**Profileform.js:**  
  
***Functions:***  
1. const handleChange = (e) => {

    const { name, value } = e.target;

    setFormData({data

      ...formData,

      [name]: value

    });

  };  
  
**Code explaination:**  
This (**handleChange)** function updates the form data.  
**(Name)** is the attribute for the input field and (**Value)** is the current value of the input field.  
**(setFormData)** - is a function used to update the state of formdata which holds the whole form fields.   
**(…formData)** this state is updated using spread operator to preserve current value of the field.  
This spread operator is used to fetch multiple objects into a new object  
  
**([name]:value)** - sets the property name to its corresponding new value.

2.  const handleFileChange = (e) => {

    setFormData({

      ...formData,

      profilePicture: e.target.files[0]

    });

  };  
**Code explaination:**

**(handleFileChange)-** This function handles the change event for the file input like profilepic.

3.  const handleSubmit = (e) => {

    e.preventDefault();

    setIsSubmitted(true);

  };  
  
**Code explaination:**

**(handleSubmit)-** this function is called when the form is submitted. It prevents the default form submission.  
if submitted is true it displays a success msg

***2.Matchlist.js:***  
  
1.useEffect(() => {

    setMatches([

      { name: 'Harish', age: 30, interests: 'Music, Travel', Occupation: 'Civil Engineer', profilepicture: 'https://randomuser.me/api/portraits/men/58.jpg' },

      { name: 'Priyanka', age: 28, interests: 'Reading, Yoga', Occupation: 'Architect',profilepicture: 'https://randomuser.me/api/portraits/women/84.jpg' },

    ]);

  }, []);  
**Code explaination:**  
**(useEffect)** - function is used here to setup initial state of matches, it oly runs once when the component is 1st rendered and this is due to empty dependency array[].  
  
**(setMatches**) -function is called to populate the matches array with hardcoded data.  
  
  
2. const handleViewClick = (index) =>   
{ if (selectedMatchIndex === index)  
 { setSelectedMatchIndex(null); } else { setSelectedMatchIndex(index); } };  
  
**Code explaination:**

**(handleViewClick)** - This function handles the view button and this fuction accepts index as a parameter.  
  
**(selectedMatchIndex === index)** - The function checks if the clicked item (index) is already the one that's currently selected.  
  
3.       {matches.map((match, index) => (

**Code explaination:**  
matches is an array, probably an array of objects.  
  
The **matches.map() -** function is used to display over the matches array and render a card for each match. Each match card contains:  
name, age, intrests,profilepics,etc…  
  
  
 ***3. Profile.js:***  
  
1.const handleChange = (e) =>  
 { const { name, value } = e.target;  
 setEditedData({  
 ...editedData,  
 [name]: value   
}); };  
  
handleChange- fun is called when the user make changes in any input fields,  
it will update the **editeData** state to reflect the changes  
**...editedData** – helps to keep the existing values and only update the changed fields.  
  
2.const handleEdit = () => { setIsEditing(true); };  
  
**Code explaination:**  
   
This function is triggered when the user clicks the **Edit Profile** button.  
   
It sets the **isEditing** state to true, which causes the profile data to switch from a view-only mode to an editable form.  
  
  
3.const handleSave = () => {

setProfileData(editedData);

setIsEditing(false);

};  
**Code explaination:**  
This function is called when the user clicks the **Save** button after editing the profile.

It updates the **profileData** state with the values in **editedData** (i.e., the changes the user made).

Then, it sets **isEditing** to false to return to the view-only mode.  
  
4.const handleCancel = () => {

setEditedData(profileData);

setIsEditing(false);

};  
**Code explaination:**  
  
 This function is called when the user clicks the **Cancel** button after editing the profile.

It resets the **editedData** state to its original value (profileData) to discard any changes made.

It also sets **isEditing** to false to return to the view-only mode