

EXPERIMENT-01

NAME: K.Swethank Rohan

SECTION: B-15

ROLL NUMBER:122010315005

Aim: Control the LED with Arduino Board and tinkercad software

Objectives: To get the knowledge of Arduino Board and control of output device (LED)

Outcomes: We will be able to write a program using Arduino IDE for Blinking LED.

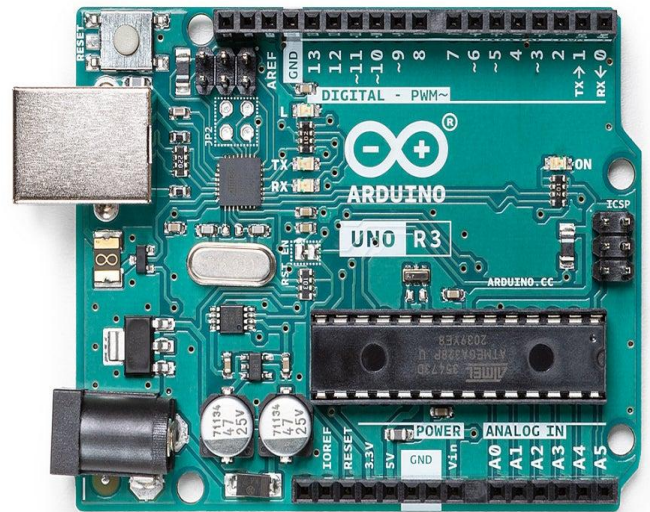
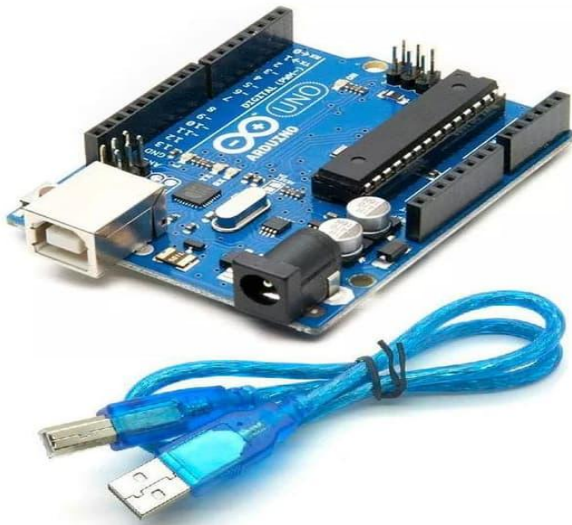
Hardware Requirements:

1. 1x Breadboard
2. 1x Arduino Uno
3. 1x LED
4. 1x 330 Ω Resistor
5. 2x JumperWires

Theory:

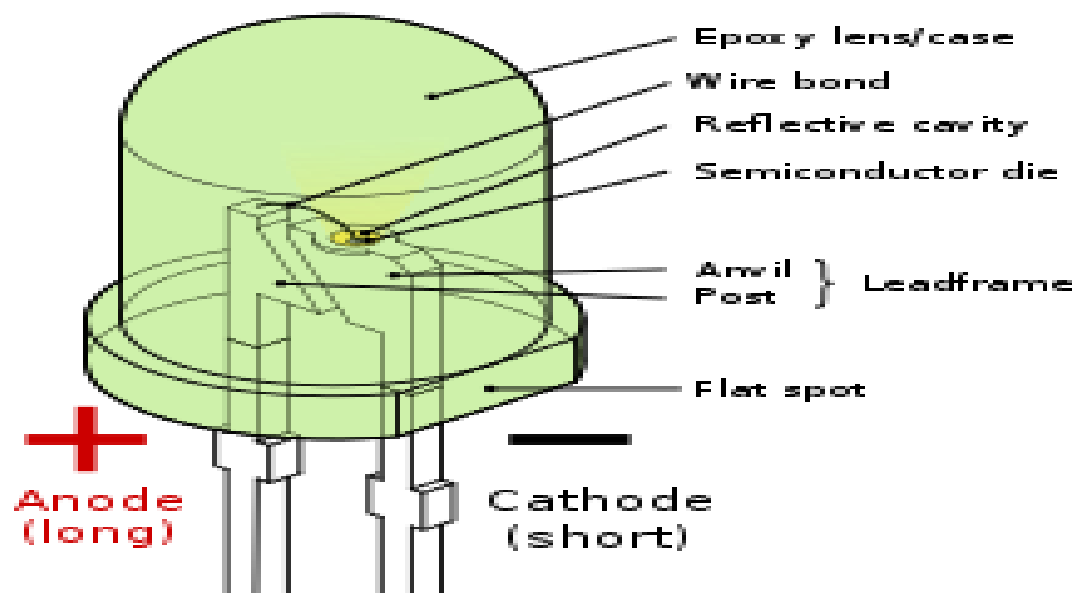
Arduino Uno:

Arduino UNO is a low-cost, flexible, and easy-to-use programmable open-source microcontroller board that can be integrated into a variety of electronic projects. This board can be interfaced with other Arduino boards, Arduino shields, Raspberry Pi boards and can control relays, LEDs, servos, and motors as an output.



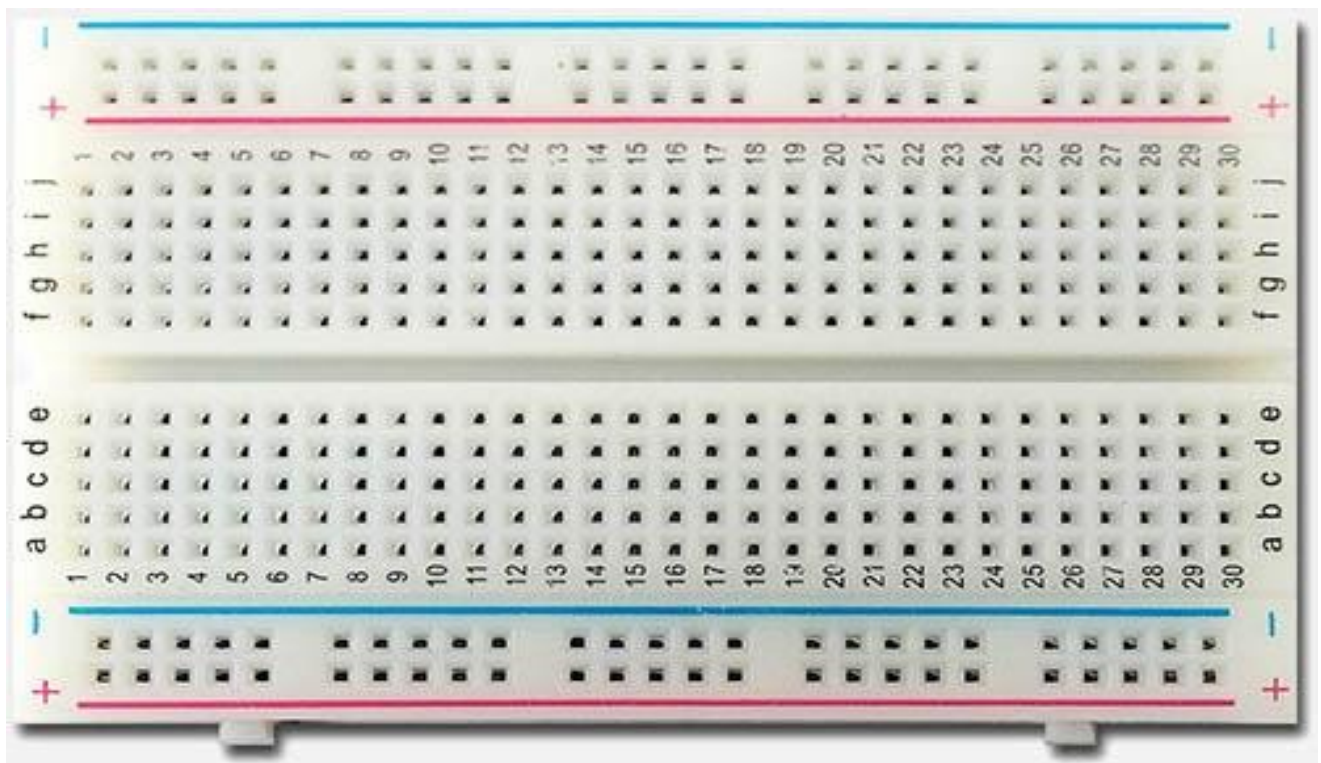
LED:

A light-emitting diode is a semiconductor light source that emits light when current flows through it. LEDs (light-emitting diodes) are small, bright, power-efficient lights commonly used in electronic products. An LED light is a polarized part, meaning it has to be connected to a circuit in a certain way to work properly.



Breadboard:

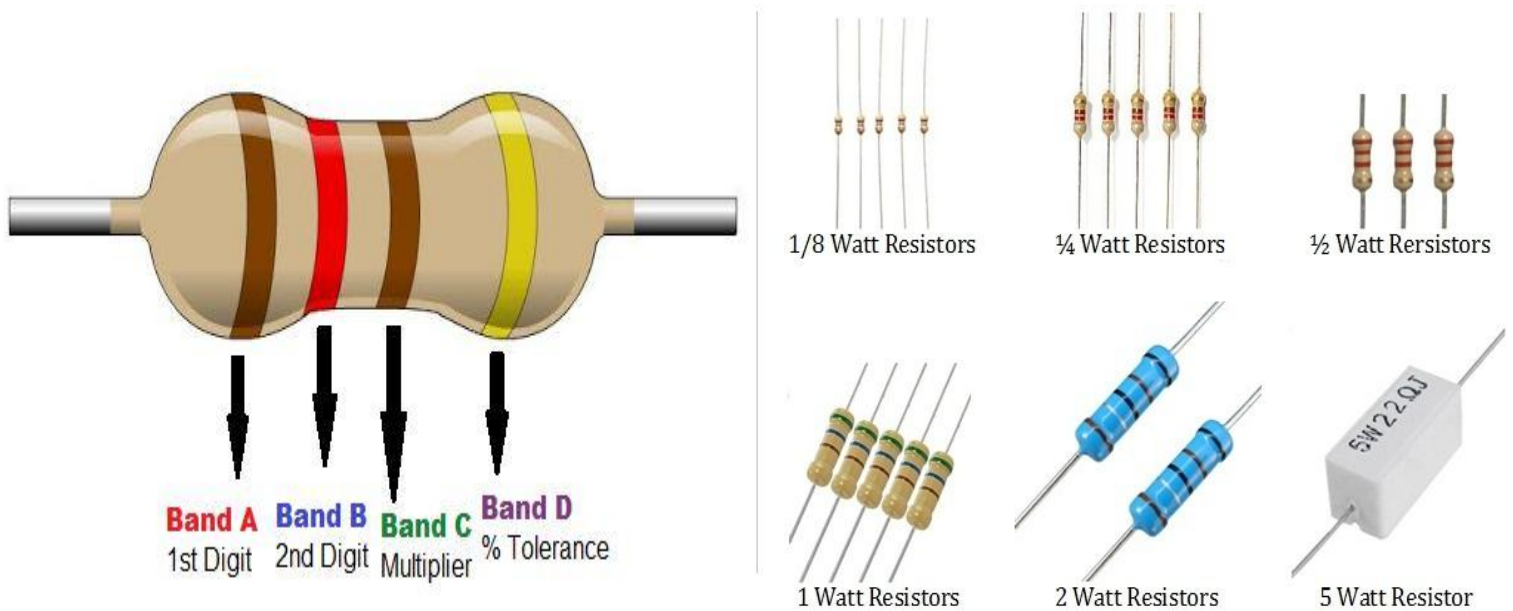
A breadboard is a rectangular plastic board with a bunch of tiny holes in it. These holes let you easily insert electronic components to prototype (meaning to build and test an early version of) an electronic circuit, like this one with a battery, switch, resistor, and an LED (light-emitting diode). It is used to build and test circuits quickly before finalizing any circuit design. The breadboard has many holes into which circuit components like ICs and resistors can be inserted.



Resistor:

A resistor is a passive two-terminal electrical component that implements electrical resistance as a circuit element. In electronic circuits, resistors are used to reduce current flow, adjust signal levels, to

divide voltages, bias active elements, and terminate transmission lines, among other uses.



Procedure:

1. Open and create a new account at www.tinkercad.com or log in with an existing Gmail account.



Welcome back

How will you sign in?

Students, join your class

Email or Username

Sign in with Google

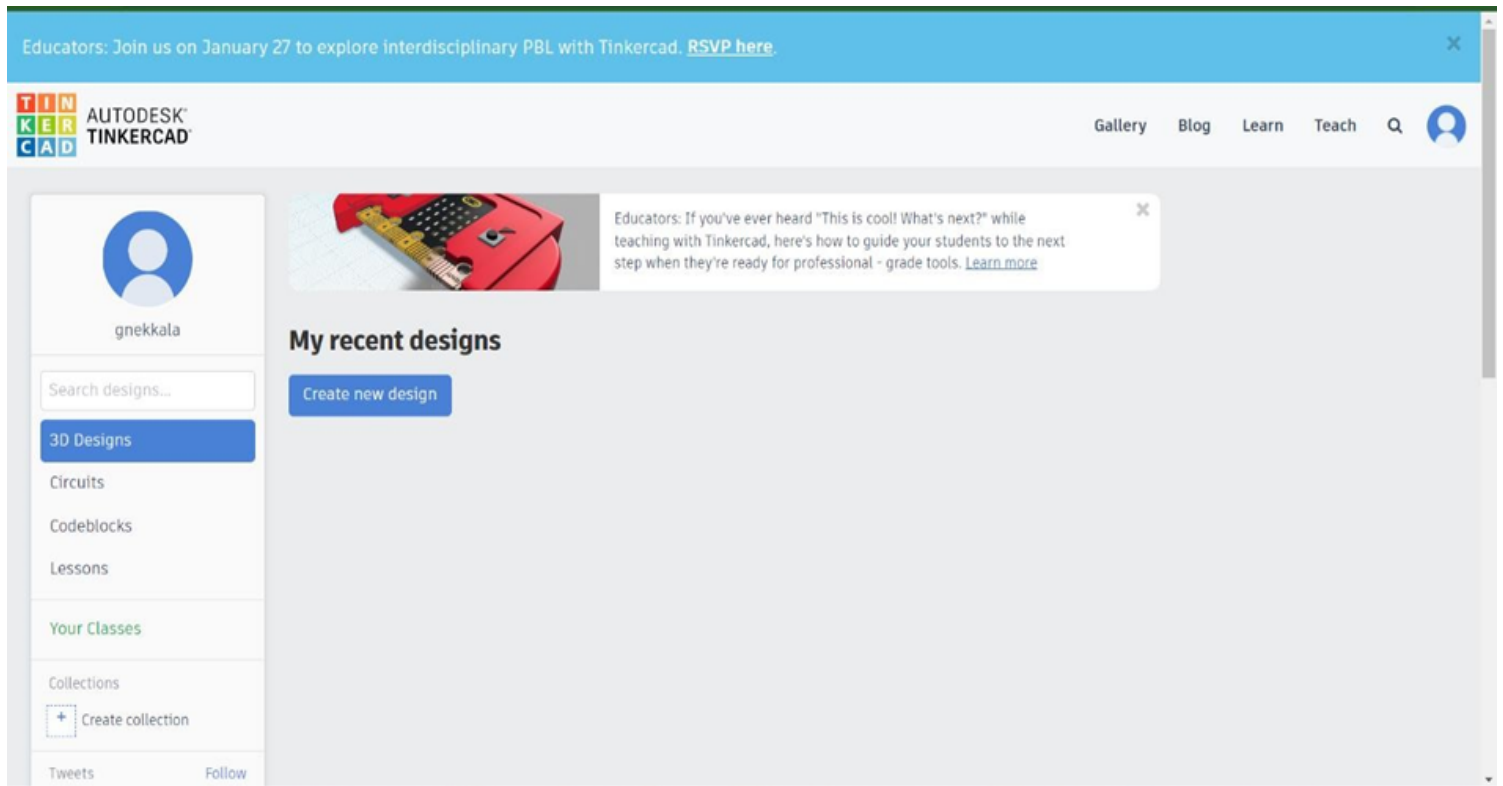
Sign in with Apple

More sign in options...

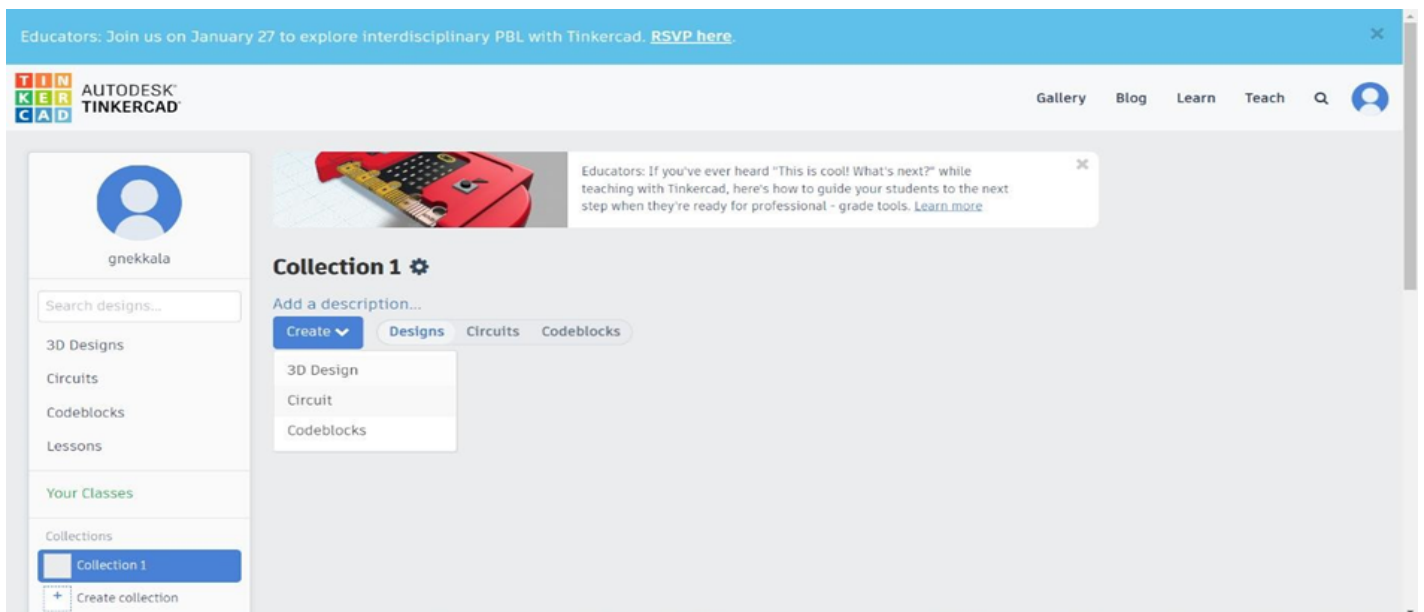
Don't have an account yet?

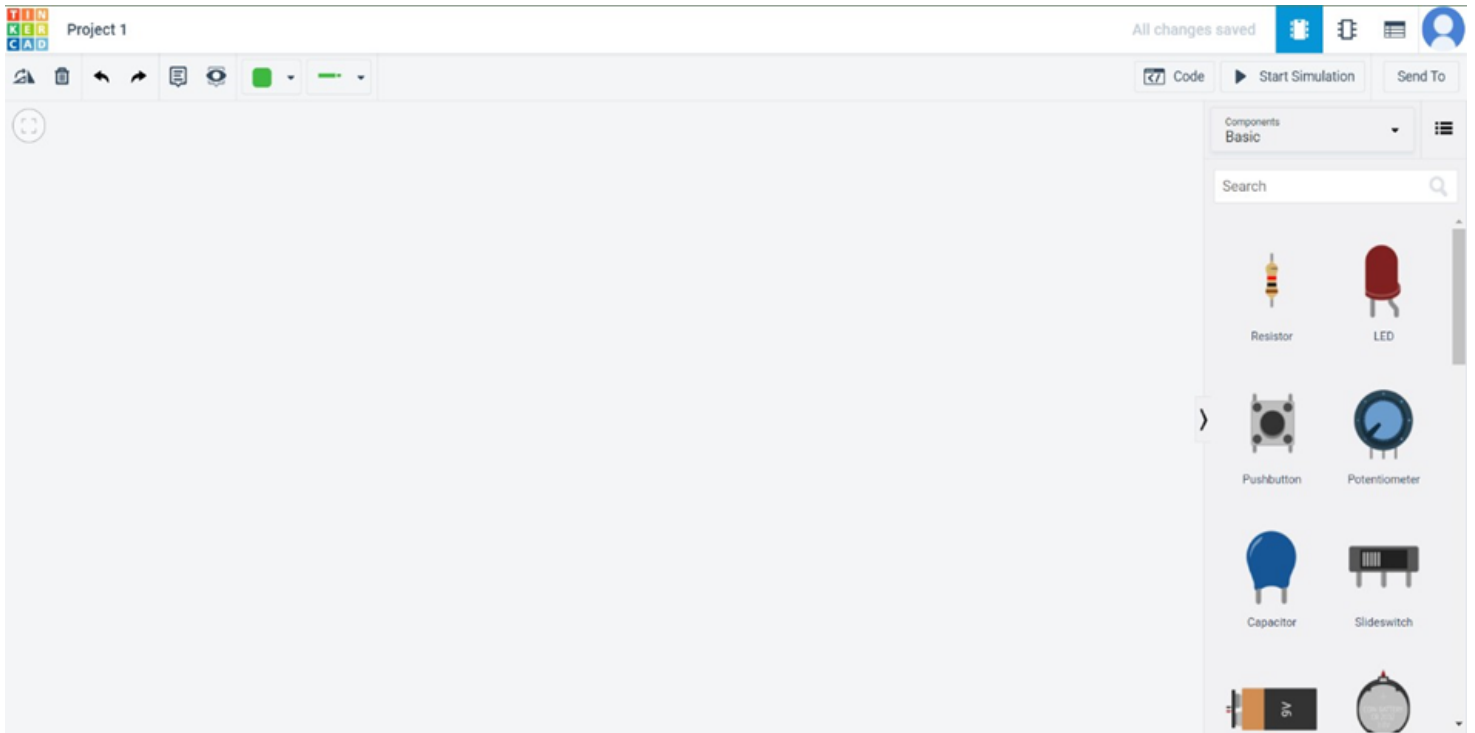
Join Tinkercad

2. Click on go to create Collection and create a new collection.

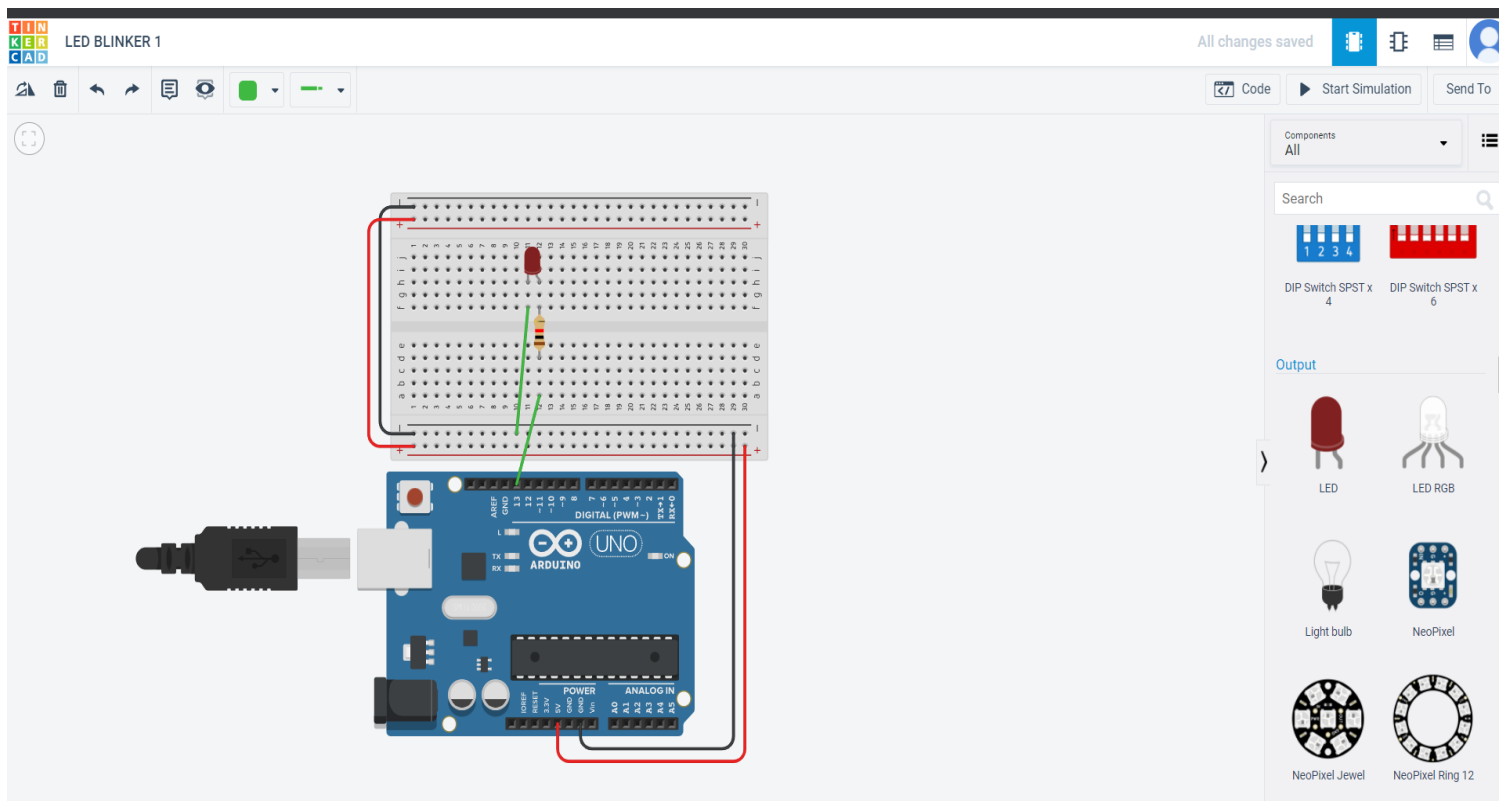
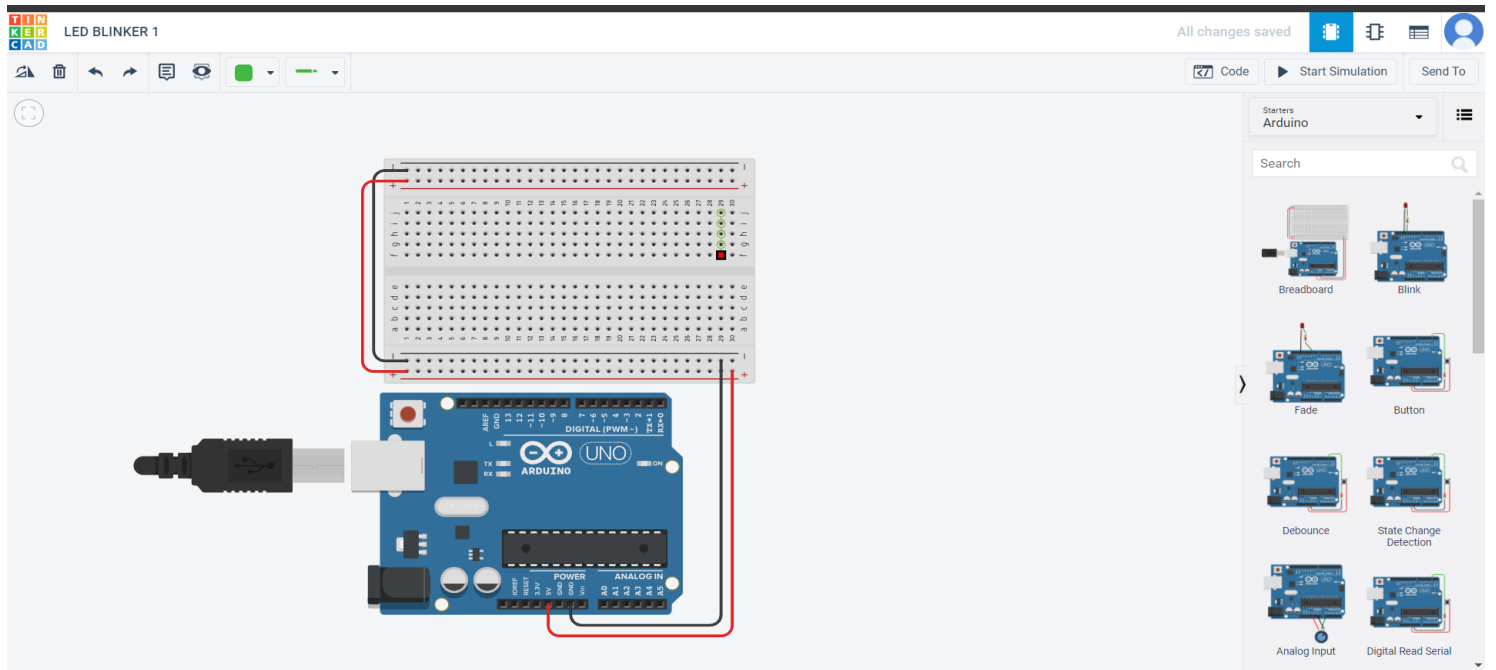


3. Go to create menu and select circuit



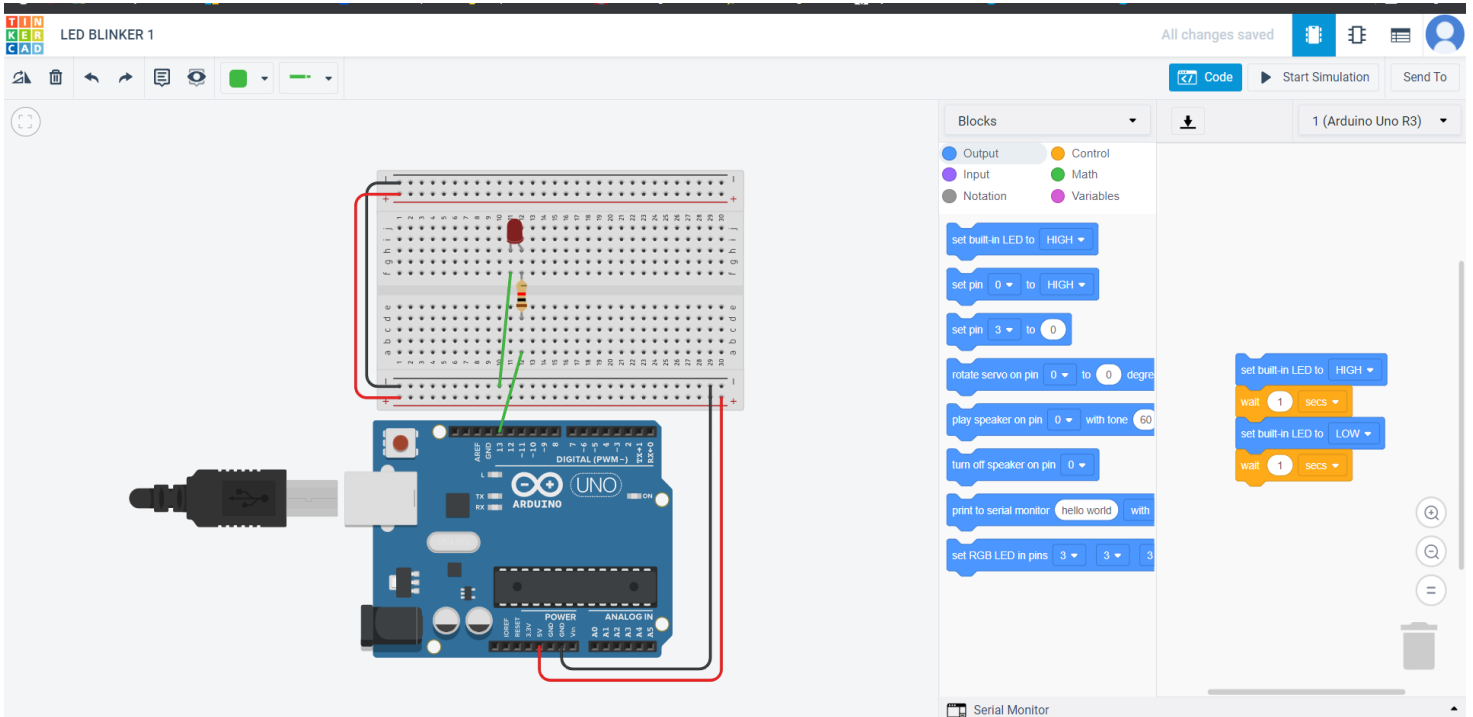


4. Select the Arduino and breadboard and place it in the design area.
5. Search the component LED and resistor and make connections.
Configure the resistor value as 330 ohms.
6. Attach the LED to an output pin of the Arduino D13.



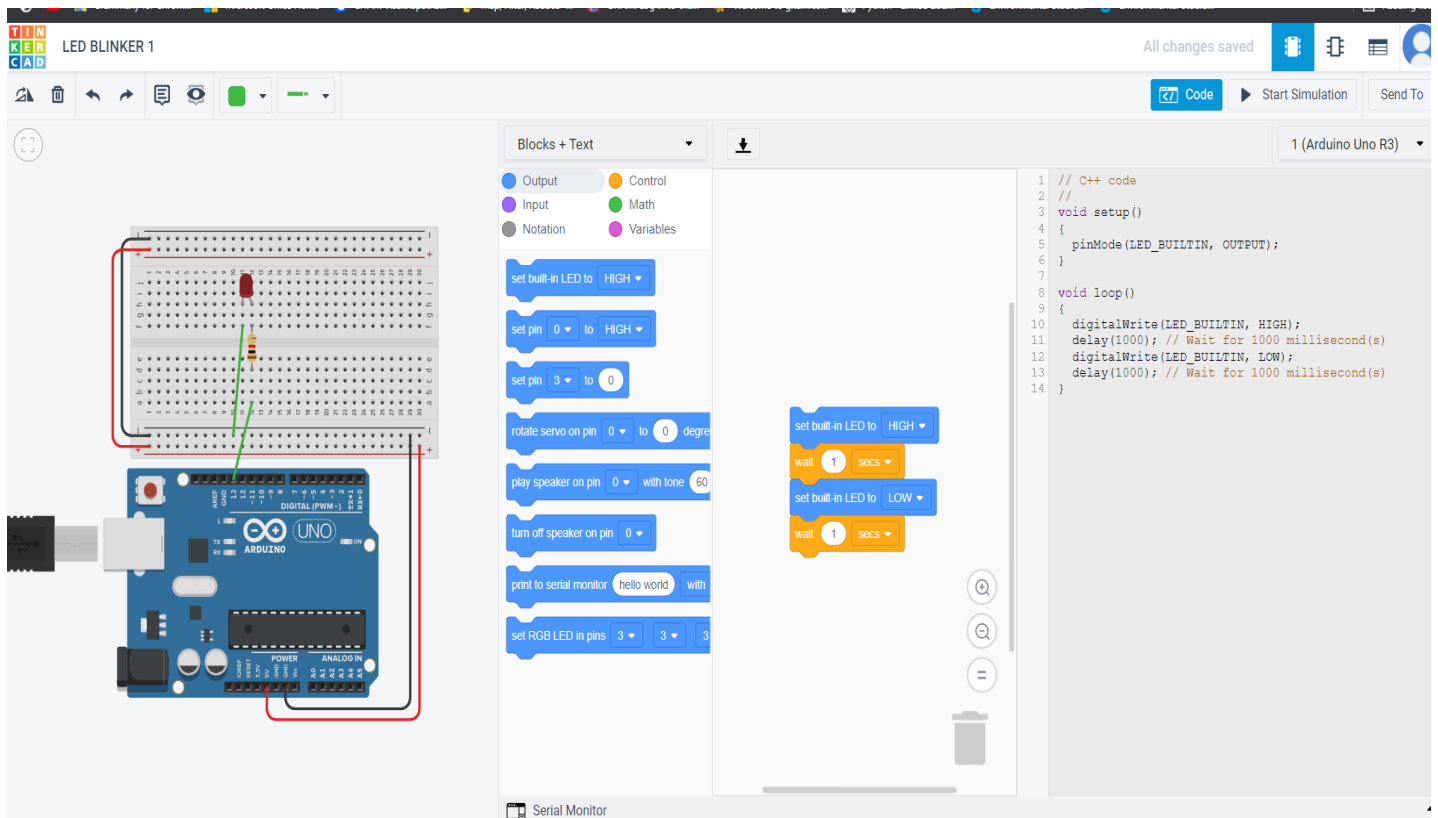
7. Once the circuit connection are ready, programming the Arduino can be done in three ways.

1. Using code blocks



The screenshot shows the Tinkercad interface for a project named "LED BLINKER 1". On the left, an Arduino Uno R3 is connected to a breadboard. A red wire connects the 5V pin of the Arduino to the positive rail of the breadboard. A green wire connects the ground pin of the Arduino to the negative rail of the breadboard. A red LED is connected to the positive rail with its anode and to the negative rail with its cathode. On the right, the code editor shows a sequence of code blocks: "set built-in LED to HIGH", "wait 1 secs", "set built-in LED to LOW", and "wait 1 secs". The "Blocks" panel on the left lists various block categories: Output, Input, Notation, Control, Math, and Variables.

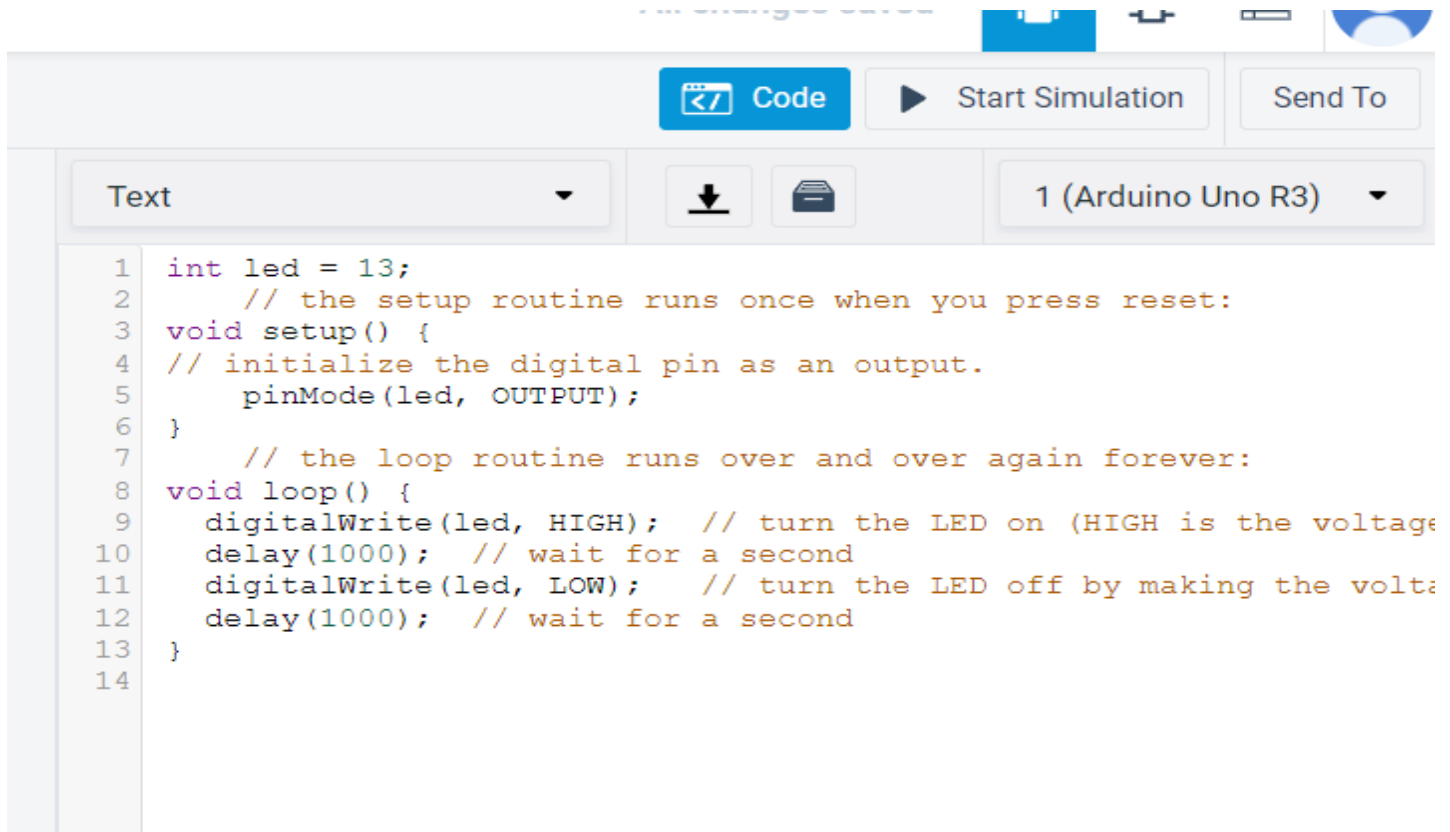
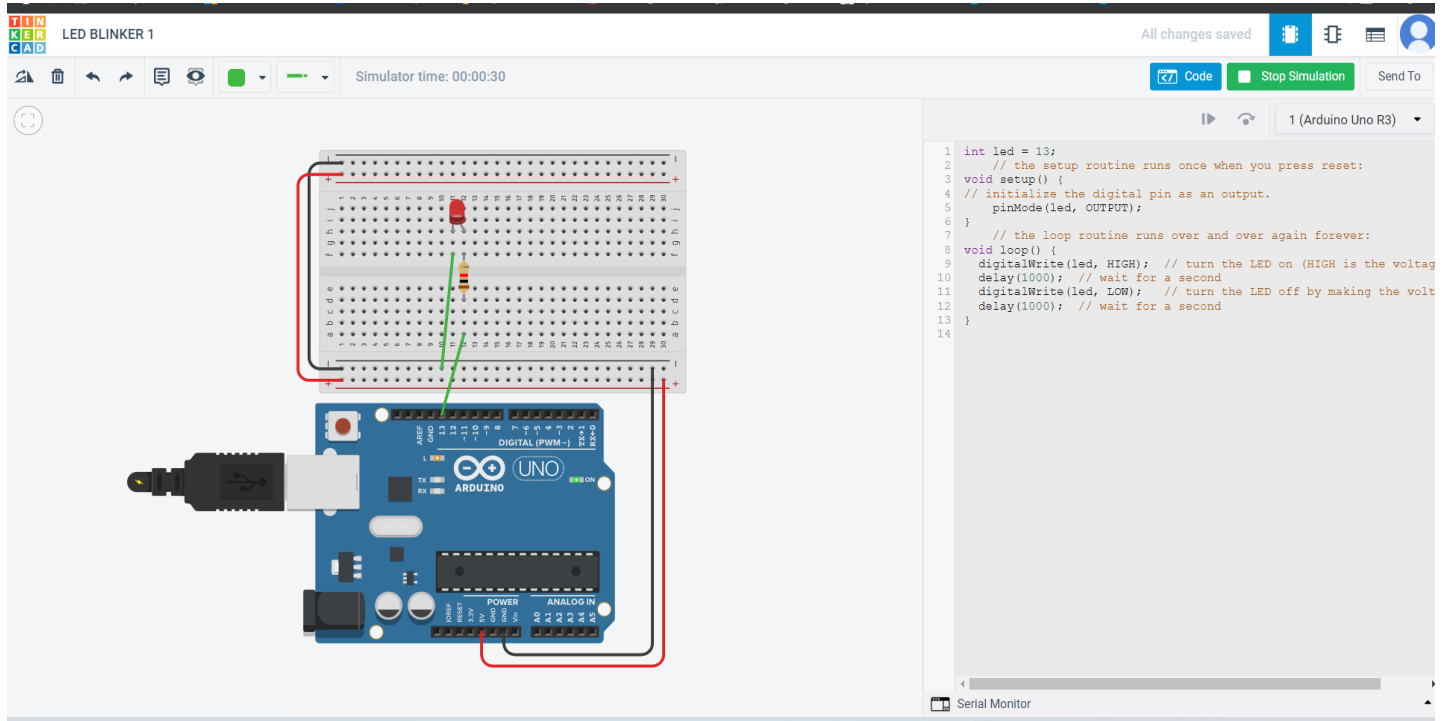
2. Using code blocks + text programming



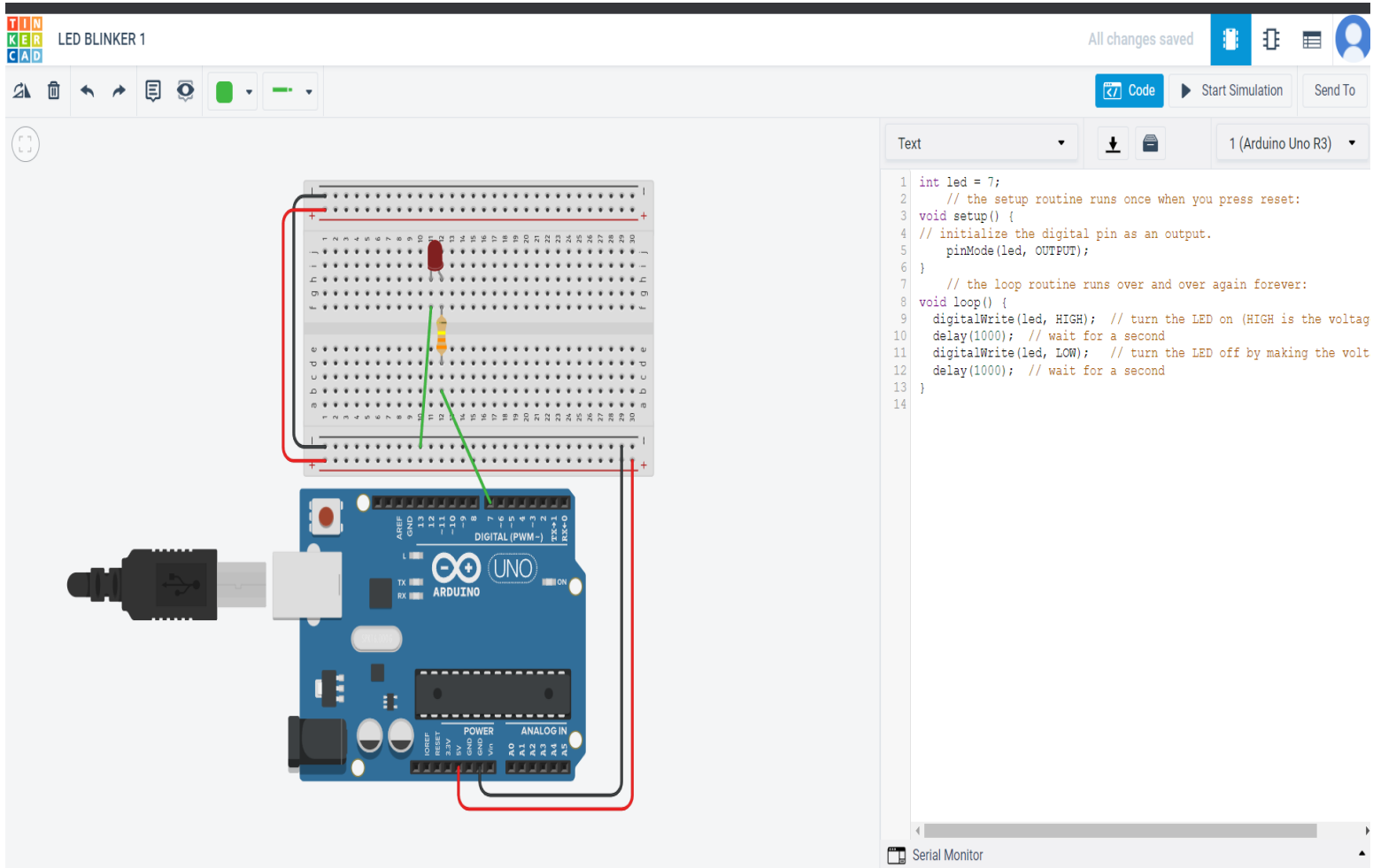
The screenshot shows the same Tinkercad interface as in the first image, but with a different code editor view. The "Blocks + Text" panel is active, and the code editor shows a C++ code block for the same LED blinking logic. The code is as follows:

```
1 // C++ code
2 //
3 void setup()
4 {
5   pinMode(LED_BUILTIN, OUTPUT);
6 }
7
8 void loop()
9 {
10  digitalWrite(LED_BUILTIN, HIGH);
11  delay(1000); // Wait for 1000 millisecond(s)
12  digitalWrite(LED_BUILTIN, LOW);
13  delay(1000); // Wait for 1000 millisecond(s)
14 }
```


2. With text program

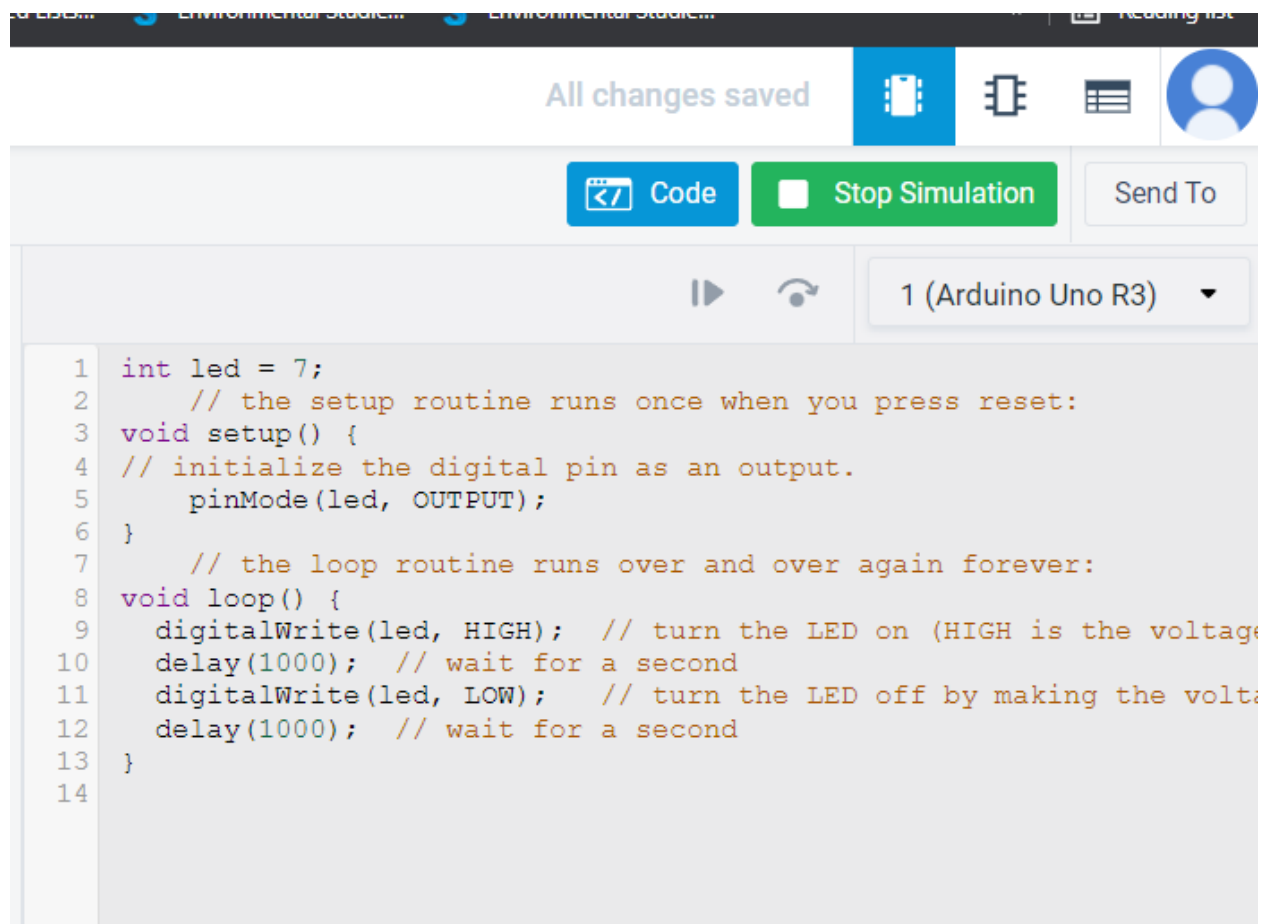


Let us try using a different pin of the Arduino – say D7. Move the red jumper lead from pin D13 to pin D7 and modify the following line near the top of the sketch:



The screenshot shows the Arduino IDE interface with a breadboard circuit and a code sketch. The breadboard circuit is connected to an Arduino Uno R3. A red LED is connected to pin D7 (labeled 7) on the Arduino. The negative terminal of the LED is connected to ground (pin D0, labeled 0). A green jumper lead connects the positive terminal of the LED to pin D7. A red jumper lead connects the positive terminal of the LED to pin D13. The Arduino is connected to a USB cable. The code sketch is as follows:

```
1 int led = 7;
2 // the setup routine runs once when you press reset:
3 void setup() {
4 // initialize the digital pin as an output.
5 pinMode(led, OUTPUT);
6 }
7 // the loop routine runs over and over again forever:
8 void loop() {
9 digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage)
10 delay(1000); // wait for a second
11 digitalWrite(led, LOW); // turn the LED off by making the voltage low
12 delay(1000); // wait for a second
13 }
14
```



The screenshot displays the Tinkercard software interface. At the top, a status bar indicates "All changes saved". Below this, there are icons for a circuit board, a microcontroller, and a list of components. A user profile icon is visible on the right. The main workspace shows a single Arduino Uno R3 board. Below the board, there is a code editor with the following C++ code:

```
1 int led = 7;
2 // the setup routine runs once when you press reset:
3 void setup() {
4 // initialize the digital pin as an output.
5   pinMode(led, OUTPUT);
6 }
7 // the loop routine runs over and over again forever:
8 void loop() {
9   digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage
10  delay(1000); // wait for a second
11  digitalWrite(led, LOW); // turn the LED off by making the voltage
12  delay(1000); // wait for a second
13 }
14
```

Result:

The controlling of LED with Aurdino Board using Tinkercard software is successfully done.

