****

**TEMPLATE EXECUTION SOFTWARE**



**Software Requirements Specification Version 0.1**

**Document Control:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Revision History** | | | | | | | | |
|  |  |  | |  |  |  |  |  |
| **Date** | **Version** | **Author** | **Brief Description of Changes** | | | | **Approver Signature** | |
| 18/11/2022 | 0.1(Draft) | Group 03 | Basic design and documentation | | | |  | |
|  |  |  |  | | | |  | |

|  |
| --- |
| **Team Members** |

|  |  |
| --- | --- |
| **Employee ID:** | **Name** |
| 46281047 | Jampala Triveni |
| 46282012 | Thatikonda Lavanya |
| 46282014 | Motapothula Vyshnavi |
| 46281049 | Vadisa Gayathri |
| 46281039 | Errabelly Archana |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |

**Table of Contents**

**1.Introduction……………………………………………………………………………………05**

* 1. Purpose
  2. Intended Audience
  3. Scope
  4. Overview

**2. Overall Design…………………………………………………………06**

2.1. Assumptions and Dependencies

2.2. Operating Environment

**3. Working……………………………………………………………….07**

**4. System Features………………………………………………............07**

4.1Functional Requirements

**5. External Interface Requirements……………………………...........08**

5.1User Interfaces

5.2 Hardware Interfaces

**6. Non-Functional Requirements………………………………............09**

6.1. Performance Requirements

6.2. Software Quality Attributes

6.3. Safety Requirements

**Template execution Software**

1. **Introduction**
   1. **Purpose**

The purpose of the project and this project is to implement a Template Execution Software (TES), where template means a file having some placeholder variables, whose values are fetched from other files. A typical invocation might be a given a template file and a number of data files, resulting in formation of output files corresponding to each data file.

* 1. **Intended Audience**

The project is designed to provide users with a Template Execution Software , where template means a file having some placeholder variables, whose values are fetched from other files. A typical invocation might be a given a template file and a number of data files, resulting in formation of output files corresponding to each data file.

* 1. **Project Scope**

The purpose of the application is shall support substitution of variables in Template file, picking up the variable (v) from data. A single invocation of TES will be done by specifying one template file and one-or-more data files.

**1.4 Overview**

The ‘TEMPLATE EXECUTION SOFTWARE’ project will developed to overcome the time consuming problem of manual system. Here we collect the one template file and multiple data files here we put the placeholder variable in the template file. Here we replace the variable with value from the data file. The project will reduce the manual process in managing examinations and all issues regarding that.

**2. Overall Description**

**2.1. Assumptions and Dependencies**

The following assumptions have been made in regards to the development of the TES Evaluation:

* The user or client organization has machines capable of running a UNIX based operating system.
* C source code can be compiled on the machines.
* The users have some storage space to store the data.

**2.2** **Operating Environment**

Operating Environment for the TES is as follows:

* Operating System: Any UNIX Based OS
* Compiler: GCC or similar to compile source code written in C programming language.

**3.Working of Template Execution Software**

The Template Execution Software is developed using the C programming language. Here we can put the all the data files into data csv which saves the memory. Here we collect the one template file and multiple data files here we put the placeholder variable in the template file. Here we replace the variable with value from the data file.

**4.System Features**

The template execution software is an application that helps the procees the data.

**4.1**. **Functional Requirements**

Following is a list of functionalities of the system:

1. Replace: It is used to replace old word with new word.
2. Load template file: It is used to load the template file.
3. Load data file: It is used to load the data file.
4. Process replacement: It is process is repeated. That means we can deal with the data files one after one.

.

**5.External Internal Requirements**

This application uses UNIX (CLI) to perform connection between the template file and data files.

Based on User choice the functions will perform tasks (Code reuse).

**5.1.** **User Interfaces**

* + GUL The application does not use Graphical User Interface
  + CLI: This application uses Command Line Interface to accept console commands by users choice and perform the needful functions.

**5.2. Hardware Interfaces**

Hardware Requirements are as follows:

* 32 bit and 64bit Machine capable of running UNIX based operating system
* Storage space to store the data.

**6. Non-Functional Requirements**

**6.1 Performance Requirements**

The Application is developed to run through CLI on UNIX based systems. As long as the machine can run the operating system along with the necessary dependencies without any flaws there are no additional requirements.

* 1. **Software Quality Attributes**

**Serviceability:**

* The system requires minimal amount of maintenance. The maintenance and upkeep can be performed by any person with a basic understanding and development experience in C.

**Reusability:**

* The application needs to be designed in a way such that the code can be easily reused and it can run in any UNIX like machine in the organization.

**Binary Compatibility**:

* This application should be compatible with any computer that has an UNIX based operating system.

**Portability:**

* The source code needs to be implemented in such a way that it is portable to any machine that can compile and run C programs.
  1. **Safety Attributes**
* The data (all fields) of the candidate should be correct.
* Atleast one of the candidate field should be unique.

THANK YOU