

AGILE TRACK SYSTEM

IIHT

Time To Complete: 2 hr

CONTENTS

1	Project Abstract	3
2	Problem Statement	3
3	Proposed Agile Track System Application Wireframe	4
3.1	Welcome Page	4
3.2	User Login	5
3.2	Admin Login	7
3	Business-Requirement	11
4	Validations	18
5	Constraints	18
6	Mandatory Assessment Guidelines	18

1 PROJECT ABSTRACT

In the rapidly evolving landscape of project management, the need for streamlined and accessible platforms to manage agile workflows has become increasingly important. With the vision of creating an innovative and user-friendly Agile Track System, the CEO of a growing software development company, Mr. X, assigns a team of developers to build an application using React.

This application aims to provide a user-friendly and efficient platform for both team members and administrators, enhancing the overall task management and team collaboration experience.

Your task is to develop a comprehensive digital solution that enables users to log in, view the tasks assigned to them, and track their progress within their respective scrum teams. Allows administrators to monitor and manage users, scrum teams, and tasks efficiently, fostering a collaborative and productive work environment. Provides detailed insights into task statuses, user assignments, and team performance, facilitating better decision-making and agile project management.

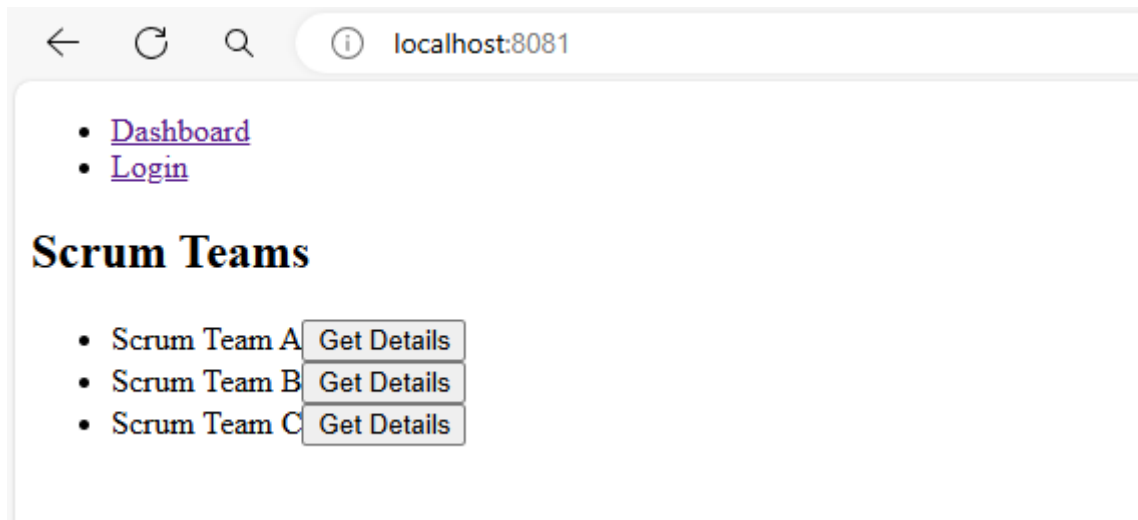
2 PROBLEM STATEMENT

The "**Agile Track System**" is a Single Page Application (SPA) designed to streamline task management and team collaboration within agile frameworks. This platform allows users and administrators to log in, view tasks assigned to users, and track progress across various scrum teams.

3 PROPOSED AGILE TRACK SYSTEM WIREFRAME

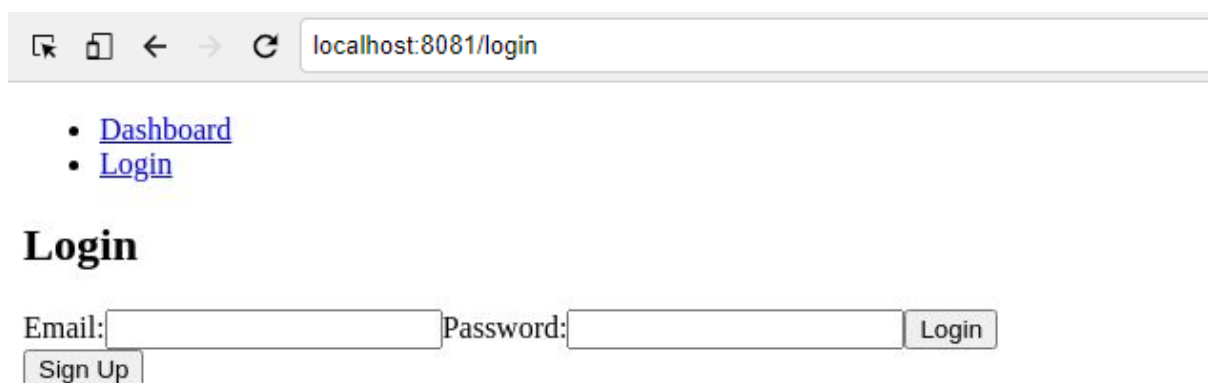
UI needs improvisation and modification as per given use case and to make test cases passed.

3.1 WELCOME PAGE

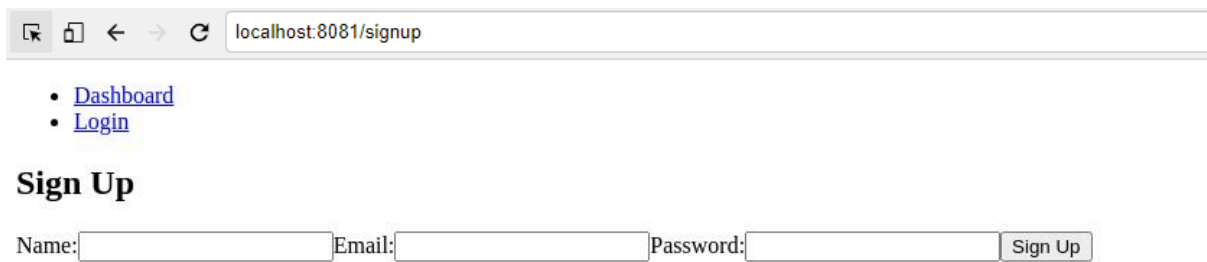


Already added users and admin with their details

*** Login Page***



*** Sign Up Page***



The screenshot shows a web browser window with the address bar displaying 'localhost:8081/signup'. Below the address bar, there are two links: 'Dashboard' and 'Login'. The main heading is 'Sign Up'. Below the heading, there are three input fields labeled 'Name:', 'Email:', and 'Password:', followed by a 'Sign Up' button.

• [Dashboard](#)

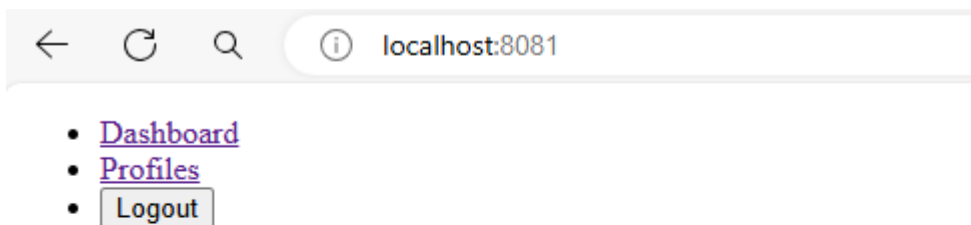
• [Login](#)

Sign Up

Name: Email: Password:

3.2 USER LOGIN

*** User Home Page***



The screenshot shows a web browser window with the address bar displaying 'localhost:8081'. Below the address bar, there are three links: 'Dashboard', 'Profiles', and 'Logout'.

• [Dashboard](#)

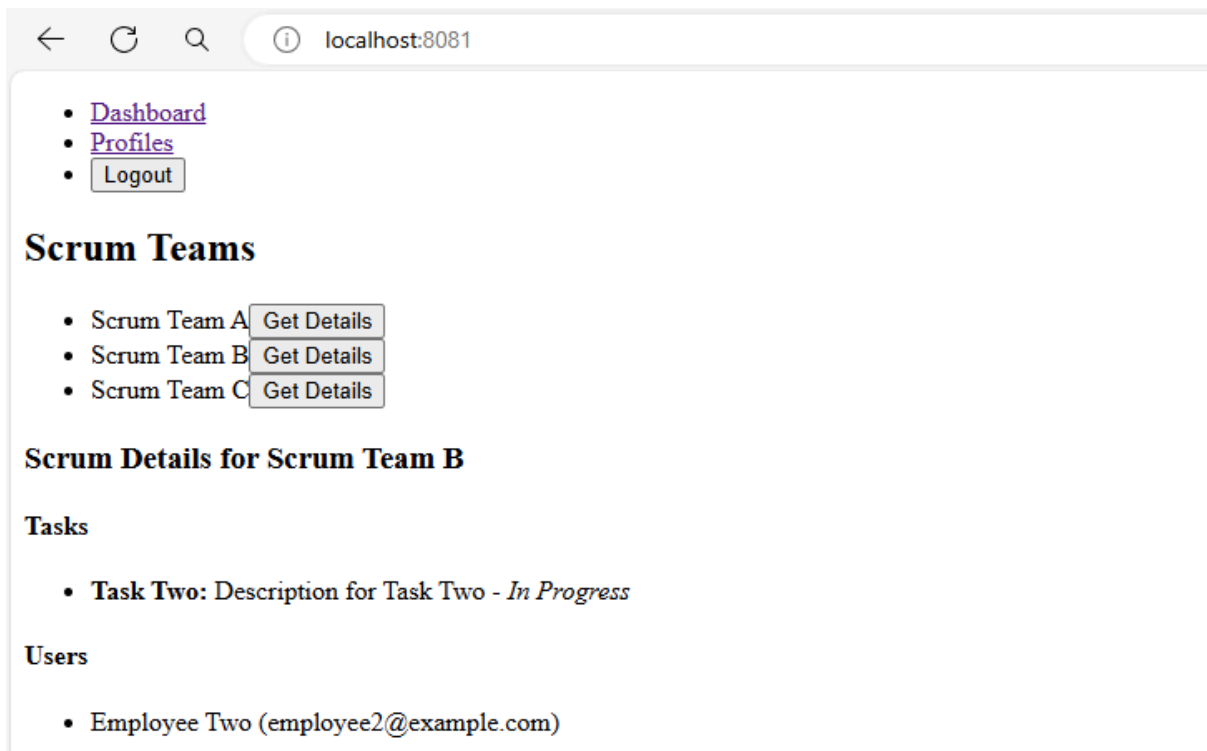
• [Profiles](#)

•

Scrum Teams

- Scrum Team A
- Scrum Team B
- Scrum Team C

*** Team Details***



• [Dashboard](#)

• [Profiles](#)

• Logout

Scrum Teams

- Scrum Team A Get Details
- Scrum Team B Get Details
- Scrum Team C Get Details

Scrum Details for Scrum Team B

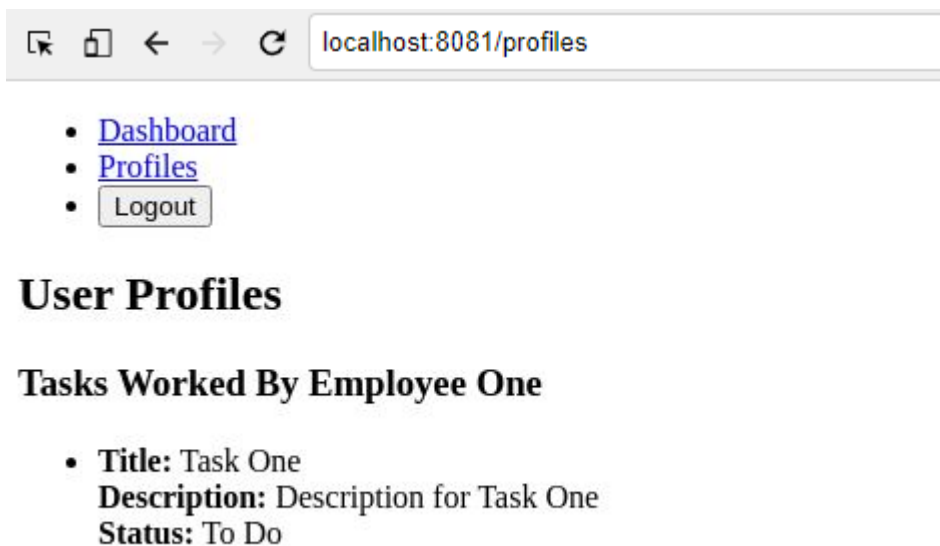
Tasks

- **Task Two:** Description for Task Two - *In Progress*

Users

- Employee Two (employee2@example.com)

*** User Profile Page***



• [Dashboard](#)

• [Profiles](#)

• Logout

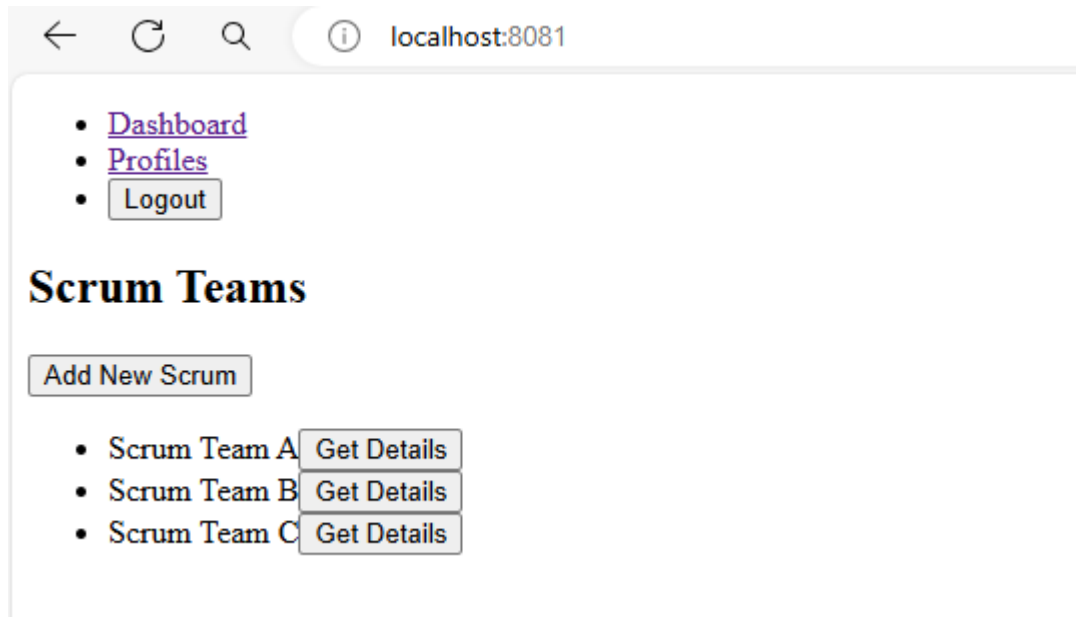
User Profiles

Tasks Worked By Employee One

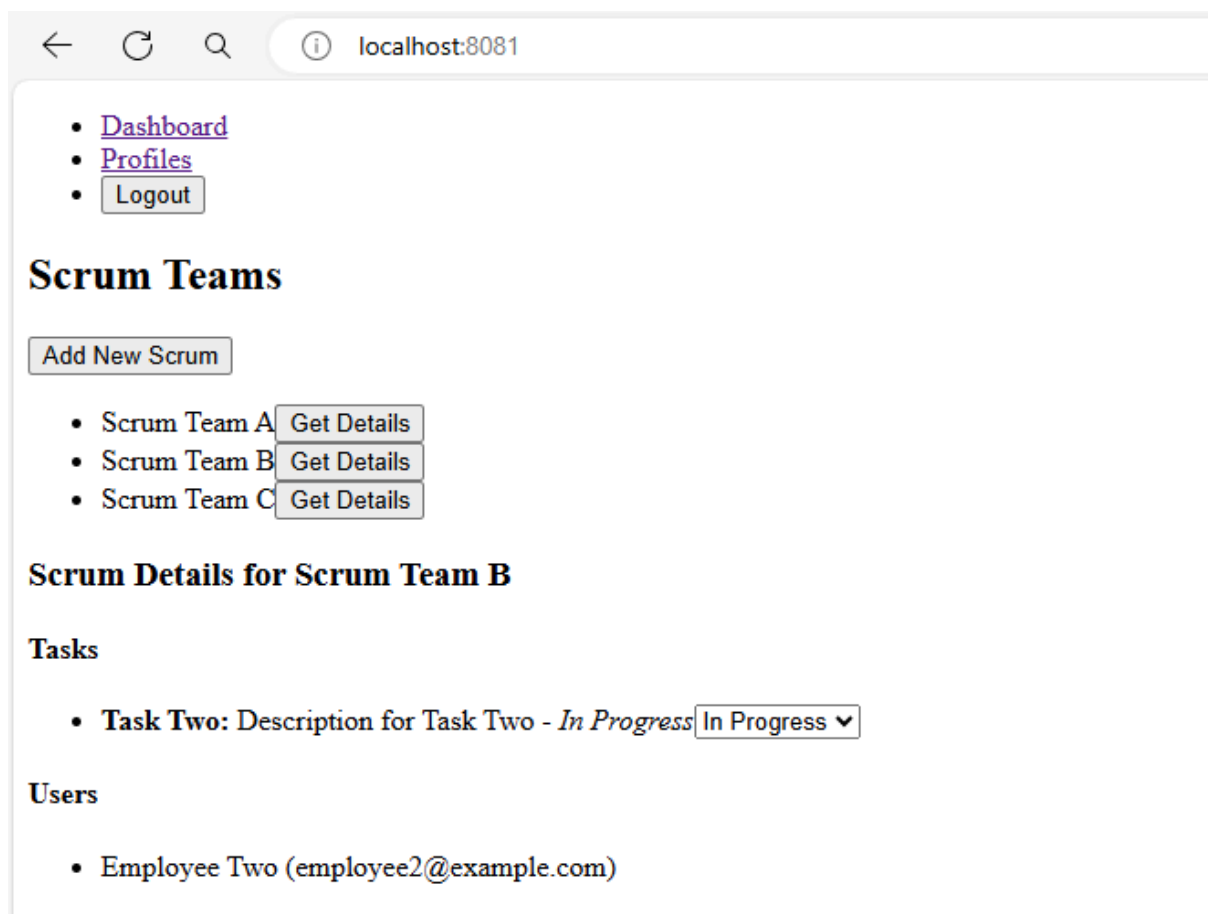
- **Title:** Task One
Description: Description for Task One
Status: To Do

3.3 ADMIN LOGIN

*** Admin Home Page***



*** Team Details***



*** Add New Scrum***

[←](#) [↺](#) [🔍](#) [ℹ](#) localhost:8081

- [Dashboard](#)
- [Profiles](#)
- [Logout](#)

Scrum Teams

[Cancel](#)
Scrum Name:
Task Title:
Task Description:
Task Status: To Do ▼
Assign To: Select a user ▼
[Create Scrum](#)

- Scrum Team A [Get Details](#)
- Scrum Team B [Get Details](#)
- Scrum Team C [Get Details](#)

*** Update Task Status***

[←](#) [↺](#) [🔍](#) [ℹ](#) localhost:8081

- [Dashboard](#)
- [Profiles](#)
- [Logout](#)

Scrum Teams

[Add New Scrum](#)

- Scrum Team A [Get Details](#)
- Scrum Team B [Get Details](#)
- Scrum Team C [Get Details](#)

Scrum Details for Scrum Team B

Tasks

- Task Two: Description for Task Two - *In Progress* In Progress ▼

Users

- Employee Two (employee2@example.com)

To Do
In Progress
Done

*** Profiles Page***



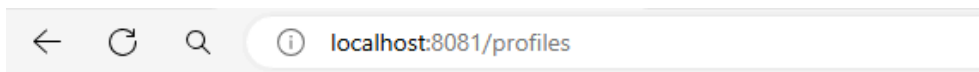
- [Dashboard](#)
- [Profiles](#)
- [Logout](#)

User Profiles

Add New User

- **Name:** Employee One
Email: employee1@example.com
[Get History](#)
- **Name:** Employee Two
Email: employee2@example.com
[Get History](#)
- **Name:** Employee Three
Email: employee3@example.com
[Get History](#)

*** Add New User***



- [Dashboard](#)
- [Profiles](#)
- [Logout](#)

User Profiles

[Cancel](#)

Name:

Email:

Password:

Role:

[Create User](#)

- **Name:** Employee One
Email: employee1@example.com
[Get History](#)
- **Name:** Employee Two
Email: employee2@example.com
[Get History](#)
- **Name:** Employee Three
Email: employee3@example.com
[Get History](#)

*** Get History of an User***

[←](#) [↻](#) [🔍](#) [localhost:8081/profiles](#)

- [Dashboard](#)
- [Profiles](#)
- [Logout](#)

User Profiles

[Add New User](#)

- **Name:** Employee One
Email: employee1@example.com
[Get History](#)
- **Name:** Employee Two
Email: employee2@example.com
[Get History](#)
- **Name:** Employee Three
Email: employee3@example.com
[Get History](#)

Tasks Worked By Employee Two

- **Title:** Task Two
Description: Description for Task Two
Status: In Progress
- **Title:** Task 3
Description: Description for Task 3
Status: In Progress

4 BUSINESS-REQUIREMENT:

As an application developer, develop the Agile Track System (Single Page App) with below guidelines:

User Story #	User Story Name	User Story
US_01	Welcome Page	<p>As a user I should be able to visit the welcome page as default page.</p> <p>Acceptance criteria:</p> <ol style="list-style-type: none">There should be 2 routes/links:<ul style="list-style-type: none">Dashboard: When clicking this link, it should display the title "Scrum Teams".Login: When clicking this link, it should navigate to the login page.Should have a title as "Scrum Teams".A list of scrum teams should be displayed, where each team includes:<ul style="list-style-type: none">The team name (e.g., Scrum Team A, Scrum Team B).A Get Details button for each team, when clicked should navigate to the login page. <p>** Kindly refer to the screenshots for any clarifications. **</p>
	Login Page	<p>As a user, I should be able to access the login page to log into the system.</p> <p>Acceptance criteria:</p> <ol style="list-style-type: none">Users can access the login page by clicking the Login link in the menu.The login page should have the title as "Login".The login page should have fields for entering an email and password.The system should validate the user credentials upon submission and provide appropriate feedback:<ul style="list-style-type: none">If the email address is invalid (e.g., missing '@'), it should display "Please include an '@'".If the credentials are wrong, it should display "Invalid email or password".Both fields are mandatory. If either field is not filled, an error message should be displayed indicating that the fields are required.

	Sign Up Page	<ol style="list-style-type: none"> 5. Fake REST APIs for logging in as a user and an admin are provided and should be used, as specified in the db.json file. 6. The login page should have a Login button to submit the login credentials. 7. After successful login, the user should be redirected to their respective home page (user or admin). 8. There should be a Sign Up button for users who don't have an account. When clicking this button, it should navigate to the sign-up page. <p>** Kindly refer to the screenshots for any clarifications. **</p> <hr/> <p>As a new user, I should be able to sign up for a new account.</p> <p>Acceptance criteria:</p> <ol style="list-style-type: none"> 1. Users can access the sign-up page by clicking the Sign Up button on the login page. 2. The sign-up page should have the title "Sign Up". 3. The sign-up page should have fields for entering a name, email, and password. 4. The system should validate the user inputs upon submission and provide appropriate feedback: <ul style="list-style-type: none"> ● If the email address is invalid (e.g., missing '@'), it should display "Please include an '@". ● All fields are mandatory. If any field is not filled, an error message should be displayed indicating that the fields are required. 5. The sign-up page should have a Sign Up button to submit the new account details and when clicked it should redirect to the home page as a user. <p>** Kindly refer to the screenshots for any clarifications. **</p>
--	--------------	---

US_02	User Page (Logged in as a user)	<p>As a user, I should be able to visit the welcome page as the default page after logging in to the Agile track system.</p> <p>Acceptance criteria:</p> <ol style="list-style-type: none"> There should be 2 routes/links: <ul style="list-style-type: none"> Dashboard: When clicking this link, it should display the title "Scrum Teams". Profiles: When clicking this link, it should navigate to the profiles page. There should be a Logout button, and when clicked, it should log the user out and navigate to the login page. <p>The above mentioned 2 routes/links & logout button should be throughout all the respective pages until logging out.</p> <ol style="list-style-type: none"> Should have a title as "Scrum Teams". A list of scrum teams should be displayed below the title, where each team includes: <ul style="list-style-type: none"> The team name (e.g., Scrum Team A, Scrum Team B) along with a Get Details button for each team should be available. <p>Get Details Button:</p> <p>Acceptance criteria:</p> <ol style="list-style-type: none"> When clicking the Get Details button for a scrum team (e.g., Scrum Team A), it should display the title "Scrum Details for [Scrum Team Name]". The details page should include: <ul style="list-style-type: none"> Tasks: A list of tasks associated with the scrum team. Each task should include: <ol style="list-style-type: none"> Task name (e.g., Task One) Task description (e.g., Description for Task One) Task status (e.g., To Do) Users: A list of users associated with the scrum team. Each user should include: <ol style="list-style-type: none"> User name (e.g., Employee One) User email (e.g., employee1@example.com) <p>The Dashboard, Profiles, and Logout links should be present throughout all the respective pages in the application.</p>
-------	---------------------------------	--

	Profiles Page	<p>** Kindly refer to the screenshots for any clarifications. **</p> <p>As a user, I should be able to view the tasks I am working on in the Agile track system</p> <p>Acceptance criteria:</p> <ol style="list-style-type: none"> 1. When clicking the Profiles link, it should display the title "User Profiles". 2. The profile page should include a section titled "Tasks Worked By [User Name]" (e.g., "Tasks Worked By Employee One"). 3. For each task the user is working on, the following details should be displayed: <ul style="list-style-type: none"> ● Title: The name of the task (e.g., Task One). ● Description: A description of the task (e.g., Description for Task One). ● Status: The current status of the task (e.g., To Do). <p>The Dashboard, Profiles, and Logout links should be present throughout all the respective pages in the application.</p> <p>** Kindly refer to the screenshots for any clarifications. **</p>
US_03	Admin Page (Logged in as a admin)	<p>As an admin, I should be able to manage and add, view scrum teams and profiles in the Agile Track System.</p> <p>Acceptance criteria:</p> <ol style="list-style-type: none"> 1. There should be 2 routes/links: <ul style="list-style-type: none"> ● Dashboard: When clicking this link, it should display the title "Scrum Teams". ● Profiles: When clicking this link, it should navigate to the profiles page where admin can view user profiles and tasks associated with them. 2. There should be a Logout button, and when clicked, it should log the admin out and navigate to the login page. <p>The above mentioned 2 routes/links & logout button should be throughout all the respective pages until logging out.</p> <ol style="list-style-type: none"> 3. Should have a title as "Scrum Teams". 4. The "Add New Scrum" button should be available to add new scrum with respective details. 5. A list of scrum teams should be displayed, where each team includes:

		<ul style="list-style-type: none"> • The team name (e.g., Scrum Team A, Scrum Team B). • A Get Details button for each team. When clicked, it should display the detailed view of that scrum team, displaying tasks and users associated with the team. <p>Add New Scrum Button:</p> <p>Acceptance criteria:</p> <ol style="list-style-type: none"> 1. When the "Add New Scrum" button is clicked: <ul style="list-style-type: none"> • A form should appear above the list of existing scrum teams. • The form should include fields for: <ol style="list-style-type: none"> 1. Scrum Name: Input field for the name of the new scrum. 2. Task Title: Input field for the title of the task. 3. Task Description: Input field for describing the task. 4. Task Status: Dropdown menu to select the status (e.g., To Do, In Progress, Done). 5. Assign To: Dropdown menu to select a user to whom the task will be assigned. • A "Create Scrum" button should be provided to submit the form and create the new scrum. • A "Cancel" button should be provided to hide the form without creating a new scrum. • All fields should be mandatory. If any field is left blank, an error message should be displayed. <p>Get Details Button:</p> <p>Acceptance criteria:</p> <ol style="list-style-type: none"> 1. When clicking the Get Details button for a scrum team (e.g., Scrum Team A), it should display the title "Scrum Details for [Scrum Team Name]". 2. The details page should include: <ul style="list-style-type: none"> • Tasks: A list of tasks associated with the scrum team. Each task should include:
--	--	---

	Profiles Page (Logged in as an Admin)	<ol style="list-style-type: none"> 1. Task name (e.g., Task One) 2. Task description (e.g., Description for Task One) 3. Task status - A dropdown menu to change the task status (e.g., To Do, In Progress, Done). <ul style="list-style-type: none"> ● Users: A list of users associated with the scrum team. Each user should include: <ol style="list-style-type: none"> 1. User name (e.g., Employee One) 2. User email (e.g., employee1@example.com) <ol style="list-style-type: none"> 3. Admin can update the status of a task by selecting the Task Status dropdown menu to update the status (e.g., To Do, In Progress, Done). <p>The Dashboard, Profiles, and Logout links should be present throughout all the respective pages in the application.</p> <p>** Kindly refer to the screenshots for any clarifications. **</p> <p>As an admin, I should be able to add, view and manage user profiles and their associated tasks in the Agile Track System.</p> <p>Acceptance criteria:</p> <ol style="list-style-type: none"> 1. The Profiles page should display the title "User Profiles". 2. The "Add New User" button should be available to add a new employee/admin with respective details. 3. A list of user profiles should be displayed, where each profile includes: <ul style="list-style-type: none"> ● Name: The name of the employee (e.g., Employee One, Employee Two). ● Email: The email address of the employee (e.g., employee1@example.com, employee2@example.com). ● A Get History button for each user. When clicked, it should display the tasks associated with that user below the user's profile.
--	--	---

Add New User Button:

Acceptance criteria:

1. When the "Add New User" button is clicked:
 - A form should appear above the list of existing user lists.
 - The page dynamically displays a form with input fields for:
 1. **Name:** Text input to enter the user's name.
 2. **Email:** Text input to enter the user's email address.
 3. **Password:** Password input to enter the user's password.
 4. **Role:** A dropdown selector to choose the user's role (e.g., Employee, Admin).
 - A "Create User" button should be provided to submit the form and add the new user to the list.
 - A "Cancel" button should be provided to hide the form and return to the initial view.
 - All fields should be mandatory. If any field is left blank, an error message should be displayed.

Get History Button:

Acceptance criteria:

1. When clicking the **Get History** button, it should display a form titled "Tasks Worked By <Employee name>".
2. The form should include the following fields:
 - **Title:** The name of the task (e.g., Task One).
 - **Description:** A description of the task (e.g., Description for Task One).
 - **Status:** The current status of the task (e.g., To Do).

The **Dashboard**, **Profiles**, and **Logout** links should be present throughout all the respective pages in the application.

**** Kindly refer to the screenshots for any clarifications. ****

5 VALIDATIONS

- All required fields must be fulfilled with valid data.
- When logging into the system all the fields must be filled.
- When adding a new scrum into the system all fields are mandatory to be filled.
- When adding a new user into the system all fields are mandatory to be filled.

6 CONSTRAINTS

- You should be able to press the “TAB” key and “SHIFT + TAB” to navigate from top field to bottom field and vice-versa.
- Once a user is logged in there should not be a “/login” url available until logged out.

7 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 the 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 React 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:8080/8081 to open project in browser -> takes 2 to 3 min
 - c. `npm run json-start` -> As we are using a json server to mimic our db.json file as a database. So, this command is useful to start a json server.
 - a. `npm run jest` -> to run all test cases and see the summary. It takes 5 to 6 min to run.
 - d. `npm run test` -> 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.