KONY BESTBUY

AND

JLL COMPONENTS

Project report submitted in partial fulfilment of the requirement for the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

By

SAKSHAM THAKUR(151242)

UNDER THE GUIDANCE OF

Mrs. KALYANI KUCHI

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology
Waknaghat, Solan-173234, Himachal Pradesh

TABLE OF CONTENTS

DECLARATION BY THE SCHOLAR

We hereby declare that the work reported in the B-Tech thesis entitled **“Kony BestBuy and JLL Components ”** submitted at **Jaypee University of Information Technology, Waknaghat, India,** is an authentic record of my work carried out under the supervision of **MRS. KALYANI KUCHI**. We have not submitted this work elsewhere for any other degree or diploma.

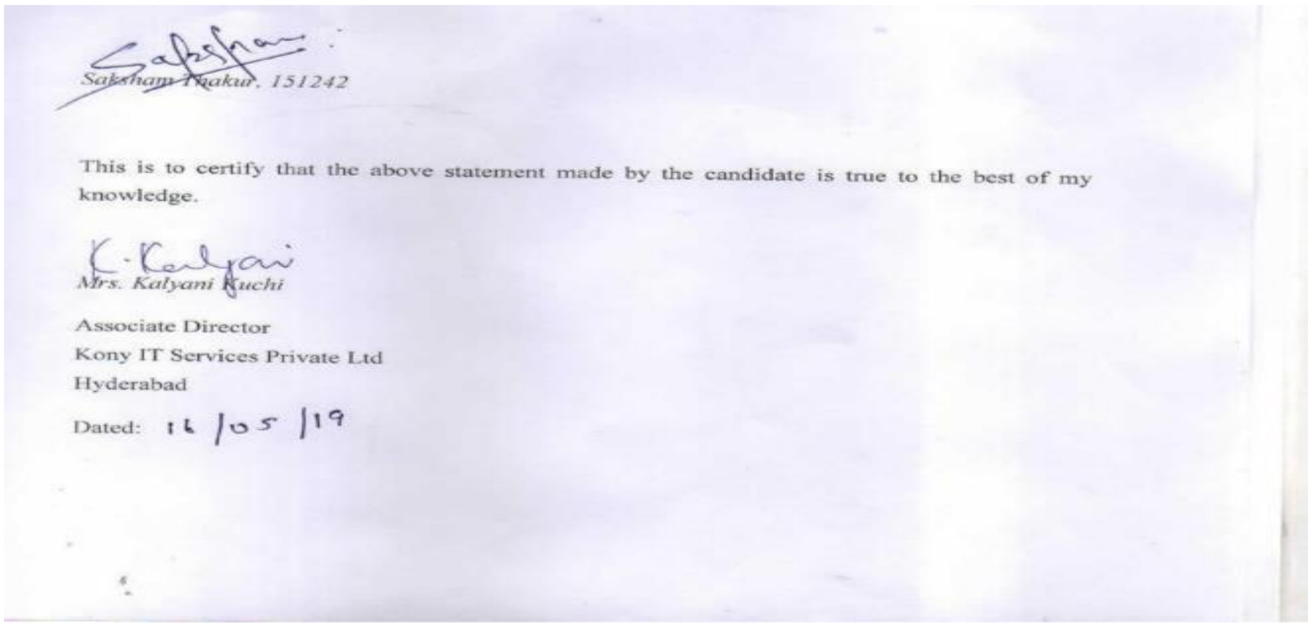
JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY

(Established by H.P. State Legislative vide Act No. 14 of 2002)
P.O. Waknaghat, Teh. Kandaghat, Distt. Solan - 173234 (H.P.) INDIA
Website: www.juit.ac.in
Phone No. (91) 01792-257999
Fax: +91-01792-245362

CERTIFICATE

I hereby declare that the work presented in this report entitled **"KONY BESTBUY AND JLL COMPONENTS"** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology, Waknaghat, is an authentic record of my own work carried out over a period from February 2019 to May 2019 under the supervision of **Mrs. Kalyani Kuchi,** Assoicate Director, Kony IT Services Private Ltd.

The matter embodied in the report has not been submitted for the award of any other degree or diploma.



[illegible]

Saksham Thakur, 151242

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

K. Kalyani

Mrs. Kalyani Kuchi

Associate Director
Kony IT Services Private Ltd
Hyderabad

Dated: 16/05/19

ACKNOWLEDGEMENT

It is a pleasure to express my deep appreciation and gratitude to my mentor and to guide Mrs. KALYANI KUCHI, Associate Director, Kony IT Services Private Ltd. Hyderabad, Telangana. Her dedication and keen interest, and primarily, his overwhelming attitude to help his team had been solely and mainly responsible for the completion of my work. Timely advice, meticulous review, scholarly advice and a scientific approach have helped me a lot.

I owe a deep sense of gratitude to all my team, respectful and supportive, who helped me to participate in the very early stages of project development and I expect them to continue to guide me constantly. Their prompt inspirations, timely suggestions with kindness, enthusiasm and dynamism allowed us to complete my thesis.

I thank very much MRS. LAKSHMI PRASANNA CHALLA, Software Engineer Kony IT Services Private Ltd., Hyderabad, Telangana, for her help and cooperation throughout my development period.

SAKSHAM THAKUR (151242)

LIST OF ABBREVIATIONS AND ACRONYMS

API	Application Program Interface
ASA	Authentication Service Agency
AUA	Authentication User Agency
CIDR	Central Identities Data Repository
CSV	Comma Separated Value
DLL	Digital Logic Library
FMR	Finger Minutiae Record
FP	Fingerprint
GDSPL	Gem Digital Security Private Limited
HSM	Hardware Security Module
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
HW	Hardware
ID	Identity
IIR	Iris Image Record
ISO	International Standards Organization
JSON	JavaScript Object Notation
MNREGA	Mahatma Gandhi National Rural Employment Guarantee Act
OTP	One Time Password/Pin
	PoS

POS	Point of Sale
RD	Registered Devices
SDK	Software Development Kit
SMS	Short Message Service
UI	User Interface
UIDAI	Unique Identification Authority of India
URL	Uniform Resource Locator
UX	User Experience
XML	Extensible Markup Language

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

This report covers two immensely crucial mobile applications which I got a chance to develop during my internship (tentatively ending in August) in Kony IT Services , Hyderabad. The first application is Best Buy Application which assists the AUAs to perform various tasks like checking status of availability of product, getting details of products, checking status in pre-production and production stages. The underlying principles behind the execution of this utility is to keep devices consistent with the protocols of product team of Kony Incorporation. The mobile application communicates with the server through http requests the APIs for which have been provided by team. The second application is JLL Components Application and as the name suggests, this application communicates with components and works towards creating a lightweight application for the users. As a future prospect, these components can also be used in other applications to ease the whole process of development and the developer could use these components directly rather than working from scratch.

CHAPTER – 1

INTRODUCTION

1.1 History of mobile applications

A versatile application or portable application is a PC program or programming application intended to keep running on a cell phone, for example, a telephone/tablet or watch. Applications were initially proposed for efficiency help, for example, Email, timetable, and contact databases, yet the open interest for applications caused fast venture into different territories, for example, versatile recreations, manufacturing plant computerization, GPS and area-based administrations, request following, and ticket buys, so that there are presently a great many applications accessible. Applications are for the most part downloaded from application circulation stages which are worked by the proprietor of the portable working framework, for example, the App Store (iOS) or Google Play Store. Some applications are free, and others have a cost, with the benefit being part between the application's maker and the dispersion stage. Portable applications regularly remain as opposed to work area applications which are intended to keep running on PCs, and web applications which keep running in versatile internet browsers as opposed to straightforwardly on the cell phone.

Now these applications have become an inseparable part of our lives and we humans cannot imagine our lives without some of these applications. Social media applications like Facebook, Instagram, Twitter and Snapchat are an internal part of our lives. All of us have a separate life on all of these and this is a life we care for a lot because it helps us to connect with the people and the world. These platforms help in forming our image in the eyes of other people. In other words, they are helping in connecting the whole humanity in a manner unforeseen. Some of the other applications like GoIbibo, MakeMyTrip etc. are helping us to organize our travel and plans with an unprecedented efficiency. Then there are other applications like Evernote, slack etc. which help us to organize ourselves and become more productive.

The general procedure of developing such an application involves learning a wide set of technologies and integrating them together to develop what we can call a good working application. This surely involves a lot of work hours and efforts of the software developer too. But the problem doesn’t just end here. The real pain hides in the fact that application must be developed differently for the various platforms available. This is with respect to the difference in the inherent nature of how the various operating systems have been developed.

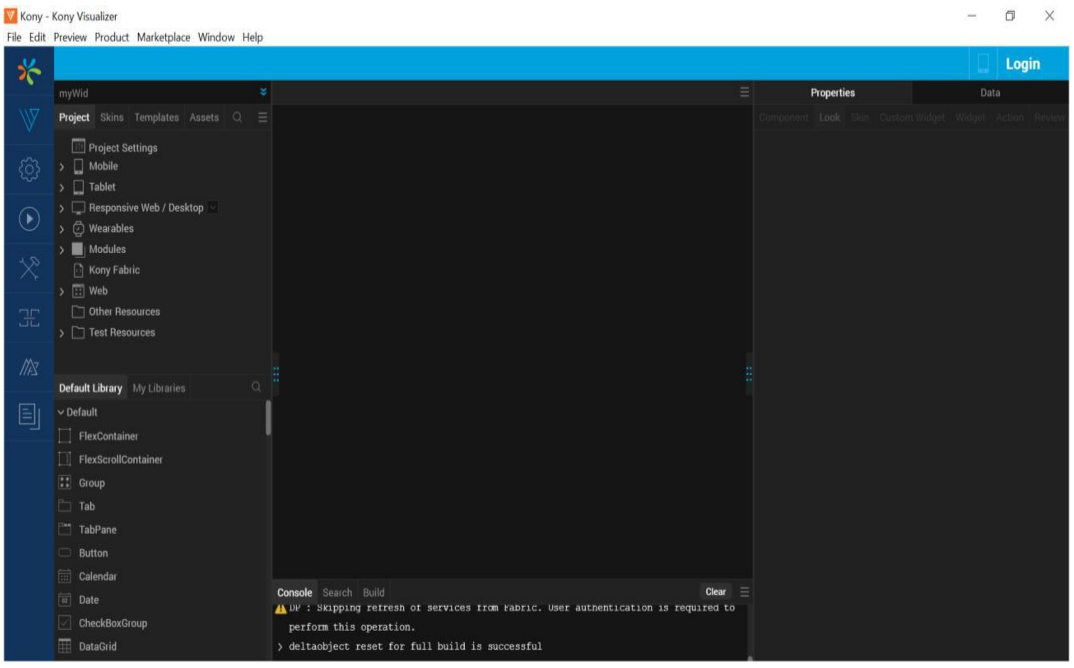
Android operating system was built on the top of Linux kernel. The methodology of building an android application generally involves learning the android studio along with the Java technology stack. Moreover, all the UI/UX (User Interface/User Experience) stuff needs to be defined using XML (Extensible Markup Language). XML is extremely lightweight which helps in loading the application swiftly. Android is an Operating System handled and operated by the big five company Google. Over the years, the process of developing an Android application has transitioned to Kotlin rather than Java. The features like better functional programming style approach, easier maintenance etc. make it a better language and developers are expected to learn the whole technology stack just to develop a small application.

The story with the second most famous mobile operating system IOS is not much different. Developed over BSD authorized UNIX based operating system it makes the process of application development task even more tough. Developers are expected to learn cumbersome languages like objective C and swift. The size of the development platform makes the whole process even more hectic.

1.2 Need of Kony Platform

This was just when we were talking about running the application on the mobile platform. What about the devices like desktops, laptops, tables and smart watches? Let's not talk about learning the whole confusing JavaScript technology stack along with the whole plethora of languages and technologies that need to be learned and practiced gaining the ability to develop such applications. Having established how complex the whole development task has become, I want to talk about the client requirement.

Suppose, there is a new startup with a great idea and wants to reach maximum amount of people. Hence, it decides to create an application for all the platforms that are available in the market i.e. mobile, desktop, tablets et cetera. Now, what needs to be done? It needs to hire different developers who are going to develop the application corresponding to one specific platform. They could also hire a developer who is skilled enough to develop the application on all the platforms and develops the app for all platforms. The first methodology is going to cost a lot more and a single developer might take a very long time to complete the allotted task. So, we need a better solution here.

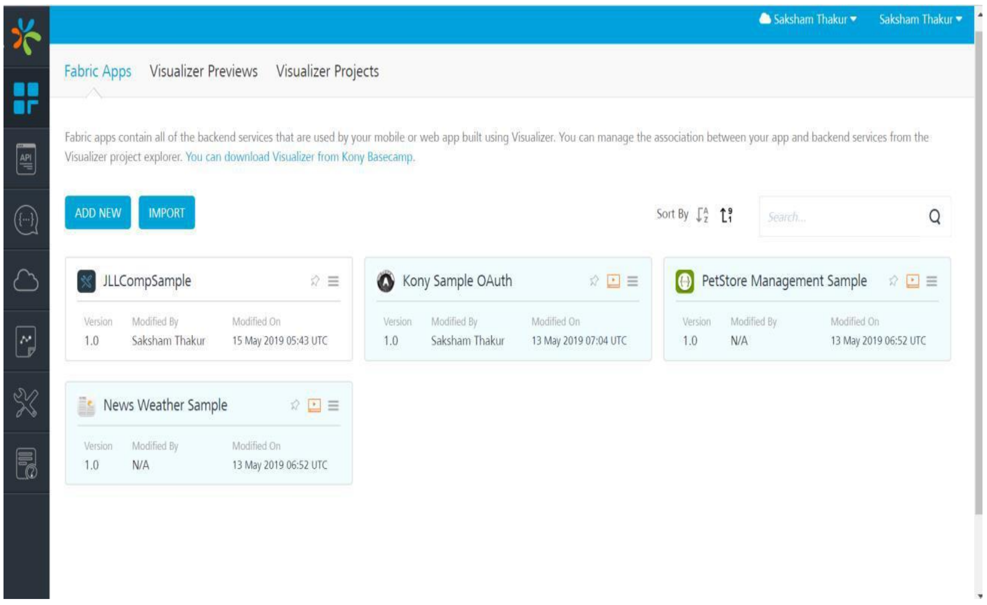


```
⚠ DP : Skipping refresh of services from Fabric. User authentication is required to
  perform this operation.
> deltaobject reset for full build is successful
```

Figure 1.1 : Kony Visualizer View

1.3 Brief Overview of Kony Platform

This is where Kony comes into the picture. Founded in 2007 in Texas, United States of America Kony INC. is a company which works towards creating a unified platform. This platform is omnichannel clubbed with the cloud platform. Omnichannel is a cross-channel content methodology that associations use to improve their client experience. As opposed to working in parallel, correspondence channels and their supporting assets are planned and organized to collaborate. Omnichannel infers coordination and arrangement of channels with the end goal that the experience of connecting over every one of the channels somebody utilizes is as, or significantly progressively, productive or wonderful than utilizing single diverts in confinement. In other words, Kony is a platform that can be used to develop applications for all the platforms necessary. You just need to learn a single technology stack and there you go. This involves some Java and a lot of Vanilla JavaScript. All of this is done in the Kony Visualizer platform. Along with this we need to learn a middleware platform called Kony Fabric. This is easy to use and can be installed in your local machine.



Fabric apps contain all of the backend services that are used by your mobile or web app built using Visualizer. You can manage the association between your app and backend services from the Visualizer project explorer. You can download Visualizer from Kony Basecamp.

JLLCompSample
Version 1.0 · Modified By Saksham Thakur · Modified On 15 May 2019 05:43 UTC

Kony Sample OAuth
Version 1.0 · Modified By Saksham Thakur · Modified On 13 May 2019 07:04 UTC

PetStore Management Sample
Version 1.0 · Modified By N/A · Modified On 13 May 2019 06:52 UTC

News Weather Sample
Version 1.0 · Modified By N/A · Modified On 13 May 2019 06:52 UTC

Figure 1.2: Kony Fabric View

Some of the other Kony INC. products have also dominated the market. The Kony DBX has been dubbed as "The future of online banking". This product is a generalized product for the various banking platforms which can be edited and adjusted to fit to the customer objectives. Kony Quantum is a product of different nature with tagline "Low code without limits". Basically, this product enables even the amateurs to create applications using minimal code.

During the internship period we were first trained briefly for a period of about one month. At the end of this training we worked on developing an application which integrated the Best Buy api. The application basically revolved around the various types of products an E-commerce website offers and dealt with displaying their detailed images, reviews, price and availability.

After working on this I was deployed into the JLL project where I worked towards creating a lot of components. Components is the part of the project which is common to the whole project. I worked on the UI and functionality of the various components and even worked on the Fabric side to manipulate the data we were getting.

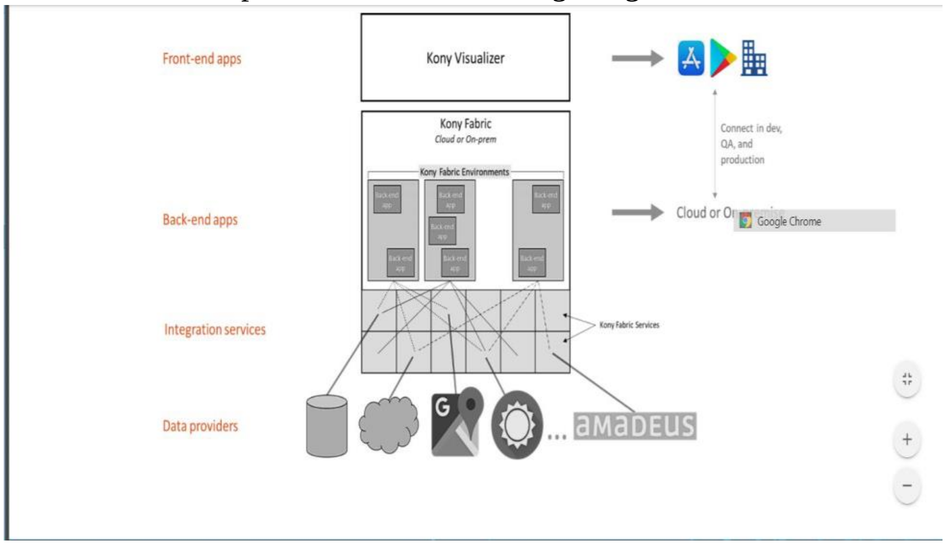


Figure 1.3: Logical Diagram of Kony Visualizer, Fabric and Backend

CHAPTER – 2

LITERATURE SURVEY

For the development of these applications , the basic understanding of the various application programming interface and the methodologies required to develop an application and other minute details have been gained from the following documents.

2.1 Kony Documentation

Documentation is a lot of records given on paper, or on the web, or on computerized or simple media, for example, sound tape or CDs. Models are client guides, white papers, on-line help, brisk reference guides. It is winding up less basic to see paper (printed version) documentation. Documentation is appropriated by means of sites, programming items, and other on-line applications.

This is a set of documents which is regularly updated by the product team with every new release of the Kony visualizer version, Kony fabric version or Kony DBX version. The documentation for Kony Visualizer documentation is hosted at https://basecamp.kony.com/s/quantum-documentation?index=0. This document contains wide variety of information which includes information from installation guide of the visualizer to the api guide which is the necessary tool in the process of development of any application on the platform.

Similarly, for the other products like Kony DBX documentation is hosted on the https://konydocs.atlassian.net/wiki/spaces/ARBD41/pages/935691289/Digital+Banking+Solution+Overview website. This documentation hosts information on Introduction, DBX App Suite, Digital Banking Platform, Data Storage, Open Banking and PSD2, Open Integration, UI/UX and Data Model Design, Extensibility and Customization, Enterprise Security and GDPR Readiness. Moreover, information about Digital Banking Platform is also there. The Kony Digital Banking Platform gives integral assets to help the full life-cycle of use improvement, testing, organization and upkeep. Kony Visualizer empowers low code improvement and the capacity to effortlessly share and use regular parts, incorporating those gave the Digital Banking Suite, through both open and private Marketplaces. Visualizer is utilized to both expand the standard suite and create custom applications. Kony Fabric gives the versatile middleware runtime, reconciliation administrations, outsider programming connectors, Identity and API the board.

2.2 Kony Boot Camp

This is the go-to website for all the doubts and the problems that a developer face. This is an inquiry and answer site for expert and aficionado software engineers. The domain is https://basecamp.kony.com/s/topic/0TO6A0000001I9ZWAU/developer-bootcamp?.

CHAPTER – 3

SYSTEM DEVELOPMENT

3.1 Best Buy Service

This was an application I developed as part of the final training project. This application revolved around the various products that a particular e-commerce website has to offer , their availability and the reviews that the customers had given.

3.1.1 Front End Features

3.1.1.1 Forms

Forms in Kony application are the biggest part that host all the parts of a UI component. They are equivalent to window in the typical web applications.

Three main forms :

1. Category Form: This contains information on the various categories and sub categories corresponding to a particular product.
2. Products Form: This lists the information on all the products that are present under a particular category.
3. Product Detail Form: This lists the detailed information on a particular product and also the various reviews that customers have given.

Common features in all forms:

Header: This is a component which can be seen in all the forms. Header consists of a back button which is a button that can be used to navigate to the previous screen. Then there is an image corresponding to the website name. At last there is a search feature in the header which enables user to search directly for a desired product and navigates him to the relevant page and loads the data.

TECHNOLOGIES USED

LANGUAGE: Javascript

VERSION: ECMASCRIPT 7

IDE: Eclipse integrated with Kony Visualizer

Middleware: Kony Fabric

API Used:

Product-
https://bestbuyapis.github.io/api-documentation/#products-api

Category-
https://bestbuyapis.github.io/api-documentation/#categories-api

Figure 3.1 : Brief Outlook

- A Form has a life cycle method that gets called when a particular event is triggered.
- Form Life Cycle Methods :
  1. Init
  2. preShow

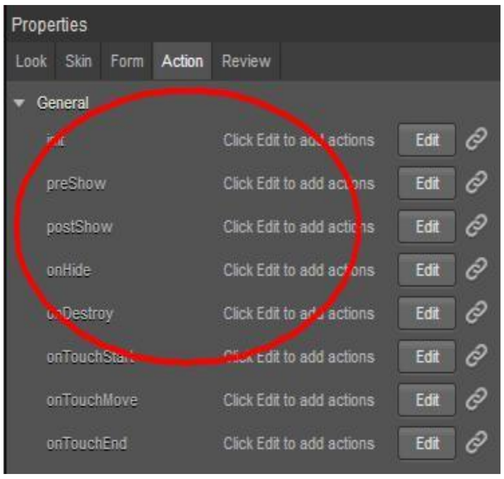


Figure 3.2: Events in visualizer

3. postShow
4. onHide
5. onDestroy

- The form lifecycle begins when we:
  1. Access a form
  2. Access a widget on a form
  3. Show a form

- Init
  1. This event is invoked only once in the form's life cycle, when the form is ready with the widgets hierarchy
  2. This is called after the addWidgets method, and is a good place to put code to initially set up the form – you'll have access to all the widgets
  3. When form is destroyed and reused again, init gets called as a part of form's lifecycle

- preShow
  1. This event is triggered just before the form is visible on the screen
  2. This event is triggered every time the form is shown whether through code, the device back button, etc.
  3. This is a good place to put code to prepare the screen for viewing

- postShow
  1. This event is triggered every time after the form is rendered on the device's screen
  2. This is a good place to put code that does things like make service calls to get data to display since postShow is not holding up the UI thread like preShow
  3. Can be used to display other forms (e.g. interstitial screen) during service calls
- onHide
  1. This event is called when the form goes out of view because some other form is displayed
  2. TonDestroy
  3. his event is called when the form and it's controller is removed from memory
  4. You can call this using API kony.application.destroyForm("frmHello");

3.1.1.2 Details of each form

Category Form: Along with the header component this page displays the whole path to a particular category or subcategory. This is displayed on a label widget.

The rest of the form contains a segment widget. This widget is used when a lot of data is to be displayed repetitively. Each row of the segment has an onRowClick event attached to it which actually triggers the form and shows the subcategories corresponding to a category.

The data that is being fetched is using the bestbuy category api. The endpoint corresponding to the api is at https://bestbuyapis.github.io/api-documentation/#categories-api. We just send a get request to this api and it returns us with the xml and fabric handles it and structures it to the required JSON format which can be handled easily. All the requests that are sent are via the fabric only. We create an MF application and associate this fabric application with our visualizer application. Now in JS code we invoke the fabric function.

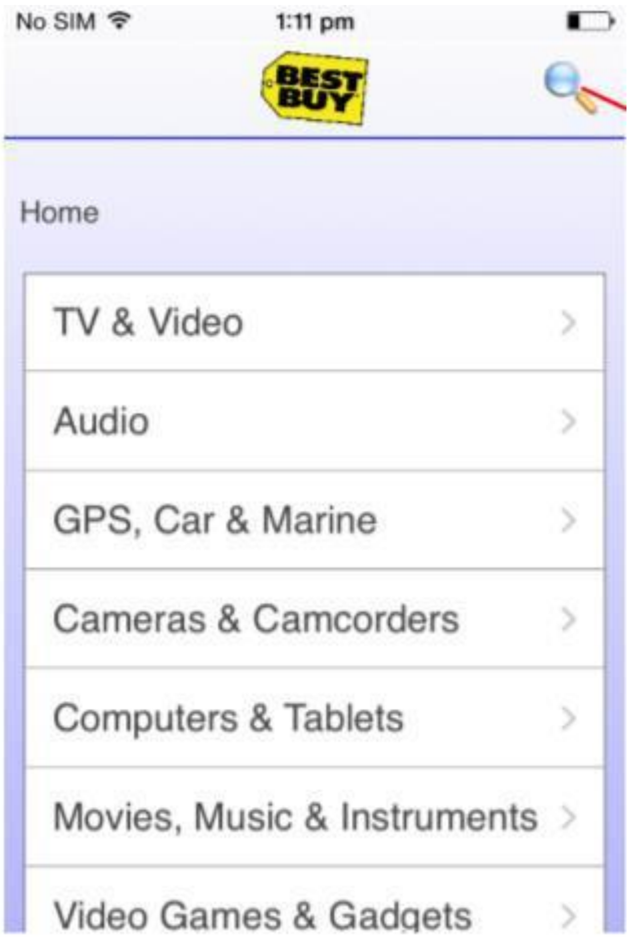
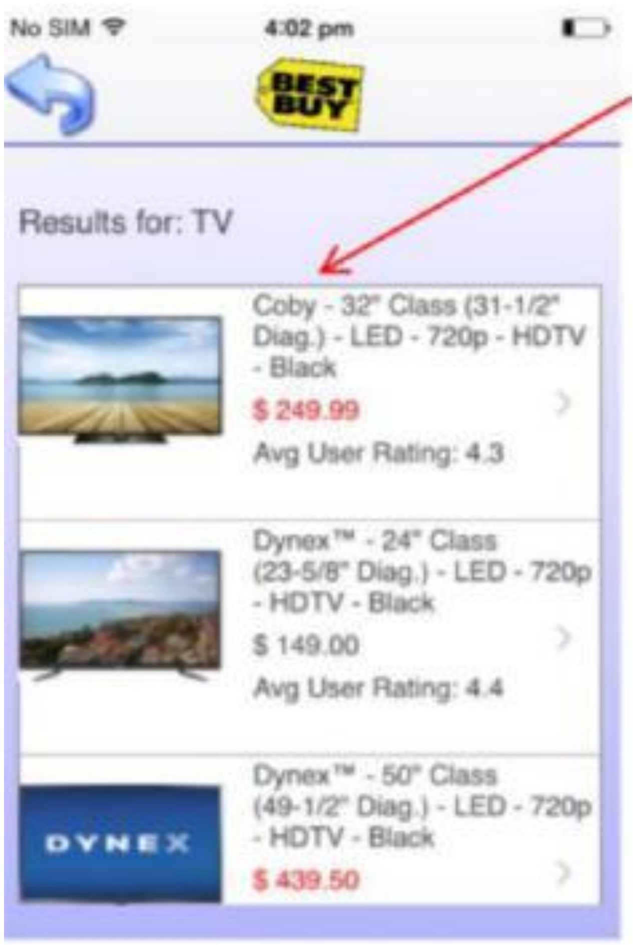


Figure 3.3: Category form

Products Form: This form again contains the header component. Along with this it consists of a segment which shows a brief outline about a particular product. Now, the data is to be displayed in different formats in the rows of this segment. This is done via the use of templates. Basically we define a particular design inside a template and later add the data to each row of the segment. This is done by modifying the JSON response that we get from the fabric.

Now the data binding in between the JSON response and the elements of the segments and templates has to be defined explicitly using the widgetDataMap function that is provided under the segment widget api. For templates declaring the name on similar lines helps in binding the data efficiently.



Results for: TV

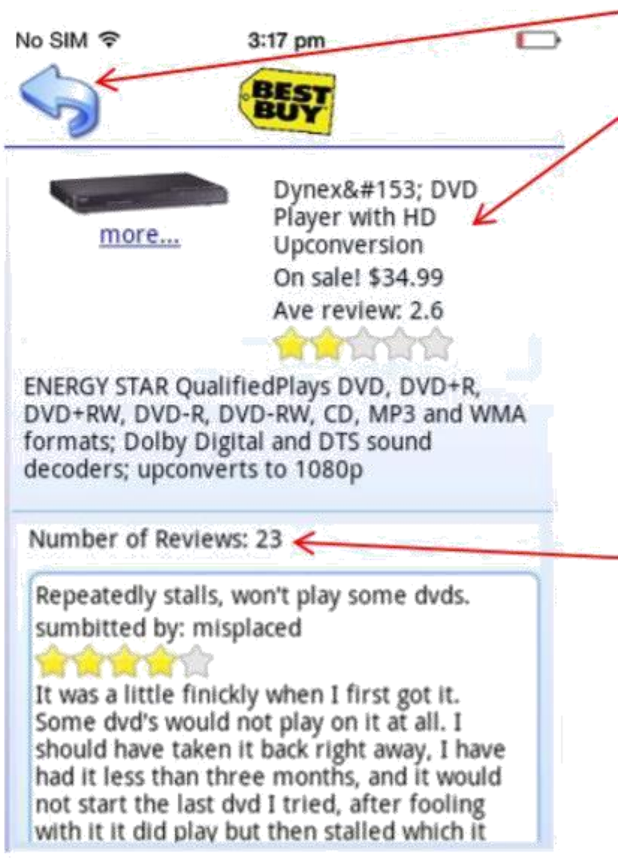
Coby - 32" Class (31-1/2" Diag.) - LED - 720p - HDTV - Black
$ 249.99
Avg User Rating: 4.3

Dynex™ - 24" Class (23-5/8" Diag.) - LED - 720p - HDTV - Black
$ 149.00
Avg User Rating: 4.4

Dynex™ - 50" Class (49-1/2" Diag.) - LED - 720p - HDTV - Black
$ 439.50

Figure 3.4: Product Page

Product List Form: This page features the details about a product, lists its specifications in detail and shows the reviews. The product details are shown in a template which also holds the average rating, sale price. Again there is a segment widget which holds the information about a particular review.


No SIM 3:17 pm

more...

Dynex&#153; DVD Player with HD Upconversion

On sale! $34.99

Ave review: 2.6

ENERGY STAR QualifiedPlays DVD, DVD+R, DVD+RW, DVD-R, DVD-RW, CD, MP3 and WMA formats; Dolby Digital and DTS sound decoders; upconverts to 1080p

Number of Reviews: 23

Repeatedly stalls, won't play some dvds.
sumbitted by: misplaced

It was a little finickly when I first got it. Some dvd's would not play on it at all. I should have taken it back right away, I have had it less than three months, and it would not start the last dvd I tried, after fooling with it it did play but then stalled which it

Figure 3.5:Product Detail form

The various endpoints to get the data are:

Get categories:
https://api.bestbuy.com/v1/categories(id=cat00000)?apiKey=YourAPIKey

Get products for a category:
https://api.bestbuy.com/v1/products(categoryPath.id=abcat0902001)?apiKey=YourAPIKey&page=1

Get products by search text: :
http://api.bestbuy.com/v1/products(search=%22hdmi+cable%22)?apiKey=YourAPIKey&page=1

Get product details: :
http://api.bestbuy.com/v1/products(productId=1218088854917)?apiKey=YourAPIKey

Get user reviews for a product:
http://api.bestbuy.com/v1/reviews(sku=1000671)?apiKey=YourAPIKey

You need to register (quick and free) to get an API key that needs to be passed to all API calls: https://developer.bestbuy.com/

3.1.2 Middleware Services

The following are the six major services offered by Kony Fabric:

1. Identity - who can access the app and use it
2. Integration - accessing backend data
3. Orchestration - combining integration services
4. Synchronization - managing offline data
5. Engagement - incorporating push messaging
6. Reporting - measuring app usage and user info

You can create "apps" in Kony Fabric where you can have one or more of each of those feature.On the client, in the application, your code can consume the features of one or more "apps" hosted by Kony Fabric

Think of a Kony Fabric "app" as a logical grouping of features – for example, you might have a Facebook "app" that has features for logging in and accessing Facebook data through web services.

We will use Identity and Integration services now.

3.1.2.1 Identity service

Kony Fabric character administrations help you secure your application by including an authentication layer.

You can set up a personality administration dependent on the sort of the clients who are permitted to get to your application. To limit access to your organization's inner gathering of people, utilize Microsoft Active Directory confirmation. To enable access of your application to a bigger crowd, you can utilize endeavor personality suppliers (Microsoft Active Directory, Kony SAP Gateway, Open LDAP, OAuth 2.0, Salesforce, Custom Identity Service, SAML, Siteminder) and social character suppliers (Google, LinkedIn, Instagram, Amazon, Microsoft, Yahoo, BOX, Facebook.).

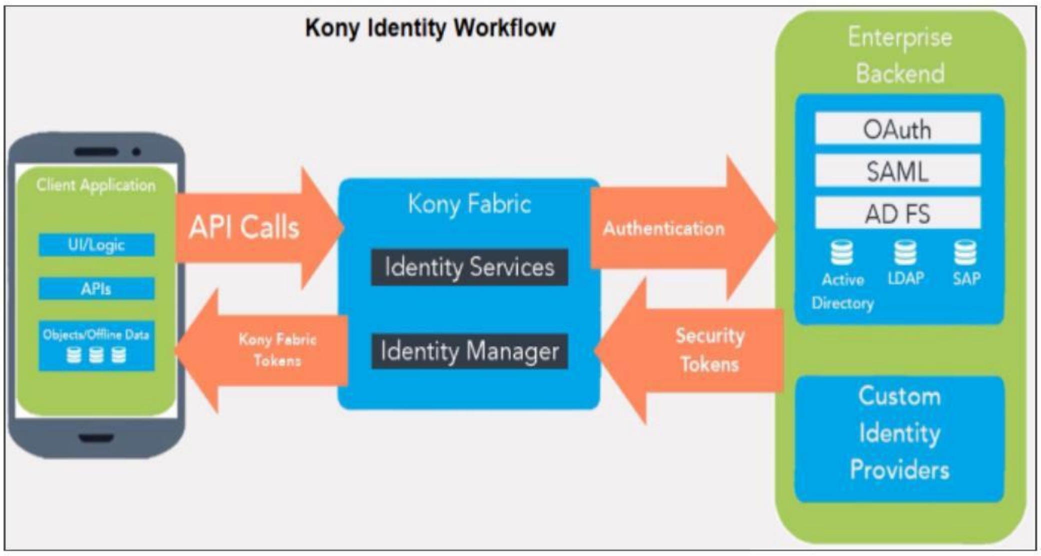
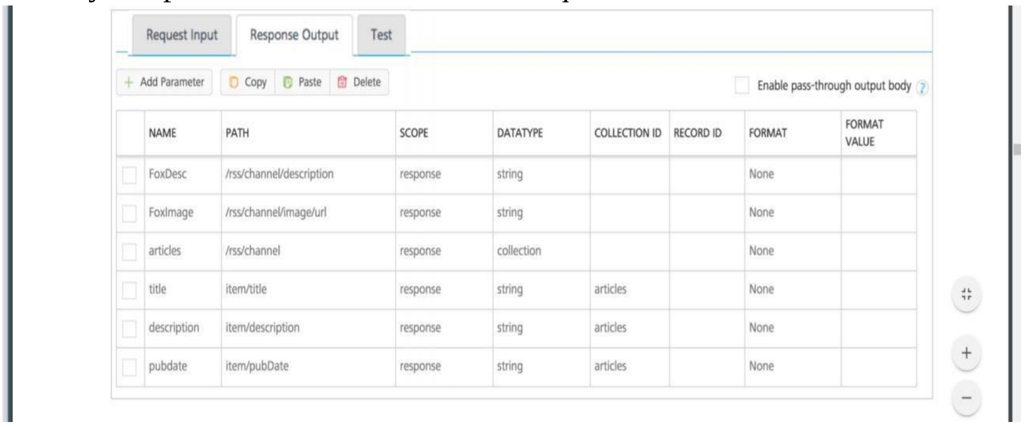


Figure 3.6: Kony Fabric View

3.1.2.2 Integration service

An Integration Service is an application segment that speaks to the application communication with an outer framework or information source. An administration definition involves the meta-information, or the designs required to trade information with the outer framework or information source. For instance, the arrangements can be administration type, Endpoint URL, administration ID, type (HTTP/HTTPS), demand parameters, reaction parameters, preprocessors and postprocessors, and validation accreditations whenever required.

We need to mention the api endpoint here. Here we can also structure the output result according to the manner we want. This requires some knowledge of xpath to actually scrape the data in the format we require.



NAME	PATH	SCOPE	DATATYPE	COLLECTION ID	RECORD ID	FORMAT	FORMAT VALUE
FoxDesc	/rss/channel/description	response	string			None	
FoxImage	/rss/channel/image/url	response	string			None	
articles	/rss/channel	response	collection			None	
title	item/title	response	string	articles		None	
description	item/description	response	string	articles		None	
pubdate	item/pubDate	response	string	articles		None	

Figure 3.7: Integration Service View

The result is stored in an array and is structured according to the way we mention in response output.

For the best buy application, we created an application that used XML response output as mentioned above.

After creating this application, we need to publish this application. This generates some keys that we use to integrate this fabric application in our visualizer application.

Moving the Integration Service to the middleware permits the developer the following advantages:

• When the information comes back to Kony Fabric, it's structured to such an extent that ONLY the fields you need on the visualizer side are sent back to the application.

• You can perform information pre-and post-processing on the server instead of on the visualizer application

• You can store information in Kony Fabric's client reserve for use in other Integration Service calls - no reason to deal with that information on the customer application

• You can process returned information before sending to the customer application

• You can make a vigorous library of designed Integration Services for use in more than one application

• Fixes/changes/upgrades can be made in Kony Fabric without influencing your versatile application - clients don't need to transfer another application on the grounds that the backend administration changed

• Kony Fabric uncovered these Integration Services through a standard API to improve the versatile application code

Using the SDK and integrating with Fabric

Visualizer on the left-hand side menu has an option which can be used to integrate the Fabric application.

1. Right Click on the Kony Fabric option and click on use existing application option.
2. Now list of all the applications currently on the Kony Fabric will be listed.
3. Associate the application you want to choose and now the visualizer and Fabric application have been linked.

3.1.2.3 API in different forms

3.1.2.3.1 Animation API

This api is used to animate the objects. This includes movements like translation, rotation, scaling or transformation which involves the combination of any of the former three.

For animation we generally follow the given procedure:

1.Create a transform object using kony.ui.makeAffineTransform().
2.This object can be next configured according to the way or according to the animation we require.
3.We define the object configurations on a scale of 0 to 100, where 0 corresponds to the position of the object at 0 time and 100 denoting the step when duration of animation is complete. For this we use kony.ui.createAnimation and mention the various states inside this as the parameter of this function.
4.Next, we need to define the duration of the animation and the mode which we are going to use for our animation.
Now for attaching the animation to any widget we just mention the widget path and use the animate function with this. This function takes the three parameters animation Object, animationConfig, animation Callbacks as input.

3.1.2.3.2 Loading Screen API

This involves showing the loading screen, the message corresponding to the loading screen and dismissing the loading screen. Two main methods are adept at dealing with this.

1.kony.model.ApplicationContext.showLoadingScreen(message)- This is used to show the loading screen. The sole parameter deals with the message that is to be displayed.

kony.model.ApplicationContext.dismissLoadingScreen()- This is used to dismiss the loading screen.

2.kony.application.showLoadingScreen(skin,text,position,blocked,show ProgressIndicator,properties)- Another methodology to display the loading screen.
Parameter:
Skin- The skin that is to be applied to the loading screen. Text- Message that we want to show on the loading screen. Position- The position where we want to place the loading screen. Options include center and full screen mode.
Blocked- If other UI elements will work when the loading screen is shown.
ShowProgressIndicator: If we want to show the progress screen indicator.

kony.application.dismissLoadingScreen():This is used to dismiss the loading screen.

3.1.2.3.3 API's to communicate with fabric

The first step deals with getting the integration service object. For this we use kony.sdk.getCurrentInstance().getIntegrationService(serviceName). This procedure is followed when we already associated our fabric application with the visualizer application. Now, we use this object to call a particular operation.

integrationObj.invokeOperation(operationName,headers,data,this.operation Success, this.operationFailure) is the function which we use to call the operation corresponding to the integration service.

Parameters:

operationName- This specifies the name of the operation which we intend to call.

Data: This includes the additional data that we give or require to call the api.

operationSuccess: This function is called if the operation is executed successfully.

operationFailure: This function is called if the operation faces some failure in its execution.

3.1.2.3.4 Mapping API

This is used to map the output in the response object to the widget we want to. This may include mapping the text output to a particular widget and so on. We generally just mention the widget path and use the widgetDataMap function with this to map the output.

Example: this.view.segOther.widgetDataMap={lblData:”name”};

3.1.2.3.5 Navigation API

This API helps us in navigating from one form to another form. The general procedure is as follows:

1.Creating a navigation object. This is done using the kony.mvc api. General format is- new kony.mvc.Navigation(formName)
2.Using the navigation object to actually navigate to the form we want.
General format is- navigationObject.navigate(data). Data consists of the additional data we want to send to the other page.

Now, the form to which we are navigating must contain the onNavigate function. This function handles and defines the actions that are to be performed when the form is loaded into the application.

3.1.2.3.6 OS API

This API deals with the system level events and is useful in getting system related information.

Kony.os.userAgent()- This gives us the information like phone name,IMEI number etc.

Kony.os.freeMemory()- This gives us information about the amount of free memory that is available.

Kony.os.deviceInfo()- This gives us information like the height , width of the screen and so on.

3.1.2.3.7 Segment API

This API is used to handle the data in the segment.

The general api’s used are:

1.setData(object): This is used to set the data of the segment. The parameter given in the input contains the data that the segment is set to.
2.setDataAt(object): This is used to set the data of a particular row in the segment to a particular value.
3.removeAll(): This enables us to remove all the existing rows and sections in a segment.

The api also gives some important events that can be used to customize the application and meet the client’s need. These events generally turn out to be very important and include:

1.selectedRowIndex: This returns the index of the row that is clicked. We generally use this to change the data or to trigger a particular event that generates important results.
2.onRowClick: This is an event triggered every time we click a particular row in the segment.
3.selectedRowItems: This returns the contents of the row that has been clicked by the user.

3.2 JLL Components

This is the project I worked on after completing the training. This client of the application is a hotel chain owner who required an application specifically for the purpose of booking the rooms, making reservation, booking the workstations, getting feedback, fixing work order, ordering food and getting the general information.

3.2.1 Forms

The application consists of several forms which are loaded on the basis of the requirement.

The first form loaded is **frmSplash**. To mention the first form we need to right click on the form we want to set as the first form and click on set as startup form option.

Now splash screen is the screen containing the adept UI and functionality corresponding to the time when the application is being loaded.

This form just contains the image in the background, but is focal from a functionality point of view. In its postshow event the function for the initialization of fabric application is written. This directly helps in attaching the fabric application to the visualizer application. Once this is done it calls the setWeatherData function. This function again calls the weatherDetails function and the operation getCurrentWeather. This gives us the details about the current weather which we are displaying in our main application. Moreover, it also returns the data about the stock price of the share of the company.

In case when all of this is executed successfully the page is navigated further to **frmLogin**. But in case of failure several things occur. If failure occurs some kind of network error the required message is shown.

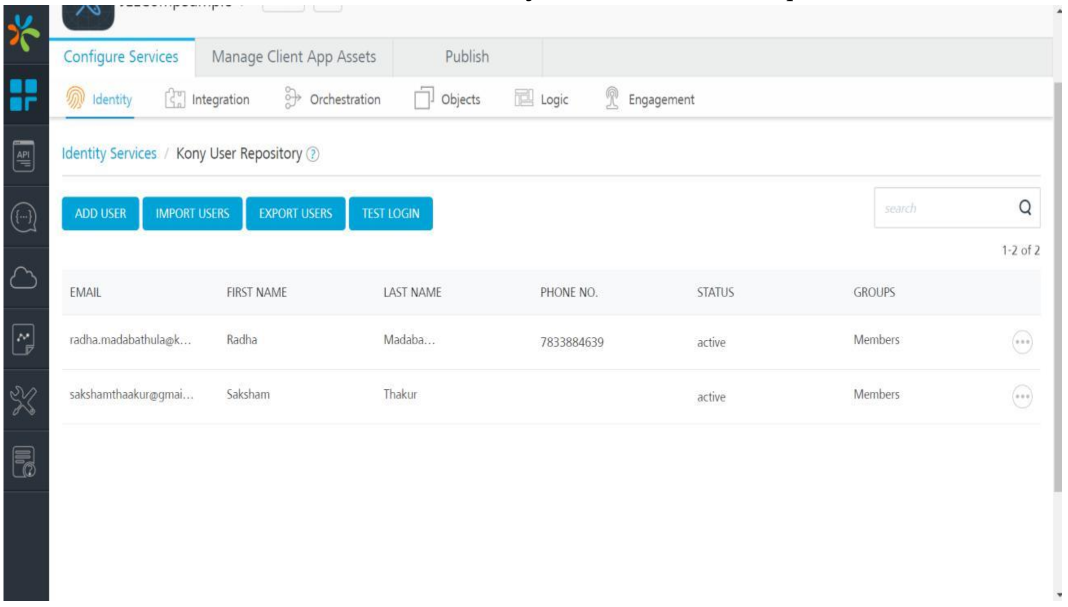
Figure 3.8 :Frm Splash

The important global information required by all the forms is stored in the globalVariables.js file. This file contains the following variables:
1.Authentication variables for connecting to fabric:

gblMyAppKey- The application specific key required to connect to the application.
gblMyAppSecret- The secret key required to connect to the fabric application and service

gblMyService_Url- The URL at which the fabric URL is up and running.

Now there are some of the objects that need to be accessed on all the forms. For this purpose we store them in this global file. Important concept revolves around this. The file structure is such that all the files containing the code that should be global are stored in the **Module** folder. Similarly, we also have a require folder.



EMAIL	FIRST NAME	LAST NAME	PHONE NO.	STATUS	GROUPS
radha.madabathula@k...	Radha	Madaba...	7833884639	active	Members
sakshamthaakur@gmai...	Saksham	Thakur		active	Members

Figure 3.9 :Identity Service

The concept of require folder revolves around making the application lightweight. Generally, whenever the application is launched all the files of the module folder are also loaded into the memory making the overall application heavyweight. The performance of the application suffers a lot due to this. Hence, we have the concept of require.This revolves around loading a particular file into the application only when it is needed. That is we just use the syntax **require(filename)** and the file loads correspondingly into the memory. This helps in improving the performance to a heavy extent. The concept is a groundbreaking improvement in the concept of modular programming and has served to make the applications and web community on a humongous scale.

2. gblKonyObject- This object stores the current instance of the fabric application that has been associated to the visualizer application.

3. gblLoginUserObj: This holds the detailed information about the user who has logged into the application.

4. gblHomeIconsObj: Holds the information on the icons that are to be displayed on the different forms.

5. gblStockWeatherObj: This object holds the return value of the weather and stock service and is useful in displaying the corresponding information on the different forms.

6. gblDirectoryData: This is useful in the security and helpdesk module and provides information on the goto people and the people to whom a particular problem should be communicated to.

Now, from flmSplash we are navigated to frmLogin. This form as the name suggests is responsible for logging in the user. The application has a component that has been imported into the application with name **com.logincomponent.login** . There are two modes to login into this application. The first is login as Employee and the second is login as Guest.

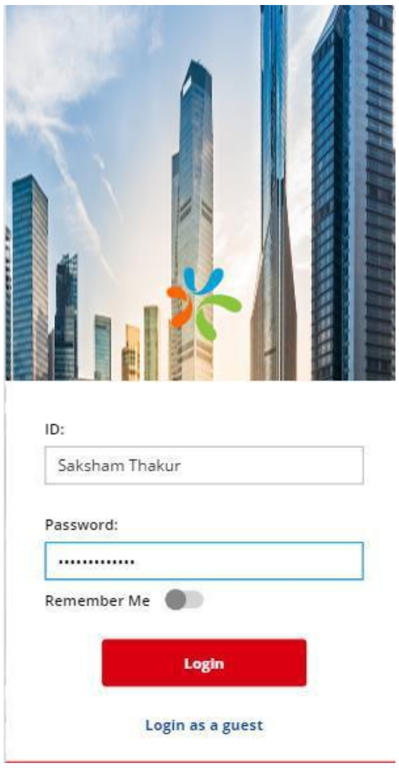


Figure 3.10 :Login Screen

For the login functionality we use the general **user repository** authentication method. This service operates in the basic manner. We just register the user, and the user later logs into the website using the email and password that he has provided at the time of registration. This is for the employee login. For the guest login feature we just ask for the guest Username and guest Email and use these for login. When the guest logs in using these we set the data correspondingly in the memory as this is also required later to display the information. For this purpose we use the kony.store api.

Kony.store API explained:

kony.store.setItem(key,value): This api stores the data in the memory as a key, value pair and uses the java hasmap data structure to store data. kony.store.removeItem(key): This is used to delete or remove the data from the memory.

kony.store.clear(): This api removes all the data from the memory , i.e. it clears the whole java hasmap .

kony.store.getItem(keyname): This gets a particular item from the memory on the basis of the keyname that has been mentioned

kony.store.key(index): This returns the data stored in the memory on the basis of the index that is mentioned in the parameter.

kony.store.length(): This returns the total number of the key, value pairs that have been stored.

The onNavigate function is triggered when we navigate to this form. This function calls the loadUI and bindEvents functions. The loadUI function works on the UI part and assigns the necessary skins to the required widgets. The bindEvents function assigns the callbacks. These are the function which are executed when an another function which is expected to take time completes. onClickGuestLogin and onClickLogin are the functions here which act as the callbacks to the two different login functionality.

Bulk of the functionality associated has been defined in the Js file associated with the component. The constructor of this file contains the association of the various events to the widgets. There are several getters and setters which assign the necessary Variables with the value that is required. The other functions Include switchRememberMe which is responsible in hiding the Remember me option in the guest Login view. clearPrefillData is another function used to clear the input fields when we visit the login page. switchGuestAndEmployee function calls the different functions which as a whole are responsible for making the form seem like the customer login view or the guest login view. isnullOrEmpty function checks if any of the input field has not been filled. guestLogin function is triggered when the guest tries to login. No identity service is required for this login and the if everything happens correctly triggers the onGuestLogin function of the original form controller. employeeLogin is the function used for logging the employee. This uses the userstore identity service and calls the login function. If everything happens successfully then identityLoginSucces is called which has some additional functionality. If the remember me option was chosen at the time of logging it, the input fields are stored in the kony store. If failure occurs then it triggers the identityLoginFailure function. This function uses another component This is the popup component and is used to display a message as a popup. Here we display Login Failure message. The validateLoginFields function is used to check up on some of the constraints that the login fields have to follow. This include minimum length of password, userId and null field constraints. onClickClose is another function which is associated with the popup component and is used to change the visibility of the component.

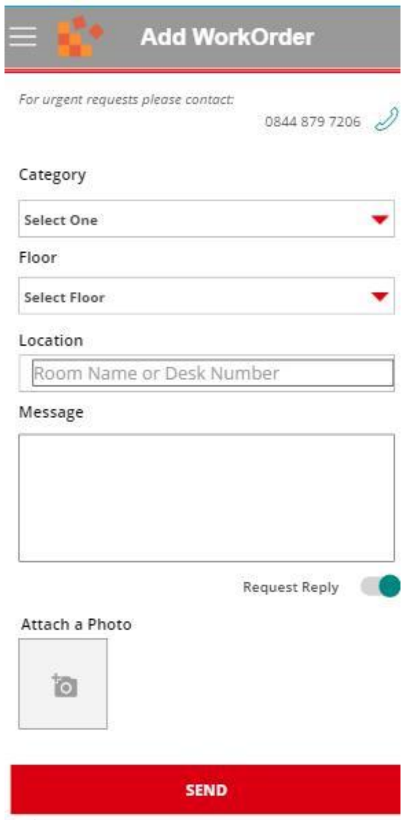
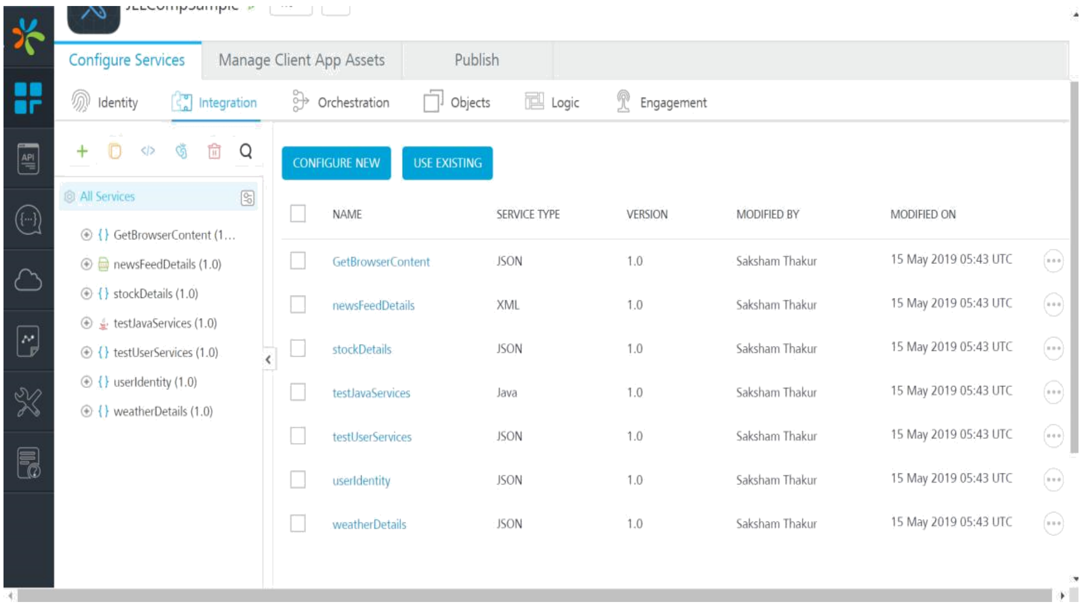


Fig 3.11:FrmWorkOr

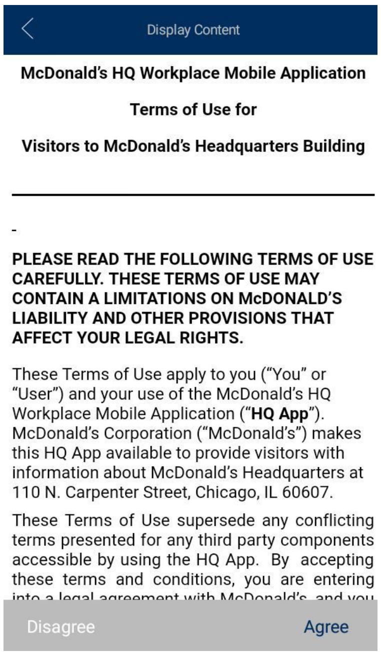
Additional function include the hideLoading and showLoading functions which are used to show the loading screen when the login request is made. Similarly, the hideLoading method is used to hide the loading screen.



NAME	SERVICE TYPE	VERSION	MODIFIED BY	MODIFIED ON
GetBrowserContent	JSON	1.0	Saksham Thakur	15 May 2019 05:43 UTC
newsFeedDetails	XML	1.0	Saksham Thakur	15 May 2019 05:43 UTC
stockDetails	JSON	1.0	Saksham Thakur	15 May 2019 05:43 UTC
testJavaServices	Java	1.0	Saksham Thakur	15 May 2019 05:43 UTC
testUserServices	JSON	1.0	Saksham Thakur	15 May 2019 05:43 UTC
userIdentity	JSON	1.0	Saksham Thakur	15 May 2019 05:43 UTC
weatherDetails	JSON	1.0	Saksham Thakur	15 May 2019 05:43 UTC

Figure 3.12 : Various Service

Now, if this login occurs successfully the user is redirected to the form **frmTandC** which stands for form Terms and conditions. This shows the basic terms that the customer has to agree before using the application.



McDonald's HQ Workplace Mobile Application

Terms of Use for

Visitors to McDonald's Headquarters Building

-

PLEASE READ THE FOLLOWING TERMS OF USE CAREFULLY. THESE TERMS OF USE MAY CONTAIN A LIMITATIONS ON McDONALD'S LIABILITY AND OTHER PROVISIONS THAT AFFECT YOUR LEGAL RIGHTS.

These Terms of Use apply to you ("You" or "User") and your use of the McDonald's HQ Workplace Mobile Application ("**HQ App**"). McDonald's Corporation ("McDonald's") makes this HQ App available to provide visitors with information about McDonald's Headquarters at 110 N. Carpenter Street, Chicago, IL 60607.

These Terms of Use supersede any conflicting terms presented for any third party components accessible by using the HQ App. By accepting these terms and conditions, you are entering into a legal agreement with McDonald's and you

Figure 3.13 : Form Terms And Conditions

Here we have a browser widget. Browser widget in Kony can be used in any form and it acts as a typical browser does. We can load its content from a local HTML file or from can hit a URL for the data that it displays. Here also we have different services for both of the methods above.

displayContent method checks what the value of the variable _sourceType is. If this variable is set to HTML Content then we call the function displayHTMLContent() and if

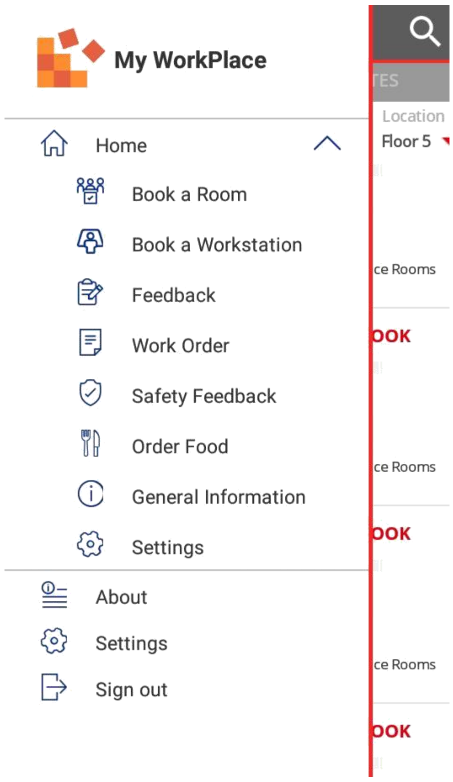
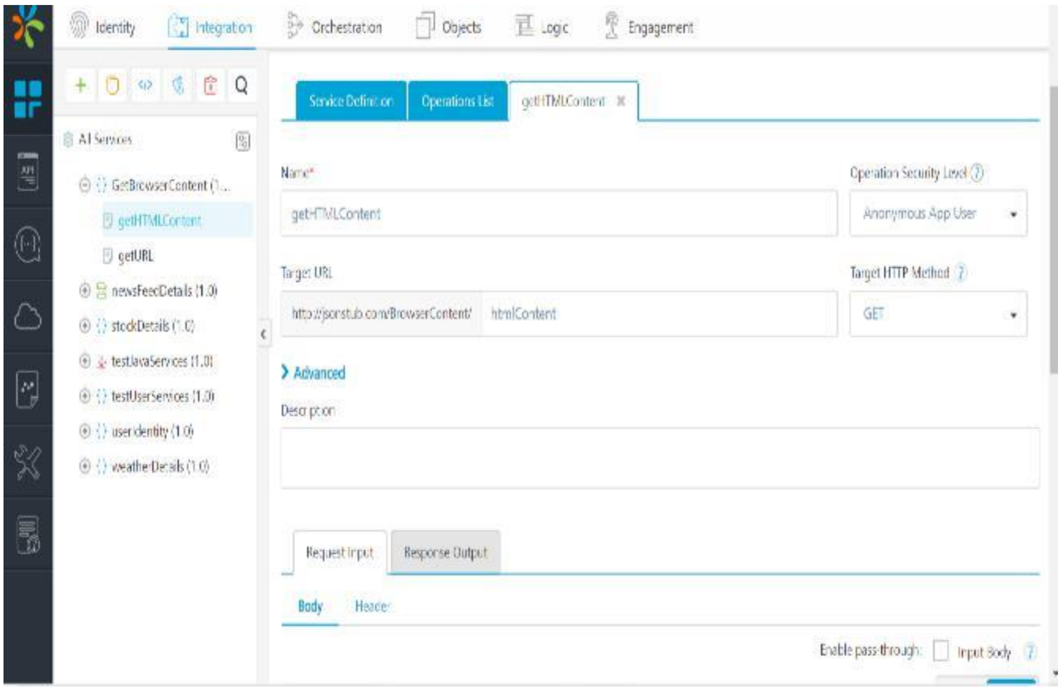


Figure 3.14 : Hamburger Component

it is set to URL then displayHTMLContent function is called. displayHTMLContent function uses an integration service. GetBrowserContent is the service name and getHTMLContent is the operation corresponding to this. The successCallback of this service sets the data in the browser widget to the returned response.

Similarly if the _sourceType is URL then the function displayContentFromURL is called. This function also triggers an operation in the integration service. The servicename is GetBrowserContent and the operation name is getURL. The success and failure callbacks have a similar nature.



Name*: getHTMLContent

Operation Security Level: Anonymous App User

Target URL: http://jsonstub.com/BrowserContent/ htmlContent

Target HTTP Method: GET

Description:

Enable pass-through: ☐ Input Body

Figure 3.15 : Integration Service

Now, the function navigateToHomeScreen is called. If the cutomer agrees with the Terms and Conditions then first we check if the customer type is employee or guest. If customer type is Employee then the callGetLoginUserDetails function is called. This function calls another integration service. The service name is userIdentity and the operation name corresponding to it is getUserDetails. This operation takes the user credentials i.e. email and password of the employee as the parameters. If the service is called successfully the user is navigated to the **frmHome** form. The response of the service contains several details about the user and sets the gblLoginUserObj object to the accurate data.

frmHome contains four different components in it. These are headercomp, hamburgerMenu, home and newsfeed. The headercomp component contains a label and a hamburger Menu option. The hamburger menu contains a segment with an animation attached which appears and disappears on the click of the image corresponding to the widget. Next comes the home component. This component has three different flex. First flex is flxStockDateWeatherInfo which displays the information about the current stock, date and weather just after the header component. This also displays the information about the current temperature at the place.

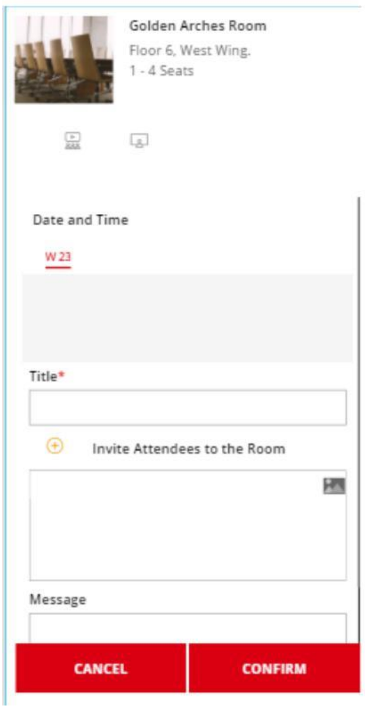


Figure 3.16 :Form Booking

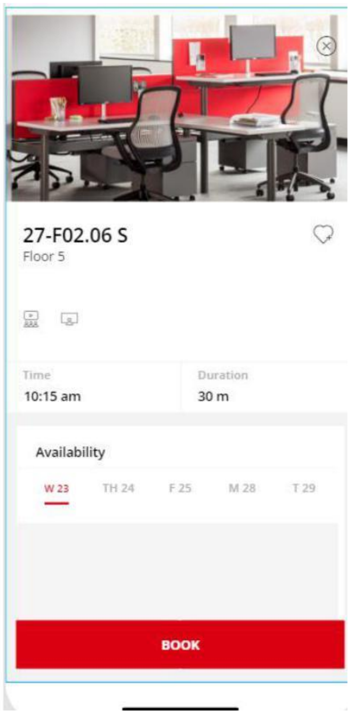
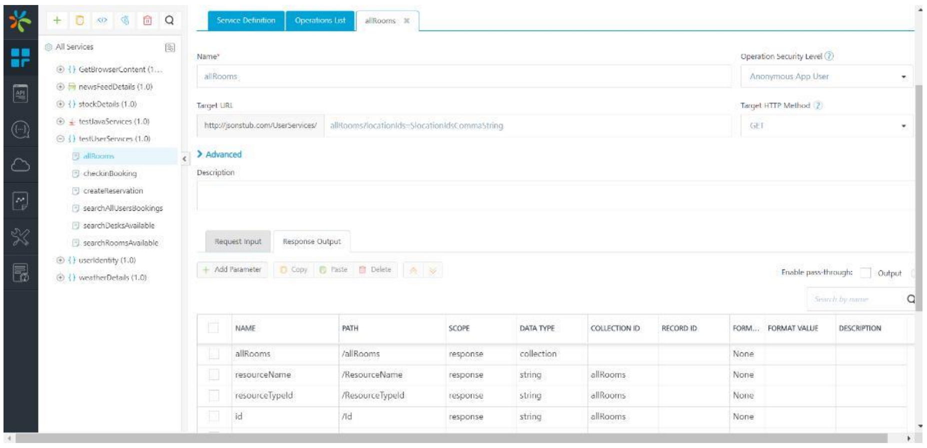


Figure 3.17 :Form Booking Detail

Then there is a flex containing information on Booked Rooms. This is actually a segment which shows information on which rooms have been booked and the exact duration related to them. At last there flxRows flex container which has three different rows. The data in these rows can be set from the frmHomeController. Each entity of these rows is actually an icon corresponding to the features that the application offers. For every item there is an icon, text defining the functionality and an event associated with it. This event is actually a navigation event that navigates the page to the appropriate form.

On navigating to the frmHome we first set the properties and events needed for the components. Then in the preshow life cycle method we call getMyBookings() function. This function corresponds to the module section of the home component. This basically gets all the bookings on the basis of date. This actually calls the callGetMyBookingsService function which calls the testUserServices and the allRooms operation. The response output consists of the following.



Name*: allRooms

Operation Security Level: Anonymous App User

Target URL: http://jsonstub.com/UserServices/ allRooms?locationIds=$locationIdsCommaString

Target HTTP Method: GET

Description:

Enable pass-through: ☐ Output

	NAME	PATH	SCOPE	DATA TYPE	COLLECTION ID	RECORD ID	FORM...	FORMAT VALUE	DESCRIPTION
☐	allRooms	/allRooms	response	collection			None		
☐	resourceName	/ResourceName	response	string	allRooms		None		
☐	resourceTypeId	/ResourceTypeId	response	string	allRooms		None		
☐	id	/Id	response	string	allRooms		None		

Figure 3.18 :Allrooms Service View

Now this response is to be set in the segment. Each row of the segment displays image of the room, the date corresponding to booking, the timing of booking and the roomName. All of this is contained in the response of the service call above.

The function getNewsFeedData is also called on preshow of this form. This actually displays the news data and news feed on the lower area of the form. This is done completely by using the newsfeed component as the footer. This again is a segment which has a template assigned to it. This template shows the headline, image related to news and the detailed news. This data is received using the integration service and the operation The response of this service contains the date, headline, headline image and the description of the news. The _setHamburgerHomeIcons is an important function which helps to add the relevant information into the hamburger menu segment and enables its functionality, such that it can be used for the purpose of navigation. Similar to this, if the user logged in as guest then we navigate him to **frmGuestHome** . This form is similar to the **frmHome** except that there is a limit on the number of features that you can access.

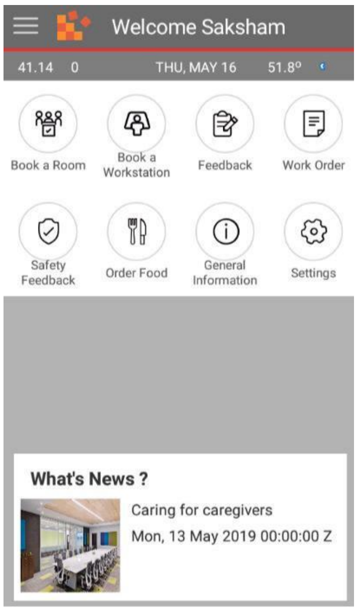
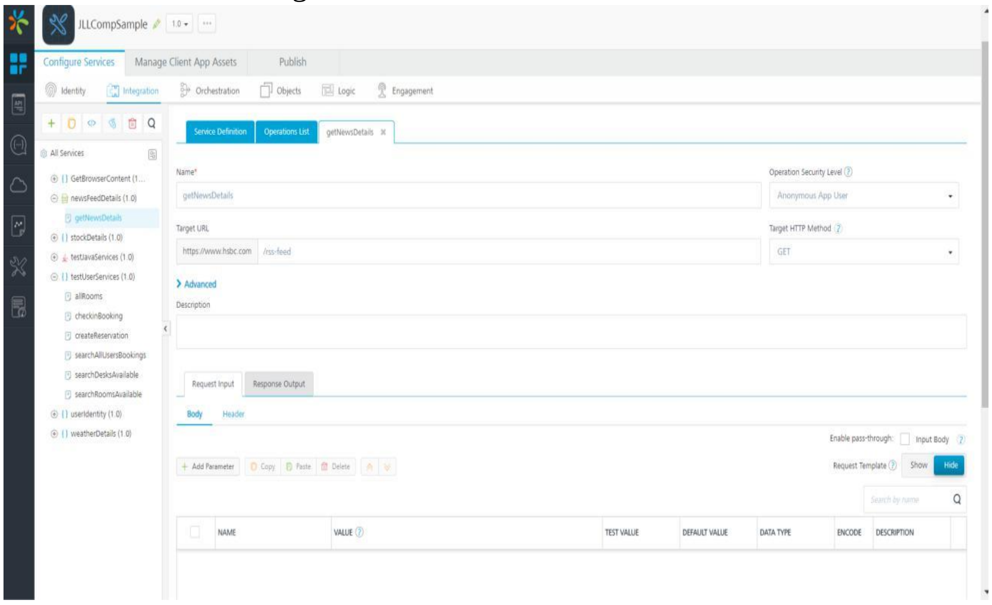


Figure 3.19 :Form Home

Now, from the home form we have a lot of features which we can use. In other words, a lot of forms we can navigate to. These are **frmAbout, frmAddWorkOrder, frmBookResource, frmFeedback, frmGeneralInformation, frmListResource, frmResourceInfo, frmSafety, frmSettings, frmTransportation**. Lets look into these forms in depth.

frmFeedback form is the form corresponding to the form where we get all the feedbacks. The forms consists of three main components headercomp, hamburgerMenu and feedback. The first two components have already been dealt with. The feedback component is the main component where feedback stuff happens.

This component has the switchcomponent imported into it which is just consists of a button. Basically if we want a reply from the user we set this button to on and if we don't this button is off. The feedback component has a flex which contains several options for setting the type of feedback. The options include: like, dislike, question and suggestion. We need to select the type of feedback before writing and filing it. Then there is a text box form taking the input. This is the actual message or feedback that the user gives. At last there is the send button.



Name*: getNewsDetails
Operation Security Level: Anonymous App User
Target URL: https://www.hsbc.com /rss-feed
Target HTTP Method: GET
Description:
Enable pass-through: Input Body
Request Template: Hide

	NAME	VALUE	TEST VALUE	DEFAULT VALUE	DATA TYPE	ENCODE	DESCRIPTION

Figure 3.20 :GetNewsDetails Operation

In the onNavigate function of the frmFeedback we call the setFeedBack function. In this we set the various parameters like setSubject, setFirstName, setLastName, setToMailAddress, setFromMailAddr, setFromEmailPassword and setCCMail. These all are the required to send the feedback mail to the appropriate authoritites. Now when the user fills the feedback, he clicks on the send button. This triggers the sendFeedbackServiceCall function.

Next,comes in the **frmGeneralInformation** . This form displays the information on the basic things and services contact list. The form contains three main components headerComp, hamburgerMenu and generalInfo.The generalInfo component has different flexes that display information of different nature. This information is on UserInfo, Directory, Locations, WIFIInfo, GuestParkingInfo, ITRequest contacts, Preferred Floors and all the Notifications for the user. Each of the flex contains a segment which is used to display the information related to the specific field. The preffered floor field is actually a checkbox which is there for the user to choose. No fabric service is used here.

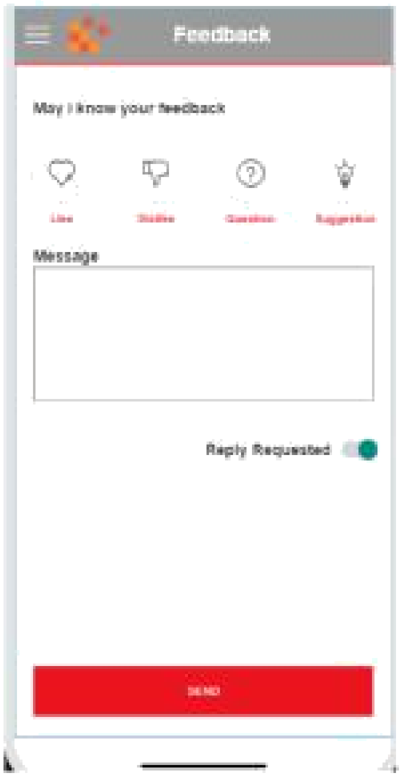
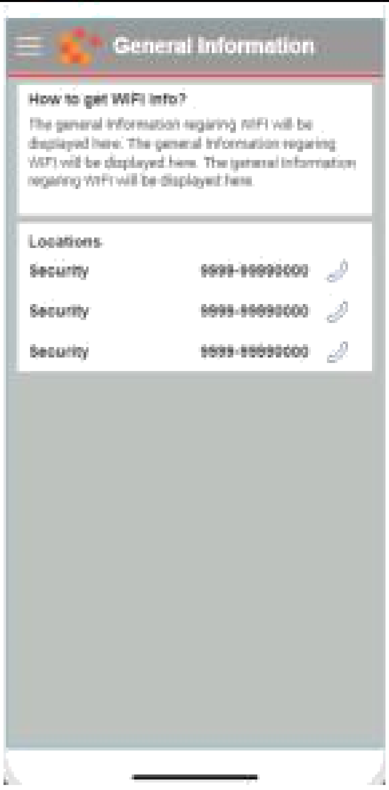


Figure 3.21 : Form Feedback



How to get WIFI info?

The general information regaring WIFI will be displayed here. The general information regaring WIFI will be displayed here. The general information regaring WIFI will be displayed here.

Locations

Security 9999-99990000

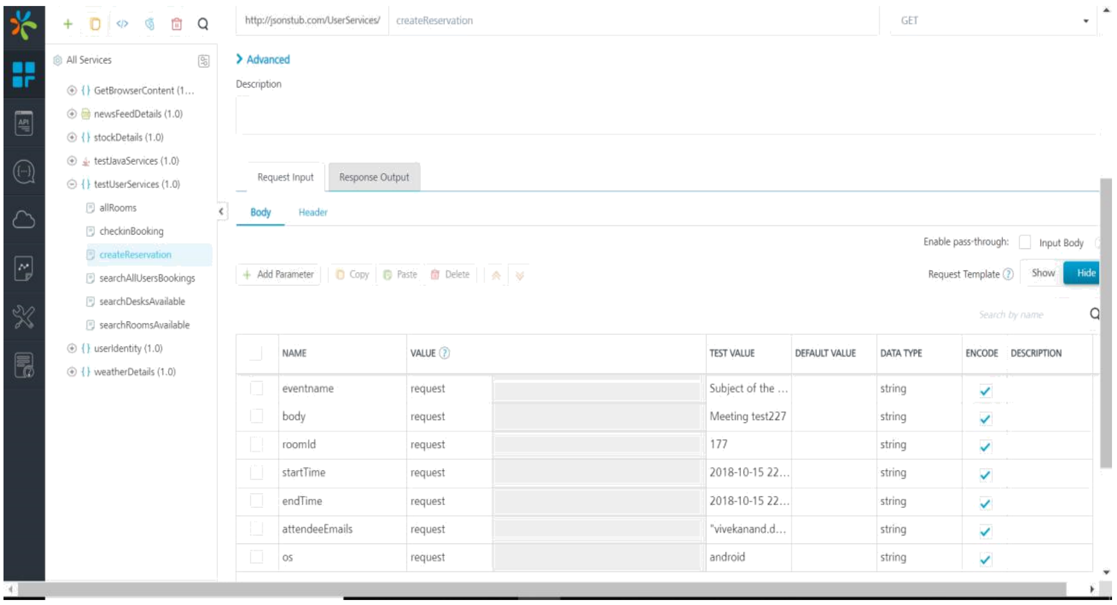
Security 9999-99990000

Security 9999-99990000

Figure 3.22 :Form General Information

The form **frmSettings** is a form similar to the **frmGeneralInformation.** This uses the same components, imports the same services. The only difference is that it is used for setting the application preferences according to the need of the customer.

Next comes in the form frmBookResource. This form is useful in booking the particular room at a particular venue according to the time the customer desires. This form just imports a single component called BookResource. This component displays the information of the resource that has been chosen. This includes image,location and seats. Next it shows the to and from date for the booking. We can add the title corresponding to the nature of the requirement. Moreover, the name of the attendees can also be mentioned for better management. At last if the customer has some other requirement then it can also be added so that it facilitates the management.



http://jsonstub.com/UserServices/ createReservation GET

All Services

- GetBrowserContent (1…
- newsFeedDetails (1.0)
- stockDetails (1.0)
- testJavaServices (1.0)
- testUserServices (1.0)
  - allRooms
  - checkinBooking
  - createReservation
  - searchAllUsersBookings
  - searchDesksAvailable
  - searchRoomsAvailable
- userIdentity (1.0)
- weatherDetails (1.0)

Advanced

Description

Request Input | Response Output

Body | Header

Enable pass-through: Input Body

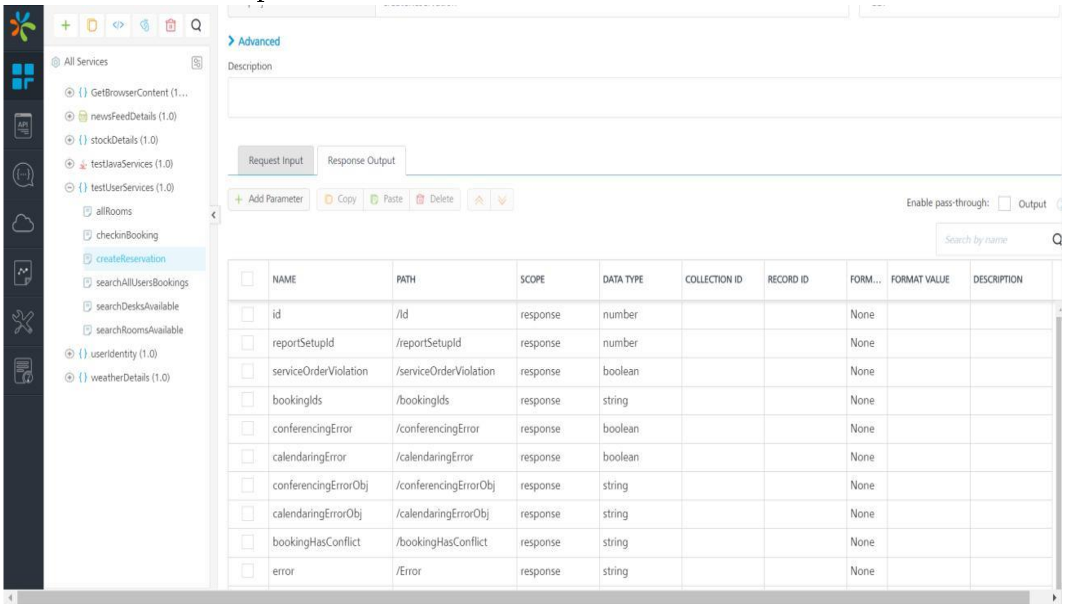
Request Template Show Hide

NAME	VALUE		TEST VALUE	DEFAULT VALUE	DATA TYPE	ENCODE	DESCRIPTION
eventname	request		Subject of the …		string	✓	
body	request		Meeting test227		string	✓	
roomId	request		177		string	✓	
startTime	request		2018-10-15 22…		string	✓	
endTime	request		2018-10-15 22…		string	✓	
attendeeEmails	request		"vivekanand.d…		string	✓	
os	request		android		string	✓	

Figure 3.23 :CreateReservation Operation

The onNavigate function of this form has onInitComponent mentioned in it. This function is defined inside the module of the corresponding component. This function first fetches all the bookings that have been made. Then we try to find out the reservations that are in the same time slot. If we are able to find such reservations then we simply display the message that the room is not available. In case if the room is available then we reserve the room for the person sending the appropriate customer requirements to the management authorities. We also have the feature to check in the user in the hotel from this component itself.

There are a lot of the services that are involved in the dynamics of this component.This includes the service testUserServices with the operation name checkinBooking. This is the service for checking in the user. Then we have the createReservation operation under the testUserServices service.



All Services

- GetBrowserContent (1…
- newsFeedDetails (1.0)
- stockDetails (1.0)
- testJavaServices (1.0)
- testUserServices (1.0)
  - allRooms
  - checkinBooking
  - createReservation
  - searchAllUsersBookings
  - searchDesksAvailable
  - searchRoomsAvailable
- userIdentity (1.0)
- weatherDetails (1.0)

Advanced

Description

Request Input | Response Output

Enable pass-through: Output

NAME	PATH	SCOPE	DATA TYPE	COLLECTION ID	RECORD ID	FORM…	FORMAT VALUE	DESCRIPTION
id	/Id	response	number			None		
reportSetupId	/reportSetupId	response	number			None		
serviceOrderViolation	/serviceOrderViolation	response	boolean			None		
bookingIds	/bookingIds	response	string			None		
conferencingError	/conferencingError	response	boolean			None		
calendaringError	/calendaringError	response	boolean			None		
conferencingErrorObj	/conferencingErrorObj	response	string			None		
calendaringErrorObj	/calendaringErrorObj	response	string			None		
bookingHasConflict	/bookingHasConflict	response	string			None		
error	/Error	response	string			None		

Figure 3.24 : Request Input Of Create Reservations

The **frmListResource** form is primordial as it lists all the resources available for the customer. This has four main components listResources, hamburgerMenu, headerComp and SearchResource. The listResources component is has several widgets. We have the date picker facility here where we enter the date on which we want a resource. We enter the date, duration and location here and then this component lists for us the available resources.

When we navigate to the **frmListResource** we call the function makeServiceCallForRooms and makeServiceCallForDesks. These functions call the integration service testUserServices under the operation allRooms and allDesks. On getting the result we push the data into a temporary variable and use it to display the resources.

Moreover, we also have the operation **searchAllUsersBookings** defined in the same service which works to find the details about the room. There are many other functions in this component. These include callServiceToGetRoomsWithFavorities which makes the service call searchRoomsAvailable to find the favorites room available for a particular user.

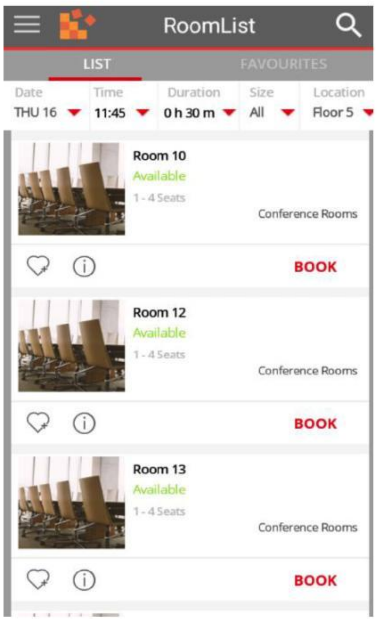


Figure 3.25 : Form RoomList

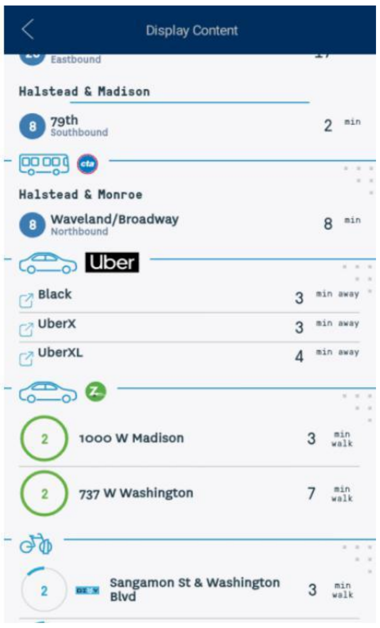


Figure 3.26 :Form Order

Running the Mobile App

Setting up the Visualizer Project

Google Account Setup for Map

This app uses Google's Map API to provide the location of the event on a map widget. The map widget requires a map key to be set up in Visualizer's settings to view the map in an app. Follow these steps to set up a Visualizer project:

Google Dev Key generation

Setting up the Key in the Visualizer project

Push Notification Setup (Visualizer)

For setting up push notification in your app, you need to configure the Visualizer app and the Kony Fabric app.

You must create an app in the Firebase Developer Console (for Firebase Cloud Messaging).

You can follow the givens steps to create an app on the Firebase Console and make the corresponding changes in Visualizer.

Registering in Firebase Developer Console

Adding senderID to project

Selecting GCM in project settings

Setting up Fabric App

LinkedIn Account Setup

The app requires LinkedIn credentials for logging in. For the LinkedIn log-in to work, you need to have an app in the LinkedIn developer portal. Follow the steps mentioned in Authenticating OAuth 2.0 to create and configure the LinkedIn app.

After you configure your LinkedIn app, you need to generate a Client ID and a Client Secret for the app. You need to configure the client details in Kony Fabric.

Follow these steps to setup the Kony Fabric identity service:

Setting up Kony Fabric Identity Service

Push Notification Setup (Kony Fabric)

After you set up an Identity Service in Kony Fabric, you need to set up push notification in your app. Follow the given steps to set up push notifications for your app:

Identity Service: customAccountLogin

Integration Service: accountLogin

Engagement Service: Android

Engagement Service: iOS

Sample Data Upload

The Visualizer project contains a sample data set. You can find this data set at the following path:

<Your Download Location>/Event/EventData.zip

Follow these steps to upload the data to the Kony Fabric app:

Uploading Sample Data

Publishing Fabric App

Once you upload the sample data to the Object Services, you need to publish the Kony Fabric app to a run-time environment. The client app accesses the data from the run-time environment.

Steps to publish Kony Fabric app

Publish Visualizer App

Once you publish the Kony Fabric app, navigate back to Visualizer and make the client app ready for preview. Use the Visualizer App Viewer mobile app to view the published client app.

You now need to publish the application to Kony Cloud to access and view it on App Viewer.

This operation reduces the need to build the app multiple times.

Steps to publish the app to the Kony Cloud

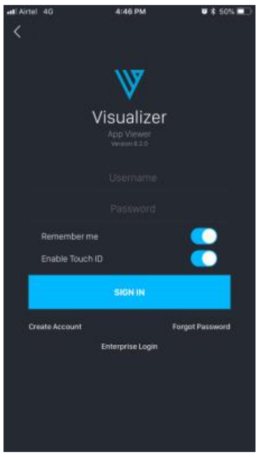
Previewing the App in Visualizer App Viewer

Apart from viewing the app, you can also use App Viewer to provide comments on the design and functionality of the app.

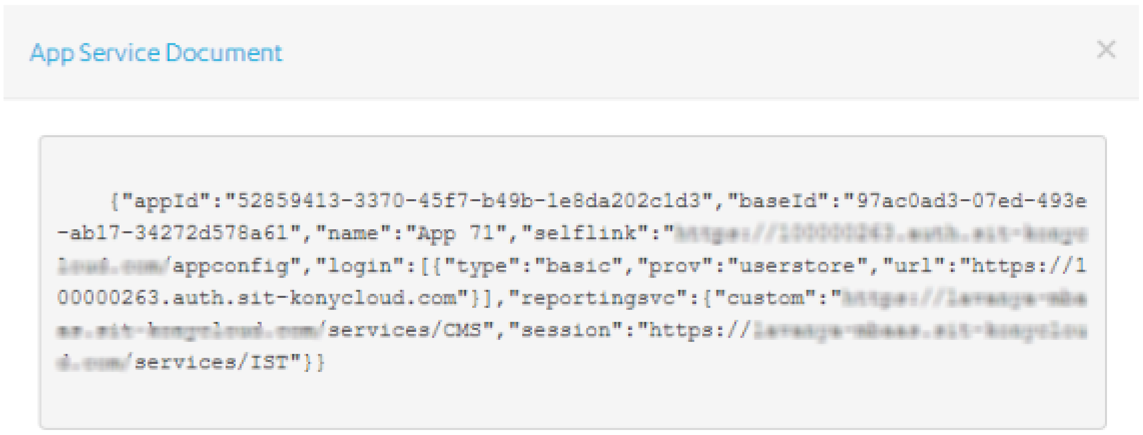
Steps to preview your app

1.Launch the Visualizer App Viewer app on your mobile.

2. Sign in with using your Kony account credentials.



3.On the landing screen of the app, enter the five-digit preview code, which you received earlier, in the App Viewer ID field.



```
{"appId":"52859413-3370-45f7-b49b-1e8da202c1d3","baseId":"97ac0ad3-07ed-493e
-ab17-34272d578a61","name":"App 71","selflink":"[redacted]/appconfig","login":[{"type":"basic","prov":"userstore","url":"https://1
00000263.auth.sit-konycloud.com"}],"reportingsvc":{"custom":"[redacted]/services/CMS","session":"https://[redacted]/services/IST"}}
```

4.Tap the right-arrow to preview the Employee Directory app on your mobile.

The debugging techniques are also similar. Normally, we just build the application and launch in the mobile device. This generates a link that we can use go to directly in our browser and use this to debug. We can use the built in debugger of the browser, as the application is mosty in javascript and can use the breakpoints and other debug options to debug the application.

CHAPTER – 4

PERFORMANCE ANALYSIS

Here we are going to compare the performance of the application structure to the architectural styles and methodology that are available in the market or that could have been adopted to create the aforementioned applications. The application is built on the top of the mvc architecture, contrary to the traditional free form javascript. MVC stands for model, view and controller. The model deals with the database and data related entities. View deals with the visual part that is the things that are to be displayed. At last controller deals with the business logic involved in carrying out the various operations of the application optimally. MVC provides a better structure to the overall application, providing the required functionality with it and makes the overall application lightweight as it injects the require module capabilities.

This application deeply deals with the components which are a part of the Kony Visualizer application since the version 8.2 fix pack 1. Components inject higher performance features into the application and this is widely evident in this application. To test this application and for the purpose of performance analysis we are going to use the standard tools here. This includes the SAP standard tools for MOM (Manufacturing operation Management) . Along with this we need the SkyMobile application installed in the device that has the application that we are going to test. This application has been specially created by the product team of Kony for this specific purpose only.

The SkyMobile Framework was developed specifically for SAP as a complimentary add-in component. It requires no extra infrastructure, other than that which SAP supports. The following notes must be reviewed with this context in mind, since Sky does not position SkyMobile as a competitor to MI, but rather as a complimentary architecture that provides extended data capture and device integration capabilities.

SkyMobile runs inside of SAP ERP/CRM/APO/BW providing better transaction management, multiple user interface options, built-in peripheral device and multi-media support and inter-product integration. There is no doubt that NetWeaver Mobile (MI) will provide good mission critical mobile applications on a grand scale, but if customisation or a highly specialised mobile application is required, the SkyMobile Business Framework may offer a better cost effective alternative. A current limitation of SAP Mobile is that the NetWeaver MI component runs outside of SAP ERP, CRM etc. and it does not provide any interface management framework within the SAP ERP or CRM environment. Thus any end-to-end management of asynchronous transactions is not possible.

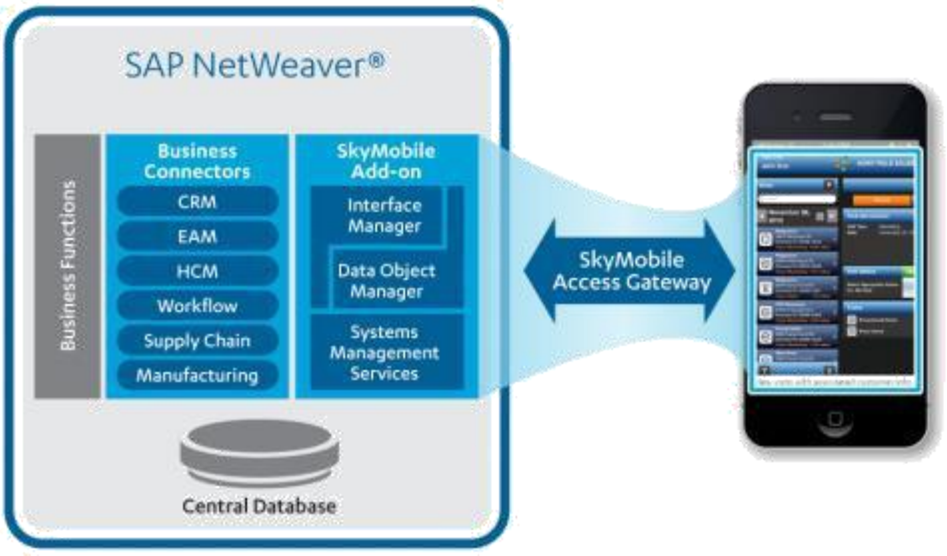


Figure 4.1 : Brief Outlook

Now, there are three basic methods to use these performance analysis services.

1.We can create a test ourselves.

2.We can record an application to create a task.

The general stages involved in both the tasks are more or less the same. These include:

1. Creating a test.
2. Running this test on the application.
3. Scrutinizing the test results and deciding upon the performance.

To record an application for creating a task we need to use the SAP MOM. Procedure is:

1. Open the application to be anlysed in the device.
2. Logon to the system containing SkyMobile and MOM and execute the YNOMR transaction.

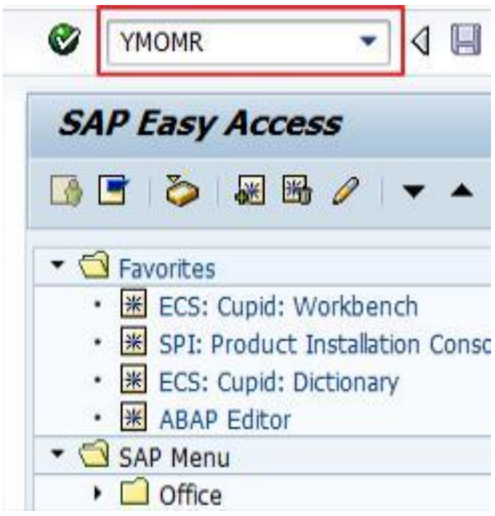
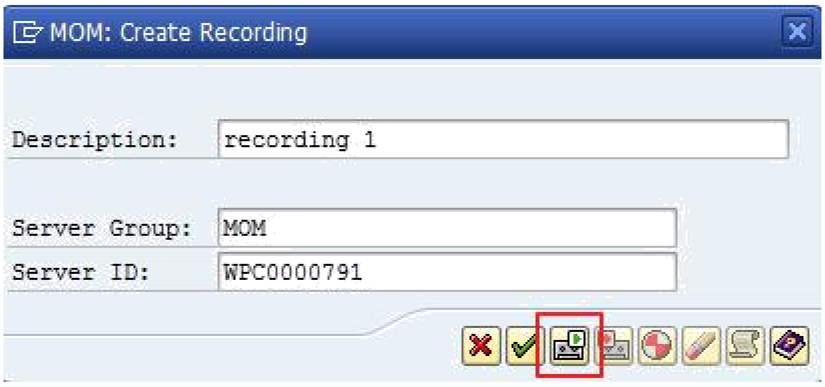


Figure 4.2 : SAP Easy Access View

3. Choose New Recording. Enter the description and other fields that emerge along with it. At last, click on create recording.
4. As we are for the testing of the Kony Fabric server here we need to enter the required details for establishing a connection with the server.

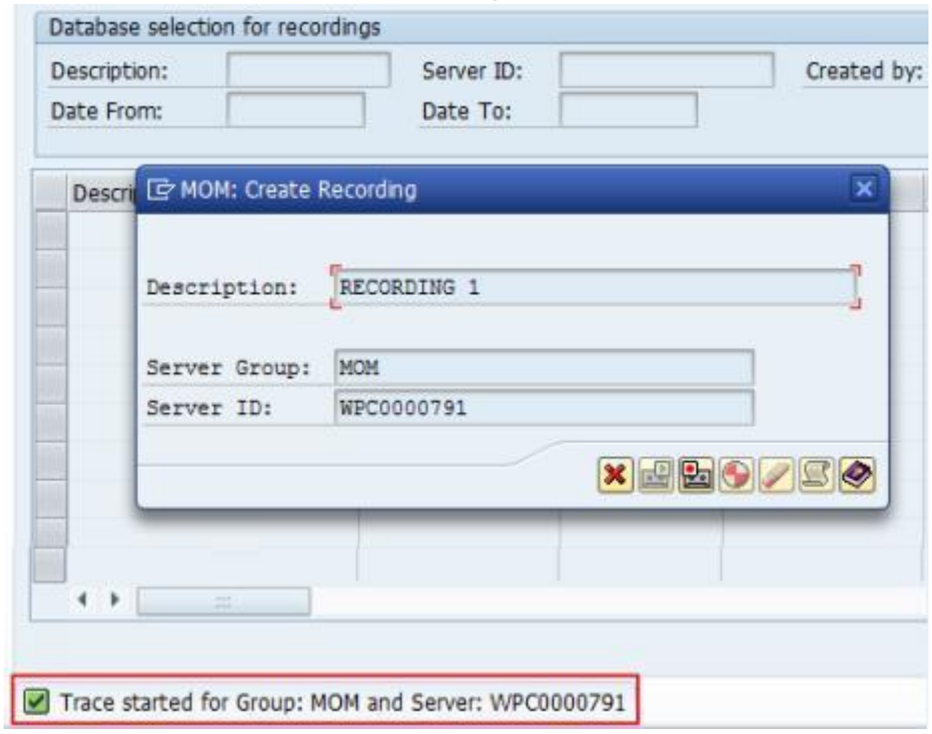


MOM: Create Recording

Description:	recording 1
Server Group:	MOM
Server ID:	WPC0000791

Figure 4.3 :Create Recording View

5. Some additional information concerned with the recording is required. Enter this detail and click on Start Recording.



Database selection for recordings

Description:		Server ID:		Created by:
Date From:		Date To:		

MOM: Create Recording

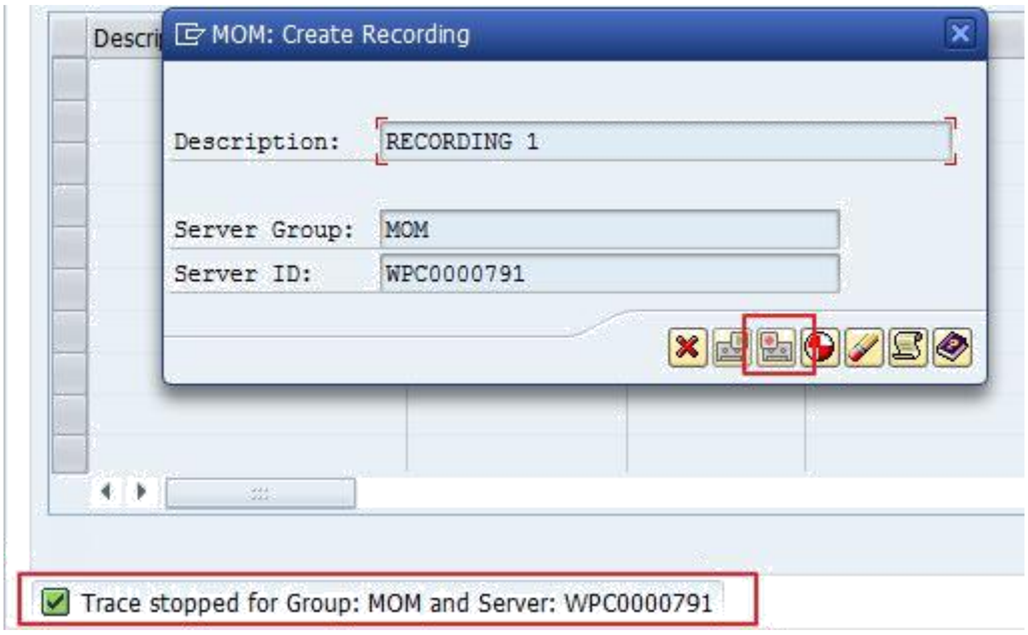
Description:	RECORDING 1
Server Group:	MOM
Server ID:	WPC0000791

Trace started for Group: MOM and Server: WPC0000791

Figure 4.4 : Recording Details View

6. Now launch and execute the application which you want to test.

7. At the end of the use, click on the stop recording button on the MOM.



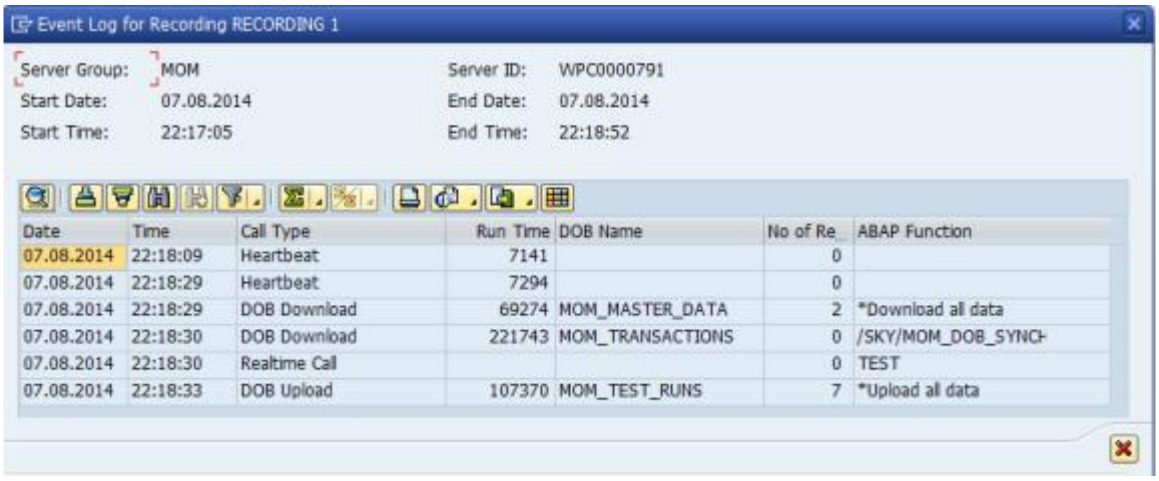
MOM: Create Recording

Description:	RECORDING 1
Server Group:	MOM
Server ID:	WPC0000791

Trace stopped for Group: MOM and Server: WPC0000791

Figure 4.5 : Recoring Completion View

8. For checking the details click on the event logs. This will show the name of the recording and just under this you can see the various events and the performance related to them.



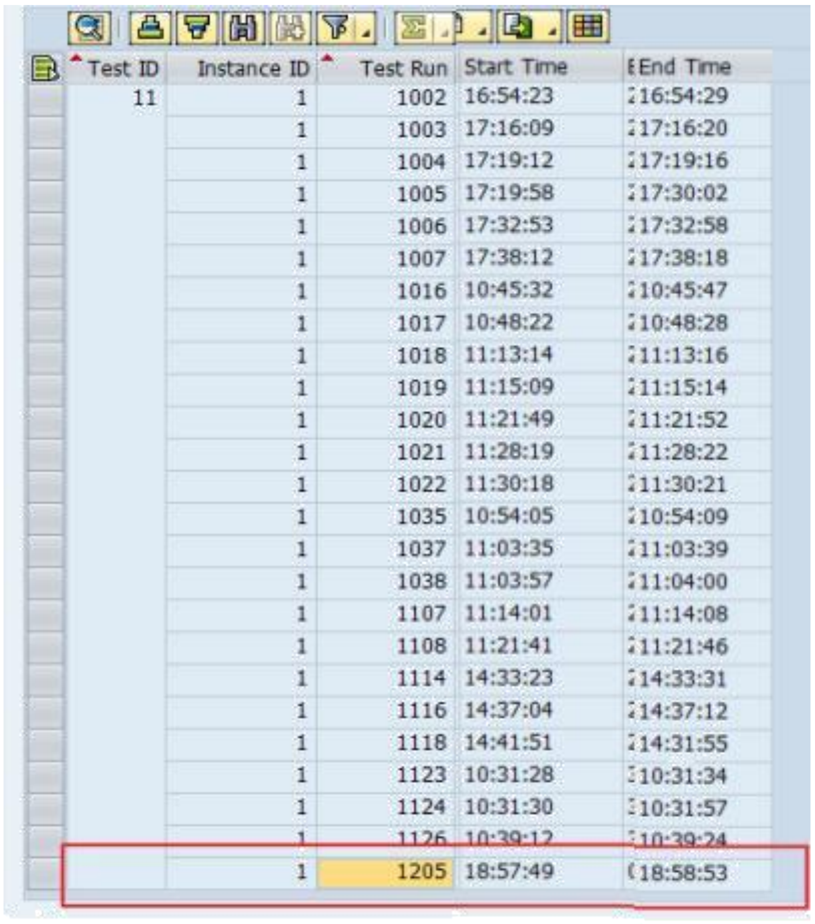
Event Log for Recording RECORDING 1

Server Group:	MOM	Server ID:	WPC0000791
Start Date:	07.08.2014	End Date:	07.08.2014
Start Time:	22:17:05	End Time:	22:18:52

Date	Time	Call Type	Run Time	DOB Name	No of Re	ABAP Function
07.08.2014	22:18:09	Heartbeat	7141		0	
07.08.2014	22:18:29	Heartbeat	7294		0	
07.08.2014	22:18:29	DOB Download	69274	MOM_MASTER_DATA	2	*Download all data
07.08.2014	22:18:30	DOB Download	221743	MOM_TRANSACTIONS	0	/SKY/MOM_DOB_SYNCH
07.08.2014	22:18:30	Realtime Call			0	TEST
07.08.2014	22:18:33	DOB Upload	107370	MOM_TEST_RUNS	7	*Upload all data

Table 4.1 Logs

A detailed report of the performance is also generated. This detail consists of all the information we need to test our application in detail and look at the results correspondingly and tweak with the parts of the application which are responsible for worsening the application and server performance.



Test ID	Instance ID	Test Run	Start Time	End Time
11	1	1002	16:54:23	16:54:29
	1	1003	17:16:09	17:16:20
	1	1004	17:19:12	17:19:16
	1	1005	17:19:58	17:30:02
	1	1006	17:32:53	17:32:58
	1	1007	17:38:12	17:38:18
	1	1016	10:45:32	10:45:47
	1	1017	10:48:22	10:48:28
	1	1018	11:13:14	11:13:16
	1	1019	11:15:09	11:15:14
	1	1020	11:21:49	11:21:52
	1	1021	11:28:19	11:28:22
	1	1022	11:30:18	11:30:21
	1	1035	10:54:05	10:54:09
	1	1037	11:03:35	11:03:39
	1	1038	11:03:57	11:04:00
	1	1107	11:14:01	11:14:08
	1	1108	11:21:41	11:21:46
	1	1114	14:33:23	14:33:31
	1	1116	14:37:04	14:37:12
	1	1118	14:41:51	14:31:55
	1	1123	10:31:28	10:31:34
	1	1124	10:31:30	10:31:57
	1	1126	10:39:12	10:39:24
	1	1205	18:57:49	18:58:53

Figure 4.2 :Result

CHAPTER – 5

CONCLUSIONS

Working at KONY IT Private Ltd. Has been as fun and productive as it has been working on these projects. Best Buy And JLL Components Application are equally significant in their own places. The former provides ease of operation to the technical team while the latter gives an insight into the enrolment procedure to a layman who's from a minimal technical background.

Kony JLL COmponents has been the most extensive application of the two as it processes complex and heavy server requests. The application needs more modifications for increasing efficiency. Also, there are certain bugs which need to be dealt with to make it deployable to the technical team of the Kony Fabric. After the aforementioned changes, the application will go through an extensive modular and system testing to make it work at it's best efficiency.

Best Buy Application on the other hand was a top level development with it's foundations in the low level SDKs containing DLLs for interaction with the biometric devices. The DLLs are full fledged and capable of exception handling and failure management which prevents application from crashing and providing a detailed error log instead. This application has a slightly flashy UI and a relatively smoother UX as it is meant to be used by users of all domains irrespective of their technical background.

This Application has been deployed successfully on our cloud storage SharePoint and is being tested and used by the sales team of JLL for demonstration to future customers. While Kony Best Buy, on the other hand, needs a bit of debugging and improvisation before staging it to SharePoint.

Overall, development of these applications has been a really learning experience in terms of writing legible, comprehensible and maintainable code, debugging to an extent that each and every module knows how to handle exceptional situations, creating a UI that is not just good to look but is equally good to use and work with.

There are indeed places which need extemporization and diligence to reach the mark of perfection and excellence. I've been working hard to reach there and will confidently accomplish it some day. I will keep these precious learnings with me forever and hope to flourish more with Kony.

REFERENCES

1. http://www.controldeasistencia.mx/uploads/5/8/3/3/58339667/3m__csd_200i_sin gle-digit_optical_fingerprint_scanner_v122015.pdf
2. https://manage.kony.com/console/
3. https://basecamp.kony.com/s/
4. https://www.kony.com/resources/blog/introducing-kony-base-camp/