VIKAS K R
TAS241

SHELL SCRIPT

Write a shell script to change the values in a file(i.e sig.conf) according to the input passed to the script. The script should ask for all four inputs from the user & also validate the input. Below are the details of input. In full bracket options are given, you have to restrict the user pass single value for each input from the provided options in the full bracket.
Input:-
1) Component Name [INGESTOR/JOINER/WRANGLER/VALIDATOR]
2) Scale **[MID/HIGH/LOW]**
3) View **[Auction/Bid]**
4) Count **[single digit number]**
Explanation of a conf file line.
<view> ; <scale> ; <component name> ; ETL ; vdopia-etl= <count>

Note:- vdopiasample stands for Auction & vdopiasample-bid is for Bid
The script should change the values in the file according to the input provided. At a time only one line of the conf file should be altered.

Step 1:

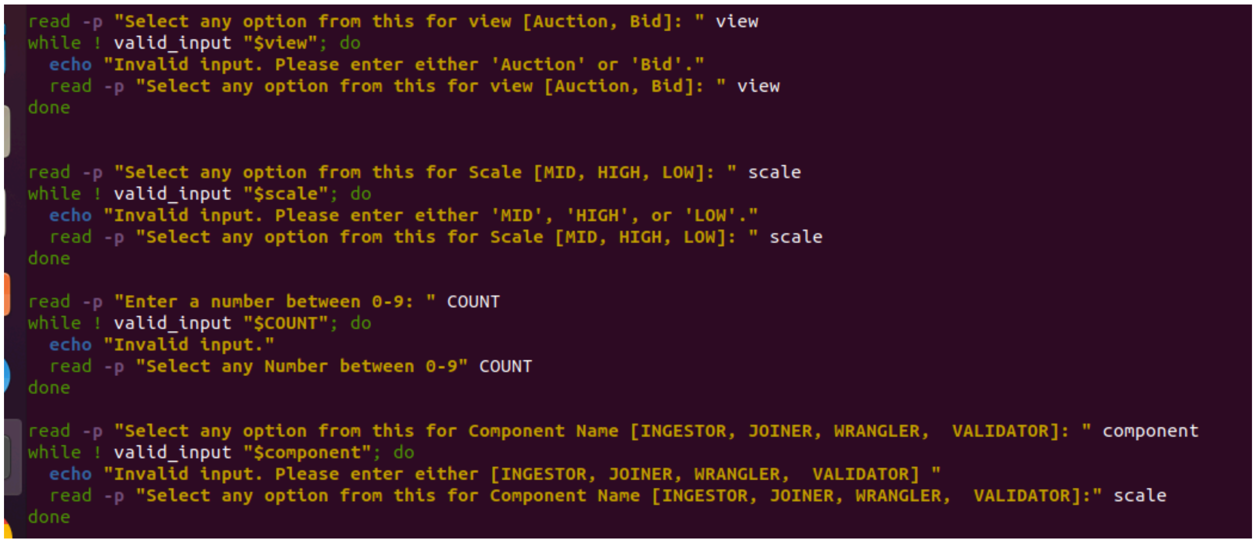
Create a **Sig.conf :-**
The Sig.conf file is a configuration file that follows a specific format. Each line in the file represents a configuration entry.
format:- <view> ; <scale> ; <component name> ; ETL ; <prefix>= <count>

Step 2:

Create a shell Script **script.sh** that validates the inputs, constructs the appropriate pattern, and replaces the matching line in the file. Here's a detailed step-by-step shell script to meet the requirements:

1. Taking input from user

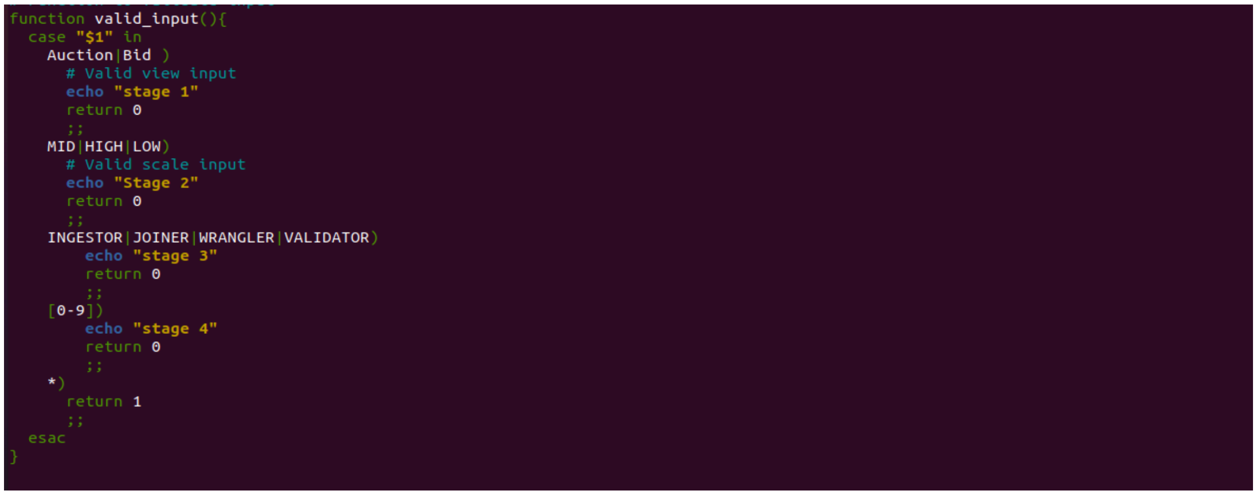
- In this script it asks the user to input values for `view`, `scale`, `COUNT`, and `component`.
- Each input is checked using the `valid_input` function to ensure it matches acceptable values. The script keeps prompting the user until a valid input is provided.
- If the user inputs an invalid value, the script displays an error message and re-prompts until a valid value is entered.



```
read -p "Select any option from this for view [Auction, Bid]: " view
while ! valid_input "$view"; do
  echo "Invalid input. Please enter either 'Auction' or 'Bid'."
  read -p "Select any option from this for view [Auction, Bid]: " view
done


read -p "Select any option from this for Scale [MID, HIGH, LOW]: " scale
while ! valid_input "$scale"; do
  echo "Invalid input. Please enter either 'MID', 'HIGH', or 'LOW'."
  read -p "Select any option from this for Scale [MID, HIGH, LOW]: " scale
done

read -p "Enter a number between 0-9: " COUNT
while ! valid_input "$COUNT"; do
  echo "Invalid input."
  read -p "Select any Number between 0-9" COUNT
done

read -p "Select any option from this for Component Name [INGESTOR, JOINER, WRANGLER,  VALIDATOR]: " component
while ! valid_input "$component"; do
  echo "Invalid input. Please enter either [INGESTOR, JOINER, WRANGLER,  VALIDATOR] "
  read -p "Select any option from this for Component Name [INGESTOR, JOINER, WRANGLER,  VALIDATOR]:" scale
done
```

2. Input Validation

- The valid_input function is designed to validate different types of input based on what you provide. It uses the case statement to handle different types of input.
- **Valid Inputs**: If the input matches one of the patterns (like Auction, MID, or a digit), the function prints a specific message and returns 0, indicating that the input is valid.
- **Invalid Inputs**: If the input does not match any of the patterns, the function returns 1, indicating that the input is invalid.

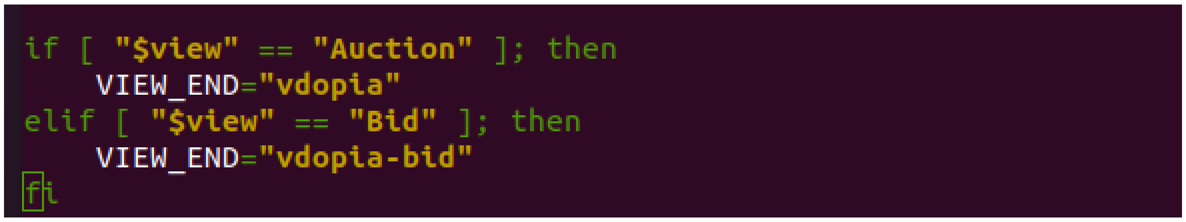


```
function valid_input(){
  case "$1" in
    Auction|Bid )
      # Valid view input
      echo "stage 1"
      return 0
      ;;
    MID|HIGH|LOW)
      # Valid scale input
      echo "Stage 2"
      return 0
      ;;
    INGESTOR|JOINER|WRANGLER|VALIDATOR)
        echo "stage 3"
        return 0
        ;;
    [0-9])
        echo "stage 4"
        return 0
        ;;
    *)
      return 1
      ;;
  esac
}
```

```
function valid_input(){
 case "$1" in
   Auction|Bid )
                                                            # Valid view input

     echo "stage 1"
     return 0
     ;;
   MID|HIGH|LOW)
                                                            # Valid scale input

     echo "Stage 2"
     return 0
     ;;
   INGESTOR|JOINER|WRANGLER|VALIDATOR)
                                                            # Valid component input

     echo "stage 3"
     return 0
     ;;
   [0-9])
                                                      # Valid count input (single digit number)

     echo "stage 4"
     return 0
     ;;
   *)
                                                            # Invalid input

     return 1
     ;;
 esac
}
```

3. Conditional Check

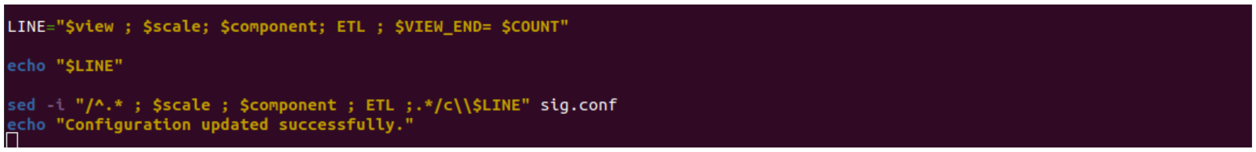
- The script checks the value of the `view` variable. If it is `"Auction"`, it sets `VIEW_END` to `"vdopia"`. If it is `"Bid"`it sets `VIEW_END` to `"vdopia-bid"`.
- Based on the value of `view`, `VIEW_END` is assigned one of two possible strings: either `"vdopia"` or `"vdopia-bid"`.



```
if [ "$view" == "Auction" ]; then
    VIEW_END="vdopia"
elif [ "$view" == "Bid" ]; then
    VIEW_END="vdopia-bid"
fi
```

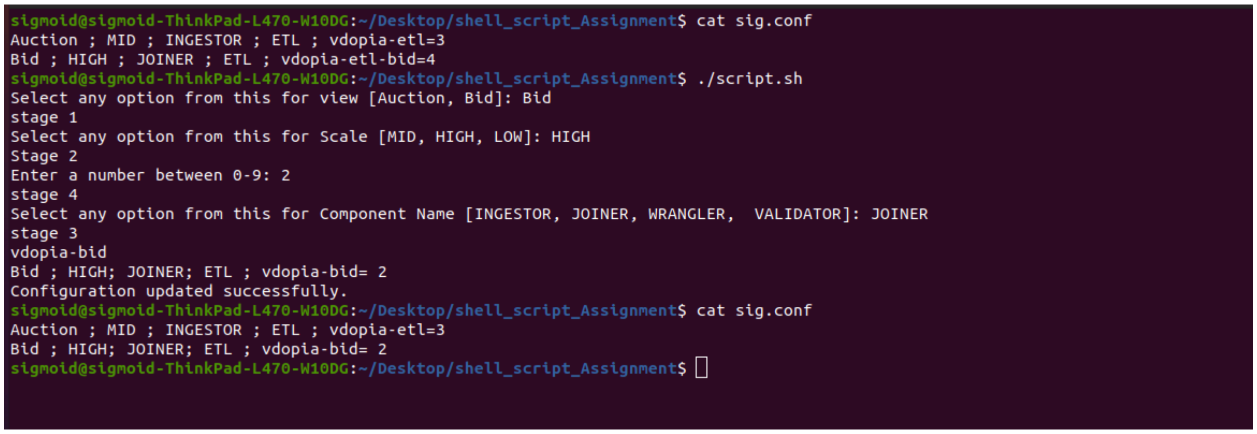
4.Combing all the value and replace in the file

- The sed command searches sig.conf for a line matching the pattern with $scale, $component, and ETL, and replaces it with the new LINE.
- The `LINE` variable is created by combining the values of `$view`, `$scale`, `$component`, `ETL`, `$VIEW_END`, and `$COUNT` into a single string.



```
LINE="$view ; $scale; $component; ETL ; $VIEW_END= $COUNT"

echo "$LINE"

sed -i "/^.* ; $scale ; $component ; ETL ;.*/c\\$LINE" sig.conf
echo "Configuration updated successfully."
```

5. OUTPUT:



```
sigmoid@sigmoid-ThinkPad-L470-W10DG:~/Desktop/shell_script_Assignment$ cat sig.conf
Auction ; MID ; INGESTOR ; ETL ; vdopia-etl=3
Bid ; HIGH ; JOINER ; ETL ; vdopia-etl-bid=4
sigmoid@sigmoid-ThinkPad-L470-W10DG:~/Desktop/shell_script_Assignment$ ./script.sh
Select any option from this for view [Auction, Bid]: Bid
stage 1
Select any option from this for Scale [MID, HIGH, LOW]: HIGH
Stage 2
Enter a number between 0-9: 2
stage 4
Select any option from this for Component Name [INGESTOR, JOINER, WRANGLER,  VALIDATOR]: JOINER
stage 3
vdopia-bid
Bid ; HIGH; JOINER; ETL ; vdopia-bid= 2
Configuration updated successfully.
sigmoid@sigmoid-ThinkPad-L470-W10DG:~/Desktop/shell_script_Assignment$ cat sig.conf
Auction ; MID ; INGESTOR ; ETL ; vdopia-etl=3
Bid ; HIGH; JOINER; ETL ; vdopia-bid= 2
sigmoid@sigmoid-ThinkPad-L470-W10DG:~/Desktop/shell_script_Assignment$ 
```

Github :

https://github.com/VikasKR123/Shell_Script