Account Management

by Jinan Kordab

Google has single Sin In implemented for many of its services. By logging in to Google account for example, one has access to Gmail, YouTube, and other services in a single browser session, without the need to sign in again.

Also, it is in users' interest, to stay with one account, meaning if he or she registers with one unique handle, it is in his or her only interest to stay with that handle or account.

Each application, software solution, intranet solution, or internet solution, should have settings and options, or variables, including users' settings, his or her history, things he or she can change, etc . . . And while the user's account is active, all these things mentioned above progress with him. An example would be things he bought for the past seven months, his wish list for each month, his friends history, his friends chat history, etc. . .

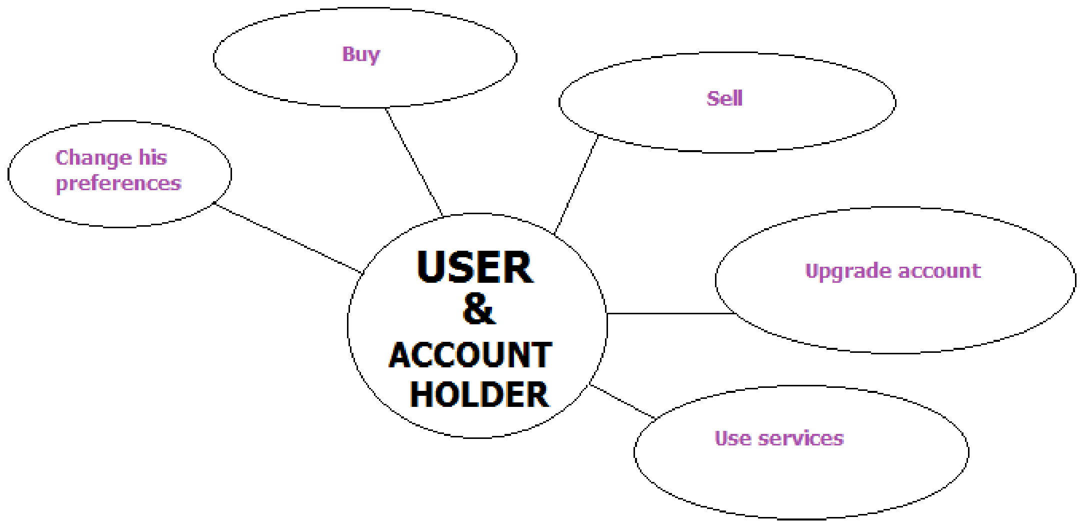
Also, and this is directly related to Account Management in a Software Solution, some social network sites nowadays, keep user's score in social media, and measure how active he or she has been. And as time moves forward, joined with user's activities and actions on different social networks, his score becomes higher.

Now imagine in the scenario mentioned above, that a user does not have a single account on such a network. This means that he will have many scores, different ones. And social network, or software solution, will not identify him or her as same single user, or holder of his account.

This is only to say that staying with single account, and opening a single account on a software solution, is solely in the interest of a user.

While building a Software Solution, this single user approach should be taken into consideration from the beginning, in the architecture itself.

Account management, is far beyond granting Authorization and then Authentication for a user. In addition to this, Account Management tells a users story over a period of time, what he can do and allowed to do. As an example, please see the image below:



One might also ask oneself: “Why would a user open many accounts, with different credentials on same software solution, be it a website or downloadable desktop software?”. There are many justifiable reasons for that, but from a software solution, it is highly recommended to create single account for oneself, and also, provide an option for a user to link all other accounts that he has created. And I mean accounts and not pages within a single account! This will enhance user’s experience and add more value to the software itself.

A software solution is trusted more, if it does not store users private information, such as address, date of birth, martial status, email, mailing address, home address, phone, and other personal id’s data on its own databases and servers. But then we ask ourselves: “How do we register a user, and let him log in with user name and password ?”. The answer is simple. There is no need for any password anymore ! For user accounts registration, we could use OAuth protocol from reputable companies like Google, Microsoft, and Yahoo.

There are two solutions to this. First is, implement your own OAuth protocol, with your custom code, in whatever language you want, build your own authorization server, your own authentication server, encrypt and hash code, and do the same for tokens. Also, if you do not want to write your own OAuth protocol, suited only for your needs, you can use many public OAuth providers, written in different languages libraries, add those libraries to your solution, and use them. But this still means in one way or another creating your own authorization and authentication server, if you want, since this is what OAuth stands for.This solution does not require an account holder or user to enter a password as well.

The second solution is to use Windows Azure, where everything is already done for you. Windows Azure has OAuth for Google, OAuth for Microsoft, and OAuth for Yahoo, and many others, since every one of those big companies write their own Authorization and Authentication servers, and libraries as well. Microsoft Azure has them all! What this means is that you implement your solution, and leave the sign in , and log in to OAuth on Windows Azure, which will handle everything for you. You , will not have to store any user’s password whatsoever. The only thing that you store, is a hashed string , or handle, that will identify a user!

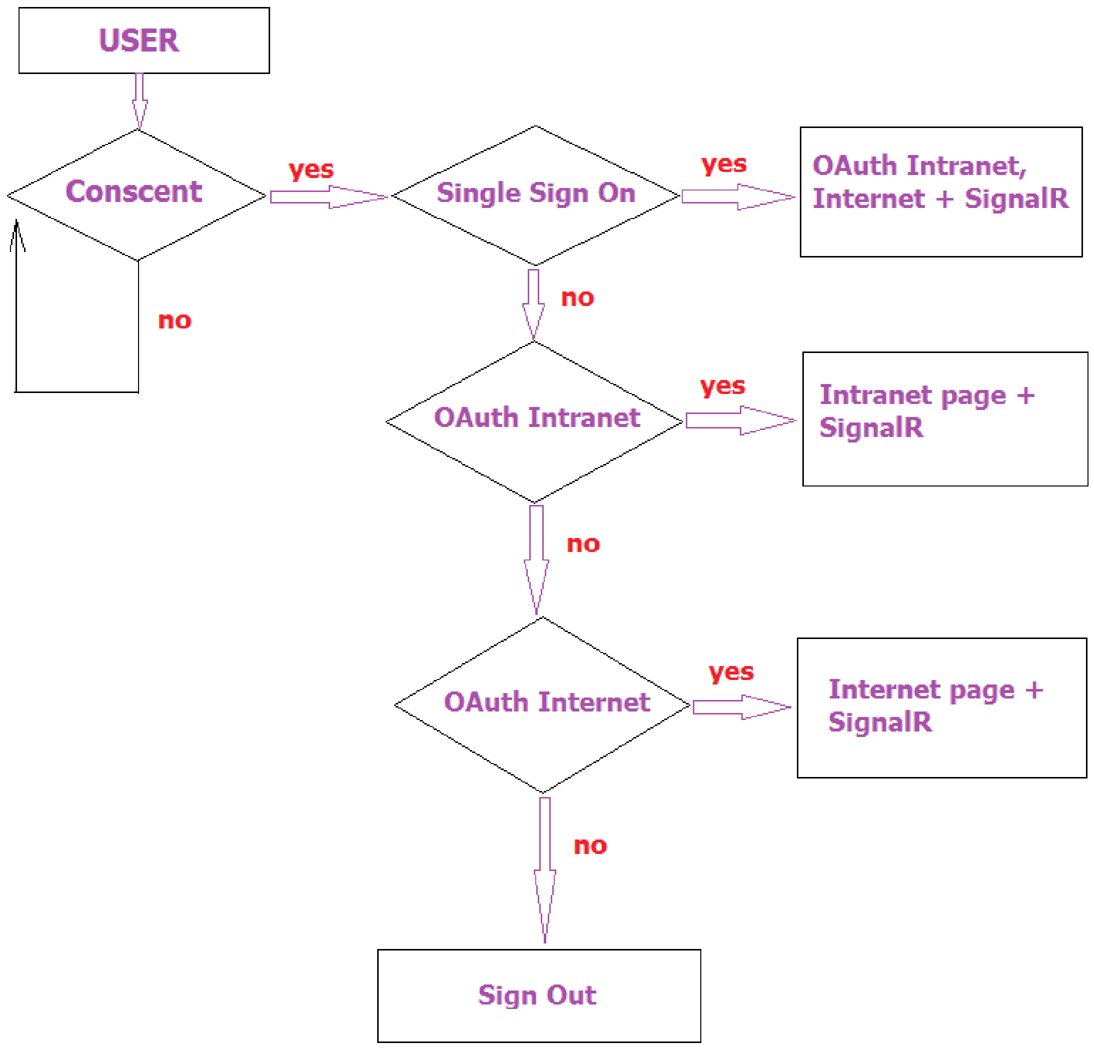
In both of these cases, you, of-course need to ask users’s CONSCENT to log in with Google Credentials, Microsoft Credentials, Yahoo credentials, and many other providers.

From an application standpoint, it does not see any password, and it does not register a user at all!

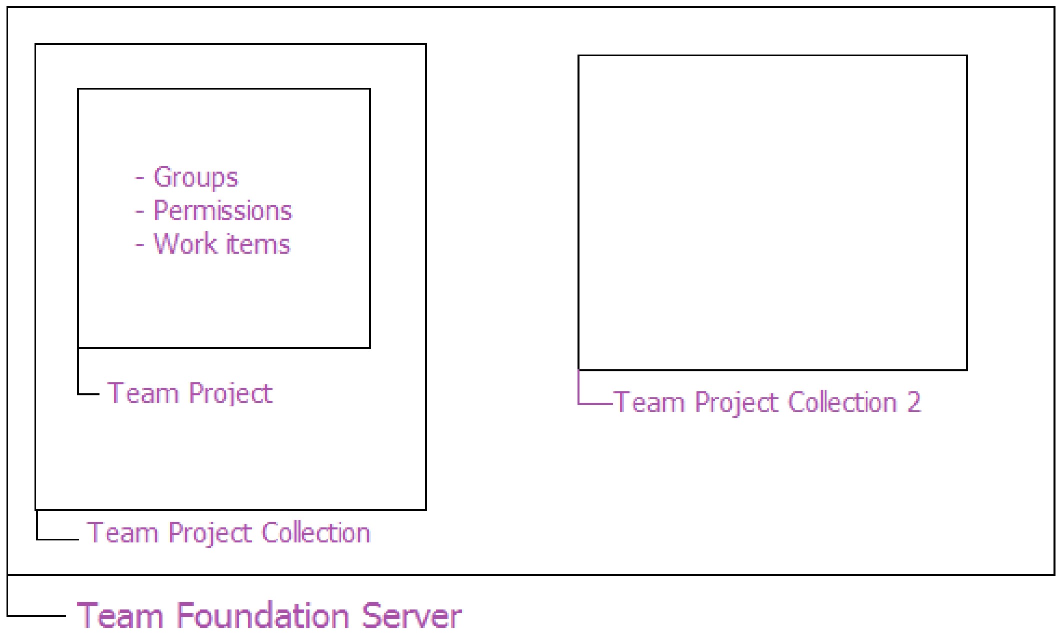
Account management, from software’s standpoint, is not customer support as well. And it is not a service oriented department. User management and account management must provide best possible value for users’ investment in one or another software solution.

Account manager, should fully understand software’s architecture that it is built upon. He or she should also be aware of different design patterns that come together to bring that architecture to life, because a single software solution might include many design patterns working together! He or she should as well be fully able to read someone else's code, or code written by someone else, because not always we end up writing our own software solution.

With OAuth, weather implementing it by oneself, using third party OAuth libraries, or using a third party hosting service, as mentioned above, it is very easy to set up Single Sign on. Single Sign On is not SPA or Single Page App. Single Page App is an app that has all its functionalities on a single page, where as Single Sign On, is a process where a user signs in, and he automatically is signed in to different domains, or accounts within the same software farm. As an example, Google. The diagram below shows my example, of a very secure and safe application design, in general, using OAuth, and applying SignalR technology if the need arises:



On the other hand, if you are planning to write or design a software solution, including good account and user management architecture, you might want to consider TFS, or Team Foundation Server for better and faster code writing and team collaboration, even if your team is only of two programmers or software developers. In its core, TFS is a version control system, that stores, and keeps track of your files written by your team members. Examples of version control system are: Source Safe, Subversion, and TFS. The picture below shows basic TFS architecture:


- Groups
- Permissions
- Work items
Team Project
Team Project Collection
Team Project Collection 2
Team Foundation Server