QR Payments

Show and scan QR. Use as merchant or consumer.

Introduction

Show and scan QR. Register as merchant and user. Enable card acceptance immediately.

The QR Payments solution was created to enable both merchants and individuals to generate and scan QR codes to perform transactions. Verestro supports the global QR standard and has its own QR standard which is an extension of the global standard. This functionality allows the user to generate his own QR code. Having generated QR code, users can order the execution of a transaction by sharing the code to the potential payer. Functionality also allows users to make a transfer by scanning the QR code shared by the potential recipient of the transfer. The scanned QR must support the global QR standard or the Verestro standard.

QR code payments can be used by issuers, fintech providers, merchants, individual consumers for various use cases starting from Person-2-Person transactions, through SME merchant payments to sophisticated use cases for digital, TV or advertisement payments.

How to connect with us?

QR Payments is a solution developed on Android and iOS. Integration of the solution is available through the mobile QR SDK offered by Verestro. Integration through Verestro SDK, requires an account at Verestro Artifactory. To create such account, please contact the Customer Service.

Mobile SDK

Verestro provides Software Development Kit (SDK) which can be used for QR generating, QR scanning, encoding data into the QR code and decode data from the QR code. Verestro team actively supports Customer with integration. More information about each module in Mobile SDK can be found in iOS QR SDK Documentation for iOS implementation and Android QR SDK Documentation for Android implementation.

Overview

This section provides general information about the solution, terminology description and a high-level description of the business and technical aspects.

Abberations and Acronyms

Abbreviations and Acronyms used in the document:

Abbreviations	**Description**
ACQ	Acquiring Institution/Acquirer
AP	Admin Panel
ACS	Access Control Server
C2C	Card to card
DC	Data Core API
P2P	Peer to peer API
MDC	Mobile Data Core API
SDK	Software Development Kit
THC	Transaction History Core
OS	Operative System
URI	Uniform Resource Identifier
Mid	Merchant Identifier

Terminology

This section explains a number of key terms and concepts used in this document:

Name	Description
Customer	Institution which is using Verestro products. This institution decides which SDK should be used and how transaction should be processed. Basicly Customer can be called Verestro client.
User	User which is using Money Transfer Hub Application. It is root of entity tree. User is identified in Wallet Server by some unique identifier which is provided after registration. User can have access to his data and operations based on session. User’s session is created after device pairing is performed. When session expires then user authentication have to be performed. Session is valid 10 minutes, however it is configurable parameter.
Card	Card belongs to the user. User can have many cards. Card is identified via internal id given after storing card on Wallet Server. Whole PAN is stored on Wallet Server which has PCI DSS certificate.
Device	Device belongs to user. When user starts using application after installation then device pairing is performed. After pairing device with some unique id, unique device installation id is generated and this installation is assigned to user. It is possible to have one active installation on specific device for specific user.
Session Token	Token which defines User. It is an authorization way of the User. This entity is created after paring device and this is needed to perform any actions in the application. When session is expired then user authentication needs to be performed. Session is valid 10 minute s, however it is configurable parameter.
Sender	Verestro Wallet user which triggers transaction to the Receiver (check User description).
Receiver	Receiver can be identified in Wallet Server (Internal) or may be an entity that does not exist in Wallet Server (External) ◦ Internal – this type of Receiver has his own unique identifier just like sender. It can also act as a Sender in the transaction process, ◦ External – this type of Receiver does not exist in Wallet Server. Transfers that are made to this type of Receiver require the entering of his card data by Sender.
Mid	Merchant identifier. This entity is representing Merchant in Acquirer’s system. Customer have to provide the mid information to enable mid configuration in the Verestro system. Required to process 3DS authentication via Verestro System.

Acquirer	External institution responsible for processing transaction and 3ds requests ordered by the Verestro Money Transfer Hub Application. Acquirer connects with banks / card issuers and returns information whether the ordered action on a given card is possible.
PAN	It is 7-15 digits of credit card number. These digits contain the Permanent Account Number (PAN) assigned by the bank to uniquely identify the account holder.
Wallet Server	Provides the backend services to support Mobile Payment Application via Verestro Wallet SDK and is responsible for managing users, devices, cards , device tokens, storing transactions history and communication with Acquirers.
PCI DSS	PCI DSS (Payment Card Industry Data Security Standard) is a security standard used in environments where the data of payment cardholders is processed. The standard covers meticulous data processing control and protection of users against violations.
QR	QR code is a type of barcode or scannable pattern that contains various forms of data like website links, account information, phone numbers, or even entire object of the transaction.

QR Transactions

QR Transaction is a technology which enables to perform transaction by QR code. This functionality allows to initialize payment by showing QR by Merchant/Receiver to the Sender, which scans it, and pays using Verestro application. The QR code is generated in accordance with the global standard, however, depending on the customer's needs, it can be extended as long as the standard is maintained. This solution requires the 3DS Authentication method, which is described later in the document and it is an individual requirement depending on the Settlement Agent.

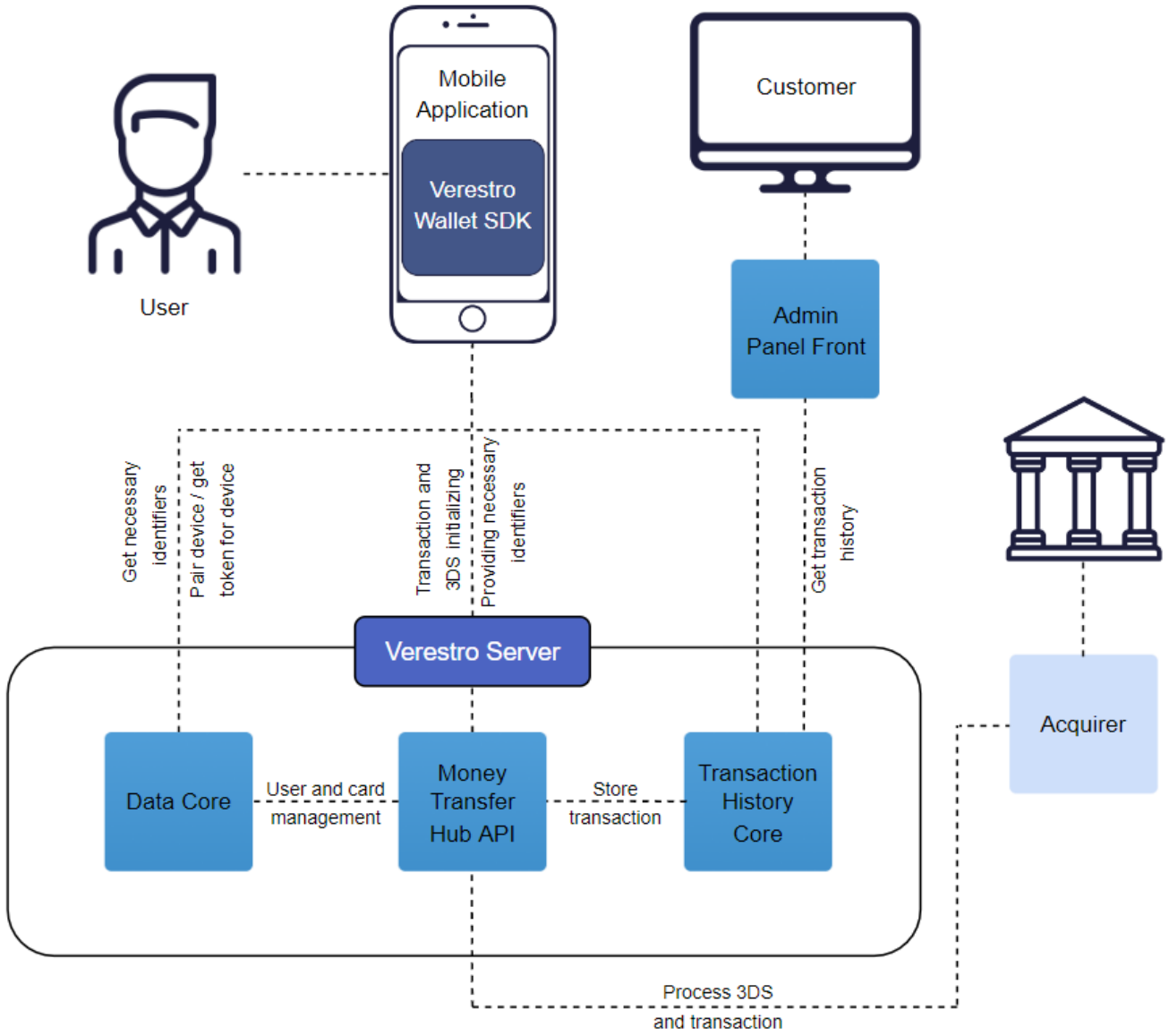
The MC Send 2.0 solution also provides the ability to perform transactions using a QR code - Mastercard QR. However the customer must be integrated with Verestro Money Transfer and Mastercard Send 2.0 to be able to use the MCQR solution.

The MCQR solution offers some features such as:

- crossboarded transfers,
- the possibility of issuing a QR for the customer’s clients,
- compliance with the global QR standard.

QR Transactions high level overview

This diagram shows high level components which are involved in whole solution:



QR Transaction Key Components

Component	Description
Verestro Wallet Server	Backend services of Money Transfer solution. In the described product, they are responsible handling data provided from QR code. On the basis of such data, the backend enables the execution of transactions and 3ds authentication by connecting to various Acquirers.
Verestro QR Wallet SDK	Provides all functionalities needed for Money Transfer Hub Solution with QR transfers. It is responsible for generate QR, parsing data embedded in it and deliver necessary data to pass all Verestro Wallet Server functionalities.
Notification Service	Delivers all necessary information about transaction statuses and other actions which was performed between individual Verestro backend components and/or external.

Admin Panel	Frontend component that allows Customer to check transaction statuses and transaction history of his clients.

Verestro QR Money Transfer Hub

Verestro Payment QR Hub is a solution that was created to provide possibility of QR generating and scanning via application. Verestro QR Money Transfer Hub provides functionalities for making transfers based on the data contained in QR codes.

Solution consists of:

- Server components:
  - Wallet Server – backend component,
  - Wallet Admin Panel – frontend component,
- Mobile components:
  - Wallet SDK – Android / iOS libs.

Wallet Types

Money Transfer Hub Solution supports one type of wallet which can be used in the implementation:

- OPEN - user registers itself in the application and provides data like PAN etc.

Note that the User can provide generated QR to external entites.

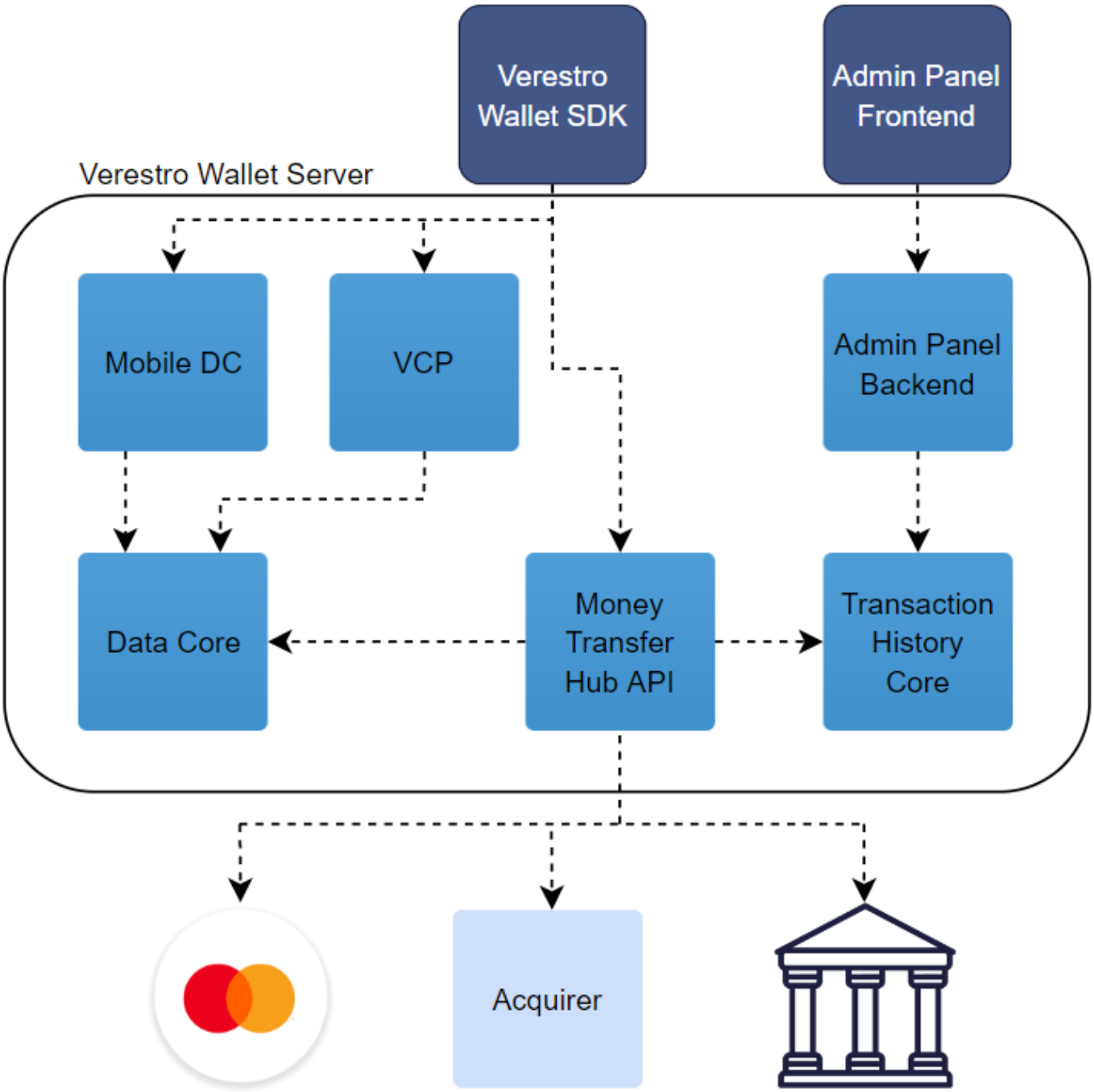
Implementation models

Verestro provides QR in Software-and-a-Service model hosting the solution for customer. Verestro provides Wallet SDK and Wallet Server. Customer is responsible for integration of provided SDK with his own application and user authentication management (based on MDC SDK).

Verestro can also provide QR payments inside its White Label Application. In this model full implementation process is on Verestro side and project can be launch quickly.

Architecture

This diagram shows big picture of Verestro Money Transfer Hub architecture:



Server Components

Server components are backend services which are designed to process requests from the mobile part, provide the necessary information such as user ID and communicate with Acquirers.

Deployment Models

In QR Money Transfer Hub Solution Server components are deployed and configured on Verestro side. Verestro is responsible for maintaining infrastructure, deploying applications and monitoring.

Wallet Server

Wallet Server is the backend component which consists of few internal services which are responsible for managing users, cards, security tokens, QR payments and transaction history. This component is also responsible for connection with Acquirers. Services included in the Wallet Server. *For more detailed information about Wallet Server component please see Money Transfer documentation.*

Wallet Server operates with domain objects like:

- User (Sender) - User which is using QR Money Transfer Hub,
- Session Token – Token which defines User. It is an authorization way of the User,
- Device – This entity is created after user registration and is required to login the User,
- Card - User Card which can be charged or recharged,
- Receiver - Verestro Wallet user or external entity which receives funds from the Sender.

Mobile Components

Mobile components are dedicated to handle the QR solution on the Android and iOS.

Wallet SDK

QR SDK is responsible for creating the appropriate QR code, parse it and for transferring the data contained in this code. Based on such data, a transaction will be initiated.

Below is a detailed list of SDKs included in Mobile Components:

- P2P Transfers SDK - supports the process of generating and reporting transactions. The share of this module in the application takes its payment functions to a higher level, enabling the initiation of transfers to a card, telephone number or QR code *(for more technical information please check "P2P Transfer SDK documentation")*.
- QR SDK - The QR module was designed to work with the applicable MPQR (Merchant Presented QR) standard developed by EMV. Thanks to the integration of this module with P2P and meeting the requirements of Mastercard, the user will be able to pay with sellers using QR codes. An additional functionality is that the user can use the code generated for his card and thus receive funds from other people within one implementation *(for more technical information please check "QR SDK documentation")*.

Access

The account at Verestro Artifactory is required to get access to Verestro repository.

Versioning

SDK version contains three numbers. For example: 1.0.0.:
(For more information check what is "Semantic Versioning" standrand)

- First version digit tracks compatibility-breaking changes in SDK public APIs. It is mandatory to update application code, to use SDK, when this is incremented.
- Second version digit tracks new, not compatibility-breaking changes in public API of SDK. It is optional to update application code, when this digit is incremented.
- Third version digit tracks internal changes in SDK. No updates in application code are necessary to update to version, which has this number incremented.

Changes not breaking compatibility:

- Adding new optional interface to SDK setup,
- Adding new method to any domain,
- Adding new enum value to input or output,
- Adding new field in input or output model.

Communication with Wallet Server

Wallet SDK at the very beginning performs authentication of application and device to Wallet Server.

Security

MDC SDK is responsible for most of the security issues. However, in the Money Transfer Hub solution, sensitive data such as PAN or CVC are processed. They are taken as an array of characters. This data is not held, but immediately wiped from RAM.

Security Checks and Data Clearing

There are performed security checks on Wallet SDK side. Security checks consists of:

- root access detection,
- hooking protection,
- debugging protection,
- custom ROM protection,
- data tampering protection.

Requirements

Wallet SDK has some mandatory requirements to make it work:

- device cannot be rooted,
- Android OS should be in version 6.0 or above,
- iOS OS should be in version 13.0 or above,
- devices cannot have enabled debugging,
- MDC SDK integration.

Configuration

The entire solution requires configuration data necessary for the product to operate in line with the Customer's expectations. *For more detailed information about customer account configuration requirements please see Money Transfer documentation.*

Use cases

This section describes a detailed description of the processes provided in the QR Payments solution and the appearance of the application from the end user point of view.

Wallet Server Money Transfer API

This section describes use cases which can be initiated using Wallet Server P2P API. This API should be used by Customers through integrated Wallet QR SDK to manage Transactions and Transaction’s Senders/Receivers, Commission calculation and determine Currencies on Wallet Server. Every method below is secured by session token. *For more detailed information about non-QR functionalities offered by Money Transfer please see Money Transfer documentation.*

Transaction process

If the user and his card are present in the Verestro system, such a user can perform QR transactions using the Verestro QR Payments Hub solution. This type of transaction allows the user to make a transfer by scanning the QR code and to track the transfer by providing his own code.

QR Payment - scan code

This diagram shows QR code transaction processes in the case that the user scans the "external" QR code:

@startuml
skinparam ParticipantPadding 30
skinparam BoxPadding 30
skinparam noteFontColor #FFFFFF
skinparam noteBackgroundColor #1C1E3F
skinparam noteBorderColor #1C1E3F
skinparam noteBorderThickness 1
skinparam sequence {
ArrowColor #1C1E3F
ArrowFontColor #1C1E3F

```
ActorBorderColor #1C1E3F
ActorBackgroundColor #FFFFFF
ActorFontStyle bold
ParticipantBorderColor #1C1E3F
ParticipantBackgroundColor #1C1E3F
ParticipantFontColor #FFFFFF
ParticipantFontStyle bold
LifeLineBackgroundColor #1C1E3F
LifeLineBorderColor #1C1E3F
}
participant "User" as user
participant "Mobile App" as mob
participant "Verestro Mobile SDK" as sdk
participant "Verestro Money Transfer API" as p2p
participant "Verestro THC API" as thc
participant "Acquirer" as acq
participant "Issuer" as acs
user->mob: 1. User scans QR code
mob->sdk: 2. Provides scanned QR code
sdk->sdk: 3. Parses obtained QR - extract data
note right of sdk: Wallet SDK uses dedicated library for QR handling - IMV Standard
mob<-sdk: 4. Returns extracted data from scanned QR code
user<-mob: 5. Shows all transaction data extracted from QR code - optional
note right of user: If QR code is static, user provides amount
user->mob: 6. Accepts transaction
note right of user: 3ds authentication may be required at this point
note right of user: 3ds authentication flow described on above diagram
mob->sdk: 7. Performs transaction
sdk->p2p: 8. Provides transaction data and execute transaction
p2p->acq: 9. Perform transaction
acq->acs: 10. Perform Funding from provided card if possible
acq<-acs: 11. Success
p2p<-acq: 12. Success + transaction id
p2p->thc: 13. Store transaction with status Funding
p2p<-thc: 14. Transaction has been stored successfully
sdk<-p2p: 15. Transaction success
mob<-sdk: 16. Transaction success
user<-mob: 17. Your transaction has been sent
@enduml
```

After scanning the received QR code the Mobile SDK decomposes the data contained in the QR code. The user sees the transaction data, such as the amount that will be sent after confirming the transfer or the currency in which the transfer will be made etc. After confirming the transfer of the funds, the SDK forwards the request to the Wallet API. Wallet API in turn forwards the request to the Acquirer. Based on the data received, the Acquirer contacts the bank to complete the transaction. If the bank agrees, the funding process is carried out - collecting funds from the

Sender's account. After the funding is completed, the funds are transferred to the appropriate Receiver.
To facilitate understanding of the process flow, each of the steps is described below in the correct order (points highlighted in blue are performed outside the Verestro Server):

1. User scans QR code using device (possible only if provided the device supports such functionality).
2. Verestro Mobile SDK gets data from the scanned QR code.
3. User confirms the transfer and authenticates himself via 3ds authentication process.
4. After going through the 3ds process, the funds transfer process begins.
5. The application orders the transfer of funds by communicating with the Verestro backend via the Verestro Mobile SDK.
6. Money Transfer API communicates with Data Core to check whether the card details belong to the user and the receiver.
   - After receiving a positive response from Data Core, Money Transfer API performs money send transaction.
7. Acquirer receives a request to make a transfer of funds.
8. The Acquirer communicates with the Issuer regarding the execution of the transfer.
   - The Sender's account is debited.
   - The Receiver’s account is credited.
9. The Acquirer receives information from the Issuer that the operation has been performed.
10. Acquirer informs Money Transfer API about the positive status of the ordered operation.
11. Money Transfer API saves transaction details in the Transaction History Core API.
12. Money Transfer API informs Mobile SDK about the positive status of the ordered operation.
13. Mobile SDK informs Mobile Application about the positive status of the ordered operation.
14. The Mobile Application displays to the user a successful transfer of funds.

QR Payment - generate code

This diagram shows QR code transaction processes in the case that the user generates QR code:

@startuml
skinparam ParticipantPadding 30
skinparam BoxPadding 30
skinparam noteFontColor #FFFFFF
skinparam noteBackgroundColor #1C1E3F
skinparam noteBorderColor #1C1E3F
skinparam noteBorderThickness 1
skinparam sequence {
ArrowColor #1C1E3F
ArrowFontColor #1C1E3F
ActorBorderColor #1C1E3F
ActorBackgroundColor #FFFFFF
ActorFontStyle bold
ParticipantBorderColor #1C1E3F

```
ParticipantBackgroundColor #1C1E3F
ParticipantFontColor #FFFFFF
ParticipantFontStyle bold
LifeLineBackgroundColor #1C1E3F
LifeLineBorderColor #1C1E3F
}
participant "User" as user
participant "Payer" as payer
participant "Mobile App" as mob
participant "Verestro Mobile SDK" as sdk
participant "Verestro Money Transfer API" as p2p
participant "Verestro THC API" as thc
participant "Acquirer" as acq
participant "Issuer" as acs
user->mob: 1. User opens QR code generating
note right of user: User generates QR
mob->sdk: 2. Provides data fulfilled by the user
sdk->sdk: 3. Coverts obrained data into QR code
note right of sdk: Wallet SDK uses dedicated library for QR handling - IMV Standard
mob<-sdk: 4. Returns QR with parsed data
user<-mob: 5. Shows all transaction data and generated QR code
note right of user: User shows generated QR to the potential payer
note right of payer: Payer scans QR code
payer->payer: 6. Scans QR code
mob->sdk: 7. Get transaction data from QR code
mob<-sdk: 8. Returns transaction data obtained from QR code
payer<-mob: 9. Shows all transaction data
payer->mob: 10. Confirms money transfer and pass 3ds
note right of payer: 3ds authentication may be required at this point
note right of payer: 3ds authentication flow described on above diagram
mob->sdk: 11. Performs transaction
sdk->p2p: 12. Provides transaction data and execute transaction
p2p->acq: 13. Perform transaction
acq->acs: 14. Perform Funding on Payer's card if possible
acq<-acs: 15. Success
p2p<-acq: 16. Success + transaction id
p2p->thc: 17. Store transaction with status Funding
p2p<-thc: 18. Transaction has been stored successfully
sdk<-p2p: 19. Transaction success
mob<-sdk: 20. Transaction success
payer<-mob: 21. Your transaction has been sent
acq->acs: 22. Perform Credit on User's card if possible
acq<-acs: 23. Success
p2p<-acq: 24. Success + transaction id
p2p->thc: 25. Update transaction status to CLEARED
@enduml
```

3DS Authentication

The QR Money Transfer Hub Solution supports the 3DS 2.0 process and it is required when user is initiating a transaction. This is an authentication method based on the alleged cardholder data check, biometric authentication and improved customer experience.

As mentioned in the "Configuration" paragraph, a specific 3DS integration is required depending on which ACQ Verestro will connect to when making a transaction.

- If the integration with a given ACQ has already been made, Verestro only need to configure the appropriate merchant identifier, which should be provided by the Customer.

If the integration with the required billing agent has not yet been completed, it will be one of the activities for which Verestro will be responsible. Note that integration with 3DS increases the scope of required development.
In card to card transfer, the authentication process is required. It is to confirm that the user is definitely the owner of the card.

This diagram shows high level 3ds authentication processes:

@startuml
skinparam ParticipantPadding 30
skinparam BoxPadding 30
skinparam noteFontColor #FFFFFF
skinparam noteBackgroundColor #1C1E3F
skinparam noteBorderColor #1C1E3F
skinparam noteBorderThickness 1
skinparam sequence {
ArrowColor #1C1E3F
ArrowFontColor #1C1E3F
ActorBorderColor #1C1E3F
ActorBackgroundColor #FFFFFF
ActorFontStyle bold
ParticipantBorderColor #1C1E3F
ParticipantBackgroundColor #1C1E3F
ParticipantFontColor #FFFFFF
ParticipantFontStyle bold
LifeLineBackgroundColor #1C1E3F
LifeLineBorderColor #1C1E3F
}
participant "User" as user
participant "Mobile App" as mob
participant "Verestro Mobile SDK" as sdk
participant "Verestro Money Transfer API" as p2p
participant "Acquirer" as acq

participant "Mastercard/VISA" as mcvisa
participant "ACS/Issuer" as acs
note left of mob: User has confirmed currencies and accepted shown currency rate
user->mob: 1. Enters CVC code
note left of mob: In this step the application begins 3DS authentication process.
note left of mob: If the bank decides that 3ds is not required, not any action will be performed.
note left of mob: Diagram shows the option requiring the user to authenticate.
mob->sdk: 2. Initialize 3DS process
sdk->p2p: 3. Initialize 3DS process
p2p->acq: 4. Initialize 3DS process
note left of acs: Bank returns decision whether it is necessary for the user to authenticate
acq->mcvisa: 5. Is 3DS authentication required?
acq<-mcvisa: 6. ThreeDS Method
p2p<-acq: 7. ThreeDS Method
p2p->acq: 8. Continue 3DS process
acq->acs: 9. Continue 3DS process
acq<-acs: 10. Challenge required
p2p<-acq: 11. Challenge required
sdk<-p2p: 12. Challenge required
mob<-sdk: 13. Challenge required
user<-mob: 14. Informs user that challenge is required
note left of mob: Bank's page content is provided
user->user: 15. Performs challenge
note right of user: After a successful challenge, the Bank sends PaRes/cRes, which is intercepted by the Wallet SDK and provided to the Money Transfer API
user-->mob
mob-->sdk
sdk->p2p: 16. Provides intercepted PaRes/cRes
p2p->acq: 17. Provides obtained PaRes/cRes
acq->acs: 18. Check authentication for provided PaRes/cRes
acq<-acs: 19. Authentication success
p2p<-acq: 20. Authentication success
sdk<-p2p: 21. Authentication success
mob<-sdk: 22. Authentication success
user<-mob: 23. Informs about the successful completion of the authentication process
@enduml

To facilitate understanding of the process flow, each of the steps is described below in the correct order *(points highlighted in blue are performed outside the Verestro Server)*:

1. Mobile application contacts the Verestro Server via the Verestro Mobile SDK to start 3ds process.
2. Mobile SDK provides all necessary data to the Money Transfer API (including user/card id and 3ds authentication request id) while calling the 3ds initialization method.

3. Having all the necessary information, Money Transfer API orders Acquirer to start the 3ds authentication process.
4. Acquirer transfers the card and user details to ACS.
5. If the user is the owner of the card, the ACS returns a positive answer and a decision whether the continuation of the authentication process is required (according to the diagram above, the scenario with the necessity to continue the process is described).
6. ACS informs the Acquirer that the 3ds process continue is required.
7. Acquirer informs Money Transfer API about ACS decision.
8. Money Transfer API requests Acquirer to continue the authentication process (the authentication id is provided.
9. Acquirer provide the request to continue the process to the ACS.
10. ACS informs about the necessity to perform the Challenge process and returns necessary parameters such as:
    - challengeHtmlFormBase64 - this field is a BASE64 encrypted html source file containing the ACS' challenge 3DSecure frame.
    - cReq - data for building a form such as challengeHtmlFormBase64.
11. Acquirer informs Verestro Money Transfer API that ACS requires Challengeand provides above parameters.
12. Verestro Money Transfer API forwards the obtained information to Verestro Mobile SDK.
13. Verestro Mobile SDK decodes the received challengeHtmlFormBase64 parameter and transmits the received frame of the mobile application.
14. The user is redirected to the bank's website where he performs the Challenge process.
15. After a successful Challenge process, the bank sends the cRes / PaRes parameter. This response is intercepted by the Verestro Mobile SDK and forwarded to the Verestro Money Transfer API.
16. Verestro Money Transfer API provides the received cRes / PaRes to Acquirer.
17. Acquirer provides the above parameters to ACS for verification.
18. If everything was done correctly, the ACS informs Acquirer about the successful completion of the 3ds authentication. Among other things, the following parameters are included in the response.
    - cavv - property determined by the ACS. The value may be used to provide proof of authentication.
    - eci - property is determined by the ACS. This property contains the two digit Electronic Commerce Indicator (ECI) value, which is to be submitted in a credit card authorization message.
19. Acquirer provides information about successful completion of the 3ds authentication among with the above parameters obtained from the ACS to Verestro Money Transfer API.
20. Verestro Money Transfer API provides information about successful completion of the 3ds authentication among with the above parameters obtained from the ACS to Verestro Mobile SDK.
21. Verestro Mobile SDK provides positive response to mobile application. The information is shown to the user.

User experience & screenshots

This chapter introduces the main actions and processes in the application associated with QR module from the end
user point of view.

> *The rest of the processes apply to "Money Transfer" as a product as a whole are described in separate Money Transfer documentation. QR payments is one of the components of the Money Transfer product.*

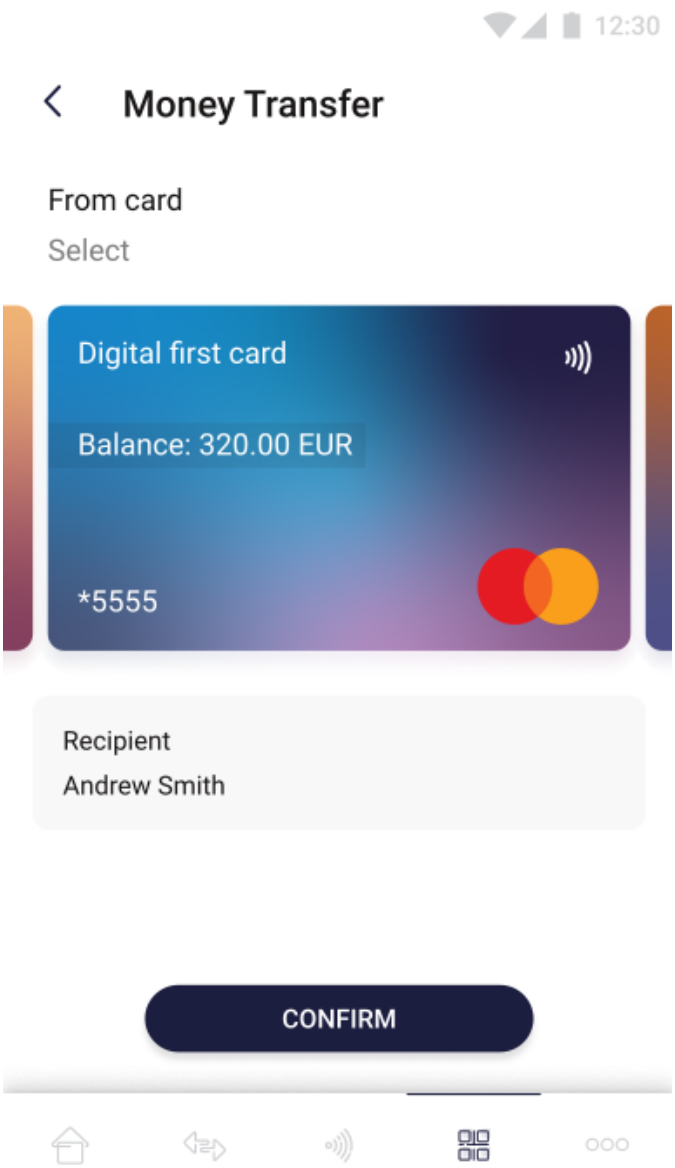
QR Payment flow

The module responsible for QR payments can be divided into two functionalities. The first one allows you to make a payment by scanning the QR code. The second functionality allows user to generate a QR code with which user can order a transfer to his card.

Scan QR code

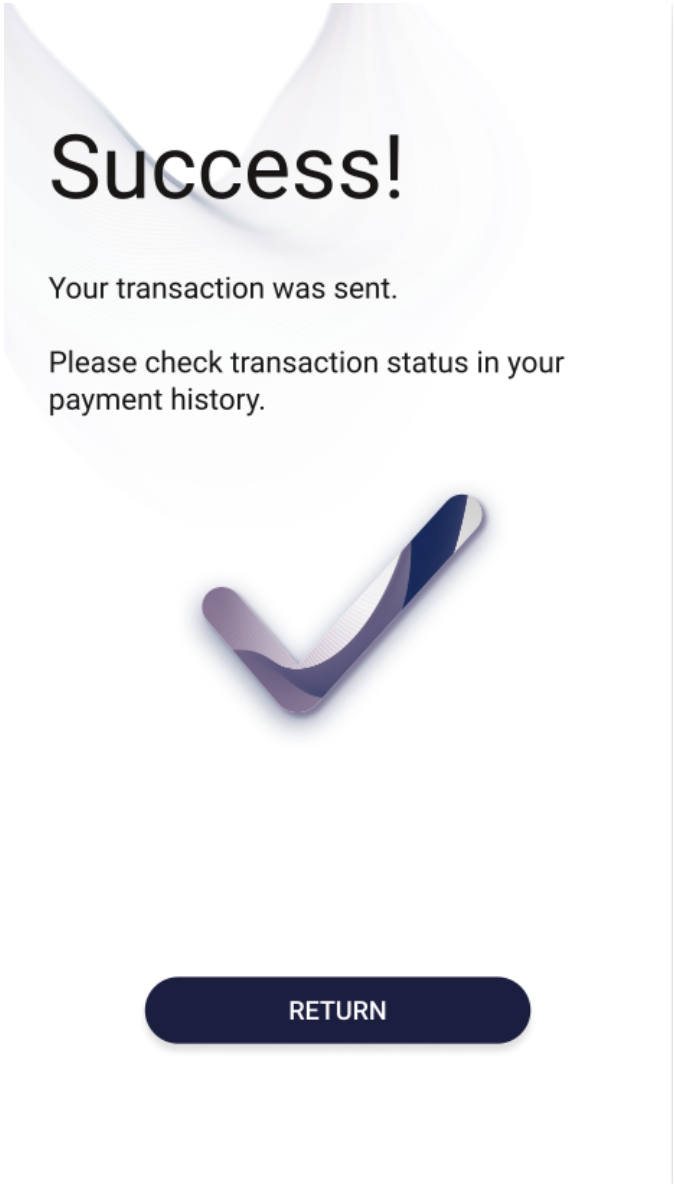
The scan QR option allows user to order a money transfer by scanning the QR code issued by the potential recipient. The execution of this path depends on whether the user's device has a QR scan function. Additionally, the application may request access to the camera of the device used.

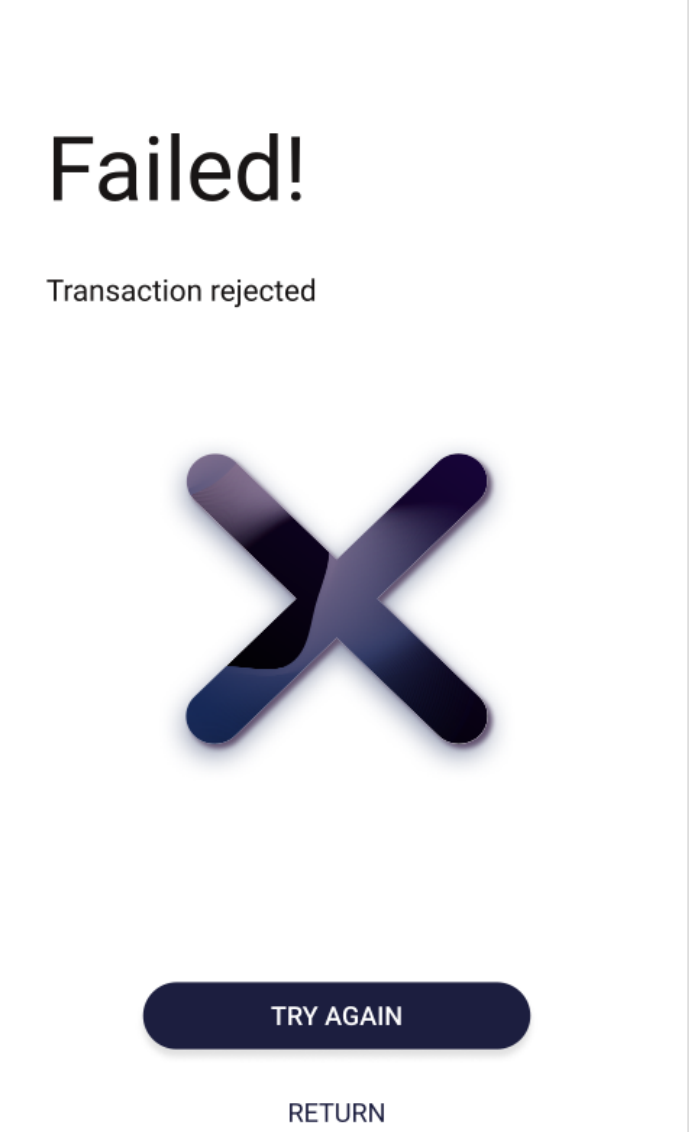
The pictures below show “scan QR code” view:



On the screen above the user selects the scan tab for the application to get all the information needed to make the transfer from the provided QR code.

On the screen above the redirection following scanning the QR code is shown. The user can see to whom the transfer will be made and which of his cards will be debited. If the user is ready to make the transfer, he confirms the action with the confirm button. Similar to a card to card transaction, the 3ds process may be required (See “Card to card” chapter).





On the screen above user is informed about the successfully money transfer. Return button will redirect the user to the main page.

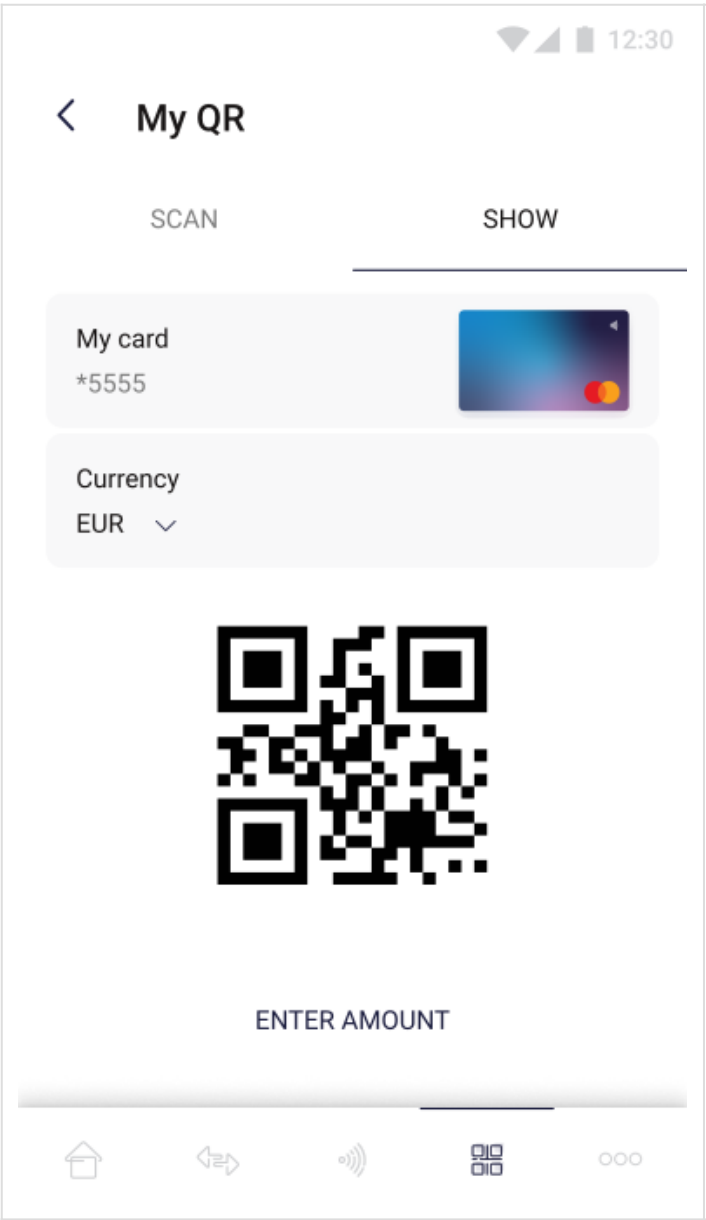
On the screen above user is informed about the unsuccessfully money transfer. In this case user is able to try again the transfer or return to the main page.
Example reasons of the transaction fail:

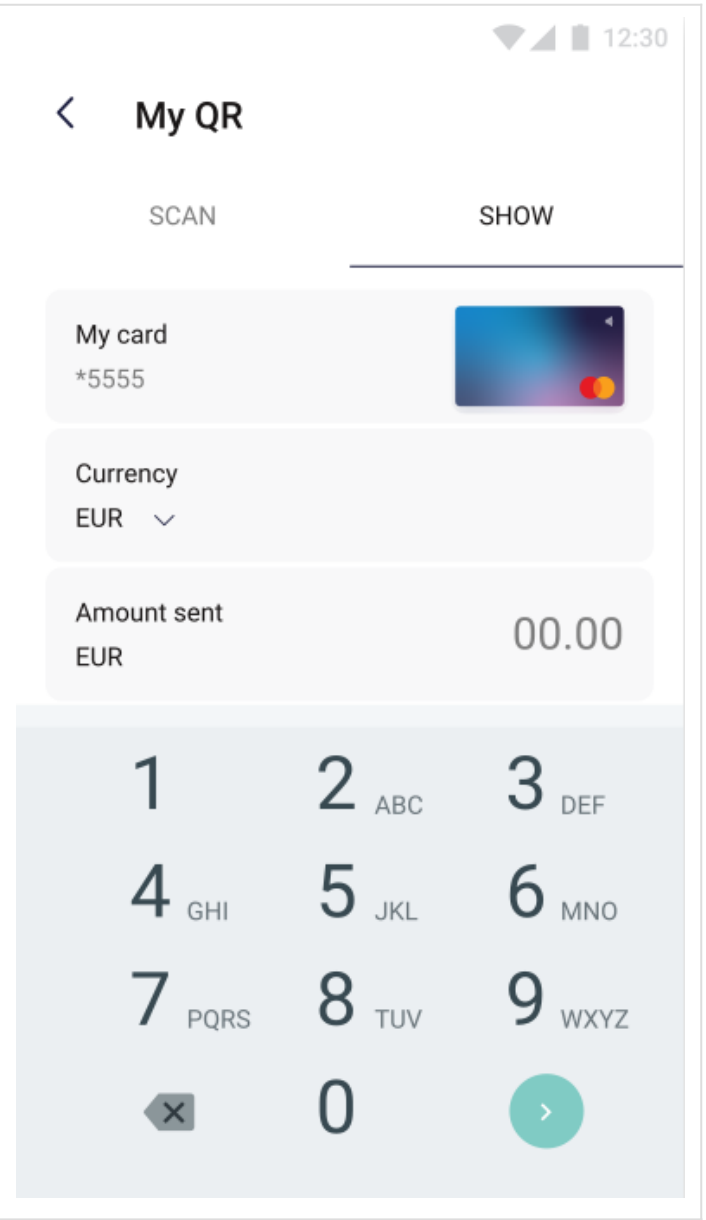
- Luck of funds,
- 3ds failed,
- Invalid card data.

Generate QR code

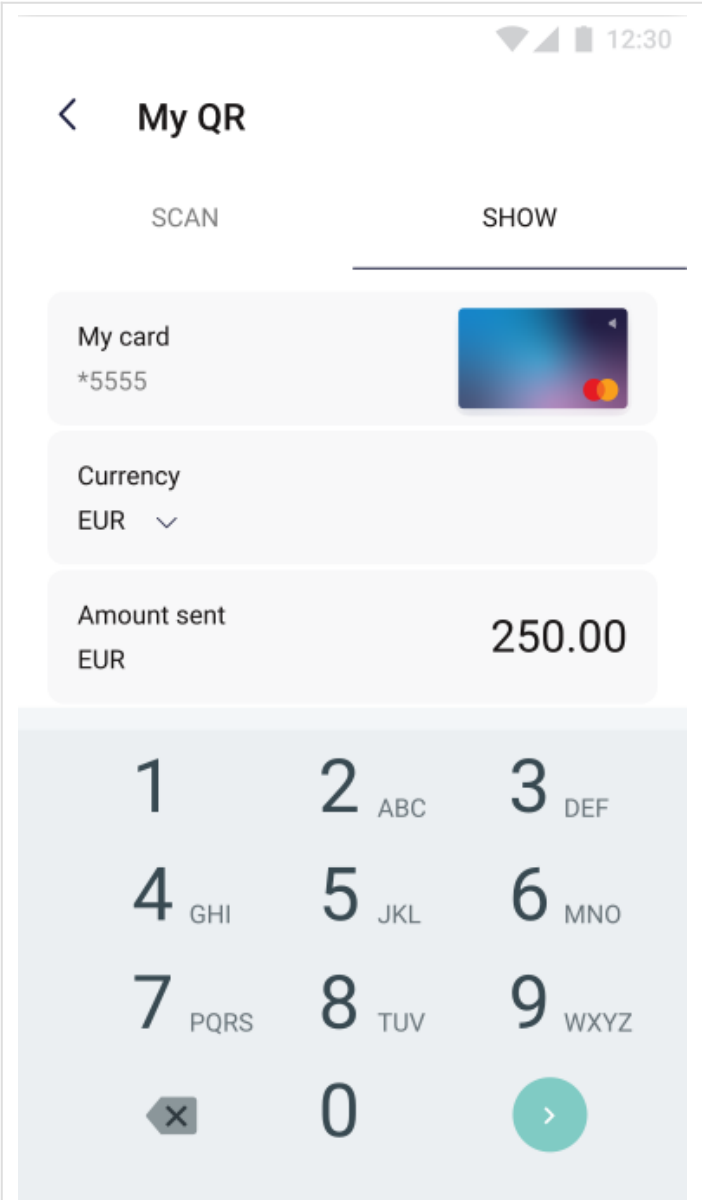
The scan QR option allows user to generate a QR code. With the generated code, the user can have his account topped up by showing the code to another person.

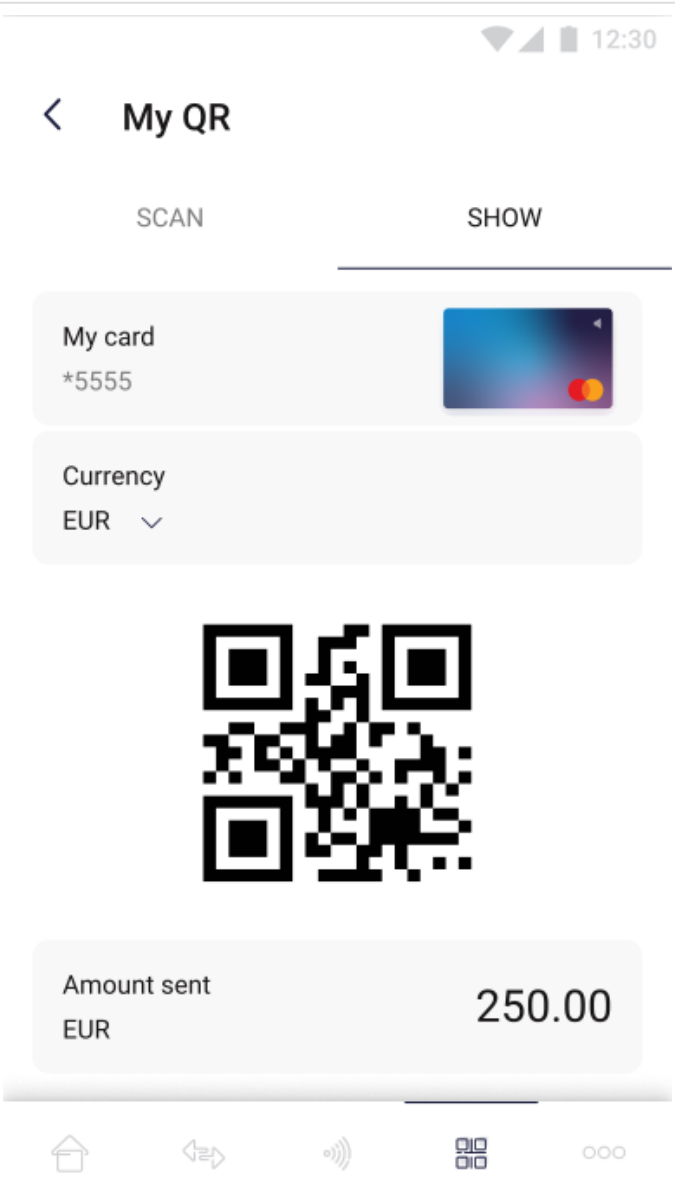
The pictures below show “generate QR code” view:





On the screen above user chooses which of his cards will be debited, and in what currency the transfer is to be made. If the above choices have been made, the user has to enter the amount he wants to receive.





On the screen above the user enters the amount he wants to receive and then confirms the selection with the green button in the lower right corner of the screen. After performing all the required actions, a QR code and information about the ordered transaction (card, currency and amount) are generated on the screen.