System Requirements Specification Index

For

Interview Tracker Application (Collaborative)

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

TABLE OF CONTENTS

1.Business-Requirement:

1.1 Problem Statement:

The purpose of this application is to allow the official staff to register, Update, delete, show all interviewees by using **user API**. On the other hand, Dashboard **API** allows you to view all Interviews, search Interview by name or interviewer name, add, Delete, Update interview and Using **Interview API** you can Add New Interview and count total number of interviews.

1.2 Following is the requirement specification:

	Interview Tracker
USERS	
1	Admin/Interviewer
2	User/interview
Interview/User Functionalities	
1	Can register itself (*Contact Details (Contact Address, mobile, email, Password)*
2	Can apply for Interview after Registration/Login.
	While applying for interview, following information is required
	a. Name
	b. Skills Type (Technology Stack like - .net, Java, Angular etc)
	c. Experience in relevant technology.
	d. Interview Date (based on user available)
	e. Interview Time (based on user available)
3	Can track status of interview schedule or not
Admin/Interviewer Functionalities	
1	Can Disable/Enable User/Interview
2	Can list all interview schedule for today based on date
3	Admin can register new interviewers.
4	If the interview is scheduled by an admin or interviewer, following information should be furnished based on user input and seeking time with the interview.
	a. Interviewer Name
	b. Type of interview (Interview Name)
	c. Interview Name (User Name)
	d. User Skills (Skills Type)
	e. Interview Date (based on user available) – if new scheduled
	f. Interview Time (based on user available) – if new scheduled
	g. Status of Interview – Enum type – Inprocess, Cancel, Done, Pass, Fail

	h. User Type must be select from User Enum List.
5	Can search interview by interview name of interviewer name
6	*Can Delete the interview if not relevant.*
7	*Can Update an interview if reschedule required.*

2. Assumptions, Dependencies, Risks / Constraints

2.1 User/Interview Constraints

- User controllers must be accessible by only Authorized users.
- While registering users check all user details, if not provide operation should throw custom exception.
- Users must be registered with their email Id and password provided. If not, throw a custom exception.
- While applying for an interview by user, if all information does not exist then the operation should throw a custom exception.
- While applying for an interview, the user must be logged in first, if use is disabled then operation should throw a custom exception.
- While fetching interview status, if Interview id does not exist then operation should throw custom exception.
- While fetching interview details of a user, if user id does not exist or user disable then operation should throw custom exception.

2.2 Admin/Interviewer Constraints

- Admin/Interviewer controller must be accessible by only Authorize Admin/Interviewer.
- While scheduling an interview all information must be provided, if not then operation should throw a custom exception or core exception.
- Based on user login function should be worked on if admin function works for admin and if there is user function based on user should work, if any other user tries to access controller should throw custom exception.
- Before schedule any interview User and Interviewer must be registered first. If not, a custom exception user or interviewer does not exist.
- While updating and deleting any interview interviewer is disable or ID does not exist, throw custom exceptions for user and interviewer that do not exist.
- If any interview is applied by user interview status should be in – Inprocess from Enum list value.
- If any interview is canceled by admin or interviewer status should be in – Cancel from Enum list value.
- If any user passes in an interview interview status should be in – Pass from Enum list value.
- If any user fails in an interview interview status should be in – Fail from Enum list value.

- If any all processes are completed in interview interview status should be in – Done from Enum list value.
- User type must be selected while updating interview information from Enum List. If not selected, then throw a custom exception.

2.3 Common Constraints

- All users must be registered as basic users and after Admin can make them different types of user based on Role.
- All users must be registered as UserEnabled to use this application later manager can disable or enable.
- For all controller receiving @RequestBody, validation check must be done and must throw custom exception if data is invalid
- All the business validations must be implemented in model classes only.
- All the database operations must be implemented on entity objects only with code first approach.

3. Business Validations on class

3.1 Interview_Master Entity

- InterviewId Int is not null, primary key
- InterviewName string is not null, min 3 and max 100 characters.
- Interviewer List of interviewers that is registered as interviewer is not null, min 3 and max 100 characters.
- Interview User is not null (Select list of user who appear in interview)
- UserSkills enum value is not null
- InterviewDate DateTime value is not null
- InterviewTime DateTime value not null.
- Interview Status enum value is not null
- UserType Enum value not null.
- Interview Status enum value is not null
- ContactAddress string is not null
- Phone long is not null, min 10 and max 12 characters.
- Email string is not null, valid format
- CreatedOn auto update date and time.

3.2 User_Master Entity

- UserId int is not null, primary key
- Name string is not null, min 3 and max 100 characters.
- Email string is not null, valid format
- ContactNumber long is not null, min 10 and max 12 characters.
- Address string is not null, min 10 and max 100 characters.
- IdProofType string value is not null
- IdProofNumber string is not null
- Skills Type Enum Value.

- Experience double not null.
- Interview Date not null
- Interview Time not null
- UserType enum value – **Only Used for admin this property**
- Enabled bool.

4. Considerations

A. For Role of application users 2 possible values must be used: -
   1. User/Interview
   2. Admin/Manager

B. For InterviewStatus of Interview following 4 possible Enum values must be used.

1. Inprocess = 1,
2. Cancel = 2,
3. Fail = 3,
4. Pass =4,
5. Done = 5

C. For UserSkills of UserMaster following possible Enum values must be used.

1. .net = 1
2. Java = 2
3. Angular = 3 and many more..

D. For UserType of Interview and UserMaster following possible Enum values must be used.

1. Trainee = 1 2. TraineeIntern = 2 3. Manager = 3 4. Developer = 4 5. FresherAssociate = 5 6. TeamLead = 6 7. TeamSupervisor = 7 8. DeliveryLead = 8 9. ProductionLead = 9 10. NotAssigned = 10

5. Rest Endpoints

Rest End-points to be exposed in the controller along with method details for the same to be created with browser and swagger API tools.

5.1 DashboardController

URL Exposed		Purpose
1. /api/Dashboard		Get list of interview
Http Method	GET	
Parameter 1	-	
Return	<IEnumerable<Interview> >	
api/Dashboard/DeleteInterview/{InterviewId}		Delete an existing interview
Http Method	Delete	
Parameter 1	InterviewId	
Return	HttpStatus code	
api/Dashboard/Updateinterview/{InterviewId}		Update an existing interview
Http Method	PUT	
Parameter 1	interviewId	
Parameter 2	Interview model	
Return	HttpStatus Code	
api/Dashboard/Getinterview/{InterviewId}		Get an Interview by InterviewId
Http Method	GET	
Parameter 1	InterviewId	
Return	Interview object	
api/Dashboard/Searchinterview/{Name}		Update an existing product Category
Http Method	GET	
Parameter 1	Name (string)	
Return	HttpStatus Code	
api/Dashboard/TotalInterview		Get Total no of Interview in Collection
Http Method	GET	
Parameter 1	-	
Return	Count of interview	

5.2 UserController

URL Exposed		Purpose
api/user		Get list of register User
Http Method	GET	
Parameter 1	-	
Return	`<IEnumerable<ApplicationUser>>`	
api/user/RegisterUser		Register new user for application
Http Method	POST	
Parameter 1	`RegisterViewModel model`	
Return	HttpStatusCode	
api/user/DeleteUser/{UserId}		Delete an existing user
Http Method	DELETE	
Parameter 1	UserId	
Return	Http Status Code	
api/user/Updateuser/{UserId}		Update an existing user information
Http Method	HTTPPUT	
Parameter 1	UserId	
Parameter 2	ApplicationUser model	
Return	Http Status Code	

5.3 InterviewController

URL Exposed		Purpose
api/Interview		Get count of total interview
Http Method	GET	
Parameter 1	-	
Return	`Count of total interview`	
api/interview/AddInterview		Add new interview
Http Method	POST	
Parameter 1	AddInterviewViewModel model	
Return	Http Status code	

6. Template Code Structure

6.1 Package: InterviewTracker

Resources

Names	Resource	Remarks
Package Structure		
controller	User, Dashboard, Interview Controller	These all controller handle all application Function, update/Edit show information and login existing user.
Startup.cs	Startup CS file	Contain all Services settings and Db Configuration.
Properties	launchSettings.json file	All URL Setting for API

6.2 Package: InterviewTracker.BusinessLayer

Resources

Names	Resource	Remarks
Package Structure		
Interface	IInterviewTracker, IUserInterviewTracker Services interface	Inside all these cs files contains all business logic functions..
Service	InterviewTracker, UserInterviewTracker Servicesclass file	Using this all class we are calling the Repository method and use it in the program and on the controller.
Repository	IInterviewTracker, InterviewTracker, IUserInterviewTracker, UserInterviewTracker Repository CS file and interface.	All these interfaces and class files contain all CRUD operation code for MongoDb.
ViewModels	AddInterview, EditInterview, Interview, Register, UserEdit ViewModel Class file	Contain all view Domain entities for show and bind data.

6.3 Package: InterviewTracker.DataLayer

Resources

Names	Resource	Remarks
Package Structure		
DataLayer	Mongosettings, MongoDBContext, IMongoDBContext cs file	All MongoDb setting class

6.4 Package: InterviewTracker.Entiities

Resources

Names	Resource	Remarks
Package Structure		
Entities and Enum	ApplicationUser, Interview, InterviewStatus, Status, TechnicalInterviewStatus, UserType CS and Enum file	All Entities/Domain attribute

6.5 Package: InterviewTracker.Tests

Resources

Note: - Under the InterviewTracker.Tests contain All Test cases for code evaluation, please don't try to alter and edit it.

7. Execution Steps to Follow

- **All actions like build, compile, running application, running test cases will be through Command Terminal.**
- **To open the command terminal the test takers need to go to the Application menu (Three horizontal lines at left top) Terminal → New Terminal.**
- **On command prompt, cd into your project folder (cd <Your-Project-folder>).**
- **To build your project use command: (Project_directory/InterviewTracker/ dotnet build)**
- **To launch your application, Run the following command to run the application: (Project_directory/InterviewTracker / dotnet run)**
- **This editor Auto Saves the code.**
- **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.**
- **To test any UI based application the second last option on the left panel of IDE, you can find Browser Preview, where you can launch the application.**

- **To run the test cases in CMD, Run the following command to test the application:**
  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)

- **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.**

- **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.**

- **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.**