NiziPOS API Documentation

The NiziPOS background service exposes a REST API on `http://127.0.0.1:9121` for controlling the connected UART display device.

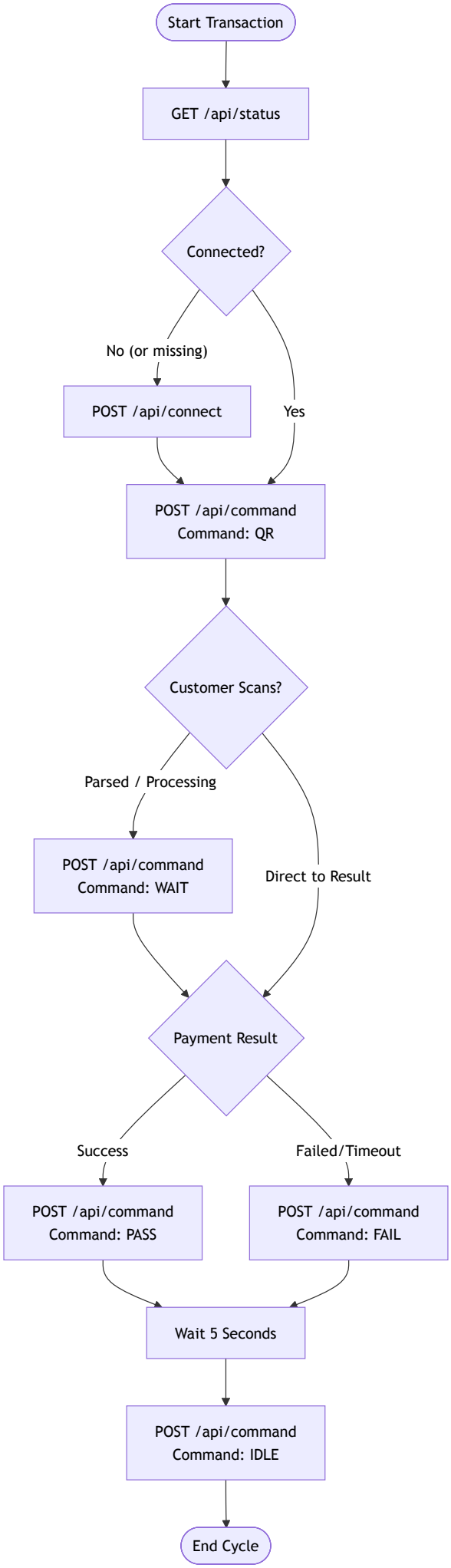
Authentication is handled via a fixed API key. This key is provided to authorized users via email and must be included in the `X-API-Key` header for all requests.

Fixed API Token: `nizipos-fixed-secret-token` (Placeholder — refer to your distribution email)

Header	Description
`X-API-Key`	Your secure API token.

Recommended Integration Flow

To handle a dynamic QR payment lifecycle, we recommend the following step-by-step developer flow.


Start Transaction
GET /api/status
Connected?
No (or missing)
Yes
POST /api/connect
POST /api/command
Command: QR
Customer Scans?
Parsed / Processing
Direct to Result
POST /api/command
Command: WAIT
Payment Result
Success
Failed/Timeout
POST /api/command
Command: PASS
POST /api/command
Command: FAIL
Wait 5 Seconds
POST /api/command
Command: IDLE
End Cycle

Endpoints

1. Connection Status

`GET /api/status`
Returns the current connection state.

Response (JSON):

```
{
  "connected": true,
  "port": "COM15"
}
```

2. Send Command

`POST /api/command`
Sends a raw command string to the device.

Common Command Strings:

- **IDLE**: `IDLE` (Returns to idle screen)
- **QR Code**: `QR**<amount>**<scan_text>**<payload>`
- **Loading/Wait**: `WAIT**<amount>**<message>`
- **Success**: `PASS**<title>**<message>`
- **Failure**: `FAIL**<amount>**<message>`

Example Body (JSON) Requests:

IDLE Request:

```
{
  "command": "IDLE"
}
```

QR Code Requests:

Standard scan message:

```
{
  "command": "QR**<amount>**<SCAN TO PAY/ Customer Name/ Bill No>**dynamic_qr_string
```

```
}
```

With client's name in the scan message:

```
{
  "command": "QR**Rs. 150.00**JOHN DOE**dynamic_qr_string_payload"
}
```

With order number or business name:

```
{
  "command": "QR**Rs. 150.00**ORDER #1024**dynamic_qr_string_payload"
}
```

Loading/Wait Request:

```
{
  "command": "WAIT**150.00**Please wait..."
}
```

Success Request:

```
{
  "command": "PASS**SUCCESS!**Payment successful"
}
```

Failure Request:

```
{
  "command": "FAIL**150.00**Payment Failed"
}
```

3. Update Settings

`POST /api/settings`
Updates device parameters.

Body (JSON):

```
{
  "volume": 80,
```

```
  "brightness": 100,
  "screentime": 300
}
```

4. Connect Device

`POST /api/connect`
Triggers a connection attempt.

Body (JSON):

```
{
  "port": "COM3"
}
```

Note: Set `"port": null` for auto-detect.

Response (JSON):

```
{
  "success": true,
  "port": "COM3",
  "error": null
}
```

5. Disconnect Device

`POST /api/disconnect`
Safely closes the serial connection.

6. Upload Image

`POST /api/upload-image`
Uploads and displays a JPEG image.

Form Data:

- `image` : Binary JPEG file.
- `size` : (Optional) Target size, e.g., `"320x480"` .

WebSocket (Socket.IO) Communication

The background service uses Socket.IO for real-time status updates and command feedback.

Connection URL: `ws://127.0.0.1:9121` or `http://127.0.0.1:9121`

Authentication

Socket.IO connections require a valid API token provided in the `auth` object during the initial handshake.

Example (Client-side):

```
const socket = io("http://127.0.0.1:9121", {
  auth: {
    token: "your-fixed-secret-token"
  }
});
```

Connections missing the token or using an invalid token will be automatically rejected by the server.

Client-to-Server Events

`send_command`

Sends a raw command to the device (alternative to the POST `/api/command` endpoint).

Payload:

```
{
  "command": "IDLE"
}
```

Server-to-Client Events

`device_status`

Emitted immediately upon connection and whenever the device connection state changes.

Payload:

```
{
  "connected": true,
  "port": "COM15"
```

```
}
```

`command_result`

Emitted after a command sent via `send_command` is processed.

Payload:

```
{
  "command": "IDLE",
  "success": true,
  "error": null
}
```

Error Handling

Status Code	Description
`200`	OK. Operation completed or data returned.
`400`	Bad Request. Missing parameters or invalid data.
`401`	Unauthorized. Missing or invalid `X-API-Key`.
`403`	Forbidden. Attempted access from non-localhost IP.