2021 International Seminar on Electron Devices Design and Production (SED)

The Client-side System for Management of the Smart Home Climate Control System

Ildar Nigmetzyanov
Department of Computer Engineering
Higher school of economics
Moscow, Russia
iinigmetzyanov@edu.hse.ru

Ivan Yatsenko
Department of Computer Engineering
Higher school of economics
Moscow, Russia
iyuyatsenko@edu.hse.ru

Danila Petin
Department of Computer Engineering
Higher school of economics
Moscow, Russia
dmpetin@edu.hse.ru

Alexey Karmanov
Department of Computer Engineering
Higher school of economics
Moscow, Russia
avkarmanov@edu.hse.ru

***Abstract*—The paper considers the development of a client-side indoor climate control system. The system consists of applications that contact the user, as well as a web server for communication between applications and devices, which purpose is managing the microclimate in the room. The mobile app controls the microcontroller using a unique scripting system developed by our project team. To identify the user and their smart home device, Google Mail authentication is added to the app. There are some features and fundamental things used in developing an WEB application. Moreover, in this paper are also shown the differences between WEB app and other parts of the project such as the scenarios creation process, statistics window and so on.**

Keywords—internet of things, smart home, iOS, Android, web app, web server

I. INTRODUCTION

Smart home systems are increasingly important nowadays. There are a huge number of smart devices on the market, both separately and in the form of integrated systems[1]. Many of them are not multifunctional and user-friendly. Often they are limited to simple applications with a few functions. If the application has many features, it becomes not convenient and incomprehensible to the user.

Also, not all smart home devices on the market allow users to control them remotely. They are often limited to Bluetooth operation.

Our application and services for managing smart home devices can solve these problems.

The originality of the project is the uniqueness of the created smart home device management system. A distinctive feature is the modularity of the system. The applications and the microcontroller work independently of each other. They connect through the server using the developed API.

This allows users to easily integrate our system entirely or completely into another smart home system.

The iOS and Android apps, as well as the web app, reflect the datas coming from the devices conveniently. Unlike many smart home management applications[2], our app has access to the data of the system's sensors in the time interval.

The applications have their own unique script creation algorithm, which allows users to set a large number of settings in a simple form. Instead of directly controlling individual devices, the user only selects the desired parameters for the preferred time intervals in the needed rooms.

The scripting system also allows users to control external devices (such as an air conditioner or humidifier). It is possible to set the periods and priority, as well as different modes (for example the silent mode).

II. LITERATURE REVIEW AND RELATED WORK

The app market has a large number of implementations of smart home applications for climate and device control. The following part of the article is an overview of existing solutions for smart home management applications.

DIGMA SmartLife app produced by Nippon Klick Systems LLP. This application takes up 63 MB of memory. The smart home application has the ability to view video from surveillance cameras, the ability to open doors automatically and trigger a scenario for devices for certain actions in the room. But the use of this functionality often leads to errors in the system or the implementation does not meet the expectations of users.

Smart Home Application VI. Produced by mimismart. This application takes up 61 MB of memory. This application has a well-developed functionality and also an interesting design. But this application is difficult to use and maintain the microclimate in the rooms. The only way to regulate the climate is to change the operation of the air conditioner and fan. Since the application does not display the indicators of the microclimate, its use is advisable only if the user is at home.

Our system has a number of advantages over the considered analogues.

- Convenient display of information. The room sensors data are displayed in separate views with explanatory pictures.
- Storing sensor metrics in a database on the server. The user can track the sensor indicators for any day at any time in the form of graphs in the statistics menu. The data is not stored on the device and does not take up space in memory, but is downloaded

978-1-6654-3941-1 @2021 IEEE

from the server if necessary. The collected data is analyzed and errors are detected based on it.

- Control of the device in the form of a scripting system. The user does not need to manually adjust the fan speed, the opening of the valve, turn on and off air conditioners and humidifiers. The user can configure the desired climate and device settings.
- OAuth 2.0 authorization. The user does not need to create another account and remember another password. They can simply use their existing Google account. Moreover, it eliminates the need to store user passwords on the server, which significantly increases the security of the system;
- Availability. The apps are available on Apple and Android devices, and there is also a web version that is accessible from the browser.

There are different ways of IoT integration between device and client application. Shiu Kumar and Seong Ro Lee (2014) suggest the easiest way of integration between device and client app. In their opinion, using a Bluetooth module that is built in a microcontroller is a fast and cost-effective way to make an integration. The absence of a database and a server allows to create a low-effort elementary connection. [1]

Rajeev Piyare (2013) describes a more advanced approach than just Bluetooth connection. He suggests using TCP/IP connection instead of it in order to make smart home systems available via Wi-Fi or 3G/4G. This variant is close to a regular web applications structure – work through HTTPs with REST API[3] so it is available not only for smartphones but for computers too.

However, in contrast to this structure, the author claims that it is better not to use a remote HTTP server, one for all system kits but a low-resource server built into an embedded system. In his opinion, using the application of such a server based on a microcontroller could be much cheaper and more secure than a remote server. [4]

In contrast to Rajeev Piyare, the authors of the article “Smart Home Implementation Based on Internet and WiFi Technology” suggest using a remote server in the system architecture. In their opinion, using Wi-Fi controllers will not be more expensive than using a separate controller as a server. Moreover, it eliminates dynamic IP problems, NAT and other home network problems.

In their solution, the database and REST API is located on a remote server while program logic still remains on the device. The authors consider such architecture best for the production version of the smart home systems with big data databases. [5]

Speaking about databases, authors of the article “Comparing Relational and NoSQL Databases for carrying IoT data” provide results of a study on performance of three database management systems (MySQL, PostgreSQL, MongoDB) in work with big data in IoT sphere.

According to test results with different data types (CLOB), different number of rows (10000 to 1000000) and with different operations (SELECT, INSERT) authors came to the conclusion that the worst candidate for big data in IoT is PostgreSQL. MongoDB shows performance slightly higher than MySQL but MongoDB is not a relational database and so could be not suitable for every project. [6]

If we look through analogs of our smart home system we will see that most manufacturers try to use remote server architecture. For example, INSYTE and Tion MagicAir provide smart home systems in basic configuration only with connection through cloud (same as remote) server. None of them use Bluetooth modules or home servers.

However, INSYTE system offer outer Bluetooth and infrared remote control modules that can work simultaneously with cloud servers. Nevertheless, all databases of both INSYTE and TION systems are stored at cloud servers so Bluetooth connection could not be a major type of IoT integration.

Home servers based on microcontrollers are not considered by Tion, INSYTE and other major manufacturers at all

In view of the above, we can see a general trend of IoT systems backend architecture. All serious projects tend to use architecture with a remote server as described in the article “Smart Home Implementation Based on Internet and WiFi Technology” and combining it with local Bluetooth connection. Thus this is the type of architecture we will use in our project.

Speaking about databases, we realized that there is no single right answer. It should be selected depending on the task and skills of the developer. However, based on the article “Comparing Relational and NoSQL Databases for carrying IoT data”, we can conclude that mySQL is the most suitable DBMS for our project.

III. System Architecture

The architecture of the system based on conclusions from the literature review with structure in Fig. 1 has been developed in the course of this work.

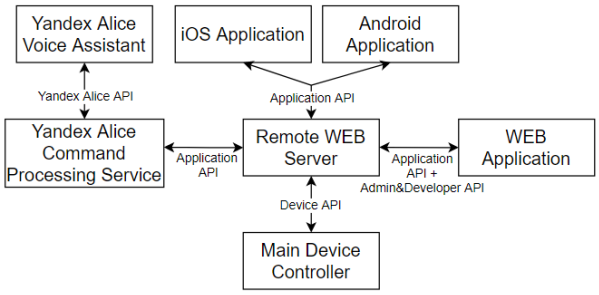

Fig. 1. System Architecture.

The main component of the system architecture is a remote web server. It is a point of the HTTP request API connection between all other components and the controlled device.

The other components of the system are front-side applications which are interacting with the user. iOS and Android applications (see “Mobile apps”) are the classic implementation of this containing all information about the controlled system and ways of controlling (see “Scenario system”).

Web application implements all the functions from mobile apps but also adds features for administrators and developers with log and analytics information and advanced control panel (see “Web application”).

Yandex Alice Voice assistant is integrated into our system application that implements basic features that can be asked by voice (see “Yandex Alice Voice assistant”).

Mobile apps are available for users as apk files/test apps and as an applications in Google Play/AppStore in a perspective. Other components are deployed on a public server in order to make them available from the Internet. In the figure 3 showed a server deployment scheme of web connected components.

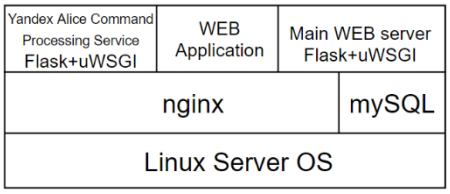

Fig. 2. System components deployment scheme.

IV. Scenario System

Scenario system is created in order to simplify management of smart home climate control devices. With this, the user does not have to manually manage fan speed, air conditioner temperature and parameters of other device components manually. The user should configure only the climate parameters (temperature, humidity and CO2 concentration) that he wants to have in his apartment. Moreover, he can assign the desired parameters for a specific day of the week, a specific time and a specific room. Further, the algorithms and device control implemented in the hardware part of the project will create the desired microclimate for the user.

Since the scenario, in which several indicators are configured for any time, each day of the week and for each room, is a complex structure, we decided to use a step-by-step system for creating the scenario. A block diagram of scenario creation can be seen in Fig. 3. Below is its detailed description.

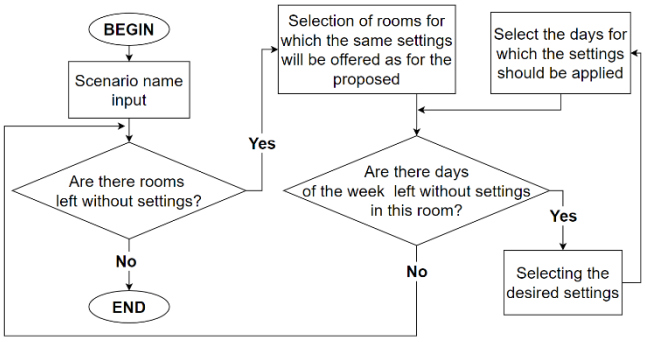

Fig. 3. Algorithm of scenario creating.

The first step in creating a script is to enter its name. This is done the only time in the entire process.

Next, the user is shown the name of the room for which he will have to set a desired settings, as well as select other rooms for which such settings will be applied. The user must go through this step at least once, but if the first time he does not select all the rooms, he will have to return to this step to set the settings for the rest of the rooms.

After selecting a room, the user is prompted to select the days of the week and settings for them. The user also goes through these steps at least once, but if the first time the user has not selected all days of the week, he will have to return to these steps to select the rest of the rooms. For the convenience of the user, it is possible to select several days of the week - the same settings will be applied to them.

After the user specifies the settings for each day of the week for each room, a new script will be created that will be available for use.

A simplified example of the created scenario can be seen in a Fig. 4.

	Room 1	Room 2 Room 3
Monday	**7:00** : temperature - 20° humidity - 60% CO2 - 600ppm **9:00** : temperature - ANY humidity - ANY CO2 - ANY **18:00** : temperature - 20° humidity - 60% CO2 - 600ppm	**7:00** : temperature - 20° humidity - 60% CO2 - 600ppm
Tuesday		
Wednesday		
Thursday		
Friday		
Saturday	**7:00** : temperature - 20° humidity - 60% CO2 - 600ppm	
Sunday		

Fig. 4. Simplified scenario example.

V. Web Server

A web server (Fig. 5) was developed for integration between client applications and the smart home device. The software was developed on the Flask microframework[7]. The mySQL DBMS was used to develop the database.

The server is deployed in configuration with uWSGI and nginx proxy server. The interaction between Flask and mysql is carried out using the SQLAlchemy library. Data exchange between the server and client applications occurs via http.

Http requests allow access to three APIs - Devices API (interaction device-server), Application API (interaction any client app - server) and Admin and Development API (interaction admin dashboard - server)

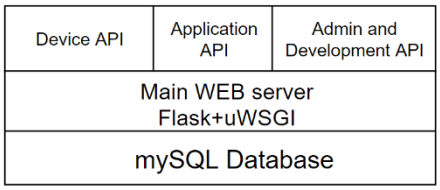

Fig. 5. Web server structure.

A. *Device API*

Allows to request data about a scenario, send sensor readings to the server, synchronize time, and several other features. Sending data to the device is implemented in several formats (JSON, XML, String) so that the microcontroller programmer can choose the data format that suits him, depending on the capabilities of the controller.

B. *Application API*

Allows to query the current indicators of sensors, statistics for a specific date, create and modify scenarios,

make various settings. In most cases, data is sent to the client device either in JSON format or as a string, depending on the complexity of the response structure.

C. Admin and Development API

It is an extension for the Application API that allows to query all the statistics, logging and analytics data. Available only for administrators and developers of the system.

D. Database

Database structure presented on a Fig. 6. Below is a brief description of its tables.

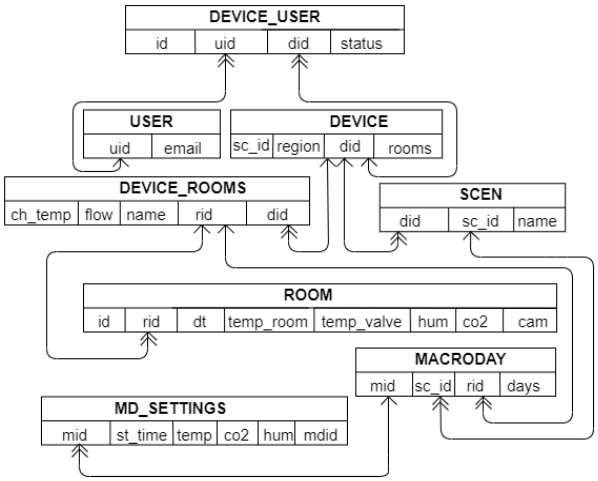

Fig. 6. Database structure.

The *User* table is responsible for storing user data. Note that thanks to OAuth 2.0 authorization, passwords are not stored on the server, which makes the system safer.

The *Device* table is responsible for storing device data. *Device_User* associates a user with one device. *Device_Rooms* represent rooms for a single device. The *Room* table stores sensor readings from a device for one room.

The *Scen*, Macroday, and *Md_Settings* tables store script settings. *Scen* - basic information on the scenario, *Macroday* - connection of rooms and configured days, *Md_Settings* - directly settings of the scenario.

VI. Mobile Apps

We have developed Android and iOS applications for managing smart home systems. Almost every person has a mobile device, so users will be able to control the climate without spending money on a new smart control device.

A. Authorization

To identify the user, the Google Authorization feature has been added to the app [8], [9], [10], [11]. Unique token, which was received during registration, will be used for all subsequent requests to the server. After first authorization the users account would be saved and the next launching application will automatically open the main window.

B. Main Window

The home screen shows all the rooms that our smart home device is connected to (Fig. 7). The name of the room is written in bold in each individual block. Each room has a name, temperature, humidity, CO2 content in the air and the number of people in the room. Room data are displayed on special cells, which open a window with detailed information about the room that the user chose. Mobile apps download room data from the server. Swipes allow the user to move to the next room. When swiping to the right, the user switches to the window of the first room. At the top of the screen there is a button that opens the menu list and white text with the name of the window.

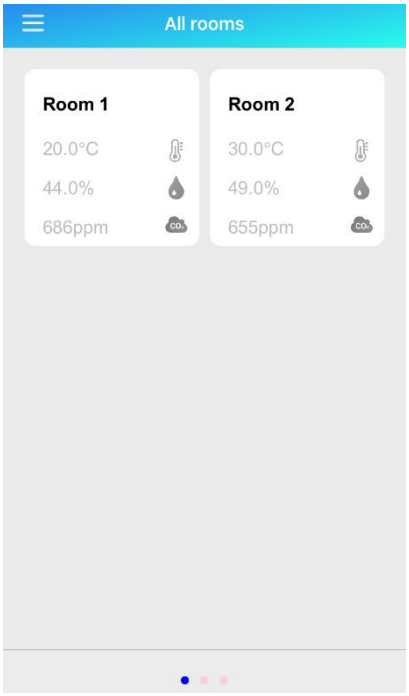

Fig. 7. Main window.

C. Detailed Room Data Window

The room data window shows more detailed information then the main window (Fig. 8). In addition to the information in the main window, data about the user-defined operating mode of the device is appended to the window of the selected room. Also users can choose ventilate and temperature change options by clicking on plus and minus on white button and blue button for ventilation. Mobile apps download room data from the server. Swipes allow the user to move to the next or previous room. When swiping to the right, the user switches to the window of the next room, when swiping to the left, the user switches to the window of the previous room. When the user views the last room in the list, nothing happens when swiping to the right, and when the user views the first room in the list, the main window opens when swiping to the left. At the top of the screen, as in Fig. 7, there is a button that opens the menu list and white text with the name of the current room. Then, there are 4 white separate blocks with the room microclimate information: temperature, CO_2 level, humidity and number of people.

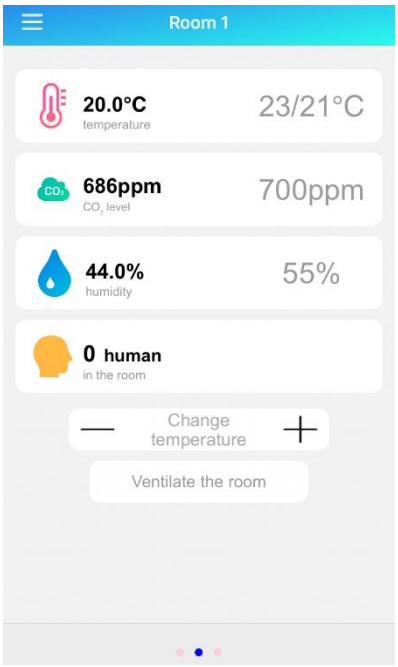

Fig. 8. Detailed room data window.

D. Menu List

The menu list [12] could be opened in the main and detailed room data windows by clicking on the button at the left top of the screen. The title of the menu has the user mail and the exit button, which returns the user to the authorization menu. The body consists of the buttons that open other windows of the application.

E. Window for Selecting Scenarios

The window for selecting scenarios could be opened from the menu list or by swiping up at the main and detailed room data windows. The various scenarios are downloaded from the server and displayed in the cells of layout. These cells consist of the names of the scenario and buttons. By clicking on these buttons the user can choose which mode of the microcontroller will operate. The right top of the screen has the button, which opens the system for creating new scenarios.

F. System for Creation New Scenarios

System for creating new scenarios consists of four windows. The first window is intended for assigning a name to the script. This layout has a text input field, the button, which cancels the script creation and the button which opens the next window if the text field is not empty. The second layout required for selecting rooms, which will be applied in the script. Also this window has a separate interface where all rooms are displayed and the user can choose needed placement. The selected rooms can be grouped as the user desires. The third window opens when the client clicks on a cell with room names. This layout has the same functionality, but instead of rooms the days of the weak are shown. The next window allows clients to set settings for the chosen days and selected rooms (Fig. 9). At the top of the screen, there are 2 buttons: on the left side there is a button “back”, which returns the users to the previous window, and on the right side there is a button “save”, which saves the created scenario. Below these buttons is an explanatory text “Add the desired settings”. Users can pick time, humidity, temperature and co2 amount. Moreover, all selected devices are listed to the right of all these icons with explanatory text. Also the client can select the silent mode of the device or the operation when the user is not at home. In addition the client can select devices, which would be used when the microcontroller is running. When all settings assigned, the client could click on the save button at the bottom of the screen and view with all data specified will appear in the separated cell. These cells can be edited and saved. Users can switch through all windows and exit to the main or scenarios window. All created data is sent to the server and the user can continue editing scenarios at any time.

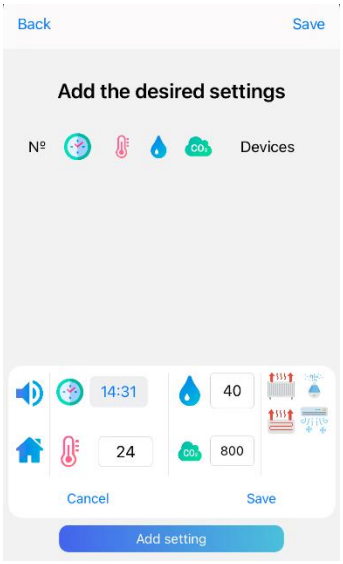

Fig. 9. Set the settings window.

VII. Web Application

Although the appearance of the WEB application differs from the mobile apps, the interface is designed in a single style. Talking about functionality, there is also authorization using a token, the main window with the display of sensor indicators such as temperature, humidity etc. and some ways to configure and manage the system. The main difference is that all the detailed information is displayed in separate blocks for each room on the same screen without the need for swipes. The same could be said about all the other windows of application. Moreover, if the user doesn't have a mobile app on their smartphone, they may also open the site from their phone in any browser and do all the necessary things.

Another major difference between Mobile and Web applications is availability of the developer and admin panel. This panel allows advanced features as logging data, analytics data (can help to detect deviations in works of devices) and features for system setup and adjustment.

Web applications interacts with web server (both application and admin API) using XMLHttpRequest (http requests in JavaScript)

VIII. Yandex Alice Voice Assistant

Yandex Alice Voice assistant is integrated into our system application that implements basic features that can be asked by voice. As presented in Fig. 10 Yandex Alice Voice assistant sends user voice command to the Yandex Alice

Command Processing Service which analyses voice command, due to its request information from remote web server and response voice answer back to Voice assistant.

Implemented voice commands allow to make a quick airing, changing of scenario temperature and some other features.

Interaction between voice assistant and processing service is carried out using http requests and Yandex API[13]. Processing service interacts with the web server using http requests and application API. Schema of interaction is presented in Fig. 10.

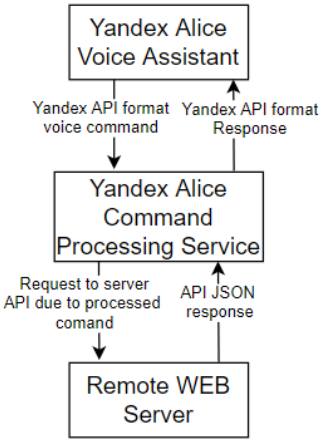

Fig 10. Yandex Alice interaction.

IX. CONCLUSION

In the course of the work, a client-side indoor climate control system was developed. The following goals were achieved:

- Developed mobile application on Android;
- Developed mobile app for iOS;
- Developed web application;
- Developed web server;
- Developed proprietary application programming interfaces (APIs) for client-server and server-managed device interaction;
- The application interacts with web servers, which interacts directly with the device. This device is developed by our colleagues using the API;
- Developed our own device management system in the form of scenarios and algorithms for creating them;
- Alice Yandex voice assistant was integrated.

As a result, we developed a unique client-side smart home device management system. In the future the system can be improved by adding new features (e.g. google home, apple home kit) or it can be integrated into other smart home systems.

REFERENCES

[1] M. Hasan, M. H. Anik, and S. Islam, “Microcontroller Based Smart Home System with Enhanced Appliance Switching Capacity,” *ITT 2018 - Inf. Technol. Trends Emerg. Technol. Artif. Intell.*, pp. 364–367, 2019, doi: 10.1109/CTIT.2018.8649518.

[2] S. Kumar and S. R. Lee, “Android based smart home system with control via Bluetooth and internet connectivity,” 2014, doi: 10.1109/ISCE.2014.6884302.

[3] Masse M., REST API Design Rulebook: O’Reilly Media, 2012 – pp. 2-10

[4] Rajeev Piyare, Seong Ro Lee “Smart Home-Control and Monitoring System Using Smart Phone.” 2013, 1st International Conference on Convergence and its Application (ICCA)

[5] YAN Wenbo, WANG Quanyu, GAO Zhenwei “Smart Home Implementation Based on Internet and WiFi Technology.” 2015

[6] Sotirios Kontogiannis , Christodoulos Asiminidis , George Kokkonis, “Comparing Relational and NoSQL Databases for carrying IoT data.”, 2019, The Journal of Scientific and Engineering Research

[7] Grienberg M., Flask Web Development: O’Reilly Media, 2018

[8] T. Hang, L. Thao, L. Hieu, N. Khoe, T. Thuong, and V. Uyen, “GOOGLE SIGN-IN FOR IOS,” vol. 2, no. 16, pp. 23–25, 2017.

[9] Q. Ye, G. Bai, K. Wang, and J. S. Dong, “Formal Analysis of a Single Sign-On Protocol Implementation for Android,” in Proceedings of the IEEE International Conference on Engineering of Complex Computer Systems, ICECCS, Jan. 2016, vol. 2016-January, pp. 90–99, doi: 10.1109/ICECCS.2015.20.

[10] S. Holla and M. M. Katti, “Android based mobile application development and its security,” Int. J. Comput. Trends Technol., vol. 3, no. 3, p. 5, 2012, doi: 10.1080/10304312.2012.706462.

[11] S. Kumar, “Ubiquitous Smart Home System Using Android Application,” *Int. J. Comput. Networks Commun.*, vol. 6, no. 1, pp. 33–43, Feb. 2014, doi: 10.5121/ijcnc.2014.6103.

[12] I. C. Morgado and A. C. R. Paiva, “Test patterns for android mobile applications,” in ACM International Conference Proceeding Series, Jul. 2015, vol. 08-12-July-2015, pp. 1–7, doi: 10.1145/2855321.2855354.

[13] Alice Voice Assistant API (Yandex) - Alice's Skills. Alice. Dialogues (yandex.ru)