Uknow InfoHub

Production Requirement Documentation

BlXLRSMB Team

Contents

1 Introduction

1.1 Background

The use of social networks spread rapidly. It became a common sense that information is collected through online information portal. This product mainly aims to help users collect useful information as well as manage their personal information in convenience.

1.1.1 Complexity of Information Resources

Nowadays there are various sources of information. Light-weight sources, such as RenRen, Sina Micro Blog (Weibo) produce news all the time. An ordinary user would use many of them, then he needs to open several different web pages, or different phone applications, in order to get all the news feeds.

Moreover, those heavy-weight sources like NetEase, Yahoo!, or individual blogs may produce information from time to time. Readers may not be willing to consecutively refresh them to get the latest article, which will be both time-consuming and bandwidth-consuming.

According to the pre-survey we made, most social network users are facing this situation that the complexity of information resources is causing them problems.

Quantity

The large quantity of web portals, social networks makes users tired of changing and choosing between web pages or applications. Users tend to only focus on specific types of information, but will suffer from the large quantity of the information presented to them.

Redundancy

It is a waste of time for users to acquire similar information through different portals. Users is long for a tool to automatically, intelligently filter the repeated information from different sources.

Therefore, an information collector with corresponding filter is in great demand, in order to simplify the flow of getting the latest information. Users can browse their favourite for one time in just one place.

It is an obvious conclusion that users are in badly needed to one browsing-friendly page that simply contains all and exactly all the information they need.

1.1.2 Analysis of Existing Information Portals

In this section, the possibility of combining information resources will be illustrated. Let's first take Sina Weibo[1], Renren[2] and Zhihu[3], which are three major information sources of different types, as examples.

1. User Interface

   It is not hard to find out that the user interfaces designed by different websites do not distinguish each other too much. There are considerably parts that follow certain similar format. The interfaces are mainly divided into four categories:

   (a) main interfaces
   (b) personal interface
   (c) managing and editing interface
   (d) other individual functions

2. Main Interface

[1] http://weibo.com
[2] http://renren.com
[3] http://zhihu.com

The main interface is the page that provides the most significant and useful information. Users mainly receive information through this page, therefore it is the information sources we should be working on.

As we can see in the following figures, the main interfaces' features are listed as followed:

Figure 1: Main Interface of Sina Weibo

Figure 2: Main Interface of Renren

Figure 3: Main Interface of Zhihu

(a) Information items are listed and presented by vertical blocks. (Fig. 1, Fig. 2, Fig. 3)

(b) The item-block consists of the author, the subtitle, the content and the picture part. (Fig. 1, Fig. 2)

Since the speciality of the format that the interfaces are following can be easily caught, it provides the great possibility to combine those pages together.

1.2 Purposes

Uknow InfoHub is designed according to the needs of users. It will provide users with a entirely new experience in gathering information – by using only one information collector. Technically, a collector which can be customized and widened is required, for it's inevitable that different users ask for different classification information. Even more, if the collector contains a recommendation system, which can be capable to suggest subscribers articles they may interested in, users will be more sticky to this collector in all probability.

1.3 Future Works

Uknow InfoHub shall provide extensible interfaces. The Open API makes it possible to expand different functions based on this collector.

For instance, a Text-to-Speech system can be used upon the collector, so that users will be able to hear the information instead of browsing. Meanwhile, based on the users behavior, a recommendation system could be most valuable to users that recommend stuff which is meaningful to individual users. An Ad-targeting system can also be built directly on this.

This collector should provide the easy functionality to be used on various operating systems as well as various devices, such as Android, iOS, Linux, Windows.

2 Functional Requirement

2.1 User Management

An user's permitted behaviors on his/her account are included in but not limited to the following:

- Register an account
- Login
- Edit his/her personal profile
- Change his/her password
- Delete his/her account, along with all the relevant data

Moreover, since Uknow is supposed to connect to multiple diversed service providers, users have to provide authorize us the use of their accounts on other

web services. OAuth[4] is an open protocol to allow secure authorization. By using OAuth, users can safely authorize Uknow to use their accounts. The principles and proecss of OAuth authorization is illustrated in Fig. 4.

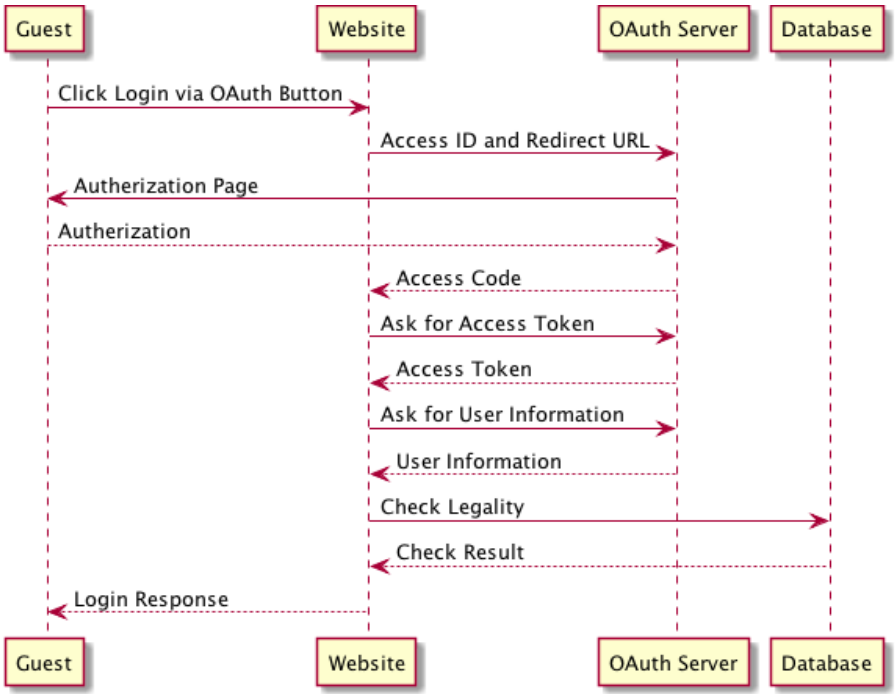


Figure 4: OAuth process

2.2 Data Management

2.2.1 Items

An item is piece of information retrieved from various sources, and shown in lists for further reading. It consists of following parts:

- Title
- Source
- Content
- Labels
- Comments

Item is the core entity in Uknow system. A user can manipulate items in following ways:

Removal

[4] http://oauth.net/

When a user find a specific item unpleasant, unattractive or redundant, he/she may remove this item using the button provided, as shown in Fig. 5. Removal operation can be reversed within a short period of time to prevent from mis-clicking the button.

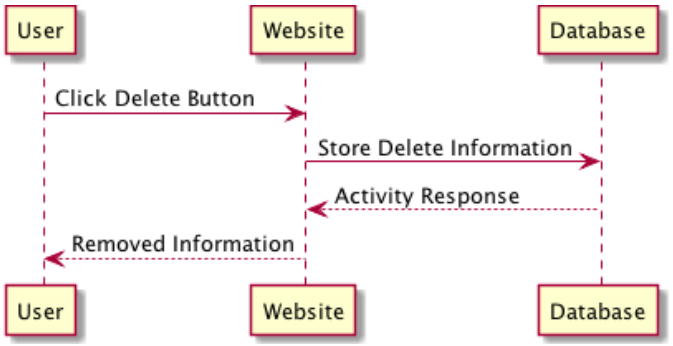


Figure 5: Removal process

Archive

User can store specific items for further reading or recapping in the future. Archived items still show up in item flow, and will be labeled as 'archived'.

Share

If an item fascinates a user, he/she may share it among popular SNS. When first sharing an item, user will authenticate its account on desired SNS website, and after confirmation, a sharing information will be posted on the chosen SNS.

Like

Users can show theirs preferences on certain items, and others will know the statistics, and this could be used as an score to evaluate an item. For a better reading experience, high score items are more likely to rank higher in the item flow than low score items.

Comment

Users can post comments under an item. Posts will be stored in this system, and can optionally be posted synchronously on item source, if the source website supports such functionality.

Comments is organized hierarchically. If a user intends to reply other's comment, he/she can click the reply button, and the reply form should show right below the comment.

When complete typing, click the reply button and the comments shall be posted, and the form will consequently disappear.

Furthermore, system can learn from user behaviors on items and to recommend items related to users' interest.

2.2.2 Labels

A label is an attribute describing items. An item can have multiple labels. A label associated with an item may be automatically assigned by system, or tagged by users. A label can be either a system-wide label, which is visible to all user, or a user specified label which indicates user preferences on an item.

Label may come from:

System Pre-tagging An item may arrive to a user with pre-tagged labels. These pre-tagged labels are vital to subsequent data processing, such as tag-filtering plugins in tabs.

User-tagging Same item could have distinct meaning to different user. User can tag item with labels cater to their taste, as well as remove labels that could lead to misunderstanding to itself, as shown in Fig. 6.

Figure 6: Tagging process

As described aforementioned, functions like 'archive' is actually a process of tagging an item by the label 'archived'

2.2.3 Plugins

Plugin is an essential concept in Uknow system, which comprises the implementations of varied functionalities in the system. A plugin should process a bunch of items, returning processed items. The number of items before and after need not to be the same, that is, a plugin can either shrink items or enrich items (filtering job) or process on contents of items.

User can choose a subset of provided plugins to employ on its items. Plugin can be either scoped to a 'tab' or can be applied system-wide.

Plugins can be configurable, but configuration is plugin-dependent.
Examples of plugins:

Tag-filtering Allowing users to choose preferred tags.

Highlight Highlight significant words in an item.

Emotion tagging Detect emotion of an item.

Face detection in image Detect faces from images of an item.

Related items Recommend related items for users to discover new contents to read.

2.2.4 Tabs

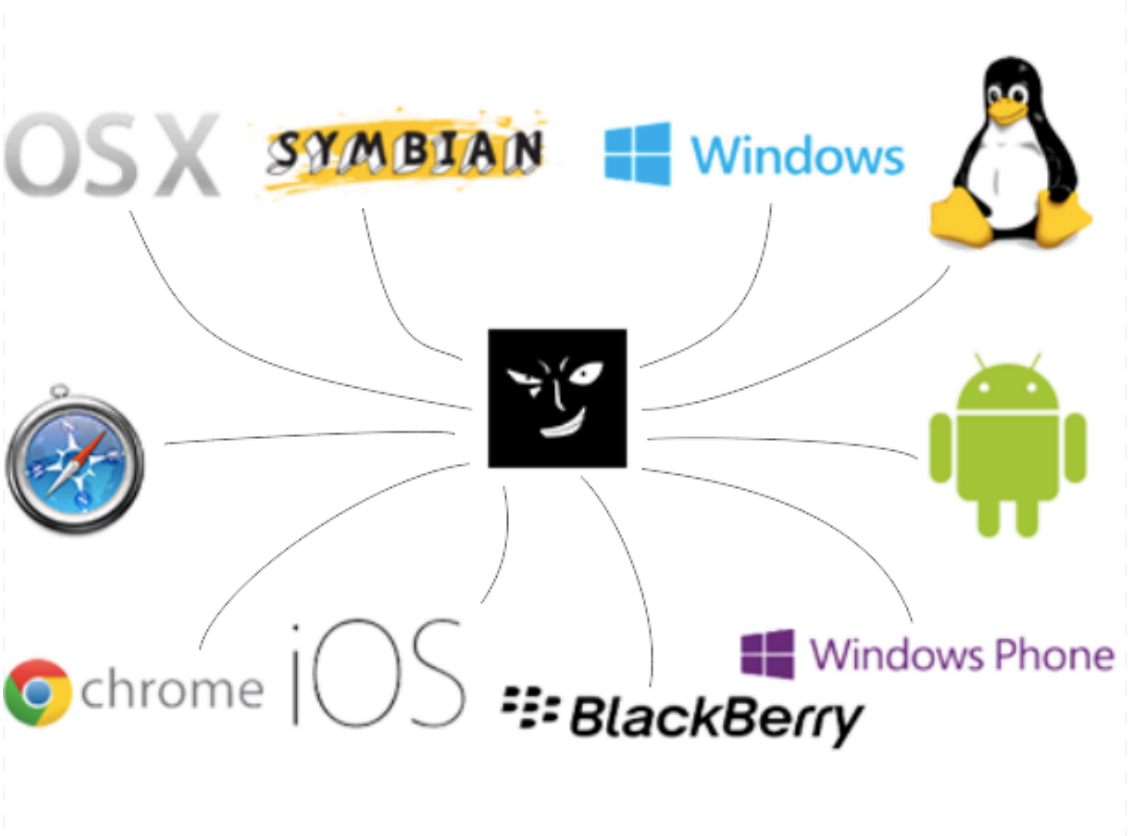
Tab is a collection of filtered items, in which the filter is defined by plugins. A typical use of tab is to divide items into different categories, which can be exclusive or not.

Tabs can be added or removed on-the-fly with 'add' or 'remove' button, add define its behaviour with plugins.

Moreover, tab layout, item flow style, etc., are also configurable via tab configuration.

2.3 Cross Platform

Uknow system shall include several clients including: web app, desktop client and mobile app. The users should have the freedom to choose their preferred platform, and get uniformed user experience.



2.3.1 Web app

Web apps are built in standards-based technologies such as HTML5, CSS3 and other modern web tech. Without any special translations, conversions or re-programming, a web app can run on pretty much any platform with a modern, standards-compliant web browser. Once a web app is launched, users on iPhones, iPads, Android phones, the Kindle Fire and Windows Phones can all access the same app and run it just as well as on any other platform.

The web app should render correctly on any of these desktop browsers:

Google Chrome/Chromium a freeware web browser developed by Google.

Mozilla Firefox a free and open source web browser by the Mozilla Foundation and its subsidiary, the Mozilla Corporation.

Safari developed by Apple Inc.

Opera developed by Opera Software.

Note that support for Internet Explorer by Microsoft is not planned.
As for mobile browsers, the following platform should be supported:

Opera Mobile running in Windows Phone and Symbian.

Safari pre-installed on all versions of iOS devices.

Chrome Mobile available in app store for iOS and Android.

2.3.2 Desktop application

Considering performance issue that web applications will always run in a sandbox and won't have full access to native resources and security and interactivity, desktop application is still needed on many popular platforms including:

Windows which dominate the world's personal computer market with over 90% market share.

Mac OS X a Unix-based graphical interface operating systems developed, marketed, and sold by Apple Inc.

Linux a Unix-like and POSIX-compliant computer operating system assembled under the model of free and open source software development and distribution.

For different platforms, different formats of installation file should be provided as:

Windows a msi or exe executable file for installation.

Mac OS X a dmg file containing a standalone app bundle.

Linux a tarball with a shell script for automatic installation.

2.3.3 Native Mobile App

Android the app should be available on Google Play Store.

iOS like Android, user can download the app on App Store.

3 Non-functional Requirement

3.1 User Management

3.1.1 Register Page

The register page contains an input form, users can only successfully register an account if:

- The username is unique in our database, and contains only letters and digits.

- The password is no less than 6 characters, and does not contains digits only.
- User repeat the password again correctly.
- User provided a valid email address.

After a valid register form is submitted, the server side would again check the input values. If the form passes, a validation email would automaticlly be sent to the user with a confirm code. Otherwise, the client will redirect the user to the original register page again, and alert about this error message. This process is illustrated well in Fig. 7.

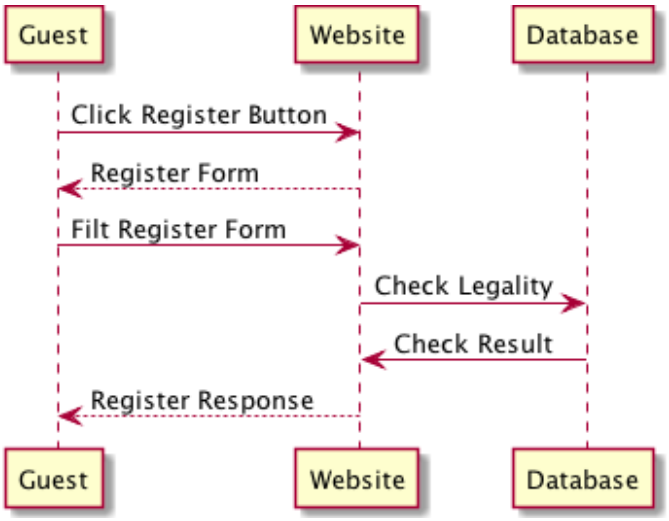


Figure 7: Registration process

3.1.2 Confirm Page

A link to the confirm page should be attached in the email sent to the user. An user account will be activated only if:

- The email address was registered before but has not been activated yet.
- User visits the confirm page within the time constraints.
- A correct confirm code (dependent on this account only) is provided by the user.

Once an account is activated, user can then edit profile, change password, and use all the other services provided by Uknow.

3.1.3 Login/Logout Page

A visitor will be redirected to login page everytime he/she tries to visit a restricted resource. In the login page, the user is asked to provide username together

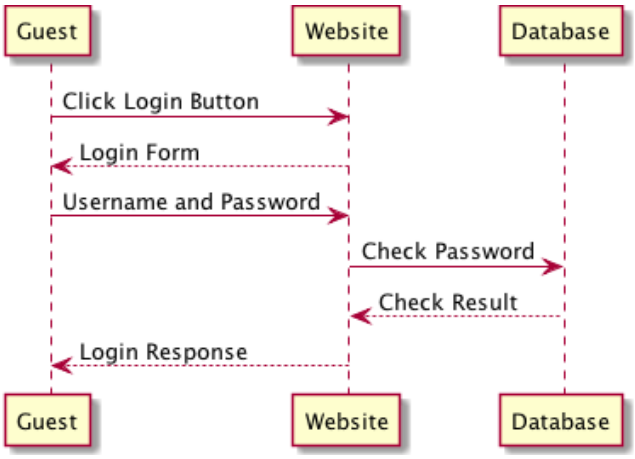


Figure 8: Login Process

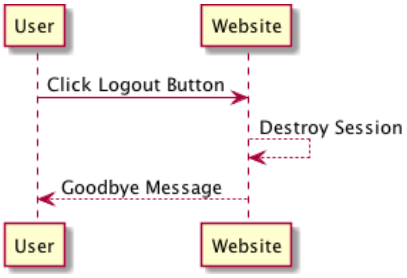


Figure 9: Logout Process

with password. A session is stored in cookie after a successful login, otherwise, a "password error" message will be displayed. On logging out, this session shall be destroyed and user will be redirected to the home page. The login/logout process is illustrated in Fig. 8 and Fig. 9.

3.1.4 Profile Page

User can view and edit the following items:

- Avatar, this could be uploaded by user or selected from Gravatar.com
- Nickname, which is displayed in the front page of the site.
- Sex, you know what I mean.
- Age, those under 16 is supposed to be warned before accessing items with specific labels.
- Email address, another validation email will be sent if this item is changed

On the other hand, users should be able to provide access to some social networking sites for us to collect information including but not limited to:

- Weibo, a twitter-like site in mainland China.
- Renren, a facebook-like site which is very popular among Chinese students
- QQ, a MSN-like IM tool running by Tencent
- Tsinghua Netclass, specially provided for Tsinghua student

Finally, if user wishes to delete his/her account, a confirm email will also be sent.

3.1.5 Change Password

The password of an account could be changed only if:

- The correct old password is provided
- A new valid password.
- The new valid password is typed twice and matched well.
- The link to a confirm page which was emailed to the user was clicked within 24h.

3.2 Data Flow and Internal Logic

The back-end of Uknow InfoHub system comprises mainly five components: item fetcher, prefilter, data storage, postfilter and API website, as illustrated in Fig. 10. Data passing between modules are encoded using json formats.

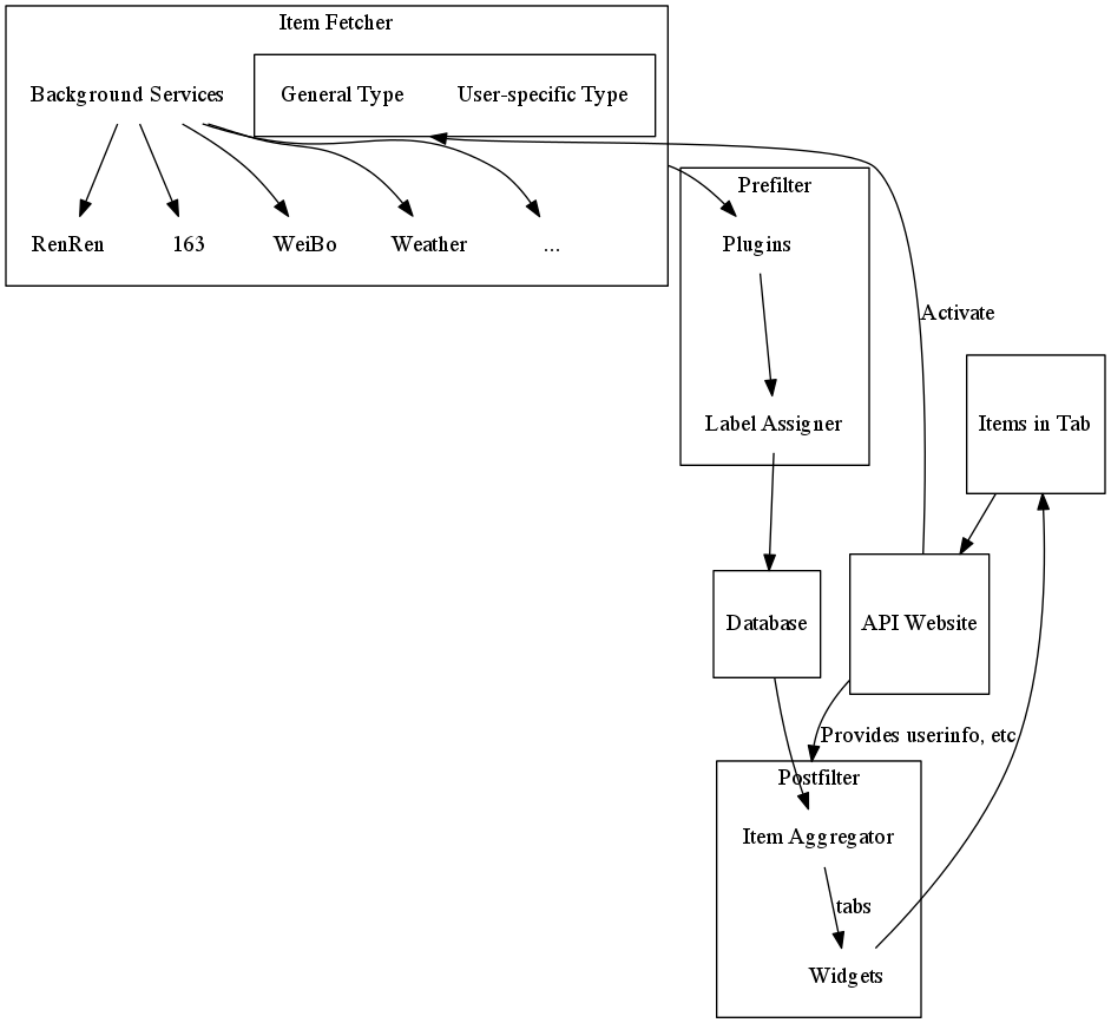


Figure 10: Overall architecture of Uknow backend

3.2.1 Item

An item is an abstract concept standing for a piece of collected information to be presented to the user. It can be almost anything, such as a status on SNS like renren, a piece of news from Netease, or a Weibo post or tweets. An item is represented as a dictionary structure programmatically. Following attributes are associated with an item:

1. An integer unique ID, which is used system-widely to identify a specific item.
2. Item fetcher type, which could be general item fetcher or aa user-specific item fetcher along with the user ID. See descriptions below for further details.
3. A set of labels, each of which describes a property of the underlying item.

Possible labels include data source (websites such as renren, Facebook), data category (news, SNS updates, and etc.), and inferred information such as its importance to the user. An item could either be public or user-specific, and the associated labels could be different for different users.

4. A brief description as human-readable plain text, which could also be used for plugins involving NLP(nature language processing) and item searching.
5. Creation time
6. Other attributes for a specific item category.

3.2.2 Item fetcher

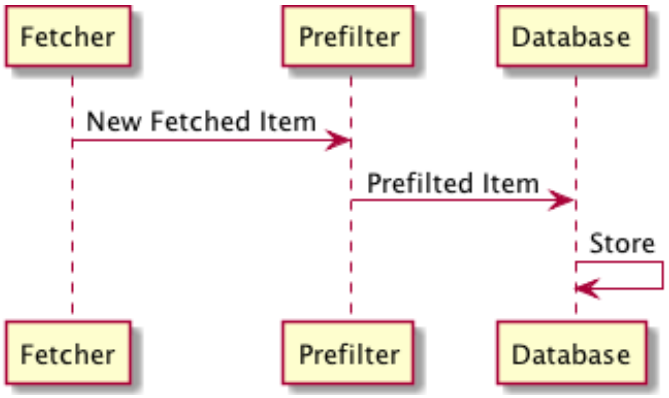


Figure 11: Workflow of Fetcher

Item fetchers simply retrieve items from various information sources, adding basic attributes such as ID, description, source URL (if available), and then forward items to prefilter for later processing, as illustrated in Fig. 11. There are two types of item fetcher:

General Item Fetcher

This kind of item fetcher collects public such as news and weather. They should always run in background and provide new data. No user login or authentication should be involved at this stage.

User-specific Item Fetcher

This kind of item fetcher is activated by user action (such as login); when activated, it would collect information that could only be gained after authentication (such as connecting to SNS sites). The API website sends an activation signal along with corresponding user ID to all this kind of item fetchers, and they should do the work in background. Note that the activating process is asynchronous and the user could only get results of some time

earlier to his request; however the latency should be fairly small if fast servers and Internet connection are guaranteed.

3.2.3 Prefilter

Prefilter is responsible for processing an item before putting it into database. Prefilter consists of a series of plugins, each of which takes the item dictionary as input and outputs a possibly modified item or aborts processing of current item and discards it. This series comprises two parts configured by the system administrator and the users respectively. For the system-wide part, common plugins include infer extra labels from item content or filtering illegal items. For items produced by a user-specific item fetcher, plugins enabled by the corresponding user is referred to as the "user part" and would be applied on this item.

3.2.4 Database Storage

After processed by prefilter, if an item is not discarded, it would be stored in the database. Due to the heterogeneous nature of data items, we decided to use MongoDB as the storage back-end. The advantages would be discussed later in Sec. 3.3.1.

3.2.5 Postfilter

Postfilter contains two stages:

1. Item Aggregator: When a user requests items in a tab, possibly with time, page number or seaching keyword constraints, the item aggratator simply finds items meeting those conditions and return them in some order.
2. Like prefilter, a sequence of system-wide and user-defined plugins are then applied to the items, which, for example, could deduplicate items in the same tab.

3.2.6 API Website

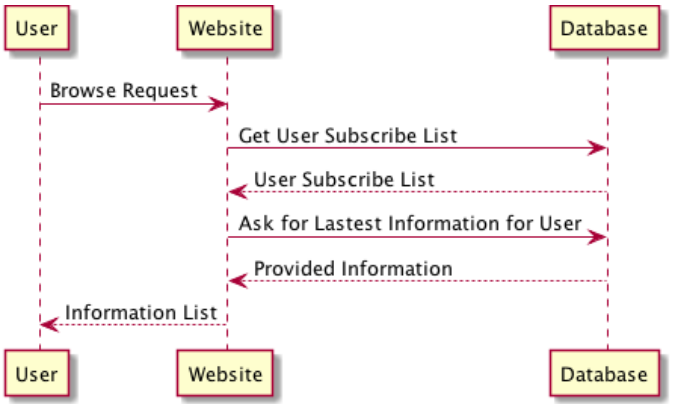


Figure 12: Workflow of the API website

API Website is the interface of the back-end system which interacts with users, though indirectly. It accepts user requests, invokes corresponding functions as described above, fetches and presents result back to user, as displayed in Fig. 12. Refer to Sec. 3.6 for further discussion about the API interfaces.

3.3 Scalability

3.3.1 Data Storage

Data storage is implemented using MongoDB[5], which is a leading NoSQL databse proven to be fast, stable, flexible and reliable. Care has to be taken on creating appropriate indexes to allow fast querying. Techniques such as replication and sharding could be adopted to enhance efficiency and reliability, only if there are enough servers. With the help of such mature production, we do not have to worry too much about data storage.

3.3.2 Background Workers

As mentioned above, there are long running background workers fetching items from public information sources. As the total number of such sources is limited, it is not necessary to deploy many of those workers, and this obviously has no scalability issue with growth of user group.

However, user-specific fetchers must be able to scale with the number of users. Here we set up a worker cluster. The cluster contains a task queue and distributed worker nodes. When a user activates a fetcher, a task is added to the queue, and some worker node then picks up the task and finishes it asynchronously. We

[5] http://www.mongodb.org/

choose Celery[6], a distributed, asynchronous task queue to be the queue framework in Uknow. For a worker node, it must be equipped with high-speed Internet connection; and we can use co-routines (e.g., greenlet[7] in Python) to reduce CPU workload.

Now the system is fully horizontally scalable; theoretically a large number of users could be served simultaneously if sufficiently many servers are available.

3.4 Availablity

The key to increasing availability is to introduce redundancy. There could be multiple queue schedulers and multiple worker nodes, failure of any of which would not influence the whole system. The replication mechanism provided by MongoDB also guaranteed the availability of data storage. For API website, we could set up an Nginx[8] server with reverse proxy to multiple API servers, which makes the whole system almost as reliable as anyone would request.

3.5 Security

The python drivers for MongoDB makes it impossible to commit attacks like SQL injection. It is hard for our system itself to contain any security vulnerabilities. Care has to be taken on system security such as server software, operating system configuration and etc.

3.6 Web API

Since we provide several different clients, the communication between client and server should be universal. Thus, web API implemented in Python and based on JSON will be used.

JSON (JavaScript Object Notation)[9] is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999[10].

3.6.1 Basic Structure

All returned valued is a dict containing some specific keys. If an error occurred during an API access, a key named `error` will be provided with a key named `msg` showing the reason of error. Otherwise, the desired data will be returned.

[6] http://www.celeryproject.org/
[7] https://pypi.python.org/pypi/greenlet
[8] http://wiki.nginx.org/Main
[9] http://www.json.org/
[10] http://www.ecma-international.org/publications/files/ecma-st/ECMA-262.pdf

3.6.2 User Management

All user management related action should be accessible by API including but not limited to:

- Register an account
- User login (validate)
- Edit profile
- Withdraw authority for specific third-party web service
- Delete an account

When a password is sent, SSL/TLS protocal should be used.

3.6.3 Data Flow

Any client of Uknow shall follow an uniformed protocal to commuticate data with Uknow backend server. Thus, Uknow server shall provide web API for clients to fetch feeds data, mainly consisting of the following parts:

1. Get new feeds
2. Tag an item
3. Update tab configurations
4. Update tab-level plugins

4 Documentation Requirements

4.1 License

4.1.1 Registration License

In order to provide better service to our users, users shall claim to agree our registration license when applying for an account. The registration license shall cover the following parts:

1. Rules on using our service, including the constrains on using, transfering the account, the limited use of the information we provided.
2. The privacy on the personal information provided by users in registration.
3. User shall understand and accept the service provided by Uknow before registration, and take full responsibilities for any kind of abuse of our service.

4.1.2 Disclaimers

In order to avoid unnecessary conflicts with our users, users must sign our disclaimers before using the information collector of Uknow InfoHub. The disclaimer shall cover the following contents:

1. Uknow is not responsible for, and expressly disclaims all liability for, damages of any kind arising out of use, reference to, or reliance on any information provided by Uknow.
2. While the information provided by Ukonw is periodically updated, no guarantee is given that the information is correct, complete, and up-to-date.
3. Though Uknow might provide links or direct data from other Internet Resources, Uknow is not responsible for any kind of damages arising out of visit to those sites.
4. Uknow doesn't own and shall not preserve copyright for any information provided in our service, but will show the user the origin of those information. Any kind of offence to the copyright of the original author of the information provided by us is seen as a direct offence to the author.
5. Though Uknow might contain advertisements or other types of endorsement for products or services from third-party companies, Uknow has not investigated and does not assure the fidelity of the claims by any advertiser. Product information is based solely on material received from suppliers.

4.1.3 Notice to Right Holders

Uknow InfoHub developer group understand and respect the right of any individual or entity. Thus, a notice to right holders must be provided on our website for right holders to read and get instructions on enforcement of their corresponding rights related to the behavior or the services provided by Uknow InfoHub developer group. The notice shall cover the following parts:

1. Copyright. While Uknow InfoHub provides simple information aggregation for our terminal users, Uknow respects and does not offend the copyright of the original author of the information presented by us.
2. Privacy. Uknow might fetch and parse information from various sources, but with the authority of our service users. Uknow will not initiatively collect private data of any individual or entity.
3. Use of our service. All users of our services are immediately granted the right of using our service according to the user agreement right after registration.

4.2 Help Manual

4.2.1 Basic Use of Our Service

For the purpose of getting users quickly involved in our product, an instruction manual on the basic use of our service is necessary. This instruction manual shall contain the following parts:

1. The functionality of our service, and the instructions of using them on our clients.
2. Examples on the use of our key functionality, including tab-like information classification, auto-tagging, etc.
3. Precautions and attentions that our users should be aware of.

4.2.2 API Documentation

Since Uknow InfoHub provides an extensible and flexible structure, allowing developers to create their own plugins to extend the functionalities of Uknow, an open API documentation is needed for potential developers to read and develop their plugins for our platform accordingly. Meanwhile, as the scale of this project grows, there will be inevitable difficulty in its maintainance and update. Therefore, an detailed API documentation of internal classes and interfaces is also necessary for the everlasting development of this project.

The API documentation of Uknow should be presented in good-looking style, in cross-platform formats(HTML, pdf, etc.), and be easy to refer to. The names of classes and terms in the documentation should be well referenced to its definition, providing readers with better readability.

5 Future