

2303A51563

batch =10

import hashlib

```
def hash_email(email):
    # Hashing email to protect user identity
    return hashlib.sha256(email.encode()).hexdigest()

name = input("Enter your name: ")
age = int(input("Enter your age: "))
email = input("Enter your email: ")

secure_email = hash_email(email)

# Store only anonymized data
user_data = {
    "name": name,
    "age": age,
    "email_hash": secure_email
}

print("User data stored securely.")
```

The screenshot shows the OnlineGDB interface. The left sidebar displays user information and navigation links like 'Create New Project', 'My Projects', 'Classroom', 'Learn Programming', 'Programming Questions', 'Upgrade', and 'Logout'. The main workspace is titled 'main.py' and contains the provided Python code. The code defines a function to hash an email, reads user input for name, age, and email, generates a secure email hash, stores the data in a dictionary, and prints a confirmation message. The code is run under 'python 3'. The output window at the bottom shows the program's execution and completion message: "...Program finished with exit code 0 Press ENTER to exit console."

```
import hashlib
def hash_email(email):
    # Hashing email to protect user identity
    return hashlib.sha256(email.encode()).hexdigest()
name = input("Enter your name: ")
age = int(input("Enter your age: "))
email = input("Enter your email: ")
secure_email = hash_email(email)
user_data = {
    "name": name,
    "age": age,
    "email_hash": secure_email
}
print("User data stored securely.")
```

```
def sentiment_analysis(text):
```

```

# Simple keyword-based sentiment analysis
positive_words = ["good", "happy", "excellent"]
negative_words = ["bad", "sad", "terrible"]

text = text.lower()

# Bias mitigation: neutral language, no offensive terms
score = 0
for word in positive_words:
    if word in text:
        score += 1
for word in negative_words:
    if word in text:
        score -= 1

if score > 0:
    return "Positive"
elif score < 0:
    return "Negative"
else:
    return "Neutral"

```

```
print(sentiment_analysis("The product is good"))
```

The screenshot shows the OnlineGDB interface with a Python 3 environment. The code in the editor is:

```

1- def sentiment_analysis(text):
2-     # Simple keyword-based sentiment analysis
3-     positive_words = ["good", "happy", "excellent"]
4-     negative_words = ["bad", "sad", "terrible"]
5-
6-     text = text.lower()
7-
8-     # Bias mitigation: neutral language, no offensive terms
9-     score = 0
10-    for word in positive_words:
11-        if word in text:
12-            score += 1
13-    for word in negative_words:
14-        if word in text:
15-            score -= 1
16-
17-    if score > 0:
18-        return "Positive"
19-    elif score < 0:
20-        return "Negative"
21-    else:
22-        return "Neutral"
23-
24- print(sentiment_analysis("The product is good"))

```

The terminal output shows:

```
Positive
...Program finished with exit code 0
Press ENTER to exit console.
```

### Task 3

```
def recommend_product(history):
    products = ["Laptop", "Mobile", "Tablet"]

    # Transparency: Show why recommendation is made
    print("Recommendations based on your browsing history")

    if "tech" in history:
        return products
    else:
        return ["General Product"]

user_history = ["tech", "gadgets"]
print(recommend_product(user_history))
```

The screenshot shows the OnlineGDB interface. On the left, there's a sidebar with user information (Welcome, 2303A51563), project options (Create New Project, My Projects, Classroom, Learn Programming, Programming Questions, Upgrade), and a Logout button. The main area has tabs for Run, Debug, Stop, Share, Save, and Beautify, with Language set to Python 3. The code editor contains the provided Python script. Below the editor is a terminal window showing the output of the program: "Positive" and "...Program finished with exit code 0 Press ENTER to exit console.".

### Task 3

```
import logging
```

```
logging.basicConfig(filename="app.log", level=logging.INFO)
```

```
def login(username):
    # Ethical logging: Do NOT log passwords or emails
    logging.info(f"User {username} logged in.")
    print("Login successful")
```

```
login("harshavardhan)
```

The screenshot shows the OnlineGDB interface. The left sidebar has a user profile with the name '2303A51563'. The main area shows a Python script named 'main.py' with the following code:

```
import logging
logging.basicConfig(filename="app.log", level=logging.INFO)
def login(username):
    # Ethical logging: Do NOT log passwords or emails
    logging.info(f"User {username} logged in.")
    print("Login successful")
login("Akshaya")
```

The bottom right window shows the console output:

```
Positive
...Program finished with exit code 0
Press ENTER to exit console.
```

## Tak 5

### Responsible AI Model Usage:

- Model accuracy depends on data quality
- Not for critical medical/legal decisions
- Bias may exist if training data is biased

\*\*\*\*

```
from sklearn.linear_model import LinearRegression
```

```
X = [[1], [2], [3], [4]]
y = [2, 4, 6, 8]
```

```
model = LinearRegression()
model.fit(X, y)
```

```
print("Prediction:", model.predict([[5]]))
```

OnlineGDB  
online compiler and debugger for c/c++

Welcome, 2303A51563 ▲

Create New Project  
My Projects  
Classroom new  
Learn Programming  
Programming Questions  
Upgrade  
Logout ▾

```
1 """
2 - Responsible AI Model Usage:
3 - Model accuracy depends on data quality
4 - Not for critical medical/legal decisions
5 - Bias may exist if training data is biased
6 """
7
8 from sklearn.linear_model import LinearRegression
9
10 X = [[1], [2], [3], [4]]
11 y = [2, 4, 6, 8]
12
13 model = LinearRegression()
14 model.fit(X, y)
15
16 print("Prediction:", model.predict([[5]]))
17 |
```

Positive

...Program finished with exit code 0  
Press ENTER to exit console.█

About • FAQ • Blog • Terms of Use •  
Contact Us • GDB Tutorial • Credits •  
Privacy  
© 2016 - 2026 GDB Online

The screenshot shows the OnlineGDB interface. On the left, there's a sidebar with user information, project links, and navigation buttons like 'Logout'. The main area has a toolbar at the top with 'Run', 'Debug', 'Stop', 'Save', and 'Beautify' buttons. The language is set to 'Python 3'. Below the toolbar is a file list with 'main.py' selected, and a status bar showing 'python' and task numbers 3, 4, 5, and 6. The code editor contains a Python script for a linear regression model. The output window below shows the script running, with the word 'Positive' entered as input and the program outputting '...Program finished with exit code 0' followed by a prompt to press 'ENTER'.