19Z701 - CRYPTOGRAPHY
Conceptual Design

21Z204 - Akil K
21Z221 - Javagar M
21Z246 - S Bharath
21Z250 - Santhosh A
21Z268 - Venkatprasadh Mari

ABAC Model Design

- **Access Policy Definitions**: Describe how user, resource, and environmental attributes are evaluated to determine access.
- **Attributes**: Detail how user attributes (role, department), resource attributes (data type, sensitivity), and environmental attributes (time, location) are used to create dynamic policies.
- **Policy Management**: Explain how policies are managed, created, and updated in response to changing requirements.

Threshold Cryptography Design

- **Cryptographic Schemes**: Use Shamir's Secret Sharing or similar threshold cryptography techniques to distribute decryption keys among multiple trusted parties.
- **Threshold Value**: Define the threshold number of parties required to decrypt the data.
- **Operation Flow**: Describe how the system verifies cryptographic shares and reconstructs the decryption key when the threshold is met.

Smart Contract Implementation

- **Policy Enforcement**: Detail how smart contracts enforce ABAC policies and verify cryptographic conditions.
- **Logging Mechanism**: Describe how smart contracts record access decisions and cryptographic operations on the blockchain.
- **Emergency Override**: Explain the emergency override mechanism and how it is implemented through smart contracts.

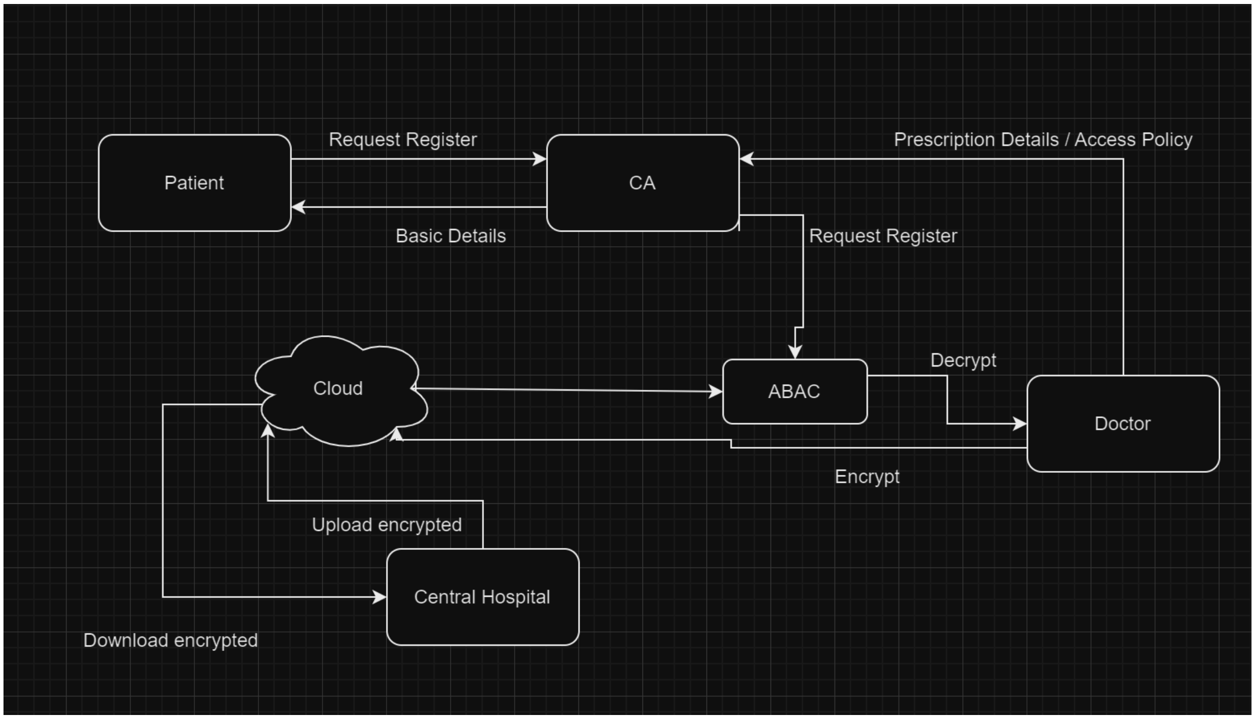
Data Security and Privacy

- **Data Encryption**: Explain how sensitive data is encrypted and decrypted securely using cryptographic techniques.
- **Access Control Integrity**: Describe how blockchain and threshold cryptography ensure that no single party can unilaterally access patient data.
- **Auditing and Transparency**: Discuss how blockchain's immutable ledger provides transparency and ensures secure access audits.

Evaluation and Performance Metrics

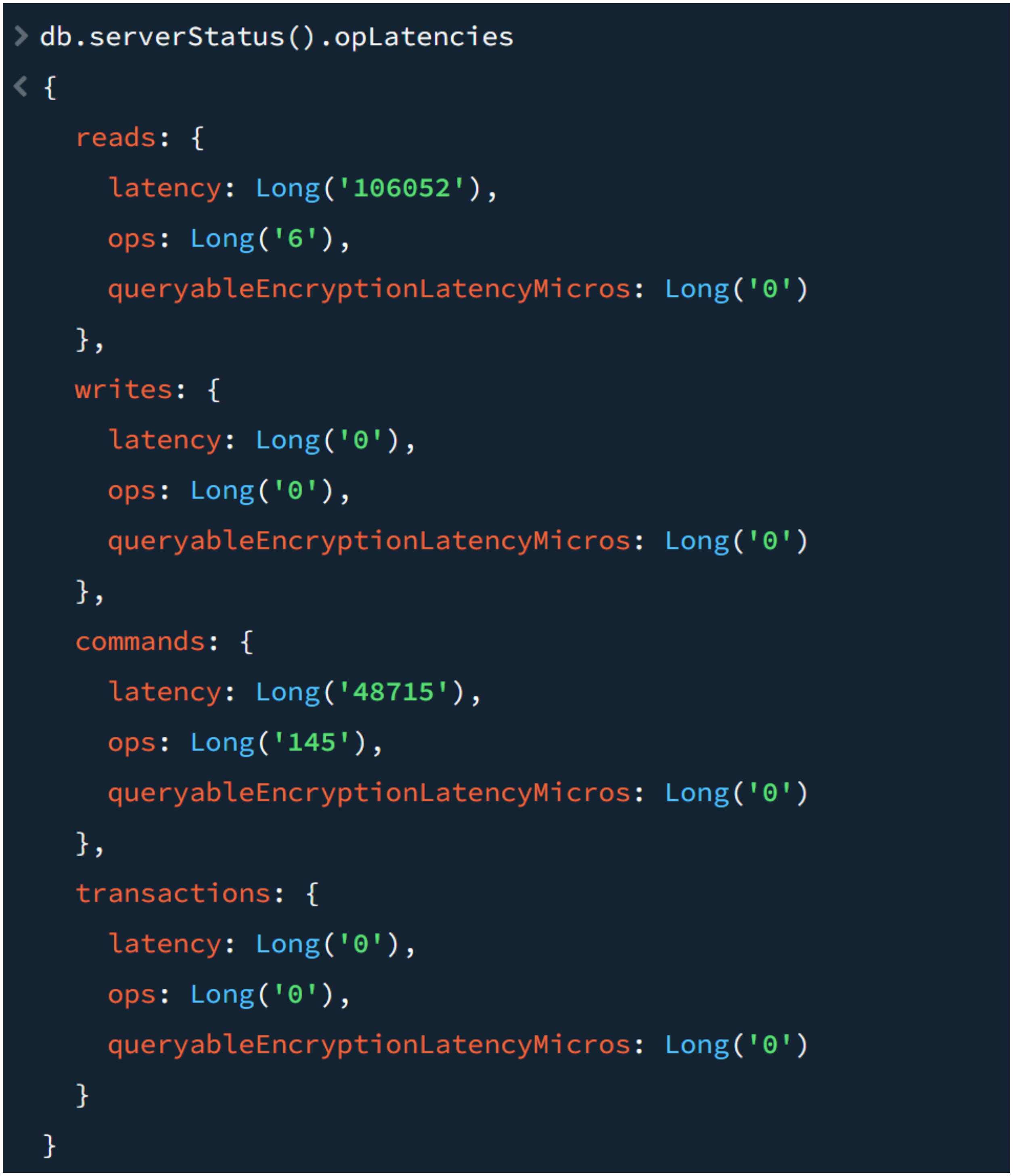
- **Security Evaluation**: Analyze the resistance of the system to unauthorized access, hacking attempts, and cryptographic attacks.
- **Performance Metrics**:
  - **Transaction Latency**: Time taken for blockchain transactions and policy enforcement.
  - **Scalability**: Evaluate the performance under a growing number of users, transactions, and data.
  - **Cryptographic Overhead**: Measure the computational cost of encryption and decryption using threshold cryptography.
- **Usability Testing**: Conduct usability testing to ensure healthcare professionals and administrators can use the system effectively.

Architectural Diagram:



Assessment of Metrics:

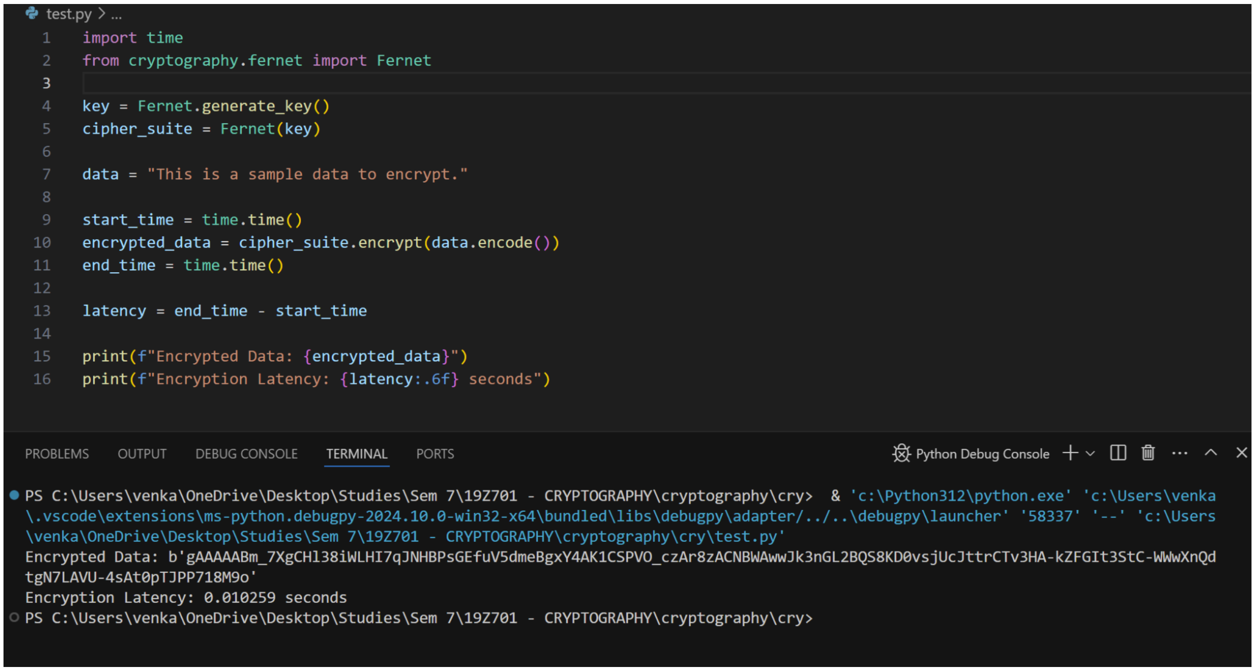
MongoDB Latency Assessment



```
> db.serverStatus().opLatencies
< {
    reads: {
      latency: Long('106052'),
      ops: Long('6'),
      queryableEncryptionLatencyMicros: Long('0')
    },
    writes: {
      latency: Long('0'),
      ops: Long('0'),
      queryableEncryptionLatencyMicros: Long('0')
    },
    commands: {
      latency: Long('48715'),
      ops: Long('145'),
      queryableEncryptionLatencyMicros: Long('0')
    },
    transactions: {
      latency: Long('0'),
      ops: Long('0'),
      queryableEncryptionLatencyMicros: Long('0')
    }
  }
```

Hashing Performance Assessment:

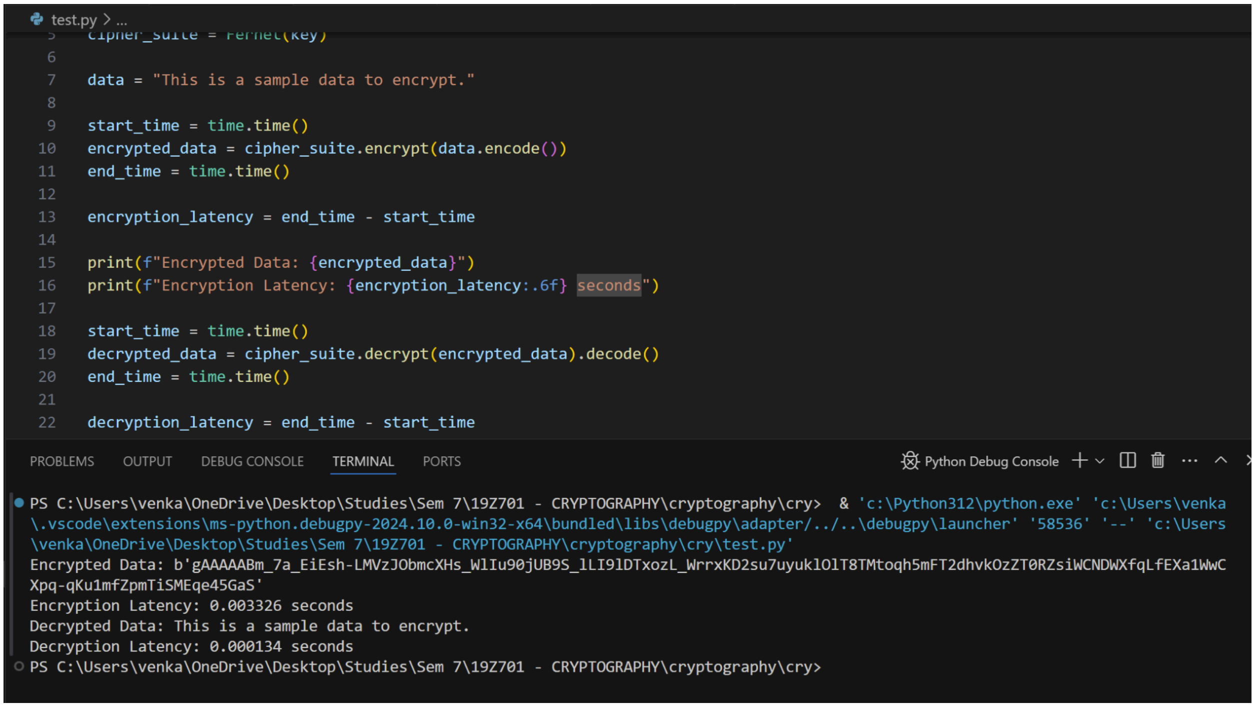
Encryption:



```python
import time
from cryptography.fernet import Fernet

key = Fernet.generate_key()
cipher_suite = Fernet(key)

data = "This is a sample data to encrypt."

start_time = time.time()
encrypted_data = cipher_suite.encrypt(data.encode())
end_time = time.time()

latency = end_time - start_time

print(f"Encrypted Data: {encrypted_data}")
print(f"Encryption Latency: {latency:.6f} seconds")
```

```
PS C:\Users\venka\OneDrive\Desktop\Studies\Sem 7\19Z701 - CRYPTOGRAPHY\cryptography\cry>  & 'c:\Python312\python.exe' 'c:\Users\venka
\.vscode\extensions\ms-python.debugpy-2024.10.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '58337' '--' 'c:\Users
\venka\OneDrive\Desktop\Studies\Sem 7\19Z701 - CRYPTOGRAPHY\cryptography\cry\test.py'
Encrypted Data: b'gAAAAABm_7XgCHl38iWLHI7qJNHBPsGEfuV5dmeBgxY4AK1CSPVO_czAr8zACNBWAwwJk3nGL2BQS8KD0vsjUcJttrCTv3HA-kZFGIt3StC-WWwXnQd
tgN7LAVU-4sAt0pTJPP718M9o'
Encryption Latency: 0.010259 seconds
PS C:\Users\venka\OneDrive\Desktop\Studies\Sem 7\19Z701 - CRYPTOGRAPHY\cryptography\cry>
```

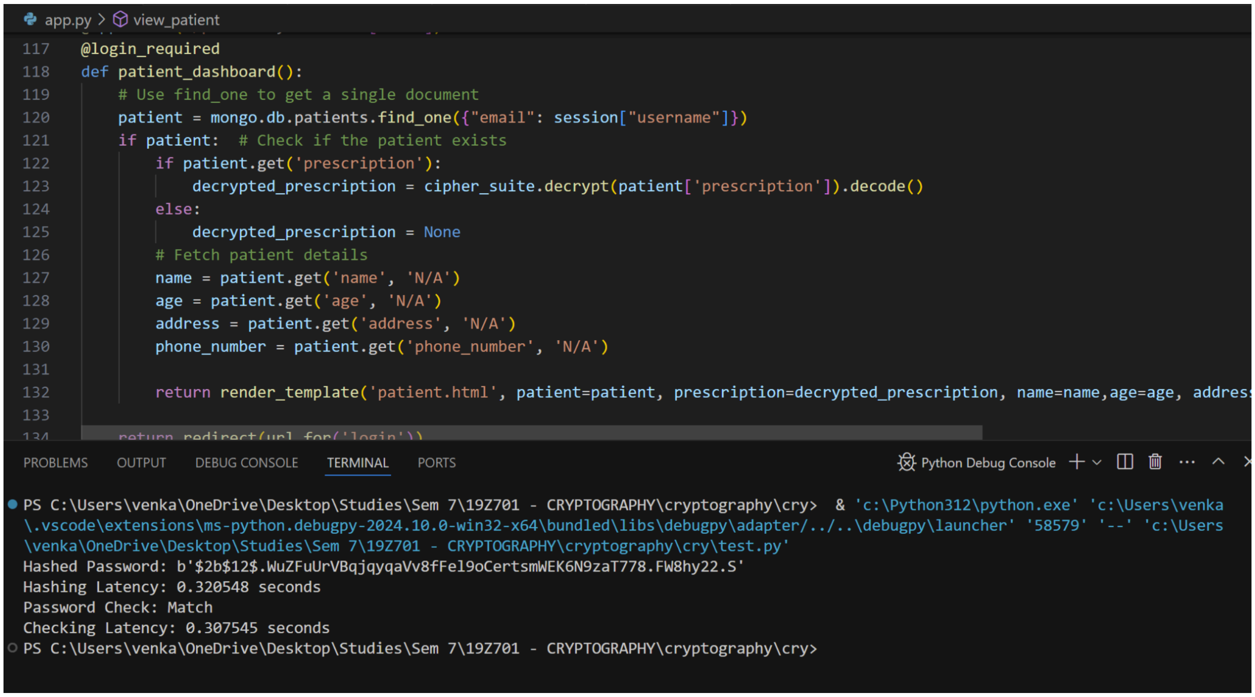
Decryption:



```python
cipher_suite = Fernet(key)

data = "This is a sample data to encrypt."

start_time = time.time()
encrypted_data = cipher_suite.encrypt(data.encode())
end_time = time.time()

encryption_latency = end_time - start_time

print(f"Encrypted Data: {encrypted_data}")
print(f"Encryption Latency: {encryption_latency:.6f} seconds")

start_time = time.time()
decrypted_data = cipher_suite.decrypt(encrypted_data).decode()
end_time = time.time()

decryption_latency = end_time - start_time
```

```
PS C:\Users\venka\OneDrive\Desktop\Studies\Sem 7\19Z701 - CRYPTOGRAPHY\cryptography\cry>  & 'c:\Python312\python.exe' 'c:\Users\venka
\.vscode\extensions\ms-python.debugpy-2024.10.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '58536' '--' 'c:\Users
\venka\OneDrive\Desktop\Studies\Sem 7\19Z701 - CRYPTOGRAPHY\cryptography\cry\test.py'
Encrypted Data: b'gAAAAABm_7a_EiEsh-LMVzJObmcXHs_WlIu90jUB9S_lLI9lDTxozL_WrrxKD2su7uyuklOlT8TMtoqh5mFT2dhvkOzZT0RZsiWCNDWXfqLfEXa1WwC
Xpq-qKu1mfZpmTiSMEqe45GaS'
Encryption Latency: 0.003326 seconds
Decrypted Data: This is a sample data to encrypt.
Decryption Latency: 0.000134 seconds
PS C:\Users\venka\OneDrive\Desktop\Studies\Sem 7\19Z701 - CRYPTOGRAPHY\cryptography\cry>
```

Password Checking:



```python
@login_required
def patient_dashboard():
    # Use find_one to get a single document
    patient = mongo.db.patients.find_one({"email": session["username"]})
    if patient:  # Check if the patient exists
        if patient.get('prescription'):
            decrypted_prescription = cipher_suite.decrypt(patient['prescription']).decode()
        else:
            decrypted_prescription = None
        # Fetch patient details
        name = patient.get('name', 'N/A')
        age = patient.get('age', 'N/A')
        address = patient.get('address', 'N/A')
        phone_number = patient.get('phone_number', 'N/A')

        return render_template('patient.html', patient=patient, prescription=decrypted_prescription, name=name,age=age, addres

    return redirect(url_for('login'))
```

```
PS C:\Users\venka\OneDrive\Desktop\Studies\Sem 7\19Z701 - CRYPTOGRAPHY\cryptography\cry>  & 'c:\Python312\python.exe' 'c:\Users\venka
\.vscode\extensions\ms-python.debugpy-2024.10.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '58579' '--' 'c:\Users
\venka\OneDrive\Desktop\Studies\Sem 7\19Z701 - CRYPTOGRAPHY\cryptography\cry\test.py'
Hashed Password: b'$2b$12$.WuZFuUrVBqjqyqaVv8fFel9oCertsmWEK6N9zaT778.FW8hy22.S'
Hashing Latency: 0.320548 seconds
Password Check: Match
Checking Latency: 0.307545 seconds
PS C:\Users\venka\OneDrive\Desktop\Studies\Sem 7\19Z701 - CRYPTOGRAPHY\cryptography\cry>
```