Software Requirements Specification

for

An Android Mobile App for Phishing Attack Detection through Delusory Login Simulation.

**Version 1.0 approved**

**Prepared by:**

Mayank Agarwal

Akshay Agrawal

Akshay Kumar

Rakhi Awari

**D. Y. Patil College Of Engineering**

**9.07.18**

**Table of Contents**

**Table of Contents [ii](#_gjdgxs)**

**Revision History [ii](#_30j0zll)**

**1. Introduction [1](#_1fob9te)**

1.1 Purpose [1](#_3znysh7)

1.2 Document Conventions [1](#_2et92p0)

1.3 Intended Audience and Reading Suggestions [1](#_tyjcwt)

1.4 Product Scope [1](#_3dy6vkm)

1.5 References [1](#_1t3h5sf)

**2. Overall Description [2](#_4d34og8)**

2.1 Product Perspective [2](#_2s8eyo1)

2.2 Product Functions [2](#_17dp8vu)

2.3 User Classes and Characteristics [2](#_3rdcrjn)

2.4 Operating Environment [2](#_26in1rg)

2.5 Design and Implementation Constraints [2](#_lnxbz9)

2.6 User Documentation 2

**3. External Interface Requirements [3](#_44sinio)**

3.1 User Interfaces [3](#_2jxsxqh)

3.2 Hardware Interfaces [3](#_z337ya)

3.3 Software Interfaces [3](#_3j2qqm3)

3.4 Communications Interfaces 3

**4. Other Nonfunctional Requirements [4](#_3whwml4)**

4.1 Performance Requirements [4](#_2bn6wsx)

4.2 Safety Requirements [5](#_qsh70q)

4.3 Security Requirements [5](#_3as4poj)

4.4 Software Quality Attributes [5](#_1pxezwc)

4.5 Business Rules [5](#_49x2ik5)

**Appendix A: Glossary [5](#_147n2zr)**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

The purpose of this document is to build an android app for phishing detection through deceptive login simulation.

## Document Conventions

This document uses the following conventions:

DB Database

IDB In App Database

PTR PhishTank Repository

## Intended Audience and Reading Suggestions

This project is a prototype for the common user and is not restricted within the college premises but accessible anywhere by any android mobile phone. This project may also be very useful for IT professionals, developers and testers.

## Product Scope

The purpose of the android app is to generate a kind of alert every time a user unwillingly access a not so legitimate web URL/page. This system is based on a repository/database which contains the already reported phishing websites and webpages. We will have this database which will be used to generate the phishing alert. Above all we hope to provide a less risky and a newbie user experience. The system has been designed for deceptive phishing. In future several other types of phishing including Malware-based phishing, Web Trojans, Man-in-the-middle attacks etc. can be implemented with the same android app.

## References

<https://krazytech.com/projects>

<https://www.phishtank.com/>

# Overall Description

## Product Perspective

Attackers use clever social engineering techniques to convince their victims into clicking a malware or deceptive login-based webpages. Most solutions for this particular problem focus more on helping desktop computer users than mobile device users.

The system enables a mobile device user to create fake login account, with fake login credentials, that mimics user login procedure every time the user opens a login webpage and generates an alert.

## Product Functions

* Registration: This page enable a user to register him/her in the app.
* Login : This page will be used for authentication purpose.
* Fake login input credentials: User need to provide fake credentials which will be used for authorization purpose in the various websites.
* Help: Allows the user to enquire about any functional ability.
* About: This section provides the information about the app.

## User Classes and Characteristics

* Typical Users, who want to use the system for avoiding online mishaps such as stealing of credentials through phishing.

(Social networks, Social Media networks, Semantic networks etc.)

* Advanced/Professional Users, such as engineers or researchers, who want to use

the system for learning, detecting and analysing different phishing attacks.

* Programmers who are interested in working on the project by further developing it

or fix existing bugs.

## Operating Environment

The system is made to operate in any mobile android system with a version of at least 4.0.

All the devices had at least 853 Mb internal memory and 1400 MHZ processor.

## Design and Implementation Constraints

The system is developed in Java, it uses XML for its visualization design and has been

built on top of the Android Studio Platform.

It uses a modular design where every feature is wrapped into a separate module and the modules depend on each other through well-written APIs.

## User Documentation

There is a quick start guide available on the website of PhishTank:

<https://www.phishtank.com>

For any query related to phishing, its types and countermeasures, one can visit the following website:

<https://www.phishtank.com/faq.php>

Additional help and information can be found at:

<https://en.wikipedia.org/wiki/PhishTank/>

# External Interface Requirements

## User Interfaces

* Registration interface
* Login interface
* Fake input credentials interface
* Every interface has two buttons: Help and About

## Hardware Interfaces

* Several small size devices such as smart-phones and tablets with Android operating

systems

* The devices link with a Asus machine installed with XAMPP server having core i5-6400 CPU with 2.70GHz and 8Gb internal memory
* All the devices had at least 853 Mb internal memory and 1400 MHZ processor.

## Software Interfaces

Following are the software used for the system:

Operating System : Andriod OS greater then version 4.0

Database : Phishtank repository for comparing website hashcodes, MYSQL.

Server : XAMPP web server

## Communications Interfaces

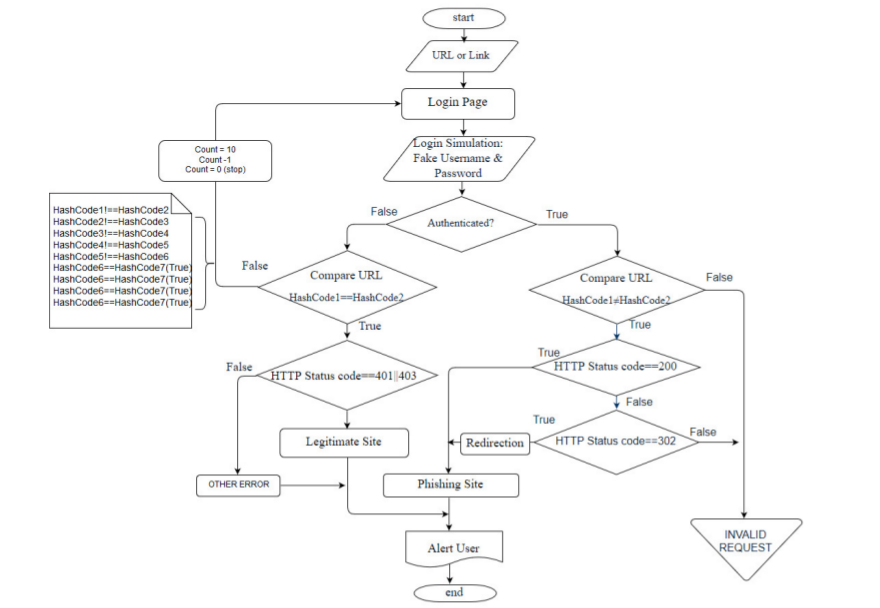
The project supports chrome web browser as of now. other web browsers like mozilla firefox and safari may also be supported in the future.

We are using simple In-app login formats for user registration and logins.

# Other Nonfunctional Requirements

## Performance Requirements

* **Workflow:** The following figure shows the workflow of the overall system



* **Normalisation:**

The basic objective of normalization is to reduce redundancy which means that information is to be stored only once. Storing information several times leads to wastage of storage space and increase in the total size of the data stored. If a database is not properly designed it can give rise to modification anomalies. Modification anomalies arise when data is added to, changed or deleted from a database table. Similarly, in traditional databases as well as improperly designed relational databases, data redundancy can be a problem. These can be eliminated by normalizing a database. Normalization is the process of breaking down a table into smaller tables. So that each table deals with a single theme. There are three different kinds of modifications of anomalies and formulated the first, second and third normal forms (3NF) is considered sufficient for most practical purposes. It should be considered only after a thorough analysis and complete understanding of its implications.

## Safety Requirements

If there is extensive damage to a wide portion of the database due to catastrophic failure, such as a disk crash, the recovery method restores a past copy of the database that was backed up to archival storage (typically tape) and reconstructs a more current state by reapplying or redoing the operations of committed transactions from the backed up log, up to the time of failure.

## Security Requirements

Security systems need database storage just like many other applications. However, the special requirements of the security market mean that vendors must choose their database partner carefully.

## Software Quality Attributes

* Correctness: It is effective in assisting users to identify phishing attacks with an accuracy of 96%.
* Adaptability: The system is adaptable to work any android mobile device having version 4.0 or above.
* Usability: The system should satisfy maximum number of user needs/requirements.
* Maintainability: The system is not very complex to maintain because of limited number of modules.

**Appendix A: Glossary**

**Phishtank**: PhishTank is a collaborative clearing house for data and information about phishing on the Internet.

**Deceptive phishing**: deceptive phishing refers to any attack by which fraudsters impersonate a legitimate company and attempt to steal people's personal information or login credentials.

**XAMPP**: XAMPP is a free and open source cross-platform web server solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters

**Flowchart**: A flowchart is a type of diagram that represents an algorithm, workflow or process. The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows.

**Hashcode**: n the Java programming language, every class implicitly or explicitly provides a hashCode() method, which digests the data stored in an instance of the class into a single hash value (a 32-bit signed integer).

**Repository**: a central location in which data is stored and managed.

**HttpConnectionStatus**: Sets whether HTTP redirects (requests with response code 3xx) should be automatically followed by this HttpURLConnection instance.