“Toll and Error”

CS39440 Web Based Gameification Project Report

Author: Damien Xavier Phillips ([bep36@aber.ac.uk](mailto:bep36@aber.ac.uk))

Supervisor: Dr Edore Akpokodje (eta@aber.ac.uk)

25th March 2023

Version 1.0 (Draft)

This report is submitted as partial fulfilment of a BSc degree in  
Computer Graphics, Vision and Games (G450)

Department of Computer Science

Aberystwyth University

Aberystwyth

Ceredigion

SY23 3DB

Wales, UK

Declaration of originality

I confirm that:

* This submission is my own work, except where clearly indicated.
* I understand that there are severe penalties for Unacceptable Academic Practice, which can lead to loss of marks or even the withholding of a degree.
* I have read the regulations on Unacceptable Academic Practice from the University’s Academic Registry (AR) and the relevant sections of the current Student Handbook of the Department of Computer Science.
* In submitting this work, I understand and agree to abide by the University’s regulations governing these issues.

Name: Damien Xavier Phillips

Date: 25/04/2023

Consent to share this work

By including my name below, I hereby agree to this project's report and technical work being made available to other students and academic staff of the Aberystwyth Computer Science Department.

Name …………………………………………

Date ……………………………………………

Acknowledgements

I am grateful to my supervisor, Dr Edore Akpokodje, for all of his help in completing this project. I couldn’t have done any of this without his support.

I’d like to thank everyone at the Ceredigion Museum for meeting with me on multiple occasions to discuss the project and for providing further information on the historical events in the game.

I would also like to thank my boyfriend, Nathan, for his continued support throughout this project, and beyond it, as well as my furry and scaly friends, Poppy, Molly, Wing Commander the Lord Flashheart, and Mara, back at home for their companionship.

Abstract

“Toll and Error” is an application that aims to educate visitors to the Ceredigion Museum, and to visitors on the website, about the historical events of the Rebecca Riots in a fun and engaging way. The Rebecca Riots, which were a series of riots that took place in Wales in the 19th century, specifically in 1839-1845 these riots were in response to the toll fees that were charged to residents for use of their own local roads. These riots involved the rioters all dressing in traditional Welsh women’s wear and refusing to pay the tolls at the gates, and in some cases tearing them down. These rioters, who were predominantly men, called themselves “Rebecca”, or, more commonly, the “Daughters of Rebecca.” These riots were an important moment in Welsh history, and their story is still told to this day.

Interactive museum displays have become very popular, as they are, in the experiences of many, an excellent way to display information and convey it in a manner that is engaging, memorable, and that makes the museum experience fun and accessible to all. “Toll and Error” is an excellent example of such an exhibit, as it allows visitors to “step in to the story,” so to speak, and take on the role of a toll booth operator in an exaggerated, over the top, “Horrible Histories” style universe.

Once fully implemented, the “Toll and Error” game will be available on the museum’s website, allowing visitors to play and learn about the Rebecca Riots from the comfort of their own homes. In addition, the game may also be displayed in the museum itself, allowing visitors to engage with the exhibit in a more immersive way, as they will be able to see the toll board in question standing in front of them as they play.

Contents

[1. Background, Analysis & Process 6](#_Toc133591710)

[1.1. Background 6](#_Toc133591711)

[1.1.1. The Ceredigion Museum 6](#_Toc133591712)

[1.1.2. The History of the Rebecca Riots, and other miscellaneous research. 6](#_Toc133591713)

[1.2. Analysis 7](#_Toc133591714)

[1.3. Process 7](#_Toc133591715)

[2. Design 8](#_Toc133591716)

[2.1. Overall Architecture 8](#_Toc133591717)

[2.2. Detailed Design 8](#_Toc133591718)

[2.2.1. Even More Detail 8](#_Toc133591719)

[2.3. User Interface Design 8](#_Toc133591720)

[2.4. Other Relevant Sections 8](#_Toc133591721)

[3. Implementation 9](#_Toc133591722)

[4. Testing 10](#_Toc133591723)

[4.1. Overall Approach to Testing 10](#_Toc133591724)

[4.2. Automated Testing 10](#_Toc133591725)

[4.2.1. Unit Tests 10](#_Toc133591726)

[4.2.2. User Interface Testing 10](#_Toc133591727)

[4.2.3. Stress Testing 10](#_Toc133591728)

[4.2.4. Other Types of Testing 11](#_Toc133591729)

[4.3. Integration Testing 11](#_Toc133591730)

[4.4. User Testing 11](#_Toc133591731)

[5. Critical Evaluation 12](#_Toc133591732)

[6. References 13](#_Toc133591733)

[7. Appendices 14](#_Toc133591734)

[A. Third-Party Code and Libraries 15](#_Toc133591735)

[B. Code Samples 16](#_Toc133591736)

# Background, Analysis & Process

## Background

To prepare for the project, research was put into the following areas:

* The Ceredigion Museum
* The History of the Rebecca Riots
* GameMaker 2

Similar systems were also researched, such as other interactive museum displays in the Eureka! Science and Discovery Centre [1] and the Xplore! Science Discovery Centre [2], as well as games and shows of similar styles. “Horrible Histories” [3] was researched to assist with the playful, silly tone of the game, while “Papers, Please” [4] was researched to help with gameplay, as it is a critically acclaimed example of the gameplay features of the type of game that was to be created.

### The Ceredigion Museum

At the beginning of this project, it was essential to research thoroughly and gather as much information as possible. It was important for those involved to visit the Ceredigion Museum and view their displays in person. This allowed us to gain a more thorough understanding of the exhibits there, and provided a lot of inspiration for the final project.

During the visit to the museum, many “star exhibits” were shown as options for the project. This provided a wide variety of ideas to consider. After much discussion, it was decided that the final game would be a narrative-heavy game that took after the gameplay of “Papers, Please” and the tone of “Horrible Histories” called “Toll and Error” that focused on two star exhibits: The traditional Welsh attire and the toll board.

The decision was made to focus on these two exhibits was based on their local significance, due to the fact that the board is from nearby, and the fact that the story of the Rebecca riots piqued the interest of the team. The Welsh clothing exhibit showed beautiful traditional attire worn by women in Wales during the time period, and the toll board shows the toll fees charged during the 19th century, prompting the Rebecca Riots.

A second visit to the museum was then made to showcase the progress made on the games concept and storyline. This allowed discussion regarding a bilingual script, which then led to the English script being sent to the museum staff for translation.

### The History of the Rebecca Riots, and other miscellaneous research.

After a decision was made regarding the game’s topic, research then had to be done on the history of the riots. A lot of time was spent researching the event via articles online, podcasts, and discussion with the museum staff. This was quite an enjoyable part of the project due to the very interesting nature of the event and its prevalence in Welsh history.

In addition to researching the history of the riots, this phase also involved looking at the social and cultural context of the time. Learning about this allowed moving forward to look into how to make the tone of the script fun for all audiences. This led to looking into specific sketches from the BBC’s educational sketch comedy “Horrible Histories”. The specific sketch that inspired many parts of the script was the “Measly Middle Ages – Pay Rise” sketch that involved a medieval peasant approaching the knight he worked for to ask for a pay rise due to the plague wiping out many of them.

The team also worked closely with museum staff to ensure that the game accurately represented the exhibits and historical events depicted in the game. The staff provided valuable insight and expertise, as well as the aforementioned Welsh translation of the script.

### GameMaker2

When developing a game, one of the critical decisions that need to be made is selecting the right game engine. For the “Toll and Error” game, the team researched different game engines to determine which one would be the best fit for the project. After careful consideration, the team decided to use GameMaker, as it looked easy to use and had the features that would allow them to create an engaging and interactive game.

To test that the software was fit for purpose, a tester game was made. This enabled exploration of the games features, and finalised the decision to use GameMaker2 over Unity to create the game.

There were challenges encountered in the process of the game, such as the more advanced features of GameMaker, and trying to code things in GML, GameMaker’s text based programming language, that weren’t possible with GML Visual, GameMaker’s “code block” style programming language.

Deciding on, and subsequently learning to use GameMaker was critical in the development of the game.

## Analysis

The issue presented was that of a museum looking for a web-based gamification of their exhibits. To solve this problem, the task had to be broken down into smaller parts to make the whole thing more manageable. There was only one proposed approach to this, which was to first visit the museum, gain knowledge on the exhibits, decide on a topic for the project, research the topic extensively to provide historical accuracy and a compelling, entertaining game that’s tonally appropriate for all ages and abilities. The next step was to research how the game was actually going to be created. The game engine had to be decided, and then had to be researched, as despite being somewhat familiar with the game engine Unity from the “Computer Graphics and Games” module from the previous semester, prior experience with the software was still very limited, as before September 2022, the user had not been exposed to any game development software whatsoever, so was still very much a beginner in that regard.

Security is a very important issue when faced with a computing project, especially one that interfaces with the public and works with people directly, such as in the case of “Toll and Error”. However, “Toll and Error” does not take any data from users whatsoever, nor does it save anything between playthroughs of the game. Each run is completely independent from the last, with the only differences coming from decisions made within the game itself, such as language choice and decisions made by the player character. So, while still very important, security wasn’t much of a concern for this project.

The goal of this project was to present an original script written by the team via the means of a narrative game published via GameMaker’s own hosting service, GX.games. The smaller goals of this project to reach the final, ultimate end goal are:

* To gain an understanding of the Rebecca Riots for the purpose of writing a sensible script.
* To create a simple, mock-up framework for the game, for the purposes of showing both developers and museum staff how the game will look at the end of development.
* To create original graphics for the game based on the mock-up design.
* To write an original script for the game, that will then be approved and subsequently translated into Welsh by museum staff.
* To create a first, beta version of the game where there are no user decisions, and only one ending.
* To create a final version of the game where there are user decisions and two endings.

## Process

An Agile development methodology was utilised for this project. There wasn’t a rigid plan followed for this project which aligned very well with the flexible nature of Agile.

Basic principles of Agile were followed throughout the project, such as frequent communication with stakeholders (in this case, museum staff) to ensure that the project was meeting their needs, as well as adapting to changes as and when they arose. Tasks were also prioritised based on efficiency, which also aligns with the principles of Agile development, as does the clear idea of what features were needed.

An agile framework, such as Scrum or Kanban, was not used, as the development of this project was adapted very specifically to suit the work style of the developer. Choosing and then sticking strictly to a specific framework was not possible, not only due to the unknowns presented by the project, but the changing needs of the developer.

# Design

You should concentrate on the more important aspects of the design. It is essential that an overview is presented before going into detail. As well as describing the design adopted it must also explain what other designs were considered and why they were rejected.

DONE UP TO HERE

The design should describe what you expected to do and might also explain areas that you had to revise after some investigation.

Typically, for an object-oriented design, the discussion will focus on the choice of objects and classes and the allocation of methods to classes. The use made of reusable components should be described and their source referenced. Particularly important decisions concerning data structures usually affect the architecture of a system and so should be described here.

How much material you include on detailed design and implementation will depend very much on the nature of the project. It should not be padded out. Think about the significant aspects of your system. For example, describe the design of the user interface if it is a critical aspect of your system, or provide detail about methods and data structures that are not trivial. Do not spend time on long lists of trivial items and repetitive descriptions. If in doubt about what is appropriate, speak to your supervisor.

You should also identify any support tools that you used. You should discuss your choice of implementation tools - programming language, compilers, database management system, program development environment, etc.

Some example sub-sections may be as follows, but the specific sections are for you to define.

## Overall Architecture

## Detailed Design

### Even More Detail

## User Interface Design

## Other Relevant Sections

# Implementation

The implementation should discuss any issues you encountered as you tried to implement your design. During the work, you might have found that elements of your design were unnecessary or overly complex; perhaps third-party libraries were available that simplified some of the functions that you intended to implement. If things were easier in some areas, then how did you adapt your project to take account of your findings?

It is more likely that things were more complex than you first thought. In particular, were there any problems or difficulties that you found during implementation that you had to address? Did such problems simply delay you or were they more significant?

You can conclude this section by reviewing the end of the implementation stage against the planned requirements.

# Testing

Detailed descriptions of every test case are definitely not what is required in this section; the place for detailed lists of tests cases is in an appendix. In this section, it is more important to show that you adopted a sensible strategy that was, in principle, capable of testing the system adequately even if you did not have the time to test the system fully.

Provide information in the body of your report and the appendix to explain the testing that has been performed. How does this testing address the requirements and design for the project?

How comprehensive is the testing within the constraints of the project? Are you testing the normal working behaviour? Are you testing the exceptional behaviour, e.g. error conditions? Are you testing security issues if they are relevant for your project?

Have you tested your system on “real users”? For example, if your system is supposed to solve a problem for a business, then it would be appropriate to present your approach to involve the users in the testing process and to record the results that you obtained. Depending on the level of detail, it is likely that you would put any detailed results in an appendix.

Whilst testing with “real users” can be useful, don't see it as a way to shortcut detailed testing of your own. Think about issues discussed in the lectures about until testing, integration testing, etc. User testing without sensible testing of your own is not a useful activity.

The following sections indicate some areas you might include. Other sections may be more appropriate to your project.

## Overall Approach to Testing

## Automated Testing

### Unit Tests

### User Interface Testing

### Stress Testing

### Other Types of Testing

## Integration Testing

## User Testing

# Critical Evaluation

Examiners expect to find a section addressing questions such as:

* Were the requirements correctly identified?
* Were the design decisions correct?
* Could a more suitable set of tools have been chosen?
* How well did the software meet the needs of those who were expecting to use it?
* How well were any other project aims achieved?
* If you were starting again, what would you do differently?

Other questions can be addressed as appropriate for a project.

The questions are an indication of issues you should consider. They are not intended as a specification of a list of sections.

The evaluation is regarded as an important part of the project report; it should demonstrate that you are capable not only of carrying out a piece of work but also of thinking critically about how you did it and how you might have done it better. This is seen as an important part of an honours degree.

There will be good things in the work and aspects of the work that could be improved. As you write this section, identify and discuss the parts of the work that went well and also consider ways in which the work could be improved.

In the latter stages of the module, we will discuss the evaluation. That will probably be around week 9, although that differs each year.

# References

|  |  |
| --- | --- |
| [1] | “Eureka! Science and Disovery Centre,” [Online]. Available: https://www.eurekadiscovery.org.uk/. |

# Appendices

The appendices are for additional content that is useful to support the discussion in the report. It is material that is not necessarily needed in the body of the report, but its inclusion in the appendices makes it easy to access.

If you have used any 3rd party code, i.e. code that you have not written yourself such as libraries, then you must include Appendix A. In that appendix, you will provide details of the 3rd party code that you have used.

For most other items, it would be better to include them in your technical submission instead of including them as an appendix. For example:

* If you have developed a Design Specification document as part of a plan-driven approach for the project, then it would be appropriate to include that document in the technical work. In this report, you would highlight the most interesting aspects of the design, referring your reader to the full specification for further detail.
* If you have taken an agile approach to developing the project, then you may be less likely to have developed a full requirements specification at the start of the project. Perhaps you used stories to keep track of the functionality and the ‘future conversations.’ If it isn’t relevant to include all those stories in the body of your report, you could detail those stores in a document in the technical work.
* If you have used manual testing, then include a document in the technical work that records the tests that have been done. In this report, you would talk about the use of those tests.

Documents included in the technical work or in the appendices are supporting evidence of the work done. Where you include documents, this report should refer to the documents. You should not be relying on detailed study of those documents in order to understand what is written in this report.

Speak to your supervisor or the module coordinator if you have questions about this.

* 1. Third-Party Code and Libraries

If you have made use of any third-party code or software libraries, i.e. any code that you have not designed and written yourself, then you must include this appendix.

As has been said in lectures, it is acceptable and likely that you will make use of third-party code and software libraries. If third-party code or libraries are used, your work will build on that to produce notable new work. The key requirement is that we understand what your original work is and what work is based on that of other people.

Therefore, you need to clearly state what you have used and where the original material can be found. Also, if you have made any changes to the original versions, you must explain what you have changed.

The following is an example of what you might say.

**Apache POI library** – The project has been used to read and write Microsoft Excel files (XLS) as part of the interaction with the client’s existing system for processing data. Version 3.10-FINAL was used. The library is open source and it is available from the Apache Software Foundation [5]. The library is released using the Apache License [6]. This library was used without modification.

Include as many declarations as appropriate for your work. The specific wording is less important than the fact that you are declaring the relevant work.

* 1. Code Samples

This is an example appendix. Include as many appendices as you need. The appendices do not count towards the overall word count for the report.

For some projects, it might be relevant to include some code extracts in an appendix. You are not expected to put all of your code here - the correct place for all of your code is in the technical submission that is made in addition to the Project Report. However, if there are some notable aspects of the code that you discuss, including that in an appendix might be useful to make it easier for your readers to access.

As a general guide, if you are discussing short extracts of code then you are advised to include such code in the body of the report. If there is a longer extract that is relevant, then you might include it as shown in the following section.

Only include code in the appendix if that code is discussed and referred to in the body of the report.

Random Number Generator

The Bayes Durham Shuffle ensures that the pseudo random numbers used in the simulation are further shuffled, ensuring minimal correlation between subsequent random outputs.

// Some example code here…