QR Payment Flow

Flow: QR Payment

Document: LLD-002 **Version:** 1.0 **Date:** 2026-02-21 **Author:** Frontend Architect (AI Agent) **Status:** Draft **Scope:** End-to-end QR payment flow covering camera handling, merchant resolution, payment confirmation, SCA, and error states for both web and mobile

---

1. Overview

QR payments allow Drop users to pay in-store by scanning a merchant's QR code. The payment is initiated via PISP (Payment Initiation Service Provider) directly from the user's bank account under the PSD2 pass-through model. Drop never holds customer funds.

QR Code Format: `drop://pay/{merchantId}`

Payment Fee: 1% of transaction amount (fee is deducted from user's balance: `totalOre = amountOre + feeOre`. Settlement to merchant is separate.)

> **Note:** Database stores amounts in ore (minor units). API accepts NOK and converts internally using `nokToOre()`. Example: 129 NOK = 12900 ore in DB.

Flow summary: Camera permission → QR scan → decode merchant ID → fetch merchant details → amount entry → SCA trigger → payment confirmation → receipt display

---

2. QR Payment Sequence Diagram

```
sequenceDiagram
    actor User
    participant App as Drop App<br/>(Web/Mobile)
    participant Camera as Camera API<br/>(Browser/Native)
    participant API as Drop API<br/>(/api/transactions)
```

```
    participant Bank as User's Bank<br/>(via PISP)
    participant DB as Database

    User->>App: Navigate to /scan
    App->>App: Check auth (useAuth / Bearer token)

    alt Camera available
        App->>Camera: Request camera permission
        Camera-->>App: Permission granted
        App->>App: Show camera viewfinder<br/>with scan frame brackets
    else Camera denied / unavailable
        App->>App: Show "Simuler skanning" button<br/>(demo mode fallback)
    end

    User->>Camera: Point at merchant QR code
    Camera->>App: Decode QR: "drop://pay/{merchantId}"

    App->>App: Parse merchantId from QR URI
    App->>API: GET /api/merchants/{merchantId}
    API->>DB: SELECT merchant WHERE id = ?
    API-->>App: { merchantId, businessName, category }

    App->>App: Show merchant info + amount input
    User->>App: Enter amount (e.g., 129 NOK)

    App->>App: Calculate fee (1% = 1.29 NOK)
    App->>App: Show payment summary<br/>(amount, fee, total, source account)

    User->>App: Tap "Betal nå" (confirm)

    App->>API: POST /api/transactions/qr-payment<br/>{ merchantId, amount }
    API->>API: Rate limit check (10/min)
    API->>DB: Verify merchant exists
    API->>DB: Get user's primary bank account
    API->>API: Calculate fee (1% of amount)

    Note over API,Bank: Production: PISP initiates<br/>payment from user's bank.<br/>Demo: Direct DB debit.

    API->>DB: BEGIN TRANSACTION
```

```
    API->>DB: UPDATE bank_accounts SET balance = balance - (amount + fee)
    API->>DB: INSERT transaction (type=qr_payment, status=completed)
    API->>DB: COMMIT

    API-->>App: 201 { transaction }

    App->>App: Show success screen<br/>(checkmark, merchant, amount, fee)

    User->>App: Tap "Tilbake til hjem"
    App->>App: Navigate to /dashboard
```

---

3. Payment Flow State Diagram

```
stateDiagram-v2
    [*] --> Idle: Navigate to /scan

    Idle --> RequestingPermission: Camera API available
    Idle --> SimulationMode: No camera / demo mode

    RequestingPermission --> Scanning: Permission granted
    RequestingPermission --> PermissionDenied: Permission denied

    PermissionDenied --> Scanning: User grants in settings
    PermissionDenied --> SimulationMode: Use simulation

    SimulationMode --> MerchantResolved: Click "Simuler skanning"

    Scanning --> Decoding: QR code detected
    Decoding --> MerchantResolved: Valid drop:// URI
    Decoding --> InvalidQR: Not a Drop QR code

    InvalidQR --> Scanning: Dismiss error, retry scan

    MerchantResolved --> AmountEntry: Merchant details loaded
    MerchantResolved --> MerchantNotFound: Merchant lookup failed

    MerchantNotFound --> Scanning: Go back, scan again
```

```
    AmountEntry --> PaymentReview: Amount entered + confirmed
    AmountEntry --> Scanning: Cancel / go back

    PaymentReview --> Processing: Tap "Betal nå"
    PaymentReview --> AmountEntry: Edit amount

    Processing --> Success: Payment completed (201)
    Processing --> InsufficientFunds: Balance too low
    Processing --> PaymentFailed: API error

    InsufficientFunds --> AmountEntry: Adjust amount
    PaymentFailed --> PaymentReview: Retry

    Success --> [*]: Navigate to dashboard
```

---

4. Camera Permission Handling

4.1 Camera Permission Table (iOS / Android)

Platform	Permission API	First Request	After Denial	Settings Redirect
iOS (Safari)	`navigator.mediaDevices.getUserMedia()`	System prompt: "getdrop.no would like to access the camera"	Blocked silently; must reset in Safari Settings → getdrop.no → Camera	Link to Settings not programmatically available
iOS (Expo)	`expo-camera` `Camera.requestCameraPermissionsAsync()`	System prompt: "Drop would like to access the camera"	Returns `{ status: 'denied' }`; use `Linking.openSettings()`	`Linking.openSettings()` → iOS Settings → Drop → Camera
Android (Chrome)	`navigator.mediaDevices.getUserMedia()`	System prompt: "Allow getdrop.no to use your camera?"	Blocked; user must tap lock icon → Site settings → Camera → Allow	Site settings accessible via address bar
Android (Expo)	`expo-camera` `Camera.requestCameraPermissionsAsync()`	System prompt: "Allow Drop to take pictures and record video?"	Returns `{ status: 'denied' }`; `{ canAskAgain: false }` after permanent deny	`Linking.openSettings()` → App Info → Permissions → Camera

4.2 Fallback Behavior

When camera is unavailable or denied:

- **Web:** Shows "Simuler skanning" button that triggers a demo merchant scan (Ahmetov Kebab, merchant_001)
- **Mobile:** Shows camera placeholder (gray box with QR icon) + "Simuler skanning" button + nearby merchants list (hardcoded: Ahmetov Kebab, Kafe Oslo, Narvesen)

---

5. QR Code Format and Validation

Field	Value	Validation
URI scheme	`drop://`	Must match exactly
Path	`pay/{merchantId}`	Must start with `pay/`
Merchant ID format	`mer_` prefix + flexible identifier	No strict regex enforced (e.g., `mer_demo1` is valid)
Example	`drop://pay/mer_demo1`	Validated by DB lookup

HMAC verification: QR codes may optionally include `timestamp` and `signature` parameters for HMAC-SHA256 verification using the merchant's `qr_hmac_key`. Verification is performed only when both `qrTimestamp` and `qrSignature` are present in the request. If omitted, the payment proceeds without cryptographic QR verification.

Invalid QR handling:

- Non-Drop QR codes: Show "Ugyldig QR-kode. Vennligst skann en Drop-butikks QR-kode."
- Malformed merchant ID: Show "Ugyldig betalingskode."
- Empty scan result: Continue scanning (do not trigger error)

---

6. Error States

Error	HTTP Status	Cause	User-Facing Message (Norwegian)	Recovery
Invalid QR	N/A (client)	Not a `drop://pay/` URI	"Ugyldig QR-kode. Skann en Drop-butikks QR-kode."	Retry scan
Merchant Not Found	404	Merchant ID not in database	"Butikken ble ikke funnet. QR-koden kan være utdatert."	Scan different QR

Error	HTTP Status	Cause	User-Facing Message (Norwegian)	Recovery
Insufficient Funds	402	Bank balance < amount + fee	"Ikke nok penger på kontoen. Saldo: {balance} NOK."	Reduce amount or top up bank
No Bank Account	400	No linked bank account	"Ingen bankkonto koblet. Koble en konto først."	Navigate to /accounts
Rate Limited	429	>10 payments/min	"For mange betalinger. Vent litt."	Wait and retry
Network Error	N/A	No connectivity	"Ingen nettverkstilkobling."	Retry when online
Server Error	500	Internal error	"Noe gikk galt. Prøv igjen."	Retry
Camera Error	N/A	Camera hardware failure	"Kameraet fungerer ikke. Bruk 'Simuler skanning'."	Use simulation mode

---

7. UI Components

7.1 Web — Scan Page (`/scan`)

State	UI Elements	Components Used
Scanning	Dark background (#0F172A), camera viewfinder with gold corner brackets (#D4A017), instruction text, "Simuler skanning" button, BankID/Vipps badges	BottomNav, Button, ArrowLeft/Camera (lucide)
Payment	White background, merchant icon (gold gradient circle), merchant name (Fraunces font), amount display (4xl), source account info, "Betal nå" button (green), "Avbryt" button	BottomNav, Button, ChevronLeft (lucide)
Paying	Loading spinner overlay	Spinner component
Success	Checkmark icon (green), transaction details, merchant name, amount, fee	BottomNav, Check/Store (lucide)

7.2 Mobile — Scan Screen (`(tabs)/scan.js`)

State	UI Elements
Scanning	Gray camera placeholder, QR icon, "Skann QR-kode" text, "Simuler skanning" button, nearby merchants list
Payment	Merchant info, amount input, confirm button
Success	Confirmation with "Tilbake til hjem" button

7.3 Figma Reference

Source of truth: `mockups/figma-make-export/src/app/screens/ScanQR.tsx`

- Dark scanning mode with gold bracket viewfinder
- Payment confirmation with merchant details and amount
- Green "Betal nå" CTA button

---

8. Data Flow

8.1 Request: POST /api/transactions/qr-payment

```
{
  "merchantId": "mer_a1b2c3d4e5f6g7h8",
  "amount": 129
}
```

8.2 Response: 201 Created

```
{
  "data": {
    "id": "tx_qr_a1b2c3d4e5f6g7h8",
    "type": "qr_payment",
    "status": "completed",
    "amount": 129,
    "currency": "NOK",
    "fee": 1.29,
    "feePercent": 1,
```

```
    "merchantName": "Ahmetov Kebab",
    "merchantId": "mer_1",
    "fromAccount": "DNB",
    "createdAt": "2026-02-21T14:30:00.000Z"
  }
}
```

8.3 Database Operations (Atomic Transaction)

```
BEGIN;
UPDATE bank_accounts SET balance = balance - 130.29 WHERE id = ? AND user_id = ?;
INSERT INTO transactions (id, user_id, type, status, amount, currency, fee, merchant_id,
created_at, completed_at)
  VALUES (?, ?, 'qr_payment', 'completed', 129, 'NOK', 1.29, ?, datetime('now'),
datetime('now'));
COMMIT;
```

---

9. Production vs Demo Differences

Aspect	Demo (Current)	Production (Phase 2+)
Camera	Simulated scan button	Real camera scanning via `expo-camera` or `getUserMedia()`
Payment execution	Direct DB balance debit	PISP initiation via Open Banking API
SCA	Not implemented	BankID SCA required for each payment
Merchant verification	Static seed data (Ahmetov Kebab)	Live Brønnøysund org number verification
Fee handling	Fee deducted from user's balance (`totalOre = amountOre + feeOre`)	Merchant settlement is separate from user debit
Settlement	Instant (DB update)	T+1 or T+2 settlement to merchant bank account

---

10. Accessibility Considerations (WCAG 2.1 AA)

Requirement	Implementation
Camera alternative	"Simuler skanning" button provides non-camera path
Amount input	Labeled with "Beløp" and suffixed with "NOK"
Confirmation	"Betal nå" button clearly labeled; "Avbryt" provides escape
Success feedback	Visual checkmark + text confirmation of payment
Color contrast	Gold (#D4A017) on dark (#0F172A) = 5.2:1 ratio (passes AA)
Screen reader	Merchant name and amount announced on payment confirmation

11. Cross-References

- **QR payment API:** `POST /api/transactions/qr-payment` — See API Reference
- **Merchant registration:** `POST /api/merchants/register` — See API Reference
- **Transaction schema:** `transactions` table — See Database Schema
- **Merchant schema:** `merchants` table — See Database Schema
- **Component overview:** See component-overview.md
- **Figma scan screen:** `mockups/figma-make-export/src/app/screens/ScanQR.tsx`
- **Web scan page:** `src/drop-app/src/app/scan/page.tsx` — See PAGES.md
- **Mobile scan screen:** `src/drop-mobile/app/(tabs)/scan.js` — See MOBILE-APP.md
- **Merchant onboarding flow:** See flow-merchant-onboarding.md
- **PSD2 PISP details:** See open-banking-aisp-pisp.md

Revision #7
Created 2026-02-21 05:58:53 UTC by John
Updated 2026-06-28 20:01:37 UTC by John