Project Name : Customer Module

Author : Kumar

Date Of Assignment: 10-06-2021

Date Of Submission: 03-07-2021

Document detail

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Description | Version | Comment | Date |
| Kumar | Initial Draft | 1.0 |  | 26-06-2021 |

Document Future Scope

|  |
| --- |
|  |
|  |
|  |
|  |
|  |

Steps to be followed

* Requirement Gathering
* Project life-cycle discussion
* Agile Methodology discussion
* Jira Setup and management
* Git Setup and Management
* Create RAML
* Develop API using RAML
* Local Testing
* Munit Testing
* Code Check-in
* Discussion about deployment modal
* Deploy application on Cloud-hub
* Apply policy

# Objective of this exercise

* To simulate a real-world scenario, where a solution architecture document will be handed down to the development team to build & implement.
* Make use of MuleSoft out of the box capabilities (as much as possible) to come out with a solution to address the design problems addressed in the use case.
* Include previous best practices experience when coming out with the solution.
* Make use of Design Center to create RAML projects and design RAML specification base RAML best practices based on the business scenario given.
* Organize and create the mule projects based on the business scenarios, and solutions described in this document.
* Creating mule message flows, making use of mule components and connectors. implement a solution based on the scenario and specifications given in this document.
* M-unit test is created to test each of the possible scenarios.
* Test scripts creation to test the end point using testing tools such as Postman / Soap UI (if needed).
* Demo the working example at the end. During the presentation, outline in detail what you are trying to solve in your Mule project and the approach to resolve it.
* Extend and enhance your application in real-time.

Project Life cycle:

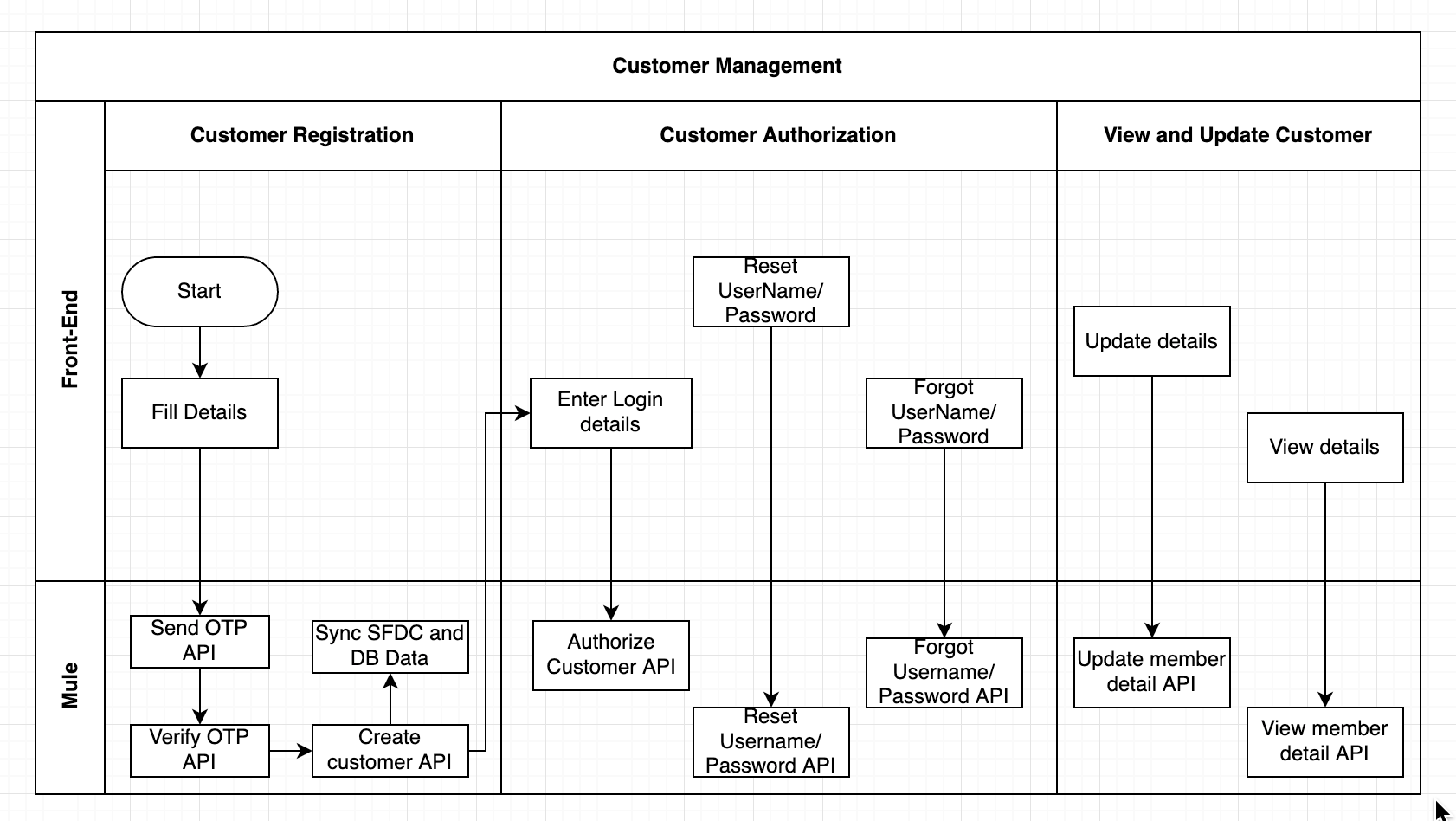
* Requirement Gathering
* Create RAML
* Develop API using RAML
* Local Testing
* Munit Testing
* Code Check-in
* Deploy application on Cloud-hub
* Apply policy
* Move Code to Higer environment ( Dev -> SIT ->UAT ->PROD)
* Support

Business Scenario

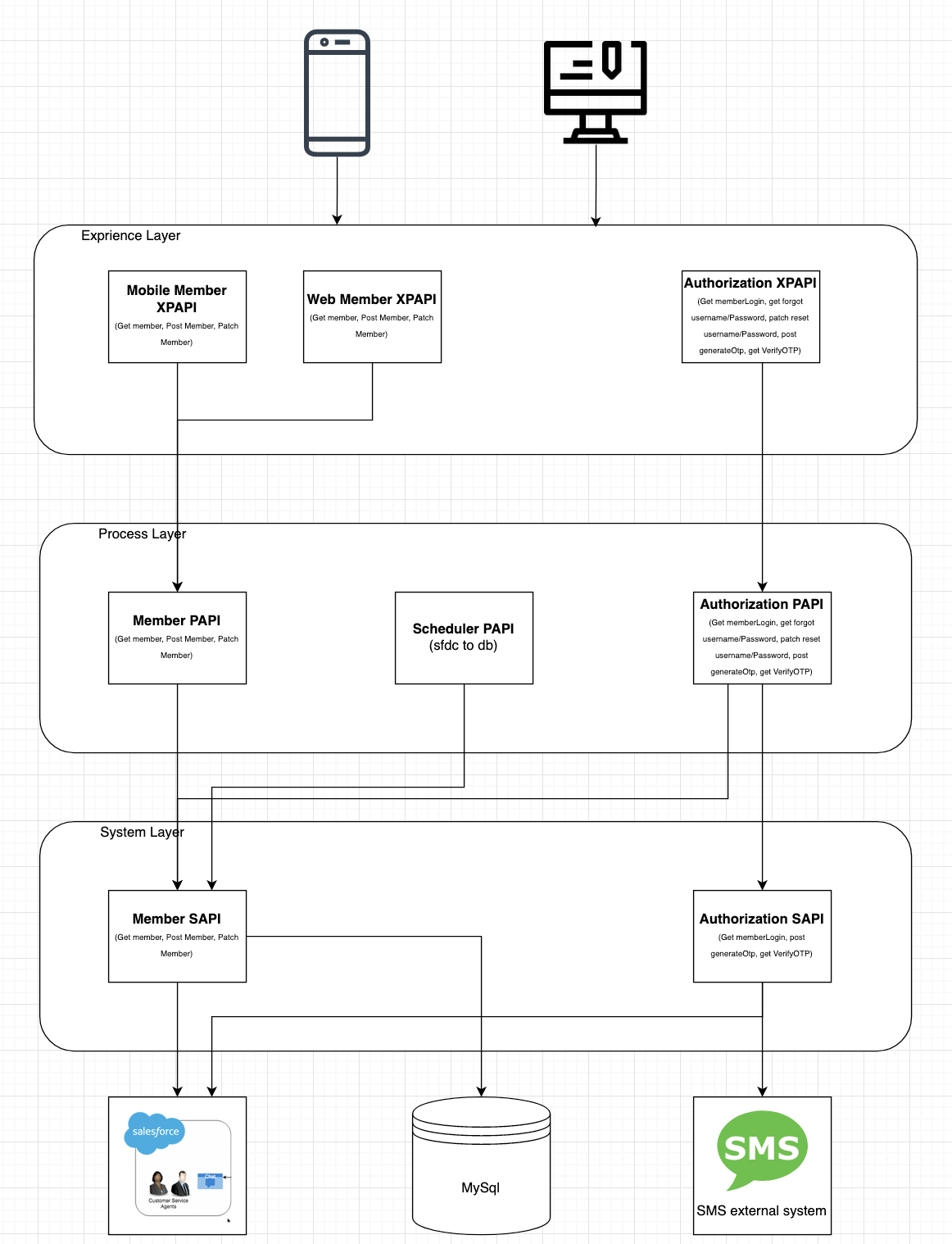
Customer management system will allow customers create account , to check and update the member details ,member login and reset member credentials. First customer will go to company portal and using signup option will create account by providing basic details. Once customers complete successful member registration, they can login using the login username and password details. If in case, customers forget the login details, they can utilize forgot password and forgot username options.

During customers login, they will receive an OTP on their mobile which is needed as part of login process. After successful login, customer can view member details, update member details, reset password.

Flow Diagram



# API Landscape



**1. Mobile member XPAPI**

**a. POST /members:** (Member registration) - Mobile-Member-Registration-XPAPI

**b. GET /members:** (Get member details) - Mobile-GetMember-Details-XPAPI

**c. PATCH /members:** (Update member details) - Mobile-Member-Update-XPAPI

**d. GET/members/{memberId}:** (Get member details) - GetMember-Details-PAPI

**2. Web member XPAPI**

**a. POST /members:** (Member registration) - WebApp-Member-Registration-XPAPI

**b. GET /members:** (Get member details) - WebAPP-GetMember-Details-XPAPI

**c. PATCH /members:** (Update member details) - WebApp-Member-Update-XPAPI

**3. Authorization XPAPI**

**a. GET /members/login:** (Member login) - Member-Login-XPAPI

**b. GET /members/credentials:** (Forgot UserName and Password) - Member-Forgot- UserNameAndPassword-XPAPI

**c. PATCH /members/credentials:** (Reset UserName And Password)- Member-Reset- UserNameAndPassword-XPAPI

**d. POST/members/{memberId}/OTP:** (Generate OTP code) - Member-Generate-OTP-XPAPI

**e. GET/members/{memberId}/OTP/{OTPId}:** ( Verify OTP code) - Member-Verify-OTP-XPAPI

**4. Member PAPI**

**a. POST/members:** (Member registration) - Member-Registration-PAPI

**b. GET/members/{memberId}:** (Get member details) - GetMember-Details-PAPI

**c. GET/members:** (Get member filter email(Optional), mobile(optional)) - Mobile- GetMember-Details-PAPI

**d. PATCH/members/{memberId}:** (Update member details) - Member-Update-PAPI

**5. Scheduler PAPI**

**a. Scheduler API:** ( To move data from SFDC to DB) – Schedular-SFDC-To-DB

**6. Authorization PAPI**

**a. GET/members/login:** (Member login) - Member-Login-PAPI

**b. POST/members/{memberId}/OTP:** (Generate OTP code) - Member-Generate-OTP-PAPI

**c. GET/members/{memberId}/OTP:** ( Verify OTP code) - Member-Verify-OTP-PAPI

**7. Member SAPI**

**a. POST/members:** (Member registration SFDC) - Member-Registration-SFDC-SAPI

**b. POST/members/sync:** (Member detail to DataBase) - Member-Registration-DB-SAPI

**c. GET/members/{memberId}:** (Get member details) - GetMember-Details-SAPI

**d. GET/members:** (Get member filter email, mobile, registrationDate) - Mobile-GetMember- Details-SAPI

**e. PATCH/members/{memberId}:** (Update member details) - Member-Update-SAPI

**8. Authorization SAPI**

**a. POST/members/OTP:** (Generate OTP code) - Member-Generate-OTP-SAPI

**b. GET/members/{memberId}/OTP:** (Get OTP code ) - Member-Get-OTP-SAPI

**External System:**

1. **Salesforce**

Salesforce is a cloud computing service as a software (SaaS) company that specialises in customer relationship management (CRM). Salesforce's services allow businesses to use cloud technology to better connect with customers, partners and potential customers. The software has become the number one for customer success and helps businesses track customer activity, market to customers and many more services.

1. **Data Base**

A database is a collection of information that is organised so that it can be easily accessed, managed and updated. Computer databases typically contain aggregations of data records or files, containing information about sales transactions or interactions with specific customers.

1. **Object Store**

Often referred to as object-based storage, is a data storage architecture for handling large amounts of unstructured data. This is data that does not transform to or cannot be organised easily into a traditional relational database with rows and columns.

1. **external SMS API**

An SMS API is a type of API that allows your business to integrate SMS (Short Message Service) messaging into your existing software platforms. SMS APIs enable you to send or receive SMS messages quickly and easily through any website or application.

1. **Frontend**

It is the website with which user interacts. It is also known as user interface.