Few clarifications about the final project

Scenario 1:

1. Firstly, I am guaranteed that I am on the specified URL which was previously saved in the Configuration Reader.
2. I clicked on the one-way type button using the click method.
3. I entered the destination and departure city using send keys and finally clicked on the first suggestion that appeared.
4. I set the number of passengers to 2 using a dropdown.
5. I also selected the Economy cabin class.
6. I entered the specified date.
7. I clicked on the search button.
8. I clicked on the first flight option.
9. I clicked on the book now button. The CSS of this button was stored in a list of Web elements since there are many book now elements on the page and I called the first button by using an index of 0 to click on the first Book Now button.
10. I entered the passenger attributes, such as name, surname, email, confirm email, and phone number using send keys.
11. I clicked on seats for two passengers. Here as well, I stored the seats in a list and called them using specific indexes. Finally, I clicked on the save button.
12. I entered the CVV, clicked on the agree button and confirmed.
13. Finally, I called the verifyMsgIsDisplayed method, which verifies that the confirmation table has appeared with the specified text and printed the booking number to the console. I obtained the booking number using the getText() method, which I then trimmed. To verify that the text has been displayed, I stored the text in a string and checked whether this substring belongs to the text contained in the confirmation table.

Scenario 2: Some of the methods are the same and the logic used is the same as well. Moreover, the methods are very clear in the code.The 11th step : It is a method called `verifyTravelerNames` that takes an array of expected traveler names as input. The method uses a for loop to iterate through each traveler name displayed on the confirmation page, which is obtained by using the `travelerNames` locator from the `ConfirmPageElements` class.For each traveler name, the actual name is obtained using the `getText()` method and is then compared to the corresponding expected name from the input array using the `Assert.assertEquals()` method. If the actual name does not match the expected name, the test will fail and an AssertionError will be thrown.Overall this method is used to verify that all the travelers' names displayed on the confirmation page match the expected names provided as input, and to ensure that the booking was made correctly for the intended travelers.

Scenario 3: The same steps are followed in this scenario as well. However, I would like to specify that I created a wait before entering the city names to select the first suggestions, and then pressed the enter button.

Very important: Each scenario has its own specific details that can only be seen specifically in the code. In my project, apart from running the code from the specified features, I have also created a main method to test the specified scenarios. **The project has been created based on all the given conditions.** Due to numerous changes in my code, certain attributes have been set, which made it possible to successfully test the code. These attributes can be easily changed. Additionally, I will attach some videos and photos to confirm this fact. Every scenario has been tested successfully but some of the videos are made yesterday when the website had many problems, so the videos are all not captured till the end. I have added to many waits also that I needed during the implementation of the project.

Thank you very much for giving me this opportunity. I had a great time and I hope that my performance was very good! I have learned many new things thanks to you.