

ChatMate

Raghav Mundhara
D15B - 41

1. Introduction
2. Problem Statement
3. Objective
4. Techstack
5. Features
6. Implementation
7. Future Scope
8. Literature Survey
9. Conclusion
10. References

Introduction

Welcome to the unveiling of a groundbreaking chat application, crafted meticulously using Flutter and Firebase. In today's fast-paced digital landscape, effective communication is paramount. This application is poised to redefine connectivity, leveraging Flutter's intuitive user interface toolkit and Firebase's robust backend infrastructure. From real-time messaging to secure authentication and a range of unique features like "Invisible Ink" for discreet conversations, this app promises to transform the way users engage.

Problem Statement

1. In today's digital communication landscape, privacy and confidentiality are paramount concerns for users of chat applications.
2. While traditional messaging platforms offer encryption and security measures, there remains a need for enhanced privacy features that allow users to share sensitive information discreetly.
3. Users face challenges in securely sharing confidential information within chat conversations, leading to concerns about data privacy and security.



Objectives

1. **User Engagement:** Increase user engagement by providing a seamless and interactive chat experience through features such as real-time messaging, multimedia sharing, and emoticons.
 2. **Scalability:** Ensure the chat application can accommodate a growing user base and handle increasing message volumes without compromising performance or reliability.
 3. **Cross-Platform Compatibility:** Develop the chat application to be compatible across multiple platforms, including mobile devices (iOS and Android), web browsers, and desktop environments, to maximize accessibility for users.
 4. **User Authentication and Security:** Implement robust user authentication mechanisms to verify user identities and ensure secure access to chat features. Employ encryption and other security measures to protect user data and communications from unauthorized access or interception.
-

TechStack

- Flutter



- Firebase



Features

1. User Authentication and Security
2. Invisible Ink
3. Integration with Cloud Storage
4. Profile Management



User Authentication and Security

Objective: Implement a secure user authentication system using email and password, ensuring user credentials are encrypted and protected against unauthorized access.

Key Result: Achieve a seamless login experience for users while maintaining robust security measures to safeguard user accounts and sensitive information.

Invisible Ink

Objective: Develop the "Invisible Ink" feature, allowing users to send confidential messages that can only be revealed by the recipient using a 4-digit secret PIN.

Key Result: Ensure that obscured messages remain encrypted and hidden from view until the correct PIN is entered, providing users with enhanced privacy and control over their communications.

Integration with Cloud Storage

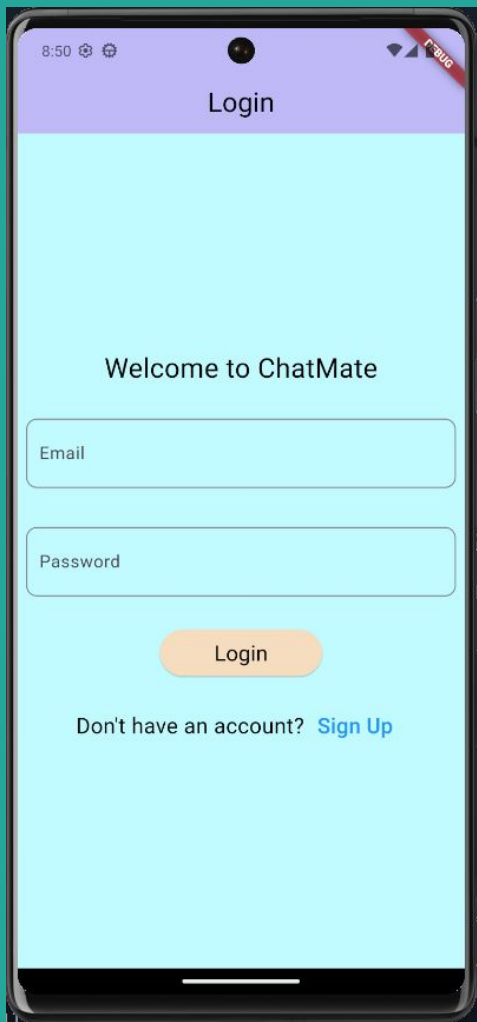
Objective: Integrate cloud storage functionality to enable users to upload and store profile pictures and other media files securely.

Key Result: Implement seamless integration with a cloud storage service (e.g., Firebase Storage) to allow users to upload, retrieve, and manage their profile pictures and other media assets from any device.

Profile Management

Objective: Provide users with the ability to update their bio and profile picture within the chat application.

Key Result: Develop user-friendly interfaces for editing and updating user profiles, including options to modify bio descriptions, upload new profile pictures, and view/update personal information.



8:50

Login

Welcome to ChatMate

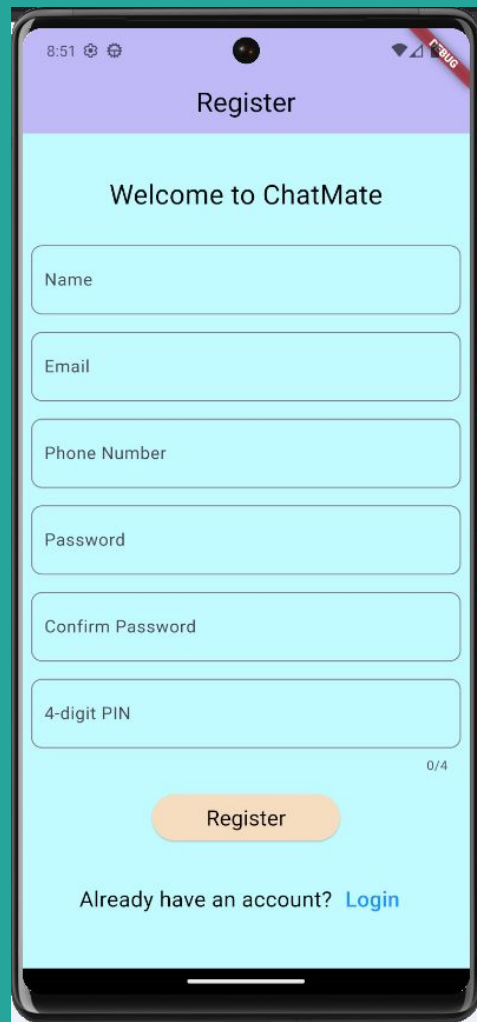
Email

Password

Login

Don't have an account? [Sign Up](#)

This is a mobile app login screen. It features a purple header with the title 'Login'. Below the header is a light blue background with the text 'Welcome to ChatMate'. There are two input fields: 'Email' and 'Password'. Below these is an orange 'Login' button. At the bottom, there is a link 'Sign Up' in blue text.



8:51

Register

Welcome to ChatMate

Name

Email

Phone Number

Password

Confirm Password

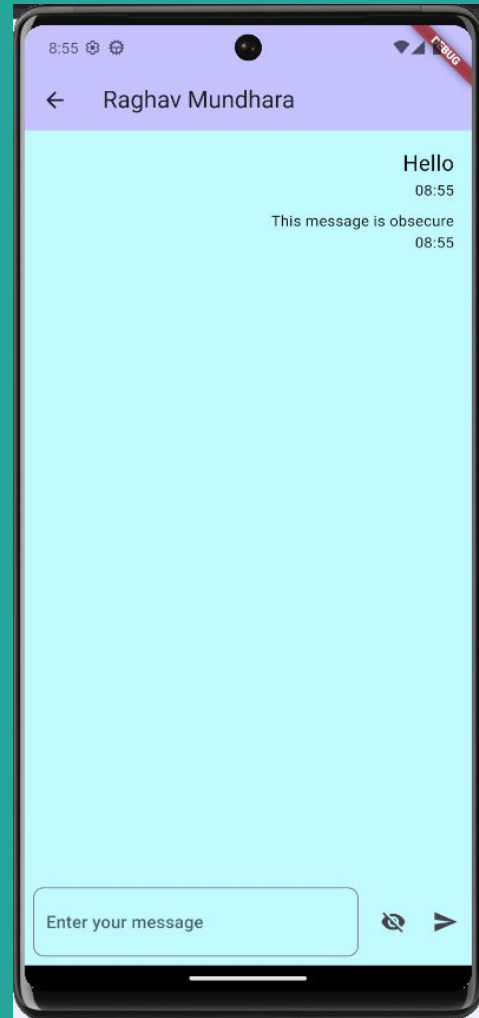
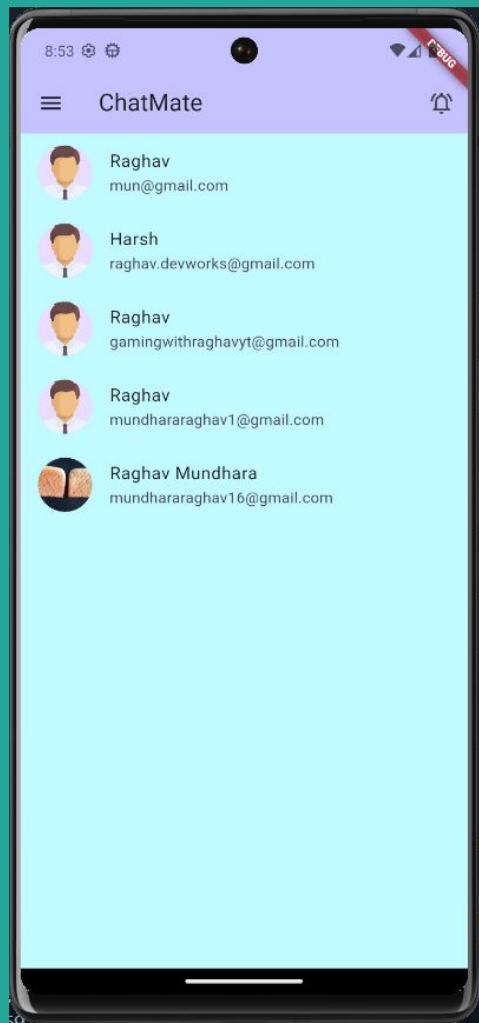
4-digit PIN

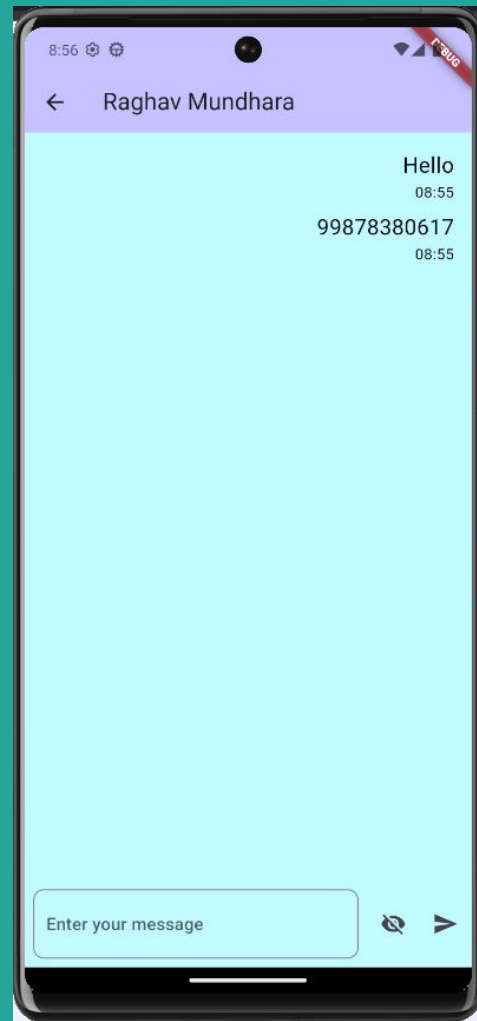
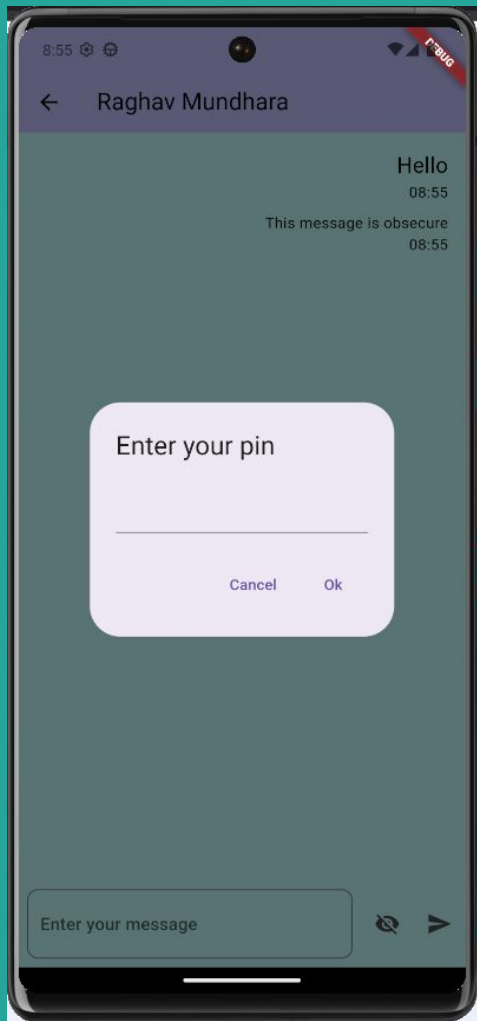
0/4

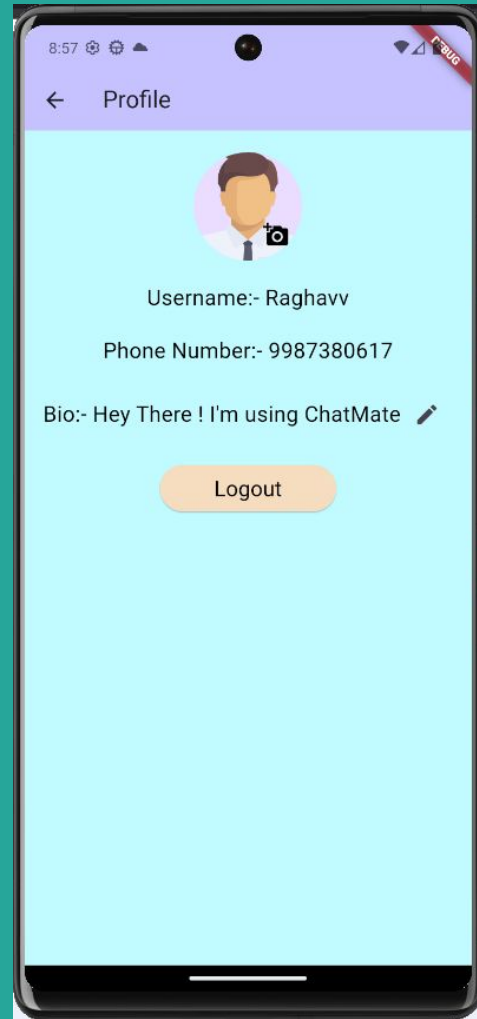
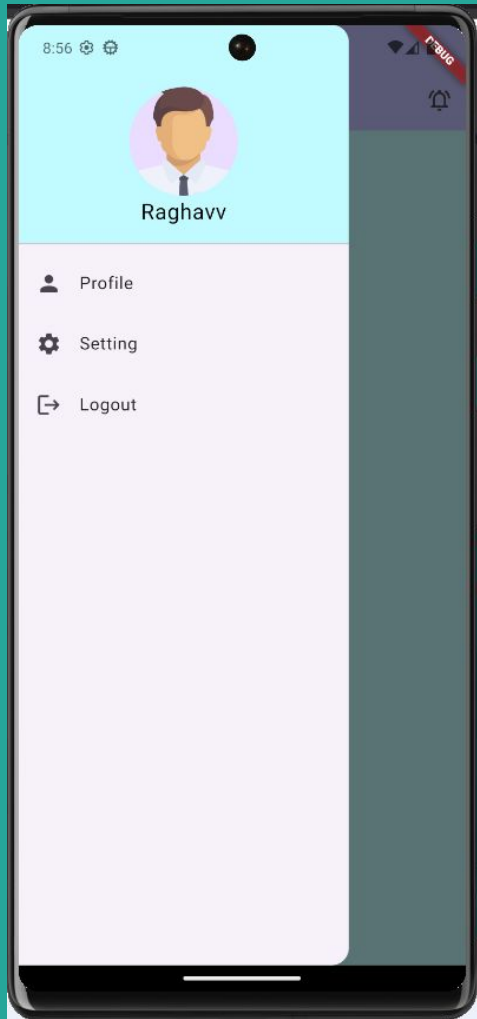
Register

Already have an account? [Login](#)

This is a mobile app registration screen. It features a purple header with the title 'Register'. Below the header is a light blue background with the text 'Welcome to ChatMate'. There are five input fields: 'Name', 'Email', 'Phone Number', 'Password', and 'Confirm Password'. Below these is a '4-digit PIN' field with a '0/4' character count indicator. Below the PIN field is an orange 'Register' button. At the bottom, there is a link 'Login' in blue text.







Future Scope

1. **Integration with Virtual Assistants:** Integrate virtual assistant functionalities to provide users with helpful features such as smart suggestions, automated responses, and natural language processing capabilities within the chat interface.
 2. **End-to-End Encryption:** Implement end-to-end encryption for all messages exchanged within the chat application to ensure maximum security and privacy for users' conversations.
 3. **Integration with Third-party Services:** Enable integration with popular third-party services such as calendar apps, task management tools, file sharing platforms, and productivity suites to enhance collaboration and workflow efficiency.
 4. **Mood-Based Chat Themes:** Implement a feature that allows users to select chat themes based on their current mood or preferences.
-

Literature Survey

[1] S. N. Reddy Lakkireddy, A. A. Thomas, T. S. Shree and T. Mamatha, "Web-based Application for Real-Time Chatting using Firebase," 2022 International Conference on Knowledge Engineering and Communication Systems (ICKES), Chickballapur, India, 2022, pp. 1-4, doi: 10.1109/ICKECS56523.2022.10060845.

The paper introduces a web-based chat application utilizing Firebase for backend support and ReactJS for frontend development. It addresses the growing need for remote communication platforms by offering real-time messaging capabilities accessible across various devices, including Android and iOS. The project aims to provide users with a seamless experience akin to popular messaging applications like Facebook and WhatsApp, emphasizing accessibility and usability on a global scale.

Literature Survey

[2]S. Prabhune and S. Sharma, "End-to-End Encryption for Chat App with Dynamic Encryption Key," 2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), Greater Noida, India, 2021, pp. 1361-1366, doi: 10.1109/ICAC3N53548.2021.9725597.

The paper discusses the implementation of end-to-end encryption in chat applications to enhance privacy and security for millions of users. It emphasizes the importance of addressing various security concerns to prevent potential threats despite the implementation of multiple security layers. The proposed solution employs dynamic key encryption using the MD5 algorithm, ensuring that only intended recipients can access the messages while protecting against attackers and unauthorized access. By generating unique keys for each message, the system enhances security and complexity, thereby safeguarding the chat application effectively.

Conclusion

Our chat application developed with Flutter and Firebase offers a seamless and secure platform for users to connect and communicate. With features like login using email and password, the unique "Invisible Ink" for confidential messaging, cloud storage integration, and profile management, we've created a versatile tool for modern communication needs.

Reference

1. <https://docs.flutter.dev/> (Flutter Documentation)
2. <https://firebase.google.com/docs> (Firebase Documentation)
3. <https://www.youtube.com/@AkshitMadan> (Youtube Tutorial)
4. <https://www.youtube.com/playlist?list=PL4cUxeGkcC9giLVXCHSQmWqIHc9BLXdVx> (Youtube Tutorial)

E-Commerce PWA

Raghav Mundhara

D15B - 41

What is PWA?

PWA stands for Progressive Web App. It's a type of application software delivered through the web, built using common web technologies like HTML, CSS, and JavaScript. PWAs are designed to work on any platform that uses a standards-compliant browser, including both desktop and mobile devices.

Installation of PWA

Installing a Progressive Web App (PWA) is crucial as it ensures the website's availability regardless of network connectivity, offering users instant access anytime, anywhere. By allowing installation, PWAs integrate seamlessly into users' devices, fostering increased engagement and retention rates. Additionally, PWAs boast optimized performance, delivering fast loading times and responsive navigation, thereby enhancing user satisfaction. Moreover, the simplified distribution process of PWAs eliminates the need for app store approval, enabling developers to reach a wider audience more efficiently. Overall, installing a PWA enhances accessibility, improves user engagement, streamlines distribution, and contributes to a seamless user experience.

Installation of PWA

Welcome to FlipShop



Product 1
Price: \$100

Buy



Product 2
Price: \$200

Buy



Product 3
Price: \$300

Buy



Product 4
Price: \$400

Buy

SERVICE WORKERS IN PWA

Service Worker is a script that works on browser background without user interaction independently. Also, It resembles a proxy that works on the user side. With this script, you can track network traffic of the page, manage push notifications and develop “offline first” web applications with Cache API.

The screenshot displays a web application titled "Welcome to FlipShop" with a grid of smartphone products. The interface includes a sidebar with navigation options like Manifest, Service workers, Storage, and Background services. The main content area shows two product cards: "Product 1" (Price: \$100) and "Product 2" (Price: \$200), each with a "Buy" button. The right sidebar shows the "Service workers" section, which is active for the URL "http://127.0.0.1:5500/". It displays the source "service-worker.js", the status "#2706 activated and is running", and a table of update cycle activities.

Version	Update Activity	Timeline
#2706	Install	
#2706	Wait	
#2706	Activate	■

Implementation of Events

In Progressive Web Apps (PWAs), fetch, push, and sync events play crucial roles in enhancing functionality and user experience.

The fetch event allows PWAs to intercept network requests, enabling them to cache resources for offline use, reduce load times, and improve performance.

Push events empower PWAs to send notifications to users even when the app is not actively running, facilitating real-time updates and engagement.

Sync events enable background synchronization of data, ensuring that the app remains up-to-date with the latest information, regardless of network availability.

Together, these events contribute to the seamless operation and enhanced functionality of PWAs, enriching the user experience across various devices and network conditions.

PUSH EVENT

Using Application Tab from Chrome Developer Tools for testing push notification.

```
Live reload enabled.
```

```
Service Worker registered with scope: http://127.0.0.1:5500/
```

```
Push notification activated
```

```
③ Service Worker: Fetching
```

```
③ Service Worker: Found in Cache
```

```
Service Worker: Push event received
```

```
▶ PushEvent {isTrusted: true, data: PushMessageData, type: 'push', target: ServiceWorkerGlobalScope, ...}
```

FETCH EVENT

You can track and manage page network traffic with this event. You can check existing cache, manage “cache first” and “network first” requests and return a response that you want.

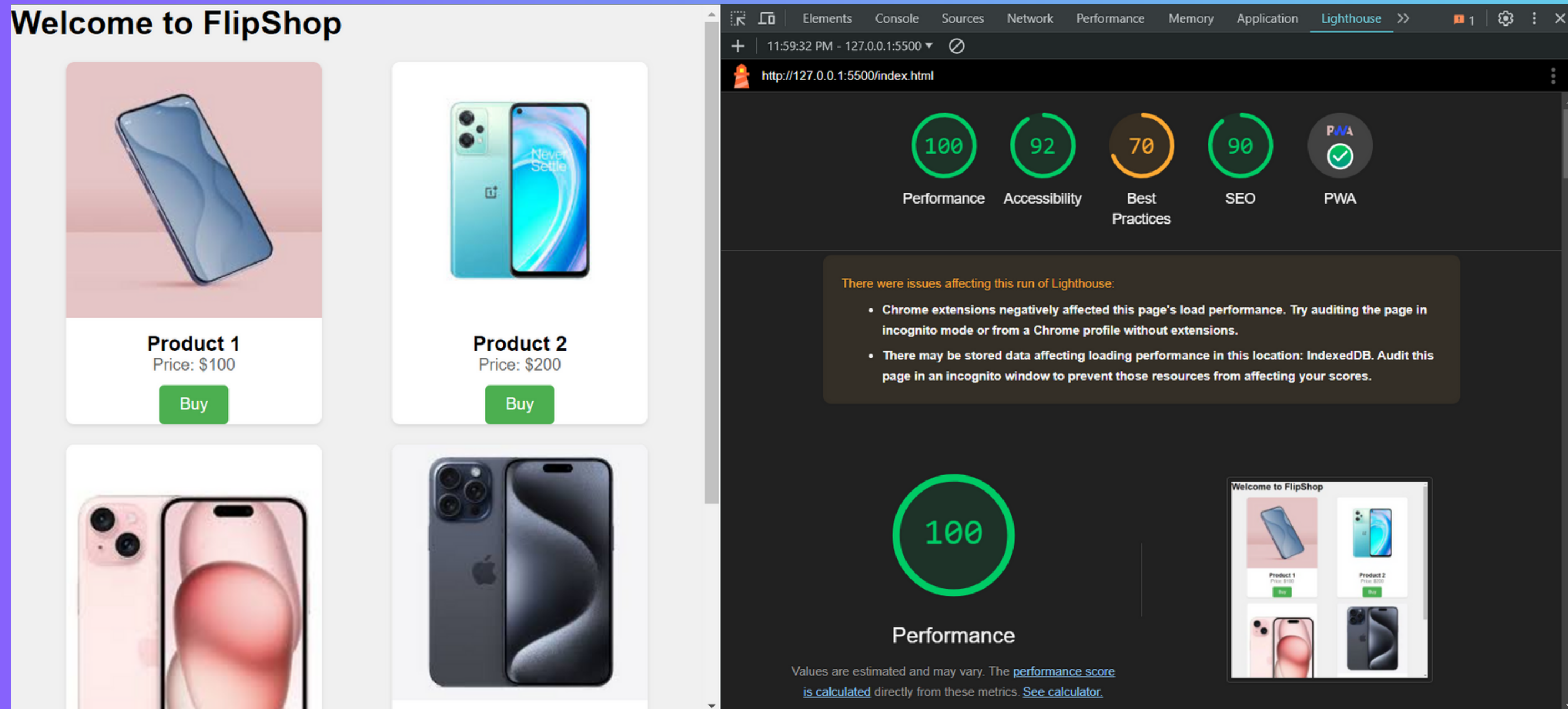
.

```
3 Service Worker: Fetching
3 Service Worker: Found in Cache
```

Performance Analysis with Google Lighthouse

Google Lighthouse is an open-source tool developed by Google that helps developers improve the quality and performance of web pages and web applications. It is commonly used to audit and analyze various aspects of a website, including performance, accessibility, best practices, SEO (Search Engine Optimization), and Progressive Web App (PWA) functionality.

Performance Analysis with Google Lighthouse



Conclusion

In conclusion, our journey in building the E-commerce Progressive Web Application (PWA) has been marked by significant achievements. From the successful installation and service worker registration to the seamless implementation of fetch, push, and sync events, followed by deployment and thorough performance analysis using Google Lighthouse, we have demonstrated the power and versatility of PWAs in modern web development. The importance of PWAs for enhancing user experience cannot be overstated, and as we look to the future, there are ample opportunities for further enhancements and innovations in our E-commerce PWA, ensuring continued growth and success in the ever-evolving digital landscape.

Thank you
for listening!