Event Storming Start

Successful Payment

Hubert Baumeister

January 2023

Here is a small document on how to start with event storming for DTU Pay. You should be starting with the successful payment scenario. Later, you can add other scenarios to your event storming diagram.

Event storming is usually done on a large paper using PostIts. However, a good alternative is the use of Miro[1] or Mural[2].

1 First step

In the first step, you should collect and discuss all important events relevant for the successful payment scenario. It is a good idea to work backwards, e.g., start with the outcome, e.g., `MoneyTransferred` and then move back to the beginning. In the diagram below this is `PaymentRequested`. Note that name of events is the past tense. Events document facts that have happened. In contrast, commands are actions intended to happen. Commands will act with a system (external or internal) and the result of commands will be events (facts that have happened).

In the following picture, I have provided you with a starting point to do event storming for the successful payment scenario. You should add more events as needed.

2 Second step

In a second step you want to order your events from left to right. An event on the left occurs before an event on the right and even may cause the event on the right.

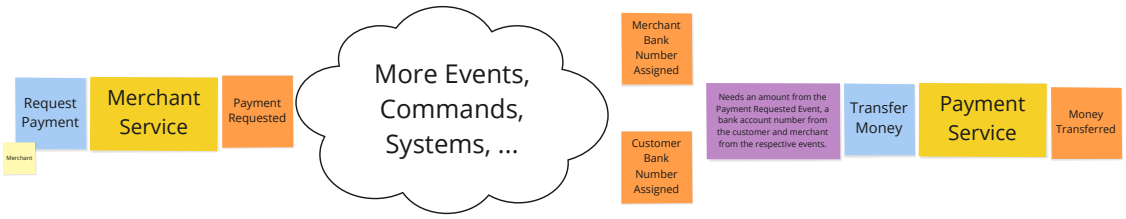
[1] `http://miro.com`
[2] `http://mural.co/`

3 Third step

In a third step, you are going to add policies, commands, and systems.

- Commands, in contrast to events, commands describe actions on a system that will cause events as a result. For example, a command could be `create customer` and the resulting event could be `customer created`.
- Policies, are rules that govern, e.g., when events cause commands to happen
- Systems are the aggregates, bounded contexts, or Microservices that execute commands and produce events. For example, the command `create customer` could be sent to the `customer account service`.

Here is a small example on how to use systems and commands for the successful payment scenario. Again, you have to fill out the rest.



4 Fourth step

You should have now identified your systems, commands, and events. Systems can map to Microservices, commands to functions offered by Microservices. Events can be used together with asyncrhonous communication to choreograph the Microservices so that the successful payment behaviour can be implemented. Have a look at the learning modules "Asynchronous Communication" and "Testing Microservices" on DTU Learn to see how to do this.

You should now write the test and implement the successful payment scenario with those microservices and functions that you have discovered.

5 Fifth step

After you have implemented and refactored your Microservices so that the successful payment scenario works, you can use and extend the event storming diagram to discuss failure scenarios and other scenarios.