System Requirements Specification Index

For

My Time Away Application

Version 1.0

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

MY TIME AWAY SYSTEM

System Requirements Specification

1.BUSINESS-REQUIREMENT:

1.1 PROBLEM STATEMENT:

My Time Away Application is a simple .Net Core 3.1 RESTFUL Web API application with MS SQL server. "My Time Away" is a comprehensive employee leave management application designed to streamline and simplify the process of managing employee leave requests within an organization. This application provides an efficient and user-friendly platform for both employees and managers to handle leave requests, ensuring a seamless workflow while maintaining compliance with company policies.

1.2 FOLLOWING IS THE REQUIREMENT SPECIFICATION:

	My Time Away Application
1	Employee
Employee Leave Module Functionalities	
	1.Create a leave
	2.Can delete a leave
	3.Get leave Info by id
	4.Fetch all leaves
	5.Update leave details
	6.Cancel leave
	7.Reject leave
	8.Approve leave
	9.Search leave by employee id, employee name, days

2. ASSUMPTIONS, DEPENDENCIES, RISKS / CONSTRAINTS

2.1 Employee Leave Constraints

- While adding a leave details, if leave is already existing, it should throw a custom exception
- While deleting the leave, ensure that leave already exists, if not, the operation should throw a custom exception

2.3 Common Constraints

- For all rest endpoints receiving @RequestBody, validation check must be done and must throw custom exception if data is invalid
- All the database operations must be implemented on entity object only
- 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

- **All business logic CRUD operations under repository class and write your business logic validation in Services class and related validation use proper user defined exceptions mentioned in above document.**
- **Controller must validate before processing any logic on the database.**

2.4 Visitors can perform the follow actions

- Allows to add a new leave
- Allows to delete an existing leave
- Allows to update leaves details
- Allowd
- Allows to search the leave on the basis of name,id,days
- Allows to display all leaves
- Allows to cancel leave
- Allows to reject leave
- Allows to cancel leave

2.5 ToolChain

- .NET Core 3.1, RESTful Web API, MS SQL Server.

3. Business Validations

3.1 Employee Leave Class Entities

- Id (long) must be not null and unique, key attribute
- Employee Id (string) is not null, min 3 and max 100 characters.
- Employee Name (string) is not null, min 3 and max 100 characters.
- Employee Email (string) is not null, min 3 and max 100 characters.
- Employee phone (string) is not null, min 3 and max 100 characters.
- Manager Email (string) is not null, min 3 and max 100 characters.
- From Date (DateTime) is not null.
- To Date (DateTime) is not null.
- Reason (string) is not null, min 3 and max 100 characters.
- Total Days (int) is not null, min 3 and max 100 characters.
- Is Processed (bool) is not null.
- Status (string) is not null, min 3 and max 100 characters.

4. Considerations

- For Role of application users must be used: -
    1. Employee

5. Rest Endpoints

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

5.1 DepartmentController

URL Exposed		Purpose
/create-leave		Add new leave Details
Http Method	Post	

Parameter 1	EmployeeLeave model	
Return	HttpResponse status code	
/update-leave		Update existing leave Details
Http Method	PUT	
Parameter 1	EmployeeLeaveView Model model	
Return	HttpResponse status code	
/delete-leave		Delete leave with given id.
Http Method	DELETE	
Parameter 1	Long id	
Return	HttpResponse status code	
/get-leave-by-id		Fetches the leave Details with the given id
Http Method	GET	
Parameter 1	Long id	
Return	<EmployeeLeave>	
/get-all-leaves		Fetches all the leaves Details
Http Method	GET	
Parameter 1	-	
Return	<<IEnumerable<EmployeeLeave>>	
/search-leaves		Fetches the leave Details with the given id, name and days
Http Method	GET	
Parameter 1	String employeeId	
Parameter 2	String employeeName	
Parameter 3	Int totalDays	
Return	<EmployeeLeave>	
/cancel-leave		Cancel the leave with the given id
Http Method	PUT	
Parameter 1	Long id	
Return	<EmployeeLeave>	
/reject-leave		Reject the leave with the given id
Http Method	PUT	
Parameter 1	Long id	
Return	<EmployeeLeave>	
/approve-leave		Approve the leave with the given id
Http Method	PUT	
Parameter 1	Long id	
Return	<EmployeeLeave>	

6. TEMPLATE CODE STRUCTURE

6.1 Package: MyTimeAway

Resources

Names	Resource	Remarks	Status
Package Structure			
controller	EmployeeLeave Controller	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

6.2 Package: MyTimeAway.BusinessLayer

Resources

Names	Resource	Remarks	Status
Package Structure			
Interface	IEmployeeLeaveService, interface	Inside all these interface files contains all business validation logic functions.	Already Implemented
Service	EmployeeLeaveService 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	IEmployeeLeaveRepository EmployeeLeaveRepository 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	EmployeeLeaveViewModel,	Contain all view Domain entities for show and bind data.	Already Implemented

		All the business validations must be implemented.	

6.3 Package: MyTimeAway.DataLayer

Resources

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

6.4 Package: MyTimeAway.Entities

Resources

Names	Resource	Remarks	Status
Package Structure			
Entities	EmployeeLeave, CS file	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	Already Implemented

6.5 Package: MyTimeAway.Tests

Resources

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

7. 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:**
   (MyTimeAway /sqlcmd -S localhost -U sa -P pass@word1)
   - **To create database from terminal -**

     1> Create Database MyTimeAwayDb
     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-**
     (MyTimeAway /dotnet-ef database update)

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

   1> Use MyTimeAwayDb
   2> Go
   1> Select * From __EFMigrationsHistory
   2> Go

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

8. **To launch your application, Run the following command to run the application:**
   (MyTimeAway /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: (MyTimeAway /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.**
