4. The remaining paths will be appended to the above, e.g.:
   `localhost:8080/`**`soap`**`/resources/CompanyServices/`

Endpoints

1. **Path**: `/company`
   Verb: DELETE
   Produces: application/json
   a. Deletes all Department, Employee and Timecard records in the database for the given company. You will need to pay attention to the Foreign Key Constraints.
   b. Input is your RIT user ID as a String passed as QueryParam

   ```
   company=company+name
   ```

   where “company+name” is your RIT user ID

   c. Output:
      i. Success:

      ```
      {
         "success":"companyName's information deleted."
      }
      ```

      ii. Error:

      ```
      {
         "error":"An appropriate error message."
      }
      ```

2. **Path**: `/department`
   Verb: GET
   Produces: application/json
   a. Returns the requested Department as a JSON String.
   b. Input as QueryParams:

   ```
   company=company+name&dept_id= id
   ```

   where “company+name” is your RIT user ID and “dept_id” is the record id of the department to retrieve.

c. Output:

i. Success:

```
{
   "dept_id":1,
   "company":"rituserid",
   "dept_name":"accounting",
   "dept_no":"d10",
   "location":"new york"
}
```

ii. Error:

```
{
   "error":"An appropriate error message."
}
```

3. **Path**: `/departments`
Verb: GET
Produces: application/json

a. Returns the requested list of Departments.

b. Input as a String in a QueryParam.

```
company=company+name
```

where “company+name” is your RIT user ID

c. Output:

i. Success:

```
[
   {
      "dept_id":1,
      "company":"rituserid",
      "dept_name":"accounting",
      "dept_no":"d10",
      "location":"new york"
   },
   {
      "dept_id":2,
      "company":"rituserid",
      "dept_name":"research",
      "dept_no":"d20",
      "location":"dallas"
   },
   {
      "dept_id":3,
      "company":"rituserid",
      "dept_name":"sales",
      "dept_no":"d30",
      "location":"chicago"
   },
   ....
]
```

ii. Error:

```
{
   "error":"An appropriate error message."
}
```

4. **Path**: `/department`
   Verb: POST
   Consumes: application/json
   Produces: application/json

   a. Additional Validation:
      dept_no must be unique among all companies; Suggestion: include company name as part of id.

   b. Returns the updated Department as a JSON String.

   c. Input as JSON:

```
{
   "company":"rituserid",
   "dept_name":"IT",
   "dept_no":"d11",
   "location":"rochester"
}
```

where "company" is your RIT user ID.

d. Output:

i. Success:

```
{
   "success": {
      "dept_id":5,
      "company":"rituserid",
      "dept_name":"IT",
      "dept_no":"d11",
      "location":"rochester"
   }
}
```

ii. Error:

```
{
   "error":"An appropriate error message."
}
```

5. **Path**: `/department`
Verb: PUT
Produces: application/json

a. Additional Validation:

i. dept_no must be unique among all companies; Suggestion: include company name as part of id.

ii. dept_id must be an existing record number for a department

b. Returns the updated Department as a JSON String.

c. Input (any values you want to change plus the record id for the Department) as FormParam:

```
{
   "dept_id" = 1
   "company" = "rituserid"
   "dept_name" = "mystery"
   "dept_no" = "d10"
   "location" = "buffalo"
}
```

where “company” is your RIT user ID.

d. Output:

i. Success:

```
{
   "success": {
      "dept_id":1,
      "company":"rituserid",
      "dept_name":"mystery",
      "dept_no":"d10",
      "location":"buffalo"
   }
}
```

ii. Error:

```
{
   "error":"An appropriate error message."
}
```

6. **Path**: `/department`
   Verb: DELETE
   Produces: application/json
   a. Returns the ID of the department deleted.
   b. Input as QueryParam:

```
"company" = "company name"
"dept_id" = id
```

where “company name” is your RIT user ID and “id” is the record id of the department to delete.

c. Output:

i. Success:

```
{
   "success":"Department 5 from rituserid deleted."
}
```

ii. Error:

```
{
   "error":"An appropriate error message."
}
```

7. **Path**: `/employee`
   Verb: GET
   Produces: application/json
   a. Returns the requested Employee as a JSON String.

   b. Input as a QueryParam

   ```
   emp_id=#
   ```

   where "emp_id" is the employee ID

   c. Output:
      i. Success:

      ```
      {
         "emp_id":2,
         "emp_name":"jones",
         "emp_no":"e2",
         "hire_date":"1981-04-01",
         "job":"manager",
         "salary":2975.0,
         "dept_id":2,
         "mng_id":1
      }
      ```

      ii. Error:

      ```
      {
         "error":"An appropriate error message."
      }
      ```

8. **Path**: `/employees`
   Verb: GET
   Produces: application/json
   a. Returns the requested list of Employees.

   b. Input as a QueryParam.

   ```
   company=company+name
   ```

where “company+name” is your RIT user ID

c. Output:

i. Success:

```
[
   {
      "emp_id":1,
      "emp_name":"king",
      "emp_no":"e1",
      "hire_date":"1981-11-16",
      "job":"president",
      "salary":5000.0,
      "dept_id":1,
      "mng_id":0
   },
   {
      "emp_id":2,
      "emp_name":"jones",
      "emp_no":"e2",
      "hire_date":"1981-04-01",
      "job":"manager",
      "salary":2975.0,
      "dept_id":2,
      "mng_id":1
   },
   {
      "emp_id":3,
      "emp_name":"ford",
      "emp_no":"e3",
      "hire_date":"1981-12-02",
      "job":"analyst",
      "salary":3000.0,
      "dept_id":2,
      "mng_id":2
   },
   ....
]
```

ii. Error:

```
{
   "error":"An appropriate error message."
}
```

9. **Path**: `/employee`
   Verb: POST
   Consumes: application/json
   Produces: application/json
   a. Additional validations:
      i. dept_id must exist as a Department in your company
      ii. mng_id must be the record id of an existing Employee in your company.
      iii. hire_date must be a valid date equal to the current date or earlier (e.g. current date or in the past)
      iv. hire_date must be a Monday, Tuesday, Wednesday, Thursday or a Friday. It cannot be Saturday or Sunday.
      v. emp_no must be unique amongst all employees in the database, including those of other companies. You may wish to include your RIT user ID in the employee number somehow.

   b. Returns the new Employee as a JSON String.

   c. Input:

```
{
   "emp_name":"french",
   "emp_no":"rituserid-e1b",
   "hire_date":"2018-06-16",
   "job":"programmer",
   "salary":5000.0,
   "dept_id":1,
   "mng_id":2
}
```

   d. Output:
      i. Success:

```
{
   "success": {
      "emp_id":15,
      "emp_name":"french",
      "emp_no":"rituserid-e1b",
      "hire_date":"2018-06-16",
      "job":"programmer",
      "salary":5000.0,
      "dept_id":1,
      "mng_id":2
   }
}
```

ii. Error:

```
{
   "error":"An appropriate error message."
}
```

10. **Path**: `/employee`
    Verb: PUT
    Produces: application/json

    a. Additional validations same as inserting an Employee plus emp_id must be a valid record id in the database.

    b. Returns the updated Employee as a JSON String.

    c. Input (any values you want to change plus the record id for the Employee) as FormParams:

```
{
   "emp_id" = 15
   "emp_name" = "french"
   "emp_no" = "rituserid-e1b"
   "hire_date" = "2018-06-16"
   "job" = "programmer"
   "salary" = 6000.0
   "dept_id" = 1
   "mng_id" = 2
}
```

    d. Output:

    i. Success:

```
{
   "success": {
      "emp_id":15,
      "emp_name":"french",
      "emp_no":"rituserid-e1b",
      "hire_date":"2018-06-16",
      "job":"programmer",
      "salary":6000.0,
      "dept_id":1,
      "mng_id":2
   }
}
```

ii. Error:

```
{
   "error":"An appropriate error message."
}
```

11. Path: `/employee`
    Verb: DELETE
    Produces: application/json
    a. Returns that the employee was deleted.
    b. Input as a QueryParam.

    ```
    emp_id=#
    ```

    where emp_id is the record id of the Employee to delete
    c. Output:
       i. Success:

       ```
       {
          "success":"Employee 15 deleted."
       }
       ```

       ii. Error:

       ```
       {
          "error":"An appropriate error message."
       }
       ```

12. **Path**: `/timecard`
    Verb: GET
    Produces: application/json

a. Returns the requested Timecard as a JSON String.

b. Input as a QueryParam

```
timecard_id=#
```

where "timecard_id" the record id of the desired Timecard

c. Output:

i. Success:

```
{
   "timecard_id":1,
   "start_time":"2018-06-14 11:30:00",
   "end_time":"2018-06-14 15:30:00",
   "emp_id":2
}
```

ii. Error:

```
{
   "error":"An appropriate error message."
}
```

13. **Path**: `/timecards`
Verb: GET
Produces: application/json

a. Returns the requested list of Timecards.

b. Input as a QueryParam

```
emp_id=#
```

where "emp_id" is the record id of the employee you want to see the Timecards for

c. Output:

i. Success:

```
[
   {
      "timecard_id":3,
      "start_time":"2018-06-14 11:30:00",
      "end_time":"2018-06-14 15:30:00"
      "end_time":"2018-06-14 15:30:00",
      "emp_id":4
   },
   {
      "timecard_id":4,
      "start_time":"2018-06-13 11:30:00",
      "end_time":"2018-06-13 15:30:00",
      "emp_id":4
   },
   {
      "timecard_id":6,
      "start_time":"2018-06-12 11:30:00",
      "end_time":"2018-06-12 15:30:00",
      "emp_id":4
   }
]
```

ii. Error:

```
{
   "error":"An appropriate error message."
}
```

14. **Path**: `/timecard`
    Verb: POST
    Consumes: application/json
    Produces: application/json
    a. Additional validations:
        i. emp_id must exist as the record id of an Employee in your company.
        ii. start_time must be a valid date and time equal to the current date or up to 1 week ago from the current date.
        iii. end_time must be a valid date and time at least 1 hour greater than the start_time and be on the same day as the start_time.
        iv. start_time and end_time must be a Monday, Tuesday, Wednesday, Thursday or a Friday. They cannot be Saturday or Sunday.
        v. start_time and end_time must be between the hours (in 24 hour format) of 06:00:00 and 18:00:00 inclusive.
        vi. start_time must not be on the same day as any other start_time for that employee.
    b. Returns the new Timecard as a JSON String.
    c. Input:

```
{
   "emp_id":2,
   "start_time":"2018-06-14 11:30:00",
   "end_time":"2018-06-14 15:30:00"
}
```

d. Output:

i. Success:

```
{
   "success": {
      "timecard_id":1,
      "start_time":"2018-06-14 11:30:00",
      "end_time":"2018-06-14 15:30:00",
      "emp_id":2
   }
}
```

ii. Error:

```
{
   "error":"An appropriate error message."
}
```

15. **Path**: `/timecard`
Verb: PUT
Produces: application/json

a. Additional validations same as inserting a Timecard, plus timecard_id must be a valid record id in the database.

b. Returns the updated Timecard as a JSON String.

c. Input (any values you want to change plus the record id for the Timecard) as FormParams:

```
{
   "timecard_id" =1
   "emp_id" = 2
   "start_time" = "2018-06-15 12:30:00"
   "end_time" = "2018-06-15 15:30:00"
}
```

d. Output:

c. Output:

i. Success:

```
{
   "success": {
      "timecard_id":0,
      "start_time":"2018-06-15 12:30:00",
      "end_time":"2018-06-15 15:30:00",
      "emp_id":2
   }
}
```

ii. Error:

```
{
   "error":"An appropriate error message."
}
```

16. **Path**: `/timecard`
    Verb: DELETE
    Produces: application/json

a. Returns the number of rows deleted.

b. Input as a QueryParam.

```
timecard_id=#
```

where timecard_id is the record id of the Timecard to delete

c. Output:

i. Success:

```
{
   "success":"Timecard 1 deleted."
}
```

ii. Error:

```
{
   "error":"An appropriate error message."
}
```