



The Standards People



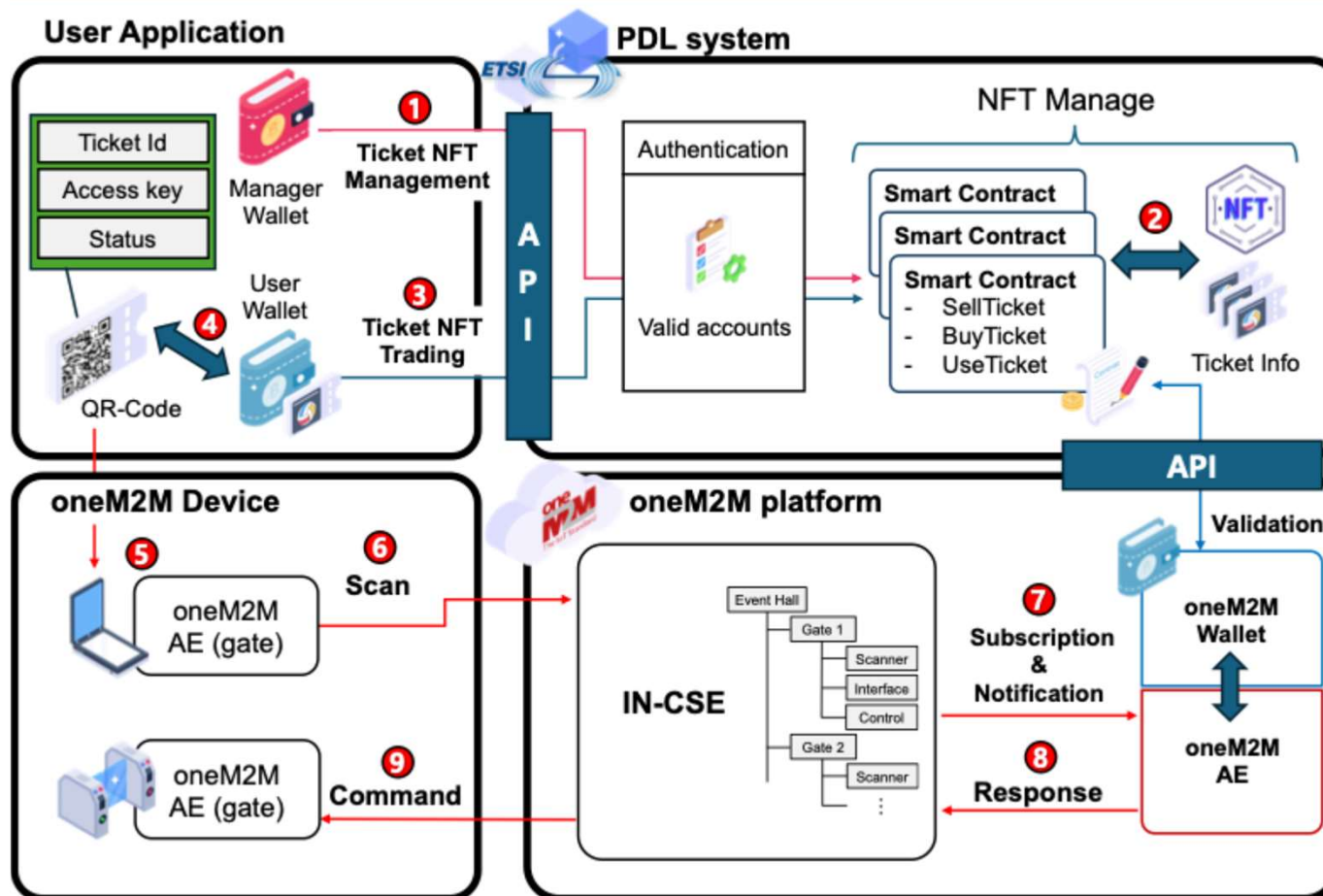
# Demo: Consideration of using Blockchain in oneM2M

Presented by: **Sejong University**  
**JiEun Lee,**  
**JiHo Lee,**  
**JaeSeung Song**

For: **PDL#18**

25.06.2024

# Overview



## About Demo

---

- ✔ In this demo, we'll be checking the APIs that the application sends to the blockchain network.
- ✔ Due to incomplete development of the private network, this demo uses the Ethereum testnet.
- ✔ Therefore, all wallet addresses are assumed to be authorized.

## About Demo

---

- ✓ In this demo, we're going to show you the following features:
  - ✓ Ticket minting
  - ✓ Get ticket information
  - ✓ Buy ticket
  - ✓ Using ticket
  - ✓ Reselling ticket

# Demo

## User application for PDL PoC#04 1.0 OAS 3.0

[/v3/api-docs](#)

This is a PoC for integration between oneM2M and PDL.

- This demo version utilizes a public testnet (Klaytn Baobab)
- Users can utilize three main functions :
  - Minting: Create an NFT ticket. 2.
  - GetInfo: Get information of NFT tickets (event name, seat number, price, status, etc.)
  - Trade: Trade NFT tickets. The transaction works in the following order
    1. Enter a price lower than the generated price and change the NFT to sell status.
    2. The user checks the sale status of the NFT and sends a purchase request.
    3. The seller chooses to accept or reject the purchase request.

Servers

### ticket-controller ^

PUT	/ticket/minter/remove	∨
PUT	/ticket/minter/add	∨
POST	/ticket/use	∨
POST	/ticket/sell	∨
POST	/ticket/mint	∨

# Demo

## 📍 Ticket minting

**POST** /ticket/mint ⌵

Parameters Try it out

No parameters

Request body *required* application/json

Example Value | Schema

```
{
  "eventName": "string",
  "seatNumber": "string",
  "expiredDate": 0,
  "price": 0
}
```

Responses

Code	Description	Links
200	OK	No links

Media type \*/\*

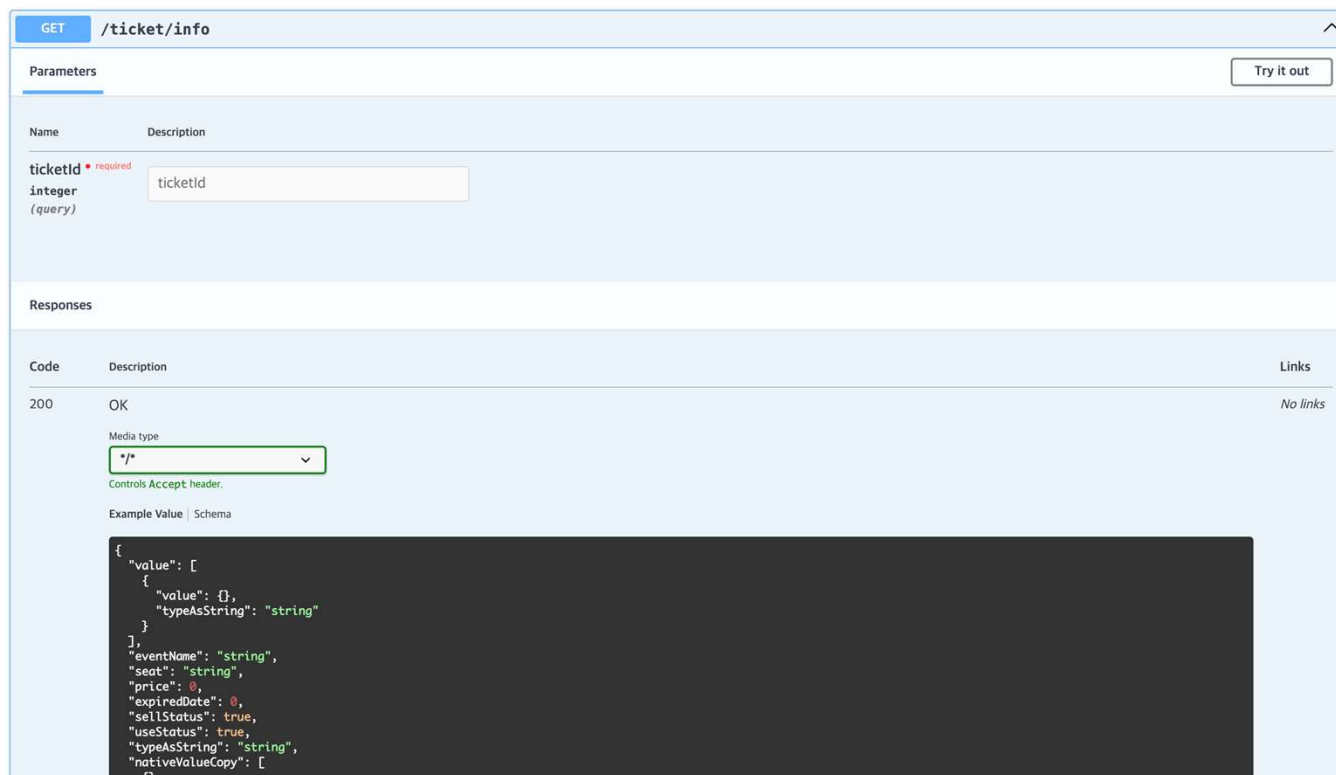
Controls Accept header.

Example Value | Schema

```
{
  "transactionHash": "string",
  "transactionIndex": 0,
  "blockHash": "string",
}
```

# Demo

## Get ticket information



The screenshot shows an API documentation interface for the endpoint `GET /ticket/info`. The interface is divided into several sections:

- Parameters:** A table with columns 'Name' and 'Description'. It lists a required parameter `ticketId` of type `integer` (query). A text input field contains the value `ticketId`. A 'Try it out' button is located in the top right of this section.
- Responses:** A table with columns 'Code', 'Description', and 'Links'. It shows a `200` status code with the description 'OK' and 'No links'.
- Media type:** A dropdown menu is set to `*/*`. Below it, the text 'Controls Accept header.' is visible.
- Example Value / Schema:** A dark-themed code block displays a JSON schema snippet:

```
{
  "value": [
    {
      "value": {},
      "typeAsString": "string"
    }
  ],
  "eventName": "string",
  "seat": "string",
  "price": 0,
  "expiredDate": 0,
  "sellStatus": true,
  "useStatus": true,
  "typeAsString": "string",
  "nativeValueCopy": [
    {}
  ]
}
```

# Demo

## Buy ticket

POST /ticket/buy-request

Parameters Try it out

Name	Description
ticketId <small>required</small>	ticketid

integer (query)

Responses

Code	Description	Links
200	OK	No links

Media type: \*/\*

Controls Accept header.

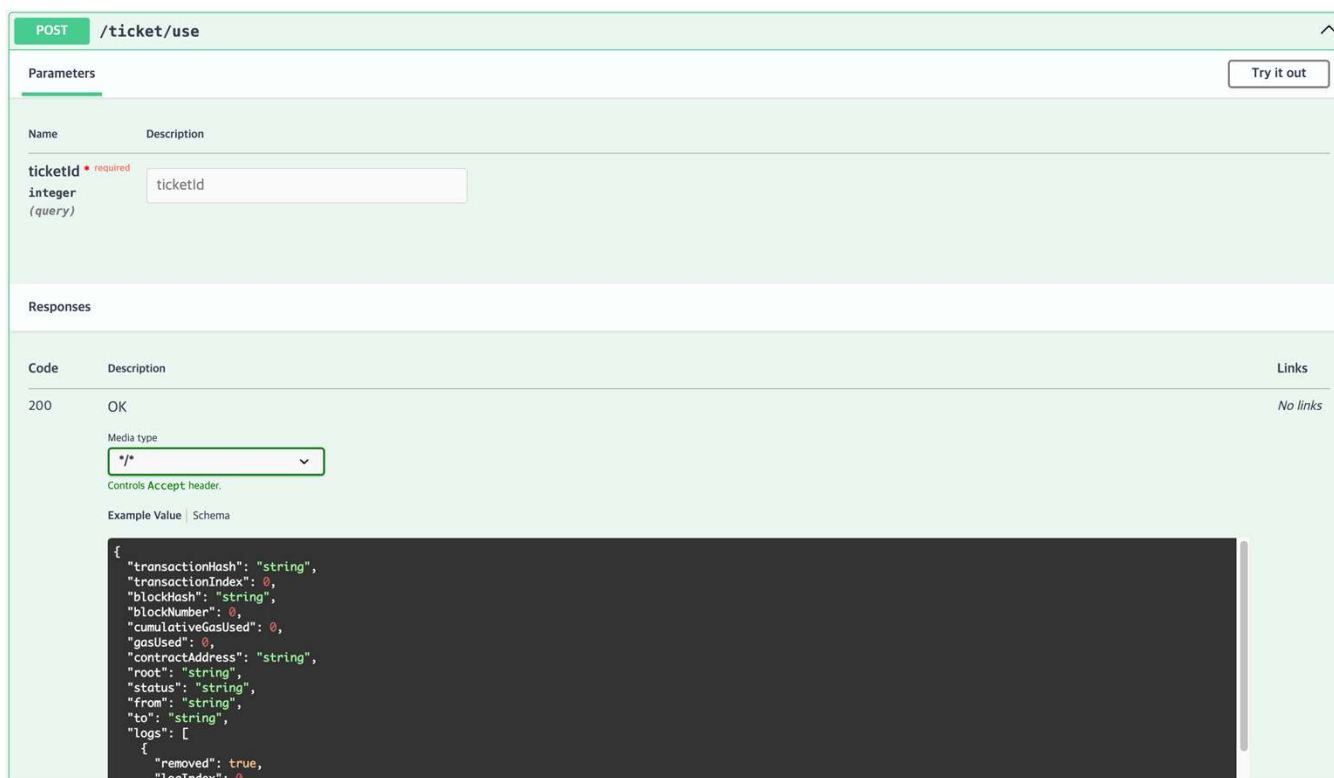
Example Value | Schema

```
{
  "transactionHash": "string",
  "transactionIndex": 0,
  "blockHash": "string",
  "blockNumber": 0,
  "cumulativeGasUsed": 0,
  "gasUsed": 0,
  "contractAddress": "string",
  "root": "string",
  "status": "string",
  "from": "string",
  "to": "string",
  "logs": [
    {
      "removed": true
```



# Demo

## Using ticket



The screenshot shows an API documentation interface for the endpoint `POST /ticket/use`. It includes a "Parameters" section with a table listing the `ticketId` parameter, which is an integer and required. Below this is a "Responses" section with a table showing a 200 OK response. The response details include a media type dropdown set to `*/*`, a note about the Accept header, and an example JSON response value.

Name	Description
<code>ticketId</code> <small>required</small>	ticketId

integer (query)

Code	Description	Links
200	OK	No links

Media type: `*/*`

Controls Accept header.

Example Value | Schema

```
{
  "transactionHash": "string",
  "transactionIndex": 0,
  "blockHash": "string",
  "blockNumber": 0,
  "cumulativeGasUsed": 0,
  "gasUsed": 0,
  "contractAddress": "string",
  "root": "string",
  "status": "string",
  "from": "string",
  "to": "string",
  "logs": [
    {
      "removed": true,
      "logIndex": 0,
    }
  ]
}
```

# Demo

## Reselling ticket

POST /ticket/approve-sale

Parameters Try it out

Name	Description
ticketId <small>required</small>	
integer <i>(query)</i>	<input type="text" value="ticketId"/>

Responses

Code	Description	Links
200	OK	No links

Media type:

Controls Accept header.

Example Value | Schema

```

{
  "transactionHash": "string",
  "transactionIndex": 0,
  "blockHash": "string",
  "blockNumber": 0,
  "cumulativeGasUsed": 0,
  "gasUsed": 0,
  "contractAddress": "string",
  "root": "string",
  "status": "string",
  "from": "string",
  "to": "string",
  "logs": [
    {
      "removed": true,
    }
  ]
}

```

---

# Q & A