Contents

[Part 1 Answer: Why Cross platform development is highly in demand nowadays - comparison with native application development . 2](#_Toc171717307)

[1. Cost Efficiency 2](#_Toc171717308)

[2. Development Speed 2](#_Toc171717309)

[3. Resource Utilization 2](#_Toc171717310)

[4. User Experience (UX) 2](#_Toc171717311)

[5. Consistency Across Platforms 2](#_Toc171717312)

[6. Market Reach 2](#_Toc171717313)

[7. Maintenance and Updates 3](#_Toc171717314)

[Overview of Mobile Application features 3](#_Toc171717315)

[Source Code:- 4](#_Toc171717316)

[Project Architecture and Design of the Application 5](#_Toc171717317)

[Folder Structure 5](#_Toc171717318)

[Data base and CRUD operations 6](#_Toc171717319)

[ToDo class 6](#_Toc171717320)

[Database Service class 7](#_Toc171717321)

[Third party libraries 9](#_Toc171717322)

[UI / UX Design 9](#_Toc171717323)

[Test case 9](#_Toc171717324)

[Possible issue and errors in applications 11](#_Toc171717325)

[References 11](#_Toc171717326)

# Part 1 Answer: Why Cross platform development is highly in demand nowadays - comparison with native application development .

### 1. **Cost Efficiency**

* **Cross-Platform:** Developing a single codebase for multiple platforms (iOS, Android) significantly reduces the development and maintenance costs.
* **Native Development:** Requires separate teams and codebases for each platform, leading to higher costs in terms of development, testing, and maintenance.

### 2. **Development Speed**

* **Cross-Platform:** Allows for faster development as a single codebase can be used across multiple platforms. This is especially beneficial for startups and companies looking to get their products to market quickly.
* **Native Development:** Takes more time as developers have to write and maintain different codebases for each platform.

### 3. **Resource Utilization**

* **Cross-Platform:** Easier to manage resources and teams since one team can handle the entire project, leveraging shared knowledge and code.
* **Native Development:** Requires more specialized developers for each platform (iOS and Android), which can be harder to manage and more expensive.

### 4. **User Experience (UX)**

* **Cross-Platform:** Modern cross-platform frameworks (e.g., Flutter, React Native) provide near-native performance and a good user experience. They offer a variety of plugins and libraries that can help achieve a native-like feel.
* **Native Development:** Offers the best possible performance and user experience as the apps are tailored specifically for the platform, making full use of the device's features and capabilities.

### 5. **Consistency Across Platforms**

* **Cross-Platform:** Ensures a consistent user experience across different devices and platforms since the same codebase is used.
* **Native Development:** While it can offer the best experience for each individual platform, maintaining consistency across different platforms can be challenging.

### 6. **Market Reach**

* **Cross-Platform:** Helps in reaching a broader audience quickly since the app is available on multiple platforms from the start.
* **Native Development:** Slower to reach all potential users as each platform needs to be developed and launched separately.

### 7. **Maintenance and Updates**

* **Cross-Platform:** Easier to manage updates and bug fixes as changes need to be made in a single codebase and deployed across all platforms.
* **Native Development:** Requires updates to be made separately for each platform, which can be time-consuming and error-prone.

# Overview of Mobile Application features

This project was mainly done create mobile application using latest cross platform technologies . The created application is ToDo Application , which helps users to manage their day to day task .

**1. Main features:**

* Users can easily create new Todo items through user friendly interface.
* User can ,create , update , delete and mark as done Todo items .

**2. Data Persistence:**

* SQLite database integration allows for efficient and reliable storage of contact information.
* Users can trust that their data is securely saved and readily accessible across app sessions.

**3. Intuitive User Interface:**

* The application boasts a clean and modern UI, courtesy of Flutter and Material UI, enhancing the overall user experience.
* Material UI components contribute to a consistent and visually appealing design.

5. **Cross-Platform Compatibility:**

* Built with Flutter, the app is designed to run seamlessly on both Android and iOS devices, ensuring a broader user reach.

# Source Code:-

**Github Link :**

https://github.com/kume12345/Flutter-ToDo-App

# Project Architecture and Design of the Application

## Folder Structure

A screenshot of a computer

Description automatically generated

* Main entry for the application is main.dart , it will boot strap the application
* All user interfaces are located under the screens folder.
* Each crud operation is handled in different layouts.
* Data base objects are located under the model folder.
* All other shared logic such as , dbHelpers , and Application colors are stored under shared folder.
* Currently project have database\_service utility which is used to connect the app with SQLLite data base .

## Data base and CRUD operations

### ToDo class

A screen shot of a computer program

Description automatically generated

In this Todo class:

* id: An optional integer ID for the todo item.
* categoryId: Integer representing the category ID to which the todo belongs.
* isDone: Integer (typically 0 or 1) indicating whether the todo is completed.
* description: String describing the todo item.

The toMap method converts a Todo object into a Map<String, dynamic> for database storage or serialization purposes. The fromMap factory method creates a Todo object from a Map<String, dynamic> retrieved from a database or any other source where data is stored in map format.

### Database Service class

A screen shot of a computer program

Description automatically generated

Database helper class mainly used to connect app with SQLite data base , it creates singleton object . Which mean every part of the application can use the same connection .

#### Insert new ToDo

A screen shot of a computer

Description automatically generated

#### Get All ToDos

A screen shot of a computer

Description automatically generated

#### Update ToDo

A computer screen with text

Description automatically generated

#### Delete ToDo

A computer code on a black background

Description automatically generated

## Third party libraries

1. SQFlite – for in memory sql lite data base
2. Flutter\_slidable – to have slidable tiles in contact list view
3. Path provider – to resolve paths easily in different platforms
4. Device preview – to test application in different screen sizes via browser
5. Flutter SVG – load svg images

## UI / UX Design

A screenshot of a phone

Description automatically generatedA screenshot of a phone

Description automatically generated

A screenshot of a phone

Description automatically generatedA screenshot of a phone

Description automatically generatedA screenshot of a phone

Description automatically generated

## Test case

This a test case which is written to test main ToDO details page

A screen shot of a computer program

Description automatically generated

## Possible issue and errors in applications

* Design is mostly created for mobile applications , might not be align with Tablet devices .
* Currently using local sql lite data base , there is potential data loss on uninstallation .

## References

1. <https://docs.flutter.dev/>
2. <https://docs.flutter.dev/cookbook>
3. <https://www.atlassian.com/git/tutorials>