|  |  |
| --- | --- |
|  | **Cognizant Academy**  **MFRP**  **Software Specification Document**  **Version 1.0** |
| |  |  |  | | --- | --- | --- | |  | **Prepared By / Last Updated By** | **Reviewed By** | | **Name** | Youvaraja Sundararaman | Janakiraman, Sudharsanan | | **Role** | Architect  (SharePoint Trainer/Mentor) | Sr. Architect | | **Signature** |  |  | | **Date** | 22 Mar 2021 | 22 Mar 2021 | |
|  |

Table of Contents

[Introduction 3](#_Toc67303857)

[Purpose of this document 3](#_Toc67303858)

[Definitions & Acronyms 3](#_Toc67303859)

[Project Overview 3](#_Toc67303860)

[Scope 3](#_Toc67303861)

[Intended Audience 3](#_Toc67303862)

[Hardware and Software Requirement 4](#_Toc67303863)

[Data Flow Diagram 5](#_Toc67303864)

[Use Case diagram 6](#_Toc67303865)

[SharePoint List Design 7](#_Toc67303866)

[Customers 7](#_Toc67303867)

[Products 7](#_Toc67303868)

[Orders 7](#_Toc67303869)

[Design Considerations 8](#_Toc67303870)

[User Requirements 9](#_Toc67303871)

[Document Deliverables 9](#_Toc67303872)

[Important Note 9](#_Toc67303873)

[Coding Standards & Evaluation 10](#_Toc67303874)

[SharePoint Online & SPFx: 10](#_Toc67303875)

[HTML 10](#_Toc67303876)

[SCSS 10](#_Toc67303877)

[TYPESCRIPT 11](#_Toc67303878)

# Introduction

## Purpose of this document

The purpose of this document is to define the software specification for MFRP.

## Definitions & Acronyms

|  |  |
| --- | --- |
| Definition / Acronym | Description |
| NPM  SPFx  Sp-Pnp-Js  IDE  HTML  CSS | Node Package Manager  SharePoint Framework  SharePoint Patterns and Practices  Integrated Development Environment  Hyper Text Markup Language  Cascading Style Sheets |

## Project Overview

Build a sales system for a medium-sized automobile company using SharePoint Online (O365)

## Scope

Develop a SPFx client side web part for Sales system in SharePoint Online.

## Intended Audience

* Product Owner
* Scrum Master
* Application Architect
* Project Manager
* Test Manager
* Development Team
* Testing Team

## Hardware and Software Requirement

1. Hardware Requirement:
   1. Developer Desktop PC/Laptop with 4GB RAM
2. Software Requirement
   1. Chrome Browser
   2. MS Visual Studio Code (IDE) Free Edition
   3. Microsoft SharePoint Online (O365) Free Developer Subscription
   4. Node version 10.18.0.
   5. NPM version 6.13.4
   6. Yeoman and Gulp.
   7. Visual Studio Code (IDE) for SPFx development.
   8. React Framework, HTML, CSS, and Microsoft Fabric UI for Front End development.
   9. Typescript, Sp-Pnp-Js for Back End development.
   10. JavaScript to perform PreSaveAction or PowerApps validation or Custom Column validation(less recommended)
   11. Workbench to perform WebPart Testing.
   12. Power Automate for automation.

# Data Flow Diagram

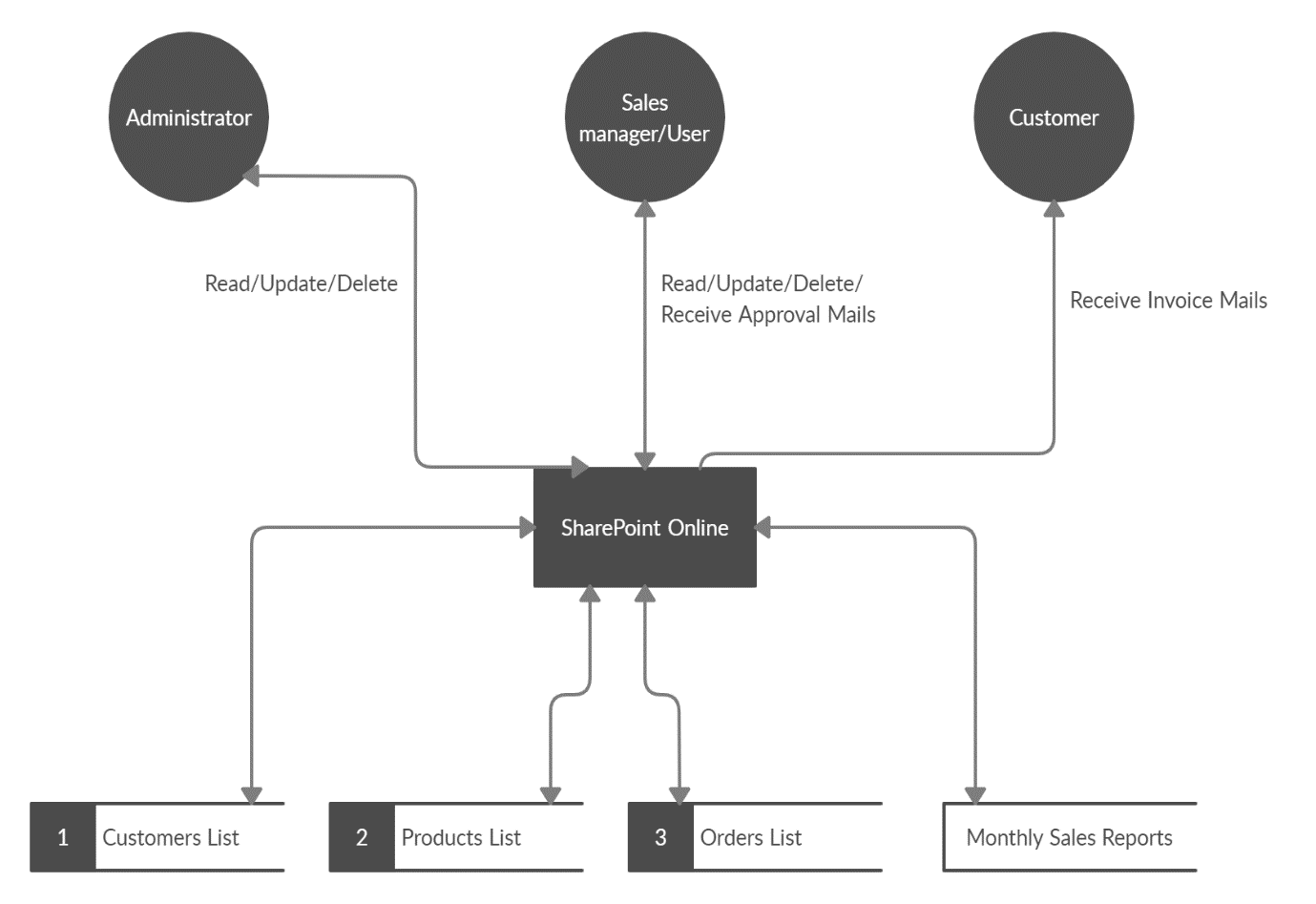


Fig.1 Data Flow Diagram

# Use Case diagram

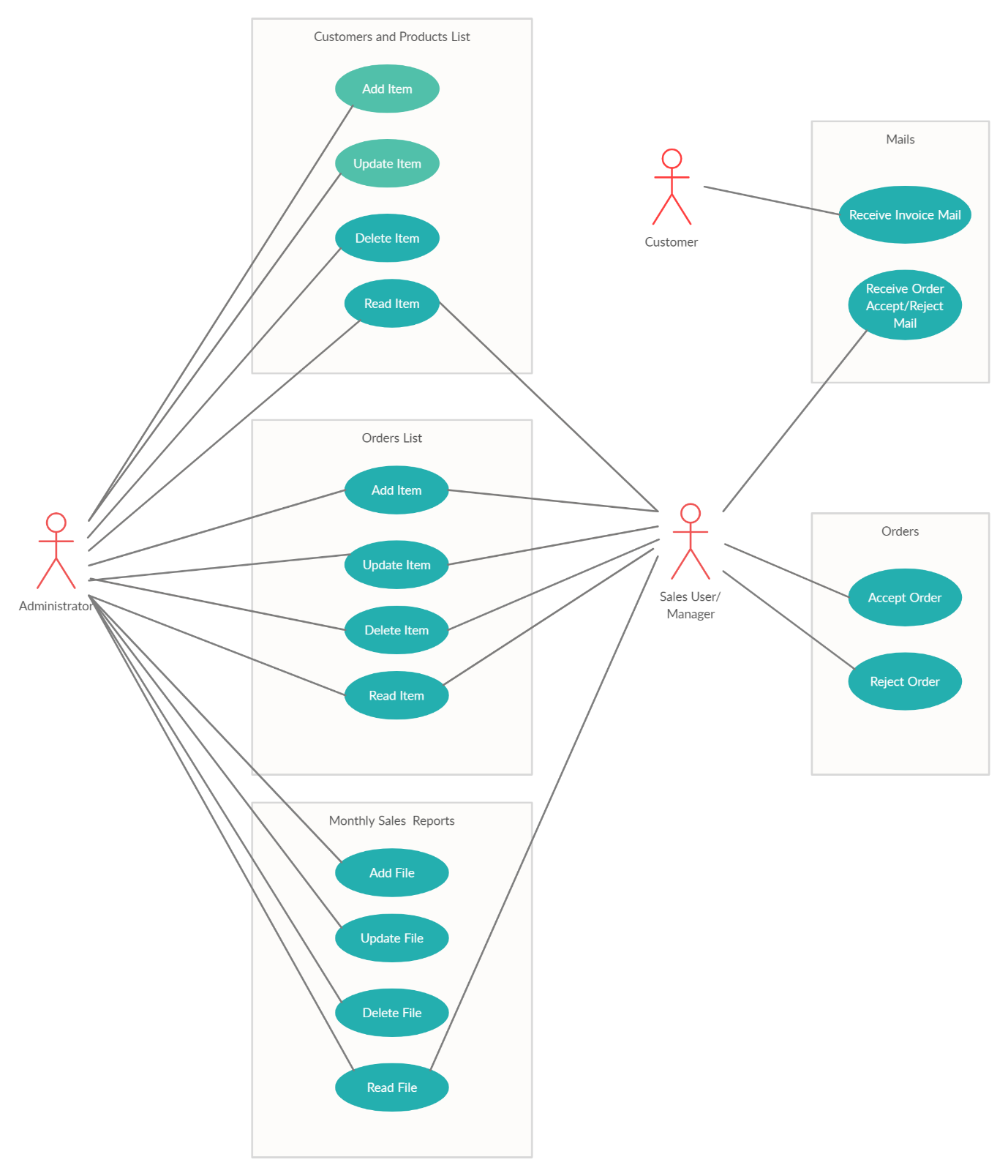


Fig.2 Use Case Diagram

# SharePoint List Design

Create following SharePoint lists in the site with the given columns:

## Customers

**Field Specification**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Column Type** | **Mandatory & Validation** |
| Customer ID | Number | Mandatory, only number must be allowed |
| Customer Name | Single line of text | Mandatory, only alphabets must be allowed |
| Customer E-mail Address | Single line of text | Mandatory, E-mail validation |

## Products

**Field Specification**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Column Type** | **Validation** |
| Product ID | Number | Mandatory, only number must be allowed |
| Product Name | Single line of text | Mandatory, only alphabets must be allowed |
| Product Unit Price | Number | Mandatory, only number must be allowed |
| Product Expiry date | Date and time | Mandatory |
| Product Type | Choice | Mandatory |

## Orders

**Field Specification**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Column Type** | **Validation** |
| Order ID | Number | Mandatory, only number must be allowed |
| Customer ID | Single line of text | Mandatory, only alphabets must be allowed |
| Product ID | Number | Mandatory, only number must be allowed |
| Units Sold | Number | Mandatory |
| Unit Price | Number | Mandatory |
| Sale Value | Number | Mandatory |
| Order Status | Choice | Values: Approve, Rejected, Completed |

# Design Considerations

1. Create a Site Collection and Team Site for Sales Portal
2. Design the Homepage with customization with Header & Footer (Application Customizer)
3. Configure Navigation – Both Top Navigation & Quick Launch Navigation as required, add appropriate links to navigate to Sales Order Form, Product, Customer & Order Lists, Reports
4. Create Custom Views and Reports as per instructions
5. Build a SPFx webpart with ReactJS UI to make the order entries which will have the following UI fields with Header and Footer
   1. Customer Name: (Dropdown list which gets data from customer list)
   2. Product Name: (Dropdown list which will display all the products)
   3. Product Type – Auto populate based on selected product name field
   4. Product Expiry date – Auto populate based on selected product name field
   5. Product Unit value – Auto populate based on selected product name field
   6. Number of Units – Text box that accepts only numeric values
   7. Sale value – Calculated based on Number of Units (\*) Unit value
   8. **Add Button** – To add the order to **Orders** list
   9. **Edit Button** – To edit the orders by passing Order ID
   10. **Delete Button** – To Delete Orders by passing Order ID
   11. **Reset Button** – To Clear all values after Insert, update operations
6. OOB Custom Views – Based on Customer Name, Product Name, Date, Sales Value > 10
7. Add appropriate validation using UsePreSaveAction in all OOB Lists and JavaScript/ JQuery validations on client side
8. Create 2 security groups in the site
   1. Administrator – Should be able to add/edit/delete records in all the lists
   2. Sales User/Manager – Will have read access to customer and Product, Contribution access to Orders list
9. Send mail to Sales users using Flow (Power Automate)
   1. Send mail to Sales Manager, Customer E-Mail for each Sales order (Invoice Format)
   2. Approval for Sales order value more than 1,00,000, update the status in orders list as Approved, Point 1 send invoice and update status as completed
   3. Generate summary report of monthly sales in excel format

# User Requirements

1. Administrator – Should be able to add/edit/delete lists
2. Sales Manager/Sales Users – Should be able to add/edit/delete records in all the Sales system lists

Will have read/write access to Customer, Products and Orders list.

1. Sales User should be able to search orders in the system

# Document Deliverables

1. Functional Requirement Specification Document, detailing the requirements
2. High Level Design Document detailing the design elements and flow of events
3. Code solution
4. Deployment packages, instructions, Flow configurations
5. A brief presentation to walk through the MFRP modules
6. Any other additional documents required by academy

## Important Note

1. The team should be in a position to talk through the reason why the current design was chosen
2. At all stages, the mentor should be informed and consulted.
3. Code should have appropriate commenting, both inline, block level and all best practices shall be followed in line with Microsoft recommendations
4. All styles shall be driven using style sheets. There shall be no inline styling and placed in Site Assets.

# Coding Standards & Evaluation

## SharePoint Online & SPFx:

1. 3 lists should be created with proper naming  convention
2. Columns should be created with proper naming convention & appropriate data types
3. Column names should be validated properly (in the list -provide validation messages wherever necessary).
4. Project Name & project folders should be created with proper naming convention.
5. Order form should be designed with rich look & feel. (UI Fabric)
6. Controls on the Order form should also be validated (provide validation messages wherever necessary).
7. Code standards should be followed for naming controls on the form and also for the methods or functions.
8. Comments should be provided explaining the functionality of the methods or functions used in the project.
9. Project should be executed in debug mode and then only should be shipped to release mode.
10. Explanation of the methods or functions in detail being used in the project and project flow.
11. Exceptions for all functions and methods with try catch block
12. Highlight he reusable codes wherever applicable
13. ID must be Auto-generated numbers, value type “”read only”

## HTML

1. Document Type should be defined as

<!DOCTYPE html>

1. Form fields must always include a <label> element with a "for" attribute matching the "id".

<label for="field-email">email</label>

<input type="email" id="field-email" name="email" value="" />

1. Each form field should have "name" attribute.
2. All tag names and attributes must be written in lowercase.

## SCSS

1. Reuse styles wherever applicable. Potential areas are Application Name, Page Title, Form Fields and Buttons
2. Selector names should be in lower case with words separated by hyphen.

           .page-title {

         margin: 20px;

     }

1. Selector class names should be relevant to the purpose of the element where the style is applied. For example if common style needs to be applied for all text boxes in an application.
2. Properties should be followed by a colon and a space
3. Properties and values should be in lower case
4. Use hexadecimal code for color definition

Example:

.login-form {

   background: #fdfdfd;

    display: block;

    margin: 0;

    margin-left: 20px;

}

## TYPESCRIPT

1. Use camelCase for variables and functions.

var firstName = "John";

function alert() {

    alert("This is an alert message.");

}

1. All variable names start with a letter.
2. Always put a space before and after an operator.

var profit = sellingPrice - buyingPrice;

1. Always end a statement with semicolon.
2. Open curly braces should be in the same line.
3. All lines following open curly braces should have one level of indent
4. Closing curly braces should be in next line with reduction of indent by one level

Example

if (time < 20) {

    greeting = "Good day";

} else {

    greeting = "Good evening";

}

***WISH YOU ALL THE BEST ☺***