

CENG 322 – Software Project

DELIVERABLE 4

Team Name: SmartShelf Innovators

Project Title: Smart Produce Management System For Grocery Stores

Group No.: 13

Group Members:

(1) Maharshkumar Raval

Student ID: N01391355

Github ID: MaharshRaval1355

(2) Ayushkumar Patel

Student ID: N01537221

Github ID: AyushPatel7221

(3) Riyankumar Patel

Student ID: N01542237

Github ID: Riyan1102




(4) Winso Tuke

Student ID: N01544313

Github ID: WinsoTuke4313

Project Goals & Final Vision:

The goal of the Smart Produce Management System is to revolutionize grocery store management by using smart sensors to monitor and manage environmental factors such as stock levels, moisture, lighting, and temperature in real-time. The final vision is to automate produce management, reducing waste, ensuring optimal freshness, and providing alerts for restocking and system anomalies. The system will also feature manual controls, visual feedback using NeoPixel LEDs, and secure access to the stockroom using RFID technology. An Android app will allow store staff to interact with and monitor the system remotely, improving operational efficiency and enhancing the customer shopping experience.

Name	Student Id	Github Id	Signature	Effort
Ayushkumar Patel	N01537221	AyushPatel7221		90%
Maharshkumar Raval	N01391355	MaharshRaval1355		85%
Riyankumar Patel	N01542237	Riyan1102		85%
Winso Tuke	N01544313	WinsoTuke4313		50%

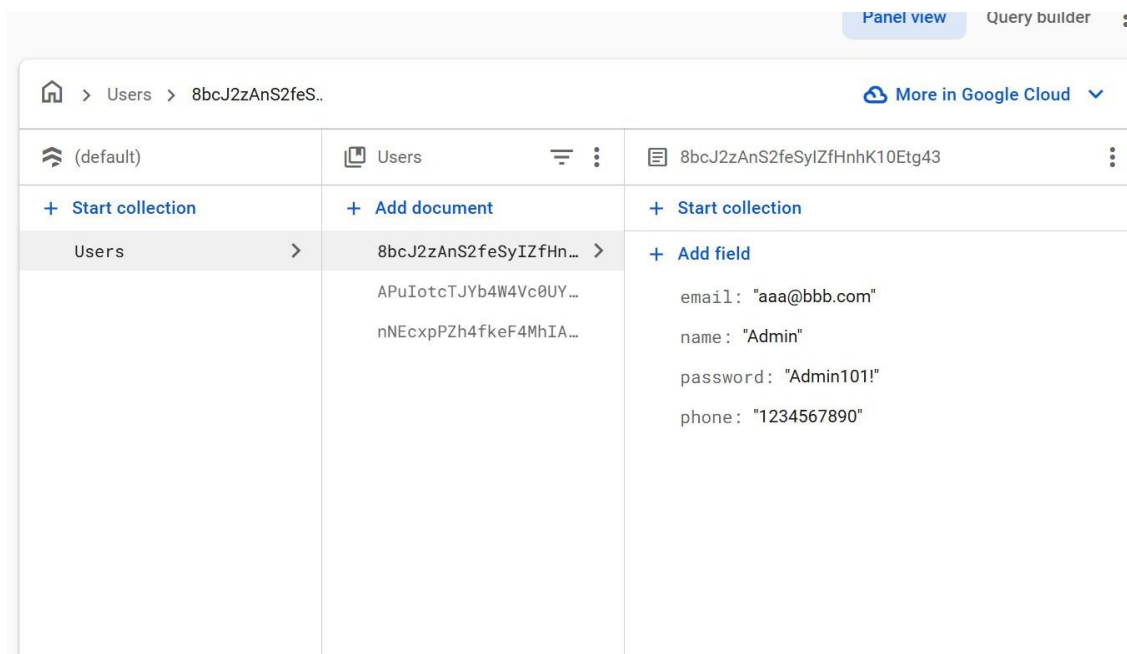
GitHub Repo Link:

<https://github.com/MaharshRaval1355/CENG322SoftwareProject.git>

16.) Credentials that will be used for login:

Email: aaa@bbb.com

Password: Admin101!



Sprint 5 Goals:

1. **Regulator Fragment Development and Setup:** The primary goal for this sprint was to develop and set up the Regulator Fragment that allows users to manually adjust settings such as lighting, temperature, and moisture levels. This involves integrating sensor data with the app interface to ensure real-time monitoring and adjustments.

2. **Offline Functionality Implementation:** Implement features that enable the app to function in offline mode. This includes making certain functionalities available without internet access, ensuring that the app remains functional and user-friendly even when connectivity is limited.
3. **JUnit Testing:** Writing and executing unit tests using JUnit to ensure that all new and existing app components function correctly. This task focuses on validating the reliability and stability of the app through systematic testing.
4. **AlertDialog and Snackbar Integration for Logout:** Setting up an AlertDialog for logout confirmation and integrating a Snackbar to provide users with immediate feedback when they attempt to log out.
5. **App Performance Improvements:** Enhancing the overall performance of the app by optimizing existing code and improving the functionality of various components. This includes reducing app crashes and ensuring smoother transitions and interactions within the app.
6. **Settings and Feedback Screen Enhancements:** Updating and refining the Settings and Feedback Screens to improve user experience. This involves enhancing the user interface and adding new functionalities to provide clearer, more responsive feedback to the users.

These goals were set to improve the app's functionality and user experience, addressing both backend and frontend components to ensure a seamless and efficient operation.

Tasks Addressing Technical Debt:

1. **Refactoring the Database Access Layer:** One of the main tasks was to refactor the database access layer of our application. This involved updating the code to use more efficient queries and improving the structure to ensure better performance and scalability. The task was crucial to eliminate redundant database calls and to enhance the response time of our app.
2. **Optimizing Image Handling and Caching:** Another task focused on the optimization of image handling and caching mechanisms within the app. Previously, inefficient handling of images was causing slow load times and

excessive use of device resources. By implementing more effective caching strategies and optimizing image resolution and size based on the device specifications, we significantly reduced the app's resource consumption and improved the user experience.

3. **Updating Legacy Code to Support New Android Versions:** This task involved updating legacy code that was not fully compatible with newer Android versions. It included replacing deprecated functions and APIs with newer ones supported in recent Android releases. Addressing this technical debt was essential to ensure the app's compatibility and functionality across all devices, preventing potential crashes and compatibility issues.

These tasks address technical debt by focusing on improving the underlying codebase and infrastructure of the app, which contributes to a more robust, efficient, and maintainable application.

Team Member Contributions in Sprint 5:

1. Maharsh:

- **Regulator Fragment Development and Setup:** Led the development and setup of the Regulator Fragment, integrating it with the existing app architecture. This task involved programming the interface and underlying logic to enable real-time adjustments and monitoring of environmental settings such as lighting and temperature.
- **App Performance Improvements:** Focused on enhancing the overall performance of the app by optimizing the existing codebase. Efforts included reducing app crashes and smoothing out UI interactions, which contributed significantly to a better user experience.

2. Ayush:

- **JUnit Testing:** Took charge of writing and executing unit tests for various components of the app using JUnit. This ensured that new functionalities introduced in the sprint, as well as existing features, were robust and reliable.

- **Firestore Database Integration:** Worked on migrating text data and similar data to Firestore for improved management and scalability. Managed the database connectivity which involved setting up and ensuring stable interactions between the app and the Firestore backend.

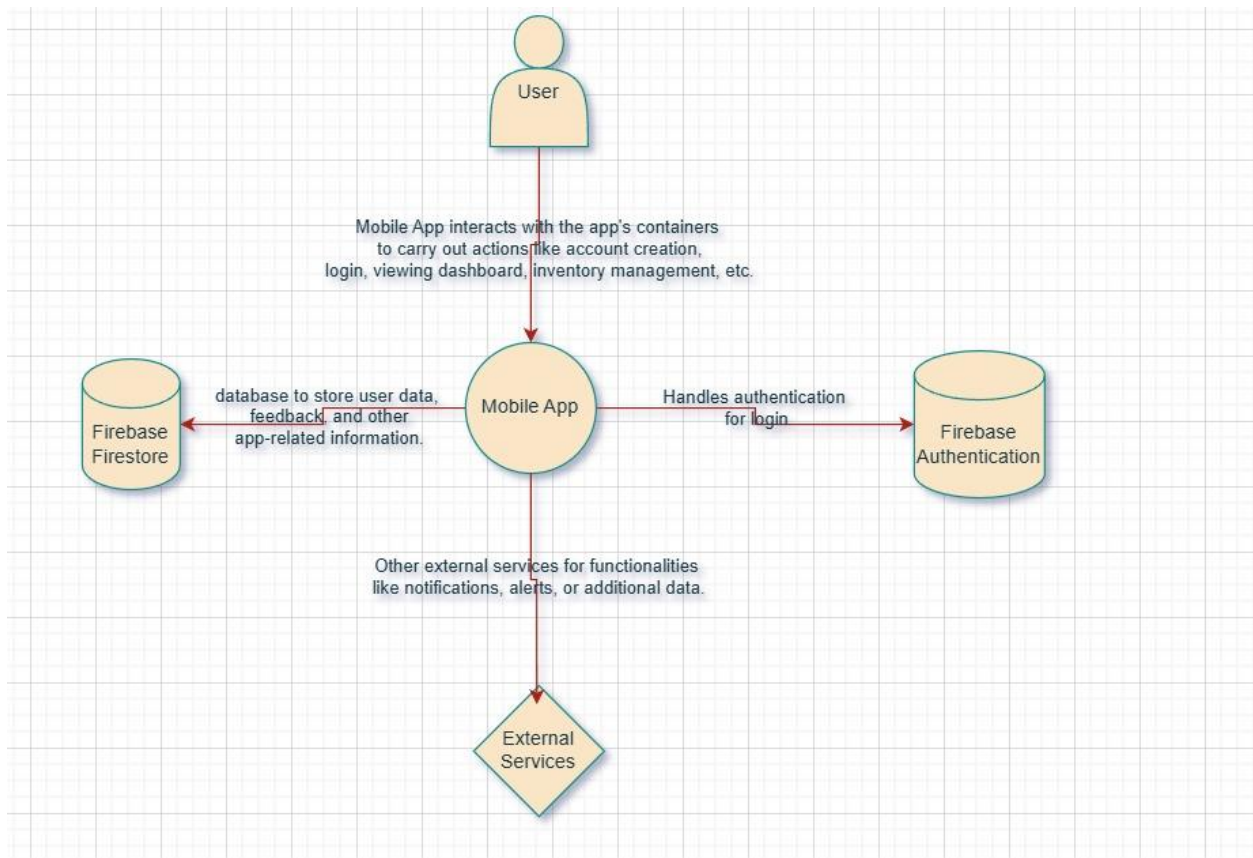
3. Riyan:

- **AlertDialog and Snackbar Integration for Logout:** Implemented an AlertDialog for confirming user logout attempts and integrated a Snackbar to provide immediate feedback upon user actions. This task was crucial for enhancing the interactive aspects of the app's logout functionality.

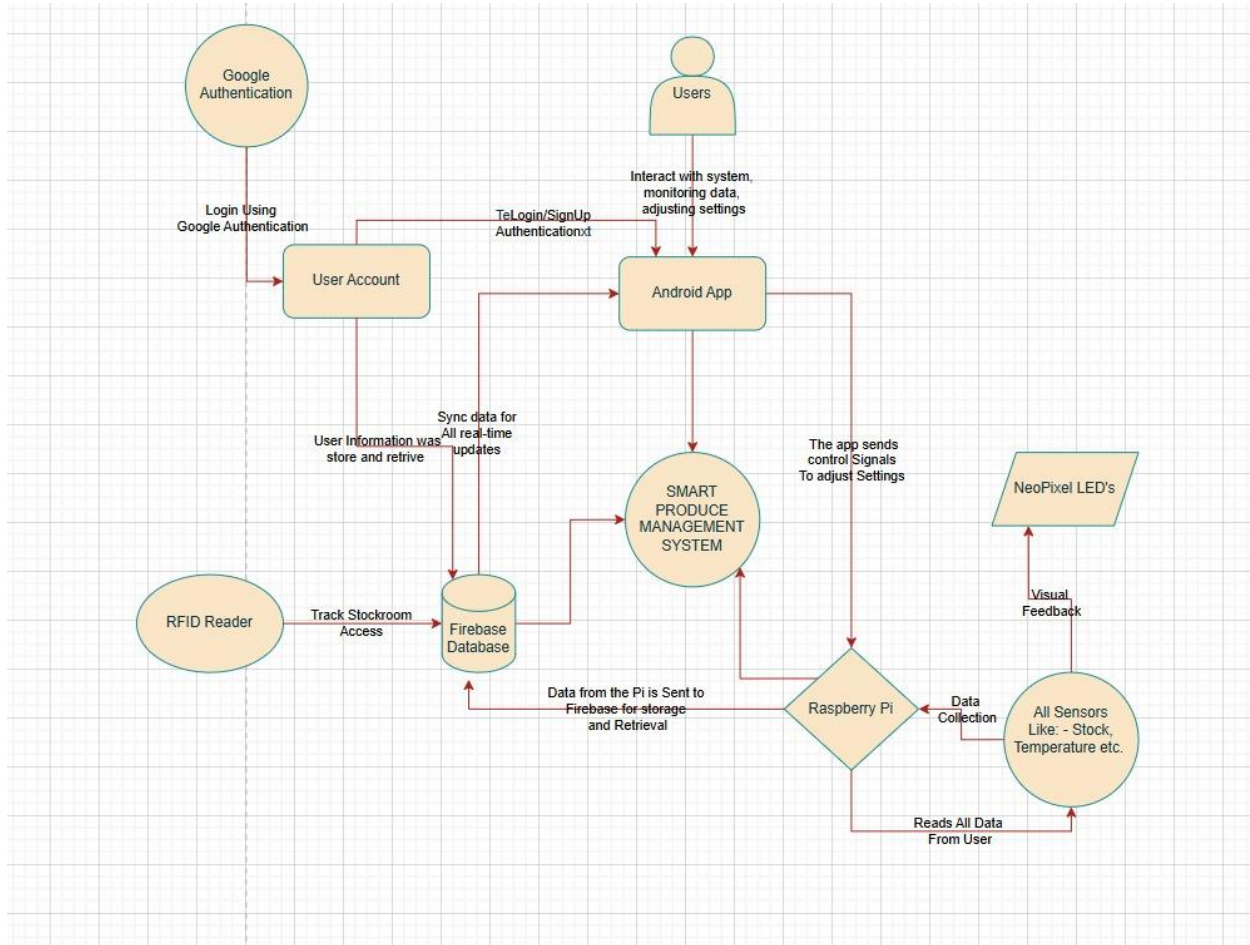
4. Winso:

- **Offline Functionality Implementation:** Assigned the task of adding features that enable the app to function effectively in offline mode. However, this task was not completed and requires further attention in the next sprint.
- **Menu Bar Addition:** Began work on adding a new menu bar to improve navigation within the app. Like the offline functionality, this task remains unfinished and will be carried forward.

Container Diagram:

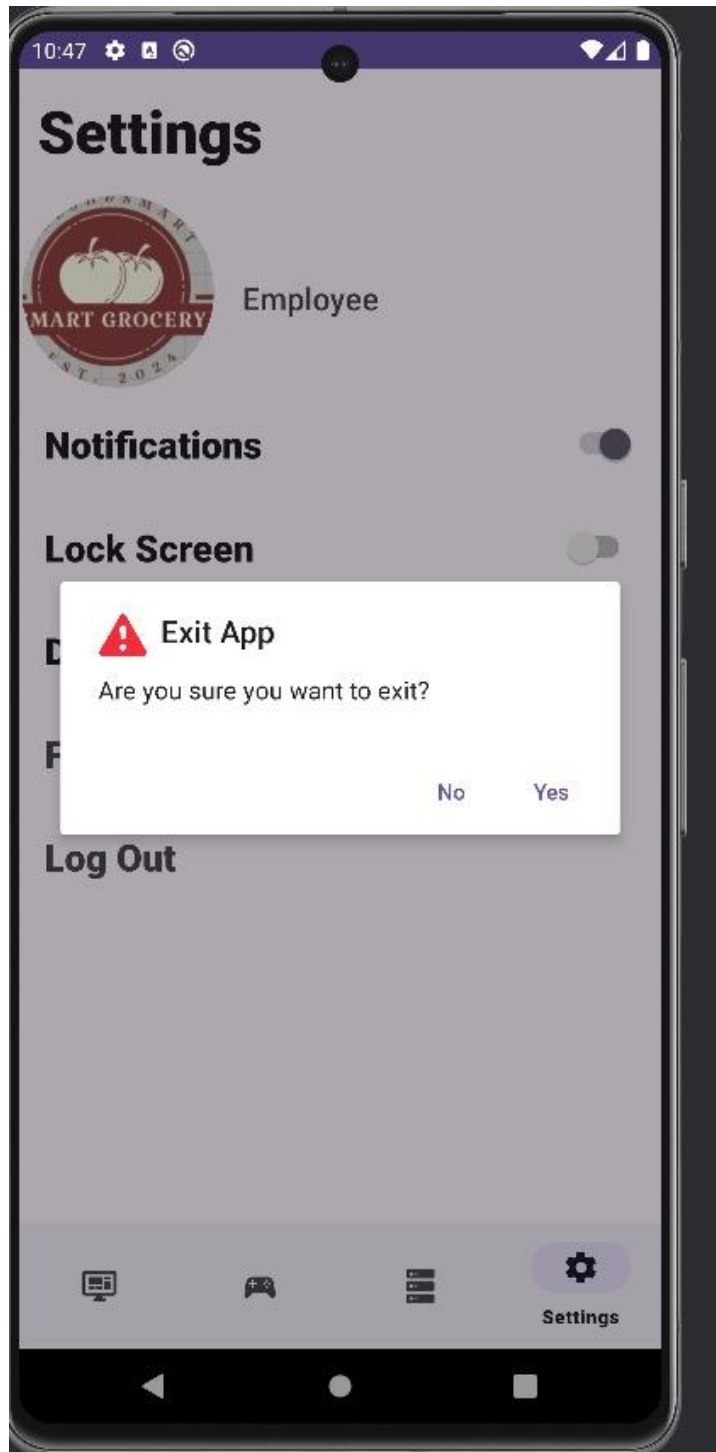


System Context Diagram:

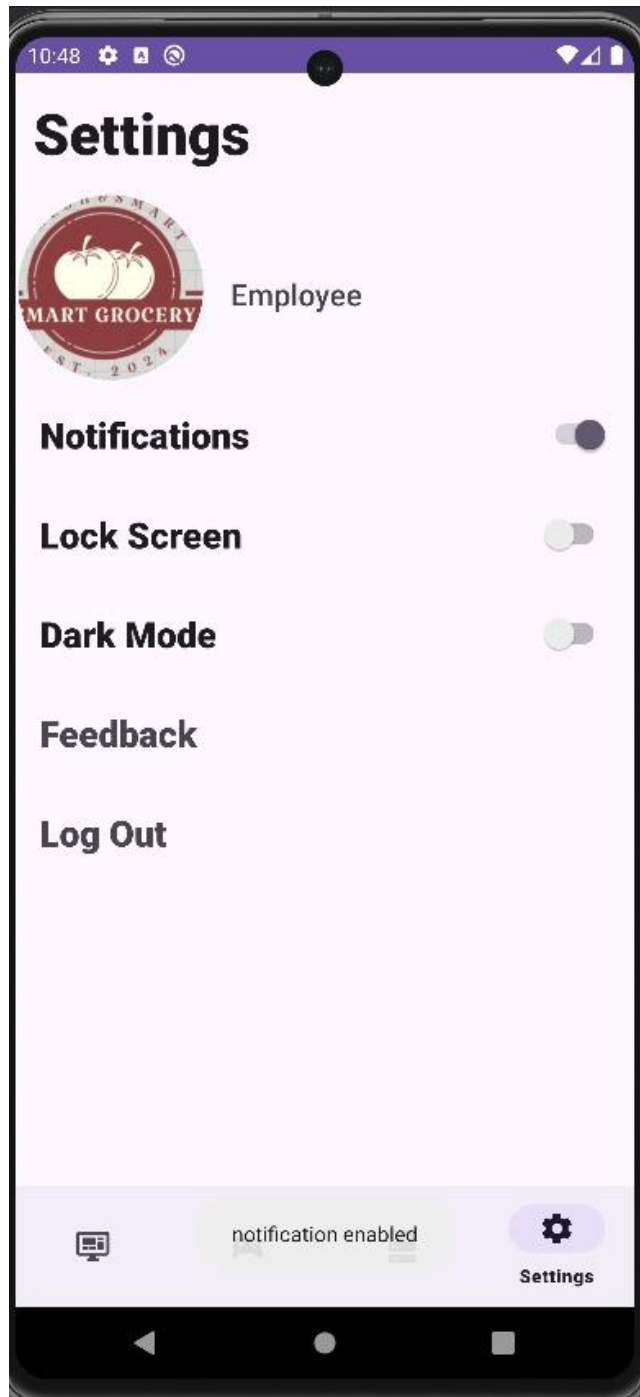


Screenshots showing AlertDialog, Toast, Snackbar, and Notification with runting handling for API 33 and above:

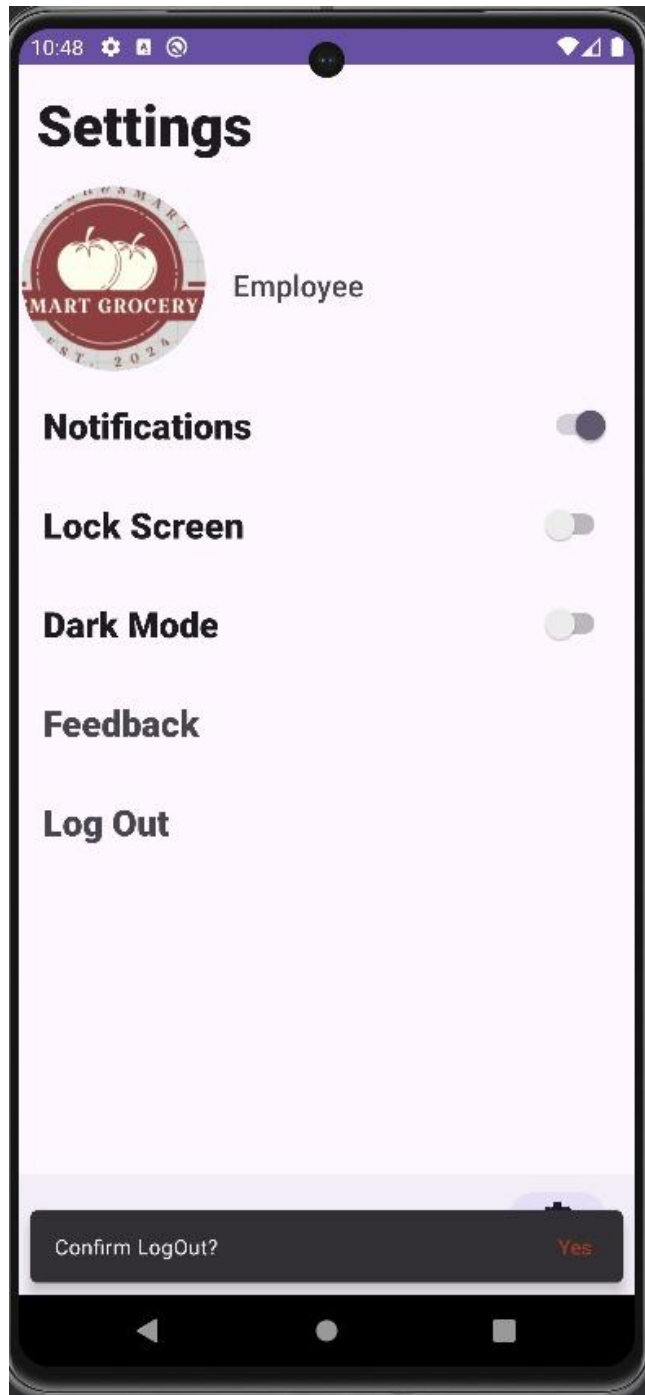
AlertDialog:



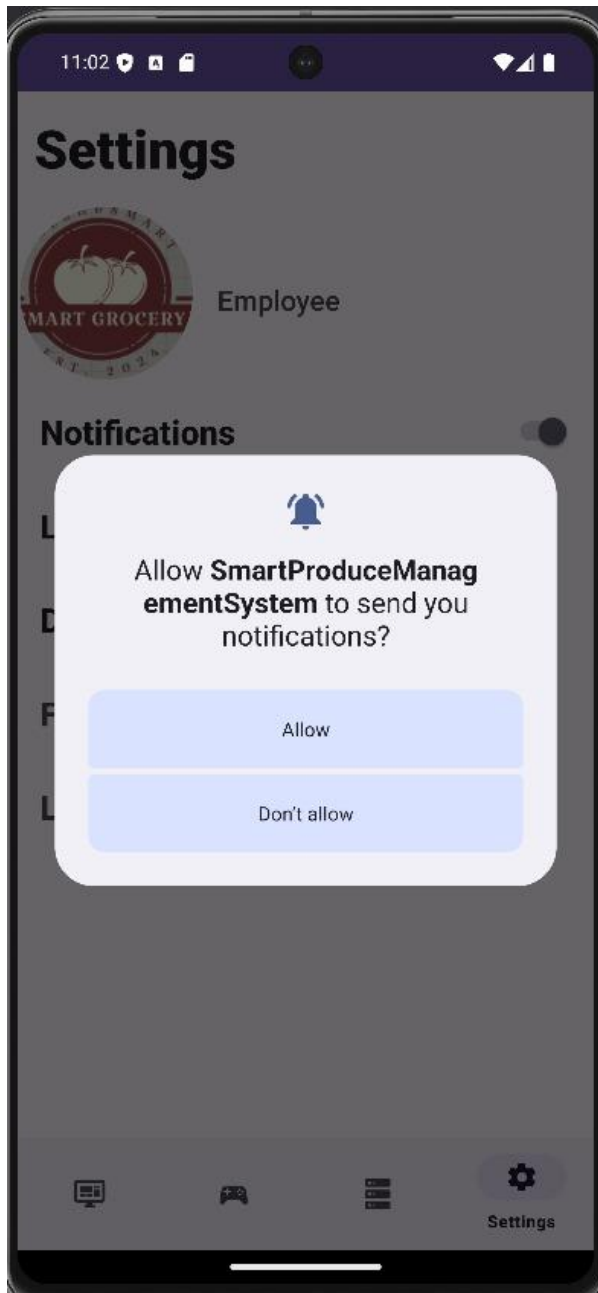
Toast:



Snackbar:



Runtime Notification for API 33:



Scrum Dashboard:

The screenshot shows the Asana Scrum Dashboard for a project named 'Smart Produce Management System'. The interface includes a sidebar with navigation options like 'Create', 'Home', 'My tasks', 'Inbox', 'Insights', 'Reporting', 'Portfolios', 'Goals', 'Projects', and 'Team'. The main area displays a list of tasks for 'Sprint 5'. Each task is represented by a row in a table with columns for Task name, Assignee, Due date, Start Date, End Date, Status, and Size. The tasks are color-coded by status: Ready (blue), Done (green), Working... (orange), and Pending (yellow). The size of each task is indicated by a colored square: Large (red), Medium (yellow), and Small (blue).

Task name	Assignee	Due date	Start Date	End Date	Status	Size
Develop the Regulator Fragment interface	MR MAHARSHK...	Today	Nov 26	Today	Ready	Large
Integrate hardcoded data for environmental sensors	MR MAHARSHK...	Today	Nov 28	Today	Done	Medium
Optimize UI transitions and response times	MR MAHARSHK...	Today	Nov 29	Today	Ready	Medium
Write JUnit tests for new functionalities	Ap Ayush patel	Today	Nov 26	Today	Done	Large
Ensure seamless database connectivity and data retrieval	Ap Ayush patel	Today	Nov 28	Today	Ready	Medium
Migrate textual data to Firebase	Ap Ayush patel	Today	Nov 29	Today	Ready	Medium
Implement logout AlertDialog	n01542237...	Today	Nov 26	Today	Ready	Medium
Integrate Snackbar for logout feedback	n01542237...	Today	Nov 29	Today	Done	Medium
Design offline mode functionality for critical app features	n01544313...	Today	Nov 29	Today	Working...	Medium
Design and develop a new menu bar for enhanced app navigation	n01544313...	Today	Nov 29	Today	Working...	Large

Post-Mortem Project Review Meeting

A. Project Performance Review:

- **Cost:** We managed to stay within our budget, which was a relief since we didn't have to seek additional funds.
- **Schedule:** Our timeline got a bit tight toward the end. Several team members had other coursework, which delayed some of the project tasks beyond the original deadlines.
- **Quality:** We met all the functional requirements, though the final product could have used more polishing. Due to the tight schedule, we didn't get to refine some features as much as we would have liked.

B. Time Management:

- Our time management was a mix of good intentions and last-minute rushes. Balancing this project with other courses proved challenging, and as a result, a significant portion of the work was pushed to the end of the sprint.

C. Quality Issues and Compromises:

- Given the rush near the project deadline, there were minor quality issues. We completed all planned features, but the lack of time meant that some functionalities lacked depth in testing and detail, which could have enhanced the app's overall polish.

D. Lessons Learned and Improvement Areas:

- Early Start: We've learned that starting tasks earlier within the sprint cycle can alleviate end-of-sprint pressure.
- Consistent Progress: We need to ensure that progress is made consistently throughout the project, not just at critical deadlines.
- Enhanced Focus on Detail: Allocating more time to detail-oriented tasks like testing and user interface polishing would likely improve the final product's quality.
- Better Time Allocation: Adopting more effective time management strategies will help us manage our course load and project tasks more efficiently.

E. Meeting Attendance:

- All team members were present at the meeting, which helped us thoroughly discuss our achievements and the areas needing improvement.

Project Review Meeting:

Addressing technical debt in our project:

In our project, we proactively managed technical debt by focusing on several key practices. We regularly refactored our code to make it cleaner and more efficient, and updated all dependencies to maintain security and compatibility. We also prioritized thorough documentation and implemented automated tests to catch bugs early. Additionally, peer code reviews were essential for maintaining high code quality. These strategies helped us minimize technical debt, ensuring our app remains robust and maintainable.

Refactored Code For Runtime Permission:

```
// Notification toggle listener
notificationSwitch.setOnCheckedChangeListener((buttonView, isChecked) -> {
    if (isChecked) {
        if (android.os.Build.VERSION.SDK_INT >= android.os.Build.VERSION_CODES.TIRAMISU) {
            // For API 33+ request runtime permission
            if (ContextCompat.checkSelfPermission(requireContext(), Manifest.permission.POST_NOTIFICATIONS)
                != PackageManager.PERMISSION_GRANTED) {
                ActivityCompat.requestPermissions(requireActivity(),
                    new String[]{Manifest.permission.POST_NOTIFICATIONS},
                    REQUEST_NOTIFICATION_PERMISSION);
            } else {
                enableNotifications();
            }
        } else {
            enableNotifications();
        }
    } else {
        disableNotifications();
    }
});
```

Why Refactoring Was Needed:

- Compliance with New API Requirements: Ensures the app remains functional and respects user permissions on newer Android versions.
- Avoiding Crashes: Prevents potential crashes due to unauthorized notification attempts, ensuring a smooth user experience.

Refactored Code For Snackbar:

```
logOutTV = view.findViewById(R.id.logOut_textview);
logOutTV.setOnClickListener(v -> {
    Snackbar snackbar = Snackbar.make(view, "Confirm LogOut?", Snackbar.LENGTH_INDEFINITE)
        .setAction("Yes", confirmView -> {
            FirebaseAuth.getInstance().signOut();
            editor.clear();
            editor.apply();
            intent = new Intent(requireActivity(), SignUp.class);
            intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK);
            startActivity(intent);
            Toast.makeText(requireContext(), "Logged Out Successfully", Toast.LENGTH_SHORT).show();
        })
        .setActionTextColor(getResources().getColor(R.color.snackbar_action));
    snackbar.show();
});
```

Why Refactoring Was Needed:

- Enhanced User Interaction: Provides a confirmation step before logging out, preventing accidental logouts.
- Improved UX Design: The Snackbar offers an interactive and friendly way for users to confirm or cancel their action, enhancing the overall user experience.

How DevOps Would Have Helped Our Project:

Incorporating DevOps practices into our project could have significantly enhanced both the development process and the overall quality of our app. By adopting a DevOps approach, we could have streamlined collaboration between team members, ensuring more consistent updates and integrations. Continuous Integration (CI) and Continuous Deployment (CD) tools would have allowed us to automatically test and deploy code changes, reducing the time to detect and fix bugs. Additionally, by leveraging Infrastructure as Code (IaC), we could have automatically managed and provisioned our development, testing, and production environments, ensuring consistency across all stages. This would have led to a more reliable and efficient development lifecycle, ultimately improving our project's delivery speed and stability.

Coding Standards Used in Our Project:

For our project, we adhered to several key coding standards to ensure our code was clean, maintainable, and efficient. We followed Java's standard naming conventions, which include using camelCase for methods and variables, and PascalCase for class names, making our code easier to read and understand. We also implemented consistent indentation and spacing, which enhanced code readability and helped prevent errors during collaborative edits. Comments and documentation were integral parts of our standards; we made sure every method and class was well-documented to explain the purpose and functionality, aiding any new team members who might join the project or when revisiting old code. These standards were crucial for maintaining a high level of code quality and fostering an environment of clear communication and effective teamwork.

Vulnerability:

```
public void onSuccess() {  
    if (rememberMeCheckBox.isChecked()) {  
        loginManager.saveLoginState(email, password);  
    }  
    navigateToMainActivity();  
}
```

One potential security vulnerability in the provided code is storing sensitive information (like passwords) in shared preferences or other unprotected storage when the "Remember Me" feature is enabled. The password is being saved in plain text if the user chooses to be remembered, which poses a security risk."

Security Vulnerability: The password is saved in shared preferences. If an attacker gains access to the app's storage (via physical access or a security exploit), they could easily retrieve the password. We will use Android's EncryptedSharedPreferences or the Android Keystore system to securely store sensitive information like passwords. This ensures that even if an attacker gains access to the app's storage, they cannot easily retrieve the password.

Not storing passwords in plain text: In the future, I will avoid storing passwords directly in the app's local storage altogether. Instead, I'll store a hashed and salted password using secure algorithms like PBKDF2, bcrypt, or scrypt, which will make it impossible to retrieve the original password even if the data is compromised.

Switching to Token-based Authentication: Instead of storing the password, I will switch to OAuth tokens or session tokens. These tokens can be securely stored and used to maintain the user's session without compromising sensitive credentials.

Progress Bar while the form is getting submitted:

The screenshot shows a mobile application interface with a purple header bar. The status bar at the top displays the time 11:49 and various system icons. The main content area has a light purple background and is titled "Feedback". It contains four text input fields with the following values: "Admin", "1234567890", "aaa@bbb.com", and "good app". Below the text fields is a row of five grey stars. At the bottom of the form are two rounded purple buttons labeled "Submit Feedback" and "Back". A large, faint grey loading spinner is centered on the screen, indicating that the form is being submitted. The bottom of the screen features a navigation bar with four icons: a monitor, a game controller, a list, and a settings gear. The "Settings" icon is highlighted with a purple circle and the label "Settings" below it. The Android navigation bar is visible at the very bottom.

11:49

Feedback

Admin

1234567890

aaa@bbb.com

good app

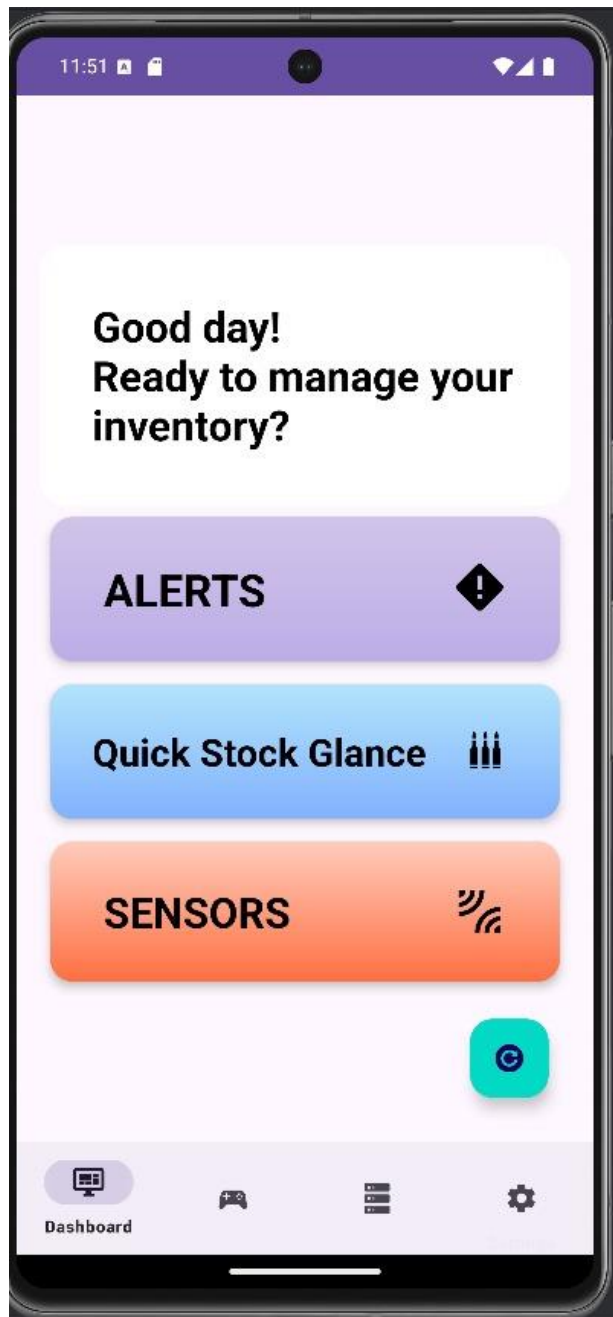
★ ★ ★ ★ ★

Submit Feedback

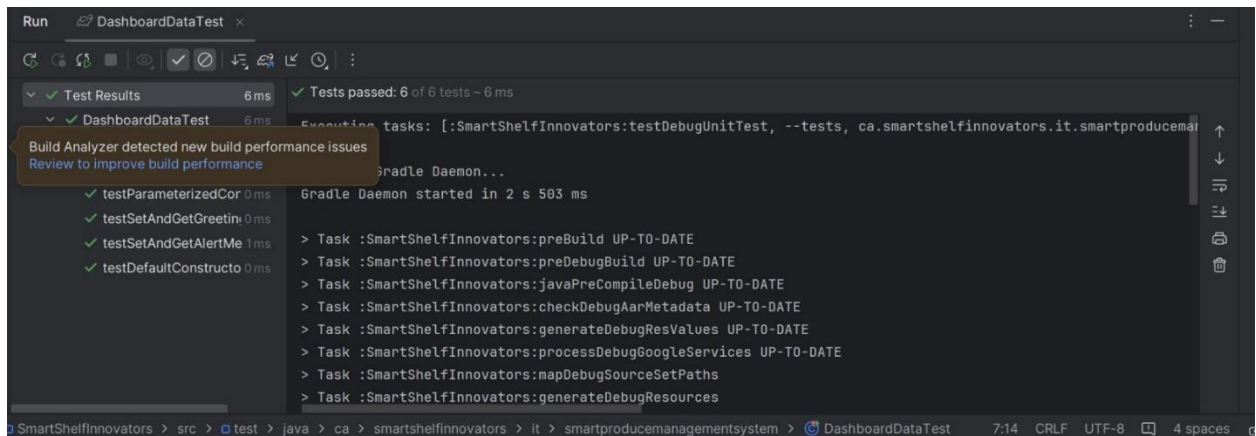
Back

Settings

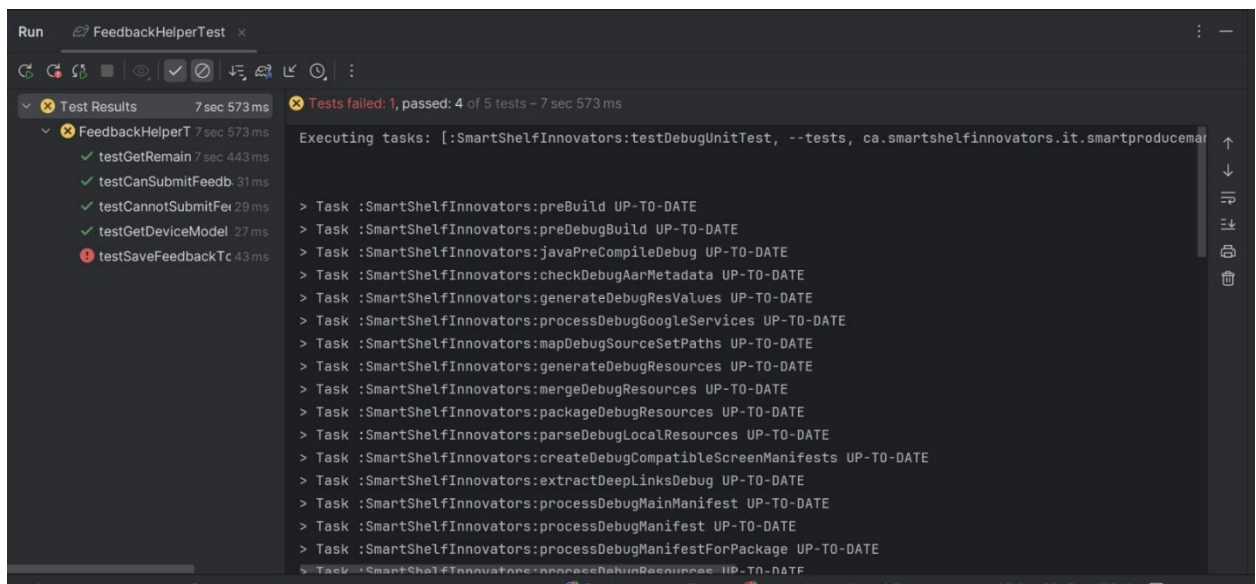
Floting Action Button:

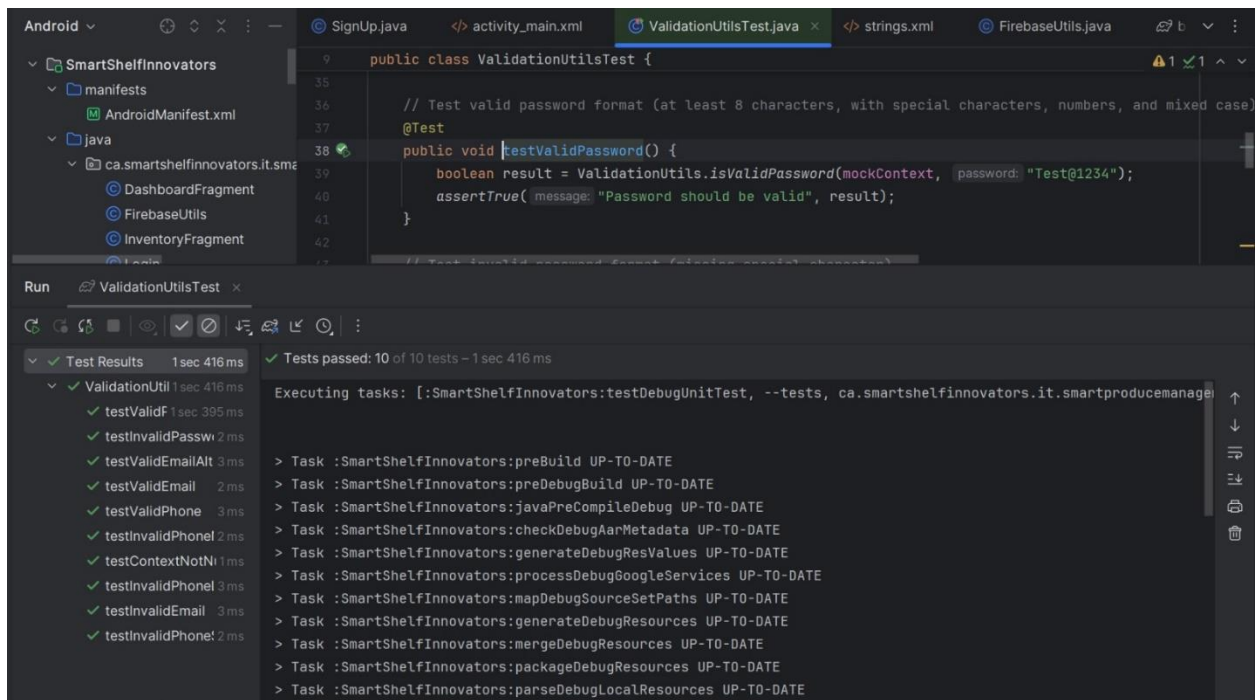


DashBoardDataTest:



Robletric Test:





Entries In Database:

Home > Feedback > 6A3WzGnqlsryS. More in Google Cloud		
(default)	Feedback	6A3WzGnqlsrySZ8WP8zW
+ Start collection	+ Add document	+ Start collection
Feedback >	6A3WzGnqlsrySZ8WP8zW >	+ Add field
Users dashboardData	NBAaBD0ApVdbLSKQHL57 PeRGi35gAEpMjhXPg19F UxfSkoc11WvcageqPE49 gm5ycxKRDIDxn1NSQJsF o1zsRsax5qfCqHVtDN5	comment: "Very Good and Very easy to use the app" deviceModel: "Google Android SDK built for x86" email: "n01542237@gmail.com" name: "Riyan Patel" phone: "1234567890" rating: 5