# Additional Questions – Discord Bot

* **What student data (specifically) will the bot have access to from MyLO? Do you have a list of potential queries that would be run, and would it have access to personally identifiable information about the student?**
  + Summary of information retrievable, specific information below:
    - Student-level information:
      * Student ID
      * First Name
      * Last Name
      * Pronouns
      * Enrolled units past and present
      * Grades (Assessment item, score, feedback)
      * Assessment completion
    - Unit Coordinator-accessible information about students
      * Campus and Tutorial Allocations (accessible to unit coordinator via queries to “group enrolment”, where groups are automatically populated for students).
      * Email
      * Role (student, tutor)
      * Last access date
      * Current MyLO online status (true/false)
    - Unit info:
      * Assessment items and due dates
      * Content items and meta data
      * MyLO Unit Identifier
  + Below is a list of API requests the bot could make, and the structure of the data returned from that API endpoint
    - Note that for \*all\* cases below, data is only retrieved using the API and never stored, with the exception of:
      * OAuth access and refresh tokens.
        + Note this is \*never\* shared with the user client, as it is only needed by the server to make MyLO API requests.
        + This token will be encrypted when it is at rest in the database.
        + This token is associated with a Discord User ID
      * Unit ID (referred to as OrgUnitID in the endpoints below) – stored for each discord server manually entered by the Unit Coordinator.
        + This number is publicly accessible on the MyLO website as it is in all the URLs there.
        + Student association with a UnitID (i.e. enrolment) is **not** able to be determined via the database structure alone
    - Authenticated Students can only access information about themselves
    - Authenticated Unit Coordinators can access information about any student in the unit
  + Basic User Information
    - Endpoint: /d2l/api/lp/(version)/users/whoami [GET]
    - Returns:

{

**"Identifier"**: <string:D2LID>,

**"FirstName"**: <string>,

**"LastName"**: <string>,

**"UniqueName"**: <string>,

**"ProfileIdentifier"**: <string>,

**"Pronouns"**: <string>

}

* + Basic Unit Information
    - Not directly obtained, instead included in responses like the one below for enrolled units
    - Structure:

{

**"Id"**: <number:D2LID>,

**"Type"**: { <composite:OrgUnit.OrgUnitTypeInfo> },

**"Name"**: <string>,

**"Code"**: <string>|**null**,

**"HomeUrl"**: <string:URL>|**null**,

**"ImageUrl"**: <string:APIURL>|**null**

}

* + Enrolled Units (for bot to confirm that the Discord user should be in the server for a given unit, and for recommending servers they can join as alumni):
    - Endpoint: /d2l/api/lp/(version)/enrollments/myenrollments/
    - Returns a list of unit info the student is enrolled in and their role within it

{

**"OrgUnit"**: { <composite:Enrollment.OrgUnitInfo> },

**"Access"**: {

**"IsActive"**: <boolean>,

**"StartDate"**: <string:UTCDateTime>|**null**,

**"EndDate"**: <string:UTCDateTime>|**null**,

**"CanAccess"**: <boolean>,

**"ClasslistRoleName"**: <string>|**null**,

**"LISRoles"**: [ <string>, ... ],

**"LastAccessed"**: <string:UTCDateTime>|**null** *// Added as of LMS v20.21.7*

},

**"PinDate"**: <string:UTCDateTime>|**null**

}

* + Gradebook information
    - Endpoint: /d2l/api/le/(version)/(orgUnitId)/grades/values/(userId)/
      * Note there are other endpoints for getting a specific gradebook item
    - Returns a list of grade items for the unit:

{

**"UserId"**: <string:D2LID>, *// Available in LE unstable API contract*

**"OrgUnitId"**: <string:D2LID>, *// Available in LE unstable API contract*

**"DisplayedGrade"**: <string>,

**"GradeObjectIdentifier"**: <string:D2LID>,

**"GradeObjectName"**: <string>,

**"GradeObjectType"**: <number:GRADEOBJ\_T>,

**"GradeObjectTypeName"**: <string>|**null**,

**"Comments"**: { <composite:RichText> },

**"PrivateComments"**: { <composite:RichText> },

**"LastModified"**: <string:UTCDateTime>|**null**,

**"LastModifiedBy"**: <string:D2LID>|**null**,

**"ReleasedDate"**: <string:UTCDateTime>|**null**

}

* + Due assessment items:
    - Endpoint: /d2l/api/le/(version)/content/myItems/due/
    - Returns a list of upcoming items and the given user’s completion status

{

**"UserId"**: <string:D2LID>,

**"OrgUnitId"**: <string:D2LID>,

**"ItemId"**: <number:D2LID>,

**"ItemName"**: <string>,

**"ItemType"**: <number:CONTENT\_T>,

**"ItemUrl"**: <string>|**null**,

**"StartDate"**: <string:UTCDateTime>|**null**,

**"EndDate"**: <string:UTCDateTime>|**null**,

**"DueDate"**: <string:UTCDateTime>|**null**,

**"CompletionType"**: <number:CONTENT\_COMPLETIONTYPE\_T>,

**"DateCompleted"**: <string:UTCDateTime>|**null**,

**"ActivityType"**: <number:ACTIVITYTYPE\_T>,

**"IsExempt"**: <boolean>

}

* + (Unit Coordinator Only) Class List (unit enrolment):
    - Endpoint: /d2l/api/le/(version)/(orgUnitId)/classlist/
    - Returns a list of enrolled students in the following format:

{

**"Identifier"**: <string:D2LID>,

**"ProfileIdentifier"**: <string>,

**"DisplayName"**: <string>,

**"UserName"**: <string>|**null**,

**"OrgDefinedId"**: <string>|**null**,

**"Email"**: <string>|**null**,

**"FirstName"**: <string>|**null**,

**"LastName"**: <string>|**null**,

**"RoleId"**: <number:D2LID>|**null**,

**"LastAccessed"**: <string:UTCDateTime>|**null**,

**"IsOnline"**: <boolean>

}

* + (Unit Coordinator Only) Group Enrolment (for determining tutorial allocation and campus)
    - Endpoint:  
      /d2l/api/lp/(version)/(orgUnitId)/groupcategories/(groupCategoryId)/groups
    - Returns a list of groups for the unit, in the following format:

{

**"GroupId"**: <number:D2LID>,

**"Name"**: <string>,

**"Code"**: <string>,

**"Description"**: { <composite:RichText> },

**"Enrollments"**: [ <number:D2LID>, ... ] *// a list of student IDs*

}

* + Other information accessed through the API is about the unit and unit content only, and so isn’t personal information about a student. This information is Intellectual Property of the University, however information is transmitted rather than stored. Such information may include:
    - Query of Assessment tasks, quizzes
      * Returning title, weighting, due date, links
    - Query of Unit content
      * Returning module title, link, metadata, structure
* **Can the bot be added to any Discord server? Are there any requirements around security that can be enforced before the bot is added?** 
  + The bot can be added to any Discord server via a link, however any MyLO-integrated commands cannot be run by Discord users without first authenticating with MyLO and storing their OAuth token.
  + There is no way for an unauthenticated Discord user to obtain the MyLO information of another user from another server or the current server, meaning that even when the bot is added to an arbitrary server, there is no risk associated with this.
  + Bots can only be added to Discord servers by a server admin.
  + When the bot is added to a server, this action is logged with the following information:
    - Timestamp
    - Discord User ID who added the bot
    - Discord Server ID of the server the bot is added to
    - Discord Server Name of the server the bot is added to
  + An option is available to disable the bot from being added to new servers, and to have the bot added manually by me to servers upon a Unit Coordinators request, however this is not desirable.
  + I can remove the bot from any server at any time.
* **Have you considered the ramifications of hacked Discord accounts – and the potential risk of student data being exposed to third parties via this method?**
  + Upon linking their Discord account to their MyLO account, students will be provided with a reminder and tips on how to maintain account security. They will also be advised that if they believe their account is compromised they can immediately revoke permission for the bot to access their information through MyLO itself (this process is explained in detail here: <https://docs.valence.desire2learn.com/basic/oauth2.html#id5> )
    - After completing this process, all API requests by the bot for this user will fail
  + If a student still has access to their Discord account, they can at any time disconnect their MyLO account from their Discord account using the “/mylo disconnect” command. To reconnect, the user needs to re-authenticate with MyLO
  + If a third-party gained access to a student Discord account linked to MyLO they would only be able to access the information of \*that\* student and not any other students.
  + Additionally the authentication of the MyLO OAuth token must be periodically refreshed (every 20 hours), which reduces the risk of a past student who no longer interacts with Discord losing their account later on not realising it has been compromised, and having their information accessed.
* **What monitoring / metrics would be available to monitor the bot's usage and calls to MyLO for malicious activity?**
  + All API calls to MyLO from the bot will be logged to the database used by the bot with the following information:
    - Timestamp
    - Authenticated Discord user ID
    - API Endpoint
    - Parameter List (in some cases, this may include the student ID)
    - Note that the result of the API call will \*not\* be stored in the database
  + All Discord command invocations are logged with the following information:
    - Timestamp
    - Authenticated Discord UserID
    - Command name
    - Parameter list
  + This information would be monitored by myself for suspicious activity, and there is the possibility of automated-detection. Suspicious activity might include:
    - Frequent requests
      * Note that the D2L platform also has rate-limiting built-in
    - Requests to unexpected api-endpoints not used by the app
      * This should not be possible, given endpoints are hard-coded into the server requests
    - API requests without a matching Discord command invocation
      * This should also not be possible, as the bot can \*only\* respond to commands issued by students via the Discord client
    - Requests with abnormal parameter lists, such as the student ID of another student
      * Again, this should not be possible
* **If a third party gained control of the bot itself or its underlying database would that then expose any student with a stored OAuth token to have their student data accessed?**
  + In the event of a third-party gaining control of the code for the bot or the underlying database, the immediate action would be to revoke the OAuth credentials for the application on the D2L administration interface.
    - This would result in all future API calls resulting in an error.
    - The only remaining information the third-party would have access would be the invalid and encyrpted OAuth token/refresh tokens, and unit IDs (publicly known).
  + If a third party gained access to the source-controlled version of the code, they would be unable to use the information there to gain access to the database or the MyLO API, since all access tokens and authentication would not be committed to source control.
  + In the event of a breach AND the OAuth credentials of the bot (different to Oauth token of each student) were compromised, only operations with the pre-defined scopes the application has been registered would be successful. These scopes would be \*read-only\* to reduce risk of malicious activity. A list of all scopes can be found here: <https://docs.valence.desire2learn.com/http-scopestable.html>. Expected scopes:
    - dropbox:folders:read
    - content:toc:read
    - content:modules:read
    - content:topics:read
    - content:file:read
    - enrollment:own\_enrollment:read
    - grades:gradeobjects:read
    - quizzing:quizzes:read
    - users:own\_profile:read
    - (unit coordinator scope only) enrollment:orgunit:read
    - (unit coordinator scope only) groups:group:read
    - (unit coordinator scope only) users:profile:read