Project Title: Unified Payments Interface (Give your own title)

Project Description:

Your goal is to build a payment gateway interface (like the Indian government's UPI) where **third party developers** can integrate your payment system into their application. Payment gateway performs the important role in processing and authorizing the payment or transactions between customer and merchants.

Note: You are supposed to ***build*** your own payment gateway, not use one. Instructions and other information follow:

Basic Features of the App:

1. **Merchant Login:** Merchant should be able to sign up using their email and Google+ OAuth. The merchant should also be able to login using the same ID. Forgot password, change password and change email address options must be provided.

2. **Dashboard:** A dashboard must be provided that must contain the list of transactions and payments or orders made by clients. They should be categorized into ***successful***, ***failed*** and ***pending*** status. A graph indicating sales, number of transactions etc., would be nice.

3. **Roles:** Admin role for yourself, a role for the merchant and other relevant roles must be provided.

Note: Other trivial features that build a meaningful web app have to be assumed and implemented.

Key Features of the App:

1. **Client Information Collection:** The client fills out an order form on the merchant’s website and enters payment card information. The merchant receives the credit card information and passes the information on, along with the order amount to your payment gateway.

2. **Authentication:** Every transaction should generate a unique ID. Your payment gateway then takes this information, encrypts the information to be sent between the browser and the merchant's bank account. The payment gateway may allow transaction data to be sent directly from the customer's browser to the gateway, bypassing the merchant's systems. This should happen without the customer being redirected to another website.

3. **Authorization**: The merchant forwards the transaction details to your payment gateway (assume that the credit card issuing bank receives the authorization request, verifies the credit or debit available and then sends a response back). Your payment gateway should send an authorization request to the acquiring bank (assuming that it exists) which redirects back to the customer.

4. **Transaction:** The merchant then accepts the fulfillment from your payment gateway and then completes the transaction. A transaction is like a verification of the order or process of payment or transaction of money. It should be a token that confirms the payment.

Exceptions and Errors to be handled:

1. What should happen when there is an internet connection failure at either end of the transaction?
2. What are the factors that lead to an infinite loop where a transaction never completes?
3. What happens when a customer enters wrong card/payment information?

4. What happens when there is no response from your payment gateway or from the bank? When will the transaction end in this case?

Note: More points to handling out more exceptions and to making sure that the app performs well in all conditions.

Evaluation Criteria:

- **Code Quality:** Every line of code has to be indented properly using best practices. Add comments wherever necessary. You will be judged on overall code quality, code structuring and modularity. Organized, scalable code architecture will be appreciated. Any extra effort made to look the code more readable will get you extra points.

- **Database Design:** The app's core database architecture and the way attributes, keys are mapped are judged. The database is expected to be designed with scalability in mind.

- **User Interface:** The app should be made keeping in mind it will be used by a layman and not a developer. It should be as intuitive as possible. The app should be as easy to understand as possible. You should think about innovative ways of delivering messages (like errors, failures) to users.

- **Originality:** The code will be checked for plagiarism, it is considered as a minus point if the code is not original, copied/borrowed from a fellow candidate or the internet.

Note: If in doubt, use your best judgement. Also add assumptions, if any, as comments in the code, wherever necessary.

Submission Guidelines

- **Source Code:** The root repository of the project has to be kept track in **git** from the beginning. The project should be uploaded to **GitHub** and a possible link has to be shared of the repository.

- **Documentation:** Include a README in the repository with steps to install/run the application and basic description of design/architecture and approach. More information on how to write a good documentation can be found here: https://github.com/dctacademy/docs-format

- **Deployment:** Extra points for deploying the application on the web. Using Heroku is recommended. If needed, we will provide you help regarding this. A link should be submitted of the deployed application. The app should be in working condition. Please do not submit incomplete or broken app.