

Assignment:

We have created some API's that you have to integrate.

1. Authenticate user using the credentials specified. API uses Bearer authentication to authenticate further API calls. Authentication API will return a bearer token which you will have to pass in subsequent API calls.

Path: https://qa2.sunbasedata.com/sunbase/portal/api/assignment_auth.jsp

Method: POST

Body:

```
{  
  "login_id" : "test@sunbasedata.com",  
  "password" : "Test@123"  
}
```

This will return you a token, that you will have to pass in as Bearer token for further calls.

2. Create a new Customer:

Path: <https://qa2.sunbasedata.com/sunbase/portal/api/assignment.jsp>

Method: POST

Parameters:

cmd : create

Header:

Authorization: Bearer token_recieved_in_authentication_API_call

Body:

```
{  
  "first_name": "Jane",  
  "last_name": "Doe",  
  "street": "Elvnu Street",  
  "address": "H no 2 ",  
  "city": "Delhi",  
  "state": "Delhi",  
  "email": "sam@gmail.com",  
  "phone": "12345678"  
}
```

first_name and last_name are mandatory parameters.

Response:

Success: 201, Successfully Created

Failure: 400, First Name or Last Name is missing

3. Get customer list:

Path: <https://qa2.sunbasedata.com/sunbase/portal/api/assignment.jsp>

Method: GET

Parameters:

cmd : get_customer_list

Header:

Authorization: Bearer token_recieved_in_authentication_API_call

Response: 200

```
{
  "first_name": "Jane",
  "last_name": "Doe",
  "street": "Elvnu Street",
  "address": "H no 2 ",
  "city": "Delhi",
  "state": "Delhi",
  "email": "sam@gmail.com",
  "phone": "12345678"
}
```

4. Delete a customer

Path: <https://qa2.sunbasedata.com/sunbase/portal/api/assignment.jsp>

Method: POST

Parameters:

cmd : delete

uuid : uuid of a specific customer

Header:

Authorization: Bearer token_recieved_in_authentication_API_call

Response:

200, Successfully deleted

500, Error Not deleted

400, UUID not found

5. Update a customer

Path: <https://qa2.sunbasedata.com/sunbase/portal/api/assignment.jsp>

Method: POST

Parameters:

cmd : update

uuid : uuid of a specific customer

Header:

Authorization: Bearer token_recieved_in_authentication_API_call

Body:

```
{
  "first_name": "Jane",
  "last_name": "Doe",
  "street": "Elvnu Street",
  "address": "H no 2 ",
  "city": "Delhi",
  "state": "Delhi",
  "email": "sam@gmail.com",
  "phone": "12345678"
}
```

Response:

200, Successfully Updated

500, UUID not found

400, Body is Empty

Notes:

if you are passing in wrong value for cmd parameter, you will get 500, Invalid Command

If for any API authentication fails you will get 401, Invalid Authorization

You can test tools like postman to test these API's.

Basic steps include:

- create a basic UI in html. does not have to look pretty. basic Html table , form and buttons would do.











- Basically we need the ability to Login, view the list of customers, create a new customer, delete and update an existing customer.

It will consist of 3 screens:

- Login Screen

Login Page

- Customer List Screen

Customer List							Delete this customer	Edit this customer
First Name	Last Name	Address	City	State	Email	Phone	Action	
Sakshi	Sharma	Hno 1	Delhi	Delhi	sam@gmail.com	10258	 	
Nikhil	Sharma	Hno 1	Delhi	Delhi	sam@gmail.com	10232	 	
Roneet	SHarma	Hno 1	Delhi	Delhi	sam@gmail.com	10267	 	
Jane	Jane	Hno 1	Delhi	Delhi	sam@gmail.com	10244	 	
Tom	Tom	Delhi	Delhi	Delhi	sam@gmail.com		 	

- Add a new customer

Customer Details

First name	Last name
Street	Address
City	State
Email	Phone

[Submit](#)

Create a new customer
on click of this button

Submission Instructions.

- Create a new Public repo on Github and push your code in this repository. Do not hardcode the login credentials anywhere in the code.
- Add a README file to specify things like what the project does and how to get it up and running.
- Add comments in your code to ensure readability of code.