* Identifies all service endpoints required for your MVP ie. what are the URLs that your client will use to access the backend of your application.
* Describes the purpose of each endpoint based on the goals of a user persona as it relates *to their data* (i.e., not their intentions with respect the feedback they want, but how what they want to do will change their data).
* Provides example requests and responses, including erroneous requests (e.g., error message responses for when bad data is input).
* Diagrams how communication will go from your user interface pages to service endpoints.

# Users

4 endpoints are required for management of the users collection. Two of them manage the user login features using POST requests. The other two manage the built-in character sheet associated with the logged in user. These methods include PUT and GET requests. Note there is no API endpoint required to manage user logout because the user context is stored in memory rather than in the database.

## /api/users/register

Method: POST

Purpose: Endpoint is responsible for managing user registration. This takes the registration form data and creates a new user with name, email, username, and password.

Request/Response

Request…

Response: 201 (created), 409 (conflict user already exists), 400 (bad request for malformed data), 500 (internal server error)

Example workflow:

A diagram of a api

AI-generated content may be incorrect.

## /api/users/login

Method: POST

Purpose: Endpoint is responsible for managing user login. This takes the login form data and sends it out for authentication such that the server can confirm credentials and generate the user context for the rest of the application.

Request/Response

Request

Respons: 200 (successful login), 401 (invalid credentials), 500 (internal server error)

Example workflow

A diagram of a api

AI-generated content may be incorrect.

## /api/users/{id}/character/form

Method: PUT

Purpose: This endpoint is responsible for creation and updates to the player character data that is stored. No POST exists as the initial creation of the document is handled when the user account is created. Any subsequent creation or update of a character comes from the submission of the form.

Request/Response

Request

Response: 204(no content: success), 401(when user not logged in), 403(when user is logged in but somehow makes a request to an id that is not theirs), 415(unsupported media type: this will just be to restrict the files to be uploaded for csy reasons), 500 (server error)

Example workflow

A diagram of a diagram

AI-generated content may be incorrect.

Method: GET

Purpose: This endpoint is responsible for getting the character data to prepopulate the character form whenever a user makes an update to their character.

Request/Response

Request

Response: 200(ok), 401(when user not logged in), 403(when user is logged in but somehow makes a request to an id that is not theirs), 500 (server error)

Example workflow

A diagram of a diagram

AI-generated content may be incorrect.

## /api/users/{id}/character

Method: GET

Purpose: This endpoint is responsible for the display of the character sheet data in a viewable format for the player. This is required not to just display data from the database, but to calculate modifiers and display relevant output.

Request/Response

Request

Response: 200(ok), 403(when user is logged in but somehow makes a request to an id that is not theirs), 404 (character not found will happen if they are not logged in), 422(un-processable entity: this is just the return if we can’t do all the business logic), 500 (server error)

Example workflow

A diagram of a diagram

AI-generated content may be incorrect.

# Adventure Logs

3 endpoints are required for management of the adventure log collection. One endpoint creates the documents with a POST method. One endpoint updates/deletes documents using PUT/DELETE methods. One endpoint is used to display the appropriate data for the various log pages using a GET method.

## /api/adventurelog

Method: POST

Purpose: This endpoint is responsible for the creation of new documents that will be held in the adventure log collection. The data will come from the form filled out on the adventure log page.

Request/Response

Example workflow

A diagram of a api

AI-generated content may be incorrect.

## /api/adventurelog/{id}

Method: PUT

Purpose: This endpoint is responsible for sending updates to the log entry. The update is made directly from the location that the entry is displayed in, so the user can only edit the title and details once the initial entry is made.

Request/Response

Example workflow

A diagram of a api

AI-generated content may be incorrect.

Method: DELETE

Purpose: This endpoint is responsible for deleting the selected log entry from the collection. The endpoint updates the database, but is not responsible for making a new request to update the displayed page for the user.

Request/Response

Example workflow

A diagram of a api

AI-generated content may be incorrect.

## /api/adventurelog?type={type}

Method: GET

Purpose: This endpoint is responsible for getting the proper log entries to display based on the selected type. The filter by type allows the single endpoint to manage the four different log entry types.

Request/Response

Example workflow

A diagram of a diamond with arrows

AI-generated content may be incorrect.

# Random Monsters

1 endpoint is required for management of the random monsters collection. The endpoint has one GET method for encounter generation.

## /api/monsters?size={size}level={level}difficulty={difficulty}

Method: GET

Purpose: This endpoint is responsible for creating a unique encounter based on three inputs: party size, party level, and difficulty. The three inputs determine a range of XP. The API selects a random amount of random monsters from the collection. It then determines the XP of that encounter based on the [rules](https://www.dndbeyond.com/sources/dnd/basic-rules-2014/building-combat-encounters?srsltid=AfmBOopWCtQZu8TLQ4TSJ1KUhkVMpha-O0vqFtk_r2g-RCMu1f9xGeQz). It checks whether the encounter is within the XP range. If it is not, a new set of monsters is selected. Once a balanced encounter is found, the results can be sent back to the front end.

Request/Response

Example workflow

A diagram of a computer application

AI-generated content may be incorrect.

# Random NPCs

1 endpoint is required for management of the random NPCs collection. The endpoint has one GET method for NPC generation.

## /api/npcs

Method: GET

Purpose: This endpoint is responsible for retrieving a random NPC from the collection. Each document includes a first and last name as well as an image. The API will then select three random documents using the first name from the first, last name from the second, and image from the third to create a new random NPC. This new combination is then returned to be displayed.

Request/Response

Example workflow

A diagram of a graph

AI-generated content may be incorrect.