**Software Design Document**

<<Company Name>>

<< Team Member >>

<< Team Member >>

<< Team Member >>

<< Team Member >>

<< Team Member >>

1. Overview

1.1 Introduction

<< Briefly describe the major aspects of the design and, if applicable, how a developer will use it. For example: “Create and post a General Ledger transaction using the glTrx routines. Perform account inquiries with gjJournal routines.”

1.2 Scope

<< Provide a brief overview of the scope of this design. Also touch on anything that will not be included in this document. >>

2. High-Level Design

2.1 High-Level Component Design

<< Create a diagram of the high-level components or modules in the program, linking them with arrows to show any dependencies. Also complete the tables to provide a description of each module as well as the table which traces components to their related requirements. >>

|  |  |
| --- | --- |
| **Component** | **Description** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Component** | **Related Requirements** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

2.2 Activity Diagrams

<< Include activity diagrams for important workflows in the program. At least one diagram should be included for the main workflow in the program. Optionally include labels that indicate which component is responsible for that part of the activity. Activity diagrams for components which perform complex tasks should also be included. >>

2.3 Class Diagram

<< Include a class diagram for all classes to be designed. Optionally include major data elements of those classes and important methods and functions that will be used by other classes. >>

2.4 Sequence Diagram

<< Include sequence diagrams for important functionality of the program to indicate control flow. These diagrams should include classes found in the class diagram and use the methods for those classes to show the interaction between them. >>

3. User Interface Design

3.1 UI Description

<< Provide a brief description of the UI that will be used in this program and how users will interact with the program. >>

3.2 UI Mockup

<< Create a mockup of the user interface. This can be a simple drawing that demonstrates key parts of the user interface or a screenshot of a prototype created within an IDE. >>

4. Data Design

4.1 Program Data

<< If the project is using a database, or otherwise storing data in some format, provide a description of what data is being stored by the program and how it is being stored. Examples include storing customer data in an Oracle database, saving program configurations in an XML file etc. >>

4.2 Data Formats

<< If necessary, include the design for any important data files developed as part of this program. For example, if a configuration file is being used to store parameters for the program, it may be important to provide a detailed design of how that data will be stored. If your program is creating some form of custom save file, it may also be useful to provide a design for that in the event that other programs need to access those save files. >>

5. Non-Functional Design

5.1 Security

<< Discuss any necessary security features and include other necessary security considerations (e.g. remote access) that are part of this design. >>

5.2 Performance

<< Discuss any relevant design issues related to performance. For example, describe special algorithms used to obtain desired performance characteristics. Additional considerations may have been given for alternatives in data normalization. >>

**6. Document Review**

<< This section contains any relevant signatures necessary to indicate that the initiation document has been reviewed and approved. >>

|  |  |  |  |
| --- | --- | --- | --- |
| Name: | <Name> | Date: | <Date> |
| Signature: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Name: | <Name> | Date: | <Date> |
| Signature: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Name: | <Name> | Date: | <Date> |
| Signature: | | | |