**WORK-ORDERS API**

Creating, Fetching and modifying workers and work orders data. Returns XML data for the workers and the work orders they are assigned.

**General URL for app:** <https://shubham-api.herokuapp.com/webapi/>

**Instructions:**

* Application is deployed on Heroku and can be accessed by [https://shubham-api.herokuapp.com/webapi/\*](https://shubham-api.herokuapp.com/webapi/*) URL.
* Append the URL parameters provided in the document
* Use any HTTP client testing tools like Postman to make REST API requests and capture XML responses.
* Use sample URL <https://shubham-api.herokuapp.com/webapi/api/createworkorder> to create a workorder record.
* Set the HTTP method as POST.
* Set the Content-Type in Request Header as application/xml.
* Define the request data payload in the body section in the XML format as

<workOrderDTO>

<deadline>…….</deadline>

<description>……. </description>

<title>…….. </title>

</workOrderDTO>

* Click on send to make request to my web service.

**Functionalities:**

* **Create a worker:**  Create a worker record in the database

1. **URL Param:** api/createworker
2. **Method:** POST
3. **Data Param:** Consumes XML format of input in data payload. Ensure the XML data sent as request to server is in this format.

**<workerDTO>**

**<name>Maxwell</name>**

**<email>Maxwell.doe@gmail.com</email>**

**<company>Unicef</company>**

**</workerDTO>**

1. **Success Response:**

Code: 204

Content: None

* **Delete a worker:**  Create a worker record in the database

1. **URL Param:** api/deleteworker/name

name: Maxwell

1. **Method:** DELETE
2. **Data Param:** Name appended to the URL. Ensure we only append the string value and the user present in the database.
3. **Success Response:**

Code: 204

Content: None

1. Failure Response:

Code: 404

Content:

**<errorMessage>**

**<errorCode>404</errorCode>**

**<errorMessage>Record for user not found and cannot be**

**deleted </errorMessage>**

**</errorMessage>**

* **Create a Work Order:** Create a work order record in the database

1. **URL Param:** api/createworkorder
2. **Method:** POST
3. **Data Param:** Consumes XML format of input in data payload. Ensure the XML data sent as request to server is in this format.

**<workOrderDTO>**

**<deadline>2019-05-08T00:00:00-04:00</deadline>**

**<description>maintaining databases </description>**

**<title>DB administer </title>**

**</workOrderDTO>**

1. **Success Response:** 
   * 1. Code: 204
     2. Content: None

* **Assigning a worker to a work order:** Assigning a work order to worker if he is present in the database records.

1. **URL Param:** api/assign/empname/task

Empname: Maxwell

Task: DB administer

1. **Method:** POST
2. **Data Param:** Empname and task appended to URL as parameters, only the data records of the employee name and the task in the database must be used as parameters for assigning.
3. **Success Response:** 
   * 1. Code: 204
     2. Content: None
4. Failure Response:
   * 1. Code: 404
     2. Content:

**<errorMessage>**

**<errorCode>404</errorCode>**

**<errorMessage>Record for user Maxwell is not found in records and cannot be assigned</errorMessage>**

**</errorMessage>**

* **Fetch all work orders for a specific worker:**  Fetch all the work orders of a specific worker.

1. **URL Param:** api/workorders/{empname}

Empname: Maxwell

1. **Method:** GET
2. **Data Param:** Empname appended to URL as parameters, only the data records of the employee name must be used as parameters for assigning.
3. **Success Response:** 
   * 1. Code: 200
     2. Content:

**<fetchDTO>**

**<deadline>2019-07-08T00:00:00-04:00</deadline>**

**<description>Teaching Assistant</description>**

**<empname>john</empname>**

**<title>GTA</title>**

**</fetchDTO>**

1. Failure Response**:**
   * 1. Code: 404
     2. Content:

**<errorMessage>**

**<errorCode>404</errorCode>**

**<errorMessage>Record for user johnie not found</errorMessage>**

**</errorMessage>**

* **Fetch all work orders sorted by deadline:** Fetch all the work orders registered in the database sorted in ascending order by deadline.
  + - 1. **URL Param:** api/workordersbydeadline
      2. **Method:** GET
      3. **Data Param:** Empname appended to URL as parameters, only the data records of the employee name must be used as parameters for assigning.

1. **Success Response:** 
   * 1. Code: 200
     2. Content:

**<workOrderDTO>**

**<deadline>2019-07-08T00:00:00-04:00</deadline>**

**<description>Teaching Assistant</description>**

**<title>GTA</title>**

**</workOrderDTO>**

1. Failure Response**:**
   * 1. Code: 404
     2. Content:

**<errorMessage>**

**<errorCode>404</errorCode>**

**<errorMessage> No Records have entered </errorMessage>**

**</errorMessage>**

**DATABASE:**

* ClearDB instance is used for creating Mysql based database server.
* 3 tables are created to store data related to this application.

1. worker(name,company,email)
2. workorders(title,description,deadline)
3. assignment(empname,title)

**Happy Path Test Cases:** Test cases are tested using JUNIT

* Entering valid data for creating worker
* Deleting worker record present in the workers table
* Entering valid data for creating work order
* Assigning valid worker present in the table with valid work order.
* Fetching work orders for valid worker present in the table.
* Fetching all the work orders sorted in ascending order.

**Edge Cases Handled:**

* Assigning work order to worker not present in the records.
* Assigning more than 5 workers to same work order.
* Trying to delete worker not present in the records.
* Fetching work orders for worker not present in the records.
* Fetching work orders sorted based on deadline from empty table.