*AUTOMATED TIME TABLE*

Test report

Version *<1.0>*

*<04/04/2016>*

VERSION HISTORY

[Provide information on how the development and distribution of the Test Report was controlled and tracked. Use the table below to provide the version number, the author implementing the version, the date of the version, the name of the person approving the version, the date that particular version was approved, and a brief description of the reason for creating the revised version.]

|  |  |  |
| --- | --- | --- |
| **Version #** | **Implemented**  **By** | **Revision**  **Date** |
| 1.0 | *14* | *04/04/2016* |
|  |  |  |
|  |  |  |
|  |  |  |

***Note to the Author***

[This document is a template of a **Test Report** document for a project. The template includes instructions to the author, example text, and fields that should be replaced with the values specific to the project.

* Blue italicized text enclosed in square brackets ([text]) provides instructions to the document author, or describes the intent, assumptions and context for content included in this document.
* Blue italicized text enclosed in angle brackets (<text>) indicates a field that should be replaced with information specific to a particular project.
* Text and tables in black are provided as examples of wording and formats that may be used or modified as appropriate to a specific project. These are offered only as suggestions to assist in developing project documents; they are not mandatory formats.

When using this template for your project document, it is recommended that you follow these steps:

1. Replace all text enclosed in angle brackets (e.g., <Project Name>) with the correct field values. These angle brackets appear in both the body of the document and in headers and footers.
2. Modify example text as appropriate to the specific project.
3. To add any new sections to the document, ensure that the appropriate header and body text styles are maintained. Styles used for the Section Headings are Heading 1, Heading 2 and Heading 3. Style used for boilerplate text is Body Text.
4. To update the Table of Contents, right-click and select “Update field” and choose the option- “Update entire table”
5. Before submission of the first draft of this document, delete this “Notes to the Author” page and all instructions to the author, which appear throughout the document as blue italicized text enclosed in square brackets.]

**Table of Contents**

[1.0 Introduction 5](#_Toc445795901)

[1.1 Purpose 5](#_Toc445795902)

[2.0 Test PLAN 5](#_Toc445795903)

[3.0 Test Assessment 5](#_Toc445795904)

[4.0 Test Results 5](#_Toc445795905)

[4.1 Unit/Module/System Testing 5](#_Toc445795906)

[4.2 System Testing 6](#_Toc445795907)

[4.3 User Acceptance Testing 6](#_Toc445795908)

[4.4 Regression Testing 7](#_Toc445795909)

[4.5 Performance Testing 7](#_Toc445795910)

[4.6 <Type of Test> 8](#_Toc445795911)

[5.0 Variances 8](#_Toc445795912)

[6.0 Test Instances 8](#_Toc445795913)

[6.1 Resolved Test Incidents 8](#_Toc445795914)

[6.2 Unresolved Test Incidents 8](#_Toc445795915)

[7.0 Recommendations 9](#_Toc445795916)

[APPENDIX A: REFERENCES 10](#_Toc445795917)

[APPENDIX B: KEY TERMS 11](#_Toc445795918)

# 1.0 Introduction

## 1.1 Purpose

This *Automated Time Table* Test Report provides a summary of the results of test performed as outlined within this document.

# 2.0 Test PLAN

We started with testing the basic working of the software. Going through all the use cases mentioned in the SRS document, we tested them first with different range of test inputs. Henceforth, we moved towards the UI part, where we performed User Acceptance Testing to test if this software is good enough for the end users who are assumed to be capable enough only to run the software and follow the commands displayed on screen.

After System and User Acceptance testing, we did Performance Testing. It includes testing the software in accordance with various nonfunctional parameters like speed, time, and reliability.

# 3.0 Test Assessment

Test cases were enough to find various major errors in the software.

Firstly, when we gave a wrong input to the software, it was not able to test its correctness or pop up some message about the unacceptable format of the input.

Next, when we altered the predefined test case by reducing the number of courses, faculties, students, rooms and changing other constraints, the application didn’t yield any output.

Moreover, the algorithm implemented is not uniform with the input size. Even for same input file (that provided by the developers), it yields different output when run several times. The software takes variable time to yield the output when same input is given on different runs. The system is not able to check the constraints on the input, and even if it does, it does not prompt a message on the screen regarding it.

Therefore, with our test cases we were able to find out all the major problems in the working of this software.

# 4.0 Test Results

1. User Interface of software is very bad. On opening the software, there are no instructions displayed for further actions.
2. No interface to specify input.
3. No interface to specify constraints.
4. No interface to input courses.
5. No interface to input courses students.
6. No interface to input courses faculties.
7. No interface to input slots.
8. No interface to input rooms and room allocations.
9. Software yields different time tables when same input is given twice creating a major issue of consistency.
10. Software takes enormous time to generate the timetable, which might irritate the user.
11. Software doesn’t give any output on some test inputs.
12. Generated Time Table is not saved in any directory. If saved, then there is no notification about it.
13. If student gives his own details, then the time table generated for him is different from the timetable generated for the entire institute which threatens the reliability of the software.
14. According to SRS, there should be a feature to notify the user right before his class timings, but no such feature has been implemented.
15. When an input is given with only 1 classroom, then software do not prompt any notification that time table could not be generated. Software continues to run for infinite amount of time.
16. THU (Thursday) is written instead of TUE (Tuesday) in the generated time table.
17. The Generation number generated during Time Table generation can be changed by scrolling on the window. It only stops if we stop the Time Table generation.
18. No function is made to search through the Time Table.
19. On giving an input with no class, no course, no prof or no room, , the software begin to hang and then, do not respond instead of giving any error message.
20. Giving a blank input file also results in not responding state.
21. If I remove the entire .cfg file from the software folder and place it somewhere else, the software does not accept it and goes into not responding state.

## Unit/Module/System Testing

Unit, module, and system integration testing activities were performed during the development of the system build or release.

## 4.2 System Testing

The table below summarizes the results of system testing:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test Case ID | Date Tested | Tester | Pass/Fail | Severity of Defect [Low/ Medium/ High] | Summary of Defect | Closed prior to Production Release? | Comments |
| ST\_1 | 02/04/2016 | Abhay | Fail | Medium | User not able to customize the input | <Yes> or <No> | After clicking the find configuration option, user should get a window to fill all the inputs. |
| ST\_2 | 02/04/2016 | Abhay | Fail | Low | No such option is available in the software for this feature. |  | There must be an option for sending notification to the students about their classes |
| ST\_3 | 02/04/2016 | Abhay | Pass | - |  |  | Software gave correct time table as output |
| ST\_4 | 02/04/2016 | Abhay | Fail | High | Software giving different output when run twice. |  | Software must make only 1 time table for a given input. |
| ST\_5 | 03/04/2016 | Deepshi | Pass | - |  |  | User (student) is able to generate his own time table, but it does not match with the original one. |
| ST\_5 | 03/04/2016 | Deepshi | Fail | Medium | Software does not respond on no input |  | Software begins to hang, then does not respond when an empty input file is given. |
| ST\_6 | 03/04/2016 | Deepshi | Fail | High | Software doesn’t run on very small input with obvious solution |  | When input is given with only 3 classes and very few courses, the software do not show any progress in Time Table generation |
| ST\_7 | 03/04/2016 | Deepshi | Fail | Medium | No search function is made to search through the Time Table |  | Not even ctrl+f works on the software. |
| ST\_8 | 04/04/2016 | Abhay | Fail | High | Software does not responds when an input is given with either no class, no prof, no room or no course. |  | Error handling is not done properly. Software must prompt an error message. |
| ST\_9 | 04/04/2016 | Raghava | Fail | High | Software does not run when redundant data is added in the input |  | Software must run with the available data. |

*[If the test case failed, list the corresponding Test Incident ID in the Comments column.]*

## User Acceptance Testing

The table below summarizes the test cases employed for user acceptance testing and the test results obtained for each test case:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test Case ID | Date Tested | Tester | Pass/Fail | Severity of Defect [Low/ Medium/ High] | Summary of Defect | Closed prior to Production Release? | Comments |
| UAT\_1 | 03/04/2016 | Raghava | Fail | High | User got different time tables when he ran the software on the same input twice. | <Yes> or <No> | Software must yield same software for same input. Different output shows randomness in the algorithm. |
| UAT\_2 | 03/04/2016 | Raghava | Pass | - | - |  | User is able to get the Time Table |
| UAT\_3 | 03/04/2016 | Raghava | Fail | High | Software takes a .cfg file as input to make the Time Table |  | There is no interface to take input from user. User is not educated to make .cfg file. |
| UAT\_4 | 03/04/2016 | Raghava | Fail | High | Student generated time table does not match with the original time table |  | Original time table is not saved anywhere. |
| UAT\_5 | 03/04/2016 | Abhay | Fail | High | Generated time table is not saved in any directory |  | User cannot save the generated time table and use it for future use. |
| UAT\_6 | 03/04/2016 | Deepshi | Fail | Medium | THU (Thursday) is written instead of TUE (Tuesday) in the generated time table |  | This type of GUI error can cause confusion to the user. |
| UAT\_7 | 04/04/2016 | Raghava | Fail | High | .cfg file selected from any other folder is not accepted by the software |  | If I place .cfg file in any other folder, the software does not accept it and goes into not responding state. |

*[If the test case failed, list the corresponding Test Incident ID in the Comments column.]*

## Regression Testing

The table below summarizes the test cases employed for regression testing and the test results obtained for each test case:

\*This software is Version 1.0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test Case ID | Date Tested | Tester | Pass/Fail | Severity of Defect [Low/ Medium/ High] | Summary of Defect | Closed prior to Production Release? *[This column will be filled by the dev team.]* | Comments |
|  |  |  |  |  |  | <Yes> or <No> |  |

*[If the test case failed, list the corresponding Test Incident ID in the Comments column.]*

## Performance Testing

The table below summarizes the test cases employed for performance testing and the test results obtained for each test case:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test Case ID | Date Tested | Tester | Pass/Fail | Severity of Defect [Low/ Medium/ High] | Summary of Defect | Closed prior to Production Release? | Comments |
| PT\_1 | 02/04/2016 | Abhay | Fail | High | Give different output on same input | <Yes> or <No> | Consistency Problem |
| PT\_2 | 02/04/2016 | Deepshi Garg | Fail | Low | Takes a lot of time to generate time table |  |  |
| PT\_3 | 03/04/2016 | Deepshi | Fail | High | Student generated time table does not match the actual timetable |  | Reliability issue. |

*[If the test case failed, list the corresponding Test Incident ID in the Comments column.]*

# 5.0 Variances

Most of the testing that was planned has been executed. Many of the features that were mentioned in the SRS have not been implemented, so we had to change some of the input test cases. Since, input is taken as a .cfg file, therefore we are not able to make a very large input test case because manually, it will take a lot of time to make a large .cfg file.

Therefore, software is tested on several boundary cases, but not for a very large test case because according to the SRS, we expected to get an interface to input our values which is not available.

# 6.0 Test Instances

1. The generation count visible during the time table generation can be changed by scrolling on the window.
2. Software didn’t responded when an empty .cfg file was given as input.
3. Software didn’t responded when a .cfg file with no classes was given as input.
4. Unable to search anything from the generated time table, not even using ctrl+f.

## Resolved Test Incidents

*[Identify all resolzved test incidents and summarize their resolutions. Reference may be made to Test Incident Reports that describe in detail the unexpected results, problems, or defects reported during testing, along with their documented resolutions, which may be included as an appendix to this document.]*

*[This will be filled by the dev team.]*

## Unresolved Test Incidents

[Identify all unresolved test incidents and provide a plan of action for their resolution. Reference may be made to Test Incident Reports that describe in detail the unexpected results, problems, or defects reported during testing, which may be included as an appendix to this document.]

*[This will be filled by the dev team.]*

# 7.0 Recommendations

Developers must make an interface to take input all the required fields for the making of timetable. If possible, a database must be maintained to prevent rerun of redundant outputs.

Also, they should try to implement a more reliable, consistent algorithm which has a lesser time complexity.

APPENDIX A: REFERENCES

[Insert the name, version number, description, and physical location of any documents referenced in this document. Add rows to the table as necessary.]

The following table summarizes the documents referenced in this document.

|  |  |  |
| --- | --- | --- |
| **Document Name** | **Version** | **Description** |
| *SRS (Automated Time Table)* | *3.0* | *This document contained the complete design of the software and all the use cases.* |

APPENDIX B: KEY TERMS

*[Insert terms and definitions used in this document. Add rows to the table as necessary.]*

The following table provides definitions for terms relevant to this document.

|  |  |
| --- | --- |
| **Term** | **Definition** |
| *.cfg* | *Extension of the file that contains all the details to make the timetable* |
| *[Insert Term]* | *[Provide definition of the term used in this document.]* |
| *[Insert Term]* | *[Provide definition of the term used in this document.]* |