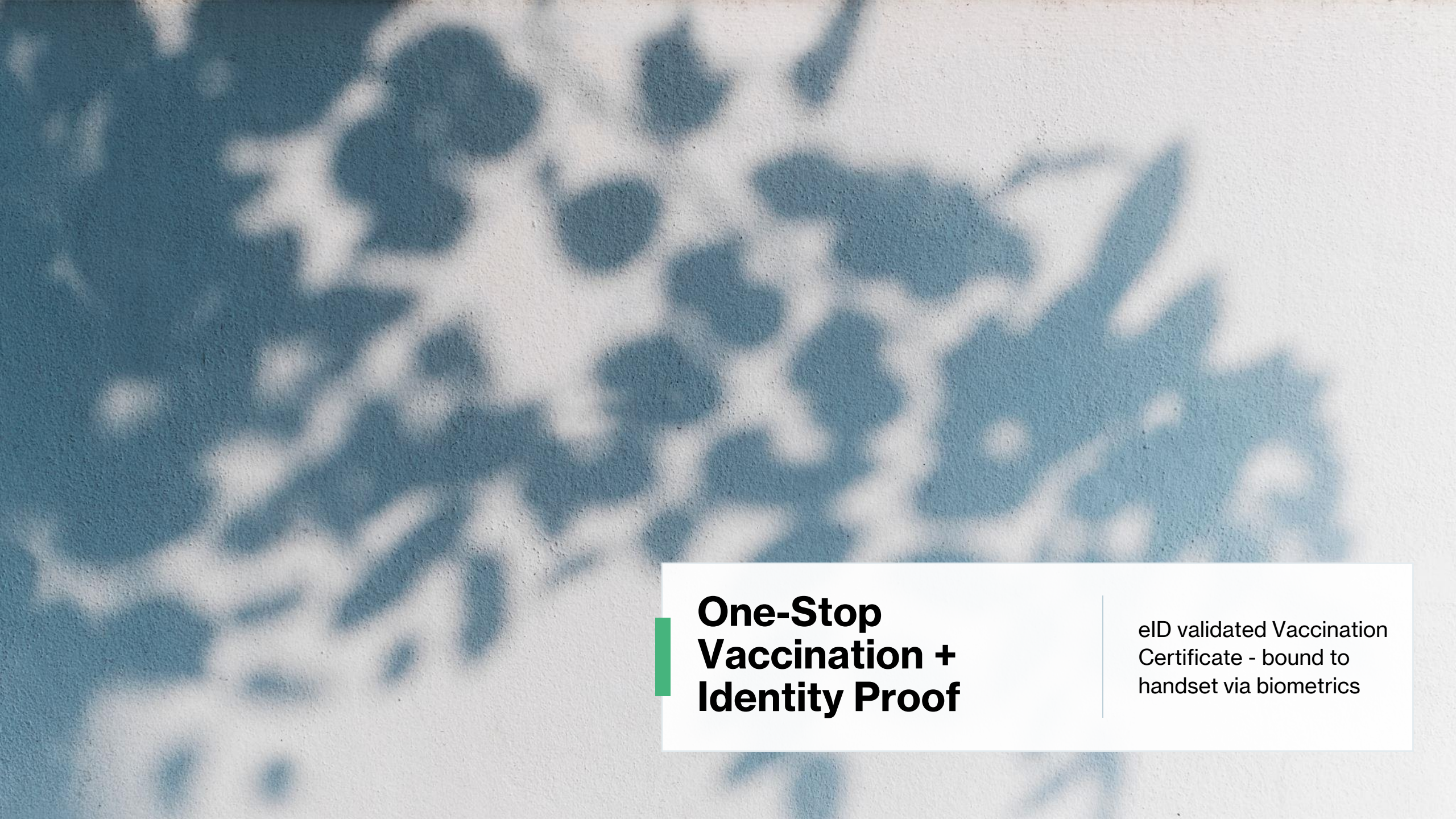


One-Stop Vaccination + Identity Proof

eID validated Vaccination Certificate - bound to handset via biometrics

One-stop Vaccination + Identity

Agenda

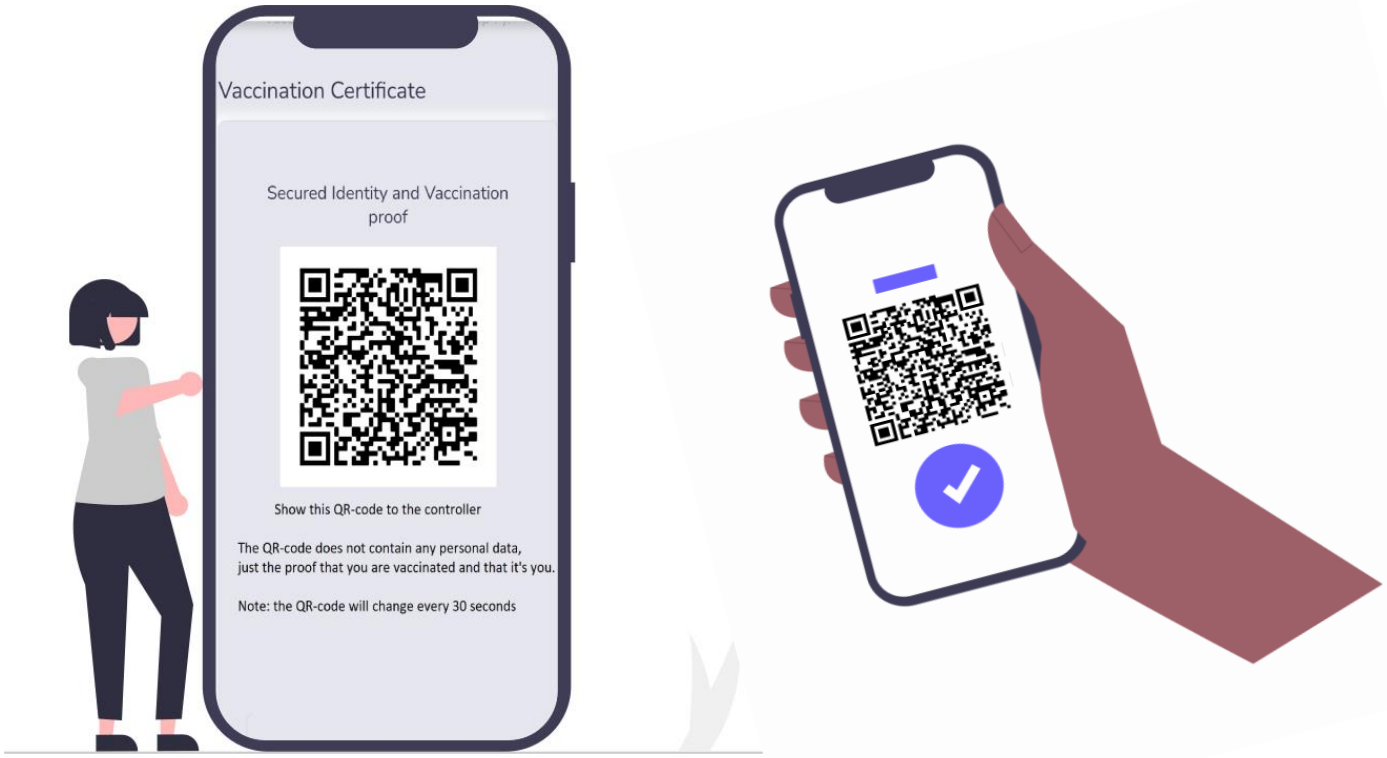
1. Pain Points

2. Solution
    - Enrolment one-stop proof
    - Check by controller

3. Benefits

4. Implementation

What‘s about: Prooven Vaccination and Identity in one QR-Code



User shows **one** dynamic QR-code that proves:

- validated vaccination
- validated identity (eID)
- validated with user‘s fingerprint for 30sec (prevent sharing screenshot with others)

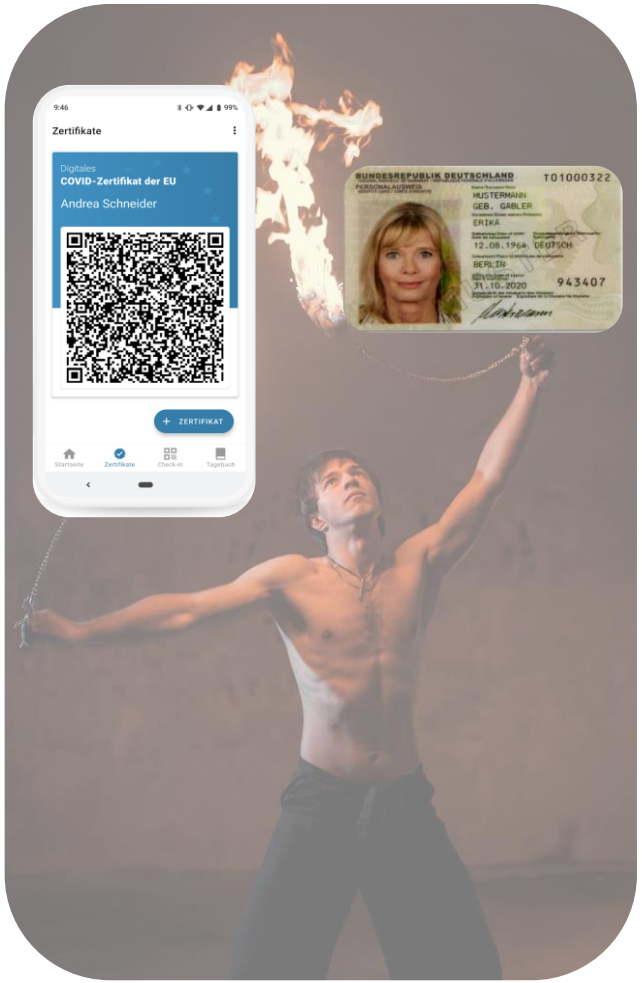
Controller: Checks Vaccination and Identity in **one scan**

No personal data of user revealed just OK or NOK

1. Pain Points

Situation Today

1. Pain Points – Situation Today: Check Vaccination + Identity



1. **Media Discontinuity: Physical ID-Card & Digital Vaccination Certificate**

   => Inconvenient; to juggle with handset and plastic card to present proofs

2. **Legitimate Privacy concerns - Controller gets all personal data of user**

   => Given Name, Name, Birthday, Vaccine, Vaccination Date, Place of Birth, ... all exposed

3. **Manual-, insecure and error-prone process**

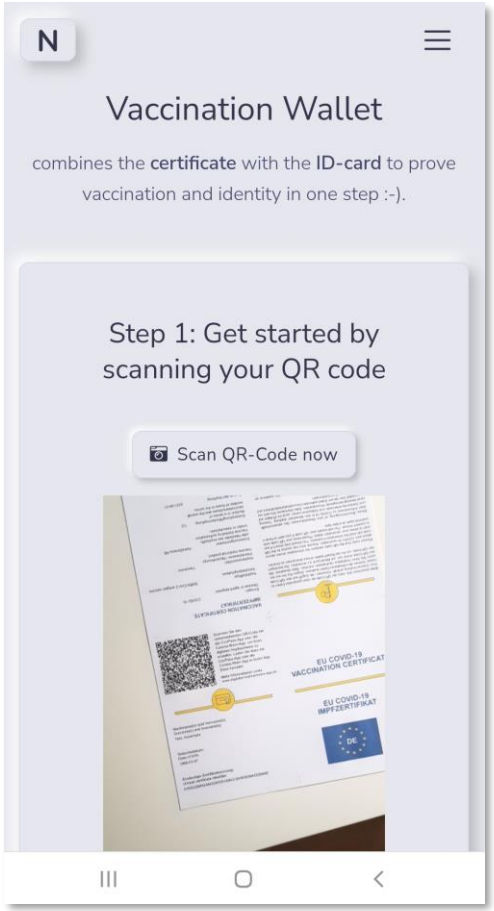
   => Most controller may not check ID-card, no automation possible, only on premises

Sources: Von Bundesregierung der Bundesrepublik Deutschland - Gesetz zur Stärkung der Sicherheit im Pass-, Ausweis- und ausländerrechtlichen Dokumentenwesen, Gemeinfrei, https://commons.wikimedia.org/w/index.php?curid=91186057
unsplash: https://unsplash.com/photos/78fYQxHwxhU

2. Solution

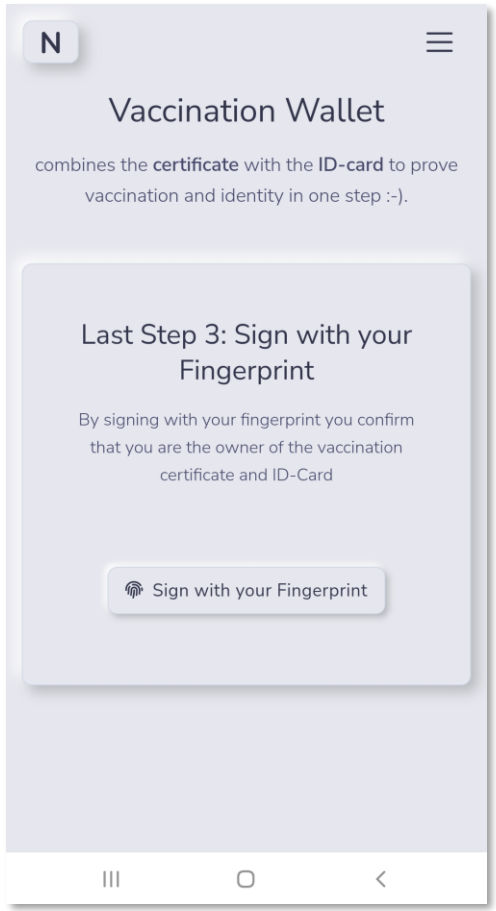
Use Case ‚Enrolment‘
Use Case ‚Check by Controller‘

2a. ‚Enrolment' Use Case: 3 easy and simple steps



1.) Scan the QR-Code of your Vaccination Certificate

2.) Validate Name and Birthday with your eID

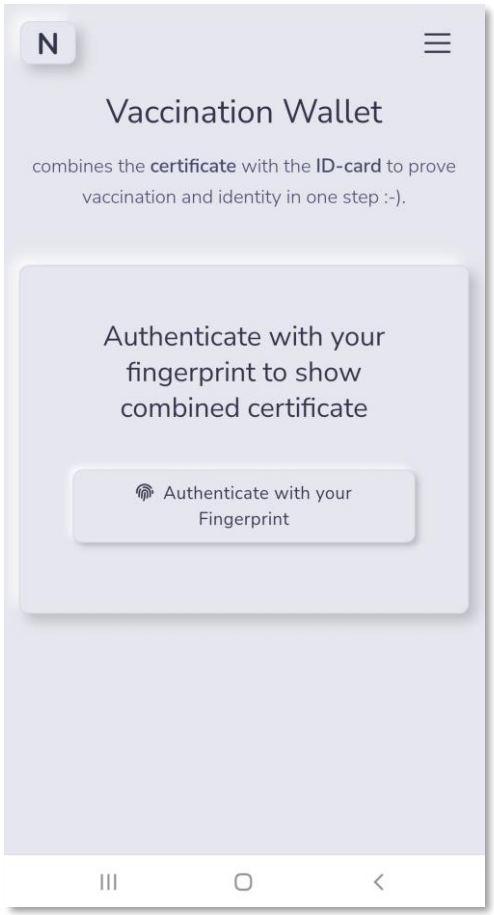


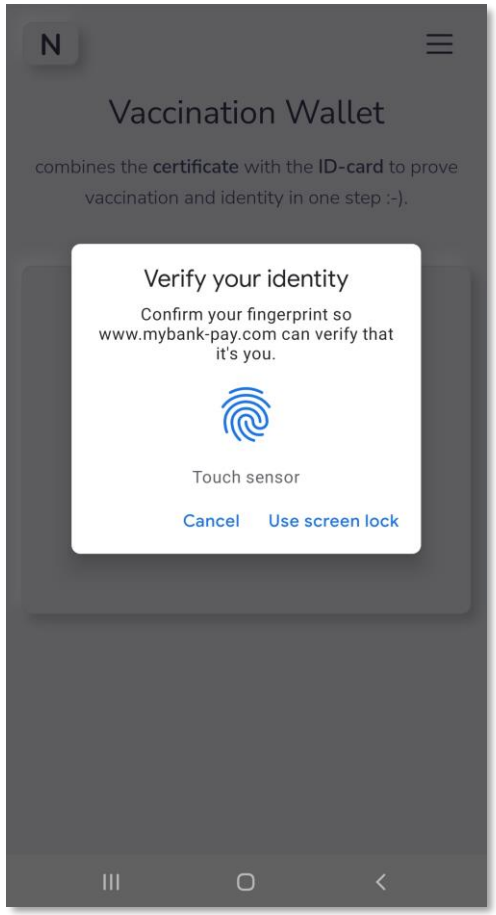
3.) Bind your personal data to Device and Fingerprint

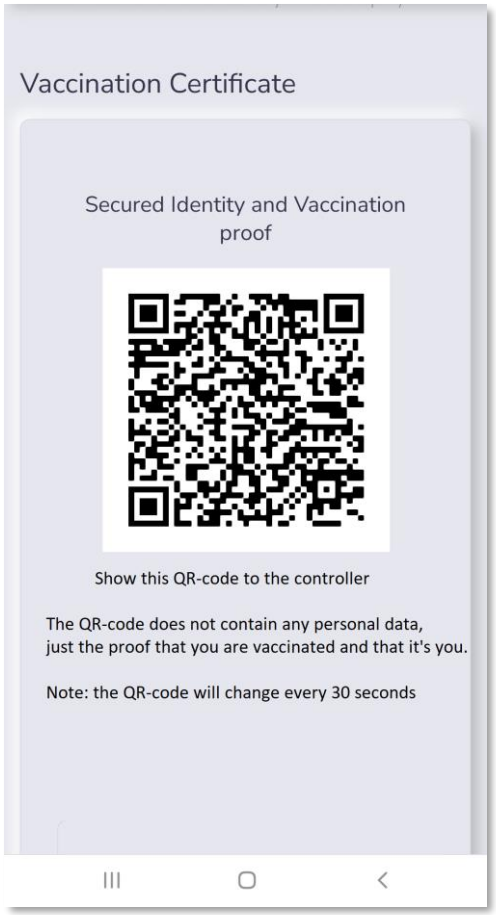
DONE

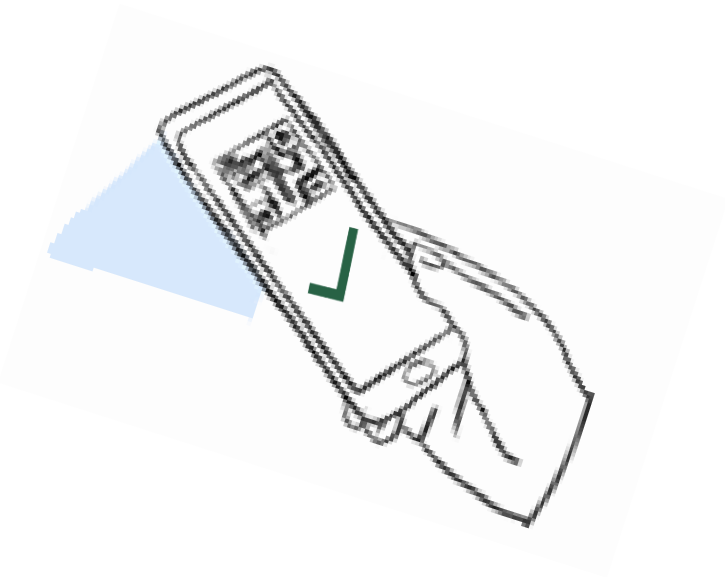
- Vaccination Certificate with eID validated
- And these personal data bound to device and user

2b. ‚Check' Use Case: One touch to prove Vaccination and Identity









Controller: Proof of vaccination and identity with QR code scan

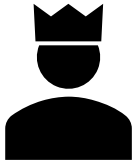
No personal data revealed

Authenticate yourself with your fingerprint – that‘s it!

3. Why

Benefits

3. Why - Unique Benefits



User

- Very convenient to use; just fingerprint
- High 2-factor security
  (Device possesion and biometric inherence)
- No personal data exposed to third parties

Controller

- Very easy to use, just scan one QR-code
- High level of trust and confidence
  (proven vaccination and identity)
- Universal usage; cinema, bars, shops, restaurants..

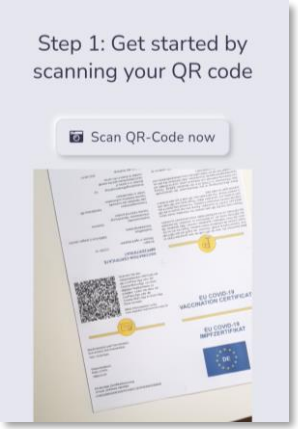
Society

- Create and practice data privacy
- Getting (really) digital
- Tons of more use cases ..

4. How

Implementation ‚Enrolment‘
Implementation ‚Check‘

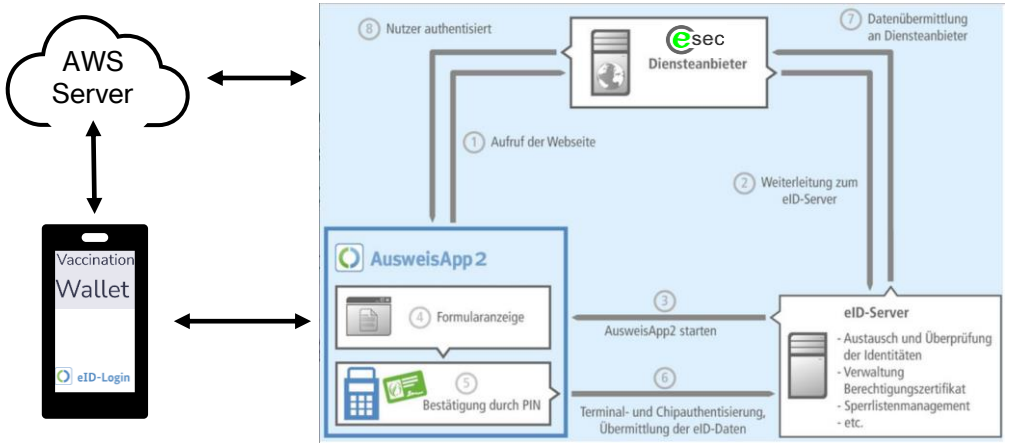
4a. How does it work – ‚Enrolment' Use Case (1/2)



1. Scan of the vaccination certificate:
- check validity of the certificate (signature, expiry date)
- extract personal data (given name, name and birthday)

Implementation:
The QR-code is read by standard JS library
The content is decoded (base45, cbor2 and z-lib decompressed)
The signature of the vaccination certificate is validated by standard JS lib

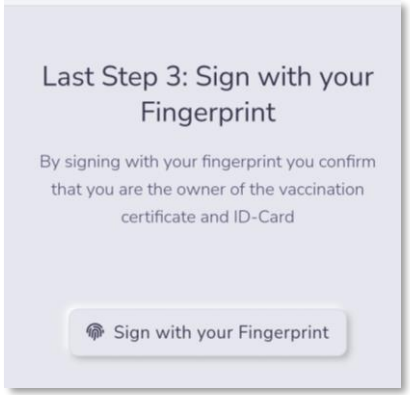
2. Reading electronic ID-card (eID):
- strong authentication of the user (physical ID-card and PIN)
- to get and verify personal data (name and birthday) from ID-card with vaccination certificate



Current Implementation:
Via machine learning, the ID card data is read and OCR'ed on the Cloud server, which performs the comparison with the data from the vaccination certificate

Planned Implementation:
The ‚vaccination wallet' uses the AusweisApp2 SDK, to authenticate the user via eID and PIN and to get proven personal data via the Identity Provider returned

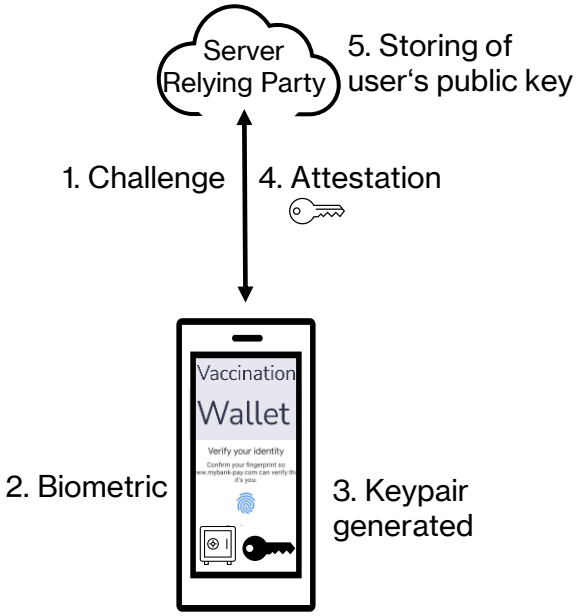
4a. How does it work – ‚Enrolment' Use Case (2/2)



3. WebAuthn/FIDO2 Registration:
- User registers new account with biometric and device to sign and confirm ownership
- Server (relying party) validates attestation response

Implementation:

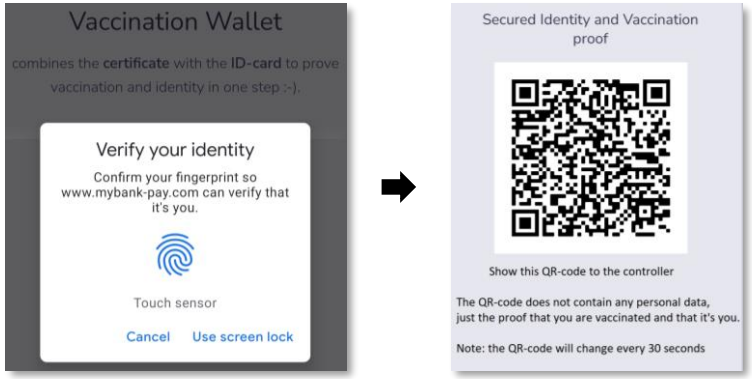
The webapp uses the WebAuthn APIs from the underlying useragent/browser via plain JS
The browser itself uses CTAP2 protocol to call secure authenticator (keystore in the device or external one like Yubikey)
The registration flow generates an attestation object, which is validated by the server (using https://github.com/lbuchs/WebAuthn)



Flow:

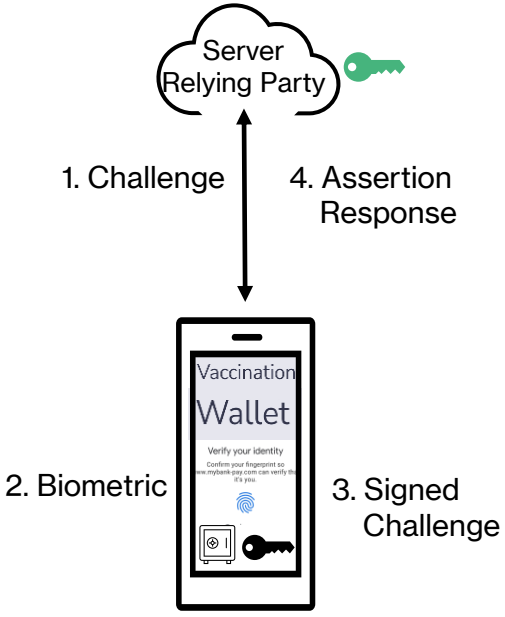
1. The server generates a challenge (hash of vaccination certificate data)
2. The device performs the user verification (biometric or device unlock PIN) before a keypair is generated (access to secret key in secure authenticator only via registered biometrics or device unlock PIN)
3. The device (secret key in the authenticator) signs the challenge and responds to server
4. The application server (acting as Relying Party) receives the attestation and public key
5. The application server stores the User's public key for subsequent authentications

4b. How does it work – ‚Check' Use Case by Controller (1/2)



1. WebAuthn/FIDO2 Authentication Flow and Dynamic QR-Code:
- User performs the authentication flow, using biometrics and device
- Server (relying party) validates assertion response

Implementation:
Use of WebAuthn API and CTAP2 protocol as during registration during enrolment
The authentication flow generates an assertion object, which is validated by the server



Flow:

1. The server generates a challenge (hash of vaccination certificate data and Unix epoch timestamp)

2. The device performs the user authentication (biometric or device unlock PIN) before signing the challenge (access to secret key in secure authenticator only via registered biometrics or device unlock PIN)

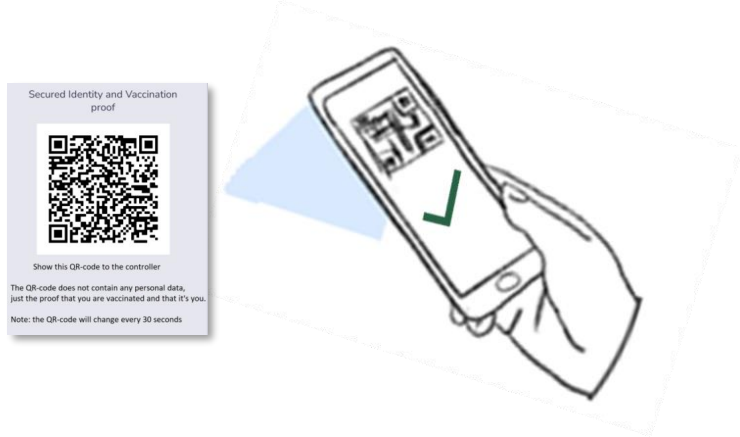
3. The device (secret key in the authenticator) signs the challenge and responds to server

4. The application server (acting as Relying Party) receives and verifies the assertion response

5. The application server signs itself the assertion response with its trusted signing key

6. The application server returns an QR-code, stating that
   a.) the user has a valid vaccination certificate,
   b.) prooven with the User's eID
   c.) bound to this device and the user's fingerprint
   d.) **and valid for 30sec (Unix epoch), to prevent screen copy and sharing with others**

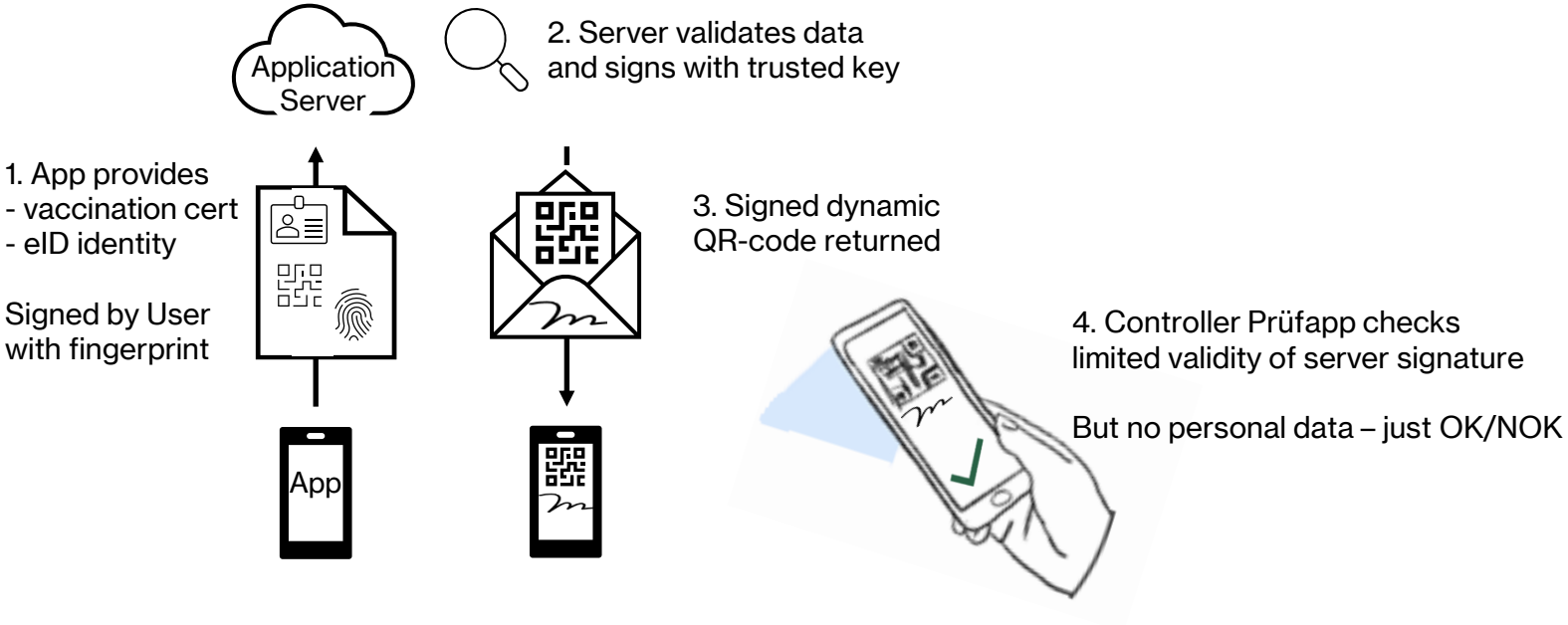
4b. How does it work – ‚Check' Use Case by Controller (2/2)



2. Controller scans the dynamic QR-code and validates signature:
- Controller scans the QR-code (does not see any personal data)
- Prüfapp extracts data and verifies signature (with extracted public key and key chain)
- Prüfapp visualizes result: Ok or NOK

Implementation:
Use plain JS to read QR-code and extract the contents
JS library to verify signature and certificate chain

Illustrative E2E Implementation (dramatically simplified ;-)



Thanks ☺