SwypePay

Terra Softworks

May 10, 2021

USAGE AND INSTALLATION

SwypePay is an innovative, simple to use and effective digital platform that spurs the growth of business by offering the fastest, seamless payment solutions that suit all business needs.

SwypePay offers businesses E-commerce, Bulk Messaging, Payment Request, and Integrated Payments solutions all in one place.

CHAPTER

ONE

RELEVANT BACKGROUND INFORMATION AND PRE-REQUISITS

To effectively use swypepay, a user should:

- Visit our website https://www.swypepay.africa/

To further understand the code developers must:

- Be familiar with Flutter and Dart.
- Read *User Interfaces*

CHAPTER

TWO

REQUIREMENTS OVERVIEW

The **software requirements** define the system from an interfaces perspective. They are split into the following sections:

- **User Interfaces** - *User Interfaces*
- **Technical Interfaces** - *Technical Interfaces*
- **Runtime Interfaces and Constraints** - *Technical Constraints / Runtime Interface Requirements*

Contents:

2.1 Installation

Installing SwypePay into your device takes only a matter of minutes.

- First go to the App Store.
- Search SwypePay on the search bar.
- Click on install
- After installation is complete click open
- You should see SwypePay listed among your other apps

2.2 Getting started

Mobile Application

After installation simply tap on SwypePay app and follow the in-app walkthrough

Web Application

Visit our website https://www.swypepay.africa/ click on create account if you are a first time user or sign in to access your account

2.3 Context

SwypePay is a multiplatform application that has a web application and mobile application. SwypePay can be accessed and used from any of supported devices that are connected to the internet.

2.4 Conventions

We follow the coding guidelines:

Table 1: Coding Guidelines

Language	Guideline	Tools
PHP	https://www.php-fig.org/psr/psr-2/	
Dart	https://dart.dev/guides/language/effective-dart	

2.5 Architecture Constraints

2.5.1 Technical Constraints / Runtime Interface Requirements

- Applied technolologies
  - Micro-Services
- Operating systems
  - Android
  - Linux
  - Mac
- Programming Languages
  - Php V7.*
  - Dart
- Databases
  - MySQL

2.6 Technical Interfaces

Foundational Interoperability

SwypePay exchanges data with payment service providers i.e M-pesa and supported Banks

Structural Interoperability

Data is exchanged in J-son format.

Semantic Interoperability

At the highest level data is exchanged in j-son format and serves to notify all the systems of success or failure of processes such as withdraw, sending and recieving of funds

2.7 User Interfaces

The user interaces with the system in the following ways:

2.7.1 Dynamic UI Behaviour

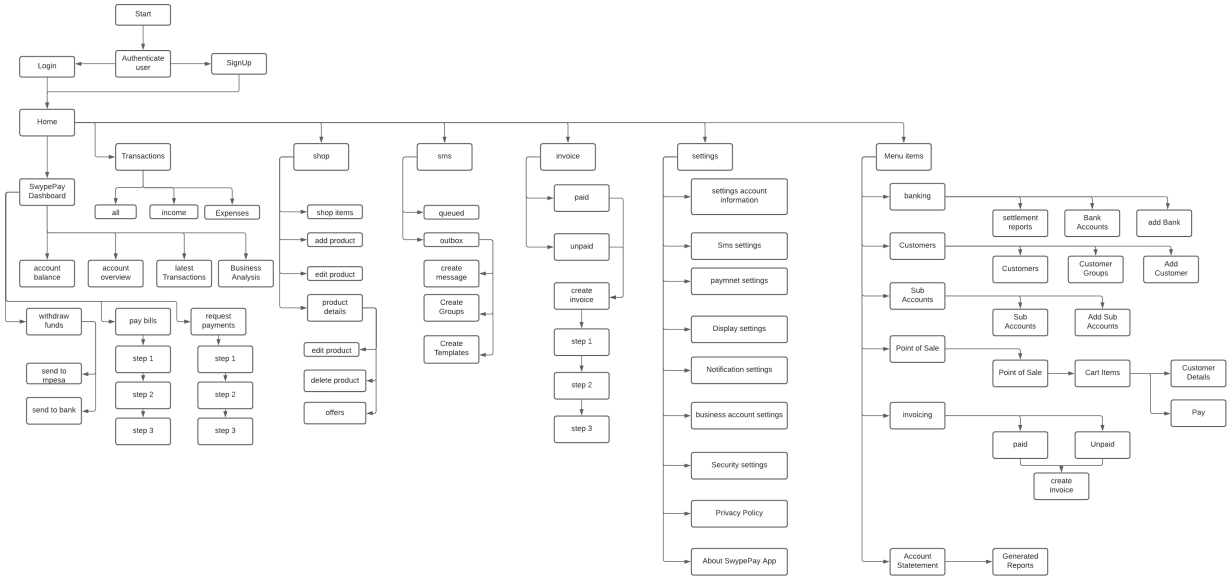

Fig. 1: Diagram showing the dynamic behaviour of the user interface.

2.7.2 Static UI

Fig. 2: Diagram showing the static ui

2.7.3 Mobile UI Flutter Widgets

Screens

- Onboarding Screen

  includes the following :

  - Text in a gesture detector on the top right that navigates to login screen.
  - Pageview that displays a column with an image and two text widgets as its child,
  - CustomButton widget at the bottom that changes the pageview index and navigates to login screen when last index is reached.

- Login Screen

  consists of a column with the following children:

  - Image of SwypePay logo
  - Two Text widgets
  - Two CustomFormField widgets to input username and passwords respectively
  - A row with a checkbox (saves users authentication details) and a text widget (navigates to ForgotScreen()) as its children
  - CustomButton to perform the login function
  - A richtext widget that navigates to sign up screen.

- Sign up

  The sign up is found in two dart files i.e sign_up_screen.dart and sign_up_step_two_screen.dart.

  - Sign_up_screen.dart

    Made up of a column with the following children:

    * CustomAppBar widget with Sign Up for SwypePay, Its free as the title.
    * Four CustomFormField widgets to input Email address, phone number, pin and pin confirmation respectively.
    * A row of two circular columns that serve as indicators.
    * A CustomButton that navigates to SignUpStepTwo.

  - Sign_up_step_two_screen.dart

    Made up of a column with the following children.

    * Two CustomFormField widgets to input business name and registration number.
    * Two CustomDropDown widgets in a row to choose the number of owners and country
    * A CustomDropDown widget to choose the business type
    * A CustomButton to perform the registration function.
    * A richtext with links to the terms of use and privacy policy.

- Forgot password screen

  made up of a column with the following children

  - A CustomLeadingIcon widget.
  - A text widget prompting the user to enter merchant code.
  - A CustomFormField widget to input code
  - A CustomButton to perform the password reset.

- OTP screen

  made up of a column with the following children

  - A CustomAppBar widget.
  - A text widget with text phone verification
  - A richtext prompting the user to enter 4-digit code and a textspan child that prompts the user to change merchant code.

  - Form with PinCodeTextField widget as a child
  - A row with a RichText (showing the number of seconds left to resend code) and An IconButton widget (perform the verification) as its children.

  The loading states of the authentication module are made up of an AlertDialog widget with a column with circular progress bar and text widget as its children.

- Sms screen

  made up of the following:

  - A scaffold with NavigationDrawer widget as its drawer.
  - A column with the following children:
  - A row with OpenDrawerIcon widget and IconButton that navigates to SearchScreen().
  - A TabBar and TabBarView that display ChatList widget.

- Invoice Screen

  consists of a Column with CustomAppBar widget a default tab controller with a TabBarView body that with PaidInvoicesTab() and UnpaidInvoicesTab() as its children. It also has a CustomFAB widget that navigates to CreateInvoiceScreen().

- UnpaidInvoicesTab

  is a column with a CustomSearchBar widget and a list of UnpaidInvoiceListItem widgets as its children.

- PaidInvoicesTab

  is a column with a a CustomSearchBar widget and a list of PaidInvoiceListItem widgets as its children.

- CreateInvoiceScreen

  is a column with CustomAppBar widget and StepOne, StepTwo and StepThree widgets as its children.

- StepOne

  is a that widget returns a column with the following children

  - A CustomFormField widget to input invoice title
  - A CustomFormField widget to input invoice description
  - A CustomFormField widget to input invoice due date
  - A row with two Columns of texts and a container as its children.
  - A Container with a row with two text widgets to display total amount
  - A CustomButton to proceed to the next Step.

- StepTwo

  returns a column with the following children

  - A CustomFormField widget to input user name
  - A CustomFormField widget to input invoice phone number.
  - A row of Text and Icon widget that calls displayAddContactSheet on onTap
  - A CustomButton to proceed to the next Step.

- StepThree

  returns a column with ClientCard, ItemsCard, TotalCard and CustomButton as its children.

- AddItemsScreen

  returns a scaffold with a column with the following children as its?s body.

  - A CustomAppBar widget,
  - A List of ItemListItem().
  - A row of Text and Icon widget that calls displayItemsBottomSheet on onTap
  - A Container with a row with two text widgets to display total amount
  - A CustomButton to proceed to the next Step.

- AddItemsBottomSheet

  shows a modalbottomSheet that returns a container with a column as a child. The children of the column are:

  - A right aligned iconbutton for closing the modal sheet
  - Five CustomFormField widgets to input item, item description, price, tax
  - A Container with a row with two text widgets to display total amount
  - A CustomButton to proceed to the next Step.

- InvoiceDetailsScreen

  is made of a column with the following children

  - Row with a CustomAppBar and an Icon in a Gesturedetector whose onTap shows CustomDropDownMenu widget.
  - ClientCard,
  - ItemsCard
  - TotalCard
  - Row with the following children
    - A column of text to display number and due date.
    - A row of containers with text as children to send reminder and Delete respectively

- AddBankScreen

  made up of a column with the following children

  - CustomAppBar
  - two CustomFormField widgets for inputing account name and account number,
  - two CustomDropDown widgets for choosing bank name and bank branch,
  - CustomFormField for inputing Mobile Till
  - CustomButton to save the details as its children.

- BankingScreen

  made up of a colum with the following children

  - CustomAppBar
  - DefaultTabController with a TabBarView Body with BankAccountTab and SettelemtReportTab as its children.

- BankAccountsTab

  made up of a Scaffold with CustomFAB as the floating action button and a Container with a ListView.builder that returns BankAccountListItem Wigdet.

- SettlementReportTab

  made up scaffold with NoData widget as its body.

- CustomersScreen

  is made up of a column with CustomAppBar and a DefaultTabController with a body of TabBarView that has CustomersTab and CustomerGroupTab as its children.

  - CustomFormField for inputing Mobile Till
  - CustomButton to save the details as its children.

- BankingScreen

  made up of a colum with the following children

  - CustomAppBar
  - DefaultTabController with a TabBarView Body with BankAccountTab and SettelemtReportTab as its children.

- customersTab

  made up of a Scaffold with CustomFAB as the floating action button and a Colum with CustomSearchBar a ListView.builder that returns BankAccountListItem Wigdet as its children

- customerGroupsTab

  made up of a Scaffold with a body of ListView.builder that returns SmsGroupListItem widget. And Container that is wrapped in an inkwell and has a row of Icon and Text as a child. The onTap method returns a Dialog with a column the following children

  - Container with a Column of two text widgets,
  - Container with a child of CustomFormField widget to input group name
  - A divider
  - A row of two Text fields wrapped in gesture detectors to create or cancel

- AddCustomerScreen

  is made a up of a column with the following children

  - CustomAppBar
  - CustomFormField to input customer name
  - CustomPhonePicker to input customer phone number
  - Row of Icon and Text
  - CustomButton

- CustomerDetailScreen

  is made up of a column with the following children

  - CustomAppBar
  - Row of two CustomFormField to input Firstname and last name.
  - CustomFormField to input email address

  - CustomDropDown to choose delivery type
  - CustomDropDown to choose location
  - A left aligned Text Widget (payment status),
  - A row of text to display subtotal,
  - A row of text to display tax
  - A Divider
  - A row of text to diplay Total
  - A CustomButton whose onPressed calls displayChargeBottomSheet().
- AccountStatementScreen

  is made up of a column with the following children
  - CustomAppBar
  - Text (Settelemnent Period)
  - Text (period)
  - Row of CustomFormField to input date range and CustomDropDown to select transaction type.
  - CustomDropDown to choose payment methods
  - CustomButton to generate reports
- AddSubAccount

  is made up of a Scaffold with CustomAppBar as the appbar and a body of a column with the following children
  - CustomFormField to input sub account name
  - CustomFormField to input account no.
  - CustomFormField to input Description.
  - CustomButton.
- GenerateAccountStatement

  is made up of a column with the following children
  - CustomAppBar
  - CustomSearchBar
  - A ListView.builder that returns GeneratedListItem.
- StatementDetail

  is made up of a column with the following children
  - Row with a CustomAppBar and an Icon in a Gesturedetector whose onTap shows CustomDropDownMenu widget.
  - Stack with the following children
    * Positioned with a container as a child
    * Positioned with a container as a child
    * Column with the following children
      · Rows of text to display sendername, beneficiary name

- Row of text to display recipient name and demo name,
- Row with children text to display debit and column of text to diplay visa and businesses.
- Row of text to display amount and currency labels
- Row of text to display amount and currency
- Row of text to display Ref No. and Order Ref labels
- Row of text to display Ref No. and Order Ref
- Left aligned text to display date label
- Left aligned text to display date.
- Container with SfBarcodeGenerator widget as child

- SubAccountScreen

  is made up of a scaffold with a body of Continer that has a list that returns SubAccounListItem as its children. A CustomAppBar widget as its appbar and CustomFAB as its floating action button. Floating action button location is centerFloat

- RequestPaymentScreen

  made up of three widgets depending on loading state, if state is loading it returns LoadingBodyScreen() if state is loading successful it returns SuccessBody() else it returns a column with the following children.

  - CustomAppBar
  - Row with three Columns with the following children
    - Circle container with an Icon as a child
    - Text to display current step
    - Text as a title of current step
  - Stepone() or StepTwo()

- StepOne

  returns a column with the following children:

  - CustomPhonePicker
  - CustomFormField to input name
  - CustomFormField to input email address
  - CustomButton to go to stepTwo

- StepTwo

  returns a column with the following children

  - CustomFormField to input amount
  - CustomFormField to input ref/invoice no
  - CustomFormField to input,
  - CustomButton to display LoadingScreen()

- NotificationsScreen

  is made up of a scaffold with a body of container with a child listview with the following children:

  - Slidable with the following secondaryActions

        * Container with a Row with the following children as its child:
        * Container with SvgPicture.string
        * Column with the following children
            · Richtext with three textspans to diplay amount received, and phone number.
            · Row of text to display date and time and ElevatedButton to pay.

- PaybillScreen is made up of three depending on loading state, if state is loading it returns LoadingBodyScreen() if state is loading successful it returns SuccessBody() else it returns a column with the following children.
  - CustomAppBar
  - Row with three Columns with the following children
    * Circle container with an Icon as a child
    * Text to display current step
    * Text as a title of current step
  - Stepone or StepTwo or StepThree
- StepOne

  returns a column with the following children
  - Container with a child Column with the following children
    * Row of Container to diplay team.png, and IconButton.
    * Text
    * Container with child row with Icon and Text (add) as its children
  - CustomFormField to input name
  - CustomPhonePicker to input recipient phone
  - CustomFormField to input email
  - CustomButton to proceed to step 2,
- StepTwo

  returns a column with the following children
  - CustomFormField to input amount
  - CustomFormField to input Ref/invoice nu
  - CustomDropDown to choose expense category?
  - CustomFormField to input description
  - CustomButton to go to step three
- StepThree

  returns a column with the following children
  - CustomFormField to input name
  - CustomPhonePicker to input number
  - CustomFormField to input email
  - CustomFormField to input amount

  - CustomFormField to input ref/invoice no
  - CustomDropDown to choose expense category.
  - CustomFormField to input description
  - CustomButton to display LoadingScreen

- Withdraw_screen.dart

  is made up of three depending on loading state, if state is loading it returns LoadingBodyScreen() if state is loading successful it returns SuccessBody() else it returns a column with the following children

  - CustomAppBar
  - Row with the following children
    - containers made with a child column with two text displaying account balance
    - containers made with a child column with two text displaying transaction fees
  - Container with child column with the following children
    - Row of Icon and Text (settlement policy)
    - Text to display policy
  - Row of text indicating current display type and to choose current type
  - Mpesa() or bank()

- Mpesa()

  returns a column with the following children

  - CustomPhonePicker
  - CustomFormField to input beneficiary name
  - CustomFormField to input amount
  - CustomFormField to input total reference
  - Row of text to display transaction charges
  - CustomButton to send to mpesa

- Bank()

  returns a column with the following children

  - CustomDropDown to choose bank
  - CustomFormField to input beneficiary name
  - CustomFormField to input to input amount
  - CustomFormField to input total reference
  - Row of text to display transaction charges
  - CustomButton to send to bank

- SettingsScreen

  is made up of a column with the following children

  - Row with OpenDrawerIcon widget as a child
  - Stack with the following children

        * CircleAvatar to display profile image
        * Icon in a Circular container positioned on the top right of the CircleAvatar
- Text (Demo account0
- Text(active)
- Listview.builder that returns SettingsListItem

SettingsScreen Scaffold has drawer of NavigationDrawer widget.

- BusinessAccountSetUpScreen

  is made up of a column with the following children
  - Row with three Columns with the following children
    * Circle container with an Icon as a child
    * Text to display current step
    * Text as a title of current step
  - Stepone(), StepTwo() or StepThree() depending on step state
- Stepone

  returns a column with the following children
  - CustomFormField to input business name
  - CustomDropDown to choose language
  - CustomDropDown to choose category
  - CustomButton to go to next step
- Steptwo

  returns a column with the following children
  - CustomFilePicker to upload registration certificate
  - CustomFilePicker to upload business permit
  - CustomFilePicker to upload tax certificate
  - CustomFilePicker to upload directors compliance certificate
  - CustomFilePicker to upload directors id
  - CustomFilePicker to upload to upload cr12 Document
  - CustomButton to go to next step
- Stepthree

  returns a Column with the following children
  - Row of checkbox and RichText with Three text Spans to agree to terms of service and Privacy Policy
  - CustomButton to continue
- DisplaySettingsScreen

  returns a column with the following children
  - CustomAppBar
  - Text

  - ListView.builder that returns CustomListItemWithSwitch widget.

- NotificationSettingsScreen

  returns a column with the following children

  - CustomAppBar
  - Text
  - Text (payments)
  - Listview.builder that returns NotificationSettingsListItem
  - Divider
  - Text (other)
  - Listview.builder that returns NotificationSettingsListItem

- PersonalInformationScreen

  returns a scaffold with a body stack with the following children

  - column with the following children
    - CustomAppBar
    - CustomFormField to input old password
    - CustomFormField to input new password
    - CustomFormField to input confirm password
  - positioned with a child CustomButton to continue

- SecuritySettingsScreen

  returns a Scaffold with appBar as CustomAppBar and a column with the following children

  - Text
  - CustomSearchBar
  - Row with following children
    - CustomDropDown to choose time
    - CustomDropDown to choose city
  - Container with child listview.builder that returns AccessLogListItem

- SmsSettingsScreen

  returns scaffold with a body a column with the following widgets

  - CustomAppBar
  - Text
  - Expanded with child defaultTabController and a body TabBarView with SmsGroupTab and SmsTemplateTab as children

  The scaffold has floatingActionButton that calls displayGroupSheet() and displayTemplateSheet() depending on the index of the tab controller.

- PaymentSettingsScreen

  returns a scaffold with a body of column with the following children

  - CustomAppBar

  - Expanded with child defaultTabController and a body TabBarView with BeneficiariesTab and PaymentGroupTab, ApproverTab as children

  The scaffold has floatingActionButton that calls displayBeneficiaryBottomSheet(), displayApproverBottomSheet and displayPaymentGroupBottomSheet() depending on the index of the tab controller.

- ApproverTab

  returns a Container with child Listview.builder that returns ApproverListItem

- PaymentGroupTab

  returns Listview.builder that returns paymentGroupListItem

- BeneficiariesTab

  returns a Container with child Listview.builder that returns BeneficiaryListItem

- SmsTemplateTab.dart

  returns a Container with child Listview.builder that returns smsTemplateListItem.

- TransactionsScreen

  returns a scaffold with a body with column with the following children

  - Row with children OpenDrawerIcon and AppBarIcon
  - Row with children Container with child Column children Text(year) and Text (date range), and CustomDropDown
  - Row with two CustomRowItems as children
  - Container with child tabbar with tabs Tab(all), Tab(income), Tab(Expense)
  - Container with child TabBarView with children TransactionList

- ShopScreen

  returns scaffold with body container with child column that has the following children

  - Row with OpenDrawerIcon and AppBarIcon as children
  - Tabbar with tabs Tab(products) and Tab(orders)
  - TabBarview with children ShopItemsTab, and onlineOrdersTab

  The scaffold also has floatingActionButton and a drawer set to NavigationDrawer widget

- ShopItemsTab

  returns a scaffold with a column body with the following children

  - CustomSearchBar
  - Row with two shadowed containers each with a child of row -with children Icon and Text.
  - ListView.builder wrapped in container that returns ShopItemsListItem
  - Text (grid format)
  - Gridview.builder wrapped in a container that returns ShopItemGridItem

- OnlineOrdersTab

  returns a Container with child column that has the following children

  - CustomSearchBar
  - Row with the following children

        * Text(sortby) centered in a container.
        * CustomContainer with child DropDownButtonHideUnderline with child DropdownButton
        * shadowedContainer with child DropDownButtonHideUnderline with child DropdownButton
        * shadowedContainer with child DropDownButtonHideUnderline with child DropdownButton
    - Listview.builder wrapped in an expanded that returns OnlineOrderListItem
- ProductDetailsScreen

  returns a scaffold with a body column with the following children
    - Stack with the following children
        * Container with child PageView.builder that returns a container with Image
        * Positioned with child CustomAppBar
        * Positioned with child container to display linear gradient
        * Positioned with child container with child _showMenu
        * Positioned with child Row wrapped in a container
    - Stack with the following children
        * Column wrapped in a container and with the following children
            · Text to display item name
            · Text to display item price
            · Row with children text to display available stock and text to display on offer.
            · Row with children text to display stock, and CustomizedSwitch
            · Text to display description label
            · Text to display description
        * Positioned with a floating action button wrapped inside an animated container.
        * _showMenu returns a Container with Column with child Visibility that has a child Listview wrapped inside an expanded.
- PointOfSaleScreen

  returns a Scaffold with body of stack that has the following children
    - Column wrapped in singlechildscrollview with the following children
        * CustomAppBar
        * CustomSearchBar
        * CustomDropDown wrapped in a container
        * Gridview.builder wrapped in a sizedbox that returns CartlistItem
    - Positioned with a child column wrapped in a container. The column has the following children
        * Row with Icon and Text (clear cart) as children
        * Row with Text (total) and Text as children
        * CustomButton to proceed to checkout

- CartItemsScreen

  returns a Scaffold with a body of stack with the following children

  - Column with the following children
    - CustomAppbar wrapped in padding
    - ListView.builder that returns ChargeListItem
  - Positioned with child column with the following children
    - Row with Icon and Text (clear cart) as children
    - Row with Two Text as children to display subtotal
    - Row with Two Text as children to display Tax
    - Row with Two Text as children to display Total
    - Row with the following children

Widgets

- dispalyAddContactSheet()

  found in add_contact_bottom_sheet.dart calls showModalBottomSheet() that returns a column wrapped in a container with the following children

  - Row of Icon and Text (Add from phone)
  - Icon wrapped in gesturedetector to close the sheet
  - CustomSearchBar
  - Row of Icon and Text (Add new contact)
  - Text(recent)
  - ListView.builder that returns ContactListItem
  - Text
  - ListView.builder that returns ContactListItem

- displayItemsBottomSheet

  found in add_items_bottom_sheet.dart calls show modal that returns column wrapped win a container with the following children

  - Icon wrapped in gesturedetector to close the sheet
  - CustomFormField to input item
  - CustomFormField to input description
  - Row with the following children
    - CustomFormField to input quantity
    - CustomFormField to input Price
  - CustomFormField to input Tax
  - Row with children Text(total) and Text (25000)
  - CustomButton

- CustomAppBar

  found in app_bar.dart returns an AppBar with title and leading of CustomLeadingIcon. It takes title as parameters

- AppBarIcon

  found in app_bar_icon.dart returns a Container with child of IconButton. It takes IconData and VoidCallback onPressed as parameters.

- CustomFilePicker

  found in custom_file_picker.dart returns a column with the following children

  - Richtext with children TextSpan(title and star)
  - Container with child row with the following children
    - Container with a centered text as child
    - Text

- CustomizedSwitch

  found in custom_switch.dart returns Switch, takes bool switchedState as parameters.

- CustomDropDown

  found in drop_down.dart returns a Column with the following children

  - Richtext with children TextSpan(title and star)
  - DropdownButtonHiddenUnderLine with child DropDownButton

  Takes String valueChosen, List ValueList, String star and String Title as parameters

- CustomButton

  found in elevated_button.dart returns a Container wrapped in a gesturedetector with a Centered Text as a child. Takes String title, voidCallback onpressed, Color colr, EdgeInsets Padding, double titleFontSize and double cornerRadius as parameters.

- CustomFAB

  found in floating_action_button.dart returns a container wrapped in an inkwell with a row of Icon and Text as a child. Takes Strings route and tagtext as parameters.

- OpenDrawerIcon

  found in floating_action-button.dart returns a Container with IconButton as child. Takes GlobalKey scaffoldKey as a parameter.

- CustomSearchBar

  found in search_bar.dart returns a Container with TextFormField as a child.

- CustomFormField

  found in text_form_field returns a Column with the following children

  - Richtext with children TextSpan(title, star and extras)
  - Container with child TextFormField

  Takes Strings title,star,hint,extras. Bools hasSuffix, isObscured, isLong and Color xtrasColor as parameters

- displayApproversBottomSheet

  found in approver_bottom_sheet.dart, calls showModalBottomSheet() which returns Container with a child column with the following children.

  - CustomFormField to input Approvers name
  - CustomFormField to input Approvers email

  - CustomPhonePicker to input phone Number
  - CustomButton

- displaybeneficiariesBottomSheet

  found in beneficiaries_bottom_sheet.dart calls showModalBottomSheet() which returns a column with the following children

  - CustomFormField to input Beneficiary name
  - CustomFormField to input Beneficiary email
  - CustomDropDown to select payment group
  - CustomPhonePicker to input phone Number
  - CustomButton

- displayChargeBottomSheet

  found in charge_bottom_sheet.dart calls showModalBootomSheet() which returns a Container with a Child column that has the following children

  - Row with the following children
    - Row of Container to display image and Text (use cash payment)
    - Icon (icons.navigate_next)
  - Divider
  - Row with the following children
    - Row of Container to display image and Text (use Mpesa payment)
    - Icon (icons.navigate_next)
  - Divider
  - Row with the following children
    - Row of Container to display image and Text (use credit card)
    - Icon (icons.navigate_next)

- ClientCard

  found in client_card.dart returns a Container with child stack with the following children

  - Container positioned at top right with child column with the following children
    - Text (name)
    - Row of Text(phone) and text(edit)
    - Text (date)
  - Container positioned to the left with a Decoration Image,
  - Text(Client) positioned to the top left
  - Icon (Icons.more_horiz) positioned to the top right.

- CustomContainer

  found in custom_container.dart returns a Container with boxDecoration,. It accepts Widget child, double cornerRadius, EdgeInsets padding as parameters.

- CustomAlertDialog

  found in custom_delete_alert_dailog.dart returns a Dialog with a Column wrapped in a container with the following children

  - Column of Text(title) and Text(body)
  - Divider
  - Row with the following children
    * Text(Delete) centered in a Container wrapped in a GestureDetector
    * Text(Cancel) centered in a Container wrapped in a GestureDetector.

2.8 Design Decisions

This can document decisions on the design of the software.

Todo: If you want to use it, keep track of decisions that someone further developing this software or using it in the future might want to know about. This might save them time or clarify expectations. You can put them as a list, or whatever.

Things to consider:

- What exactly is the challenge?
- Why is it relevant for the architecture?
- What consequences does the decision have?
- Which constraints do you have to keep in mind?
- What factors influence the decision?
- Which assumption have you made?
- How can you check those assumptions?
- Which risks are you facing?
- Which alternative options did you consider?
- How do you judge each one?
- Which alternatives are you excluding deliberately?
- Who (if not you) has decided?
- How has the decision been justified?
- When did you decide?

2.9 API

Visit http://34.66.157.228/docs/ to view SwypePay's API documentation

2.10 Solution Strategy

Todo: Describe your solution strategy.

Contents.

A short summary and explanation of the fundamental solution ideas and strategies.

Motivation.

An architecture is often based upon some key solution ideas or strategies. These ideas should be familiar to everyone involved into the architecture.

Form.

Diagrams and / or text, as appropriate. Keep it short, i.e. 1 or 2 pages at most!

2.11 Test Strategy

Todo: If you have any tests describe:

- Where the tests are kept.
- How the tests are invoked.
- What you can/need to test manually.

2.11.1 System Tests

2.11.2 Integration Tests

2.11.3 Unit Tests

2.12 Building Block View

2.12.1 Overview

Todo: Inset a building block view:

- Static decomposition of the system into building blocks and the relationships thereof.
- Description of libraries and software used
-

2.13 Runtime View

Todo: Add a runtime view of i.e. the startup of your code. This should help people understand how your code operates and where to look if something is not working.

Contents. alternative terms: - Dynamic view - Process view - Workflow view This view describes the behavior and interaction of the system?s building blocks as runtime elements (processes, tasks, activities, threads, ?).

Select interesting runtime scenarios such as:

- How are the most important use cases executed by the architectural building blocks?
- Which instances of architectural building blocks are created at runtime and how are they started, controlled, and stopped.
- How do the system?s components co-operate with external and pre-existing components?
- How is the system started (covering e.g. required start scripts, dependencies on external systems, databases, communications systems, etc.)?

  Note

  The main criterion for the choice of possible scenarios (sequences, workflows) is their **architectural relevancy**. It is **not** important to describe a large number of scenarios. You should rather document a representative selection.

  Candidates are:

  1. The top 3 ? 5 use cases
  2. System startup
  3. The system?s behavior on its most important external interfaces
  4. The system?s behavior in the most important error situations

  Motivation.

  Especially for object-oriented architectures it is not sufficient to specify the building blocks with their interfaces, but also how instances of building blocks interact during runtime.

  Form.

  Document the chosen scenarios using UML sequence, activity or communications diagrams. Enumerated lists are sometimes feasible.

  Using object diagrams you can depict snapshots of existing runtime objects as well as instantiated relationships. The UML allows to distinguish between active and passive objects.

2.14 Deployment View

Todo: Add a deployment diagram. **Contents.**

This view describes the environment within which the system is executed. It describes the geographic distribution of the system or the structure of the hardware components that execute the software. It documents workstations, processors, network topologies and channels, as well as other elements of the physical system environment.

Motivation.

Software is not much use without hardware. These models should enable the operator to properly install the software. You can separate this into different levels…

2.15 Libraries and external Software

- laravel
- Smile_identity Apis
- intl_phone_field: ^1.3.0
- flutter_bloc: ^7.0.0
- equatable: ^2.0.0
- dotted_decoration: ^2.0.0
- cupertino_icons: ^1.0.2
- circular_check_box: ^1.0.4
- pin_code_fields: ^7.0.0
- flutter_svg: ^0.21.0+1
- syncfusion_flutter_gauges: ^19.1.55
- showcaseview: ^0.1.6
- shared_preferences: ^2.0.5
- flutter_slidable: ^0.6.0
- fl_chart: ^0.35.0
- sizer: ^1.1.8
- syncfusion_flutter_barcodes: ^19.1.57
- flutter_speed_dial: ^3.0.5
- adobe_xd: ^2.0.0+1