Database Schemas

Collections:

- Users
    1. username: String - User's username.
    2. email: String - User's email address.
    3. emailVerified: Boolean - Indicates whether the user's email has been verified.
    4. photo: Buffer - Binary data representing the user's profile photo.
    5. uid: String - User ID.
    6. createdAt: Date - Date and time when the user account was created. It defaults to the current date and time if not specified.
    7. lastLoginAt: Date - Date and time of the user's last login.
    8. role: Object - User's role within the system, including:
        - Organizer: Object - Details specific to users with the organizer role, including:
        - Organizations: Array of Strings - Names of organizations the user is associated with as an organizer.
        - Student: Object - Details specific to users with the student role, including:
        - Organizations: Array of Strings - Names of organizations the user is associated with as a student.
        - Moderator: Object - Details specific to users with the moderator role, including:
        - AllOrganizationsAccess: Boolean - Indicates whether the moderator has access to all organizations.
        - The default values for each role are provided if not specified explicitly.
    9. source: String - Source of user registration (e.g., "Google", "Facebook").
    10. save: Array of Strings - IDs of events saved by the user.
    11. registered: Array of Strings - IDs of events the user is registered with.

- Organizations
    1. name: String - Name of organization
    2. members: Array of String: ID of user who are member of this organization
    3. events: Array of Object ID: Id of events created under this organiation,

- Events
    1. title: String - Title of the event.
    2. description: String - Description of the event.
    3. location: String - Location where the event will take place.
    4. beginDate: Date - Start date and time of the event.
    5. completeDate: Date - End date and time of the event.
    6. recurring: Object - Information about recurring events, including:
        - frequency: String - Frequency of recurrence (e.g., daily, weekly).

    - specificDay: String - Specific day of recurrence (if applicable).
7. images: Buffer - Binary data representing images related to the event.
8. contactInfo: Object - Contact information for the event, including:
    - email: String - Email address for contacting event organizers.
    - phoneNumber: String - Phone number for contacting event organizers.
    - discordHandle: String - Discord handle for contacting event organizers.
9. seats: Object - Information about available seats for the event, including:
    - totalSeat: Number - Total number of seats available.
    - seatsRemaining: Number - Number of seats remaining.
10. tags: Array of Strings - Tags associated with the event for categorization or search purposes.
11. eventRegistration: Object - Information about event registration, including:
    - internalExternalForm: Boolean - Indicates whether registration is handled internally or externally.
    - externalFormLink: String - Link to an external registration form (if applicable).
    - fields: Object - Fields required for registration, including:
    - fullName: Boolean - Indicates whether full name is required for registration.
    - email: Boolean - Indicates whether email is required for registration.
    - major: Boolean - Indicates whether major is required for registration.
    - classYear: Boolean - Indicates whether class year is required for registration.
12. track: Array of ObjectIds - References to users who are tracking or interested in the event.

GET Endpoints
All parameters for GET should be put into a query string (req.query). All other method parameters in body (req.body)

- /users
    - /users/dash
        - return the corresponding user info for current user: containing username, email, photo, uid, role, saved, registered
        - require to pass authentication
    -
- /organizations
    - /organizations
        - Get a list of organization names with their id
        - No parameters
    - /organizations/:id
        - Get info on an organization with id :id
        - No parameters
- /events
    - /events
        - Get a list of all events
        - Parameters:
            - sort_by (optional): get events by some order
                - Options: start_date, end_date, none (default)
            - filter_by (optional): filter events by a filter
                - Options: start_date, end_date, recurring, tags, none (default)
            - filter (required if filter_by != none) : filter used to filter events
                - Must be a valid date for start_date and end_date
                - Must be a valid recurrence for recurring (check schema for enum)
                - Must be a valid tag for tags
                - Ignore for none
    - /events/:eventid
        - Get info on an event with id :eventid
        - No parameters

- /dash
    - authenticate user and redirect to dashboard

POST Endpoints

- /users
    - /users/login
        - verify the session token and redirect user to the proper page

        - require idToken and csrfToken
    - /users/signin
        - create a new user on Mongo DB base on the credentials credentials given
        - require idToken
    - /users/signinWithPassword
        - create a new user on Mongo DB
        - require idToken, csrfToken, profileImage buffer

- /user
    - /user/sessionLogout
        - revoke the token and redirect user to login
        - require session cookie
    - /user/registerEvent
        - register user for the given event base on event id
        - require eventid
    - /user/unregisterEvent
        - unregister user from given event
        - require eventid
- /organizations
    - /organizations
        - Create new organization
            - Generate unique organization id (auto-increment?) if parameters are valid
            - Only admins should be allowed to create organizations
        - Parameters:
            - name (required): name of organization
                - Must be a name not already in the organization collection
            - members (required): list of members (usernames) in organization
                - Must have at least one member and all members must be valid usernames
                    - On creation, all members should be given the "Organizer" role
            - api_token (required): api token of user requesting this
                - Must be an admin token
- events
    - events
        - Create new event
        - require title, description, location, beginDate (Date object), completeDate (date object), recurring, seats, tags, eventRegistration, eventImage (image buffer)

PUT Endpoints
Not making mass updates because I think that sounds like a terrible idea

- /users
    - /users/:username
        - Update user info
            - Only info that should be modifiable is email, password, role, and organization
            - Only admins should be able to change role and organization
            - Don't allow access if API token isn't an admin or username's token
        - Parameters: (continue from here, should be similar to POST /users)
            - email (optional)
                - Must be a valid @rpi.edu email and cannot already be in collection
            - password (optional): new plaintext password that user wants
                - Must not be an empty string
            - role (optional): new role of user
                - Only admins can change role of a user
            - organization (optional)
                - Only admins can add/remove organization of a user
            - events_registered (optional): new list of events to register for
                - Will need to update all old events to remove this user and then new events to add this user to registered list
            - api_token (required): API token of user making the request

- /user/
    - /user/updateUsernames
        - update username
        - require NewUserName
    - users/updateProfileImage
        - update profile image
        - require NewPhoto

- /organizations
    - /organization/:id
        - Update organization info
            - Only admins can access this endpoint
        - Parameters:
            - name (optional): new name of organization
            - members (optional): new list of members (usernames)
                - When doing this, remove organization from all members' organization list and organizer role from all members.

                Then add all new members and assign them this organization + organizer role
            - All usernames must be valid. Ignore duplicates
        - api_token (required): API token of user making request
- /events
    - /events/:id
        - Update event info
            - Only event organizers and admins can do this
        - Parameters:
            - title (optional)
            - description (optional)
            - start_date (optional)
            - end_date (optional)
            - recurring (optional)
            - images (optional): new list of images
            - contact_type (optional)
            - contact_info (optional)
            - unlimited_seats (optional)
            - total_seats (optional)
            - users_registered (optional): new list of users registered for event
                - Need to update old users to not be registered for this and then new users to be registered for this
                - All usernames must be valid
            - tags (optional): new list of tags
            - internal_form (optional)
            - form_link (required if internal_form = true)
            - organization (optional): new list of organizations hosting event
                - Only admins can change this
            - api_token (required): api token of user making request

DELETE Endpoints
Not making mass deletes because again, it's a bad idea

- /users
    - /users/:username
        - Delete user with username :username
            - Any admin has access
        - Parameters:
            - api_token (required)
                - If api token doesn't match :username's api token and isn't an admin token, deny access
- /organizations
    - /organizations/:id
        - Delete organization with id :id
            - Only admins can do this (organizer makes request and then admin makes sure people in organization want to delete it before deleting)
            - Remove organization from all events (if event has no organization delete it)
            - Remove organization from all users (if user has no organization and isn't admin, update role to Regular)
        - Parameters:
            - api_token (required)
- /events
    - /events/:id
        - Delete event with id :id
            - Any admin has access
            - Remove event from user's registered events list and organization's hosted events list
        - Parameters:
            - api_token (required)