VisualPro - Research Proposal

1st Given Edward Patch
Software Engineer Student (of BSc Year 3)
Independent Project
University of Wales Trinity St. Davids (of Mike Dacey)
Swansea, Wales
Student ID: 1801492

CONTENTS

I Introduction 1
I-A Potential growth to VisualPro 1
I-B Sections 1

II Literature Review 1
II-A Productivity and Visual Scripting 1
II-B Markup Languages 2

III Aims and Objectives 2
III-A Aims 2
III-B Objectives 3

IV Project Design 3
IV-A Outline of Project Process 3
IV-B Data Collection 3
IV-C Risks and Limitations 3
IV-D Ethical Issues 3

V Resources and Planning 3
V-A Resources Required 3
V-B Planning Chart 4
V-C Mindmap - Ideas 5
V-D Mindmap - VisualPro 6

VI Terminology 7

VII Appendices 7

VIII Reference List 7

***Abstract*— This document provides the proposal of the Independent Project. After doing in-depth research and planning, the working title for this project is VisualPro.**

Hypothesis: If visual scripting can write structure and logic of any desired language the user wishes, will this:-

- **A: produce a more productive and better work environment.**
- **B: help novices code in any language without knowing different syntax per language.**

***Index Terms*—Visual Programming, Proposal, Research.**

I. INTRODUCTION

VisualPro aims to be a lightweight visual programming pad, which includes the following features:-

- Program in any language with ease.
- Enables users to write code in seconds.
- A simple Graphical User Interface (GUI) that changes the way Visual Scripting already works.

The prototypes will demonstrate C/C++ libraries working together with a different language, which plays a massive part to archive this task.

A. Potential growth to VisualPro

The dynamics of the custom languages allow new scripting languages. Consequently, it implies but is not limited to, Notebook and Stylus companies benefiting from this product. Take a look at (reference page) for more information on this topic.

B. Sections

The VisualPro Proposal document includes a Literature Review, Aims and Objective, Project Design, Resources and Planning and Reference List.

The Literature Review section contains segments of academic books and articles related to the project proposal. Within the extensive research, hand-picked books and articles to show what technologies are required to make the idea plausible, narrowing down the project's association towards the Author's study area.

After the Literature Review section, the Aims and Objectives section will give the reader an idea of how the final product will impact society and its user base.

The third section, Project Design, describes the outline of the project process, data collection, risks and limitations and ethical issues. These are essential points that decide if the project breaks any laws or pushes too many boundaries in today's society.

The Resources and Planning section include two subsections; Resources Required and Planning Chart. Resources Required sub-section involves Physical Resources *(if any hardware and equipment are required)*, Human Resources *(if any staff are required)* and Other Resources *(if any software or other categories not been mentioned are required)*. The Planning Chart sub-section displays the tasks and time frame within a Gantt Chart, constructed by Microsoft Project [1].

II. LITERATURE REVIEW

A. Productivity and Visual Scripting

Multiple books are necessary to capture this question, "If visual scripting can write structure and logic of any desired language the user wishes, will this produce a more productive and better work environment". The research also helps understand why visual scripting or interpreted programming languages/scripts are more popular than compiled programming languages and how productivity expands with new access to different tools.

Productivity, specifically within the software engineering field, Rethinking Productivity in Software Engineering Book states, "Productivity is a challenging concept to define, describe, and measure for any knowledge work that involves nonroutine creative tasks. Software development is a prime example of knowledge work, as it too often involves poorly defined tasks relying on extensive collaborative and creative endeavors. As in other areas of knowledge work, defining productivity in software development has been a challenge facing both researchers and practitioners who may want to understand and improve it by introducing new tools or processes." When reading further into the chapter, the following points evaluate how we should think about productivity:-

- "Velocity: How fast works gets done."
- "Quality: How well works gets done."
- "Satisfaction: How satisfying the work is."

After thinking about this, VisualPro, in theory, should increase the rapidity of the individual's work. Suppose VisualPro changes the way Visual Scripting works today, making it lightweight and understandable compared to existing Visual Scripting (For example, Unreal Engine's Blueprints [2]), with the ability to generate any desired code. In that case, the quality and satisfaction of the developer amplify.

To compare the theory to conclude, **Hypothesis A** and **B**, page 1, two books, Dr. C. Sadowski and Dr. Thomas Zimmerman's [3] Book, "Rethinking Productivity in Software Engineering", and A. M. Winn and T. J. Smedley's [4] Book, "Multimedia Workshop: Exploring the Benefits of a Visual Scripting Language", helps decide the decision.

A: produce a more productive and better work environment.
B: help novices code in any language without knowing
different syntax per language.

After examining Chapter 1, "Are Slower Programmers Just More Careful?" [3], figure 1-3 is shown. The dialogue suggested that O symbols represent "...dynamically typed scripting" and **X** symbols represent "...statically typed programs" within the graph. The author was trying to show that script languages might produce productivity, as they require fewer lines of code, but programming languages offer more efficient execution time. In order to complete the analysis of **Hypothesis A**, it is essential to understand that the proposed idea should act as a script but generate a programming language whilst making an easy-to-use environment. This statement, including the figure shown, with the points mentioned, concludes that **Hypothesis A** is true if administered accurately.

A. M. Winn and T. J. Smedley's [4] Book, section 3.2, "The graphical user-interface", suggests that Multimedia Workshop (MMW) Visual Scripting is excellent for junior and experienced developers, "...MMW was carefully designed for easy and natural use...". The Author's of this Book mentions that a poor graphical user interface could "...hinder an assessment of the usefulness of a visual scripting language". To prove **Hypothesis B** effective on novices stands true on condition the software is executed correctly and is not complicated or chunky when designing software. This project requires an easy drag and drop method with a simple layout to meet **Hypothesis B**'s aim.

B. Markup Languages

What is a Markup Language?
A markup language is essentially a script, which consists of tags and attributes as Erik T. Ray's [5] Book implies in Chapter 2 (Markup and Core Concepts), page 49, "Just as skeletons give us vertebrates shape and structure, markup does the same for text." This quote implies that markup languages offer structures, principally for designs, layouts and arrangement responsibilities. Due to the nature of Markup Languages with the ease of adding tags and attributes, Markup Languages will do the heavy lifting of setting available libraries and syntax for each language.

How will Extensible Markup Language (XML) support custom languages in this project?
According to G. Costa and R. Ortale [6], Machine learning techniques for XML (co-)clustering by structure and-constrained phrases Book, Chapter 2.1, "*An **XML tree** is a rooted, labeled, tree* $\mathbf{t} = (\mathbf{V^t}, \mathbf{r^t}, \mathbf{E^t}, \lambda^t)$ *where:*

- $\mathbf{V^t} \subseteq \mathbb{N}$ is a set of nodes, with $\mathbf{r^t} \in \mathbf{V^t}$ being the root of **t**;
- $\mathbf{E^t} \subseteq \mathbf{V^t} \times \mathbf{V^t}$ is a set of edges, catching parent-child relationships between nodes of **t**;
- $\lambda : \mathbf{V^t} \rightarrow \sum$ is a labelling function, with $\sum$ being the domain of the node tags (or equivalently, labels)."

The facility of having nodes and attributes means we can expand the XML file with many different syntaxes for different opportunities. If nested nodes are within the parent nodes, then it will increase the performance time of execution. The library that reads through the XML document will only need to select the relevant sub-nodes listed under the parent node instead of reading through every possible node outcome. In theory, users can append languages by reading through the XML document quite efficiently, and the software would read through the XML document without any issues.

III. Aims and Objectives

A. Aims

VisualPro software aims to increase existing software engineers, web developers, and other users in different professions. Productivity would increase and allow individuals to code in many languages without understanding different languages. The software aims to provide custom languages by a markup language with an easy-to-grasp script. Read section II Literature Review, sub-section II-B, page 2 to understand markup language.

B. Objectives

To achieve the aims mentioned in section (a reference in aims and objectives), sub-section (a reference in arms). In order to meet the aims during the production process of the project, the following objectives is a must. These objectives are as follows:-

- Project planning (section: V Resources and Planning, sub-section: V-B Planning Chart, page 4).
- Create prototypes (Existing programs [7] and libraries [8] to make this possible.).
- Conduct surveying, backend unit testing and front-end user interface testing relating to 'Program in any language with ease' section I, page 1.
- Create a front-end to work with existing libraries, which helps recreate Visual Scripting technology.
- Create a backend to communicate with the front-end and generate code based on values the user enters, which should help the user to compose code in seconds.

IV. Project Design

A. Outline of Project Process

Outline of the project process:-

- Design and produce a front-end to dock the backend libraries.
- Convert existing libraries and programs to Dynamic-Link Library (DLL) archives that work with the front-end.
- Create a LanguageCompiler (similarly to Programming Planner [7]) but with custom languages (XML).
- Design and create an XML script that tells the backend of what languages are available. The script must be understandable by the user.
- Use C *_declspec(dllexport)* [9] to export entry points within the DLL libraries.

B. Data Collection

Survey data will not ask for the personal details of the surveyed as it is not applicable data. When using the software, VisualPro, user data is not needed to operate this application. However, if the software is enhanced, author information may happen to be recorded in the future.

C. Risks and Limitations

The risks of the project are limited since no personal data is collected. Penetrations through the software are unlikely as it does not require any online connection or database to post or retrieve any information. Any communication from the front-end to the backend is restricted and limited, so no commands can be sent to delete any critical data. However, as the front-end will offer a File Explorer, Save Dialog, to save the code file, the user can still delete any necessary data if the user loaded the application with elevated privileges.

Limitations of the software will include the following due to time frame issues:-

- Code Logic (inc. Loop and Conditional Logic.) -
  The software will have the implementation space to allow this in further development. Unfortunately, only multiple languages, the structure the code including parent-child logic and the front-end, can be covered to the time frame.
- Nested Logic -
  Again, this will use very similar logic to the Parent-Child syntax for classes, functions and variables. However, these will ideally fit the time frame when the code logic feature implements in the software.

When creating the backend and front-end of this software, reusable functions are a must. Without reusable functions, the schedule will get delayed. Reusable functions also enable the developer's working on VisualPro the ability to implement features agile and precise.

D. Ethical Issues

The software proposes an increase in productivity. However, screen time still matters, and the user needs to rest when feeling strained in front of the screen. The idea of VisualPad is to improve the work environment for many different fields in a comfortable, peaceful and healthy manner.

V. Resources and Planning

A. Resources Required

Physical Resources

Requirements of physical resources are research material, technical equipment and office equipment, like paper and pens. In order to plan a project like this, pen and paper is a must. Some problems are harder to tackle without pen and paper planning, especially if unique problems appear. Other physical resources also include a computer system to carry out the project. Take a look at the Project Schedule, sub-section V-B Planning Chart, page 4.

Human Resources

This project would usually require a Project Manager, Designer, Developer and a Tester. Ideally, a four-member team would usually be required to create a much more efficient project. However, the Project Manager of this project, Edward Patch, will be doing the four suggested roles to gain broad experience managing a project, making a more desirable leader in a project environment. Sub-section V-B Planning Chart, page 4, contains the work allocation for each role of this project, i.

Other Resources

Software is the key to this project. Source control is a must to keep track of the project and showcase any premade prototypes or end applications created during this process. A development platform is ideal for creating the project contents itself. This platform must fit a C/C++ backend and a suitable front-end environment with a code editor, compiler and GUI designer. Visual Studio supports many different languages and frameworks from C++, C# and JavaScript.

Software Required:

- GitHub [10] (Cloud).
- Git [11] (Source Control).
- Visual Studio [12] (Development Platform).

B. Planning Chart

The schedule contains seven milestones *(deadlines inside brackets)*; Proposal *(18/10/21)*, Produce Prototypes *(03/01/22)*, Surveying *(12/01/22)*, Frontend *(09/02/22)*, Backend *(15/03/22)*, Final Report *(12/04/22)* and Exhibition *(28/04/22)*. The important deadlines are 18/10/21, Proposal *(leaving a contingency of 0 days)*, 28/04/22, Final Report *(leaving a contingency of 16 days)* and 06/05/22, Exhibition *(leaving a contingency of 8 days)*. In total, 25 days are available for the contingency plan if anything gets off course. To understand the schedule better, make sure to browse the Schedule of VisualPro Microsoft Project Schedule [13] and Evidence of the Microsoft Word Schedule [14].

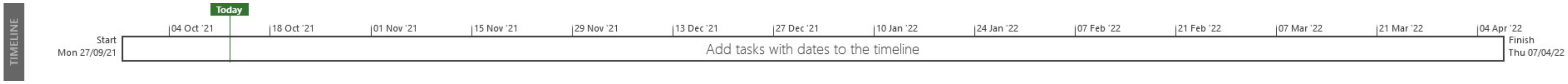
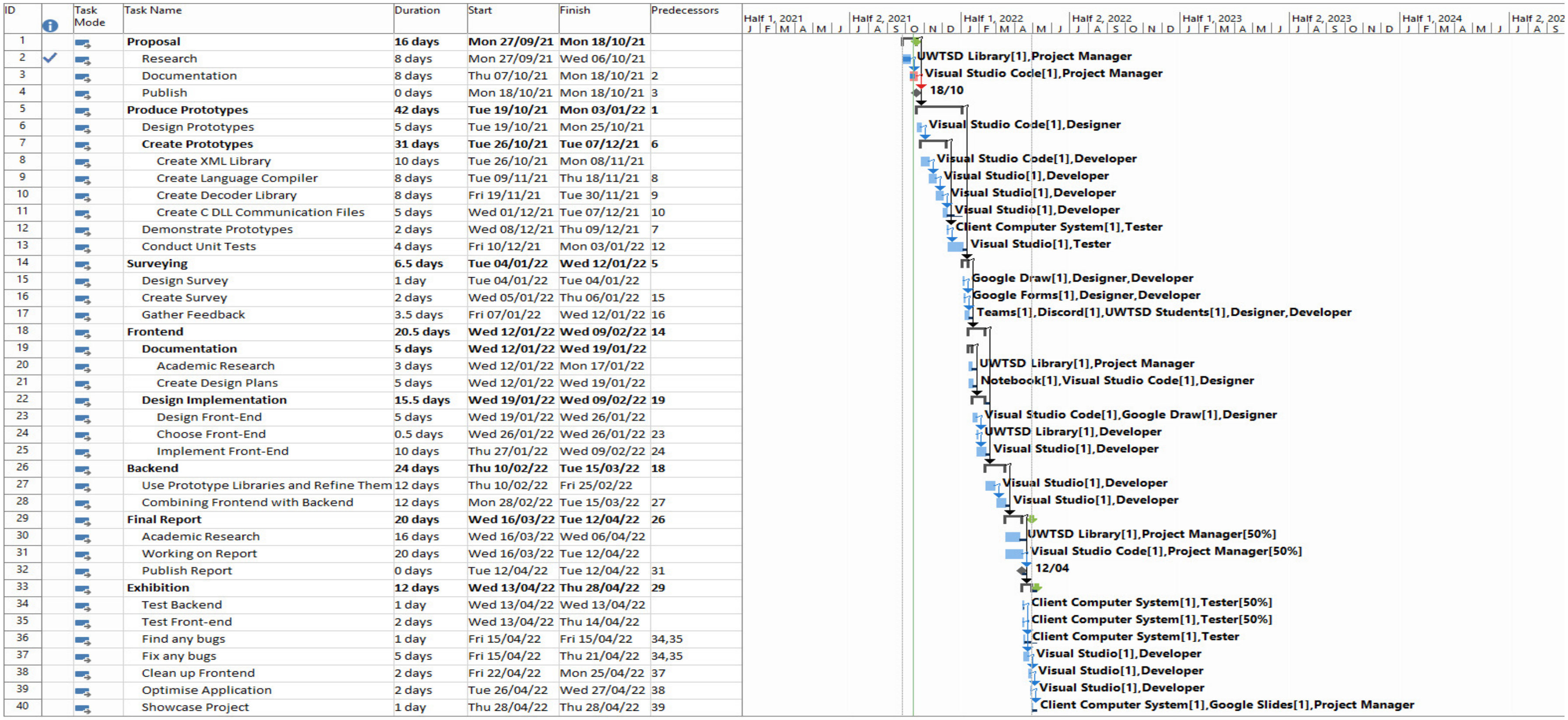


Fig. 1. VisualPro Timeline



ID		Task Mode	Task Name	Duration	Start	Finish	Predecessors
1			**Proposal**	**16 days**	**Mon 27/09/21**	**Mon 18/10/21**	
2	✓		Research	8 days	Mon 27/09/21	Wed 06/10/21	
3			Documentation	8 days	Thu 07/10/21	Mon 18/10/21	2
4			Publish	0 days	Mon 18/10/21	Mon 18/10/21	3
5			**Produce Prototypes**	**42 days**	**Tue 19/10/21**	**Mon 03/01/22**	**1**
6			Design Prototypes	5 days	Tue 19/10/21	Mon 25/10/21	
7			**Create Prototypes**	**31 days**	**Tue 26/10/21**	**Tue 07/12/21**	**6**
8			Create XML Library	10 days	Tue 26/10/21	Mon 08/11/21	
9			Create Language Compiler	8 days	Tue 09/11/21	Thu 18/11/21	8
10			Create Decoder Library	8 days	Fri 19/11/21	Tue 30/11/21	9
11			Create C DLL Communication Files	5 days	Wed 01/12/21	Tue 07/12/21	10
12			Demonstrate Prototypes	2 days	Wed 08/12/21	Thu 09/12/21	7
13			Conduct Unit Tests	4 days	Fri 10/12/21	Mon 03/01/22	12
14			**Surveying**	**6.5 days**	**Tue 04/01/22**	**Wed 12/01/22**	**5**
15			Design Survey	1 day	Tue 04/01/22	Tue 04/01/22	
16			Create Survey	2 days	Wed 05/01/22	Thu 06/01/22	15
17			Gather Feedback	3.5 days	Fri 07/01/22	Wed 12/01/22	16
18			**Frontend**	**20.5 days**	**Wed 12/01/22**	**Wed 09/02/22**	**14**
19			**Documentation**	**5 days**	**Wed 12/01/22**	**Wed 19/01/22**	
20			Academic Research	3 days	Wed 12/01/22	Mon 17/01/22	
21			Create Design Plans	5 days	Wed 12/01/22	Wed 19/01/22	
22			**Design Implementation**	**15.5 days**	**Wed 19/01/22**	**Wed 09/02/22**	**19**
23			Design Front-End	5 days	Wed 19/01/22	Wed 26/01/22	
24			Choose Front-End	0.5 days	Wed 26/01/22	Wed 26/01/22	23
25			Implement Front-End	10 days	Thu 27/01/22	Wed 09/02/22	24
26			**Backend**	**24 days**	**Thu 10/02/22**	**Tue 15/03/22**	**18**
27			Use Prototype Libraries and Refine Them	12 days	Thu 10/02/22	Fri 25/02/22	
28			Combining Frontend with Backend	12 days	Mon 28/02/22	Tue 15/03/22	27
29			**Final Report**	**20 days**	**Wed 16/03/22**	**Tue 12/04/22**	**26**
30			Academic Research	16 days	Wed 16/03/22	Wed 06/04/22	
31			Working on Report	20 days	Wed 16/03/22	Tue 12/04/22	
32			Publish Report	0 days	Tue 12/04/22	Tue 12/04/22	31
33			**Exhibition**	**12 days**	**Wed 13/04/22**	**Thu 28/04/22**	**29**
34			Test Backend	1 day	Wed 13/04/22	Wed 13/04/22	
35			Test Front-end	2 days	Wed 13/04/22	Thu 14/04/22	
36			Find any bugs	1 day	Fri 15/04/22	Fri 15/04/22	34,35
37			Fix any bugs	5 days	Fri 15/04/22	Thu 21/04/22	34,35
38			Clean up Frontend	2 days	Fri 22/04/22	Mon 25/04/22	37
39			Optimise Application	2 days	Tue 26/04/22	Wed 27/04/22	38
40			Showcase Project	1 day	Thu 28/04/22	Thu 28/04/22	39

Half 1, 2021
Half 2, 2021
Half 1, 2022
Half 2, 2022
Half 1, 2023
Half 2, 2023
Half 1, 2024
Half 2, 202
J F M A M J J A S O N D J F M A M J J A S O N D J F M A M J J A S O N D J F M A M J J A S
UWTSD Library[1],Project Manager
Visual Studio Code[1],Project Manager
18/10
Visual Studio Code[1],Designer
Visual Studio Code[1],Developer
Visual Studio[1],Developer
Visual Studio[1],Developer
Visual Studio[1],Developer
Client Computer System[1],Tester
Visual Studio[1],Tester
Google Draw[1],Designer,Developer
Google Forms[1],Designer,Developer
Teams[1],Discord[1],UWTSD Students[1],Designer,Developer
UWTSD Library[1],Project Manager
Notebook[1],Visual Studio Code[1],Designer
Visual Studio Code[1],Google Draw[1],Designer
UWTSD Library[1],Developer
Visual Studio[1],Developer
Visual Studio[1],Developer
Visual Studio[1],Developer
UWTSD Library[1],Project Manager[50%]
Visual Studio Code[1],Project Manager[50%]
12/04
Client Computer System[1],Tester[50%]
Client Computer System[1],Tester[50%]
Client Computer System[1],Tester
Visual Studio[1],Developer
Visual Studio[1],Developer
Visual Studio[1],Developer
Client Computer System[1],Google Slides[1],Project Manager

Fig. 2. VisualPro Gantt Chart

C. *Mindmap - Ideas*

This mindmap demonstrates the process and current experience, which helped get the idea for the final project.

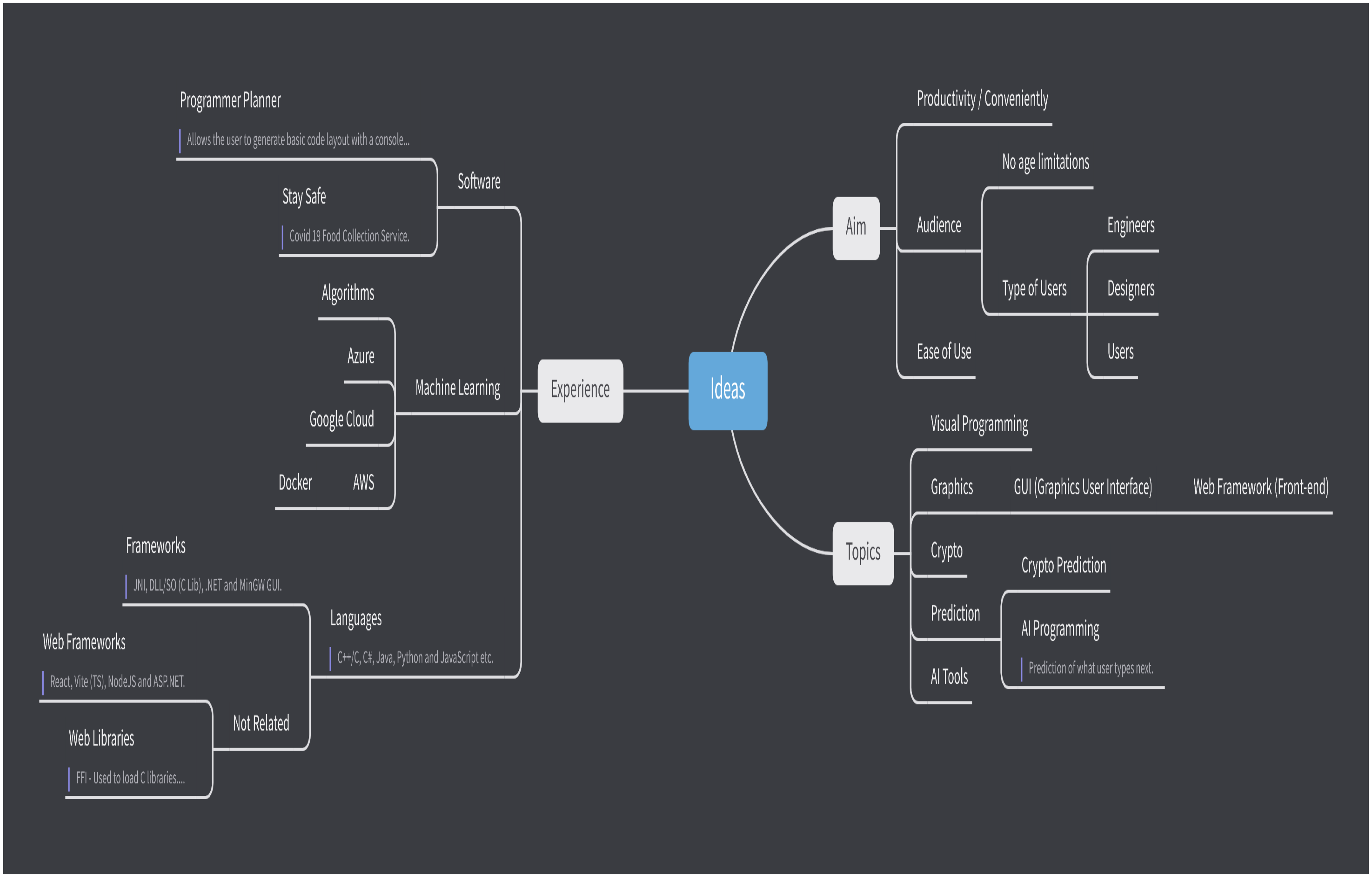

Ideas
Experience
Software
Programmer Planner
Allows the user to generate basic code layout with a console...
Stay Safe
Covid 19 Food Collection Service.
Machine Learning
Algorithms
Azure
Google Cloud
Docker
AWS
Languages
C++/C, C#, Java, Python and JavaScript etc.
Frameworks
JNI, DLL/SO (C Lib), .NET and MinGW GUI.
Not Related
Web Frameworks
React, Vite (TS), NodeJS and ASP.NET.
Web Libraries
FFI - Used to load C libraries....
Aim
Productivity / Conveniently
Audience
No age limitations
Type of Users
Engineers
Designers
Users
Ease of Use
Topics
Visual Programming
Graphics
GUI (Graphics User Interface)
Web Framework (Front-end)
Crypto
Prediction
Crypto Prediction
AI Programming
Prediction of what user types next.
AI Tools

Fig. 3. VisualPro Ideas Mindmap

D. Mindmap - VisualPro

This mindmap displays existing and future libraries or problems within the final project that may prove resourceful within the development of VisualPro.

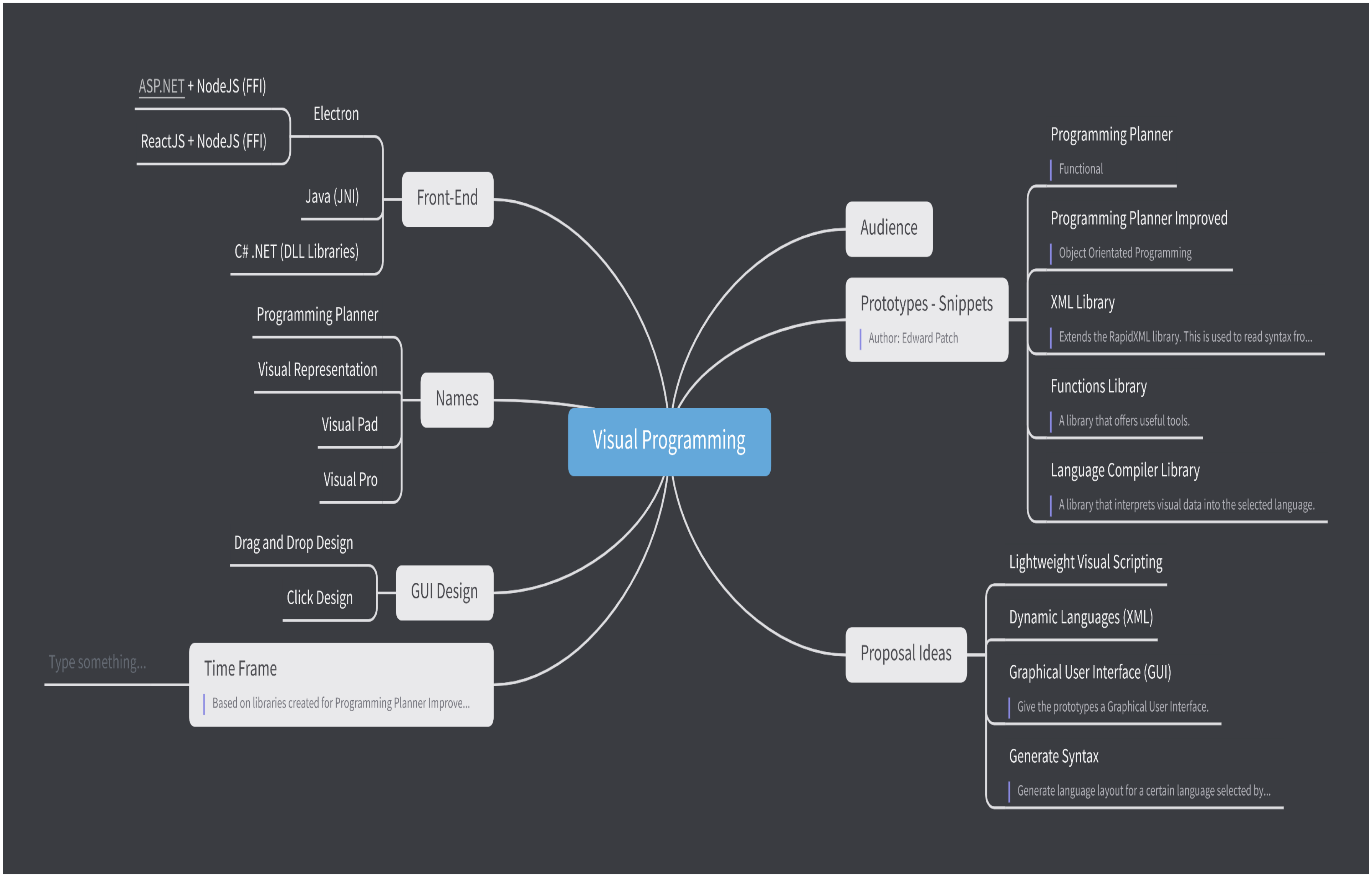

Visual Programming
Front-End
Electron
ASP.NET + NodeJS (FFI)
ReactJS + NodeJS (FFI)
Java (JNI)
C# .NET (DLL Libraries)
Names
Programming Planner
Visual Representation
Visual Pad
Visual Pro
GUI Design
Drag and Drop Design
Click Design
Time Frame
Based on libraries created for Programming Planner Improve...
Type something...
Audience
Prototypes - Snippets
Author: Edward Patch
Programming Planner
Functional
Programming Planner Improved
Object Orientated Programming
XML Library
Extends the RapidXML library. This is used to read syntax fro...
Functions Library
A library that offers useful tools.
Language Compiler Library
A library that interprets visual data into the selected language.
Proposal Ideas
Lightweight Visual Scripting
Dynamic Languages (XML)
Graphical User Interface (GUI)
Give the prototypes a Graphical User Interface.
Generate Syntax
Generate language layout for a certain language selected by...

Fig. 4. VisualPro Mindmap

VI. Terminology

List of terminologies used in this document:-

- GUI - Graphical User Interface.
- XML - Extensible Markup Language.
- DLL - Dynamic-Link Library.
- MMW - Multimedia Workshop.

VII. Appendices

VIII. Reference List

[1] Microsoft, “Compare Project Management Solutions and Costs | Microsoft Project.” [Online]. Available: https://www.microsoft.com/en-gb/microsoft-365/project/compare-microsoft-project-management-software

[2] Unreal Engine, “Introduction to Blueprints.” [Online]. Available: https://docs.unrealengine.com/4.27/en-US/ProgrammingAndScripting/Blueprints/GettingStarted/

[3] Dr. Caitlin Sadowski and Dr. Thomas Zimmerman, Eds., *Rethinking Productivity in Software Engineering*, 1st ed. Apress, 2019. [Online]. Available: https://doi.org/10.1007/978-1-4842-4221-6

[4] A. M. Winn and T. J. Smedley, “Multimedia Workshop: Exploring the Benefits of a Visual Scripting Language.” IEEE Computer Society, Sep. 1998, pp. 280–280. [Online]. Available: https://www.computer.org/csdl/proceedings-article/vl/1998/87120280/12OmNBl6EH3

[5] E. T. Ray, *Learning XML*, 2nd ed. Sebastopol, California: O’REILLY.

[6] G. Costa and R. Ortale, “Machine learning techniques for XML (co-)clustering by structure-constrained phrases,” *Information Retrieval*, vol. 21, no. 1, pp. 24–55, Feb. 2018, num Pages: 24-55 Place: Dordrecht, Netherlands Publisher: Springer Nature B.V. [Online]. Available: http://www.proquest.com/docview/2002401888/abstract/100992F450AA4EEFPQ/1

[7] E. Patch, “Programming Planner Software,” Oct. 2021, original-date: 2021-09-24T08:51:54Z. [Online]. Available: https://github.com/ShinkuKira21/VisualPro-FinalProject

[8] ——, “Libraries,” Oct. 2021, original-date: 2021-09-24T08:51:54Z. [Online]. Available: https://github.com/ShinkuKira21/VisualPro-FinalProject

[9] Microsoft, “Exporting from a DLL Using __declspec(dllexport),” May 2019. [Online]. Available: https://docs.microsoft.com/en-us/cpp/build/exporting-from-a-dll-using-declspec-dllexport

[10] GitHub, “GitHub: Where the world builds software.” [Online]. Available: https://github.com/

[11] Git, “Git.” [Online]. Available: https://git-scm.com/

[12] Microsoft, “Visual Studio: IDE and Code Editor for Software Developers and Teams.” [Online]. Available: https://visualstudio.microsoft.com/

[13] E. Patch, “VisualPro Project Plan,” Sep. 2021. [Online]. Available: https://github.com/ShinkuKira21/VisualPro-FinalProject/blob/main/Project/VisualPro.mpp

[14] ——, “VisualPro Plan Evidence,” Sep. 2021. [Online]. Available: https://github.com/ShinkuKira21/VisualPro-FinalProject/blob/main/Project/Plan.docx