“Creation of “Internet of Things (IoT)” laboratory, IoT demo stand, and training” Grant report

Organizing IoT classes for teenagers using Arduino and Raspberry Pi controllers was a challenging task. We defined the learning objectives, that guided us in selecting the appropriate curriculum, creating lesson plans, and choosing the necessary hardware and software. We had to consider the level of technical expertise and interests of our students to make sure the objectives are achievable.

Arduino and Raspberry Pi are popular platforms for building IoT projects. Arduino is simpler and more affordable, while Raspberry Pi is more powerful and versatile. The chosen platforms best suited to our objectives and students’ skill levels.

A curriculum was developed: A well-structured curriculum helped students understand and learn the necessary skills to build IoT projects. The courses were divided into modules that covered the fundamental concepts and tools. They included;

Theoretical course – This course helped students to understand the value of using the Internet of Things for businesses and how they can help through smart product and data collection. Participants got knowledge about IOT architecture, communication systems, IOT platforms, and visualization solutions. We also trained out the physical basis of the operation of individual electronic components and components from the ecosystem such as

- temperature, humidity, illumination, distance, motion sensors,
- various executing devices such as buttons, switches, LEDs, motors
- LCD display devices and multi-digit indicators

Interesting and colorful presentations were prepared for all theoretical classes.

Basic Practical course – this course helped to get basic knowledge and experience in smart electronics design. During the course, it was shown, how to work with different development boards such as Arduino, ESP8266, ESP32, and Raspberry Pi as well as connect with them a lot of sensors and actuators. During the classes, a detailed analysis of both the methods of connecting devices and the program code was carried out.

Advanced practical course – During the course, students got acquainted with the basics of the functioning of the Internet, organizing network connections both in local and global networks, building web servers and various cloud applications. Students studied the components of IoT systems, the MQTT brokers, as well as other components of IoT systems.

Students also:

- learned how to connect to a cloud server and configure it. They received information about the MQTT broker and web server setup,
- learned how to write MQTT broker publishing and subscription queries, as well as web server applications, so that users can activate smart devices by pressing the appropriate buttons in the remote computer or phone browser,
- learned how to create applications for smart devices.

During the course, students developed real IoT systems that can be used to control equipment over the phone at home or in the office.

Technical support – In order to improve students’ skills and experience in the IoT field, the most of trained students undergo internships. The internship program aimed to optimally balance students’ knowledge in the field of hardware and software. Selected participants during this internship:

- worked on creating training articles on implemented IoT projects and solutions
- had the opportunity to participate in the implementation of state-of-the-art IoT solutions, such as lighting automation, and automatic temperature control in Smart Home and Smart GreenHouse projects.

We also included practical projects that students can work on to apply what they've learned. Online resources like tutorials, and videos, that supplemented the curriculum, as well as Zoom, Tinkercad, and Kahoot platforms, were also used. Considering the high rates of development of the Internet of Things, we created an Internet of Things lab, where high school students got acquainted with the fields of application of the Internet of Things, implemented projects, as well as got the opportunity to gain theoretical and practical knowledge of the Internet of Things. In the laboratory students studied to:

- Develop software for microcomputers that will provide "smart stuff" functionality
- Create sensors and systems of actuators based on microcontrollers ensuring their interaction through the network
- Understand the specifics of device interaction protocols and select the protocol that suits the participant's problem
- Analyze the architecture of IoT solutions and hardware-software systems
- Orient in IoT systems that can solve the problems of production, transport, health, and energy systems
- Build prototypes of IoT solutions using different cloud IoT platforms from the hardware level to client application building.

The course program included both theoretical and practical work.

Materials and resources, necessary for students, including hardware components like sensors, actuators, and controllers, as well as cables and tools like breadboards and jumper wires were provided, as well as other supporting materials: code samples, documentation, and project ideas.

Classes provided to students included the following lectures:

- Internet of Things, application in various fields: industry, household, agriculture, cargo transportation, healthcare, ecology, industry
- SmartHome and SmartCity solutions
- Problems of technical implementation of Internet items
- Ways to provide Internet connection
- Adaptive self-regulating networks, mesh solutions
- Possibilities of using Arduino, ESP8266, ESP32 and Raspberry equipment in IoT solutions
- HTTP and MQTT protocols
- IoT sensors and performing equipment
- IoT platforms
- Equipment identification on IoT platforms.
- Software update on IoT platforms
- Cloud IoT solutions IBM Bluemix, Microsoft Azure, AWS
- Ensuring the security of IoT systems.

The lab also offered additional introductory courses on Internet-related items, such as

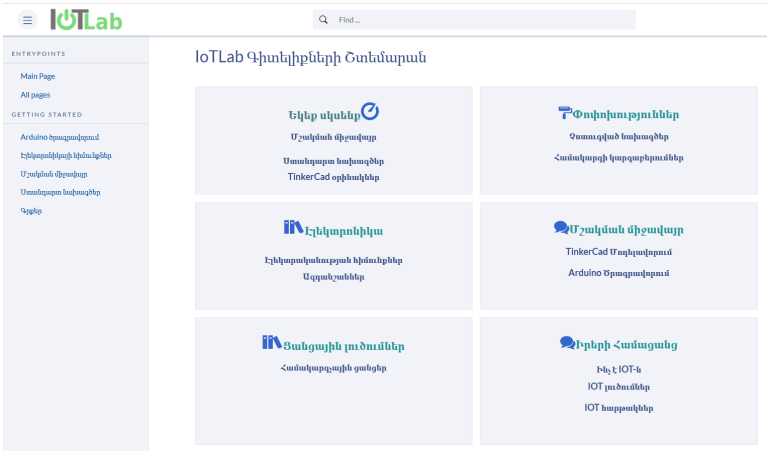
- PCB design and embedded systems development.
- computer networks and monitoring systems.

- robotics, machine learning and artificial intelligence.
- marketing of Internet of Things and business solutions for Internet of Things

The laboratory is equipped with the following equipment:
- Computers
- Soldering station
- Microcontrollers
- Arduino Uno
- Arduino Nano
- ESP8266ESP32
- Microcomputers
- Raspberry Pi
- Sensors: temperature, humidity, enlightenment, color, movement, dust, vibration, CO2
- Engines: step-by-step, power converters, relays, keys

During classes, students were guided through the process of building their IoT projects. Classes started with introductory lectures and demonstrations, followed by hands-on projects that allowed students to practice their skills. We also provided technical support and guidance to help students overcome any difficulties they encounter. Students were encouraged to experiment and explore.
At the end, students were evaluated and assessed using tests, quizzes, and assignments to assess their understanding of the concepts and tools covered in the course. We also evaluated their practical projects to see if they can apply what they've learned to build functional IoT projects.

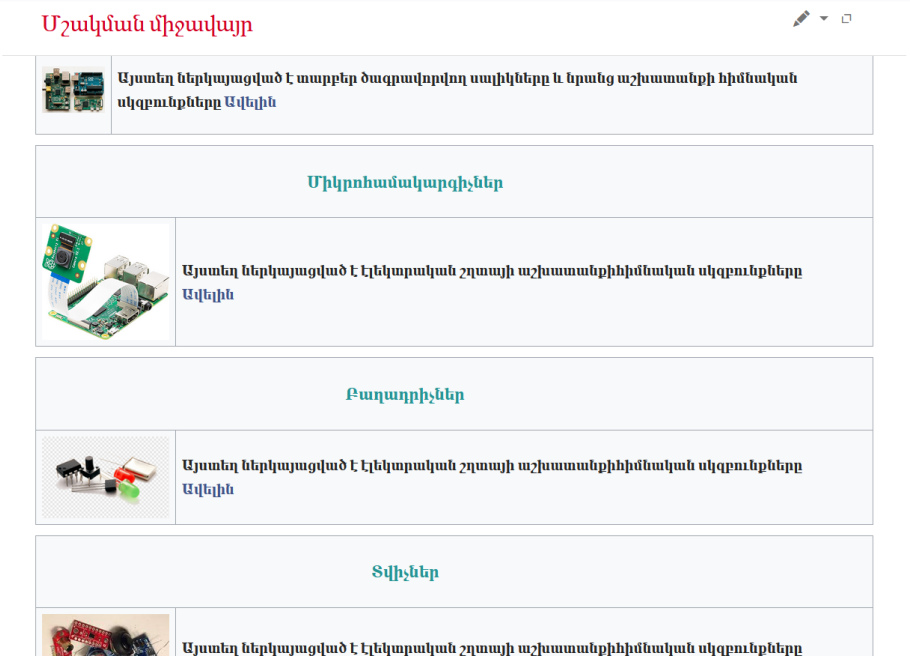
Additionally, an Armenian language wiki knowledge base was developed. The screenshot of it is shown below.



Here is the English translation of the wiki knowledge base sections:

IoT lab	Find	
Entry Points Main page All pages Getting started Arduino programming Basics of electronics Development environment Standard projects Books	IoT lab knowledge base	
	Let's start Development environment Standard projects ThinkerCad examples	Changes Not verified projects System corrections
	Electronics Basics of electronics Signals	Development environment ThinkerCad modeling Arduino programming
	Network Solutions Computer networks	Internet of Things IoT solutions IoT platforms

Information in the repository shows the basic equipment and components for building embedded IOT devices.



The English translation of the repository content is here:

Arduino environment	
	Different programmable plates are presented here More
Microcomputers	
	Electronic circuits operation basic principles are presented here More
Ingredients	
	Electronic circuits operation basic principles are presented here More
Sensors	
	Electronic circuits operation basic principles are presented here More

Standard projects

The repository contains many examples and projects built on Arduino and ESP microcontrollers. IoT devises programming languages and commands.

Here is the English translation of the sections:

Arduino programming		
Arduino language directory		
Operators	Data	Functions
Manageable operators	Confirmations	Digital in/out
Syntaxis	Data types	Analog in/out
Arithmetic operators		In/out additional functions
		Operations with time
		Mathematical functions

Standard projects

Standard projects	
Arduino projects	ESP projects

A detailed description is provided for each project, including a description of the operation, explanations of the equipment connections, and a detailed analysis of the operation of the software code.

Arduino projects

1. Flashing LED
2. Using the button
3. Light control
4. Distance measurement
5. Temperature measurement
6. Irrigation of plants
7. Dali protocol
8. Engine control

9. 1:48 DC motor
10. Stepper motor
11. Controlling engines using Jostik
12. KY-008 laser circuit
13. Laser safety system
14. Measurement of air pollution CO2
15. Connecting the Bluetooth module to the Arduino board
16. Controlled traffic light
17. Relay module
18. Connecting the RFID module to the Arduino
19. Connecting equipment to Arduino via SPI

ESP projects

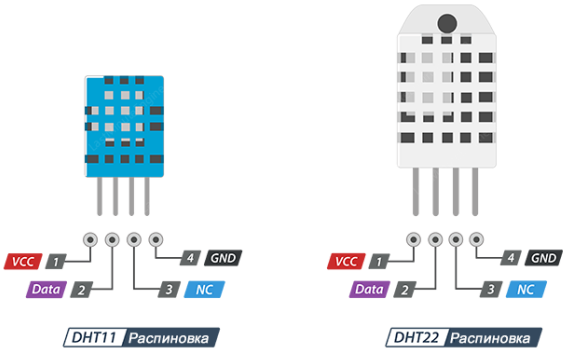
1. Wiring of esp8266
2. Building an ESP8266 web server
3. Weather station based on ESP8266 and BME280
4. Multiple DS18B20 data mirroring via ESP8266
5. ESP8266 DC motor control
6. Getting time from the NTP server with ESP8266
7. ESP8266 software update via OTA
8. ESP8266 intelligent repair system
9. ESP8266 RFID lock
10. Access control system built on ESP8266 and RFIDs

In the description of the projects, the working principles of the component equipment, and connections are presented in detail, as well as a detailed overview of the software solution is carried out.

Measuring a temperature

Ջերմաստիճանի չափում

DHT11 և DHT22/AM2302 սենսորները բավականին հեշտ են միացվում: Նրանք ունեն չորս եզրակացություն.



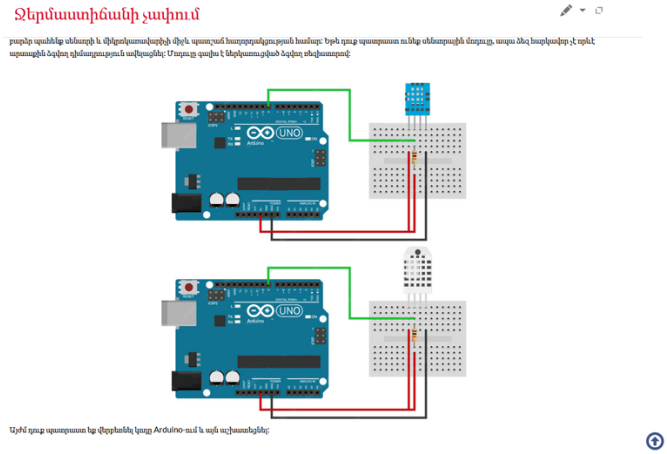
- VCC քորոցը ապահովում է սենսորին էներգիա: Թեև թույլատրվում է սնուցման լարումը 3,3-ից 5,5 Վ միջակայքում, խորհուրդ է տրվում 5 Վ լարում: Այնուամենայնիվ, 3,3 Վ սնուցման լարման դեպքում մալուխի երկարությունը չպետք է գերազանցի 1 մետրը: Հակառակ դեպքում, գծի լարման անկումը կհանգեցնի չափման սխալների:
- Data pin-ն օգտագործվում է սենսորի և միկրոկառավարիչի միջև հաղորդակցվելու համար:
- NC-ը միացված չէ
- GND-ը պետք է միացված լինի Arduino գետնին:

DHT11 և DHT22/AM2302 միացում Arduino UNO-ին

Այժմ, երբ մենք լիովին հասկանում ենք, թե ինչպես է աշխատում DHT սենսորը, մենք կարող ենք սկսել այն միացնել մեր Arduino տախտակին:

Բարեբախտաբար, DHT11, DHT22/AM2302 սենսորները Arduino-ին միացնելը բավականին անհշան է: Նրանք ունեն բավականին երկար կապումներ 0,1 դյույմ (2,54 մ) հարթության վրա, այնպես որ կարող եք հեշտությամբ միացնել դրանք ցանկացած հացահատիկի մեջ: Միացրեք սենսորը 5 Վ-ով և միացրեք հողը: Ի վերջո, միացրեք տվյալների ոտիճը Arduino-ի թվային մուտքի 2-ին:

Measurement of a temperature



Measurement of a temperature

Ջերմաստիճանի չափում

```
}
```

Էսքիզը սկսվում է՝ ներառելով DHT գրադարանը: Հաջորդը, մենք պետք է որոշենք Arduino փին համարը, որին միացված է մեր սենսորի տվյալների փին և ստեղծենք DHT օբյեկտ: Այսպես մենք կարող ենք մուտք գործել գրադարանի հետ կապված հատուկ գործառույթներ.

```
#include <dht.h>
#define dataPin 8 // Նշում է փին համարը, որին միացված է սենսորը:
dht DHT;          // Ստեղծում է DHT օբյեկտ
```

Setup() ֆունկցիայի մեջ մենք պետք է գործարկենք սերիական հաղորդակցման միջերեսը, քանի որ արդյունքները ցուցադրելու համար կօգտագործենք սերիական մոնիտորը:

```
void setup()
{
  Serial.begin(9600);
}
```

Loop() ֆունկցիայի մեջ մենք կօգտագործենք read22() ֆունկցիան, որը կարդում է տվյալները DHT22/AM2302-ից: Ըդպես պարամետր այն վերցնում է սենսորային տվյալների ելքի թիվը: Եթե դուք աշխատում եք DHT11-ով, ապա պետք է օգտագործեք read11() ֆունկցիան: Դուք կարող եք դա անել՝ չմեկնաբանելով երկրորդ տողը:

```
// Մեկնաբանություններից հանեք ձեզ անհրաժեշտ տողը, կախված ձեր օգտագործած սենսորից
int readData = DHT.read22(dataPin);   // DHT22/AM2302
//int readData = DHT.read11(dataPin); // DHT11
```

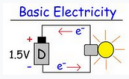
Խոնավության և ջերմաստիճանի արժեքները հաշվարկելուց հետո մենք կարող ենք մուտք գործել դրանք.

```
float t = DHT.temperature; // Ստացեք ջերմաստիճանի արժեքը
float h = DHT.humidity;    // Ստացեք հարաբերական խոնավություն
```

Fundamentals of electrical engineering and electronics

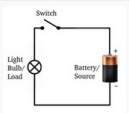
Էլեկտրոնիկայի հիմունքներ

Էլեկտրականություն



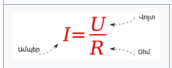
Այստեղ ներկայացված է էլեկտրական շղթայի աշխատանքիհիմնական սկզբունքները
Ավելին

Միացման սխեմաներ



Էլեկտրական շղթաները պատկերելու համար, օգտագործվում են էլեկտրական սխեմաներ:
Ավելին

Էլեկտրականության օրենքներ



Այս օրինակը Ձեր առաջին փորձն է Arduino սալիկը աշխատացնելուԱվելին

Էլեկտրականության կառավարում

Here is the English translation of the section:

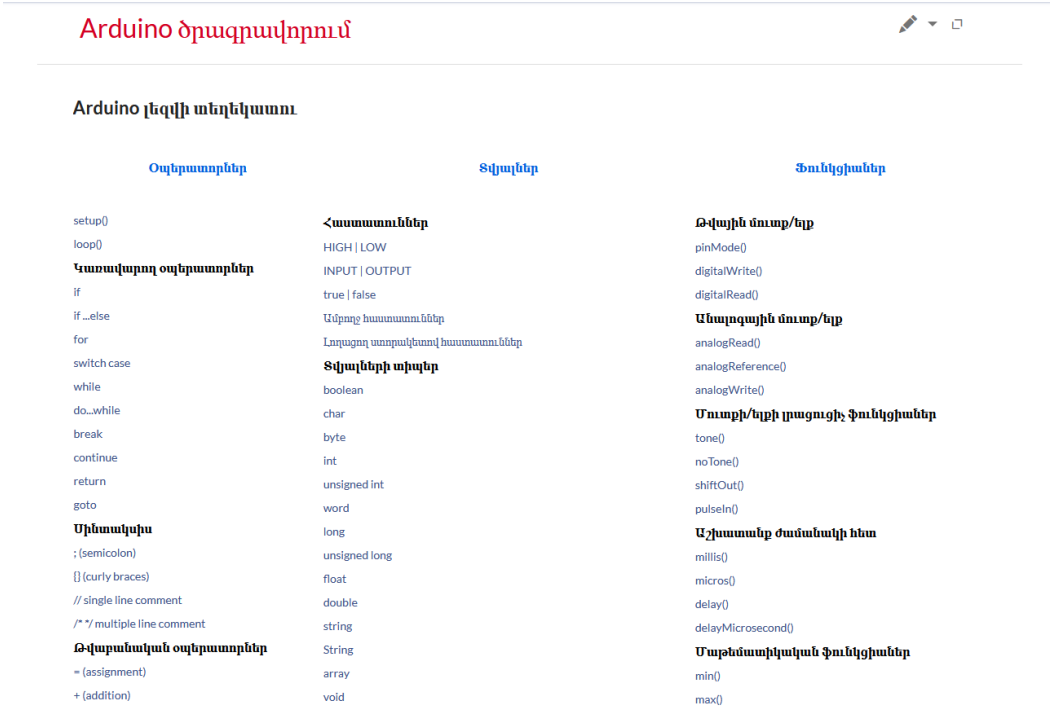
Fundamentals of electronics

	Fundamentals of electrical circuits operation are presented here More
Connection diagrams	
	Electrical diagrams are used for presenting electrical circuits More
Laws of electricity	
	This example is your firtst example of making the Arduino board operational More
Electricity control	

Herein, the structure, characteristics and working principles of microcontrollers, components, sensors and actuators for the development of IOT devices are described.
Examples of circuits built on Arduino and ESP boards are also described in the database.
This section describes:
1. Fundamentals of electric circuit operation
2. Circuit diagrams are used to represent electrical circuits
3. Laws of electricity
4. Electricity management methods
5. Principles of chain construction
6. Principles and applications of the capacitor, resistor, diode, light emitting diode, transistor, motor, and speaker

Signals
Digital and analog signals, their common properties and differences, and processing methods are described here. Digital-to-analog and analog-to-digital converters.

Arduino Programming
The system includes an Arduino C++ programming guide, as well as a guide to configuring and using the Arduino IDE development environment.



The courses were organized into two streams. An announcement was made on social networks and educational institutions about the organization of courses. Then interviews were organized with the candidates for participation.

In the case of the first stream, 10 participants were selected from 28 candidates, and 8 continued to attend later.

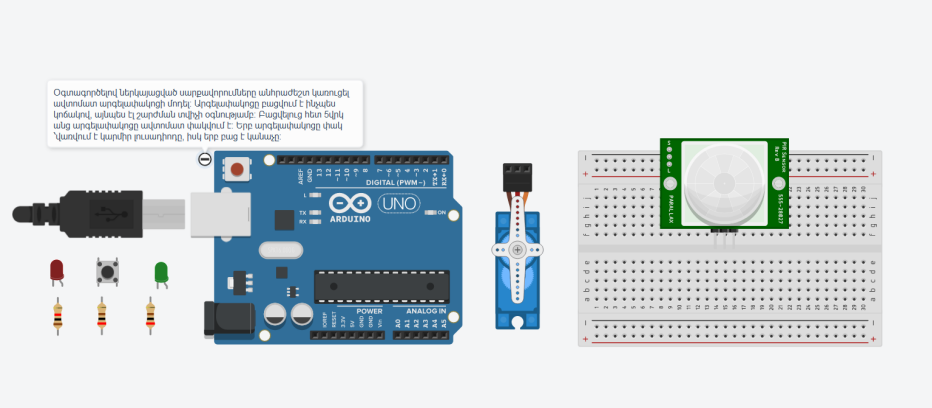
In the case of the second stream, 10 participants were selected from 27 candidates, and 9 continued to attend later.

The list of participants:

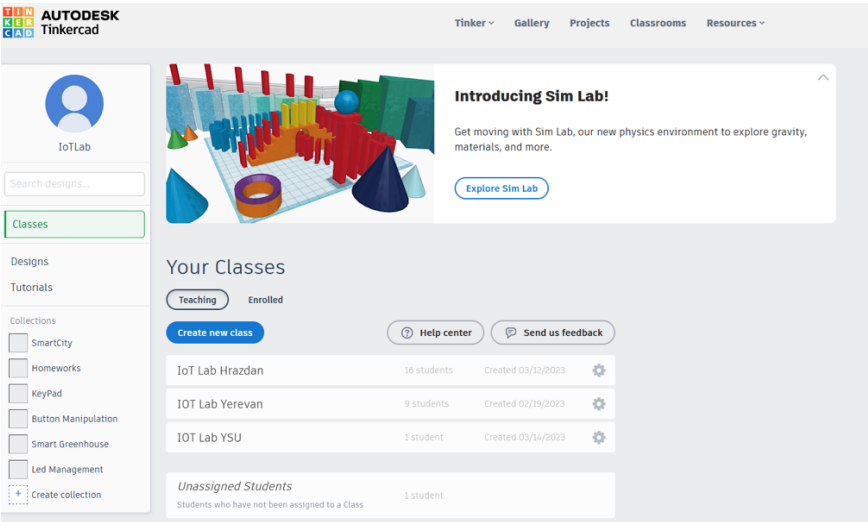
First name	**Last name**	**E-mail**
First Stream		
Lilit	Gevorgyan	lilit.gevorgyan18@edu.ysu.am
Ashot	Kostanyan	kostanyanashot2007@gmail.com
Emma	Karapetyan	emma.karapetyan2007@gmail.com
Edgar	Hovhanisyan	edgarhovhannisyan9622@gmail.com
Srbuhi	Khachatryan	srbuhi2003@gmail.com
Vazgen	Davtyan	jjanartur@mail.ru
Manvel	Qocharyan	manvelqocharyan77@gmail.com
Marianna	Faltajanyan	faltmarianna@gmail.com
Miqayel	Mkrtchyan	miqayel05015@gmail.com
Harutyun	Mnacakanyan	harut.mnatsakanayn@bk.ru
Second Stream		
Hasmik	Hovhanisyan	hashovhannisyan03@mail.ru
Miqayel	Ovanesov	movanesov7@gmail.com
Sona	Hovhanisyan	sonahovhannisyan145@gmail.com

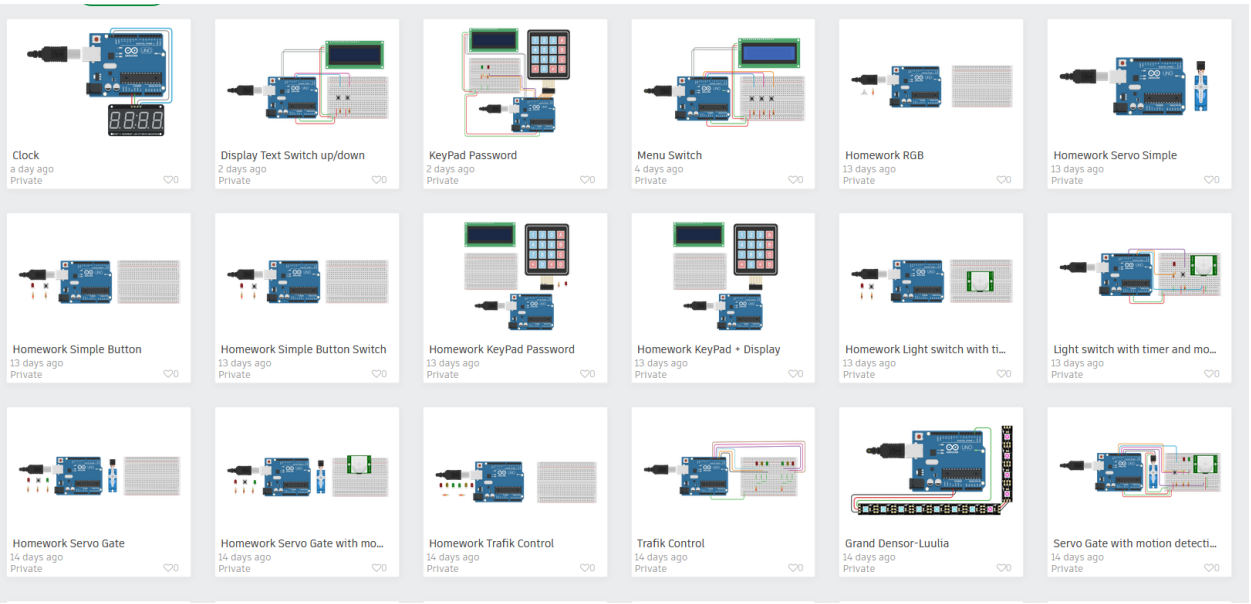
Afina	Agajanyan	Afinochka07@gmail.com
Narek	Stepanyan	stepanyann938@gmail.com
Neli	Babajanyan	neli96369nelli@gmail.com
Hovsep	Kashanyan	hos_kashanyan@gmail.com
Miqayel	Buduryan	mikaelbuduryan@gmail.com
Davit	Hovhannisyan	hovhannisyandavid2007@gmail.com
Inga	Matevosyan	ingamatevosyan46@gmail.com

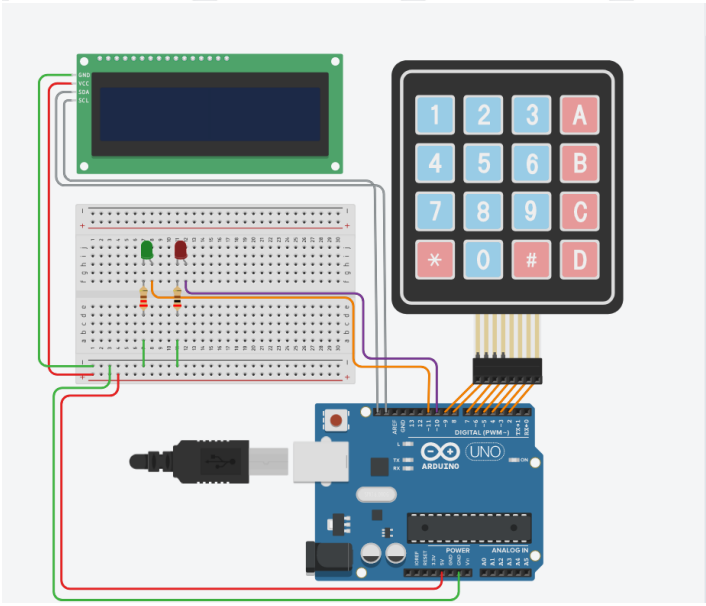
The orientation of the courses was mainly practical, but there were courses where theoretical knowledge was also provided, the PowerPoint presentations of such courses are attached. During practical courses, participants worked directly on physical Arduino Uno, ESP8266, and ESP32 boards. In addition, online tools such as https://tinkercad.com were widely used. Through the platform, participants were also given individual homework, and each participant's work was further discussed in detail. This methodology allowed for strengthening the participants' knowledge.



Images of participants' works follows.

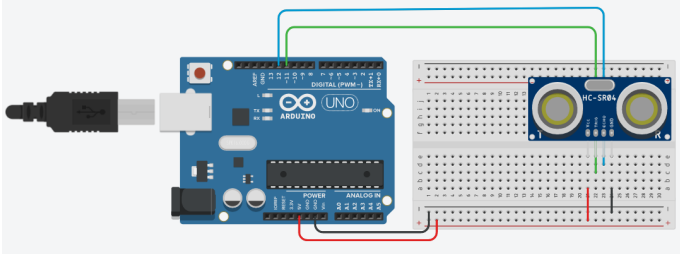




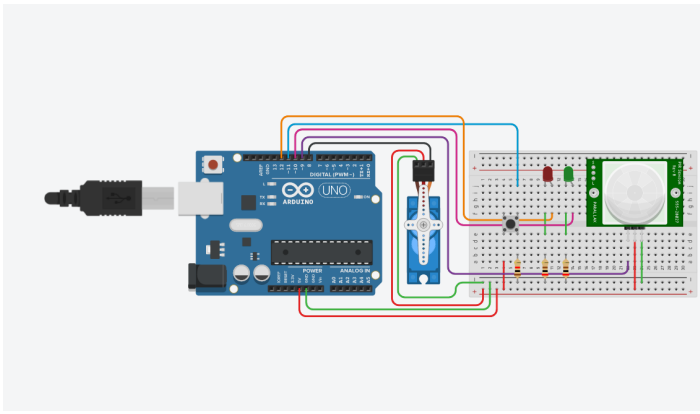


Text

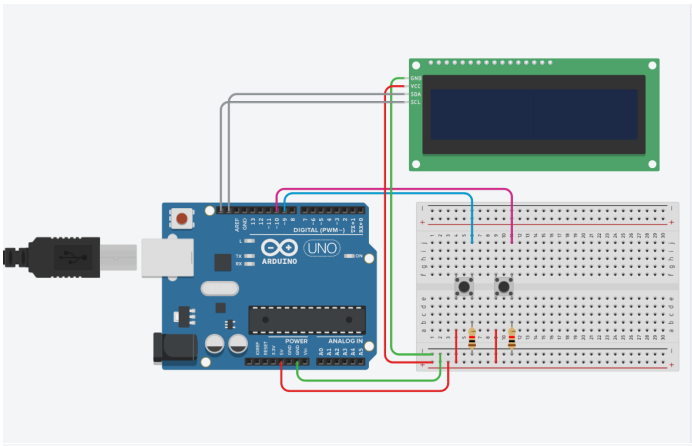
```
// C++ code
//
#include <Wire.h>
#include <Keypad.h>
#include <Adafruit_LiquidCrystal.h>

Adafruit_LiquidCrystal lcd_1(0);

const byte ROWS = 4;
const byte COLS = 4;

int redledPin = 10;
int greenledPin = 11;

char hexaKeys[ROWS][COLS] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};

byte rowPins[ROWS] = {9, 8, 7, 6};
byte colPins[COLS] = {5, 4, 3, 2};

const String password = "1234A";
String input_password;
bool first;

Keypad customKeypad = Keypad(makeKeymap(hex

void setup()
{
  lcd_1.begin(16, 2);
  input_password="";
  lcd_1.setCursor(0, 0);
  pinMode(redledPin, OUTPUT);
  pinMode(greenledPin, OUTPUT);
  digitalWrite(redledPin, HIGH);
  digitalWrite(greenledPin, LOW);
  }

void loop(){
  char customKey = customKeypad.getKey();
  if (customKey)

  {
```



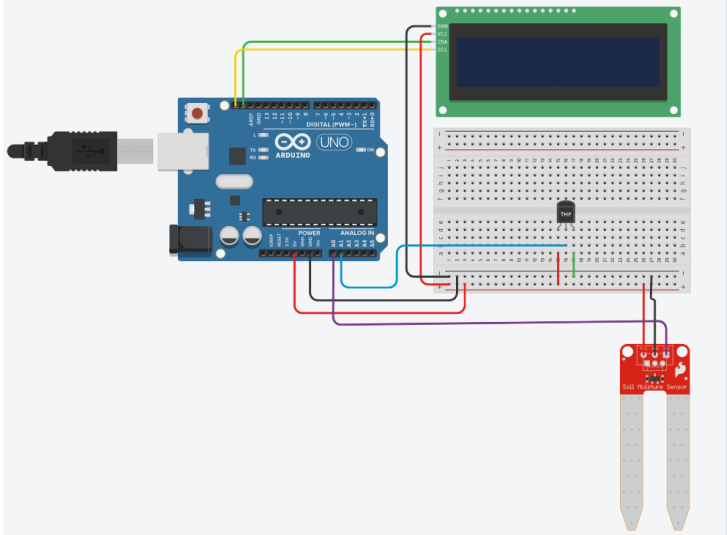

```
#define PIN_TRIG 11
#define PIN_ECHO 12

long duration, cm;

void setup() {

  // Инициализируем взаимодействие по последовательному порту

  Serial.begin (9600);
  //Определяем входы и выходы
  pinMode(PIN_TRIG, OUTPUT);
  pinMode(PIN_ECHO, INPUT);
}

void loop() {

  // Сначала генерируем короткий импульс длительностью 2-5 микро

  digitalWrite(PIN_TRIG, LOW);
  delayMicroseconds(5);
  digitalWrite(PIN_TRIG, HIGH);

  // Выставим высокий уровень сигнала, ждем около 10 микросекунд
  delayMicroseconds(10);
  digitalWrite(PIN_TRIG, LOW);

  //  Время задержки акустического сигнала на эхолокаторе.
  duration = pulseIn(PIN_ECHO, HIGH);

  // Теперь осталось преобразовать время в расстояние
  cm = (duration / 2) / 29.1;

  Serial.print("Distance: ");
  Serial.print(cm);
  Serial.println(" cm.");

  // Задержка между измерениями для корректной работы скеча
  delay(250);
}
```



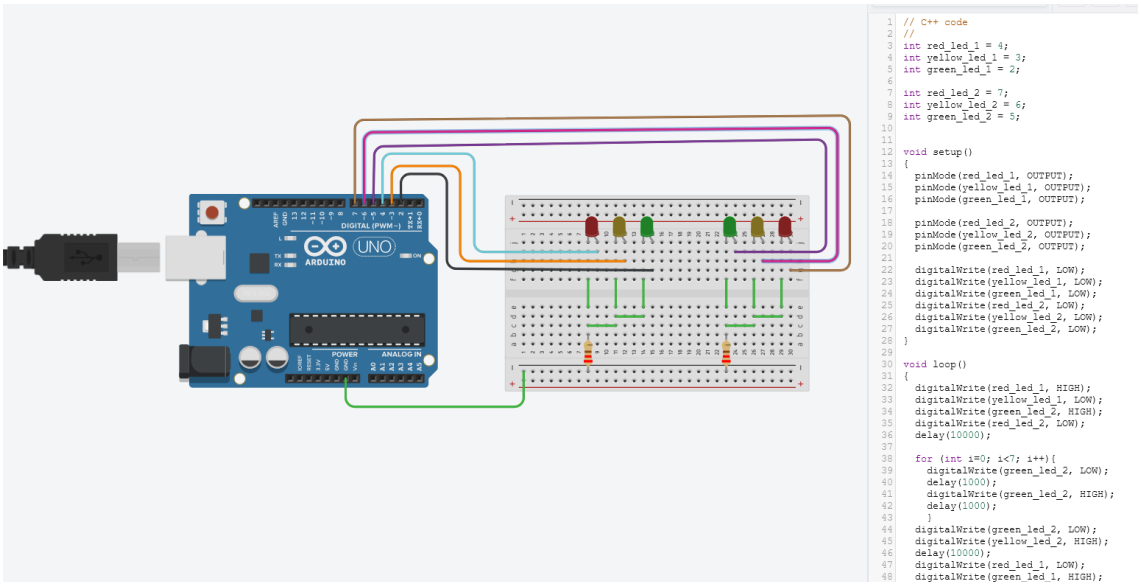
```
// C++ code
//

#include <Servo.h>

int pos = 0;

Servo servo_8;

unsigned long ms_from_start =0;
unsigned long ms_prevIous_read_LED   = 0;
unsigned long LED_interval=5000;

int buttonPin = 11;
bool buttonState=0;
bool lastbuttonState=0;

#define LED1 12
#define LED2 10
#define PIN_PIRSENSOR 9

int LED_state=0;
//int LED2_state=0;

void setup() {
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(buttonPin, INPUT);
  pinMode(PIN_PIRSENSOR, INPUT);
  servo_8.attach(8, 500, 2500);
  Serial.begin(9600);
  for (pos = 90; pos >= 0; pos -= 1) servo_8.write(pos);
}

void loop() {
  buttonState = digitalRead(buttonPin);
  int movement_detected = digitalRead(PIN_PIRSENSOR);
  if (((buttonState != lastbuttonState)&&(buttonState == HIGH)&&(LED
    {
```



```
// C++ code
//
#include <Adafruit_LiquidCrystal.h>

Adafruit_LiquidCrystal lcd_1(0);

String Menu[4] = {"Temperature", "Moisture","Light", "Air Polution"
int buttonUpPin = 10;
int buttonDownPin = 9;

int MenuIndex=0;
int lastMenuIndex=0;

bool buttonUpState;
bool lastbuttonUpState;
bool buttonDownState;
bool lastbuttonDownState;


void setup()
{
 pinMode(buttonUpPin, INPUT);
 pinMode(buttonDownPin, INPUT);
 lcd_1.begin(16, 2);
 lcd_1.clear();
 lcd_1.setCursor(0, 0);
 lcd_1.print(Menu[0]);
}

void loop()
{
 buttonUpState = digitalRead(buttonUpPin);
 buttonDownState = digitalRead(buttonDownPin);
 if ((buttonUpState != lastbuttonUpState)&&(buttonUpState == HIGH
 {
   if (MenuIndex <3)
     MenuIndex++;
 }
 else if ((buttonDownState != lastbuttonDownState)&&(buttonDownSt
 {
 if (MenuIndex >0)
     MenuIndex--;
 }
 //if (((buttonUpState != lastbuttonUpState)&&(buttonUpState == H
 if ((((buttonUpState != lastbuttonUpState)&&(buttonUpState == HI
```

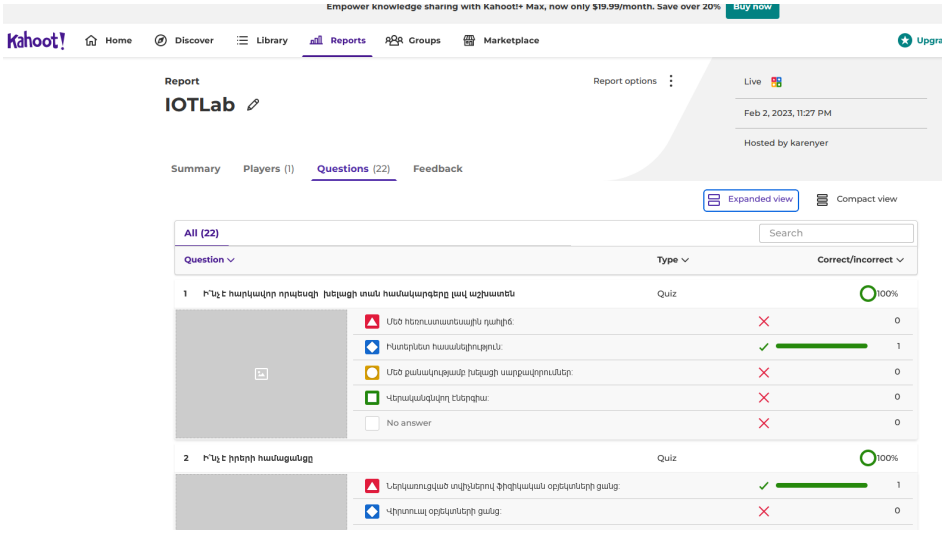


```
// C++ code
//
#include <Adafruit_LiquidCrystal.h>
//#include <LiquidCrystal_I2C.h>

int seconds = 0;
float moisture;

float temperature;

Adafruit_LiquidCrystal lcd_1(0);

void setup()
{
  pinMode(A0,INPUT);
  pinMode(A1,INPUT);
  lcd_1.begin(16, 2);
}

void loop()
{
  moisture=analogRead(A0);
  lcd_1.clear();
  lcd_1.setCursor(0, 0);
  DisplayOutput("Moisture", moisture);
  delay(5000);

  temperature=analogRead(A1);
  lcd_1.clear();
  DisplayOutput("Temperature", temperature);
  delay(5000);

}


int DisplayOutput(String name, float value)
{
  lcd_1.setCursor(0, 0);
  lcd_1.print(name);
  lcd_1.setCursor(0, 1);
  lcd_1.print("      ");
  lcd_1.setCursor(0, 1);
  lcd_1.print(value);
}
```



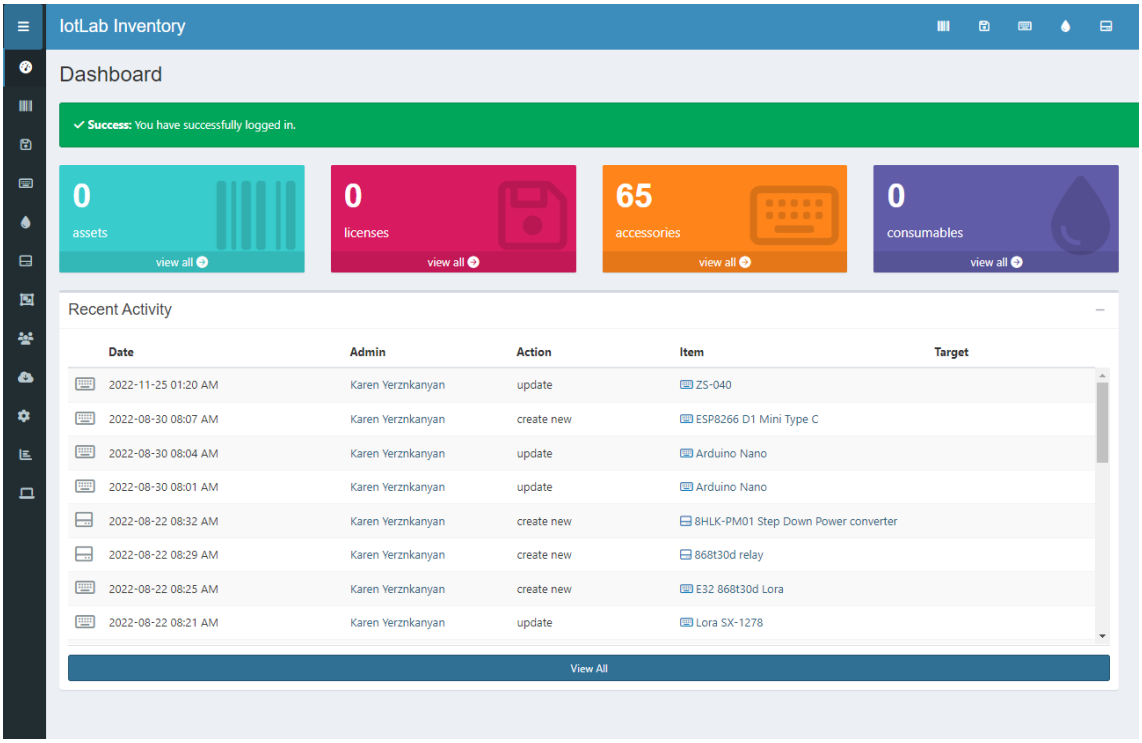
```
// C++ code
//
int red_led_1 = 4;
int yellow_led_1 = 3;
int green_led_1 = 2;

int red_led_2 = 7;
int yellow_led_2 = 6;
int green_led_2 = 5;


void setup()
{
  pinMode(red_led_1, OUTPUT);
  pinMode(yellow_led_1, OUTPUT);
  pinMode(green_led_1, OUTPUT);

  pinMode(red_led_2, OUTPUT);
  pinMode(yellow_led_2, OUTPUT);
  pinMode(green_led_2, OUTPUT);

  digitalWrite(red_led_1, LOW);
  digitalWrite(yellow_led_1, LOW);
  digitalWrite(green_led_1, LOW);
  digitalWrite(red_led_2, LOW);
  digitalWrite(yellow_led_2, LOW);
  digitalWrite(green_led_2, LOW);
}

void loop()
{
  digitalWrite(red_led_1, HIGH);
  digitalWrite(yellow_led_1, LOW);
  digitalWrite(green_led_2, HIGH);
  digitalWrite(red_led_2, LOW);
  delay(10000);

  for (int i=0; i<7; i++){
    digitalWrite(green_led_2, LOW);
    delay(1000);
    digitalWrite(green_led_2, HIGH);
    delay(1000);
    }
  digitalWrite(green_led_2, LOW);
  digitalWrite(yellow_led_2, HIGH);
  delay(10000);
  digitalWrite(red_led_1, LOW);
  digitalWrite(green_led_1, HIGH);
```

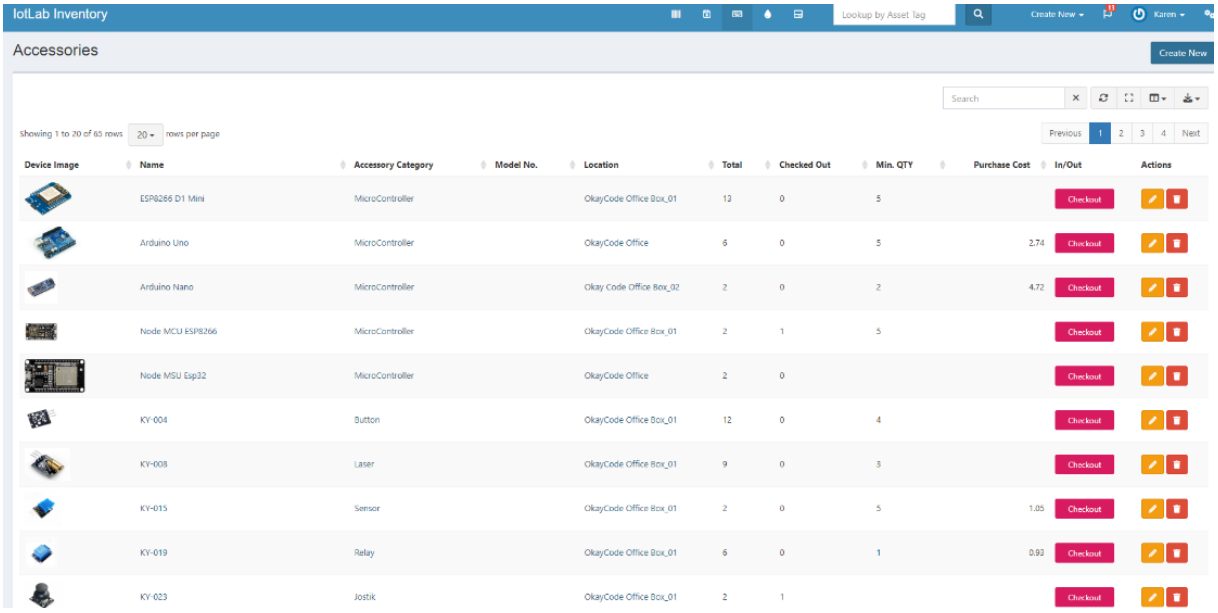
In order to test and strengthen the theoretical and practical knowledge of participants, four questions were prepared for each stream in a game way. The features of the https://kahoot.com online platform were used for that.



Necessary equipment and tools, processing boards, and electronic nodes were purchased for the Laboratory.

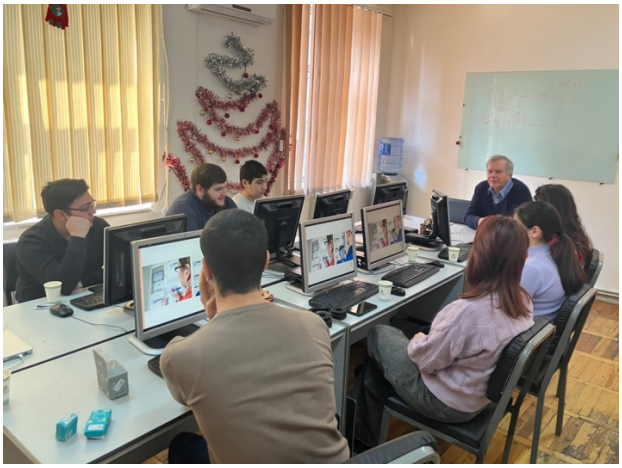
In order to effectively organize the inventory of the equipment of the IOT laboratory, an electronic platform for the inventory of equipment and electronic components was built http://inventory.iotlab.am.

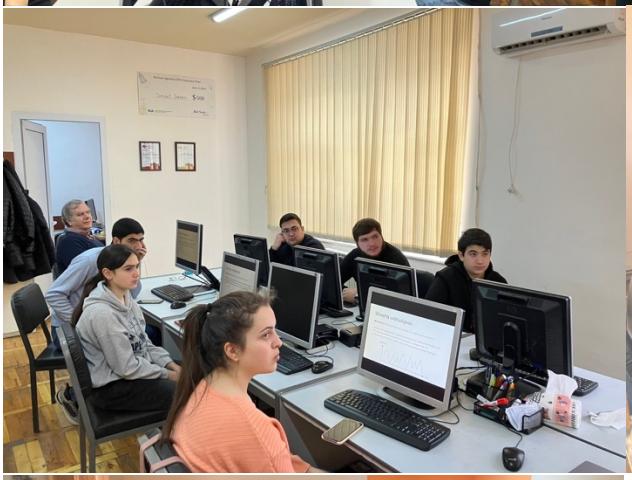

IotLab Inventory
Dashboard
Success: You have successfully logged in.
0
assets
view all
0
licenses
view all
65
accessories
view all
0
consumables
view all
Recent Activity
Date
Admin
Action
Item
Target
2022-11-25 01:20 AM
Karen Yerznkanyan
update
ZS-040
2022-08-30 08:07 AM
Karen Yerznkanyan
create new
ESP8266 D1 Mini Type C
2022-08-30 08:04 AM
Karen Yerznkanyan
update
Arduino Nano
2022-08-30 08:01 AM
Karen Yerznkanyan
update
Arduino Nano
2022-08-22 08:32 AM
Karen Yerznkanyan
create new
8HLK-PM01 Step Down Power converter
2022-08-22 08:29 AM
Karen Yerznkanyan
create new
868t30d relay
2022-08-22 08:25 AM
Karen Yerznkanyan
create new
E32 868t30d Lora
2022-08-22 08:21 AM
Karen Yerznkanyan
update
Lora SX-1278
View All

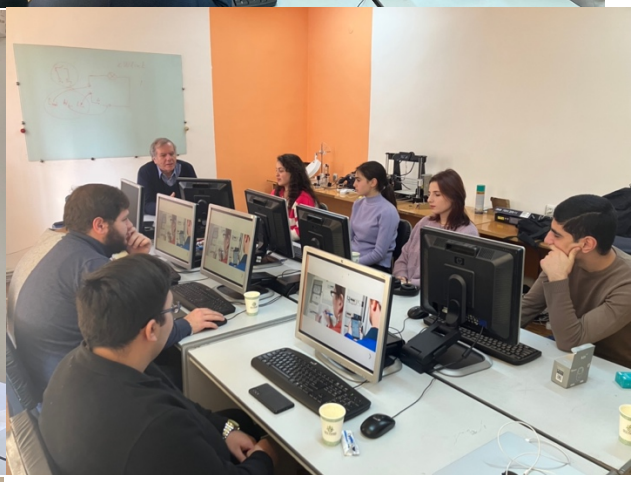

IotLab Inventory
Lookup by Asset Tag
Create New
Karen
Accessories
Create New
Search
Showing 1 to 20 of 65 rows
20
rows per page
Previous
1
2
3
4
Next
Device Image
Name
Accessory Category
Model No.
Location
Total
Checked Out
Min. QTY
Purchase Cost
In/Out
Actions
ESP8266 D1 Mini
MicroController
OkayCode Office Box_01
13
0
5
Checkout
Arduino Uno
MicroController
OkayCode Office
6
0
5
2.74
Checkout
Arduino Nano
MicroController
Okay Code Office Box_02
2
0
2
4.72
Checkout
Node MCU ESP8266
MicroController
OkayCode Office Box_01
2
1
5
Checkout
Node MSU Esp32
MicroController
OkayCode Office
2
0
Checkout
KY-004
Button
OkayCode Office Box_01
12
0
4
Checkout
KY-008
Laser
OkayCode Office Box_01
9
0
3
Checkout
KY-015
Sensor
OkayCode Office Box_01
2
0
5
1.05
Checkout
KY-019
Relay
OkayCode Office Box_01
6
0
1
0.93
Checkout
KY-023
Jostik
OkayCode Office Box_01
2
1
Checkout

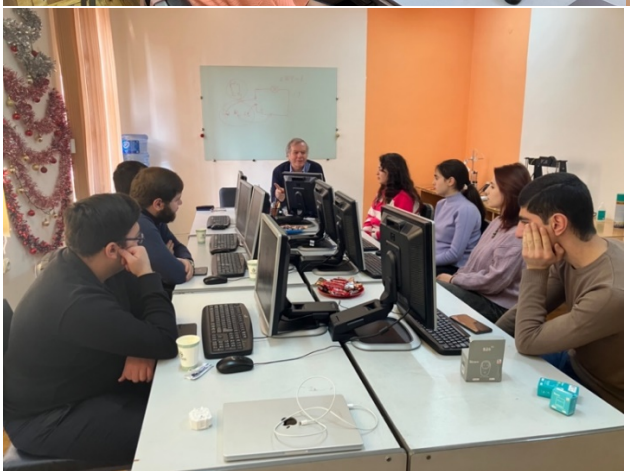
65 types of equipment and components were registered in the system.
As part of the implementation of the project, models of smart lighting and smart greenhouses were built in the laboratory.

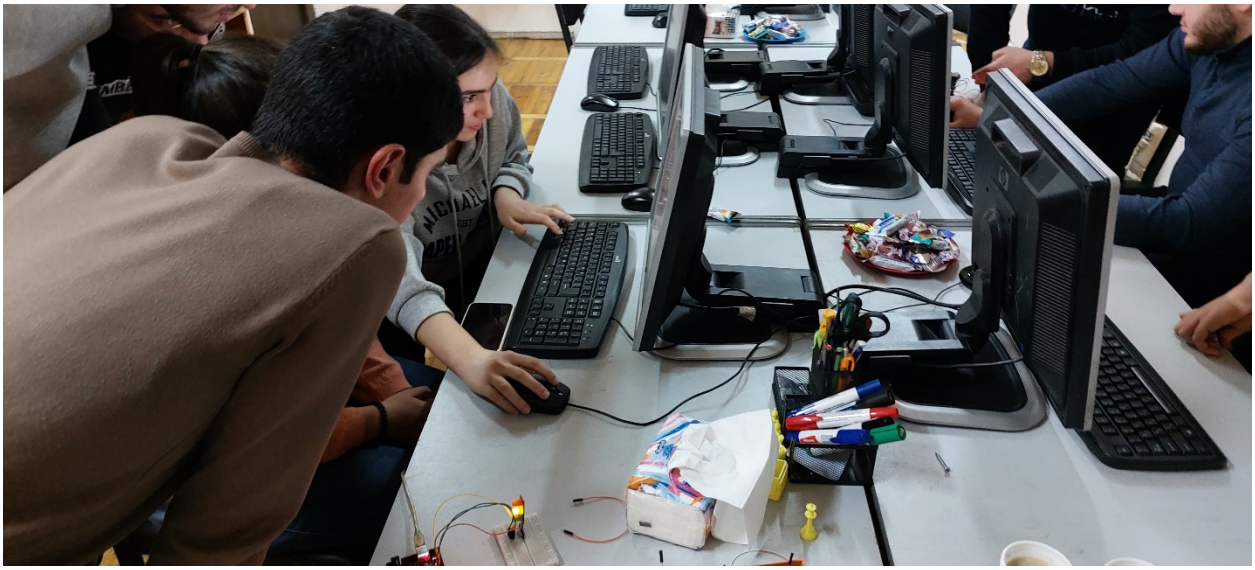
Photos of classes

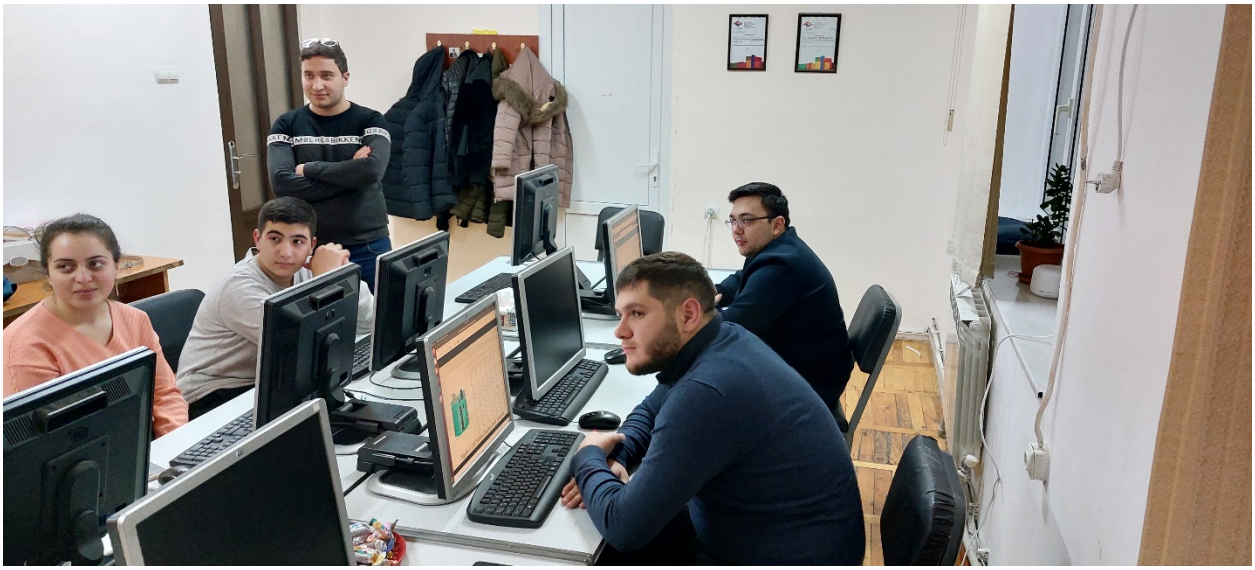


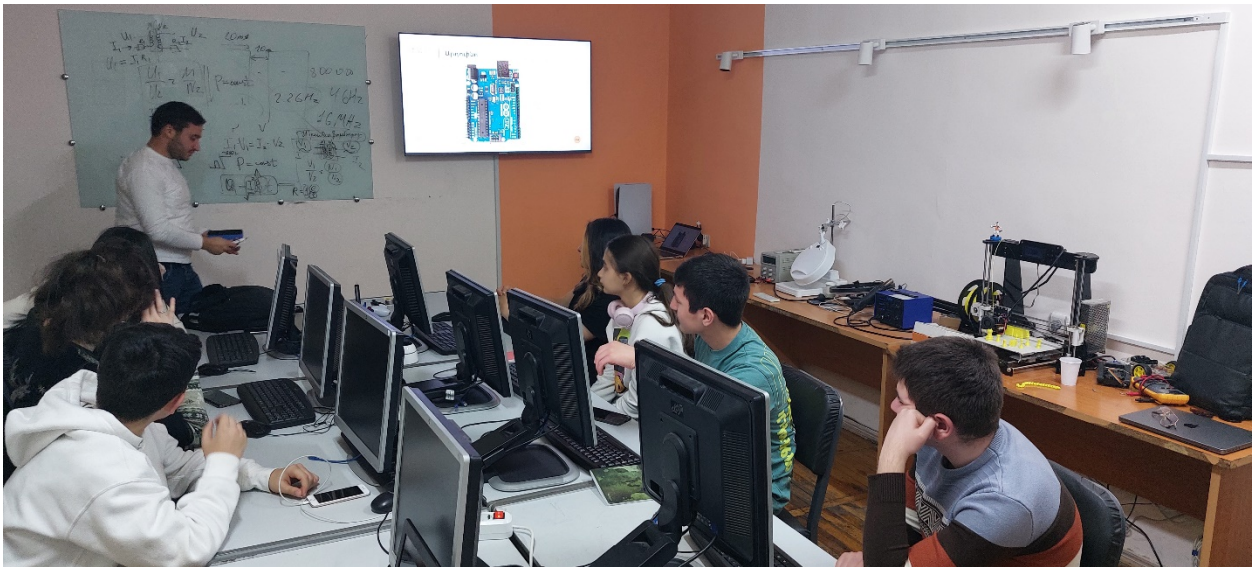


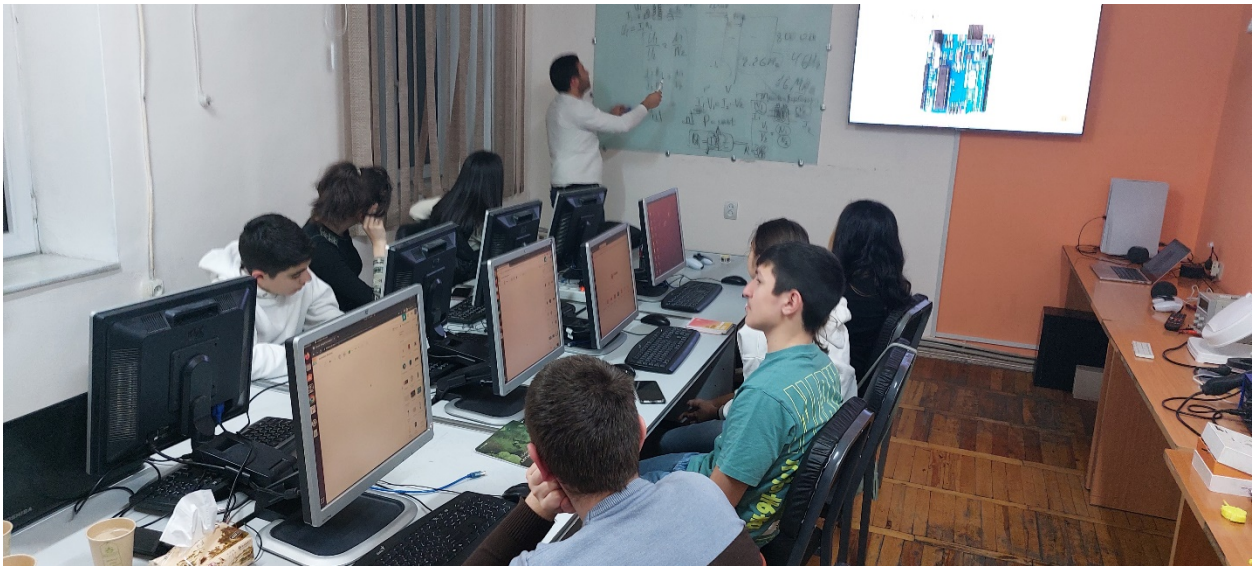




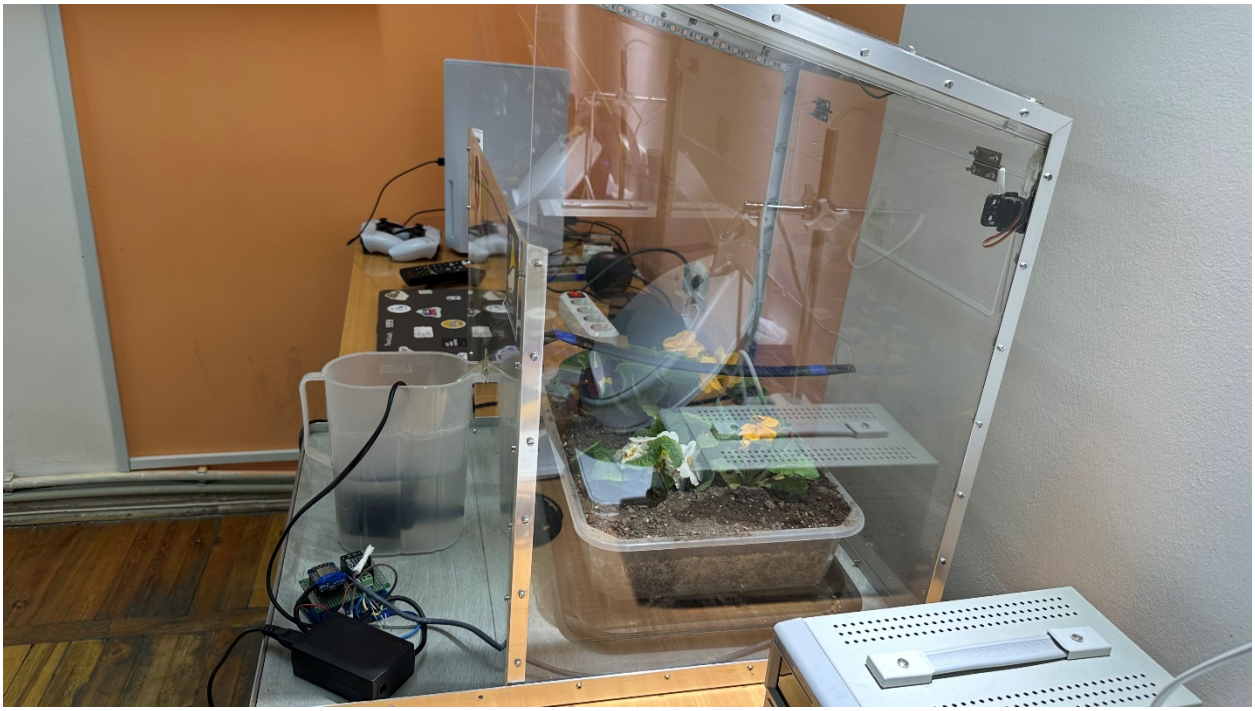






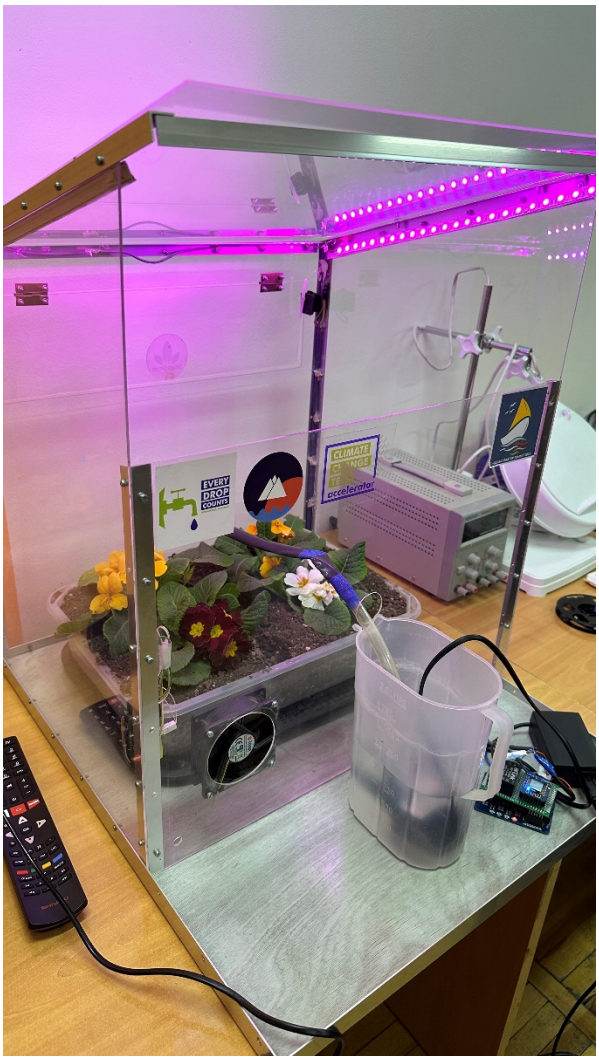


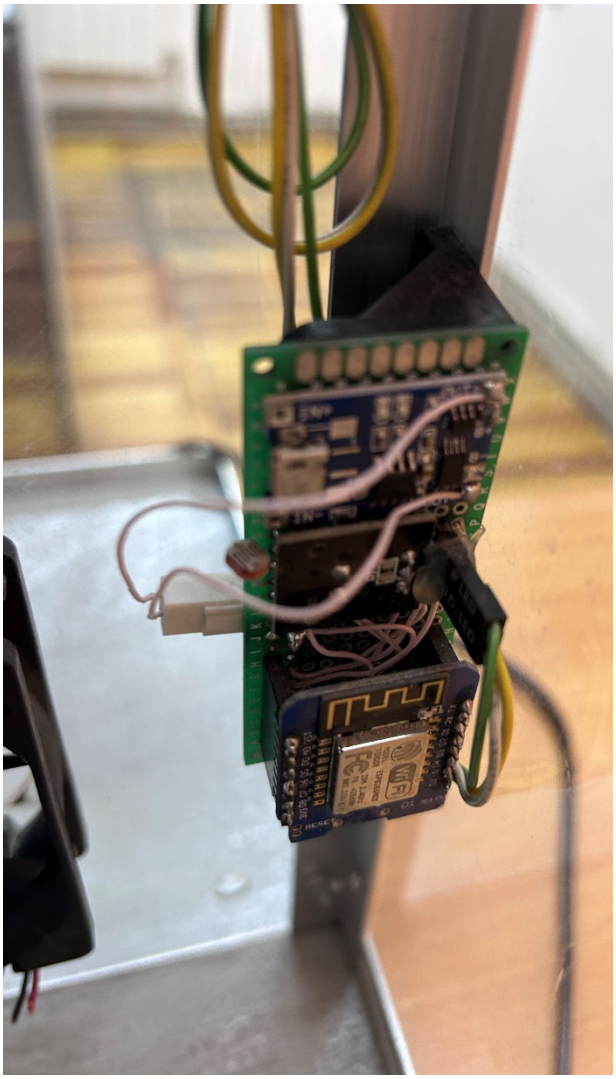
Smart Greenhouse demonstration model consist from automatic irrigation, heating, cooling and lighting systems. Sensors and actuators are based on ESP8266 boards, and main greenhouse automated system is based on Raspberry PI microcomputer.




EVERY
H2O
COUNTS
CLIMATE
CHANGE
TECH
accelerator


EVERY DROP COUNTS
CLIMATE CHANGE TECH
accelerator


EVERY DROP COUNTS
accelerator



SmartHome demo model is based on Sonoff smart lights and smart power switch devices. SmartHome lighting system is integrated with Amazon Alexa and Yandex Alisa voice assistants.