# Reapware API Documentation

# Reapware API Documentation

## Overview

The **Reapware API** is a secure Node.js-based backend service that integrates with Apples APIs and Wallet services. It allows users to add credit or debit cards to Apple Wallet using Apple Pay, while handling merchant validation and secure session creation.

This API performs:

- Merchant validation using Apple Pay JS API
- Card tokenization handshake with Apples servers
- Secure communication via Apple Pay certificates
- Wallet pass generation and delivery
- Merchant domain verification compliance

## Technologies

- **Node.js + Express**
- **Apple Pay JS API**
- **Apple Merchant Validation**
- **Apple Wallet PassKit**
- **HTTPS w/ Certificates**
- **Frontend Integration (Safari only)**

## Folder Structure

```
apple-pay-reapware/
 app.js                        # Main backend logic
 package.json                  # Node.js dependencies and scripts
 public/
    .well-known/
        apple-developer-merchantid-domain-association
 certs/
    Certificate.pem            # Apple Pay merchant cert
    key.pem                    # Private key for merchant cert
    wwdr.pem                   # Apple WWDR cert
```

## Next Steps

Next, well write:
1. Endpoint documentation (`/api/validate-merchant`, `/api/add-card`, etc.)
2. Step-by-step setup for devs
3. Apple Wallet integration flow

4. Security and troubleshooting

## API Endpoints

### POST `/api/validate-merchant`

Description:
Starts an Apple Pay session by validating the merchant session with Apple.

Request Body:
```json
{
  "validationURL": "https://apple-pay-gateway.apple.com/paymentservices/startSession"
}
```

Response:
- Returns the full merchant session from Apple

Usage:
This endpoint is called from the frontend during `beginApplePaySession` to securely verify your domain.

### POST `/api/add-card`

Description:
Handles secure communication with Apple to tokenize card info and optionally prepare a Wallet pass.

Request Body:
```json
{
  "cardInfo": { ... },
  "session": "apple-session-token"
}
```

> This is a placeholder; your app would likely tokenize on the client, then confirm and store on server.

Response:
- Success confirmation or tokenized payment pass

### GET `/api/wallet-pass`

Description:
Returns a `.pkpass` file for Apple Wallet

Query Params (optional):
- `userId`
- `iban` or `kban`

Response:
- Wallet pass download (MIME type: `application/vnd.apple.pkpass`)

## Apple Wallet Integration Flow

1. **Frontend JS Integration**
   - Use Apple Pay JS to trigger `beginApplePaySession`
   - Pass validationURL to `/api/validate-merchant`

2. **Merchant Validation**
   - Server sends certs + validationURL to Apple
   - Apple returns a merchant session
   - Return to frontend to continue the Apple Pay flow

3. **Secure Tokenization**
   - Frontend sends encrypted card data
   - Backend confirms session & prepares pass

4. **Pass Download (Optional)**
   - User can download a `.pkpass` from the `/api/wallet-pass` route

## Developer Setup Guide

### Prerequisites

- Apple Developer account
- Apple Pay Merchant ID
- TLS certificate + Apple Pay certs:
  - `Certificate.pem` (Apple Pay merchant cert)
  - `key.pem` (private key)
  - `wwdr.pem` (WWDR root cert)
- Domain verification:
  - Upload `apple-developer-merchantid-domain-association` to:
    ```
    https://yourdomain.com/.well-known/apple-developer-merchantid-domain-association
    ```

### How to Run

```bash
# Install dependencies
```

```
npm install

# Start server
node app.js
```

## Security Considerations

- Always use HTTPS
- Apple Pay sessions expire validate per session
- Never store full card numbers or CVV
- Tokens should be short-lived and validated against device fingerprints if needed

---

## Troubleshooting

- `Error validating merchant`:
  - Check domain association file is reachable
  - Make sure certs are correctly formatted
- `Invalid .pkpass`:
  - Validate your pass JSON and WWDR cert chain
- `Cannot add card`:
  - Apple Pay card provisioning must be done on a real device + Safari

---

## Prerequisites

To use the Reapware API to integrate with Apple Pay, you must have the following:

### Apple Developer Requirements:
- An active **Apple Developer Account**
- A registered **Merchant ID**
- A valid **Payment Processing Certificate**
- Completed **Merchant Domain Verification**
- Access to **Apple Pay API Documentation**: [Apple Pay Developer Docs](https://developer.apple.com/documentation/passkit/apple_pay)

These are all required to securely validate sessions, sign requests, and communicate with Apples Wallet and Payment systems.