Project Thor

Requirements Document

Version 1.0

Team Members:

Adonay Pichardo
Jared Blanco
Josh Temel
Luke Bonenberger

Faculty Advisor:
Ph.D, Sid Bhattacharyya

Client:
Ph.D, Amitabh Nag

Florida Institute of Technology Senior Design

Sep 22, 2021

Table of Contents

1. Introduction

1.1. Purpose

This Requirements Document (RD) is dedicated to providing a comprehensive set of requirements applicable to the project together with relevant justifications. Within will be a detailed guide to ensure the agreeable development, testing, and implementation of the project between the Project Thor team members and the client. Additionally, it will act as a resource for both the team members and faculty advisor to reference throughout the course of the project to monitor and determine the project's status, proper fulfillment of customer's needs, and overall project success.

1.2. Document Conventions

To allow formal traceability of the different requirements, a unique identifier is used in line with a set of acronyms (i.e. Data to show Database Requirements, Web to show Web Application Requirements). In addition to these identifiers, the requirements follow a numbering format that shows their relation to each other using a numbered list method. Furthermore, the requirements have been broken down by type. The types are defined as follows:

- Performance: How well a task or action must be completed.
- Interface: How information must be displayed back to the user.
- Maintainability: Related to the ease of how the product must be maintained, repaired, reformatted, or replaced.
- Availability: Defines when the solution must be able to be used or accessed by users.
- Supportability / Sustainability: Related to compatibility and abilities to test, adapt, configure, install, scale, localize, etc.
- Security: Defines the security of the systems and services.
- Safety: Requirements that are defined for the purpose of risk reduction.
- Human Factors: Refer to the ease, usability, and user satisfaction.

1.3. Intended Audience

While the software requirement specification (SRS) document is commonly written for a more general audience, this document is intended for individuals directly

involved in the development of the project. This includes the project's team members and faculty advisors. This document need not be read sequentially; users are encouraged to jump to any section they find relevant. However, it is recommended that users who are new to the project begin reading with the Project Scope and Possible Solution. In order to gain a better understanding of the project.

1.4. References

1.4.1. Public Project Website: https://killjaqular.github.io/thor
1.4.2. Private Github Repository: https://github.com/killjaqular/thor
1.4.3. IEEE Recommended Practice for Software Requirements Specifications: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=720574

2. Scope

2.1. Problem

Society has placed an ever-increasing dependency on using computers in day-to-day life. From making our personal lives easier through instant messaging and social media to controlling power stations and traffic management systems. Along with this increased dependency on computers came an increased need for those computers to encrypt and transmit data securely. Unfortunately, modern software and hardware advances have led to a decrease in the reliability of these encryption methods. The problem lies in the predictability of the pseudo-random numbers used to seed these algorithms. Therefore, we need to find a non-deterministic process of gathering truly random numbers from lightning phenomena to help make our encrypted data more secure compared to current practices.

2.2. Solution

Current best practices generally employ natural phenomena to inject randomness into the data sets used to generate random numbers. A company in Switzerland uses a point in time at which a radioactive source decays to create more random numbers, and Random.org uses a method that incorporates electromagnetic atmospheric noise samples. Both methods work but are cumbersome. Therefore, our team seeks to use data collected from lightning strikes across the globe to create a data set with a higher level of variation than those used in today's industry and implement it in a web application that generates encryption keys.

2.3. Project Scope

In an ideal world, the concepts and methods developed throughout this project will be published and adapted as an industry standard. However, this project's scope is much more realistic and will be constricted to three deliverables along with the results of some data analysis. Our deliverables will consist of a database containing a limited data set of measurements from lightning strikes, a web application that uses that data to create encryption keys for popular encryption algorithms, educates the user about that process, and automation that transfers the raw ASCII data into the database for its continued growth and evolution. Furthermore, the database will come with an analysis that determines and describes the entropy of the data. Although applicable, the data used in this project will not be considered for other computer science applications like game theory and the prediction of weather patterns, but we will instead focus solely on the creation of high entropy, random numbers for use with encryption algorithms.

3. Web Application Requirements

In the following section, the functional and non-functional requirements pertaining to the web application are listed. Each requirement is identified by a unique requirement ID and further defined by the type of requirement it is.

3.1. Functional Requirements

Req ID	Type	Functional Requirement
WEB-1	Interface	The web application shall display a tooltip whenever the user's cursor has hovered over icons or command buttons.
WEB-1.1	Interface / Human Factor	The web application's tooltips shall have a minimum of 9 point font.
WEB-2	Interface	The web application shall be capable of presenting the generated key from the database to the user.
WEB-3	Interface	The web application shall display a graphic showing the location where the generated number was collected.
WEB-3.1	Interface	The graphic will show the geographical Lat-Lon, time interval, peak current, rise time, and peak to zero.
WEB-4	Supportability/ Sustainability	The web application shall be loadable on modern desktop and mobile operating systems. This type of design is referred to as responsive web design, where the web pages render well on various device sizes (i.e. Windows, macOS, iPadOS, iOS, Linux, Android, etc.).
WEB-5	Supportability/ Sustainability	The web application shall support web browsers: Google Chrome, Safari, Firefox.

3.2. Non-functional Requirements

Req ID	Type	Non-functional Requirement
WEB-6	Performance	The web application shall be capable of loading and displaying data from the database quickly, within 3 seconds.

WEB-7	Performance	The web application shall be able to execute commands within 1 second of receiving input.
WEB-8	Interface	The web application shall show a simplistic interface displaying all commands.
WEB-8.1	Interface	The web application shall have a “Generate Key” command which presents the user with a random number (aka a key).
WEB-8.2	Interface	The web application shall have a “Contact” command that presents the user with a contact form to send a message to the project team.
WEB-9	Availability	The web application shall be available 24 hrs a day when not under maintenance.
WEB-10	Security	The web application shall only allow public read-only access.
WEB-11	Human Factors	The web application shall have all text sized in a minimum of a 9 pts font.
WEB-12	Interface	The web application shall have a minimum of 3 pages.
WEB-12.1	Interface	The web application shall have a page to provide users with the Thor service detailing how the key was generated, and what details it has.
WEB-12.2	Interface	The web application shall have a page to facilitate contacting Thor Team members.
WEB-12.3	Interface	The web application shall have an about page
WEB-12.4	Interface	The web application shall have a link to the separate project website

4. Database Requirements

In the following section, both the functional and non-functional requirements that pertain to the database are listed. Each requirement is identified by a unique requirement ID and further defined by the type of requirement it is.

4.1. Functional Requirements

Req ID	Type	Functional Requirement
DATA-1	Availability	The database shall be available for maintenance and use through SSH.
DATA-2	Maintainability	The database shall be able to roll back to previous states.
DATA-3	Sustainability	The database shall be hosted on a team member's private local machine.
DATA-4	Security	The database shall utilize standard sanitization techniques of SQL commands.

4.2. Non-functional Requirements

Req ID	Type	Non-functional Requirement
DATA-5	Performance	The database shall return SQL requests in real-time.
DATA-6	Performance	The database shall accept SQL requests from the web application.
DATA-6.1	Performance	The database shall be capable of updating the database as the ASCII data is updated in real-time.
DATA-7	Availability	The database shall be capable of being duplicated to a cloud service.
DATA-7.1	Maintainability	The database shall be hosted from a raspberry pi so ownership can be transferred.
DATA-8	Maintainability	The database shall have the ability to be scalable as required.

DATA-9	Availability	The database shall be available 24 hrs per day when not under maintenance.
DATA-10	Security	The database shall require authorization to send SQL requests.
DATA-11	Performance	The database shall have preventative measures for poor data being added.
DATA-11.1	Performance	The database shall prevent low-entropy numbers from being added to the data set.
DATA-11.2	Performance	The database shall have mitigation measures in place to prevent duplicate data from being added.
DATA-12	Performance	The database shall have a data set with a non-0 high variance.