System Requirements Specification Index

For

Python Movie Tickets BookingConsole Application

Version 1.0

IIHT Pvt. Ltd.
fullstack@iiht.com

TABLE OF CONTENTS

1. Project Abstract

Provide a code solution to add movie details, show available movies, search movie and book the movie tickets.

2. Common Constraints with description:

1. Create a class MovieInventory with

    a. Anall_moviesasa static collection data member
    b. add_movie(movie_obj) – To add movie details
    c. show_movie_inventory() – To show movie details
    d. find_movie(expected, finder) - To search required move
    e. book_movie_tickets (movie_name,tickets)-To book movie

2. Add Movies and Tickets

    a. From MainClass, take input details of Movie and store in a collectioni.eall_movies.
    b. If movie adds in the collection successfully return a message "Movie Added".
    c. Make sure that Movie name is unique, else throw a custom exception(MovieAlreadyAvailableError)

3. Create a class Movie with
    a. movie_name as String
    b. available_tickets as int
    c. ticket_price as int

4. Search Movie
   From MainClass, take input of details to search movie. If movie details found then return and displaythe movie details else display a message 'No movies found'.

5. Book Movie Tickets
   From MainClass, take input of details to book movie.
    a. If movie is available book movie and return total costs of tickets.
       else throw a custom exception (MovieNotAvailableError)
    b. If tickets not available or tickets are insufficientthen, throw a custom exception (TicketsNotAvailableError)

3. Template Code Structure

Resources(Instances)

Class	Description	Status
Movie	This class contains all the properties of the Moviein the constructor.	Already implemented.

Resources(Business Logic)

Class	Description	Status
MovieInventory	This class contains all the methods which are used to write the business logic for the application.	Partially implemented (available only template code).

Resources (Exceptions)

Class	Description	Status
MovieAlreadyAvailableError	Custom Exception to be thrown when trying to add a movie in the inventory which is already exists.	Already created.
MovieNotAvailableError	Custom Exception to be thrown when trying to book movie tickets if movie does not exist.	Already created.

TicketsNotAvailableError	Custom Exception to be thrown when trying to book movie tickets if tickets are insufficient or does not exist.	Already created.

Resources(Main class)

Class	**Description**	**Status**
MainClass	This is the class which is having methods to be used to read data from the user and display the data. All the business logic methods of the **MovieInventory** class will be called from this class.	Partially implemented (available only template code).

4. Execution steps to be follow:

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. The editor Auto Saves the code.
4. If you want to exit(logout) and to 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 run application, use the following command

   ```
   python3 main.py
   ```

7. Mandatory: Before final submission run the following command

   ```
   python3 -m unittest
   ```

8. 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.
9. 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.