Software Requirements Specification

for

Platform.io

Version 1.0

Prepared by Jason Anrico

Southern Connecticut State University

September 5th, 2019

Table Of Contents

1. Introduction

This document is a Software Requirement Specification (SRS) for Platform.io. This is the initial draft for the SRS and it will be used for the extensions. This document is prepared by the following IEEE conventions for software requirement specification.

Platform.io is a website that provides entertainment for anyone. Regardless if your looking for entertainment, growing a brand, searching for jobs or employees. Platform.io allows you to view all of your media in one place along with others.

The purpose of this project is to provide a way for a user to track all their social media in one place i.e (Twitter, Youtube, Twitch, Facebook, Github, Instagram). But also along with other users on the site.

1.1 Purpose

The purpose of this document is to specify complete description of Platform.io. Through this document, It will clarify how development will proceed with this being a solo project. To be more specific, this document is going to describe functionality, external interfaces, performance, attributes and the design constraints of how the system will be developed. Therefore, the intended reader for this requirement specification are users / professors.

1.2 Scope

This project is intended for making use of API's and giving people a good product. There are so many API's to pick from that come along with social media that we all use every day. I want to explore further into them by trying to capture the idea of "What if all these sites came together as one".

By combining all these API's into one site, this would allow a user to essentially manage and look at all their social medias that are important to them in one place.

1.3 Overview

We are going to focus on describing this site in terms of a product perspective, product functions, user characteristics, assumptions and dependencies on the following section of this document. Next, we will discuss specific requirements of the system, external interface requirements, requirements of the system, performance requirements, and other requirements.

1.4 References

For this project I have used references to make this more of a smooth learning process. The Passport Documentation was super helpful for authorizing users and creating a session. W3Schools is always a saving grace, mainly used for HTML/CSS and structure of a Nodejs project.

2. Overall Description

This section gives background information about specific requirements of Platform.io. This wont go into detail about everything but will describe the factors that affect the final product.

2.1 Product Perspective

This software product is intended for anyone and can be used by anyone as long as they have access to the internet. This product will be deployed to the web. This website will have a pretty large scale to it, it is definitely not a small scale project.

To have access beyond the index page you must register by a local username or with Google+. When a new user is registered, all required data will be stored in the database with an encrypted password with the use of bcrypt. The only passwords that are not stored are Google+ accounts, this is handled by passport-google-oauth.

From a users point of view, once they have registered and logged in, they will be prompted to edit their profile and if they would like to link other social media accounts to their profile. The user can choose to skip the previous step if they would like to.

After setting up a profile, you will be brought to a discover page or in other words a real time feed. Here you can make a status post, find other users on the site, view their profile, and connect with them in messages or in comments.

2.2 Product Functions

Platform.io must have features which allow users to use the functionalities explain above. The required functionalities can be put into 5 categories; user management, socket programming, background page updates, api requirements. Overall descriptions of the requirements can be found below.

2.2.1 User Management Requirements

This category of requirements is related to user authentication. Each user will have their own credentials to log in and use the site. Users will perform all of their actions on the site under these credentials.

2.2.2 Socket Programming and File Server Requirements

This category is related to direct messaging and file server requirements. Each user will have the ability to send messages to another user, and upload files such as profile pictures and header photos. These functionalities require socket programming allowing for messaging and a file server to store images rather than a database.

2.2.3 Background Tasks Requirements

This website needs to be scalable. I would like to have a real time feed, and this will require only an update of your news feed not the entire page. Background processes must be ran in order to apply this to front end HTML. This is what the user will be looking at most of the time, they shouldn't have to refresh the whole page to receive some new information.

2.2.4 API’s Requirements

As an important part of this project, api’s will be used and called a lot. This is my main focus and priority to this project. API’s will only be active for a user if they choose to link them. Using passport, I can authenticate users and ask for their consent to link their accounts, no passwords will be stored. Each api account that is linked to a user will appear on their profile along with their activity on their accounts.

User data is handled by getting the information on the backend and sending it to a view, using scripts to bring that data to html tags. API responses are typically all .JSON objects

2.2.5 User Interface Requirements

This group of requirements is related to what the website will look like and how the user will interact with it. This website will be responsive so no matter what device you are on, it will fit to screen so nothing is out of place and functions properly. The interface will be minimal to ease confusion, but enough to know what everything does.

2.3 User Classes and Characteristics

Users of Platform.io can be anyone. The main target audiences, however can be content creators, streamers, growing brands, and pure entertainment conosours. I believe with this target audience that an average user will have a basic understanding of how to navigate web pages and other available social media pages. Also a clear documentation page about Platform.io will be available for unregistered and registered users.

2.4 Operating Environment

This website will be hosted with Google Cloud, an IP assigned a url. This website is intended to be compatible with all browsers. Platform.io is built with Nodejs, Express, ejs, and MySQL.

2.5 Constraints

Since i am the only developer of this project I had to make priorities on what comes first.

I am aware that this site must be secure, with that I myself will be the only one who can be an admin of the site. The only passwords stored are local accounts with usernames in which the password is encrypted. No other passwords are stored, verification is handled by passport.

The main feature of this site is to track your social media, and I must assure that it does exactly that while also providing more at the same time. Focus on the important items first, worry about bigger parts later.

2.6 User Documentation

In production of this website I did use manuals and online help. The manuals and online help that I have used thus far are the Passport, ejs, express, html and css documentations. W3schools has also helped me a lot with my html and css.

2.7 Assumptions and Dependencies

Platform.io does in fact use open source dependencies. In my package.json file i currently have the following dependencies installed using npm.

Dependency	Version
Ejs	2.6.2
Express	4.17.1
Express-session	1.16.2
Morgan	1.9.1
Mysql	2.17.1
Nodemon	1.19.1
Passport	1.0.0
Passport-local	1.0.0
Passport-facebook	3.0.0
Passport-twitter	

Passport-twitch	
Passport-google-oauth	2.0.0
Bcrypt	3.0.6
uuid	3.3.2
Body-parser	1.19.0
Cookie-parser	1.4.4
Connect-flash	0.1.1
Request	2.88.0
request-promise	4.2.4

3. Specific Requirements

With this section and later, we will describe the requirements of the software in detail. We will be categorizing these requirements in 2, which are User interfaces, Hardware & Software Interfaces, and Communications Interfaces.

3.1 User Interfaces

I have quite a few html pages along with a bunch of backend code started already. Instead of describing all the pages, I can just show what I have so far.

3.2 Hardware & Software Interfaces

This website will not be very hardware dependent. My goal is to make this website as flexible as possible. Intended devices are (Desktops (PC), Laptops, Phones and Tablets).

No matter what operating system the device is running, or what browser they are on they should have no issues on the site. The software used to create this site are as follows:

- Ubuntu 18.04 Kernel Version:
- MySQL Server 5
- Npm
- Nodejs
- Text Editor - Atom w/ Terminal Package

3.3 Communications Interfaces

This website will make use of File Transfer Protocol (FTP) this will be used for uploading files and communicating with users. This will allow users to share more than just plain text.

4. System Features

In this section, we will examine the features of Platform.io in detail according to their functionality. For each feature, we will give an introduction, purpose, diagram/image and an example response.

4.1 Register

4.1.1 Description

Goal in Context: Purpose of this feature is to register user to website
Trigger: User wants to register to website

Case 1: Local Signup

- User opens register page
- User specifies information
- System validates information
- User is registered and logged into system

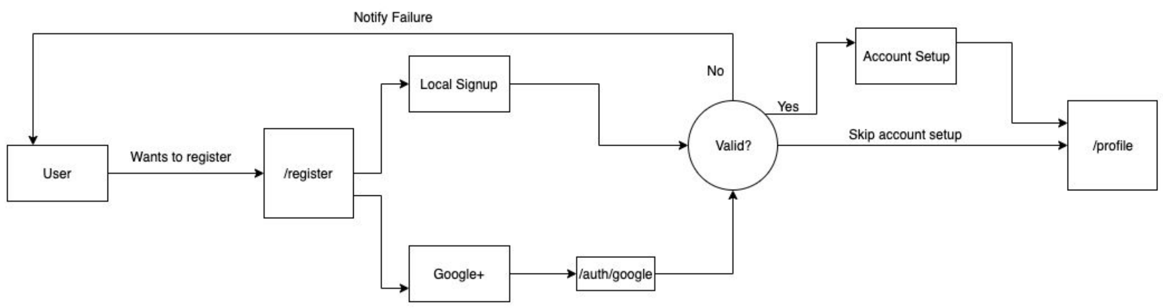
Case 2: Google+ Register

- User Clicks Register with Google+
- User enters google+ email and password
- Passport validates information
- User is registered and logged into system

Case 3: Incorrect Information

- User inputs incorrect information
- Prompted to retry

4.1.2 Diagram



4.1.3 Functional Requirements

1. User Management Requirement
2. API's Requirement
3. User Interface Requirement

4.2 Login

4.2.1 Description

Goal in Context: Purpose of this feature is to log a user in. Google+ logins assume that a user is already logged in to their google+ account
Trigger: The user wants to login

Case 1: Local Login

- User opens login page
- User specifies information
- Compare Sync Password
- User is logged in

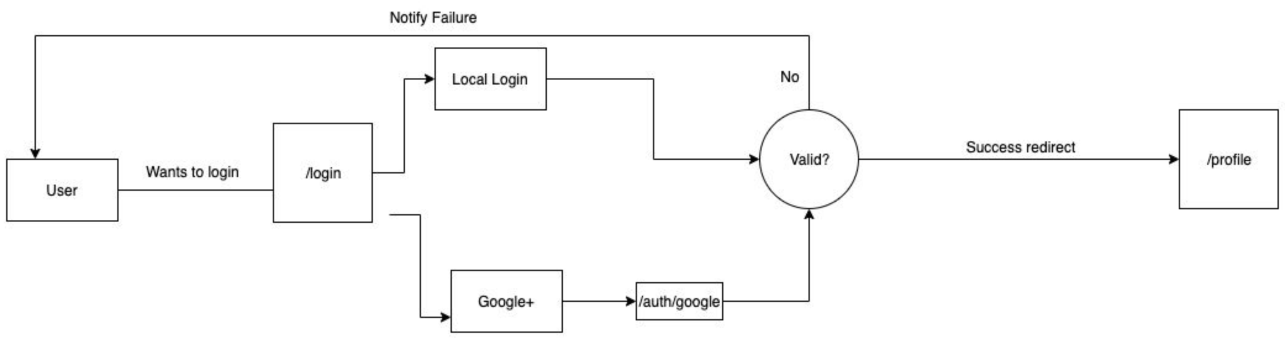
Case 2: Google Login

- User open login page
- User clicks Google+ icon
- User is logged in

Case 3: Incorrect Credentials

- User inputs incorrect information
- Prompted to retry

4.2.2 Diagram



4.2.3 Functional Requirements
1. User Management Requirements
2. API’s Requirement
3. User Interface Requirement

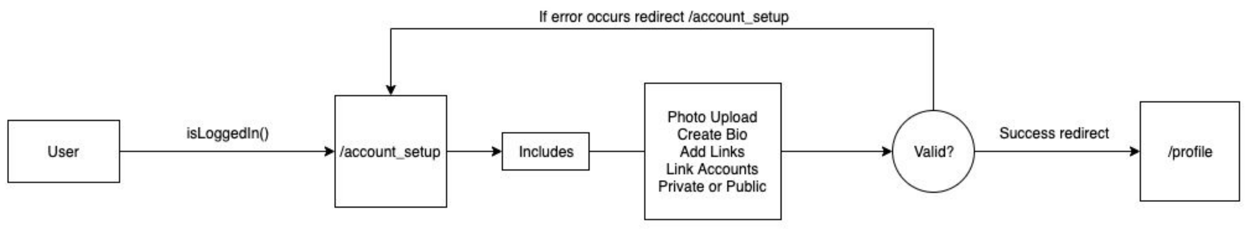
4.3 Account Setup
A user can set up their profile even further once they have made an account.

4.3.1 Description

Goal In Context: Purpose of this feature is to give a users account more personalization.
Trigger: A user wants to set up their account

Case 1: Photo Upload - User selects photo from current device - Upload to server - If error occurs notify user **---** **Case 2: Create Bio** - User enters a few brief statements about themselves - Inserted to database - If error occurs notify user **--** **Case 3: Add Links** - User may add links (i.e portfolio or website url) - Inserted to database - If error occurs notify user	**Case 4: Linking Media Accounts** - User wants to link a social media account - Click icon media choice - Next url /auth/<medianame> (i.e /auth/google /auth/twitter) - Enters valid user info - Only store email to database - If successful /auth/<medianame>/callback (i.e /auth/twitter/callback) - If error occurs notify user to retry. --- **Case 5: Private Account** - User wants to make account private (public by default) - Choose private (boolean) - User is now private - If error occurs notify user

4.3.2 Diagram



4.3.3 Functional Requirements

1.User Interface Requirement
2. User Management Requirement
3. API’s Requirement
4. File Server Requirements

4.4 Linking Media Account

A user of the site may want to link a social media account. These social media accounts consist of Twitter, Twitch, Github, Youtube, Facebook, and Instagram. This will allow a user to display their statistics of all their accounts on their profile.

4.4.1 Description

Goal in Context: Purpose of this feature is to allow a user to give their account more personalization and use the website to its fullest potential.
Trigger: A user wants to link a social media account(s).

Case 1: Twitter

- User wants to link Twitter account
- Brought to /auth/twitter
- Enter credentials
- Passport-twitter checks for authorization
- If successful /auth/twitter/callback
- Redirect to /account_setup
- If error display error

Case 2: Twitch

- User wants to link Twitch account
- Brought to /auth/twitch
- Enter credentials
- Passport-twitch checks for authorization
- If successful /auth/twitch/callback
- Redirect to /account_setup
- If error display error

Case 3: Youtube

- User wants to link Youtube account
- Brought to /auth/youtube
- Enter credentials
- Passport-youtube checks for authorization
- If successful /auth/youtube/callback
- Redirect to /account_setup
- If error display error

Case 4: Github

- User wants to link Github account
- Brought to /auth/github
- Enter credentials
- Passport-github checks for authorization
- If successful /auth/github/callback
- Redirect to /account_setup
- If error display error

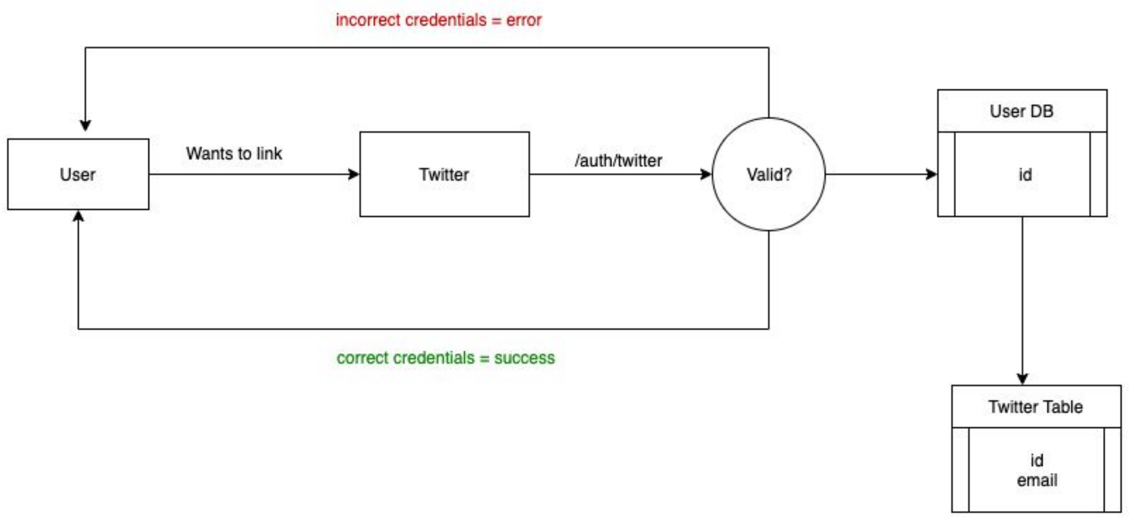
Case 5: Facebook

- User wants to link Facebook account
- Brought to /auth/facebook
- Enter credentials
- Passport-facebook checks for authorization
- If successful /auth/facebook/callback
- Redirect to /account_setup
- If error display error

Case 6: Instagram

- User wants to link Instagram account
- Brought to /auth/instagram
- Enter credentials
- Passport checks for authorization
- If successful /auth/instagram/callback
- Redirect to /account_setup
- If error display error

4.4.2 Diagram



4.4.3 Functional Requirements

1. API's Requirement
2. User Interface Requirement
3. User Management Requirement

4.5 Private or Public

On this site a user can decide if they would like to have a public or private account. A user is Public by default. Public accounts can be seen and followed by anyone. Private Accounts can only be seen by followers and follow requests must be accepted.

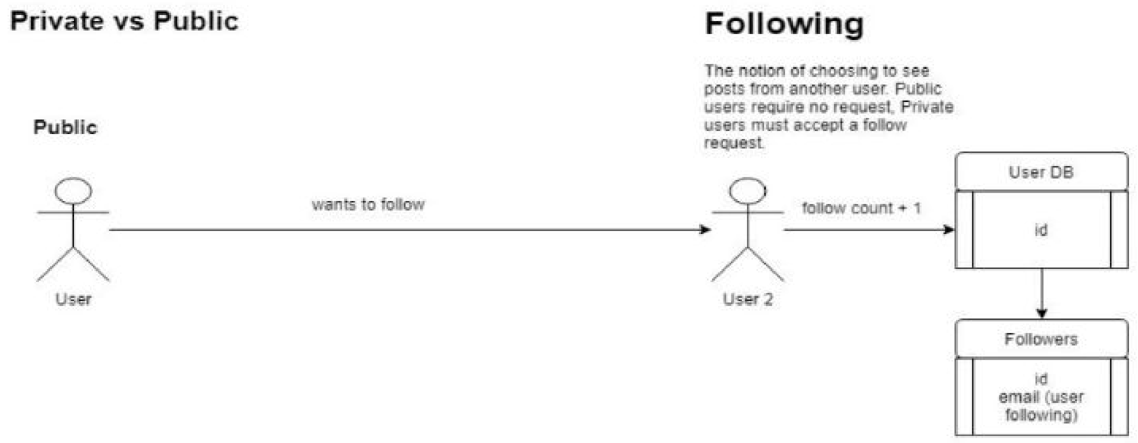
4.5.1 Description

Goal in Context: Purpose is to allow users to decide who can follow them and view their profile.
Trigger: A user wants to change their account privacy.

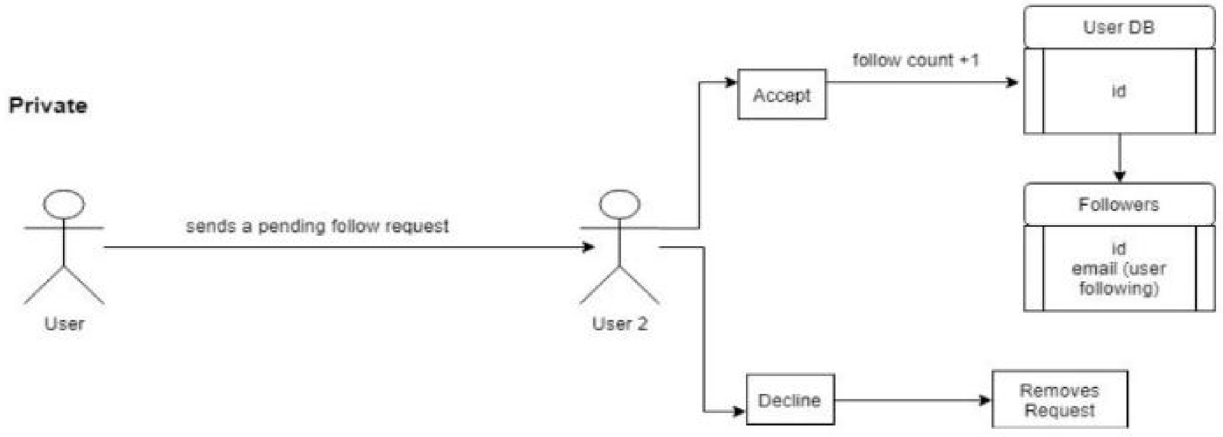
Case 1: Go Private	**Case 2: From Private to Public**
- In Account Setup - User chooses private - Private boolean value changed to 1 - User is now Private	- In Account Setup - User chooses go public - Private boolean value changed to 0 - User is not Public

4.5.2 Diagram

Public


Private vs Public
Following
The notion of choosing to see posts from another user. Public users require no request, Private users must accept a follow request.
Public
wants to follow
User
User 2
follow count + 1
User DB
id
Followers
id
email (user following)

Private



4.5.3 Functional Requirements

1. User Management Requirement
2. User Interface Requirement

4.6 Realtime Feed

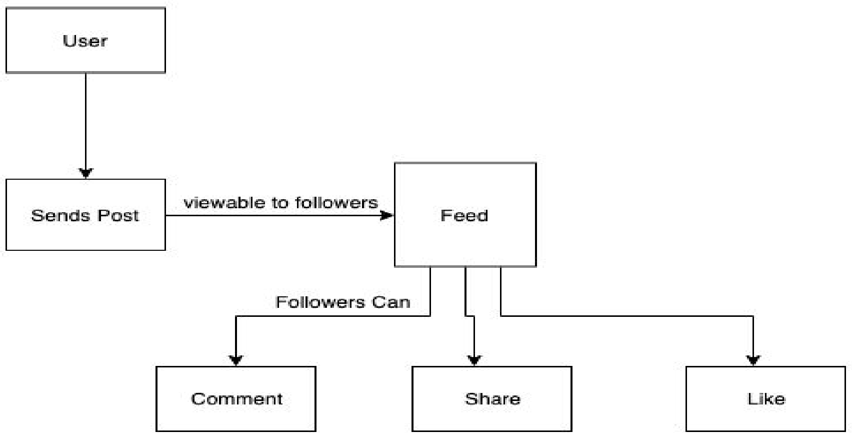
Platform.io consists of a real time activity feed. This allows users to see posts from other users that they follow. This is real time, so once a post is made just the feed refreshes not the whole page.

4.6.1 Description

Goal in Context: Purpose of the realtime feed is to bring more to the site and allow users to interact with each other.
Trigger: A user wants to see what other users are posting.

Case 1: Posting - User fills post form - Post is added to the feed - If public, all site users can view - If private, only followers can view -- **Case 2: Liking** - Once a post is made, other users are allowed to like the post - Click like button - Like as added to post - Author of post is notified of like	**Case 3: Commenting** - Once a post is made, other users are allowed to comment on post - User fills comment form - Comment is added to post - Author of post is notified of comment -- **Case 4: Replies** - When a post has comments, other users are allowed to reply to comments - User fills in reply field - Reply is added to comment - Author of post and comment are notified.

4.6.2 Diagram



4.6.3 Functional Requirements
1.User Management Requirement
2. User Interface Requirement
3. Socket Programming Requirements

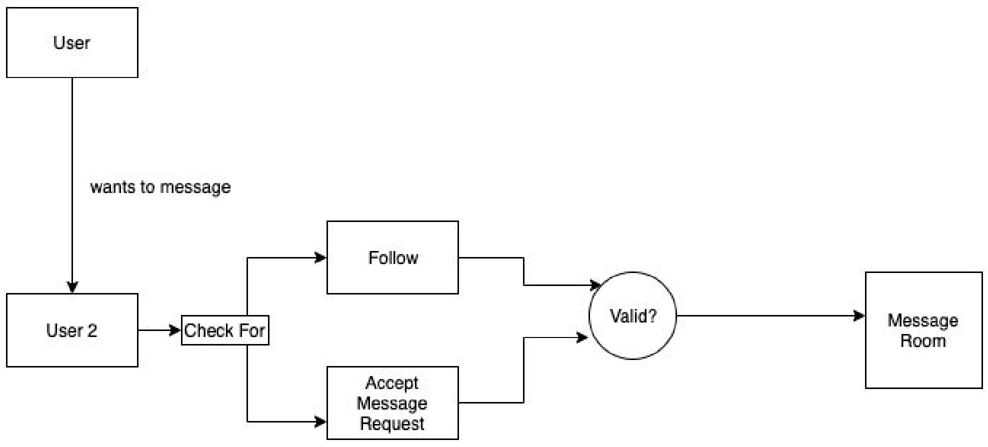
4.7 Messaging
Platform.io allows users to message each other. This is essential for users to be able to communicate and interact further.

4.7.1 Description
Goal in Context: Purpose of messaging is to allow users to communicate more.
Trigger: A user wants to send a message to another user.

Case 1: Message Request
- A user must send a message request to be accepted
- Once accepted, message room is open.

Case 2: Rejected Message Request
- A user sends a message request to a user
- User does not accept
- No message room is open

4.7.2 Diagram



4.7.3 Functional Requirements
1.User Management Requirement
2. Socket Programming Requirement
3. User Interface Requirement

4.8 Notifications

Platform.io wants the user to constantly be notified about the activity of their account.

4.8.1 Description

Goal in Context: Purpose of notifications is to allow a user to stay updated about their account.
Trigger: A user receives an update on account status

Case 1: Message Notification

- User receives message
- Notified that a message has been sent to them

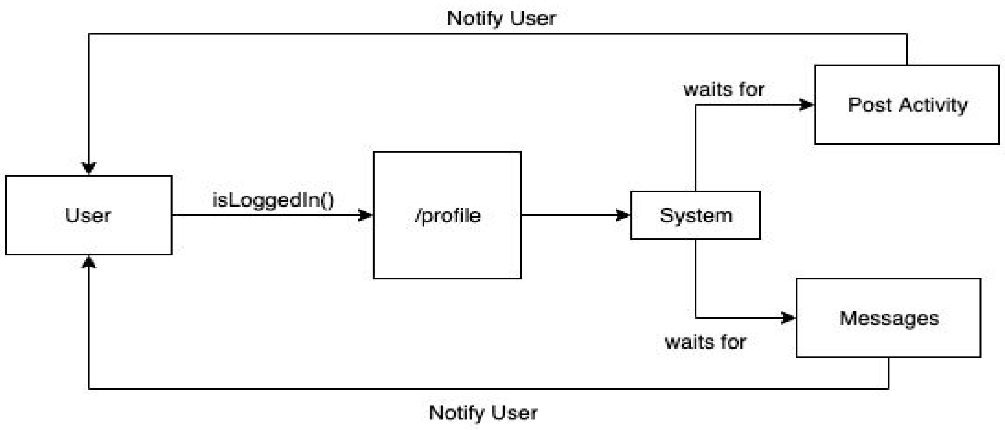
Case 2: Comment and Like Notification

- User receives comments or likes on post
- Notified of activity on post

Case 3: Flash Messages

- Flash messages are small messages that appear on screen.
- New posts on feed
- Error messages
- Logging in / Logging out

4.8.2 Diagram



4.8.3 Functional Requirements

1. User Management Requirement
2. User Interface Requirement
3. Socket Programming Requirement

5. Other Nonfunctional Requirements

In this section, last group of requirements which is non-functional requirements will be explained in detail. Non-functional Requirements include performance requirements and security requirements

5.1 Performance Requirements
Since Platform.io is going to be web based, it will require a moderately powerful CPU and connections for messaging should allow to handle multiple users at a time.

This web application will be developed so it can work on the most common devices today. (i.e Phones, Tablets, Laptops. Desktop (PC's)).

My expected number of simultaneous users should be around 100. System should be able to deal with 100 users at the same time. Also database of the system should be able to hold a sizable amount of data and perform correct queries accordingly.

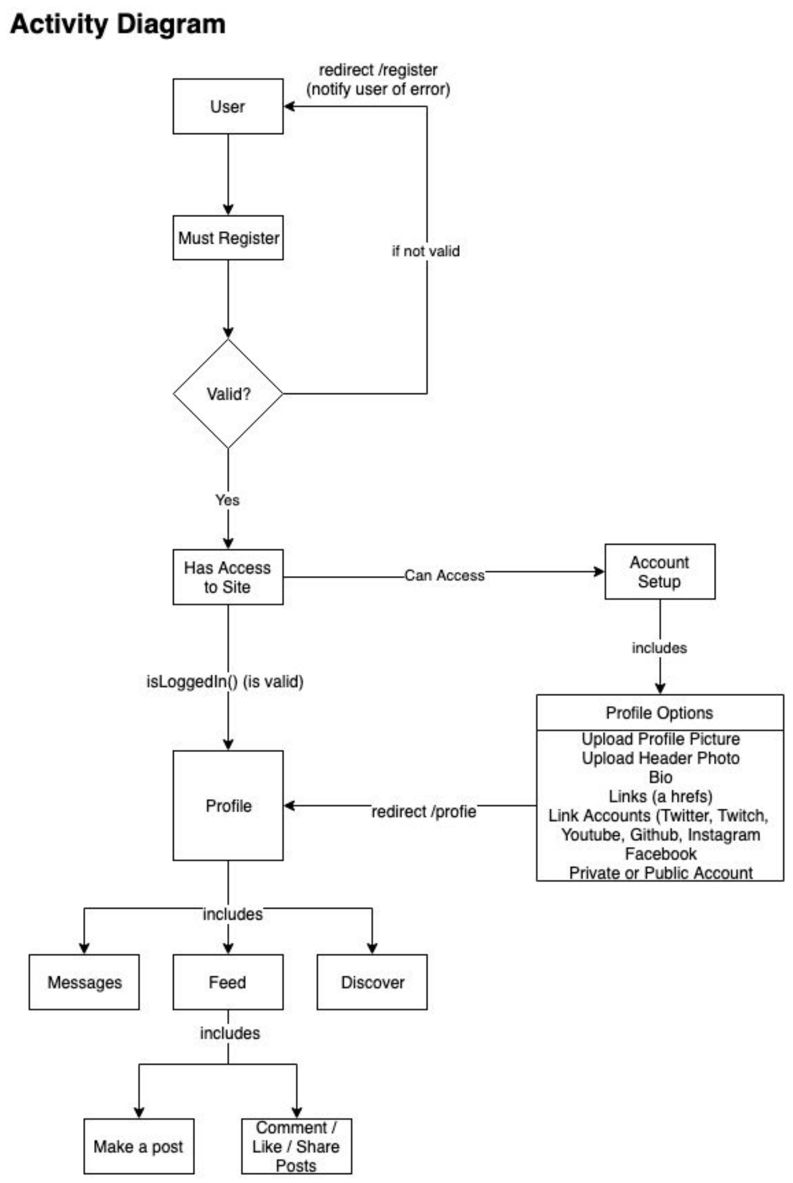
5.3 Security Requirements
Since Platform.io will be a hosted website and could face potential attacks to get user information. Product should be able to protect the privacy of user data. Accounts will only be able to be accessed with correct credentials.

Platform.io is going to store the least amount of information as possible for users. All passwords are encrypted.

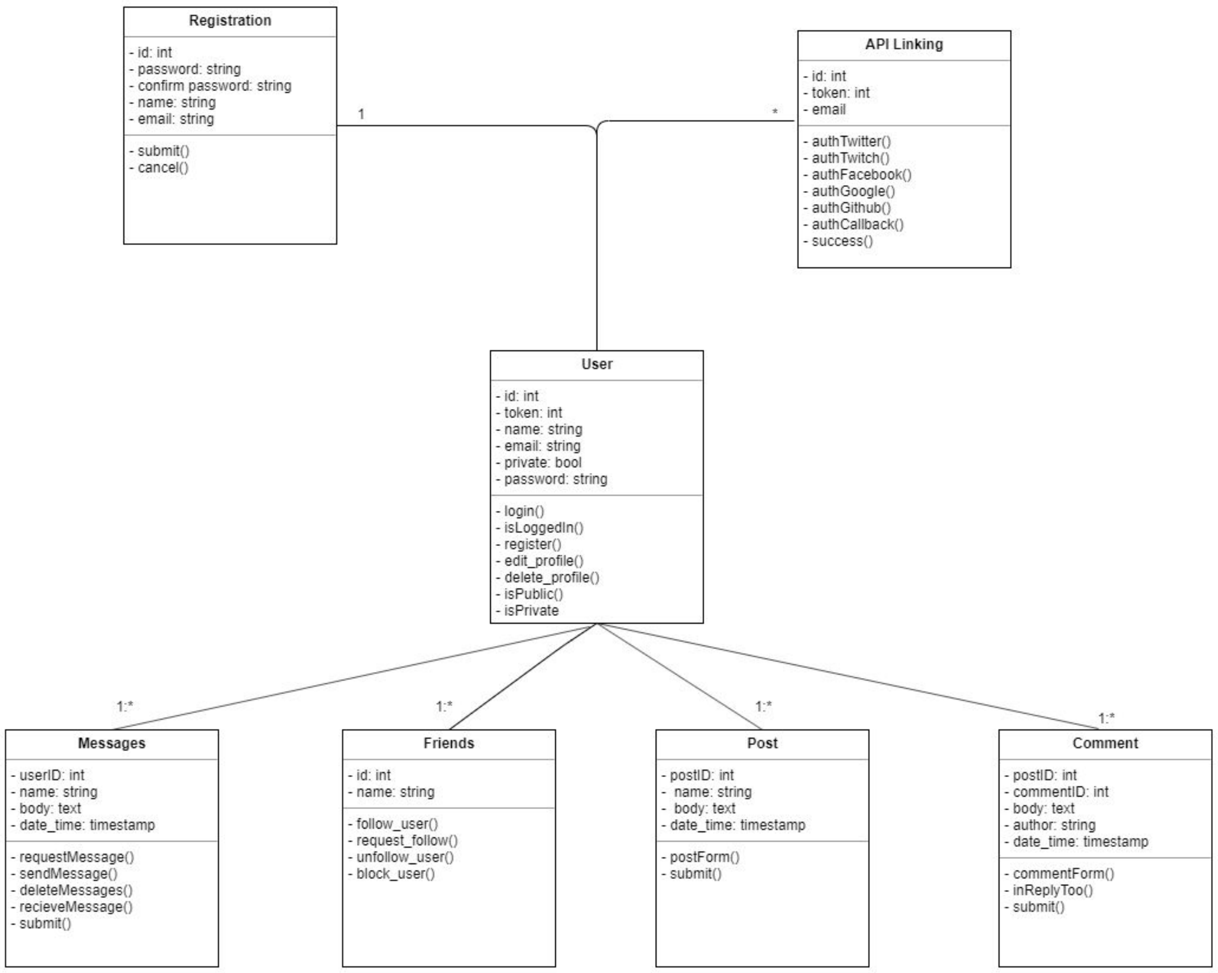
6. Other Requirements

Appendix B: Analysis Models

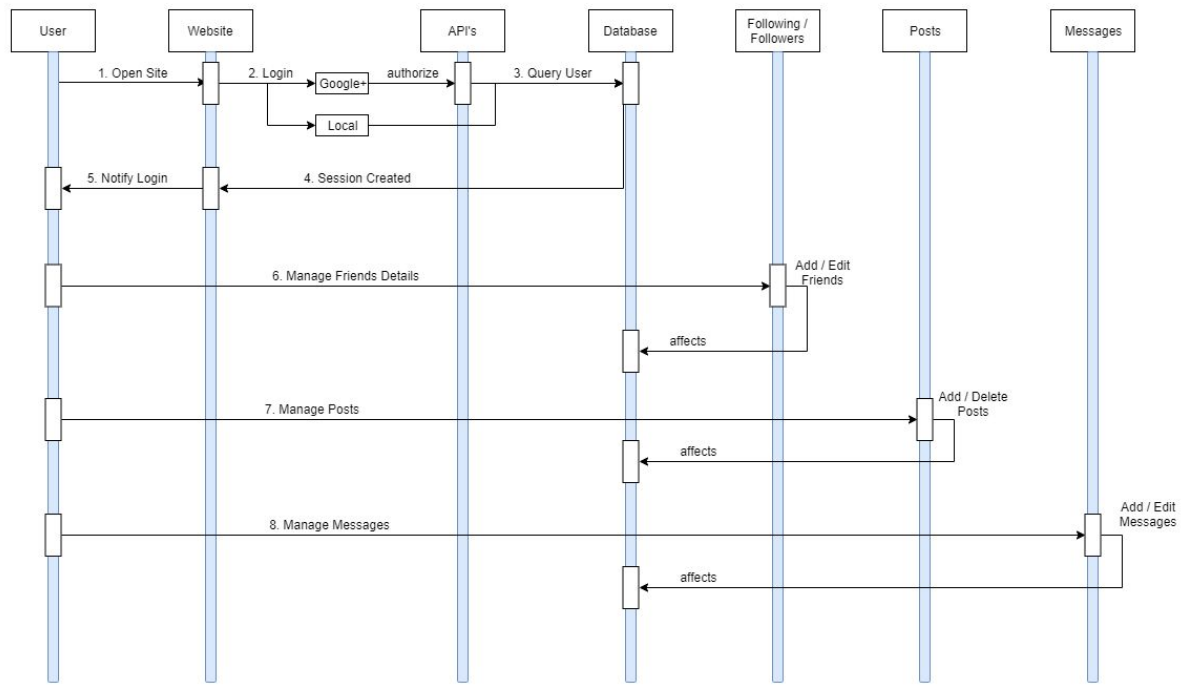
Activity Diagram



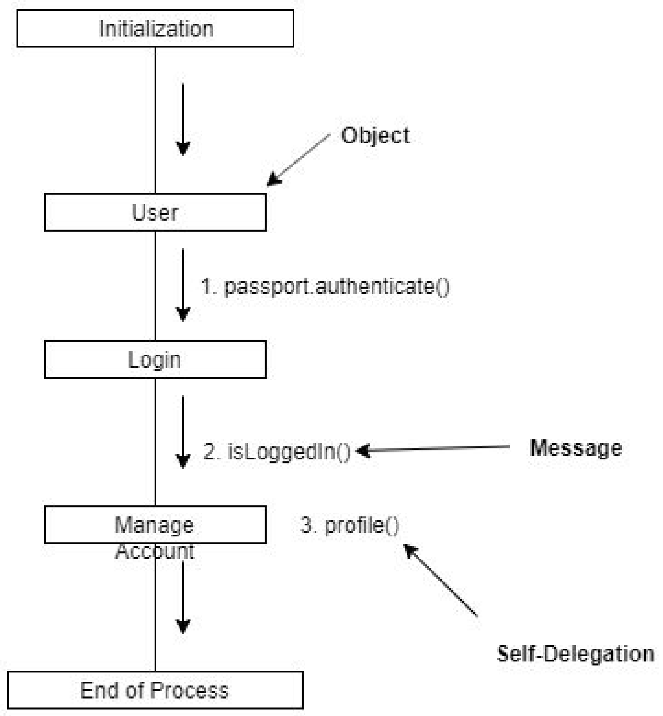
Class Diagram



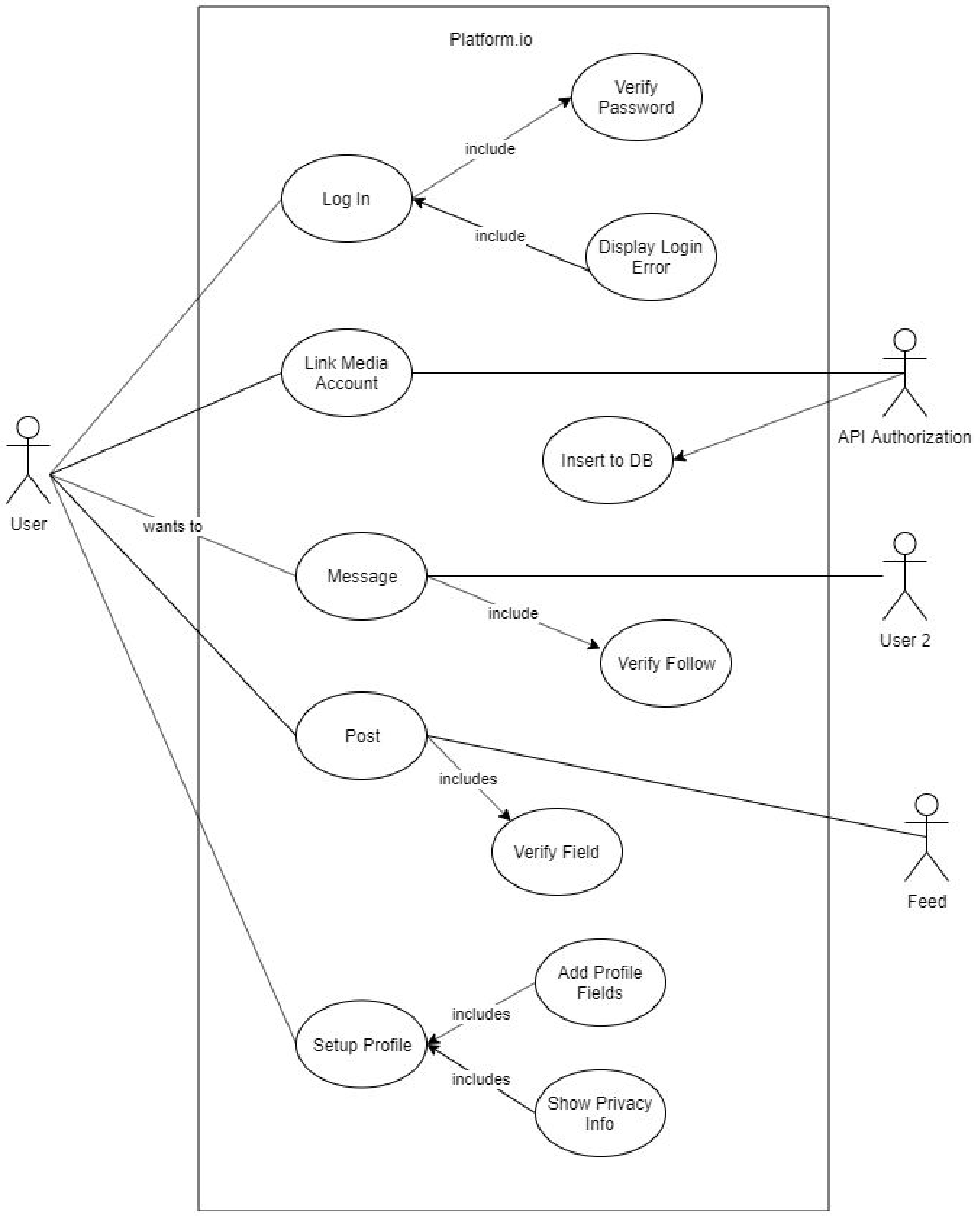
Sequence Diagram



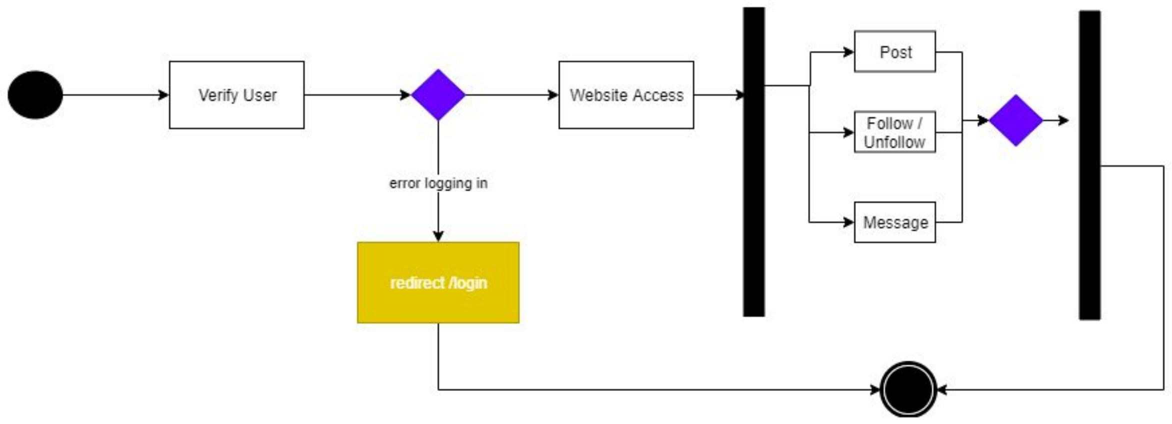
Collaboration Diagram



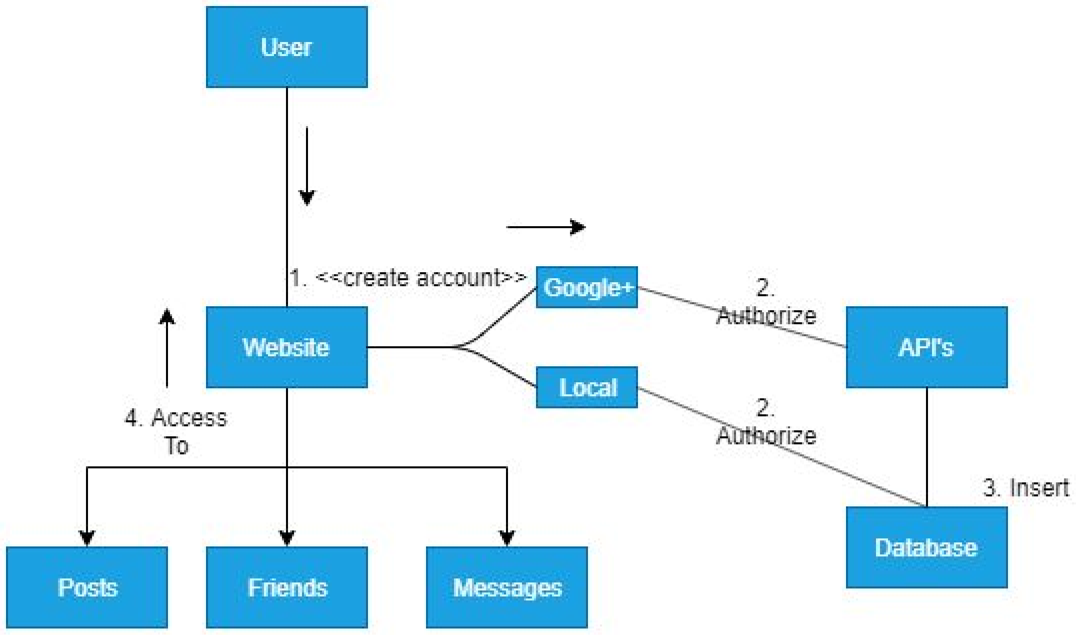
Use Case Diagram



State Diagram



Communication Diagram


User
1. <<create account>>
Google+
Local
Website
2. Authorize
2. Authorize
API's
3. Insert
Database
4. Access To
Posts
Friends
Messages