The Mess Matrix

Software Requirements Specification (SRS) Document

SRS-TheMess(Draft).docx

Version 1.1

January 28th, 2019

The Mess

1 Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Initial (1.0)	Skyler Knecht Ebraheem Aldubis Sam Jones Mike Jeromin	Initial Draft	2/4/2019
Final (1.1)	Skyler Knecht Ebraheem Aldubis Sam Jones Mike Jeromin	Final Draft	2/6/2019

2 Contents

3 INTRODUCTION 4

3.1 Purpose 4

3.2 Description 4

3.3 Overview 4

4 OVERALL DESCRIPTION 5

4.1 Product Perspective 5

4.2 Product Functions 5

4.3 Similar Systems 5

4.4 User Characteristics 5

4.5 User Objectives 5

4.6 Constraints 5

5 SPECIFIC REQUIREMENTS 6

5.1 Functional Requirements 6

5.2 User Interface Requirements 8
5.2.1 User Interface: Graphical (GUI) or Command-Line (CLI) 8
5.2.2 Application Programming Interface (API) 8
5.2.3 Diagnostics (Error Reporting and Usage Logs) 9
5.2.4 System Requirements 9
5.2.5 Hardware Interfaces 9
5.2.6 Communication Interfaces 9
5.2.7 Software Interfaces 9

5.3 Non-Functional Requirements 9
5.3.1 Usability 9
5.3.2 Reliability 9
5.3.3 Security 9
5.3.4 Maintainability 9

5.4 Logical Database Requirements 10

6 SOFTWARE LIFE CYCLE MODEL 11

6.1 Choice of Software Life Cycle Model 11

6.2 Justification for Choice of Model 11

3 Introduction

3.1 Purpose

The purpose of this document is to specify the functionality of the Mess Matrix and the requirements for the system’s development.

3.2 Description

The Mess Matrix will allow users to protect themselves against an observer stealing a text password at login by using a 6x6 matrix, which is filled by random characters from a-z and 0-9. Through using a matrix-based approach the Mess Matrix will provide better security against shoulder surfing attacks, since users utilize a mouse to enter their passwords rather than typing on a keyboard which may be observed by a potential password thief.

3.3 Overview

The following section will provide an overall description of the project detailing the product perspective, product functions, similar systems, user characteristics, user objectives and constraints. Section five will detail the specific requirements that need to be met throughout this project. Finally, section six will describe the life cycle model chosen and why it was chosen. To find a section quickly please refer to the table of contents on page 3.

4 Overall Description

4.1 Product Perspective

This product will be a web-based application that Users can access via a web browser where they will be greeted with a login form.

4.2 Product Functions

The intended function of this product is to provide a login form that has the ability to enter a password in two ways. The first method of logging in would be a standard keyboard-based password authentication. The second method would be the ability to click a button to open the Pass Matrix. This pass matrix should allow a user to manually enter in their password via mouse clicks in a 6x6 matrix.

4.3 Similar Systems

Although there are many papers written on the subject there are few similar systems to the one described in this document.

4.4 User Characteristics

The intended user is anyone who has used a keyboard-based authentication before and is interested in alternative authentication methods to avoid shoulder surfing. Users will need to maintain the ability to navigate through web pages and know how to successfully enter their password in a 6x6 matrix.

4.5 User Objectives

Users should expect a fully-functional login system, with the ability to register, login, and modify their username and password after a successful login. Users should also expect a high level of usability, security and accessibility.

4.6 Constraints

Passwords for this proof-of-concept system are limited to alpha-numeric characters only. Punctuation and other special characters are not allowed as legal password characters for the Mess Matrix system, to keep the number of characters that must be displayed in the matrix to a minimum.

5 Specific Requirements

5.1 Functional Requirements

Priority Scale: Low (1) – Medium (2) – High (3)

1. Low: Items that can be eliminated should the need arise, without adversely affecting the product. These items are not urgent and not as important to the final project.
2. Medium: Items that are desired by the customer and/or users of the system, but that may be postponed until a future release. These items are not urgent and but are important parts of the final project.
3. High: Items that are mission critical and without which the system cannot function in a manner that is satisfactory to the customer. These items are urgently needed and important to the success of the final project.

Functional Requirement 1

Name: Main Page
Description: Create a main website for the Mess Matrix. Allows user to login and create a new account.
Priority: High
Technical Issues: website shall be properly coded and Wi-Fi will be needed.
Risks: The system may not display the homepage
Dependencies: None.
Function Type: Functional

Functional Requirement 2

Name: Login Page
Description: Users who have login credentials for the Mess Matrix website will already have a user account in the system and can login using their Mess Matrix username and password. Logging in will bring the user to their profile page.
Priority: High
Technical Issues: The user will have to be connected to Wi-Fi and the database server.
Risks: The user will not be able to login.
Dependencies: 1,4
Function Type: Functional

Functional Requirement 3

Name: New User Page
Description: Users who do not currently have login credentials for the Mess Matrix website will be able to set up a new account for themselves by clicking the new user button. Setting up a user account will allow the user to create a username and password for themselves. Logging in will bring the user to their profile page.
Priority: High
Technical Issues: The user will have to be connected to Wi-Fi and the database server.

Risks: The user will not be able to create an account.
Dependencies: 1,4
Function Type: Functional

Functional Requirement 4

Name: Database/File
Description: Create a database or file that will store the user's usernames and passwords.
Priority: High
Technical Issues: The user will have to be connected to webpages
Risks: If failed the whole project would fail
Dependencies: 2,3,6
Function Type: Functional

Functional Requirement 5

Name: Mess Matrix
Description: Create a screen GUI to allow the user to input their password. The matrix is to change characters after each click
Priority: High
Technical Issues: The user will have to be connected to Wi-Fi and the database server.
Risks: Certain characters will not be displayed.
Dependencies: 4
Function Type: Functional

Functional Requirement 6

Name: Profile or User Account Page
Description: Page dedicated for showing saved events that the user is interested in. From here, the user can view information for their events.
Priority: High
Technical Issues: The user will have to be connected to Wi-Fi and the database server.
Risks: The user will not be able to view their profile information.
Dependencies: 2,3,4
Function Type: Functional

Functional Requirement 7

Name: Encrypt maker
Description: Creating an Encryption algorithm that allows the passwords to be encrypted
Priority: Medium
Technical Issues: The user will have to be connected to the database server.
Risks: The user will not be able to encrypt the password.
Dependencies: 4
Function Type: Functional

Functional Requirement 8

Name: Forgotten Password
Description: User can't remember her/his password, the system asks user a security question, or sends an e-mail to the e-mail address stored in file in the user's profile.
Priority: Low
Technical Issues: The user will have to be connected to the database server.
Risks: The user can't answer the security questions correctly and loss of password/account.
Dependencies: 4,6
Function Type: Non-Functional

Functional Requirement 9

Name: Special Characters
Description: If the user attempts to enter special characters or spaces when settig her/his password the first time, the system must not allow them to be included in the password. An error message must be displayed and the user must re-enter the password on the keyboard.
Priority: High
Technical Issues: The user will have to be connected to the database server.
Risks: Only alpha-numeric characters are allowed in passwords in this system.
Dependencies: 4,6
Function Type: Functional

Functional Requirement 10

Name: User/Usage Logs
Description: Keeps track of User log-in and usage of system.
Priority: Low
Technical Issues: The user will have to be connected to the database server.
Risks: Keeps all sensitive info in one place.
Dependencies: 4
Function Type: Non-Functional

5.2 User Interface Requirements

5.2.1 User Interface: Graphical (GUI) or Command-Line (CLI)

The Mess Matrix will be accessible through a GUI in a standard web browser. This GUI will contain text fields and buttons for users to navigate the application appropriately. Upon navigating the webpage users may be prompted with messages (e.g., successful and unsuccessful login attempts).

5.2.2 Application Programming Interface (API)

The Mess Matrix will only be accessible through a web application.

5.2.3 Diagnostics (Error Reporting and Usage Logs)

Administrators will have access to the system logs via logging into the server where the web application is being hosted.

5.2.4 System Requirements

The Mess Matrix requires users to maintain an up-to-date browser and access to the internet.

5.2.5 Hardware Interfaces

The Mess Matrix will be hosted on a standard web server so there will be no hardware interfaces required.

5.2.6 Communication Interfaces

The Mess Matrix will be hosted on a standard web server and must be able to communicate with the file explorer and database to function correctly.

5.2.7 Software Interfaces

Users should have an up to date operating system such as:

OSX 10.9 or newer

Windows 7 or newer

5.3 Non-Functional Requirements

5.3.1 Usability

The mess matrix should be fairly easy to navigate to brand new users.

5.3.2 Reliability

As long as there is access to the internet and the web server is running properly the application should run as expected.

5.3.3 Security

Due to the nature of the application there shall be some type of encryption or hashing of saved user credentials. All database software shall be fully updated to avoid viruses, malware and malicious injections.

5.3.4 Maintainability

The program will be very maintainable with the correct software life cycle model in place. This life cycle model will enable maintainability pre and post competition of the project.

5.4 Logical Database Requirements

The program requires a database to store user credentials and/or personal information (e.g., first and last name).

6 Software Life Cycle Model

6.1 Choice of Software Life Cycle Model

For the development of the program, the Kanban Life Cycle Model will be implemented.

6.2 Justification for Choice of Model

As a team, we decided that the Kanban life cycle model would be the best method to use for our program. We believe this because Kanban was originally an Agile life cycle model that focused primarily on continuously delivery of a project without overworking the development team. Kanban would allow for every member in the project to know exactly where they were in the project development cycle by being able to visualize the work through a well-planned out task board. To implement this feature, we will be digitalizing our board by using an app that is called SmartSheet. We will also be able to release multiple versions of the application by prioritizing features to guarantee that all must add value to the system to make it into the application.