not to initiate class-action lawsuits against Licenser in relation to this license and to compensate Licenser for any legal fees, cost or attorney fees should any claim brought by Licensee against Licenser be denied, in part or in full

Chapter 2

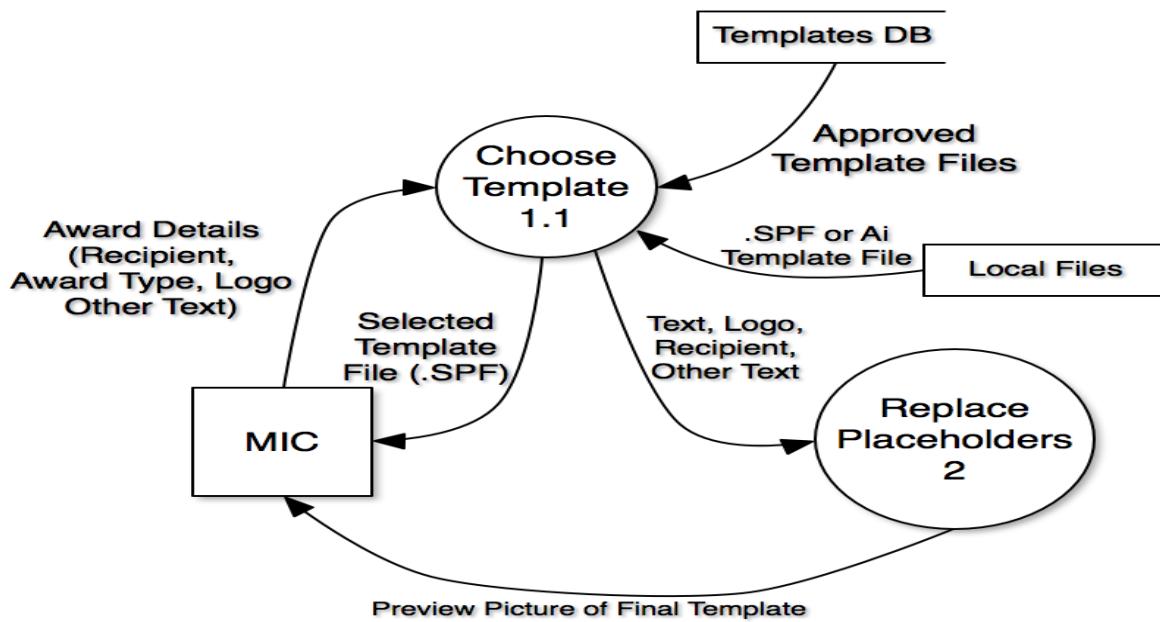
Planning and Designing

2.1 Context Diagram and Data flow diagrams ¹

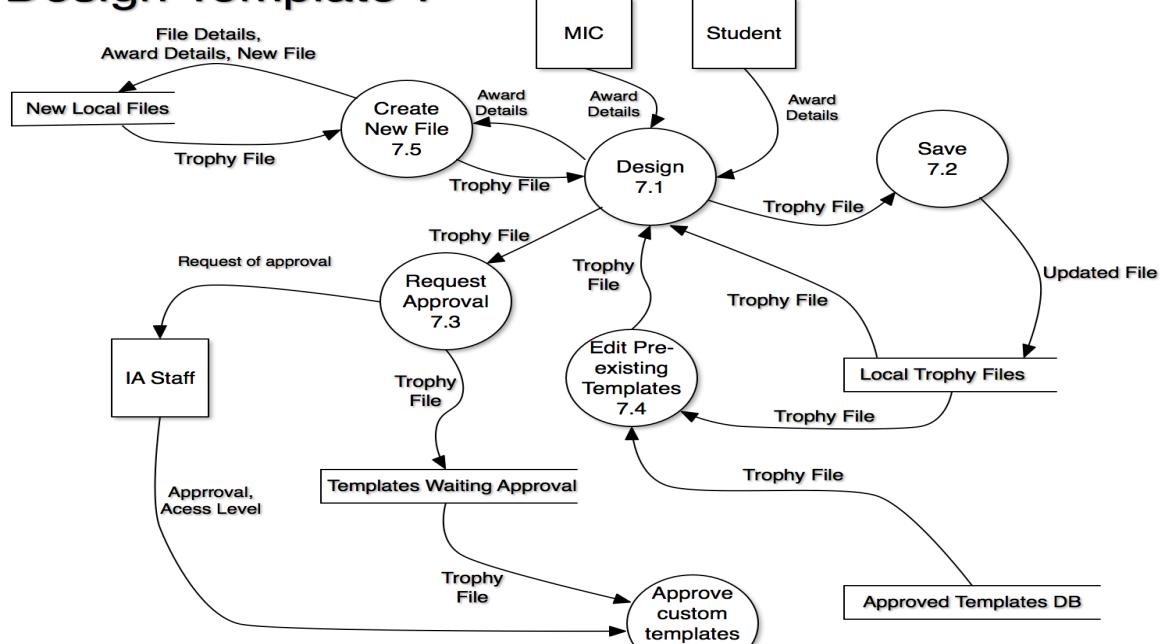


¹Context and DFD by Max

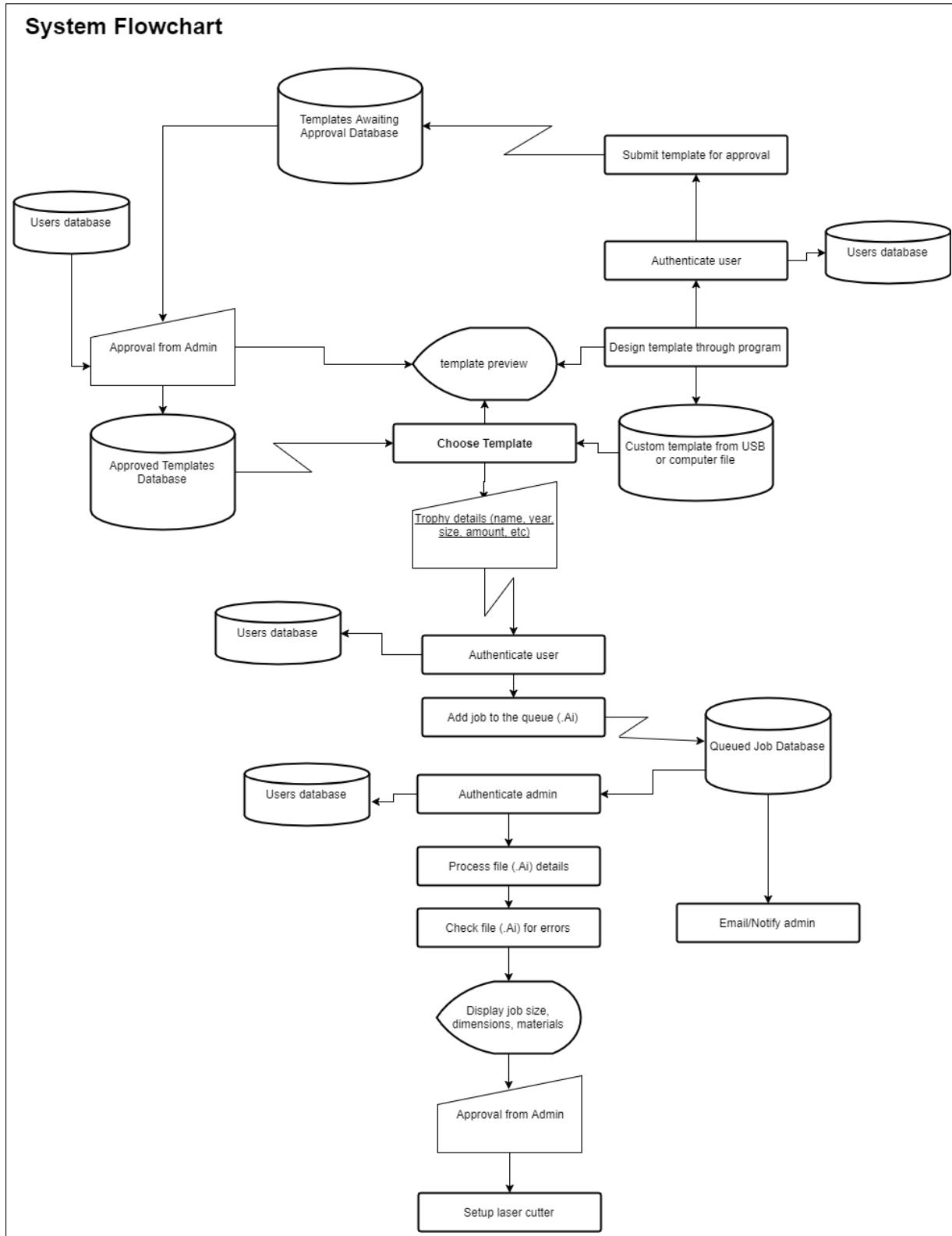
Select Template 1



Design Template 7



2.2 System Flowchart²



²System Flowchart By Jack



2.3 IPO Chart ³

2.3.1 Select Template

Input	Process	Output
Array of SPF files	Obtains Template From Database Displays all template designs in the template database Client selects appropriate template from the database	Partial SPF File containing data for the template selected
Names (string) Number of Awards (int) Args*(vary)	Adds Data to the SPF file containing the type of template already chosen <i>* based on the template chosen, the user will be prompted for certain data. An example is for a trophy that requires both a name and a score. The user will be prompted to enter the name as well as the score. If left blank, the input will be considered invalid unless otherwise specified.</i>	Complete SPF File containing both the selected template data as well as the variation data specified by the user

2.3.2 Design Template

Input	Process	Output
Array of SPF files	Creates Template from Selection Selection From Client <ul style="list-style-type: none"> • Presented as a selection GUI • Displays all template designs in the template data base • Client selects appropriate template from the database 	Partial SPF File containing data for the template selected

³IPO Chart by Shourov



Raw AI / EPS Template File	Edits Template with Graphical Editor Graphical Editor <ul style="list-style-type: none"> • Editor Based off of CAD software • Tools include Line creation • Ability to add images • Ability to add dynamic text ⁴ • Ability to add static text ⁵ • Ability to create CUT lines and ENGRAVE lines 	AI / EPS Template File
Login / Password (string)	Authenticates User	Authentication Level (Admin / Teacher / Student)
AI / EPS Template File Authentication level of Client(integer)	Sends to get verified	None

2.3.3 Queue Job

Input	Process	Output
Login / Password (String)	Authentication	Authentication Level (Admin / Teacher / Student)
SPF File containing <ul style="list-style-type: none"> • Template Selected from Template Database • Variation Data unique to the job required to client • Authentication level of Client • Client Details 	Add to Job Queue Database	Reponse code

2.3.4 Approve Job

Input	Process	Output
Login / Password (String)	Authentication	IF Authentication level is not Admin deny Approval ability

⁴Dynamic text is text which is different for each job from the template file. An example of Dynamic text is the name on a trophy

⁵Static text is text which appears on every job from the template file. An example of Static text is the year on a trophy



SPF File from Job Queue DB	<p>Approval Process Admin level client will see</p> <ul style="list-style-type: none"> • Preview of SPF File • Line weight detail of the SPF file • Authentication level of the user • Client Details 	<p>IF Approved the job is sent to a folder to be laser cut The SPF File is converted to AI / EPS files to match the format of the Laser Cutter. If required, the job is split into multiple files IF Not Approved the job is deleted</p>
-----------------------------------	--	--

2.3.5 Set Up Laser Cutter

Input	Process	Output
Login / Pass-word (String)	Authentication	IF Authentication level is not Admin deny Approval ability
AI / EPS File	<p>Laser Cutter Process</p> <ul style="list-style-type: none"> • An Admin must place the resource within the Laser Cutter • Open the files on the Laser Cutter computer • Print via Laser Cutter printer to the Laser Cutter • Remain by the Laser Cutter until the job is finished • Remove finished job and replace material if need be 	Final Job

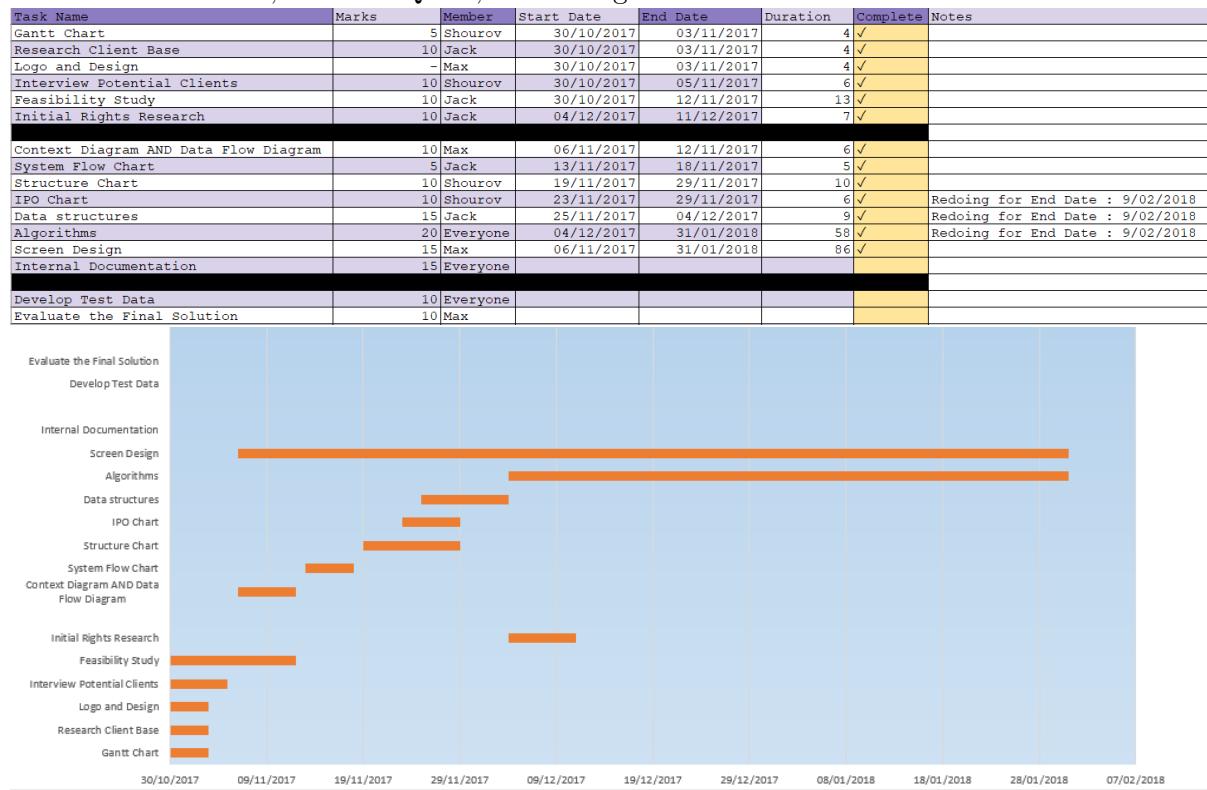
2.4 Gantt Chart ⁶

⁶Gantt Chart by Shourov

L.A.M.P

Waxy LASER Solutions

Max Wharton-Jones, Shourov Quazi, Jack Jiang



2.5 Data Dictionary ⁷

Data Structure	Data Type	Length ⁸	Description
tblTemplates Array (TemplateRecord)	Array(Of Record) Auto.ID (Auto Number) DXF (String) tags (Array(Of String)) material (String) length (real) height (real) thickness (real) preview (Array(Of Image)) creatorId (Integer) approverId (Integer) dynamicTextList(Array(Of String)) textLocation(Array(Of Point))	32 10000 20x20 10 8 8 8 3x4MB 4 4 10x10 10x16	Contains details about approved templates created using the program. can be saved/loaded as .SPF files Index 0-2147,483,647 (for database). Primary Key Line and text data encoded using the DXF format the criteria that the template meets the material that the template uses the length of each piece, in millimeters the height of each piece, in millimeters the thickness of each piece, in millimeters preview images for each template id of the creator, joined to tblUser.Auto.ID id of the approver, joined to tblUser.Auto.ID Dynamic texts are text labels that are filled in by the user. They describe the name of the label (e.g. year) the location where the dynamic text will be filled
tblSubmittedJobs Array (JobRecord)	Array(Of Record) Auto.ID (Auto Number) template.ID submitterId (Integer) approverId (Integer) approved (boolean) submitDate (date)	32 32 4 4 1 32	Contains information about jobs submitted to the IA staff for cutting. Index 0-2147,483,647 (for database). Primary Key the id of the spf file to cut. Joined to tblTemplates.Auto.ID the id of submitter. Joined to tblUser..ID the id of approver. Joined to tblUser..ID whether or not the job is approved the date in which the job is submitted
tblUsers Array (UserRecord)	Array(Of Record) Auto.ID (Auto Number) email (string) password (string) accesslevel (integer) name (string)	32 20 20 4 20	Contains information about all the users that are stored in the database Index 0-2147,483,647 (for database). Primary Key the email used to sign up password used to sign in 1=student, 2=teacher, 3=admin. Determines if the user can do certain actions full name of the user
lines Array (new-line)	Array (Of Record) point1x (real) point1y (real)	8 8	Stores data about a shape or line drawn using the template designer. Lines is the array containing all the currently drawn lines start point x co-ordinate start point y co-ordinate

⁷Data Dictionary by Jack



	point2x (real) point2y(real) lineType (string) color (string) lineNumber (integer)	8 8 20 20 4	end point x co-ordinate end point y co-ordinate the type of shape drawn color of line the index of the line in the array
Point	Record x1 (real) y1 (real)	8 8	Stores data of 1 point in 2d space location on x axis location on y axis
Unapproved Templates	Array (Of JobRecord)	20 x 10MB	Stores all the unapproved Templates

SCOPE	Identifier	Data Type	Length	Description
Global	running	boolean	1	whether the program is running or not
	userTags	Array (string)	20	Shows the tags that the user currently want to look at
	CurrentUser	UserRecord	1	Stores the currently logged on user
	loggedIn	boolean	1	shows if the user is logged in or not
	dataConnection	OleDbConnection	1	Used to connect to the database
	dataAdapter	OleDbDataAdapter	1	used to read and write data to the database
	tool	string	1	currently selected tool
	linesCollection	Array (Line)	10000	stores all the lines on the screen
	USER	constant integer	1	accesslevel of user=1
	TEACHER	constant integer	1	accesslevel of teacher=2
	ADMIN	constant integer	1	accesslevel of admin=3
	sortBy	string	1	how to sort the database, either by date or material
FilterTemplate	MatchingTemplate	Array (Template)	100	An array containing the templates that match a user's preferences
Global	prefCount	integer	1	Counter for looping through userpreferences
	tagCount	integer	1	Counter for looping through the tags in a template
SortByDate SortByMaterial SortByName	sortedTillElement	integer	1	The index that the array is sorted up to. Starts at 0 (before the first element)
	largestpos	integer	1	the position of the largest element in the sub-array after index sortedTillElement
	upto	integer	1	the index of the element that the algorithm is currently checking
	nextUnsorted	integer	1	the index of the next unsorted element in the array
checkValidSPF	file	file (SPF)	1	the spf file to check
	lineData	Array (Line)	10000	the lines that the spf file contains

2.6 Algorithms

```

1 BEGIN MAIN — Jack —
2   running is a boolean = true
3
4   InitializeDatastructures
5   Login
6   LoadTemplates
7
8   WHILE running = true
9     selectModule(Max)
10  END WHILE
11 END MAIN
12
13 BEGIN SUB selectModule — Max —
14   Get user selection
15   CASEWHERE user selection
16     Select Template : selectTemplate
17     Design Template : designTemplate
18     Queue Job : queueJob
19     Submit Template : submitTemplate
20     Approve Template : approveTemplate
21     Approve Job(s) : approveJob
22     Set up Laser Cutter : setupLaser
23     Change Settings : changeSetting
24   END CASEWHERE
25 END SUB
26
27 BEGIN SUB passwordIsStrong(password) — Shourov —
28   strong (boolean) = false
29   IF password contains numbers AND password.length >= 8 THEN

```

⁸Length in bytes

⁹Length in elements

```

30     strong = true
31 END IF
32 RETURN strong
END SUB

34
35 BEGIN SUB validEmail(email) — Shourov —
36     IF email contains characters before a "@" symbol AND ends with "sbhs.nsw.edu.au" THEN
37         RETURN true
38     ELSE
39         RETURN false
40     END IF
41 END SUB

42
43 BEGIN SUB IsTeacherEmail(email) — Shourov —
44     IF first character of email is NOT a number THEN
45         RETURN true
46     ELSE
47         RETURN false
48     END IF
49 END SUB

50
51 BEGIN SUB Login — Max —
52     loggedIn = false
53     email = "" (string)
54     password = "" (string)
55     found = false (boolean)

56
57     get email, password
58     IF email = "" AND password = "" THEN
59         print "Please enter a Username and Password."
60         succeeded = false
61     ELSE
62         index = 1
63         WHILE index <= tblUsers.length AND found <> true
64             IF tblUsers(index).email = email THEN
65                 found = true
66                 IF tblUsers(index).password = password THEN
67                     loggedIn = true
68                     print "login successful"
69                 ELSE
70                     loggedIn = false // wrong password!
71                     print "Wrong password!"
72                 END IF
73             END IF
74             index = index + 1
75         END WHILE
76         IF found <> true THEN
77             print "email not found, please sign up"
78         END IF
79     END IF

80     RETURN loggedIn
81 END SUB

83
84 BEGIN SUB SortTemplates(type) — Shourov —
85     CASEWHERE type
86         'date': SortByDate
87         'material': SortByMaterial
88         'name': SortByName
89         otherwise : print "Invalid type supplied for sort!"
90     END CASE
91 END SUB

92
93 BEGIN SUB SignUp — Max —
94     failed = false (boolean)
95     email = "" (string)
96     password = "" (string)
97     name = "" (string)
98 
```



```

99   Get email, password, name from user input
100  IF email = "" OR password= "" OR name = "" THEN
101      display "Enter an email, password and name"
102      failed = true
103  END IF
104
105  IF failed <> true THEN
106      IF validEmail(email) = true THEN
107          IF IsTeacherEmail(email) = true THEN
108              accesslevel = 1
109          ELSE
110              accesslevel = 0
111          END IF
112      ELSE
113          failed = true
114          display "Invalid email address!"
115      END IF
116  END IF
117
118  IF failed <> true THEN
119      IF passwordIsStrong(password) = true THEN
120          IF CreateUser(email, password, name, accesslevel) = true THEN
121              Login
122          ELSE
123              failed = true
124              print "error creating account"
125          END IF
126      ELSE
127          print "password must be 8 characters or longer, and contain numbers and text"
128          failed = true
129      END IF
130  END IF
131
132  succeed (Boolean) = true
133  IF failed = true THEN
134      succeed = false
135  ELSE
136      RefreshDatabases
137      // Refresh database so new user is added to tblUsers
138  END IF
139
140  RETURN succeed
141 END SUB
142
143
144 BEGIN SUB CreateUser(email, password, name, accesslevel) — Jack —
145     succeeded = true (boolean)
146     IF SearchForUser(email) = false THEN
147         // check that the user is not in the database already
148         Open UsersDatabase for relative access
149         Write email to UsersDatabase.email
150
151         Write password UsersDatabase.password
152         Write name to UsersDatabase.name
153         Write accesslevel to UsersDatabase.accesslevel
154
155         Close UsersDatabase
156         RefreshDatabases
157     ELSE
158         succeeded = false
159         print "email already used!"
160     END IF
161     RETURN succeeded
162 END SUB
163
164
165 BEGIN SUB SearchForUser (email) — Jack —
166     index = 1
167     found (boolean) = false

```



```

168
169 WHILE index <= tblUsers.length AND found <> true
170   IF tblUsers(index).email = email THEN
171     found = true
172   END IF
173   index = index + 1
174 END WHILE
175 RETURN found
176 END SUB
177
178 BEGIN SUB LoadTemplates — Jack —
179   IF loggedIn <> true THEN
180     Login
181   END IF
182
183   IF loggedIn = true THEN //loggedIn will be set to true of Login is successful
184     RefreshDatabases
185   ELSE
186     Print "You must login to access the templates"
187   END IF
188 END SUB
189
190 BEGIN SUB InitializeDatastructures — Jack —
191   TemplateRecord is a record containing
192     Auto_id (integer)
193     DXF (string) // this contains all the line data on the template
194     tags (Array(string)) //list of categories that apply to the temp
195     material (string) // the material the template is to be cut on
196     length (Integer)
197     height (Integer)
198     thickness (Integer)
199     preview (Array(Image)) // preview images of the template. Optional
200     creatorName (string)
201     creator_id (Integer)
202     approverName (string)
203     approver_id (Integer)
204     complete (boolean)
205     dynamicTextLabel (Array(Of String))
206     textLocation (Array(Of Point))
207   END TemplateRecord
208
209   JobRecord is a record containing
210     Auto_id (integer)
211     template (TemplateRecord)
212     submitterName (string)
213     submitterID (Integer)
214     approverName (string)
215     approverID (Integer)
216     approved (boolean)
217     submitDate (date)
218   END JobRecord
219
220   UserRecord is a record containing
221     Auto_id (integer)
222     email (string)
223     password (string)
224     accesslevel (string)
225     name (string)
226   END UserRecord
227 // database tables
228 tblTemplates is an array of TemplateRecord indexed from 1 to EOF
229 tblSubmittedJobs is an array of JobRecord indexed from 1 to EOF
230 tblUsers is an array of UserRecord indexed from 1 to EOF
231 tblUnApprovedTemplates is an array of TemplateRecord indexed from 1 to EOF
232 RefreshDatabases
233
234 UserTags is an array of string indexed from 1 to EOF
235 CurrentUser (UserRecord) = null
236 Tool = "" (string) // currently selected tool
237 loggedIn = false (boolean)

```



```

238   USER = 0 (constant integer)
239   TEACHER = 1 (constant integer)
240   ADMIN = 2 (constant integer)
241   sortBy = "date" (string)
242 END SUB
243
244 BEGIN SUB RefreshDatabases — Jack —
245   Open AllDatabases for sequential reading
246   Get tblTemplates, tblSubmittedJobs, tblUsers, tblUnApprovedTemplates
247   FilterTemplates
248   SortTemplates(sortBy)
249
250   Close AllDatabases
251 END SUB
252
253 // Templates can be filtered according to criteria , stored in the tags
254 // on the template item
255 BEGIN SUB FilterTemplates — Jack —
256   MatchingTemplates is an array of TemplateRecord indexed from 1 to EOF
257   currentTemplate = null (TemplateRecord)
258   index = 1 (integer)
259
260   WHILE index <= tblTemplates.Length
261     currentTemplate = tblTemplates(index)
262     IF TagIsPresent(currentTemplate) = true THEN
263       Append current to MatchingTemplates
264     END IF
265     index = index + 1
266   END WHILE
267   tblTemplates = MatchingTemplates
268 END SUB
269
270 BEGIN SUB TagIsPresent(template) — Jack —
271   // check if one of the tags in UserPreferences
272   prefCount = 1 (integer)
273   tagCount = 1 (integer)
274   found = false (boolean)
275   userTag = "" (string)
276   currentTag = "" (string)
277   // loop through each element in preferences , and check it with each
278   // tag in the template
279   WHILE prefCount <= UserPreferences.length AND found <> true
280     tagCount = 1
281     userTag = UserPreferences(prefCount)
282     WHILE tagCount <= template.tags.length AND found <> true
283       currentTag = template.tags(tagCount)
284       IF currentTag = userTag THEN
285         found = true
286       END IF
287       tagCount = tagCount + 1
288     END WHILE
289   END WHILE
290   RETURN found
291 END SUB
292
293 BEGIN SUB SortByDate — Jack —
294   // sorts tblTemplates by submitted date , from latest to oldest
295   // (descending) how much of the list is sorted , so we don't resort it
296   // as it is already in descending order
297
298   sortedTillElement = 0 (Integer)
299
300   temp = null (TemplateRecord)
301   current = null (TemplateRecord)
302
303   // the location of the largest item in the list
304   largestpos = 1 (integer)
305
306   // what element the sort is at
307   upto = 1 (integer)

```



```

308      // the next element that is unsorted
309      nextUnsorted = 1 (integer)
310
311      // selection sort : will continually find the largest item in the list
312      // and move it to the right position till the entire list is sorted
313      WHILE sortedTillElement <= tblTemplates.length
314          // set upto to the first element that is not sorted yet
315          upto = sortedTillElement + 1
316
317          // assume largest item is on the first element to be tested
318          largestpos = upto
319
320          WHILE upto <= tblTemplates.length
321              current = tblTemplates(upto)
322
323              IF current.date > tblTemplates(largestpos).date THEN
324                  // found newer largest item!
325                  largestpos = upto
326              END IF
327              upto = upto + 1
328          END WHILE
329
330          // largestpos is now the index of the largest element
331          // swap it with the next unsorted item
332          nextUnsorted = sortedTillElement + 1
333
334          temp = tblTemplates(next)
335          tblTemplates(next) = tblTemplates(largest)
336          tblTemplates(largest) = temp
337          sortedTillElement = sortedTillElement + 1
338
339      END WHILE
340  END SUB

```

```

341
342 BEGIN SUB SortByMaterial — Shourov —
343     // sorts tblTemplates by submitted material
344     // (descending) how much of the list is sorted, so we don't resort it
345     // as it is already in descending order
346
347     sortedTillElement = 0 (Integer)
348
349     temp = null (TemplateRecord)
350     current = null (TemplateRecord)
351
352     // the location of the largest item in the list
353     largestpos = 1 (integer)
354
355     // what element the sort is at
356     upto = 1 (integer)
357
358     // the next element that is unsorted
359     nextUnsorted = 1 (integer)
360
361     // selection sort : will continually find the largest item in the list
362     // and move it to the right position till the entire list is sorted
363     WHILE sortedTillElement <= tblTemplates.length
364         // set upto to the first element that is not sorted yet
365         upto = sortedTillElement + 1
366
367         // assume largest item is on the first element to be tested
368         largestpos = upto
369
370         WHILE upto <= tblTemplates.length
371             current = tblTemplates(upto)
372
373             IF current.material > tblTemplates(largestpos).material THEN
374                 // found newer largest item!
375                 largestpos = upto
376             END IF

```

```

377         upto = upto + 1
378     END WHILE
379
380     // largestpos is now the index of the largest element
381     // swap it with the next unsorted item
382     nextUnsorted = sortedTillElement + 1
383
384     temp = tblTemplates(next)
385     tblTemplates(next) = tblTemplates(largest)
386     tblTemplates(largest) = temp
387     sortedTillElement = sortedTillElement + 1
388 END WHILE
389 END SUB
390
391 BEGIN SUB SortByName — Shourov —
392     // sorts tblTemplates by submitted tags
393     // (descending) how much of the list is sorted, so we don't resort it
394     // as it is already in descending order
395     sortedTillElement = 0 (Integer)
396
397     temp = null (TemplateRecord)
398     current = null (TemplateRecord)
399
400     // the location of the largest item in the list
401     largestpos = 1 (integer)
402
403     // what element the sort is at
404     upto = 1 (integer)
405
406     // the next element that is unsorted
407     nextUnsorted = 1 (integer)
408
409     // selection sort : will continually find the largest item in the list
410     // and move it to the right position till the entire list is sorted
411     WHILE sortedTillElement <= tblTemplates.length
412         // set upto to the first element that is not sorted yet
413         upto = sortedTillElement + 1
414
415         // assume largest item is on the first element to be tested
416         largestpos = upto
417
418         WHILE upto <= tblTemplates.length
419             current = tblTemplates(upto)
420
421             IF current.tags(1) > tblTemplates(largestpos).tags(1) THEN
422                 // found newer largest item!
423                 largestpos = upto
424             END IF
425             upto = upto + 1
426         END WHILE
427
428         // largestpos is now the index of the largest element
429         // swap it with the next unsorted item
430         nextUnsorted = sortedTillElement + 1
431
432         temp = tblTemplates(next)
433         tblTemplates(next) = tblTemplates(largest)
434         tblTemplates(largest) = temp
435         sortedTillElement = sortedTillElement + 1
436     END WHILE
437 END SUB
438
439 BEGIN SUB PrintJobDetails(job) — Shourov —
440     Initialize Printer Driver
441     print "Job ID: ", job.id
442     print "Date, time", job.time
443     print "Submitter: ", job.submitter, "Approver:", job.approver
444     print "Dimensions: " job.jobWidth, "x", job.jobHeight
445     Close Printer Driver
446 END SUB

```



```

447
448 BEGIN SUB SelectTemplate — Shourov —
449     LoadTemplates
450     Display tblTemplates on GUI
451     selectedTemplate = null (TemplateRecord)
452     get selectedTemplate
453     Display inputForm(size of selectedTemplate.dynamicTextLabel)
454     dynamicText = empty (array(string))
455     dynamicText = data from inputForm
456     selectedTemplate.dynamicTextLabel = dynamicText
457     Send selectedTemplate to queueJob
458 END SUB
459
460 BEGIN SUB inputForm(rows) — Shourov —
461     Display input table with num Rows
462 END SUB
463
464 BEGIN SUB scrollC — Max —
465     Tool = "Move Screen"
466     //Moves the Field of the designer
467     REPEAT
468         CASEWHERE mouse Moves:
469             Increased x: Design Field moves down by the same amount
470             Decreased x: Design Field moves up by the same amount
471             Increased y: Design Field moves left by the same amount
472             Decreased y: Design Field moves right by the same amount
473         END CASE
474         UNTIL user releases middle click
475 END SUB
476
477 BEGIN SUB rightC — Max —
478     IF user clicks Line THEN
479         Ask user for linetype
480         //(Cut, Raster, Vector, Includes descriptions of each)
481         CASEWHERE user selection is
482             Cut: linesCollection(lineNumber).lines.lineType = "Cut"
483             Vector: linesCollection(lineNumber).lines.lineType = "Vector"
484             Raster: linesCollection(lineNumber).lines.lineType = "Raster"
485         END CASEWHERE
486     END IF
487 END SUB
488
489 BEGIN SUB designTemplate — Max —
490     lineIndex is an integer = 1
491     REPEAT
492         lines is a Record containing:
493             x1 (Integer)
494             y1 (Integer)
495             x2 (Integer)
496             y2 (Integer)
497             lineType (string)
498             color (string)
499             lineNumber (Integer)
500         END lines
501         linesCollection is an Array of lines indexed from 1 to EOF
502         CASEWHERE user selection is
503             Straight line: drawStr
504             Free line: drawFree
505             Edit Logo: editLogo
506             Edit Text: editText
507             New Object: newObj
508             Select Parts: partSelect
509             Line Type: lineType
510             Move: move
511             Left Click: leftC
512             Right Click: rightC
513             Middle Mouse: scrollC
514         END CASEWHERE
515         Tool = ""
516         UNTIL Done = True

```



```

517 genImg
518 END SUB
519
520
521 BEGIN SUB moveScreen — Max —
522     Tool = "Move"
523     //Moves the Field of the designer
524     IF left click is pressed THEN
525         REPEAT
526             CASEWHERE mouse Moves:
527                 Increased x: Design Field moves down by the same amount
528                 Decreased x: Design Field moves up by the same amount
529                 Increased y: Design Field moves left by the same amount
530                 Decreased y: Design Field moves right by the same amount
531         END CASEWHERE
532         UNTIL user releases left click
533     END IF
534 END SUB
535
536 BEGIN SUB partSelect — Max —
537     Tool = "Select Part"
538     leftC
539 END SUB
540
541 BEGIN SUB leftC — Max —
542     Highlight ends of clicked line
543     WHILE mouse is hovering over line
544         Display co-ordinates of Line ends next to mouse
545     END WHILE
546     IF left click is pressed AND a line is selected THEN
547         // Moves the line
548         REPEAT
549             CASEWHERE mouse Moves:
550                 Increased x: Line moves up by the same amount
551                 Decreased x: Line moves down by the same amount
552                 Increased y: Line moves right by the same amount
553                 Decreased y: Line moves left by the same amount
554         END CASEWHERE
555         UNTIL Left click is released
556     END IF
557
558     IF left click is pressed AND a logo is selected THEN
559         // move the logo
560         REPEAT
561             CASEWHERE mouse Moves:
562                 Increased x: Logo moves up by the same amount
563                 Decreased x: Logo moves down by the same amount
564                 Increased y: Logo moves right by the same amount
565                 Decreased y: Logo moves left by the same amount
566         END CASEWHERE
567         UNTIL Left click is released
568     END IF
569
570     IF left click is pressed AND a TextBox is selected THEN
571         // move the logo
572         REPEAT
573             CASEWHERE mouse Moves:
574                 Increased x: TextBox moves up by the same amount
575                 Decreased x: TextBox moves down by the same amount
576                 Increased y: TextBox moves right by the same amount
577                 Decreased y: TextBox moves left by the same amount
578         END CASEWHERE
579         UNTIL Left click is released
580     END IF
581 END SUB
582
583 BEGIN SUB lineType — Max —
584     Tool = "Line Type"
585     IF user left Clicks on a Line Segment THEN
586         CASEWHERE user selection is

```

L.A.M.P

Waxy LASER Solutions

Max Wharton-Jones, Shourov Quazi, Jack Jiang



```
587     Cut: linesCollection(lineNumber).lines.lineType = "Cut"
588     Vector: linesCollection(lineNumber).lines.lineType = "Vector"
589     Raster: linesCollection(lineNumber).lines.lineType = "Raster"
590 END CASEWHERE
591 END IF
592 END SUB
593
594 BEGIN SUB newObj — Max —
595     Get user input
596     // Logo, text Box, Line
597     CASEWHERE User selection is
598         Logo: Create a logo in Design Field with center at Mouse coordinates
599         Text Box: Create a Text Box in Design Field with center at mouse coordinates
600         Line: drawStr
601     END CASEWHERE
602 END SUB
603
604 BEGIN SUB editText — Max —
605     IF User left clicks Text box THEN
606         Get user input (String)
607     END IF
608     Textbox.Text = user input
609 END SUB
610
611 BEGIN SUB editLogo — Max —
612     REPEAT
613         Access = "Denied" (String)
614         Ask user for a logo
615         Prompt File viewer
616         IF the File is a PNG OR the file is a JPEG OR the File is a JPG THEN
617             File = "Approved"
618         ELSE
619             Deny
620         END IF
621         UNTIL File = "Approved"
622 END SUB
623
624 BEGIN SUB drawFree — Max —
625     Ask user for linetype
626     //((Cut, Raster, Vector, Includes descriptions of each))
627     Tool = "Free Line"
628     freeTimer = 0 (integer)
629     IF user clicks Left Click THEN
630         REPEAT
631             linesCollection(lineIndex).lines.x1 = mouse position x in Design Field
632             linesCollection(lineIndex).lines.y1 = mouse position y in Design Field
633             freeTimer = freeTimer + 1
634             Wait 0.01 Seconds
635             linesCollection(lineIndex).lines.x2 = mouse position x in Design Field
636             linesCollection(lineIndex).lines.y2 = mouse position y in Design Field
637             lineIndex = lineIndex + 1
638         UNTIL Left Click is released
639     END IF
640 END SUB
641
642 BEGIN SUB drawStr — Max —
643     Ask user for linetype
644     //((Cut, Raster, Vector, Includes descriptions of each))
645     Tool = "straightLine"
646     IF User clicks left mouse AND Tool = straightLine THEN
647         linesCollection(lineIndex).lines.x1 = mouse position x in Design Field
648         linesCollection(lineIndex).lines.y1 = mouse position y in Design Field
649         IF user clicks left mouse THEN
650             linesCollection(lineIndex).lines.x2 = mouse position x in Design Field
651             linesCollection(lineIndex).lines.y2 = mouse position y in Design Field
652         ELSE
653             linesCollection(lineIndex).lines.x1
654             linesCollection(lineIndex).lines.y1
655             lineIndex = lineIndex - 1
656     END IF
```

```

657     END IF
658     lineIndex = lineIndex + 1
659 END SUB
660
661 BEGIN SUB genIMG — Max —
662     Prompt File viewer
663     Load Ai File
664     Take image from Ai Logo
665     Use image as icon for file in file viewer
666 END SUB
667
668 BEGIN SUB checkValidSPF( file ) — Shourov —
669     Open file for sequential reading
670     lineData = null (Array of line , from 0 to EOF)
671     get lineData from file
672
673     valid = true
674     index = 1
675     WHILE lineData.Length <= index AND valid <> false
676         IF lineData(index).color <> "RED" AND lineData(index).color <> "BLUE" AND lineData(index).color
677             valid = false
678             print "Lines must be red , black or blue"
679         END IF
680         index = index + 1
681     END WHILE
682     close file
683     RETURN valid
684 END SUB
685
686 BEGIN SUB queueJob — Jack —
687     IF loggedIn = false THEN
688         Login
689     END IF
690     currentDate = null (date)
691     get currentDate
692     IF loggedIn = true AND CurrentUser.accesslevel >= TEACHER THEN
693         // teacher/admin , and logged in
694         Get SPF file
695         IF file.complete = true AND checkValidSPF( file ) = true THEN
696             // send it
697             Open JobsDatabase for relative access
698             write file to JobsDatabase.template
699             write CurrentUser.name to JobsDatabase.submitterName
700             write CurrentUser.Auto_ID to JobsDatabase.submitterID
701             write "" to JobsDatabase.approverName
702             write -1 to JobsDatabase.approverID
703             write false to JobsDatabase.approved
704             write currentDate to JobsDatabase.submitDate
705             Close JobsDatabase
706             RefreshDatabases
707         ELSE
708             print "File is incomplete , or contains errors"
709         END IF
710     ELSE
711         print "Only teachers/administrators can submit jobs"
712     END IF
713 END SUB
714
715 BEGIN SUB submitTemplate — Jack —
716     LoadTemplates\\
717     templatefile = null (SPF file)
718     get templateFile
719     IF loggedIn <> true THEN
720         Login
721     END IF
722     IF loggedIn = true AND CurrentUser.accesslevel >= TEACHER THEN
723         Open unapprovedTemplateDatabase
724         write templateFile to unapprovedTemplateDatabase
725         Close unapprovedTemplateDatabase
726     ELSE

```



```

727     print "please login to submit templates"
728 END IF
729 END SUB
730
731 BEGIN SUB approveJob — Jack —
732     IF loggedIn = false THEN
733         Login
734     END IF
735
736     selectedJob = null (JobRecord)
737     confirmation = false (boolean)
738
739     IF loggedIn = true AND CurrentUser.accesslevel >= ADMIN THEN
740         // admin, and logged in
741         display tblSubmittedJobs
742         get selectedJob
743         print "Approve job?"
744         get confirmation
745         IF confirmation = true THEN
746             print "Do you want to print job details to paper?"
747             get confirmation
748             IF confirmation = true THEN
749                 PrintJobDetails(selectedJob)
750             END IF
751             selectedJob.approverName = CurrentUser.name
752             selectedJob.approverID = CurrentUser.Auto_ID
753             selectedJob.approved = true
754         ELSE
755             print "Cancelled"
756         END IF
757     ELSE
758         print "Only administrators can approve jobs"
759     END IF
760 END SUB
761
762 BEGIN SUB setupLaser — Shourov —
763     Initialize Laser Driver
764     Process SPF file as DXF file in lasercutter printer
765     Close Laser Driver
766 END SUB
767
768 BEGIN SUB ChangeSort(criteria) — Shourov —
769     CASEWHERE criteria
770         date: sortBy = "date"
771         material: sortBy = "material"
772         name: sortBy = "name"
773         otherwise: print "Invalid criteria"
774     END CASE
775 END SUB
776
777 BEGIN SUB changeSettings — Shourov —
778     CASEWHERE userinput
779         add category: AddTag(tag)
780         remove category: RemoveTag(tag)
781         change sort: ChangeSort(criteria)
782         otherwise: print "invalid input"
783     END CASE
784 END SUB
785
786
787 BEGIN SUB approveTemplate — Jack —
788     IF loggedIn = false
789         Login
790     END IF
791     selectedTemplate = null (JobRecord)
792     confirmation = false (boolean)
793
794     IF loggedIn = true AND CurrentUser.accesslevel >= ADMIN THEN
795         // admin, and logged in

```



```

796     display tblUnapprovedTemplates
797     get selectedTemplate
798
799     print "Approve template?"
800     get confirmation
801     IF confirmation = true THEN
802         delete selectedTemplate from tblUnapprovedTemplates
803         append selectedTemplate to tblTemplates
804     ELSE
805         print " Cancelling"
806     END IF
807     ELSE
808         print " Only administrators can approve templates"
809     END IF
810 END SUB
811
812 BEGIN SUB AddTag(tag) — Shourov —
813     // add a category to the user settings
814     UserTags(UserTags.length+1) = tag
815 END SUB
816
817 BEGIN SUB RemoveTag(tag) — Shourov —
818     // remove a category to the user settings
819     index = 1
820     found = false
821     WHILE index <= UserTags.length AND found <> true THEN
822         IF UserTags(count) = tag THEN
823             delete index field from UserTags
824             found = true
825         END IF
826         index = index + 1
827     END WHILE
828     IF found <> true THEN
829         print " Cannot find tag"
830     END IF
831 END SUB

```

2.7 Screen Design Principals¹⁰

2.7.1 Philosophy

Functionality was the main focus of this program, and thus sacrifices were made in terms of the user interface, to accommodate a high number of functions and options per screen.

Aesthetics and Clarity

- + Ensuring information is presented in an organised and predictable manner ←-- using groupings and clear alignments
- + As much content on each screen without too much clutter, organised with group boxes
- + Easy to read fonts and non-intrusive colour scheme using neutral or cool colours
- + Simple colour palettes
- + No unnecessarily colourful or large icons in menus

¹⁰Screen Design by Max

2.7.2 Storyboard

Screen Design Storyboard

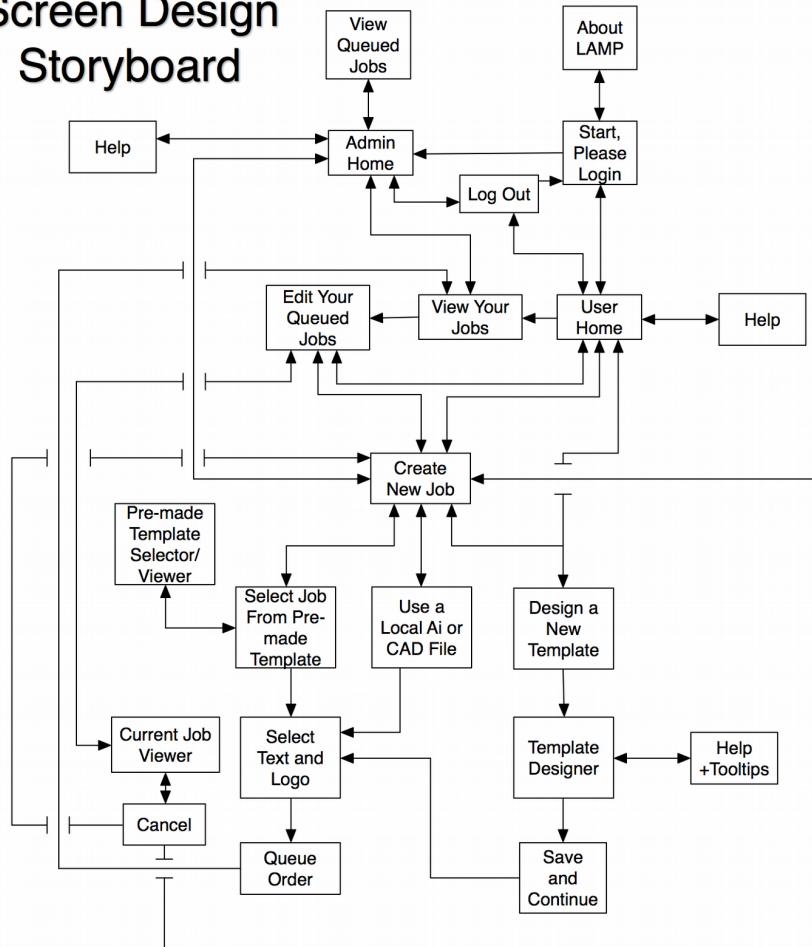


Figure 2.1: The storyboard, showing connected screens



2.7.3 Login Screen

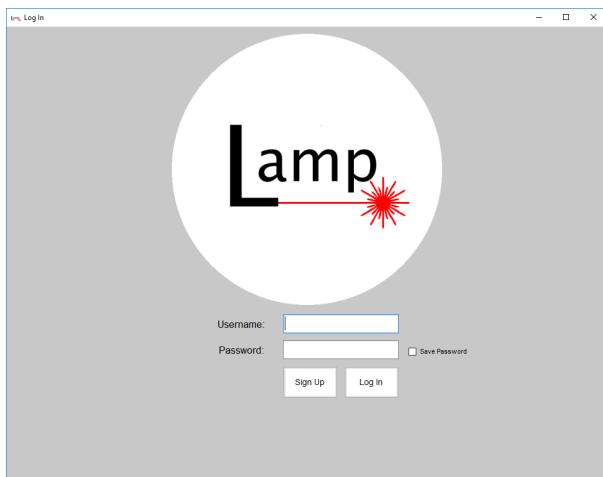


Figure 2.2: The Login Screen was created with simplicity in mind, and so the only things on the screen upon the start-up are the logo and the textboxes that you login with. The form uses a light grey background colour, which is consistent throughout all the forms of the program.

2.7.4 Sign Up]

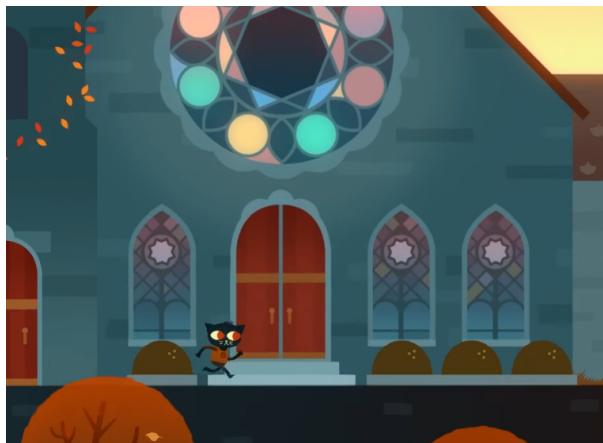


Figure 2.3: This screen contains only text boxes and labels to guide the user as to how to sign up and create an account.

2.7.5 Home Screen

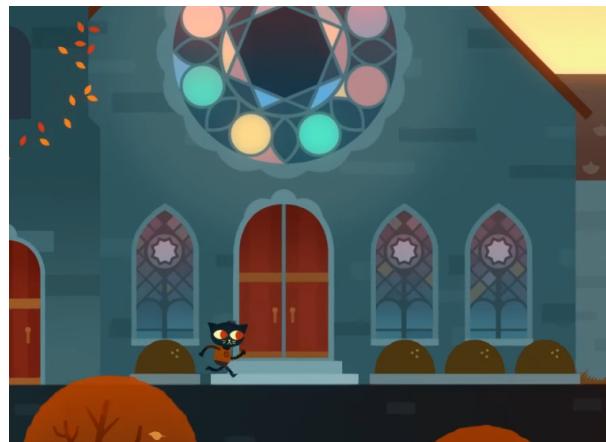


Figure 2.4: This screen has a toolbar which is also consistent throughout the program. It also contains links to the other forms

2.7.6 Create New Design

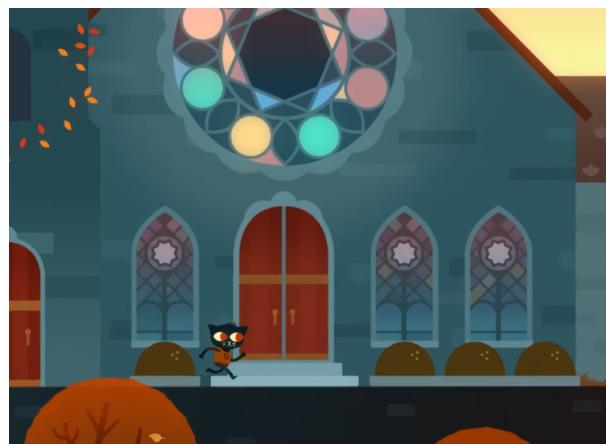


Figure 2.5

2.7.7 Home Screen

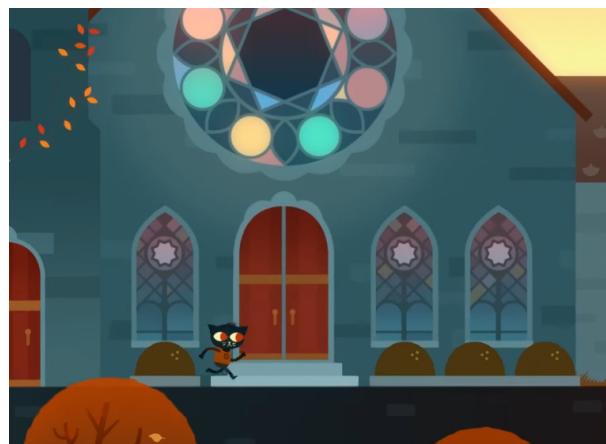


Figure 2.6: This screen has a toolbar which is also consistent throughout the program. It also contains links to the other forms

2.7.8 Create New Design

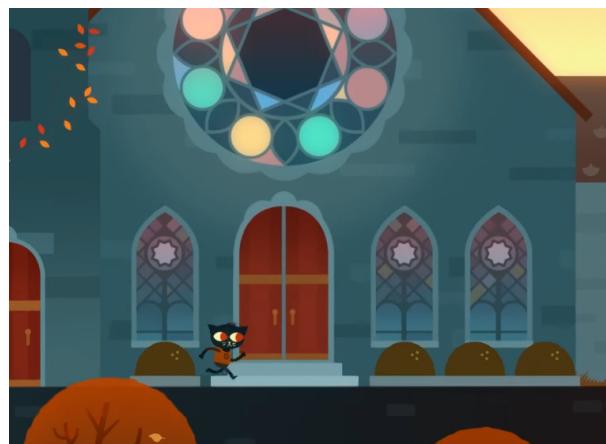


Figure 2.7

2.7.9 Create New Job

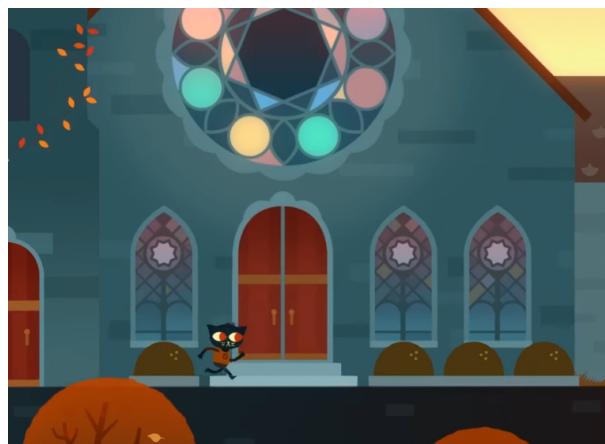


Figure 2.8

2.7.10 View Jobs

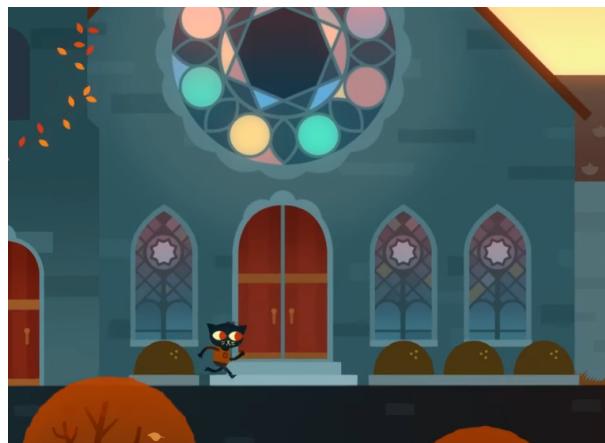


Figure 2.9

2.7.11 View Designs

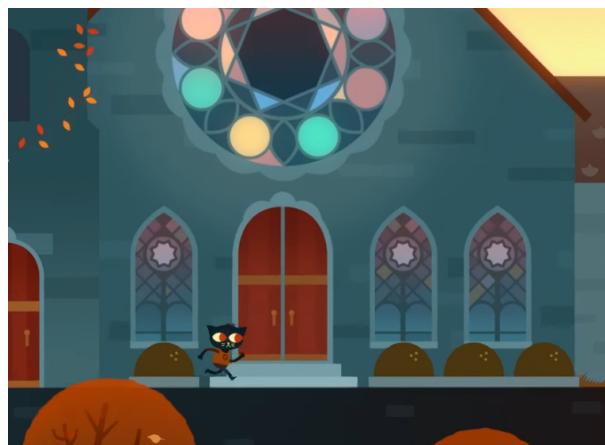


Figure 2.10

2.7.12 Admin Tools

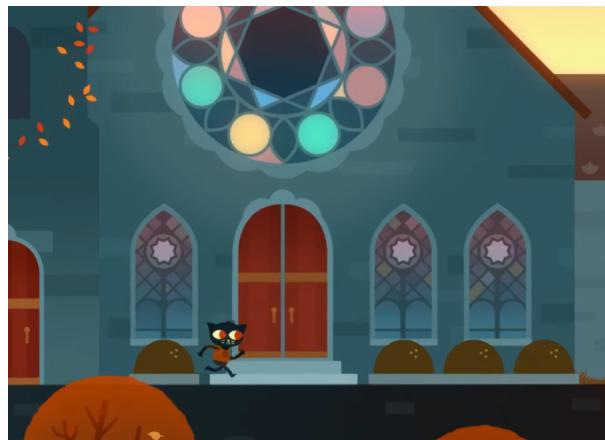


Figure 2.11

2.7.13 Approve/Reject Templates

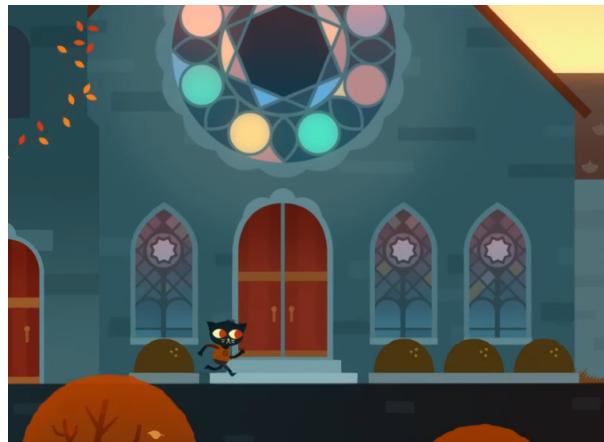


Figure 2.12

2.7.14 Dialogs

Help

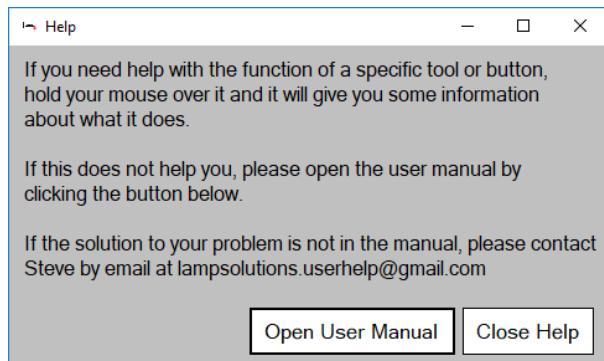


Figure 2.13: The help dialog informs the user how to use tooltips and provides a link to the user manual

L.A.M.P

Waxy LASER Solutions

Max Wharton-Jones, Shourov Quazi, Jack Jiang



About

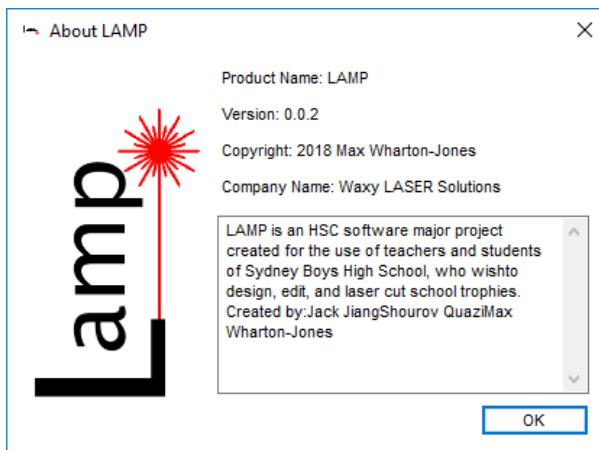


Figure 2.14: The about dialog provides information about the program along with relevant copyright and licensing terms