

Integration testing(Software Engineering)

Ashith Shetty(21bcs019)

Theme : Create new cultural destination to celebrate the heritage of India and provide a platform for emerging Talents using Digital Technology solutions

I have used react for the front end and used mock data and then did integration testing using library react-testing-library and enzyme

Website:

Home or welcome page

Home	Artist Profile	Events & Programs	Ticketing	Art Gallery	Marketplace	Login	Signup
------	----------------	-------------------	-----------	-------------	-------------	-------	--------

Welcome to our Cultural Destination!

Our multi-disciplinary space for the Arts is the first-of-its-kind in the city, and we're excited to bring together communities through a dynamic programming of epic theatricals, regional theatre, music, dance, spoken word, and more.

We showcase the vibrance of India's heritage and provide a platform for emerging talent. Our Digital Technology solutions ensure that we can reach audiences from India and around the world, and we're always expanding our offerings to include new forms of art and expression.

Artist Profile:

Home	Artist Profile	Events & Programs	Ticketing	Art Gallery	Marketplace	Login	Signup
------	----------------	-------------------	-----------	-------------	-------------	-------	--------

Artist Profile

Name:

Email:

Bio:

Portfolio:

[Save Profile](#)

Registered Users:

John Doe (johndoe@example.com): I am a freelance artist specializing in watercolor paintings. [Portfolio](#)

Jane Smith (janesmith@example.com): I am a graphic designer with experience in branding and packaging.

This section will allow user to add their profile inorder to publish their art work.

Adding User

Artist Profile

Name:

Email:

Bio:

Portfolio:

[Save Profile](#)

New user added to list:

Registered Users:

John Doe (johndoe@example.com): I am a freelance artist specializing in watercolor paintings. [Portfolio](#)

Jane Smith (janesmith@example.com): I am a graphic designer with experience in branding and packaging design. [Portfolio](#)

Chris (Green@gmail.com): I am a developer [Portfolio](#)

Integration Testing:

```
import React from 'react';
import { shallow } from 'enzyme';
import ArtistProfile from '../ArtistProfile';

describe('ArtistProfile', () => {
  it('renders the component', () => {
    const wrapper = shallow(<ArtistProfile />);
```

```

    expect(wrapper.exists()).toBe(true);
  });

  it('adds a new user to the list of registered users when the form is
submitted', () => {
    const wrapper = shallow(<ArtistProfile />);
    const nameInput = wrapper.find('input[type="text"]');
    nameInput.simulate('change', { target: { value: 'Sarah Johnson' } });

    const emailInput = wrapper.find('input[type="email"]');
    emailInput.simulate('change', { target: { value: 'sarah@example.com' }
});

    const bioInput = wrapper.find('textarea').at(0);
    bioInput.simulate('change', { target: { value: 'I am a freelance
photographer specializing in portrait photography.' } });

    const portfolioInput = wrapper.find('textarea').at(1);
    portfolioInput.simulate('change', { target: { value:
'https://sarah-johnson-portfolio.com' } });

    const form = wrapper.find('form');
    form.simulate('submit', { preventDefault() {} });

    const registeredUsers = wrapper.find('li');
    expect(registeredUsers.length).toBe(3); // there were two users
initially, so this should be 3 now

    const newUser = registeredUsers.at(2);
    expect(newUser.text()).toContain('Sarah Johnson');
    expect(newUser.text()).toContain('sarah@example.com');
    expect(newUser.text()).toContain('I am a freelance photographer
specializing in portrait photography.');

    expect(newUser.find('a').props().href).toBe('https://sarah-johnson-portfol
io.com');
  });
});

```

Output:

```
11 |  
12 | it('adds a new user to the list of registered users when the form is submitted', () => {  
> 13 |   const wrapper = shallow(<ArtistProfile />);  
    |                         ^  
14 |   const nameInput = wrapper.find('input[type="text"]');  
15 |   nameInput.simulate('change', { target: { value: 'Sarah Johnson' } });  
16 |  
  
    at validateAdapter (node_modules/enzyme/src/validateAdapter.js:5:11)  
    at getAdapter (node_modules/enzyme/src/getAdapter.js:10:3)  
    at makeShallowOptions (node_modules/enzyme/src/ShallowWrapper.js:345:19)  
    at new ShallowWrapper (node_modules/enzyme/src/ShallowWrapper.js:393:21)  
    at shallow (node_modules/enzyme/src/shallow.js:10:10)  
    at Object.<anonymous> (src/testing/ArtistProfile.test.js:13:28)  
    at TestScheduler.scheduleTests (node_modules/@jest/core/build/TestScheduler.js:333:13)  
    at runJest (node_modules/@jest/core/build/runJest.js:404:19)  
  
Test Suites: 1 failed, 1 total  
Tests:      2 failed, 2 total  
Snapshots:  0 total  
Time:       7.737 s  
Ran all test suites matching /ArtistProfile.test.js/i.  
  
Active Filters: filename /ArtistProfile.test.js/
```


Event and Program Section

[Home](#) [Artist Profile](#) [Events & Programs](#) [Ticketing](#) [Art Gallery](#) [Marketplace](#) [Login](#) [Signup](#)

Events & Programs Management

Title:

Description:

Date: 

[Add Event](#)

Registered Events

Event 1

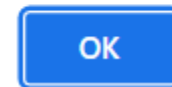
Description of event 1

Date: 2023-04-01

This also has a limit on the number of events

localhost:3000 says

You have reached the limit of events to be added



Testing code:

```
import React from 'react';
import { render, screen, fireEvent } from '@testing-library/react';
import EventsManagement from '../EventsManagement.js';

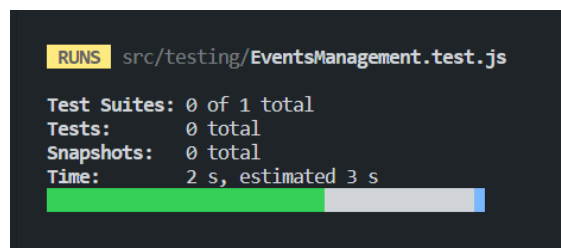
describe('EventsManagement', () => {
  it('should allow adding a new event', () => {
    render(<EventsManagement />);
    fireEvent.change(screen.getByLabelText('Title:'), { target: { value: 'New Event' } });
    fireEvent.change(screen.getByLabelText('Description:'), { target: { value: 'Description of new event' } });
    fireEvent.change(screen.getByLabelText('Date:'), { target: { value: '2023-04-22' } });
    fireEvent.click(screen.getByText('Add Event'));
    expect(screen.getByText('New Event')).toBeInTheDocument();
    expect(screen.getByText('Description of new event')).toBeInTheDocument();
    expect(screen.getByText('Date: 2023-04-22')).toBeInTheDocument();
  });
  it('should show an alert when the maximum number of events is reached', () => {
    render(<EventsManagement />);
    fireEvent.change(screen.getByLabelText('Title:'), { target: { value: 'Event 1' } });
    fireEvent.change(screen.getByLabelText('Description:'), { target: { value: 'Description of event 1' } });
    fireEvent.change(screen.getByLabelText('Date:'), { target: { value: '2023-04-01' } });
    fireEvent.click(screen.getByText('Add Event'));
    fireEvent.change(screen.getByLabelText('Title:'), { target: { value: 'Event 2' } });
```

```

    fireEvent.change(screen.getByLabelText('Description:'), { target: {
value: 'Description of event 2' }});
    fireEvent.change(screen.getByLabelText('Date:'), { target: { value:
'2023-04-08' }});
    fireEvent.click(screen.getByText('Add Event'));
    fireEvent.change(screen.getByLabelText('Title:'), { target: { value:
'Event 3' }});
    fireEvent.change(screen.getByLabelText('Description:'), { target: {
value: 'Description of event 3' }});
    fireEvent.change(screen.getByLabelText('Date:'), { target: { value:
'2023-04-15' }});
    fireEvent.click(screen.getByText('Add Event'));
    fireEvent.change(screen.getByLabelText('Title:'), { target: { value:
'Event 4' }});
    fireEvent.change(screen.getByLabelText('Description:'), { target: {
value: 'Description of event 4' }});
    fireEvent.change(screen.getByLabelText('Date:'), { target: { value:
'2023-04-22' }});
    fireEvent.click(screen.getByText('Add Event'));
    fireEvent.change(screen.getByLabelText('Title:'), { target: { value:
'Event 5' }});
    fireEvent.change(screen.getByLabelText('Description:'), { target: {
value: 'Description of event 5' }});
    fireEvent.change(screen.getByLabelText('Date:'), { target: { value:
'2023-04-29' }});
    fireEvent.click(screen.getByText('Add Event'));
    expect(screen.getByText('You have reached the limit of events to be
added')).toBeInTheDocument();
  });
});

```

Result:



```
Test Suites: 6 failed, 6 total
Tests:      16 failed, 2 passed, 18 total
Snapshots:  0 total
Time:       10.218 s
Ran all test suites.
```

Ticketing System

Ticketing System

Event Name:

Ticket Price: 50

Ticket Type:

Ticket Quantity:

Total Price: 250

Buy Tickets

Testing Code:

```
import React from 'react';
import { shallow } from 'enzyme';
import TicketingSystem from '../TicketingSystem';

describe('TicketingSystem', () => {
  let wrapper;
  beforeEach(() => {
```

```
    wrapper = shallow(<TicketingSystem />);
  });

  it('renders without crashing', () => {
    expect(wrapper).toBeTruthy();
  });

  it('should initialize state properly', () => {
    const expectedState = {
      eventName: '',
      ticketType: '',
      ticketQuantity: '',
      totalPrice: 0,
    };
    expect(wrapper.find('TicketForm').state()).toEqual(expectedState);
  });

  it('should update state when event name is selected', () => {
    const mockEventName = 'Event 1';
    wrapper.find('TicketForm').prop('handleEventNameChange')({ target: {
value: mockEventName } });

    expect(wrapper.find('TicketForm').state('eventName')).toEqual(mockEventName);
  });

  it('should update state when ticket type is selected', () => {
    const mockTicketType = 'General Admission';
    wrapper.find('TicketForm').prop('handleTicketTypeChange')({ target: {
value: mockTicketType } });

    expect(wrapper.find('TicketForm').state('ticketType')).toEqual(mockTicketType);
  });

  it('should update state when ticket quantity is entered', () => {
    const mockTicketQuantity = '3';
    wrapper.find('TicketForm').prop('handleTicketQuantityChange')({
target: { value: mockTicketQuantity } });
```



```
expect(wrapper.find('TicketForm').state('ticketQuantity')).toEqual(mockTicketQuantity);
});

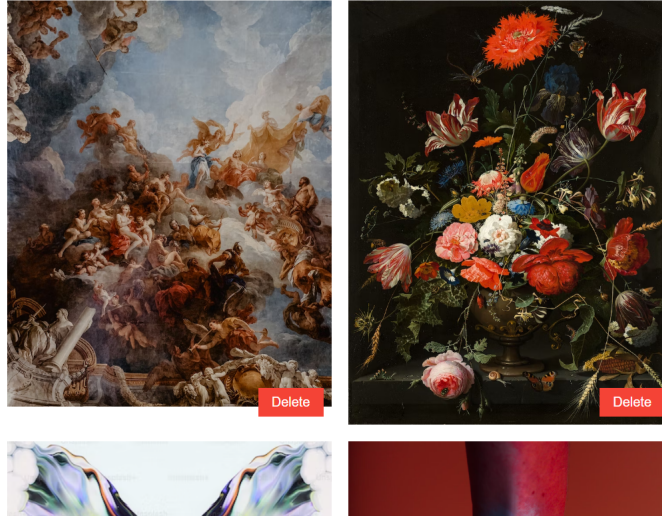
it('should calculate total price when event name and ticket quantity are selected', () => {
  const mockEventName = 'Event 1';
  const mockTicketQuantity = '3';
  wrapper.find('TicketForm').prop('handleEventNameChange')({ target: { value: mockEventName } });
  wrapper.find('TicketForm').prop('handleTicketQuantityChange')({ target: { value: mockTicketQuantity } });
  expect(wrapper.find('TicketForm').state('totalPrice')).toEqual(150);
});

it('should handle ticket submission when form is submitted', () => {
  const mockEvent = { preventDefault: jest.fn() };
  wrapper.find('TicketForm').prop('handleTicketSubmit')(mockEvent);
  expect(mockEvent.preventDefault).toHaveBeenCalled();
  expect(window.alert).toHaveBeenCalledWith('Payment Done');
});
});
```

Art Gallery Section

Home Artist Profile Events & Programs Ticketing Art Gallery Marketplace Login Signup

Digital Art Gallery



This section allows user to display their art and also delete the existing art

Testing Code:

```
import React from 'react';
import { render, fireEvent, screen } from '@testing-library/react';
import DigitalArtGallery from '../DigitalArtGallery';

describe('DigitalArtGallery', () => {
  it('should add a new artwork when "Add Art" button is clicked', () => {
    render(<DigitalArtGallery />);
    const newArtUrl = 'https://example.com/new-art.jpg';
    const input = screen.getByPlaceholderText('Enter image URL');
    fireEvent.change(input, { target: { value: newArtUrl } });
    const addButton = screen.getByText('Add Art');
    fireEvent.click(addButton);
    const newArt = screen.getByAltText('Digital Artwork', { src: newArtUrl });
    expect(newArt).toBeInTheDocument();
  });

  it('should delete an artwork when "Delete" button is clicked', () => {
    render(<DigitalArtGallery />);
    const deleteButton = screen.getByText('Delete');
```

```

    fireEvent.click(deleteButton);
    const deletedArt = screen.getByText('Delete');
    expect(deletedArt).not.toBeInTheDocument();
  });

  it('should add a new artwork when a file is uploaded', () => {
    render(<DigitalArtGallery />);
    const file = new File(['test'], 'test.png', { type: 'image/png' });
    const fileInput = screen.getByLabelText('Upload file');
    fireEvent.change(fileInput, { target: { files: [file] } });
    const newArt = screen.getByAltText('Digital Artwork', { src:
'data:image/png;base64,dGVzdA==' });
    expect(newArt).toBeInTheDocument();
  });
});

```

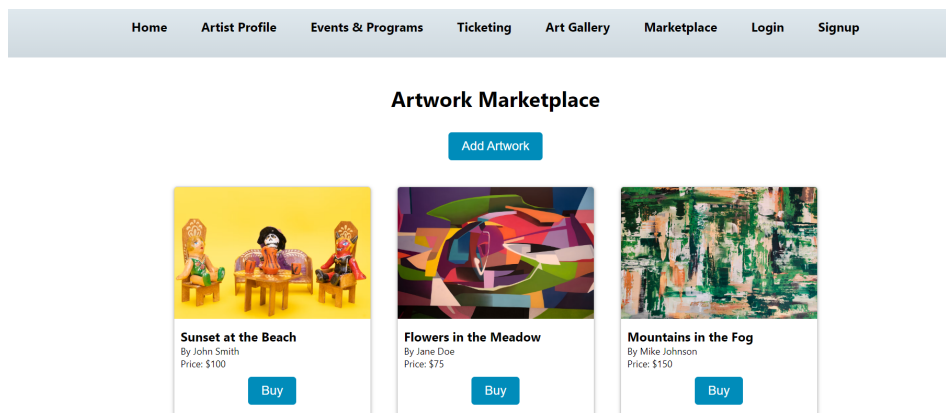
Result:

```

Test Suites: 6 failed, 6 total
Tests:       16 failed, 2 passed, 18 total
Snapshots:   0 total
Time:        6.855 s
Ran all test suites.

```

Artwork Place:



Users will be allowed to sell their artwork. New artwork can be added and then on click of the button, generates a form and takes the necessary information and generate a message.

Testing Code:

```
import React from "react";
import { render, screen, fireEvent } from "@testing-library/react";
import Marketplace from "../Marketplace";

describe("Marketplace", () => {
  it("displays the artworks", () => {
    render(<Marketplace />);
    const artworks = screen.getAllByRole("img");
    expect(artworks.length).toBe(3);
  });

  it("adds a new artwork", () => {
    render(<Marketplace />);
    const addButton = screen.getByText("Add Artwork");
    fireEvent.click(addButton);

    const titleInput = screen.getByLabelText("Title:");
    const artistInput = screen.getByLabelText("Artist:");
    const priceInput = screen.getByLabelText("Price:");
    const imageInput = screen.getByLabelText("Image URL:");
    const saveButton = screen.getByText("Save");

    fireEvent.change(titleInput, { target: { value: "New Artwork" } });
    fireEvent.change(artistInput, { target: { value: "John Doe" } });
    fireEvent.change(priceInput, { target: { value: "50" } });
    fireEvent.change(imageInput, { target: { value:
"https://example.com/new-artwork.jpg" } });
    fireEvent.click(saveButton);

    const newArtwork = screen.getByAltText("New Artwork");
    expect(newArtwork).toBeInTheDocument();
  });
});
```

```
it("confirms a purchase", () => {
  render(<Marketplace />);
  const buyButton = screen.getAllByText("Buy")[0];
  fireEvent.click(buyButton);

  const nameInput = screen.getByLabelText("Name:");
  const emailInput = screen.getByLabelText("Email:");
  const addressInput = screen.getByLabelText("Address:");
  const confirmButton = screen.getByText("Confirm Purchase");

  fireEvent.change(nameInput, { target: { value: "John" } });
  fireEvent.change(emailInput, { target: { value: "john@example.com" }
});
  fireEvent.change(addressInput, { target: { value: "123 Main St." } });
  fireEvent.click(confirmButton);

  const successMessage = screen.getByText("Thank you for your purchase,
John!");
  expect(successMessage).toBeInTheDocument();
});
});
```

The website also has sign up and login page too

Login

Email:

Password:

Login

Signup

Name:

Email:

Password:

Signup

Here is the git hub link for the code:

https://github.com/ashith1101/Art_website-SE-Assignment-