GitGoing Code Review

# Software Design Document

Names: Travis Lamb, Christina Nguyen, Michael Bloomquist, Sarah Ramanazi

Date: (11/09/2020)

**TABLE OF CONTENTS**

1. [INTRODUCTION 2](#_bookmark0)
   1. [Purpose 2](#_bookmark0)
   2. [Scope 2](#_bookmark0)
   3. Definitions, Acronyms, and Abbreviations
   4. [References 2](#_bookmark0)
2. [SYSTEM OVERVIEW 2](#_bookmark0)
3. [SYSTEM Components 2](#_bookmark0)
   1. [Decomposition Description 3](#_bookmark1)
   2. [Dependency Description 3](#_bookmark1)
   3. Interface Description 3
      * Module Interfaces
      * User Interfaces (GUI)
4. [DETAILED DESIGN 3](#_bookmark1)
   1. [Module Detailed Design 3](#_bookmark1)
   2. [Data Detailed Design 3](#_bookmark1)
   3. RTM 3

### INTRODUCTION

## Purpose

This document contains a complete description of the design of GitGoing, and all relevant third-party dependencies. The expected audience is the Computer Science department of Bellevue College, including Professor Sara Farag, and other top-level Computer Science department members. Other Computer Science Department members may also find use of this document.

## Scope

The basic architecture is database back-end hosting a web application for clients. The database is a MySQL relational database, paired to a React web application that utilizes HTML and JavaScript. As development continues, the ideal goal of GitGoing is to provide a more intuitive version control website that does not scare away clients by being overly complex. Intuitive UI elements and detailed descriptions of valid actions will be used to achieve this goal.

## Definitions, Acronyms, and Abbreviations

*This section is optional.*

Provide definitions of all terms, acronyms, and abbreviations that might exist to properly interpret the SDD. These definitions should be items used in the SDD that are most likely not known to the audience.

## References

*This section is optional.*

List any documents, if any, which were used as sources of information for the test plan.

### SYSTEM OVERVIEW

Give a general description of the functionality, context and design of your project. Provide any background information if necessary.

### SYSTEM Components

## Decomposition Description

Provide a decomposition of the subsystems in the architectural design. Supplement with text as needed. You may choose to give a functional description or an object-oriented description. For a functional description, put top-level data flow diagram (DFD) and structural decomposition diagrams. For an OO description, put subsystem model, object diagrams, generalization hierarchy diagram(s) (if any), aggregation hierarchy diagram(s) (if any), interface specifications, and sequence diagrams here.

## Dependency Description

Develop a modular program structure and explain the relationships between the modules to achieve the complete functionality of the system. This is a high-level overview of how Software Design Document 3 responsibilities of the system were partitioned and then assigned to subsystems. Identify each high-level subsystem and the roles or responsibilities assigned to it. Describe how these subsystems collaborate with each other in order to achieve the desired functionality. Don’t go into too much detail about the individual subsystems. The main purpose is to gain a general understanding of how and why the system was decomposed, and how the individual parts work together. Provide a diagram showing the major subsystems and data repositories and their interconnections. Describe the diagram if required.

## Interface Description

## Module Interfaces

## User Interfaces (GUI)

### DETAILED DESIGN

## Module Detailed Design

Represent design through classes, activity diagrams, sequence diagrams, pseudocode, and/or tabular forms. Also summarize the relation with SRS. In this section, we take a closer look at what each component does in a more systematic way. If Software Design Document you gave a functional description in section 3.2, provide a summary of your algorithm for each function listed in 3.2 in procedural description language (PDL) or pseudocode. If you gave an OO description, summarize each object member function for all the objects listed in 3.2 in PDL or pseudocode. Describe any local data when necessary.

## Data Detailed Design

Describe how the major data or system entities are stored, processed and organized. νList any databases or data storage items.

## RTM

Provide a cross-reference that traces components and data structures to the requirements in your SRS document (later in STD and codes).νUse a tabular format to show which system components satisfy each of the functional requirements from the SRS. νRefer to the functional requirements by the numbers/codes that you gave them in the SRS.

### APPENDICES

*This section is optional.*

Appendices may be included, either directly or by reference, to provide supporting details that could aid in the understanding of the Software Design Document.

### 