**Wunderlist Implementation**

Steps to get client ID and secret for Oauth:

1. Goto the below link and register your App. <https://developer.wunderlist.com/>
2. Enter a valid call back url.
3. Now you'll be shown all your registered apps in below link. You can edit your call back url whenever you want <https://developer.wunderlist.com/apps>
4. Note down the client ID and secret which we are going to use them for Oauth.

Steps to get Access Token of the user using the above credentials:

1. Firstly, redirect the user along with user's teamchat mail-id and room-id to 'servlet\_url' noted in properies file.
2. Rerieve the mail-id and room-id in servlet to which the user is redirected.
3. Retrieve the parameters mail-id and room-id and apppend them seperated by a comma.
4. Now redirect user to the authorization URL which is <https://www.wunderlist.com/oauth/authorize?client_id>={client\_id}& redirect\_uri={redirectUrl}&state={mail-id,room-id}
5. Now response will be redirected to the call back url above provided.
6. Now retrieve the 'code' parameter and state as we sent.
7. Now send POST request to <https://www.wunderlist.com/oauth/access_token>
8. This POST should have a JSON body with client\_id, client\_secret and code which we recieved above.
9. In response, we'll get JSON object with access\_token.
10. Now store the access\_token in database and notify the user that he's connected.

Functionalities(keywords) providing here:

1. lists – It displays the names of lists present in the user's account.
2. addlist – It creates a new List in user's account.
3. viewinbox – It displays all the tasks present in the inbox.
4. Viewtasks – It displays all the tasks other than inbox's.
5. Addtask – To add a task to user's inbox.
6. Notify – To manage user's notifications.

Lists:

1. Send a GET request to <https://a.wunderlist.com/api/v1/lists?client_id>= {client\_id}&access\_token={access-token}
2. Get the JSON response and parse it to get the names of all the lists.

Addlist:

1. Ask the user to enter the name of the list.
2. Send a POST request to https://a.wunderlist.com/api/v1/lists
3. The above request should contain a JSON body with sinlge element title and add access-token, client-id in header.

Viewinbox:

1. Firstly, create a method to retrieve inbox-id. As there will be only one inbox it'll be easy to retrieve that.
2. Send GET request to https://a.wunderlist.com/api/v1/tasks?client\_id={client\_id}&access\_token={accessToken}&list\_id={listid}
3. Parse the recieved JSON to get the tasks in Inbox.

Viewtasks:

1. Send a GET request to <https://a.wunderlist.com/api/v1/lists?client_id>={client\_id} &access\_token={access-token}
2. Parse the JSON response omit the tasks present in the list whose type is inbox and print.

Addtask:

1. Send POST request to <https://a.wunderlist.com/api/v1/tasks>
2. Add JSON body with list\_id, title, starred(boolean)

Notify:

1. If user selects ON then create webhooks.

* Send a POST request to <https://a.wunderlist.com/api/v1/webhooks>
* Add JSON body containing list\_id,url(call\_back), configuration, processor\_type(always generic).
* Once it is created we'll get response code as 200.

1. If user selects OFF then delete webhooks

* Send http DELETE request to <https://a.wunderlist.com/api/v1/webhooks/>{webhook\_id[j]}?client\_id={client\_id}& access\_token={accessToken} &revision={revision}
* In the above parameters we can get user's revision by sending GET to <https://a.wunderlist.com/api/v1/user?client_id>={client\_id}& access\_token={accessToken}
* Notify the user that notificatications are turned OFF successfully on getting response code 200.
* Delete the user's details in webhook database.

Extra Details:

1. Using \*\*\*.properties file to store all the details.
2. Using embedded url so that everything will be opened in Teamchat app itself.