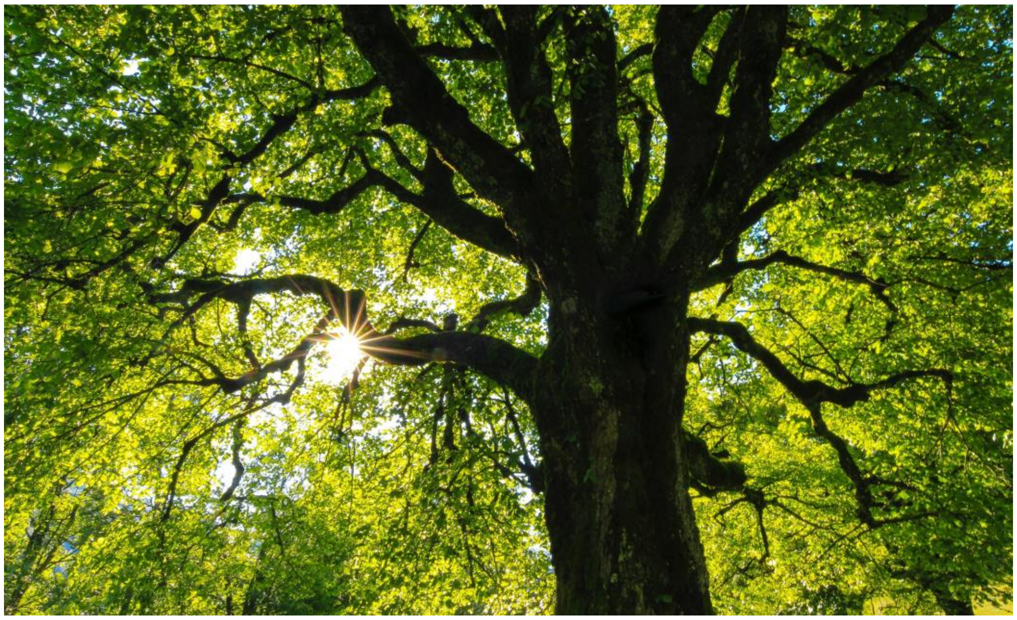


FAMILY TREE GENERATOR

System Proposal – 11/8/22

PREPARED BY:
MATTHEW FISK

Seattle Pacific University

Table of Contents

Executive Summary

The SPU Computer Science Department is tasked with creating a family tree generator, dubbed hereafter as the FTG. The following document is an overview of the system's goals, proposed functions, monetization model, risks, and values. The FTG will be a publically available website with no paywalls funded through advertising. It is focused entirely on creating family tree diagrams and will not have any other focus.

The project is deemed feasible after considering various benefits and risks, detailed more closely in the Expected Costs and Benefits, Constraints, and Feasibility Analysis sections. Some financial risks may include low user retention rates and server hosting costs. Still, the FTG is predicted to be financially successful because of a lack of strong competitors and the widespread use of family trees.

1.0 Introduction and Overview

Problem Statement

Family trees are diagrams that are used worldwide. Everyone, no matter their background or status, has a family lineage that can be uniquely mapped out in a way that matters to each person on the individual level. It's possible to do this on paper or with generalized graphing tools, but they can often be clunky and hard to plan out without considering all family members. This presents an opportunity for a graphing application specialized in making family trees. One that dynamically updates and organizes itself as the user places family member information.

Project Vision and Scope

The vision of this project is to create a straightforward, dynamically updating family tree generator. It will be a web-based application that will run in the user's browser and not require a download. To use the application, the user will create visual nodes on a blank canvas containing important family member information like a name, gender, age, portrait, and more. By right-clicking the nodes, the user can create a line that starts at the node and follows the mouse until it is attached to another family node, which is set as a parent or child of the original. Connecting floating family nodes to larger sections of the tree will automatically organize them.

Requirements Summary

The most crucial system requirements include the FTG's ability to:

- Add family nodes and edit information
- Upload custom portraits to family nodes
- Attach family nodes to their parents and children
- Have customizable color-coding of lines between parents and children
- Have real-time organizing of family nodes
- Drag and drop family nodes

- Pan and zoom around the canvas
- Create a website account through an email and a password
- Save user progress over the cloud
- Export a final product as a PDF, PNG, or JPEG
- Display advertisements
- Run the application within a browser

Stakeholders and Their Interests

The entities that will be directly or indirectly affected by the FTG include:

- **SPU Computer Science Department** – We are the ones that will develop the FTG, so we must deliver a product that fulfills all expectations. It must be bug-free and have all the expected functions. If we do not deliver, the whole product faces jeopardy.
- **Advertisers** – Advertisers will be the product's primary revenue source, so in return they get clicks from users. If the FTG flops, no users will see or interact with their advertisements. The advertisers also want to ensure the product they are placing their ads on is appropriate.
- **Web-Server Host** – The client will not host this website and will be outsourced to another company specializing in server hosting. If this company has slow server access or frequent outages, the FTG may suffer and lose users and revenue. If the FTG does not perform well because of poor server service, the project may be pulled, and the web server company may lose a customer.
- **Users** – The users expect the product to work intuitively and without bugs. If this isn't the case, the user will waste their time.

Expected Costs and Benefits

Expected Benefits:

The most apparent motivation for the project is revenue from advertising, but there are many other general benefits. Creating a system like this can be a worthwhile experience in creating other graphing software or interactive UIs. We can likely use code from this system in many future projects. The FTG also presents an opportunity to grow our portfolio; it is something of substance that can be shown to future clients to help prove our skills and capabilities. Lastly, the project will increase name recognition and may create entirely new opportunities for different systems to design.

Expected Costs:

There are two types of costs we will experience during the development and lifetime of the FTG. The first is the upfront costs, which comprise paying developer wages, tester wages, and equipment to test or create the project. These costs will be during the development, before the project's release. The other type of cost is an ongoing cost, which comprises paying for website hosting services and any potential libraries or licenses that the FTG may need. This type of cost will be after the FTG is finished and published.

Constraints

With the nature of the FTG, several problems may arise:

- An internet connection is required to run the application
  - We are most likely unable to do anything to fix this problem fundamentally, but there are some ways it can be mitigated. Since the FTG is a browser-based application, users can continue using the core functions offline if they access the website beforehand with an internet connection. Other functionalities, like cloud saving and logging in or out of the website, will not work offline. These issues may be mitigated by a downloadable build or a downloadable custom file type if we want to add these to the project scope in the future.
- Users use many unique internet browsers, so it isn't practical to assume the FTG will work smoothly on all of them
  - We will mitigate this by thoroughly testing the most commonly used browsers. We guarantee that all aspects of the FTG will work well with Google Chrome, Chromium, Mozilla Firefox, Microsoft Edge, Opera, Brave, and Safari. These browsers collectively make up the overwhelming majority of internet browsers commonly used. Any other browser coming up with an issue is unlikely to make any financial impact on the success of the FTG.
- The application scope is limited only to PC
  - Users won't likely be able to operate the FTG on devices other than a PC. These devices might include mobile phones, tablets, gaming consoles, or smart applications. This is unlikely to be important because most users are predicted to use a PC, regardless. The only device other than a PC that may need thought is a mobile device, which can be mitigated by implementing a mobile build if the scope is expanded.
- The application relies on text to communicate functionality
  - Using text to explain the functions of the FTG will provide a few communication issues. If someone does not speak the language the application is written in, it is unlikely they will be able to use it to its fullest extent. The application will be written in English by default, but this may be mitigated by adding other commonly used language localizations if necessary. The second major communication issue is with the visually impaired. If a user is blind or doesn't have good eyesight, they cannot read the text in the same way as most users. This can be resolved entirely because most visually impaired individuals have other software capable of reading text on a screen. We will simply have to embed the text into the browser with these types of applications in mind.

Recommendation

This document serves several purposes for the creation of the FTG. The primary purpose is as a high-level overview of the system's vision and functionality. This includes a list of what the FTG is expected to do and some core insights like the problem being solved and the monetization model. Another significant purpose of this document is to identify risks and propose solutions. It's important to note that while considerable effort is put into these aspects, the system design is not set in stone. Some minor tweaks may be made to the design of the FTG while in production. It is also possible that additional risks will be identified later on or that an existing one may have been misidentified. Ultimately, this document is a valuable tool for guiding the project, but we cannot predict the future of the FTG with absolute certainty. That should always be a consideration until the system is fully finished and released to the public.

Document Overview

The rest of the document contains the following sections:

- **Project Initiation Request**
- **Feasibility Assessment**
- **Functional Requirements**
- **Data Requirements**
- **Non-Functional Requirements**
- **Requirements Model**
- **System Evolution**
- **Conclusions and Recommendations**
- **Appendices**
- **Glossary**
- **Bibliography**

2.0 System Initiation

Project Initiation Request (PIR)

PIR-00000 *[PIR Number to be assigned by the Project Office]* Project Initiation Request (PIR) – L1 v5.0

Project Name: ________Family Tree Generator________ Student Name: ______Matthew Fisk___

0. General Project Information

Project Name:	Family Tree Generator
Two Sentence Request Description:	The Family Tree Generator is a small-scale program to help users create a family tree. The user would input their information and see the tree generated in real-time.
Requested Launch Date(s):	November 25th, 2022
Department(s) Affected By Project:	Computer Science
Project's Customers:	People who want to make a family tree for personal use
Date Request Submitted:	9/30/22

1. Project Sponsor and Manager

Project Sponsor

Name:	Andy Cameron
Title:	Professor
Department:	Computer Science – SPU
eMail:	acameron@spu.edu

Business Project Manager & Requestor

Name:	Matthew Fisk
Title:	Student
Department:	Computer Science – SPU
eMail:	fiskm@spu.edu

2. Business Problem or Opportunity: The motivation for this request

A family tree is a common type of diagram used worldwide by all sorts of people. It has a very specific use and doesn't have very many high-level uses, but it's something any person could feasibly want to make. Since there is negligible professional reason to use a family tree generator, it will not be a paid program. Instead, the system will be placed online for free and will use advertisements to fund the web servers, backend maintenance, and generate revenue.

The application will allow the user to add people that will appear as free-floating UI boxes on a blank canvas. In these UI boxes, the user can input information like name, age, gender, and a portrait. The user will also be able to assign existing people as a child/parent or create a new child/parent directly from the UI box.

3. Justification, Impact, and Importance

Assumptions

▪ Returning customers will not be common
▪ Backend and server costs will be cheap
▪ Customers vary widely and are not using this program for professional purposes

Competitive Landscape / Context

▪ There are many available similar programs online for free
▪ Many similar free programs are not robust and have limiting design decisions
▪ Most free programs do not even have advertisements
▪ It is common for a family tree generator to be one of many free diagrams on its' host website
▪ FamilyEcho - Very robust software that works exactly how you'd expect it. UI design is very basic
▪ DNAweekly - Deliberate design decisions that hurt the software. Extremely slow to use. UI design is somewhat confusing but works well at times

Return, Opportunity, or Impact	One Time	On-Going
▪ Revenue from advertisements	$ 0	$ 50 / month
▪ Sponsorships	$ 200	$ 0

Intangible Benefits	Impact or Value
▪ Increased name recognition for future and past projects	$ N/A
▪ GUI and tree code can be reused for future projects	$ N/A
▪ Project can be added to a portfolio to show future clients	$ N/A

4. Project Requirements

4.1. Must Haves

4.1.1.	Export options for PDF, PNG, and JPEG
4.1.2.	Creating lines from parents to children
4.1.3.	All people from the same generation line up on the same line
4.1.4.	Drag and drop people
4.1.5.	Portrait image uploading
4.1.6.	Customizable color coding of people and their lines to children
4.1.7.	Account system allowing users to sign into the website
4.1.8.	Save progress on the cloud
4.1.9.	Adspace

4.2. Nice to Haves

4.2.1.	Log in with Google or Meta account
4.2.2.	Customizable visual themes
4.2.3.	Optional preset visual themes
4.2.4.	Exportable custom filetype

4.3. Don't Do's (Out of Scope)

4.3.1.	Localization integration other than English
4.3.2.	Pop-up video advertisements
4.3.3.	Paid no-advertisement version

5. Project Costs (Operating and Capital, Onetime and Recurring

Labor Costs

Type	Team(s) Affected	Low (hrs)	High (hrs)
Analysis & Design		3	8
Development		21	32
Testing and Quality Assurance		4	8
Systems Integration		8	16
Deployment		1	2
Support and Maintenance		1	2
Sales and Marketing		0	0
Total		**38**	**68**

Comments: None.

Maintenance Costs

Type	Hours / Month Low	Hours / Month High
System / User Support	1	4
Business / Process Support	0	0
Total Support & Maintenance	**1**	**4**

Comments: None.

Capital Costs

Description	Quantity	Cost ($)
Web server costs	N/A	$ 5 / month
Total		**$5.00 / month**

3.0 Feasibility Assessment

Introduction

There are several aspects of the FTG, and some are more feasible for us to create than others. To communicate the challenges and strengths regarding building the FTG, several unique aspects will be analyzed for their feasibility. The following list is the rating scale that will be used.

- **Very Low** – There are a lot of risks and challenges that come with this aspect of the application. There's a great deal of uncertainty, and we will likely encounter unexpected problems. An excessive amount of time will be spent dealing with this aspect. It is recommended to fundamentally change some aspects of the application to mitigate these issues.
- **Low** – This aspect of the application is somewhat risky to build. Several prominent or a few major issues are projected to slow down development. We may run into unexpected problems relating to this application aspect.
- **Moderate** – There are a few risks with this aspect of the application, but it is doable. It's unlikely that any unexpected problems concerning this aspect will appear. The development is projected to go smoothly. No changes to the application are recommended.
- **High** – No substantial risks are present in this aspect of the application. It is extremely unlikely that any unexpected problems regarding this aspect will appear. The development will go fast, and we are thoroughly prepared to deliver a strong product regarding this aspect.
- **Very High** – The circumstances of this application are perfect for this aspect. There is a near-zero chance that we will encounter any unexpected problems. The application's development will be fast-tracked, and we are prepared to deal with any challenges that this aspect poses.

Feasibility Analysis

Technical Feasibility – **High**

- User Familiarity: Applications that run in the browser are very common across the web, and it is unlikely that the users will be confused by this aspect. The concept of a family tree is also well-known and easy to understand, so the user should be familiar with at least the most basic functions of the FTG without a tutorial.
- Developer Familiarity: We have made applications that run in the browser and can be displayed on a website. This type of application is very well documented across the web, so if there are any problems, they are likely to be fixed quickly. We also have experience in web-based databases. We do not have any experience creating graphing software or creating online website accounts.
- Project Size and Scope: The FTG's scope is relatively basic, and no unnecessarily complicated aspects are being requested. There is no problem regarding the size or scope.

Resource Feasibility – **Very High**

- Manpower: The project scope suggests it is possible to complete the FTG with a team as small as two or three developers, but we have more than that on hand. Assigning more developers will make the project go by much more smoothly.
- Equipment: Nothing about the system suggests there is any special equipment necessary. We likely have all of the software and hardware that will need for the project on hand already.
- Libraries and Licenses: Since the project isn't incredibly complex, there probably won't be any special data libraries or licenses that will have to be purchased. They will probably be simple and inexpensive if we need to purchase these.

Schedule Feasibility – **High**

- There is a deadline for the project to be completed on December 30th, which is reasonable. The project scope is small, and we have extra developers, so it will probably go by smoothly. We estimate we will finish the prototype within three weeks and a final product within six weeks. We estimate that there may be issues with creating an account creation system, which may require additional time.

Organizational Feasibility – **Moderate**

- The biggest risk of the FTG is the organizational aspect. Many unique actors will interact with the system differently, which can easily be unpredictable. The data servers and website hosting servers are going to be outsourced. If the companies that you choose are unreliable, this could hurt the project's success. If the FTG is a success and the project grows, it may require a much greater organizational need to keep it running. This shouldn't be an issue as long as good server hosting companies are picked to outsource to, and future project scope is expanded safely.

Legal Feasibility – **High**

- Nothing about the FTG requires uncommon legal licenses. The system will not deal with highly sensitive data that needs extra attention, like credit cards or SSIDs. The most sensitive data the project will deal with is the user's email, password, and potential relatives they enter into the FTG. If this data is not handled appropriately and is leaked, it may cause issues for the users. Passwords should be encrypted across the web, and all data should be stored in a location that is not easily physically accessible to mitigate this.

Contractual Feasibility – **High**

- The project is tied to two server hosting companies at most, a database server and a website hosting server. Although it is likely a single company will be chosen to do both tasks. The project's success relies on this company. It is also worth noting that this company will have access to the IP addresses of the users who visit the website, along with their password, email, and family tree data. It is essential to select a company that can be trusted.

Additional Comments

This feasibility analysis was done with the project launch in mind. As the FTG grows in scope after its launch, new issues that do not accurately reflect this analysis should be expected.

Conclusion

Ultimately, the project scope is estimated to be **high**. There are a lot of benefits to the lack of complexity of the project, and it is estimated that the production will go smoothly. There are some notable risks, but they are not unique and can be easily mitigated. The biggest risk may be finding a reliable third-party company or companies to handle the database and website hosting servers. Still, this risk can be solved if considerable effort and time go into finding reliable companies.

4.0 Requirements Definition

Introduction

The following section will cover the **functional**, **non-functional**, and **data** requirements for this application. If a requirement is functional, it describes what the application should be doing. If a requirement is non-functional, it describes anything that isn't a functional requirement, usually an external constraint on the system. Data requirements describe ways the system must handle data.

Functional Requirements

- **Adding family nodes (1) –** Upon right-clicking a blank spot on the canvas, a pop-up will appear that allows the user to click a "Create family member" button. There will also be a UI button on the top of the screen with a plus symbol. Hovering over it will display "Create family member." Interacting either way will generate a family node on a blank spot on the canvas. This forms a horizontal rectangle with a portrait slot on the left, a bold name on the top right, and other information like gender and age under the name on the right. The user will immediately enter edit mode of a newly created family node.

- **Editing family nodes (2) –** The user can enter edit mode of any family node by right-clicking and selecting an "Edit" button on a pop-up or by double-clicking. Once in edit mode, the camera will zoom in, so the node fills up most of the screen space. All text information will become an inset and editable text box. This includes age, gender, and name. Age only accepts integers, and gender and name accept any string. There will be a small circle with a plus button on the top right of the portrait slot, and selecting it will open the user's files on the computer. The portrait will only accept PNG, JPEG, or WEBP file types. Once selected, the image will show up on the family node.

- **Creating lines to and from children (4, 12) –** When hovering over a family member node, there will be two small circle buttons with pluses on the top and bottom middle. Hovering over the top displays "Add parent" and hovering over the bottom displays "Add child." Clicking on either will draw a thick black line connected to the middle of the node that only travels in cardinal directions. At this point, the user is free to zoom out and pan around the screen. The line will follow their mouse. Once the user selects another family node, that node is set as a parent or child of the original node, depending on the button clicked. A family node cannot have a parent with a younger age or a child with an older age. By right-clicking a line in or out of edit mode, the user will be prompted with a "Change color" pop-up. This pop-up features eight colored circles, black, gray, red, orange, yellow, green, blue, and violet. Clicking any of these will change the color of the line.

- **Real-time organization (4) –** Once a family node is assigned a parent or child, it should automatically move to be in a horizontal line with all other children of the same parent or parents of the same child.

- **Dragging and dropping family nodes (5) –** Hovering over the edge of a family node outside of the edit mode will turn the cursor into a move symbol. The user can drag the family node around by holding left click and drop it into the desired location by releasing left click. If there are any lines to children or parents, they automatically update throughout the process.
- **Zoom and pan (6, 7) –** The user will be able to zoom into and out of the canvas with the scroll wheel. The user can also middle click and drag the mouse around or use WASD to pan around the canvas
- **Exporting as PDF, PNG, or JPEG (8) –** On the top right of the screen, there will be a button labeled "Export," and clicking it will open an "Export as:" pop-up that has "PDF," "PNG," and "JPEG" buttons. Clicking any of them will download an image of the family tree in its respective file type. The image is fit to encompass all family nodes with a margin of 32px at the highest, lowest, leftmost, and rightmost points.
- **Creating and logging into accounts (9, 10) –** On the top right of the screen, to the left of the "export" button, there is a "Log-in/Sign up" button. Pressing it will open a large pop-up locked in the middle of the screen. The pop-up is labeled "Log-in" at the top and has a text box for an email and a text box for a password. A small hyperlink at the bottom reads, "Don't have an account? Sign up!" Pressing it turns the log-in pop-up into a "Sign up" pop-up. It has a text box for an email, a password, and a confirm password.
- **Saving over the cloud (3) –** If the user is logged in, the system should automatically save their work to the cloud database every 30 seconds. If the user presses Ctrl+S, the tree should be saved. If the user presses Ctrl+S when not logged in, they will be prompted to log in.

Data Requirements

- **Custom saves –** Conventional data structures can not fully describe the family trees generated by the FTG. Although the core of the data is a tree, the fact that nodes can have multiple parents allows for much more complex relationships that need to be considered. Conventional tree data types will not be used on their own. It is also important that the backend database servers save not only the tree itself but the locations of the nodes and a coordinate system. This requirement will be further expanded upon in the System Specification Document.

- **Emails –** The FTG design requires user emails for account creation. These are in the database server, but their only purpose is to be tied to an account. These emails must not be abused internally to ensure users feel respected and happy using the FTG. Emails must not be sold to a third party, and we must not send out advertisements or software update information directly to the user emails. Users should only receive emails to confirm user accounts, reset passwords, inform the user of a log-in on a new device, or inform the user of a password reset.

- **Passwords –** With user account functionality, passwords are required. It is almost certain that users will have nothing confidential directly on their accounts. Still, it is crucial that passwords are encrypted, and appropriate security measures are considered. We can not trust that users will not use the same password for the FTG as their bank or other important institution. If we do not take proper care of their data and it gets leaked, real harm could come to someone, even if it has nothing to do with our system directly.

- **Cookies for Advertisers –** The advertiser service we use may want to use targeted advertising. This can vary depending on the service that is chosen, but it's something that should be considered.

Non-functional Requirements

Operational Requirements:

- The system is required to work on PCs. It should detect keyboard presses, mouse clicks, and mouse movement. It should be designed to fit on a 16:9 screen ratio but should also expand responsively to wider screen sizes.

- The system is required to work thoroughly on many different browsers. It should be tested on Google Chrome, Chromium, Mozilla Firefox, Microsoft Edge, Opera, Brave, and Safari. Each of which with varying screen sizes.

Advertisement Requirements:

- Advertisements must not obscure the workspace of the user in any way.

- Advertisements must not play background sound to the user.

- Advertisements must be ensured they are appropriate and safe for work.

- Advertisements must not pop up and disturb the user.

Protection Requirements:

- User passwords and emails must be encrypted in transit.
- All user data must be stored in a protected physical location.

5.0 Requirements Model

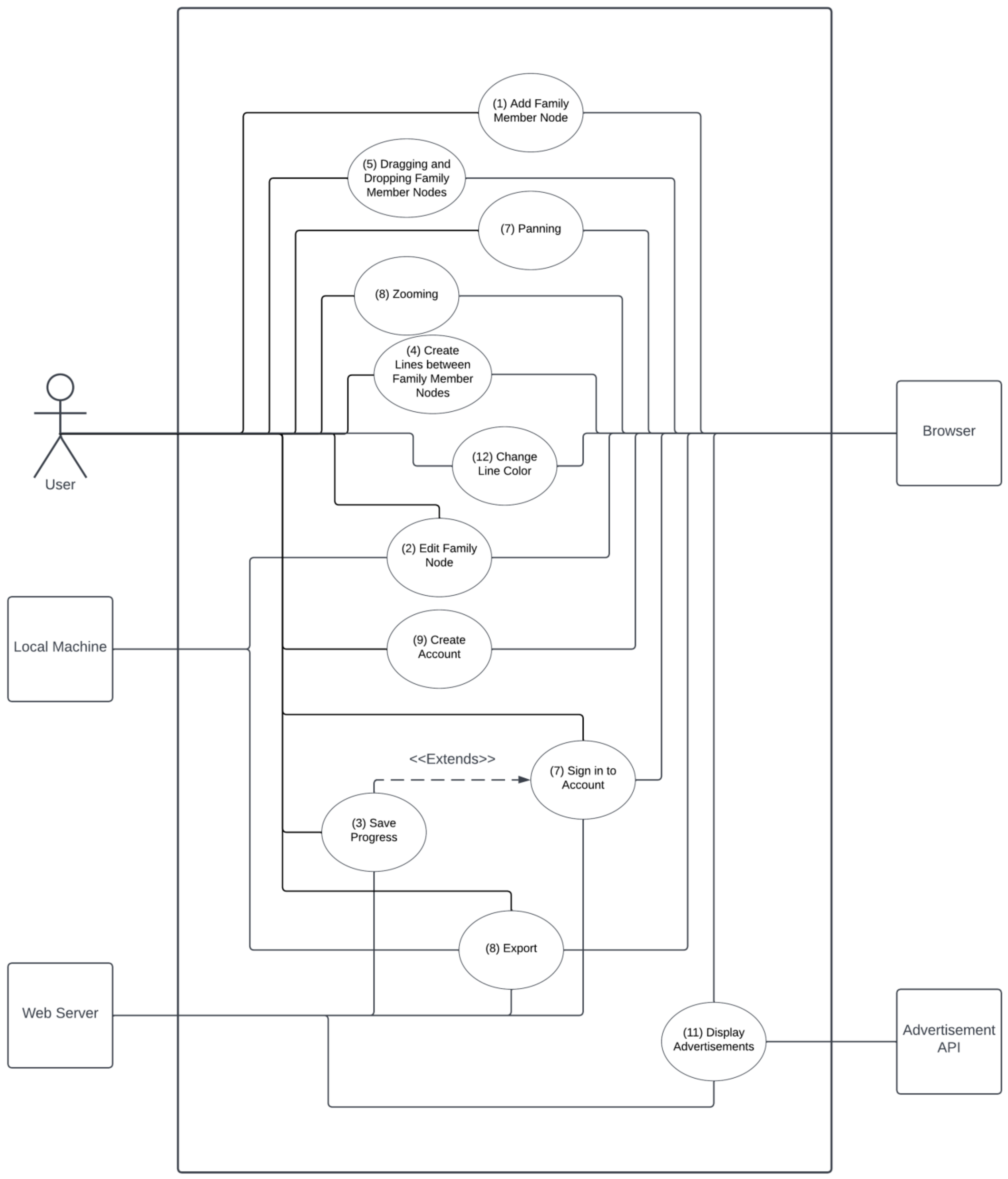
Introduction

Pages 19-40 cover the system in much closer detail than this document has previously. The overview is not as low-level as the System Specification but still gives more thorough descriptions of the functional requirements.

The Use Case Diagram is a grouping of 12 use cases and what actors they are directly affected by. Use cases are specific functions that the FTG will perform. Actors are external people, systems, or machines interacting with the FTG from the outside. Their relationships to the use cases are drawn by lines. Each use case has an ID number that is associated with it. This diagram is still relatively high level.

The Use Case Descriptions are much more in-depth descriptions of each use case. They go over which functions the system should perform, how it should perform it, and what actors are affecting it. They are ordered by the ID numbers of the use cases.

Use-Case Diagram



Use-Case Descriptions

Use Case Name: Add Family Node	**ID**: 1	**Importance**: High

Primary Actor: User	**Use Case Type**: Detail, Essential

Supporting Actors: Browser

Stakeholders and Interests:

User – They want to create a family member to attach to other family members. Without the ability to add family nodes the user cannot use the rest of the system.

SPU Computer Science Department – We want to make sure that the user is happy with the system, if this core function does not work well there is a risk to the user leaving and not creating advertisement revenue.

Server Hosting Company – If this core function does not work well, less people will use this system, and they may lose us as a customer.

Brief Description:

User creates an empty family node to later be edited and set as a parent or child.

Triggers:

Right click on an empty spot on the canvas and selecting "Add family member" from dropdown.

Left click on the "Add family member" button.

Type (mark one): _X_ External ___ Temporal

Relationships:

Association: User, Browser

Include: None

Extend: None

Generalization: None

The Normal Flow of Events:

1. User clicks the "Add family member" button on the top of the screen.
2. Empty family member node is generated in the middle of the user's screen.
3. User is automatically placed into edit mode, see use case 2.

Sub-flows:

2a. If there is no empty canvas in the middle of the screen, the node is generated at the spot closest to the center without overlapping on any other nodes.

Alternate/Exceptional Flows:

1. User right clicks an empty spot on the canvas where they want to add a family member.
2. User selects the "Add family member" button on drop-down list that appears.
3. Empty family member node is generated on the canvas at right click location.
4. User is automatically placed into edit mode, see use case 2.

Special Requirements: None

To do/Issues: None

Use Case Name: Edit Family Node	**ID**: 2	**Importance**: High
Primary Actor: User	**Use Case Type**: Detail, Essential	
Supporting Actors: Browser, Local Machine		
Stakeholders and Interests: User – They want to add information to the family member so that it is unique and has a meaningful relationship to other family members. SPU Computer Science Department – We want the user to be able to use the system, otherwise there was no point in making it. Server Hosting Company – If this core function does not work well, less people will use this system, and they may lose us as a customer.		
Brief Description: User edits a family node. They can change the name, gender, age, and upload a profile picture.		
Triggers: Double clicking on a family member node. Right clicking on a family node and selecting the "Edit family member" button in the dropdown. **Type** (mark one): _**X**_ External ___ Temporal		
Relationships: **Association**: User, Browser, Local Machine **Include**: None **Extend**: None **Generalization**: None		
The Normal Flow of Events: 1. User double clicks on the family member node. 2. Screen zooms in on the family member node. 3. All text becomes editable. 4. User adds a name. 5. User adds a gender. 6. User adds an age. 7. User uploads a profile picture to the family member node.		

8. User edits clicks off of the family node and exits edit mode.
Sub-flows: 6a. User presses the plus button on the profile picture. 6b. User's local files are opened. 6c. User selects a PNG, JPEG, or WEBP filetype. 6d. File is set as profile picture.
Alternate/Exceptional Flows: 1. User right clicks on family member node. 2. User selects the "Edit family member" button from dropdown list. 3. Continued from 2 on the normal flow.
Special Requirements: User must allow the website to have access to their local files.
To do/Issues: None

Use Case Name: Save Progress	**ID**: 3	**Importance**: High
Primary Actor: Web Server	**Use Case Type**: Detail, Essential	
Supporting Actors: User, Browser		
Stakeholders and Interests: User – The user may want to create a family tree that is too large to do in one session, or they may just not have very much time to spare. They will want to save their progress so they can come back later without having to keep a browser open in between sessions. User also has trust that their password information will be protected. SPU Computer Science Department – We are partly responsible for keeping the data safe and accessible. Server Hosting Company – They own the web and database servers that the system is running on, it costs them money to use their web and database servers for the system. They also are taking a risk that comes with storing potentially confidential information like passwords.		
Brief Description: User's progress is saved in the cloud. It is saved on the database servers connected to the web servers.		
Trigger: Automatically saves every 30 seconds. User presses Ctrl+S. **Type** (mark one): _**X**_ External _**X**_ Temporal		
Relationships: **Association**: Web Server, User, Browser **Include**: None **Extend**: Sign into Account **Generalization**: None		
The Normal Flow of Events: 1. Web server sends a pull request to signed-in session. 2. User's computer sends save information and account to the web server. 3. Web server stores it in database server. 4. Web server waits 30 seconds. 5. Web server repeats step 1.		

Sub-flows:

3a. Database server recieves save information.

3b. Database server frees previous save information from memory.

Alternate/Exceptional Flows:

1. User presses Ctrl+S.
2. User's computer sends save information and account to the web server.
3. Web server recieves save information and account.
4. Continue from step 3 on the normal flow.

1. User presses Ctrl+S.
2. User's computer sends save information and no account to the web server.
3. User is prompted to sign in, see use case 7.

1. User presses Ctrl+S.
2. User's computer attempts to send save information to the web server.
3. User's computer is not able to send information because there is no interent connection.
4. User recieves an error message telling them that they are not connected to the internet.

Special Requirements:

User must be connected to the internet.

User must be signed into an account.

Web servers and database servers must be operational.

To do/Issues: None

Use Case Name: Create Lines Between Family Nodes | **ID**: 4 | **Importance**: High

Primary Actor: User | **Use Case Type**: Detail, Essential

Supporting Actors: Browser

Stakeholders and Interests:

User – There is no point in a family tree if you can't draw meaningful relationships between its family members. The user wants to be able to create those relationships.

SPU Computer Science Department – We want the user to use our system, if this core function does not work well they may not use it.

Server Hosting Company – If this core function does not work well, less people will use this system, and they may lose us as a customer.

Brief Description:

User creates a parent/child relationship between two family nodes. This is visualized as a line appearing on the top of the child and connecting to the bottom of the parent.

Triggers:

Left click on the "Add parent" button.

Left click on the "Add child" button.

Type (mark one): _**X**_ External ___ Temporal

Relationships:

Association: User, Browser

Include: None

Extend: None

Generalization: None

The Normal Flow of Events:

1. User left clicks on the "Add parent" button on the desired family node.
2. A black line is drawn from the top middle of the family member node to the mouse location every frame.
3. User left clicks on another family member node.

4. The line is no longer drawn to the mouse location and is connected to the bottom middle of the clicked family member node.
5. The clicked family member node is set as the parent in the first family member node.
6. The first family member node is set as the child in the second family member node.
7. Family node lines up with the rest of the other children.

Sub-flows: None

Alternate/Exceptional Flows:

1. User left clicks on the "Add child" button on the desired family node.
2. A black line is drawn from the bottom middle of the family member node to the mouse location every frame.
3. User left clicks on another family member node.
4. The line is no longer drawn to the mouse location and is connected to the top middle of the clicked family member node.
5. The clicked family member node is set as the child in the first family member node.
6. The first family member node is set as the parent in the second family member node.
7. Family node lines up with the rest of the other parents.

Special Requirements: None

To do/Issues: None

Use Case Name: Dragging and Dropping Family Nodes	**ID**: 5	**Importance**: Moderate
Primary Actor: User	**Use Case Type**: Detail, Essential	
Supporting Actors: Browser		
Stakeholders and Interests: User – They want to reorganize the family tree in a way that is more uniquely meaningful. They can do this by manually moving and setting the family member nodes where they want them to go.		
Brief Description: User drags the family member node by holding left click, and it is set down when left click is let go.		
Trigger: Holding left click on a family member node. **Type** (mark one): _**X**_ External ___ Temporal		
Relationships: **Association**: User, Browser **Include**: None **Extend**: None **Generalization**: None		
The Normal Flow of Events: 1. User holds left click on family member node. 2. Family member node is drawn to the mouse location every frame. 3. User lets go of left click at desired location. 4. Family member node is set at desired location.		
Sub-flows: None		
Alternate/Exceptional Flows: 4. Family member node is set on top of another node. 5. Family member node is moved to the closest position on the canvas.		
Special Requirements: None		
To do/Issues: None		

Use Case Name: Zooming	**ID**: 6	**Importance**: Moderate
Primary Actor: User	**Use Case Type**: Detail, Essential	

Supporting Actors: Browser

Stakeholders and Interests:

User – It can be hard to comprehend the full scope of the family tree, so the user wants to be able to zoom out to be able to see the larger connections between nodes. They also want to be able to zoom in to get a closer look at specific nodes.

Brief Description:

User zooms in and out of the screen to a minumum and maximum zoom level using the scroll wheel.

Trigger:

Scroll wheel.

Type (mark one): _**X**_ External ___ Temporal

Relationships:

Association: User, Browser

Include: None

Extend: None

Generalization: None

The Normal Flow of Events:

1. User scrolls up.
2. Screen size decreases, level of detail increases.

1. User scrolls down.
2. Screen size increases, level of detail decreases.

Sub-flows: None

Alternate/Exceptional Flows:

1. User scrolls up.
2. Minimum zoom level has been reached, nothing happens.

1. User scrolls down. 2. Maximum zoom level has been reached, nothing happens.
Special Requirements: None
To do/Issues: None

Use Case Name: Panning | **ID**: 7 | **Importance**: High

Primary Actor: User | **Use Case Type**: Detail, Essential

Supporting Actors: Browser

Stakeholders and Interests:

User – They want to be able to see the family tree as a whole without compromising level of detail. They also want to be able to navigate to different sections of the tree.

Brief Description:

User holds middle or right click, mouse cursor locks onto the screen and user shifts entire view by moving mouse.

Triggers:

Holding middle click.

Holding right click.

Type (mark one): _**X**_ External ___ Temporal

Relationships:

Association: User, Browser

Include: None

Extend: None

Generalization: None

The Normal Flow of Events:

1. User holds middle or right click.
2. Mouse cursor is locked to the canvas.
3. Mouse cursor turns into pan-symbol.
4. User shifts view desired direction and amount.
5. User lets go of middle or right click.

Sub-flows: None

Alternate/Exceptional Flows: None

Special Requirements: None

To do/Issues: None

Use Case Name: Export	**ID**: 8	**Importance**: High
Primary Actor: User	**Use Case Type**: Detail, Essential	
Supporting Actors: Browser, Web Server, Local Machine		
Stakeholders and Interests: User – They will need to have access to the final product that they made, there would otherwise be no purpose to making the family tree. SPU Computer Science Department – We want the user to be happy with our system, if this core function does not work they will get upset at us because we wasted their time.		
Brief Description: The family tree as shown on screen is uploaded form the web server to the user's local computer as a PDF, PNG, or JPEG.		
Trigger: Selecting the "Export" button on the top of the screen. **Type** (mark one): _**X**_ External ___ Temporal		
Relationships: **Association**: User, Browser, Web Server, Local Machine **Include**: None **Extend**: None **Generalization**: None		
The Normal Flow of Events: 1. User clicks on the "Export" button on the top of the screen. 2. User selects "PDF," "PNG," or "JPEG" from a drop-down list. 3. Web server saves progress, see use case 3. 4. Web server generates a file of the save in the user's chosen filetype. 5. Family tree is downloaded by the user's computer in chosen filetype. It contains all content on screen with 32px margins to the farthest points on the left, right, top, and bottom.		
Sub-flows: None		
Alternate/Exceptional Flows:		

3. Website does not have access to user's local files. 4. Website requests access to local files. 5. User accepts. 6. Family tree is downloaded by the user's computer. It contains all content on screen with 32px margins to the farthest points on the left, right, top, and bottom.
Special Requirements: User must be connected to the internet. Website must have access to the user's local files. Web server and database server must be operational.
To do/Issues: None

Use Case Name: Create Account	**ID**: 9	**Importance**: High
Primary Actor: User	**Use Case Type**: Detail, Essential	
Supporting Actors: Browser, Web Server		

Stakeholders and Interests:

User – They would like to be able to save their progress, and that is not functional without an account.

SPU Computer Science Department – We want the customer to be able to use the system, if this f unction does not work well then we risk upsetting the user. If enough users create accounts, we can boast about that to future clients.

Server Hosting Company – They have to store the data on their database server and they need to access it with their web server. This use case contains the user's password, which is sensitive information. They are taking a risk that the information is going to be protected.

Brief Description:

User creates a personal account on the website tied to an email and a chosen password.

Triggers:

Selecting the "Log-in/Sign up" button on the top of the screen.

Type (mark one): _**X**_ External ___ Temporal

Relationships:

Association: User, Browser, Web Server

Include: None

Extend: None

Generalization: None

The Normal Flow of Events:

1. User selects the "Log-in/Sign up" button on the top of the screen.
2. A pop-up is generated in the middle of the screen.
3. User selects the "Don't have an account? Sign up!" link on the bottom of the pop-up.
4. User enters an email.
5. User enters a password.
6. User confirms password.
7. User selects "confirm" button.
8. Email, IP address, and password are encrypted.

9. Information is sent to the web server.
10. Web server confirms that email is not already associated with another account.
11. Email and password are stored locally in the database server.
12. User's current family tree is saved on the database server under their account, see use case 3.
13. User recieves message that their account was created.
14. User is automatically signed in, see use case 10.

Sub-flows: None

Alternate/Exceptional Flows:

9. Web server finds that email is already associated with another account.
10. User recieves error message that email is already in use.

Special Requirements:

User must be connected to the internet.

Web server and database server must be operational.

To do/Issues: None

Use Case Name: Sign into Account	**ID**: 10	**Importance**: High
Primary Actor: User	**Use Case Type**: Detail, Essential	
Supporting Actors: Browser, Web Server		
Stakeholders and Interests: User – The user may want to create a family tree that is too large to feasible do in one session or they may just not have very much time to spare. The ability to create an account is essential for saving capibilities. The user will also want to be able to sign in on different devices to continue working on their family tree. SPU Computer Science Department –		
Brief Description: User is able to sign in to their existing account		
Trigger: Selecting the "Log-in/Sign up" button on the top of the page. **Type** (mark one): _**X**_ External ___ Temporal		
Relationships: **Association**: User, Browser, Web Server **Include**: None **Extend**: None **Generalization**: None		
The Normal Flow of Events: 1. User selects the "Log-in/Sign up" button on the top of the screen. 2. A pop-up is generated in the middle of the screen. 3. User enters their email. 4. User enters their password. 5. Email, password, and IP address are encrypted and sent to the web server. 6. Web server verifies that the email is associated with an account in the database server. 7. Web server confirms password sent by the user matches the password in the database server. 8. User is signed into their account. 9. Previous family tree progress is reinstated in their browser.		
Sub-flows: None		

Alternate/Exceptional Flows: 7. Web server does not find email associated with any account in the database server. 8. User recieves an error message that the email or password is incorrect. 9. Encrypted password sent by user does not match password in database server. 10. User recieves an error message that the email or password is incorrect.
Special Requirements: User must already have an account. User must be connected to the internet. Web server and database server must be operational.
To do/Issues: None

Use Case Name: Display Advertisements	**ID**: 11	**Importance**: High
Primary Actor: Web Server	**Use Case Type**: Detail, Essential	
Supporting Actors: Browser, Advertisement API		

Stakeholders and Interests:

Advertisers – They want to advertise their product to more people.

Advertisement Service – Our system gives them a new opportunity to find more users for their advertisers to reach. They will take a small portion of the advertisement revenue.

SPU Computer Science Department – This is the main motivation of the FTG. The revenue from the advertisers will fund the servers, fund the development costs, and generate profit.

User – The user may see an advertisement that they are interested in. It is more likely they will have a worse experience because the advertisements distract them.

Brief Description:

The browser displays various advertisements from a third-party advertisement API.

Trigger:

Website connection.

Type (mark one): _**X**_ External ___ Temporal

Relationships:

Association: User, Browser, Web Server, Advertisement API

Include: None

Extend: None

Generalization: None

The Normal Flow of Events:

1. User connects to website.
2. Web server fetches advertisements from advertisement API.
3. Browser displays advertisements.

Sub-flows: None

Alternate/Exceptional Flows: None

Special Requirements:

User must be connected to the internet.
To do/Issues: None

Use Case Name: Change Line Color	**ID**: 12	**Importance**: Low
Primary Actor: User	**Use Case Type**: Detail, Essential	
Supporting Actors: Browser		
Stakeholders and Interests: User – It may be hard to keep track of which children belong to which parents if a family tree gets complex enough. Being able to color code the lines between nodes makes this much easier.		
Brief Description: Ability to change the color of a line between two family nodes. The user can select from black, gray, red, orange, yellow, green, blue, and violet.		
Trigger: Right click on line. **Type** (mark one): _**X**_ External ___ Temporal		
Relationships: **Association**: User, Browser **Include**: None **Extend**: None **Generalization**: None		
The Normal Flow of Events: 1. User right clicks on line. 2. User is prompted to select from black, gray, red, orange, yellow, green, blue, and violet. 3. User selects desired color. 4. Line color updates.		
Sub-flows: None		
Alternate/Exceptional Flows: None		
Special Requirements: None		
To do/Issues: None		

6.0 System Evolution

Most of the previous information in the document was aimed at the core functions required for the MVP. After the FTG is built and released, updates to it may be worthwhile. Some potential new functionalities may include:

- **Multiple save files –** The MVP version of the FTG only includes one save file. This is not predicted to be a significant issue because it's improbable that a user will need to work on multiple family trees simultaneously. Their family tree may become large enough to warrant a save file, but the user will probably be able to finish even a large family tree before making another one. That being said, it is still possible that the user will want to keep track of a family as it changes. In addition to this family tree, they may want to make other family trees.
- **Logging in with Meta or Google accounts –** It's not uncommon for people to be hesitant to create an account on a website they don't plan on using very much. Most people have many online accounts in various places, and adding to that list may feel cumbersome or annoying. However, almost everyone has a Google or Meta account. If we allow our system to link to these existing accounts, this problem may be mitigated.
- **Themes –** The current plan for the FTG will create functional and readable family trees, but they will not look stylized. Allowing custom themes will allow the user to add more personality to their family trees.
- **Downloadable file type for the user –** One problem with the projected version of the FTG is that the user requires internet access to export and save their work. If the user has been working on their family tree and did not realize they were offline, they may lose considerable progress on the family tree. If we develop a custom file type that the user can save locally, they will not have to worry about losing their internet connection in the middle of their work.
- **Investing in owning servers instead of renting –** Renting your server access has a lot of benefits in the short term, but that may not be the case in thinking long term. Outsourcing servers do not have the upfront investment cost of ownership but are still more expensive since the hosting company needs to make a profit. Owning servers will also give more control over the data sent by users.

7.0 Conclusions and Recommendations

Conclusions

Ultimately, this project has a lot of promise. While the problem has been solved before, many competitors offer convoluted systems, or users are forced to use an application that is not specialized in making family trees. Family trees are a significant demand because they are such a typical diagram worldwide. The project scope is also not too high to make the project unfeasible or financially risky.

There are more things going for the FTG than against it, but some risks still exist. They include but are not limited to a low user retention rate because of an infrequent need for family trees, data protections partially being handled by a third-party company, and a long-term financial burden of renting server hosting.

Recommendations

- Do not implement any form of a paid version. User retention is likely too low for this to be practical.
- Invest a lot of time and resources into finding the right third-party company to host the web and database servers. Picking the wrong company may lead to unnecessary server costs or inadequate data protection. The entire project may be jeopardized if the wrong third-party company is selected for this role.
- Consider buying internal web servers and database servers. While a third-party server hosting company is necessary for a start-up because of financial costs, if the FTG is successful enough, it's possible to do the server hosting ourselves. Outsourcing the servers is cheaper in the short term but will probably be more expensive in the long term. It also may be beneficial to reduce the number of independent actors working with the system.

Appendices

There was no reference material provided.

Glossary

Advertisement API – In this system it is refering to an external system that sends advertisements to be displayed by the browser on the website. It may also have functionality to track the user and send targetted advertisements.

Browser – The program on a computer that can read and interact with websites on the internet.

Cardinal – Strictly defined directions of north/up, east/right, south/down, and west/left.

Cloud – A way of storing data by sending it to a faraway physical database over the internet. It is not stored locally and can be fetched from multiple devices.

Family node – A family member that is displayed on the family tree. They connect to other family nodes as children or parents.

FTG – Family Tree Generator. Shorthand for the project being developed.

Hyperlink – Clickable text that the user can interact with. It often takes them to another web page or does something within their web page.

Move-symbol – A four-way symmetrical cross of a vertical and a horizontal double-sided arrow.

MVP – Minimum Viable Product. It is the version with just the core functionalities, often times the first one released.

Pan-symbol – A hand cursor that has all five fingers in a fist, as if it were grabbing.

Bibliography

Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative by Craig Larman. ISBN 978-0131489066.

Class Notes

Lectures by Professor Cameron.