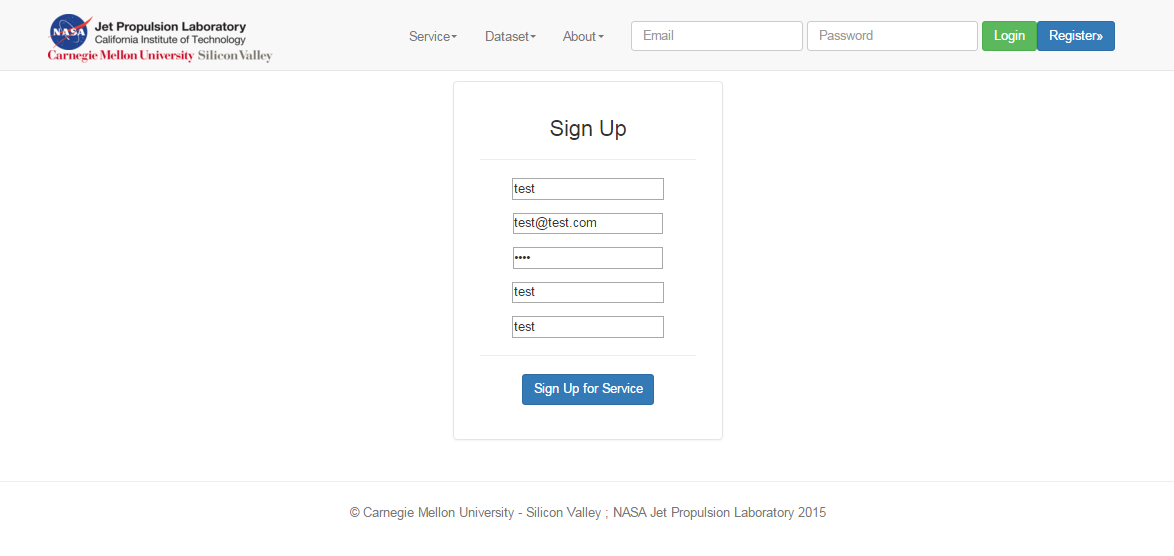
**Test suite**

We will test the following functions of our website and analyze the result:

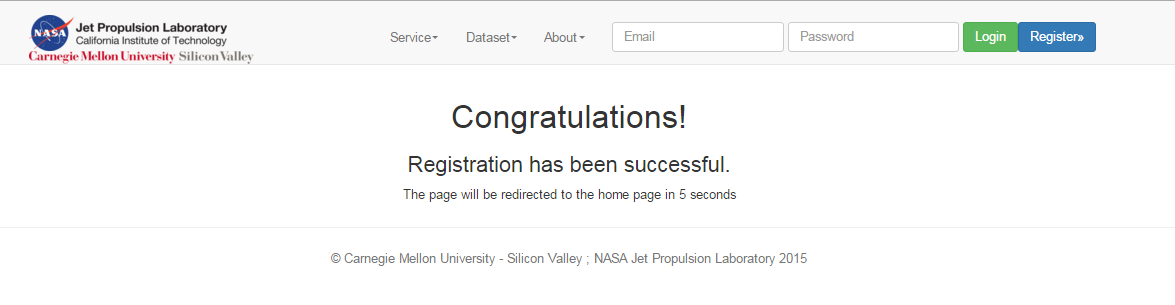
* Sign-up and login.
* Comment a service with rating star grades in comment area.
* Add a friend
* Use “@” hashtag to mention a friend
* Search a hashtag “#” after mention a service
* Search a service
* Get top 3 graded services
* Get most recently used service

***1) Sign-up and login experiment:***

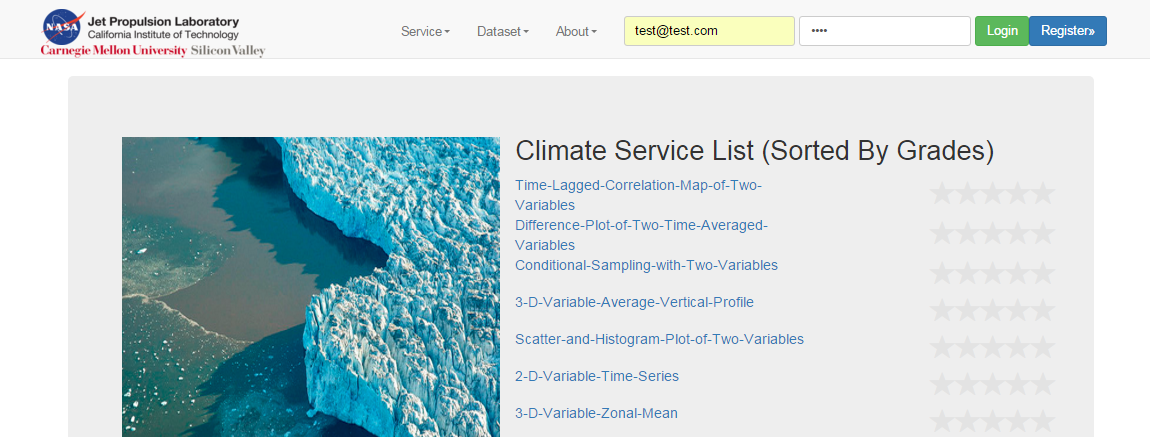
Sign-up page



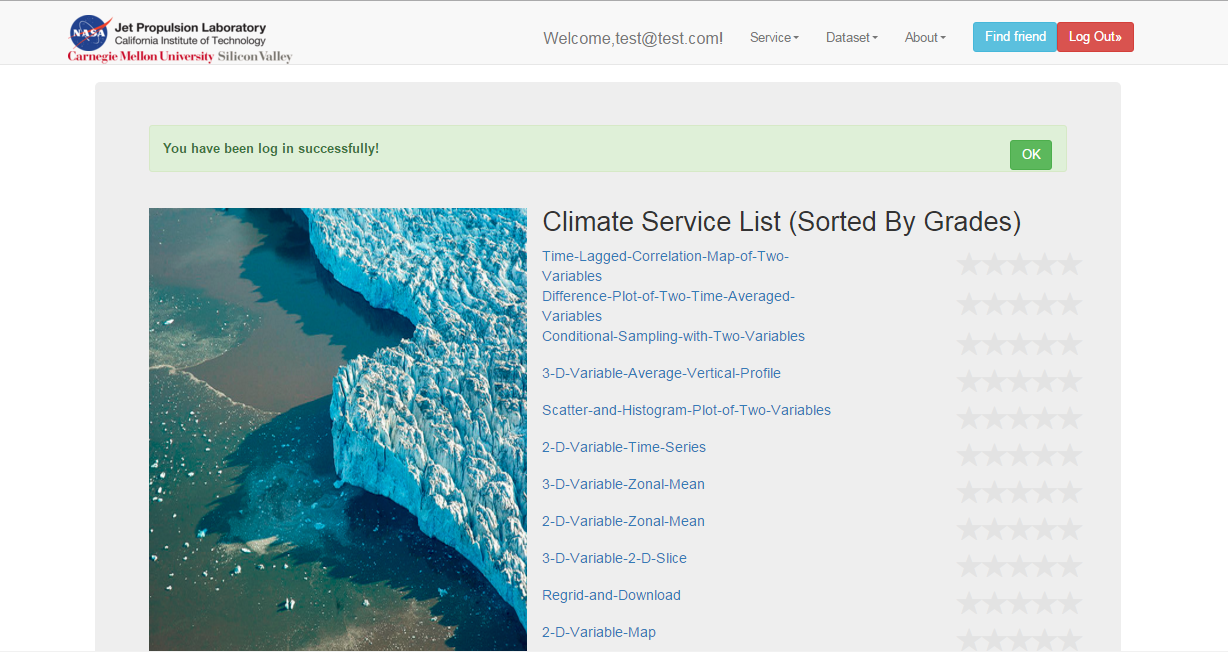
Registered successfully



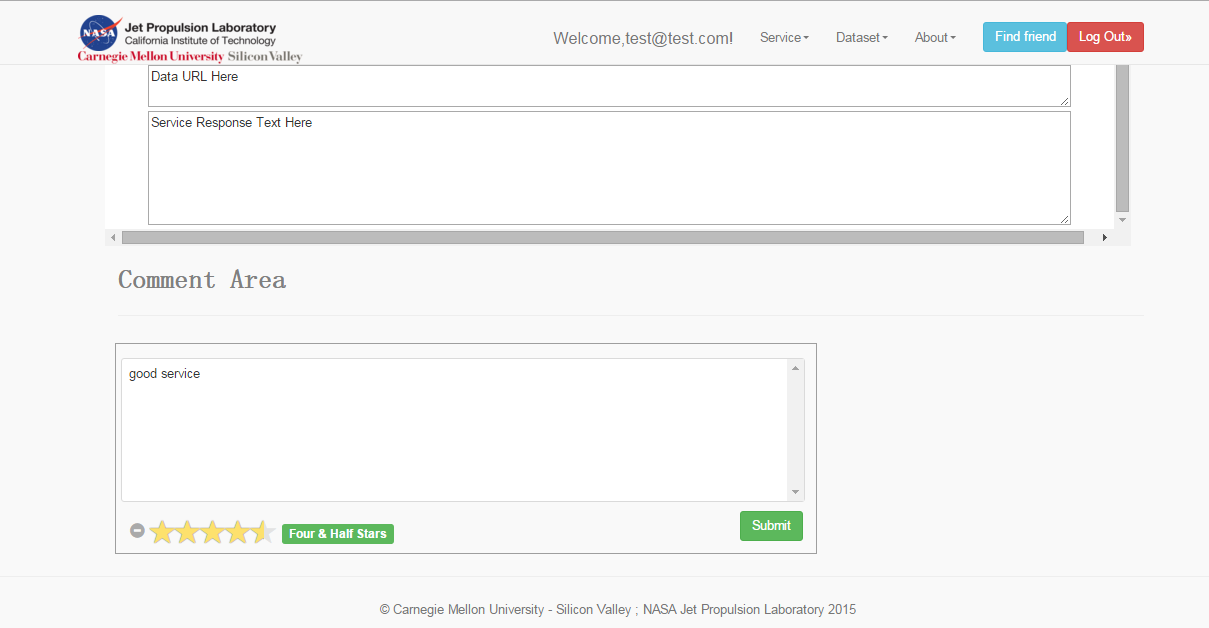
Login test:



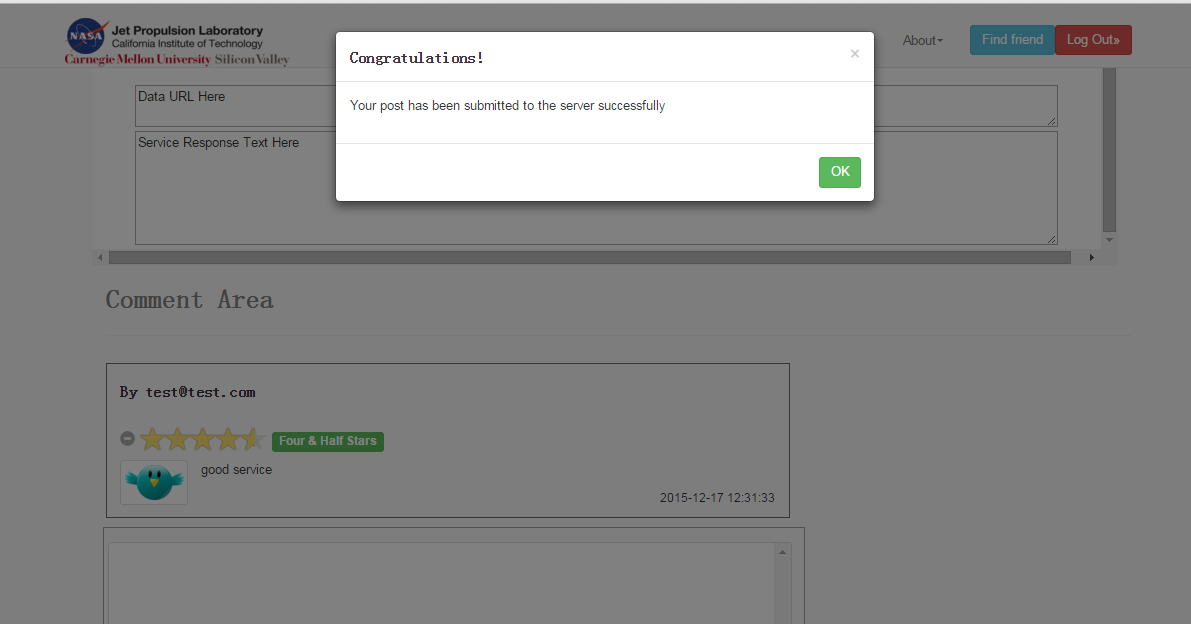
Successfully login the website:



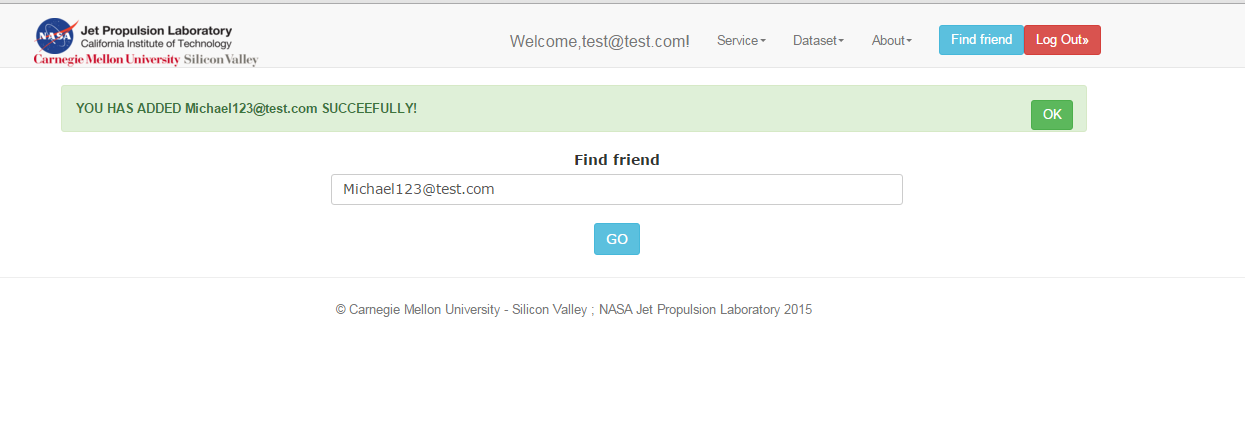
***2) Experiment of commenting a service with rating star grades in comment area.***



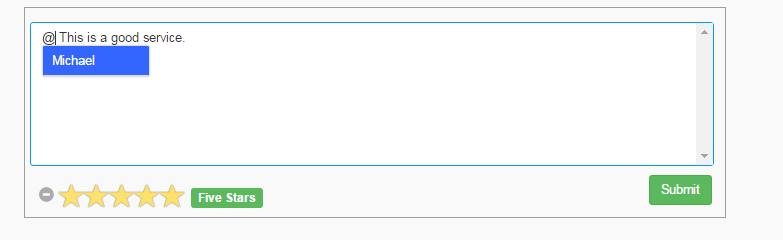
After commented a service successfully:



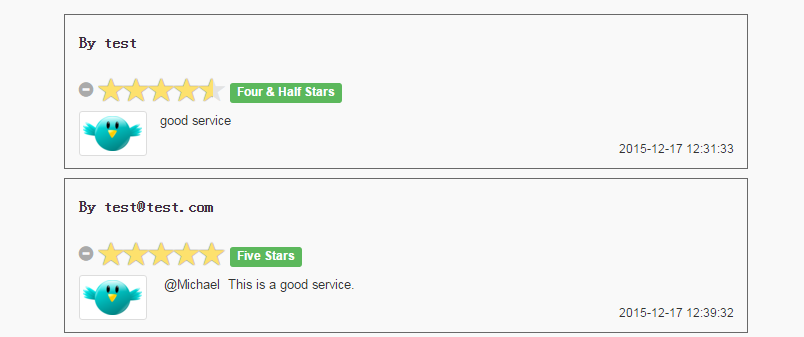
***3) Experiment - Add a friend***



***4) Experiment - Use “@” hashtag to mention a friend***

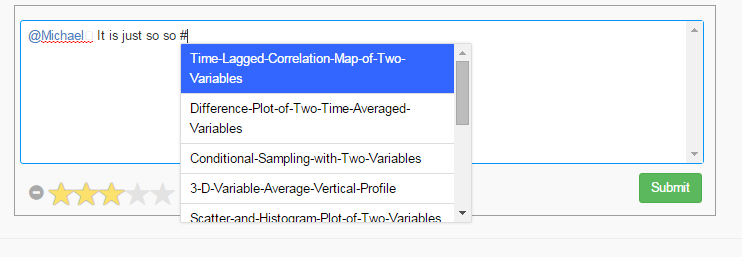


Comment area after successfully commented this service:

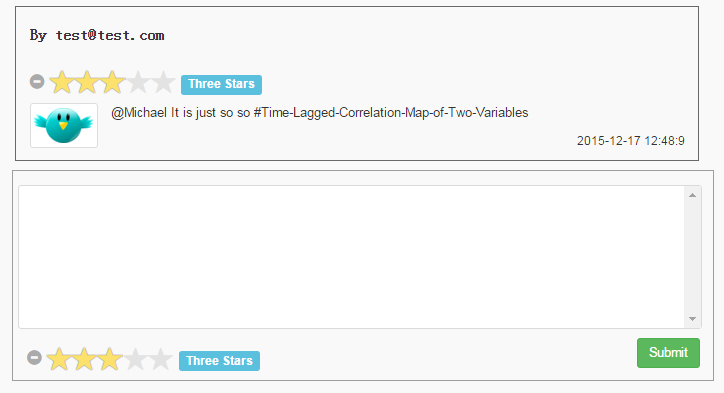


***5) Experiment - Search a hashtag “#” after mention a service***

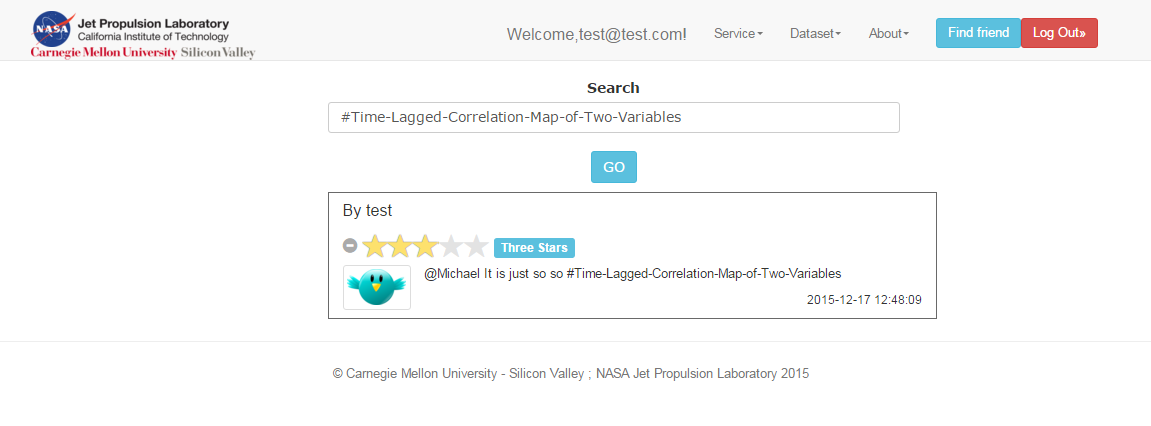
Using “#” to mention a service:



Comment area after using the hashtag “#” to mention a service:

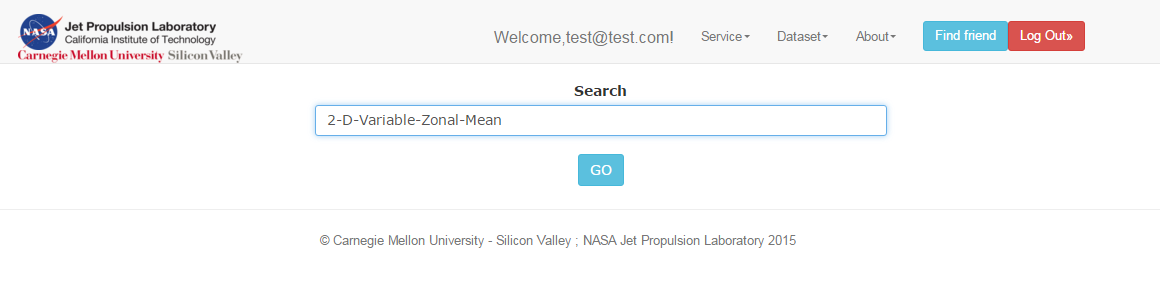


Search the hashtag with the service mentioned:

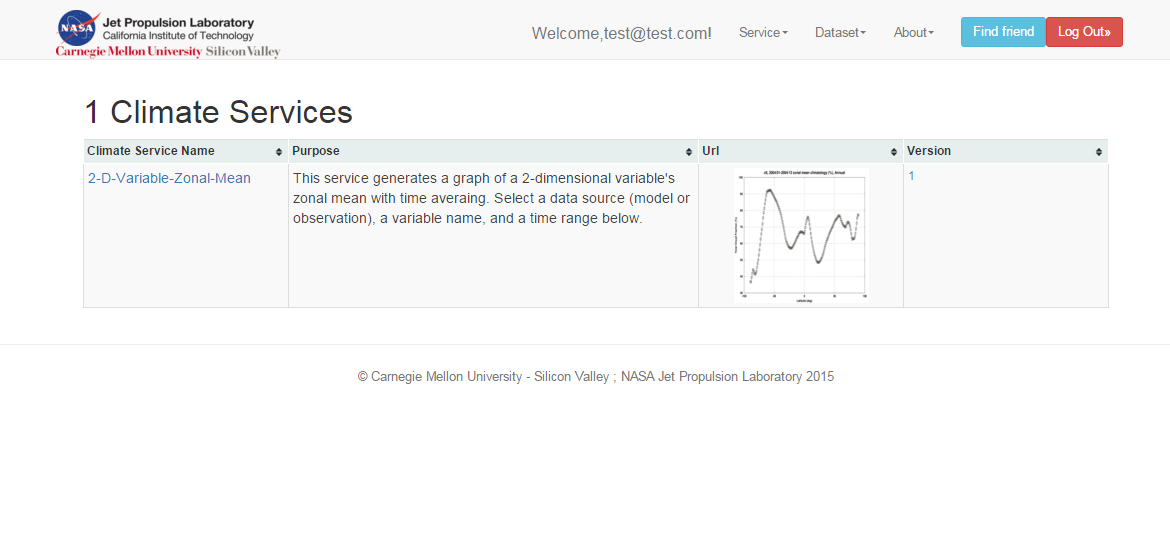


***6) Experiment - Search a service***

Search:

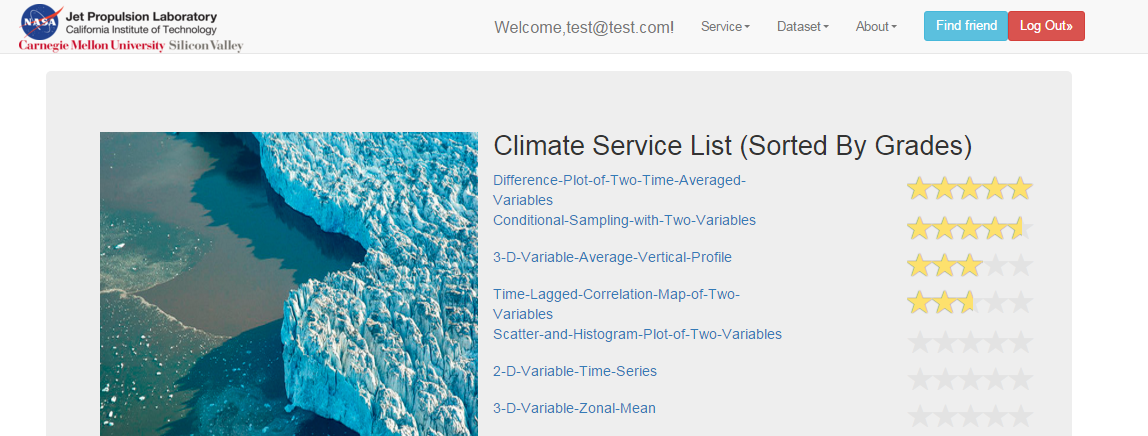


Result:

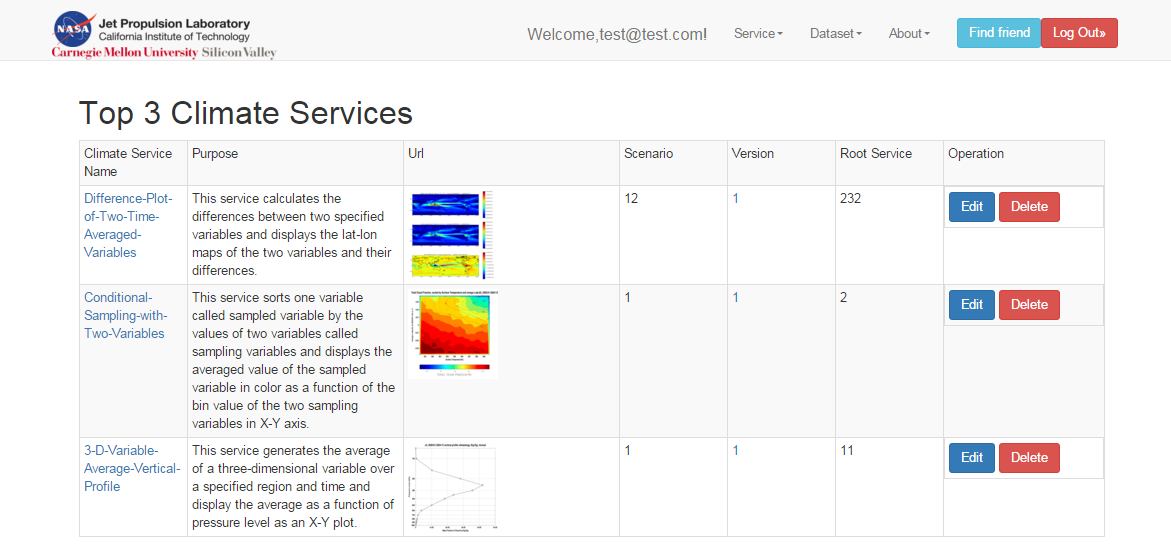


***7) Experiment - Get top 3 graded services:***

Home page service list sorted by grades:

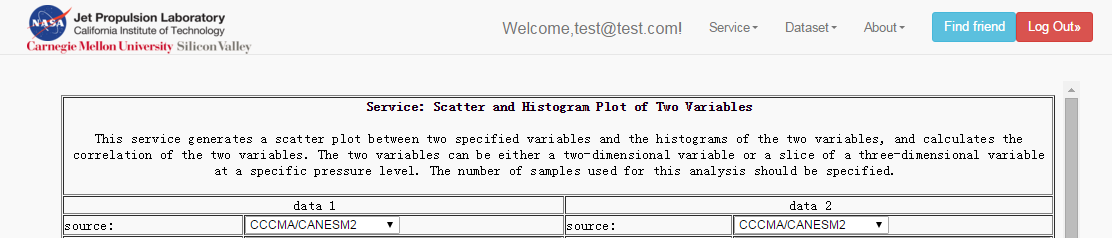


Results of getting the top three grads link

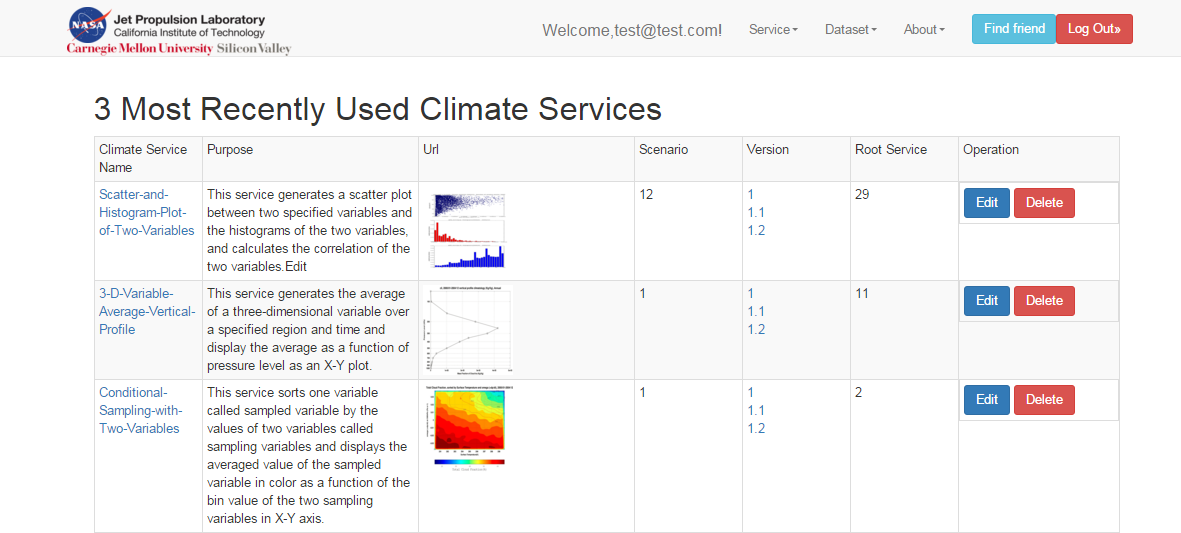


***8) Experiment - Get most recently used service***

First we will click one service and we will check if it is the most recently used service:



Result of getting most recently used service:



As you can see, now Scatter-and-Histogram-plot-of-Two-Variables is on the top of this list.

**7. Conclusions and future work**

***Conclusions***

For this project, we have implemented all of the requirements as follows:

* Implement a Climate Service public main page to list all climate services
* A user can discover a climate web service
* A user can view a climate web service (version control)
* A user can evaluate a climate web service
* Show popular climate services in Climate Service public main page
* Within a user’s comment, a user can mention a friend using sign “@”
* Within a user’s comment, a user can mention a climate service using sign “#”

In addition, we also implemented three additional functions as follows:

* Build sign-up and login pages and related functions by ourselves.
* Adding friends function.
* Get top 3 clicks climate services in a list.

***Future work***

To improve our website in future work, there are three possible ways:

* Build a home page for each registered user.
* User can send messages to each other.
* System can notice a user that when he or she has been mentioned with “@” by a friend.
* Users can manage the climate services that they have posted.

**8. Contribution of each team member**

**Front-end server:**

*Jiyu Shi*

* Implement service discovery and searching function and related controllers and web pages in front-end server.
* Implement login and sign-up functions and related web pages in front-end server and relate communications mechanism with back-end server.
* Implement adding-friends function in front-end server and communications mechanism for interacting with back-end server.
* Implement climate service version control in front-end server.
* Implement functions for ranking the top 3 grades climate services and most recently used services in front-end servers.

*Zhongao Tang*

* Implement the Climate Service public main page to list all climate services sorted by grades in front-end server.
* Implement the comment area in which user can post comment and rate a service with star rating and related communication mechanism with back-end server.
* Implement the function that a user can mention a friend using sign “@”within the user’s comment in front-end server.
* Implement the function that a user can mention a climate service using sign “#” within the user’s comment in front-end server.
* Implement the function for displaying comments in searching result area that mention a service with “#” in searching result in front-end server.

**Back-end server:**

*Yuan Liu*

* Implement data model class design and corresponding CRDU methods for a climate service and a user.
* Implement API for providing a list of all services to the front-end server.
* Implement API for searching top 3 grades services and 3 most recently used services in back-end server.
* Implement APIs for responding users’ login and sign-up requests, and related controllers to store the users’ information in the database.
* Implements API for managing all comments and related controllers for manage the list of comments of each climate service.

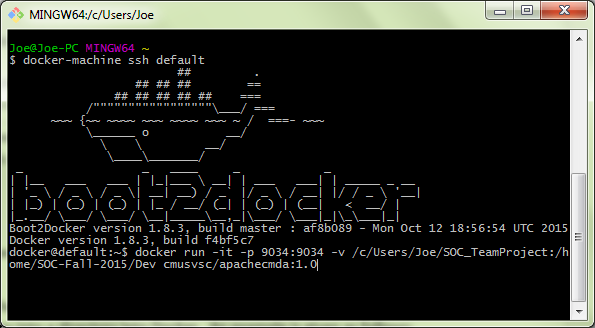
*Xiaoyu Wang*

* Implement data model class design and corresponding CRDU methods for users’ comment and users’ friend list.
* Design the database table structure in back-end server.
* Implements APIs’ structure design for communicating with front-end server.
* Implements APIs for responding users’ searching requests and related controllers’ functions.
* Implements API for a user’s adding friend request and related controllers’ functions and database operations.

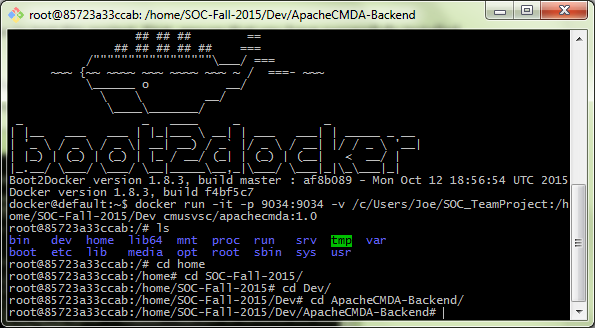
**9. Tutorial**

We have provided a tutorial for how to test this project. First, ensure that you have successfully installed Docker and related software.

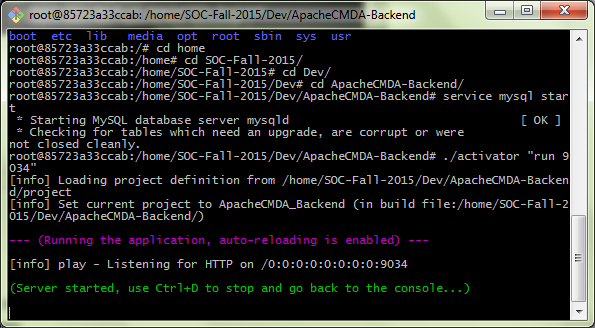
1) Load the provided image which has already configured the environment for this project. In addition, mount the back-end server directory into a directory into Docker. An example is given as follows:



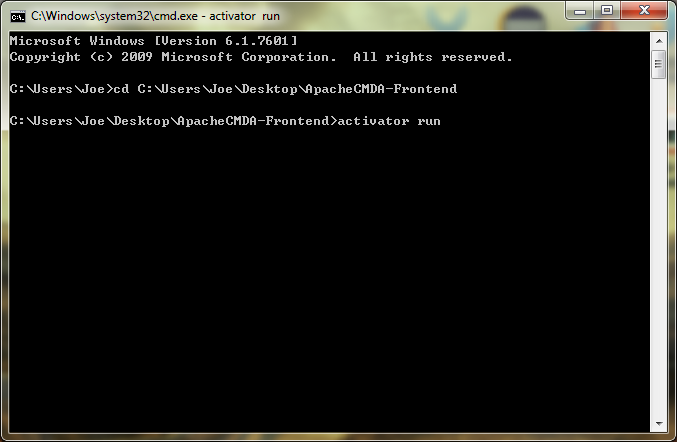
2) Change the directory in the Docker image to the directory mapped to the back-end directory in local machine. An example is given as follows:



3) Start the mysql service and start the back-end server at port 9034 with the command: ./activator “run 9034”. An example is given as follows:



4) Start front-end server in its directory at local machine, and an example is given as follows:

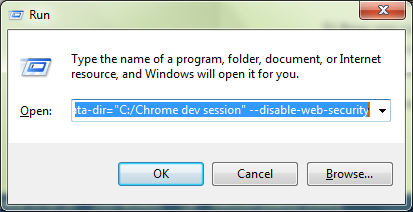


5) Now you can test our website in local port “9000”, please disable the web-security and enable cross-origin resource sharing of your browser because it might block our local AJAX request to back-end server. An example is given as follows:

Run the chrome browser in windows system:

chrome.exe --user-data-dir="C:/Chrome dev session" --disable-web-security

Following is the example graph:



6) Test the website at the browser and the URL is <http://localhost:9000/>, and the following is an example:

