User Guide

The endpoints available to client applications for DTU Pay (Custemr client, merchant and manager) are also described in the OpenAPI specifications attached. Here is a description of the major use cases:

Customer

- Register
  - The use case for registering a user, this is invoked by making a post request containing a Customer object to the facade microservice, the facade then routes this request to the endpoint within the accountmanager microservice which handles the registration of the new customer. The id of the newly registered customer is returned.
- Update
  - The use case for updating an existing user, this is invoked by making a put request to the facade micro service, the put request should contain the id of the customer to be updated, and a customer object containing the updated customer data. This is then routed to the accountmanager microservice, which then updates the customer and returns either true of false, depeding on the result of the update.
- Delete
  - The use case for deleting an existing user, this is invoked by making a delete request to the facade microservice containing customer id of the customer to be deleted, the request is then routed to the accountmanager microservice, which handles the deletion of the customer and returns either true or false, depending on the result of the deletion.

Merchant

- Register
  - The use case for registering a merchant, this is invoked by making a post request containing a Merchant object to the facade microservice, the facade then routes this request to the endpoint within the accountmanager microservice which handles the registration of the new merchant. The id of the newly registered merchant is returned.
- Pay

Reporting

- Creation of customer report regarding all transactions
- Creation of merchant report regarding all transactions

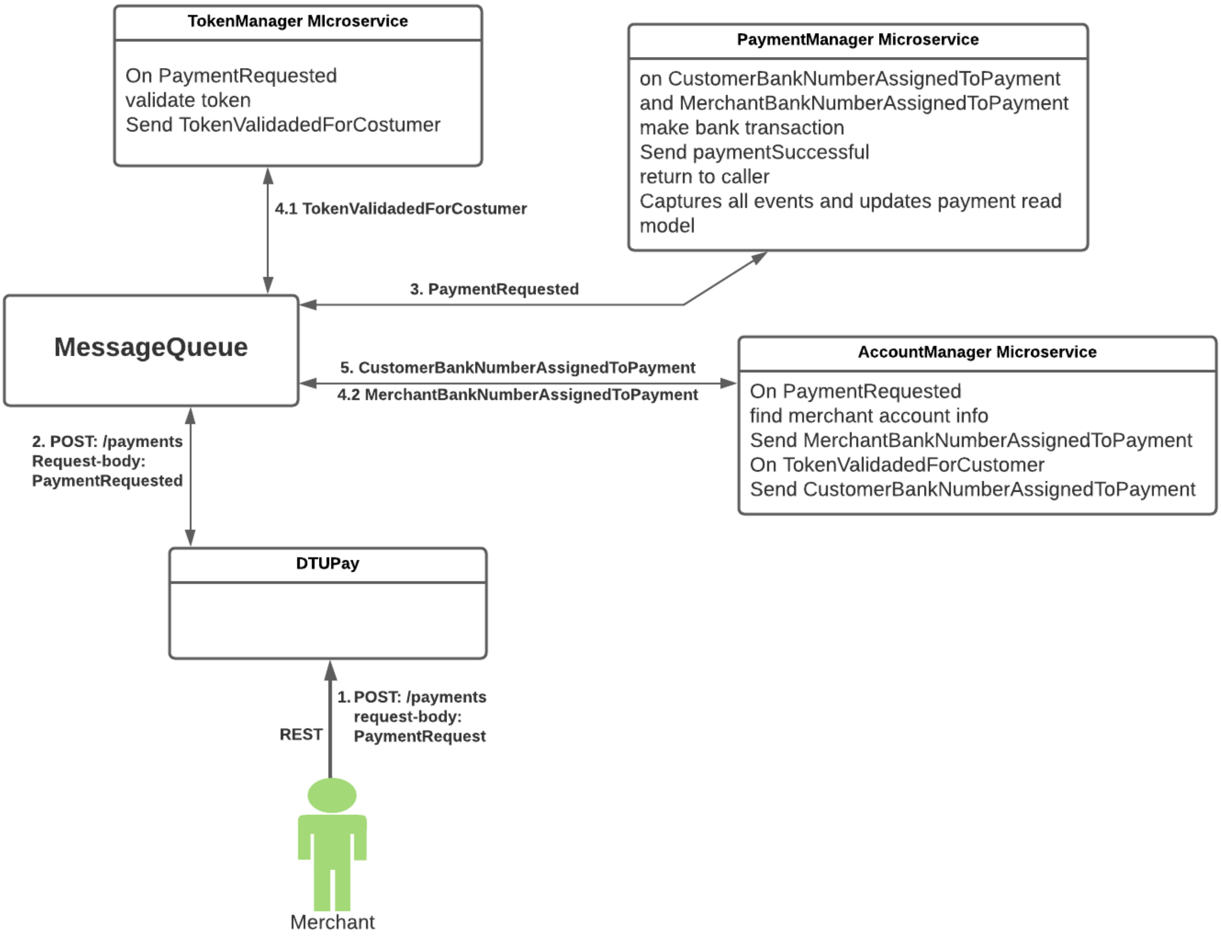
Errors

The errors can also be seen in the OpenAPI specifications.

- To be handled by Customer client.
  - When getting tokens, a customer must have no more than 1 token, and can have a maximum of 6 tokens in total - HTTP 403 (see OpenAPI specification)
- To be handled by Merchant client.
  - Customer must have at least 1 unused valid token in order to perform the payment successfully - HTTP 400
  - Customer must have sufficient funds in order to perform the payment successfully - HTTP 400
  - The token that the customer uses must have been registered with and received from DTU pay - HTTP 401
  - Both the creditor and the debtor have to have valid bank accounts - HTTP 400

Messages

When a Merchant requests a payment (REST) to DTUPay MS (Step 1), DTUPay MS forwards the PaymentRequest (REST) to PaymentManager MS (step 2).
The message ”PaymentRequested”is sent from PaymentManager MS, to MessageQueue, which TokenManager MS and AccountManager MS catches (step 3).
TokenManager MS then sends the message ”TokenValidatedForCustomer”to MessageQueue which AccountManager MS (step 4(1)) catches. At the same time AccountManager MS sends the message ”MerchantBankNumberAssignedToPayment”to MessageQueue that is caught by PaymentManager MS (step 4(2)).
After receiving ”TokenValidatedForCustomer”AccountManager MS sends ”CustomerBankNumberAssignedToPayment”to MessageQueue and caught by PaymentManager MS (step 5), and the payment is done.


TokenManager MIcroservice
On PaymentRequested
validate token
Send TokenValidadedForCostumer
4.1 TokenValidadedForCostumer
PaymentManager Microservice
on CustomerBankNumberAssignedToPayment
and MerchantBankNumberAssignedToPayment
make bank transaction
Send paymentSuccessful
return to caller
Captures all events and updates payment read model
3. PaymentRequested
MessageQueue
5. CustomerBankNumberAssignedToPayment
4.2 MerchantBankNumberAssignedToPayment
AccountManager Microservice
On PaymentRequested
find merchant account info
Send MerchantBankNumberAssignedToPayment
On TokenValidadedForCustomer
Send CustomerBankNumberAssignedToPayment
2. POST: /payments
Request-body:
PaymentRequested
DTUPay
1. POST: /payments
request-body:
PaymentRequest
REST
Merchant

Figur 1: Message Queue