

Test Cases

Black-Box Test Cases

These test cases are defined based on the available documentation and execution of the program. The code was not inspected.

#	Test case (very brief description)	Preconditions (any required setup)	Test steps (steps executed during testing)	Expectation	Observation ("pass" or failure description)
1	Application Launch	None	1. Launch the application.	The application starts and displays the main screen	Pass
2	Main Screen Display	The application is launched.	1. Observe the main screen displayed at the start.	The main screen prompts the user with the question: "What is your field of interest?"	Pass
3	Input Field of Interest - Valid Input	The application is asking for the field of interest.	1. Enter a valid field of interest (e.g. "computer science") and press Enter or click Send.	The input is accepted, and the application proceeds to ask for the program.	Pass
4	Input Field of Interest - Invalid Input	The application is asking for the field of interest.	1. Enter an invalid field of interest (e.g. "history") and press Enter or click Send.	An alert is shown indicating "Invalid Input. Please enter a valid field of interest."	Pass
5	Input Program - Valid Input	The application is asking for the program.	1. Enter a valid program (e.g. "MSc") and press Enter or click Send.	The input is accepted, and the application proceeds to ask for the city.	Pass
6	Input Program - Invalid Input	The application is asking for the program.	1. Enter an invalid program (e.g. "bachelor") and press Enter or click Send.	An alert is shown indicating "Invalid	Pass

				Input. Please enter a valid program."	
7	Input City - Valid Input	The application is asking for the city.	1. Enter a valid city (e.g. "Munich") and press Enter or click Send.	The input is accepted, and the application proceeds to generate recommendations.	Pass
8	Input City - Invalid Input	The application is asking for the city.	1. Enter an invalid city (e.g. "Paris") and press Enter or click Send.	An alert is shown indicating "Invalid Input. Please enter a valid city."	Pass
9	University Recommendations	Valid inputs for field of interest, program, and city have been provided.	1. Observe the university recommendations provided by the application.	The application lists universities that match the criteria or indicates that no matches were found.	Pass
10	No Matching Universities	The CSV data contains no universities matching the given criteria.	Enter valid inputs for field of interest, program, and city that do not match any university in the CSV data.	The application indicates "No universities found matching your criteria."	Pass
11	Input Field of Interest - Unsupported Characters	The application is asking for the field of interest.	1. Enter a field of interest with unsupported characters (e.g. "comp\$uter sci&ence") and press Enter or click Send.	The input should be rejected with an appropriate alert message.	Pass
12	Input Program - Numeric Input	The application is asking for the program.	1. Enter a numeric input (e.g. "12345") and press Enter or click Send.	The input should be rejected with an appropriate alert message.	Pass
13	Input City - Extra Spaces	The application is asking for the city.	1. Enter a city name with leading or trailing spaces (e.g. " Munich ") and press Enter or click Send.	The input should be accepted and processed correctly without extra spaces.	Pass

14	Input Program - Case Sensitivity	The application is asking for the program.	1. Enter a valid program with different casing (e.g., "msc" instead of "MSc") and press Enter or click Send.	The input should be accepted regardless of case.	Pass
15	Empty Input Handling	The application is asking for any input (field of interest, program or city).	1. Press Enter or click Send without entering any input.	The input should be rejected with an appropriate alert message.	Pass

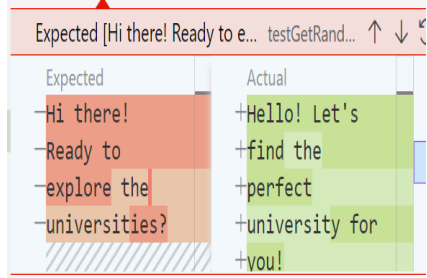
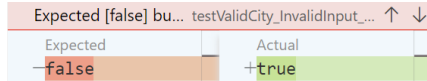
White-Box Test Cases

These additional test cases were defined during inspection of the code.

Class: **AppGUI**

#	Test case (very brief description)	Preconditions (any required setup)	Test steps (steps executed during testing)	Expectation	Observation ("pass" or failure description)
1	Verify that the getRandomGreeting method returns a valid greeting	An instance of AppGUI is created	<ol style="list-style-type: none"> 1. Create an instance of the AppGUI class. 2. Call the getRandomGreeting method. 3. Check if the returned greeting is one of the expected greetings from the GREETINGS array. 	The method should return a valid greeting from the GREETINGS array.	p
2	Verify that the getRandomGreeting method returns different greetings on subsequent calls.	N	<ol style="list-style-type: none"> 1. Create an instance of the AppGUI class. 2. Call the getRandomGreeting 	The method should return different greetings on subsequent calls.	Pass

			<p>method and store the returned greeting.</p> <ol style="list-style-type: none"> 3. Call the getRandomGreeting method again and store the returned greeting. 4. Check if the two greetings are different. 		
3	Verify that the getRandomGreeting method returns a non-empty string	<p>An instance of AppGUI is created.</p> <p>currentQuestionIndex is set to 0.</p> <p>chatArea is initialized as a VBox.</p>	<ol style="list-style-type: none"> 1. Create an instance of the AppGUI class. 2. Call the getRandomGreeting method and store the returned greeting. 3. Check if the returned greeting is not an empty string. 	The method should return a non-empty string.	P

4	Verify that the getRandomGreeting method returns a non-null string		<ol style="list-style-type: none"> 1. Create an instance of the AppGUI class. 2. Call the getRandomGreeting method and store the returned greeting. 3. Check if the returned greeting is not null. 	The method should return a non-null string.	<p>Fail</p> 
5	Verify that the getRandomGreeting method returns unique greetings	N	<ol style="list-style-type: none"> 1. Create an instance of the AppGUI class. 2. Call the getRandomGreeting method repeatedly and store the returned greetings in an array. 3. Check if the array contains all the unique greetings from the GREETINGS array. 	The method should return all unique greetings from the GREETINGS array.	p
6	Verify that the validateInput method accepts a valid city input.	Set the currentQuestionIndex to 2 (assuming	<ol style="list-style-type: none"> 1. Create an instance of the AppGUI class. 	The method should return true for a valid city input.	<p>Fail</p> 

		this is the index for the city question).	<ol style="list-style-type: none"> Set the currentQuestionIndex to 2. Call the validateInput method with a valid city input (e.g., "munich"). Check if the method returns true. 		
7	Verify that the validateInput method rejects an invalid city input (case-sensitive)	Set the currentQuestionIndex to 2 (assuming this is the index for the city question).	<ol style="list-style-type: none"> Create an instance of the AppGUI class. Set the currentQuestionIndex to 2. Call the validateInput method with an invalid city input (e.g., "MUNICH"). Check if the method returns false. 	The method should return false for an invalid city input (case-sensitive)	p
8					
9					

Class: **CSVDataReaderTest**

#	Test case (very brief description)	Preconditions (any required setup)	Test steps (steps executed during testing)	Expectation	Observation ("pass" or failure description)
---	---------------------------------------	---------------------------------------	---	-------------	--

1	Constructor	A valid file Path is inputted	<ol style="list-style-type: none"> 1. Performed checks if the file properly reads .csv files in file paths. 2. check if a file path with file not a .csv format enter if it returns an argument exception 	Expected to read a valid file path and a valid .csv file in the path directory and throw an exception if the file is not a CSV	Fail
2	readCSVDataReader Method	None	<ol style="list-style-type: none"> 1. performed test to check if the file in the file pass is Null 2. Performed test to check if the file in the file path is empty. 3. Takes the csv file from the constructor and accesses the data inside through the header. 	It is expected for the method to read all datas in the csv file correctly based on the header specified	Fail <ol style="list-style-type: none"> 1. "String GmatorGRERequirement = csvRecord.get("Requirements");" is suppose to return GRE requirement but it is returning GPA requirements. 2. "String cgpaRequirement = csvRecord.get("GRE/GMAT");" is expected to return GPA requirement but it return GRE requirements

Class: **University**

#	Test case (very brief description)	Preconditions (any required setup)	Test steps (steps executed during testing)	Expectation	Observation ("pass" or failure description)
1	Constructor and Getters	None	1. Create a 'University' object with the following values:	The getter methods return the values	Pass

			<ul style="list-style-type: none"> - Field of Interest: "computer science" - Program: "master" - City: "munich" - Name: "Technical University of Munich" - GRE/GMAT Requirement: "GRE Score: 325" - CGPA Requirement: "Minimum GPA: 3.2" <p>2. Call getter methods to retrieve each value.</p>	passed to the constructor.	
2	ToString Method	None	<p>1. Create a 'University' object with the following values:</p> <ul style="list-style-type: none"> - Field of Interest: "computer science" - Program: "master" - City: "munich" - Name: "Technical University of Munich" - GRE/GMAT Requirement: "GRE Score: 325" - CGPA Requirement: "Minimum GPA: 3.2" <p>2. Call the 'toString' method.</p>	The 'toString' method returns a string representation of the 'University' object matching the expected format.	Pass
3	Set Field of Interest	A 'University' object is created.	<p>1. Call 'setFieldOfInterest' with the value "biology".</p> <p>2. Call 'getFieldOfInterest' to verify the change.</p>	The field of interest is updated to "biology".	Pass
4	Set Program	A 'University' object is created.	<p>1. Call 'setProgram' with the value "MBA".</p> <p>2. Call 'getProgram' to verify the change.</p>	The program is updated to "MBA".	Pass

5	Set City	A 'University' object is created.	1. Call 'setCity' with the value "Berlin". 2. Call 'getCity' to verify the change.	The city is updated to "Berlin".	Pass
6	Set Name	A 'University' object is created.	1. Call 'setName' with the value "Humboldt University of Berlin". 2. Call 'getName' to verify the change.	The name is updated to "Humboldt University of Berlin".	Pass
7	Set GRE/GMAT Requirement	A 'University' object is created.	1. Call 'setGmatorGRERequirement' with the value "GMAT Score: 700". 2. Call 'getGmatorGRERequirement' to verify the change.	The GRE/GMAT requirement is updated to "GMAT Score: 700".	Pass
8	Set CGPA Requirement	A 'University' object is created.	1. Call 'setCgpaRequirement' with the value "Minimum GPA: 3.5". 2. Call 'getcpgaRequirement' to verify the change.	The CGPA requirement is updated to "Minimum GPA: 3.5".	Pass
9	Set Field of Interest Empty String	A 'University' object is created.	1. Call 'setFieldOfInterest' with an empty string. 2. Call 'getFieldOfInterest' to verify the change.	The field of interest is updated to an empty string.	Pass
10	Set Field of Interest Null	A 'University' object is created.	1. Call 'setFieldOfInterest' with 'null'. 2. Call 'getFieldOfInterest' to verify the change.	The field of interest is updated to 'null'.	Pass
11	Set Program Empty String	A 'University' object is created.	1. Call 'setProgram' with an empty string. 2. Call 'getProgram' to verify the change.	The program is updated to an empty string.	Pass
12	Set Program Null	A 'University' object is created.	1. Call 'setProgram' with 'null'. 2. Call 'getProgram' to verify the change.	The program is updated to 'null'.	Pass
13	Set City Empty String	A 'University' object is created.	1. Call 'setCity' with an empty string. 2. Call 'getCity' to verify the change.	The city is updated to an empty string.	Pass
14	Set City Null	A 'University' object is created.	1. Call 'setCity' with 'null'. 2. Call 'getCity' to verify the change.	The city is updated to 'null'.	Pass
15	Set Name Empty String	A 'University' object is created.	1. Call 'setName' with an empty string. 2. Call 'getName' to verify the change.	The name is updated to an empty string.	Pass

16	Set Name Null	A 'University' object is created.	1. Call 'setName' with 'null'. 2. Call 'getName' to verify the change.	The name is updated to 'null'.	Pass
17	Set GRE/GMAT Requirement Empty String	A 'University' object is created.	1. Call 'setGmatorGRERequirement' with an empty string. 2. Call 'getGmatorGRERequirement' to verify the change.	The GRE/GMAT requirement is updated to an empty string.	Pass
18	Set GRE/GMAT Requirement Null	A 'University' object is created.	1. Call 'setGmatorGRERequirement' with 'null'. 2. Call 'getGmatorGRERequirement' to verify the change.	The GRE/GMAT requirement is updated to 'null'.	Pass
19	Set CGPA Requirement Empty String	A 'University' object is created.	1. Call 'setCgpaRequirement' with an empty string. 2. Call 'getcpgaRequirement' to verify the change.	The CGPA requirement is updated to an empty string.	Pass
20	Set CGPA Requirement Null	A 'University' object is created.	1. Call 'setCgpaRequirement' with 'null'. 2. Call 'getcpgaRequirement' to verify the change.	The CGPA requirement is updated to 'null'.	Pass
21	case-insensitive for the fieldOfInterest field	when user type in chat Biology	expected is biology (from excel database), however when useer type Biology then test occurs	biology	Fail