# UTAS Discord Bot MyLO Integration

This document shows how the UTAS Discord Bot would operate with MyLO integration. Specifically are examples for the commands in the screenshot below:

A screenshot of a computer

Description automatically generated with medium confidence

I have provided in italics additional information regarding the security issues at each step. *Note that all command uses are logged by the bot, with an audit log indicating:*

* *Timestamp*
* *Invoking Discord User ID*
* *Command Name and arguments*
* *The resulting response of a command is NOT stored, since that likely contains information like grades.*

## Example account linking flow using **/mylo connect**

1. User types /mylo connect or any other /mylo command
2. Private message with a Discord OAuth Link is generated
   1. *This message is only shown to the user who typed the command*

Graphical user interface, text, application, chat or text message

Description automatically generated

1. A Discord OAuth popup appears (inside the Discord app). This is done to ensure the bot knows exactly who clicked the link.

Graphical user interface, text

Description automatically generated

1. After Discord authorization, a browser window with a MyLO OAuth login screen is shown. In this sample, we just pass through student ID because we don’t know what MyLO OAuth provides back other than token
   1. *The redirect from the Authorize button generated by Discord to the MyLO page contains an encrypted state payload. This payload contains:*
      1. *The Discord message id of the original private message*
      2. *The Discord server id where the original private message was posted*
      3. *The Discord user id of the authenticated user*
   2. *The server id and user id are publicly accessible to any Discord user, however the Discord message id cannot be determined by other users, and thus cannot be spoofed.*

A picture containing diagram

Description automatically generated

1. After MyLO Oauth Completed, redirected to Bot server with token and info. At this point token and info is stored in database
   1. *The encrypted state payload from the previous step will be passed through the MyLO Oauth flow as with the Discord Oauth flow.*
   2. *Upon successful login, the original private message id is cross-checked with the authenticated Discord user id*
   3. *A tuple of [ Discord User ID, MyLO Oauth Token ] is stored in the database, nothing else is stored.*
   4. *The Oauth token at all times remains on the server-side and is not needed by the Discord client*

Text

Description automatically generated

1. Original private message automatically updated with a student id fetched using MyLO API to show the connection was successful.
   1. *All discord private messages (ephemeral messages) are not stored permanently on the Discord server chat logs, and only appear client-side for a short time (~15 minutes). They can be dismissed by the user at any time, and are irretrievable.*

Graphical user interface, text, application

Description automatically generated

## Disconnecting accounts using **/mylo disconnect**

1. User types /mylo disconnect

Graphical user interface, application, website

Description automatically generated

1. User shown a confirmation button to disconnect
   1. *This message is only visible to the user*
   2. *Only the authenticated user can press this button, there is no way for another user to spoof this button event as it goes through Discord’s servers*

A screenshot of a computer

Description automatically generated with medium confidence

1. User clicks the disconnect button and the private message automatically updates to reflect that the operation was successful.

Graphical user interface, text, application, chat or text message

Description automatically generated

## Example Mylo-enabled command: **/mylo grades**

1. User types in /mylo grades
   1. *The bot checks the database for a stored oauth token for the authenticated Discord user who invoked the command.*
   2. *There is no external interface to the bot which can request an oauth token for a given Discord ID, thus there is no way to obtain an oauth token for another user*
   3. *The oauth token at all times remains on the server-side, and is not needed by the Discord client.*
2. If an oauth token is found, it is used to run the required MyLO API request, and the bot returns formatted output as a private message
   1. *This message is only visible to the user who ran the command*
   2. *As before, all private (ephemeral) messages like this are not stored on Discords server chat logs*
   3. *The MyLO API request using the Oauth token is done on the server side with tested code to ensure only the correct Oauth token is used*
   4. *Regardless of the MyLO API request, the following safeguards are in place inherent to using Oauth:*
      1. *Only data that the authenticated MyLO student user could access can be returned. This is further limited by the registered scopes of the application*
      2. *The user will be authenticated with a scope to only do read-only actions, thus even if a write request was made it would return an error*
   5. *If no oauth token is found for the authenticated Discord user, the user is prompted to link their account.*

Text

Description automatically generated