Clist Bot (Discord Bot)

Software Testing

Tatvam(B20CS077)
Maniya Yash Rajeshbhai(B20CS033)

1. Black Box Testing Technique

- Command - register :

  It registers the handle given to the discord server.

  ;register Tatvam_Shah :- registers the given handle on the discord server for the user who wrote the command on the server.

- Command - meme :

  It gives a random programming related meme from reddit. No argument is passed in this command.

  ;meme :- gives a meme.

  ;meme 10 :- gives a meme.

- Command - clear :

  It takes positive integers as an argument and clears the latest n messages from the channel.

  ;clear -1 :- No action is taken on the discord server .

  ;clear 3 :- Clears latest 3 messages of the channel.

  ;clear 1.2 :- No action is taken on the discord server .

- Command - stalk :

  It takes codeforces handle as an argument and gives information about that particular codeforces handle.

  ;stalk Tatvam_Shah :- Gives information about the codeforces handle Tatvam_Shah since it is a registered handle on codeforces website.

  ;stalk codeheck_00235 :- No output is shown on the channel where the user gives the command.

2. Boundary Value Analysis

- Command - clear :

  It takes positive integers as an argument and clears the latest n messages from the channel.

  ;clear -1 :- No action is taken on the discord server where the bot is being used while there is an error at the heroku server.

  ;clear 3 :- Clears latest 3 messages of the channel.

  ;clear 1.2 :- No action is taken on the discord server where the bot is being used while there is an error at the heroku server.

- aliases :

  For clear command there are other aliases defined in the code : clr, delete, Clear.

  ;clr 3 :- Clears latest 3 messages of the channel.

  ;clr -1 :- No action is taken on the discord server where the bot is being used while there is an error at the heroku server.

;cleare 1 :- No action is taken on the discord server where the bot is being used while there is an error at the heroku server since there is no alias of clear defined as cleare in the code.

- Command - clist future :

  It takes a positive integer as an argument and gives the given number of future contests scheduled on codeforces platform.

  ;clist future 2 :- Gives the information about the latest upcoming 2 contests on codeforces.

  ;clist future -1 :- No output is shown on the channel where the user gives the command but there is an error at the heroku server since the number isn't a positive integer.

  ;clist future 1.7 :- No output is shown on the channel where the user gives the command but there is an error at the heroku server since the number isn't a positive integer.

- Command - stalk() :

  It takes codeforces handle as an argument and gives information about that particular codeforces handle.

  ;stalk Tatvam_Shah :- Gives information about the codeforces handle Tatvam_Shah since it is a registered handle on codeforces website.

  ;stalk 1994.00$#@* :- No output is shown on the channel where the user gives the command but there is an error at the heroku server since the number isn't a positive integer.

3. Equivalence Class Partitioning

- Command - help :

  It gives all the possible sub commands for a given command or the information about the command

  Equivalence Class Partioning :

Invalid	Valid
;help stalke	;help stalk
;help 123	;help future

- Command - clear :

  It takes positive integers as an argument and clears the latest n messages from the channel.

  Equivalence Class Partioning :

Invalid	Valid
;clear -2	;clear 2
;clear 4.8	;clear 21

- Command - stalk :

  It takes codeforces handle as an argument and gives information about that particular codeforces handle.

  Equivalence Class Partioning :

Invalid	Valid
;stalk #$%	;stalk Tatvam_Shah
;stalk %^&-=!@5	;stalk Mangalam_01

- Command - clist future :

  It takes a positive integer as an argument and gives the given number of future contests scheduled on codeforces platform.

  Equivalence Class Partitioning :

Invalid	Valid
;clist future -2	;clist future 10
;clist future @*	;clist future
;clist future xd	;clist future long
;clist future 5.8	;clist future short

4. Integration test cases

Integration testing tests integration or interfaces between components, interactions to different parts of the system such as an operating system, file system and hardware or interfaces between systems.

Discord Application Authentication :

- In our case, we are making a discord bot which needs authentication from the discord app developers portal.
- This is done by using a unique discord bot TOKEN. This token is to be supplied to the heroku server where our script is getting executed.
- If the wrong token is included in the script, the discord bot will not come online and the user can't use it.

Heroku - Github Integration :

- We are using Github as our version control system. So, the code in the github repository needs to integrate with the heroku server to execute the same code on its server backend.
- If the heroku-github integration is not done correctly, the bot will crash.

Script - API Database Integration :

- The API is used when a particular command requests some data from Codeforces / Reddit API. And if the user passes wrong inputs to the command, API returns an error message in the request fetched by our script.

5. Script for Automated Testing :

We can test the bot manually on discord server channels (common practice).

However as mentioned in the question, we are going to automate the testing process using the cordejs library.

Code goes here :

```
const { group, test, command, beforeStart, afterAll, action } = require("corde");
const { client, loginBot } = require("discord.js");

beforeStart(() => {
    loginBot({username:'Yash_Maniya',auth:'your_password'});
});

group("main commands", () => {
    test(";help jsahdkjahs", () => {
        expect().shouldReturn(None);
    });
});
setTimeout(() => {  console.log("Testing Next Command"); }, 2000);

group("main commands", () => {
    test(";help clist", () => {
        expect("Clist Future Help Portal").shouldReturn(discord.Embed);
    });
});
setTimeout(() => {  console.log("Testing Next Command"); }, 2000);

group("main commands", () => {
    test(";clear -15.2", () => {
        expect(discord.ext.commands.errors.BadArgument).shouldReturn(None);
    });
});
setTimeout(() => {  console.log("Testing Next Command"); }, 2000);

group("main commands", () => {
    test(";clear 5", () => {
        expect().action(discord.Channel.prune(5));
    });
```

```
});

setTimeout(() => {  console.log("Testing Next Command"); }, 2000);

group("main commands", () => {
    test(";register unregistered_handle", () => {
        expect().shouldReturn(None);
    });
});
setTimeout(() => {  console.log("Testing Next Command"); }, 2000);

group("main commands", () => {
    test(";register codheck_0", () => {
        expect().action(UpdateAttr);
    });
});
setTimeout(() => {  console.log("Testing Next Command"); }, 2000);

group("main commands", () => {
    test(";clist 10 future", () => {
        expect(discord.ext.commands.errors.CommandInvokeError).return(None);
    });
});
setTimeout(() => {  console.log("Testing Next Command"); }, 2000);

group("main commands", () => {
    test(";clist future 6", () => {
        expect(discord.Embed).return(discord.Embed);
    });
});
setTimeout(() => {  console.log("Testing Next Command"); }, 2000);

afterAll(() => {
    client.destroy();
});
```