FAKULTI TEKNOLOGI MAKLUMAT DAN KOMUNIKASI

SEM 1 2023/2024

BITP 3453 MOBILE APPLICATION DEVELOPMENT

PROJECT ARTIFACT DECLARATION

TITLE: INSTANT TEXT SUMMARIZER

GROUP NUMBER: 15

Matric No	Name
B032210015	AHNUSIYAA A/P SEKARAN
B032110312	SAEDATUL IZZAH BT AZIZUL RAHMAN
B032110121	RASHMIKA A/P KATHIRAVAN

Description of the Project

Write a brief description of the project in this section. The description shall include the link to the project demonstration on YouTube.

KeyPoint is a mobile application that was designed to transform lengthy texts into interactive flashcards. By extracting and presenting the key points, it is efficient for users to learn. By developing this mobile application, it is easier for users to transform their knowledge on the go.

Video link: https://youtu.be/Cf2HWMJewo4

The Application FIRST SCREEN

Insert a screenshot of the application's first screen in this section.

THE OTHER SCREEN 1

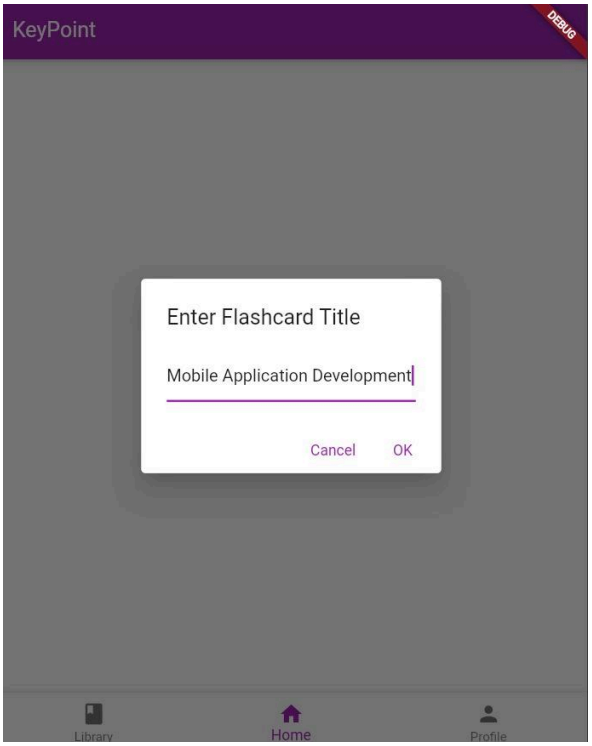
Insert the other screen with the description in this section. The description shall describe the State the file name, class and method that are used to implement the screen.


```dart
class HomeScreenContent extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Container(
      decoration: BoxDecoration(
        image: DecorationImage(
          image: AssetImage('assets/background.jpg'),
          fit: BoxFit.cover,
        ), // DecorationImage
      ), // BoxDecoration
      child: Center(
        child: Container(
          decoration: BoxDecoration(
            color: Colors.white70,
            borderRadius: BorderRadius.circular(10),
          ), // BoxDecoration
          padding: EdgeInsets.all(16),
          child: ElevatedButton(
            onPressed: () {
              _showFlashcardTitleDialog(context);
            },
            child: Text('Add Flashcard', style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold)),
          ), // ElevatedButton
        ), // Container
      ), // Center
    ); // Container
  }
```

The above code sample is written in Dart and is a component of a Flutter application. The content of the home screen is represented by a class called `HomeScreenContent`, which extends `StatelessWidget`. It generates a container with a backdrop picture and a centered child container inside the `build` method. The child container features an elevated "Add Flashcard" button and a white, semi-transparent backdrop with rounded corners. This button displays a dialog for entering a flashcard title when it is pressed. This dialog is triggered by the `_showFlashcardTitleDialog} method, which is probably implemented in another area of the codebase. The Flutter framework handles navigation and user interface interaction through the `context` argument. The file name where this class is defined is not specified.

THE OTHER SCREEN 2

Insert the other screen with the description in this section. The description shall describe the State the file name, class and method that are used to implement the screen.

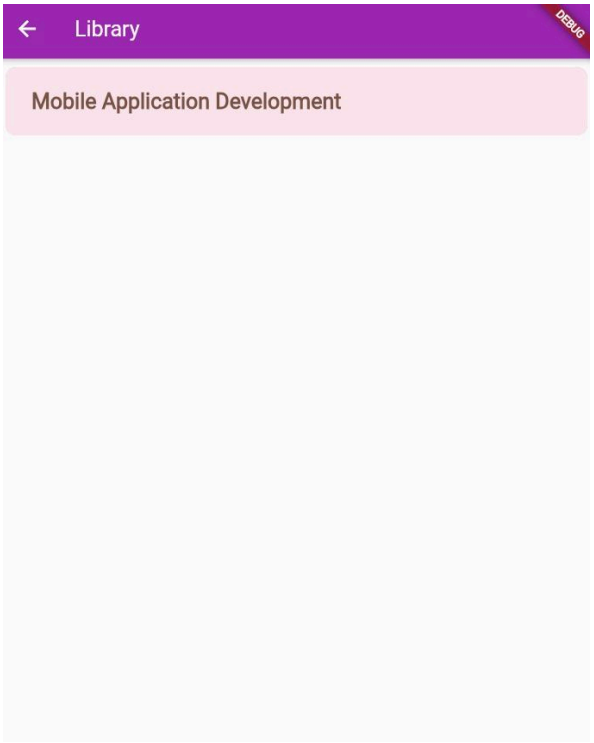



```
Future<void> _showFlashcardTitleDialog(BuildContext context) async {
  TextEditingController controller = TextEditingController();
  return showDialog<void>(
    context: context,
    builder: (BuildContext context) {
      return AlertDialog(
        title: Text('Enter Flashcard Title'),
        content: TextField(controller: controller),
        actions: <Widget>[
          TextButton(
            onPressed: () {
              Navigator.of(context).pop();
            },
            child: Text('Cancel'),
          ), // TextButton
          TextButton(
            onPressed: () {
              Navigator.of(context).pop();
              Navigator.push(
                context,
                MaterialPageRoute(
                  builder: (context) => ParagraphScreen(flashcardTitle: controller.text),
                ), // MaterialPageRoute
              );
            },
            child: Text('OK'),
          ), // TextButton
        ], // <Widget>[]
      ); // AlertDialog
    },
  );
```

The screen above is the Enter FlashCard Title. The given code snippet was probably written in a Flutter application and is written in Dart. It defines a `_showFlashcardTitleDialog} method inside an unknown class that is most likely related to a Flutter widget. This function is in charge of causing the application to display a dialog box asking the user to input a flashcard's title. The dialog box has two action buttons: "Cancel" and "OK," and a Text input field where the user can type the title. The supplied title is given to a new instance of the `ParagraphScreen} widget via the `MaterialPageRoute} when the 'OK' button is pushed. Presumably, the purpose of the `ParagraphScreen} widget is to show content associated with the flashcard title that was entered. The file name and class in which this method is located are not specified in the code.

THE OTHER SCREEN 3

Insert the other screen with the description in this section. The description shall describe the State the file name, class and method that are used to implement the screen.

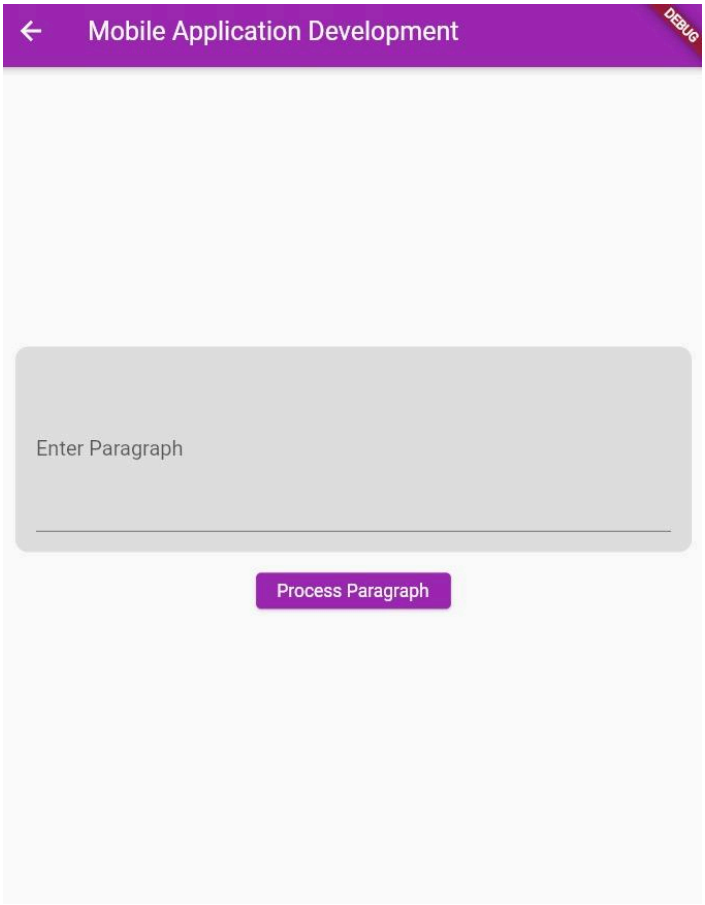



```dart
// j) The library Screen is shown.

class LibraryScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text('Library')),
      body: ListView.builder(
        itemCount: Library.flashcards.length,
        itemBuilder: (context, index) {
          return FlashcardItem(flashcard: Library.flashcards[index]);
        },
      ),  // ListView.builder
    );  // Scaffold
  }
}
```

The above code snippet is a component of a Flutter application and was written in Dart. It defines the `LibraryScreen` class, which is an application screen that extends `StatelessWidget`. The body of this screen is a `ListView.builder}, and the app bar is scaffolded with the title 'Library'. For every flashcard in the `Library.flashcards} list, the builder uses the `FlashcardItem` widget to dynamically produce a list of objects. The screen is responsible for presenting the flashcards in a scrollable list view. It appears that the `Library` class has a static list of flashcards. The code provided does not include any more methods, nor does it specify the file name containing the definition of this class. According to the implementation, this screen is intended to

THE OTHER SCREEN 4

Insert the other screen with the description in this section. The description shall describe the State the file name, class and method that are used to implement the screen.




```
@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(title: Text(widget.flashcardTitle)),
    body: Padding(
      padding: const EdgeInsets.all(16.0),
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          Container(
            padding: EdgeInsets.all(16),
            decoration: BoxDecoration(
              color: Colors.grey.withOpacity(0.3), // Set the background color with opacity
              borderRadius: BorderRadius.circular(10),
            ),  // BoxDecoration
            child: TextField(
              controller: paragraphController,
              maxLines: 5,
              decoration: InputDecoration(labelText: 'Enter Paragraph'),
            ),  // TextField
          ),  // Container
```

The above code snippet is a component of a Flutter application and was written in Dart. It describes a widget class, perhaps called `ParagraphScreen}, and its `build` method. This class is linked to a screen that lets the user enter paragraphs and extends a Flutter {StatefulWidget`. The body of the screen has a padding column with a container holding a text field where paragraphs can be entered, and the screen is scaffolded with an app bar showing the title of the flashcard. The text field's background is a grey color that is semi-transparent and has rounded corners. The text input of the widget is managed by a controller called {paragraphController}. It is not specified which file contains the definition of this class. According to the implementation, this screen is meant to be used for capturing

THE OTHER SCREEN 5

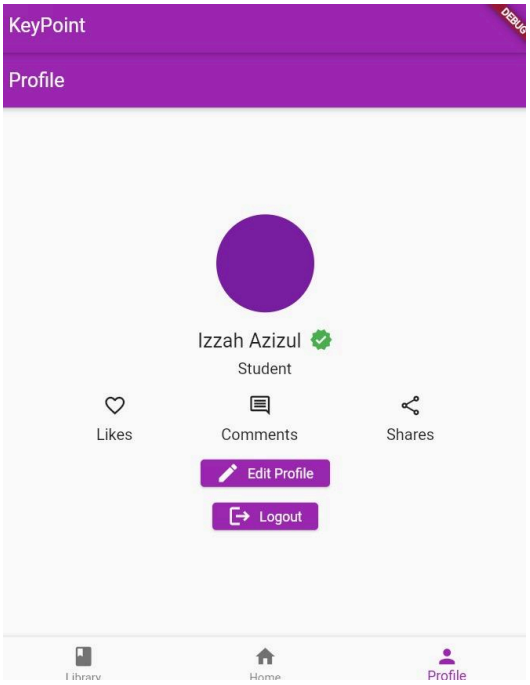
Insert the other screen with the description in this section. The description shall describe the State the file name, class and method that are used to implement the screen.


```
void _processParagraph() {
  String paragraph = paragraphController.text;
  List<String> extractedSentences = paragraph.split('. ');
  sentences.clear();
  sentences.addAll(extractedSentences);
  currentSentenceIndex = 0;
```

The above code snippet is a component of a Flutter application and was written in Dart. It describes a widget class, perhaps called `FlashcardDisplayScreen}, and its `build` method. This class represents a screen that shows flashcard content by extending a Flutter {StatefulWidget`. A `Column` widget scaffolds the screen, holding a gradient-colored container that shows the current sentence from the flashcard, page information, and navigation buttons. A `LinearGradient} between pink tones provides the gradient background color. The text inside the container has a bold font, size 20, and is centered and styled in white. The code provided does not include any more methods, nor does it specify the file name containing the definition of this class. The display provides

THE OTHER SCREEN 6

Insert the other screen with the description in this section. The description shall describe the State the file name, class and method that are used to implement the screen.

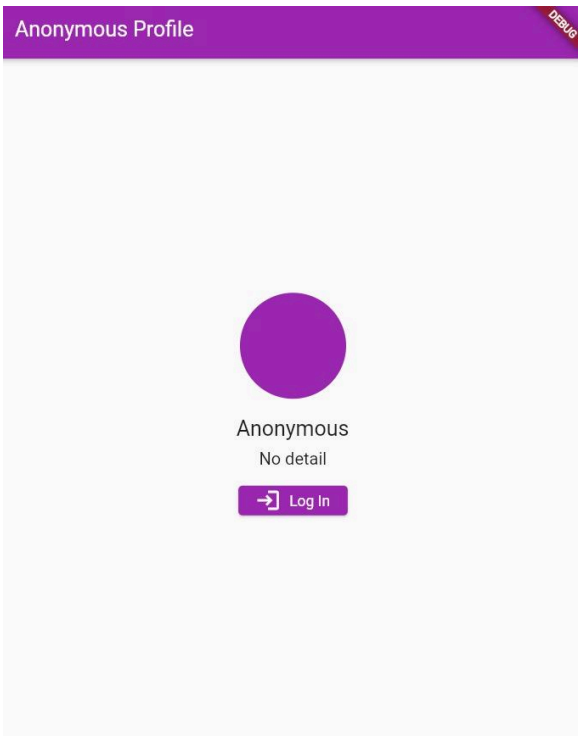



```
class ProfileScreen extends StatefulWidget {
  @override
  _ProfileScreenState createState() => _ProfileScreenState();
}

class _ProfileScreenState extends State<ProfileScreen> {
  bool isLoggedIn = true;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text('Profile')),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
```

The given code snippet replaces the `build` function with the `@override` annotation, implementing a Flutter screen in Dart. It is most likely a member of a class called `_ProfileScreenState`. The corresponding widget is called "ProfileScreen}," and this class extends {State<ProfileScreen>}. Screen scaffolding includes a `AppBar} with the title 'Profile,' and a `Column} with multiple widgets including a `CircleAvatar} for the profile picture, text elements for the username and role, engagement metrics, and buttons for editing, logging in, and logging out. The class also contains state management using a boolean variable `isLoggedIn`, managing the visibility of particular items based on whether the user is logged in. There is functionality on the screen to navigate to an

THE OTHER SCREEN 7

Insert the other screen with the description in this section. The description shall describe the State the file name, class and method that are used to implement the screen.




```dart
class AnonymousProfilePage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text('Anonymous Profile')),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            CircleAvatar(
              radius: 50,
              backgroundColor: Colors.purple, // Placeholder for anonymous profile image
            ), // CircleAvatar
            SizedBox(height: 16),
            Row(
              mainAxisAlignment: MainAxisAlignment.center,
              children: [
                Text(
                  'Anonymous',
                  style: TextStyle(fontSize: 20),
                ), // Text
              ],
```

This code sample extends `StatelessWidget` and implements a Flutter screen in Dart using the `AnonymousProfilePage` class. This screen is a representation of an anonymous user's profile page. It has a purple circular avatar placeholder, the username "Anonymous," and a succinct description of "No detail." It has a button that, when pushed, takes the user to the

{WelcomeScreen}, suggesting that it probably takes you to the login screen. It is not specified which file contains the definition of this class. To create a straightforward and user-friendly anonymous profile page, the screen is scaffolded with a `AppBar} that displays the headline 'Anonymous Profile' and a `Column} that organizes different widgets. The code clearly invites users to log in and reflects the static nature of an anonymous profile.