Software Project Deliverable 3

Buckle Up Inc.

Smart Aqua

Team Number

Group 6

Student's names and ID's

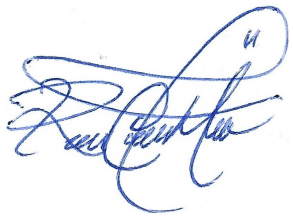
Alvaro Rodrigo Chavez Moya	N01455107
Denis Shwaloff	N01422583
Nicholas Dibiase	N01367109
Paolo Adrian Quezon	N01424883

Table of Contents

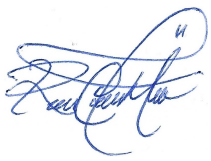
Project Description

Smart Aqua is an easy-to-use Android app that provides real-time water quality monitoring and notifications for effective water treatment system management. The project entails requirement gathering, user-friendly UI design, effective database construction, and seamless sensor integration. Continuous improvements and comprehensive testing are carried out in response to beneficial user input to guarantee that all objectives and requirements are satisfied properly.

Team Signature List

Alvaro Rodrigo Chavez Moya	Denis Shwaloff	Nicholas Dibiase	Paolo Adrian Quezon
			

Members Info and Participation

Name	ID	Signature	Effort
Denis Shwaloff	N01422583		100
Alvaro Rodrigo Chavez Moya	N01455107		100
Nicholas Dibiase	N01367109		100
Paolo Adrian Quezon	N01424883		100

GitHub Repository Link

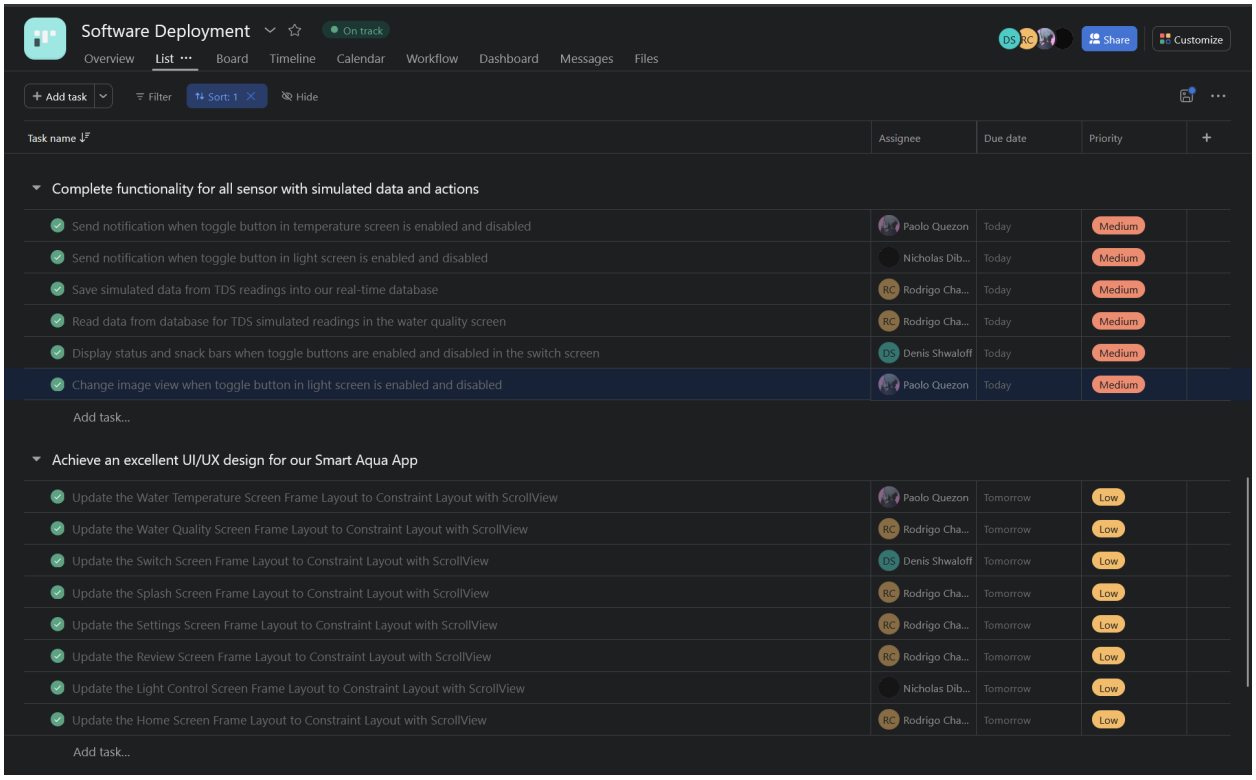
https://github.com/DenisShwaloff2583/SmartAqua.git

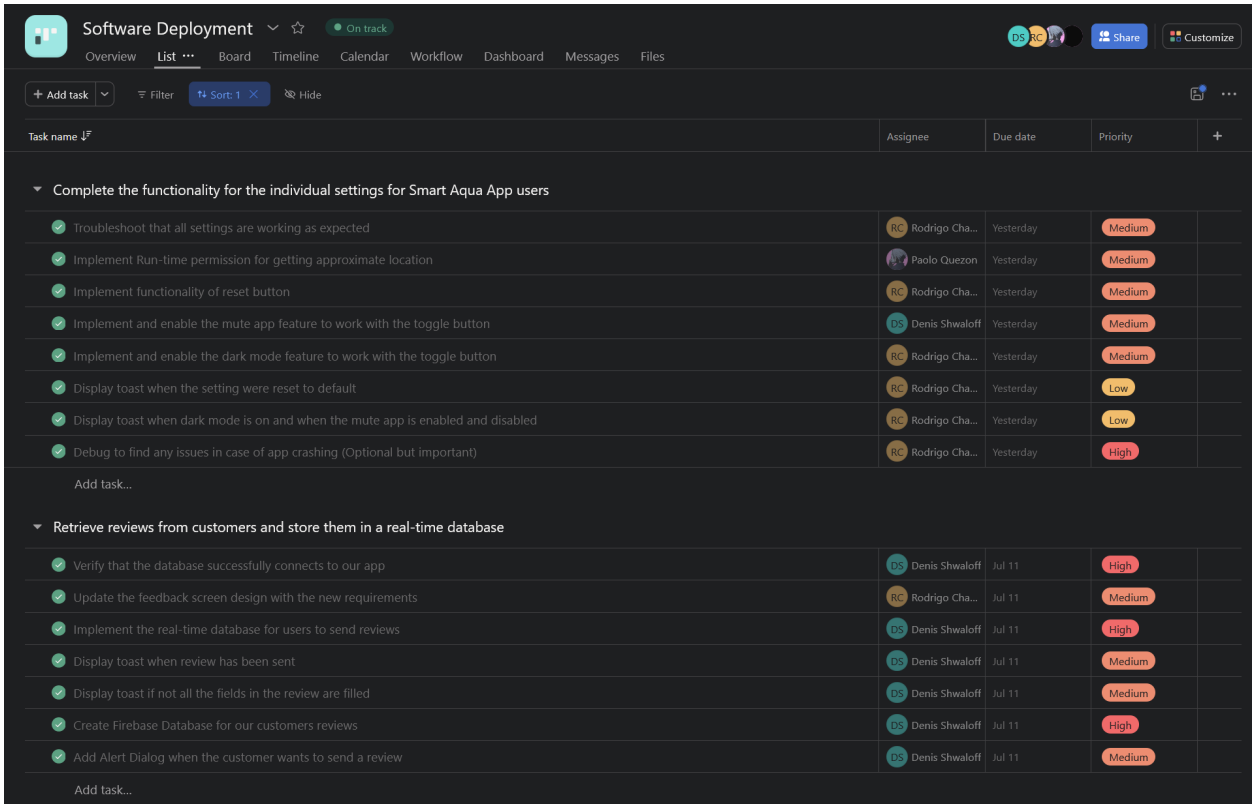
Sprint Goals

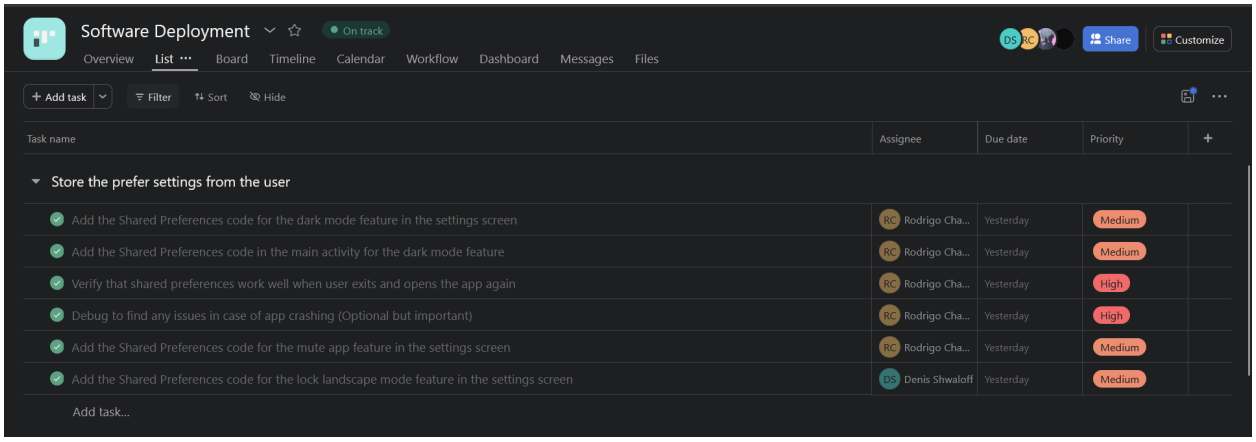
As our team transitions from deliverable 2 to 3 our sprint goals for the current sprint are as follows:

- Complete the implementation of all group member's sensors.
- Complete the feedback/review page implementation with validation.
- Complete the implementation of the settings screen.
- Implement runtime permission with functionality and use of AsyncTask.
- Connect android application with database, which includes storing sensor readings and client reviews.
- Store user preference in settings screen with shared pref, to save user functionality when the user exits the app or switches between screens.

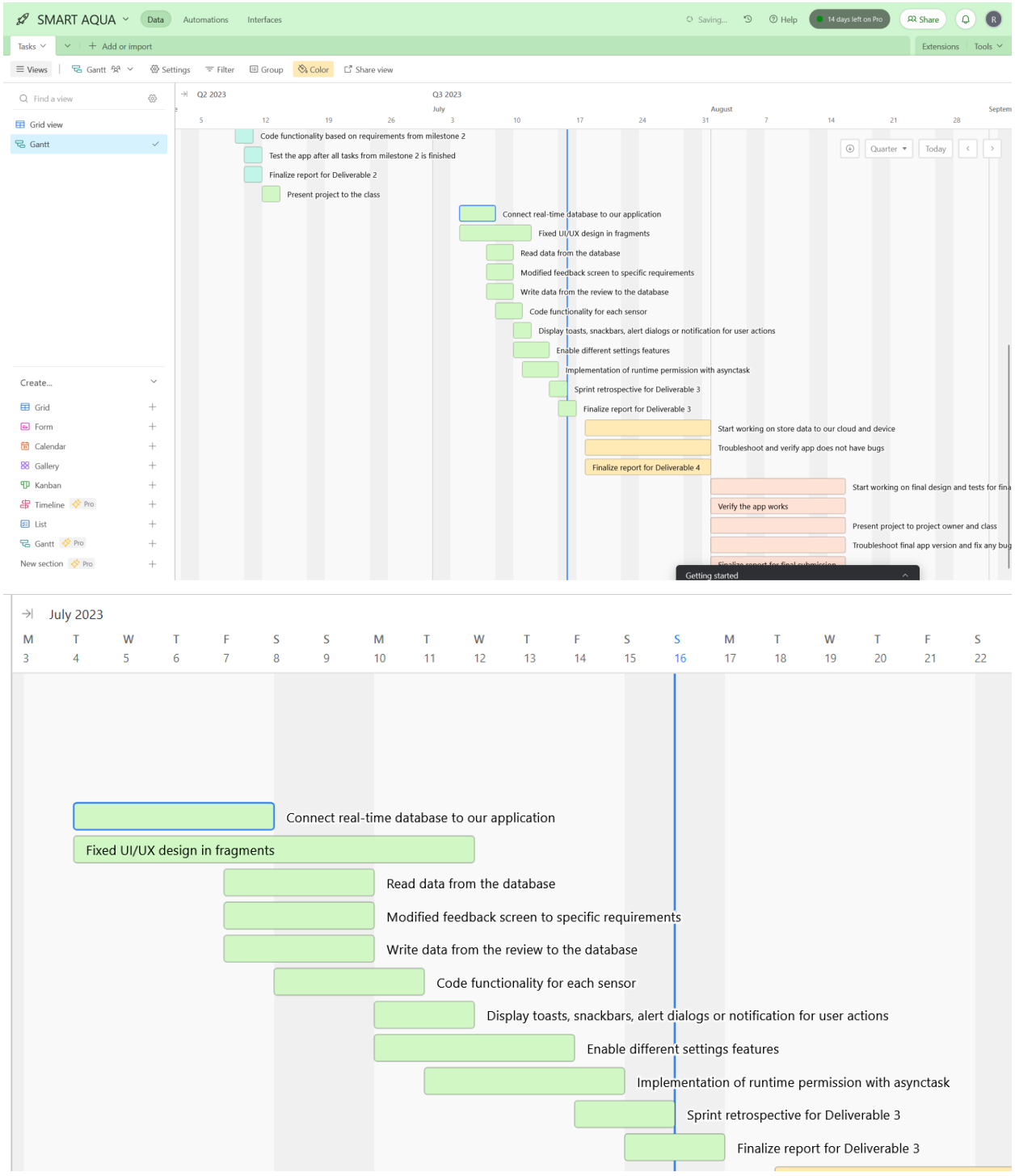
Project Stories




Software Deployment
On track
Overview
List
Board
Timeline
Calendar
Workflow
Dashboard
Messages
Files
Share
Customize
Add task
Filter
Sort: 1
Hide
Task name
Assignee
Due date
Priority
Complete the functionality for the individual settings for Smart Aqua App users
Troubleshoot that all settings are working as expected
Rodrigo Cha...
Yesterday
Medium
Implement Run-time permission for getting approximate location
Paolo Quezon
Yesterday
Medium
Implement functionality of reset button
Rodrigo Cha...
Yesterday
Medium
Implement and enable the mute app feature to work with the toggle button
Denis Shwaloff
Yesterday
Medium
Implement and enable the dark mode feature to work with the toggle button
Rodrigo Cha...
Yesterday
Medium
Display toast when the setting were reset to default
Rodrigo Cha...
Yesterday
Low
Display toast when dark mode is on and when the mute app is enabled and disabled
Rodrigo Cha...
Yesterday
Low
Debug to find any issues in case of app crashing (Optional but important)
Rodrigo Cha...
Yesterday
High
Add task...
Retrieve reviews from customers and store them in a real-time database
Verify that the database successfully connects to our app
Denis Shwaloff
Jul 11
High
Update the feedback screen design with the new requirements
Rodrigo Cha...
Jul 11
Medium
Implement the real-time database for users to send reviews
Denis Shwaloff
Jul 11
High
Display toast when review has been sent
Denis Shwaloff
Jul 11
Medium
Display toast if not all the fields in the review are filled
Denis Shwaloff
Jul 11
Medium
Create Firebase Database for our customers reviews
Denis Shwaloff
Jul 11
High
Add Alert Dialog when the customer wants to send a review
Denis Shwaloff
Jul 11
Medium
Add task...


Software Deployment
On track
Overview
List
Board
Timeline
Calendar
Workflow
Dashboard
Messages
Files
Share
Customize
Add task
Filter
Sort
Hide
Task name
Assignee
Due date
Priority
Store the prefer settings from the user
Add the Shared Preferences code for the dark mode feature in the settings screen
Rodrigo Cha...
Yesterday
Medium
Add the Shared Preferences code in the main activity for the dark mode feature
Rodrigo Cha...
Yesterday
Medium
Verify that shared preferences work well when user exits and opens the app again
Rodrigo Cha...
Yesterday
High
Debug to find any issues in case of app crashing (Optional but important)
Rodrigo Cha...
Yesterday
High
Add the Shared Preferences code for the mute app feature in the settings screen
Rodrigo Cha...
Yesterday
Medium
Add the Shared Preferences code for the lock landscape mode feature in the settings screen
Denis Shwaloff
Yesterday
Medium
Add task...

Gantt Chart


SMART AQUA
Data
Automations
Interfaces
Saving...
Help
14 days left on Pro
Share
Tasks
Add or import
Extensions
Tools
Views
Gantt
Settings
Filter
Group
Color
Share view
Find a view
Grid view
Gantt
Q2 2023
Q3 2023
July
August
Septem
Quarter
Today
Code functionality based on requirements from milestone 2
Test the app after all tasks from milestone 2 is finished
Finalize report for Deliverable 2
Present project to the class
Connect real-time database to our application
Fixed UI/UX design in fragments
Read data from the database
Modified feedback screen to specific requirements
Write data from the review to the database
Code functionality for each sensor
Display toasts, snackbars, alert dialogs or notification for user actions
Enable different settings features
Implementation of runtime permission with asynctask
Sprint retrospective for Deliverable 3
Finalize report for Deliverable 3
Start working on store data to our cloud and device
Troubleshoot and verify app does not have bugs
Finalize report for Deliverable 4
Verify the app works
Present project to project owner and class
Troubleshoot final app version and fix any bug
Create...
Grid
Form
Calendar
Gallery
Kanban
Timeline
List
Gantt
New section
Getting started
July 2023
M T W T F S S M T W T F S S M T W T F S
3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
Connect real-time database to our application
Fixed UI/UX design in fragments
Read data from the database
Modified feedback screen to specific requirements
Write data from the review to the database
Code functionality for each sensor
Display toasts, snackbars, alert dialogs or notification for user actions
Enable different settings features
Implementation of runtime permission with asynctask
Sprint retrospective for Deliverable 3
Finalize report for Deliverable 3

Daily Stand-Ups

DAILY STAND UP

July 5

SPRINT 3

Alvaro Rodrigo Chavez

What did you work on yesterday?

Read through the deliverable document and analyze what implementation I must complete for my sensor

What will you work on today?

Fixed different aspects of the mobile application by following the feedback given in Deliverable 2

Any roadblocks?

Might run into errors in case of coding wrong in the java class for Smart Aqua Water Quality

Nicholas Dibiase

What did you work on yesterday?

Read through the deliverable document and started setting priority on task to see what to start first

What will you work on today?

Fixed light screen design giving feedback after presentation of deliverable 2

Any roadblocks?

Might run into erros with the light-bulb image switch in the following days

Paolo Quezon

What did you work on yesterday?

Briefed over deliverable guidelines and reviewed over feedback from the previous sprint. Jotted down key app functionality and document requirements.

What will you work on today?

Change implementation of land scape mode for the splash and temperature screens according to feedback received from deliverable 2.

Any roadblocks?

Knowing which code should be retained according to feedback and what can be added to improve it.

Denis Shwaloff

What did you work on yesterday?

Went over the professor's comments and corrections for deliverable 2. Applied fixes to the app sections I'm responsible for

What will you work on today?

Update the design for Settings screen and go over the layout files and re-make landscape layouts

Any roadblocks?

Had a problem with implementing user run-time permission. Stumbled across depreciated classes and code, so did not know how to approach.

DAILY STAND UP

July 7

SPRINT 3

Alvaro Rodrigo Chavez

What did you work on yesterday?

After fixing the aspects given by the feedback. I started to work on the functionality of the water quality screen. Also changed minor XML details

What will you work on today?

I will start the design for the feedback screen following the guideline given in Deliverable 3

Any roadblocks?

No blockers at the moment

Nicholas Dibiase

What did you work on yesterday?

FIxed the deign for the light screen planning on starting to do the functionality for the light screen

What will you work on today?

Today i started to help a fellow group member, on the temperature screen i was doing the functionality of the seek bar

Any roadblocks?

Ran into some problems with the seek bar where whenever i moved the bar either left or right it would give me a higher value then the one i set in the code

Paolo Quezon

What did you work on yesterday?

Attempted to fix the landscape layouts for my respective screens. Updated necessary logic to the temperature screen.

What will you work on today?

Update the positioning of assigned XML files and implement the use of notification functionality on the toggle button. Add logic for seekbar conversion and appropriate color coding for temp readings.

Any roadblocks?

Allowing notifications to be sent to the emulator depending on the temperature screen's status. Enable notification permissions for app to send a message via appropriate channel.

Denis Shwaloff

What did you work on yesterday?

Added functionality to my Switch screen. Re-arranged objects on the layout

What will you work on today?

Implement functionality for the Review screen and send user feedback to the database

Any roadblocks?

No problems at this moment

DAILY STAND UP

July 11

SPRINT 3

Alvaro Rodrigo Chavez

What did you work on yesterday?

After designing the feedback screen. I started to check if I could improve my code for the Smart Aqua Water Quality class

What will you work on today?

I will start to implement the dark mode feature and others in our settings screen. As well as the shared preferences requirement

Any roadblocks?

Might run into errors at coding the functionality but if this happens I will research and debug my code to solve them

Nicholas Dibiase

What did you work on yesterday?

Yesterday i went through my screen and finalized the design layout for the new constraint layout

What will you work on today?

Today i have implemented the notification onto my light screen, when the user turns the light on a notification will appear telling the user the light is on

Any roadblocks?

No blockers.

Paolo Quezon

What did you work on yesterday?

Implemented the location runtime permission to display current device location. Also revised code to use strings instead of hard coded text and updated necessary string files.

What will you work on today?

Remove unnecessary user prompt for location runtime permission and implement the use of Async task to set location permissions.

Any roadblocks?

According to user's decision the device's location should/not be shown via long and lat. Setting the proper logic to set the emulator permissions to correspond with based of user selection.

Denis Shwaloff

What did you work on yesterday?

Updated our project's strings file. Added new strings and translations. Fixed translations for strings that were changed

What will you work on today?

Add AlertDialog and Toasts to reflect user made choices on the Review screen

Any roadblocks?

For some reason, Android Studio was throwing an error that some strings already had a translation, even though they were brand new

DAILY STAND UP

July 13

SPRINT 3

Alvaro Rodrigo Chavez

What did you work on yesterday?

Implemented functionality in the settings screen. In addition, I switched most of the xml designs from Frame Layout to Constrain Layout

What will you work on today?

I had trouble working with the dark mode and shared preferences for the app, therefore I am going to fix them by debugging as I have done before

Any roadblocks

Dark mode is not functional as I want, therefore I have to research a solution to launch the app ignoring the phone theme

Nicholas Dibiase

What did you work on yesterday?

Yesterday I implemented the notification functionality that appears when the user turns the light on and off

What will you work on today?

Today i will fix all the UI elements on the light screen from framelayout to contraintlayout

Any roadblocks?

Had some issues with the layout, i had a hard time getting switch and image in the right place but with the help of my fellow group members i was able to fix this problem

Paolo Quezon

What did you work on yesterday?

Modified seekbar range and logic in temp fragment. Modified XML layout from frame to constraint to ensure better app consistency and display across devices.

What will you work on today?

Fix UI elements for screens that are not supported with constraint layout. Implemented logic for light image on the light fragment, when toggle status is on/off.

Any roadblocks?

Displaying OFF light when status is off and ON light when status is on. Initial logic had the images layered on eachother which was good, but could use adjustments to look more cleaner.

Denis Shwaloff

What did you work on yesterday?

Changed the way user reviews are parsed to the database. Changed the substitute for "." character

What will you work on today?

Update Switch screen Frame Layout to the Constraint Layout with Scroll View

Any roadblocks?

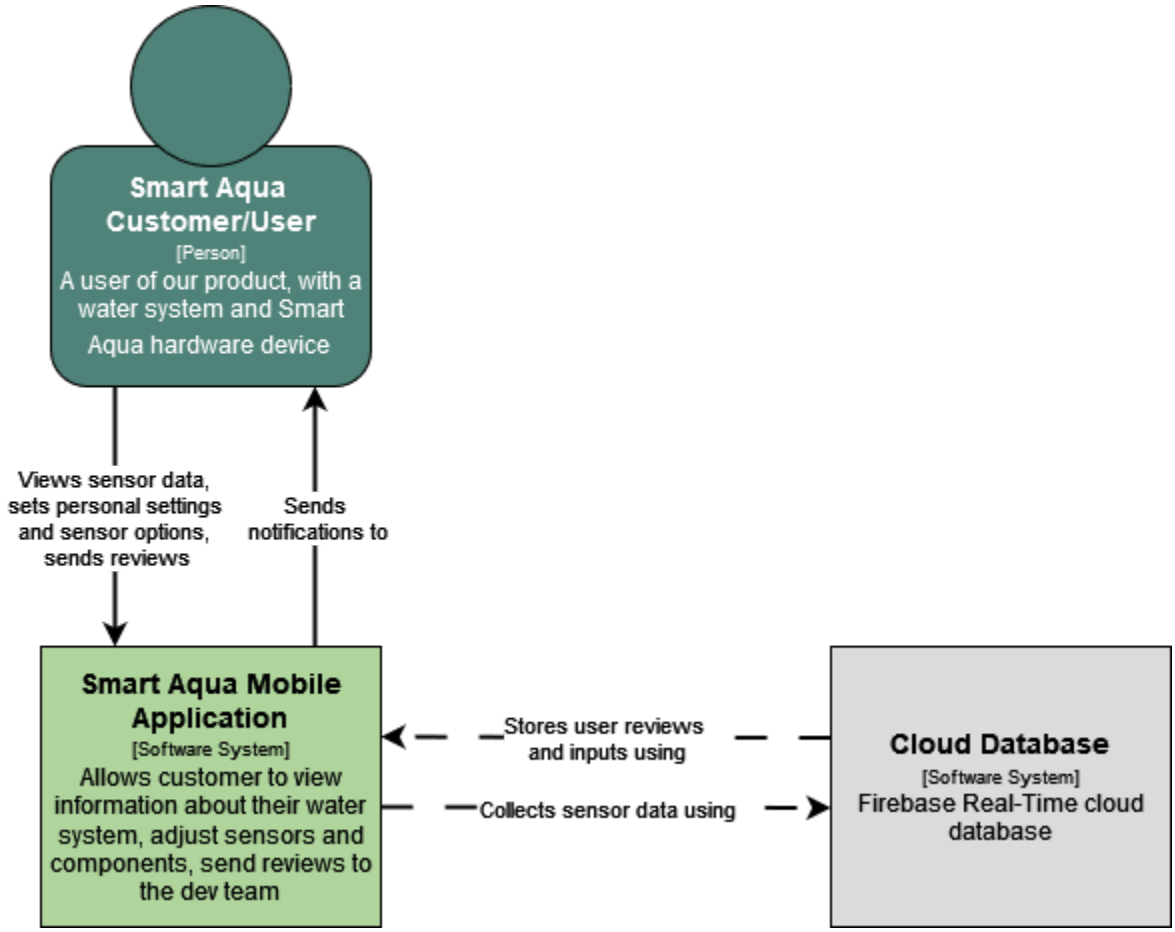
No problems so far

Sprint Retrospective

Buckle Up Inc.
Smart Aqua

Sprint Retrospective

C4 - Level 1: System Context Diagram



Design Patterns and Principles

Design Patterns

Creational Pattern

Builder

SmartAquaFeedback > AlertDialog

```
AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());

builder.setTitle(getResources().getString(R.string.reviewSendAlertTitle));

builder.setMessage(getResources().getString(R.string.reviewSendAlertBody));
              builder.setIcon(R.drawable.icon_error_48px);
              builder.setPositiveButton(R.string.reviewSendAlert, (dialogInterface,
i) -> {

                     SmartAquaReviews reviews = new SmartAquaReviews(name, email,
comment, model, number, rating);
                     dbReference =
FirebaseDatabase.getInstance().getReference("SmartAquaReviews");
                     DatabaseReference childReference =
dbReference.child(email.replace(".", ","));
                     childReference.setValue(reviews);

                     Toast.makeText(getActivity(), R.string.reviewSent,
Toast.LENGTH_SHORT).show();

                     ETname.setText("");
                     ETemail.setText("");
                     ETcomment.setText("");
                     ETnumber.setText("");
                     RBrating.setRating(0);
              });

             builder.setNegativeButton(R.string.no, (dialogInterface, i) -> {
                     Snackbar deniedBar = Snackbar.make(view,
R.string.reviewSendDenied, Snackbar.LENGTH_SHORT);
                     deniedBar.show();
              });

              AlertDialog alert = builder.create();
              alert.show();
```

The *builder* above process creates an AlertDialog with different properties.

Behavioral Pattern

Observer

SmartAquaQuality / SmartAquaQualityData > dbRef.child(reading_TDS_str).setValue(qualityData)

```
Button dbButton = view.findViewById(R.id.SmartAquaWQButton);

dbButton.setOnClickListener(view1 ->{
       // ...

       SmartAquaQualityData qualityData = new SmartAquaQualityData(reading_TDS_str,
status_TDS_str);
       dbRef = FirebaseDatabase.getInstance().getReference("QualityDataReadings");
       dbRef.child(reading_TDS_str).setValue(qualityData);

       // ...
});
```

In the SmartAquaQuality class, the SmartAquaQualityData objects are being created and saved to a Firebase Realtime Database. The SmartAquaQuality class acts as the subject or the publisher, while the Firebase Realtime Database can be considered as the observer or subscriber. Whenever new data is generated in the SmartAquaQuality class, it notifies the Firebase Realtime Database by setting the value in the database.

Design Principles

KISS - Keep it simple, stupid!

This is the most used principle that we try to use, since it is easy to understand and straightforward.

SmartAquaSettings > OnCreateView

```
ToggleButton darkTB = view.findViewById(R.id.SmartAquaDarkModeToggleBtn);
darkTB.setOnCheckedChangeListener(null); // Remove previous listener temporarily

// Get the initial state from SharedPreferences
final boolean[] darkModeCheckState =
{settingsPreferences.getBoolean("DarkModeToggleState", false)};
darkTB.setChecked(darkModeCheckState[0]);

darkTB.setOnCheckedChangeListener((buttonView, isChecked) -> {
   // Check if the state has actually changed
   if (isChecked != darkModeCheckState[0]) {
       if (isChecked) {
           AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_YES);
           Toast.makeText(getActivity(), R.string.darkModeON,
Toast.LENGTH_SHORT).show();
       } else {
           AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_NO);
       }
       settingsPreferences.edit().putBoolean("DarkModeToggleState",
isChecked).apply();
       darkModeCheckState[0] = isChecked; // Update the initial state
   }
});
```

In this snippet of code, we made the toggle button for the dark mode feature. Then using Shared Preferences we set it up for the user to exit and open the app keeping the setting chosen. Then if/else statements are used to identify when the toggle button is enabled to change the app theme.

SmartAquaTemperature > OnCreateView

```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
   View view = inflater.inflate(R.layout.fragment_smart_aqua_temperature, container,
false);
   seekBar = view.findViewById(R.id.SmartAquaTempSeekBar);
   textView = view.findViewById(R.id.SmartAquaTempReading3);

   sharedPreferences = PreferenceManager.getDefaultSharedPreferences(getContext());
   int savedProgress = sharedPreferences.getInt("TemperatureProgress", 0);
   int savedTemperatureRange = sharedPreferences.getInt("TemperatureRange", 18); //
Default value is 18
   boolean toggleState = sharedPreferences.getBoolean("TempPref", false);

   seekBar.setProgress(savedProgress); // Set the saved progress
   setTemperatureText(savedTemperatureRange); // Set the saved temperature range

   seekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
       @Override
       public void onProgressChanged(SeekBar seekBar, int progress, boolean b) {
           int temperatureRange = (int) (progress * 0.09) + 18; // Map progress from
18-27
           setTemperatureText(temperatureRange); // Update temperature text
```

```
            // Save the progress and temperature range in shared preferences
            SharedPreferences.Editor editor = sharedPreferences.edit();
            editor.putInt("TemperatureProgress", progress);
            editor.putInt("TemperatureRange", temperatureRange);
            editor.apply();
        }

        @Override
        public void onStartTrackingTouch(SeekBar seekBar) {
        }

        @Override
        public void onStopTrackingTouch(SeekBar seekBar) {
        }
    });

    ToggleButton toggleButton = view.findViewById(R.id.SmartAquaTempToggleButton);
    toggleButton.setChecked(toggleState);
    toggleButton.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
        @Override
        public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {
            String message = isChecked ? getString(R.string.tempNoti_ON) :
getString(R.string.tempNoti_OFF);
            displayNotification(message);

            // Save the toggle state in shared preferences
            SharedPreferences.Editor editor = sharedPreferences.edit();
            editor.putBoolean("TempPref", isChecked);
            editor.apply();
        }
    });

    return view;
}
```

The snippet of code above adheres to the KISS (Keep it Simple, Stupid) design principle as it aims to keep the code simple yet straightforward, while avoiding unncessary functions. By separating the functionality into smaller methods like the 'setTemperatureText' and 'displayNotification' it makes the code easier to manage and maintain. Generally speaking the code above keeps the code implementation simple and priortizes readability.

Single Responsibility Principle (SRP)

This principle is used to display the simulated data from the database.
The SmartAquaQuality class uses the SmartQualityData to retrieve the readings displaying the app to then being stored in the database.

SmarAquaQualityData

```
public class SmartAquaQualityData {
    String reading_TDS, status_TDS;
```

```
    public SmartAquaQualityData() {
    }

    public SmartAquaQualityData(String reading_TDS, String status_TDS) {
        this.reading_TDS = reading_TDS;
        this.status_TDS = status_TDS;
    }

    public String getReading_TDS() {
        return reading_TDS;
    }

    public void setReading_TDS(String reading_TDS) {
        this.reading_TDS = reading_TDS;
    }

    public String getStatus_TDS() {
        return status_TDS;
    }

    public void setStatus_TDS(String status_TDS) {
        this.status_TDS = status_TDS;
    }
}
```

SmartAquaQuality > OnCreateView

```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
    // Inflate the layout for this fragment
    View view;

    view = inflater.inflate(R.layout.fragment_smart_aqua_quality, container, false);

    readings_TDS = view.findViewById(R.id.SmartAquaWaterReadings);
    status_TDS = view.findViewById(R.id.SmartAquaWStatus);

    Button dbButton = view.findViewById(R.id.SmartAquaWQButton);

    dbButton.setOnClickListener(view1 ->{
        String reading_TDS_str = readings_TDS.getText().toString();
        String status_TDS_str = status_TDS.getText().toString();

        SmartAquaQualityData qualityData = new SmartAquaQualityData(reading_TDS_str,
status_TDS_str);
        dbRef = FirebaseDatabase.getInstance().getReference("QualityDataReadings");
        dbRef.child(reading_TDS_str).setValue(qualityData);
        Toast.makeText(getActivity(), R.string.save_data, Toast.LENGTH_SHORT).show();
    });

    readDataFromDatabase();

    return view;
}
```

Coding work progress since Deliverable 2.

What additional features/functionality added since Deliverable 2.

We've made substantial improvements to the Smart Aqua app since Deliverable 2. We redesigned the feedback screen page to meet new requirements and introduced valuable functionality by integrating with Firebase Real-time Cloud Database. This integration enables the Quality Screen to seamlessly retrieve and display data directly from the Cloud Database. Our efforts also focused on enhancing the Settings screen, resulting in significant improvements such as the addition of Dark Mode, Mute, and Reset Settings options. These additions provide users with more control and customization over their app experience. To ensure a seamless user experience, we implemented Shared Preferences, allowing the app to remember and store user inputs and settings whenever they close the app. This feature saves users' time and ensures their preferences are preserved across sessions. Moreover, we upgraded the Location request feature with AsyncTask functionality, intelligently managing when users are prompted to share their location. The app now stores users' location sharing preferences and prompts them only when there is a substantial change in their current location. This streamlined approach reduces interruptions and respects users' choices. The home screen received a visual makeover, offering clear visuals that effectively communicate the purpose and contents of the app. Additionally, we implemented an app-wide theme, supporting both regular/light and dark mode options. This cohesive design approach provides a unified and consistent look across the Smart Aqua app, enhancing its visual appeal and user experience. Lastly, we optimized all screens by migrating to Constraint Layouts with Scrollview. This adaptation ensures compatibility across various screen resolutions, providing a seamless experience for users on any device. In summary, our substantial updates to the feedback screen, including the redesigned interface, Firebase integration, enriched Settings screen, and improved features such as Shared Preferences and enhanced Location request, significantly elevate the overall user experience and visual appeal of the Smart Aqua app.

Runtime permission and Implementation

Overview

The Smart Aqua app has a feature that will use the location runtime permission to determine the approximate position of the user's device. For testing reasons, the emulator's position is displayed as a toast message using longitude and latitude coordinates.

Functionality

We've included a location button in our app's settings section to allow users to modify their location permissions When the user taps the location button, they are presented with three options:

Always Allow: If the user selects this option, the device's location permission is set to "Allow only while using the app" with precise location enabled.

Allow Just Once: Selecting this option prompts the device to ask the user for location access every time the app is launched.

Deny: If the user rejects the app permission to access their location, future clicks on the location button will not trigger the permission popup.

Behavior

When the user hits "Allow" or "Allow Just Once," the longitude and latitude of the user's device appears on the screen as a toast message. This data may be utilized by the app for a variety of purposes.

If the user chooses to decline permission to access their device's location, future tapping on the location button will not trigger the permission popup. The app will no longer be able to access the user's location.

Implementation

The code uses the 'ACCESS_COARSE_LOCATION' permission to grant runtime access to the device's coarse location. It checks and requests permission asynchronously using an AsyncTask named 'LocationPermissionTask'. If the permission is granted, the task's 'doInBackground' function shows the current location using the 'displayCurrentLocation' method. If the permission is not given, the 'onPostExecute' function asks the user for it. By carrying out the permission check and request in the background, the code effectively manages the user's answer to the permission request and ensures the app remains responsive.

Conclusion

Our app's implementation of the location runtime permission gives users control over their location data. Users can change their settings for location access within the app by selecting "Always Allow," "Allow Just Once," or "Deny" from the prompt choices. This

approach ensures that our software protects user privacy while still providing location-based features and functionality.

Cloud Database

Switched to Firebase Real-Time database due to better integration with Android Studio and Android Framework.