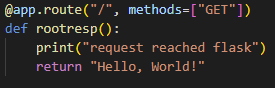
Python Payment-Backend:

We use a Python Flask, deployed in Firebase Functions, to securely handle API calls to Stripe and PayPal services and our App.

A Python Flask Function looks like this:



The private/secret Keys are only available on the Backend and not Clientside and can only be changed there.

The Backend follows the Stipe and PayPal documentation for implementation of API calls to the respective services (PayPal API documentation: https://developer.paypal.com/api/ , Stripe API documentation: <https://stripe.com/docs/api> and implementation guide: <https://stripe.com/docs> ). To call to the Stripe API we use the ‘stripe’ Package, which is available for Python, which handles the API calls while providing readability.

A Python Flask Function with a Stripe API call looks like this:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

To call to the PayPal API we make a REST API call, which is done through the ‘request’ Package from Python. A first-Party Solution over GraphQL from Braintree was not an options, as we went with Firebase to host our Front- and Backend.

A Python Flask Function with a PayPal REST API call looks like this:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Paymethodes:

The Paymethode widget is given the Project/Event Id, the type of the Id, the Donation Amount, the status and whether the donation is being made as a subscription and how long or not.

The Paymethode widget changes depending on whether the user uses the site via a web browser or the mobile app. This is detected by the ‘GetPlatform’ class:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

When viewed with a web browser the site displays a button for payments over Stripe with the label ‘Karte’ and a button for payments over PayPal. When viewed with the mobile app the site displays a for payments over Stripe with the label ‘Karte’ and a button for payments over GooglePay, which makes payments over stripe, but with the use of the users google account.



Note that when the user uses the subscribe functionality, only a button for payments over stripe is displayed.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung­

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

The OnPressed classes are also handled via GetPlatform:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

The GooglePay button is being provided, by the ‘pay’ Package by Google and is configured using the ‘paymentItems’ in the widget and the ‘default\_payment\_profile\_google\_pay.json’. The Stripe button in the mobile app is using a paymentsheet provided by stripe over the ‘flutter\_stripe’ Package. The API calls to the Payment Backend are being made with the ‘http’ Package like this:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

The response data is being returned as needed, in this case with the Checkout link, the description of the payment and the total amount in the String List ‘paypalObject’.

Depending on whether the payment was cancel or successful the return url from Stripe and Paypal directs the user to the ‘Paymethodecancel’ or ‘Paymethodesuccess’:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

When the user is redirected to ‘Paymethodecancel’ a Snackbar is displayed with a notification, that the Payment was cancelled and the user is redirected back to ‘Paymethode’, were they can try again.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

When the Payment was successful the user is being redirected to ‘Paymethodesuccess’, where they are being immediately redirected to ‘DonationReceived’.