

CS 320 Course Project Final Report

for

Medival Maps

Prepared by

Rabid Cabbage Collective

|  |  |  |
| --- | --- | --- |
| Patrick Wellington | <student #> | [Patrick.wellington@wsu.edu](mailto:Patrick.wellington@wsu.edu) |
| Lucas Gubala | 11543921 | [Lucas.gubala@wsu.edu](mailto:Lucas.gubala@wsu.edu) |
| Kevin Ngo | 11518172 | [Kevin.ngo2@wsu.edu](mailto:Kevin.ngo2@wsu.edu) |
|  |  |  |

|  |  |
| --- | --- |
|  |  |
| Date: | 12/13/2018 |
|  |  |
|  |  |
|  |  |

Contents

Contents ii

1 Introduction 1

1.1 Project Overview 1

1.2 Definitions, Acronyms and Abbreviations 1

1.3 References and Acknowledgments 1

2 Design 2

2.1 System Modeling 2

2.2 Interface Design 2

3 Implementation 3

3.1 Development Environment 3

3.2 Task Distribution 3

3.3 Challenges 3

4 Testing 4

4.1 Testing Plan 4

4.2 Tests for Functional Requirements 4

4.3 Tests for Non-functional Requirements 4

4.4 Hardware and Software Requirements 4

5 Analysis 5

6 Conclusion 6

Appendix A - Group Log 7

# 

# Introduction

*<TO DO: Please provide a brief introduction to your project.>*

## Project Overview

< A brief description of the project.

TO DO: Write 1-2 paragraphs describing the project. >

Medival Maps is a map generation tool designed for use with table top board games such as Dungeons and Dragons. MM randomly generates city like grids, that are populated with buildings and characters that bring the town to life. The player can choose what size of town to generate, as well as the fantasy races that will inhabit this town.

Medival Maps are not meant to recreate a real city, rather provide a simple layout that a Dungeon Master (the person responsible for leading the table top game), can rely on to be geographically sound. Instead of creating arbitrary distances between buildings, and coming up with what sort of buildings are in the town on the spot, DM’s can autogenerate the town.

## Definitions, Acronyms and Abbreviations

<Define all the terms necessary to properly interpret the report, including acronyms and abbreviations.

TO DO: Please provide a list of all abbreviations and acronyms used in this document sorted in alphabetical order.>

DnD: Dungeons and Dragons, a table top role playing game.

DM: Dungeon Master, the person who runs the table top role playing game. Responsible for creating the world.

MM: Shorthand for Medieval Maps.

## References and Acknowledgments

<List any other documents or Web addresses to which this document refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document.

TO DO: Use the standard IEEE citation guide for this section.>

# Design

## System Modeling

< Update your UML diagrams in milestone 2, to reflect the real implementation of this software.

TO DO: Provide an updated version of the UML diagrams, including use case diagrams, sequence (or state) diagrams, activities diagrams, and class diagrams. If you don’t have an updated version, just mention: “our implementation strictly follows the design document (milestone 2)”. >

## Interface Design

<Provide several screenshots to illustrate your interface design.

TO DO:

For each subsystem, pick one or two representative screenshots and paste here.>

# Implementation

## Development Environment

<Describe the devleopment environment you were using for the project.

TO DO: List the programming lanagues, IDEs, tools, etc.>

## Task Distribution

*<Describ how the implementation tasks are distributed among team members.*

*TO DO: For each team member, describe his/her main implementation tasks in this project.*

*If this is a one-person project, mention: “all the work presented here is done by \*\*\* (your name).” >*

## Challenges

*<This section is optional. Describ the challenges in the implementation, if there are any, and how you dealt with them.*

*TO DO: If you don’t have anything to fill in, just leave this section blank.>*

# Testing

## <*This section is a summary of your testing report>*

## Testing Plan

<Describe your testing plan for the project.

TODO: Give a list of items or functions you want to test, and also a schedule for performing the testing. >

## Tests for Functional Requirements

<Describe your test results for the functional requirements.

TODO: Provide a list of use cases or functions you have tested, as well as the testing results (whether or not the system passed the tests).>

## Tests for Non-functional Requirements

<Similar to the Section 4.2, but this section is for the non-functional requirements. >

## Hardware and Software Requirements

<Describe the hardware and software requirements for performing the tests. >

# Analysis

<In this Section you need to analyze the effort that has been put on this project.

TODO: Describe how many hours (approximately) each team member spent on the project, for each milestone, which milestone takes the most effort and why. >

# Conclusion

<Conclude the document with what you have learned through working on the project.>

Appendix A - Group Log

< Describe how frequently the group meembers meet during the semester, and how effective the communication is. This is optional for one-person projects.>