**POWER BI**

**A PROJECT REPORT**

**Submitted in partial fulfillment of the**

**Requirements for the Degree of**

**MASTER OF COMPUTER APPLICATIONS**

**By**

**JAI SINGH**

**University Roll No 1900290140017**

**Under the supervision of**

**Ms Neelam Rawat**

**KIET group of institutions**



**Submitted to**

**Department Of Computer Applications**

**KIET Group of Institutions, Ghaziabad**

**Uttar Pradesh-201206**

**May,2022**

**Declaration**

I undersigned hereby declare that the project report ( “**POWER BI**”) , submitted for partial fulfillment of the requirements for the requirement for the award of the degree of Master of Computer Applications by the ‘KIET GROUP OF INSTITUTION’S GHAZIABAD’ is a bonafide work done by me under supervision of (Pro. Neelam Rawat). This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission.

Name : Jai Singh

Roll No: 1900290140017

Branch: MCA (6th Sem)

**(Candidate Signature)**

**Certificate**

Certified that **Jai Singh (enrollment no. 1900290140017)** has carried out the project work having “Order Guide” for Master of Computer Applications from Dr. A.P.J. Abdul Kalam Technical University (AKTU) (formerly UPTU), Technical University, Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself/herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

**Date:**

This is to certify that the above statement made by the candidate is correct to the best

of my knowledge.

**Date**:

**Pro. Neelam Rawat**

**Associate Professor**

**Department of Computer Applications**

**KIET Group of Institutions, Ghaziabad**

**Signature of Internal Examiner Signature of External Examiner**

**Abstract**

### 

POWER BI is a great tool for Business Organization who have access to a Graphical Representation. It is useful for Bussiness who are working professionals and wish to improve. It allow business living in remote areas to attend organization This method of Representation has many benefits like ease of Bussiness. The POWER BI is installed on the smartphone and Desktop can be used anytime. It gives access to both live sessions and pre-recorded to Organization. People use audio-visual mode for Representation. people engage in the power bi and interact with other bussiness very easily. There are many BI tools offering millions of Representation. Many Organization and Industry also give the benefit of this tools. These allow to choose visualization of their interest. There are both paid and free visualization available to choose from. Organization even get a certificate of completion when they finish a project

**ACKNOWLEDGEMENT**

I extend my deepest appreciation to Successive Technologies Pvt. Ltd .My esteemed guide, for providing me with the possibility to complete this project with the right guidance and advice.

Special gratitude I give to my respected head of the division **Dr. Ajay Kr Srivastav, Head of Department, and all faculties Master of Computer Applications**, for allowing me to use the facilities available and also help me to coordinate my project

Furthermore, I would also like to acknowledge with much appreciation the crucial role of faculty members on this occasion.

Last but not least, I would like to thank friends who help me to assemble the parts and gave a suggestion about the project.

.

**TABLEOFCONTENT**

|  |  |  |
| --- | --- | --- |
|  | | |
|  | |  |
|  | | Declaration |
|  | | Certificate |
|  | | Abstract  Training Certificate |
|  | | Acknowledgement |
|  | |  |
| **BI 1-** | | **INTRODUCTION 9-10** |
| 1.1 | | INTRODUCTION 9 |
| 1.2 | | PURPOSE 9 |
| 1.3 | | SCOPE 10 |
| **BI 2-** | | **PROJECT CATEGORY 11-12** |
| 2.1 | | TECHNOLOGY USED 11 |
| 2.2 | | LANGUAGE USED 12 |
| **BI 3-** | | **SOFTWARE REQUIREMENT SPECIFICATION 13-18** |
| 3.1 | | HARDWARE CONFIGURATION 13 |
| 3.2 | | SOFTWARE REQUIREMENT 13 |
| 3.3 | | FUCTIONAL REQUIREMENT 14 |
| 3.4  3.5  3.6  3.7  3.8  3.9 | | NON- FUNCTIONAL REQUIREMENT 14  SOFTWARE SYSTEM ATTRIBUTE 14  FEATURE 14  PRELIMINARY INVESTIGATION 15  APPROACH USED : AGILE APPROACHED 16  PRELIMINARY DESCRIPTION 18 |
| **BI 4-** | | **FEASIBILITY STUDY 19** |
| 4.1  4.2  4.3 | | ECONOMIC FEASIBILITY 19  TECHNICAL FEASIBILITY 19  OPERATINAL FEASIBILITY 19 |
| **BI 5-** | **PLANNING SCHEDULING AND FLOW 20** | | |  |
| 5.1 | GANT CHART 20 | | |  |
| **BI 6- ANALYSING AND DESIGN 21**  6.1 ANALYSIS 21  6.2 DESIGN 21  **BI 7- DATA FLOW DIAGRAM 22-24**  7.1 DESCRIPTION 22  7.2 DIAGRAM 24    **BI – ER NOTATION 25-26**  7.1 DESCRIPTION 25  7.2 DIAGRAM 26  **BI 9- IMPLEMENTATION AND SYSTEM TESTING 27**  9.1 SYSTEM TESTING 27  **BI 9- VALUATION 28-84** | | | |  |
| **CONCLUSION 85**  **DECLARATION 86** | | | |  |

|  |  |  |
| --- | --- | --- |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  |  | | |  |
|  |  | | |  |
|  | | | |  |
|  | | | |  |
|  | | | |  |
|  | | | |  |
|  | | | |  |

**BI 1**

**INTRODUCTION**

**1.1 PROJECT DESCRIPION**

The online applications for academic purposes are called POWER BI. Such applications make use of an internet connection. The POWER BI can be accessed from a smartphone or a Desktop. It is a technology-based BI tool that enables information sharing. It is commonly known as mobile apps and desktop app for visualization.

POWER BI is a great Representation tool for Organization who do not have access to a physical Visualization. It is useful for Organization who are working professionals and wish to improve. It allows Organization living in remote areas to attend Visualization.

This method of Representation has many benefits like ease of education. An POWER BI is installed on the smartphone and can be used anytime. It gives access to both live sessions and pre-recorded Visualization to Organization. Trainers use audio-visual mode for Representing. Organization engage in the online Visualization and interact with other Organization very easily.

There are many POWER BI offering millions of Visualization. Many schools and colleges also give Organization the benefit of attending Visualization online in admissions. These allow Organization to choose subjects of their interest. There are both paid and free Visualization available to choose from. Organization even get a certificate of completion when they finish a POWER BI.

The method of POWER BIs is user-friendly. Organization who attend the session have to mark their attendance. Organization get assignments to complete within deadlines. Organization can track their performances. They can even give feedback for the improvement of Representation and Representing methods.

**Purpose:-**

An POWER BI is installed on the smartphone and can be used anytime. It gives access to both live sessions and pre-recorded Visualization to Organization. Trainers use audio-visual mode for Representing. Organization engage in the online Visualization and interact with other Organization very easily.

**SCOPE:-**

The main purpose of education is to achieve upward mobility. Online Visualization certification programs have been able to provide inexpensive education to the masses and also save time, energy and money. Electronic-Representation through certified online Visualization provides a wide range of Visualization that caters to the core interests of the student, thus creating a fertile arena for future advancement .There is a misplaced notion, that employers prefer Organization with traditional brick and mortar college degrees.

On the contrary, corporate organizations in India are recognizing the high skill levels of Organization who have undergone online Visualization certification programs from highly acclaimed educational institutions.

**BI 2**

**PROJECT CATEGORY**

**2.1 Technology Used**

**POWER BI**

POWER BI is an open-source Microsoft software created by Meta Platforms, Inc. It is used to develop applications for Android, Android TV, iOS, macOS, tvOS, Web, Windows and UWP by enabling developers to use the React framework along with native platform capabilities.  It provides a slick, smooth and responsive user interface, while significantly reducing load time. It's also much faster and cheaper to build apps in React Native as opposed to building native ones, without the need to compromise on quality and functionality.

**Software and Applications Used**

APPLICATION : POWER BI

OPERATING SYSTEM : WINDOWS 10

FRONT END : Visualization

BACK END : SQlite , DAX

**Back-end :**

* **Visualization:-** It is realtime database developed by and then acquired by Google in 2014 used for developing the high quality DASHBOARD act as a storage of information of various data.
* **SQlite:-** SQLite is a database engine written in the C language. It is not a standalone app; rather, it is a library that software developers embed in their apps. As such, it belongs to the family of embedded databases

**2.2Language Used**

This project has been developed in React Native and JavaScript

* + **React Native :** React Native is an open-source UI software framework created by Meta Platforms, Inc. It is used to develop applications for Android, Android TV, iOS, macOS, tvOS, Web, Windows and UWP by enabling developers to use the React framework along with native platform capabilities.
  + **Stylesheet :** StyleSheet is React Native StyleSheet is a way of styling an application using JavaScript code, the main function of react native StyleSheet is concerned with styling and structuring of components in an application, all components of react native make use of a prop known as style, names and properties values work in a similar way as the web StyleSheet . There are three type of styling in react native
* Inline
* Internal or Embedded CSS
* External CSS
* **SQlite:-** SQLite is a database engine written in the C language. It is not a standalone app; rather, it is a library that software developers embed in their apps. As such, it belongs to the family of embedded databases

**BI 3**

**Software Requirement Specification**

**3.1 Hardware Configuration :**

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware.

Memory – All software, when run, resides in the random access memory (RAM) of computer. Memory requirements are defined after considering demands of the application, operating system, supporting software and files, and other running processes. Optimal performance of other unrelated software running on a multi-tasking computer system is also considered when defining this requirement.

|  |  |
| --- | --- |
| Number | Description |
| 1 | PC with 500 GB or more Hard disk |
| 2 | PC with 8 GB RAM. |
| 3 | 2nd generation Intel Core or newer |

**Table 3.1 Hardware Requirements**

**3.2 Software Requirements**

The software requirements are description of features and functionalities of the target system. Requirements convey the expectations of users from the software product. The requirements can be obvious or hidden, known or unknown, expected or unexpected from client’s point of view .Every project needs software. We should try to understand what sort of requirements may arise in the requirement elicitation phase and what kinds of requirements are expected from the software system.

|  |  |  |
| --- | --- | --- |
| Number | Description | Type |
| 1 | Operating System | Windows XP / Windows |
| 2 | Language | React Native |
| 3 | Database | SQlite |
| 4 | Android Studio | Emulator |
| 5 | Server | NPM |

**Table 3.2 Software Requirements**

**3.3 Functional Requirements:**

**Internet Connectivity:**

As discussed that Application will work on Online mode so it need regular Internet Connectivity to signup and login.

**Facebook Account:**

User can directly login to the facebook account to access this application they don’t need to signin in apps environment for which facebook account is mandatory.

**Email id and Mobile Number:**

To access the application and to signin or login user must have email id and mobile number to fill the mandatory field in the form.

**3.4 Non-functional Requirements:**

Performance Requirements

* User friendly**:** The system should be user friendly so that it can easily be understand by the user without any difficulty.
* Ease of maintenance :- System should be easy to maintain and use.
* Less time consuming: The system should be less time consuming which could be achieved by good programming.
* Error free: The system should easily handle the user error in any case.
* Static: Application runs on stand alone machine i.e. Android mobile phone of API level 16 and onward. Support only single user.

**3.5 Software System Attributes:**

1. **Security**: The system should be secure from the unauthorized access and should be password protected so that no other user can access it.

If the user is new then he needs to Signup with required details and a can also login with the facebook.

1. **Portability**:- The system should be machine independent.
2. **Maintainability**: The system will be designed in a maintainable order. The system can be easily modified and renewed according to the need of the organization.

**3.6 Features**

* Security of data.
* Ensures data accuracy.
* Minimize manpower.
* Minimum time consumption.
* Greater efficiency.
* Fast
* Better services.
* User friendliness and interactive.
* Minimum time required.
* Easy to update
* User friendly
* Free for the user
* knowing about entry time and exit time

The online applications for academic purposes are called POWER BI. Such applications make use of an internet connection. An POWER BI can be accessed from a smartphone. It is a technology-based study tool that enables information sharing. It is commonly known as mobile apps for Representation.

POWER BI is a great Representation tool for Organization who do not have access to a physical Visualization. It is useful for Organization who are working professionals and wish to improve. It allows Organization living in remote areas to attend Visualization.

This method of Representation has many benefits like ease of education. An POWER BI is installed on the smartphone and can be used anytime. It gives access to both live sessions and pre-recorded Visualization to Organization. Trainers use audio-visual mode for Representing. Organization engage in the online Visualization and interact with other Organization very easily.

There are many POWER BIs offering millions of Visualization. Many schools and colleges also give Organization the benefit of attending Visualization online in admissions. These allow Organization to choose subjects of their interest. There are both paid and free Visualization available to choose from. Organization even get a certificate of completion when they finish a POWER BI.

**3.7 Preliminary investigation:**

Fact Finding:

After obtaining the background knowledge, we began to collect data on the existing system.

The tools that are used in information gathering are as follows:

* On-site observation.
* Questionnaire.
* Review of the peoples.

The model we have used is Waterfall Model. In this model, first of all the existing system is observed, then customer requirements are taken in consideration then planning, modelling, construction and finally deployment.

**3.8Approach used: Agile Approach**

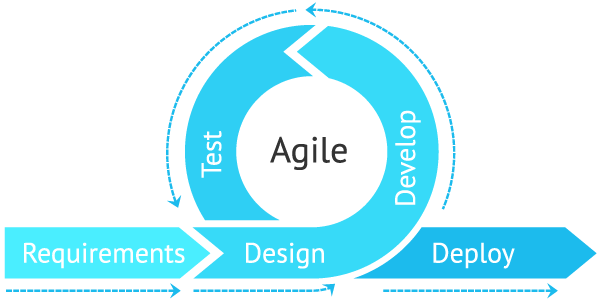


Fig. 3.1

Agile is **an iterative approach to project management and software development that helps teams deliver value to their customers faster and with fewer headaches**. Instead of betting everything on a "big bang" launch, an agile team delivers work in small, but consumable, increments.

**3.9 Preliminary Description:**

The first step in the system development life cycle is the preliminary investigation to determine the feasibility of the system. The purpose of preliminary investigation is to evaluate project requests. It is not a design study nor does it include the collection of details to describe the system in all respect. Rather, it is the collecting of information that helps committee members to evaluate the merits of project request and make an informed judgment about the feasibility of the proposed project.

**Analyst working on the preliminary investigation should accomplish the following objectives:**

* Clarify and understand the project request.
* Determine the size of the project.
* Access costs and benefits of alternative approaches.
* Determine the technical and operational feasibility of alternative approaches.
* Report the findings to management with recommendations outlining the acceptance and rejection of the proposal.

**BI 4**

**Feasibility study**

After studying and analyzing all the existing and requires functionalities of the system, the next task is to do the feasibility study for the project. Feasibility study includes consideration of all the possible ways to provide a solution to a given problem. The proposed solution should satisfy all the user requirements and should be flexible enough so that future changes can be easily done based on the future upcoming requirements.

**4.1 Economical Feasibility:**

It will be freely available on the Google play store without having any cost.

* 1. **Technical feasibility:**

This included the study of function, performance and constraints that may affect the ability to achieve an acceptable system. For this feasibility study, we studied complete functionalities to be provided in the system, as described in the System Requirement Specification (SRS), and checked if everything was possible using different type of front end and back end platform.

* 1. **Operational Feasibility:**

No doubt the technically growing Bihar needs more enhancement in technology, this apps is very user friendly and all inputs to be taken all self-explanatory even to a layman.

**BI 5**

**Planning and Scheduling and Flow**

5.1 **Gantt chart**

A Gantt chart can be developed for the entire project or a separate chart can be developed for each function. A tabular form is maintained where rows indicate the task with milestones and columns indicate duration (weeks/months).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Days  Process | 1-5 | | 6-25 | 26-30 | | | 30-80 | | 80-85 | | | 85-90 | |
| Requirement  Gathering |  |  |  |  | | |  | |  | | |  | |
| Design |  |  |  |  | | |  | |  | | |  | |
| Test Cases |  | |  |  |  |  | |  | | |  | | | |  |
| Coding |  | |  |  |  | |  | |  | | |  | |
| Testing |  | |  |  | | |  | |  |  | |  |  |
| Build |  | |  |  | | |  | |  | | |  |  |

**BI 6**

**Analysis and Design**

**6.1 Analysis:**

In present all visitor work done on the paper. The whole year visitor is stored in the registers. We can’t generate reports as per our requirements because its take more time to calculate the visitors report.

**Disadvantage of present system:**

* **Not user friendly:** The present system not user friendly because data is not stored in structure and proper format.
* **Manual Control:** All report calculation is done manually so there is a chance of error.
* **Lots of paper work:** Visitors maintain in the register so lots of paper require storing details.
* **Time consuming**

**6.2 Design Introduction:**

Design is the first step in the development phase for any techniques and principles for the purpose of defining a device, a process or system in sufficient detail to permit its physical realization.

Once the software requirements have been analyzed and specified the software design involves three technical activities - design, coding, implementation and testing that are required to build and verify the software.

The design activities are of main importance in this phase, because in this activity, decisions ultimately affecting the success of the software implementation and its ease of maintenance are made. These decisions have the final bearing upon reliability and maintainability of the system. Design is the only way to accurately translate the customer’s requirements into finished software or a system.

Design is the place where quality is fostered in development. Software design is a process through which requirements are translated into a representation of software. Software design is conducted in two steps. Preliminary design is concerned with the transformation of requirements into data.

**BI 7**

**Data Flow Diagram**

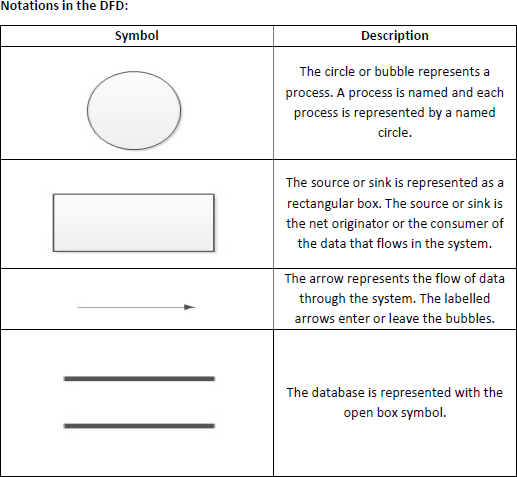
A **data-flow diagram** is a way of representing a flow of data through a [process](https://en.wikipedia.org/wiki/Process) or a system (usually an [information system](https://en.wikipedia.org/wiki/Information_system)). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow — there are no decision rules and no loops. Specific operations based on the data can be represented by a [flowchart](https://en.wikipedia.org/wiki/Flowchart).

There are several notations for displaying data-flow diagrams. The notation presented above was described in 1979 by [Tom DeMarco](https://en.wikipedia.org/wiki/Tom_DeMarco) as part of [structured analysis](https://en.wikipedia.org/wiki/Structured_analysis).

For each data flow, at least one of the endpoints (source and / or destination) must exist in a process. The refined representation of a process can be done in another data-flow diagram, which subdivides this process into sub-processes.

The data-flow diagram is a tool that is part of [structured analysis](https://en.wikipedia.org/wiki/Structured_analysis) and [data modeling](https://en.wikipedia.org/wiki/Data_modeling). When using [UML](https://en.wikipedia.org/wiki/Unified_Modeling_Language), the [activity diagram](https://en.wikipedia.org/wiki/Activity_diagram) typically takes over the role of the data-flow diagram. A special form of data-flow plan is a site-oriented data-flow plan.

Data-flow diagrams can be regarded as inverted [Petri nets](https://en.wikipedia.org/wiki/Petri_nets), because places in such networks correspond to the semantics of data memories. Analogously, the semantics of transitions from Petri nets and data flows and functions from data-flow diagrams should be considered equivalent.



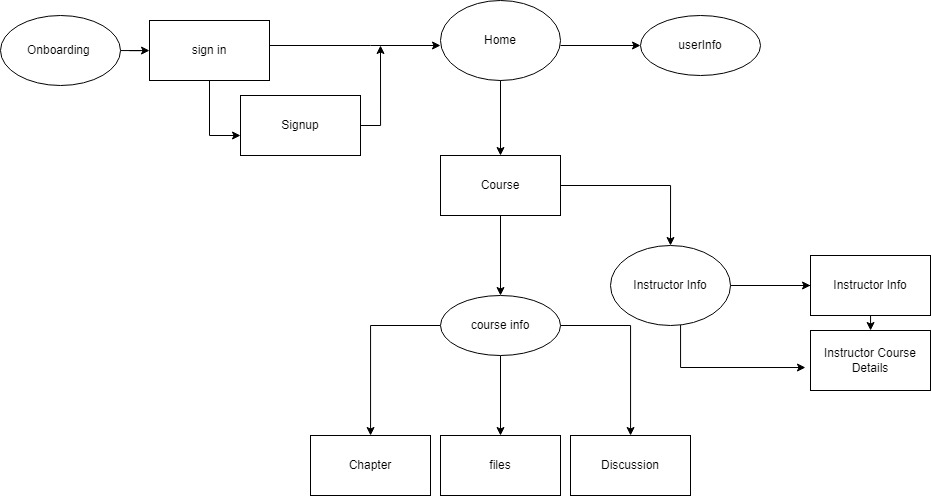
****

Fig 6.1

**BI 8**

**ER Diagram:**

The Entity-Relationship (ER) model was originally proposed by Peter in 1976 [Chen76] as a way to unify the network and relational database views. Simply stated the ER model is a conceptual data model that views the real world as entities and relationships. A basic component of the model is the Entity-Relationship diagram which is used to visually represent data objects. Since Chen wrote his paper the model has been extended and today it is commonly used for database design for the database designer, the utility of the ER model is:

* It maps well to the relational model. The constructs used in the ER model can easily be transformed into relational tables.
* It is simple and easy to understand with a minimum of training. Therefore, the model can be used by the database designer to communicate the design to the end user.
* In addition, the model can be used as a design plan by the database developer to implement a data model in specific database management software.

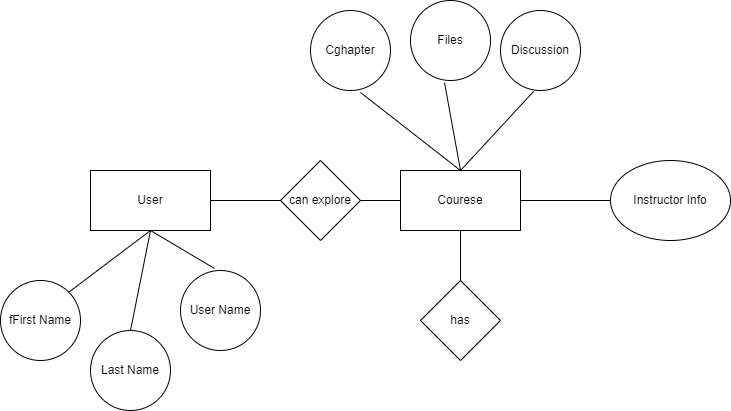
**7.1 ER Notation**

There is no standard for representing data objects in ER diagrams. Each modeling methodology uses its own notation. The original notation used by Chen is widely used in academics texts and journals but rarely seen in either CASE tools or publications by non-academics. Today, there are a number of notations used; among the more common are Bachman, crow's foot, and IDEFIX.

All notational styles represent entities as rectangular boxes and relationships as lines connecting boxes. Each style uses a special set of symbols to represent the cardinality of a connection. The notation used in this document is from Martin. The symbols used for the basic ER constructs are:

* **Entities** are represented by labeled rectangles. The label is the name of the entity. Entity names should be singular nouns.
* **Relationships** are represented by a solid line connecting two entities. The name of the relationship is written above the line. Relationship names should be verbs
* **Attributes**, when included, are listed inside the entity rectangle. Attributes which are identifiers are underlined. Attribute names should be singular nouns.
* **Cardinality** of many is represented by a line ending in a crow's foot. If the crow's foot is omitted, the cardinality is one.

**Existence** is represented by placing a circle or a perpendicular bar on the line. Mandatory existence is shown by the bar (looks like a 1) next to the entity for an instance is required. Optional existence is shown by placing a circle next to the entity that is optional.



**BI 9**

**Implementation and System Testing**

After all phase have been perfectly done, the system will be implemented to the server and the system can be used.

**8.1 System Testing**

The goal of the system testing process was to determine all faults in our project.The program was subjected to a set of test inputs and many explanations were made and based on these explanations it will be decided whether the program behaves as expected or not. Our Project went through two levels of testing

1. Unit testing

2. Integration testing

**UNIT TESTING**

Unit testing is commenced when a unit has been created and effectively reviewed .In order to test a single module we need to provide a complete environment i.e. besides the section we would require

* The procedures belonging to other units that the unit under test calls
* Non local data structures that module accesses
* A procedure to call the functions of the unit under test with appropriate parameters

**1. Test for the admin module**

* **Testing login form-** This form is used for log in of administrator of the system. In this form we enter the username and password if both are correct administration page will open otherwise if any of data is wrong it will get redirected back to the login page and again ask the details.

**INTEGRATION TESTING**

In the Integration testing we test various combination of the project module by providing its features.

The primary objective is to test the module interfaces in order to confirm that no errors are occurring when one module invokes the other module.

**BI 10**

**Valuation**

**Features:**

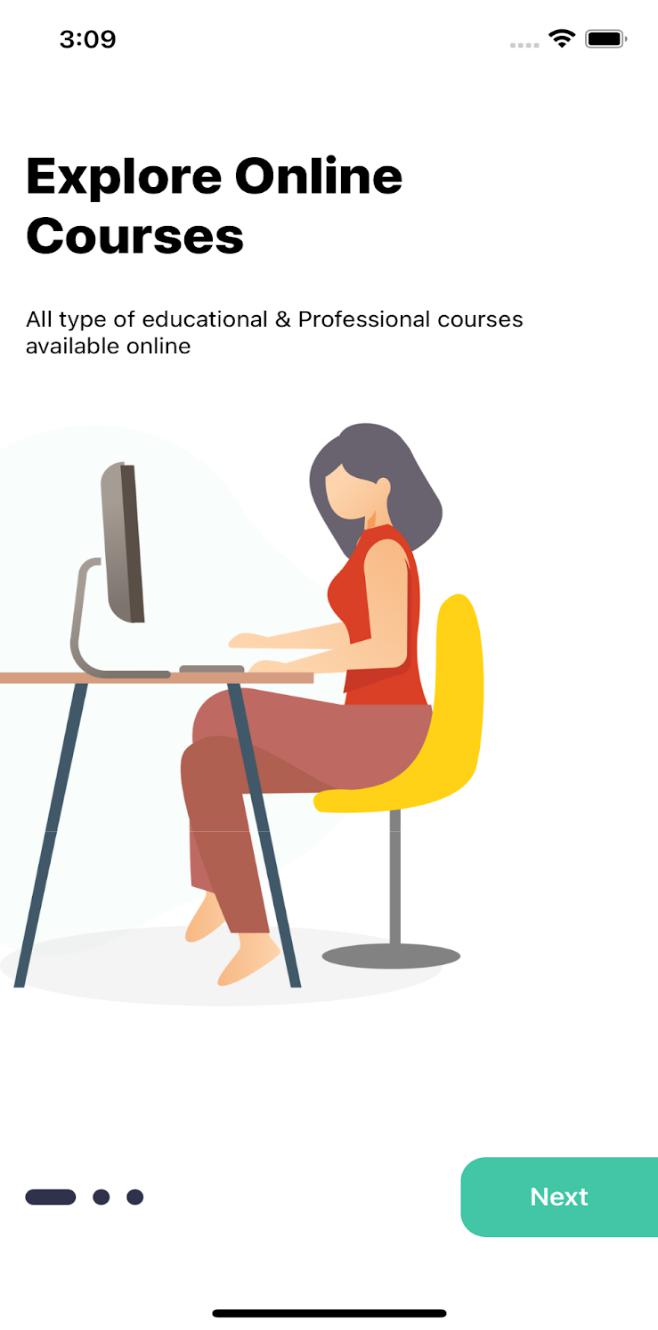
These are Features of the application that help the user

**9.1 Walkthrough Screen :**

****

Walkthrough screens give brief introduction of application

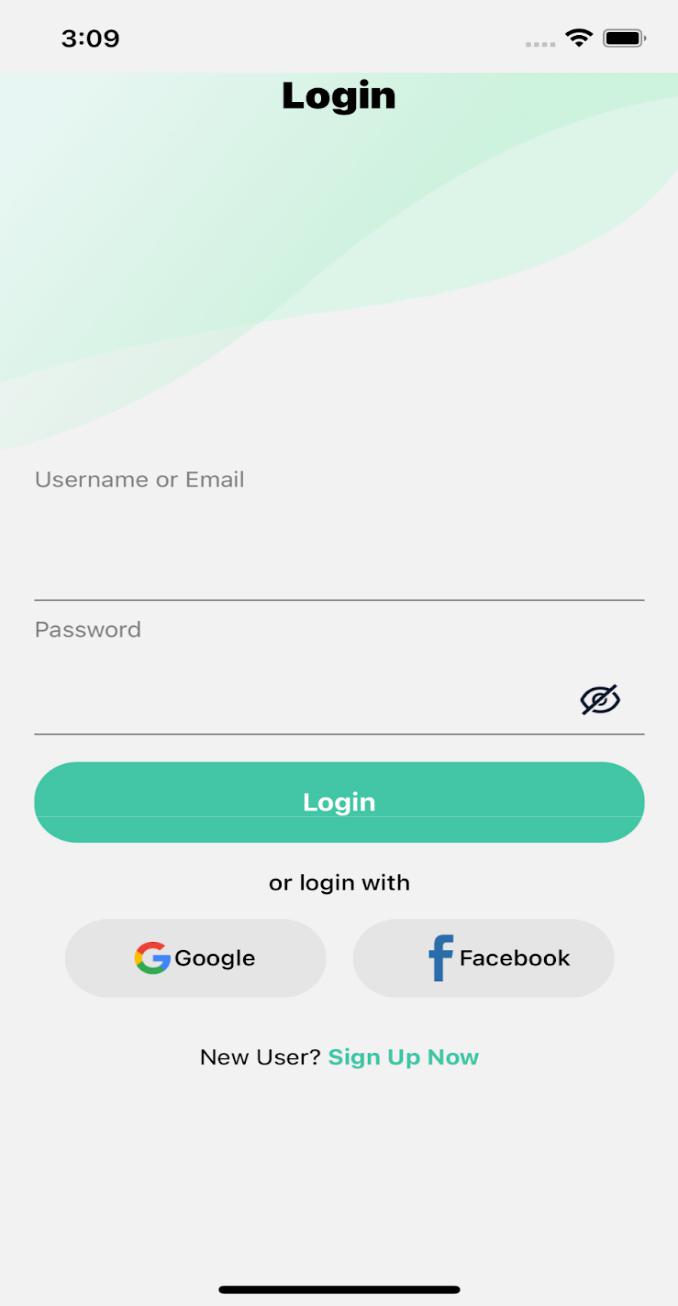
* This is App Intro Slider Screen
* That Slide and show Intro of the App



|  |
| --- |
| import React, {memo, useRef} from 'react'; |
|  | import {Animated, SafeAreaView, StatusBar, View} from 'react-native'; |
|  | import {Constant} from '../../config'; |
|  | import {COLORS, selectedTheme, SIZES} from '../../config/Themes'; |
|  | import {NavigationDataTypes} from '../../models'; |
|  | import RenderItems from './Renderitem'; |
|  | import styles from './style'; |
|  |  |
|  | const onBoardings = Constant.walkthrough; |
|  |  |
|  | const OnboardingScreen = (props: NavigationDataTypes) => { |
|  | const {navigation} = props; |
|  | const [completed, setCompleted] = React.useState(false); |
|  |  |
|  | const flatlistRef = useRef<any | null>(null); |
|  |  |
|  | const scrollToIndex = (data: any) => { |
|  | flatlistRef.current.scrollToIndex({animated: true, index: data + 1}); |
|  | }; |
|  |  |
|  | const scrollX = new Animated.Value(0); |
|  |  |
|  | React.useEffect(() => { |
|  | scrollX.addListener(({value}) => { |
|  | if (Math.floor(value / SIZES.width) === onBoardings.length - 1) { |
|  | setCompleted(true); |
|  | } |
|  | }); |
|  |  |
|  | return () => scrollX.removeAllListeners(); |
|  | }, []); |
|  |  |
|  | const renderDots = () => { |
|  | const dotPosition = Animated.divide(scrollX, SIZES.width); |
|  |  |
|  | return ( |
|  | <View style={styles(selectedTheme).dotsContainer}> |
|  | <StatusBar backgroundColor={COLORS.additionalColor13} barStyle={'dark-content'} /> |
|  | {onBoardings.map((item, index) => { |
|  | const dotSize = dotPosition.interpolate({ |
|  | inputRange: [index - 1, index, index + 1], |
|  | outputRange: [10, 17, 10], |
|  | extrapolate: 'clamp', |
|  | }); |
|  |  |
|  | return ( |
|  | <Animated.View |
|  | key={index} |
|  | style={[ |
|  | styles(selectedTheme).dot, |
|  | { |
|  | width: dotSize, |
|  | height: dotSize, |
|  | }, |
|  | ]} |
|  | /> |
|  | ); |
|  | })} |
|  | </View> |
|  | ); |
|  | }; |
|  |  |
|  | return ( |
|  | <SafeAreaView style={styles(selectedTheme).container}> |
|  | <View> |
|  | <Animated.FlatList |
|  | ref={flatlistRef} |
|  | horizontal |
|  | pagingEnabled={true} |
|  | data={onBoardings} |
|  | showsHorizontalScrollIndicator={false} |
|  | keyExtractor={(item, index) => item + index.toString()} |
|  | renderItem={({item}) => ( |
|  | <RenderItems |
|  | item={item} |
|  | navigation={navigation} |
|  | nextdata={() => { |
|  | scrollToIndex(item.id); |
|  | }} |
|  | /> |
|  | )} |
|  | onScroll={e => scrollX.setValue(e.nativeEvent.contentOffset.x)} |
|  | /> |
|  | <View style={styles(selectedTheme).dotsRootContainer}> |
|  | {renderDots()} |
|  | </View> |
|  | </View> |
|  | </SafeAreaView> |
|  | ); |
|  | }; |
|  |  |
|  | export default memo(OnboardingScreen); |

**9.2. Login Screen :**

* Login with credentials (username/email,password)
* Google Login
* Facebook Login



|  |
| --- |
| import React, {memo} from 'react'; |
|  | import { |
|  | View, |
|  | Text, |
|  | TextInput, |
|  | TouchableOpacity, |
|  | Image, |
|  | SafeAreaView, |
|  | ImageBackground, |
|  | ScrollView, |
|  | } from 'react-native'; |
|  | import {selectedTheme} from '../../config/Themes'; |
|  | import { |
|  | buttons, |
|  | form, |
|  | Icon, |
|  | Images, |
|  | RouteScreens, |
|  | screensData, |
|  | } from '../../config'; |
|  | import styles from './style'; |
|  |  |
|  | interface loginpageprops { |
|  | navigation: any; |
|  | visible: boolean; |
|  | setvisibe: React.Dispatch<React.SetStateAction<boolean>>; |
|  | isvalidEmail: boolean; |
|  | isvalidPassword: boolean; |
|  | Emailinpute: (text: string) => void; |
|  | EnterPassword: (text: string) => void; |
|  | isEnabled: boolean; |
|  | submit: any; |
|  | signIn: any; |
|  | Facebooklogin: any; |
|  | } |
|  |  |
|  | const LoginScreen = (props: loginpageprops) => { |
|  | const { |
|  | navigation, |
|  | visible, |
|  | setvisibe, |
|  | isvalidEmail, |
|  | isvalidPassword, |
|  | Emailinpute, |
|  | EnterPassword, |
|  | isEnabled, |
|  | signIn, |
|  | Facebooklogin, |
|  | submit, |
|  | } = props; |
|  |  |
|  | return ( |
|  | <SafeAreaView style={styles(selectedTheme).mainContainer}> |
|  | <ImageBackground |
|  | imageStyle={styles(selectedTheme).imageBackgroundStyle} |
|  | source={selectedTheme.name == 'light' ? Images.BG : Images.BG\_DARK} |
|  | style={styles(selectedTheme).bgImage}> |
|  | <Text style={styles(selectedTheme).heading}>{screensData.LOGIN}</Text> |
|  | </ImageBackground> |
|  | <ScrollView |
|  | style={styles(selectedTheme).container} |
|  | showsVerticalScrollIndicator={false}> |
|  | <View style={styles(selectedTheme).input}> |
|  | <Text style={styles(selectedTheme).formText}> |
|  | {form.USERNAMEOREMAIL} |
|  | </Text> |
|  | <View style={styles(selectedTheme).inputContainer}> |
|  | <TextInput |
|  | keyboardType={'email-address'} |
|  | onChangeText={Emailinpute} |
|  | style={styles(selectedTheme).inputText} |
|  | /> |
|  | </View> |
|  | {!isvalidEmail ? ( |
|  | <Text style={styles(selectedTheme).invalidText}> |
|  | {form.validation.EMAILVALIDATION} |
|  | </Text> |
|  | ) : null} |
|  | </View> |
|  | <View style={styles(selectedTheme).input}> |
|  | <Text style={styles(selectedTheme).formText}>{form.PASSSWORD}</Text> |
|  | <View style={styles(selectedTheme).inputContainer}> |
|  | <TextInput |
|  | style={styles(selectedTheme).inputText} |
|  | secureTextEntry={!visible} |
|  | onChangeText={EnterPassword} |
|  | /> |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).eyeContainer} |
|  | onPress={() => (visible ? setvisibe(false) : setvisibe(true))}> |
|  | <Image |
|  | source={visible ? Icon.EYE : Icon.EYE\_CLOSE} |
|  | style={styles(selectedTheme).eyeIcon} |
|  | /> |
|  | </TouchableOpacity> |
|  | </View> |
|  | {!isvalidPassword ? ( |
|  | <Text style={styles(selectedTheme).invalidText}> |
|  | {form.validation.PASSWORDVALIDATION} |
|  | </Text> |
|  | ) : null} |
|  | </View> |
|  |  |
|  | <TouchableOpacity style={styles(selectedTheme).button} onPress={submit}> |
|  | <Text style={styles(selectedTheme).buttonText}>{buttons.LOGIN}</Text> |
|  | </TouchableOpacity> |
|  |  |
|  | <Text style={styles(selectedTheme).text}> |
|  | {screensData.loginPage.OR\_LOGIN\_WITH} |
|  | </Text> |
|  |  |
|  | <View style={styles(selectedTheme).socialButtonContainer}> |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).socialButton} |
|  | onPress={() => signIn()}> |
|  | <Image source={Icon.GOOGLE} style={styles(selectedTheme).icon} /> |
|  | <Text style={styles(selectedTheme).socialButtonText}> |
|  | {buttons.GOOGLE} |
|  | </Text> |
|  | </TouchableOpacity> |
|  |  |
|  | <TouchableOpacity |
|  | onPress={() => Facebooklogin()} |
|  | style={styles(selectedTheme).socialButton}> |
|  | <Image source={Icon.FACEBOOK} style={styles(selectedTheme).icon} /> |
|  | <Text style={styles(selectedTheme).socialButtonText}> |
|  | {buttons.FACEBOOK} |
|  | </Text> |
|  | </TouchableOpacity> |
|  | </View> |
|  |  |
|  | <View style={styles(selectedTheme).buttonsContainer}> |
|  | <Text style={styles(selectedTheme).newUserText}> |
|  | {screensData.loginPage.NEW\_USER\_} |
|  | </Text> |
|  |  |
|  | <TouchableOpacity |
|  | onPress={() => navigation.navigate(RouteScreens.REGISTERMODEL)}> |
|  | <Text style={styles(selectedTheme).signupText}> |
|  | {buttons.SIGNUPNOW} |
|  | </Text> |
|  | </TouchableOpacity> |
|  | </View> |
|  | </ScrollView> |
|  | </SafeAreaView> |
|  | ); |
|  | }; |
|  | export default memo(LoginScreen); |

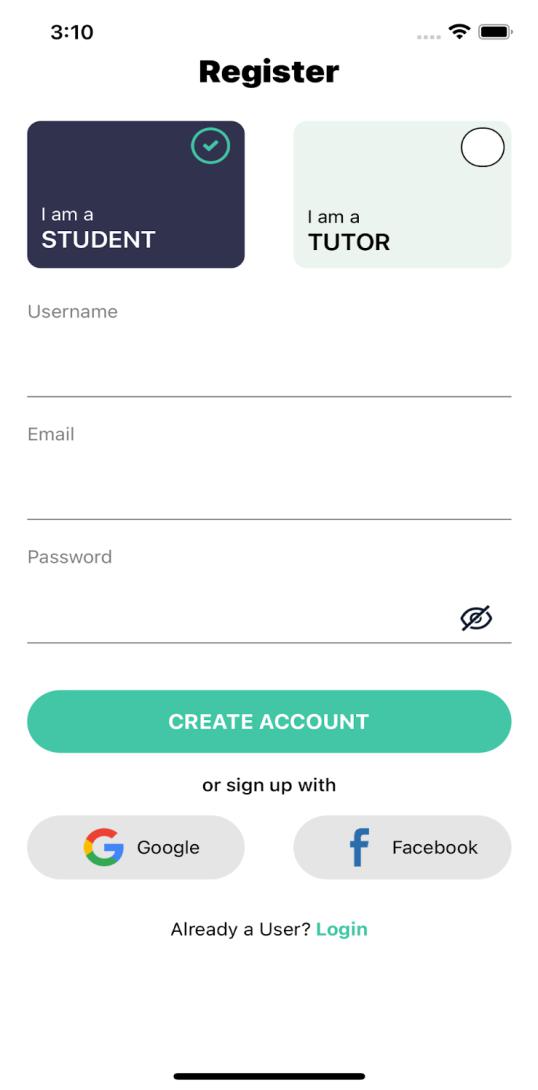
**9.3. Register Screen :**

* Users can register as a Organization or Industry
* Users can register by filling all the input fields

Here is some other and quick way to register in this

Application :

* Register via Google
* Register via Facebook

****

|  |
| --- |
| import React, {useRef} from 'react'; |
|  | import { |
|  | View, |
|  | Text, |
|  | Image, |
|  | TextInput, |
|  | TouchableOpacity, |
|  | ScrollView, |
|  | StatusBar, |
|  | Animated, |
|  | SafeAreaView, |
|  | } from 'react-native'; |
|  | import {buttons, form, Icon, RouteScreens, screensData} from '../../config'; |
|  | import {COLORS, selectedTheme} from '../../config/Themes'; |
|  | import styles from './style'; |
|  |  |
|  | interface Registerprops { |
|  | navigation: any; |
|  | visible: boolean; |
|  | setvisibe: React.Dispatch<React.SetStateAction<boolean>>; |
|  | selected: any; |
|  | setselected: any; |
|  | isvalidEmail: boolean; |
|  | isvalidPassword: boolean; |
|  | isvalidusername: boolean; |
|  | Emailinpute: (text: string) => void; |
|  | EnterPassword: (text: string) => void; |
|  | EnterUserName: (text: string) => void; |
|  | submit: any; |
|  | register: { |
|  | id: number; |
|  | label: string; |
|  | }[]; |
|  | defaultitem: any; |
|  | setdefaultitem: any; |
|  | GoogleSignin: () => Promise<void>; |
|  | Facebooklogin: () => void; |
|  | } |
|  |  |
|  | const RegisterScreen = (props: Registerprops) => { |
|  | const { |
|  | navigation, |
|  | visible, |
|  | setvisibe, |
|  | selected, |
|  | setselected, |
|  | isvalidEmail, |
|  | isvalidPassword, |
|  | isvalidusername, |
|  | Emailinpute, |
|  | EnterPassword, |
|  | EnterUserName, |
|  | register, |
|  | defaultitem, |
|  | setdefaultitem, |
|  | GoogleSignin, |
|  | Facebooklogin, |
|  | submit, |
|  | } = props; |
|  |  |
|  | const animationValue = useRef(new Animated.Value(0)).current; |
|  | const scaleValue = useRef(0); |
|  |  |
|  | const NEWButton = Animated.createAnimatedComponent(TouchableOpacity); |
|  |  |
|  | const runAnimationOnClick = () => { |
|  | scaleValue.current = scaleValue.current === 0 ? 1 : 0; |
|  | Animated.spring(animationValue, { |
|  | toValue: scaleValue.current, |
|  | delay: 1, |
|  | friction: 0.5, |
|  | useNativeDriver: true, |
|  | }).start(); |
|  | }; |
|  |  |
|  | return ( |
|  | <SafeAreaView style={styles(selectedTheme).mainConatainer}> |
|  | <StatusBar backgroundColor={'rgba(0,0,0,0)'} barStyle={'dark-content'} /> |
|  |  |
|  | <Text style={styles(selectedTheme).heading}>{screensData.REGISTER}</Text> |
|  |  |
|  | <ScrollView style={styles(selectedTheme).flex}> |
|  | <View style={styles(selectedTheme).boxContainer}> |
|  | {register.map((item, index) => { |
|  | return ( |
|  | <NEWButton |
|  | key={index} |
|  | onPress={() => { |
|  | setdefaultitem(item.id), runAnimationOnClick(); |
|  | }} |
|  | style={[ |
|  | styles(selectedTheme).box, |
|  | { |
|  | backgroundColor: |
|  | defaultitem == item.id |
|  | ? selectedTheme.backgroundblueNblack |
|  | : selectedTheme.backgroundgray10Ngray70, |
|  | borderColor: |
|  | defaultitem == item.id |
|  | ? selectedTheme.borderColor1 |
|  | : selectedTheme.borderColor1, |
|  | }, |
|  | ]}> |
|  | <Animated.View |
|  | style={[ |
|  | styles(selectedTheme).checkedContainer, |
|  | { |
|  | transform: |
|  | defaultitem == item.id |
|  | ? [ |
|  | { |
|  | translateX: animationValue.interpolate({ |
|  | inputRange: [0, 1], |
|  | outputRange: [10, 15], |
|  | }), |
|  | }, |
|  | ] |
|  | : [], |
|  | }, |
|  | ]}> |
|  | {defaultitem == item.id ? ( |
|  | <Image |
|  | source={Icon.CHECKBOX\_ON\_DARK} |
|  | style={styles(selectedTheme).icons} |
|  | /> |
|  | ) : ( |
|  | <View style={styles(selectedTheme).icons}></View> |
|  | )} |
|  | </Animated.View> |
|  | <Text |
|  | style={{ |
|  | color: defaultitem == item.id ? COLORS.white : COLORS.black, |
|  | }}> |
|  | {screensData.register.I\_AM\_A}{' '} |
|  | </Text> |
|  | <Text |
|  | style={[ |
|  | styles(selectedTheme).labelText, |
|  | { |
|  | color: |
|  | defaultitem == item.id ? COLORS.white : COLORS.black, |
|  | }, |
|  | ]}> |
|  | {item.label} |
|  | </Text> |
|  | </NEWButton> |
|  | ); |
|  | })} |
|  | </View> |
|  | <View style={styles(selectedTheme).container}> |
|  | <View style={styles(selectedTheme).input}> |
|  | <Text style={styles(selectedTheme).formText}>{form.USER}</Text> |
|  | <View style={styles(selectedTheme).inputContainer}> |
|  | <TextInput |
|  | style={styles(selectedTheme).inputText} |
|  | onChangeText={EnterUserName} |
|  | /> |
|  | </View> |
|  | {!isvalidusername ? ( |
|  | <Text style={styles(selectedTheme).invalidText}> |
|  | {form.validation.USERNAME} |
|  | </Text> |
|  | ) : null} |
|  | </View> |
|  |  |
|  | <View style={styles(selectedTheme).input}> |
|  | <Text style={styles(selectedTheme).formText}>{form.EMAIL}</Text> |
|  | <View style={styles(selectedTheme).inputContainer}> |
|  | <TextInput |
|  | keyboardType={'email-address'} |
|  | onChangeText={Emailinpute} |
|  | style={styles(selectedTheme).inputText} |
|  | /> |
|  | </View> |
|  | {!isvalidEmail ? ( |
|  | <Text style={styles(selectedTheme).invalidText}> |
|  | {form.validation.EMAILVALIDATION} |
|  | </Text> |
|  | ) : null} |
|  | </View> |
|  |  |
|  | <View style={styles(selectedTheme).input}> |
|  | <Text style={styles(selectedTheme).formText}>{form.PASSSWORD}</Text> |
|  | <View style={styles(selectedTheme).inputContainer}> |
|  | <TextInput |
|  | style={styles(selectedTheme).inputText} |
|  | secureTextEntry={!visible} |
|  | onChangeText={EnterPassword} |
|  | /> |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).eyeContainer} |
|  | onPress={() => (visible ? setvisibe(false) : setvisibe(true))}> |
|  | <Image |
|  | source={visible ? Icon.EYE : Icon.EYE\_CLOSE} |
|  | style={styles(selectedTheme).eyeIcon} |
|  | /> |
|  | </TouchableOpacity> |
|  | </View> |
|  | {!isvalidPassword ? ( |
|  | <Text style={styles(selectedTheme).invalidText}> |
|  | {form.validation.PASSWORDVALIDATION} |
|  | </Text> |
|  | ) : null} |
|  | </View> |
|  |  |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).button} |
|  | onPress={submit}> |
|  | <Text style={styles(selectedTheme).buttonText}> |
|  | {buttons.CREATE\_ACCOUNT} |
|  | </Text> |
|  | </TouchableOpacity> |
|  |  |
|  | <Text style={styles(selectedTheme).text}> |
|  | {screensData.register.OR\_SIGNUP\_WITH} |
|  | </Text> |
|  |  |
|  | <View style={styles(selectedTheme).socialButtonContainer}> |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).socialButton} |
|  | onPress={() => GoogleSignin()}> |
|  | <Image source={Icon.GOOGLE} style={styles(selectedTheme).icon} /> |
|  | <Text style={styles(selectedTheme).socialButtonText}> |
|  | {buttons.GOOGLE} |
|  | </Text> |
|  | </TouchableOpacity> |
|  |  |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).socialButton} |
|  | onPress={() => Facebooklogin()}> |
|  | <Image |
|  | source={Icon.FACEBOOK} |
|  | style={styles(selectedTheme).icon} |
|  | /> |
|  | <Text style={styles(selectedTheme).socialButtonText}> |
|  | {buttons.FACEBOOK} |
|  | </Text> |
|  | </TouchableOpacity> |
|  | </View> |
|  |  |
|  | <View style={styles(selectedTheme).buttonContainer}> |
|  | <Text style={styles(selectedTheme).newUserText}> |
|  | {screensData.register.ALREDY\_A\_USER} |
|  | </Text> |
|  |  |
|  | <TouchableOpacity |
|  | onPress={() => navigation.navigate(RouteScreens.LOGINMODEL)}> |
|  | <Text style={styles(selectedTheme).loginText}> |
|  | {screensData.register.LOGIN} |
|  | </Text> |
|  | </TouchableOpacity> |
|  | </View> |
|  | </View> |
|  | </ScrollView> |
|  | </SafeAreaView> |
|  | ); |
|  | }; |
|  | export default RegisterScreen; |

**9.4 Category Screen :**

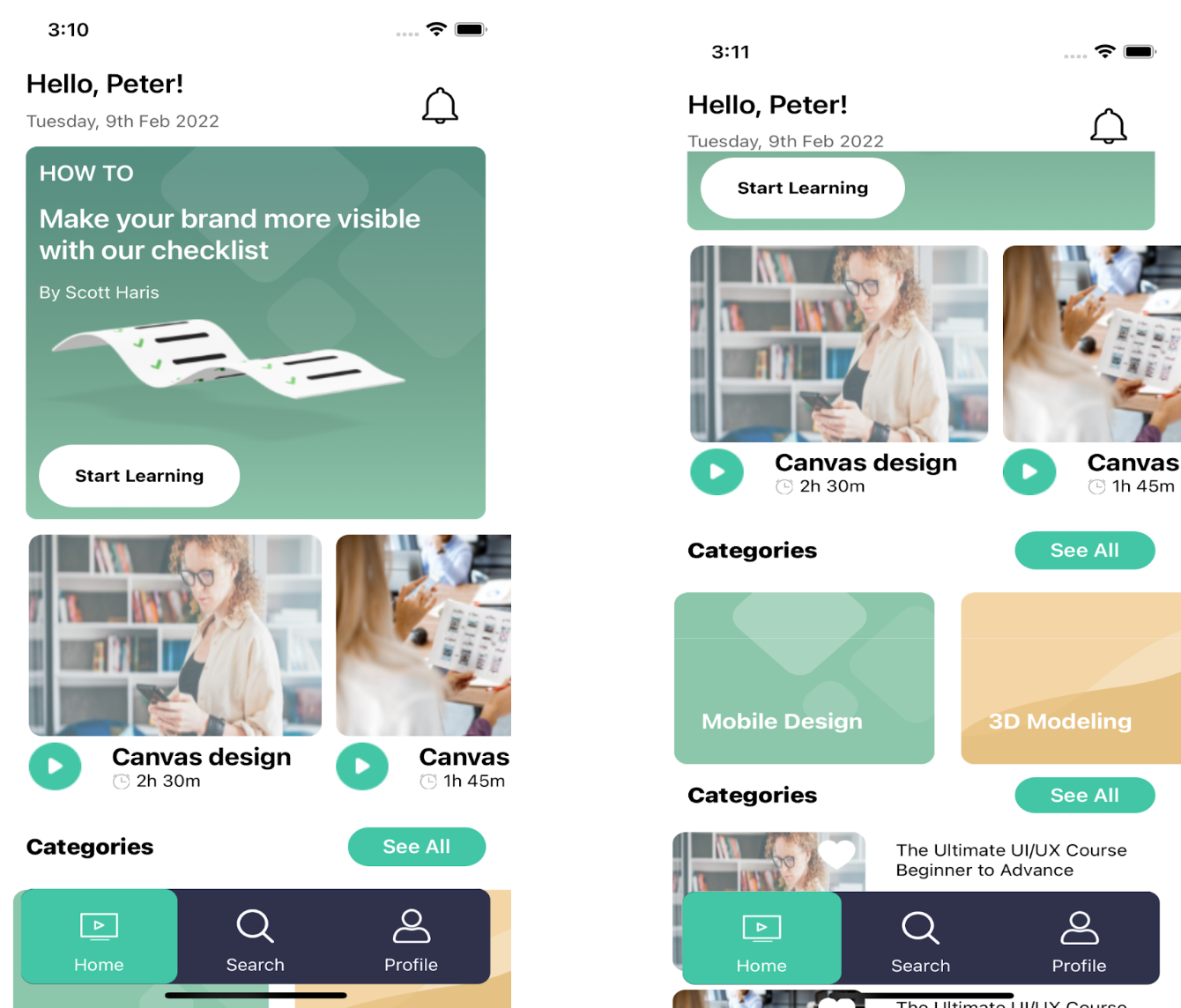
* Users can choose categories and navigate according to selected category

****

|  |
| --- |
| import React, {memo} from 'react'; |
|  | import { |
|  | FlatList, |
|  | Image, |
|  | SafeAreaView, |
|  | Text, |
|  | TouchableOpacity, |
|  | View, |
|  | } from 'react-native'; |
|  | import {selectedTheme} from '../../config/Themes'; |
|  | import {Icon, Constant, screensData, buttons, RouteScreens} from '../../config'; |
|  | import Renderitem from './Renderitem'; |
|  | import styles from './style'; |
|  |  |
|  | interface CategoryProps { |
|  | navigation: any; |
|  | defaultitem: string; |
|  | setdefaultitem: React.Dispatch<React.SetStateAction<string>>; |
|  | defaulticon: undefined; |
|  | setdefaulticon: React.Dispatch<React.SetStateAction<undefined>>; |
|  | } |
|  |  |
|  | const CategoryScreen = (props: CategoryProps) => { |
|  | const {navigation, defaultitem, setdefaultitem, setdefaulticon, defaulticon} = |
|  | props; |
|  |  |
|  | return ( |
|  | <SafeAreaView style={styles(selectedTheme).mainConatiner}> |
|  | <View style={styles(selectedTheme).container}> |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).leftButton} |
|  | onPress={() => navigation.goBack()}> |
|  | <Image |
|  | source={Icon.BACK} |
|  | style={styles(selectedTheme).headerLeftIcon} |
|  | /> |
|  | </TouchableOpacity> |
|  | <Text style={styles(selectedTheme).text}> |
|  | {screensData.home.CATEGORIES} |
|  | </Text> |
|  |  |
|  | <View style={styles(selectedTheme).flatlistContainer}> |
|  | <FlatList |
|  | data={Constant.categories} |
|  | extraData={Constant.categories} |
|  | renderItem={({item, index}) => ( |
|  | <Renderitem |
|  | item={item} |
|  | index={index} |
|  | defaultitem={defaultitem} |
|  | setdefaultitem={setdefaultitem} |
|  | setdefaulticon={setdefaulticon} |
|  | /> |
|  | )} |
|  | numColumns={3} |
|  | keyExtractor={(\_, index) => index.toString()} |
|  | /> |
|  |  |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).button} |
|  | onPress={() => |
|  | navigation.navigate(RouteScreens.POWER BILISTINGMODEL, { |
|  | Header: defaultitem, |
|  | Icon: defaulticon, |
|  | }) |
|  | }> |
|  | <Text style={styles(selectedTheme).buttonText}> |
|  | {buttons.CONTINUE}{' '} |
|  | </Text> |
|  | </TouchableOpacity> |
|  | </View> |
|  | </View> |
|  | </SafeAreaView> |
|  | ); |
|  | }; |
|  |  |
|  | export default memo(CategoryScreen); |

**9.5 Home Screen :**

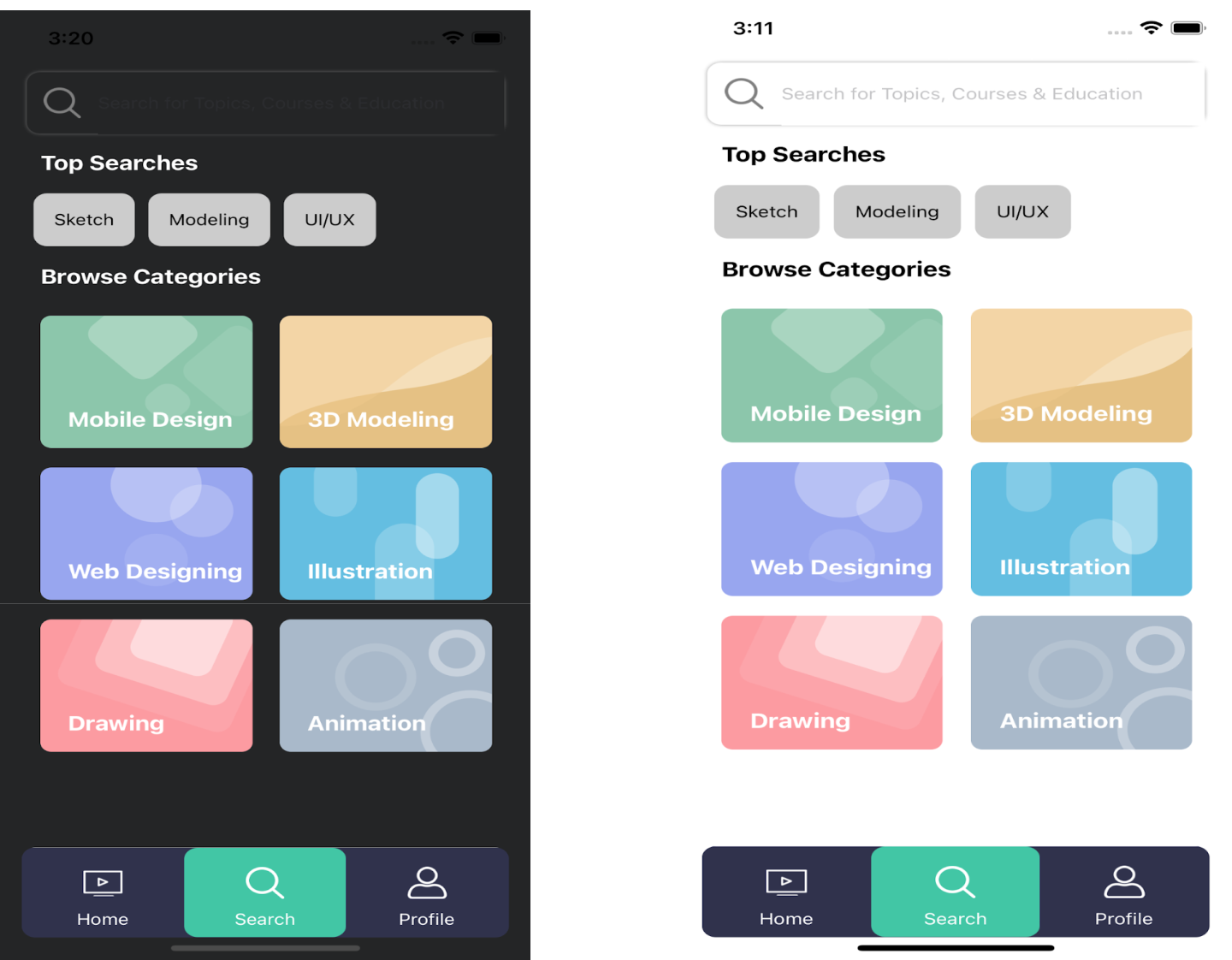
* This is the Home screen and there are several categories like(Popular Visualization and categories)
* User can navigate to any screen according to selected category
* This also contains a bottom tab navigation which opens Home , Search , Profile Screen
* At the right top right corner there is notification icon which opens Notification Screen



|  |
| --- |
| import React, {memo} from 'react'; |
|  | import { |
|  | View, |
|  | Text, |
|  | TouchableOpacity, |
|  | Image, |
|  | ScrollView, |
|  | ImageBackground, |
|  | Animated, |
|  | SafeAreaView, |
|  | } from 'react-native'; |
|  | import {selectedTheme} from '../../config/Themes'; |
|  | import {buttons, Icon, Images, RouteScreens, screensData} from '../../config'; |
|  | import RenderItem from './RenderItem'; |
|  | import styles from './style'; |
|  |  |
|  | interface Homepageprops { |
|  | navigation: any; |
|  | POWER BIlist1: { |
|  | id: number; |
|  | title: string; |
|  | duration: string; |
|  | thumbnail: any; |
|  | }[]; |
|  | category: { |
|  | id: number; |
|  | title: string; |
|  | thumbnail: any; |
|  | icon: any; |
|  | }[]; |
|  | POWER BIlist2: { |
|  | id: number; |
|  | title: string; |
|  | clsss\_level: string; |
|  | creted\_on: string; |
|  | duration: number; |
|  | instructor: string; |
|  | ratings: number; |
|  | price: number; |
|  | is\_favourite: boolean; |
|  | thumbnail: any; |
|  | }[]; |
|  | setseeall: any; |
|  | isfavourite: any; |
|  | setisfavourite: any; |
|  | } |
|  |  |
|  | const HomeScreen = (props: Homepageprops) => { |
|  | const { |
|  | navigation, |
|  | POWER BIlist1, |
|  | POWER BIlist2, |
|  | category, |
|  | setseeall, |
|  | isfavourite, |
|  | setisfavourite, |
|  | } = props; |
|  |  |
|  | const animation = new Animated.Value(0); |
|  |  |
|  | const runAnimationOnClick = () => { |
|  | Animated.loop( |
|  | Animated.sequence([ |
|  | Animated.timing(animation, { |
|  | toValue: -1, |
|  | duration: 100, |
|  | useNativeDriver: true, |
|  | }), |
|  | Animated.timing(animation, { |
|  | toValue: 1, |
|  | duration: 100, |
|  | useNativeDriver: true, |
|  | }), |
|  | Animated.timing(animation, { |
|  | toValue: 0, |
|  | duration: 100, |
|  | useNativeDriver: true, |
|  | }), |
|  | ]), |
|  | { |
|  | iterations: 4, |
|  | }, |
|  | ).start(); |
|  | }; |
|  |  |
|  | const rotateanimation = animation.interpolate({ |
|  | inputRange: [-1, 1, 2], |
|  | outputRange: ['-20deg', '20deg', '0deg'], |
|  | }); |
|  | return ( |
|  | <SafeAreaView style={styles(selectedTheme).container}> |
|  | <View style={styles(selectedTheme).mainSubContainer}> |
|  | <View style={styles(selectedTheme).headercontainer}> |
|  | <View style={styles(selectedTheme).nameAndDate}> |
|  | <Text style={styles(selectedTheme).text}> |
|  | {screensData.home.NAME} |
|  | </Text> |
|  | <Text style={styles(selectedTheme).subText}> |
|  | {screensData.home.DATE} |
|  | </Text> |
|  | </View> |
|  | <Animated.View |
|  | onTouchStart={runAnimationOnClick} |
|  | style={{ |
|  | transform: [{rotate: rotateanimation}], |
|  | }}> |
|  | <TouchableOpacity |
|  | onPress={() => navigation.navigate(RouteScreens.NOTIFICATIONTAB)}> |
|  | <Image |
|  | source={Icon.NOTIFICATION} |
|  | style={styles(selectedTheme).notifiactionicon} |
|  | /> |
|  | </TouchableOpacity> |
|  | </Animated.View> |
|  | </View> |
|  | <ScrollView |
|  | nestedScrollEnabled={true} |
|  | showsVerticalScrollIndicator={false} |
|  | style={styles(selectedTheme).mainScrollView}> |
|  | <ImageBackground |
|  | source={Images.FEATURED\_BG\_IMAGE} |
|  | imageStyle={styles(selectedTheme).backgroundImage} |
|  | style={styles(selectedTheme).subContainer}> |
|  | <Text style={styles(selectedTheme).howToText}> |
|  | {screensData.home.HOWTO} |
|  | </Text> |
|  | <Text style={styles(selectedTheme).subtextdata}> |
|  | {screensData.home.SUBPARAGRAPH} |
|  | </Text> |
|  | <Text style={styles(selectedTheme).author}> |
|  | {screensData.home.AUTHORNAME} |
|  | </Text> |
|  | <Image |
|  | source={Images.START\_REPRESENTATION} |
|  | style={styles(selectedTheme).bannerImage} |
|  | /> |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).learnButton} |
|  | onPress={() => navigation.navigate(RouteScreens.CATEGORYMODEL)}> |
|  | <Text style={styles(selectedTheme).startRepresentationText}> |
|  | {screensData.home.STARTLERNING} |
|  | </Text> |
|  | </TouchableOpacity> |
|  | </ImageBackground> |
|  | <ScrollView |
|  | horizontal |
|  | showsHorizontalScrollIndicator={false} |
|  | style={styles(selectedTheme).thirdContainer}> |
|  | {POWER BIlist1.map((item, index) => { |
|  | return ( |
|  | <TouchableOpacity |
|  | key={index} |
|  | onPress={() => |
|  | navigation.navigate(RouteScreens.POWER BIMODEL, { |
|  | Title: item.title, |
|  | }) |
|  | }> |
|  | <Image |
|  | source={item.thumbnail} |
|  | style={styles(selectedTheme).thumbnail} |
|  | /> |
|  | <View style={styles(selectedTheme).row}> |
|  | <Image |
|  | source={Icon.PLAY\_1} |
|  | style={styles(selectedTheme).playicon} |
|  | /> |
|  | <View style={styles(selectedTheme).textAndTimeContainer}> |
|  | <Text |
|  | style={styles(selectedTheme).thirdContainerTextTitle}> |
|  | {item.title} |
|  | </Text> |
|  | <View style={styles(selectedTheme).row}> |
|  | <Image |
|  | source={Icon.TIME} |
|  | style={styles(selectedTheme).timeIcon} |
|  | /> |
|  | <Text style={styles(selectedTheme).time}> |
|  | {' '} |
|  | {item.duration} |
|  | </Text> |
|  | </View> |
|  | </View> |
|  | </View> |
|  | </TouchableOpacity> |
|  | ); |
|  | })} |
|  | </ScrollView> |
|  |  |
|  | <View style={styles(selectedTheme).itemContainer}> |
|  | <Text style={styles(selectedTheme).thirdContainerTextTitle}> |
|  | {screensData.home.CATEGORIES} |
|  | </Text> |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).seeAllButton} |
|  | onPress={() => navigation.navigate(RouteScreens.CATEGORYMODEL)}> |
|  | <Text style={styles(selectedTheme).seeAllText}> |
|  | {buttons.SEEALL} |
|  | </Text> |
|  | </TouchableOpacity> |
|  | </View> |
|  |  |
|  | <ScrollView horizontal showsHorizontalScrollIndicator={false}> |
|  | {category.map((item, index) => { |
|  | return ( |
|  | <TouchableOpacity |
|  | key={index} |
|  | onPress={() => |
|  | navigation.navigate(RouteScreens.POWER BILISTINGMODEL, { |
|  | Header: item.title, |
|  | Icon: item.icon, |
|  | }) |
|  | } |
|  | style={styles(selectedTheme).item}> |
|  | <ImageBackground |
|  | source={item.thumbnail} |
|  | imageStyle={styles(selectedTheme).itemIcons} |
|  | style={styles(selectedTheme).icons}> |
|  | <Text style={[styles(selectedTheme).itemText]}> |
|  | {item.title} |
|  | </Text> |
|  | </ImageBackground> |
|  | </TouchableOpacity> |
|  | ); |
|  | })} |
|  | </ScrollView> |
|  |  |
|  | <View style={styles(selectedTheme).itemContainer}> |
|  | <Text style={styles(selectedTheme).thirdContainerTextTitle}> |
|  | {screensData.home.POPULARVISUALIZATION} |
|  | </Text> |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).seeAllButton} |
|  | onPress={() => setseeall(true)}> |
|  | <Text style={styles(selectedTheme).seeAllText}> |
|  | {buttons.SEEALL} |
|  | </Text> |
|  | </TouchableOpacity> |
|  | </View> |
|  | <ScrollView> |
|  | {POWER BIlist2.map((item, index) => { |
|  | return ( |
|  | <RenderItem |
|  | navigation={navigation} |
|  | key={index} |
|  | item={item} |
|  | index={index} |
|  | isfavourite={isfavourite} |
|  | setisfavourite={setisfavourite} |
|  | /> |
|  | ); |
|  | })} |
|  | </ScrollView> |
|  | </ScrollView> |
|  | </View> |
|  | </SafeAreaView> |
|  | ); |
|  | }; |
|  | export default memo(HomeScreen); |

**9.6 Search Screen:**

* This is the Search Screen which contains a search input field and Top searches suggestions and also categories
* User can navigate to a new screen according to the selected category

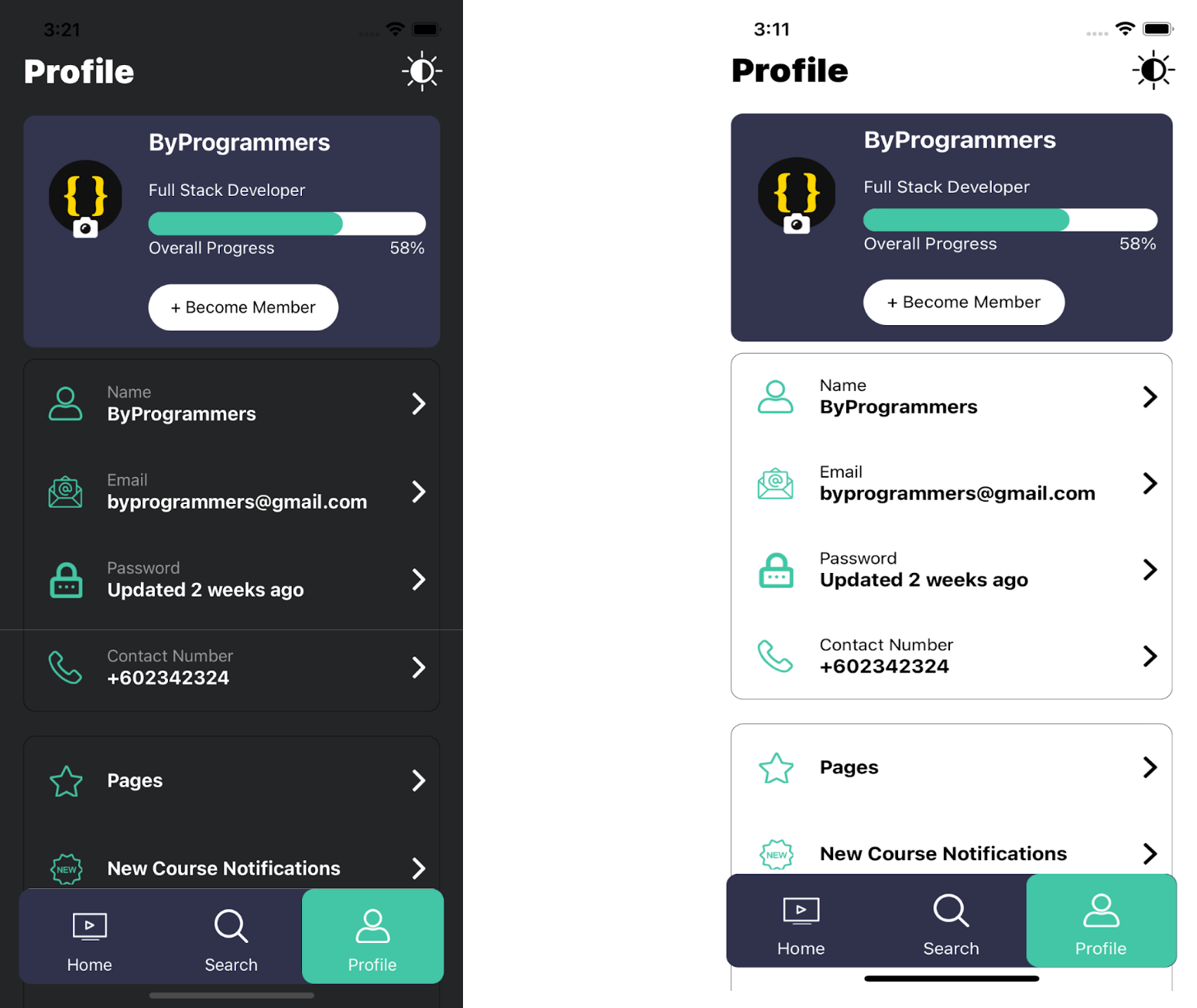


(Dark Theme) (Light Theme)

|  |
| --- |
| import React from 'react'; |
|  | import { |
|  | FlatList, |
|  | Image, |
|  | SafeAreaView, |
|  | ScrollView, |
|  | Text, |
|  | TextInput, |
|  | TouchableOpacity, |
|  | View, |
|  | } from 'react-native'; |
|  | import {selectedTheme} from '../../config/Themes'; |
|  | import {Icon, screensData, DummyData} from '../../config'; |
|  | import RenderItem from '../Home/RenderItem'; |
|  | import Renderitem from './Renderitem'; |
|  | import styles from './style'; |
|  |  |
|  | interface SearchScreenProps { |
|  | navigation: any; |
|  | visible: any; |
|  | setvisibe: any; |
|  | defaultitem: number; |
|  | setdefaultitem: React.Dispatch<React.SetStateAction<number>>; |
|  | SearchTexthandler: any; |
|  | isSearch: boolean; |
|  | searchText: string; |
|  | setisSearch: any; |
|  | resetsearch: any; |
|  | flatlistdata: { |
|  | id: number; |
|  | title: string; |
|  | thumbnail: any; |
|  | icon: any; |
|  | }[]; |
|  | DATA: any; |
|  | isfavourite: any; |
|  | setisfavourite: any; |
|  | } |
|  |  |
|  | const SearchScreen = (props: SearchScreenProps) => { |
|  | const { |
|  | navigation, |
|  | visible, |
|  | setvisibe, |
|  | defaultitem, |
|  | setdefaultitem, |
|  | SearchTexthandler, |
|  | isSearch, |
|  | searchText, |
|  | setisSearch, |
|  | resetsearch, |
|  | flatlistdata, |
|  | DATA, |
|  | isfavourite, |
|  | setisfavourite, |
|  | } = props; |
|  |  |
|  | const handleEmpty = () => { |
|  | return <Text> {screensData.search.NO\_DATA\_PRESENT}</Text>; |
|  | }; |
|  |  |
|  | return ( |
|  | <SafeAreaView style={styles(selectedTheme).mainContainer}> |
|  | <View style={styles(selectedTheme).container}> |
|  | <View style={styles(selectedTheme).searchBar}> |
|  | <TouchableOpacity> |
|  | <Image |
|  | source={Icon.SEARCH} |
|  | style={styles(selectedTheme).headerLeftIcon} |
|  | /> |
|  | </TouchableOpacity> |
|  |  |
|  | <TextInput |
|  | style={[{width: !isSearch ? '100%' : '85%'}]} |
|  | onChangeText={SearchTexthandler} |
|  | placeholder={screensData.search.PLACEHOLDER} |
|  | value={searchText} |
|  | placeholderTextColor={selectedTheme.textgray8Ngray4} |
|  | /> |
|  | {isSearch ? ( |
|  | <TouchableOpacity |
|  | onPress={() => { |
|  | setisSearch(false), resetsearch(Text); |
|  | }}> |
|  | <Image |
|  | source={Icon.CROSS} |
|  | style={[styles(selectedTheme).searchBarImage]} |
|  | /> |
|  | </TouchableOpacity> |
|  | ) : null} |
|  | </View> |
|  |  |
|  | {!isSearch ? ( |
|  | <View> |
|  | <Text style={styles(selectedTheme).text}> |
|  | {screensData.search.TOPSEARCHES} |
|  | </Text> |
|  | <ScrollView horizontal showsHorizontalScrollIndicator={false}> |
|  | <View style={styles(selectedTheme).topSearch}> |
|  | {DummyData.topSearches.map((item, index) => { |
|  | return ( |
|  | <View key={index} style={{flex: 1}}> |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).topSearchContainer} |
|  | onPress={() => SearchTexthandler(item.label)}> |
|  | <Text |
|  | key={index} |
|  | style={styles(selectedTheme).searchList}> |
|  | {item.label} |
|  | </Text> |
|  | </TouchableOpacity> |
|  | </View> |
|  | ); |
|  | })} |
|  | </View> |
|  | </ScrollView> |
|  |  |
|  | <FlatList |
|  | key={1} |
|  | data={flatlistdata} |
|  | extraData={flatlistdata} |
|  | showsVerticalScrollIndicator={false} |
|  | renderItem={({item, index}) => ( |
|  | <Renderitem item={item} index={index} navigation={navigation} /> |
|  | )} |
|  | numColumns={2} |
|  | keyExtractor={(\_, index) => index.toString()} |
|  | ListHeaderComponent={ |
|  | <Text style={styles(selectedTheme).text}> |
|  | {screensData.search.BROWSECATEGORIES} |
|  | </Text> |
|  | } |
|  | /> |
|  | </View> |
|  | ) : ( |
|  | <FlatList |
|  | data={DATA} |
|  | showsVerticalScrollIndicator={false} |
|  | ListEmptyComponent={handleEmpty} |
|  | keyExtractor={(item, index) => item + index.toString()} |
|  | renderItem={({item}) => ( |
|  | <RenderItem |
|  | item={item} |
|  | isfavourite={isfavourite} |
|  | setisfavourite={setisfavourite} |
|  | navigation={navigation} |
|  | /> |
|  | )} |
|  | /> |
|  | )} |
|  | </View> |
|  | </SafeAreaView> |
|  | ); |
|  | }; |
|  |  |
|  | export default SearchScreen; |

**9.7 Profile Screen :**

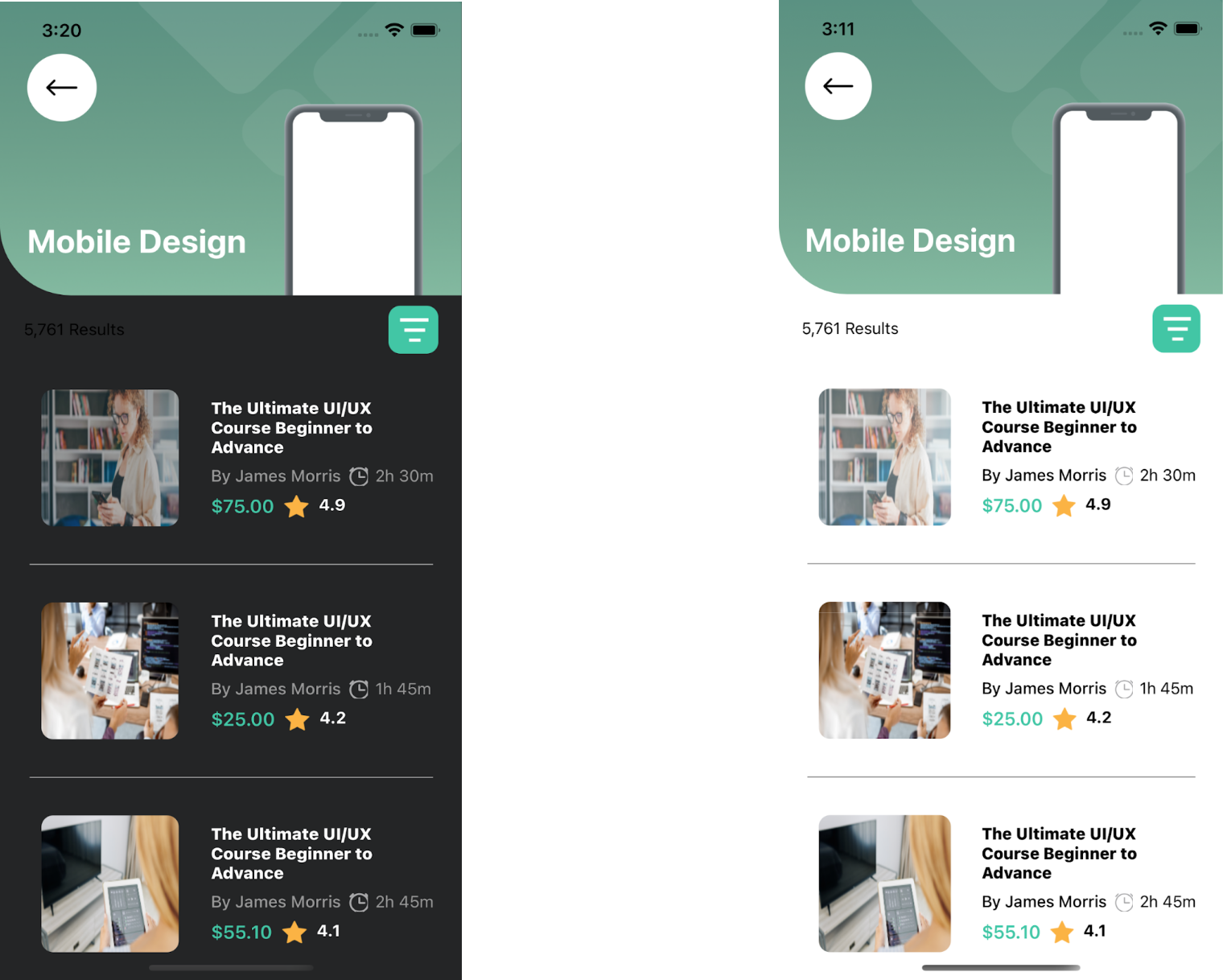
* This is the Profile screen which consists of a theme change Icon which change theme into light to dark and vice-versa
* This Screen shows the details of user
* It also consists of a change profile functionality
* After Pressing the Become member button it opens a Become membership Screen



|  |
| --- |
| import React, {memo, useState} from 'react'; |
|  | import { |
|  | Image, |
|  | Modal, |
|  | SafeAreaView, |
|  | ScrollView, |
|  | Switch, |
|  | Text, |
|  | TouchableOpacity, |
|  | TouchableWithoutFeedback, |
|  | View, |
|  | } from 'react-native'; |
|  | import {ProgressBar} from '../../components'; |
|  | import {COLORS, selectedTheme, changeTheme} from '../../config/Themes'; |
|  | import {buttons, Icon, screensData, DummyData} from '../../config'; |
|  | import styles from './style'; |
|  |  |
|  | interface profileprops { |
|  | navigation: any; |
|  | visible: any; |
|  | setvisibe: any; |
|  | setrender: any; |
|  | render: any; |
|  | modalVisible: any; |
|  | setModalVisible: any; |
|  | launchLibrary: any; |
|  | openCamara: any; |
|  | imageUriGallary: any; |
|  | dispachcall: any; |
|  | } |
|  | const ProfileScreen = (props: profileprops) => { |
|  | const { |
|  | navigation, |
|  | visible, |
|  | setvisibe, |
|  | setrender, |
|  | render, |
|  | modalVisible, |
|  | setModalVisible, |
|  | openCamara, |
|  | launchLibrary, |
|  | imageUriGallary, |
|  | dispachcall, |
|  | } = props; |
|  |  |
|  | return ( |
|  | <SafeAreaView style={styles(selectedTheme).mainContainer}> |
|  | <View style={styles(selectedTheme).mainSubContainer}> |
|  | <View style={styles(selectedTheme).headerContainer}> |
|  | <Text style={styles(selectedTheme).headerText}> |
|  | {screensData.Profile.NAME} |
|  | </Text> |
|  | <TouchableOpacity |
|  | onPress={() => { |
|  | setrender(!render), changeTheme(), dispachcall(); |
|  | }}> |
|  | <Image |
|  | source={Icon.SUN} |
|  | style={styles(selectedTheme).headerIcons} |
|  | /> |
|  | </TouchableOpacity> |
|  | </View> |
|  | <ScrollView |
|  | showsVerticalScrollIndicator={false} |
|  | style={{marginTop: 20}}> |
|  | <View style={styles(selectedTheme).subContainer}> |
|  | <View style={{flexDirection: 'row'}}> |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).imageContainer} |
|  | onPress={() => setModalVisible(true)}> |
|  | <Image |
|  | source={{uri: imageUriGallary}} |
|  | style={styles(selectedTheme).profileIcon} |
|  | /> |
|  | <View style={styles(selectedTheme).imageContainerView}> |
|  | <View style={styles(selectedTheme).imageViewContainer}> |
|  | <Image |
|  | source={Icon.CAMERA} |
|  | style={styles(selectedTheme).cameraIcon} |
|  | /> |
|  | </View> |
|  | </View> |
|  | </TouchableOpacity> |
|  | <View style={styles(selectedTheme).userHeaderDataContainer}> |
|  | <Text style={styles(selectedTheme).username}> |
|  | {screensData.instructor.NAME}{' '} |
|  | </Text> |
|  | <Text style={styles(selectedTheme).body}> |
|  | {screensData.instructor.DES} |
|  | </Text> |
|  | <ProgressBar |
|  | containerstyle={styles(selectedTheme).progressBar} |
|  | progress="90%" |
|  | /> |
|  | <View style={styles(selectedTheme).progressTextContainer}> |
|  | <Text style={styles(selectedTheme).progressText}> |
|  | {screensData.Profile.OVERALLPROGRESS} |
|  | </Text> |
|  | <Text style={styles(selectedTheme).progressText}> |
|  | {screensData.Profile.PERCENTAGE} |
|  | </Text> |
|  | </View> |
|  | </View> |
|  | </View> |
|  | <View> |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).learnButton} |
|  | onPress={() => navigation.navigate('MembershipModel')}> |
|  | <Text style={styles(selectedTheme).learnButtonText}> |
|  | {buttons.BECOMEMEMBER} |
|  | </Text> |
|  | </TouchableOpacity> |
|  | </View> |
|  | </View> |
|  | <View style={styles(selectedTheme).userDataContainer}> |
|  | {DummyData.UserData.map((item: any, index: any) => { |
|  | return ( |
|  | <View key={index}> |
|  | <View style={styles(selectedTheme).userContainer}> |
|  | <View style={styles(selectedTheme).subUserContainer}> |
|  | <Image |
|  | source={item.icon} |
|  | style={styles(selectedTheme).icon} |
|  | /> |
|  | <View style={styles(selectedTheme).textContainer}> |
|  | <Text style={styles(selectedTheme).label}> |
|  | {item.label} |
|  | </Text> |
|  | <Text style={styles(selectedTheme).value}> |
|  | {item.Value} |
|  | </Text> |
|  | </View> |
|  | </View> |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).rightImageContainer}> |
|  | <Image |
|  | source={Icon.RIGHT\_ARROW} |
|  | style={styles(selectedTheme).rightArrowIcon} |
|  | /> |
|  | </TouchableOpacity> |
|  | </View> |
|  | {index != 3 ? ( |
|  | <View style={styles(selectedTheme).itemSeperator} /> |
|  | ) : null} |
|  | </View> |
|  | ); |
|  | })} |
|  | </View> |
|  | <View style={styles(selectedTheme).userDataContainer}> |
|  | {DummyData.UserData2.map((item: any, index: any) => { |
|  | const [isEnabled, setIsEnabled] = useState(false); |
|  | const toggleSwitch = () => |
|  | setIsEnabled(previousState => !previousState); |
|  | return ( |
|  | <View key={index}> |
|  | <View style={styles(selectedTheme).userContainer}> |
|  | <View style={styles(selectedTheme).subUserContainer}> |
|  | <Image |
|  | source={item.icon} |
|  | style={styles(selectedTheme).icon} |
|  | /> |
|  | <View style={styles(selectedTheme).textContainer}> |
|  | <Text style={styles(selectedTheme).lableData}> |
|  | {item.label} |
|  | </Text> |
|  | </View> |
|  | </View> |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).rightImageContainer}> |
|  | {index == 0 ? ( |
|  | <Image |
|  | source={Icon.RIGHT\_ARROW} |
|  | style={styles(selectedTheme).rightArrowIcon} |
|  | /> |
|  | ) : ( |
|  | <Switch |
|  | trackColor={{false: COLORS.gray20, true: COLORS.additionalColor13}} |
|  | thumbColor={isEnabled ? COLORS.primary : COLORS.gray40} |
|  | ios\_backgroundColor="#3e3e3e" |
|  | onValueChange={toggleSwitch} |
|  | value={isEnabled} |
|  | /> |
|  | )} |
|  | </TouchableOpacity> |
|  | </View> |
|  | {index != 2 ? ( |
|  | <View style={styles(selectedTheme).itemSeperator} /> |
|  | ) : null} |
|  | </View> |
|  | ); |
|  | })} |
|  | </View> |
|  | </ScrollView> |
|  | <Modal |
|  | animationType="slide" |
|  | transparent={true} |
|  | visible={modalVisible} |
|  | onRequestClose={() => { |
|  | setModalVisible(!modalVisible); |
|  | }}> |
|  | <TouchableWithoutFeedback |
|  | onPress={() => setModalVisible(!modalVisible)}> |
|  | <View style={styles(selectedTheme).drawerContainer} /> |
|  | </TouchableWithoutFeedback> |
|  |  |
|  | <View style={styles(selectedTheme).modalView}> |
|  | <TouchableOpacity onPress={() => setModalVisible(!modalVisible)}> |
|  | <Image |
|  | source={Icon.CROSS} |
|  | style={styles(selectedTheme).modalIcon} |
|  | /> |
|  | <Text style={styles(selectedTheme).modalText}> |
|  | {buttons.CANCEL} |
|  | </Text> |
|  | </TouchableOpacity> |
|  |  |
|  | <TouchableOpacity onPress={() => openCamara()}> |
|  | <Image |
|  | source={Icon.CAMERA} |
|  | style={styles(selectedTheme).modalIcon} |
|  | /> |
|  | <Text style={styles(selectedTheme).modalText}> |
|  | {buttons.CAMERA} |
|  | </Text> |
|  | </TouchableOpacity> |
|  |  |
|  | <TouchableOpacity onPress={() => launchLibrary()}> |
|  | <Image |
|  | source={Icon.GALLERY} |
|  | style={styles(selectedTheme).modalIcon} |
|  | /> |
|  | <Text style={styles(selectedTheme).modalText}> |
|  | {buttons.GALLERY} |
|  | </Text> |
|  | </TouchableOpacity> |
|  | </View> |
|  | </Modal> |
|  | </View> |
|  | </SafeAreaView> |
|  | ); |
|  | }; |
|  |  |
|  | export default memo(ProfileScreen); |

**2.8 POWER BI Listing Screen :**

* This Screen contains various Visualization
* It contains a filter button that opens the filter modal that’s help the user to filter his choice of Visualization
* After Clicking the particular POWER BI it opens a POWER BI BI screen

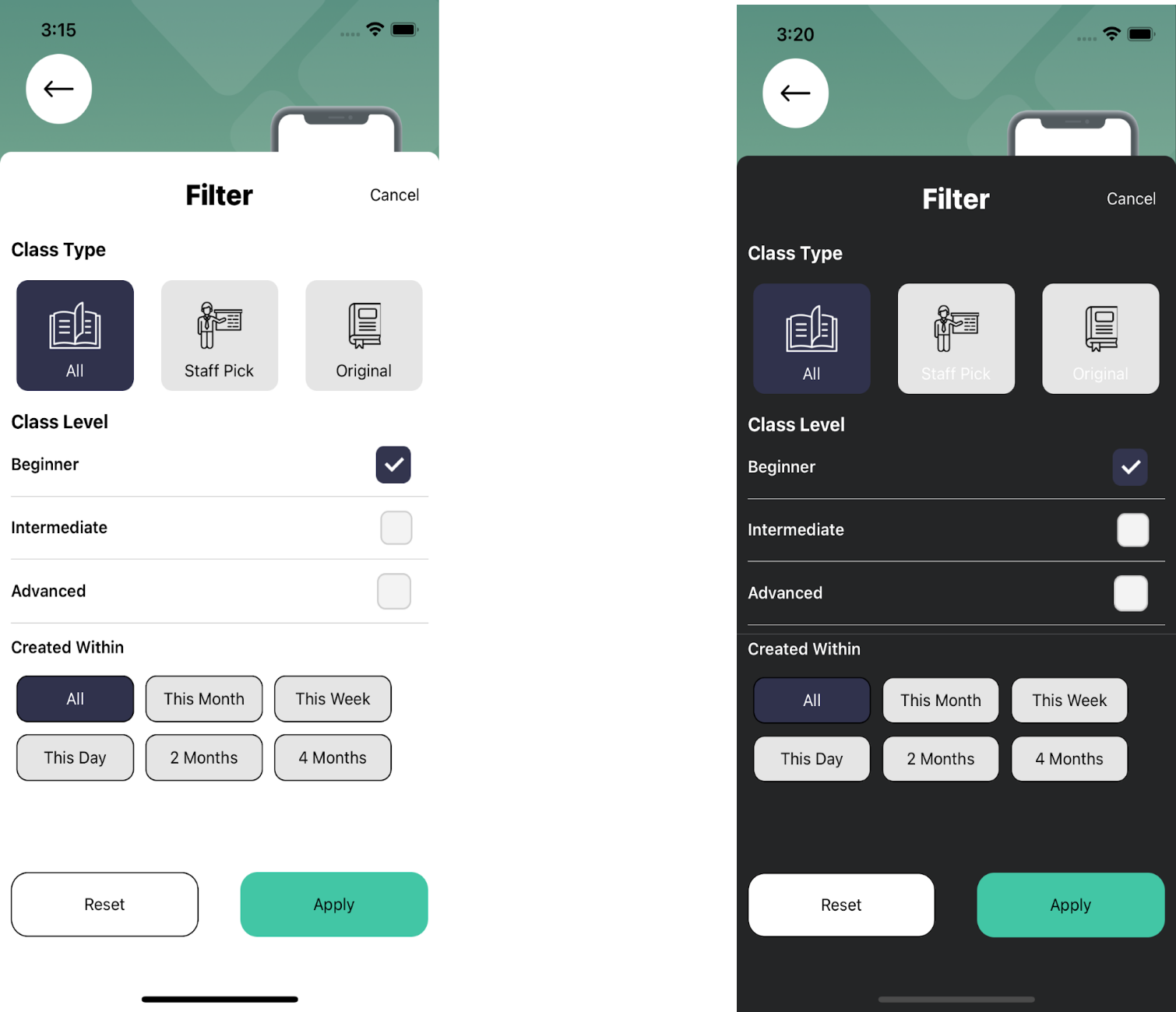


( Dark Theme ) (Light Theme)

**2.9 Filter Modal:**

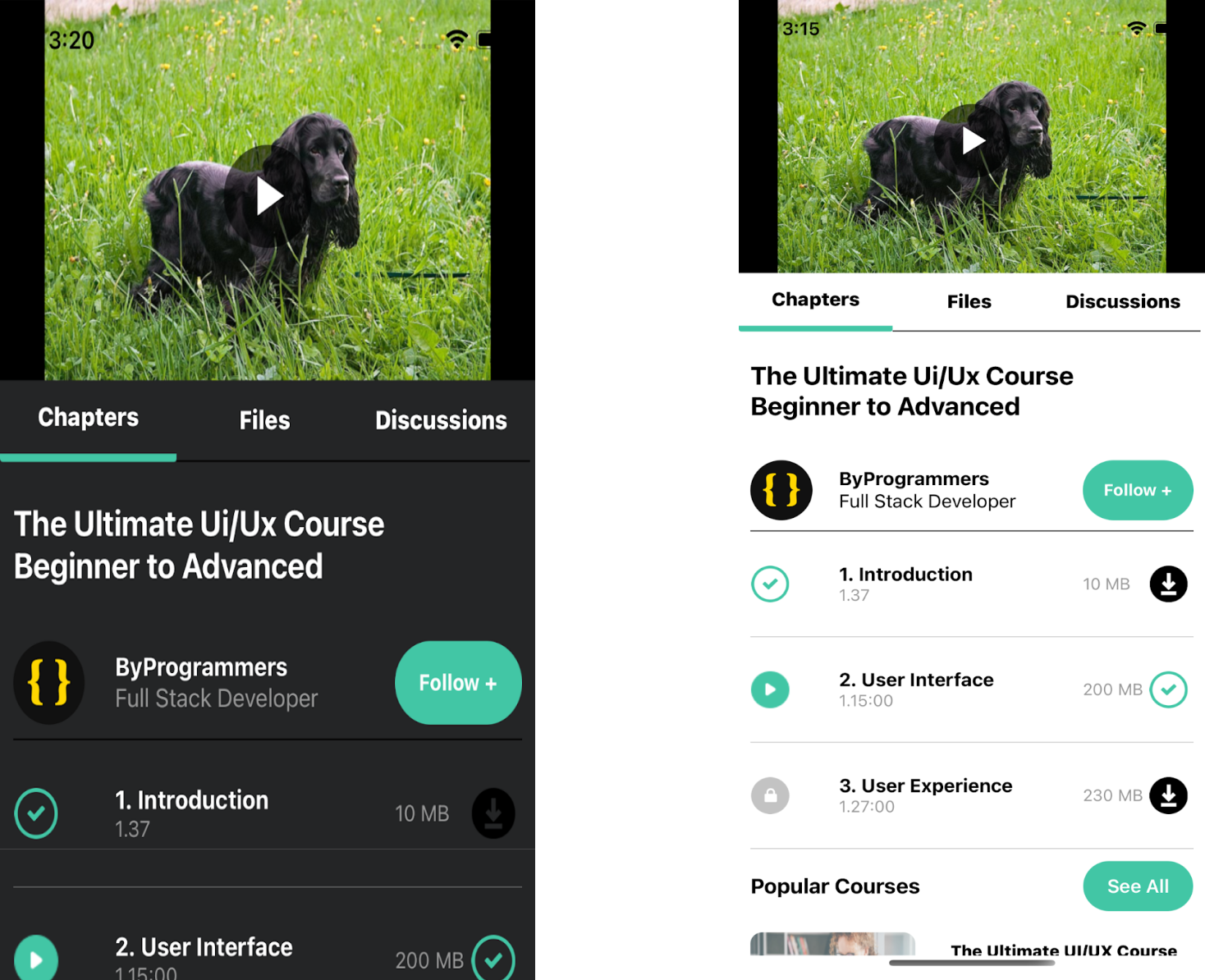
****

* This is a Filter Modal Screen that filters data according to user selected type
* In this screen there are some option like class type, class level witch help the user to filter the Visualization as according his choice



**2.10 POWER BI Screen :**

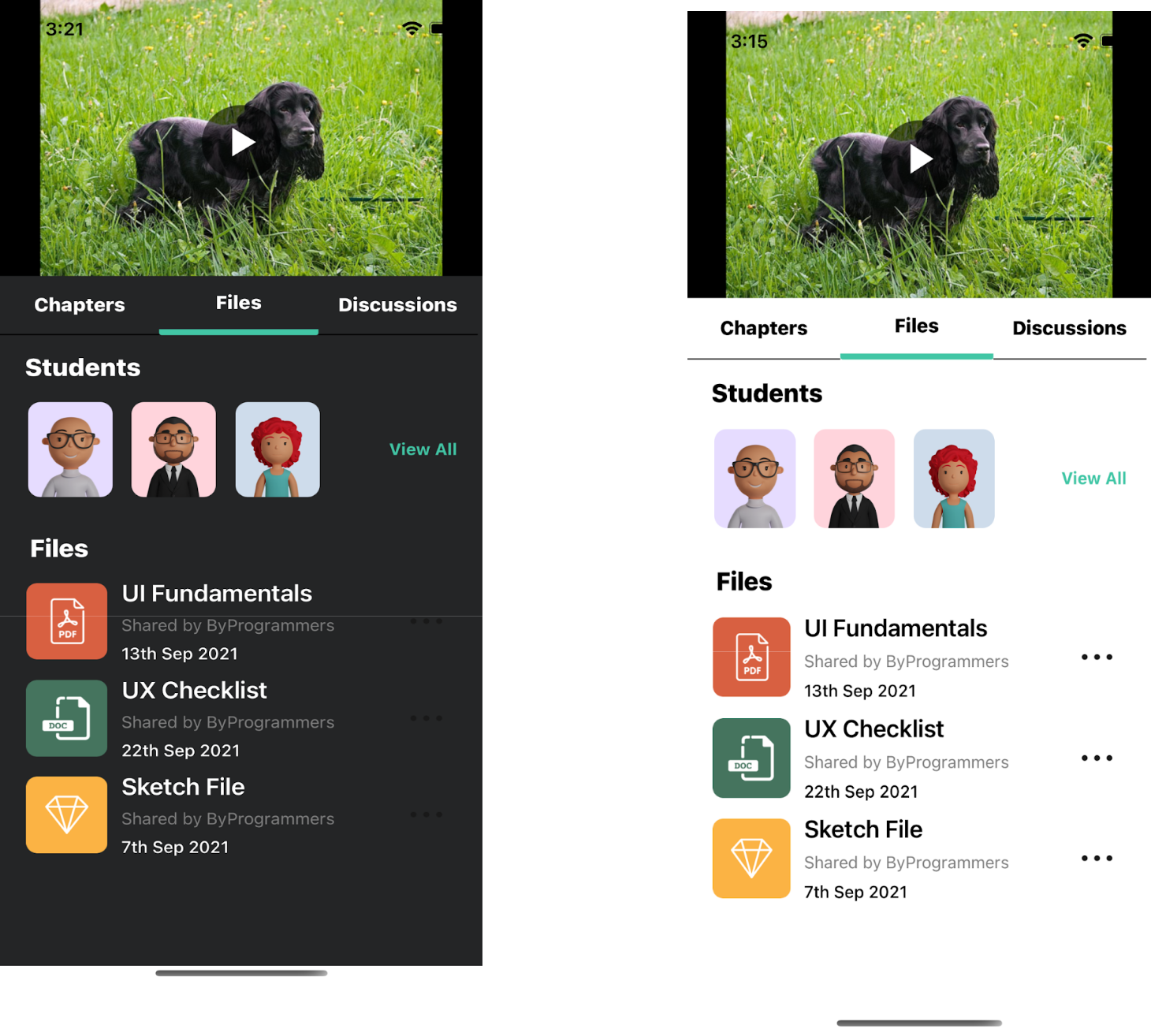
* This Screen shows the of a particular Visualization
* User can watch a Videos of Visualization
* Users can also go to Instructor profile by clicking on follow button
* It also suggests the popular Visualization related to POWER BI



**2.11 BI Files Discussion :**

* In this screen shows No. of Organization

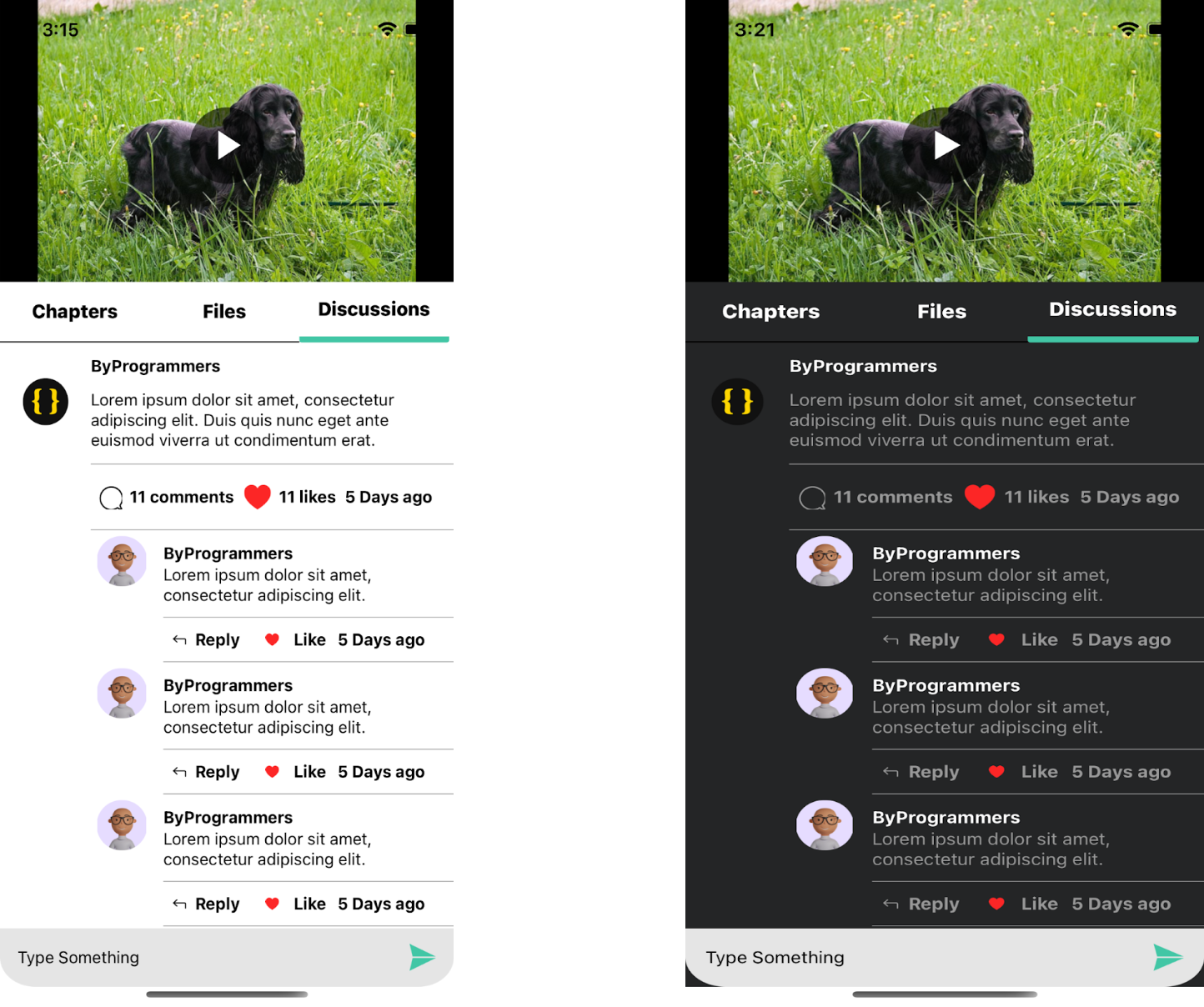
* It also contains a POWER BI Representation kit like(UI Fundamentals,UX Checklist)



|  |
| --- |
| import React, {memo} from 'react'; |
|  | import { |
|  | Text, |
|  | View, |
|  | Image, |
|  | TouchableOpacity, |
|  | ScrollView, |
|  | SafeAreaView, |
|  | } from 'react-native'; |
|  | import {selectedTheme} from '../../config/Themes'; |
|  | import {buttons, Icon, Images, RouteScreens, screensData} from '../../config'; |
|  | import {POWER BIDetailsModel} from '../../models'; |
|  | import RenderItem from '../Home/RenderItem'; |
|  | import styles from './style'; |
|  |  |
|  | interface Corselistingprops { |
|  | navigation: any; |
|  | DATA: POWER BIDetailsModel; |
|  | flatlistdata: { |
|  | id: number; |
|  | title: string; |
|  | clsss\_level: string; |
|  | creted\_on: string; |
|  | duration: number; |
|  | instructor: string; |
|  | ratings: number; |
|  | price: number; |
|  | is\_favourite: boolean; |
|  | thumbnail: any; |
|  | }[]; |
|  | Title: string; |
|  | isfavourite: any; |
|  | setisfavourite: any; |
|  | } |
|  |  |
|  | const POWER BIBIScreen = (props: Corselistingprops) => { |
|  | const {DATA, navigation, flatlistdata, Title, isfavourite, setisfavourite} = |
|  | props; |
|  | return ( |
|  | <SafeAreaView style={styles(selectedTheme).mainContainer}> |
|  | <View style={styles(selectedTheme).conatiner}> |
|  | <View style={styles(selectedTheme).headerContainer}> |
|  | <Text style={styles(selectedTheme).userTitle}>{Title}</Text> |
|  | <View style={styles(selectedTheme).studentData}> |
|  | <Text style={styles(selectedTheme).smallText}> |
|  | {DATA.number\_of\_Organization} |
|  | </Text> |
|  | <Image source={Icon.TIME} style={styles(selectedTheme).timeIcon} /> |
|  | <Text style={styles(selectedTheme).smallText}>{DATA.duration}</Text> |
|  | </View> |
|  | </View> |
|  | <View style={styles(selectedTheme).profileContainer}> |
|  | <View style={styles(selectedTheme).subProfileContainer}> |
|  | <Image |
|  | source={Images.PROFILE} |
|  | style={styles(selectedTheme).profileIcon} |
|  | /> |
|  | <View> |
|  | <Text style={styles(selectedTheme).userNameText}> |
|  | {DATA.instructor.name} |
|  | </Text> |
|  | <Text style={styles(selectedTheme).programmerText}> |
|  | {DATA.instructor.title} |
|  | </Text> |
|  | </View> |
|  | </View> |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).followButton} |
|  | onPress={() => |
|  | navigation.navigate(RouteScreens.INSTRUCTORPROFILEMODEL) |
|  | }> |
|  | <Text style={styles(selectedTheme).followText}> |
|  | {buttons.FOLLOW} + |
|  | </Text> |
|  | </TouchableOpacity> |
|  | </View> |
|  | </View> |
|  | <View style={styles(selectedTheme).seperator} /> |
|  | <View style={styles(selectedTheme).conatiner}> |
|  | <View> |
|  | {DATA.videos.map((item, index) => { |
|  | return ( |
|  | <View key={index} style={styles(selectedTheme).videoContainer}> |
|  | <View style={styles(selectedTheme).subVideoContainer}> |
|  | <View style={styles(selectedTheme).playButton}> |
|  | <Image |
|  | source={Icon.PLAY} |
|  | style={styles(selectedTheme).videoIcon} |
|  | /> |
|  | </View> |
|  | <View> |
|  | <Text style={styles(selectedTheme).title}> |
|  | {item.title} |
|  | </Text> |
|  | <Text style={styles(selectedTheme).timeText}> |
|  | {item.duration} |
|  | </Text> |
|  | </View> |
|  | </View> |
|  | <View style={styles(selectedTheme).buttonContainer}> |
|  | <Text style={styles(selectedTheme).timeText}> |
|  | {item.size} |
|  | </Text> |
|  | <TouchableOpacity> |
|  | <Image |
|  | source={Icon.DOWNLOAD} |
|  | style={styles(selectedTheme).downloadIcon} |
|  | /> |
|  | </TouchableOpacity> |
|  | </View> |
|  | </View> |
|  | ); |
|  | })} |
|  | </View> |
|  |  |
|  | <View style={styles(selectedTheme).itemContainer}> |
|  | <Text style={styles(selectedTheme).popularVisualizationText}> |
|  | {screensData.home.POPULARVISUALIZATION} |
|  | </Text> |
|  | <TouchableOpacity style={styles(selectedTheme).seeAllButton}> |
|  | <Text style={styles(selectedTheme).seeAllText}> |
|  | {buttons.SEEALL} |
|  | </Text> |
|  | </TouchableOpacity> |
|  | </View> |
|  |  |
|  | <ScrollView> |
|  | {flatlistdata.map((item, index) => { |
|  | return ( |
|  | <RenderItem |
|  | key={index} |
|  | navigation={navigation} |
|  | item={item} |
|  | index={index} |
|  | isfavourite={isfavourite} |
|  | setisfavourite={setisfavourite} |
|  | /> |
|  | ); |
|  | })} |
|  | </ScrollView> |
|  | </View> |
|  | </SafeAreaView> |
|  | ); |
|  | }; |
|  | export default memo(POWER BIBIScreen); |

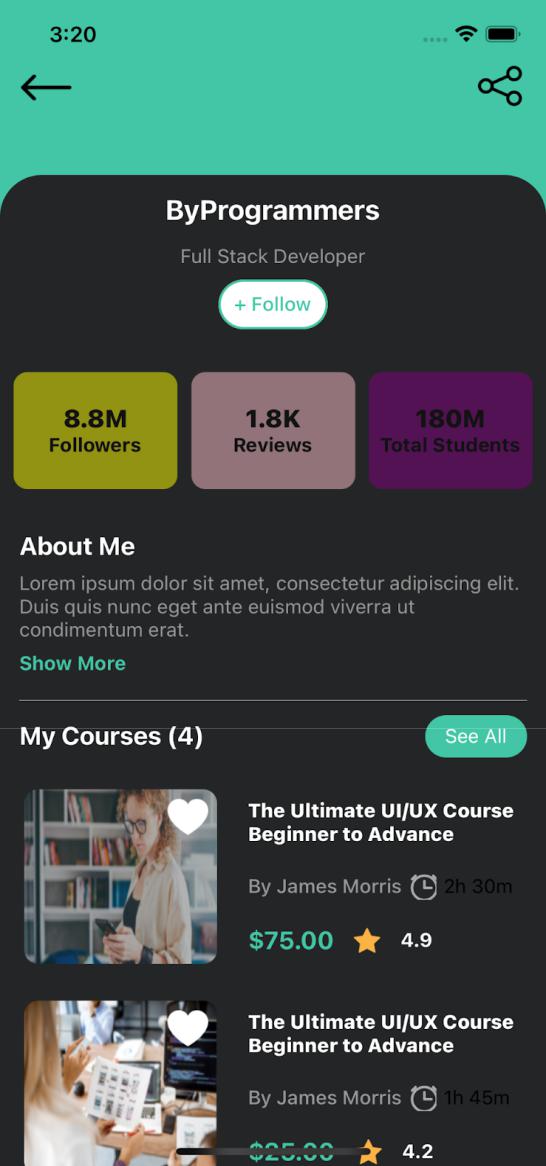
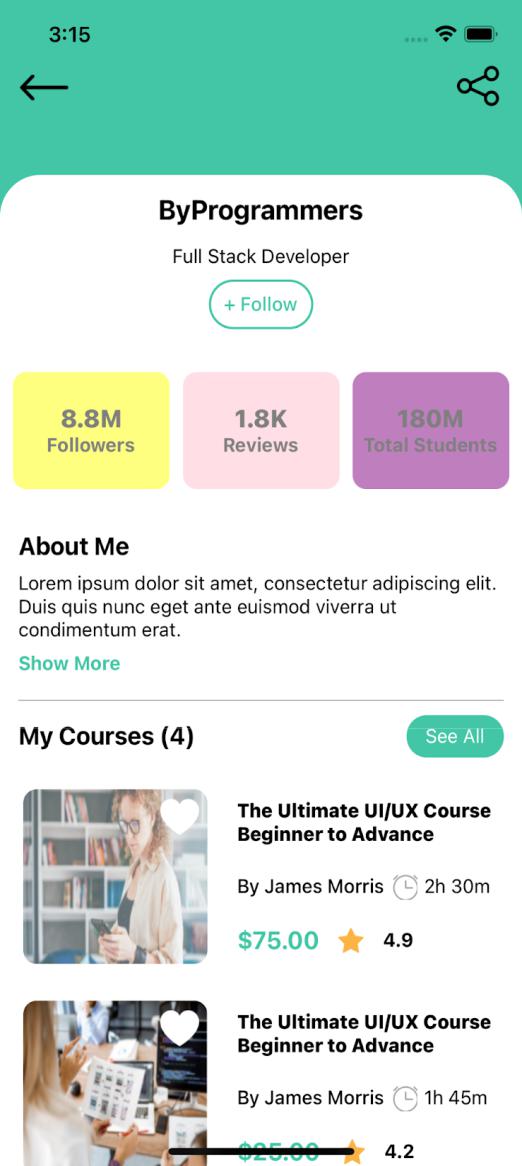
**2.12 POWER BI Discussion Screen :**

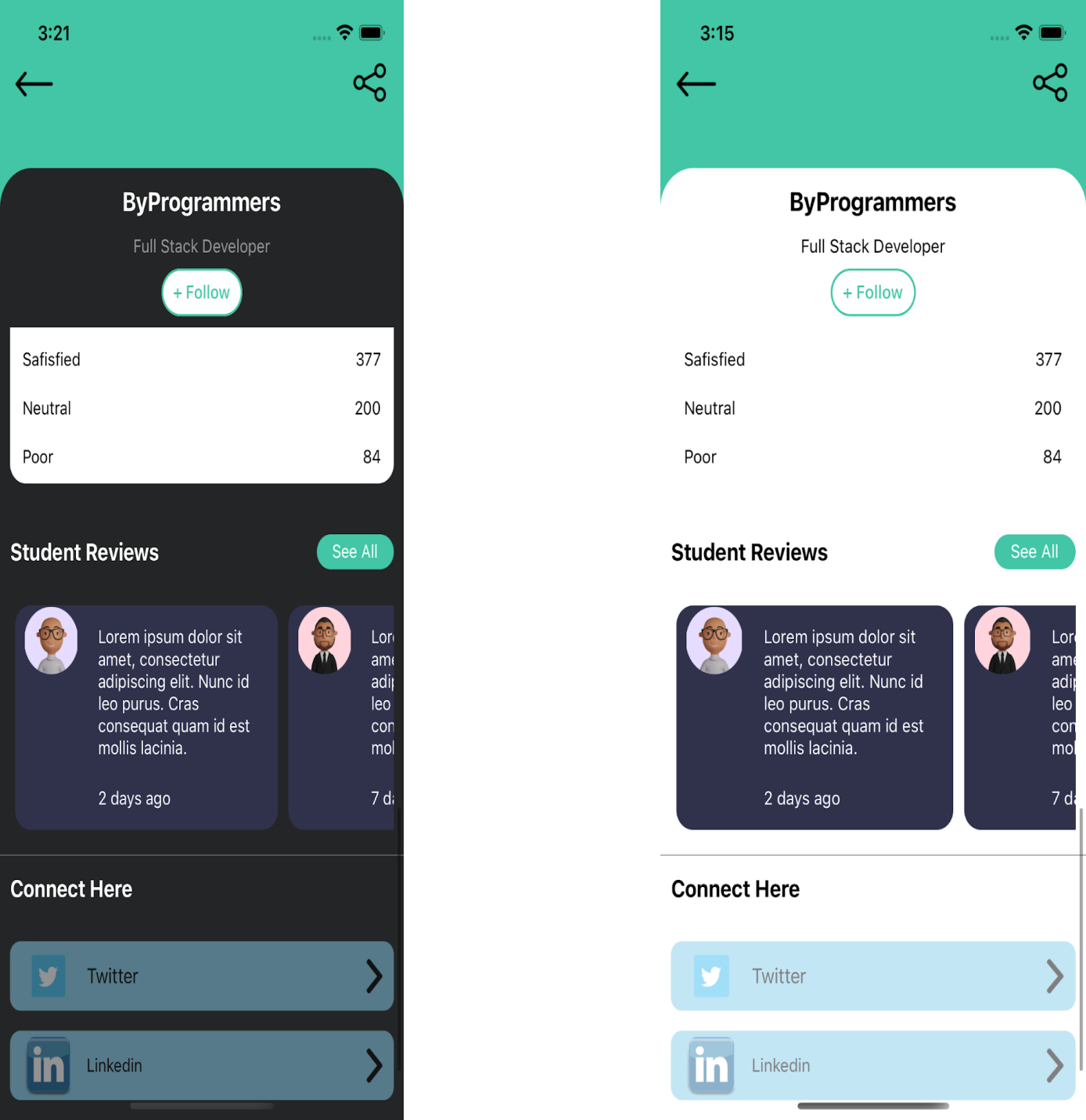
* This Screen contains the discussion section like comments and doubt related to Visualization
* Users can read and write new comments



**2.13 Instructor Screen :**

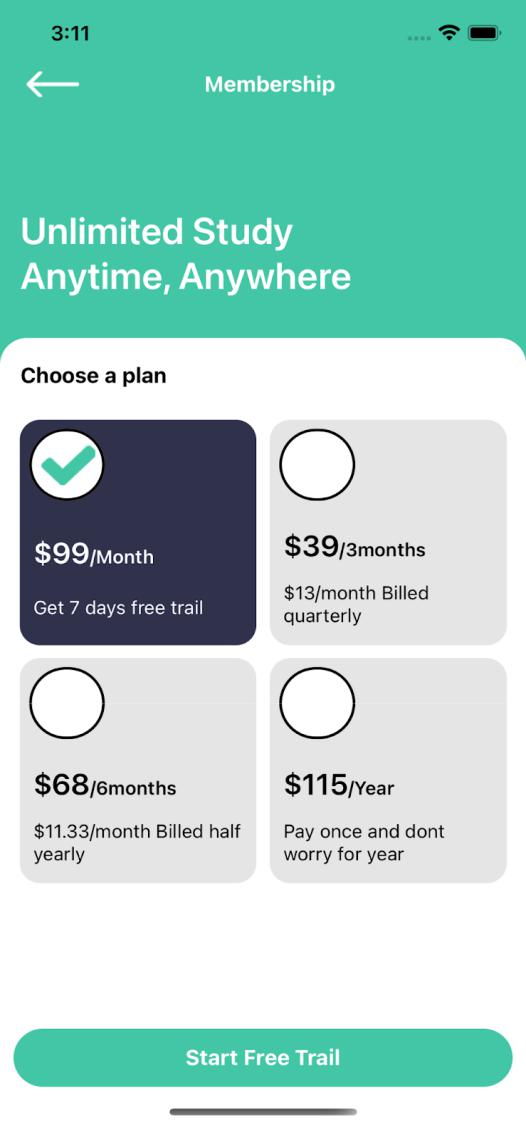
* This is the Instructor profile screen which shows some basic information of Instructor
* Users can also follow the Instructor
* It also shows how many Visualization a users have
* Reviews submitted by Organization for the Instructor
* Users can also connect Instructor to social platform (Twitter, linkedin) by clicking the Twitter, linkedin Buttons





**2.14 Membership Screen :**

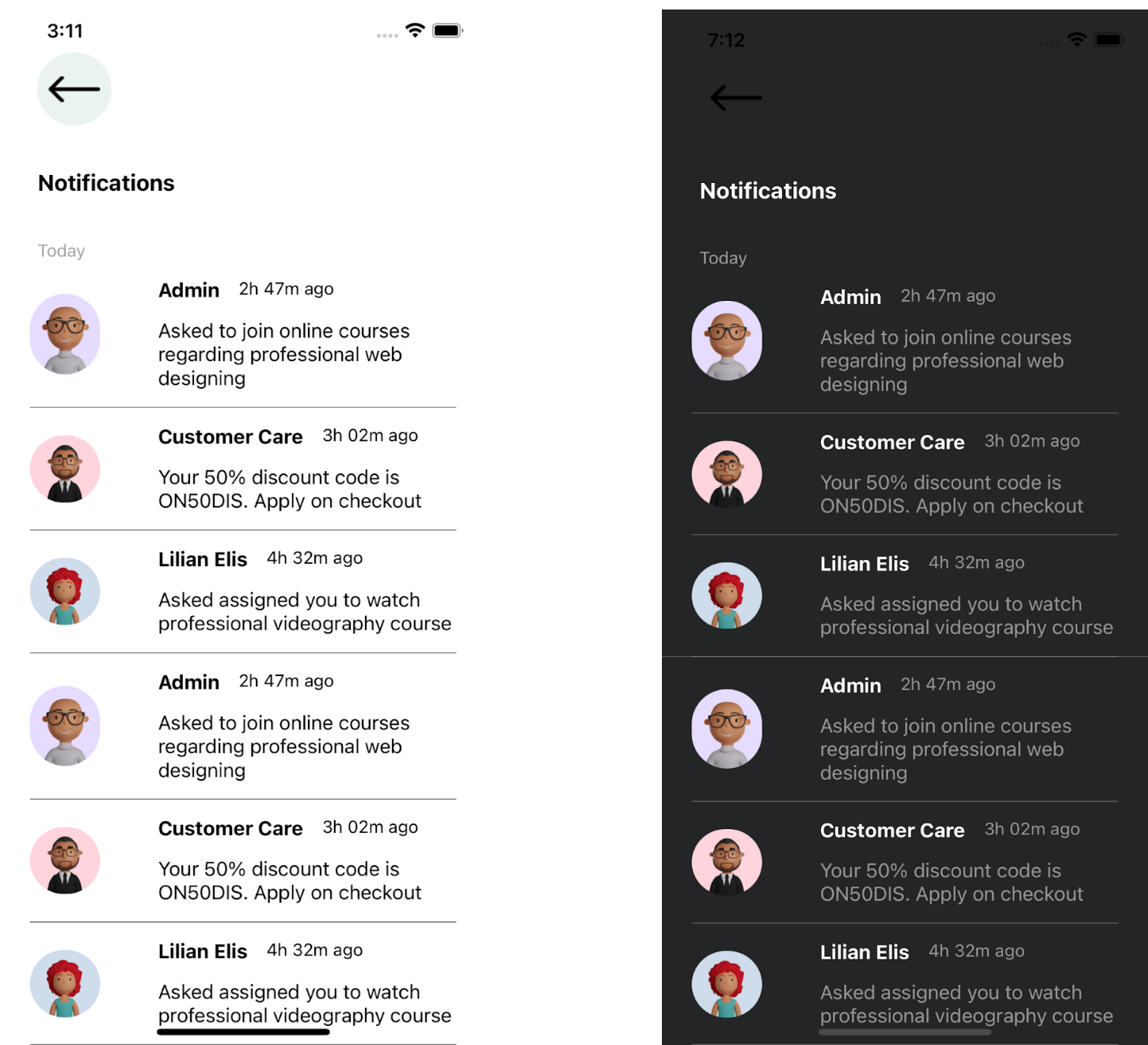
* This screen contains information of membership of Visualization



|  |
| --- |
| import React, {memo} from 'react'; |
|  | import { |
|  | FlatList, |
|  | Image, |
|  | Pressable, |
|  | SafeAreaView, |
|  | Text, |
|  | TouchableOpacity, |
|  | View, |
|  | } from 'react-native'; |
|  | import {selectedTheme} from '../../config/Themes'; |
|  | import {buttons, Icon, screensData, Constant, RouteScreens} from '../../config'; |
|  | import Renderitem from './Renderitem'; |
|  | import styles from './style'; |
|  |  |
|  | interface CategoryProps { |
|  | navigation: any; |
|  | visible: any; |
|  | setvisibe: any; |
|  | defaultitem: number; |
|  | setdefaultitem: React.Dispatch<React.SetStateAction<number>>; |
|  | } |
|  |  |
|  | const MembershipScreen = (props: CategoryProps) => { |
|  | const {navigation, visible, setvisibe, defaultitem, setdefaultitem} = props; |
|  |  |
|  | return ( |
|  | <SafeAreaView style={styles(selectedTheme).mainContainer}> |
|  | <View style={styles(selectedTheme).headContentContainer}> |
|  | <View style={styles(selectedTheme).header}> |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).leftButton} |
|  | onPress={() => navigation.goBack()}> |
|  | <Image |
|  | source={Icon.LEFT\_ARROW} |
|  | style={styles(selectedTheme).headerLeftIcon} |
|  | /> |
|  | </TouchableOpacity> |
|  | <Text style={styles(selectedTheme).headerText}> |
|  | {screensData.membership.NAME} |
|  | </Text> |
|  | </View> |
|  | <Text style={styles(selectedTheme).text}> |
|  | {screensData.membership.HEADER} |
|  | </Text> |
|  | </View> |
|  |  |
|  | <View style={styles(selectedTheme).container}> |
|  | <Text style={styles(selectedTheme).itemBigText}> |
|  | {screensData.membership.DATA} |
|  | </Text> |
|  |  |
|  | <View style={styles(selectedTheme).flatlistView}> |
|  | <FlatList |
|  | data={Constant.membership} |
|  | showsVerticalScrollIndicator={false} |
|  | extraData={Constant.membership} |
|  | renderItem={({item, index}) => ( |
|  | <Renderitem |
|  | item={item} |
|  | index={index} |
|  | defaultitem={defaultitem} |
|  | setdefaultitem={setdefaultitem} |
|  | /> |
|  | )} |
|  | numColumns={2} |
|  | keyExtractor={(\_, index) => index.toString()} |
|  | /> |
|  |  |
|  | <TouchableOpacity |
|  | style={styles(selectedTheme).button} |
|  | onPress={() => navigation.navigate(RouteScreens.ROOTTAB)}> |
|  | <Text style={styles(selectedTheme).buttonText}> |
|  | {buttons.STARTFREETRAIL} |
|  | </Text> |
|  | </TouchableOpacity> |
|  | </View> |
|  | </View> |
|  | </SafeAreaView> |
|  | ); |
|  | |  |  | | --- | --- | |  | export default memo(MembershipScreen); | |
|  |  |

**2.15 Notification Screen :**

* This screen shows notifications like offers, Users message new Visualization etc



|  |
| --- |
| import React, {memo} from 'react'; |
|  | import { |
|  | Text, |
|  | View, |
|  | SectionList, |
|  | Image, |
|  | TouchableOpacity, |
|  | SafeAreaView, |
|  | } from 'react-native'; |
|  | import {selectedTheme} from '../../config/Themes'; |
|  | import {Icon, Images, screensData} from '../../config'; |
|  | import RenderItems from './Renderitem'; |
|  | import styles from './style'; |
|  |  |
|  | interface NotificationTabprops { |
|  | navigation: any; |
|  | DATA: { |
|  | title: string; |
|  | data: { |
|  | id: number; |
|  | avatar: any; |
|  | name: string; |
|  | created\_at: string; |
|  | message: string; |
|  | }[]; |
|  | }[]; |
|  | } |
|  |  |
|  | const NotificationTab = (props: NotificationTabprops) => { |
|  | const {DATA, navigation} = props; |
|  | return ( |
|  | <SafeAreaView style={styles(selectedTheme).mainContainer}> |
|  | <Image |
|  | source={selectedTheme.name == 'light' ? Images.BG : Images.BG\_DARK} |
|  | style={styles(selectedTheme).bgImage} |
|  | /> |
|  | <View style={styles(selectedTheme).container}> |
|  | <TouchableOpacity |
|  | onPress={() => navigation.goBack()} |
|  | style={styles(selectedTheme).leftButton}> |
|  | <Image |
|  | source={Icon.BACK} |
|  | style={styles(selectedTheme).headerLeftIcon} |
|  | /> |
|  | </TouchableOpacity> |
|  | <Text style={styles(selectedTheme).headerText}> |
|  | {screensData.notification.NOTIFICATIONS} |
|  | </Text> |
|  |  |
|  | <SectionList |
|  | sections={DATA} |
|  | showsVerticalScrollIndicator={false} |
|  | keyExtractor={(item, index) => item + index.toString()} |
|  | renderItem={({item}) => <RenderItems item={item} />} |
|  | renderSectionHeader={({section: {title}}) => ( |
|  | <Text style={styles(selectedTheme).header}>{title}</Text> |
|  | )} |
|  | /> |
|  | </View> |
|  | </SafeAreaView> |
|  | ); |
|  | }; |
|  | export default memo(NotificationTab); |

**Conclusion**

**POWER BI TOOL is beneficial to the Organization, tutors and the institution offering these Visualization**. I would therefore recommend that POWER BI TOOL be implemented on all Representation institutions and research on how to improve this Representation process should be carried out. POWER BI TOOL is not just a change of technology. It is part of a redefinition of how we as a species transmit knowledge, skills, and values to younger generations of workers and Organization. I will end this book by daring to make a few predictions of how e-Representation and the functions it serves will continue to develop.

**Declaration**

I hereby declare that The Project entitled is an outcome of my own efforts under the guidance of **prof.Neelam Rawat**. The project is submitted to the department of MCA. For the partial fulfilment of Master of Computer Application 2019-22.

I also declare that project report is not submitted in any of the university previously.

Date: 24-May-2022

Place: Noida (Uttar Pradesh)

**BIBLIOGRAPHY :**

* **GOOGLE**
* **Project Github Link :**