STM JOURNALS

Journal of
Mobile Computing, Communications & Mobile Networks

ISSN: 2349-901X
Volume 11, Issue 3, 2024
September–December
DOI (Journal): 10.37591/JoMCCMN

https://journals.stmjournals.com/jomccmn

Review JoMCCMN

Performance and Stability in Cross-platform Application Development

Lekha Achuth[1,*], Naresh J.[2]

Abstract

In the field of construction and contracting, effective communication between contractors and clients is crucial to the successful implementation of projects. This mobile-based application serves as a critical medium to build an open, friendly, and transparent environment between the two stakeholders involved. The application offers contractors a deep search-and-select platform where they can offer quality service and communicate comprehensively with clients well before the final contracts are closed. The platform allows contractors to understand the requirements of the project, what a client is expecting, and what resources are necessary to begin with. The application also provides a comprehensive overview of the project details to the client, including timelines, resource needs, and an itemized breakdown of the expenses involved. This level of transparency affords the client the ability to make relevant decisions regarding their needs and express those needs professionally. Transparency according to the application ensures that contractors and clients share the same understanding regarding the project in question, which boosts collaboration and minimizes the possibility of any misunderstandings. This shared understanding is important in ensuring highly satisfactory outcomes, increasing trust, and allowing smooth and efficient project execution. Finally, the application improves interaction among contractors and clients harmoniously and cooperatively which ensures project objectives are realized with effect.

Keywords: Business-to-Customer (B2C), flutter, Django REST framework, react, cross-platform development

INTRODUCTION

The difficulties involved in achieving maximum performance and stability in the process of cross-platform application development are significant and require careful attention from developers. This paper highlights the challenges and primary causes that developers face in building applications for both Android and iOS, which can seamlessly perform across a wide range of platforms. Developers need to balance the need for consistency between platforms with the necessity of ensuring a user experience that is as fast, fluid, and dependable as possible. This becomes incredibly challenging when working with cross-platform frameworks such as Flutter and React Native. By studying performance metrics and understanding the drivers of stability, developers are better positioned to make informed choices regarding which framework best meets their project needs.

***Author for Correspondence**
Lekha Achuth
E-mail: lekha@pes.edu

[1]Associate Professor, Department of Computer Applications, PSE University, Bengaluru, Karnataka, India
[2]Student, Department of Computer Applications, PSE University, Bengaluru, Karnataka, India

Received Date: August 16, 2024
Accepted Date: September 16, 2024
Published Date: October 11, 2024

Citation: Lekha Achuth, Naresh J. Performance and stability in Cross-platform Application Development. Journal of Mobile Computing, Communications & Mobile Networks. 2024; 11(3): 28–34p.

The overall documentation, be it for React Native or Flutter, would be beneficial for developers and organizations targeting cross-platform app development. In addition, these resources not only outline the technical differences between the two

frameworks but also provide guidelines on various optimization techniques that can improve performance and enhance user experience. Both frameworks have different pros in place, but knowing how to use them correctly in specific cases is a key point.

There is also a mobile app that seeks to help the construction industry improve communication between contractors and clients through easy collaboration on projects. The app allows for the creation of a contract by easily picking tasks while enabling deep discussions between the parties involved in a specific project. It is set up to handle the entire project cycle. More specifically, it targets smaller projects, ensuring that even the smallest initiatives benefit from clear communication and structured planning.

The application provides a platform on which transparency pervades the environment, allowing contractors and clients to communicate effectively, thereby reducing the chance of miscommunication or disputes. It allows for discussions over project details, such as expected expenses, schedules, and work conditions, ensuring that both parties are on the same page in advance, which is very important for the smooth success of the entire project. The application acts as an intermediary, ensuring more transparency and building mutual understanding, which is important for satisfactory outcomes.

Moreover, the service will ease project decisions on the part of contractors, while easier decision-making is allowed on the part of the client. Contractors can select projects that best suit their skills and preferences, while clients can be accorded all opportunities to negotiate terms that best suit their needs and budgets. This place optimizes not only the process of contract development but also communication since both parties entering the contract are well-informed and in agreement.

Therefore, the key objective of the application is to facilitate the negotiation of contracts while developing a mutual understanding between contractors and clients. By achieving this objective, the application also tries to ensure that there is a greater likelihood of successful completion of projects and heightened customer satisfaction. This application is specifically meant to address each step in the construction project life cycle, with keen attention to the special needs of small enterprises and projects. However, its versatility allows it to tackle a wide range of construction projects and, as such, makes it suitable for the different needs that arise within the entire construction sector.

Fundamentally, this platform represents a versatile mechanism for overseeing interactions between clients and contractors, particularly in the context of small-scale construction initiatives. It effectively confronts the unique obstacles encountered by small enterprises, while maintaining a sufficiently extensive framework to address a wide array of project needs. Consequently, it serves as an optimal instrument for improving communication, fostering transparency, and facilitating favorable project results throughout the construction sector.

LITERATURE REVIEW

The development of cross-platform mobile applications has evolved as one of the most effective methods of application development, running smoothly on both Android and iOS platforms. Within this domain, React Native and Flutter have emerged as the two leading frameworks. The React Native, from Facebook, and Flutter, from Google, are marked with various architectural differences, along with the respective advantages and disadvantages that are relevant for developers and organizations when it comes to evaluating cross-platform options.

Flutter represents a freely available open-source framework developed by Google that offers an extensive array of predesigned widgets for the development of cross-platform applications. The documentation associated with Flutter is thorough and provides developers with a direct approach to creating visually appealing and high-performance applications. The framework utilizes the Dart programming language, which includes the inherent support for object-oriented programming principles. Conversely, React Native employs JavaScript, a language that enjoys widespread usage and presents certain advantages, while also introducing some performance overhead when compared to Flutter.

Kishore et al. (2022) also noted that during the assessment of applications for Android platforms, React Native needs higher storage capacity and resource allocation compared to Flutter. In their findings, they show that applications based on React Native have higher power consumption and a higher demand for CPU resources, as portrayed by three different measurement tools. The performance difference is relevant to developers looking forward to enhancing applications for better battery life and storage efficiency [1].

Wu (2018) conducted an in-depth comparative study between native acts and Flutter through a discussion of their basic principles, advantages, and limitations. Issues such as performance, efficiency, and learning curves are discussed for each of them. This thesis will provide rich insights into the reengineering of an existing React Native app using Flutter, which will help developers understand the practical differences between the two in real-life situations [2].

Although cross-platform frameworks present an appealing opportunity for code sharing across various platforms, White (2018) underscores the benefits associated with native application development, especially the uninterrupted access to APIs that are specific to each platform. Native applications enable developers to engage directly with features unique to the device, including the camera, GPS, accelerometer, and microphone, thereby offering a more tailored and cohesive user experience. This, in turn, enables a niching of platform-specific capabilities that often results in better performance and a more "native" feel than may be possible with cross-platform frameworks [3].

In the realm of web application development, Jazayeri (2007) showed the great use of component-oriented development methodologies in current web applications. The reason for this is the use of reusable and standardized components, such as search and tagging functionalities, that give rise to greater collaboration and user interaction in web applications. This trend has been followed by frameworks such as Django, which was built upon the Model-View-Controller(MVC) architecture [4].

Varghese (2019) explains that the architecture of Django effectively separates concerns, which enables developers to have simpler management of the data layer with models, a user interface with views, and a presentation layer with templates. This architecture enables the maintainability and scalability of applications built on Django attributes, which are very important in building robust and long-lasting web solutions [5].

A study by Suryadiputra Liawatimena et al. (2018) evaluated the quality of code in Django by applying software metrics using tools such as Radon and Pylint to measure complexity and maintainability in Django codebases. In this respect, metrics concerning code quality are fundamental for assessing the long-term viability and maintainability of web applications [6].

Thakur et al. (2023) emphasized the differentiation between front-end and back-end development within web applications, especially in the context of frameworks such as Django. Front-end developers focus on the design and functionality of user interfaces, whereas back-end developers oversee data-transfer processes and computational tasks. Collaborative interaction between front-end and back-end development is crucial for achieving seamless and effective web functionalities, particularly in intricate applications [7].

Annenkov et al. (2013) have discussed domain-specific languages. They describe a textual language that can be utilized to model domains inside web frameworks, such as Django. In this paper, they describe the use of rewriting rules and strategies to generate syntax trees and perform their transformation, and therefore one with efficient code generation. This helps automate parts of the development, thereby supporting the handling of the complex [8].

The emergence of B2C e-commerce platforms has transformed purchasing processes in the retail industry. Paris et al. (2016) highlighted the implementation processes of B2C e-commerce systems and

their extended role in connecting business-to-customer transactions. Large companies, namely Tesco, Zara, and Samsung, have introduced mobile applications within their business operations that enable them to facilitate smoother transactional processes and increase their marketing exposure [9].

Business-to-customer (B2C) e-commerce is a widely embraced paradigm in IT applications. It highlights the prevalence of B2C platforms in major retail establishments, such as Tesco, Zara, and Samsung, among others. These entities have successfully integrated mobile applications into their business models to facilitate transactions and enable the sale of products and services directly to customers [9]. By leveraging mobile technology, B2C e-commerce platforms aim to enhance customer convenience, streamline purchasing processes, and expand their market reach. This study offers insights into the growing significance of B2C e-commerce in modern business landscapes, and its implications for both consumers and retailers [10].

A very interesting case study could be Urban Company, a large home services platform that is operational in several countries around the globe, including India, the UAE, Australia, and Singapore. Unlike many other service platforms, urban companies follow a full-stack approach to service delivery, integrating native and cross-platform mobile applications. This approach allows a company to control every touch point in the experience of the service, thereby building trust with customers and guaranteeing high service delivery. Despite initial growth challenges, the Urban Company's commitment to a full-stack model positions it as an industry leader in customer satisfaction (Krishna G. Palepu, 2020) [11].

Urban Companies are prominent home service platforms operating across Asia, offering services in countries such as India, the UAE, Australia, and Singapore. Unlike typical marketplace platforms, urban companies adopt a full-stack and seamless native and cross-platform mobile app, assuming end-to-end responsibility for service delivery [11]. By actively participating in service delivery, the Urban Company prioritizes customer trust and satisfaction, setting itself apart from its competitors. Despite initial challenges and slower growth, the company's founders believe in the long-term viability of this strategy, and urban companies plan to strengthen their differentiation by doubling down on their full-stack approach, aiming to deepen customer trust and loyalty [11]. It explores expansion into new markets and verticals by focusing on maintaining high-quality standards and fostering customer trust to drive sustainable growth and success.

METHODOLOGY

Understand the requirements of the API, including endpoints, data models, authentication, and permissions.

Design the API endpoints, data models (using Django models), and overall structure of the application. Define the URL patterns in Django's urls.py file to map endpoints to the views. Create views (or view sets for RESTful APIs) to handle incoming requests, process data, and return responses. Views can utilize Django's class-based or function-based views, as shown in Figures 1–4.

RESULTS

Figures 5 and 6 illustrate the process by which contractors can access the project details listed by the clients. Contractors can initiate communication with clients via chats to discuss projects. If the client is satisfied with the communication and the proposed deal, they can accept the request.

```
Class Serializer(serializers.ModelSerializer):
  class Meta:
    model = model
    fields = '   all   '
```

Figure 1. Serializer.


```
@csrf_exempt
Def Api(request):
   if request.method=='MethodType':
      try:
         return Response
      except:
         return Response
```

Figure 2. API Request View.

```
urlpatterns = [
   path('callApi/', Api)
]
```

Figure 3. URL Pattern.

```
class newModel(models.Model):
   name=models.CharField()

   def __str__(self) -> str:
      return self.name
```

Figure 4. Model.

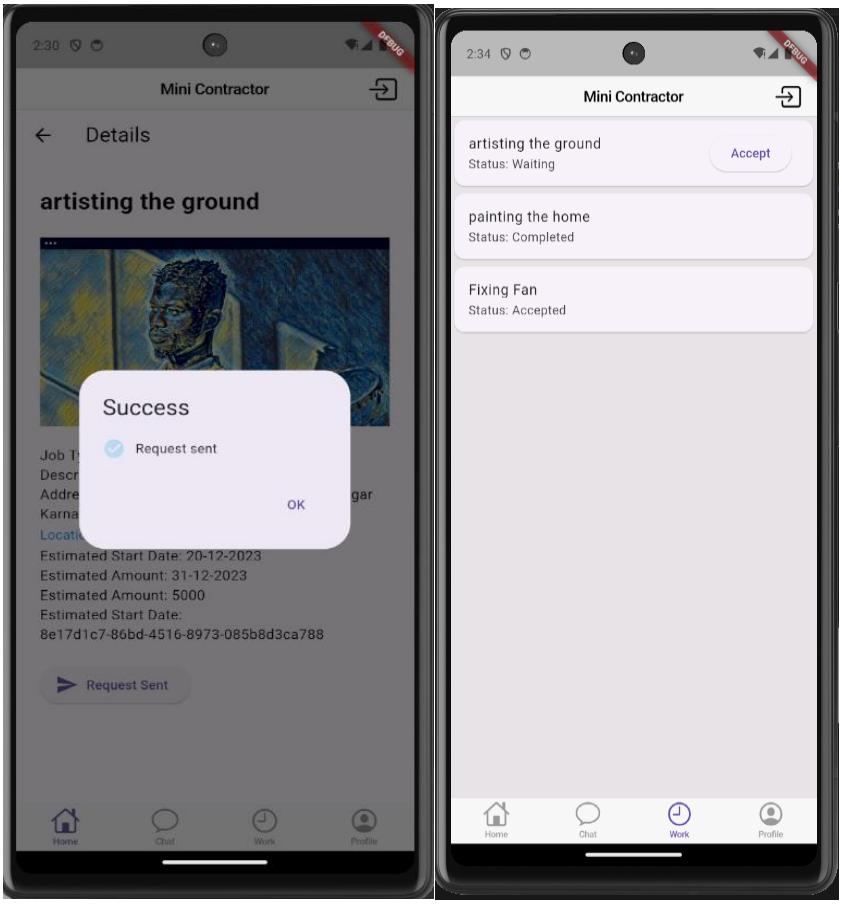

Figure 5. Send and accept requests.

Once accepted, the contractor can initiate the project. Upon completion, the client can leave feedback to finalize the procedure and close the project. In case of any issue, both the client and contractor can raise complaints with comments, prompting admin intervention to resolve the problem effectively.

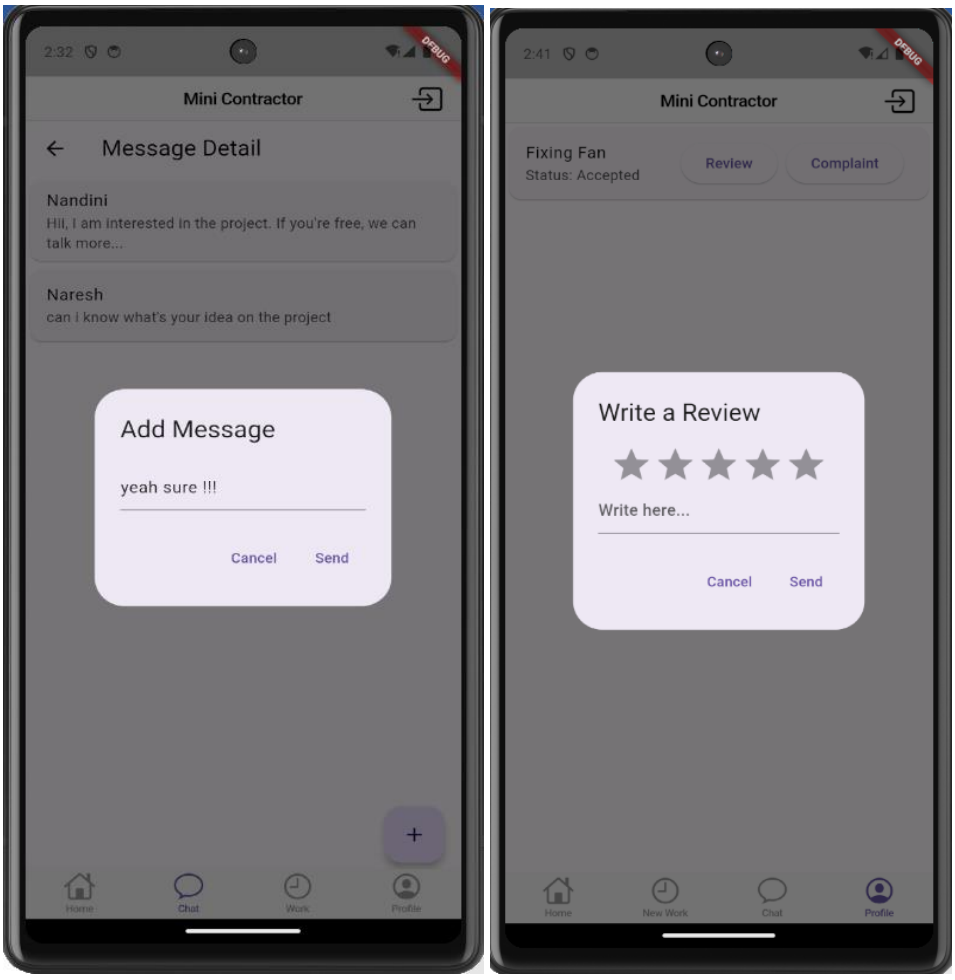

Figure 6. Add message and review.

DISCUSSION

API requests enable interactions with remote servers or services to access, send, or manipulate data. They offer several benefits, including flexibility and customization options for building tailored applications as well as scalability by leveraging existing APIs instead of building every feature from scratch.

CONCLUSION

This mobile application provides a cross-platform for small-scale contractors to grow their businesses by acquiring projects in a better way. It also provides a user-friendly environment.

Both the Flutter and Django frameworks were tested in different environments and not only the UI but also every possible factor responsible for the high performance. This performance and API integration. Using a streamlined project acquisition approach for the growth of small-scale contractors.

Acknowledgment

I thank my guide, Professor Dr. Lekha A, for the valuable advice and guidance. I also thank my Head of Department of Computer Applications, Dr. Veena S, for this opportunity and support.

REFERENCES

1. Kishore K, Khare S, Uniyal V, Verma S. Performance and stability comparison of react and flutter: Cross-platform application development. In: International Conference on Cyber Resilience (ICCR); 2022. p. 1–4. DOI: 10.1109/ICCR56254.2022.9996039.

2. Wu W. React Native vs Flutter. Cross-platform mobile application frameworks. 2018;1:25. Available from: https://www.theseus.fi/bitstream/handle/10024/146232/thesis.pdf.
3. White J. Going native (or not): Five questions to ask mobile application developers. Australas Med J. 2013;6:7–14. DOI: 10.4066/AMJ.2013.1576, PubMed: 23424610.
4. Jazayeri M. Some trends in web application development. In: Future of Software Engineering (FOSE'07); 2007. p. 199–213. DOI: 10.1109/FOSE.2007.26.
5. Varghese JJ. DJANGO web framework technology. Scribed. 2019. Available from: https://www.scribd.com/document/405415167/IEEE-Journal-on-Django.
6. Liawatimena S, Hendric Spits Warnars HL, Trisetyarso A, Abdurahman E, Soewito B, Wibowo A, et al. Django web framework software metrics measurement using radon and pylint. In: Indonesian Association for Pattern Recognition International Conference (INAPR); 2018. p. 218–22. DOI: 10.1109/INAPR.2018.8627009.
7. Thakur P, Jadon P. Django: Developing web using Python. In: 3rd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE); 2023. p. 303–6. DOI: 10.1109/ICACITE57410.2023.10183246.
8. Annenkov DV, Cherkashin EA. Generation technique for Django MVC web framework using the stratego transformation language. In: 36th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO); 2013. p. 1084–7.
9. Paris DL, Bahari M, Iahad NA. Business-to-customer (B2C) electronic commerce: An implementation process view. In: 3rd International Conference on Computer and Information Sciences (ICCOINS); 2016. p. 19–24. DOI: 10.1109/ICCOINS.2016.7783182.
10. Bin C, Qiang Y. B2C e-commerce's customer relationship management based on the long tail. In: International Conference on Management Science and Engineering; 2007 Aug 20. p. 204–9.
11. Palepu KG. Urban Company-Case-Faculty & research-Harvard Business School. Hbs.edu. 2020; Vols. 121–041. Available from: https://www.hbs.edu/faculty/Pages/item.aspx?num=59382.

