

Integration Guide

AuroPay Android and iOS Flutter Plugin

VERSION 1.0

Aurionpro Payment – Trusted partner to solve your payment problems, while you focus on your business

Personalised experience for your customers for Faster receivables

Aurionpro Solutions Limited | Synergia It Park, Plot No-R-270, Navi Mumbai, Maharashtra 400701

Table of Contents

1. Introduction 3
1.1 Prerequisites 3
1.2 Retrieve API Credentials 3
2. Integration Steps 6
2.1 Add the AuroPay Plugin 6
2.2 Initialize SDK 7
2.3 Initiate a Payment Request 9
3. Additional Resources 10
4. Need Help? 10

RECORD OF REVISIONS

UAT All Versions:

Version No.	Release Date	Features
1.0.0	19-10-2023	Base release
1.0.1	19-10-2023	Minor Bugfix
1.0.2	19-10-2023	Minor Bugfix
1.0.3	19-10-2023	Minor Bugfix
1.0.4	16-09-2024	Enable SDK support for multiple countries.
1.0.5	19-09-2024	iOS SDK improvements
1.0.6	11-10-2024	Fix the mobile number issue in android.
1.0.7	22-10-2024	Update Android and iOS SDK
1.0.8	08-01-2025	Removed full card number from convenience fee Api

PROD All Versions:

Version No.	Release Date	Changes
1.0.0	20-10-2023	Base release
1.0.1	9-10-2024	Enable SDK support for multiple countries
1.0.2	14-10-2024	Fix the mobile number issue in android.
1.0.3	23-10-2024	• Added support for multiple currencies • Performance improved.
1.0.4	13-01-2025	Removed full card number from convenience fee Api.

AuroPay Flutter Plugin Integration Guide

1. Introduction

The AuroPay Flutter Plugin enables seamless payment processing in Android and iOS Flutter applications. This guide walks you through integrating the AuroPay Flutter Plugin into your Flutter application.

1.1 Prerequisites

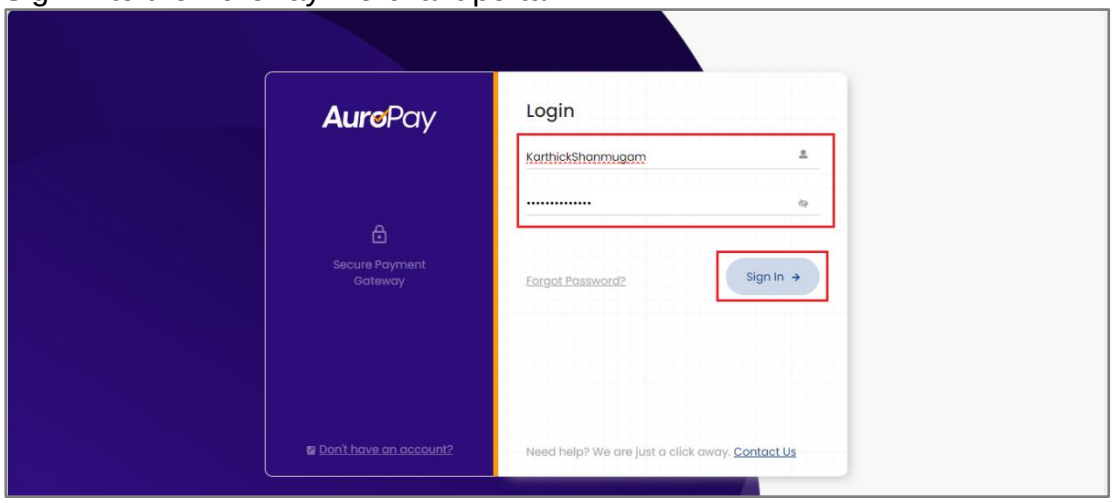
Before integrating the AuroPay SDK, ensure you have the following:

- A registered account on the AuroPay merchant portal.
- Merchant portal credentials for login.
- You need a **Subdomain ID**, **Access Key**, and **Secret Key** (available on the AuroPay Merchant Portal).

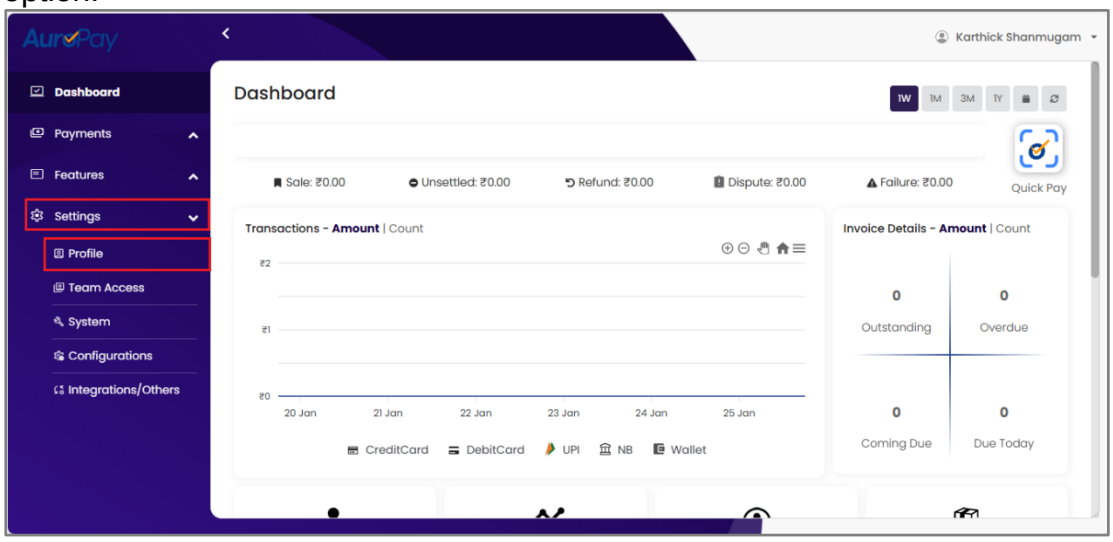
1.2 Retrieve API Credentials

To obtain the Subdomain ID, Access Key, and Secret Key, follow these steps:

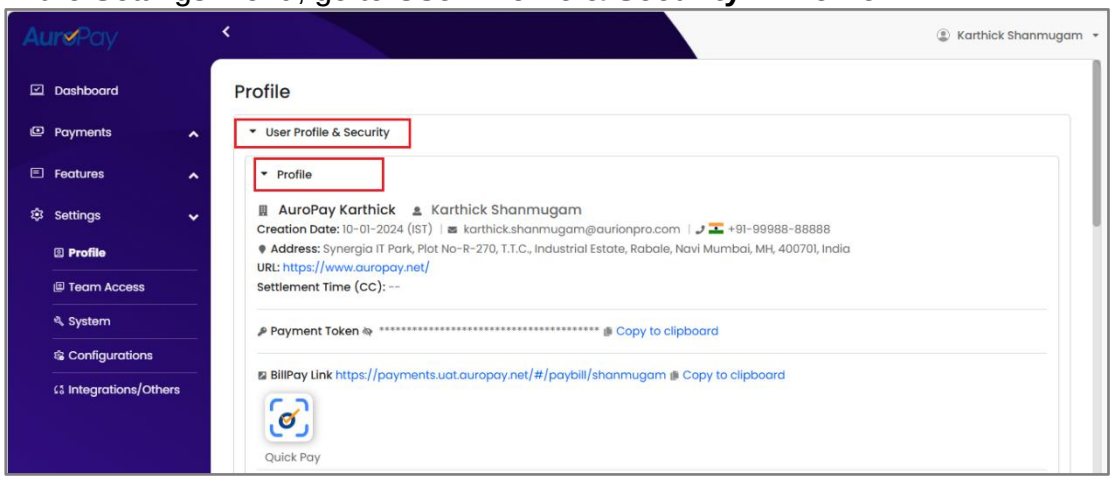
1) Sign in to the AuroPay merchant portal.



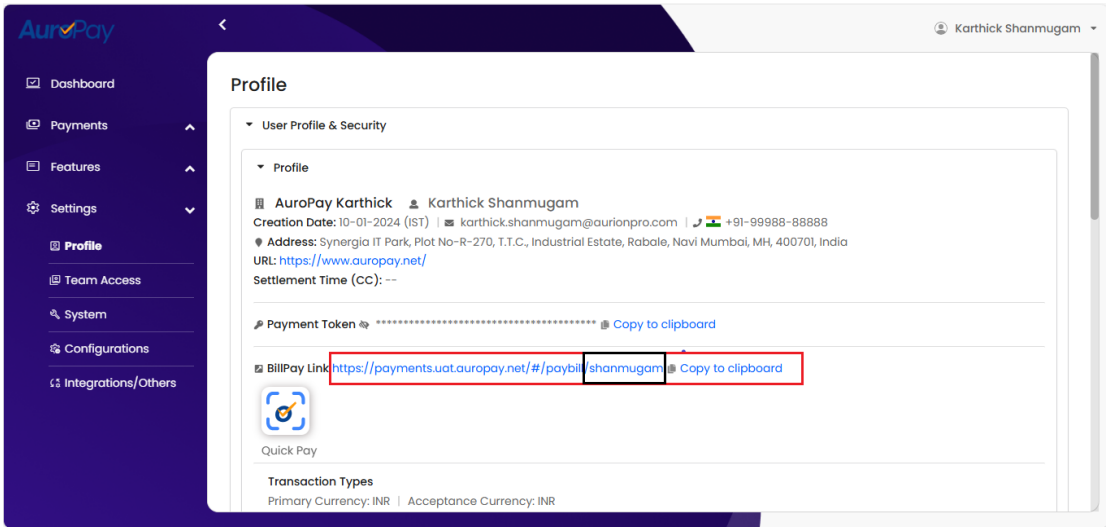
2) Within the AuroPay portal, go to the **Settings** dropdown and click the **profile** option.



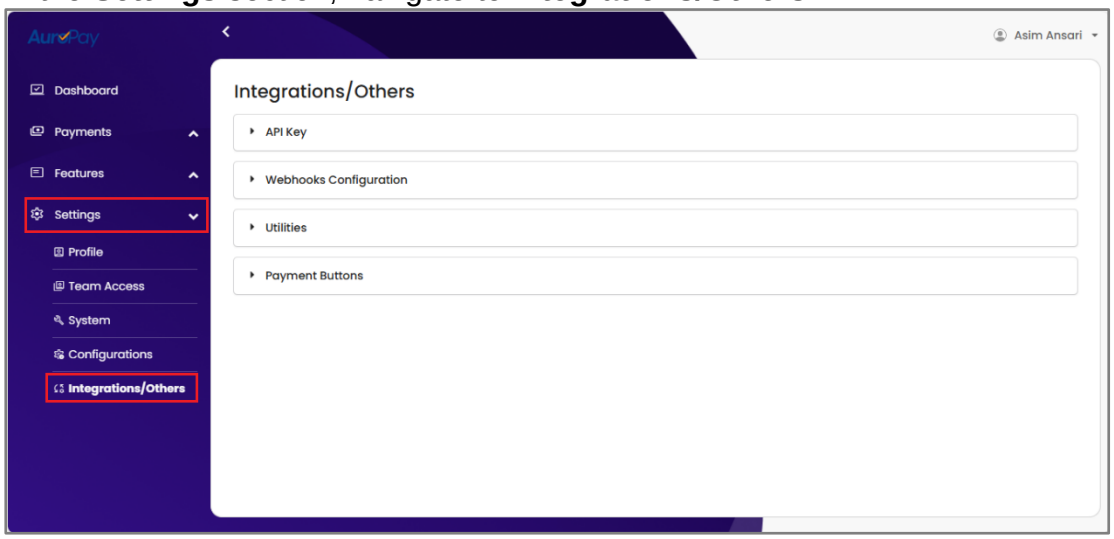
3) In the Settings menu, go to **User Profile & Security** > **Profile**.



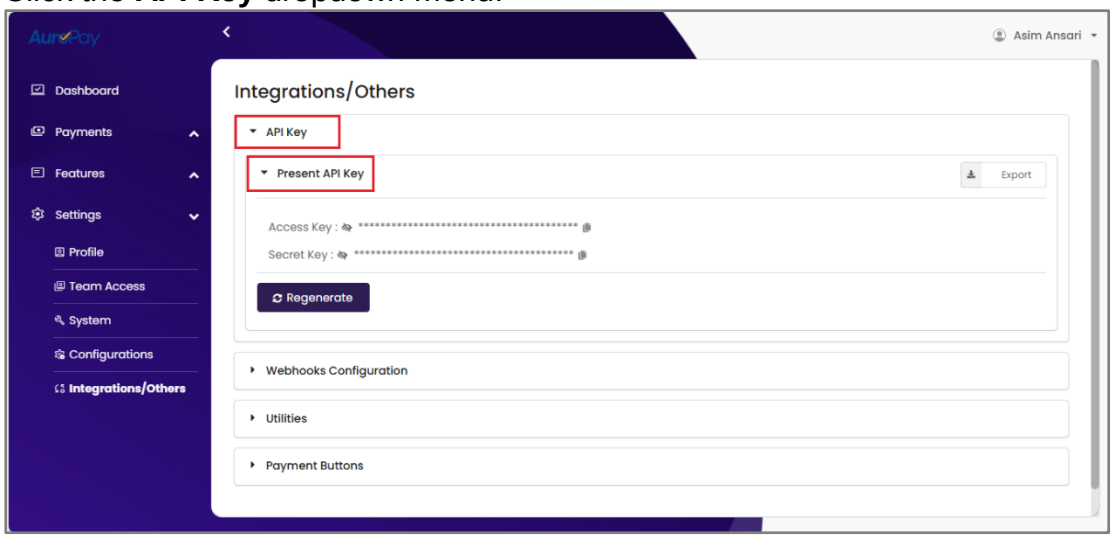
4) In the **profile** section you can view your **Subdomain ID** next to **BillPay Link** text.



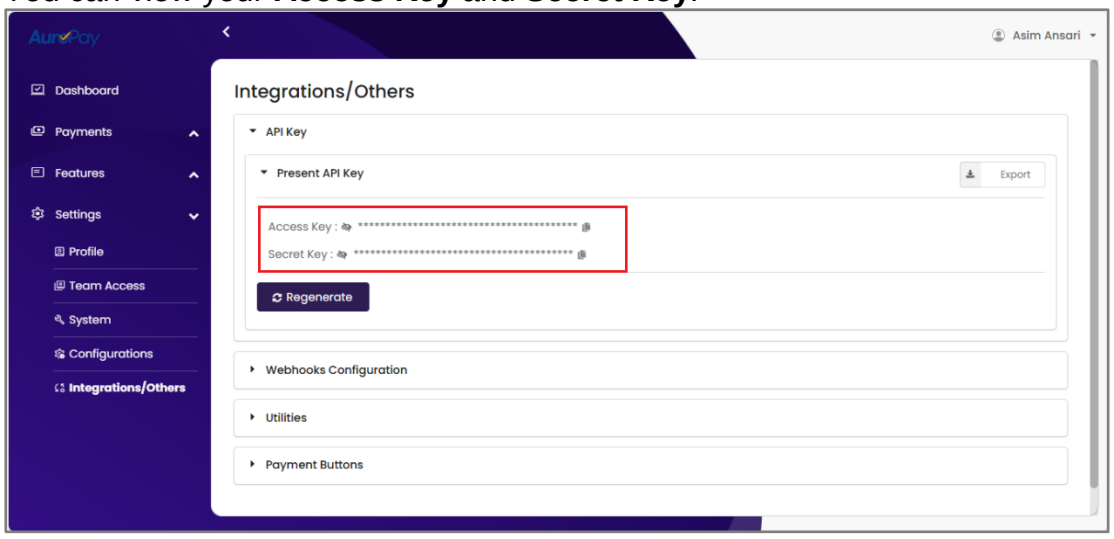
5) In the **Settings** section, navigate to **Integrations/Others**.



6) Click the **API Key** dropdown menu.



7) You can view your **Access Key** and **Secret Key**.



Note: - *Do not hardcode your API and access keys in your application. Always store them securely to protect against unauthorized access.*

2. Integration Steps

SDK Requirements

- **Minimum Android API Level:** 23 (Marshmallow).
- **iOS API Level:** iOS 13 and above.
- **Recommended Target API Level:** Latest available version.s

2.1 Add the AuroPay Plugin

AuroPay SDK is hosted on pub.dev. Follow these steps to add the SDK to your project:

Update pubspec.yaml

Add the following dependencies to your **pubspec.yaml** file:

```
auropay_payments: ^ latest version          # Production SDK
auropay_payments_sandbox: ^ latest version   # Sandbox SDK
```

Alternatively, run the following commands in the terminal:

```
flutter pub add auropay_payments         # Production SDK
flutter pub add auropay_payments_sandbox # Sandbox SDK
```

Run**:**

```
flutter pub get
```

Configure Platform-Specific Settings

For Android

Add internet permission in **AndroidManifest.xml**:

```
<uses-permission android:name="android.permission.INTERNET"/>
```

For Android Gradle Plugin **8.0 and above**, add the following in **android/settings.gradle**:

```
pluginManagement {
    buildscript {
        repositories {
            mavenCentral()
            maven {
                url = uri("https://storage.googleapis.com/r8-releases/raw")
            }
        }
        dependencies {
            classpath("com.android.tools:r8:8.2.24")
        }
    }
}
```

For iOS

Add camera permission in **Info.plist**:

```
<key>NSCameraUsageDescription</key>
<string>Needs access to your camera to scan your card.</string>
```

2.2 Initialize SDK

To process payments using the AuroPay Flutter plugin, initialize the SDK before initiating any payment request. The SDK can be initialized in **main.dart**.

To initialize the SDK, add the following lines in your screen widget:

Note: - *Never hardcode API keys in your app for security reasons. Always send them from your backend API.*

Import Dependencies

For **Production SDK**:

```
import 'package:auropay_payments/auropay_payments.dart';
import 'package:auropay_payments/models/auropay_builder.dart';
import 'package:auropay_payments/models/auropay_response.dart';
```

For **Sandbox SDK**:

```
import 'package:auropay_payments_sandbox/auropay_payments.dart';
import 'package:auropay_payments_sandbox/models/auropay_builder.dart';
import
'package:auropay_payments_sandbox/models/auropay_response.dart';
```

Initialize AuroPay

```
final _auropay = AuropayPayments();
```

Create Customer Profile

```
CustomerProfile customerProfile = CustomerProfile(
    title: "Title",
    firstName: "First Name",
    middleName: "Middle Name",
    lastName: "Last Name",
    phone: "01234567890",
    email: "Test@gmail.com"
);
```

Configure Payment Settings

```
final builder = AuropayBuilder(
    subDomainId: "Your subdomain id",
    accessKey: "Your access key",
    secretKey: "Your secret key",
    customerProfile: customerProfile
)
.setShowReceipt(true) // Default: true
.askForCustomerDetail(false) // Default: false
.getDetailedResponse(false); // Default: false

builder.build();
```

Optional: Customize the Theme

```
Theme myTheme = AuropayTheme(
    primaryColor: const Color(0xff236C6C),
    colorOnPrimary: Colors.black,
    colorOnSecondary: Colors.redAccent,
    secondaryColor: Colors.white
);
builder.setTheme(myTheme);
```

2.3 Initiate a Payment Request

Initiate a Payment Request

Use the following code snippet to initiate a payment request and receive callback:

```
auropayResponse = await auropay.doPayment(
    builder: builder,
    amount: double.parse(amountController.text),
    referenceNumber: "unique"  // Optional: null or unique
);
```

Handle Payment Response

```
final paymentStatus = ResponseType.success == auropayResponse?.type
    ? PaymentStatus.success
    : PaymentStatus.failed;

String? message;
if (auropayResponse?.data is FailureData) {
    FailureData data = auropayResponse?.data as FailureData;
    message = 'Payment Failed:\n${data.message}';
}

if (auropayResponse?.data is SuccessData) {
    SuccessData data = auropayResponse?.data as SuccessData;
    message = 'Payment Success\nTransaction ID: ${data.transactionId}';
}
```

Retrieve Detailed Payment Results

To get more detailed payment results, modify the builder configuration:

```
builder.getDetailedResponse(true); // Default: false
```

If **getDetailedResponse** is set to **true**, use the following snippet to retrieve detailed transaction data:

```
if (auropayResponse?.data is SuccessDetail) {
    SuccessDetail data = auropayResponse?.data as SuccessDetail;
    message = 'Payment Success\nTransaction ID: ${data.transactionId}';
}
```

3. Additional Resources

Pub.dev Links:

- Sandbox, Dev, UAT
- Production

GitHub Sample Implementation:

- Sandbox, Dev, UAT
- Production

4. Need Help?

If you encounter any issues, reach out to our support team at support@auropay.net.