Payment Gateway.API Developer Notes

**Assumptions**

1. Only merchants access the application so, secret\_key is directly added in the merchant table.
2. The Bank Account information for merchants has been considered out of scope.
3. The request send to the MockAcquiringBankService is totally an assumption.

**Improvements/ ToDo**

1. Add Application Logging (preferably database logging)
2. Add Performance tests.
3. Add Unit tests.
4. In a few places the objects are being created from one type to another manually. Add more AutoMapper profiles to create them.
5. Enrich the StartUp.cs to automatically switch the dependency injection of the IAcquirerService from MockAcquiringBankService to AcquiringBankService based on the environment (dev or production)
6. Store secret\_keys in separate table with a user\_id which can either reference to the merchant\_id or any other user\_id.

**Authentication**

* API Key based authentication has been used.
* **ApiKeyAuthenticatonMiddleWare.cs** is responsible to extracting the key from the Authorization header and validating it.
* **IRequestInfo** : The Authentication middleware sets the merchant\_id for the secret key and a newly generated transaction\_reference guid here. This is further used for authentication is the correct merchant is making the request.
* The transaction\_reference was added so that it can be used for distributed tracing/logging and is also sent to the acquiring bank so that in a ideal scenario when we have event based communication it can be sent back and used as an identifier

**Encryption**

* Microsoft.AspnetCore.DataProtection is used for encrypting decrypting sensitive information.
* AES-256-CBC Encryption Algortithm is used by default which is PCI DSS Compliant
* The secret\_keys, Card number and CVV are encrypted in the database.
* The encryption decryption logic has been kept in the repository layer

**Data storage**

* SQLLite has been used for datastorage
* PaymentGateway.db file in the PaymentGateway.API project is the db file

**API client**

* Swagger has been used as an API Client
* Controller/Action level comments have been added to provide an overview
* The Authorize button can be used to provide the key in the Authorization header. Sample key already in db : “key\_uCLUxBlfJBkN9fmmrvWPVlG3s1OcRvpCasTBMQ9vjhE=”

