* Collections
  + Users (I think I will just include the character data)
    - Name - String
    - Email - String
    - Username -String
    - Password -String
    - Ability scores - Integers
      * Strength
      * Constitution
      * Dexterity
      * Intelligence
      * Wisdom
      * Charisma
    - Skill Proficiencies – List[String]
    - Level – Integer
    - Saving Throws – List[String]
    - Armor – String
    - Unarmored defense – Boolean
    - Shield – Boolean
    - Hit Points – Integer
    - Character Name – String
    - Image Path – String
  + Adventure Log
    - Type – List[String]
    - Description – Text
    - Title – String
    - Date - Date
  + Random NPCs
    - Name – String
    - Image – String (or picture if I can?)
  + Random Monsters
    - Name – String
    - Image – String (or picture?)
    - XP – Integer
    - Ref stat block – String

# Database Rationale

The chosen database for the application is MongoDB Atlas. MongoDB Atlas is a cloud-hosted, document-oriented NoSQL database that aligns well with the technical requirements and structure of the D&D Toolkit application. Below is the technical rationale for the decision.

## Schema Flexibility

The document database allows for a less rigid structure. This is helpful for the application since the various tables will include data that frequently has missing values and nested structures. The nested structures are especially useful as they allow the storage of more complex data in an easy to use and read fashion. For example the ability scores can be stored as one attribute that has multiple key-value pairs for Strength, Dexterity, etc.

## Non-Relational Data

The data being used for the application is generally not related. While it can be useful to join data such as a user with their character sheet or include what user entered a specific log, this does not require intense interaction between tables in the database. For the MVP, the user will only be allowed to have one character at a time, so the character data will be directly stored with the user collection.

## Cost Effective

MongoDB Atlas is cloud-native and fully managed. There is no need to set up a server or handle the hosting details. This helps reduce development time and lower overhead costs. The platform also has a generous free tier which should be more than sufficient for the MVP. It is also scalable if more data requirement become necessary.

# Database Structure

The database for the application is made up of 4 collections. Two are used to persist data entered by users, while the other two are used to store data pertaining to random generation.