System Requirements Specification Index

For

Supplier Information Management System

Version 4.0

IIHT Pvt. Ltd.
IIHT Ltd, No: 15, 2nd Floor, Sri Lakshmi Complex, Off MG Road, Near SBI LHO,
Bangalore, Karnataka – 560001, India
fullstack@iiht.com

SUPPLIER INFORMATION MANAGEMENT SYSTEM

System Requirements Specification

1.Business-Requirement:

1.1 Problem Statement:

Supplier management system: Develop Supplier Management Details application using in C#, .NET Core 3.1 WebApi with MS Sql Server. Implements all the checks so that there are no errors when Records are added, removed, updated, searched from collections and use Entity Framework.

1.2 Following is the requirement specification:

	Supplier Information Management Application
Supplier module	Adding and inserting items to a collection for Supplier data like ID, Name, phone, Contact Person, Address, etc.
	Removing items from a collection for Supplier data by ID.
	Finding, searching items for Supplier data like search by ID, name,role or department.
	Replacing items when Supplier data is updated.
Product Module	Using Entity Framework to manipulate Product data includes adding, removing, finding,and inserting data in database,
	Adding and inserting items for Product contains data like Productid, Productname,Price,Quantity.

2. Assumptions, Dependencies, Risks / Constraints

2.1 Common Constraints

- Develop application Supplier Management System using Collections, Classes, Exception handling, in C#, .NET Core 3.1 WebApi.
- Define appropriate classes and objects for a given scenario.
- Build the application using C# New Features like Default interface methods Nullable reference types.
- Using Entity Framework to manipulate Supplier data includes adding, removing, finding,and inserting data
- Use custom exceptions and other built-in exceptions like Database Exceptions, NullReferenceException at required places in applications.
- Create the Entity context class to connect the database and use appropriate methods to execute the CRUD operations for the Supplier.
- Do not change, add, remove any existing methods in service layer
- In Repository interfaces, custom methods can be added as per requirements.
- All RestEndpoint methods and Exception Handlers must return data wrapped in **ResponseEntity**

3. Business Validations

3.1 SupplierData Class Specifications:

```
public int Supplier ID { get; set; }
public string Supplier company_Name { get; set; }
public string Contact person { get; set; }
public string Email { get; set; }
public int Phone number { get; set; }
public string  Address { get; set; }
```

3.2 ProductData Class Specifications:

```
public int Product ID { get; set; }
public string ProductName { get; set; }
public int Price { get; set; }
public string Quantity { get; set; }
public int  SupplierId { get; set; }
```

4. Rest Endpoints

Rest End-points to be exposed in the controller along with method details for the same to be created

4.1 SupplierController

URL Exposed		Purpose
/Supplier/Add-Supplier		Add a Supplier
Http Method	POST	
Parameter 1	SupplierViewModel	
Return	Http Status Code	
/Supplier/Update-Supplier		Update a Supplier
Http Method	PUT	
Parameter 1	SupplierViewModel	
Return	Http Status Code	
/Supplier/All-Suppliers		Fetches the list of all Suppliers
Http Method	GET	
Parameter 1	-	
Return	<IEnumerable<SupplierData>>	
/Supplier/Get-Supplier/{SupplierId}		Fetches the details of a Supplier
Http Method	GET	
Parameter 1	int(SupplierId)	
Return	<SupplierData>	
/Supplier/Delete-Supplier/{SupplierId}		Delete a Supplier
Http Method	DELETE	
Parameter 1	int(SupplierId)	
Return	Http Status Code	
/Product/Add-Product		Add a Product
Http Method	POST	
Parameter 1	ProductViewModel	
Return	Http Status Code	
/Product/Update-Product/{ProductId}		Update a Product
Http Method	PUT	
Parameter 1	int(ProductId)	
Return	Http Status Code	
/Product/Delete-Product/{ProductId}		Delete a Product
Http Method	DELETE	

Parameter 1	int(ProductId)	
Return	Http Status Code	
/Product/Get-Product/{ProductId}		Get a Product by id.
Http Method	GET	
Parameter 1	int(ProductId)	
Return	<ProductData>	
/Product/All-Products		Get all products.
Http Method	GET	
Parameter 1	-	
Return	<IEnumerable<ProductData>>	

5. Template Code Structure

5.1 Package: SupplierManagement

Resources

Names	Resource	Remarks	Status
Package Structure			
controller	SupplierController	Controller class to expose all rest-endpoints for auction related activities.	Partially implemented
Startup.cs	Startup CS file	Contain all Services settings and SQL server Configuration.	Already Implemented
Properties	launchSettings.json file	All URL Setting for API	Already Implemented
	appsettings.json	Contain connection string for database	Already Implemented

5.2 Package: SupplierManagement.BusinessLayer

Resources

Names	Resource	Remarks	Status
Package Structure			
Interface	ISupplierServices interface IProductServices interface	Inside all these interface files contains all business validation logic functions.	Already Implemented
Service	Supplier Services CS file Product Services CS file	Using this all class we are calling the Repository method and use it in the program and on the controller.	Partially Implemented
Repository	ISupplierRepository SupplierRepository IProductRepository ProductRepository CS file and interface.	All these interfaces and class files contain all CRUD operation code for the database. Need to provide implementation for service related functionalities	Partially Implemented
ViewModels	SupplierViewModel, ProductViewModel	Contain all view Domain entities for show and bind data. All the business validations must be implemented.	Partially Implemented

5.3 Package: SupplierManagement.dataLayer

Resources

Names	Resource	Remarks	Status
Package Structure			
DataLayer	SupplierDbContext cs file	All database Connection and collection setting class	Already Implemented

5.4 Package: SupplierManagement.Entities

Resources

Names	Resource	Remarks	Status
Package Structure			
Entities	Supplier,Product	All Entities/Domain attribute are used for pass the data in controller Annotate this class with proper annotation to declare it as an entity class with **Id** as primary key. Generate the **Id** using the **IDENTITY** strategy	Partially Implemented

5.5 Package: SupplierManagement.Tests

Resources

The SupplierManagement.Tests project contains all test case classes and functions for code evaluation. Don't edit or change anything inside this project.

6. Execution Steps to Follow

1. All actions like build, compile, running application, running test cases will be through Command Terminal.

2. To open the command terminal the test takers need to go to the Application menu (Three horizontal lines at left top) Terminal → New Terminal.

3. On command prompt, cd into your project folder (cd <Your-Project-folder>).

4. To connect SQL server from terminal:
   (SupplierManagement /sqlcmd -S localhost -U sa -P pass@word1)
   - To create database from terminal -

     ```
     1> Create Database SupplierManagement_Db
     2> Go
     ```

5. Steps to Apply Migration(Code first approach):
   - Press Ctrl+C to get back to command prompt
   - Run following command to apply migration-
     (SupplierManagement /dotnet-ef database update)

6. To check whether migrations are applied from terminal:
   (SupplierManagement /sqlcmd -S localhost -U sa -P pass@word1)

   ```
   1> Use SupplierManagement_Db
   2> Go
   1> Select * From __EFMigrationsHistory
   2> Go
   ```

7. To build your project use command:
   (SupplierManagement /dotnet build)

8. To launch your application, Run the following command to run the application:
   (SupplierManagement /dotnet run)

9. This editor Auto Saves the code.

10. To test any Restful application, the last option on the left panel of IDE, you can find ThunderClient, which is the lightweight equivalent of POSTMAN.

11. **To test web-based applications on a browser, use the internal browser in the workspace. Click on the second last option on the left panel of IDE, you can find Browser Preview, where you can launch the application.**

    Note: The application will not run in the local browser

12. **To run the test cases in CMD, Run the following command to test the application: (SupplierManagement /dotnet test --logger "console;verbosity=detailed")**
    (You can run this command multiple times to identify the test case status,and refactor code to make maximum test cases passed before final submission)

13. **If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page) then you need to use CTRL+Shift+B - command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.**

14. **These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.**

15. **You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.**

HEALTH CLUB

IIHT

Time To Complete: 10 to 12 hr

Contents

1 Problem Statement

Health Club is SPA (Single Page Application) for placing a request for appointment with personal physical trainer, manage appointments and dropping a query

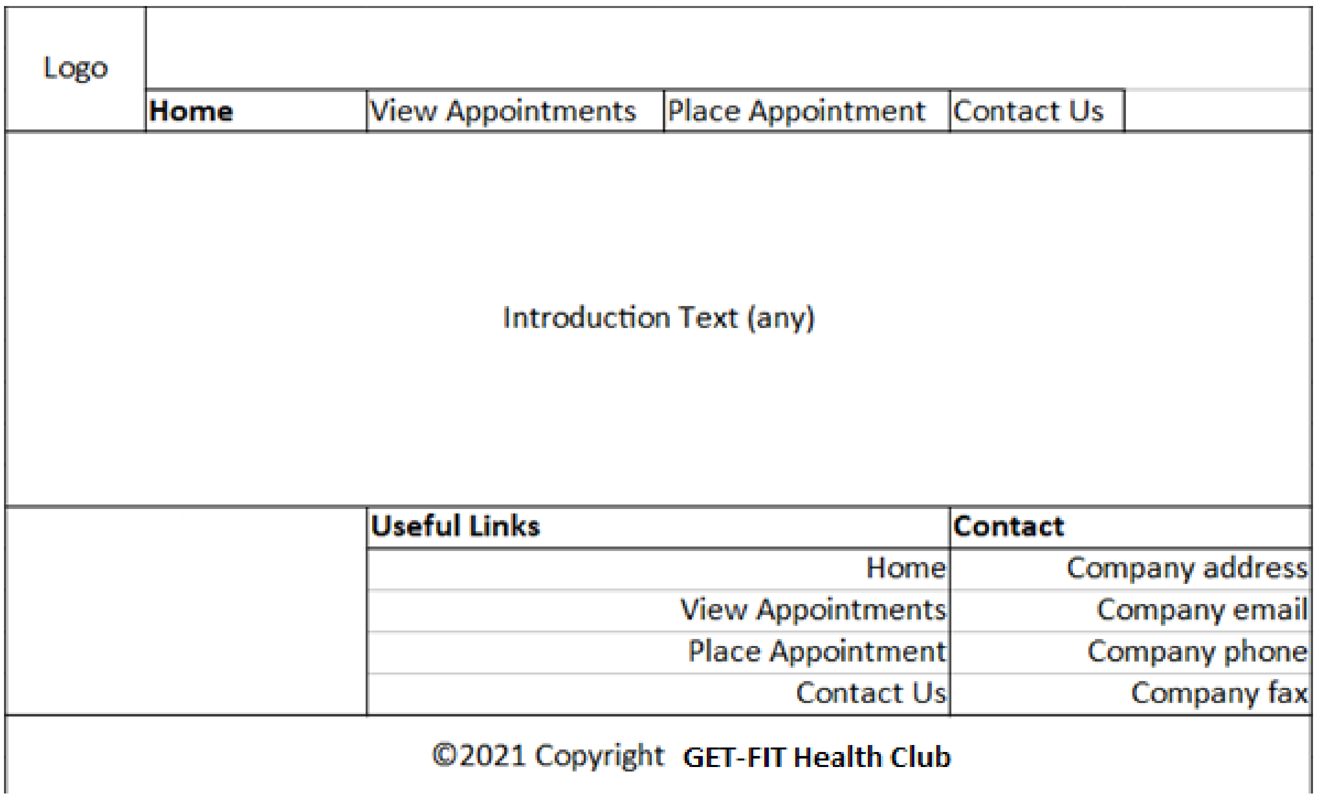
The core modules of Health Club app are:

1. Welcome Page
2. Apply for an appointment
3. View Appointment
4. Query

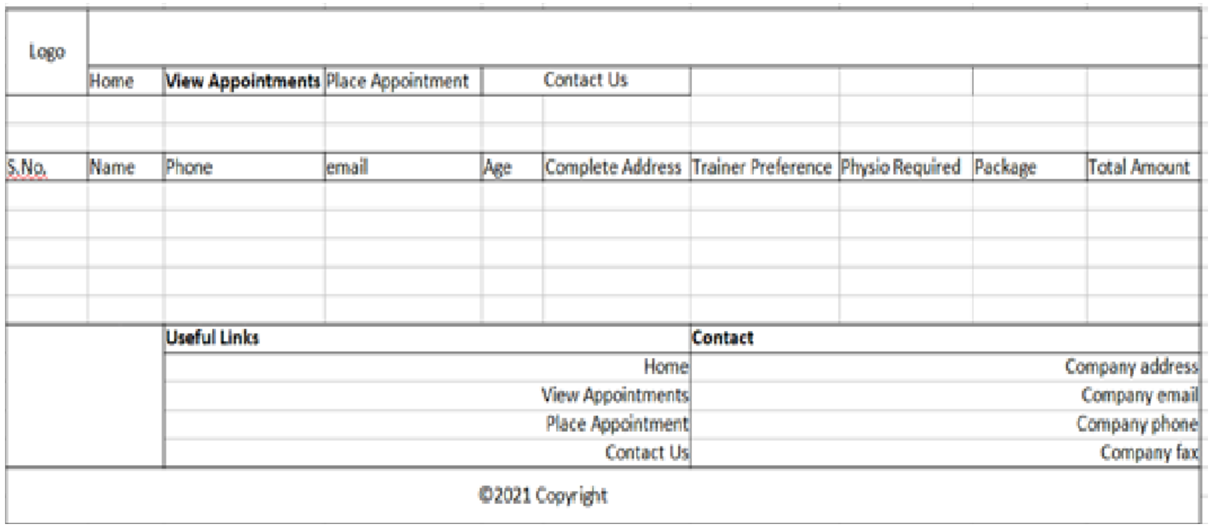
2 Proposed Health Club Wireframe

UI needs improvisation and modification as per given use case and to make test cases passed.

2.1 Welcome page



2.2 View Appointments

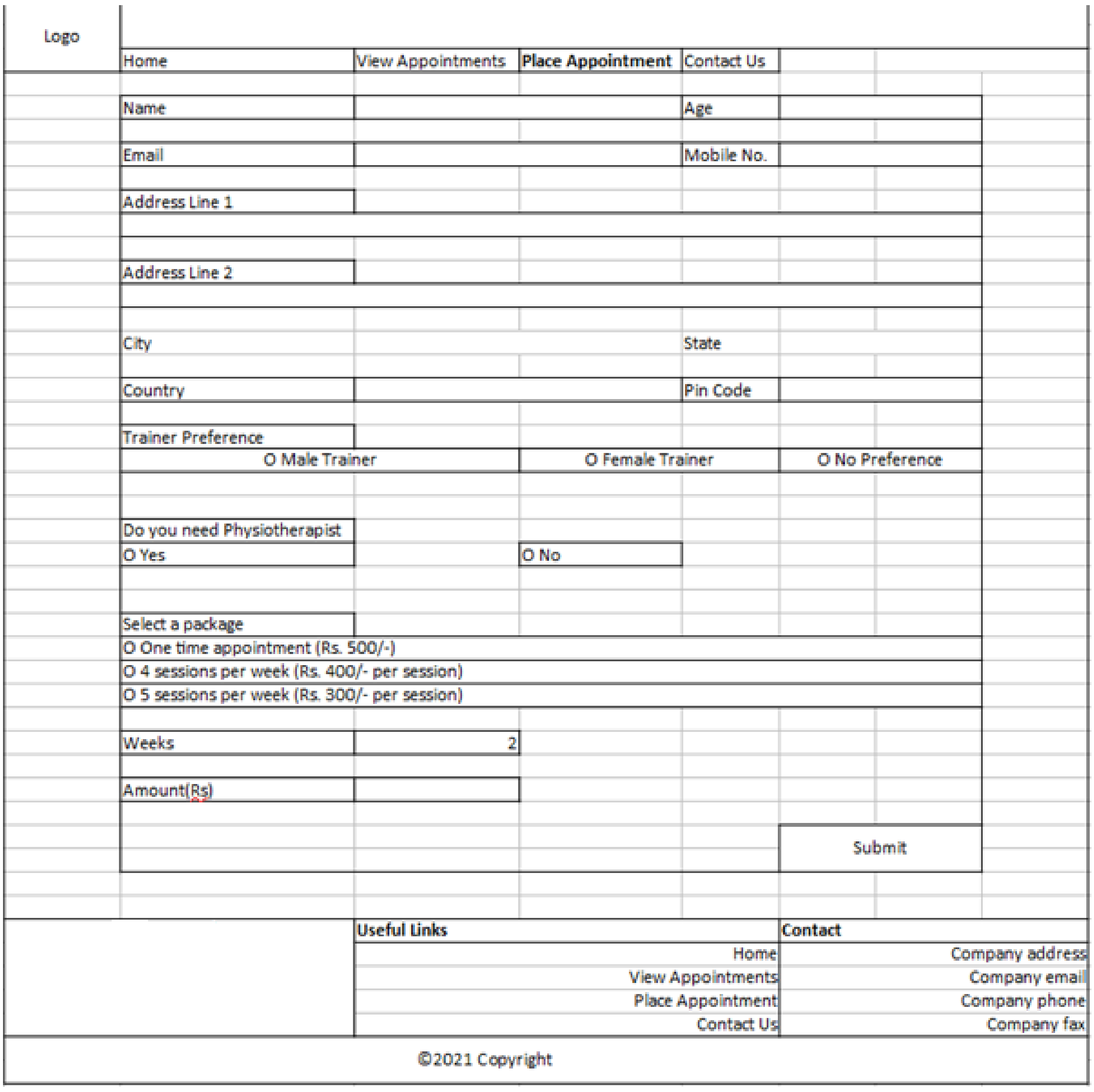


Logo									
	Home	**View Appointments**	Place Appointment	Contact Us					
S.No.	Name	Phone	email	Age	Complete Address	Trainer Preference	Physio Required	Package	Total Amount

	Useful Links	**Contact**
	Home	Company address
	View Appointments	Company email
	Place Appointment	Company phone
	Contact Us	Company fax

©2021 Copyright

2.3 Place Appointment



Logo

Home	View Appointments	**Place Appointment**	Contact Us

Name		Age	
Email		Mobile No.	
Address Line 1			
Address Line 2			
City		State	
Country		Pin Code	

Trainer Preference

O Male Trainer | O Female Trainer | O No Preference

Do you need Physiotherapist

O Yes | O No

Select a package

O One time appointment (Rs. 500/-)

O 4 sessions per week (Rs. 400/- per session)

O 5 sessions per week (Rs. 300/- per session)

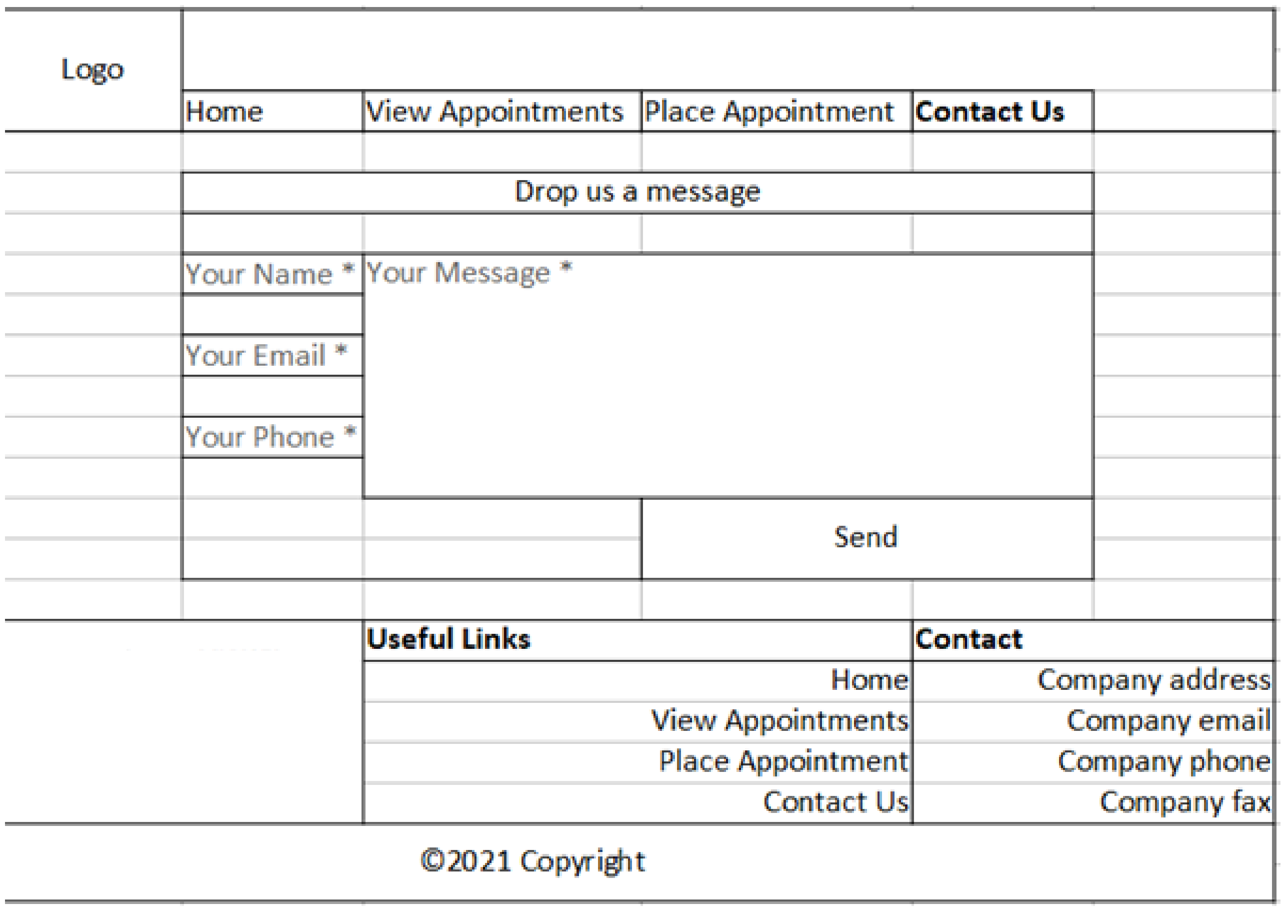
Weeks	2
Amount(Rs)	

Submit

Useful Links	**Contact**
Home	Company address
View Appointments	Company email
Place Appointment	Company phone
Contact Us	Company fax

©2021 Copyright

2.4 QUERY


Logo
Home
View Appointments
Place Appointment
Contact Us
Drop us a message
Your Name *
Your Message *
Your Email *
Your Phone *
Send
Useful Links
Contact
Home
Company address
View Appointments
Company email
Place Appointment
Company phone
Contact Us
Company fax
©2021 Copyright

3 Business-Requirement:

As an application developer, develop the Health Club App (Single Page App) with below guidelines:

User Story #	User Story Name	User Story
US_01	Welcome Page	As a user I should be able to visit the welcome page as default page. Acceptance criteria: 1. User can click any button given in the menu bar.
US_02	Post Appointment	As a user I should be able to post an appointment Acceptance criteria: 1. As a user I should be able to furnish following details at the time of placing an appointment 1.1 Name 1.2 Age 1.3 Email 1.4 Mobile No 1.5 Address Line 1 1.6 Address Line 2 1.7 City 1.8 State 1.9 Country 1.10 Pin Code 1.11 Trainer Preference 1.12 Physiotherapist requirement (Yes or No) 1.13 Select a package 1.14 Weeks 1.15 Amount 2. Weeks number type input box should be visible when 2nd or 3rd package option is selected. 3. If physiotherapist is required add additional 200/- in final amount

		4. Amount should be disabled and should be calculated automatically based on the selected package. 2. All details fields must be mandatory. 3. Address line 2 may contain the same address as address line 1. 4. Email& Mobile must be unique. 5. If any constraint is not satisfied, a validation message must be shown. 6. A success or failure message should be visible after the submit button is clicked.
US_03	Manage Appointment	1. As a user I should be able to view all appointment requests, and after selecting any appointment Acceptance criteria: 1. Message should be visible if no appointment is available to show.
US_04	Query	As a user I should be able to post a feedback/query/message Acceptance criteria: 1. As a user I should be able to furnish following details at the time of filling contact us form a. Name b. Email c. Phone d. Message 2. Message should not go beyond 200 characters. 3. All four fields must be mandatory. 4. A success or failure message should be visible after the submit button is clicked.

4 Constraints

1. On the page load, input focus must come to the first name input field.
2. You should be able to press the “TAB” key and “SHIFT + TAB” to navigate from top field to bottom field and vice-versa.
3. On click of “Submit” button, appointment details must be saved via fake-rest API in health-club.json.
4. Fake rest api is implemented with json-server.

Example JSON for reference of fields to be used for placing appointment:

```
{
    "firstname": "test",
    "lastname": "test",
    "age": 24,
    "phonenumber": 9988776655,
    "email": "test@test.com",
    "streetaddress": "test",
    "city": "test",
    "state": "test",
    "country": "india",
    "pincode": 560058,
    "trainerpreference": "Male Trainer",
    "physiotherapist": "Yes",
    "packages": "500",
    "inr": 1000,
    "paisa": 10,
    "id": 1
  }
```

5 Mandatory Assessment Guidelines

1. All actions like build, compile, running application, running test cases will be through Command Terminal.
2. To open the command terminal the test takers, need to go to
   Application menu (Three horizontal lines at left top) -> Terminal ->New Terminal.
3. This editor Auto Saves the code.
4. If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page) then you need to use CTRL+Shift+B-command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in next login.
5. These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.
6. To test any Restful application, the last option on the left panel of IDE, you can find ThunderClient, which is the lightweight equivalent of POSTMAN.
7. This is a web-based application, to run the application on a browser, use the internal browser in the workspace. Click on the second last option on the left panel of IDE, you can find Browser Preview, where you can launch the application.
   Note: The application will not run in the local browser
8. You can follow series of command to setup Angular environment once you are in your project-name folder:
   a. npm install -> Will install all dependencies -> takes 10 to 15 min
   b. npm run start -> To compile and deploy the project in browser. You can press <Ctrl> key while clicking on localhost:4200 to open project in browser -> takes 2 to 3 min
   c. npm run json-server -> to deploy fake rest api created with json-server -> takes 10 to 15 seconds
   d. npm run jest -> to run all test cases. It is mandatory to run this command before submission of workspace -> takes 5 to 6 min

9. **You may also run "npm run jest" while developing the solution to re-factor the code to pass the test-cases.**
10. **Once you are done with development and ready with submission, you may navigate to the previous tab and submit the workspace. It is mandatory to click on "Submit Assessment" after you are done with code.**
11. **You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.**