A webtool for e-textiles design

Roope Lehikoinen

School of Electrical Engineering

Bachelor's thesis
Espoo 15.12.2019

Supervisor

Prof. Samuli Aalto

Advisor

Asst. Yu Xiao

Aalto University
School of Electrical
Engineering

Aalto University
School of Electrical
Engineering

Aalto University, P.O. BOX 11000, 00076 AALTO
www.aalto.fi
Abstract of the bachelor's thesis

Author Roope Lehikoinen

Title A webtool for e-textiles design

Degree programme Automation and information technology

Major Information Technology **Code of major** ELEC3015

Teacher in charge Prof. Samuli Aalto

Advisor Asst. Yu Xiao

Date 15.12.2019 **Number of pages** 18+3 **Language** English

Abstract

The goal of this thesis is the development of a web-based tool for e-textile design following a user-centered design process and methods. Achieving this goal could ease the work of designers and allow them to focus on more important parts of designing. It could also ease the work of new designers and amateurs. The first research topic of this thesis is designing an application using user-centered design process and methods. What are user-centered design methods and how end-users are a part of the design process. The design should have enough complexity to help in more difficult tasks, but still be intuitive and user-friendly. The second topic is developing a web application. What kinds of tools could be used and how to achieve the desired design.

The application was developed in three parts. A rudimentary first version with basic features to build upon. The second version, which had more features and was functional as an e-textile design tool. Lastly, a user-test was held in the form of a workshop, where end-users could try the application and give feedback on it. Most testers found that being able to view a simulated three-dimensional view of the fabric was useful. They also found that a lot of features were left missing, such as basic editing tools and a library of predetermined woven structures. Most testers expressed that the application could be useful especially for learners and people not familiar with woven structures.

Keywords E-textiles, smart textiles, weaving, multilayered , web application, user-centered design

Aalto University
School of Electrical Engineering

Aalto-yliopisto, PL 11000, 00076 AALTO
www.aalto.fi
Tekniikan kandidaatintyön tiivistelmä

Tekijä Roope Lehikoinen

Työn nimi A webtool for e-textiles design

Koulutusohjelma Automaatio- ja informaatioteknologia

Pääaine Informaatioteknologia **Pääaineen koodi** ELEC3015

Vastuuopettaja Prof. Samuli Aalto

Työn ohjaaja Asst. Yu Xiao

Päivämäärä 15.12.2019 **Sivumäärä** 18+3 **Kieli** Englanti

Tiivistelmä

E-tekstiilejä on ollut olemassa jo kaksi vuosisataa, ja ne ovat edelleen kasvava tieteen- ja teollisuudenala. E-tekstiilit ovat kankaita, joiden sisään on kudottu sähköä johtavia materiaaleja. Jo 1800-luvun loppupuolella on käytetty vaatteita, joissa on johtavia materiaaleja, kuten kultaa tai hopeaa. Vasta myöhemmin näitä materiaaleja on alettu käyttää hyväksi muun muassa asentamalla kankaisiin erilaisia sensoreita tai valoja. E-tekstiilien suunnittelumenetelmät ovat kuitenkin jääneet jälkeen. Nykyiset suunnittelumenetelmät koostuvat paperille piirretyistä malleista, jotka siirretään tietokoneelle kuvankäsittelyohjelman avulla. Tästä ohjelmasta kuva siirretään kangaspuita ohjaavaan sovellukseen, josta kangas lopulta kudotaan. Tämä suunnittelutapa on hidas ja altis pienvirheille.

Työ toteutettiin kolmessa vaiheessa. Ensimmäinen vaihe oli yksinkertainen versio sovelluksesta, jonka tarkoitus oli olla pohja, johon voidaan lisätä toimintoja. Toinen vaihe sisälsi enemmän toimintoja ja sen tarkoitus oli olla valmis applikaatio, jolla käyttäjä kykenee suunnittelemaan ja kutomaan kankaita. Viimeinen vaihe oli toteuttaa käyttäjäkokeilu Aalto Yliopiston tekstiilien suunnittelu opiskelijoilla.

Työkalu toteutettiin tässä työssä verkkopohjaisena applikaationa. Applikaation yläreunassa sijaitsee valikko, josta ohjelma tarjoaa käyttäjälle useimmat toiminnot. Tämä valikon tyyli on standardisoitunut tietokoneohjelmien tyylissä ja on useimmille käyttäjille valmiiksi tuttu ja intuitiivinen. Ohjelman vasemmassa reunassa on työkaluvalikko, jonka vaihtoehdot muokkaavat kuinka käyttäjä vuorovaikuttaa ohjelman kanssa. Esimerkiksi liikutustyökalu valittuna käyttäjä voi liikuttaa applikaation näkymää, mutta ei kykene muokkaamaan projektia. Työkaluvalikko on nykyään laajasti käytössä työkaluohjelmissa, erityisesti kuvanmuokkausohjelmissa, kuten esimerkiksi Photoshopissa tai Gimpissä. Applikaation projektinäkymä, eli osuus ohjelman keskellä, jota käyttäjä muokkaa, koostuu oletuksena kahdesta eri näkymästä. Ensimmäinen näkymä on niin sanottu pikselinäkymä, jossa käyttäjä vaihtaa pikseleitä päälle ja pois, eli mustiksi tai valkoisiksi pikseleiksi. Muokkaukset pikselinäkymässä vaikuttavat toiseen näkymään eli kolmiulotteiseen näkymään. Tämä näkymä on kolmiulotteinen simulaatio siitä miltä lopullinen kudottu kangas näyttää.

Työssä toteutettu käyttäjäkokeilu tehtiin työpajan muodossa, jossa testaajat saivat kokeilla sovellusta ja antaa palautetta siitä. Työpajan lopuksi pidettiin vapaamuotoinen keskustelu sovelluksesta. Useimmat testaajat olivat sitä mieltä, että kolmiulotteinen kankaan simulointi on hyödyllistä. Useiden testaajien mielestä sovelluksesta jäi jotain uupumaan. Puuttuvia ominaisuuksia olivat muun muassa yksinkertaiset editointi työkalut, kuten kopioi, leikkaa ja liimaa. Näiden lisäksi useiden testaajien mielestä sovelluksessa olisi voinut olla kirjasto valmiiksi määriteltyjä kudos rakenteita. Useiden testaajien mielestä sovellus olisi hyödyllinen erityisesti aloittelijoille, jotka oppivat visuaalisesti tai jotka eivät vielä tunne kudos rakenteita.

Avainsanat E-tekstiili, älytekstiili, kudonta, web-sovellus, tietokanta, tietokoneohjelma

Contents

Symbols and abbreviations

Abbreviations

2D	Two-dimensional
3D	Three-dimensional
API	Application programming interface
CSS	Cascading Style Sheets
GPU	Graphics processing unit
HTML	Hypertext markup language
TC2	Thread controller 2
TIFF	Tagged image file format
UCD	User-centered design
Webapp	Web-based application
WebGL	Web graphics library

1 Introduction

Electronic textiles (e-textiles) have been around for two centuries and are still a growing industry. They are wearable pieces of clothing that have conductive materials woven into the fabric. Since the late 1800s people have worn clothing with conductive materials, such as gold or silver. Only later have designers started using the conductivity to create circuitry out of clothing [1]. Most of the early uses of e-textiles involved integrating some forms of lights into clothing and jewelry.

E-textiles can be created in different ways such as weaving, knitting and crochet. This thesis focuses on the aspect of weaving. It can be argued that the methods for designing woven e-textiles have stagnated. Current methods involve paper sketches, an example of which is shown in figure 1, complemented by computer-controlled weaving machines. The machines receive instructions from images created in Photoshop or in a similar software from the paper sketches. The image is then converted into a file readable by the software of the weaving machine, after which it is entered into said software. Simple revisions and changes to the design require re-editing of the image, which is time consuming, unintuitive, and prone to small errors.

The goal of this thesis is the development of a web-based tool for e-textile design following a user-centered design process and methods. Achieving this goal could ease the work of designers and allow them to focus on the more important parts of designing. It could also ease the work of new designers and amateurs. The first research topic of this thesis is designing an application using user-centered design process and methods. What are user-centered design methods and how the end-users are a part of the design process. The design should have enough complexity to help in more difficult tasks, but still be intuitive and user-friendly. The second topic is developing a web application. What kinds of tools could be used and how to achieve the desired design.

The research adheres to the following restrictions. It includes the basics of weaving and textile design as they are important building blocks for the fundamental features of the application. For example, how the pixel-based textile designing works and how it is translated into the finished fabric. However, extensive research of specific weaving techniques is not included as these are unimportant to the basic features of the application. The application has the basic tools to create weaves but does not expand the user's knowledge of weaving techniques.

The application was built step by step, and the design process and the implementation are examined in sections 3 and 4 respectively. The first version is a rudimentary template, the objective of which is to be a foundation, onto which the tool can be expanded. The second version adds more functionality to the application and allows it to function as a design tool. After the second version was complete a user-test was held in the form of a workshop, where end-users were able to try the application and give feedback on it. The details and findings of the workshop are looked at in section 5.

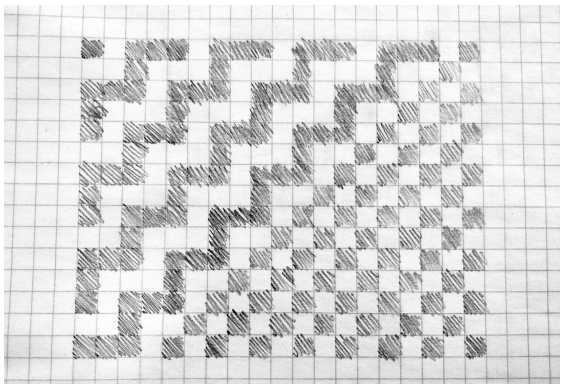

Figure 1: An example of a hand-drawn weave design.

2 Background

The following sections introduce important background information to the subject of weaving and web applications. First section 2.1 introduces the basics of weaving, onto which, section 2.2 expands upon by introducing multilayered weaving. After this section 2.3 introduces the Web graphics library and how it is used in the implementation of the application. Lastly, some previous research of textile designing applications is looked at in section 2.4.

2.1 Weaving

Fabrics are woven by interlacing two different yarns, called the warp and the weft, with each other at 90-degree angles as is illustrated in figure 2 [2–4]. Warps represent the vertical yarns and weaves the horizontal yarns. Weaving machines (looms) hold the warps inside small crevices called heddles, that can be lifted or lowered. Older wooden looms, or shaft looms, have varying amounts of heddles and often require multiple warps to be threaded through the same heddle, which reduces the available complexity of the fabric. Modern power looms have up to 10000 heddles, which allows for greater complexity.

The weaving process can be divided into three parts that are repeated in succession [3, 4]. First is shedding, where some warps are raised, and some are lowered. The lifted warps are determined by the person weaving or by a file submitted to a software in modern power looms. After this is picking, where the weft is carried through in between the warps that are lifted and the ones that are down. Lastly is beating-up, where the weft is pushed back against the fabric to make is tight. The pattern that the interlaced warp and weft create is called a weave. The simplest weave, a plain weave, is achieved when for every row, every other warp is lifted. This creates a sort of chess pattern.

In 1804 Joseph Marie Jacquard invented the Jacquard machine, which can be fitted on to a power loom to simplify the process of weaving [5]. Unlike the older shaft looms where multiple warps were threaded through the same heddles, the Jacquard machine has a heddle for each warp. The first Jacquard machines were controlled by

cards with holes punched into them. Each row on the cards was a row in the fabric and the holes marked, which warps should be lifted for that row.

Modern Jacquard power looms are controlled by programs that read tagged image file format (TIFF) bitmaps with black and white pixels. The files closely mimic the hole punched cards of older Jacquard looms. The pixels tell the weaving machine row by row, which warps should be lifted. If a pixel is black, the warp should be lifted and vice versa. These files can be created by photoshop or by industry software [6]. An example of the resulting TIFF files is shown in figure 3.

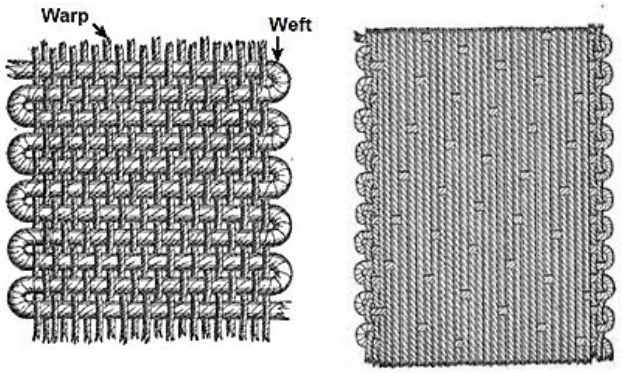

Figure 2: An illustration of basic method of weaving with the warp and the weft being interlaced with each other.

2.2 Multilayered weaving and applications in e-textiles

The basis of multilayered weaving is that the warps and wefts are divided into layers that stay separate from each other unless overlap is needed [7]. For example, if a plain weave as a single layer fabric is converted into two layers, as can be seen in figure 3, every other warp is assigned to the top fabric and every other to the bottom fabric. Then the points are looked at, where the top warps meet the top wefts. In the case of a two layered fabric this includes four points. The single layered plain weave is then translated onto these four points. After this, the pattern is translated onto the points where the bottom warps meet the bottom wefts. In the case of a plain weave, the result is a diagonal line from the bottom left corner to the top right corner.

While weaving, the points of the different layers must be kept separate unless overlap is desired [7]. This means that when the loom is working on a top layer, the warps of the bottom layer must be kept down in order to separate them. The same is true for the bottom layer, the warps of the top layers must be lifted. This means black pixels must be added to the points where the top warps meet the bottom wefts. The result is the weave seen in figure 3. In this example a 2x2 weave becomes a 4x4 weave. The same method can also be used for three layers and above. Complexity to the weaving arises when the weave employs varying amounts of layers in different parts of the fabric. Designing this type of weave with pen and paper requires unreasonable amounts of resources.

E-textiles often employ multilayered weaving to create complexity and protection to the conductive threads. Instead of having conductive materials on the outside of the fabric, the materials can be woven in between the layers [8]. This protects the conductors form outside forces. For example, it is not desirable to have a jacket with conductors on the outside exposed to rain and other weather phenomenon. Multilayered weaving also offers designers multiple sheets of circuitry. In other words it is possible to weave different electric circuits on the same piece of fabric, which increases the amount of possible implementations of e-textiles.

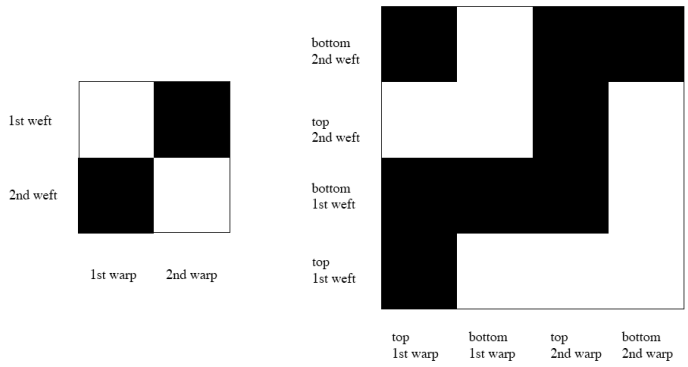

Figure 3: A single layered plain weave is converted into two layers. A black pixel in the figure signifies that a warp runs over a weft. Likewise a white pixel signifies that a warp runs under a weft.

2.3 Web graphics library and three.js

Web graphics library (WebGL) is an application programming interface (API) for JavaScript [9, 10]. It allows for the rendering of two- and three-dimensional (2D and 3D) objects inside a web browser. It was released in 2011 and was soon integrated into Chrome and other web browsers natively [11]. WebGL was built off of the Open Graphics Library for Embedded Systems (OpenGL ES), which is a 3D rendering software [9, 10]. Embedded systems refers to the fact that it was designed for embedded systems, such as smartphones and tablets.

WebGL is the browser-based version of OpenGL ES and, as such, allows for hardware-accelerated graphics [9, 10]. This means that WebGL employs the graphics processing unit (GPU) of the computer for graphical calculation. This allows the removal of dependency on a server's calculation capacity as most of the calculation is accomplished on the client computer. Overall, it grants graphics calculations of greater intensity inside a web browser. WebGL rendering is controlled by JavaScript, and shading is controlled by OpenGL shading language (GLSL) [9, 10]. GLSL is what allows WebGL to employ the GPU of the client computer.

On top of WebGL, the application described in this thesis uses a library called three.js. It is an open source library that provides high level, developer friendly, functions for WebGL [9, 10, 12, 13]. WebGL itself is quite a low-level programming library, but with three.js providing higher level functions, programming is much

more intuitive. Most functions of three.js are implemented for WebGL 1.0, which is why the application uses it instead of WebGL 2.0. WebGL 1.0 is also more widely available on web browsers.

The application uses WebGL with three.js as the base of simulating a fabric. WebGL can be used to create 3D simulations that can also be interacted with, allowing for a good enough base for the application. This is expanded upon in sections 3 and 4.

2.4 Previous research

In 2019 Mikhaila Friske et al. [14] released a paper on designing and developing an application called AdaCAD. AdaCAD addressed many of the problems in e-textile design by creating a user-friendly application with many features tailored for e-textile design. Friske et al. did not however, find simulating the finished fabric instructive enough. The result of weaving is largely based on the user, their technique, and the materials used. It is difficult for the simulation of the finished fabric to take these into consideration. A focus point of the application developed in this thesis is 3D simulation of the finished fabric. Despite the difficulty to represent real fabrics, 3D simulation is an excellent tool for visualising the results of different weaves. It is also useful for catching mistakes in the fabric before weaving, which cuts down wasted material and time used weaving.

Other applications for textile design exist. These include WeavePoint, pixeLoom, Arahweave, Fiberworks PCW, Weavelt and many others. Most of these have the functionality to do very complicated weaving but often lack in intuitiveness. In addition, none of these applications have the ability to simulate a 3D model of the resulting fabric.

3 Designing the application

The research followed a user-centered design (UCD) process and was conducted in three parts, which can be seen in figure 4. First, a basic version that supported single layer weaving and had user input functionality was created. Second, features were added to the first version. Lastly, the research included a user test on textile design students in Aalto University via a workshop.

UCD is introduced in section 3.1. After that, the usefulness and features of web applications are introduced in section 3.2. Lastly, desirable features of the application developed in this thesis are introduced in section 3.3.

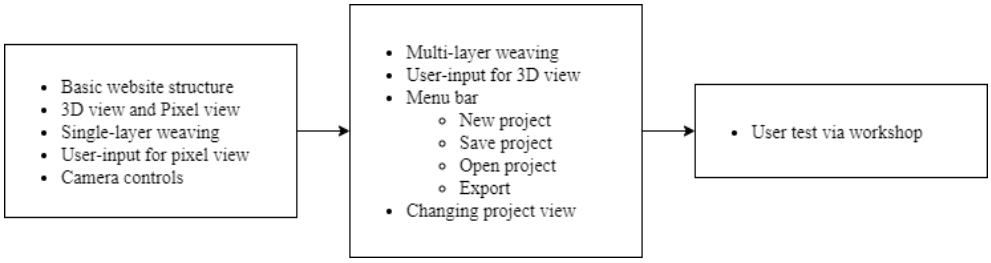

Figure 4: The iterative design process of the application.

3.1 User-centered design (UCD) process

User-centered design is a method of product design where end-users have a part to play [15–17]. The extent to which the end-users participate, varies greatly from product to product. For example, one process of UCD may ask the users what features they would like to have. Another process may have the users participate in the development and thus have a greater impact on the product. By involving the end-user in the design process, the focus of the design choices inherently changes. The design choices start following a user-friendly pattern, which includes at least four features [15, 16]. The design should make it easy to determine all possible actions at any given time, make possible actions visible and make it easy to determine the state of the system. Lastly, it should make clear connections between intentions and required actions, actions and effects, and visible information and the state of the system. For example, application should always give some visual indication of what the user is doing. Without this, the application is hard to read and confusing to work with.

The application developed in this thesis uses a kind of UCD process where most of the features presented are suggested by end-users. In discussions with Emmi Pouta, a researcher at the Department of Communications and Networking (Comnet) and a potential end-user of the application, the basic features for the application were agreed upon. These features included viewing of the woven structure in a 3D view and a pixel-based view, a library of predetermined woven structures, and easy conversions between fabrics of different layers. After the second version of the application was completed, a workshop was held for students of textile design. It

allowed the students to try the application and give feedback on it via a questionnaire. The Questionnaire and the answers are included in appendix A and more about the workshop and the answers are reviewed in section 5.

3.2 Web applications

The tool described in this thesis is a web-based application (web app). The definition of a web application is relatively unclear, but in this thesis web applications refer to dynamic web pages. Web apps have almost identical functionality to standalone applications. However, using web apps over standalone applications has pros and cons. Web apps can be run on any computer with internet access and a modern browser. Also, as presented in section 2.3, when using WebGL for 3D graphics, a burden is removed from the server side. WebGL however, also adds requirements for the client computer. Firstly, a compatible browser, which when using WebGL 1.0 should not be an issue when using updated versions of most modern browsers. Secondly, a compatible GPU is required. Requiring internet access can be viewed as a complication as well. While it allows for sharing of files and dynamic updates, a lack of it literally renders the application useless.

Saving files in web apps is more difficult than standalone applications for security reasons. This leaves three options for saving project-files and exporting TIF-files. First option is to create a database for data storage. This option is better for the end-user, as they do not have to save files on their own computers, but it places more requirements on the host server. The other option is to have the user download the project-files and the export-files. The last option is web storage.

Web storage is an API most modern web-browsers have, that allows programs to store data on the browser [18]. This feature supersedes cookies that are used to store user-data on the user's computer. The API has two different kinds of storage, session storage, and local storage. Session storage keeps saved objects for the lifetime of a tab, which is not useful in the case of backups. Local storage, however, stores up to 10MB of data and is not deleted even when the browser is exited. This means it would be an ideal method for backing up project data.

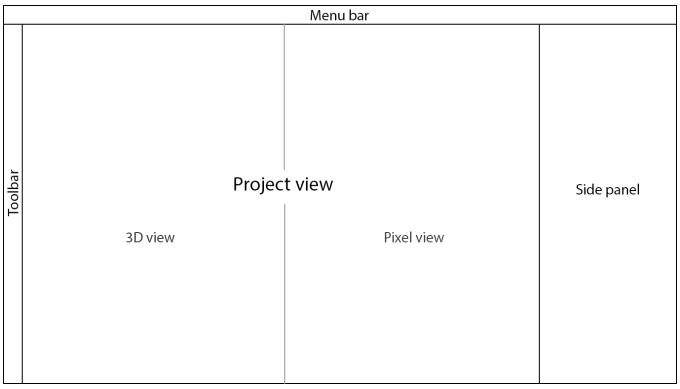

Figure 5: An illustration of the placement of elements in the application

3.3 Features

Most modern image editing applications follow an almost standardized style. The style comprises of a multitude of elements, including a menu bar, which holds most of the functionality of the application. The menu bar is located at the top of the application as shown in figure 5. The options presented to the user in the menu bar include saving and exporting files, creating new projects and changing how the project is viewed. Also presented are editing tools i.e. undo, redo, copy, cut, paste. The editing tools are a vital part of editing software. Undo reverts the project to the state before the latest change and redo reverts the actions of undo. A solution to undo and redo for this application could be a stack where the state of the project or the pixels changed are stored when undoing. Redo then recalls the item from the top of the stack. If the user edits the project after undoing the stack is reset and the user is no longer able to redo the actions before the previous undo.

The copy, cut and paste functions require a way to select items in the application. This usually means highlighting a part of the project. After highlighting the user is able to use the functions on the selection. Copy and cut are similar actions, where both add the selections to a clipboard, but cut also removes the selection from the project. In the case of this research the cut function could revert the selection to white as removing it from the project entirely would cause problems. Paste applies the clipboard to a selection. There are multiple ways of doing this. For example, if the user has a selection that is smaller than the selection in the clipboard. The clipboard selection could either go beyond the user's selection or stay confined. Also, if the user has a selection larger than the selection in the clipboard. The clipboard selection could then either paste itself to an area as large as the selection in the clipboard and leave the rest of the user selection untouched or it could multiply itself to fill the missing space. In the case of this research both methods of both examples can be useful to the user and should be implemented.

Most image and text editing software also employ a toolbar, formerly known as ribbon. The options of the toolbar change how the user interacts with the application. For example, tools in the toolbar could be a move tool and an edit tool. If the move tool is selected the user is unable to edit the project and instead is allowed to move and rotate it. Then if the edit tool is selected, the user is unable to move the project but is able to edit it. The toolbar and tools are useful since the user only has a limited number of buttons at their disposal. If the tools didn't exist, an unreasonable amount of key combinations would be required to operate most editing software. The tools add to the user experience by making the application more intuitive to use. In the application developed in this thesis, at least a move tool and an editing tool would be useful to the end-user.

In some common applications such as the Microsoft Office lineup the toolbar is located at the top of the screen. The application design of this thesis places the toolbar to the side as shown in figure 5. This is more in line with the style of Photoshop. Placing the toolbar at the top of the screen would remove screen real estate from the project view. This in not preferable as the project view should be emphasized to give the user more control over the project.

In addition to a menu bar and a toolbar, some applications utilize a side panel. This is a panel often on the right side of the application and holds all contextual information and parameters. Thus if, for example, the user highlights a part of the project, the side panel has the options the user has to interact with that selection. The side panel is hidden unless a context is defined. The side panel adds an extra layer of interactivity and user friendliness by displaying all the possible interactions to the user.

The project view, visible in figure 5, is divided into two parts. In this thesis the parts are referred to as views, the first of which is called pixel view. Pixel view is the main editing part of the application. It is composed of a grid of black and white squares, referred to as pixels, in the style visible in figure 3. These pixels can be turned on and off by the user. The view also has panning and zooming to aid visualization of the project. The second view is called the three-dimensional (3D) view. This view is a representation of the fabric after weaving. The blue yarns represent warps and the green yarns represent wefts as can be seen in figure 6. This view also allows editing by interacting with the warps. Clicking on a section of a warp makes it wrap around the next available weft. 3D view can be panned, zoomed and rotated to find the ideal viewing angle. The viewing options available in the menu bar affect the projects view. The options are to have both the pixel and the 3D view visible at the same time, or either view by themselves.

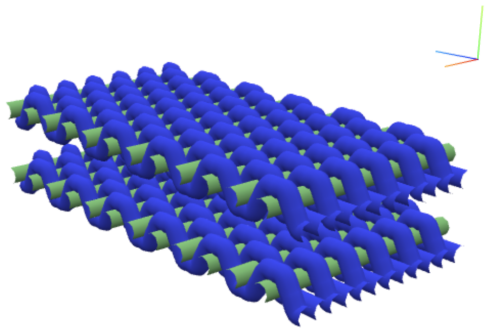

Figure 6: A three-dimensional view of a double layered plain weave simulated by the application developed in this thesis.

4 Implementation of the application and the features

The methods and algorithms of implementation are introduced in this section. Most of the application's visual side, apart from the project view, is implemented via HTML and Cascading Style Sheets (CSS). The rest, i.e. the project view and user interaction are implemented via JavaScript with the help of three.js. Introduced in section 4.1, are the methods for implementing the key features of the application. These features include the basic WebGL canvas using three.js, adding pixels and tubes to the two views, configuring two different cameras for the views. In addition, the section described a way to user input via interaction with the two views or via HTML elements from the menu bar. Section 4.2 takes a deeper look at the JavaScript behind the application by reviewing some key algorithms. The resulting application using these methods and tools is shown in figure 7.

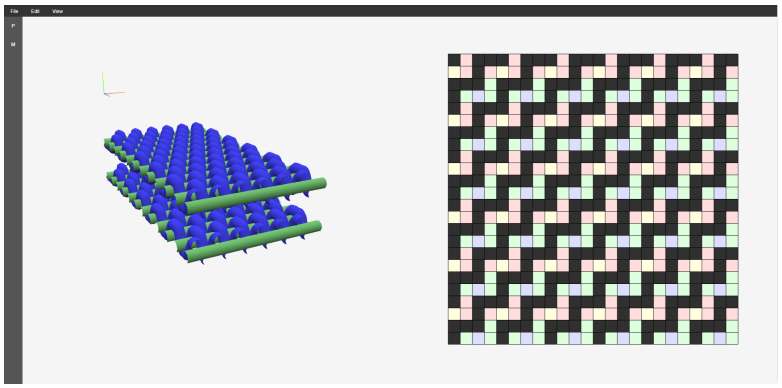

Figure 7: A screenshot of the finished webtool.

4.1 Methods

The project view is created by using a JavaScript program to attach a WebGL canvas to an HTML5 canvas, to create a WebGL canvas. This creates a canvas that can be modified and rendered onto using WebGL. Three.js, introduced in section 2.3, helps by providing functions that automate this process. The Three.js objects are accessed from the THREE library. Two WebGL canvases are created, for both the pixel view and the 3D view. After this Three.js provides an object called THREE.Renderer for controlling what is rendered onto the WebGL canvases, and from which angle it is viewed from. Other essential objects provided by the library are THREE.Scene and THREE.Camera. For every frame the THREE.Scene and the THREE.Camera are provided to the renderer, which renders the THREE.Scene from the point of view of the THREE.Camera onto the WebGL canvas. The THREE.Scene object holds a list of added THREE.Object3D objects.

THREE.Object3D objects have a THREE.Geometry and a THREE.Material. The THREE.Geometry defines the shape and the location of the object. In the

case of the pixel view, the geometry is a THREE.PlainGeometry, which defines the shape as a 2D plain in 3D space. In the 3D view the geometry is instead a THREE.TubeBufferGeometry, which creates a tube of determined radius and follows a determined path. The material is an object that defines the color of the object and the shading of the light that hits the object. The material in used in pixel view is a THREE.MeshBasicMaterial. This material does not create shadows or reflect light, which is not necessary in the pixel view where the view comprises of a flat plain of pixels. Shading could make the viewing more difficult. In the 3D view the material used is a THREE.MeshLambertMaterial. This material does shade when combined with a light source. The lightsource used is a THREE.HemisphereLight, which creates an even light from a single direction, and it is placed above the other objects. The combination of the material used, and the light causes the tubes to have a shadow on their bottom side. This helps the user determine the three-dimensionality of the view and helps separate the tubes from each other. When the geometry and the material are created, they are combined into a THREE.Mesh object, which is a subclass of THREE.Object3D. This mesh object can then be added to the scene.

The THREE.Camera object has two variations: a perspective camera and an orthographic camera. THREE.PerspectiveCamera is comprised of the location, direction and the viewing angle. It uses these to angle itself in the THREE.Scene and capture the objects in it as shown in figure 8. The perspective camera is best used for capturing 3D objects. In the application of this research it is used in the 3D view. Orthographic camera is instead used for capturing flat objects. It does not have a viewing angle. Instead it uses the entire canvas size as the viewport as seen in figure 8. This removes perspective warping from scenes as it removes depth. All objects appear their actual size regardless of their distance to the camera. Therefore orthographic camera is used in the pixel view. THREE.Camera objects can be controlled by a THREE.OrbitControl class. This class defines user controls for panning, zooming and rotating of the camera. The 3D scene employs all of these, but rotating is removed from the pixel view as it is unnecessary since the view and the objects in it flat planes.

User input in the two views is achieved with THREE.Raycaster. Both views in the project view keep track of the mouse of the user by updating its location when it is moved. This is done by adding an event listener to mousemove that triggers a function that stores the new location of the mouse. For every frame The THREE.Raycaster is then set to send a ray from the camera to the mouse location and return a list of all the objects that intersect with that ray. This list can then be used to interact with the objects. For example, in the 3D view, if the mouse is over a warp, that warp is turned red to signify that it can be interacted with. By adding an event listener to mouse click, the object being interacted with can then be toggled on or off. The same applies for the pixel view.

Another way user input is achieved is via HTML elements. The menu bar at the top of the application window has a list of HTML buttons. These buttons can be fixed to trigger a function upon clicking the button. Functionality from these buttons includes creating, saving and opening projects, as well as exporting a project as a TIFF file. Also included buttons to change the projects view to include only

one of the two views, i.e. 3D view or pixel view, or both.

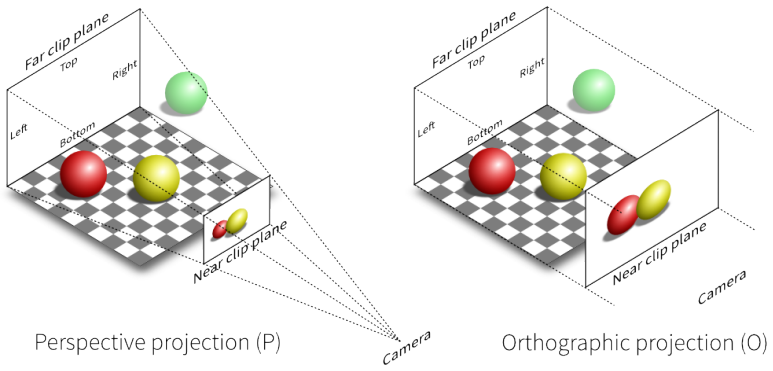

Figure 8: The difference between a perspective camera and an orthographic camera.

4.2 Algorithms

The pixel view of the application can be pictured as a matrix. Zeros representing white pixels and ones representing black pixels. This matrix is stored as a multidimensional array in the application. The state, i.e. one or zero, of a pixel is referred to as a toggle in the application. The toggle can be retrieved from the matrix by coordinates as shown in function below.

```
1 // Return the toggle state at coordinates
2 function getToggle(coordinates) {
3   return matrix[coordinates.y][coordinates.x];
4 }
```

The state of the toggle can then be switched with the following function.

```
1 // Toggle coordinates
2 function toggle(coordinates) {
3   if (getToggle(coordinates) == 0) {
4     matrix[coordinates.y][coordinates.x] = 1;
5   } else {
6     matrix[coordinates.y][coordinates.x] = 0;
7   }
8 }
```

A problem arises when working with multilayered weaving. Unlike with single layered weaving, multilayered weaving assigns multiple pixels to represent a single crossing of a warp and a weft. In the application these are called sets. As shown in figure 3, in double layered weaving every other row represents the top weft and every other represents the bottom weft. So then if the top weft is black and the bottom weft is white, the application is faced with a warp that goes above the top weft and below the bottom weft, which is an impossibility. So instead of allowing the user to toggle any pixel, the application tests if the toggle is possible and returns if it is not. This is

shown in the function below. A pattern can be realized when looking at the possible pixel combinations. The black pixels must always be above the white pixels, or the set is impossible. This can be used to make testing the possible sets easier.

```
// Try to toggle the coordinates in set
function tryToggle(coordinates) {
  // Toggle the point at coordinates to test if it is ok
  toggle(coordinates);
  let previousCoordinates = getStartCoordinates(coordinates);
  // Test set if a 0 precedes a 1
  for (let i = 0; i < layers; i++) {
    let nextCoordinates = getNextPointInSet(previousCoordinates);
    if (
      getToggle(previousCoordinates) == 0 &&
      getToggle(nextCoordinates) == 1
    ) {
      // reset the toggle of coordinates and return
      toggle(coordinates);
      return;
    } else {
      previousCoordinates = nextCoordinates;
    }
  }
  // Toggle was successful. Add coordinates to a list of changed
     coordinates
  changedCoordinates.push(coordinates);
}
```

A way to rotate the toggle state of a set should be added, especially to allow easier editing of the 3D view. In the application the function starts from the top of the set and finds the first zero. It then toggles that zero and the state of the set has changed. If no zeros are found, then all pixels are ones. In this case the function turns all pixels in the set to zero. This function is presented below.

```
// Rotate the toggle state of a set
function rotateToggle(coordinates) {
  // Get the coordinates of the first point in a set
  let coordinates = getStartCoordinates(coordinates);
  // Try to toggle the first 0
  for (let i = 0; i < layers; i++) {
    if (getToggle(coordinates) == 0) {
      tryToggle(coordinates);
      return;
    }
    coordinates = getNextPointInSet(coordinates);
  }
  // All points are 1 so turn them to 0
  coordinates = getPreviousPointInSet(coordinates);
  for (let i = 0; i < layers; i++) {
    if (getToggle(coordinates) == 1) {
      tryToggle(coordinates);
    }
    coordinates = getPreviousPointInSet(coordinates);
```

```
20    }
21 }
```

4.3 Future improvements

Due to time constraints, some desired features are missing from the application. These features include editing tools, the toolbar, and the side panel. These features and their desired effects were introduced in section 3.3.

A major issue found during the development of the application is speed and efficiency. The application slows down greatly when either creating a project with large dimensions or resetting the the projects too many times. The issue of creating large projects can be improved with better algorithms. However, part of the problem could arise from the predetermined functions of three.js. If this is the case, then the only way to improve performance noticeably is to create the application using only WebGL. The problem of resetting the project too many times stems from the built-in removal of THREE.Scene in three.js. The methods provided in three.js for this have been found lacking and nonworking. If the THREE.Scene is not removed properly, references to the objects in it remain and they are not removed from the memory allocation of the browser. On top of this the THREE.Renderer leaves references to the objects and continues to loop through them, causing more time to pass with every tick, slowing the rendering. The solutions could be to either find a way to completely remove all objects from a THREE.Scene, or again create the application using only WebGL.

Another improvement to the application would be the storage of user data. Currently the only way to save projects is to download a project file and store it on the user's computer. A way to improve this could be a database on the server-side. Users would log in to the database and could save and open projects from it. This, however, does not help with contingency of the projects. If the browser suddenly crashes, all progress not saved manually is lost. The solution could be to auto-save the project at regular intervals to either the database or a web storage.

5 User test

The user test implemented in this thesis was done in the form of a workshop. Students of textile design from Aalto University were invited to try the application and give their opinions on the design, the features and future improvements. The Workshop began by giving the testers time to try the application by themselves with minimal instructions. They were then asked to answer questions about their experience in the form of a questionnaire. The questions are listed in appendix A. After this, an informal discussion was held.

The questionnaire was structured so it had a clear line to follow. Starting from asking for the first impressions, following up with what the testers found useful. After the testers had seen what they could do with the application, the next question was what they found was missing. Lastly, they were asked what kinds of future improvements could be added to the application and their overall opinion on whether the app is useful to the students of Aalto.

Several conclusions can be drawn from the results of the workshop. First, what the testers found useful. Most testers found that the ability to view the final fabric in 3D was useful. They expressed that it assists the learning of woven structures with visual aid and also helps with visualizing the woven structure. One tester mentioned, "The possibility to simulate structures without any material waste." This references to the current methods of designing fabrics, where the only way to view the resulting fabric is to weave it. If there are errors in the weave, the process needs to be redone. The reason the tester talks about material waste is that it is difficult to recover yarns from failed fabrics.

A large portion of the features the testers found were missing were grouped under user control. This means that the features would enable the user with more control over the creative side of the application. Things such as defining different colors for warps and wefts, resizing them, and letting the user define the amount of space between them. These features would help the user better visualize the woven structure and allow for a better user experience. Another feature some testers found the application was lacking were editing tools, which were mentioned as important future improvements in section 4.3. Many of the features the users found were missing from the application were also mentioned as possible future improvements. One of these was an improved library of predetermined weaves for users to choose from. Despite the missing features, most of the testers thought that the application could be useful for the students of Aalto University.

The features that the testers found were missing and their suggestions for future improvements are not implemented in the application. They are however, important features that should, in case of future development of updates, be added among the first things.

6 Summary

This thesis looked at the current state of woven textile and e-textile design, and their methods and the tools used. It can be argued that the methods have stagnated with unintuitive tools and error prone workflows that unnecessarily waste material and time. Solutions to the problem have been developed, such as AdaCAD [14]. Some of these are however, not accessible to the public and most software is only for industrial use. In addition, few applications currently available have the ability to simulate a 3D model of the weave structure.

An application was developed in this thesis via a UCD process, looking to fill a hole in the design process. The UCD process was done in three parts. A rudimentary first version with basic functionality, a second version with more features and usable for designing textiles, and finally a workshop for end-users to test the application and give feedback on what was good and what was missing. During the first stages of the design process, some features were discussed and agreed upon with Emmi Pouta, a researcher at the Department of Communications and Networking (Comnet) and a potential end-user of the application. These features included viewing of the woven structure in a 3D view and a pixel-based view, a library of predetermined woven structures, and easy conversions between fabrics of different layers.

The application was created using HTML, CSS, WebGL, three.js, and JavaScript. It is a web application which allows it to be used on nearly any computer with access to internet and a modern web browser. It also allows for easier updates and sharing of projects. Features implemented in the application are saving, opening, and creating new projects, as well as exporting projects as Tagged Image Files (TIFF). In addition, a user can create fabrics of up to four layers and view them in 3D or in pixels. Both the 3D and the pixel view can be edited, panned, and rotated.

The workshop was conducted on textile design students from Aalto University. The workshop consisted of the testers trying the application, answering questions on a questionnaire, and a free form discussion at the end. Most testers found that being able to view a simulated three-dimensional view of the fabric was useful. They also found that a lot of features were left missing, such as basic editing tools and a library of predetermined woven structures. Most testers expressed that the application could be useful especially for learners and people not familiar with woven structures.

Using the information acquired from the workshop and observation of the performance of the application, future improvements could be done to it. These improvements could include. Improving efficiency should be a major priority of improvement. Currently creating large projects hinders the performance considerably. In addition, resetting the project multiple times, begins slowing the application down due to the objects provided by three.js leaving traces of deleted objects in memory. This causes memory leaks in the browser and thus, hindrance of performance. Another future improvement would be to implement the features the testers found were missing from the application.

References

[1] K. Flood, "The original creators: Diana dew," *VICE Media LLC*, 2011. [Online]. Available: https://web.archive.org/web/20111219075111/http://www.thecreatorsproject.com/blog/the-original-creators-diana-dew

[2] J. Shenton and E. Ridsdale, *Woven Textile Design*. London, England: Laurence King Publishing, 2014, ISBN: 9781780675695.

[3] W. H. Dooley, *Textiles*. Boston, USA: D.C. Heath and Co, 2007. [Online]. Available: www.gutenberg.org/files/24077/24077-h/24077-h.htm

[4] A. M. Collier, *A Handbook of Textiles*, 3rd ed. Oxford, England: Wheaton, 1980, ISBN: 9780080249742.

[5] J. Essinger, *Jacquard's web : how a hand-loom led to the birth of the information age*. Oxford, England: Oxford University Press, 2007. [Online]. Available: https://ebookcentral-proquest-com.libproxy.aalto.fi/lib/aalto-ebooks/detail.action?docID=422876

[6] "Tc2 digital jacquard loom." [Online]. Available: https://www.digitalweaving.no/tc2-loom/

[7] M. Salolainen and M. Fagerlund, "Woven structures," p. 6. [Online]. Available: https://mycourses.aalto.fi/pluginfile.php/841617/mod_resource/content/1/Woven_Structures_2018-highlights.pdf

[8] S. Eriksson, L. Berglin, E. Gunnarsson, L. Guo, H. Lindholm, and L. Sandsjö, "Three-dimensional multilayer fabric structures for interactive textiles," *Proceedings of the 3rd World Conference on 3D Fabrics and Their Applications*, pp. 63–67, Jan 2010. [Online]. Available: https://pdfs.semanticscholar.org/b628/0ea19bdbb5fb9dffc6f828cf951b8e545924.pdf

[9] D. Cantor and B. Jones, *WebGL Beginner's Guide*. Packt Publishing, 2012. [Online]. Available: https://www.oreilly.com/library/view/webgl-beginners-guide/9781849691727

[10] T. Parisi, *WebGL: Up and Running*. O'Reilly Media, 2012, ISBN: 9781449323578. [Online]. Available: https://www.oreilly.com/library/view/webgl-beginners-guide/9781849691727

[11] E. Kay and A. Boodman, "A dash of speed, 3d and apps," 2011. [Online]. Available: web.archive.org/web/20150411184902/http://chrome.blogspot.co.uk/2011/02/dash-of-speed-3d-and-apps.html

[12] "Three.js library." [Online]. Available: https://threejs.org/docs/index.html#manual/en/introduction/Creating-a-scene

[13] J. Dirksen, *Learning three.js : the JavaScript 3D library for WebGL*. Birmingham, England: Packt Publishing, 2013. [Online]. Available: https://ebookcentral-proquest-com.libproxy.aalto.fi/lib/aalto-ebooks/detail.action?docID=1481113

[14] M. Friske, S. Wu, and L. Devendorf, "Adacad: Crafting software for smart textiles design," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. Glasgow, Scotland UK: moi, May 2019, pp. 290–294, doi: 10.1145/3290605.3300575.

[15] C. Abras, D. Maloney-krichmar, and J. Preece, "User-centered design," in *Encyclopedia of Human-Computer Interaction*, vol. 2. Bainbridge, Washington: Thousand Oaks: Sage Publications, 2004, pp. 763–768, doi: 10.1.1.94.381.

[16] D. A. Norman, *User centered system design : new perspectives on human-computer interaction*. Hillsdale, New Jersey: Erlbaum, 1986, ISBN: 0-89859-781-1.

[17] T. Lowdermilk, *User-Centered Design*. England: O'Reilly Media, Inc, 2013, ISBN: 9781449359805. [Online]. Available: https://learning.oreilly.com/library/view/user-centered-design/9781449359812/?ar

[18] I. Hickson, "Web storage (second edition)." [Online]. Available: https://www.w3.org/TR/webstorage/

A Students' opinions on the application at a workshop

What was your first impression of the application?			
Age	**Gender**	**Weaving Expertise**	**Answer**
48	Female	Intermediate	Could become useful for visualizing the structure in jacquard design.
26	Female	Expert	Quite easy layout to use for anyone who uses computers.
26	Male	Intermediate	First, it looked like an old computer game. After I started gaining experience with the program, it had been very good in the beginning for me to learn different weaving structures since you could see how the yarns go through.
32	Female	Intermediate	Useful to be able to see the fabric from different angles
22	Non-Binary	Intermediate	Useful, visual simulation to help learn more thoroughly the technicalities of woven fabrics. Especially useful to experiment with the simulations.
34	Female	Novice	it's sparse and minimal
30	Female	Intermediate	for one layer it's quite easy to understand and imaging the structure
What did you find useful?			
48	Female	Intermediate	Visualizing the structure
26	Female	Expert	It's nice to see both the pixel structure + the 3d impression of the fabric. Zooming and all works nicely as well as the rotation.
26	Male	Intermediate	It's practical to see both the 3D fabric and also the file of the structure. It helps visual learners better to understand woven structures. I think with updates this would be very useful for learning woven structures. Redo and undo were practical tools too.
32	Female	Intermediate	Same as above, though in triple canvas the 3D view was a bit confusing
22	Non-Binary	Intermediate	Three-dimensionality of the woven structures, different colours. The possibility to simulate structures without any material waste.
34	Female	Novice	having dual view and 3 dimensional view
30	Female	Intermediate	3d image explain the structure really well

What do you think was missing?			
48	Female	Intermediate	Colors of the yarns, commands to be like redo, undo, edit
26	Female	Expert	At the moment when making for example a 3-layered structure, I find the 3d model a bit messy and too dense/yarns too bulky .The structure (on right) is quite hard to use: I don't know why when pressing the structure to make a warp yarn "black" it positions itself on top "row" automatically and not where I want it to be (for example two pixels beneath it) instantly.
26	Male	Intermediate	The triple layer fabric option is a bit unpractical. The colors are too light and one cannot see properly the binding points. Also I started to understand the triple fabric with the 3D picture.
32	Female	Intermediate	a possibility to color the weft yarns with different colors.
22	Non-Binary	Intermediate	Library of the basic structures, to see them already ready in 3D.
34	Female	Novice	tools
30	Female	Intermediate	it's a bit confusing for link the double (multiple)layer, cause the warp overlap each other
Suggestions for future improvements			
48	Female	Intermediate	Colored yarns for better visualization of the design. Different yarn structures and thicknesses?
26	Female	Expert	Adding an option for different colour weft yarns could help and maybe even having the option to give them a name. Then these different weft yarn could be displayed with their names on the sides of the structure. 3d model could show the colours as they are. Maybe there could also be an option to somehow "separate"/widen the gap between the loom yarns in the 3d model. This might help to see more clearly into the structure.

26	Male	Intermediate	The colors of the triple layered fabric could be brighter. I wish there had been the photoshop equipment, so I could see the whole program in action. It would also be good to see different qualities of the yarns since some yarns are not that thick as the one in the tool.
32	Female	Intermediate	Possibly the different thicknesses of yarns could be added to the 3d, so that the designer could see how that affects the multi-layer design
22	Non-Binary	Intermediate	Structure swatches/library in 3D. The program could maybe "show" if something would not work in real life.
34	Female	Novice	add colour options and tools. make it easier to navigate the different layers and views, perhaps similar views as Rhino. it also needs to be faster.
30	Female	Intermediate	1.for the overlapping part(previous answer) could have a little animation when doing the interwoven between different layers, 2.and also adding marks for different colours(about the position) could also be helpful,
Could this application in your opinion be useful for the students at Aalto?			
48	Female	Intermediate	This could be useful
26	Female	Expert	This could be useful
26	Male	Intermediate	This could be useful
32	Female	Intermediate	Could be useful, but maybe mostly to see the fabric in 3D. For me it seems easier to do the designs in Photoshop, though that might be just because I'm used to it.
22	Non-Binary	Intermediate	This is exactly what I've been missing!
34	Female	Novice	This could be useful
30	Female	Intermediate	This could be useful