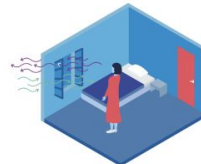


Natural Ventilation Estimation Tool

Final presentation



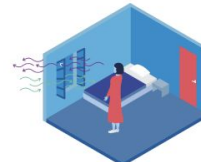
Our Team



Felipe Azank
POLIMI



Felipe Bagni
POLIMI



Project Customer



The United Nations agency working to
promote health, keep the world safe and
serve the vulnerable

Represented in this project by:

- Anna Silenzi
- Luca Fontana

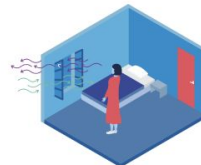


Table of Contents

1

Objectives

2

Design Process

3

Technology Stack

4

Architecture

5

Development and Functionalities

6

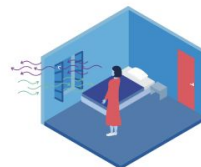
Testing

7

Future Improvements

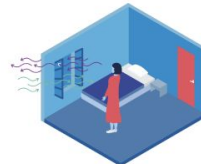
8

Live Demo



Objectives and App Concept

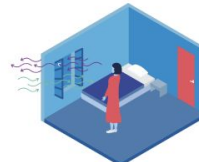
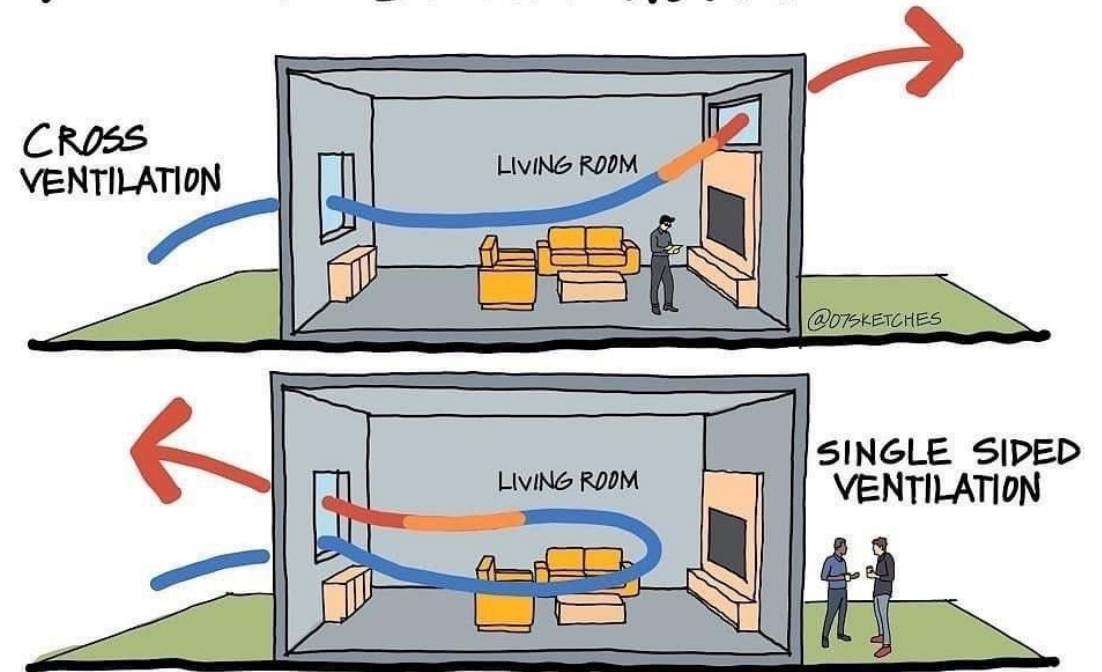
Understanding the Problem, the Requirements and the what's to be done



Problem & Solution: Mission

- Proper natural ventilation plays a key role on controlling and preventing infections in healthcare.
- **Maximizing the amount of people able to use natural ventilation to fight infections** can play a huge role in everyday life and in possible upcoming epidemics.

TYPES OF VENTILATION @07SKETCHES

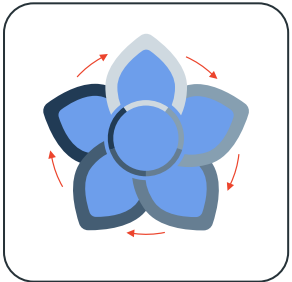


Problem & Solution: Mission



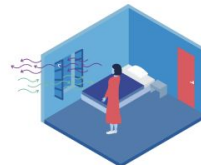
UN guidelines - 2007:

- Natural ventilation is considered for the first time among the effective measures to control infections in health care:
“The guideline describes how an airborne precaution room and its adjacent areas can be designed to provide natural ventilation control of infections.”



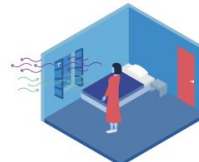
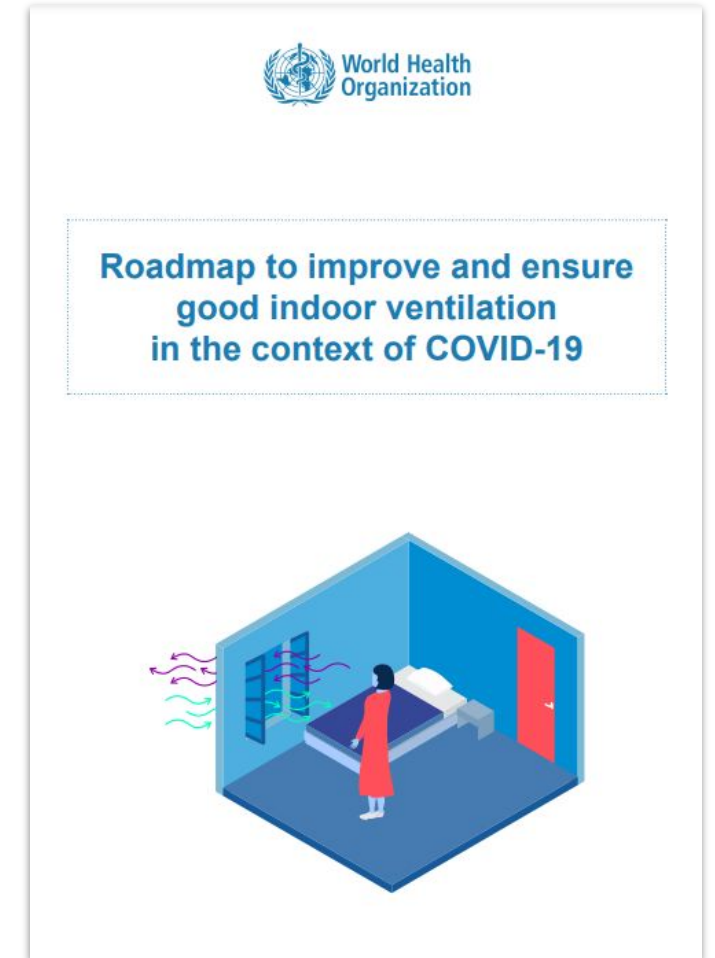
Air Flow Estimation:

- Proper estimating the air flow and the risks on a natural ventilation system is complicated and requires specialist on the field, which is something most people don't have access to



Problem & Solution: Proposition

- Create a simple and democratic application that can help **non-specialized users** on correctively estimating the ventilation rate within an indoor environment.
- This is a key step to consider the strategies to improve indoor ventilation as suggested in the document **"Roadmap to improve and ensure good indoor ventilation in the context of COVID-19"** from WHO.



Solution: job to be done

Side 1: toward exterior

opening typology 1

Length (m) 1.9

Height (m) 0.8

Mosquito net no

% Opening completely

opening typology 2

Length (m)

Height (m)

Mosquito net

% Opening

opening typology 3

Length (m)

Height (m)

Mosquito net

% Opening

Side 2: toward exterior

opening typology 1

Length (m) 1.8

Height (m) 1.2

Mosquito net yes

% Opening completely

opening typology 2

Length (m)

Height (m)

Mosquito net

% Opening

opening typology 3

Length (m)

Height (m)

Mosquito net

% Opening



Ventilation Calculator

Hey, there 🙋

Let's calculate your room ventilation!

Select your setting of interest

Residential/Comercial

What are the room's dimensions?

length

height

width

Enter dimensions

Length 5 meters

Height 3 meters

Width 3 meters

How is your room ventilated?

Naturally Intentional building

Mechanically Powered fans

Ventilation Calculator

Hey, there 🙋

Let's calculate your room ventilation!

Select your setting of interest

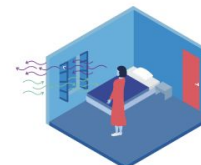
Residential/Comercial

What are the room's dimensions?

length

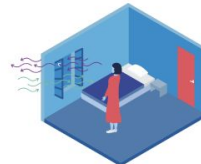
height

width

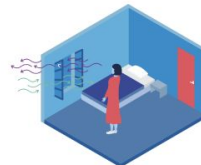
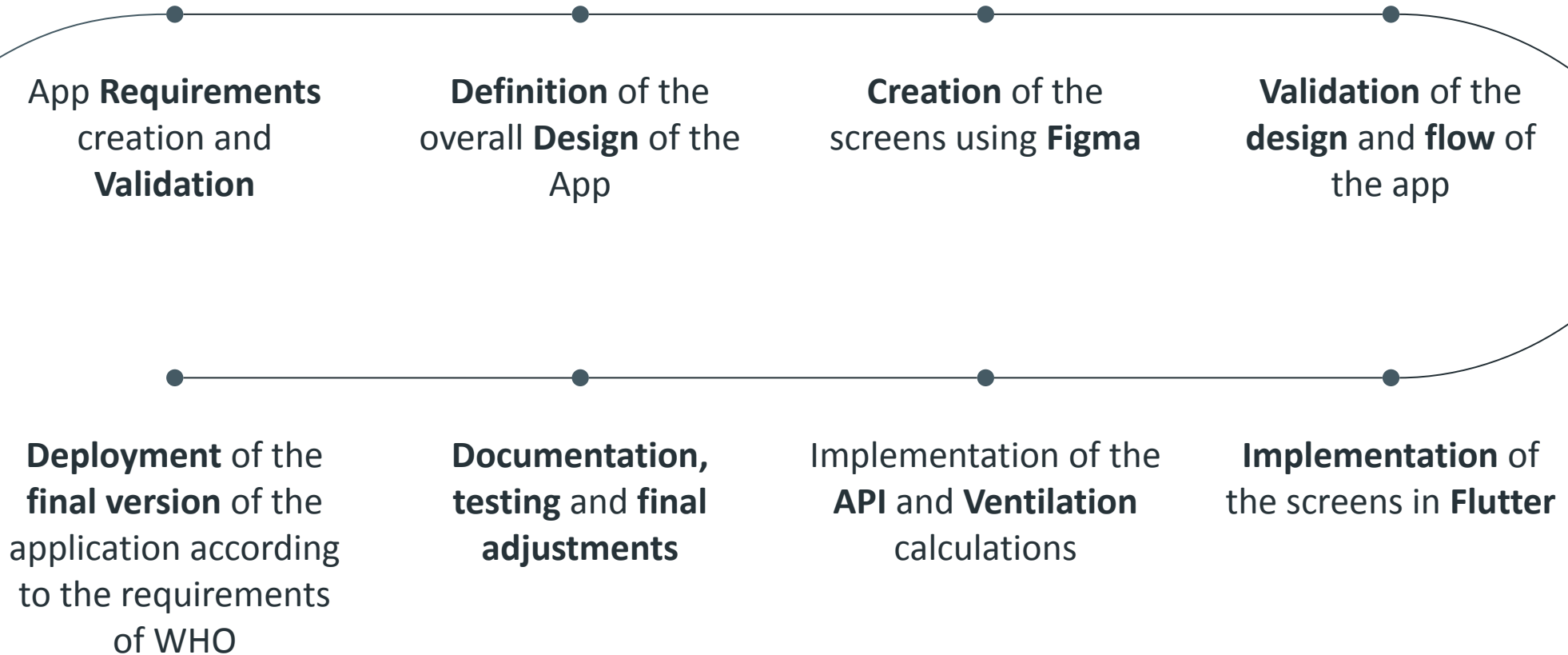


Design Process

From the requirements definition, to the design setup and first screens

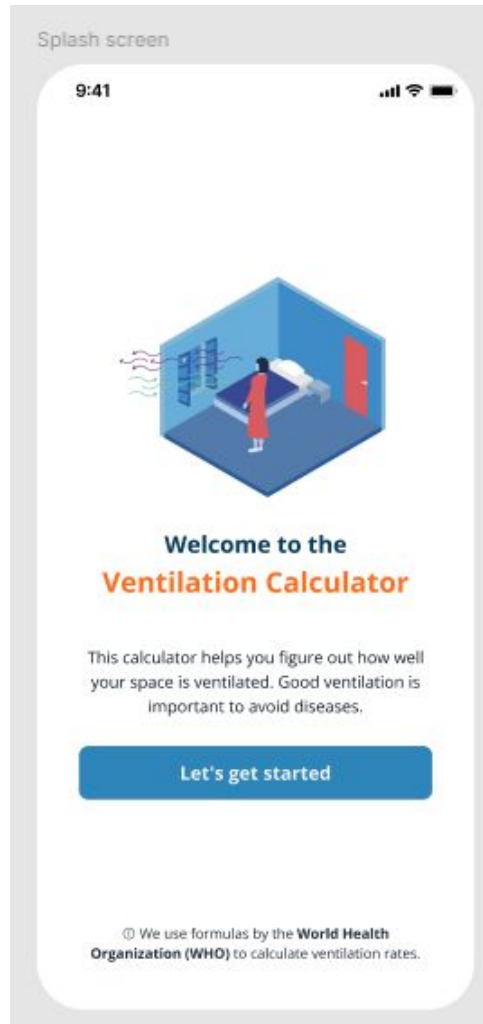


App Development Plan



Overall Design of the App

- Clear colors
- Simple layout
- Fonts that are easy to read
- Clear instructions

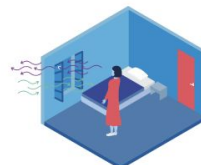


Main Background color

Second Color for highlights

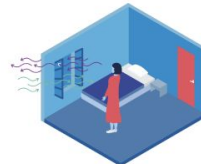
Main Color for buttons and inputs

Text Color in black



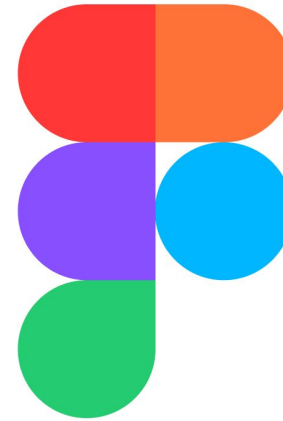
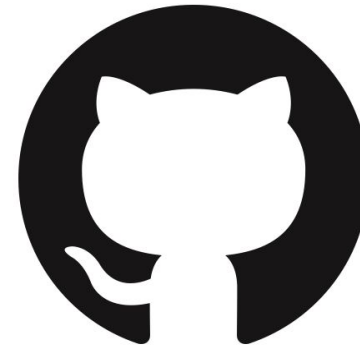
Technology Stack

Main technologies used during development



Technology Stack

- **Version control:** GitHub
- **Design and Front End:** Figma
- **Development and Testing:** Flutter
- **API Testing:** Postman
- **Emulating:** Android Studio



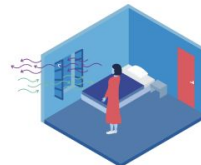
Flutter

Android Studio

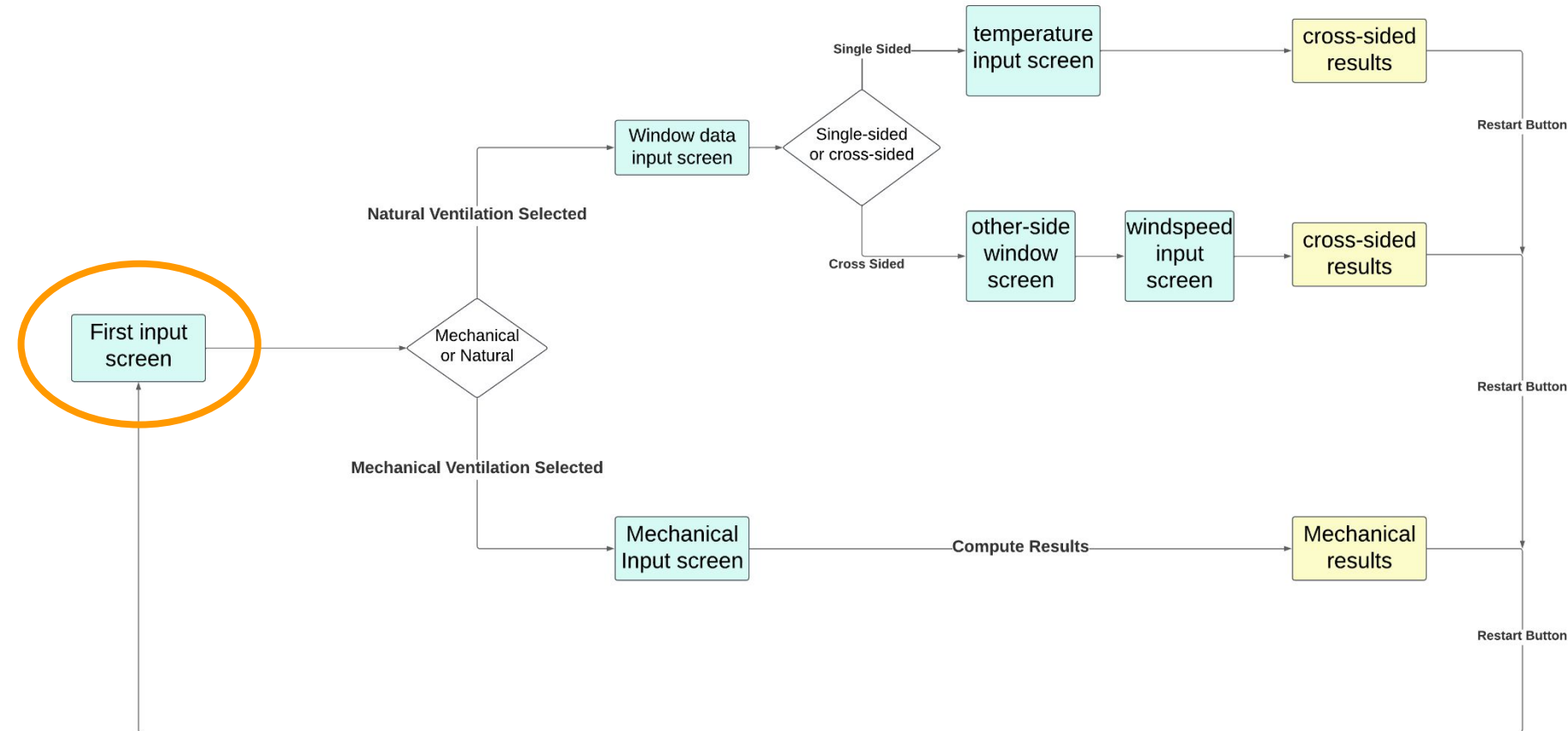
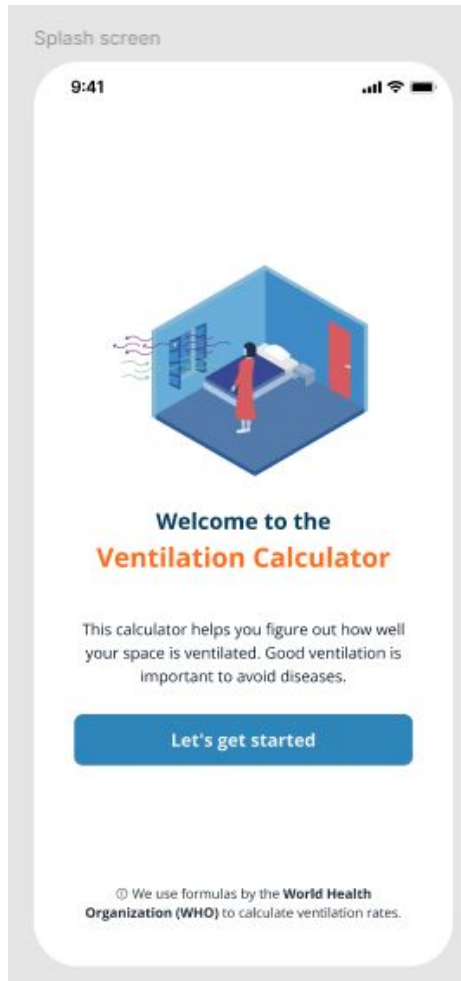


Architecture and Data Flow

Handling inputs and choices from the user



Flow of the application



Flow of the application

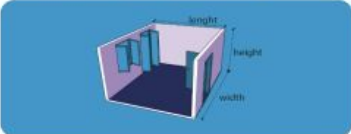
9:41

Hey, there 🌞
Let's calculate your room ventilation!

Select your setting of interest

Residential setting ▼

What are the room's dimensions?



Enter dimensions

Length: Unit: **meters** ▼

Height: Unit: **meters** ▼

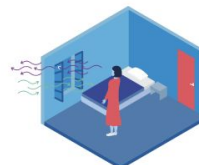
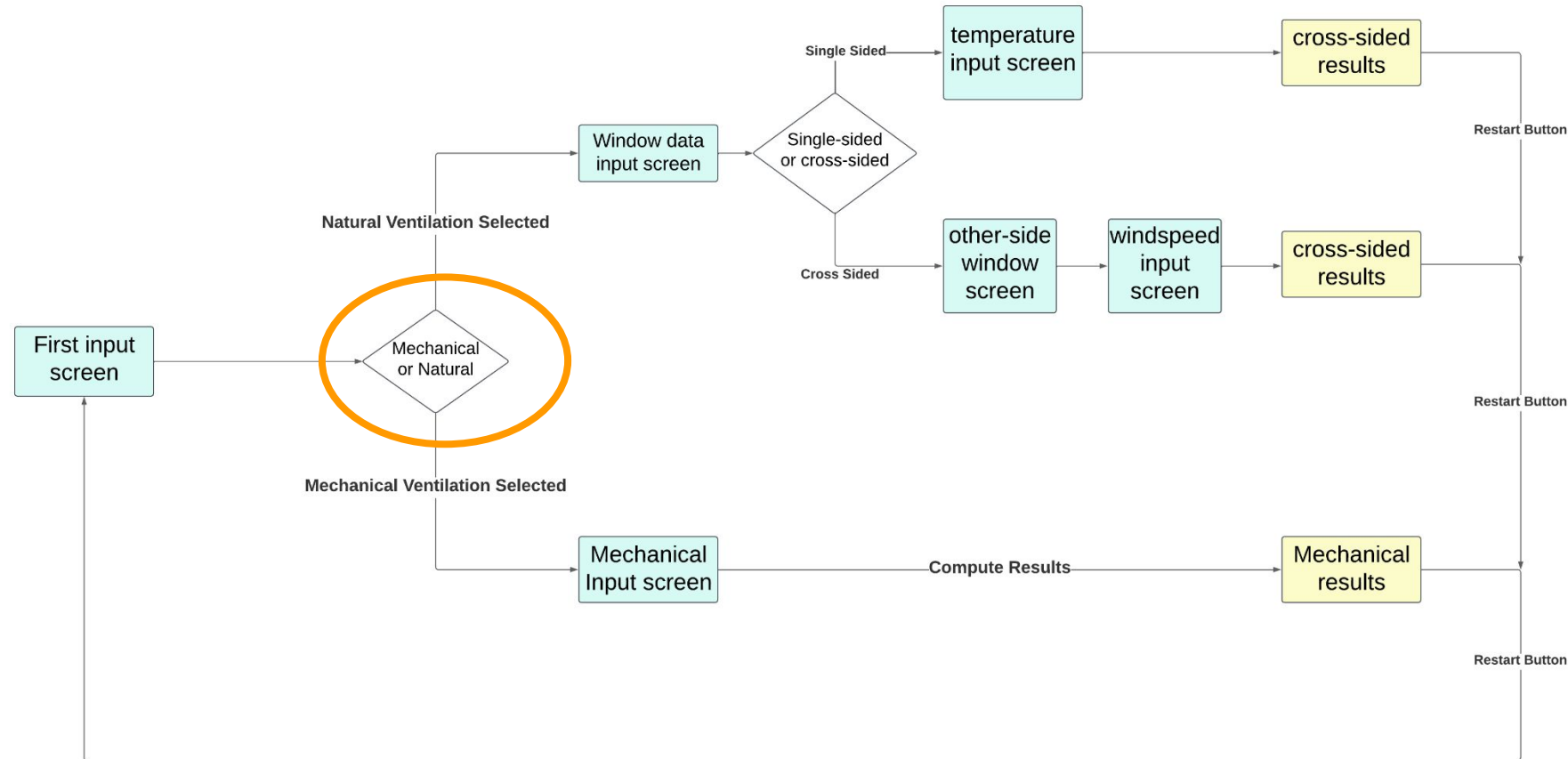
Width: Unit: **meters** ▼

How is your room ventilated?

Naturally
Intentional building openings

Mechanically
Powered fans or blowers

Next



Flow of the application

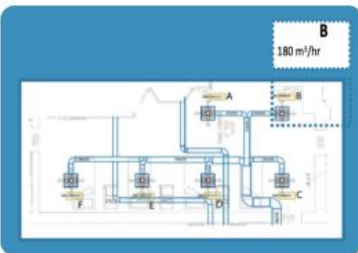
9:41

Air inlet/outlet vents

Manual Input

If you have information about the airflow in your vents, you can enter it here. This helps **estimate how often the air in your room gets replaced**.

If you're not sure, we can try to estimate it for you.



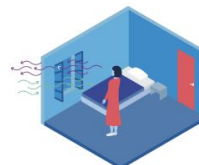
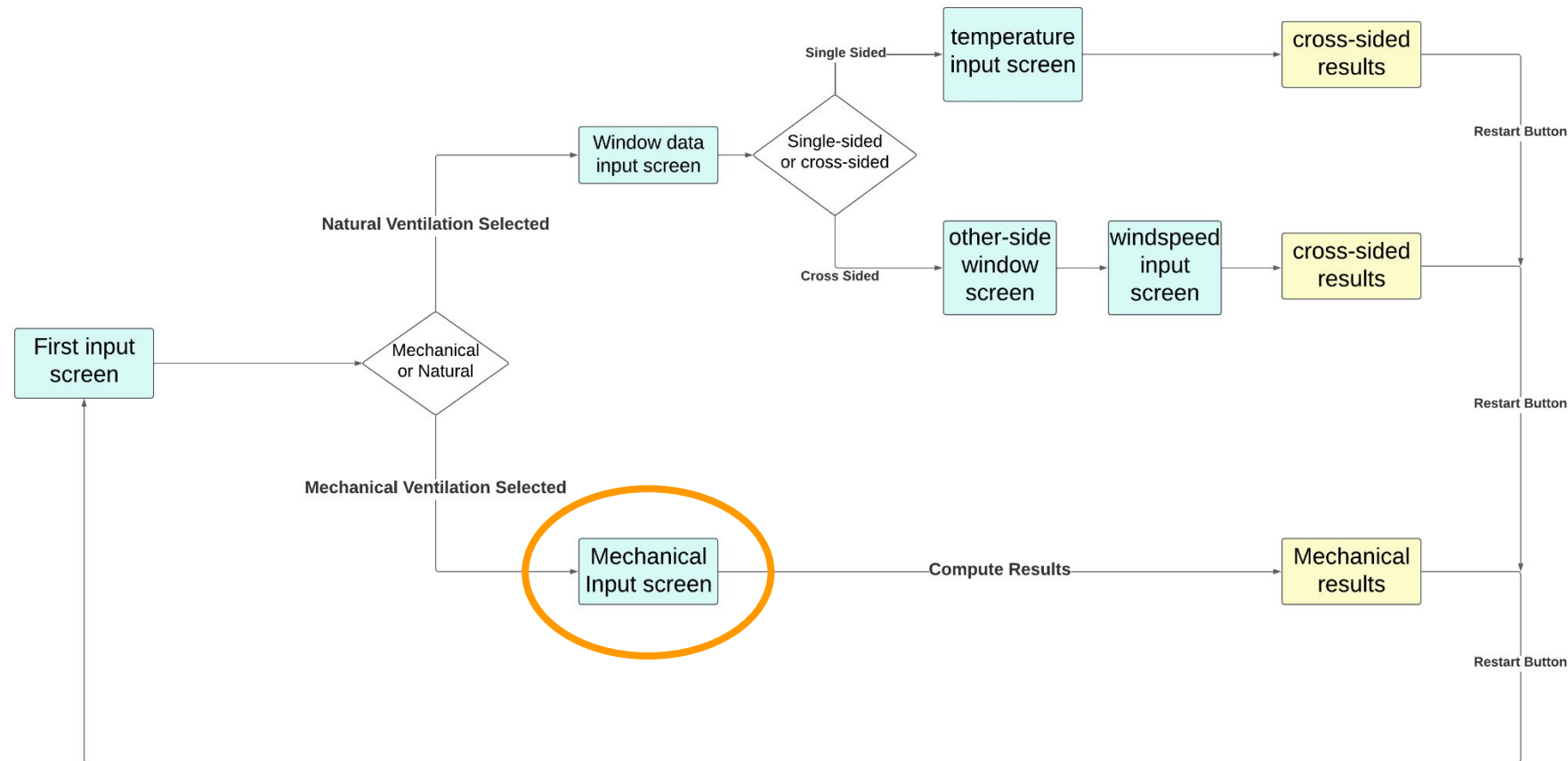
Enter dimensions

Vent A	Unit
1	m³/s
Vent B	Unit
20	m³/s
Vent C	Unit
20	m³/s

+ Add inlet/outlet vents

Air Changes per Hour (ACH)

[Calculated Value] Unit m³/s



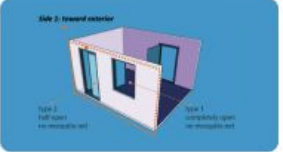
Flow of the application

9:41

Natural Ventilation

Openings Characteristics

We are now referring to the **facade toward the exterior**. Note that several typologies of opening could be available on the same wall (i.e. two different types of window or one window and one door).



What does "typologies of opening" mean? [See more](#)

Door/window number 1

Number of openings with the same size:

How much do you usually open it?

50%

Enter dimensions

Length Unit **meters**

Height Unit **meters**

Does it have a mosquito net? ☐

Door/window number 2

Number of openings with the same size:

How much do you usually open it?

80%

Enter dimensions

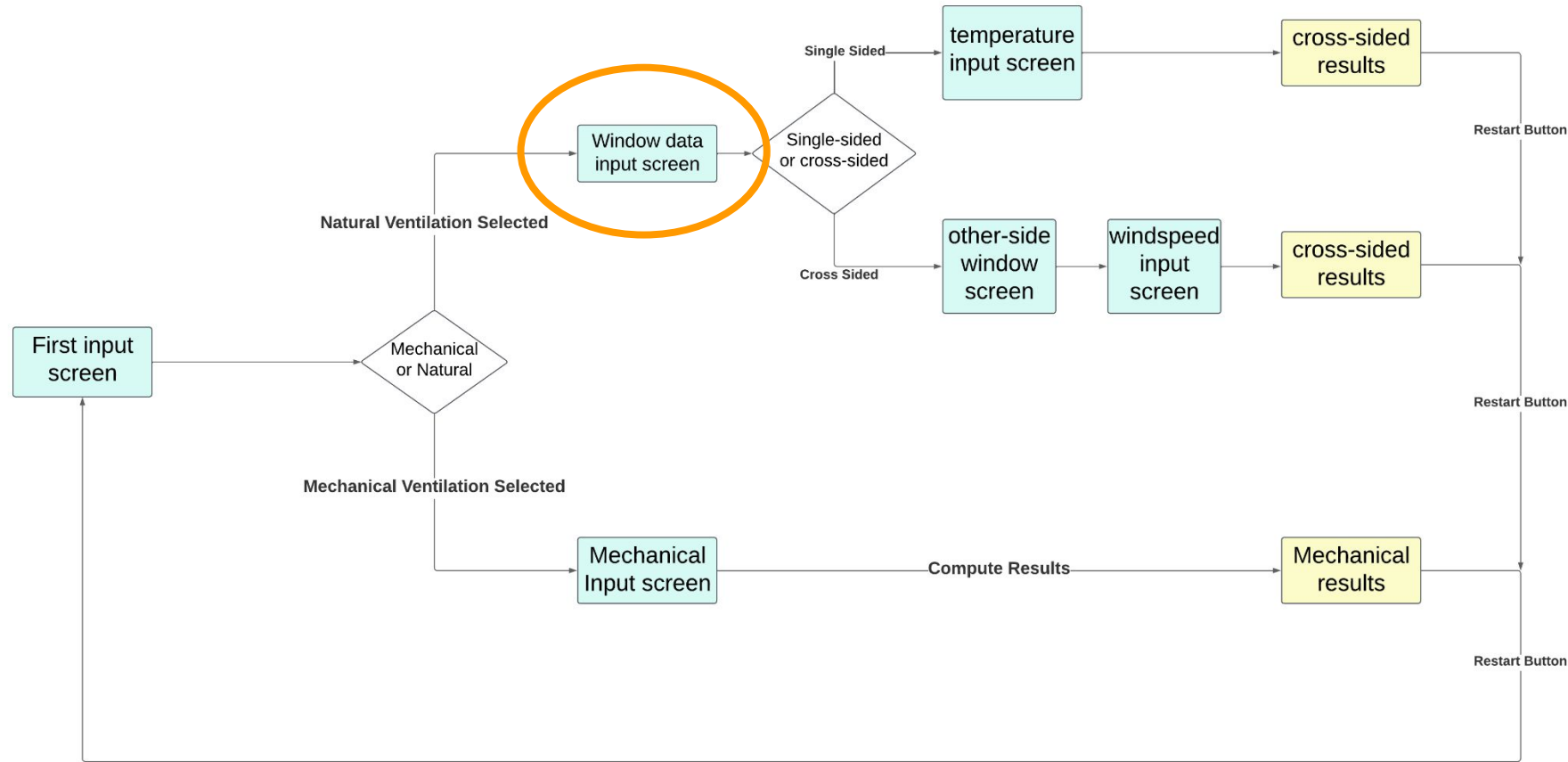
Length Unit **meters**

Height Unit **meters**

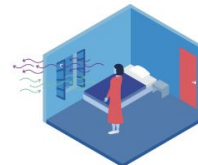
Does it have a mosquito net? ☒

[+ Add new opening type](#)

Next



Are there openings available in other side of the room?



Flow of the application

Opening Characteristics - Temperature Data

Temperature Data

Please Enter the temperature data for the location of the building as well as the overall temperature inside the room.

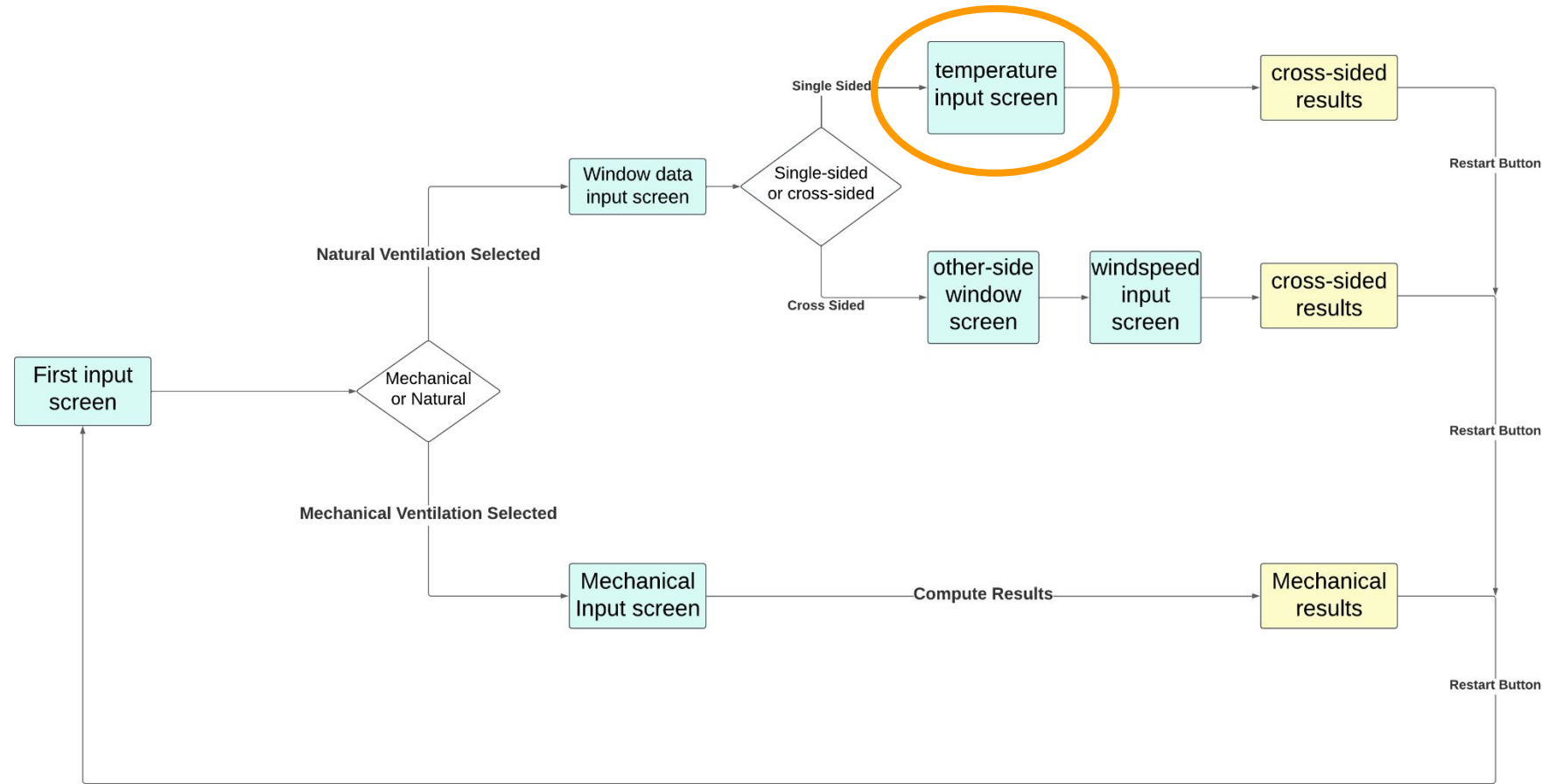
① If no data is available, enter "1".

Inside Temp °C

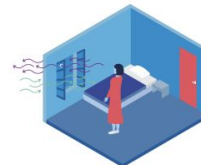
Location permission denied: enter manually the Outside Temperature.

Outside Temp °C

See Result for Single Opening

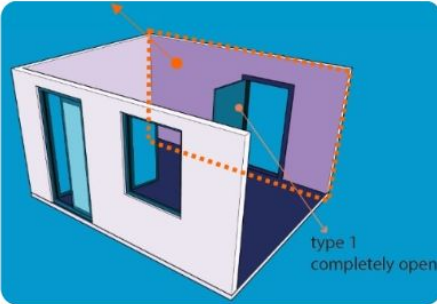


Are there openings available in other side of the room?



Flow of the application

Now for the openings in the **other side of the room**. Usually, these are doors that lead to the inside of the room.



What does "typologies of opening" mean? [Learn more](#)

Door/window characteristics

Number of openings with the same size:

How much do you usually open it?

Enter dimensions

Width meters

Height meters

[Next](#)

Opening Characteristics - Wind Data

Wind Speed

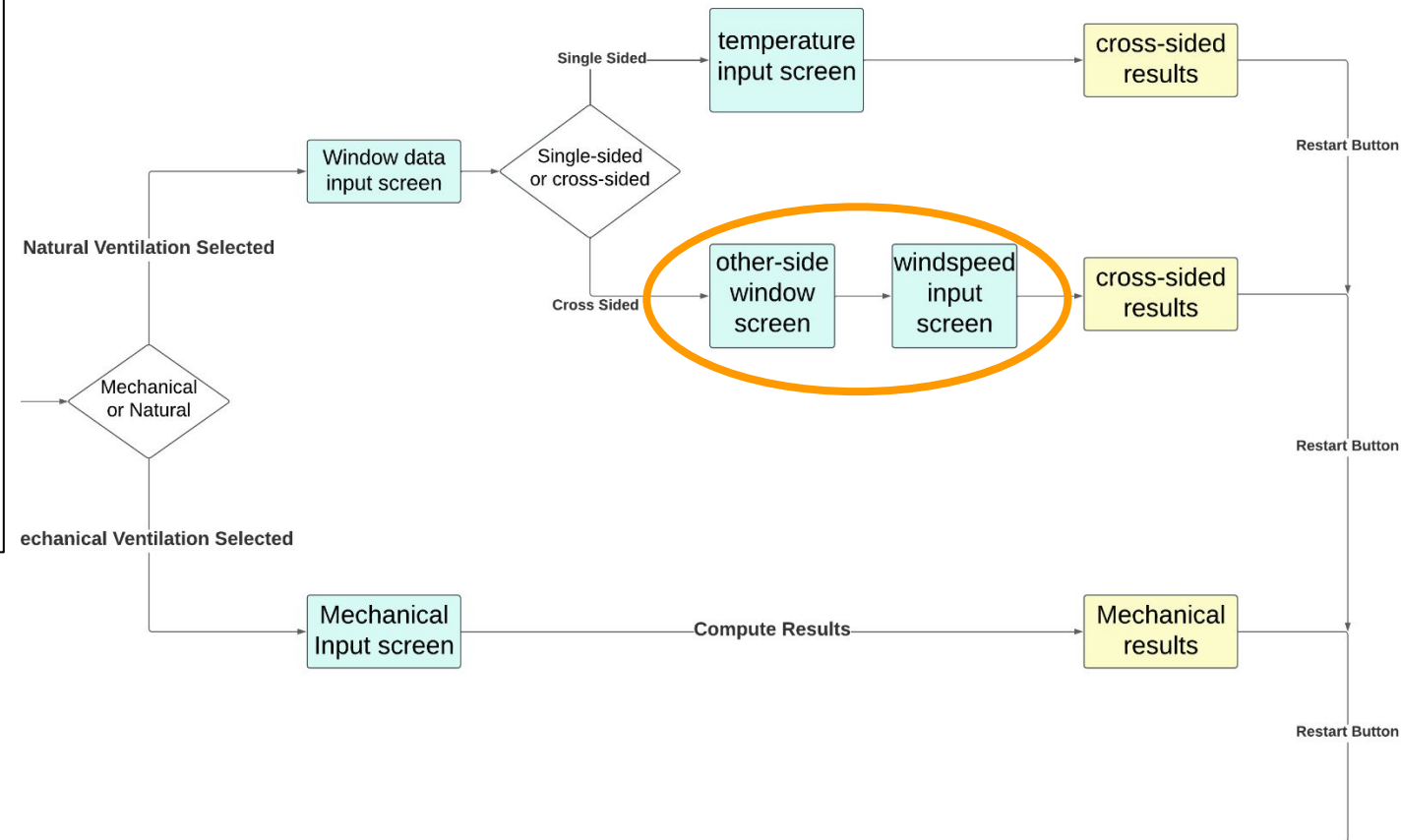
Enter the average wind speed at your building's location. This should be the average value at the building height, away from any obstructions.

① If no data is available, enter "1".

Location permission denied: enter manually the Wind Speed.

Wind Speed m/s

[See Results for Cross Ventilation](#)



Are there openings available in other side of the room?



Flow of the application

9:41

Ventilation Results

Residential settings

The **Air Changes per Hour (ACH)** value shows how many times the air in a room is completely replaced with outdoor air in one hour. This tool calculates the **maximum number of occupants** based on WHO standards.

Note: This is based on ventilation only. You still need to maintain physical distancing.

Actual ventilation estimate

Estimated Ventilation: [number] ACH

WHO Recommendation: [number] l/s/person

Requirement: [number] n. People

How many people do you want to fit in the space?

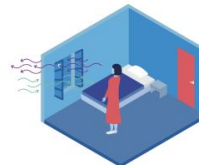
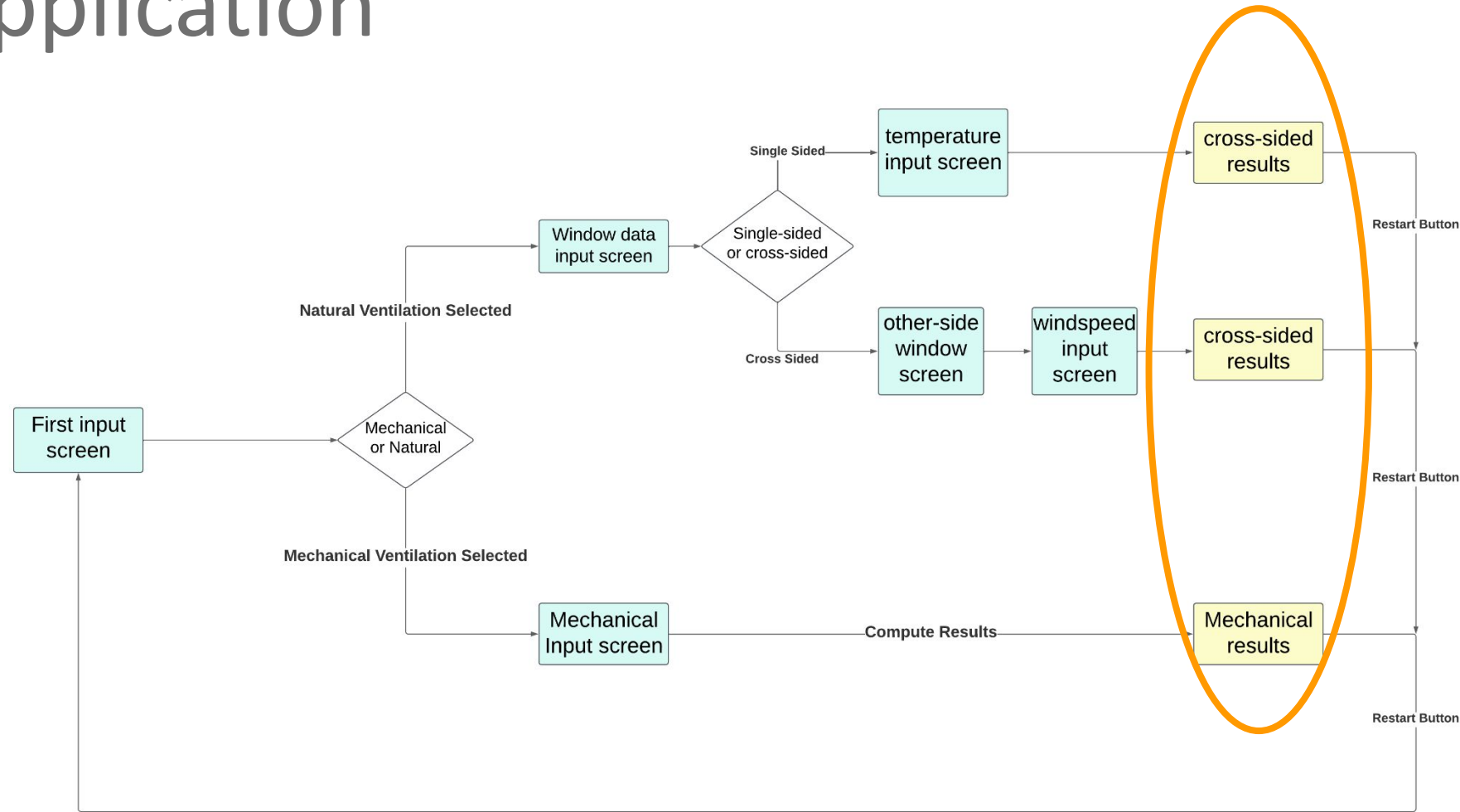
1 people

Note: Physical distancing should still be practiced.

How much you need to improve the ventilation to meet WHO standard.

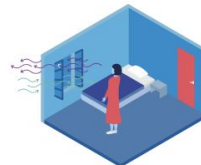
[Calculated Value] Unit ACH

Restart



Development and Functionalities

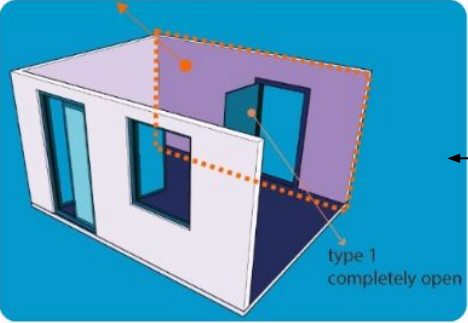
Key pieces of the development of the application



Common Usage Widgets

Opening Characteristics - Other Side

Now for the openings in the **other side of the room**. Usually, these are doors that lead to the inside of the room.



What does "typologies of opening " mean? [Learn more](#)

Door/window characteristics

Number of openings with the same size:

How much do you usually open it?

70%

Enter dimensions

Width

meters

Height

meters

Next

TextEntry()

_buildRichTextContent()

OpeningImage()

_buildHyperLinkText()

TextEntry()

CustomInputWidget()

Slider0To100()

DimensionInputRow()

NextButton()



Routes

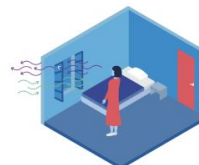
- Overview

- Handles routing.
- Uses a switch statement to return the correct MaterialPageRoute for each screen.
- Handles undefined routes with _errorRoute().

- Key Features

- Dynamic Routing: Efficiently maps URLs to screens.
- Scalability: Easily add new routes.
- Error Handling: Ensures undefined routes don't crash the app.

```
15 class RouteGenerator {
16   static Route generateRoute(RouteSettings settings) {
17     // Getting arguments passed in while calling Navigator.pushNamed
18     // final args = settings.arguments;
19
20     switch (settings.name) {
21       case '/':
22         return MaterialPageRoute(builder: (_) => SplashScreen());
23       case '/second':
24         // Validation of correct data type
25         return MaterialPageRoute(builder: (_) => const Instructions());
26       case '/input_1':
27         return MaterialPageRoute(builder: (_) => Input1());
28       case '/input_nat_2':
29         return MaterialPageRoute(builder: (_) => InputNat2());
30       case '/input_nat_3':
31         return MaterialPageRoute(builder: (_) => InputNat3());
32       case '/nat_wind_speed':
33         return MaterialPageRoute(builder: (_) => NatWindSpeed());
34       case '/nat_temperature':
35         return MaterialPageRoute(builder: (_) => NatTemperature());
36       case '/nat_results_single':
37         return MaterialPageRoute(builder: (_) => NatResultsSingle());
38       case '/nat_results_cross':
39         return MaterialPageRoute(builder: (_) => NatResultsCross());
40       case '/input_mech_2':
41         return MaterialPageRoute(builder: (_) => InputMec2());
42       case '/mec_results':
43         return MaterialPageRoute(builder: (_) => MecResults());
44       case '/input_nat_more_opens':
45         return MaterialPageRoute(builder: (_) => InputNatMoreOpens());
46       // If args is not of the correct type, return an error page.
47       // You can also throw an exception while in development.
48       default:
49         // If there is no such named route in the switch statement, e.g. /third
50         return _errorRoute();
51     }
52   }
53 }
```



State Manager: CalculationState

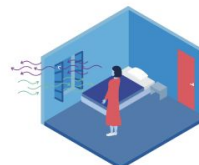
- Overview:

- Manages room settings and ventilation parameters using ChangeNotifier.

- Key Features:

- Real-time updates with ChangeNotifier.
- Uses maps for unit selection and boolean toggles for options.
- Modular methods for dynamic updates to dimensions, rates, and environmental conditions.

```
3 class CalculationState with ChangeNotifier {
4   //3-input_1
5   Map<String, bool> _settingOfInterest = {
6     'Residential/Comercial': true,
7     'Hospital Setting': false
8   };
9
10  String _lenght = '0';
11  String _height = '0';
12  String _width = '0';
13
14  Map<String, bool> _unitLeght = {'meters': true, 'inches': false};
15
16  Map<String, bool> _unitHeight = {'meters': true, 'inches': false};
17
18  Map<String, bool> _unitWidth = {'meters': true, 'inches': false};
19
20  Map<String, bool> _ventType = {'nat': true, 'mec': false};
21
22  Map<String, bool> get settingOfInterest => _settingOfInterest;
23  String get lenght => _lenght;
24  String get height => _height;
25  String get width => _width;
26  Map<String, bool> get unitLeght => _unitLeght;
27  Map<String, bool> get unitHeight => _unitHeight;
28  Map<String, bool> get unitWidth => _unitWidth;
29  Map<String, bool> get ventType => _ventType;
```



OpenWeather API (1/3)



HTTP ... / https://api.openweathermap.org/data/2.5/weather?lat=52.377956&lon=4.897070&appid=42635a3997 &u... Save Share

GET https://api.openweathermap.org/data/2.5/weather?lat=45.464664&lon=9.188540&appid=42635a3997; units=metric Send

Params Authorization Headers (6) Body Scripts Tests Settings Cookies

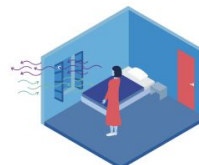
Query Params

<input checked="" type="checkbox"/>	Key	Value	Description	Bulk Edit
<input checked="" type="checkbox"/>	lat	45.464664		
<input checked="" type="checkbox"/>	lon	9.188540		
<input checked="" type="checkbox"/>	appid	42635a3997c7		
<input checked="" type="checkbox"/>	units	metric		
	Key	Value	Description	

Body Cookies Headers (9) Test Results 200 OK 99 ms 836 B Save Response

XML Preview Visualize

```
1 {"coord":{"lon":9.1896,"lat":45.4642},"weather":[{"id":800,"main":"Clear","description":"clear sky","icon":"01n"}],"base":"stations",
  "main":{"temp":2.41,"feels_like":2.41,"temp_min":1.37,"temp_max":3.89,"pressure":1019,"humidity":81,"sea_level":1019,
  "grnd_level":1004},"visibility":10000,"wind":{"speed":0.51,"deg":360},"clouds":{"all":0},"dt":1736620031,"sys":{"type":1,"id":6742,
  "country":"IT","sunrise":1736578875,"sunset":1736611241},"timezone":3600,"id":3173435,"name":"Milan","cod":200}
```



OpenWeather API (2/3):

geolocation.dart

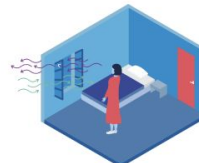
● Overview

- The *geolocation.dart* file provides the *LocationService* class to retrieve the user's current location.

● Key Features

- Service & Permission Checks: ensures location services are active and handles permission requests or denials.
- Accurate Location Retrieval: uses high-accuracy positioning via *Geolocator.getCurrentPosition()*.
- Error Handling: handles disabled services and permission issues with clear exceptions.

```
1  import 'package:geolocator/geolocator.dart';
2
3  class LocationService {
4      Future<Position> getCurrentLocation() async {
5          bool serviceEnabled = await Geolocator.isLocationServiceEnabled();
6          if (!serviceEnabled) {
7              throw Exception('Location services are disabled.');
```



OpenWeather API (3/3)

weather.dart

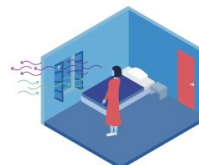
● Overview

- The *weather.dart* file defines the *WeatherService* class to fetch real-time weather data using OpenWeather API based on the user's location.

● Key Features

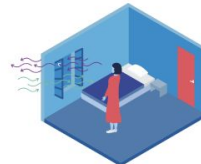
- API Integration: Constructs API calls with latitude, longitude, and API key parameters.
- HTTP Requests: Sends asynchronous GET requests to the OpenWeather API.
- Error Handling: Throws exceptions for non-200 HTTP responses.
- JSON Parsing: Decodes and returns the weather data in a structured format.

```
1  import 'package:geolocator/geolocator.dart';
2  import 'package:http/http.dart' as http;
3  import 'dart:convert';
4  import 'package:ventilation_app/api_key.dart';
5
6  class WeatherService {
7    // final String apiKey = apiKey; // API key
8
9    Future<Map<String, dynamic>> getWeatherData(Position position) async {
10      String apiUrl = 'https://api.openweathermap.org/data/2.5/weather'
11        '?lat=${position.latitude}&lon=${position.longitude}&appid=$apiKey&units=metric';
12
13      final response = await http.get(Uri.parse(apiUrl));
14
15      if (response.statusCode == 200) {
16        return jsonDecode(response.body);
17      } else {
18        throw Exception('Failed to load weather data');
19      }
20    }
21  }
```



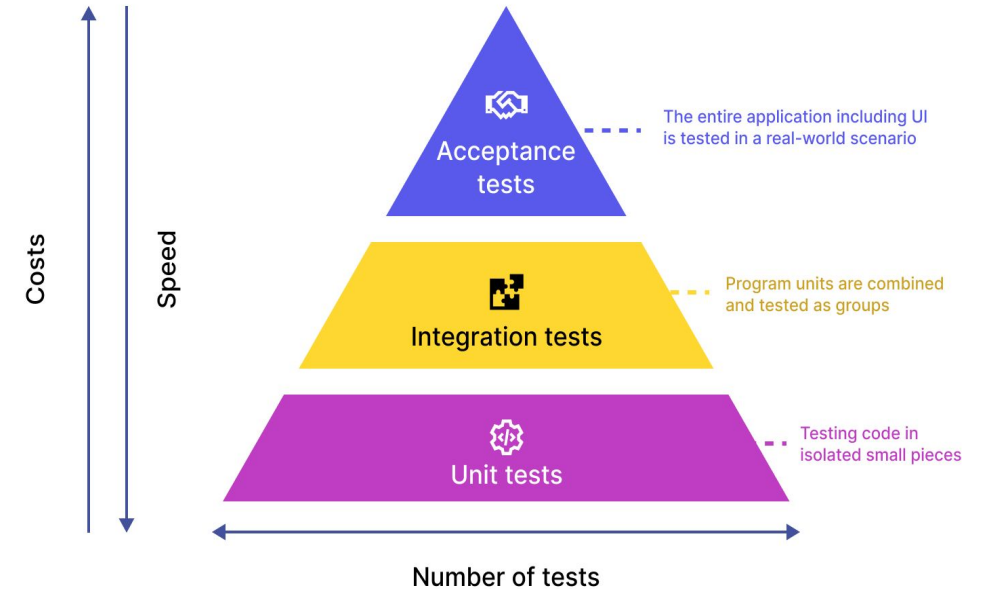
Testing

The assessments made to make sure the requirements were satisfied

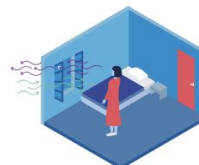


Testing: Unit Tests

- Tests developed using the *flutter_test* library.
- Coverage of 57 Unit Tests, mainly to test each Widget individually
- 2 Integration tests:
 - One to assert that the file path interacts correctly with the system
 - One to test the API procedure



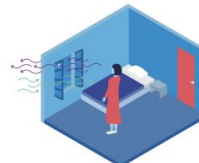
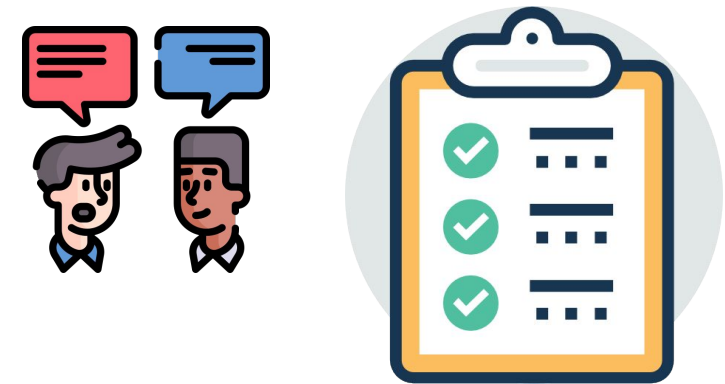
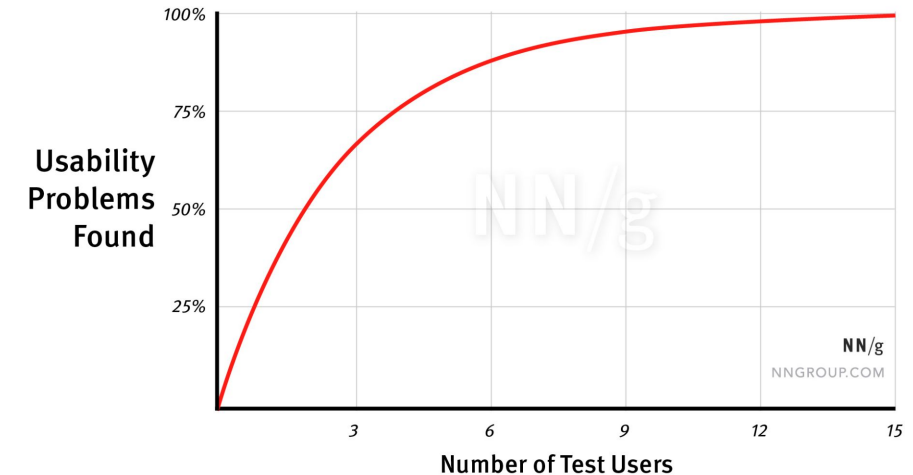
```
00:16 +59: All tests passed!  
→ ventilation_app git:(shadow) flutter
```



Testing: Acceptance Test

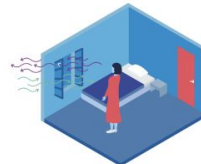
- Based on this work: [Why You Only Need to Test with 5 Users](#) by Jakob Nielsen
- We invited 5 friends to test the app by running the emulator.
- We collected some feedback after each test.
- All 5 managed to get a result and found no major problems.
- Minor problems noted added to “future work”.

5 Users: The Optimal Sample Size for Qualitative Usability Studies



Demonstration

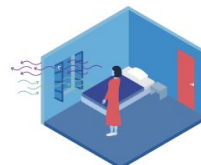
Use Cases



Demonstrations (1/3)

● Scenario 01

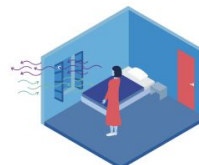
- Giacomo wants to have a dinner reception in his new house, but he is not sure on how many people he can invite to his living room in order to respect WHO recommendations to avoid respiratory diseases.
- Room dimensions (l,h,w):
 - 12.5m x 3.1m x 7.2m
- Side 01:
 - 1 windows of 0.8m x 1.2m
 - No mosquito nets
 - Opened 20%
- Side 02:
 - 2 windows of 2.1m x 1.5m
 - No mosquito nets
 - Opened 60%
- He has enabled the location permissions on his smartphone.



Demonstrations (2/3)

● Scenario 02

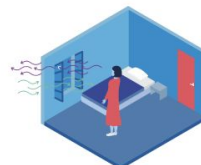
- Maria is responsible for a refugee hospital and she wants to check the natural ventilation of a room to see how many patients fit it in order to control and prevent infections.
- Room dimensions (l,h,w):
 - 8.5m x 4.2m x 5m
- Single Side:
 - 2 windows of 1.5m x 1.5m
 - With mosquito nets
 - Opened 50%
 - 1 window of 2m x 1.5m
 - With mosquito nets
 - Opened 40%
- The temperature inside: 25 °C
- The temperature outside: 30 °C
- She disabled the location service of the app on her tablet.



Demonstrations (3/3)

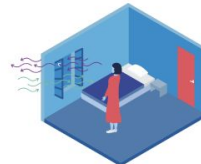
● Scenario 03

- Paolo has just been promoted in his company and his first task as the new Head of Ventilation is to check if the meeting room for 20 people is in agreement with the WHO recommendations to control and prevent respiratory infections.
- Room dimensions (l,h,w):
 - 10.2m x 4.3m x 6.2m
- Mechanical Ventilation, with 2 the specifications:
 - 30 l/s
 - 2 ACH (Air Change per Hour)
- He is using his tablet.



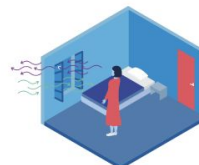
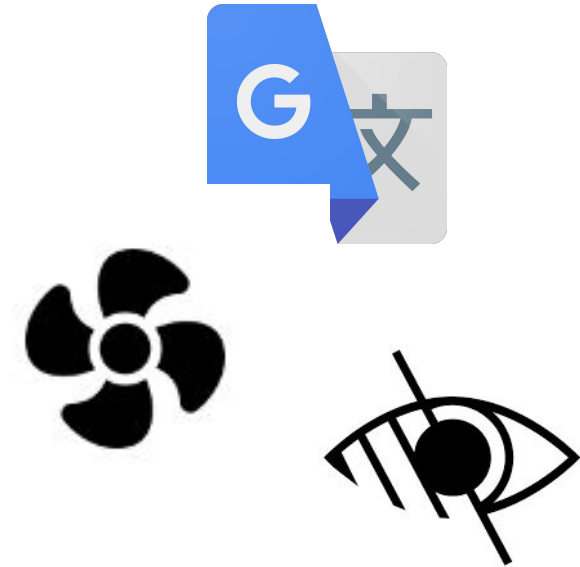
Future Improvements

Improvements based on the interviews taken



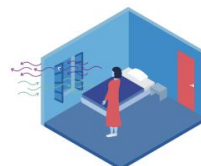
Future Improvements

- Implement a tool to add more languages to the app, making it more universal for non-english speakers
- Develop a better system of visually impaired people
- Tools that help in the measurement process of the window such as guides for estimating
- Add a window and ventilator databases so that people can also chose to select their data by brand



Thanks for your attention!

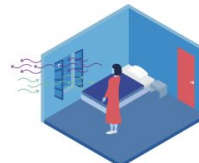
Questions?



Widget Example:

```
46 class DimensionInputRow extends StatefulWidget {
47   final String labelText; // Label for the text field
48   final List<String> dropdownItems; // List of dropdown items
49   final String? initialNumber; // Initial value for the text field
50   final String? initialDropdownValue; // Initial value for the dropdown
51   final Function(String?, String?)? onChanged; // Callback for value changes
52
53   const DimensionInputRow({
54     Key? key, // Key for the widget to interact with the statemanager
55     required this.labelText,
56     required this.dropdownItems,
57     this.initialNumber,
58     this.initialDropdownValue,
59     this.onChanged,
60   }) : super(key: key);
61
62   @override
63   DimensionInputRowState createState() => DimensionInputRowState();
64 }
```

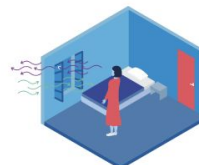
```
66 class DimensionInputRowState extends State<DimensionInputRow> {
67   late TextEditingController
68   |   _textController; // Persistent controller for the text field
69   String? selectedDimensionValue;
70
71   @override
72   void initState() {
73     super.initState();
74     // Initialize the text controller and dropdown value
75     _textController = TextEditingController(text: widget.initialNumber);
76     selectedDimensionValue = widget.initialDropdownValue;
77   }
78
79   @override
80   void dispose() {
81     _textController.dispose(); // Clean up the controller
82     super.dispose();
83   }
```



Widget Example:

```
85 @override
86 Widget build(BuildContext context) {
87   return Row(
88     mainAxisAlignment: MainAxisAlignment.start,
89     children: [
90       SizedBox(
91         width: 150.0,
92         child: TextField(
93           decoration: InputDecoration(
94             border: const OutlineInputBorder(),
95             labelText: widget.labelText,
96           ), // InputDecoration
97           keyboardType: TextInputType.number,
98           controller: _textController, // Use the persistent controller
99           onChanged: (value) {
100             // Notify parent whenever the text changes
101             if (widget.onChanged != null) {
102               widget.onChanged!(value, selectedDimensionValue);
103             }
104           },
105         ), // TextField
106       ), // SizedBox
107       const SizedBox(width: 20.0),
108       Flexible(
```

```
108 Flexible(
109   child: DropdownButtonFormField<String>(
110     value: selectedDimensionValue,
111     items: widget.dropdownItems.map((item) {
112       return DropdownMenuItem(
113         value: item,
114         child: Text(item),
115       ); // DropdownMenuItem
116     }).toList(),
117     onChanged: (value) {
118       setState(() {
119         selectedDimensionValue = value;
120       });
121       // Notify parent whenever the dropdown value changes
122       if (widget.onChanged != null) {
123         widget.onChanged!(_textController.text, selectedDimensionValue);
124       }
125     },
126   ), // DropdownButtonFormField
127 ), // Flexible
128 ],
129 ); // Row
```



State Manager: CalculationState (2/2)

- Highlights:

- Real-time updates with ChangeNotifier.
- Uses maps for unit selection and boolean toggles for options.
- Modular methods for dynamic updates to dimensions, rates, and environmental conditions.

```
31 void updateSetOfInterest(bool res) {
32     _settingOfInterest['Residential/Comercial'] = res;
33     _settingOfInterest['Hospital Setting'] = !res;
34     notifyListeners();
35 }
36
37 void updateRoomDimensions(
38     String h, String l, String w, Map unitH, Map unitL, Map unitW) {
39     _lenght = l;
40     _height = h;
41     _width = w;
42     _unitLeght = unitL.cast<String, bool>();
43     _unitHeight = unitH.cast<String, bool>();
44     _unitWidth = unitW.cast<String, bool>();
45     notifyListeners();
46 }
47
48 void updateVentType(bool nat) {
49     _ventType['nat'] = nat;
50     _ventType['mec'] = !nat;
51     notifyListeners();
52 }
```

