**System implementation, testing and validation report for <project name>**

*(Formatting is important: Please use Times new Roman, Font 12, spacing should be 1.5, justified alignment, All chapters Must start on a new page, you should have a list of tables, list of figures if any, Table of contents. Table captions must appear at the top of the table, figure captions must be placed at the bottom of the figure, page sections before chapter 1 must be numbers as ii,iii etc, Do not show the page number on the first page )*

|  |  |
| --- | --- |
| Document No: |  |
| Prepared by: |  |
| Date: | DD-MMM-YYYY |
| Version: |  |

Document Approval

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Role** | **Date** | **Signature** |
|  | Author(s) |  |  |
|  | Validation |  |  |
|  | Client |  |  |

**Table of Contents**

**List of Figures**

**List of Tables**

**Chapter 1: Introduction**

**1.1 Background and scope of the project**

**1.2 Overview of the document**

This document describes the implementation, testing and validation findings for the xxx system. It is divided into the following sections:

Section 1: This section gives an overview of the document

**2. System Specifications**

*The section describes and specifies the system completely and is the basis for the validation process.*

**2.1 Version of requirements and Version Control**

*Version of, and changes applied to, the requirements specification. Specify the version of your requirements document. Remember you used to make changes. If you made changes between versions, please specify which changes were made to get which versions. Also, explain why the changes were made*

*How to identify different versions of the computer system and to distinguish output from the individual versions.if you have different versions of your software, how do you know which version is 1.0, or 1.2?? Did you use tools like GIT, TOTROISE ETC to manage versioning of your code?*

**2.2 Input**

*All inputs the computer system will receive. Include ranges, limits, defaults, response to illegal inputs, etc. Please explain in detail what your inputs are for example, if one has a security system GPS coordinates:*

*Input 1: The coordinates are received from the GPS system. When one is attached, the system picks the coordinates and extracts the corresponding locations name and …. which is sent to all phone numbers in the contact list.*

*Input 2: (please don’t simply state the inputs, go ahead and explain them so that the reader fully understands your system)*

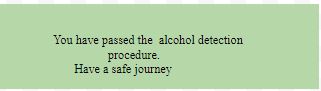
Input 1: Amount of light going through the skin. When this is recorded, the system determines whether the alcohol content in a person’s blood is above acceptable level or not.

**2.3 Output**

*All outputs the computer system will produce. Includes data formats(eg. Images, text, video, etc.), screen presentations, data storage media, printouts, automated generation of documents, etc. do a thorough explanation of all the outputs here*

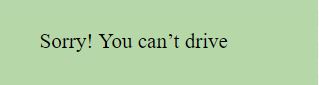
Output 1: User Alcohol Level Acceptable

On successful detection and acceptable level detected, the above message is displayed.

****

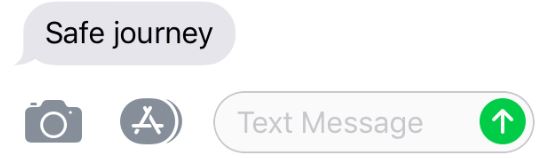
Output 2: User Alcohol Level Unacceptable

When the alcohol level detected is beyond that accepted, the user will be view the message below.

****

Output 3:SMS Notification

The user will receive such an SMS regarding the test results.

****

**2.4 Functionality**

*All functions the computer system will provide. Includes performance requirements, such as data throughput, reliability, timing, user interface features, etc. Explain all the functionalities your system in detail.*

**2.5 Limitations and safety**

*All acceptable and stated limitations in the computer system. What limits the use of the system. Please identify the limitations of the system* ***during its use*** *and NOT during development. Things like The system requires an active Internet connection. If there is no Internet connection, messages sent from users cannot be received. Etc. For embedded systems, they have energy/power limitations, web systems have …*

*All precautions taken to prevent overflow and malfunction due to incorrect input or use. When the system is in use, such and such must be done to prevent incorrect input or use of the system. Eg The administrators shall enter all eligible districts. This is to ensure that only eligible villages are entered into the system. (Give others)*

**2.6 Default settings**

*All settings applied after power-up such as default input values, default instrument or program control settings, and options selected by default. Includes information on how to manage and maintain the default settings. EG By default, when the system is first installed, it contains only one user (give the username and password). For security reasons, the password can be changed via (give the menu name). etc.*

**2.7 Special requirements**

*Requirements the laboratory is committed to, security, confi­dentiality, change control and back-up of records, protection of code and data, precautions, risks in case of errors in the computer system, etc.*

**2.8 Errors and alarms**

*Identify the errors that could arise from the use of the system. How would you handle the errors?*

Sensors not connected well.

SMS could not be sent

Alcohol level content not detected.

*C***hapter 3: Design output**

**3.1 Implementation (coding and compilation)**

*Development tools used to implement the system, notes on anomalies, module and integration details, etc.*

*All device interfaces and equipment to be supported. Identify the devices and explain how they are used in your system. For those doing embedded systems, you have sensors, key pads etc, mobile applications, you have mobile phones, web, you have computers …*

*The hardware and software operating environment in which to use the computer system. E.g. laboratory or office computer, the actual operating system, network, third-party ex­ecuta­bles such as Microsoft® Ex­cel and Word, the actual version of the platform, etc.*

The devices used are shown and listed below.

Spark Fun Triad Spectroscopy Sensor - AS7265x (Qwiic)

GSM module

LCD display

Power Supply

LED Conductor

|  |
| --- |
| **3.4 Documentation**  *What Kind of Documentation has been provided as output from the Design and what is its role to the readers?* |
| ***Table No: Design details (check all that apply to your project. Make sure you can defend what you tick )*** |

| *Topics* | **Design output** | |
| --- | --- | --- |
| **Good programming practice**  *Efforts made to meet the recommendations for good programming practice...* | Source code is... | Source code contains... |
| **Windows programming**  *If implementing Windows applications... remove this row* | Comments: | |
| **Dynamic testing**  *Step-by-step testing made dynamically during the implementation...* | Comments: | |

## Chapter 4: Inspection and testing

**4.1 Introduction**

*The inspection and testing of the computer system is planned and documented in a test plan. The ex­tent of the testing is in compli­ance with the requirements, the system acceptance test specification, the approach, complexity, risks, and the in­tended and expected use of the computer system. (Check what applies)*

**Table No: Inspection plan and performance**

| *Topics* | **3.3.1 Inspection plan and performance** | *Date / Initials* |
| --- | --- | --- |
| **Design output**  *Results from the Design Output section inspected...* | Comments: | Add dates please and initials of members concerned |
| **Documentation**  *Documentation inspected...* | Comments: |  |
| **Software development environment**  *Environment elements inspected...* | Comments: |  |
| **Result of inspection**  *Approval ofinspection.* | Comments: |  |

**4.2 Test plan and performance**

*The test plan is created during the development or reverse engineering phase and identify all elements that are about to be tested. The test plan should explicitly describe what to test, what to expect, and how to do the testing. Subse­quently it should be confirmed what was done, what was the result, and if the result was approved.*

|  |
| --- |
| **4.2.1 Test objectives**  *Description of the test in terms of what, why, and how. Ie why do the test? Steps taken and What was tested* |
| **4.2.2 Scope and Relevancy of tests**  *In terms of coverage, volumes, and system complex­ity. Relative to objectives and required operational use.* |
|  |
| **4.2.3 Levels of tests**  *Module test, integration test, and system acceptance test.* |
| **4.2.4 Types of tests**  *E.g. input, functionality, boundaries, performance, and us­ability.* |
| **4.2.5 Sequence of tests**  *Test cases, test procedures, test data and expected results.* |
| **4.2.6 Configuration and calculation tests**  *Platform, network, and inte­gration with other systems.*  *Calculation tests confirm that known inputs lead to specified outputs.* |
| **4.3 Precautions** |

**4.3.1 Anomalous conditions**

*When operating in a third-party software environment, such as Microsoft® Windows and Office, some undesirable, inappropriate, or anomalous operating conditions may exist. A dis­crepancy between the description of the way an instrument should operate, and the way it actually does, may be regarded as an anomaly as well. Minor errors in a computer system may sometimes be acceptable if they are documented and/or properly cir­cumvented*. *Anomalous operating conditions associated with the computer system itself, Anomalous operating conditions in the used instruments.*

**4.3.2** Precautionary **steps taken**

*The steps taken to workaround anomalous, inappropriate, or undesired operating conditions are verified and tested. Precautionary steps taken in e.g. Windows settings, Precautionary steps taken to workaround problems with the used instruments, Precautionary steps taken to workaround problems with the computer system itself.*

## Chapter 5: Installation and system acceptance test

*The validation of the installation process ensures that all system ele­ments are properly installed in the host system and that the user obtains a safe and complete installation, especially when installing software products.*

**5.1** **Input files**

*List of (relevant) files on the installation media and what they are used for*

Sample files for development. This should however be changed to the user installation files.

However being an embedded system, we are likely to use pictures more to describe our setup.

Arduino IDE

contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.

Proteus

The Proteus Design Suite is a proprietary software tool suite used primarily for electronic design automation. The software is used mainly by electronic design engineers and technicians to create schematics and electronic prints for manufacturing printed circuit boards.

|  |
| --- |
| **5.2 Supplementary files**  *Readme files, License agreements, examples, etc.*  Arduino IDE is developed and maintained by the Arduino team. The IDE is licensed under GPL.  SoftwareSerial library and examples written by Peter Lerup. Distributed under LGPL 2.1.  Spectrometer license |
| **5.3 Installation qualification**  *Steps to ensure and document that each component is installed correctly.*   You can choose between the Installer (.exe) and the Zip packages.  We suggest you use the first one that installs directly everything you need to use the Arduino Software (IDE), including the drivers.  With the Zip package you need to install the drivers manually.  The Zip file is also useful if you want to create a portable installation.  Please allow the driver installation process when you get a warning from the operating system.  Image missing  Choose the components to install  Image missing  Choose the installation directory (we suggest to keep the default one)  Image missing  The process will extract and install all the required files to execute properly the Arduino Software (IDE). |

**Table no: Checklist of the Installation and system acceptance test**

| *Topics* | **Installation summary** |
| --- | --- |
| **Installation method**  *Automatic or manual installation...* | Comments: |
| **Installation media**  *Media containing the in­stallation files...* | Comments: |
| **Installed files**  *List of (relevant) installed files, e.g. EXE- and DLL-files, spreadsheet Add-ins and Templates, On-line Help, etc.* | * DBK files * HEX files * PWI files * Plg files * BAK files * Opt files * C files |

**Table no: Installation Procedure Check**

| *Topics* | **Installation procedure** | *Date / Initials* |
| --- | --- | --- |
| **Authorization**  *Approval of installation in actual environment.* | Person responsible: Keeya Emmanuel. |  |
| **Installation test**  *The following installations have been performed and approved...* | Comments: |  |

## Chapter 6: Performance, servicing, maintenance, and phase out

**6.1 Service and maintenance**

*Documentation of service and sup­port concerning maintenance, fu­ture updates, problem solutions, requested modifications, etc.*

We will provide software support and maintenance services to the end user of the Supported Product.

Support request via email- -none

Support requests via phone--none

Support requests via Chat--available

Service hours of operation

No service will be given :-

* To any modifications to the supported product whether by third parties or anyone else.
* Problems that cannot be reproduced by the team.
* Use of the product in a manner it is not designed for.

**Future Updates**

Feature updates will not be given to the product for:-

* Feature request not urgent
* Problems that have no business impact.

**Problem Solutions**

* Solutions will not be given when:-
* Issue affects only a small group of users.
* Severity decreases when a workaround has been provided.
* Cannot guarantee that every question, problem, issue or Problem reported by a user can or will be resolved.
* Minor/Procedural issue or question e.g., programming or configuration related questions, questions relating to functionality, operability, or formatting or cosmetic problems

**Requested Modifications**

These will not be done when they lead to:-

* Major system disruption a major disruption in business-critical system operability or functionality, or total system failure.
* Single function failure e.g., a minor disruption in operability or functionality that does not impact the entire system such as: timekeeping issues, isolated component failure.

Answer general questions not addressed in the Documentation

**6.2 Performance and Maintenance**

*Which are the requirements for service, maintenance, performance ( Maximum time taken before giving output) , and support(what kind of support do you expect to give to the clients at this stage). This phase is where all activities during performance reside and where deci­sions about changes (What would be the causes of incorporating changes?), upgrades(How should the software be upgraded? ), revalidation, and phase out are made(How do you move from the old system to the new one you have just come up with?? Ie How do you move data from the old system to the new system?). [EXPLAIN THESE IN SOME PARAGRAPHS /PARAGRAPH AND LATER FILL THE TABLE BELOW]*

An example of an End of Life product is one that will no longer be supported.

Defects in the Software due to accident, hardware malfunction, abuse or improper use.

Training, customization, integration and any issues arising from non‐standard usage of the system.

Interface modification

The electronics industry is one of the fastest growing sectors of the world economy. Those new electronic components with faster speed, smaller size and lower power consumption will quickly dominate the market. Therefore, the occasion might arise in which electronic components which are the component parts of the product have a shorter life cycle than the actual life cycle of the product.

If the product is not popular in the market and becomes unprofitable.

Peripheral interface obsolete.

The peripheral interface standard is developing. A new standard will rapidly enter the mainstream based on its improved specifications. For instance, the USB (Universal Serial Bus) has become the most popular peripheral interface standard for consumer products during the past few years. The earlier IEEE 1284 parallel interface is no longer able to be supported in most devices. Thus, it becomes a possibility that long life cycle systems will suffer from the problem of interface mismatch because of these modern peripherals.

Component obsolete.

A component is a product provided by a vendor, which for the majority, contains some unique properties and cannot be replaced by a product from other vendors. Component obsolescence is a severe case since it frequently occurs.

The availability of newer and better architectures (processors, interconnections and interface blocks) can provide the motivation for the reengineering of a product.

Better circuit technology migration

Better performance, lower price or being friendlier towards the environment would force the system to migrate to a new circuit technology.

Vendor device migration: Vendor portability is a special issue for …………..

Function change requirement

For different requirements, add, delete or modification of functions are inevitable. For example……..

The software application of an embedded system is usually programmed in a high level language (e.g. C), which always contains some processor dependent code. Even if it is a non-processor-specific code, the device drivers and device management, initialization and locator modules and initial boot-up record data require modifications when the hardware changes. The majority of the currently available software written for embedded systems is almost 100% target dependents. Any change to the hardware requires a change in the software. If the microcontroller is replaced by a different one, the software has to be consistently modified. The development environment, such as the compiler and library, can also be different.

**Support currently offered**

**Table no: Performance and maintenance details**

| *Topics* | **Performance and maintenance** | *Date / Initials* |
| --- | --- | --- |
| **Problem / solution** | *Detection of system problems causing operating troubles. A first step could be to suggest or set up a well-documented temporary solution or workaround.*  The voltage required by the spectrometer was lower than we were producing. Lowering of this had to be done with a shield or resistors.  Phone numbers has to be hardcoded unlike our earlier desire to use earlier registration. | Dates must be filled in  12/07/2020 |
| **Functional maintenance** | *E.g. if the computer system is committed to in­ternational standards, and these standards are changed, the computer system, or the way it is used, should be updated ac­cordingly.* |  |
| **Functional expansion and performance im­provement** | *List of suggestions and requests, which can improve the performance of the computer system. eg*   * Inclusion of a PH sensor to determine the soil PH levels to allow for automatic neutralization of the soil. * Use of wireless technology instead of wired connections. |  |

# Chapter 7: Conclusion and Recommendations

*Make a conclusion of your whole report*

**Appendix A: User Manual**

*Give details on how the system can be used, where to go for help etc accompany your explanations with screenshots*

(*For those doing embedded systems, please add your assembled diagram, those doing other projects can add some screenshots to their manual. The screenshots must appear in a logical order)*

| **Final approval for use** | | |
| --- | --- | --- |
| Identification: | |  |
| Responsible for validation: | |  |
| Remarks: | | |
| Date: | Signature: | |