**Project: PaymentGateway**

This has been created in .NetCore. There are 3 projects in the solution:

1. PaymentGateway
2. BankService
3. WebClient
4. **Payment Gateway**

This is the Payment Gateway Api . It does the following functions:

Receives payment requests

Validates payment requests

Processes payments using Bank Service

Stores details of payments processed

Can retrieve the payments made through the api

Sends sensitive information like card number , cvv after encryption

1. **Bank Service**

This is a mock up of the acquiring bank to which the api sends payment requests and receives back a success/ failure response

1. **WebClient**

This is a client which calls the various methods on the api. This is a console application and will allow user to choose options to get previous payment details from api or to make a payment by providing all payment information.

**Features**

1. Dependency injection has been used for PaymentService
2. Salt and hash has been used to encrypt the card details like card number and cvv. The details when passed to the merchant will be in encrypted format. The merchant can use the same salt and hash with their records for reconciliation.
3. Returning api response with a ResultCode and ResultText which gives information to the caller about the success of the call as well.
4. Validations to make sure all required information for making a payment is passed and is valid. If a payment has been processed already ( request with same uid is sent twice), we do not make double payment.
5. All data is passed in json format.

**Areas Of Improvement**

1. Can add a database connection to store the payment details . Currently it is being managed in memory.
2. The features mentioned in extra mile bonus points in project spec can be added

**Methods on API and example responses**

It has below methods

**GET: api/payment**

To fetch all payments made

Example response:

{

"resultCode": 1,

"resultText": "2 payments retrieved",

"paymentDetails": [

{

"uid": "bbb",

"cardNumber": "mkv6xwu4VMmvR6cSmXKSCjmwI9ukL1WfMuxVWbkNJie9HAYatisUwMQ=",

"cardExpiry": "2021-12",

"paymentDate": "2019-06-12T23:24:24.4352726Z",

"cardCvv": "L+aeGW3pKz9/l7CYxpcHYxUJpOdARfRVr5iGOz1w0P2fsKMqoQ==",

"amount": 8000,

"currency": "GBP",

"success": true

},

{

"uid": "aaa",

"cardNumber": "dC/ijbs6ZGcwRpBqdJ9ywZodPRImnh+SMqeRWCukW/B5dho=",

"cardExpiry": "2019-6",

"paymentDate": "2019-06-12T23:25:05.8947143Z",

"cardCvv": "6RcCr+SkkhSCxiPfi3sJWxzef9b1pxW9ZIJbg9qsKBCAoWVRj7Lt7JCGV1I=",

"amount": 8000,

"currency": "GBP",

"success": true

}

]

}

**GET: api/payment/uid**

To fetch payments made with a specific uid

Example response:

{

"resultCode": 1,

"resultText": "Found payment details",

"paymentDetails": [

{

"uid": "bbb",

"cardNumber": "pQzOHvy/SR9jSMWtj+/uWusBFpeBMo8EpBL8v1v4mNJoHzLtwC7DSwE=",

"cardExpiryDate": "2021-12-31T00:00:00",

"cardExpiry": "2021-12",

"paymentDate": "2019-06-12T23:16:06.6587381Z",

"cardCvv": "skN5+3WI68MmS3lJYU+2K0/pibTU4rTF8UFUKzde6+EGUCj73eP3tUxb1XlOdRk=",

"amount": 8000,

"currency": "GBP",

"success": true

}

]

}

**POST: api/payment**

To make payments by passing in details like Uid, CardNumber, CardExpiry(yyyy-mm format, CardCvv, Amount, Currency(3 letter format) :

Example Input:

{

Uid: "aaa",

CardNumber: "1111222233334444",

CardExpiry: "2019-6",

CardCvv: "555",

Amount: 8000,

Currency: "GBP",

}

Example Response:

{

"resultCode": 1,

"resultText": "Processed Payment successfully",

"paymentDetails": []

}