Category (ex. Login, Home Page, Organization Data)

Request type (ex. Get, Post, Delete…)

Description – (just write a description of what the endpoint does)

Endpoint – (what information are we getting passed from the UI)

Query – (what you need to get from the database and the tables they come from)

Return – (what information you are returning to the UI in JSON)

**Child Account**

/childaccount/post

**Description –** The Post Child Account endpoint takes information for a new child account page and creates in on the database for later use.

**createQuery(organization id, permission list, email) –** The server will call the INSERT INTO function of SQL to add the passed organization id and the email to the User table on the database. Another SQL INSERT INTO function will add the permissions list to the User Type Permissions table.

**sendConfirmation() –** A confirmation should be sent back to the UI when the endpoint is complete.

/childaccount/get

**Description –** The Get Child Account endpoint will find all child accounts on the database that are correlated with a given organization and then return them to the UI in JSON formatting.

**createQuery(organization id) –** The server will send an SQL string to the database to SELECT user\_name, user\_type\_id, and email from the User table where their organization\_id equals the organization id passed by the UI. Then another SELECT should get the permission list from the User Type Permissions table where user\_type\_id is equal to the user type id previously selected. Then another SELECT will be used to get the city\_name from the City table where the organization\_id is equal to the organization id passed by the UI.

**toJson() –** A JSON string should be returned representing multiple objects for child accounts by using an array that includes the following. [city\_name, account\_name, permission\_list, email]

/childaccount/put

**Description –** The Put Child Account endpoint takes information for a child account that needs to modify and updates the information on the database for later use.

**createQuery(organization id, account id, permission list, email) –** The server will call the MODIFY function of SQL to update the passed organization id and the email to the User table on the database where the user\_id equals the passed account id.. Another MODIFY function will update the permissions list to the User Type Permissions table.

**sendConfirmation() –** A confirmation should be sent back to the UI when the endpoint is complete.

/childaccount/delete

**Description –** The Delete Child Account endpoint will delete a given child account.

**createQuery(account id) –** The server will call the DELETE function to delete a user from the User table where the user\_id equals the account id passed by the UI

**sendConfirmation() –** A confirmation should be sent back to the UI when the endpoint is complete.

**Organization Data**

/citydata/post

**Description –** The city data sent from UI will be sent to the database. First we will check if the county\_name, city name, and state\_name does exist in City table. If not we will add; if does we will query the city\_id and insert with organization data in Organization table.

**createQuery(organization id, organization name, city name, county name) –** Check (select city\_id from City table where state\_name = ‘state\_name’ and .county\_name = ‘county\_name’ and city\_name = ‘city\_name’).

Insert into Organization Table (organization\_id, city\_id, organization\_name) Values (organization\_id, city\_id ,‘organization\_name)

**sendConfirmation() –** A confirmation should be sent back to the UI when the endpoint is complete.

/organizationData/get

**Description -** The get organization data endpoint will find all data on the database that are correlated with the Organization ID passed from the UI and then return them to the UI in JSON formatting.

**createQuery(organization id) -** The Server will pass a string to the Database to SELECT all data FROM the organization table WHERE the organization id = the organization id passed by the UI.

**toJSON() -** A JSON string should be returned representing all objects for organization Data.

/organizationData/paygrade/put

**Description -** edit and update paygrade info from UI to database

**createQuery(organization\_id, pay\_grade\_size, pay\_grade\_steps, user\_name) -**

UPDATE Pay Grade Scale

SET num\_pay\_grades = “pay\_grade\_size”

pay\_grade\_steps = “pay\_grade\_steps”

last\_updated\_by = SELECT user\_id FROM User

WHERE user\_name = “user\_name”

last\_updated\_date = Date()

WHERE organization\_id = “organization\_id”

**sendConfirmation() –** A confirmation should be sent back to the UI when the endpoint is complete.

/organizationData/paygrademinmax/put

**Description -** edit and update paygrade\_min\_max info from UI to database

**createQuery(organization\_id, grade\_level, paygrade\_max, paygrade\_min, user\_name) -**

UPDATE Grade

SET grade\_letter = “grade\_level”

pay\_grade\_max = “paygrade\_max”

pay\_grade\_min = “paygrade\_min”

last\_updated\_by = SELECT user\_id FROM User

WHERE user\_name = “user\_name”

last\_updated\_date = Date()

WHERE organization\_id = “organization\_id”

**sendConfirmation() –** A confirmation should be sent back to the UI when the endpoint is complete.

Post create Job (organization-id, department, job-title, description, paygrade, wage, worker\_Name)

**Position**

/position/put

**Description -** Updates a position by changing one of the values stored in the column of the job

**createQuery(organization-id, department, position\_id, position-title, description, paygrade\_level, current\_wage, worker\_Name) -** The server will call the MODIFY function of SQL to update one or more of the values to the position\_id passed from the UI.

**Return -** A confirmation should be sent back to the UI when the endpoint is complete.

/position/post

**Description -** Creates a new job in the database with the values sent from the user.

**createQuery(organization id, department, job-title, description, paygrade level, current wage, worker name) -** The server will call the INSERT INTO function of SQL followed by a list of the columns in the job title table and the values that were passed in that correlate to those columns

**sendConfirmation() –** A confirmation should be sent back to the UI when the endpoint is complete.