**FACEBOOK IDS**

**Azure Classes and Controllers:**

* Startup
  + Sets up routing configuration for all controllers
  + Initializes database context using Entity Framework Code First
  + Adds several test users, households, and household members to the context
* Friend Controller
  + Primary role is to retrieve the households of Facebook friends of the current user who also use the app, but the user is not yet a member of. This is accomplished via an authenticated GET request.
  + The friend controller is called on the app’s main/welcome screen after the user logs in to allow the user to see the households of his or her friends that also use the app. The user needs to view these households in order to request to join them.
  + If the user is already a part of a friend’s household, this household will not appear in this request’s results.
* Household Controller
  + The primary role of the household controller is to get, create and update data related to households. A household is the basic communal space that users can be a part of. The relationship is a user can have many household memberships (HouseholdMember), and a household can have many memberships. Two other functionalities are also associated with households- voting and messages. Household members can participate in voting and messaging which is associated with a particular household. All household members can view and participate in the voting and messaging.
  + GET request that returns a list of households that a user is a member of.
  + GET request that returns a specific household by its id with the following info:
    - Household name
    - Household description
    - Name of currency
    - Name of the Landlord
    - List of household members
  + GET request that returns a household membership (HouseholdMember) based on a household id and user id.
  + POST action that creates a new household. This is called from the client app's home screen when the user would like to create a new household. The user that created the household will automatically become the landlord. Returns a message indicating success or failure of the creation.
  + POST action that updates the basic info about a household such as the household's name, a description, and the name of the currency. This method is called from a household's home screen to allow the landlord to edit these fields. Only the landlord may edit these fields:
    - Household name
    - Household description
    - Name of currency
  + GET request that returns list of household members for a specified household id.
* Message Controller
  + The primary role of the message controller is to store and retrieve messages for each household in Azure Document DB. Household members can create/post and view messages for a household, which are viewable to the entire household. Document DB is useful for this purpose because it allows for flexible schema which is useful if other data fields should eventually be added to the messaging functionality, and it also allows for Azure Search which will allow users to search prior messages.
  + GET request that returns all of the messages in the collection
  + GET request that returns all messages for a specified household.
  + POST action that creates a new message in Azure Document DB
* Registration Controller
  + The primary role of the registration controller is to register a user, i.e. save their info in Azure SQL, once they've authenticated through Facebook. This allows their Facebook user ID and name to be stored in Azure SQL rather than having to make a call out to Facebook's Graph every time that info is needed. The user does not have to initiate this action: once they authenticate through Facebook these methods are automatically called by the client app.
  + GET method that returns the registration status of the current user.
  + POST method that allows any authenticated user to register. This method is called automatically by the client app after the user logs in / authenticates with Facebook. If the user already exists no action is taken, but if it’s a new Facebook user their info is added to the Azure SQL database.
* User Controller
  + The primary role of the user controller is to retrieve data about users.
  + GET method that returns all users (mainly used for dev/testing)
  + GET method that returns a user by the id specified.
  + GET method that returns the current, authorized user.
* Voting controller
  + The primary role of the vote controller is to allow users to view, vote, and create proposals in the household. A "vote"/"proposal" relates to changes in membership, user karma, and the landlord. For example, if a user wanted to join a household, there's a "Join" button on the client app that will create a new vote (NewMember type) via this controller. The new vote will then appear in the list of votes for the households, available for all household members to view. Votes must be approved by a majority of the existing members in order to pass. The current version of this app allows for new member votes and karma change votes. Future releases will allow votes on electing a landlord and evicting existing members.
  + GET method that returns all the votes for a household. This is called by the client's "Voting" screen to disply all votes to the user. Returns the following for each vote:
    - Type of proposal (i.e. “Karma”, “New Landlord”, “New Member Approval”, “Evict Member”)
    - Status of the vote (ex. “2 members for, 1 against, 3 needed for majority” – 3 needed for a household of 5 for example for a karma vote)
    - Member the vote applies to (member if karma vote or landlord vote, user if a new member vote)
    - Member who proposed the vote (or anonymous)
    - Brief one-line message/description
    - List of member ids that voted for/against the vote
  + POST action that creates a new vote for a household.
    - Household ID
    - Type of proposal (i.e. “Karma”, “New Landlord”, or “Evict Member”)
    - Member to which it applies
    - If karma proposal, amount of karma to be added/deducted
    - Whether proposal should be made anonymously
    - Brief one-line message/description
  + POST action that applies a household member's vote to an existing proposal/vote. Each member is only given one vote per proposal and may not take back their vote. The controller will accept a vote containing:
    - The id of the vote
    - The vote, “For” or “Against”