## **CONTRIBUTIONS:**

- 1. SIDDHANT KUMAR(PES2201800129):RECOMMENDER SYSTEM(RECOMMENDING MOVIES BASED ON USER ATTRIBUTES)
- 2. SKANDAN KA(PES2201800064):PREDICTING THE MOST LIKED VIDEOS AND RECENTLY VIEWED
- 3. SAMEER PRASAD SUBHEDAR(PES2201800323): PREDICTING MOST PLAYED VIDEOS

Object Oriented Analysis and Design with Software Engineering Laboratory

Subject Code: UE18CS355

Test Case ID	Name of Module	Test case description	Pre-condition s	Test Steps	Test data	Expected Results	Actual Result	Test Result
1.	Recommender System	Recommen d Movies based on User Behaviour( Collaborativ e Filtering)	The movie to be recommended or should be present in the dataset	1.We will be using the KNN algorithm to compute similarity with cosine distance metric which is very fast and more preferable than pearson coefficient.  2.We first check if the movie name input is in the database and if it is we use our	Movie:" Memento "	Movies Similar to "Mement o" should be displayed( recomme nded)	Movies Similar to "Memento " should be displayed(r ecommend ed)	PASS

	1	,	<b>T</b>		1	1	1	
				recommendati on system to find similar movies and sort them based on their similarity distance and output only the top 10 movies with their distances from the input movie.				
2.	Recommender System	Recommen d Movies based on User Behaviour( Collaborativ e Filtering)	The movie to be recommended or should be present in the dataset	1.We will be using the KNN algorithm to compute similarity with cosine distance metric which is very fast and more preferable than pearson coefficient.  2.We first check if the movie name input is in the database and if it is we use our recommendati on system to find similar movies and sort them based on their similarity distance and output only the top 10 movies with their distances from the input movie.	Movie:"T oy Story"	Movies Similar to "Toy Story" should be displayed( recomme nded)	Movies Similar to "Toy Story" should be displayed(r ecommend ed)	PASS

3.	Recommender System	Recommen d Movies based on User Behaviour( Collaborativ e Filtering)	The movie to be recommended or should be present in the dataset	1.We will be using the KNN algorithm to compute similarity with cosine distance metric which is very fast and more preferable than pearson coefficient.  2.We first check if the movie name input is in the database and if it is we use our recommendati on system to find similar movies and sort them based on their similarity distance and output only the top 10 movies with their distances	Movie:" Dusk Till Dawn"	Movies Similar to "Dusk Till Dawn" should be displayed( recomme nded)	Movies Similar to "Dusk Till Dawn" should be displayed(r ecommend ed)	PASS
4		7		from the input movie.				D. CC
4.	Recommender System	Recommen d Movies based on User Behaviour( Collaborativ e Filtering)	The movie to be recommended or should be present in the dataset	1.We will be using the KNN algorithm to compute similarity with cosine distance metric which is very fast and more preferable than pearson coefficient.	Movie:" Wonder Woman"	Movies Similar to "Wonder Woman" should be displayed( recomme nded)	Movies Similar to "Wonder Woman" should be displayed(r ecommend ed)	PASS

2.We first check if the movie name input is in the database and if it is we use our recommendati on system to find similar movies and sort them based on their similarity distance and output only the top 10 movies with their distances from the input movie.  5. Recomender System  System  Recommen d Movies based on User Behaviour( Collaborativ e Filetring)  The movie to be using the KNN algorithm to compute similarity with compute similarity with compute similarity with cosine distance metric which is very fast and more preferable than pearson coefficient.  2.We first check if the movie name input is in the database and if it is we use our recommendati on system to find similar
movies and sort them based on their similarity

	1			autout anle		I		
				output only the <b>top 10</b>				
				movies with				
				their distances				
				from the input				
				movie.				
6.	Recommender	Recommen	The movie to	1.We will be	Movie:"	Movies	Movies	PASS
	System	d Movies	be	using the KNN	Usual	Similar to	Similar to	
		based on	recommended	algorithm to	Suspects"	"Usual	"Usual	
		User	or should be	compute		Suspects" should be	Suspects" should be	
		Behaviour( Collaborativ	present in the dataset	similarity with cosine		displayed(	displayed(r	
		e Filtering)	datasci	distance		recomme	ecommend	
		c i ntering)		metric which		nded)	ed)	
				is very fast				
				and more				
				preferable				
				than <u>pearson</u>				
				coefficient.				
				2.We first				
				check <u>if the</u>				
				movie name				
				input is in the				
				<u>database</u> and				
				if it is we use				
				our				
				recommendati on system to				
				find similar				
				movies and				
				sort them				
				based on their				
				similarity				
				distance and				
				output only				
				the <b>top 10</b> movies with				
				their distances				
				from the input				
				movie.				
7.	Recommender	Recommen	The movie to	1.We will be	Movie:"B	Movies	Movies	PASS
	System	d Movies	be	using <u>the KNN</u>	ottle	Similar to	Similar to	
		based on	recommended	algorithm to	Rocket"	"Bottle	"Bottle	
		User	or should be	compute		Rocket"	Rocket"	
		Behaviour(	present in the	similarity with		should be	should be	
		Collaborativ e Filtering)	dataset	cosine distance		displayed( recomme	displayed(r ecommend	
				metric which		nded)	ed)	
				is very fast		11404)		
	1			,	!		·	

		Γ		1	_	Γ		· · · · · · · · · · · · · · · · · · ·
				and more				
				preferable				
				than <u>pearson</u>				
				coefficient.				
				2.We first				
				check <u>if the</u>				
				movie name input is in the				
				<u>database</u> and				
				if it is we use				
				our				
				recommendati				
				on system to				
				find similar				
				movies and				
				sort them				
				based on their				
				similarity				
				distance and				
				output only				
				the <b>top 10</b>				
				movies with				
				their distances				
				from the input				
				movie.				
8.	Recommender	Recommen	The movie to	1.We will be	Movie:"S	Movies	Movies	FAIL
	System	d Movies	be	using <u>the KNN</u>	hazam''	Similar to	Similar to	
		based on	recommended	algorithm to		"Shazam"	"Shazam"	
		User	or should be	compute		should be	will not be	
		Behaviour(	present in the	similarity with		displayed(	displayed(r	
		Collaborativ	dataset	<u>cosine</u>		recomme	ecommend	
		e Filtering)		distance metric which		nded)	ed) as this movie isn't	
				is very fast			there in the	
				and more			dataset.	
				preferable			dataset.	
				than <u>pearson</u>				
				coefficient.				
				2.We first				
				check <u>if the</u>				
				movie name				
	1			input is in the				
				<u>database</u> and				
				database and if it is we use				
				database and if it is we use our				
				database and if it is we use our recommendati				
				database and if it is we use our				

9. Recommender System Recommen duser Behaviour (Collaborative Filtering)  Filtering  9. Recommender System Recommen der User Behaviour (Collaborative Filtering)  Filtering  Filtering  Filtering  Recommender System Recommen der User Behaviour (Collaborative Filtering)  Filtering  Fall Movies busing the KNN algorithm to compute similarity with cosine distance metric which is very fast and more preferable than pearson coefficient.  Similar to "Kung Fu Panda" should be displayed (recommended) will not be displayed (recommended) at his movic isn't there.  Similar to "Kung Fu Panda" should be displayed (recommended) is very fast and more preferable than pearson coefficient.  Similar to "Kung Fu Panda" should be displayed (recommended) is very fast and more preferable than pearson coefficient.  Similar to "Kung Fu Panda" should be displayed (recommended) is very fast and more preferable than pearson coefficient.  Similar to "Kung Fu Panda" should be displayed (recommended) is very fast and more preferable than pearson coefficient.  Similar to "Kung Fu Panda" should be displayed (recommended) is very fast and more preferable than pearson coefficient.  Similar to "Kung Fu Panda" should be displayed (recommended) is very fast and more preferable than pearson coefficient.  Similar to "Kung Fu Panda" should be displayed (recommended) is very fast and more preferable than pearson coefficient.  Similar to "Kung Fu Panda" should be displayed (recommended) is very fast and more preferable than pearson coefficient.  Similar to "Kung Fu Panda" should be displayed (recommended) is very fast and more preferable than pearson coefficient.  Similar to "Kung Fu Panda" should be displayed (recommended) is very fast and more preferable than pearson coefficient.  Similar to "Kung Fu Panda" should be displayed (recommended) is very fast and more preferable than pearson coefficient.  Similar to "Kung Fu Panda" should be displayed (recommended) is very fast and more preferable than pearson coefficient.
the top 10 movies with their distances from the input

10.	Recommender	Recommen	The movie to	1.We will be	Movie:"	Movies	Movies	FAIL
	System	d Movies	be	using the KNN	Avengers	Similar to	Similar to	
		based on	recommended	algorithm to	End	"Avenger	"Avengers	
		User	or should be	compute	Game"	s End	EndGame"	
		Behaviour(	present in the	similarity with		Game"	will not be	
		Collaborativ	dataset	cosine		should be	displayed(r	
		e Filtering)		distance		displayed(	ecommend	
		]		metric which		recomme	ed) as this	
				is very fast		nded)	movie is	
				and more		,	not in the	
				preferable			dataset.	
				than <u>pearson</u>				
				coefficient.				
				2.We first				
				check if the				
				movie name				
				input is in the				
				database and				
				if it is we use				
				our				
				recommendati				
				on system to				
				find similar				
				movies and				
				sort them				
				based on their				
				similarity				
				distance and				
				output only				
				the <b>top 10</b>				
				movies with				
				their distances				
				from the input				
				movie.				

Test Case ID	Name of Module	Test case description	Pre-conditi ons	Test Steps	Test data	Expecte d Results	Actual Result	Test Result
1.	Recently Viewed	Displays most recently viewed	Movie should be in the dataset used	1.Import the Dataset. 2.Sort based on Timestamp. 3.Display the required	data.head(1)	Toy Story	Toy Story	PASS

				Data				
2.	Recently Viewed	Displays 3 most recently viewed	Movie should be in the dataset used	1.Import the Dataset. 2.Sort based on Timestamp. 3.Display the required Data	data.head(3)	Toy Story Grumpi er Old Men Usual Suspec	Toy Story Grumpie r Old Men Usual Suspects	PASS
3.	Recently Viewed	Displays 5 most	Movie should be	1.Import the Dataset.	data.head(5)	Toy Story	Toy Story	PASS
	Viewed	recently viewed	in the dataset used	2.Sort based on Timestamp. 3.Display the required Data		Grumpi er Old Men Usual Suspec ts Dusk Till Dawn	Grumpie r Old Men Usual Suspec ts Dusk Till Dawn	
						Bottle Rocket	Rocket	

4.	Recently Viewed	Displays 4 most recently viewed	Movie should be in the dataset used	1.Import the Dataset. 2.Sort based on Timestamp. 3.Display the required Data	data.head(4)	Toy Story Grumpi er Old Men Usual Suspec ts Dusk Till Dawn	Toy Story Grumpie r Old Men Usual Suspec ts Dusk Till Dawn	PASS
5.	Recently Viewed	Displays 2 most recently viewed	Movie should be in the dataset used	1.Import the Dataset. 2.Sort based on Timestamp. 3.Display the required Data	data.head(2)	Toy Story Grumpi er Old Men	Toy Story Grumpie r Old Men	PASS

Test Case ID	Name of Modul e	Test case description	Pre-conditi ons	Test Steps	Test data	Expected Results	Actual Result	Test Result
1.	Liked	Displays	Movie	Import the	data.drop_	Bill Hicks:	Bill Hicks:	PASS

		most liked movies	displayed should be from the dataset	dataset, drop the duplicates following which we sort the movies based on rating	duplicates( subset='mo vield').sort _values('rat ing', ascending = False).head (1)	Revelatio ns (1993)	Revelations (1993)	
2.	Liked	Displays most liked movies	Movie displayed should be from the dataset	Import the dataset, drop the duplicates following which we sort the movies based on rating	data.drop_ duplicates( subset='mo vield').sort _values('rat ing', ascending = False).head (2)	Bill Hicks: Revelatio ns (1993) Rain (2001)	Bill Hicks: Revelations (1993) Rain (2001)	PASS
3.	Liked	Displays most liked movies	Movie displayed should be from the dataset	Import the dataset, drop the duplicates following which we sort the movies based on rating	data.drop_ duplicates( subset='mo vield').sort _values('rat ing', ascending = False).head (3)	Bill Hicks: Revelatio ns (1993)  Rain (2001)  George Carlin: Life Is Worth Losing (2005)	Bill Hicks: Revelations (1993) Rain (2001) George Carlin: Life Is Worth Losing (2005)	PASS
4.	Liked	Displays most liked movies	Movie displayed should be from the dataset	Import the dataset, drop the duplicates following which we sort the movies based on rating	data.drop_ duplicates( subset='mo vield').sort _values('rat ing', ascending = False).head (4)	Bill Hicks: Revelatio ns (1993)  Rain (2001)  George Carlin: Life Is Worth Losing	Bill Hicks: Revelations (1993)Your Juice in the Hood Rain (2001) George Carlin: Life Is Worth Losing (2005) What Men Talk	PASS

	(2005)	About (2010)	
	What Men Talk About (2010)		

Test	Name of	Test case	Pre-conditions	Test Steps	Test data	Expected	Actual	Test
Case	Module	description		l test steps	10st data	Results	Result	Result
ID		1						
1.	Most Played	Will display	The videos	Import the tags	for a in y:	The	The movie	PASS
		the most	played will be	dataset, make	try:	movie	Virtuosity	
		played	from the	a list of tuples	print('The	Virtuosity	(1995) has	
		videos	dataset	of the number	movie',m	(1995)	been	
				of occurrences	ovies['titl	has been	viewed	
				of a particular	e'][a[1]],'	viewed	181 times	
				movie. Get the	has been	181 times		
				movie name	viewed',a			
				from the	[0],'times'			
				movies dataset	)			
				and map it to	except:			
				the number of	,.			
				occurrences of	continue			
	M 4 D1 1	XX7:11 1: 1	T1 '1	movieId.	c ·	771	T1 .	DACC
2.	Most Played	Will display the most	The videos	Import the tags	for a in y:	The	The movie Billy Elliot	PASS
		played	played will be from the	dataset, make a list of tuples	try: print('The	movie Billy	(2000) has	
		videos	dataset	of the number	movie',m	Elliot	been (2000) Has	
		VidCos	dataset	of occurrences	ovies['titl	(2000)	viewed 54	
				of a particular	e'][a[1]],'	has been	times	
				movie. Get the	has been	viewed 54	times	
				movie name	viewed',a	times		
				from the	[0],'times'	times		
				movies dataset	)			
				and map it to	except:			
				the number of				
				occurrences of	continue			
				movieId.				
3.	Most Played	Will display	The videos	Import the tags	for a in y:	The	The movie	PASS
		the most	played will be	dataset, make	try:	movie	Grand Day	
		played	from the	a list of tuples	print('The	Grand	Out with	
		videos	dataset	of the number	movie',m	Day Out	Wallace	
				of occurrences	ovies['titl	with	and	

		I		C : 1	175 5177 1	XX 7 11	I a · · ·	<u> </u>
				of a particular movie. Get the movie name from the movies dataset and map it to the number of occurrences of movieId.	e'][a[1]],' has been viewed',a [0],'times' ) except: continue	Wallace and Gromit, A (1989) has been viewed 41 times	Gromit, A (1989) has been viewed 41 times	
4.	Most Played	Will display the most played videos	The videos played will be from the dataset	Import the tags dataset, make a list of tuples of the number of occurrences of a particular movie. Get the movie name from the movies dataset and map it to the number of occurrences of movieId.	for a in y: try: print('The movie',m ovies['titl e'][a[1]],' has been viewed',a [0],'times' ) except: continue	The movie Underneat h (1995) has been viewed 35 times	The movie Underneat h (1995) has been viewed 35 times	PASS
5.	Most Played	Will display the most played videos	The videos played will be from the dataset	Import the tags dataset, make a list of tuples of the number of occurrences of a particular movie. Get the movie name from the movies dataset and map it to the number of occurrences of movieId.	for a in y: try: print('The movie',m ovies['titl e'][a[1]],' has been viewed',a [0],'times' ) except: continue	The movie TiMER (2009) has been viewed 34 times	The movie TiMER (2009) has been viewed 34 times	PASS
6.	Most Played	Will display the most played videos	The videos played will be from the dataset	Import the tags dataset, make a list of tuples of the number of occurrences of a particular movie. Get the movie name from the movies dataset and map it to the number of	for a in y: try: print('The movie',m ovies['titl e'][a[1]],' has been viewed',a [0],'times' ) except: continue	The movie Tales from the Darkside: The Movie (1990) has been viewed 32 times	The movie Tales from the Darkside: The Movie (1990) has been viewed 32 times	PASS

	I	T	T		Γ	I	Ι	<del>- 1</del>
				occurrences of movieId.				
7.	Most Played	Will display the most played videos	The videos played will be from the dataset	Import the tags dataset, make a list of tuples of the number of occurrences of a particular movie. Get the movie name from the movies dataset and map it to the number of occurrences of movieId.	for a in y: try: print('The movie',m ovies['titl e'][a[1]],' has been viewed',a [0],'times' ) except: continue	The movie Quiz Show (1994) has been viewed 26 times	The movie Quiz Show (1994) has been viewed 26 times	PASS
8.	Most Played	Will display the most played videos	The videos played will be from the dataset	Import the tags dataset, make a list of tuples of the number of occurrences of a particular movie. Get the movie name from the movies dataset and map it to the number of occurrences of movieId.	for a in y: try: print('The movie',m ovies['titl e'][a[1]],' has been viewed',a [0],'times' ) except: continue	The movie Tenet (2021) has been viewed 78 times	Key error	FAIL
9.	Most Played	Will display the most played videos	The videos played will be from the dataset	Import the tags dataset, make a list of tuples of the number of occurrences of a particular movie. Get the movie name from the movies dataset and map it to the number of occurrences of movieId.	for a in y: try: print('The movie',m ovies['titl e'][a[1]],' has been viewed',a [0],'times' ) except: continue	The movie End Game (2019) has been viewed 104 times	Key error	FAIL
10.	Most Played	Will display the most played videos	The videos played will be from the dataset	Import the tags dataset, make a list of tuples of the number of occurrences of a particular	for a in y: try: print('The movie',m ovies['titl e'][a[1]],'	The movie John Wick (2014) has been	Key error	FAIL

		movie. Get the	has been	viewed 78	
		movie name	viewed',a	times	
		from the	[0],'times'		
		movies dataset	)		
		and map it to	except:		
		the number of	_		
		occurrences of	continue		
		movieId.			