

Functional document

Stop COVID19 Cat

by **opentrends**

April 2020

TABLE OF CONTENTS

Introduction	4
Operational sequence	5
Initial verifications	5
Registration process	6
Autotest process	9
Historical autotest process	12
User profile viewing process	12
Location sending process	13
Reminder sending process	13
Dynamic data	15
Colors and names of the autotest results	15
Dynamic buttons of the Start page	15
Dynamic information of the Start page	16
Screens and components	17
Main menu	17
Action buttons (autotest, etc.)	17
Available buttons	18
Autotest button	18
Visible buttons	19
Start screen	19
Autotest screen	20
Historical screen	20
Profile screen	21
QR reader screen	22
Back-end processes	22
User registration	22

Autotest result calculation algorithm	23
Data model	26
Data model entities	26
USERS table	26
DEVICES table	27
AUTOTEST table	27
ADDITIONAL AUTOTEST INFORMATION table	28
LOCATIONS table (in another instance of DB)	28
Database structure	29
ANNEXES	30
ANNEX I: Dynamic information of the application	30
ANNEX II: Data for calculating the autotest result	34

Introduction

This document describes the functional requirements for the mobile application (app) STOP COVID-19 CAT and the web platform and associated back-ends.

In the face of the COVID-19 pandemic, the Agency for Health Quality and Assessment of Catalonia (AQuAS) has been charged with developing a mobile application that, by operating on the two most popular platforms, IOS and Android, facilitates meeting citizen service needs and at the same time calculating the extent of the disease in any part of the territory.

To that end, the citizen will be offered the opportunity to register and monitor their state of health, in particular with regard to the symptoms of the disease, which in the event of an anomaly will be reported to healthcare devices that will channel the most appropriate response to the situation at hand.

Operational sequence

The operational sequence of different functionalities of the application is described below.

Initial verifications

When a user opens the application, whether for the first time or subsequent times, the following steps will be followed:

The splash screen is displayed and the following verifications are made:

1. Verify if there is an Internet connection. If there is no connection, inform the user. This is a block: they cannot continue with the next steps until there is a connection.
2. Verify if there is a new version of the application. If there is one, direct the user to download it. This is a block: if the new version is not downloaded, they cannot continue with the next steps.
3. Display spinner on the splash.
4. Recover dynamic data from the application.
5. If the user is not registered (the application does not have token-jwt saved), the [Registration process](#) will continue. If they have registered, they continue with the next step.
6. Validate the *token-jwt*. If it is not valid, cancel the user data of the application and go to the registration process.
7. If changes to the user data are received, update the local information. There can be changes in permissions and settings of the application for a certain user.
8. The [Start screen](#) is displayed.

Registration process

This is the process that a user has to follow in order to register on the application. It is the first thing they have to do when they open the application for the first time, and they will not be able to see the Start screen until they have completed it.

There will be buttons to go back to the previous step.

Images that act as a guide to the layout of the elements on the screen in the steps where there is interaction with the user:

<p>STEP 1 Language selection</p>	<p>STEP 3 Legal text</p>	<p>STEP 3b Extended legal text</p>	<p>STEP 4 Welcome</p>	<p>STEP 5 CIP</p>
<p>STEP 5 b Alternative ID</p>	<p>STEP 7 Phone number</p>	<p>STEP 8 Text message with validation code</p>	<p>STEP 11 Validation code</p>	<p>STEP 15 Location and notifications</p>

Steps that will be followed to register a user:

1. **The user selects the language. Languages available: Català, Castellano, English, Français and 中文 (internal codes: ca, es, en, fr and zh).**
2. **The language is set for the next screens according to what the user has chosen.**
3. **The abbreviated text on personal data protection, GDPR, a link to see more information and a button to agree. The user has to agree to the data protection policy in order to continue. When the link to see more information is clicked on, an extended text is displayed, which will be on the application.**
4. **The application's welcome information screen is displayed.**
5. **The user has to enter an identifier: they are first given the opportunity to enter their CIP (health card personal identification code), with the chance to enter other identifiers: ID card (DNI), Foreigner's ID card (NIE) or passport. In this case, the user will indicate what kind of document they have used and the document number.**
6. **The number entered will be verified to ensure it has fewer than 50 characters and the correct format:**
 - a. **CIP: It will be validated by calling a back-end.**
 - b. **DNI, NIE: The format will be validated locally.**
 - c. **Passport: No validation will be done.**

Continuing with the registration process will not be permitted if the format is not valid.

7. **The phone number will be requested and the user will be informed that we will send a confirmation text message.**
8. **We have all the data to do the registration, but the phone number has to be confirmed by text message: a request will be sent to the back-end to send the text message with the verification code. All the registration data will be sent to the request.**

Registration process in the back-end defined in back-end processes – [User registration](#).

9. **If the back-end returns an error in the format of the phone number, a message is displayed informing the user that the format of the phone number is incorrect and they cannot continue.**
10. **If the back-end returns any other error, a generic error message is displayed and they cannot continue. If it returns OK, they continue with the next step.**

11. A screen will be displayed where:

- a. The user is informed that a text message has been sent to verify the phone number.**
- b. The expiration time of the verification code is notified: 30 minutes.**

- c. **There is a text box for the user to enter the verification code received by text message.**
 - d. **There is a button to re-send the text message.**
- 12. The user enters the code and it is verified with back-end to ensure it is correct.**

Back-end sets a limit of 5 validation attempts. Once the limit is exceeded, back-end will not do any more verifications and will always return error.
- 13. If the verification code is correct:**
 - a. **The user is registered in the USERS table (more information in Back-end processes – [User registration](#)).**
 - b. **The device is registered in the DEVICES table.**
 - c. **The user permissions, settings data for this user (QR color and action buttons to which they have access) and their token-jw are received from back-end.**
 - d. **The next step is carried out: ask user to agree to location and notifications.**
- 14. If the verification code is not correct or has expired: a message is displayed to the user and the process does not continue.**
- 15. The user will be asked to agree to share the data of the location and to receive PUSH notification. If they do not agree, the registration process will still continue (access to the user's location will then be requested again when the autotest is done). If they agree:**
 - a. **The user's location is saved at time of registration by calling back-end.**
- 16. The start screen is displayed.**

At any time, depending on the application:

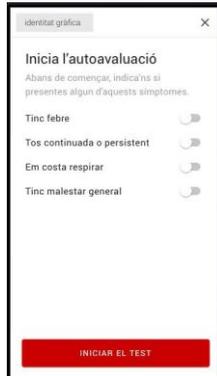
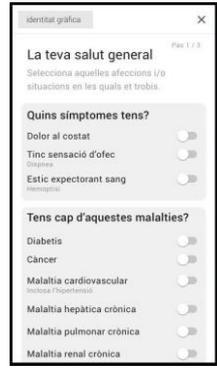
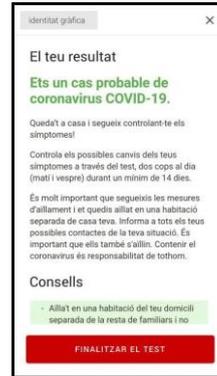
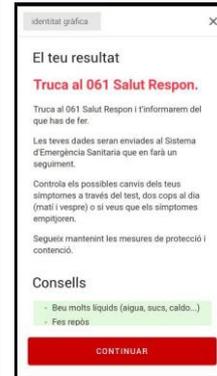
- 17. The asynchronous call is made to obtain and save the user's *token-push* in the database, which is needed to send PUSH notifications to the device.**

Autotest process

This is the process that is followed when a user does an autotest, using the button enabled for this action on the start page.

This section only describes the steps that are exclusive to the autotest. The steps prior to the first autotest screen (requires data verifications) are described in the section [Action buttons](#).

These are the screen that will be displayed during the process:

				
<p>FORM 1</p>	<p>RESULT 0</p>	<p>FORM 2</p>	<p>Example RESULT 3 or 4</p>	<p>Example RESULT 1 or 2</p>
				
<p>FORM 3</p>	<p>THANK YOU SCREEN RESULT 0</p>	<p>THANK YOU SCREEN RESULT 3 or 4</p>	<p>THANK YOU SCREEN RESULT 1 or 2</p>	

1. FORM 1: Symptoms

- a. **The user is asked to check what symptoms they have.**
- b. **If they do not have symptoms:**
 - i. **The autotest is saved with these values in the answers field:**

```
{"page1":{"persistent_cough":false, "breathing_difficulty":false,  
"temperature":false, "general_malaise ":false}}
```

- ii. **The CASE RESULT 0 is displayed.**

- c. **If they have symptoms: when a symptom is checked, a message is displayed on the same screen. When Continue is clicked on, the results are not saved, FORM 2 is displayed.**

2. CASE RESULT 0: The message “You present no symptoms...” is displayed with a button that closes the autotest screen and goes back to the application.**3. FORM 2 (a symptom case):**

- a. **The user enters data on symptoms and medical history, etc. on three consecutive pages. It is compulsory to answer all the questions (except the first page, which can check nothing).**
- b. **The information entered is sent to back-end and the result is received.**
- c. **Depending on the result, the respective result screens will be displayed:**

- i. **Result = 0 → CASE RESULT 0**
- ii. **Result = 1 → CASE RESULT 1**
- iii. **Result = 2 → CASE RESULT 2**
- iv. **etc.**

- d. **If the result is 1 or 2, FORM 3 is displayed after the respective result screen.**

4. CASE RESULT 3 and CASE RESULT 4: a text message similar to “You are a likely case...” is displayed (it can be different depending on the case), with a yellow background and a button that passes onto the THANK YOU SCREEN.**5. CASE RESULT 1 and CASE RESULT 2: a text message similar to “Call 061...” is displayed (it can be different depending on the case), with a red background and Continue button. When the button is clicked on, FORM 3 is displayed.****6. FORM 3 (cases 1 and 2):**

- a. **If you click on the Cancel (Skip) button, the THANK YOU SCREEN is displayed.**

- b. The user enters name, additional phone number and address and clicks on Continue.**
 - i. Maximum length of the three fields: 200 characters.**
- c. This information is sent to back-end.**
- d. The THANK YOU SCREEN is displayed.**

7. THANK YOU SCREEN:

- a. A message of thanks is displayed and, depending on the result of the autotest, the following buttons:**
 - i. RESULT 0: button to go back to the application.**
 - ii. RESULTS 3 and 4: button to share the result and button to go back to the application.**
 - iii. RESULTS 1 and 2: button to share the result and button to go back to the application.**
- b. When the button is clicked on, the flow of the application will be restored, displaying the screen from which the autotest was opened, after carrying out the indicated task (open the share tool or open the telephone keypad to call), if applicable.**

Autotest historical process

When a user does an autotest, the answers and the result of the autotest are saved in the database. Information on the age used to calculate the result is also saved.

A user will be able to consult the result of all the autotests they have done on the [History screen](#).

When they enter this screen, the results saved in the database are obtained and displayed.

User profile viewing process

As described on the [Profile screen](#), there are user data that can be modified by the actual user. These data include age and gender. This information is required to do an autotest, since age is used in the calculation of the result. This is why, in order to start an autotest, the user has to have provided this information beforehand.

There are two ways to provide it: by filling in the information from the Profile screen or filling in the information just before starting the autotest. If the user has no age or gender information, the application will display a screen so that they can enter these data.

The user will be able to modify the age and gender information at any time from the Profile screen.

The user will be able to choose the language in which they want the application to operate. They choose it when registering, the first time they open the application. Once registered, they can modify the Language at any time from the Profile screen. When a user modifies the language, the application passes onto be displayed in the chosen language.

Location sending process

A location sending system will have been prepared so that, when requested, the registered devices send their location to the server.

For the location sending system to function for a registered user, two things are required:

- **The user has to have agreed to receive notifications for the device.**
- **The user has to have agreed to share the location.**

When the server decides to receive the location from a certain device, which has to have agreed to receive notifications, the following steps are followed:

- 1. A notification is sent to the device, which has to have agreed to receive notifications.**
- 2. When the device receives the notification, the application verifies whether the user has agreed to share the location.**
 - a. If they have not agreed, it does nothing.**
 - b. If they have agreed, it sends the device's current location to back-end.**

Reminder sending process

There will be a process for sending notifications to remind a user that they have to do the autotest again.

When a user does an autotest, the data are sent to the server. The server calculates the result of the autotest. If this result is different from 0 (at the moment it can be 1, 2, 3 or 4), then a reminder has to be sent to the user to do the autotest again.

The reminder has to be sent with the following conditions:

- **If the result of the autotest is 1 or 2: it is sent 12 hours later (after whatever time the autotest was done).**
- **If the result of the autotest is 3 or 4: it is sent 24 hours later.**

The periods within which the reminders have to be sent (now 12 and 24 hours) can vary, so it has to be possible for them to be configured without needing to create a new version of the application.

A single reminder will be sent and all the reminders will have the same text, which will be similar to this:

Remember to follow up on your symptoms on STOP COVID19 CAT

The notification will have no title. Depending on the device, the title can be the name of the application. The reminder will be sent in the language set by the user for the device.

A notification will be sent.

If the user does a second autotest after having done an autotest with a result different from 0, then any notification that was scheduled is cancelled. If the result of the second autotest is different from 0, a notification is rescheduled.

Dynamic data

Each time the application is logged into, it will receive the following information:

1. Colors and names for each possible autotest result to be used on the autotest history screen.
2. Possible dynamic buttons of the Start page.
3. Dynamic information of the Start page.

This information will be obtained from a static end-point (Amazon S3).

Full example of the data received in [ANNEX I](#).

Each element is described below.

1. Colors and names of the autotest results

This information will be used in the autotest history to display the list of autotests done by the user.

Initially, the values will be defined for cases 0, 1, 2, 3 and 4.

The information includes the colour associated with each result and the name that is displayed for each result, in the different languages.

Full example in [ANNEX I](#), within the element “historicalLegend”.

2. Dynamic buttons of the Start page

Information will be received on the buttons that the user will be able to see on the Start page. At the moment, information is received on all the available buttons. Later, depending on each user's permissions, a variety of buttons will be displayed.

Each element from the list will have these attributes:

- **id:** Unique identifier of the button
- **icon:** identifier of the icon to display within the button
- **name:** name of the button
- **url:** URL to open when the button is clicked on

Full example in [ANNEX I](#), within the element “homeTest”.

More information on how the logic of the buttons operates on the [Start screen](#).

More details on the permissions in the database section, in the [USERS table](#) field **SETTINGS_APP.**

3. Dynamic information of the Start page

On the Start page there will be a section where information will be displayed in a dynamic manner. This information concerns elements placed one under the other, with a title and an icon. When the element is clicked on, a URL is opened in a new screen.

They will be visible to all users. At the moment there is no plan for different information to be viewed according to user permissions.

Full example in [ANNEX I](#), within the element “homeOther”.

Screens and components

Main menu

There will always be a menu visible on the application with the following options:

1. Button to go to the [Start screen](#).
2. Button to view the autotest history, opens the [History screen](#).
3. Button to access the profile, opens the [Profile screen](#).
4. Button to open the QR code reader, opens the [QR reader screen](#) (planned in the future in subsequent versions and only visible if the user has permission to read QR).

Action buttons (autotest, etc.)

On the start screen there will be a main action button and, depending on the user, there may be secondary actions buttons that will be able to be viewed by clicking next to the main button. A user can have only one main button or have one main button and one or more secondary buttons.

The names of the buttons and the functionality that is carried out when they are clicked on (URL) is defined within the dynamic information that is recovered from the server each time the application is logged into ([Dynamic buttons of the Start page](#)).

The buttons that a user can view depend on the application settings defined for each user saved in the **USERS** table (more information in [USERS table](#) SETTINGS_APP field). By default, all users will only be able to view the Autotest button.

If the user has information on their main button and what their secondary buttons are, the main button will be displayed (which can be different from the Autotest button) and they will be able to view and use the rest of the secondary buttons that they have assigned.

The information received by each button is the following (technical description in [Dynamic buttons of the Start page](#)):

1. The name that is displayed within the button
2. The icon that is displayed within the button

3. The URL that is opened when the button is clicked on

When a user clicks on an action button, the following is done:

1. **It is verified whether the user has agreed to share their location. If they have not agreed, they are asked again to agree. They do not move onto the next step if they do not agree. If the user decides not to share their location, the window where permission to share the location is requested closes and the screen where they were before is displayed. If they agree to share the location or they had already previously agreed, they move onto the next step.**
2. **If the user does not have their age saved in their profile, a window is displayed for them to enter the age and gender information. If they do not enter this information, they do not move onto the next step. If the user does not enter the data and closes the window, the process is cancelled and the screen where they were before is displayed. If they enter the data or if they were already saved, they move onto the next step.**
3. **A window with the URL of the action button opens, which passes the following information:**
 - **The user's *token-jwt* (identifies the user and the device)**
 - **Location**
 - **Age**
 - **The language saved by the user in the profile.**

Available buttons

There will initially only be one available button: the autotest button. This button's parameters and operation are described below.

Autotest button

This button will be the main default button. If a user has no specific information on the buttons that they can view, they will only view this button.

Characteristics:

- **The user must have agreed to share the location**
- **The user must have informed of their age in their profile.**

This is used to do an autotest. When it is clicked on, the [autotest screen](#) opens.

Example of data contained in the initial information of the application for this button in [Dynamic buttons of the Start page](#).

Visible buttons

As has been previously explained, in order to decide which buttons are viewed on the Start screen, this information will be used:

- The information obtained when the application is logged into ([Dynamic buttons of the Start page](#))
- The information on permissions contained in the user's *token-jwt*. This information comes from the database, from the SETTINGS_APP field of the USERS table described in the [USERS Table](#))

In this way, a user will only see the buttons they have assigned in the database. Further, in order to view them, these buttons have to be included in the information obtained when the application is logged into.

Start screen

This is the main screen that is displayed when a registered user logs into the application.

It will display:

- The menu
- The action buttons

More information on the operation of the action buttons in [Action buttons](#).

- Informative chips: the chips received from back-end will be displayed. Each chip will have an icon and a title and, when clicked on, a new window will open with the chip's URL.

The information on the chips is received from back-end. Format defined in [Dynamic information of the Start page](#).

Autotest screen

The autotest screen opens when the [Autotest button](#) is clicked on.

Before opening the screen, the operation indicated in [Action buttons](#) will be followed.

On the open screen, the indicated URL will be displayed and the procedure described in [Autotest process](#) will be followed

History screen

The following elements will be displayed:

1. The result of the last autotest done in the form of highlights (title, subtitle, and date).
2. A button to do an autotest again.
3. A list with the rest of the results of the autotest done by the user, in descending chronological order (the first being the most recent).
 - a. Each element of the list will have a title and a different color, according to the level of severity. Initially, there are these possible levels: 0 = no symptoms; 3 and 4 = mild; 1 and 2 = serious. The colors and names that will be displayed will follow the one that is defined in the information obtained when the application is logged into, described in [Colors and names of the autotest result](#).
 - b. Each element will have information on the date on which the autotest is done.

No area of the screen will be clickable: no additional information on any autotest will be displayed.

Profile screen

The following information will be displayed on this screen:

- 1. QR code: will only be visible if the user has the respective permission (VIEW_QR).**
 - a. If the user has a specific color assigned in the USERS table, the QR code will be displayed in this color. If they have no color assigned, black will be displayed.**
 - b. It is generated by linking together: “profile/” + username**
username: user identifier received in the token-jwt.
 - c. When the QR code is clicked on, the QR code is displayed in a larger size.**

- 2. Modifiable profile data:**
 - Language
 - Age
 - Gender

When the user changes any of these data and saves the changes, this information will be sent to back-end to update the database.

- 3. Non-modifiable profile data:**
 - CIP or NIF/NIE/passport
- 4. Permissions to receive notifications and share location (modifiable)**
 - a. If the receive notifications value is changed and the user decides not to receive notifications, it is notified in back-end in order to modify the DEVICES table of the database.**
- 5. Button to logout**
 - a. The application closes when this is clicked on, erasing the local data. If the user logs onto the application again, they need to register again. If they register with the same CIP or NIF/NIE/passport, and with the same phone number, they will see any autotest history they already had. Back-end is called to notify that the device is no longer active.**
- 6. Legal conditions button: it opens the same text that opens from the button to view the expanded text on the GDPR screen during the registration process.**

QR reader screen

This functionality is planned for future versions and is not within the scope of the current version:

Any users who have permission to read QR (READ_QR) will in the future be able to access this screen via a button from the main menu. The rest of the users will not have this menu option and will not be able to view this screen.

This screen will enable a QR code to be read using the device's camera.

When the QR code has been read, the information that is read will be sent to back-end, along with the user's token-jwt.

Back-end processes

User registration

The application will send the following data to back-end:

- 1. Language**
- 2. Identifying number: the CIP or DNI/NIE/passport (maximum 50 characters)**
- 3. Type of document to which the identifying number corresponds (one of the following values: 1 = CIP, 2 = DNI, 3 = NIE, 4 = passport)**
- 4. Phone number**
- 5. Age (if it has it)**
- 6. Gender (if it has it)**
- 7. Device ID (installation ID)**

Back-end will carry out the following steps:

- 1. Verify that all the required information arrives: language, identifying number (CIP or DNI, etc.), phone number, device ID. If any information is missing, it does not continue; it returns error.**

2. **Generate a verification PIN**
3. **Save the data temporarily in Redis, valid for 30 minutes**
4. **Send the text message with the verification PIN**

Once the user enters the verification code into the application, the following data will be sent to back-end to confirm that the user can be registered:

1. **Temporary token**
2. **Verification PIN**

Back-end will follow these steps:

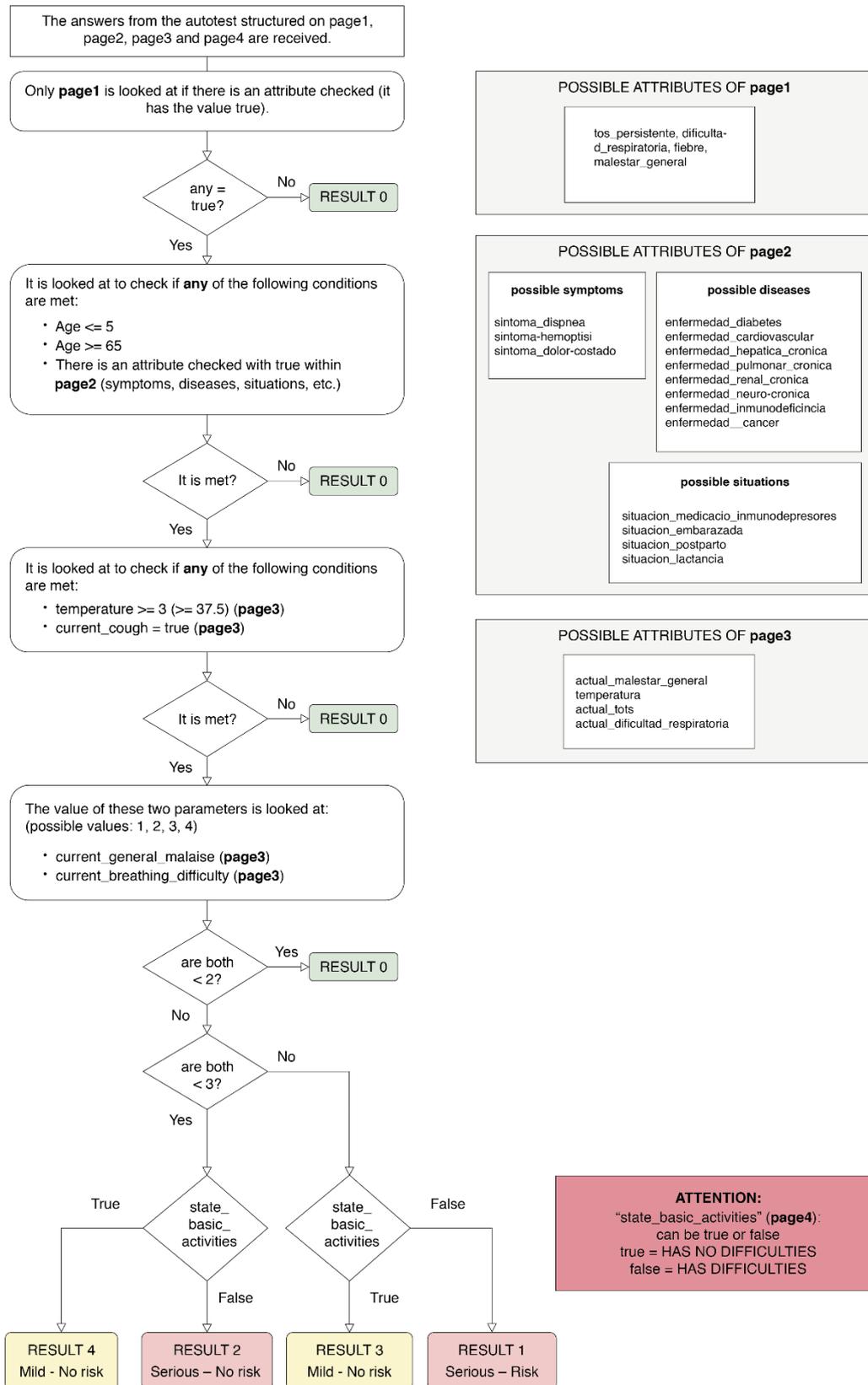
1. **Verify if the temporary token + PIN are correct**
 - a. **If they are incorrect, it returns error**
 - b. **If they are correct, it continues with the next point**
2. **Verify if there is already a registration in the USERS table with the same identifying number and phone number.**
 - a. **If it already exists: it does nothing**
 - b. **If it does not exist:**
 - i. **It saves the data in the USERS table, with the values:**
 1. **LANGUAGE:** value received
 2. **IDENTIFIER:** value received
 3. **IDENTIFIER TYPE**
 4. **PHONE NUMBER:** value received
 5. **PERMISSIONS:** blank (cannot view or read QR)
 6. **APP SETTINGS:** it assigns the available button by default: **AUTOTEST.**
 - ii. **It creates the registration in the DEVICES table**
3. **Return if the PIN is correct**
4. **Return the *token-jwt***

Autotest result calculation algorithm

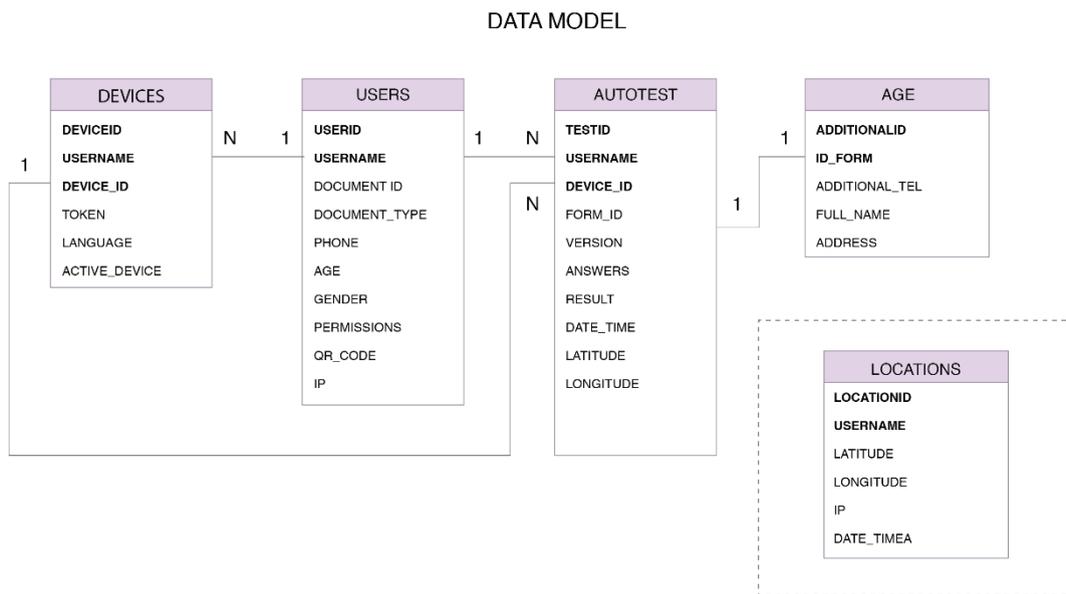
An algorithm will be followed to calculate the result of an autotest. This algorithm represented graphically in a diagram can be seen overleaf.

In ANNEX II an example of the data that are used to do the calculation can be seen.

AUTOTEST RESULT CALCULATION – openVID



Data model



Data model entities

USERS table

- **USERID (auto-generated)**
- **DOCUMENTID: the CIP or DNI/NIE/Passport number (maximum 50 characters)**
- **DOCUMENT_TYPE: indicates what DOCUMENTID corresponds to: 1 = CIP, 2 = DNI, 3 = NIE, 4 = passport**
- **PHONE: phone number (alphanumeric, we accept +34)**
- **AGE: full number. Valid values: from 0 to 200**
- **GENDER: 1 = man, 2 = woman**
- **PERMISSIONS: permissions that the user has, separated by commas. It will initially be blank, they will have no permission.**
 - **Possible permissions:**
 - **VIEW_QR: the user can view their QR code.**
 - **READ_QR: the user can read QR codes.**

- **Example of the field value for a user who can view their QR code and can read QR codes: “VIEW_QR,READ_QR”.**
- **APP_SETTINGS: specific application settings by the user in JSON format.**
 - **Contents:**
 - **QR_COLOR: color of the QR code in HTML format (hexadecimal).**
 - **BUTTON_ORDER: list of identifiers of the buttons to show to the user. This is essential data that all users must have. It will have the default value “AUTOTEST”. The first value makes reference to the main button and the other to the secondary buttons that the user can view.**
 - **Structure of the JSON: the field value will be similar to this example:**

```
{
  "qr_color": "#000000",
  "button_order": ["AUTOTEST", "TEST2", "TEST3"]
}
```
 - **Value that all users will initially have in this field:**

```
{"button_order": ["AUTOTEST"]}
```
- **IP: that of the user at time of registering**
- **DATE_CREATED: date on which the registration is created**
- **DATE_UPDATED: date on which the registration is updated**

DEVICES table

- **DEVICE ID (auto-generated)**
- **USERID (corresponds to the USERID field of the USERS table)**
- **DEVICE_CODE: code of the device or of the installation received**
- **LANGUAGE: possible values: ca es en fr zh**
- **TOKEN: identifier of the PUSH ID assigned from Firebase**
- **ACTIVE_DEVICE: 1 or 0. To validate whether the user had logged out. In these cases, PUSH should not be sent.**
- **DATE_CREATED: date on which the registration is created**
- **DATE_UPDATED: date on which the registration is updated**

AUTOTEST Table

- **IDTEST (auto-generated)**

- **USERID (corresponds to the USERID field of the USERS table)**
- **DEVICEID: ID auto-generated from the devices table**
- **VERSION: the version of the autotest form. Initially: "1.0.0"**
- **ANSWERS: json**
- **RESULT: full number, at the beginning: 0, 1, 2, 3 or 4, but can have other values**
- **LATITUDE: latitude of the location when the autotest is done**
- **LONGITUDE: longitude of the location when the autotest is done**
- **IP: IP of the user when the autotest is done**
- **AGE: age of the user notified when the autotest is done, used to calculate the result**
- **FORM_ID: ID shared with the ADDITIONALAUTOTEST INFORMATION table**
- **DATE_CREATED: date on which the registration is created**
- **DATE_UPDATED: date on which the registration is updated**

ADDITIONAL AUTOTEST INFORMATION table

- **ADDITIONALID: auto-generated (qualify if necessary with FORM_ID functionally is sufficient)**
- **FORM_ID: ID shared with the AUTOTEST table**
- **ADDITIONAL_TEL: value entered by the user into the additional data form**
- **FULL_NAME: value entered by the user into the additional data form**
- **ADDRESS: value entered by the user into the additional data form**
- **DATE_CREATED: date on which the registration is created**
- **DATE_UPDATED: date on which the registration is updated**

LOCATIONS table (in another instance of DB)

- **LOCATIONID (auto-generated)**
- **LATITUDE (latitude): location of the device**
- **LONGITUDE (longitude): location of the device**
- **USERID (username)**
- **DEVICEID: ID auto-generated from the devices table**
- **TIMESTAMP: date on which the registration is created**
- **IP: IP of the device when the location is sent**

Database structure

The image shows a screenshot of a database management tool displaying the structure of four tables: **autotest**, **autotest_additional**, **devices**, and **profile**. Each table's structure is listed with its column names and data types. The **autotest** table has 16 columns, **autotest_additional** has 8 columns, **devices** has 11 columns, and **profile** has 11 columns. Each table view includes an 'Indexes' section at the bottom.

Table Name	Column Name	Data Type	
autotest	id	BIGINT(20)	
	username	VARCHAR(255)	
	form	TEXT	
	form_id	VARCHAR(128)	
	form_version	VARCHAR(45)	
	result	INT(11)	
	age	INT(11)	
	latitude	DOUBLE	
	longitude	DOUBLE	
	ip	VARCHAR(255)	
	device_id	BIGINT(20)	
	created	DATETIME	
	modified	DATETIME	
	autotest_additional	id	BIGINT(20)
		username	VARCHAR(255)
		form_id	VARCHAR(128)
address		VARCHAR(255)	
full_name		VARCHAR(255)	
alternative_phone		VARCHAR(255)	
created		DATETIME	
modified		DATETIME	
devices	id	BIGINT(20)	
	device_uuid	VARCHAR(255)	
	username	VARCHAR(255)	
	language	VARCHAR(10)	
	token	VARCHAR(4096)	
	active	TINYINT(4)	
	created	DATETIME	
	modified	DATETIME	
	profile	id	BIGINT(20)
		username	VARCHAR(255)
		created	DATETIME
modified		DATETIME	
document_id		VARCHAR(255)	
document_id_type		INT(11)	
ip		VARCHAR(255)	
phone		VARCHAR(255)	
age		INT(11)	
gender		INT(11)	
scopes		VARCHAR(255)	

ANNEXES

ANNEX I: dynamic information of the application

Example of JSON downloaded when the application is logged into. It includes all the dynamic information of the application: names and colors associated with each autotest result, possible action buttons and dynamic information of the Start page.

```
{
  "historicalLegend":{
    "0":{
      "color":"#8CBCF9",
      "title":{
        "ca":"No presents símptomes, però no et confiïs!",
        "es":"No presentes síntomas, però no et confiïs!",
        "en":"No presentes síntomas, però no et confiïs!",
        "fr":"No presentes symptômes, però no et confiïs!",
        "zh":"No presentes síntomas, però no et confiïs!"
      },
      "headline":{
        "ca":"",
        "es":"",
        "en":"",
        "fr":"",
        "zh":""
      }
    },
    "1":{
      "color":"#FF6056",
      "title":{
        "ca":"Deixa'ns les teves dades i et trucarem. Queda't a casa.  
Ets un cas probable de coronavirus COVID-19.",
        "es":"Deixa'ns las tevas dades i et trucarem. Queda't a casa.  
Ets un caso probable de coronavirus COVID-19.",
        "en":"Deixa'ns les teves dades i et trucarem. Queda't a casa.  
Ets un case probable of coronavirus COVID-19.",
        "fr":"Deixa'ns les teves données i et truchez. Restez à la maison.  
Ets un cas probable de coronavirus COVID-19.",
        "zh":"Deixa'ns les teves dades i et trucarem. Queda't a casa.  
Ets un cas probable de coronavirus COVID-19."
      },
      "headline":{
        "ca":"",
        "es":"",
        "en":"",
        "fr":"",
        "zh":""
      }
    },
    "2":{
```

```

    "color": "#FF6056",
    "title": {
      "ca": "Truca al 061 Salut Respon i t'informarem del que has de
fer. Ets un cas probable de coronavirus COVID-19.",
      "es": "Truca al 061 Salut Respon i t'informarem del que has de
fer. Ets un cas probable de coronavirus COVID-19.",
      "en": "Truca al 061 Salut Respon i t'informarem del que has de
fer. Ets un cas probable de coronavirus COVID-19.",
      "fr": "Truca al 061 Salut Respon i t'informarem del que has de
fer. Ets un cas probable de coronavirus COVID-19.",
      "zh": "Truca al 061 Salut Respon i t'informarem del que has de
fer. Ets un cas probable de coronavirus COVID-19."
    },
    "headline": {
      "ca": "",
      "es": "",
      "en": "",
      "fr": "",
      "zh": ""
    }
  },
  "3": {
    "color": "#FFB761",
    "title": {
      "ca": "Ets un cas probable de coronavirus COVID-19.",
      "es": "Ets un cas probable de coronavirus COVID-19.",
      "en": "Ets un cas probable de coronavirus COVID-19.",
      "fr": "Ets un cas probable de coronavirus COVID-19.",
      "zh": "Ets un cas probable de coronavirus COVID-19."
    },
    "headline": {
      "ca": "",
      "es": "",
      "en": "",
      "fr": "",
      "zh": ""
    }
  },
  "4": {
    "color": "#FFB761",
    "title": {
      "ca": "Ets un cas probable de coronavirus COVID-19.",
      "es": "Ets un cas probable de coronavirus COVID-19.",
      "en": "Ets un cas probable de coronavirus COVID-19.",
      "fr": "Ets un cas probable de coronavirus COVID-19.",
      "zh": "Ets un cas probable de coronavirus COVID-19."
    },
    "headline": {
      "ca": "",
      "es": "",
      "en": "",
      "fr": "",
      "zh": ""
    }
  }
}

```

```

    },
    "homeTest": [
      {
        "id": "TEST_1",
        "icon": "staroflife",
        "name": "Vull fer el test",

"url": "http://frontend.stopcovid19.com.s3-website-eu-west-1.amazonaws.com"
      },
      {
        "id": "TEST_2",
        "icon": "staroflife",
        "name": "Vull fer el test",

"url": "http://frontend.stopcovid19.com.s3-website-eu-west-1.amazonaws.com"
      },
      {
        "id": "TEST_3",
        "icon": "staroflife",
        "name": "Vull fer el test",

"url": "http://frontend.stopcovid19.com.s3-website-eu-west-1.amazonaws.com"
      }
    ],
    "homeOther": [
      {
        "icon": "mask",
        "title": {
          "ca": "Gestió Emocional",
          "es": "Gestió Emocional.",
          "en": "Gestió Emocional",
          "fr": "Ets un cas probable de coronavirus COVID-19.",
          "zh": "Ets un cas probable de coronavirus COVID-19."
        },
        "url": {
          "ca": "https://www.gestioemocional.cat/",
          "es": "https://www.gestioemocional.cat/",
          "en": "https://www.gestioemocional.cat/",
          "fr": "https://www.gestioemocional.cat/",
          "zh": "https://www.gestioemocional.cat/"
        }
      },
      {
        "icon": "mask",
        "title": {
          "ca": "Consells de salut",
          "es": "Consells de salut",
          "en": "Consells de salut",
          "fr": "Consells de salut",
          "zh": "Consells de salut"
        },
        "url": {
          "ca": "https://canalsalut.gencat.cat/web/.content/_A-Z/C/coronavirus-2019-nc

```

```
ov/material-divulgatiu/coronavirus-aillament-domiciliari-ciutadania-A4.pdf"
,
"es":"https://canalsalut.gencat.cat/web/.content/_A-Z/C/coronavirus-2019-nc
ov/material-divulgatiu/coronavirus-aillament-domiciliari-ciutadania-A4.pdf"
,
"en":"https://canalsalut.gencat.cat/web/.content/_A-Z/C/coronavirus-2019-nc
ov/material-divulgatiu/coronavirus-aillament-domiciliari-ciutadania-A4.pdf"
,
"fr":"https://canalsalut.gencat.cat/web/.content/_A-Z/C/coronavirus-2019-nc
ov/material-divulgatiu/coronavirus-aillament-domiciliari-ciutadania-A4.pdf"
,
"zh":"https://canalsalut.gencat.cat/web/.content/_A-Z/C/coronavirus-2019-nc
ov/material-divulgatiu/coronavirus-aillament-domiciliari-ciutadania-A4.pdf"
    }
  }
]
}
```

ANNEX II: data for calculating the autotest result

An example of the data in JSON format that are sent to back-end when a user does an autotest is shown below. To calculate the autotest result, only the data contained in the “age” (user’s age) and “form” (answers on the autotest form) attributes are used. The rest of the data will be saved in the database, along with the age and the answers on the form.

```
{
  "latitude": 43.3432423423,
  "logitude": 2.4555666666,
  "form_version": "1.0.0",
  "age": 66,
  "form": {
    "page1":{
      "persistent_cough":true,
      "breathing_difficulty":true,
      "temperature":true, "
      general_malaise":true
    },
    "page2":{
      "symptom_dyspnea":false,
      "symptom_hemoptysis":false,
      "symptom_side_pain":false,
      "disease_diabetes":false,
      "disease_cardiovascular":false,
      "disease_liver_chronic":false,
      "disease_lung_chronic":false,
      "disease_kidney_chronic":false,
      "disease_neuro_chronic":false,
      "disease_immunodeficiency":false,
      "disease_cancer":false,
      "situation_medicated_immunodepressors":false,
      "situation_pregnant":false,
      "situation_postpartum":false,
      "situation_lactation":false
    },
    "page3":{
      "temperature":3,
      "current_general_malaise":1,
      "current_breathing_difficulty":1
    },
    "page4":{
      "state_basic_activities ":true
    }
  }
}
```