Distributed System

In our shop we’ve got the business rule “If an order that costs more than 100 euro gets marked as sent, the customer will receive a voucher worth 5 euro.“. We would like to hand out exactly one voucher if the business rule applies or none if not. This is very important for our business. However, the business states that the voucher doesn’t have to be created immediately.

Current state

When our shop was developed, a single application with a single database was all we needed to start our business quickly. Applying the business rule to the order and voucher module is as straight forward as starting a transaction, mark the order as sent and saving it. If the business rule applies, we create a voucher and save it as well. Then we commit the transaction. If there is an error in between, we would rollback everything. The whole business transaction succeeds or fails together. This works because the individual modules share a logical database and everything runs in one process. This means that the code can be invoked directly from one another.

Desired state

As our shop grows and more and more developers are hired, a single application with a single database will no longer scale. One reasons for this is because different teams have different release cycles and therefore want to manage their own deployment pipelines. The desired state defines that the order module becomes an independent application with its own physical database. The same applies to the voucher module. We therefore no longer share a database connection with which we can use a transaction. This brings some more challenges as we are now building a distributed system. How the applications communicate in order to consistently apply our business rule is an open question that is still being answered by our architect.

Get the job done

Surprise, you're playing the role of our architect. Your job is to build a system which behaves as described as in the desired state. You don’t have to build a whole shop. In this task you should put your focus on applying the business rule. You’ve free choice of infrastructure and third party dependencies to get the job done. Whether you write desktop applications, cli applications or web applications is up to you.

Do you have other technical solutions in mind? If so, what are the advantages and disadvantages of your solution compared to the other solutions?

Hint: You can design the system first before writing code.
Hint: Line by line, consider what will happen if one of your applications crashes and what this means for our business rule.