STM JOURNALS

International Journal of
Electronics Automation

Volume 2, Issue 1, 2024
January–June
DOI (Journal): 10.37591/IJEA

https://journals.stmjournals.com/ijea

Review IJEA

A Comprehensive Guide to Application Development Using Flutter

Atin Sharma[1], Ravishankar Gupta[2], Bharat Raj Yadav[3], Sameer Awasthi[4,*]

Abstract

Developers are forced to either write the same application many times for other operating systems or use a similar but inferior solution that compromises native performance and accuracy in favor of portability. Flutter is a toolkit developed by Google that helps programmers create apps that look and feel great on different devices like phones, tablets, and computers. What is special about Flutter is that one can write code once and use it to build apps for different platforms like iOS, Android, and even web browsers. It is like having a magic wand to make apps that work everywhere! And all this magic happens using a programming language called Dart. So, with Flutter, developers can make apps quickly and easily without having to write separate code for each device. The demand for applications that can seamlessly run on multiple operating systems like iOS and Android has skyrocketed. Developers face the challenge of either writing separate codebases for each platform or opting for cross-platform solutions that might sacrifice some native performance for the sake of portability. Developers can construct robust and high-performing iOS and Android mobile apps with the open-source Flutter SDK. With cross-platform mobile app development becoming more and more important, Google's Flutter open-source UI toolkit is a formidable competitor. This research looks at the advantages and disadvantages of Flutter for creating cross-platform mobile applications. The objective is to assess how well the framework can accelerate app development and improve app performance across several platforms. A thorough investigation was carried out by creating Flutter prototype apps, examining the development process, and contrasting the performance metrics with those of conventional cross-platform frameworks. Our research shows that Flutter offers a consistent user experience on both iOS and Android while also drastically cutting down on development time. A possible paradigm change in cross-platform mobile app development is presented by Flutter. It can take the lead in the industry with sustained community support and advancements in native integrations. Hot reloading operates by adding new source code files to the Dart Virtual Machine that is currently running.

Keywords: Virtual machine, Flutter, iOS, Android application, UI toolkit, operating system, Dart

***Author for Correspondence**
Sameer Awasthi
E-mail: hodbansal.aiml@gmail.com

[1-3]Student, Department of Computer Artificial Intelligence, Bansal Institute of Engineering and Technology, Lucknow, Uttar Pradesh, India
[4]HOD, Department of Computer Artificial Intelligence, Bansal Institute of Engineering and Technology, Lucknow, Uttar Pradesh, India

Received Date: April 25, 2024
Accepted Date: May 10, 2024
Published Date: June 29, 2024

Citation: Atin Sharma, Ravishankar Gupta, Bharat Raj Yadav, Sameer Awasthi. A Comprehensive Guide to Application Development Using Flutter. International Journal of Electronics Automation. 2024; 2(1): 35–41p.

INTRODUCTION

The search for effective cross-platform frameworks never stops in the ever-changing world of mobile application development. Google's open-source UI toolkit Flutter has become a disruptive force, bringing with it the possibility of paradigm changes in the way developers approach creating cross-platform mobile apps. Flutter aims to provide a uniform and excellent user experience across a range of platforms in addition to streamlining the development process with its distinctive architecture and cutting-edge capabilities. The importance of mobile applications in our daily lives is becoming more and more important. Since

November 2016, mobile devices have generated 48.19% more network traffic than desktop or laptop computers (47%) [1].

To reach the widest audience, a mobile application must effectively cater to two major platforms: Android and iOS. Historically, mobile app developers have grappled with the challenge of developing applications that seamlessly operate on both platforms. Traditional methods often necessitate separate codebases for each platform, resulting in time-consuming development, high resource requirements, and inconsistencies in user interface and performance. Flutter, built on the Dart programming language, presents an innovative solution by enabling developers to craft a single codebase that compiles to native ARM code. This approach ensures a native-like experience across both iOS and Android platforms. This paper explores the core elements of Flutter as a cross-platform mobile app development framework. It examines its architecture, essential features, and the potential impact of adopting Flutter on development workflows and user satisfaction. Through practical examples and insights gained from real-world applications, we aim to provide a comprehensive understanding of Flutter's strengths, challenges, and its role in shaping the future of mobile app development. This paper explores the fundamental aspects of Flutter as a cross-platform mobile app development framework. It delves into architecture, key features, and the potential impact of adopting Flutter on development workflows and end-user satisfaction. By delving into practical examples and drawing insights from real-world applications, our aim is to offer a comprehensive understanding of Flutter's strengths, challenges, and its role in shaping the future of mobile app development. As we embark on this exploration of Flutter, it becomes evident that its adoption has the potential to revolutionize how developers conceptualize, construct, and maintain cross-platform mobile applications. Our research seeks to provide insights into the landscape of cross-platform frameworks, including those resembling React Native and Flutter, which have been implemented by numerous companies in the past. However, these predecessors have often fallen short of meeting the demands of industrial development. Despite the shortcomings of previous frameworks, React Native and Flutter, backed by Facebook and Google, respectively, have garnered significant attention and optimism about their prospects. This article serves as a comprehensive guide to application development using Flutter. We will delve into the fundamentals of Flutter, explore its key features, and discuss how it simplifies and streamlines the development process for mobile applications. However, these predecessors have often fallen short of meeting the demands of industrial development.

Key Features of Flutter

Unified Codebase

According to WalkingTree Technologies, Flutter drastically cuts down on development time and resources by enabling developers to create a single codebase that functions flawlessly across a variety of platforms, including iOS, Android, web, and desktop.

Hot Reload

This feature expedites the development process by allowing developers to view the effects of code changes nearly quickly without having to restart the application (Xavor Corporation).

Rich Widget Library

Flutter's widget library consists of Cupertino (iOS-style) and Material Design components, which facilitate the creation of highly customizable and aesthetically pleasing user interfaces (UIs) (Successful Digital).

Performance

Flutter apps offer near-native performance on all platforms since they are compiled straight to machine code (Xavor Corporation).

Getting Started with Flutter

Installation

Installing the Flutter SDK from the official Flutter website is the first step in getting started with Flutter installation. Ascertain that your system satisfies the prerequisites and is equipped with utilities such as git, bash, and unzip (Xavor Corporation).

Configuring the Environment for Development

For an improved development experience, use an IDE such as Visual Studio Code, Android Studio, or IntelliJ IDEA with the required Flutter and Dart plugins (Xavor Corporation).

Starting a New Initiative

To start a new Flutter project, use the flutter create command. To get your development underway, this command creates the fundamental project structure (Xavor Corporation).

Comprehending the Project's Structure

The primary entry point is lib/main.dart, where you use the widget-based architecture of Flutter (Xavor Corporation) to define the functionality and organisation of your app.

FLUTTER

Flutter is a toolkit developed by Google that allows developers to create apps for mobile, web, and desktop platforms using a single codebase. It uses the Dart programming language and offers a collection of ready-to-use widgets for building user interfaces. What makes Flutter stand out is its reliance on the device's own widgets rather than web views, which results in high-performance applications. Flutter's architecture involves compiling Dart code into native code using Android's Native Development Kit (NDK) and LLVM for iOS. Its hot reload feature, known as Stateful hot reload, speeds up development by allowing developers to instantly see changes made to the code without losing the app's state or structure [2]. To get started with Flutter, developers need to install the Flutter SDK and Dart, set up their development environment, and use the Flutter CLI to create and run their applications.

Key Features of Flutter Quick Development

Flutter's hot reload function makes development quicker and more effective by enabling developers to view changes they make to the code right away without having to restart the application.

Expressive UI

Developers can construct stunning and expressive user interfaces with Flutter because of its extensive collection of configurable widgets [3].

Native Performance

Flutter apps have excellent performance and fluid animations since they are compiled straight to native ARM code.

Single Codebase

Developers can save time and work by writing code only once and have it deployed across various platforms with Flutter.

Rich Ecosystem

Flutter has an expanding ecosystem of plugins and packages that increase its capabilities and make it possible to integrate it with other platforms and services.

DART

Dart is a programming language developed by Google, widely used for building various applications, including web, mobile, and desktop. It is designed with a syntax like languages like C, C++, or Java,

making it easy for developers familiar with those languages to pick up. Dart follows object-oriented principles, allowing for the creation of classes, objects, and inheritance.

One of Dart's strengths is its static typing, which means that variable types are declared at compile-time, helping catch errors early in the development process. Additionally, Dart supports asynchronous programming, making it suitable for handling tasks that require waiting for operations like I/O or network requests.

Dart is notably used in the Flutter framework, also developed by Google, which enables developers to create native applications for multiple platforms from a single codebase. In Flutter development, Dart serves as the primary language, contributing to the framework's efficiency and performance.

PROPOSED METHODOLOGY

The framework for handling routine research chores is made to give researchers an organized and effective workflow. To make sure that everyday actions are in line with the study's larger objectives, it starts with the formulation of precise research goals and objectives. Essential elements include job planning and prioritization, which entail making a thorough daily task list and using project management software to arrange and monitor work. The systematic management of literature review operations makes use of reference management technologies to ensure efficient organization and citation [4]. A structured approach is used for collecting and organizing data, while robust methods are employed for cataloguing. Data analysis is made easier with the right tools and detailed documentation. Writing and documentation follow a dedicated strategy, including version control for drafts and records. Effective collaboration is ensured through a designated platform, enabling smooth communication among team members. Regular progress checks, automation tools, and contingency plans help manage challenges efficiently. A well-organized workspace and adaptability contribute to daily task management [5]. Documentation integrity and version control are maintained, with quality assurance mechanisms and feedback loops in place. Project milestones are tracked against a clear timeline for timely completion. Continuous learning, skill development, and researcher well-being are emphasized. In short, this system offers a comprehensive and flexible framework for navigating daily research tasks. Regular evaluations and refinements ensure optimal efficiency and productivity throughout the research process.

PROPOSED SYSTEM

The two components of our system are "the client" and "the server." A mobile device running iOS or Android is used to implement the client side. Windows IOS is used to deploy the server side [6]. the Android 10 smartphone's client side.

Android System

The client side on Samsung Android 10 mobile phone is shown in Figure 1.

Workflow Diagram

We created a straightforward Todo application [7]. The user can input information about the task they intend to complete and when it will be completed using the system [8]. In this the system is designed to carry out every CRUD task. For this project, REST services have also been put into place [9, 10]. The user can perform these operations like adding task, delete task, update task, and view all tabs (Figure 2). Process flowchart of proposed system is shown in Figure 3.

State Diagram

In this application, users must log in or sign in to access the website. New users are required to register and sign up first. Once registered and logged in, the user interface will be displayed on the screen. Users can input information about the tasks they intend to complete and their respective deadlines. Additionally, the website offers an option to edit their profile. After this, users can add tasks and perform other operations.

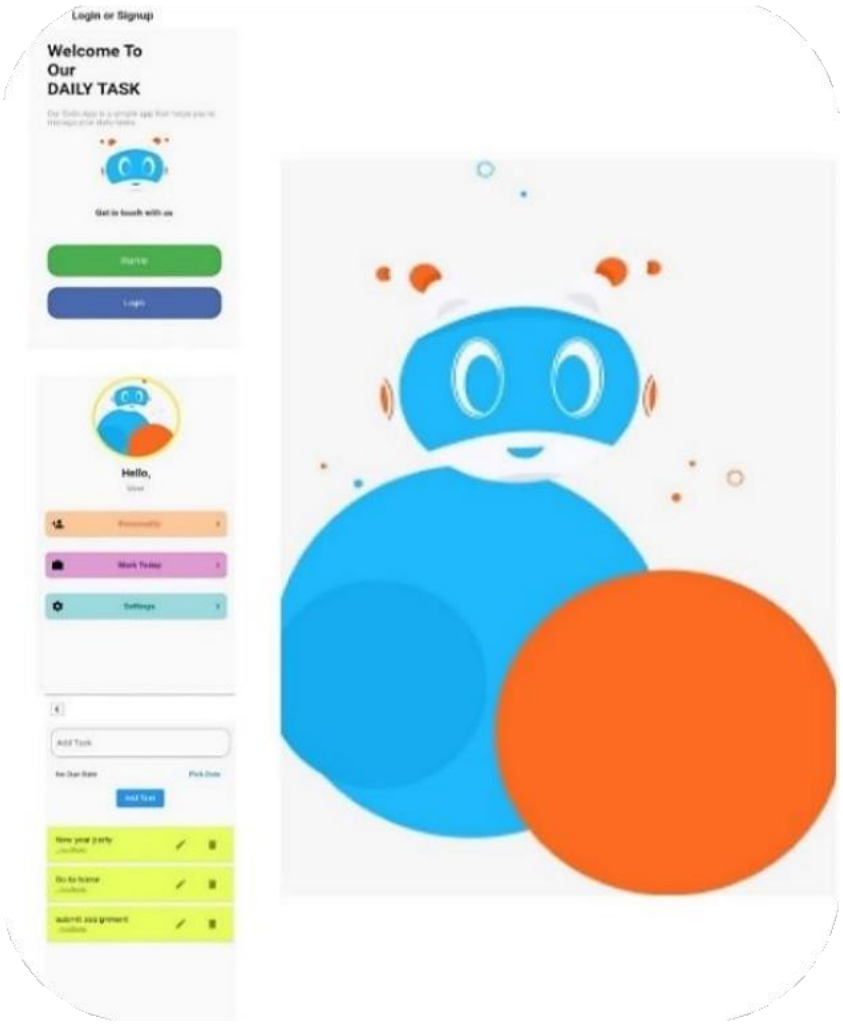

Figure 1. Client side user interface on mobile phone.

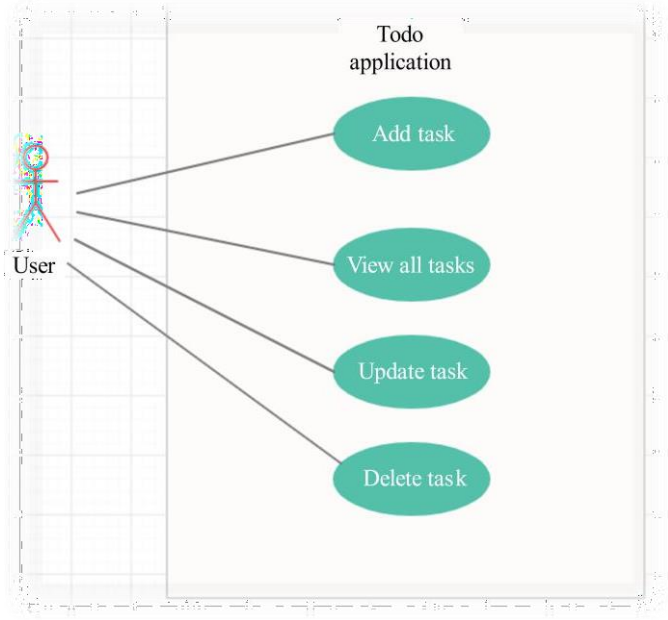

Figure 2. Tools application.

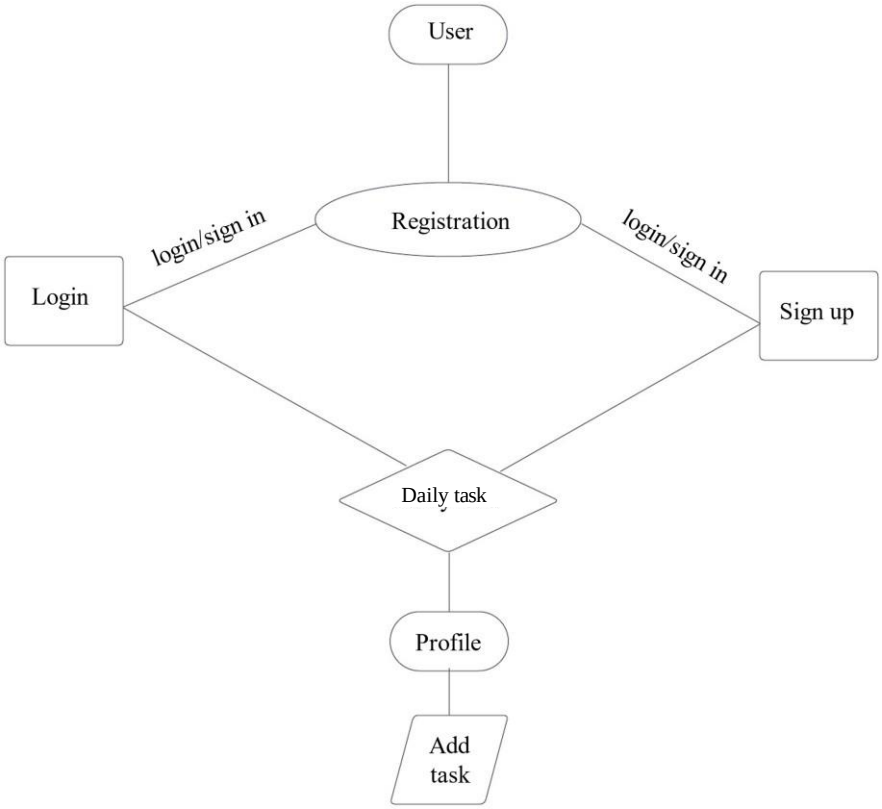

Figure 3. Process flowchart of proposed system.

CONCLUSION

Flutter is a powerful framework for building high-quality, cross-platform applications with a single codebase. Its fast development cycle, expressive UI, and native performance make it an ideal choice for developers looking to create beautiful and performant apps for a variety of platforms. As we wrap up the development phase of our Flutter daily task app, it is gratifying to reflect on the journey we have undertaken. This project aims to provide a streamlined and intuitive platform for task management, and we are pleased to report that we have achieved our objectives. Today's environment does demand the development of cross-platform mobile applications, and developers are always looking for ways to strike the ideal balance between portability, performance, and user experience.

REFERENCES

1. Aycock J. A brief history of just-in-time. ACM Comput Surv. 2003; 35 (2): 97–113. doi: 10.1145/857076.857077.
2. Cronbach LJ. Coefficient alpha and the internal structure of tests. Psychometrika. 1951; 16 (3): 297–334.
3. Bagozzi RP, Yi Y, Phillips LW. Assessing construct validity in organisational research. Admin Sci Q. 1991; 36 (3): 421–458.
4. Agarwal R, Prasad J. The role of innovation characteristics and perceived voluntariness in the acceptance of information technologies. Decis Sci. 1997; 28 (3): 557–582.
5. Bellotti V, Ducheneaut, N, Howard M, Smith IE. Taskmaster: recasting email as task management. Working paper. Palo Alto, CA, USA: Palo Alto Research Center; 2002.
6. Paleczny M, Vick CA, Click C. The Java Hotspot server compiler. In: Proceedings of the Java Virtual Machine Research and Technology Symposium. Vol. 1. Monterey, CA, USA: April 23–24, 2001.
7. Morales-Morell A. Usability Aspects of a Location-Aware TODO List Application. Master's Thesis. Mayaguez, Puerto Rico: University of Puerto Rico; 2001.

8. Adams R. Decision and stress: cognition and e-accessibility in the information workplace. Springer Universal Access Inform Soc. 2007; 5 (4): 363–379.
9. Belkin NJ. Anomalous states of knowledge as a basis for information retrieval. Can J Inform Sci. 1980; 5: 133–143.
10. Pugh EW, Johnson LR, Palmer JH. IBM's 360 and Early 370 Systems. Cambridge, MA, USA: MIT Press; 1991. pp. 160–161.

