Team Management System in the Champions League
Format of scenarios and use cases

Setting Up the Scenarios

Name	Class	Scenery
Setup1	TeamTest	Team != null [{ "name": "Real Madrid", "country": "Spain", “Points”: “2” "titles": 14, "coefficient": 98.5 }]
Setup2	TeamTest	name = "", country = "",“Points”: “2”, titles = 1, coefficient = 10
Setup3	TeamTest	name = "A”, country = "Spain",“Points”: “2”, titles = -10, coefficient = 95.5
Setup4	MatchTest	Match != null [{ "local": "Barcelona", "visitor": "PSG", "goalsLocal": 2, "goalsVisitor": 1, "date": "2025-04-15" }]
Setup5	MatchTest	Match != null [{ "local": "Barcelona", "visitor": "PSG", "goalsLocal": -2, "goalsVisitor": -1, "date": "2025-04-15" }]
Setup6	MatchTest	Match != null [{ "local": "Barcelona", "visitor": "PSG", "goalsLocal": 2, "goalsVisitor": 1, "date": "2025-75-35" }]
Setup7	RankingTest	priority queue != null
Setup8	ActionTest	actions != null [{ “Team [ ] ” : “Barcelona” }]
Setup9	ControllerTest	addTeam[{ “name”: “Real Madrid” “country”: “Spain” “tites”: “6” “UEFAcoefficient”: “12” }]

Setup10	ControllerTest	addTeam[{ “name”: “Real Madrid” “country”: “spain” “tites”: “six” “UEFAcoefficient”: “12” }]
Setup11	ControllerTest	addTeam[{ “name”: “”#$%&/()” “country”: “Spain” “tites”: “eight” “UEFAcoefficient”: “12” }]
Setup12	ControllerTest	addMatch[{ “homeTeam” : “Barcelona” “awayTeam” : “Real Madrid” “homegoals” : “3” “awaygoals” : “2” “date” : “26/04/25” }]
Setup13	ControllerTest	addMatch[{ “homeTeam” : “Barcelona” “awayTeam” : “Real Madrid” “homegoals” : “3” “awaygoals” : “2” “date” : “26/04/25” }]
Setup14	ControllerTest	addMatch[{ “homeTeam” : “Barcelona” “awayTeam” : “Real Madrid” “homegoals” : “one” “awaygoals” : “2” “date” : “26/04/25” }]
Setup15	ControllerTest	[{ "action": "addTeam", "team": { "name": "Chelsea", "country": "Inglaterra", "titles": 2, "coefficient": 15 } }]
Setup16	ControllerTest	[{ "action": "addTeam", "team": { "name": "Chelsea", "country": "", "titles": 2, "coefficient": 15 } }]
Setup17	ControllerTest	[{ "action": "addTeam", "team": { "name": "Chelsea", "country": "Inglaterra", "titles": 2, "coefficient": 15 } }]
Setup18	ControllerTest	[{ "matchId": "Match 0" }]
Setup19	ControllerTest	[{ "matchId": "Match 43" }]
Setup20	ControllerTest	[{ "matchId": " Match 0" }]

Setup21	ControllerTest	[{ "enqueuedMatchId": "Match 0" }]
Setup22	ControllerTest	[{ "enqueuedMatchId": "#$%&/(" }]
Setup23	ControllerTest	[{ "enqueuedMatchId": "" }]
Setup24	ControllerTest	[{ "teams": ["Real Madrid", "Chelsea"] }]
Setup25	ControllerTest	[{ "teams": ["", ""] }]
Setup26	ControllerTest	[{ "teams": ["Real Madrid", "Chelsea"] }
Setup27	ControllerTest	[{ "searchTeam": "Real Madrid" }]
Setup28	ControllerTest	[{ "searchTeam": "" }]
Setup29	ControllerTest	[{ "searchTeam": "Real $%&Madrid" }]
Setup30	ControllerTest	[{ "Match": "Match 0" }]
Setup31	ControllerTest	[{ "Match": "32" }]
Setup32	ControllerTest	[{ "searchMatch": “Match 0 " "searchMatch": “Match 0 " }]
Setup33	NodeTest	Node != null [{ "data": "Elemento A" }]
Setup34	NodeTest	Node == null [{ "data": "" }]
Setup35	NodeTest	Node != null [{ "data": " " }]
Setup36	LinkedListTest	[{ "data": "Elemento A" }, { "data": "Elemento B" }, { "data": "Elemento C" }]
Setup37	LinkedListTest	[{ "data": "" }]
Setup38	LinkedListTest	[{ "data": "Element A", "Element A"}]
Setup39	QueueTest	[{ "data": “5” }, { "data": “10” }, { "data": “15” }]
Setup40	QueueTest	[{ "data": }]
Setup41	QueueTest	[{ "data": “5” }]
Setup42	StackTest	[{ "data": "X" }, { "data": "Y" }, { "data": "Z" }]
Setup43	StackTest	[{ "data": "" }]

Setup44	StackTest	[{ "data": "X" }]
Setup45	HashTableTest	[{ "key": "A", "value": 1 }, { "key": "B", "value": 2 }]
Setup46	HashTableTest	[{ "key": "", "value": }]
Setup47	HashTableTest	[{ "key": "A", "value": 1 }]
Setup48	HeapTest	[{ "data": 10 }, { "data": 20 }, { "data": 5 }]
Setup49	HeapTest	[{ "data": }]
Setup50	HeapTest	[{ "data": 10 , "data": 10 }]
Setup51	PriorityQueueTest	[{ "data": 10 }, { "data": 20 }, { "data": 5 }]
Setup52	PriorityQueueTest	[{ "data": “” }]
Setup53	PriorityQueueTest	[{ "data": “10” , "data": “10” }]

Test Case Design

TeamTest

Test Objective: Verify that a team object is correctly created with valid attributes and ensure all attribute getters return the correct values.

Class	Method	Scenery	Input Values	Expected result
TeamTest	Team()	Setup1	name = "Real Madrid", country = "Spain", titles = 14, coefficient = 98.5	Team object created with the provided attributes, not null.
TeamTest	Team()	Setup2	name = "", country = "", titles = -1, coefficient = -10	Team object created with an empty name (if no validation).
TeamTest	Team()	Setup3	name = "A”, country = "Spain", titles = -10, coefficient = 95.5	Titles cannot be negative.
TeamTest	getName()	Setup1	-	Returns "Real Madrid".
TeamTest	getCountry()	Setup1	-	Returns "Spain".
TeamTest	getTitles()	Setup1	-	Returns 14.

TeamTest	getCoefficient()	Setup1	-	Returns 98.5.
TeamTest	addPoint()	Setup1	Team != null [{ "name": "Real Madrid", "country": "Spain", “Points”: “2” "titles": 14, "coefficient": 98.5 }]	totalPoints is incremented by 3.
TeamTest	addPoint()	Setup2	name = "", country = "",“Points”: “2”, titles = 1, coefficient = 10	totalPoints is decremented by 5, no validation on negative points.
TeamTest	addPoint()	Setup3	name = "A”, country = "Spain",“Points”: “2”, titles = -10, coefficient = 95.5	totalPoints remains the same.

MatchTest

Test Objective: Verify that a match object is correctly created with valid attributes and ensure all attribute getters return the correct values.

Class	Method	Scenery	Input Values	Expected result
MatchTest	Match()	Setup4	local = "Barcelona", visitor = "PSG", goalsLocal = 2, goalsVisitor = 1, date = "2025-04-15"	Match object created with the provided attributes, not null.
MatchTest	Match()	Setup5	Match != null [{ "local": "Barcelona", "visitor": "PSG", "goalsLocal": -2, "goalsVisitor": -1, "date": "2025-04-15" }]	Throws exception or error for null homeTeam (if validation exists).
MatchTest	Match()	Setup6		Match object created with the provided attributes.
MatchTest	getHomeTeam()	Setup4	-	Returns "Barcelona".
MatchTest	getAwayTeam()	Setup4	-	Returns "PSG".
MatchTest	getHomeGoals()	Setup4	-	Returns 2.
MatchTest	getAwayGoals()	Setup4	-	Returns 1.
MatchTest	getDate()	Setup4	-	Returns "2025-04-15".

RankingTest

Test Objective: Verify that a team is correctly added to the priority queue and that the queue is not null after adding a team.

Class	Method	Scenery	Input Values	Expected result
RankingTest	addTeam()	Setup7	[{ "name": "Bayern", }]	"Team successfully added to the priority queue."
RankingTest	addTeam()	Setup7	[{ "name": "Bayern", }]	Throws exception or error for null team input.
RankingTest	addTeam()	Setup7	[{ "name": "Bayern", }]	The "Bayern", team has already been added to the priority queue.
RankingTest	removeTeam()	Setup7	team = "Real Madrid"	Successfully removes team "Real Madrid" from priorityQueue.
RankingTest	removeTeam()	Setup7	[{“team” = ""}]	Throws exception or error for null team input.
RankingTest	removeTeam()	Setup7	[{“team” = "Bayer"}]	The team has not been found

ActionTest

Test Objective: Verify that actions are not null and that actions are properly recorded when a team is added and ranked.				
Class	**Method**	**Scenery**	**Input Values**	**Expected result**
ActionTest	addAction()	Setup8	[{ "Team": "Barcelona" }]	"Action successfully recorded and actions queue is not null."
ActionTest	addAction()	Setup8	[{ "Team": ""}]	Throws exception or error when team is null.
ActionTest	addAction()	Setup8	[{ "Team": "Barcelona#&"}]	unregistered action

ControllerTest

Test Objective: Verify that the Controller correctly adds teams and matches, performs actions, enqueues matches, and handles search and undo operations.				
Class	**Method**	**Scenery**	**Input Values**	**Expected result**
Controller Test	addTeam()	Setup9	[{ "name": "Real Madrid", "country": "Spain", "titles": 6, "UEFAcoefficient": 12 }]	"Team Real Madrid added successfully."
Controller Test	addTeam()	Setup10	[{ "name": "", "country": "", "titles": 6, "UEFAcoefficient": 12 }]	"Error: Team name cannot be empty."
Controller Test	addTeam()	Setup11	[{ "name": "Real Madrid", "country": "Spain", "titles": 0,	"Team Real Madrid added successfully with 0 titles."

			"UEFAcoefficient": 0 }]	
Controller Test	addMatch()	Setup12	[{ "homeTeam": "Barcelona", "awayTeam": "Real Madrid", "homegoals": 3, "awaygoals": 2, "date": "26/04/25" }]	"Match Match 0 added successfully."
Controller Test	addMatch()	Setup13	[{ "homeTeam": "Barcelona", "awayTeam": "Real Madrid", "homegoals": -3, "awaygoals": -2, "date": "26/04/25" }]	"Error: Goals cannot be negative."
Controller Test	addMatch()	Setup14	[{ "homeTeam": "Barcelona", "awayTeam": "Real Madrid", "homegoals": 0, "awaygoals": 0, "date": "26/04/25" }]	"Match added successfully with no goals scored."
Controller Test	addAction()	Setup15	[{ "action": "addTeam", "team": { "name": "Chelsea", "country": "Inglaterra", "titles": 2, "coefficient": 105 } }]	"Action successfully recorded."
Controller Test	addAction()	Setup16	[{ "action": "addTeam", "team": { "name": "", "country": "I", "titles": 2, "coefficient": 105 } }]	"Error: Team data is null."
Controller Test	addAction()	Setup17	action = "updateMatch", team = {name = "Arsenal", country = "Inglaterra", titles = 3, coefficient = 120}	"Special action successfully recorded for updating match."
Controller Test	undo()	Setup18	[{ "matchId": "Match 0" }]	"Undo successful: Match Match 0 has been deleted and points reverted."
Controller Test	undo()	Setup19	[{ "matchId": "" }]	"Error: Match ID does not exist."
Controller Test	undo()	Setup20	[{ "matchId": "Match 0" }]	"Undo successful: Match 0 has been deleted with incomplete details."
Controller Test	enqueueMatch()	Setup21	[{ "enqueuedMatchId": "Match 0" }]	"Match Match 0 enqueued successfully."
Controller Test	enqueueMatch()	Setup23	[{ "enqueuedMatchId": "" }]	"Error: Match ID cannot be null."

Controller Test	enqueueMatch()	Setup22	[{ "enqueuedMatchId": "#$%&/(" }]	"Error: Incomplete match details for enqueue."
Controller Test	teamRanking()	Setup24	[{ "teams": ["Real Madrid", "Chelsea"] }]	"Team ranking displayed."
Controller Test	teamRanking()	Setup28	[{ "teams": ["", ""] }]	"Error: No teams available for ranking."
Controller Test	teamRanking()	Setup29	[{ "teams": ["Real Madrid", "Chelsea"] }]	"Team ranking displayed with some missing teams."
Controller Test	publicSearchMatch()	Setup30	[{ "searchMatch": "Match 0" }]	"Match found: Match 0" “homeTeam” : “Barcelona” “awayTeam” : “Real Madrid” “homegoals” : “3” “awaygoals” : “2” “date” : “26/04/25”
Controller Test	publicSearchMatch()	Setup31	[{ "searchMatch": "Match 64" }]	"Error: Match not found."
Controller Test	publicSearchMatch()	Setup32	[{ "searchMatch": "Match33" }]	"Match found: Match33 with incomplete details."

Structures

Test Objective: Verify the correct operation of creation, insertion and basic manipulation of data structures.

Class	Method	Scenery	Input Values	Expected result
NodeTest	Node()	Setup33	data = "Element A"	Node created with data "Element A" and next node as null.
NodeTest	Node()	Setup34	data = ""	Error or Node created with null data.
NodeTest	Node()	Setup35	data = " "	Node created with empty string as data.
LinkedListTest	add()	Setup36	data = "Element A", then "Element B", then "Element C"	Linked list contains ["Element A", "Element B", "Element C"] in order.
LinkedListTest	add()	Setup37	data = ""	Error: Cannot add null element.
LinkedListTest	add()	Setup38	data = "Element A", "Element A"	Linked list allows duplicates: ["Element A", "Element A"]
LinkedListTest	remove()	Setup36	data = "Element A", then "Element B", then "Element C"	Linked list contains ["Element A", "Element C"] after removing "Element B".

LinkedListTest	remove()	Setup37	data = ""	Error: Element not found.
LinkedListTest	remove()	Setup38	"Element A", "Element A"	List becomes ["Element C"]
LinkedListTest	search()	Setup36	data = "Element A", then "Element B", then "Element C"	Linked list contains ["Element A", "Element C"] after removing "Element B".
LinkedListTest	search()	Setup37	data = ""	Not found.
LinkedListTest	search()	Setup38	data = "Element A", "Element A"	Found "Element A" at head position.
QueueTest	enqueue()	Setup39	data = 5, then 10, then 15	Queue contains [5, 10, 15] in insertion order.
QueueTest	enqueue()	Setup40	data = “”	Error: Cannot enqueue null value.
QueueTest	enqueue()	Setup41	data =”10”,”5”, “5”	Queue allows duplicates: [10, 5, 5]
QueueTest	dequeue()	Setup39	[{ "data": “5” }, { "data": “10” }, { "data": “15” }]	Returns and removes first element (5); queue becomes [10, 15].
QueueTest	dequeue()	Setup40	[{ "data": }]	Error: Cannot dequeue from empty queue.
QueueTest	dequeue()	Setup41	[{ "data": “5” }]	Queue becomes empty after one dequeue.
StackTest	push()	Setup42	data = "X", then "Y", then "Z"	Stack contains ["X", "Y", "Z"] ("Z" on top).
StackTest	push()	Setup43	null	Error: Cannot push null element.
StackTest	push()	Setup44	data = "X", then "X"	Stack contains two "X" entries.
StackTest	pop()	Setup42	[{ "data": "X" }, { "data": "Y" }, { "data": "Z" }]	Returns and removes last inserted element ("Z"); stack becomes ["X", "Y"].
StackTest	pop()	Setup43	[{ "data": "" }]	Error: Cannot pop from empty stack.
StackTest	pop()	Setup44	[{ "data": "X" }]	Stack becomes empty after pop.
HashTableTest	put()	Setup45	[{ "key": "A", "value": 1 },	Hash table contains {"A":1, "B":2}.
HashTableTest	put()	Setup46	[{ "key": "", "value": }]	Error: Key cannot be null.

HashTableTest	put()	Setup45	overwrite key="A" with value=99	{"A":99, "B":2} (A updated)
HashTableTest	delete()	Setup45	[{ "key": "A", "value": 1 }, { "key": "B", "value": 2 }]	Hash table contains {"B":2} after removing "A".
HashTableTest	delete()	Setup46	[{ "key": "", "value": }]	Error: Key not found.
HashTableTest	delete()	Setup47	[{ "key": "A", "value": 1 }]	HashTable becomes empty.
HeapTest	insert()	Setup48	[{ "data": 10 }, { "data": 20 }, { "data": 5 }]	Heap organizes internally to maintain heap property (e.g., min-heap would be [5, 20, 10]).
HeapTest	insert()	Setup49	[{ "data": “” }]	Error: Cannot insert null into heap.
HeapTest	insert()	Setup50	duplicate values (10, 10, 5)	Heap handles duplicates properly.
HeapTest	removeTop()	Setup48	[{ "data": 10 }, { "data": 20 }, { "data": 5 }]	Removes the minimum (5 in min-heap) or maximum (20 in max-heap) depending on heap type.
HeapTest	removeTop()	Setup49	[{ "data": “” }]	Error: Cannot remove from empty heap.
HeapTest	removeTop()	Setup50	[{ "data": “10” , "data": “10” }]	Next smallest/largest becomes top.
PriorityQueue Test	enqueue()	Setup48	[{ "data": 10 }, { "data": 20 }, { "data": 5 }]	Priority queue organizes elements so the highest/lowest priority is accessible first.
PriorityQueue Test	enqueue()	Setup49	[{ "data": “” }]	Error: Cannot enqueue null.
PriorityQueue Test	enqueue()	Setup50	duplicate priority values (10, 10)	Handles same-priority elements properly.
PriorityQueue Test	dequeue()	Setup51	[{ "data": 10 }, { "data": 20 }, { "data": 5 }]	Removes and returns the element with the highest/lowest priority.
PriorityQueue Test	dequeue()	Setup52	[{ "data": “” }]	Error: Cannot dequeue from empty queue.
PriorityQueue Test	dequeue()	Setup53	[{ "data": “10” , "data": “10” }]	Returns one "10", leaves another properly ordered.