Skybot

by Technica

Joi Anderson - jna2123
Lesley Cordero - lc2958
Maria Javier - mj2729
Klarizsa Padilla - ksp2127

Second Iteration

Github Repository:

https://github.com/TeamTechnica/Skybot

Change Suggestions from IA:

None. However, Ashna, our Project Mentor, was pleased with our progress and encouraged the use of CodeCov as our code coverage tool.

Equivalence Partitions & Boundary Conditions

Scenario 1

A user is prompted to provide their Columbia UNI, so that they can be sent a verification email.

Valid input: 2-3 letters ['a'-'z' 'A'-'Z'] followed by 4 digits.

Invalid input: anything else.

Test Scenario #	Test Scenario Description	Expected Outcome
1	Enter a UNI with fewer than 6 characters Example: User enters 'a2124'	System should not accept
2	Enters a UNI with 2-3 letters followed by 4 digits. Example: User enters 'abc2123'	System should accept
3	Enter a UNI with more than 7 characters Example: User enters '21234'	System should not accept

Note: Users are restricted to students

Scenario 2

A user is prompted to provide the verification code that was sent to their email.

Valid input: 6 digit number [0-9] with no leading 0.

Invalid input: anything else.

Test Scenario #	Test Scenario Description	Expected Outcome
1	Enter a numbers with fewer than 6 digit number Example: User enters 99999	System should not accept
2	Enters a UNI with 2-3 letters followed by 4 digits. Example: User enters 100000	System should accept
3	Enter a number with 7 or more digit Example: User enters ‘21234’	System should not accept

Scenario 3

A user is prompted to choose which airport they want to travel to with a list of three options:
(1) JFK , (2) LGA, (3) EWR.

Valid input: a single digit from 1 - 3.

Invalid input: anything else.

Test Scenario #	Test Scenario Description	Expected Outcome
1	Enter a numbers a number less than 1 Example: User enters 0	System should not accept
2	Enters a number between 1 and 3 Example: User enters 2	System should accept
3	Enter a number greater than 3 Example: User enters 4	System should not accept

Scenario 4

A user is prompted to enter the date of their flight. They are promoted to enter the date in the following format MM-DD-YYYY

Valid input: 2 digits from 01 to 12, followed by a dash, 2 digits from 01 to 31, followed by a dash, and a year that equal to or greater than the current year. This date must be equal to or after the current date.

Invalid input: anything else.

Test Scenario #	Test Scenario Description	Expected Outcome
1	Enter a numbers fewer than eight digits Example: 1210	System should not accept
2	Enters an eight digit number Example: 11102019	System should accept
3	Enter a number greater than eight digits Example: 1110201951	System should not accept

Scenario 5

A user is prompted to enter the departure time of their flight in form HHMMSS in 24 hour military form.

Valid input: 6 digits from 1-24 for the first two digits 1-60 for the second and third set of two digits.

Invalid input: anything else.

Test Scenario #	Test Scenario Description	Expected Outcome
1	Enters a number with fewer than 6 digits Example: User enters 10101	System should not accept
2	Enters a number with 6 digits Example: User enters 112519	System should accept
3	Enters a number with more than 6 digits Example: User enters 11252019	System should not accept

Scenario 6

A user is prompted to enter the maximum number of additional passengers they are willing to include in their ride. They are permitted to have up to additional people.

Valid input: a single digit from 1 - 2.

Invalid input: anything else.

Test Scenario #	Test Scenario Description	Expected Outcome
1	Enters a number that is less than 1 Example: User enters 0	System should not accept
2	Enters a number between 1 to 2 Example:User enters 2	System should accept
3	Enters a number greater than 2 Example: User enters 3	System should not accept

Our Test Suite

Our test suite can be found in the test directory in our repository:

https://github.com/TeamTechnica/Skybot/tree/master/test

Derived Use Cases

At this stage in the development process, we were able to demo the following user story:

- Name: Account Creation
- Summary: A mobile number texts the Skybot, and provided the user responds with the required information, a Skybot account is created for the user
- Rationale: While looking for a passenger to carpool with, many users find that if there is not a match for them right away, they would like to be contacted later if someone else has similar flight credentials. The account creation feature allows for their data to be able to be stored and them to later be contacted if someone queries the database with similar flight information. At other times, the

user may receive a match on their first query of the database and like to reference their match details at a later time.

- Users: All users
- Preconditions: An account with the user's uni and queried flight information has not already been created
- Basic Course of Events:
    1. A user texts Skybot an initializing text.
    2. Skybot responds with a request for their uni.
    3. The user responds with their uni
    4. Skybot messages uni@columbia.edu with a verification code
    5. The user responds to Skybot's text with their verification code.
    6. Skybot requests the user's flight information and stores it in the database
- Alternative Path(s): The user has an account created with their uni but would like to add another flight to request a match for. The postcondition is the same, aside from the fact the the user's contact information is not added to the database again.
- Postconditions: The user's contact and flight information is stored in the database

Measuring the Branch Coverage

CodeCov is the tool our team is using for measuring the code coverage of our Python Skybot code. CodeCov monitors the Skybot program, noting which parts of the code have been executed, then analyzes the source to identify code that could have been executed but was not. We use our CodeCov measurements to gauge the effectiveness of tests. It shows us which parts of the Skybot code are being exercised by tests, and which are not.