***Distributed Computing Report – Ali Khalil [17734503]***

1. To use the web application run the project in the IDE. The browser should open. Navigate to the [DnD/main\_dnd.html](http://localhost:60378/DnD/main_dnd.html) page from the root address. To create a character choose “Create Character”. To view all previously created characters choose “View All Characters”. To search for a character by their name, enter the term in the search bar and click “View Character”.

When creating a character make sure to abide by the limitations of each field. Click “Submit” when done.

When editing a character abide by the limitations. “Click” submit when done.

When deleting a character you will be prompted for confirmation. Click “Yes” to delete and “No” not to.

When viewing a character you cant edit the fields. You have three buttons on the right that allow you to edit, delete or download an XML file of the character, respectively.

When viewing all characters click the “View” text to view the selected character.

1. Successfully completed:

* Create Character
* View Character List
* View Character
* Edit Character
* Delete Character
* Download Character

Limitations:

* Racial bonuses are not added to each of the ability scores in the database. The race, however, is so viewing the character will display the racial bonuses.
* Html encoding is performed on the whole JSON string rather than just the elements within the JSON. There was some difficulty decoding this returned string in the client. To get around this problem the encoded string was placed into an invisible text-area which decoded the string. There was an attempt to try and iterate through the JSON object and encode each element but that resulted in a similar issue.

1. **Get All Races**

* URL: DnD/DnDAssignment/Races
* Method: GET
* URL Params: None
* Data Params: None
* Success Response:
  + Code: 200
    - Content: {“count”:9,“results”:[{"name":"Dwarf","url":"http://www.dnd5eapi.co/api/races/1"], …}
* Error Response:
  + Code: 500
    - Content: “An error occurred communicating with the DnD5e Api” OR “An error occurred on the server”
* Sample Call:
  + xmlreq.open('GET', 'DnDAssignment/Races/', true);
  + xmlreq.onreadystatechange = some\_func();
  + xmlreq.send();

**Get All Classes**

* URL: DnD/DnDAssignment/Classes
* Method: GET
* URL Params: None
* Data Params: None
* Success Response:
  + Code: 200
    - Content: {“count”:9,“results”:[{"name":"Barbarian","url":"http://www.dnd5eapi.co/api/classes/1"], …}
* Error Response:
  + Code: 500
    - Content: “An error occurred communicating with the DnD5e Api” OR “An error occurred on the server”
* Sample Call:
  + xmlreq.open('GET', 'DnDAssignment/Classes/', true);
  + xmlreq.onreadystatechange = some\_func();
  + xmlreq.send();

**Get A Race**

* URL: DnD/DnDAssignment/Races/{id}
* Method: GET
* URL Params: id=[integer]
* Data Params: None
* Success Response:
  + Code: 200
    - Content (using id=1 as an example): {"\_id":"5bce91525b7768e792018382","index":1,"name":"Dwarf","speed":30,"ability\_bonuses":[0,0,2,0,0,0],"alignment":"Most dwarves are lawful, believing firmly in the benefits of a well-ordered society. They tend toward good as well, with a strong sense of fair play and a belief that everyone deserves to share in the benefits of a just order.","age":"Dwarves mature at the same rate as humans, but they’re considered young until they reach the age of 50. On average, they live about 350 years.","size":"Medium","size\_description":"Dwarves stand between 4 and 5 feet tall and average about 150 pounds. Your size is Medium.","starting\_proficiencies":[{"name":"Battleaxes","url":"http://www.dnd5eapi.co/api/proficiencies/20"},{"name":"Handaxes","url":"http://www.dnd5eapi.co/api/proficiencies/24"},{"name":"Light hammers","url":"http://www.dnd5eapi.co/api/proficiencies/26"},{"name":"Warhammers","url":"http://www.dnd5eapi.co/api/proficiencies/51"}],"starting\_proficiency\_options":{"choose":1,"type":"proficiencies","from":[{"name":"Smith's tools","url":"http://www.dnd5eapi.co/api/proficiencies/71"},{"name":"Brewer's supplies","url":"http://www.dnd5eapi.co/api/proficiencies/59"},{"name":"Mason's tools","url":"http://www.dnd5eapi.co/api/proficiencies/68"}]},"languages":[{"name":"Common","url":"http://www.dnd5eapi.co/api/languages/1"},{"name":"Dwarvish","url":"http://www.dnd5eapi.co/api/languages/2"}],"language\_desc":"You can speak, read, and write Common and Dwarvish. Dwarvish is full of hard consonants and guttural sounds, and those characteristics spill over into whatever other language a dwarf might speak.","traits":[{"url":"http://www.dnd5eapi.co/api/traits/1","name":"Darkvision (Dwarf)"},{"url":"http://www.dnd5eapi.co/api/traits/2","name":"Dwarven Resilience"},{"url":"http://www.dnd5eapi.co/api/traits/3","name":"Stonecunning"}],"subraces":[{"name":"Hill Dwarf","url":"http://www.dnd5eapi.co/api/subraces/1"},{"name":"Mountain Dwarf","url":"http://www.dnd5eapi.co/api/subraces/4"}],"url":"http://www.dnd5eapi.co/api/races/1"}
* Error Response:
  + Code: 500
    - Content: “An error occurred communicating with the DnD5e Api” OR “An error occurred on the server”
* Sample Call:
  + xmlreq.open('GET', 'DnDAssignment/Classes/1', true);
  + xmlreq.onreadystatechange = some\_func();
  + xmlreq.send();

**Get A Class**

* URL: DnD/DnDAssignment/Classes/{id}
* Method: GET
* URL Params: id=[integer]
* Data Params: None
* Success Response:
  + Code: 200
    - Content (using id=1 as an example): {"\_id":"5bce90ff5b7768e792017cf3","index":1,"name":"Barbarian","hit\_die":12,"proficiency\_choices":[{"from":[{"name":"Skill: Animal Handling","url":"http://www.dnd5eapi.co/api/proficiencies/106"},{"name":"Skill: Athletics","url":"http://www.dnd5eapi.co/api/proficiencies/108"},{"name":"Skill: Intimidation","url":"http://www.dnd5eapi.co/api/proficiencies/112"},{"name":"Skill: Nature","url":"http://www.dnd5eapi.co/api/proficiencies/115"},{"name":"Skill: Perception","url":"http://www.dnd5eapi.co/api/proficiencies/116"},{"name":"Skill: Survival","url":"http://www.dnd5eapi.co/api/proficiencies/122"}],"type":"proficiencies","choose":2}],"proficiencies":[{"url":"http://www.dnd5eapi.co/api/proficiencies/1","name":"Light armor"},{"url":"http://www.dnd5eapi.co/api/proficiencies/2","name":"Medium armor"},{"url":"http://www.dnd5eapi.co/api/proficiencies/18","name":"Shields"},{"url":"http://www.dnd5eapi.co/api/proficiencies/19","name":"Simple weapons"},{"url":"http://www.dnd5eapi.co/api/proficiencies/20","name":"Martial weapons"}],"saving\_throws":[{"url":"http://www.dnd5eapi.co/api/ability-scores/1","name":"STR"},{"url":"http://www.dnd5eapi.co/api/ability-scores/3","name":"CON"}],"starting\_equipment":{"url":"http://www.dnd5eapi.co/api/startingequipment/1","class":"Barbarian"},"class\_levels":{"url":"http://www.dnd5eapi.co/api/classes/Barbarian/levels","class":"Barbarian"},"subclasses":[{"name":"Berserker","url":"http://www.dnd5eapi.co/api/subclasses/1"}],"url":"http://www.dnd5eapi.co/api/classes/1"}
* Error Response:
  + Code: 500
    - Content: “An error occurred communicating with the DnD5e Api” OR “An error occurred on the server”
* Sample Call:
  + xmlreq.open('GET', 'DnDAssignment/Classes/1', true);
  + xmlreq.onreadystatechange = some\_func();
  + xmlreq.send();

**Get All Ability Types**

* URL: DnD/DnDAssignment/AbilityTypes
* Method: GET
* URL Params: None
* Data Params: None
* Success Response:
  + Code: 200
    - Content: {"count":6,"results":[{"name":"STR","url":"http://www.dnd5eapi.co/api/ability-scores/1"},{"name":"DEX","url":"http://www.dnd5eapi.co/api/ability-scores/2"},{"name":"CON","url":"http://www.dnd5eapi.co/api/ability-scores/3"},{"name":"INT","url":"http://www.dnd5eapi.co/api/ability-scores/4"},{"name":"WIS","url":"http://www.dnd5eapi.co/api/ability-scores/5"},{"name":"CHA","url":"http://www.dnd5eapi.co/api/ability-scores/6"}]}
* Error Response:
  + Code: 500
    - Content: “An error occurred communicating with the DnD5e Api” OR “An error occurred on the server”
* Sample Call:
  + xmlreq.open('GET', 'DnDAssignment/AbilityTypes/', true);
  + xmlreq.onreadystatechange = some\_func();
  + xmlreq.send();

**Add A Character**

* URL: DnD/DnDAssignment/Characters
* Method: POST
* URL Params: None
* Data Params: "name": "Name",
* "age": "12",
* "gender": "Male",
* "biography": "Bio",
* "level”: "12",
* "race": "Human",
* "class": "Barbarian",
* "spellcaster": "1",
* "hit\_points": "43",
* "ab\_str": "0",
* "ab\_con": "0",
* "ab\_dex": "0",
* "ab\_cha": "0",
* "ab\_int": "0",
* "ab\_wis": "0"
* Success Response:
  + Code: 204
    - Content: None
* Error Response:
  + Code: 500
    - Content: “An error occurred communicating with the DnD5e Api” OR “An error occurred on the server”
  + Code 400
    - Content: “Age must not be empty\nGender must not be empty…”
* Sample Call:
  + post\_charac\_xmlreq.open('POST', 'DnDAssignment/Characters', true);
  + post\_charac\_xmlreq.onreadystatechange = some\_func();
  + post\_charac\_xmlreq.send();

**Update A Character**

* URL: DnD/DnDAssignment/Characters/{id}
* Method: PUT
* URL Params: id=[integer]
* Data Params: “id”: ”2”,
* "name": "Name",
* "age": "12",
* "gender": "Male",
* "biography": "Bio",
* "level”: "12",
* "race": "Human",
* "class": "Barbarian",
* "spellcaster": "1",
* "hit\_points": "43",
* "ab\_str": "0",
* "ab\_con": "0",
* "ab\_dex": "0",
* "ab\_cha": "0",
* "ab\_int": "0",
* "ab\_wis": "0"
* Success Response:
  + Code: 204
    - Content: None
* Error Response:
  + Code: 500
    - Content: “An error occurred communicating with the DnD5e Api” OR “An error occurred on the server”
  + Code 400
    - Content: “Age must not be empty\nGender must not be empty…”
* Sample Call:
  + post\_charac\_xmlreq.open('POST', 'DnDAssignment/Characters/' + parseInt(char\_id), true);
  + post\_charac\_xmlreq.onreadystatechange = some\_func();
  + post\_charac\_xmlreq.send();

**Get All Characters**

* URL: DnD/DnDAssignment/Characters
* Method: GET
* URL Params: None
* Data Params: None
* Success Response:
  + Code: 200
    - Content: “[{"id":"2","name":"name","age":"8","gender":"Male","biography":"Bio","level":"10","race":"Elf","class":"Barbarian","spellcaster":"1","hit\_points":"121","ab\_con":"1","ab\_dex":"0","ab\_str":"8","ab\_cha":"6","ab\_int":"0","ab\_wis":"4"},{"id":"4","name":"Ganondorf","age":"96","gender":"Male","biography":"Ganondorf, known as the King of Thieves, the Great King of Evil, the Emperor of the Dark Realm, or the Dark Lord, is the primary antagonist of the majority of games in the Legend of Zelda series. Ganondorf was born a member of the Gerudo; as the only male member to be born in a hundred years, Ganondorf is made either the","level":"9","race":"Human","class":"Sorcerer","spellcaster":"1","hit\_points":"54","ab\_con":"0","ab\_dex":"0","ab\_str":"10","ab\_cha":"0","ab\_int":"10","ab\_wis":"0"}]”
* Error Response:
  + Code: 500
    - Content: “An error occurred communicating with the DnD5e Api” OR “An error occurred on the server”
* Sample Call:
  + xmlreq.open('GET', 'DnDAssignment/Characters/', true);
  + xmlreq.onreadystatechange = some\_func();
  + xmlreq.send();

**Get A Character**

* URL: DnD/DnDAssignment/Characters/{id}
* Method: GET
* URL Params: id=[Integer]
* Data Params: None
* Success Response:
  + Code: 200
    - Content: "{"id":"1","name":"Dave","age":"12","gender":"Male","biography":"Bio","level":"12","race":"Human","class":"Barbarian","spellcaster":"0","hit\_points":"15","ab\_con":"1","ab\_dex":"1","ab\_str":"1","ab\_cha":"1","ab\_int":"1","ab\_wis":"1"}"
* Error Response:
  + Code: 500
    - Content: “An error occurred communicating with the DnD5e Api” OR “An error occurred on the server”
* Sample Call:
  + xmlreq.open('GET', 'DnDAssignment/Characters/' + parseInt(id), true);
  + xmlreq.onreadystatechange = some\_func();
  + xmlreq.send();

**Get A Character**

* URL: DnD/DnDAssignment/Characters/Name/
* Method: POST
* URL Params: None
* Data Params: "name": "Name",
* Success Response:
  + Code: 200
    - Content: "{"id":"1","name":"Dave","age":"12","gender":"Male","biography":"Bio","level":"12","race":"Human","class":"Barbarian","spellcaster":"0","hit\_points":"15","ab\_con":"1","ab\_dex":"1","ab\_str":"1","ab\_cha":"1","ab\_int":"1","ab\_wis":"1"}"
* Error Response:
  + Code: 500
    - Content: “An error occurred communicating with the DnD5e Api” OR “An error occurred on the server”
* Sample Call:
  + post\_charac\_xmlreq.open('POST', 'DnDAssignment/Characters/Name’, true);
  + post\_charac\_xmlreq.onreadystatechange = some\_func();
  + post\_charac\_xmlreq.send();

**Delete A Character**

* URL: DnD/DnDAssignment/Characters/{id}
* Method: DELETE
* URL Params: id=[integer]
* Data Params: None
* Success Response:
  + Code: 204
    - Content: None
* Error Response:
  + Code: 500
    - Content: “An error occurred communicating with the DnD5e Api” OR “An error occurred on the server”
* Sample Call:
  + post\_charac\_xmlreq.open(‘DELETE’, 'DnDAssignment/Characters/' + parseInt(char\_id), true);
  + post\_charac\_xmlreq.onreadystatechange = some\_func();
  + post\_charac\_xmlreq.send();

**Create Character XML**

* URL: DnD/DnDAssignment/Characters/XML/{id}
* Method: GET
* URL Params: id=[integer]
* Data Params: None
* Success Response:
  + Code: 200
    - Content:

<?xml version="1.0" encoding="utf-8"?><Character><ID>6</ID><Name>Government of Western Australia Department of Transport</Name><Age>499</Age><Gender>Government Body</Gender><Biography>This account will become a TAX INVOICE for GST purposes when a valid receipt is attached.</Biography><Level>19</Level><Race>Human</Race><Class>Barbarian</Class><Spellcaster>No</Spellcaster><Hit\_Points>228</Hit\_Points><Constitution\_Ability>0</Constitution\_Ability><Dexterity\_Ability>0</Dexterity\_Ability><Strength\_Ability>0</Strength\_Ability><Charisma\_Ability>0</Charisma\_Ability><Intelligence\_Ability>0</Intelligence\_Ability><Wisdom\_Ability>0</Wisdom\_Ability></Character>

* Error Response:
  + Code: 500
    - Content: “An error occurred communicating with the DnD5e Api” OR “An error occurred on the server” OR “An error occurred while creating XML file” OR “An error occurred retrieving character from database"
* Sample Call:
  + post\_charac\_xmlreq.open(‘GET, 'DnDAssignment/Characters/' + parseInt(char\_id), true);
  + post\_charac\_xmlreq.onreadystatechange = some\_func();
  + post\_charac\_xmlreq.send();

1. **Sequence Diagrams start in the next page onwards**

A picture containing screenshot

Description automatically generated

A screenshot of a cell phone

Description automatically generated

A screenshot of a social media post

Description automatically generated

**Design Essay**

The web application consists of a single webpage that, on several different events, will retrieve or input data to and from a server. This server, another component of the web application, communicates with both a 3rd party API (DnD5eAPI) and an SQLite database. The central theme of the application’s traffic design is to gather information when needed in order to spread out the instances of requests made to the server, striking a balance between frequency and load.

Traffic is generated during events that require communication with the server. These include retrieving all characters, races, and classes, retrieving detailed information about the aforementioned items, input into the database.

Requests to the server are made first and foremost when the application launches on the browser and the page is loaded. These requests include getting all characters, races and classes, and these requests are launched back-to-back. This was required into order to allow the user to seamlessly transition from one subsection of the application to another, such as moving from the list of characters to the “Create a Character” menu.

An alternative to this approach would be to load the necessary for each subsection of the web application the user is in. For example, getting all the characters may only be done once the client has clicked on the “View All Characters” button. One of the benefits of this approach is that it may decrease traffic if the user does not hop from one subsection of the web application to another in quick succession as the frequency of requests is limited to the interactivity of the user. However, if the user *does* move between subsections of the web application frequently then this would have the opposite effect. Each click would necessitate the application to make a request as there may be updated information in the database input by another client.

Within the “Create a Character” menu there is a list of races and classes that were initially retrieved on load of the web page. On selection of a class or race a request is made to acquire more information about the race/class selected. This was designed so that the initial launch of the web page, and the traffic to and from the server, is not slowed down by having to load all the individual race and class information onto top loading in the initial list of classes/races; the list of classes and races has the capacity to grow and the initial retrieval was enough to acquire a basic list. As such this design strikes a middle ground between being updated and traffic congestion, particularly so if multiple clients are utilizing the web application. The alternative is to load all the classes and races at once and store them in the webpage but, as mentioned earlier, would place extra load on the server which may be too much if there a multiple clients using the application.

On submitting a character there is some client-side validation being done to determine if there are faults with the input information. While client-side validation can be bypassed it can serve to stop requests to the server where it would be stopped but at the cost of traffic load.

On viewing and editing a character the application will reload the list of races and classes to remain updated regarding any list changes. This could have been mitigated by retaining the current list of races and classes but this may result in miscommunication with the DnD5eAPI when it comes to deleted and/or updated information.

The XML file creation was done on the server side to have the XML only contain the latest valid update of a particular character. This has the downside of imposing more load on the server but this is offset by the fact that only one character can have an XML file made using it’s information a time. The alternative would be to perform the XML file creation client-side which would prevent the additional load to the server. This comes at the cost of having the information the XML is based on open to alteration by users.

As mentioned earlier the web application is a single webpage divided into a couple of subsections. Ultimately however moving from one section of the webpage to another does not require the loss of any state information as there are no page loading outside of the initial load. This presents the benefit of minimizing the need for requests in some circumstances like transitioning between viewing, editing, and creating a character. In this case the list of races/classes is retained within the HTML of the page. Given this however the information retained can be easily edited by users opening the processing of the page to some parsing or calculation issues. Alternatively the different functions of the application could have been different pages which would lessen the need for hidden fields and HTML data retention but this would have increased the frequency of requests to the server.

The end points of the application (using the titles given in section 3 of this report) are there to be invoked by the requests of the client. They are:

* Get All Races
* Get All Classes
* Get All Ability Types
* Get A Race
* Get A Class
* Add A Character
* Update A Character
* Get All Characters
* Get A Character
* Get A Character
* Delete A Character
* Create Character XML

The first two retrieve a list of all races, classes and ability types from the DnD5eAPI. This was necessary in order to acquire a list of races, classes and ability types to interact with. Alternatively they could have been the same method call which would eliminate the need for three separate calls. In hindsight this would have been a better idea as every time the request to acquire all classes is made so is the one to acquire all abilities and races.

The next two retrieve a specific class or race from the DnD5eAPI. This was necessary as unlike the above situation they are called at different times and serve to acquire more information about the selected race/class.

The next one (Add a character) was necessary to place a character in the database. It operates by first validating the received request body and then, if valid, storing it in the database. After careful consideration there was no viable alternative to having this function.