EXPERIMENT 3

AES algorithm for 128 bit key

3.1 Aim

To write a program for implementing AES algorithm for 128 bit key.

3.2 Algorithm

1. START
2. Create a Node.js project to work .
3. Create the AesEncryption.js file.
4. Import the internal crypto module of Node.js .
5. Use the desired algorithm for encryption (aes-128-cbc) and assign to a variable named *algorithm.*
6. Use crypto.randomBytes() method to generate cryptographically built random data.
7. Generate 16 bytes of random data using the above function and assign to variable *initializationVector.*
8. Declare a constant variable *message* as input message.
9. Generate 16 bytes of random data using method mentioned in Step 6 and assign to a variable *Securitykey.*
10. Create a *cipher* object using the *crypto.createCipheriv* function of the crypto module .
11. Pass the first argument as the *algorithm* which is being used, the second argument as the *Securitykey* , and *initializationVector* as the third argument.
12. To encrypt the message, use the update() method on the cipher.
13. Pass the first argument as the *message* , the second argument as *utf-8* (input encoding), and *hex* (output encoding) as the third argument.
14. Call cipher.final() to stop encryption. Once the cipher.final() method has been called, the Cipher object can no longer be used to encrypt data.
15. Log the encrypted message in the console.
16. Create a *decipher* object using the *crypto.createDecipheriv* function of the crypto module.
17. To Decrypt the message, use the update() method on the decipher object.
18. Call cipher.final() to stop decryption. Once the decipher.final() method has been called, the decipher object can no longer be used to decrypt data.
19. Log the decrypted message in the console.
20. STOP

3.3 Program

```
// import crypto module
const crypto = require("crypto");

// define the algorithm to use
const algorithm = "aes-128-cbc";

// generate 16 bytes of random data for initializationVector
const initializationVector = crypto.randomBytes(16);

// Define the message to encrypt
const message = "This is a secret message";

console.log("Original message: " , message);

// generate 32 bytes of random data as security key
const Securitykey = crypto.randomBytes(16);

// the cipher function
const cipher = crypto.createCipheriv(algorithm, Securitykey,
initializationVector);

// encrypt the message

// input encoding  ==> utf-8
// output encoding ==> hex
let encryptedData = cipher.update(message, "utf-8", "hex");

encryptedData += cipher.final("hex");

console.log("Encrypted message: " , encryptedData);

// the decipher function
const decipher = crypto.createDecipheriv(algorithm, Securitykey,
initializationVector);

let decryptedData = decipher.update(encryptedData, "hex", "utf-8");

decryptedData += decipher.final("utf8");

console.log("Decrypted message: " , decryptedData);
```

3.4 Output

```
PS C:\Users\cinoy\OneDrive\Documents\aes encryption> node AesEncryption.js
Original message:  This is a secret message
Encrypted message:  4146c139cf4774440d3d80123df70f66ea3614875a8ea48db345e2fcc92c9581
Decrypted message:  This is a secret message
PS C:\Users\cinoy\OneDrive\Documents\aes encryption>
```

3.5 Result

The AES algorithm was implemented successfully.