MSSE SOFTWARE, INC.

Test Plan v1.0 for

GolfScore

Revision 1.1

Contents

1.0 Introduction

1.1. Objective

The Test Plan outlines the entire testing process for the GolfScore project, covering unit tests, system verification tests (SVT), and regression tests. It identifies requirements, schedules, resources, assumptions, risks, and mitigation strategies to ensure GolfScore software meets the defined functionality in the Software Requirements Specification (SRS).

1.2. Project Description

GolfScore is a command-line application designed to process scores from a golf tournament and produce reports based on these scores. The system generates three types of reports: Tournament Ranking, Golfer Report, and Course Report. Input is read from a file, and outputs are produced as text files.

1.3. Process Tailoring

The GolfScore project will primarily use specification testing, functional testing, and performance testing. Given the nature of the software, stress testing and GUI testing are unnecessary. Regression testing will be performed after fixing defects uncovered during the initial testing phases.

1.4. Referenced Documents

- **Software Requirements Specifications (SRS) for GolfScore,** Rev 1.1, July 18, 2017
- **TestPlan Example for Advanced Color Module**

2.0 Assumptions/Dependencies

- Code completion for all core functionalities (including report generation and error handling) by the development team is assumed by a specified date.
- Input data files must conform to the format specified in the SRS.
- Output directories should exist and have sufficient space for report files.
- Necessary test environments (Windows PC running GolfScore) are available and functional.

3.0 Test Requirements

The following system test requirements are derived from the GolfScore SRS:

1. **Report Generation:**
    a. Tournament Ranking Report (Section 2.5.1)
    b. Golfer Report (Section 2.5.2)
    c. Course Report (Section 2.5.3)
2. **Input Handling:**
    a. Proper parsing of input files (Section 2.4)
    b. Detection of errors in input data and parameters (Sections 2.6.1, 2.6.2)
3. **Command-line Execution:**
    a. Proper handling of command-line options (Section 2.2)
4. **Scoring System:**
    a. Correct scoring for each hole and course (Section 2.3.2)
5. **Error Handling:**
    a. Appropriate error messages for incorrect inputs or missing files (Sections 2.6.1 – 2.6.3)

4.0 Test Tools

- **Unit Test Framework:** A framework such as CUnit for C/C++ unit tests.
- **Automated Testing Tools:** To verify command-line execution and input-output tests.
- **Error Reporting:** A simple logging mechanism to track errors and execution states.

5.0 Resource Requirements

- 2 test engineers for functional and (SVT).
- One test system (PC running Windows 2000 or later).
- Software tools (e.g., a text editor and command prompt for manual testing).

6.0 Test Schedule

Test Phase	Start Date (DD/MM/YYYY)	End Date (DD/MM/YYYY)
Test Plan Development	9/9/2024	13/9/2024
Unit Testing	16/9/2024	20/9/2024
System Verification Test	23/9/2024	27/9/2024
Regression Testing	30/9/2024	4/10/2024

7.0 Risks/Mitigation

Risk	Mitigation Strategy
Delay in completing code	Set internal deadlines for the development team.
Errors in input file format	Thoroughly document input format for testers.
Failure to meet performance requirements	Conduct performance testing early in the cycle.

8.0 Metrics

The following metrics will be tracked:

- Number of defects found during each phase.
- Time taken to execute test cases.
- Number of test cases passed vs. failed.
- Defects found during regression testing.

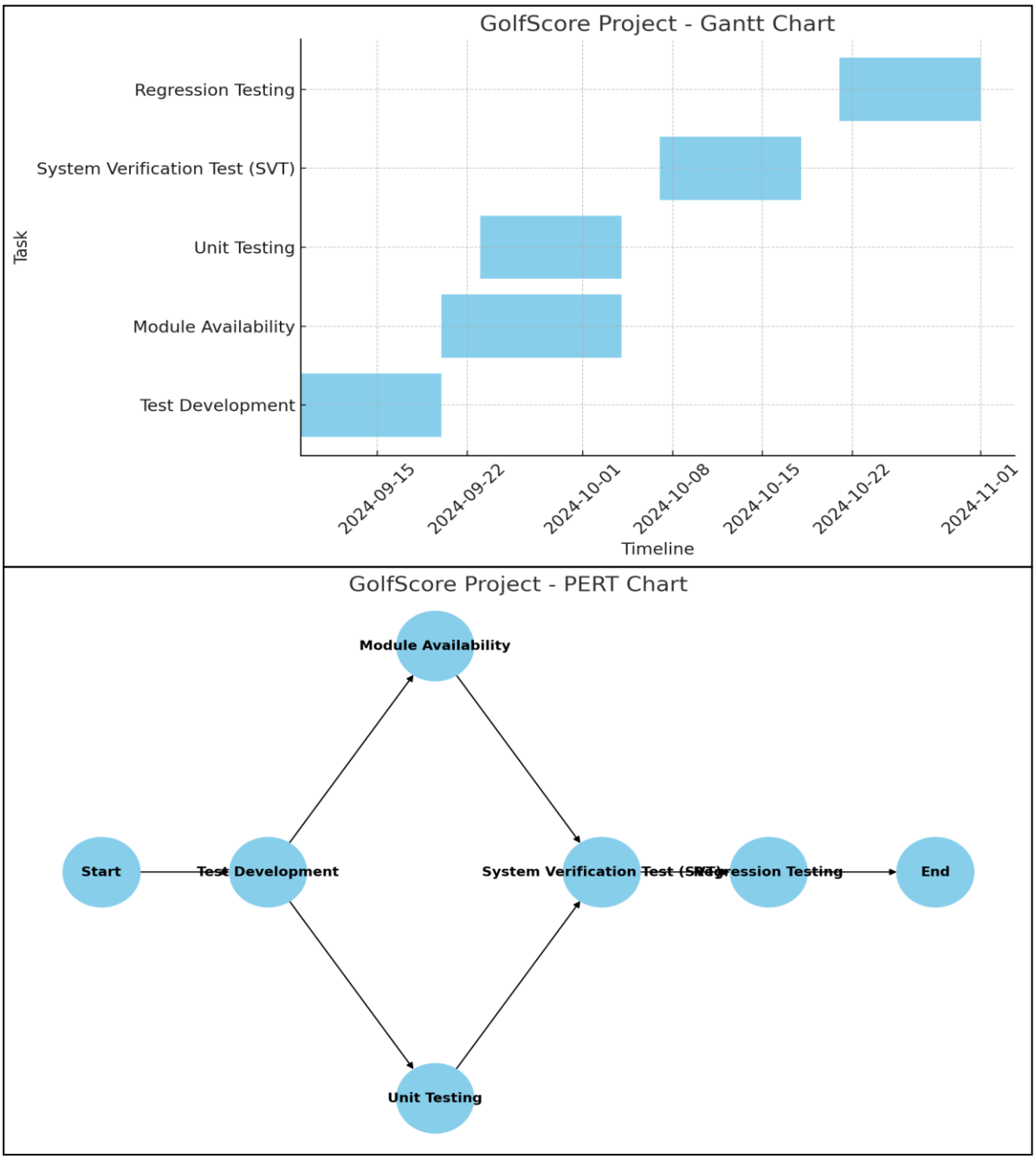
Appendix A – Detailed Resource Requirements

The resource requirements for the **GolfScore** project involve estimating the personnel, tools, and time needed for all testing activities. Detailed activities and resource allocation include:

1. **Personnel Requirements:**
    - **Lead Test Engineer:** Responsible for coordinating the testing effort, defining test cases, and ensuring alignment with project goals. Estimated effort: **50 hours.**
    - **Test Engineer:** Executes test cases, documents defects, and assists in the regression testing phase. Estimated effort: **80 hours.**
    - **Software Developer:** Provides support for test environment setup, addresses critical defects, and assists with performance testing. Estimated effort: **30 hours.**
2. **Testing Tools:**
    - **CUnit or equivalent:** A C/C++ unit testing framework will be required for verifying individual code modules of GolfScore.
    - **Automated Test Scripts:** To simulate command-line inputs and verify outputs, automation scripts will be developed. Custom automation for output verification may also be required.
    - **Log analysis tools:** To evaluate error messages and output files for correctness and defect detection.
3. **Test Environment:**
    - **Hardware:** One PC running **Windows 2000 or later** to host the GolfScore executable and perform testing.
    - **Input Files:** Pre-formatted test files for input records will be created and stored locally.
    - **Output Directories:** Directory for storing report files generated during tests.
4. **Time Allocation:**
    - **Unit Testing:** 1 week of dedicated time, primarily for testing individual functions (e.g., score calculation, report generation).
    - **System Verification Test (SVT):** 2 weeks to ensure all requirements are met as specified in the SRS.
    - **Regression Testing:** 1 week after defect resolution to validate that fixes do not affect other functionality.
5. **Miscellaneous:**
    - **Documentation Tools:** For maintaining test records, logging defects, and tracking metrics.

These requirements ensure that the **GolfScore** project testing aligns with project timelines, resources, and objectives.

Appendix B – Detailed Test Schedule



The **Gantt Chart** displays the timeline for each testing phase, including test development, module availability, unit testing, system verification, and regression testing.

The **PERT Chart** illustrates the dependencies between the tasks, showing the sequence from the start of test development through to regression testing and project completion.

Appendix C – Example Test Cases

Test Case 1: Verify Correct Tournament Ranking Report Generation

Test ID: TC_001

Objective: Verify that the **Tournament Ranking Report** is generated correctly and includes all required fields as specified in the SRS.

Preconditions: GolfScore is installed, input data file is prepared with valid records, and the application is run from the command line.

Test Steps:

1. Prepare an input file with the necessary Course Records and Golfer Records.
2. Call the application with the following command: >golf -t <input_file> <output_directory>
3. Verify that the **Tournament Ranking Report** is generated in the specified output directory (trank.rep).
4. Open the generated report and verify that:
    i. The report includes the golfer's name.
    ii. The scores for each course are listed.
    iii. The total tournament score is calculated.
    iv. The golfers are ranked in descending order of total score.

Expected Results: The **Tournament Ranking Report** is generated with correct scores and rankings, and it matches the input data.

Test Case 2: Verify Golfer Report Sorting

Test ID: TC_002

Objective: Ensure the **Golfer Report** is generated and correctly sorted alphabetically by golfer name.

Preconditions: The GolfScore executable is running, and the input file contains records for multiple golfers.

Test Steps:

1. Prepare an input file with several golfer records (ensure names vary).
2. Call the application with the following command: >golf -g <input_file> <output_directory>.

3. Verify that the **Golfer Report** is generated in the output directory (golfer.rep).
4. Open the generated report and confirm the following:
    i. All golfer names are listed alphabetically.
    ii. Scores and rankings are correctly reflected for each golfer.

Expected Results: The golfers are listed in alphabetical order with accurate scoring information in the **Golfer Report.**

Test Case 3: Verify Course Report Generation

Test ID: TC_003

Objective: Verify that the **Course Report** is generated and contains the correct scores for each golfer per course.

Preconditions: Valid input file with multiple courses and golfer data.

Test Steps:

1. Prepare an input file with multiple Course Records and Golfer Records.
2. Run the GolfScore application with the following command: >golf -c <input_file> <output_directory>.
3. Verify that the **Course Report** is generated in the output directory (course.rep).
4. Open the generated report and verify that:
    i. For each course, the list of golfers includes their hole-by-hole stroke count.
    ii. The total score for each course is displayed.
    iii. The golfers are ranked correctly based on their scores on each course.

Expected Results: The **Course Report** is generated with accurate stroke counts and ranks golfers correctly for each course.

Test Case 4: Validate Error Handling for Missing Input File

Test ID: TC_004

Objective: Ensure that an appropriate error message is displayed when the input file is missing.

Preconditions: The GolfScore application is installed but no input file is present.

Test Steps:

1. Run the application with the following command: >golf -t missing_file.txt <output.directory>.
2. Verify that the application stops and displays an appropriate error message indicating that the input file is missing.

Expected Results: The application should display an error message stating: File missing_file.txt does not exist and terminate without generating any reports.

Test Case 5: Verify Help Option

Test ID: TC_005

Objective: Ensure that the help information is displayed when the -h option is used.

Preconditions: The GolfScore executable is installed and running.

Test Steps:

1. Run the application with the help option: >golf -h.
2. Verify that the help message is displayed, and the information matches the command-line options defined in the SRS.

Expected Results: The help message is displayed on the screen with correct command syntax and descriptions.

Test Case 6: Validate Error Handling for Invalid Parameters

Test ID: TC_006

Objective: Ensure that the system handles invalid command-line parameters gracefully.

Preconditions: The GolfScore executable is installed and available.

Test Steps:

1. Run the application with an invalid option: >golf -x <input_file> <output_directory>.
2. Verify that the system terminates and displays an appropriate error message for the unrecognised option.

Expected Results: The application should display an error message: Unrecognised option -x. Use -h for help.