# Vinay Israni Batch :- B

**D15A\_24**

# Experiment no. :- 06

**Aim:** To connect firebase database with flutter ui

## Theory:

Firebase is a great backend solution for anyone that wants to use authentication, databases, cloud functions, ads, and countless other features within an app.

CODE –

import 'package:animate\_do/animate\_do.dart';

import 'package:flutter/material.dart';

import 'package:url\_launcher/url\_launcher.dart';

import '../../../core/base/view/base\_view.dart';

import '../../../core/components/circle\_load/circle\_load.dart';

import '../viewmodel/repository\_view\_model.dart';

import '../../../core/extensions/app\_extensions.dart';

import '../../../product/components/appbar/special\_appbar.dart';

class RepositoryView extends StatelessWidget {

final String username;

const RepositoryView({Key? key, required this.username}) : super(key: key);

@override

Widget build(BuildContext context) => BaseView<RepositoryViewModel>(

viewModel: RepositoryViewModel(),

onModelReady: (model) {

model.setContext(context);

model.init();

model.fetchAllRepositoryItems(username);

},

onPageBuilder: (BuildContext context, RepositoryViewModel viewModel) =>

Scaffold(

appBar: \_appBar(context),

body: viewModel.isLoad

? \_loadedBody(context, viewModel)

: const CircleLoad(),

),

);

SpeacialAppBar \_appBar(BuildContext context) => SpeacialAppBar(

context: context,

title: "Repository",

iconThemeData: IconThemeData(color: context.textColor),

);

FlipInX \_loadedBody(BuildContext context, RepositoryViewModel viewModel) =>

FlipInX(

child: ListView.separated(

padding: context.paddingNormal,

itemBuilder: (context, index) => \_item(viewModel, index, context),

separatorBuilder: (context, index) => \_divider(context),

itemCount: viewModel.repositoryModel.length,

),

);

ListTile \_item(

RepositoryViewModel viewModel, int index, BuildContext context) =>

ListTile(

onTap: () => \_launchURL(viewModel, index),

title: \_title(viewModel, index),

subtitle: \_subtitle(context, viewModel, index),

);

Text \_title(RepositoryViewModel viewModel, int index) =>

Text(viewModel.repositoryModel[index].name ?? "NULL\_REPO");

Padding \_subtitle(

BuildContext context, RepositoryViewModel viewModel, int index) =>

Padding(

padding: context.verticalPaddingLow,

child: Wrap(

crossAxisAlignment: WrapCrossAlignment.center,

spacing: context.lowValue,

children: [

CircleAvatar(

radius: 10,

backgroundColor: context.randomColor,

),

Text(viewModel.repositoryModel[index].language ?? "Readme"),

context.emptySizedWidthBoxLow,

Text(

"Updated ${viewModel.repositoryModel[index].updated\_at!.substring(0, 10)}.")

],

),

);

Padding \_divider(BuildContext context) => Padding(

padding: EdgeInsets.fromLTRB(

0,

context.normalValue,

0,

context.lowValue,

),

child: const Divider(

thickness: 2,

),

);

void \_launchURL(RepositoryViewModel viewModel, int index) async {

if (await canLaunch(viewModel.repositoryModel[index].html\_url ?? "")) {

await launch(viewModel.repositoryModel[index].html\_url ?? "",

forceSafariVC: true, forceWebView: true, enableJavaScript: true);

}

}

}

OUTPUT - 

