**Software Requirements Specification**

**for**

ONE CLICK AUTH

**Version 1.0**

**Prepared by R.Mohan Rajan, R.Karthick**

**College of Engineering Guindy,  
Anna University -95.**

**17-July-2016**

**Table of Contents**

**Revision History**

**1.** **Introduction**

1.1 Purpose

1.2 Intended Audience and Reading Suggestions

1.3 Project Scope

**2.** **Overall Description**

2.1 Product Features

2.2 Operating Environment

2.3 Design and Implementation Constraints

2.4 Assumptions and Dependencies

**3.** **Functional Requirements**

3.1 Session Management - Standard:

3.3 Session Management - Search:

3.4 Session Management – Sub Navigation:

**4.** **External Interface Requirements**

4.1 User Interface

4.2 Hardware Interfaces

4.3 Software Interfaces

4.4 Communications Interfaces

**5.** **Other Nonfunctional Requirements**

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Software Quality Attributes

**6.** **Platform And Technologies**

**7. Estimation using COCOMO Model**

**8. References**

# Introduction

## Purpose

The purpose of this project is to use single authentication for sign in of multiple websites without filling forms in every page using different social network. One-Click Auth provides a simple and a secure way to provide developers to create a simple login button that will take care of all the API calls required to connect users with the social networking sites. This makes sure that the user information is safely retrieved and utilized. It is a B2C (Business to Customer) platform where it tends to help the developers as well as the users with the ease of signing in, hence reducing the time spent.

## Intended Audience and Reading Suggestions

The document is intended for developers who wants to know about the internal functioning of the system and the API calls along with it. The user can access the information in this document if and only if he is a part of the organization developing this software. This document contains the relevant details for modifying the system to their needs.

## Project Scope

The objective of this project is to develop a single authentication system which can be accessed by the developers to include in their site and provide an interface to connect to login provided by various social networking sites. This reduces the effort taken by the developers to integrate social networking logins in to their site.

# Overall Description

## Product description

The objective of our project is to develop a system that enables developers to easily amalgamate social networking logins in an effective manner. It can be agreeable for developers to have this feature in order to deploy their website quickly without much effort. In a straightforward manner, the developers need not look at the entire documentation of each social networking which they intend to coalesce into their site. Instead they can read the documentation of this project and they can easily combine various social networking logins, thus helping them to develop a site in an expeditious manner. The dashboard will be provided for developers who can control the entire action performed through their website. The use of this system provides security and integrity of the data, so that users need not worry about the data being stolen by others. This ensures trustworthiness of this system.

## Operating Environment

The system will serve as Application Programming Interface(API). In addition to this, the developers can manage accounts of various users with the help of dashboard provided for each and every developer which is an additional feature provided to the them so that they can monitor the user and can able to blacklist individual user in case of any illegal activity. The intent of this application is to provide a rapid application development for the developers with minimal coding for their site as well as providing maximum efficiency.

## Design and Implementation Constraints

The system must be hosted on a cloud hosting service. The system runs on linux distribution platform with API calls specified to run the application by the developers. The main constraint to consider in developing the system is how many concurrent API calls the application can handle at a particular point of time. The database schema must be designed efficiently so that only the vital information which the developer will need are to be stored thereby saving memory space. Moreover the system should be secure throughout the whole process, the API calls provided by the API generated by this application should be secure apart from the calls done by the other network API’s.

## Assumptions and Dependencies

The Users who are using this system has to have a fair knowledge in the use of base networking sites such as Facebook, Google etc. The developers also has to specify the target users who will be using the system. This application is dependent on the users and made sure it is understandable to every user. Developers need to make sure that they utilize this system in a coherent way so that they can minimize the API calls made to the system to increase the speed of serving to the user. The application is dependent on the API’s of the networking sites and its version changes. So for each version change, there will be an update to change the API calls given to them respectively.

# Functional Requirements

## Session Management - Standard Login:

FR-1: The developer shall be able to create a new account by providing his details. If he already has an account he can login straightaway.

> If the login is correct , then the user will be prompted in front of the dashboard

> If the login is incorrect, an “Error:Login” Statement is thrown and the log is recorded

> If the developer signed up successfully, then the developer can create an app.

FR-2: The user(different login screen) shall be able to create a new account by providing his details. If he already has an account he can login straightaway.

> If the login is correct , then the user will be prompted in front of the website the developer created

> If the login is incorrect, an “Error:Login” Statement is thrown and the log is recorded

> If the user signed up successfully, then the developer will let the user to enter their website

## Session Management – Dashboard:

FR-1: The developer can view the documentation to implement the various API calls

FR-2: The developer on clicking the User icon can view the users who are in the application

FR-3: The developer can check user profiles in the user icon and can see their login attempts along with their last logins.

FR-4: The developer can check for any logs in the server if in case he has to check for any errors recorded.

FR-5: The developer has the option to enable multi factor authentication and email notifications in the dashboard. This will send notifications with content specified.

# External Interface Requirements

## User Interface

UI-1 Login: The Developers will be provided with a login portal to login to their dashboard.

UI-2 Dashboard(Main): The Dashboard will show stats of users who have logged in using API calls, users who have tried to login and new sign ups. This will also ask the developer whether to allow Multi Factor Authentication and Email Notifications.

UI-3 Documentation: The developer has a documentation page to know the type of url’s the developer has to call to do the JSON requests.It also gives examples of what all JSON requests will look like.

UI-4 Dashboard(Users): This interface will have a list of users who have signed up. Each user upon creation will have a profile created.

UI-5 Dashboard(Users): This interface will have a list of users who have signed up. Each user upon creation will have a profile created.

UI-6 Logs Board: This will show the list of logs that the server is facing, if incase the developer wants the application to be transparent. This allows developer to be safe around a transparent application.

## Hardware Interfaces

This will be an web application, and as such, will be designed to interface with the hardware present on the Ubuntu Operating System running Apache server with MySQL Support. In theory the application will be able to run by other devices that can emulate websites ( like Chrome, Firefox in Mobile and Desktop versions) which will be a major consideration in the design.

The RAM and the storage provided for each application the developer creates will be minimal and will expand according to his need. The application to run in a large scale must have a 4GB RAM, running on a 50GB SSD with 5TB Bandwidth capacity needs. We leave the developers to choose their own server capacity.

## Software Interfaces

The software interface of the system can classified into 3 types:

1.Operating System : This runs on a Ubuntu operating system which is a linux distribution which provides the application with only command line activities possible. This is because of the fact that this Ubuntu is run in a cloud server which is booted with the help of virtual machines.

2.Server : The webserver is run using Apache installed on ubuntu. Apache is used because of the open source and its lightweightness with security inbuilt.

3.Bankend DB: MySQL provides the support of database. PHP is used as the backend language along with JS and to connect database with Mysql , phpmyadmin is used to manage the database.

# Other Nonfunctional Requirements

## 5.1 Performance Requirements

The primary performance requirement is speed of the network and the sdk implementation of respective social networking sites. The performance depends on the effective retrieval from the database, so designing the database schema can improve the presentation to the user as they need not wait for time taken needed to retrieve data and the reliability of the network.

## 5.2 Safety Requirements

There are no safety requirements with this application. Only thing to be considered about safety is that the information kept on the main server is not accessible to everyone and no physical security breach is possible.

## 5.3 Security Requirements

The system must be able to link up with various social networking sites and the information collected from various social sites are stored in a well organised manner in order for users to properly log in and be identified. This information must be kept invulnerable. In case of any security breach, the information stored must be encrypted so that the data cannot be stolen and sold to others.

## 5.4 Software Quality Attributes

The primary attribute of this application will be usability given the large amounts of data and information collected from various social sites will be presented on such a small screen which we provide it as a dashboard to the developers. In addition to this, it depends on the developer's ability to handle the data and apply it into their site in a sensible manner so that they can administer the site in an effortless way.

As usability is concerned, it solely depends on the developer as how he can make the website simple but rather a powerful one. The feedback needs to be gathered initially in order to determine if the application can generally be considered usable and easy for the developers. Portability is not an issue as it solely depends on the reliability of the server and the API calls made to the system. We don’t want to make the application heavy as it can indirectly cause a lag when the API calls are made to the system.

# Platform and Technologies

The platform used behind this application is fairly simple. The app uses Ubuntu 14.04 distribution to felicitate the server running Apache. Technology behind the API calls are the Facebook Graph API, Google’s API, and Github’s API on start. The application is striving for the main goal of providing API calls to over 10+ network API calls.

**Frontend**: HTML 5,CSS 3

**Backend**: JS, PHP 7.0.

**Server:** Apache

**Operating System**: Ubuntu 14.04 LTS Distribution.

# Estimation using COCOMO Model

The **Constructive Cost Model** (**COCOMO**) is an procedural [software cost estimation model](https://en.wikipedia.org/wiki/Estimation_in_software_engineering) developed by [Barry W. Boehm](https://en.wikipedia.org/wiki/Barry_Boehm). The model uses a basic [regression](https://en.wikipedia.org/wiki/Regression_analysis) formula with parameters that are derived from historical project data and current as well as future project characteristics.

**Basic COCOMO** compute software development effort (and cost) as a function of program size. Program size is expressed in estimated thousands of source lines of code ([SLOC](https://en.wikipedia.org/wiki/Source_lines_of_code), [KLOC](https://en.wikipedia.org/wiki/Source_lines_of_code#Related_terms)).

COCOMO applies to three classes of software projects:

* Organic projects - "small" teams with "good" experience working with "less than rigid" requirements
* Semi-detached projects - "medium" teams with mixed experience working with a mix of rigid and less than rigid requirements
* Embedded projects - developed within a set of "tight" constraints. It is also combination of organic and semi-detached projects.(hardware, software, operational, ...)

Our Application is based on organic project.

The basic COCOMO equations take the form

**Effort Applied (E)** = ab(KLOC)bb **[** [**man-months**](https://en.wikipedia.org/wiki/Man-month) **]**

**Development Time (D)** = cb(Effort Applied)db **[months]**

**People required (P)** = Effort Applied / Development Time **[count]**

where, **KLOC** is the estimated number of delivered lines (expressed in thousands ) of code for project. The coefficients *ab*, *bb*, *cb* and *db* are given in the following table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Software project** | ***ab*** | ***bb*** | ***cb*** | ***db*** |
| Organic | 2.4 | 1.05 | 2.5 | 0.38 |

**KLOC = (estimated) 1.4 (in thousand)**

**E = 2.4(1.4)^1.05 =**

**D = 2.5(E)^0.38 =**

**P = D/ E =**

Basic COCOMO is good for quick estimate of software costs. However it does not account for differences in hardware constraints, personnel quality and experience, use of modern tools and techniques, and so on.

1. **References**
2. Password authentication with insecure communication <http://dl.acm.org/citation.cfm?id=358797>
3. Two Factor Authentication Using Mobile Phones

<https://www.researchgate.net/profile/Fadi_Aloul/publication/221429455_Two_factor_authentication_using_mobile_phones/links/00463523225999d53c000000.pdf>

1. Plan 9 authentication in linux

<http://dl.acm.org/citation.cfm?id=1400101&CFID=644831282&CFTOKEN=81898493>