The Smart Blind Addon

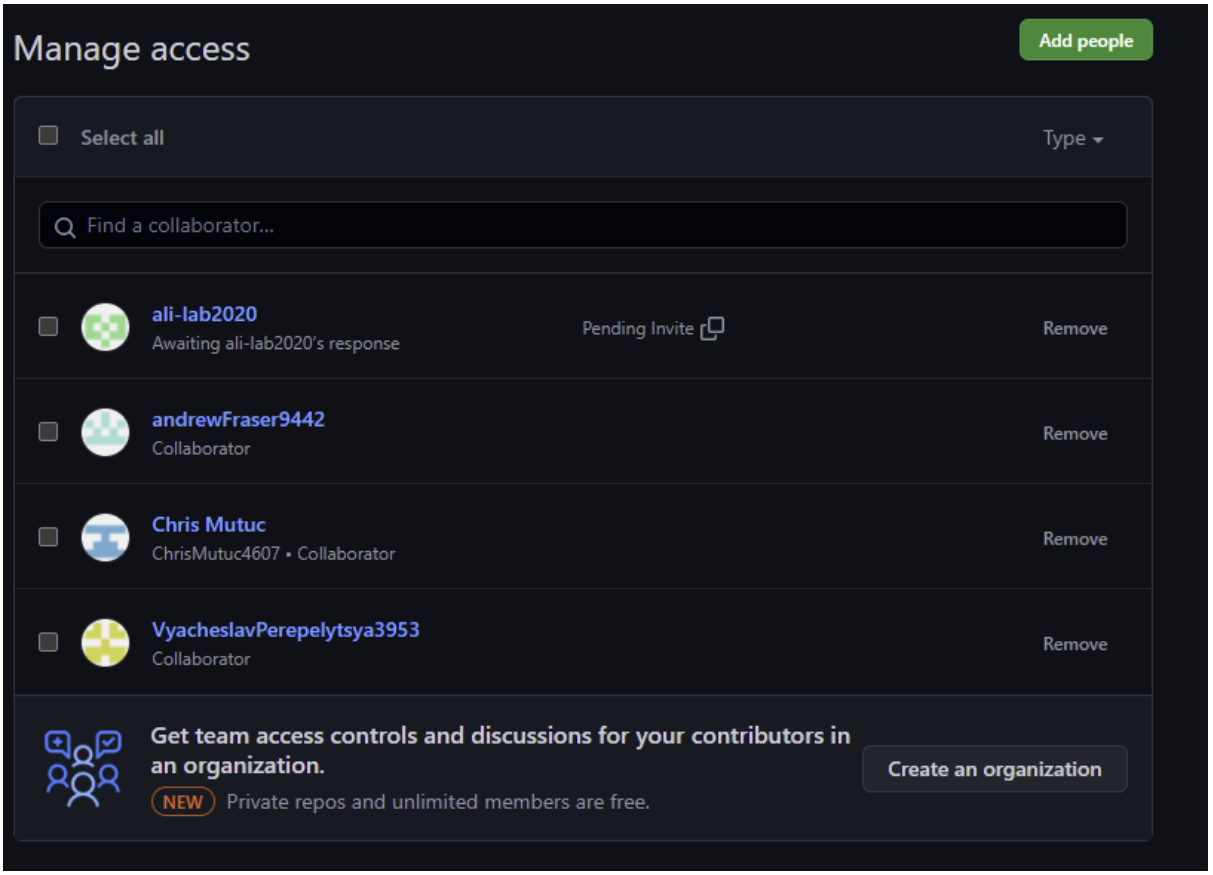
Developed by: Blind Dev

GitHub

Link

https://github.com/AmitPunit3930/SmartBlindAddon

Screenshot



Developers

Vyacheslav Perepelytsya n01133953
Amit Punit n01203930
Chris Janelle Mutuc n01314607
Andrew Fraser n01309442

Developer Signatures

Amit Punit

Chris Mutuc

Andrew Fraser

Vyacheslav Perepelytsya

Project Scope

The Smart Blind addon is an add-on device that is used to wirelessly operate corded blinds. It will use a dc motor, light sensor, distance sensor, and temperature sensor in order to control and operate blinds wirelessly in real-time.

The device is basically a box that connects operation controls of most blinds that contains a motor that will control the opening and closing of the blinds, and a temperature sensor to monitor the room so that if the room gets too hot then the blinds will automatically close in order to cool the room and the temperature will be set by the user using the app. The next sensor is the light sensor, this will monitor the light in the room and if the room gets too bright then it will automatically close the blind (In potential subversions AI will also record data to learn how bright the user wants the room). And the final sensor is a distance sensor to calibrate the device to prevent damage and ensure the correct function of the blind. The project will be complete when the device is able to operate a window blind based on the user input onto the real-time Firebase database.

Sensors

Raspberry Pi Controller

This is the main controller of the device, this will interpret commands from the app to either get data from the sensors or activate the motor. This will also periodically send data from sensors to a Firebase database to be displayed on the device as well as to get important data from the user.

5V DC Motor and L2953D

The DC motor is what will connect to the blind operating mechanism and provide the kinetic function that is needed. The L293D H-bridge is what will supply power to the motor, as well as, based on the commands of the controller, spin the motor clockwise or counterclockwise which will open or close the blind. The L293D IC chip will be soldered onto the device PCB board and will most likely be wired to the Raspberry PI's headers. The DC motor will be connected to the board via jumper wires. To power the motor a detachable 6V battery pack will be connected to the PCB board.

IR Distance Sensor (GP2Y0A02YK0F)

The IR distance sensor is there to prevent the blinds from going over the length of the window. It will be placed facing down where the blinds will drop and monitor the maximum distance the blinds can go down. The height measurement will be provided via user input to the app and ultimately the Firebase database.

Light Sensor (LM393)

The light sensor will record the amount of ambient light in the environment of the blind. This data will be sent to a Firebase database where the data will then be read from the app and displayed. The value will be updated periodically.

Temperature Sensor (MLX90614ESF)

The temperature sensor is for detecting the ambient temperature of the room and having it trigger either to close the blinds or open it. For example, if the room is too hot the sensor will indicate this and send it to the motor to open the blinds.

App Functions

Manage Blinds

This page will allow users to delete and add blinds. This will be done by either adding data or deleting data from the Firebase database. All blinds that are connected to the user will be listed here as well.

Add Blind

This will be a fragment that will be accessible from the Manage Blinds fragment. This page will connect to the device via the Firebase database. At this time the user can enter the height parameters as well as other information about the blind to be stored in the database.

Connect Blind

This will be done on the homepage or main activity, the connection will be made through Firebase and this is how the phone will send the command to open or close the blind. This process would be hidden from the user and will only interact with customers if they encounter errors.

Operate Blind

This action will be done on the home page with buttons when pressed it will update the device status on the Firebase real-time database. Thus when the change is made it will trigger the appropriate function on the device. There will be two buttons, one to open the blind and one to close the blind.

Scheduled time blinds to open and close

This function will take in either an edit text field or a date text field so the user can enter the desired time and date the blinds should be closed or opened. The operation of this function is that the device will get the data via a connection to the database and either create or update a stored variable on the device.

Troubleshooting Steps

This page can be accessed from the navigation drawer, this fragment will have instructions on how to add or delete a blind, how to install the device, as well as common errors a user might run into.

Company Contact Page

This page will be accessible from the navigation drawer. This will have a list of names of the developers of the device. This will also have a form where users can submit feedback. This data will also be stored on Firebase.

Display Room Data

This feature will be displayed on the home page. This data will be retrieved from the sensors via the Firebase database. This data will also be used to learn when the user closes and opens their blinds so it can eventually operate the blinds automatically without the user's input.

Layout

Navigation Drawer

We find this overall layout will be best for the app. In the drawer, there will be useful information like the name and email address as well as an option to schedule times for the blinds to automatically open and close. The option to manage blinds will be listed in the middle. The next option would be to access the troubleshooting steps page. Finally, there will be an option for the company contact page.

Screen Flow

The flow of pages for the app will be broken down into three ways the page can be accessed, the first way is through the navigation drawer layout, the second is through the top menu options, and the final way is through buttons that will be on the page.

The pages that will be accessible through the navigation drawer layout are the troubleshooting page, the contact form, the blind management page, and the schedule blinds page.

The pages that will be accessible from the top menu are the login page which will be accessible through the menu overflow. The other page that will be accessible is the homepage, which will have an icon that will always be displayed.

The pages that will be accessible from buttons will be a new user registration form that will be accessible from a button on the login page as well as the page for reporting an issue, which will be accessed via a button on the troubleshooting page.

Login Screen

This screen will allow the user to log in or register to the application. It will have a text field for the user's name, address (optional), phone number (optional), and email address. The password that the user enters will be encrypted.

Home Page

This will be the first page that will be displayed to the user once he opens the app, on this page, there will be an XML-generated layout of all the connected blinds. On each blind display, there will be two buttons to open and close the blinds. At the top of the page, there will be a display of the time and date, the inside temperature, and the current light reading.

Menu

The menu will have the app name, as well as an option to log in as a user in the overflow menu. There will also be a home icon that will take the user to the homepage when pressed.

Manage Blinds

At the top of the page, the app will have two buttons, one for adding a blind and the other for deleting a blind. When the user presses one of these buttons the appropriate form will be displayed. Under the form, there will be a list of all the blinds that are connected to the user. The add blind form will have an edit text field to enter the blind name, location, and length of the blind. The delete form will display all the connected blinds in a radio button group and the user will select one they want to delete and press a submit button to make the change.

Schedule Blinds

There will be two edit text fields where a user can enter a blind and the time the blind will be scheduled to operate as well as the operation to be executed, there will also be a

button to submit the data. On successful data entry, there will be a toast that will be displayed to inform the user of it.

Troubleshooting

There will be a list view with the potential and common issues that will be covered in the fragment, when an item is selected text will be displayed with the relevant information to the issue.

Contact Forum

The contact forum will have four edit text fields where the user can enter their name, email, phone number, and description of their issue. The data will then be sent to the Firebase database so a developer can review the issue and either guide the user to solve the problem or patch the app so the issue is resolved.

Integration

The overall integration of the app and the device is that both the app and the device will use a Firebase database to set and exchange data as well as to authenticate users. For direct control of the device, the Firebase real-time database will be used to execute basic commands, like opening and closing blinds. The reason for using Firebase instead of other database services like SQL is the high integration with mobile development and real-time capabilities of the service.

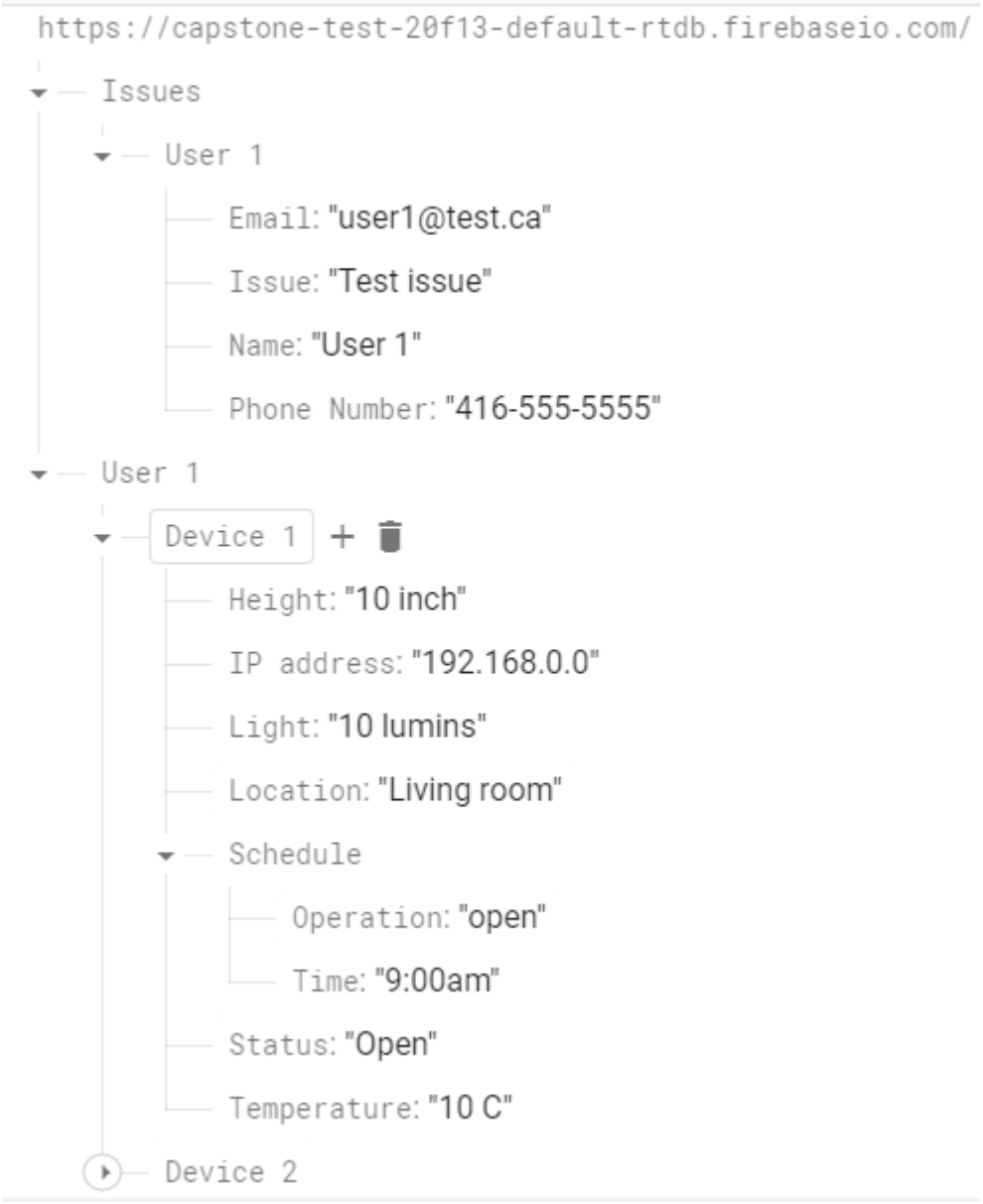
User Authentication

For user authentication, users can either use an email or password method as sign-in credentials or sign in with a google account using Firebase built-in authentication tools. When a user is created Firebase automatically creates a user ID, this will be used to create the real-time database and it will be sent to the device so it can also use the same database for sending sensor data.

Data Storage Structure

The Firebase real-time database will be organized with all relevant data being under the users. The user's id will be created when they are authenticated by Firebase, once a device is created it can be added to the appropriate users, and sensor data will then be organized under the device.

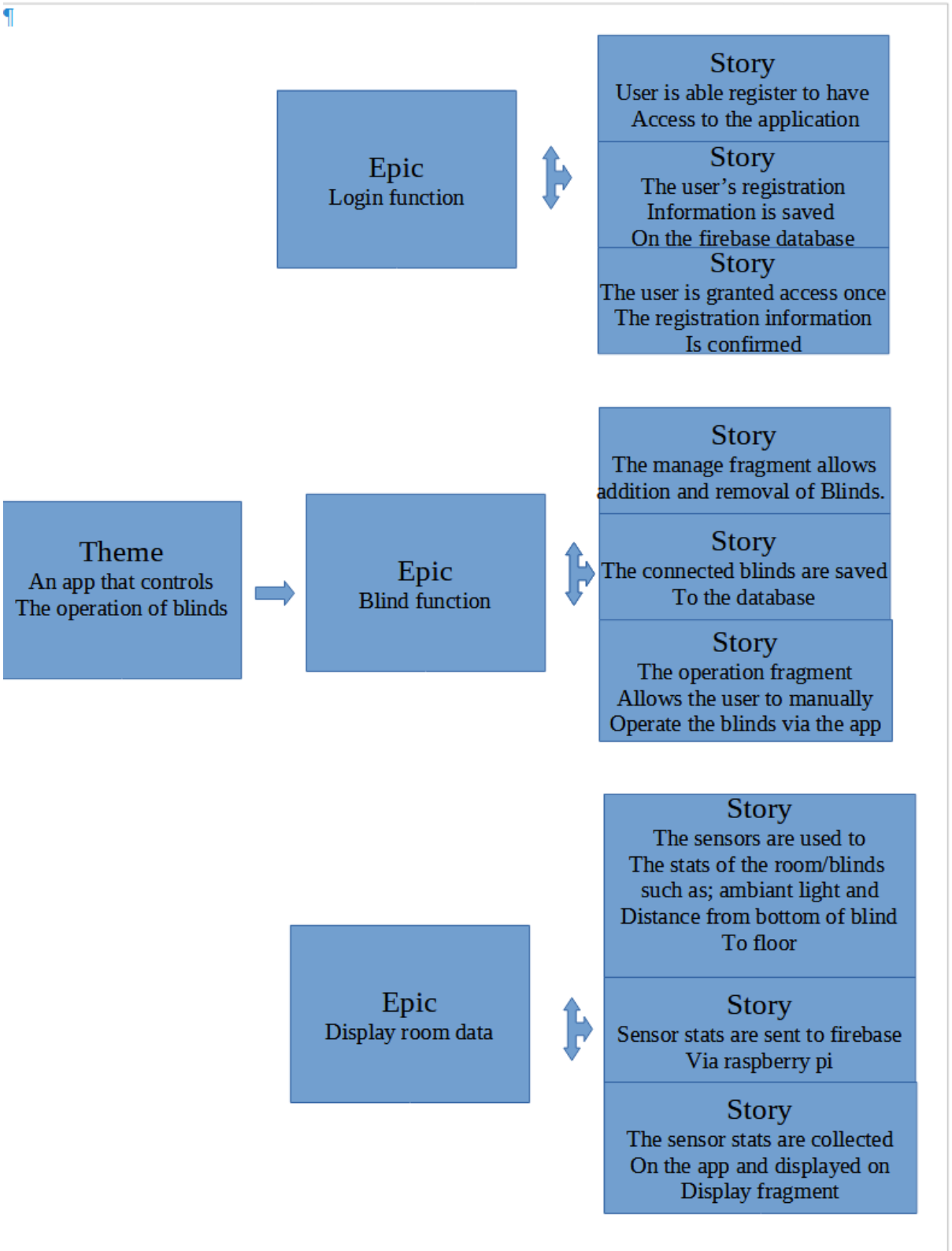
The app will read the light value, temperature, and Ip address from the Firebase real-time database. The Device will read the height, location, status, and schedule from the database. The app will write the height, location, status, and schedule, and the device will write the light and temperature values to the database. Issues that the user might want to report will be organized under the issues section with a users subsection for further organization. Below is a visual representation of the firebase real-time database.



Client Feedback Integration

We would first analyze the feedback and, if feasible, apply necessary changes to the app or explore alternative solutions with the client. Analyzing each client's feedback will maintain the quality of the app, making sure the changes proposed are valid or feasible. Checking case-by-case feedback will help us identify good feedback and bad feedback. Having scheduled meetings among the team and discussing the feedback as a team will help and approach the feedback precisely. Creating and maintaining a track record for all the client's feedback will help us keep track as well as generate test cases regarding the feedback changes, and also keep track of errors from the tests.

Theme


Epic
Login function
Story
User is able register to have
Access to the application
Story
The user's registration
Information is saved
On the firebase database
Story
The user is granted access once
The registration information
Is confirmed
Theme
An app that controls
The operation of blinds
Epic
Blind function
Story
The manage fragment allows
addition and removal of Blinds.
Story
The connected blinds are saved
To the database
Story
The operation fragment
Allows the user to manually
Operate the blinds via the app
Epic
Display room data
Story
The sensors are used to
The stats of the room/blinds
such as; ambiant light and
Distance from bottom of blind
To floor
Story
Sensor stats are sent to firebase
Via raspberry pi
Story
The sensor stats are collected
On the app and displayed on
Display fragment