

Deliverable 4

Team Name: FurGuardian

Project Name: Pet Wellness App

Team Group: 8

Student Names and ID's: Justin Chipman – N01598472

Imran Zafurallah - N01585098

Zane Aransevia - N01351168

Tevadi Brookes - N01582563

Name	Student Id	Github Id	Signature	Effort
Justin Chipman	N01598472	Chipman8472	Justin C	100%
Imran Zafurallah	N01585098	ImranZafurallah5098	Imran Z	100%
Zane Aransevia	N01351168	Zayine917	Zane A	100%
Tevadi Brookes	N01582563	TevadiBrookes2563	Tevadi B	100%

GitHub Link: <https://github.com/Chipman8472/FurGuardian.git>

Issue FurGuardian solves







FurGuardian addresses the market's lack of comprehensive pet wellness solutions by integrating health, activity, and interaction features into a single platform. It solves the problem of fragmented pet care by offering a centralized way for pet owners to manage health records, track fitness, and monitor wellness insights, ensuring pets receive optimal care. Additionally, FurGuardian provides tailored health tips and real-time tracking (planned) to address potential health issues proactively.

Comparison of two apps with FurGuardian

Pros and Cons:

- **PetCube:**
 - **Pros:**
 - Great for remote pet interaction (camera, treats, audio).
 - Offers fun and engaging features like live streaming and treat dispensing.
 - Well-suited for entertainment and keeping pets engaged.
 - **Cons:**
 - Lacks detailed health records or medical history.
 - No fitness tracking or activity monitoring beyond interaction.
 - Does not offer pet wellness insights or health tips.
- **Whistle:**
 - **Pros:**
 - Excellent for **GPS tracking** and **activity monitoring**.
 - Provides insights into pet activity, sleep patterns, and overall fitness.
 - It is ideal for tracking physical health and pet safety.
 - **Cons:**
 - No interaction features (camera, treats).
 - Does not track detailed health records or offer health tips.
 - Limited to fitness and safety, it lacks comprehensive wellness features.
- **FurGaurdian:**
 - **Pros:**
 - **Comprehensive health management:** Tracks fitness, activity, health records, and wellness tips.
 - Ability to manage pet profiles with medical history, vet visits, and more.
 - **Planned features** like device control (e.g., treat dispenser, food schedule).
 - **Real-time tracking** and tailored health tips.
 - We are focused on holistic pet care, blending health monitoring and interaction.
 - **Cons:**
 - No **GPS tracking** (unless integrated with external devices).
 - **Device control** is still a planned feature (not yet implemented).
 - Relies on user input for health metrics (e.g., weight) rather than real-time automated data.

Feature/Aspect	PetCube	Whistle	FurGuardian
Main Focus	Pet interaction (camera, treats, audio)	GPS tracking and activity monitoring	Comprehensive pet health and wellness management
Pet Interaction	✓ Two-way audio, treat dispensing, camera streaming	✗ No remote interaction features	✓ Planned features for device control (e.g., food dispenser)
Activity Tracking	✓ Monitors pet movement, interaction	✓ Tracks physical activity (steps, playtime, sleep)	✓ Activity tracking (steps, weight, hydration, etc.)
Health Records	✗ No detailed health records or vet visit tracking	✗ Limited health records	✓ Detailed health records, vet visits, medical history
GPS Tracking	✗ No GPS tracking	✓ Real-time GPS tracking for pet location	✗ No built-in GPS tracking
Health Tips	✗ No health tips	✗ No health tips	✓ Tailored health tips, including weight management
Pet Profiles	✗ Limited to interaction features	✓ Basic pet profile (focus on activity)	✓ Detailed pet profiles with breed, medical history, etc.
Device Control	✓ Treat dispensing, camera interaction	✗ No device control	✓ Planned control of pet devices (e.g., food dispenser)
User Experience	✓ Interactive and engaging	✓ Focused on fitness tracking	✓ Comprehensive wellness, health records, and activity logs
Real-Time Health Tracking	✗ No real-time health metrics	✗ Limited real-time health tracking	✓ Planned real-time health tracking (heart rate, hydration)
Pet Wellness Insights	✗ No wellness insights	✗ Limited wellness insights	✓ Health and fitness insights based on tracked data

Battery & Alerts Alerts for device status Alerts for battery life and activity changes Alerts for health updates and pet status**Cost** Higher device cost (camera and treat dispenser) Device cost for GPS tracking collar No cost

FurGuardian stands out as the best option due to its comprehensive health and wellness features, including tracking fitness, activity and maintaining detailed health records like vet visits and medical history. It offers tailored health tips, wellness insights, and planned real-time health tracking, providing a holistic approach unmatched by competitors. Unlike PetCube and Whistle, FurGuardian supports detailed pet profiles, allowing owners to manage breed, medical history, and other vital information. Combining health monitoring, planned device control, and interaction features bridges the gap between wellness and engagement, making it ideal for overall pet care. Its app-based solution is also more cost-effective, avoiding the expensive devices required by competitors. With plans for smart device integration, FurGuardian continues to evolve as a forward-thinking and comprehensive pet wellness platform.

What we did this sprint

Justin:

I worked on implementing the hashing algorithm to encrypt the passwords before being stored into the database. This way the account passwords are not displayed in plain text in our database. I also worked on fixing our PetEdFragments performance issues. It was loading very slow and even crashing our app sometimes. I found that the cause of our issue was heavy sequential processing needed in the creation of the fragment between the geocoder and the webview. My solution was to implement multithreading so that the components can be initialized in parallel. In terms of the crashing I found many uncaught exceptions especially with `requireContext()` calls. So I simply handled them accordingly to fix the crashing. As well I updated the feedback functionality to incorporate more input handling and graying out the submit button for 24 hours preventing one user from flooding our feedback potentially diluting our data findings later on. Another aspect of the app I worked on this sprint was unit testing. I implemented 10 tests for `LoginActivity` to ensure `sharedprefs`, proper intent transfer and other crucial components functioned correctly.

Tevadi:

In this sprint I improved on the Records fragment completely changing the layout, implemented a pet medical record management system that was integrated with Firebase Realtime Database. The Records Fragment was changed to dynamically display medical records and a button for retrieval of that data. A RecordsAdapter was created to bind the Firebase data to a custom card layout called card medical records(xml), displaying specific details about that pet's record with functionality to delete records. Troubleshooting steps were taken to resolve issues with unresolved IDs and missing resources ensuring proper linkage between layouts and code. The final implementation successfully integrates Firebase for data storage and retrieval, dynamically displays records and provides functionality to manage and add new medical records. I also worked on the creation of the C4 Model "System Context Diagram" and the "Container Diagram".

Imran:

This Sprint I worked on the PetEd fragment to optimize it and redid the fragment layout and introduce the spinner class with webview for a full integration. The PetEd fragment will now open the web pages and youtube videos within the app and only take you out if you need to make a schedule for vaccinations or check up within the calendar app, all other activities are done within the app for convenience and simplicity.

+

Zayne:






I collaborated with Tevadi in redesigning the records fragment. We ran into some issues with not being able to store an actual file without paying for firebase storage. So for the meantime we implemented just text and dates for the records.

Sprint Goals





















Increase account security
Write unit tests for LoginActivity
Modify simulated data to mimic more realistic behavior
Relieve technical debt in PetEdFragment
Improve RecordFragment Design

Sprint Dashboard

▼ Sprint 4

<input type="checkbox"/>	Task	Owner	Status ⓘ	Due date ⓘ	Priority	T-Shirt Size	Timeline ⓘ	Last updated	+
<input type="checkbox"/>	> Food and Water Monitoring 1  	 IZ	Not Started	 Dec 10	Low	small	-	 58 minutes ago	
<input type="checkbox"/>	+ Add task			Dec 10			-		
















▼ Completed
















<input type="checkbox"/>	Task	Owner	Status ⓘ	Due date ⓘ	Priority	T-Shirt Size	Timeline ⓘ	Last updated	+
<input type="checkbox"/>	> Update Data simulation for real world 5 		Done	 Nov-19	Medium	medium	 Nov 7 - 19	 16 minutes ago	
<input type="checkbox"/>	> Implement Unit Tests for LoginActivity 5 		Done	 Nov-19	Medium	small	 Nov 7 - 19	 5 minutes ago	
<input type="checkbox"/>	> Update database and Increase Accou... 5 		Done	 Nov-19	High	medium	 Nov 7 - 19	 2 minutes ago	
<input type="checkbox"/>	> Update and Relieve technical debt in P... 5 		Done	 Nov-19	Critical ⚠	large	 Nov 7 - 19	 Just now	
<input type="checkbox"/>	+ Add task								

Nov 19

Nov 7 - 19

▼ Completed

<input type="checkbox"/>	Task	Owner	Status ⓘ	Due date ⓘ	Priority	T-Shirt Size	Timeline ⓘ	Last updated	+
<input type="checkbox"/>	▼ Update Data simulation for real world 5 		Done	 Nov-19	Medium	medium	 Nov 7 - 19	 Just now	
<input type="checkbox"/>	Subitem	Owner	Status	Date	+				
<input type="checkbox"/>	Make heart rate more realistic 		Done	Nov 18					
<input type="checkbox"/>	make steps more realistic 		Done	Nov 18					
<input type="checkbox"/>	make sleep more realistic 		Done	Nov 18					
<input type="checkbox"/>	make distance more realistic 		Done	Nov 18					
<input type="checkbox"/>	make weight more realistic 		Done	Nov 18					
<input type="checkbox"/>	+ Add subitem								

<input type="checkbox"/>	▼ Implement Unit Tests for LoginActivity 5 		Done	 Nov-19	Medium	small	 Nov 7 - 19	 Just now	
<input type="checkbox"/>	Subitem	Owner	Status	Date	+				
<input type="checkbox"/>	Create tests for settings sharedprefs 		Done						
<input type="checkbox"/>	create tests for successful intent tran... 		Done						
<input type="checkbox"/>	create tests for inputs 		Done						
<input type="checkbox"/>	create tests for login sharedprefs 		Done						
<input type="checkbox"/>	create tests for database 		Done						
<input type="checkbox"/>	+ Add subitem								

<input type="checkbox"/>	▼ Update database and Increase Accou...	5			Done	✓	Nov-19	High	medium	✓ Nov 7 - 19		Just now
<input type="checkbox"/>	Subitem			Owner	Status		Date		+			
<input type="checkbox"/>	Implement hashing algorithm for pass...			Done								
<input type="checkbox"/>	update database to store encrypted p...			Done								
<input type="checkbox"/>	update database to store salt			Done								
<input type="checkbox"/>	Add pet information to database			Done								
<input type="checkbox"/>	implement store and retrieving of pet ...			Done								
<input type="checkbox"/>	+ Add subitem											

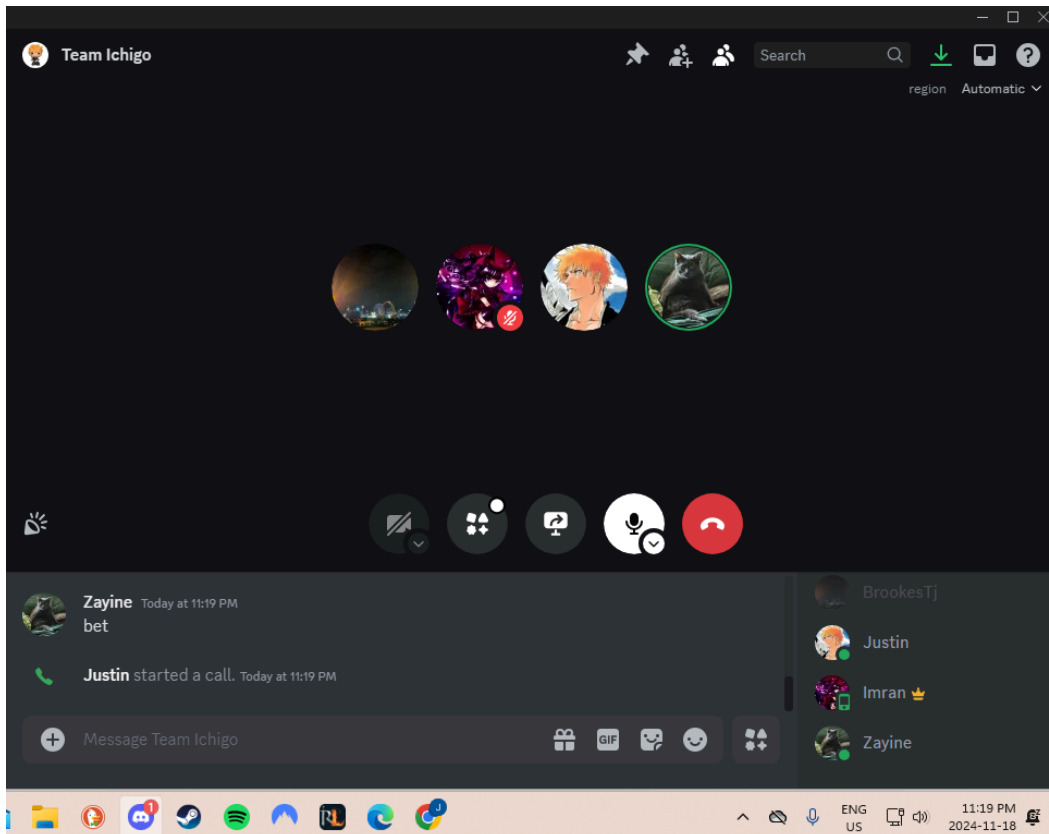
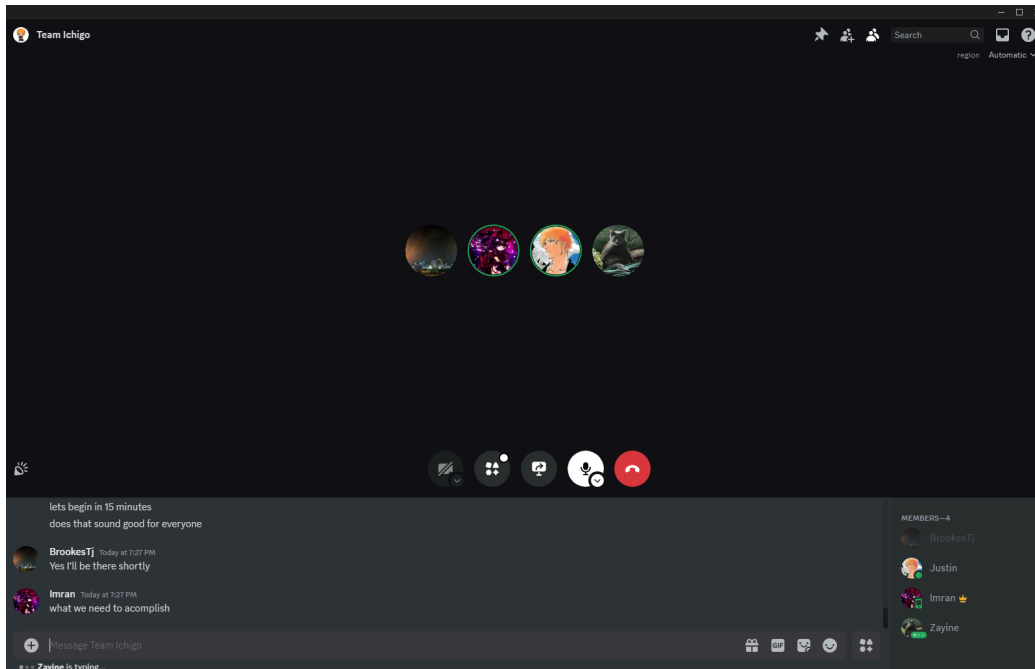
<input type="checkbox"/>	▼ Update and Relieve technical debt in P...	5			Done	✓	Nov-19	Critical ⚠	large	✓ Nov 7 - 19		Just now
<input type="checkbox"/>	Subitem			Owner	Status		Date		+			
<input type="checkbox"/>	Fix slow loading speed			Done								
<input type="checkbox"/>	Keep user in app for websites			Done								
<input type="checkbox"/>	Cover all unhandled exceptions			Done								
<input type="checkbox"/>	Update design			Done								
<input type="checkbox"/>	Implement google calendar functiona...			Done								
<input type="checkbox"/>	+ Add subitem											

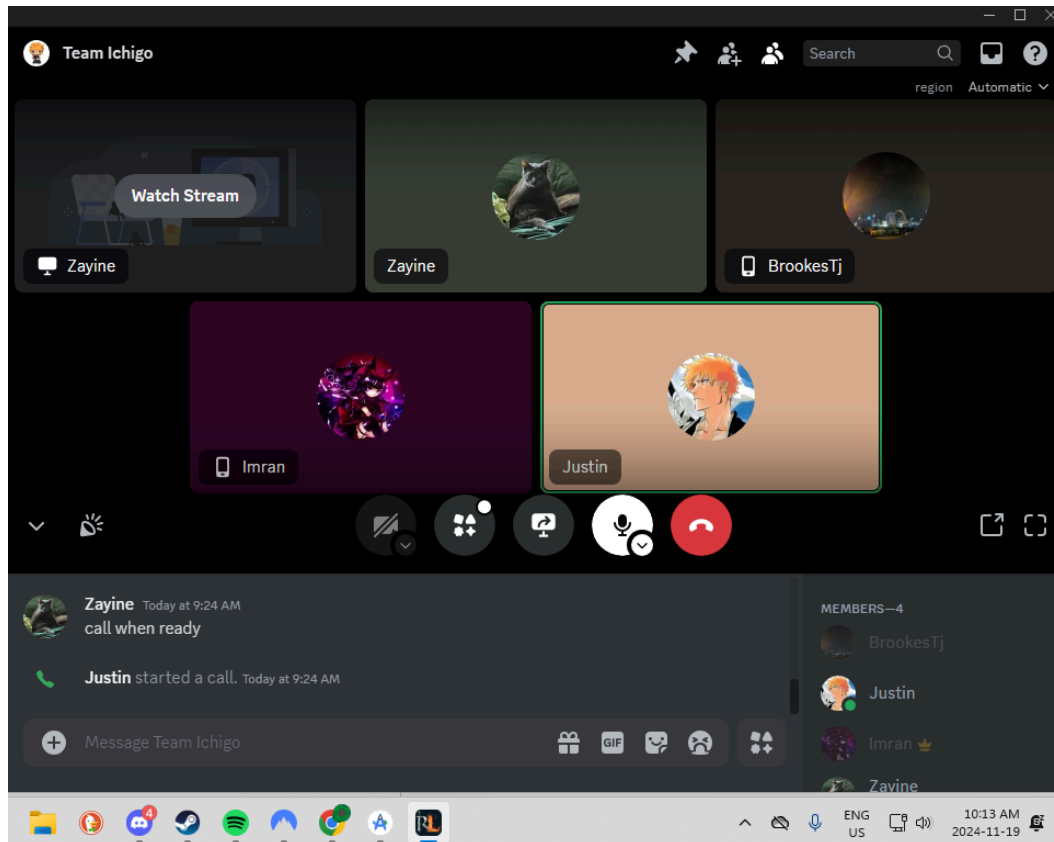
Team Discussions

Our discussions were mostly voice call based so here is three screenshots of different voice call dates of our group.

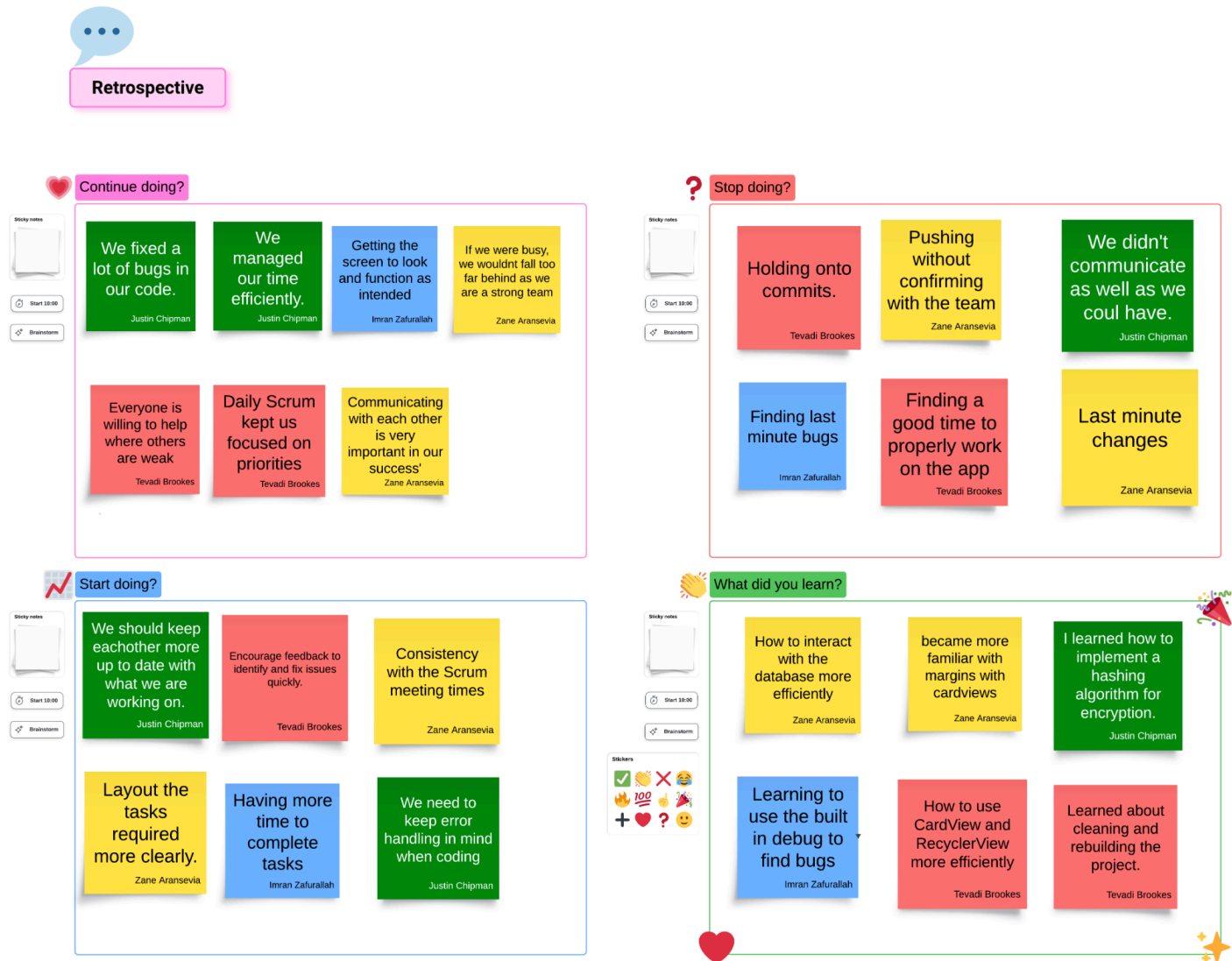
topics discussed:

Improving petEd fragments performance
 updating record fragment design
 implementing encryption for our passwords
 increasing our error handling

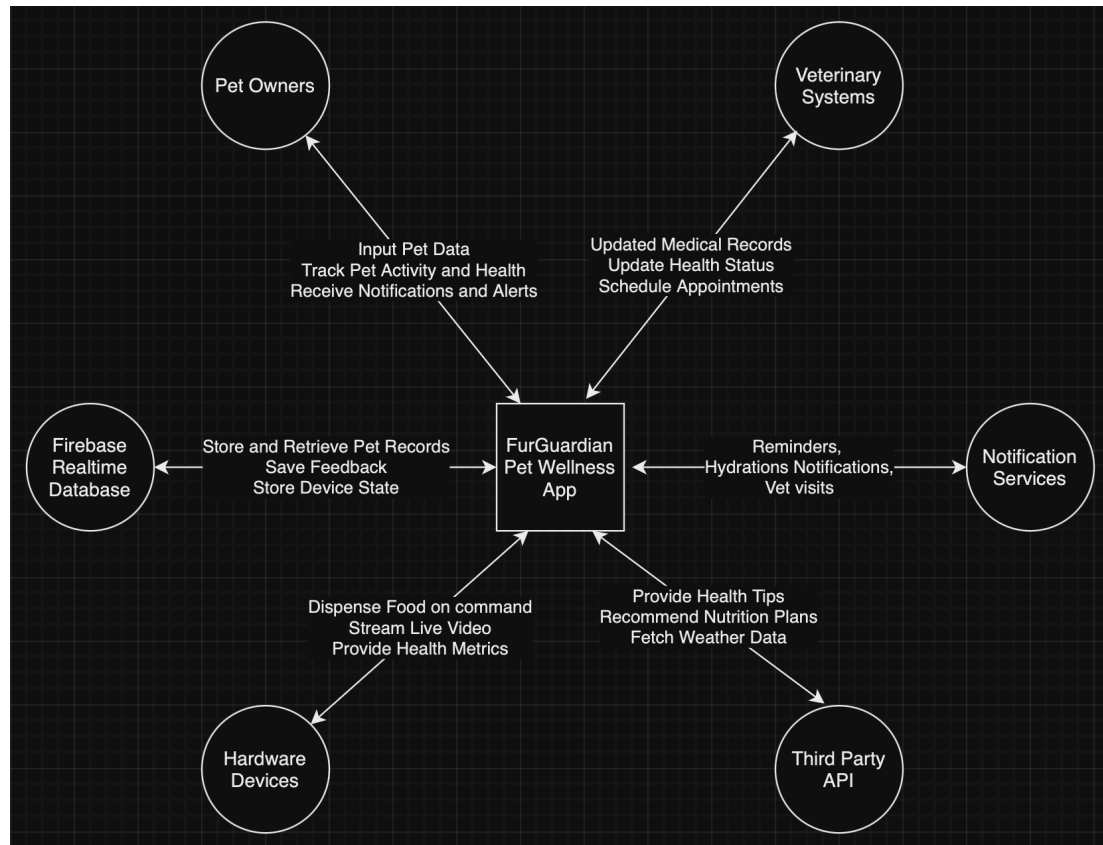




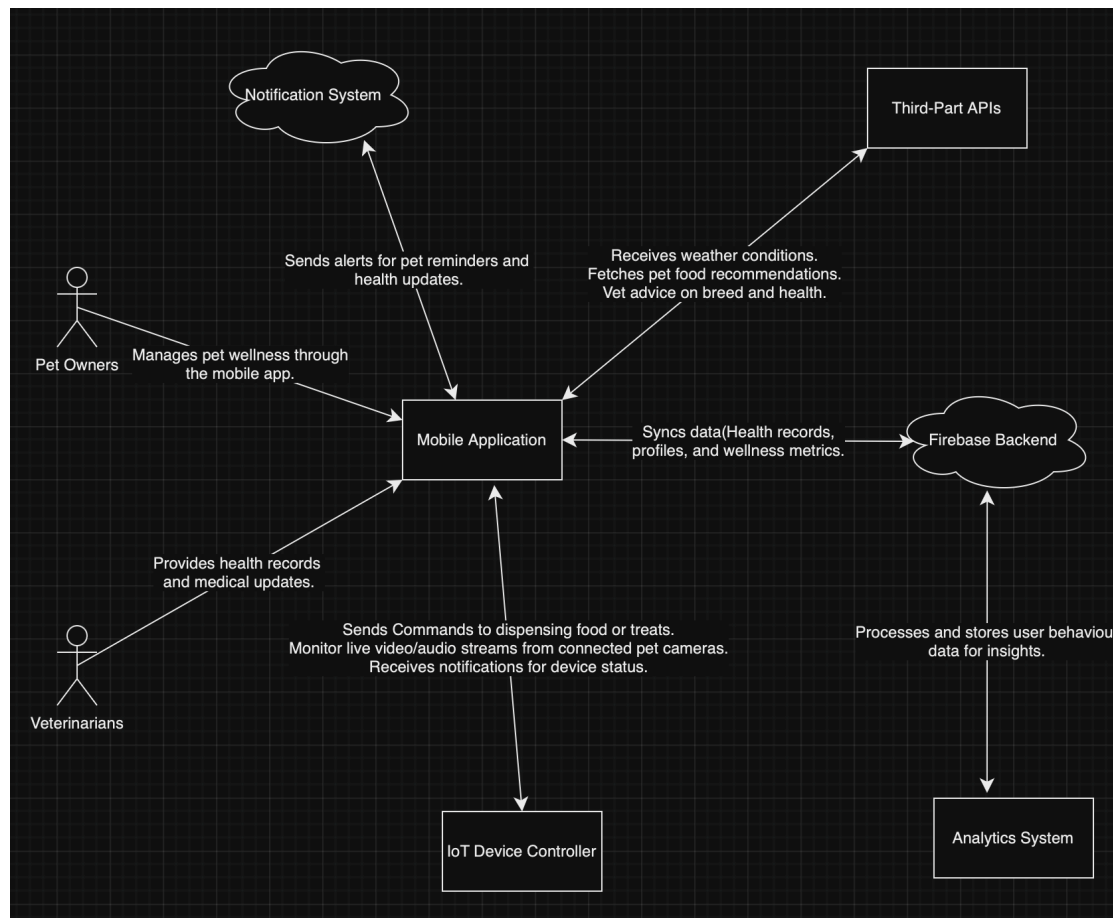
Sprint Retrospective



C4 Model System Context Diagram



C4 Model Container Diagram



Design Patterns

Singleton

```
public UserModel() {  
    FirebaseDatabase database = FirebaseDatabase.getInstance();  
    this.usersRef = database.getReference("users");  
}
```

Here we demonstrate the use of a singleton object. Since we don't want to create multiple instances of our database connection we use `getInstance()` so that if there is a connection already opened we access that one instead of making a new one.

Builder

```
// Show success dialog  
new AlertDialog.Builder(requireContext())  
    .setTitle("Thank You!")
```

```
.setMessage("Your feedback has been submitted successfully!")
.setPositiveButton("OK", (dialog, which) -> resetFields()) .setCancelable(false)
.show();
}, 2000);
```

Here we demonstrate the use of the builder design pattern. Which is used to build the alertDialog piece by piece and allowing us to choose what features we want to add. This is the benefit of the builder pattern: it allows for the same building process to produce different products.

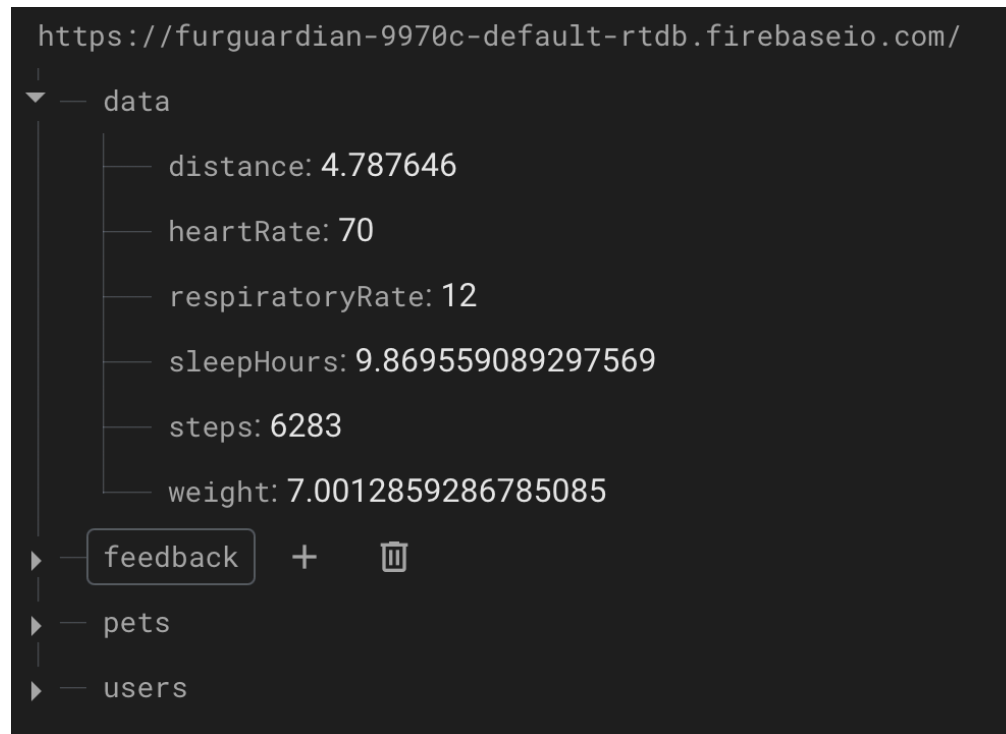
Test case screenshot

The screenshot shows the 'Run' tab in Android Studio with the 'LoginActivityTest' test suite selected. The status bar indicates '10 passed' and '10 tests, 1 m 6 s 321 ms'. The test results are displayed in a table with columns for 'Tests', 'Duration', and 'Medium_Phone_API_34'.

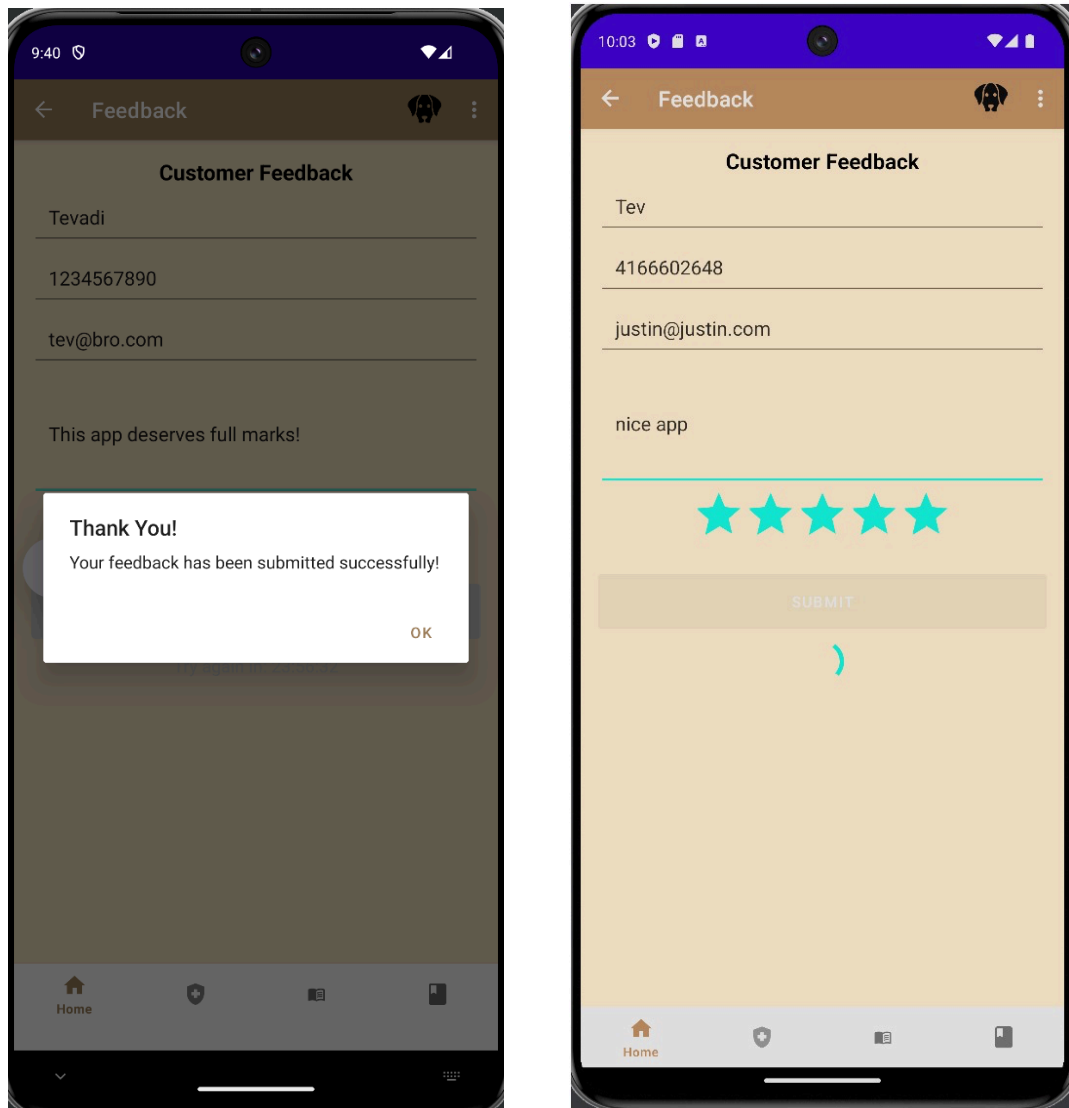
Tests	Duration	Medium_Phone_API_34
✓ Test Results	50 s	10/10
✓ LoginActivityTest	50 s	10/10
✓ testOrientationLockPreference	42 ms	✓
✓ testActivityTransitionOnSuccessfulLogin	46 s	✓
✓ testDefaultLoggedInValue	9 ms	✓
✓ testRememberMeCheckBoxDefaultState	1 s	✓
✓ testPasswordPreference	10 ms	✓
✓ testEmailPreference	36 ms	✓
✓ testEmptyLoginFields	1 s	✓
✓ testSharedPreferencesInitialization	8 ms	✓
✓ testDarkModePreference	8 ms	✓
✓ testRememberMePreference	11 ms	✓

The bottom of the screenshot shows the project path: Guardian > FurGuardian > src > androidTest > java > ca > furguardian > it > petwellness > LoginActivityTest >

Data being stored in Database



Screenshot showing the AlertDialog & Screenshot showing the progress bar



How we satisfied the feedback requirement is by storing in sharedpreferences if they submitted a feedback or not and then changed the submit button to gray and started a countdown timer for 24 hours. When the timer is up it will update the sharedpreferences.

Main features implemented

Pet education: This fragment is a one stop shop for information an owner might need for their pet. With location services to help tailor the links towards the user's local area.

Health: This fragment actively communicates with the stored sensor data from the database and displays it. It also gives recommendations based on the read data. (i.e. go for a walk, get some more sleep, etc.)