Name : **Patella Nitin Sai**  Roll No - **21bcs080**

Assignment - SE

**Theme : Create new cultural destination to celebrate the heritage of India and provide a platform**

**for emerging Talents using Digital Technology solutions**

**Aim :**

* Creating doors for a first-of-its-kind, multi-disciplinary space for the Arts in cities
* Encourage Visual art space and captivating array of public art
* Bring together communities through a dynamic programming of epic theatricals , regional theatre, music , dance , spoken word etc.
* Major attraction is to provide a platform for emerging talent and showcases the vibrance of India’s heritage
* Generate source of income for the Art communities through collaborations, aggregators and accelerators investments

**Target audiences :**

* Home to Art, Artists, the audience from India and around the world.

**Assignment scope :**

1. Identify various requirements for the above program initiative that can be developed as a digital solutions
2. Use ChatGPT platform an generate code for the above requirements
   1. Generate code and run the program in Goggle Colab/Jupiter Notebook/Visual Code/PyCharm
   2. Perform integrated testing. Add integration testing code in the same program.
3. Modify the same program. Write APIs to access the data from the public domain and test the program for regression testing the same program

**Deliverables :**

Working Program with test scripts embedded in the same program.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Here is a Python code that can be used to create a new cultural destination in India and provide a platform for emerging talents using digital technology solutions:

In fiie 1 alpha1.py code, we define two classes: CulturalDestination and DigitalSolutions. The CulturalDestination class has attributes for the city, arts space, and programming, as well as methods to display the description of the destination and generate income for the Art communities. The DigitalSolutions subclass inherits from the CulturalDestination class and adds an attribute for technology, as well as a method to allow visitors to interact with the digital technology solutions.

We create an instance of the CulturalDestination class and an instance of the DigitalSolutions subclass, and call the methods to display the description of the destination, generate income for the Art communities, and allow visitors to interact with the digital technology solutions.

**Identify various requirements for the above program initiative that can be developed as a digital solutions**

Here are some potential requirements for digital solutions that could be developed for the cultural destination program initiative:

1. A website or mobile app that provides information about the destination, including its location, hours of operation, programming schedule, and ticketing options.
2. A virtual tour of the arts space that allows visitors to explore the facility and its various offerings remotely.
3. An online platform that enables emerging artists to showcase their work and connect with potential collaborators and investors.
4. A social media presence for the destination that promotes upcoming events, shares images and videos of past performances, and encourages community engagement.
5. An e-commerce platform that sells merchandise related to the destination and its programming, such as souvenirs, books, and recordings.
6. A digital archive of past performances and exhibitions that allows visitors to revisit and share their favorite moments.
7. A system for collecting and analyzing visitor data, such as demographics and feedback, to inform programming decisions and marketing strategies.
8. A digital marketing campaign that targets potential visitors based on their interests, location, and demographics, and encourages them to visit the destination.
9. A digital map or guide that helps visitors navigate the local area and discover other cultural landmarks and attractions.
10. A mobile app that allows visitors to interact with programming elements in real-time, such as voting for their favorite acts, submitting questions to performers, or sharing their own creative content.

**Perform integrated testing. Add integration testing code in the same program:**

Integrated testing is a type of software testing that checks the interfaces and interactions between different modules or components of an application. In the context of our program, integrated testing would involve verifying that the various digital solutions we have developed work together seamlessly to provide a cohesive experience for users.

In file 2 alpha2.py code, we are using the Python unittest module to define a test suite with three test cases:

1. test\_homepage\_loads verifies that the homepage of the website loads correctly and has the expected title.
2. test\_virtual\_tour checks that the virtual tour feature works by clicking the virtual tour button and verifying that the video player appears.
3. test\_artist\_platform tests the artist platform by clicking the artist platform link and checking that there are at least some artists listed.

Each test case starts by setting up a new Selenium WebDriver instance and opening the website. The tearDown method is called after each test case to close the WebDriver and clean up resources.

To run these tests, we can simply execute the Python script. If any of the tests fail, we will see a detailed error message indicating which assertion failed and what the expected vs. actual results were. If all tests pass, we can be confident that the various digital solutions we have developed are working together as intended.

**Modify the same program. Write APIs to access the data from the public domain and test the program for regression testing the same program:**

To modify the program to include APIs for accessing data from the public domain, we can use a Python web framework such as Flask or Django.

In this file, we are defining two API endpoints using Flask's @app.route decorator. The get\_artists endpoint retrieves a list of artists from the public domain data source, while the get\_events endpoint retrieves a list of upcoming events.

To test the program for regression testing, we can use a testing framework such as pytest.

In file 4 alpha4.py, we are using the requests library to send HTTP requests to the API endpoints we defined earlier. The test\_get\_artists function verifies that the get\_artists endpoint returns a status code of 200 (indicating success) and a non-empty list of artists. Similarly, the test\_get\_events function checks that the get\_events endpoint returns a status code of 200 and a non-empty list of events.

To run these tests, we can simply execute the pytest command in the terminal. If any of the tests fail, we will see a detailed error message indicating which assertion failed and what the expected vs. actual results were. If all tests pass, we can be confident that the program is functioning correctly and that our changes have not introduced any regressions.