Software Requirements Specification (SRS)

ADACTIN Application

Author(s): **Suresh**
Version : **1.0**
Date First Issued:
Updated:

Contents

1.Introduction

This Software Requirements Specification (SRS) describes requirements for Adactin Operations. This SRS document describes the software functional and non-functional requirements of the Adactin System. This document is intended to be used by the members of the project team that will implement and verify the correct functioning of the system. Unless otherwise noted, all requirements specified here are high priorities and committed for release.

Reference Documents

Document Title	Version No.

2.PURPOSE OF THE SYSTEM

To search a required hotel across different location. User is able to book a hotel and if required booking can be canceled by providing proper reason.

Scope of the System

1. New user Registration
2. Login into application
3. Search a Hotel
4. Select a hotel
5. Booking a hotel
6. Order Id generation
7. Cancel order
8. Logout

3.NUMBER OF MODULES

The system after careful analysis has been identified to be presented with the following modules:

The Modules involved are

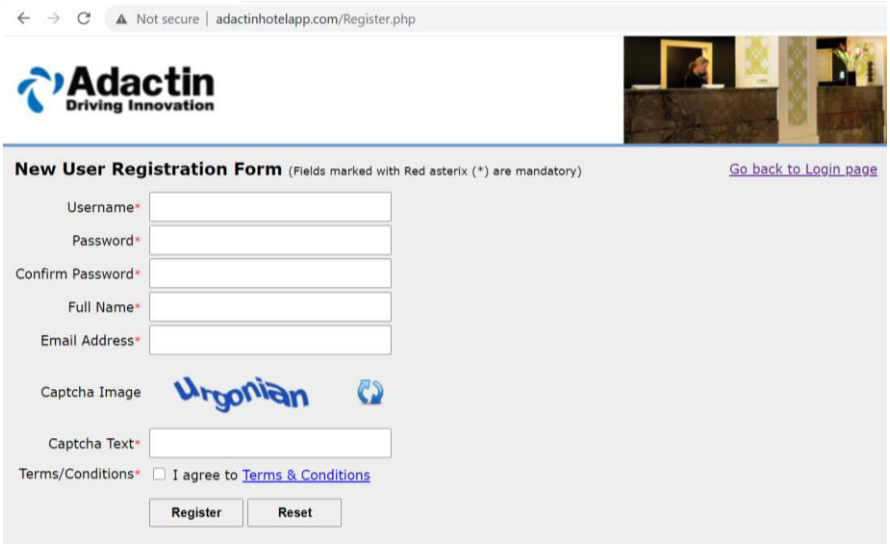
1. New User Registration
2. Login
3. Search Hotel
4. Select Hotel
5. Book Hotel
6. Order Generation
7. Booked Itinerary

3.1 New User Registration:

New user Registration is used to get login detail for the application. To start using this application first we need to complete new user registration by using below form.

Navigation :

Launch Browser →Enter url →click on **New user Register Here** link



Enter all the required fields and clickon Register button .After some we need to recive an email from admin team with login detail.

3.2Login :

This module is used to login into application by using valid login details.in case of user entered in valid login details proper error message need to be displayed.

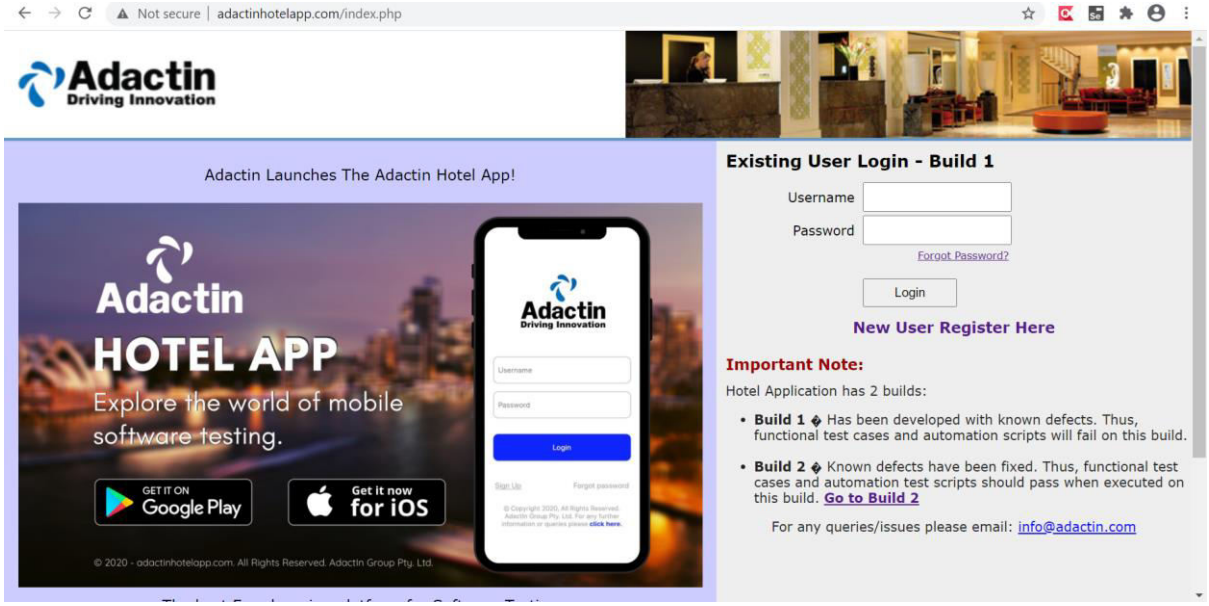
Navigation : Launch Browser →Enter url :

1. Login page should be open.
2. In Login Page Username and password Textboxes should be displayed along With the login button.
4. Enter valid username and password.
5. Press Log in Button.
6. Search Hotel page need to be displayed

→Login Page should be displayed with required fields as below:

Fields information:

Company Logo, images, Welcome text, Login fields , links ,Header and footer .



3.3Search Hotel :

This module is used to search a required hotel by proving all the required inputs.

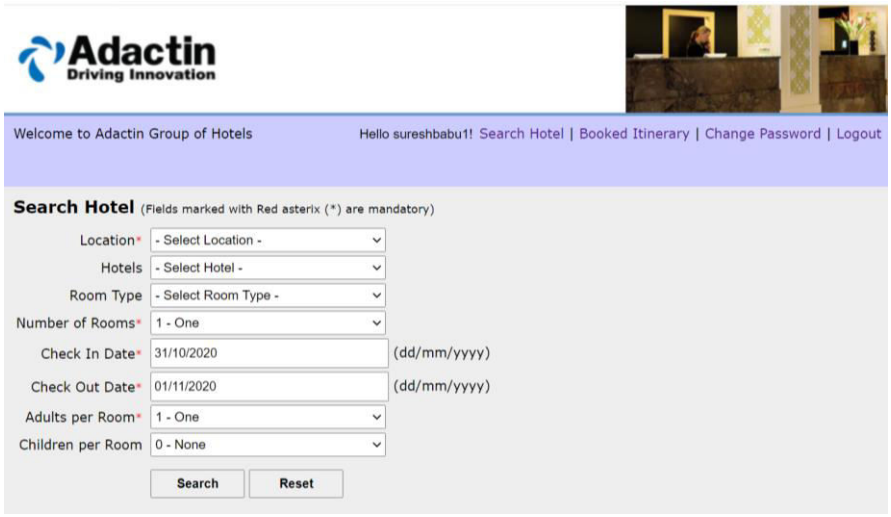
Navigation : Login into application →Search a Hotel

1.Login into application

2.Select all the required fields

3.Click on Search button

4.Select Hotel Page need to be displayed.



3.4 Select Hotel :

This module is used to select a required hotel by proving all the required inputs.

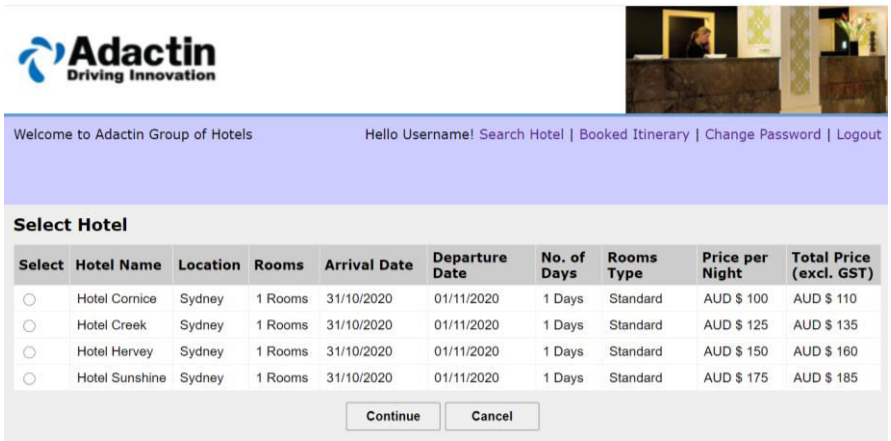
Navigation : Login into application →Search a Hotel→Select a Hotel

1.Login into application

2.Select all the required fields

3.Click on Search button

4.Select a required Hotel

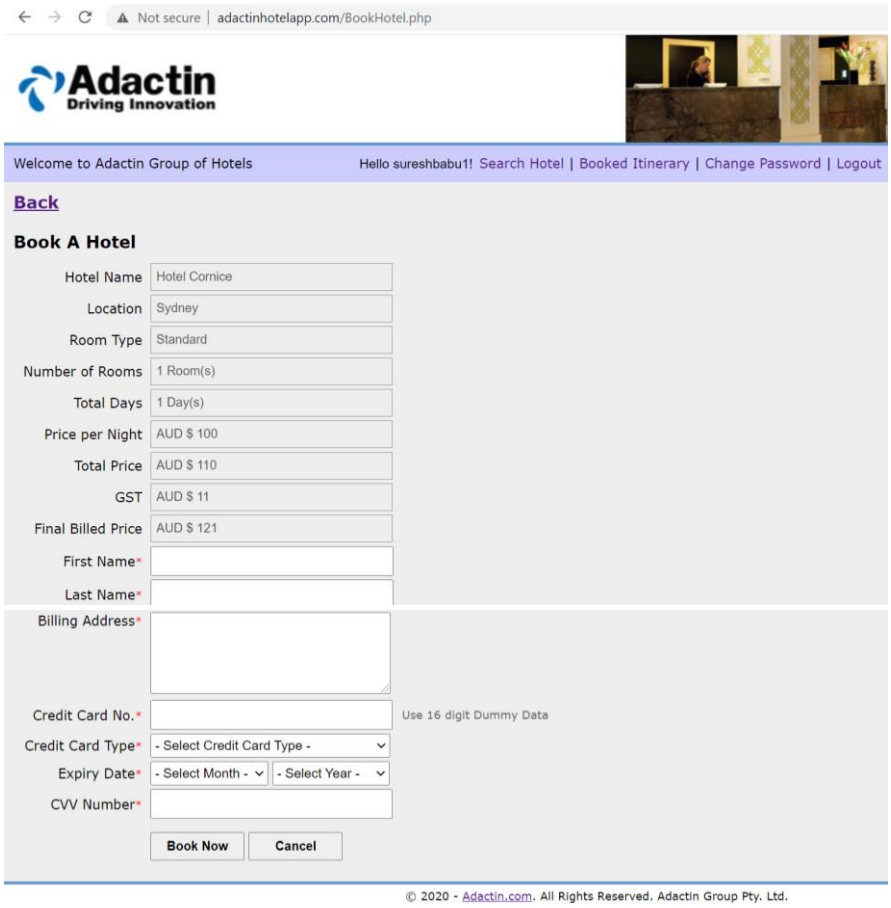


3.5 Book a Hotel :

This module is used to book a hotel by proving all the required inputs.

Navigation : Login into application →Search a Hotel→Select a Hotel→Book a hotel

1.Login into application
2.Select all the required fields
3.Click on Search button
4.Select a required Hotel
5.Provide all the required details
6.click on Book now button
7.Order number should be generated



3.6 Order Generation:

This module is used to refer order number had been generated.

Navigation : Login into application →Search a Hotel→Select a Hotel→Book a hotel→Order number should be auto-generated.

1.Login into application
2.Select all the required fields
3.Click on Search button

4.Select a required Hotel

5.Provide all the required details

6.click on Book now button

7.Order number should be auto-generated

3.7 Booked Itinerary:

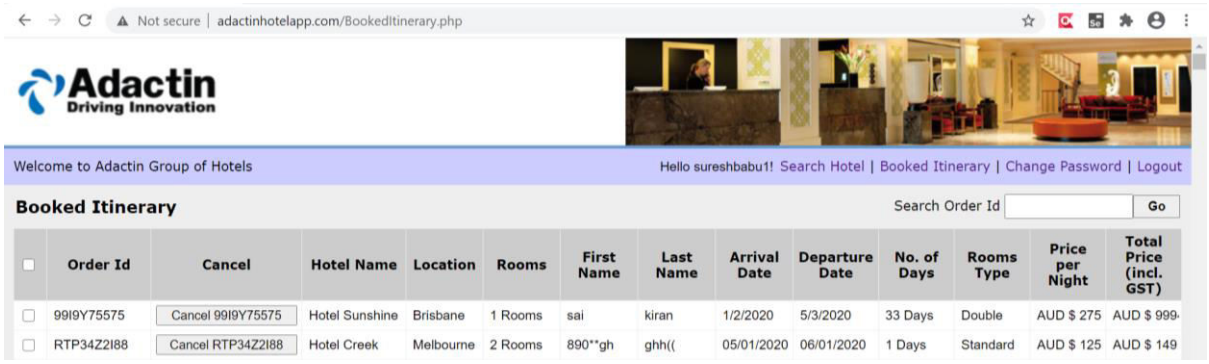
This module is used to confirm booking details by providing valid order number.

Navigation : Login into application →Click on Booked Itinerary link→Enter valid Order number.

1.Login into application

2.Click on Booked Itinerary link

3.Enter valid Order number

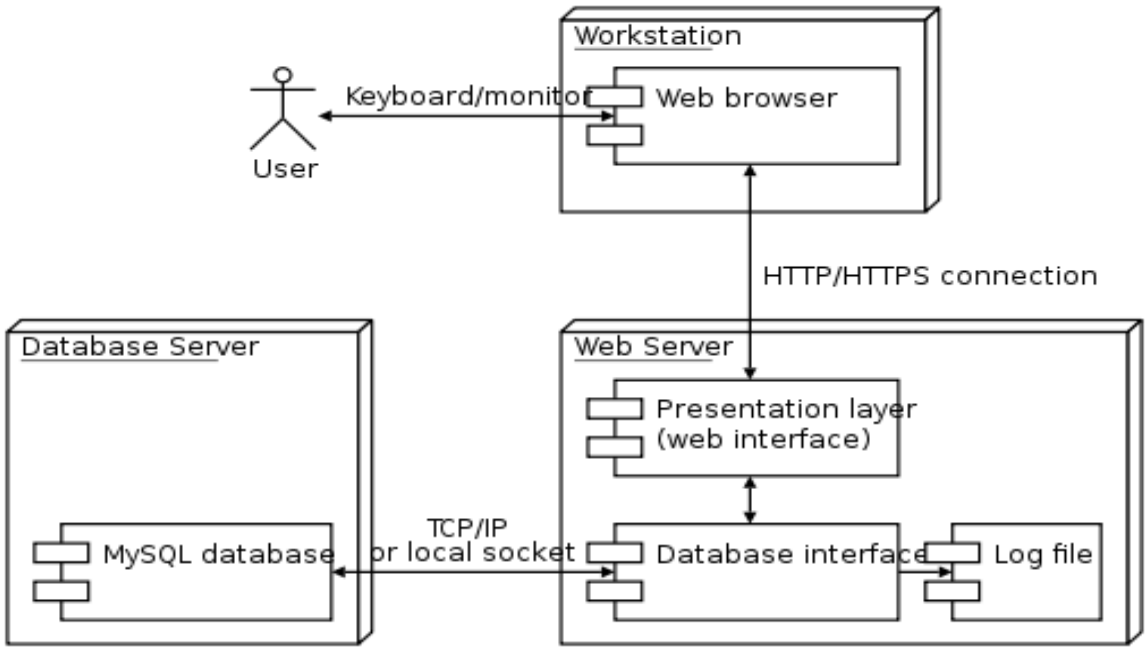


4. Functional components of the project

Following are the functional needs of the system. More functionality can be added to enhance the application.

1. user must have a valid User Id and password to login to the system
2. If a wrong password is given thrice in succession, that account will be locked and the customer will not be able to use it. When an invalid password is entered a warning is given to the user that his account is going to get locked.
3. After the valid user logs then search hotel page need to be displayed
4. User need be able to provide all the required fields
5. On clicking Search button Select Hotel page need to displayed
6. User should be able to select required hotel and On clicking on Continue button
7. User had to enter all the personal detail including payment details and On clicking book now button Order Id need to be generated
8. With the order number need to able to book confirmation details and if required need to able to cancel the booking.

5. Deployment Diagrams



6. APPLICATION DEVELOPMENT:

N-Tier Applications:

N-Tier Applications can easily implement the concepts of Distributed Application Design and Architecture. The N-Tier Applications provide strategic benefits to Enterprise Solutions. While 2-tier, client-server can help us create quick and easy solutions and may be used for Rapid Prototyping, they can easily become a maintenance and security night mare

The N-tier Applications provide specific advantages that are vital to the business continuity of the enterprise. Typical features of a real life n-tier may include the following:

- Security
- Availability and Scalability
- Manageability
- Easy Maintenance
- Data Abstraction

The above mentioned points are some of the key design goals of a successful n-tier application that intends to provide a good Business Solution.

Definition:

Simply stated, an n-tier application helps us distribute the overall functionality into various tiers or layers:

- Presentation Layer
- Business Rules Layer
- Data Access Layer
- Database/Data Store

Each layer can be developed independently of the other provided that it adheres to the standards and communicates with the other layers as per the specifications.

This is the one of the biggest advantages of the n-tier application. Each layer can potentially treat the other layer as a 'Block-Box'.

In other words, each layer does not care how other layer processes the data as long as it sends the right data in a correct format.

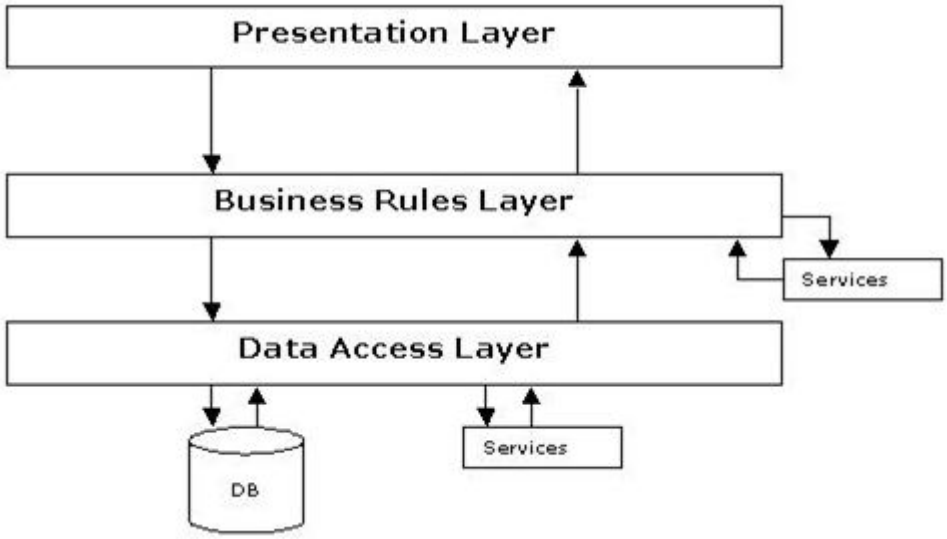


Fig 1.1-N-Tier Architecture

1. **The Presentation Layer:**

   Also called as the client layer comprises of components that are dedicated to presenting the data to the user. For example: Windows/Web Forms and buttons, edit boxes, Text boxes, labels, grids, etc.

2. **The Business Rules Layer:**

   This layer encapsulates the Business rules or the business logic of the encapsulations. To have a separate layer for business logic is of a great advantage. This is because any changes in Business Rules can be easily handled in this layer. As long as the interface between the layers remains the same, any changes to the functionality/processing logic in this layer can be made without impacting the others. A lot of client-server

apps failed to implement successfully as changing the business logic was a painful process.

3. **The Data Access Layer:**

   This layer comprises of components that help in accessing the Database. If used in the right way, this layer provides a level of abstraction for the database structures. Simply put changes made to the database, tables, etc do not affect the rest of the application because of the Data Access layer. The different application layers send the data requests to this layer and receive the response from this layer.

4. **The Database Layer:**

   This layer comprises of the Database Components such as DB Files, Tables, Views, etc. The Actual database could be created using Mysql, Oracle etc.

   In an n-tier application, the entire application can be implemented in such a way that it is independent of the actual Database. For instance, you could change the Database Location with minimal changes to Data Access Layer. The rest of the Application should remain unaffected

7. PERFORMANCE REQUIREMENTS:

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely in the part of the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use.

The requirement specification for any system can be broadly stated as given below:

- The system should be able to interface with the existing system

- The system should be accurate
- The system should be better than the existing system

The existing system is completely dependent on the user to perform all the duties.

8. Specifications:

Software Specifications

- **Operating System** : Windows
- **Database Server** : MySQL
- **Client** : Java Script
- **Tools** : Xampp
- **User Interface** : HTML with Ajax
- **Code Behind** : PHP
- **Web server** : Apache

Hardware Specification:

- **Processor** : Dual Core
- **RAM** : 1GB Ram
- **Hard Disk** : PC with 20GB

Approvals:

Name	Signature	Date

Software Requirements Specification (SRS)

API Documentation

WebApplication

Author(s): **Suresh**
Version : **2.0**
Date First Issued:
Updated:

API Testing : API stands for Application Programming Interface which acts as an intermediate of communication between two applications. Due to this intermediary role of API (Application Programming Interface) two applications talk to each other and performs the required actions efficiently. API contains a set of rules and guidelines based on which the applications are developed. So in simple we can say an API acts as an interface between two software applications so that two software applications can communicate with each other. In API Testing, instead of using standard user inputs(keyboard) and outputs, you use software to send calls to the API, get output, and note down the system's response. API tests are very different from GUI Tests and won't concentrate on the look and feel of an application. It mainly concentrates on the business logic layer of the software architecture.

As part of API testing need to test below functionality

- Data accuracy.
- Response time.
- Duplicate or missing functionality.
- Authorization checks.
- Multithreaded issues.
- Security and performance issues.
- Error codes if API returns.
- Reliability issues.

1. **url format**
   protocall//serverAddress/queryparameters/endpoint
   https://reqres.in/api/users?page=2
2. **HTTP response status codes**

The server should always return the right HTTP status code to the client.

Standard HTTP status codes

200 – OK – Everything is working

201 – OK – New resource has been created

204 – OK – The resource was successfully deleted

304 – Not Modified – The client can use cached data

400 – Bad Request – The request was invalid or cannot be served. The exact error should be explained in the error payload. E.g. „The JSON is not valid"

401 – Unauthorized – The request requires a user authentication

403 – Forbidden – The server understood the request but is refusing it or the access is not allowed.
404 – Not found – There is no resource behind the URI.
422 – Unprocessable Entity – Should be used if the server cannot process the entity, e.g. if an image cannot be formatted or mandatory fields are missing in the payload.
500 – Internal Server Error

Request	GET	POST	PUT	DELETE
Status Code	200	201	200	204

3. Common Functionality of API Testing

protocall//serverAddress/queryparameters/endpoint
https://reqres.in/api/users?page=2

3.1. url should be work with only https protocol
3.2. domain address should be followed by .com/.in/.info
3.3. server address should be reqres.in
3.4. Query parameter should be working with ? symbol
3.5. Special characters should be accepted only // : ? + &

4. Status Code Fun:

4.1. GET Request

Single user : To retrieve one user information
Query : https://reqres.in/api/users/2
Result : 200


```
{
   "data": {
      "id": 2,
      "email": "janet.weaver@reqres.in",
      "first_name": "Janet",
      "last_name": "Weaver",
      "avatar": "https://reqres.in/img/faces/2-image.jpg"
   }
```


List users : To get list of users information
Query : https://reqres.in/api/users?page=2
Result : 200

```
{
   "page": 2,
   "per_page": 6,
   "total": 12,
   "total_pages": 2,
   "data": [
      {
         "id": 7,
         "email": "michael.lawson@reqres.in",
         "first_name": "Michael",
         "last_name": "Lawson",
         "avatar": "https://reqres.in/img/faces/7-image.jpg"
      },
      {
         "id": 8,
         "email": "lindsay.ferguson@reqres.in",
         "first_name": "Lindsay",
         "last_name": "Ferguson",
         "avatar": "https://reqres.in/img/faces/8-image.jpg"
      },
      {
         "id": 9,
         "email": "tobias.funke@reqres.in",
         "first_name": "Tobias",
         "last_name": "Funke",
         "avatar": "https://reqres.in/img/faces/9-image.jpg"
      },
      {
         "id": 10,
         "email": "byron.fields@reqres.in",
         "first_name": "Byron",
         "last_name": "Fields",
         "avatar": "https://reqres.in/img/faces/10-image.jpg"
      },
      {
         "id": 11,
```

```
        "email": "george.edwards@reqres.in",
        "first_name": "George",
        "last_name": "Edwards",
        "avatar": "https://reqres.in/img/faces/11-image.jpg"
    },
    {
        "id": 12,
        "email": "rachel.howell@reqres.in",
        "first_name": "Rachel",
        "last_name": "Howell",
        "avatar": "https://reqres.in/img/faces/12-image.jpg"
    }
],
```


4.2. POST Request

POST : To insert the data to server

Query : https://reqres.in/api/users

Body :


```
{
    "name": "Suresh",
    "job": "Trainer"
}
```


Result : 201 , ID should be generated along with Created AT Message

4.3. PUT Request

PUT : To update the data to server

Query : https://reqres.in/api/users/2

Body :


```
{
    "name": "Suresh",
    "job": "softwareTrainer"
}
```


Result : 200 , Updated AT Message should be displayed

4.4. DELETE Request

POST : To Delete the data from server

Query : https://reqres.in/api/users/2

Body : { }

Result : 204

