# AUTOMATIC LAUNDRY SYSTEM

# SOFTWARE EVALUATION CRITERIA

***Group: 05***
Fardin Momtaj-201614012
Umair Sifat-201614015
Imtiaz Ahmed-201614030
S M Arif Ahmed-201514042
Farhan Saif Chowdhury-201414077

Department Of Computer Science and Engineering
MILITARY INSTITITE OF SCIENCE AND TECHNOLOGY

# Contents

**Revision History** **1**

**1 Introduction** **2**

1.1 Objective . . . . . 2

**2 Checklist of Criteria** **3**

**3 Detailed Software Evaluation Report** **4**

3.1 Usability Evaluation . . . . . 4

3.2 Sustainability and Maintainability Evaluation . . . . . 8

**4 References** **15**

# Revision History

| **Revision** | **Date** | **Author(s)** | **Description** |
|---|---|---|---|
| 1.0 | 11.09.2019 | | Chapter 1,Chapter 2 |
| 2.0 | 19.09.2019 | | Chapter 3 |

# Chapter 1

# Introduction

There are two types of software evaluation approach: criteria-based assessment and tutorial-based assessment according to the Software Sustainability Institute. We have adopted the criteria-based assessment for our integrated design project. Our project is Automatic Laundry System and the aim of this project is

- Make the laundry system automatic integrated with ironing, folding and packaging.
- Reduce the time it takes to do the task manually by human being.
- Design a cost effective way to do this automatic system.

Criteria-based assessment is a quantitative assessment of the software in terms of sustainability, maintainability, and usability. This can inform high-level decisions on specific areas for software improvement. A criteria-based assessment gives a measurement of quality in a number of areas. These areas are derived from ISO/IEC 9126-1 Software engineering — Product quality and include usability, sustainability and maintainability. The rest of this document covers each category in greater depth, with lists of questions that is used at the Software Sustainability Institute when compiling detailed software evaluation reports [1]

## 1.1 Objective

The assessment involves checking whether the software, and the project that develops it, conforms to various characteristics or exhibits various qualities that are expected of sustainable software. The more characteristics that are satisfied, the more sustainable the software.

# Chapter 2

# Checklist of Criteria

These assessment criteria for "Criteria-based Software Evaluation" is established by the Software Sustainability Institute which cultivates better, more sustainable, research software to enable world-class research. The assessment criteria are grouped as follows [1] -

| **Criteria** | **Sub-Criteria** | **Notes – to what extent is/does the software. . .** |
|---|---|---|
| Usablity | Understandability | Easily understood? |
| | Documentation | Comprehensive, appropriate, well-structured user documentation? |
| | Buildability | Straightforward to build on a supported system? |
| | Installability | Straightforward to install on a supported system? |
| | Learnability | Easy to learn how to use its functions? |
| Sustainability and maintainability | Identity | Project/software identity is clear and unique? |
| | Copyright | Easy to see who owns the project/software? |
| | Licensing | Adoption of appropriate licence? |
| | Governance | Easy to understand how the project is run and the development of the software managed? |
| | Community | Evidence of current/future community? |
| | Accessibility | Evidence of current/future ability to download? |
| | Testability | Easy to test correctness of source code? |
| | Portability | Usable on multiple platforms? |
| | Supportability | Evidence of current/future developer support? |
| | Analysability | Easy to understand at the source level? |
| | Changeability | Easy to modify and contribute changes to developers? |
| | Evolvability | Evidence of current/future development? |
| | Interoperability | Interoperable with other required/related software? |

# Chapter 3

# Detailed Software Evaluation Report

## 3.1 Usability Evaluation

Usability is the ease of use and learnability of a human-made object such as a tool or device. In software engineering, usability is the degree to which a software can be used by specified consumers to achieve quantified objectives with effectiveness, efficiency, and satisfaction in a quantified context of use [2]. The sub-criteria are evaluated in details with respect to our project below.

| **Understandability**<br>How straightforward is it to understand?<br>• What the software does and its purpose?<br>• The intended market and users of the software?<br>• The software's basic functions?<br>• The software's advanced functions? | Yes/No, supporting comments if warranted |
|---|---|
| High-level description of what/who the software is for is available. | Yes, Available in the project proposal. |
| High-level description of what the software does is available. | Yes, Available in the project proposal. |
| High-level description of how the software works is available. | Yes, Available in the UX design. |
| Design rationale is available – why it does it the way it does. | No |
| Architectural overview, with diagrams, is available. | Yes, Available in the system architecture. |
| Descriptions of intended use cases are available. | No |
| Case studies of use are available. | No |

| | |
|---|---|
| **Documentation**<br>Looking at the user documentation, what is its<br>• Quality?<br>• Completeness?<br>• Accuracy?<br>• Appropriateness?<br>• Clarity? | Yes/No, supporting comments if warranted |
| Provides a high-level overview of the software. | Yes, Available in the software requirement specification. |
| Partitioned into sections for users, user-developers and developers (depending on the software). | No |
| Lists resources for further information. | Yes, Available in the software requirement specification. |
| Is task-oriented. | Yes, Available in the software requirement specification. |
| Consists of clear, step-by-step instructions. | Yes, Available in the software requirement specification. |
| Gives examples of what the user can see at each step e.g. screen shots or command-line excerpts. | No |
| For problems and error messages, the symptoms and step-by-step solutions are provided. | No |
| Limitations/constraints are provided clearly in documentation. | Yes, Available in the software requirement specification. |
| Is on the project web site. | No |
| Documentation on the project web site makes it clear what version of the software the documentation applies to. | No |

| **Buildability**<br>How straightforward is it to?<br>• Meet the pre-requisites for building the software on a build platform?<br>• Build the software on a build platform? | Yes/No, supporting comments if warranted |
|---|---|
| Software has instructions for building the software. | No |
| Source distributions list all third-party dependencies that are not bundled, along with web addresses, suitable versions, licences and whether these are mandatory or optional. | Yes |
| Dependency management is used to automatically download dependencies (e.g. ANT, Ivy, Maven or custom solution). | Yes, Available in the software test plan. |
| All mandatory third-party dependencies are currently available. | Yes, Available in the software test plan. |
| All optional third-party dependencies are currently available. | Yes, Available in the software test plan. |
| Tests are provided to verify the build has succeeded. | Yes, Available in the software test plan. |

| | |
|---|---|
| **Installability**<br>How straightforward is it to?<br>• Meet the pre-requisites for the software on a target platform?<br>• Install the software onto a target platform?<br>• Configure the software following installation for use?<br>• Verify the installation for use?<br>Note that in some cases build and install may be one and the same. | Yes/No, supporting comments if warranted |
| Software has instructions for installing the software. | Yes, Available in the help document. |
| Source distributions list all third-party dependencies that are not bundled, along with web addresses, suitable versions, licences and whether these are mandatory or optional. | Yes, Available in the help document. |
| Dependency management is used to automatically download dependencies (e.g. ANT, Ivy, Maven or custom solution). | Yes, Available in the help document. |
| All mandatory third-party dependencies are currently available. | Yes, Available in the help document. |
| All optional third-party dependencies are currently available. | Yes, Available in the help document. |
| Tests are provided to verify the install has succeeded. | Yes, Available in the software test plan. |
| All GUIs contain a Help menu with commands to see the project name, web site, how/where to get help, version, date, licence and copyright (or where to find this information), location of entry point into user doc. | Yes, Available in the software UX design. |
| Installers allow user to select where to install software. | No |
| Uninstallers uninstall every file or warns user of any files that were not removed and where these are. | No |

| | |
|---|---|
| **Learnability**<br>How straightforward is it to learn how to achieve?<br>• Basic functional tasks?<br>• Advanced functional tasks? | Yes/No, supporting comments if warranted |
| A getting started guide is provided outlining a basic example of using the software. | No |
| Instructions are provided for many basic use cases. | Yes, Available in the software requirement specification. |
| Instructions are provided supporting all use cases. | Yes, Available in the software requirement specification. |
| Reference guides are provided for all command-line, GUI and configuration options. | No |
| API documentation is provided for user-developers and developers. | No |

## 3.2 Sustainability and Maintainability Evaluation

Sustainable development aims to meet present needs while ensuring sustainability of natural systems and the environment so as to not compromise the ability of future generations to meet their own needs. Software maintainability is defined as the ease with which a software system or a component can be modified, to correct faults, improve performance or other attributes, or adapt to a changed environment [3] [4]. The sub-criteria are evaluated in details with respect to our project below.

**Sustainability and maintainability**

| | |
|---|---|
| **Identity**<br>How straightforward is it to learn how to achieve?<br>To what extent is the identity of the project/software clear and unique both within its application domain and generally? | Yes/No, supporting comments if warranted |
| Project/software has its own domain name. | No |
| Project/software has a logo. | No |
| Project/software has a distinct name within its application area. A search by Google on the name plus keywords from the application area throws up the project web site in the first page of matches. | No |
| Project/software name does not violate an existing trade-mark. | Yes, Available in the project report. |
| Project/software name is trade-marked. | No |

| **Copyright**<br>To what extent is it clear who wrote the software and owns its copyright? | Yes/No, supporting comments if warranted |
|---|---|
| Project/software states copyright. | No |
| Project/software states who developed/develops the software, funders etc. | Yes, Available in the application. |
| If there are multiple Project/software then these all state exactly the same copyright, licencing and authorship. | No |
| Each source code file has a copyright statement. | No |
| Each source code file has a licence header. | No |

| **Licencing**<br>Has an appropriate licence been adopted? | Yes/No, supporting comments if warranted |
|---|---|
| Project/software states licence. | No |
| Project/software (source and binaries) has a licence. | No |
| Project/software has an open source licence. | No |
| Project/software has an Open Software Initiative (OSI)-recognised licence. | No |

| **Governance**<br>To what extent does the project make its management, or how its software development is managed, transparent? | Yes/No, supporting comments if warranted |
|---|---|
| Project has defined a governance policy. | No |
| Governance policy is publicly available. | No |

| **Community**<br>To what extent does/will an active user community exist for this product? | Yes/No, supporting comments if warranted |
|---|---|
| Project/software has statement of number of users/developers/members. | No |
| Project/software has success stories. | No |
| Project/software has quotes from satisfied users. | No |
| Project/software has list of important partners or collaborators. | No |
| Project/software has list of the project's publications. | No |

| **Accessibility**<br>To what extent is the software accessible? | Yes/No, supporting comments if warranted |
|---|---|
| Binary distributions are available (whether for free, payment, registration). | No |
| Source distributions are available (whether for free, payment, registration). | No |
| Access to source code repository is available (whether for free, payment, registration). | No |
| Ability to browse source code repository online. | No |
| Repository is hosted externally to a single organisation/institution in a sustainable third-party repository (e.g. SourceForge, GoogleCode, LaunchPad, GitHub) which will live beyond the lifetime of any current funding line. | No |
| Downloads page shows evidence of regular releases (e.g. six monthly, bi-weekly, etc.). | No |

| **Testability**<br>How straightforward is it to test the software to verify modifications? | Yes/No, supporting comments if warranted |
|---|---|
| Project has unit tests. | Yes, Available in the software test plan. |
| Project has component tests. | No |
| Project has integration tests. | Yes, Available in the software test plan. |
| GUI tests are available for project. | No |
| Project has scripts for testing scenarios. | No |
| Project uses automated testing tools. | Yes, Available in the software test plan. |
| Project has automated tests to check conformance to coding standards. | No |
| Continuous integration is supported – tests are automatically run whenever the source code changes. | Yes, Available in the software test plan. |
| Test results are visible to all developers/members. | Yes, We have created a github project. |
| Test results are visible publicly. | No |
| Tests create their own files, database tables etc. | Yes, Tables are shown in the firebase database |

| **Portability**<br>To what extent can the software be used on other platforms? | Yes/No, supporting comments if warranted |
|---|---|
| Application can be built on and run under Windows. | No |
| Application can be built on and run under UNIX/Linux. | No |
| Application can be built on and run under MacOSX. | No |
| Browser applications run under Internet Explorer. | No |
| Browser applications run under Mozilla Firefox. | No |
| Browser applications run under Google Chrome. | No |

| **Supportability**<br>To what extent will the product be supported currently and in the future? | Yes/No, supporting comments if warranted |
|---|---|
| Project/software has page describing how to get support. | No |
| User doc has page describing how to get support. | No |
| Software describes how to get support (in a README for command-line tools or a Help=>About window in a GUI). | No |
| Project has an e-mail address. | No |
| Project e-mail address has project domain name. | No |
| Project/software has site map or index. | No |
| Project/software has search facility. | No |
| Project resources are hosted externally to a single organisation/institution in a sustainable third-party repository (e.g. SourceForge, GoogleCode, LaunchPad, GitHub) which will live beyond the lifetime of the current project. | Yes, Github project is available |
| If there is a blog, is it is regularly used. | No |
| E-mail lists or forums, if present, have regular posts. | No |

| | |
|---|---|
| **Analysability**<br>How straightforward is it to analyse the software's source release to?<br>• To understand its implementation architecture?<br>• To understand individual source code files and how they fit into the implementation architecture? | Yes/No, supporting comments if warranted |
| Source code is structured into modules or packages. | Yes, Available in the android studio project file. |
| Source code structure relates clearly to the architecture or design. | Yes, Available in the android studio project file and system structure. |
| Project files for IDEs are provided. | |
| Source code is commented. | Yes, Available in the android studio project file. |
| Source code comments are written in an API document generation mark-up language e.g. JavaDoc or Doxygen. | No |
| Source code is laid out and indented well. | No |
| Source code uses sensible class, package and variable names. | Yes, Available in the android studio project file. |
| Project-specific coding standards are consistent with community or generic coding standards (e.g. for C, Java, FORTRAN etc.). | Yes, Available in the android studio project file. |

| | |
|---|---|
| **Changeability**<br>How straightforward is it to modify the software to?<br>• Address issues?<br>• Modify functionality?<br>• Add new functionality? | Yes/No, supporting comments if warranted |
| Project has defined a contributions policy. | No |
| Contributors retain copyright/IP of their contributions. | No |
| Users, user-developers and developers who are not project members can contribute. | No |
| Releases document removed/changed components/APIs in that release. | No |
| Changes in the source code repository are e-mailed to a mailing list. | No |

| | |
|---|---|
| **Evolvability**<br>To what extent will the product be developed in the future:<br>• For a future release?<br>• Within a roadmap for the product? | Yes/No, supporting comments if warranted |
| Project/software describes project roadmap or plans or milestones (either on a web page or within a ticketing system). | Yes, Available in the android studio project schedule. |
| Project/software describes how project is funded/sustained. | No |
| Project/software describes end dates of current funding lines. | No |

| | |
|---|---|
| **Interoperability**<br>To what extent does the software's interoperability:<br>• Meet appropriate open standards?<br>• Function with required third-party components?<br>• Function with optional third-party components? | Yes/No, supporting comments if warranted |
| Uses open standards. | Yes |
| Uses mature, ratified, non-draft open standards. | Yes |
| Provides tests demonstrating compliance to open standards. | Yes |

# Chapter 4

# References

[1] https://software.ac.uk/sites/default/files/SSI-SoftwareEvaluationTutorial.pdf.

[2] https://en.wikipedia.org/wiki/Usability

[3] https://isr.uci.edu/content/software-engineering-sustainability-se4s

[4] https://ieeexplore.ieee.org/document/7965364