Create IAM Policies for following scenarios

- Allow a user to view S3 buckets and read/write access to a specific bucket and objects of all types within the bucket.
    - Bucket name: xyz-media

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowUserToSeeBucketListInTheConsole",
            "Effect": "Allow",
            "Action": ["s3:ListAllMyBuckets"],
            "Resource": "*"
        },
        {
            "Sid": "AllowUserToListObjectsInTheBucket",
            "Effect": "Allow",
            "Action": ["s3:ListBucket"],
            "Resource": "arn:aws:s3:::xyz-media"
        },
        {
            "Sid": "AllowUserToReadWriteObjectsInTheBucket",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject",
      "s3:DeleteObject"
            ],
            "Resource": "arn:aws:s3:::xyz-media/*"
        }
    ]
}
```

- A user requires administrative access to all resources and read only access to IAM. However users should be able to perform following actions on their own IAM user
    - changing password
    - add/modifying MFA
    - add/remove access keys.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AdminAccessToAllResources",
            "Effect": "Allow",
            "Action": "*",
            "Resource": "*"
        },
        {
            "Sid": "ReadOnlyAccessToIAM",
            "Effect": "Allow",
            "Action": [
                "iam:Get*",
                "iam:List*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "SelfManageOwnIAMUser",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:UpdateAccessKey",
                "iam:ListAccessKeys",
                "iam:CreateVirtualMFADevice",
                "iam:DeleteVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:ResyncMFADevice",
                "iam:DeactivateMFADevice",
                "iam:ListMFADevices"
            ],
            "Resource": [
                "arn:aws:iam::*:user/${aws:username}"
            ]
        }
    ]
}
```

- You have configured RDS password rotation via secret manager for two users. Each secret has the username configured as a prefix. Craft a policy that would only allow users to read their own secrets. However this policy should not block the users ability to read any other secrets in the secret manager.
  - Secret naming convention - -rds-credentials
  - Usernames –
    - mark
    - harry

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowReadOwnSecrets",
            "Effect": "Allow",
            "Action": "secretsmanager:GetSecretValue",
            "Resource": [
                "arn:aws:secretsmanager:*:*:secret:mark-rds-credentials*",
                "arn:aws:secretsmanager:*:*:secret:harry-rds-credentials*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:username": [
                        "mark",
                        "harry"
                    ]
                }
            }
        },
        {
            "Sid": "AllowReadAnyOtherSecrets",
            "Effect": "Allow",
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "*"
        }
    ]
}
```

You have been tasked to create a bash script to accomplish the following. Provide a script file and provide steps on how you would implement it on an EC2 instance (Ubuntu 22.04) which would execute this script at instance startup or restart.

● Check if Java is installed, if not install the latest OpenJDK version.

● Check if security updates are configured as unattended updates and if not enable them.

● Install the following package if not already installed -> https://www.elastic.co/guide/en/fleet/current/install-standalone-elastic-agent.html

```
#!/bin/bash

# Update package lists
sudo apt-get update -y

# Check if Java is installed, if not install the latest OpenJDK version
if ! java -version &>/dev/null; then
    echo "Java is not installed. Installing OpenJDK..."
    sudo apt-get install -y default-jdk
else
    echo "Java is already installed."
fi

# Check if unattended-upgrades is installed and configure it if not
if ! dpkg -l | grep -qw unattended-upgrades; then
    echo "Unattended-upgrades is not installed. Installing..."
    sudo apt-get install -y unattended-upgrades
fi

# Ensure unattended-upgrades is enabled
echo "Enabling unattended-upgrades..."
sudo dpkg-reconfigure -plow unattended-upgrades

# Check if Elastic Agent is installed, if not install it
if ! dpkg -l | grep -qw elastic-agent; then
    echo "Elastic Agent is not installed. Installing..."
    curl -L -O https://artifacts.elastic.co/downloads/beats/elastic-agent/elastic-agent-8.6.1-linux-x86_64.tar.gz
    tar xzvf elastic-agent-8.6.1-linux-x86_64.tar.gz
    sudo ./elastic-agent-8.6.1-linux-x86_64/elastic-agent install
else
    echo "Elastic Agent is already installed."
fi
```

Create the Script File:

Create a new script file and open it with a text editor.

```
sudo nano /home/ubuntu/setup.sh
```

Copy and paste the script content into the file.Save the file and exit the text editor.

Make the Script Executable:Change the permissions to make the script executable.

```
sudo chmod +x /home/ubuntu/setup.sh
```

Create a Systemd Service to Run the Script at Startup:Create a new service file for the script.

```
sudo nano /etc/systemd/system/setup.service
```

Add the following content to the service file:

```
 [Unit]
Description=Run setup script at startup

[Service]
ExecStart=/home/ubuntu/setup.sh
Restart=always
User=ubuntu

[Install]
WantedBy=multi-user.target
```

Save the file and exit the text editor.

Enable and Start the Service:Reload systemd to recognize the new service.

```
sudo systemctl daemon-reload
```

Enable the service to run at startup.

```
sudo systemctl enable setup.service
```

Start the service immediately.

```
sudo systemctl start setup.service
```

Verify the Service:Check the status of the service to ensure it is running correctly.

```
sudo systemctl status setup.service
```