Keyboard controlled display system

1. Aim of the project

The aim of this project is to design a basic Human–Machine Interface using an **Arduino Uno**, a **4x4 matrix keypad**, and a **16x2 LCD display**. The system reads key presses from the keypad and displays the corresponding key on the LCD screen in real-time. using the Tinkercad simulation platform for circuit design, testing and troubleshooting.

---

2. Purpose of the Project

This project demonstrates how to interface a matrix keypad and an LCD with Arduino, enabling user interaction with embedded systems. It is a foundational project for understanding input handling, user interface design and real-time display updates in microcontroller-based applications.

---

3. Application of the Project

- Digital Door Lock Systems:
  Used for entering a PIN/password to unlock a secure system.
- Access Control Systems:
  Commonly used in offices, labs, or secure zones to restrict unauthorized access.
- Vending Machines:
  Allows users to input codes for item selection.
- Home Automation Interfaces:
  To control appliances based on user inputs.
- ATM Interface Prototypes:
  Simulates how users enter PINs on ATMs.
- Menu Navigation Systems:
  Used in devices where users scroll through options using keypad input.

---

4. Components Required

S. No.	Quantity	Components
1.	1	Arduino Uno
2.	1	LCD Display
3.	1	4x4 matrix keypad
4.	1	Resistor
5.	multiple	Jumper wire
6.	1	Breadboard

---

5. Some Key Concept

4x4 Matrix Keypad:

The keypad consists of 16 keys arranged in a matrix of 4 rows and 4 columns. Each key press connects a row and a column, which is detected using scanning algorithms. The Keypad library simplifies the detection of key presses by handling row–column mapping internally.

16x2 LCD Display:

A 16x2 character LCD can display 2 lines with 16 characters per line. It is used here to show which key is being pressed. The **LiquidCrystal library** allows easy interfacing and printing of text on the LCD.

Potentiometer:

a potentiometer is connected to the LCD module to control its display contrast. The contrast of a 16x2 LCD screen can vary depending on lighting conditions and viewing angles. If the contrast is too low or too high, the characters on the display may be difficult to read.

---

6. Pin Configuration & Function

4x4 Keypad :

Keypad Pin	Connected to Arduino Pin	Function
Row1	A0	Detect key press row 1
Row2	A1	Detect key press row 2
Row3	A2	Detect key press row 3
Row4	A3	Detect key press row 4
Column1	A4	Detect key press column 1
Column2	A5	Detect key press column 2
Column3	6	Detect key press column 3
Column4	7	Detect key press column 4

16x2 LCD:

LCD Pin	Connected to Arduino Pin / Component	Function
VSS	GND	Ground
VDD	5V	Power supply
V0	Potentiometer middle Pin	Contrast control
RS	12	Register Select
RW	GND	Read/write
E	11	Enable Signal
D4	5	Data Pin 4
D5	4	Data Pin 5
D6	3	Data Pin 6
D7	2	Data Pin 7
A(LED+)	5V	Backlight Power
K(LED-)	GND	Backlight ground

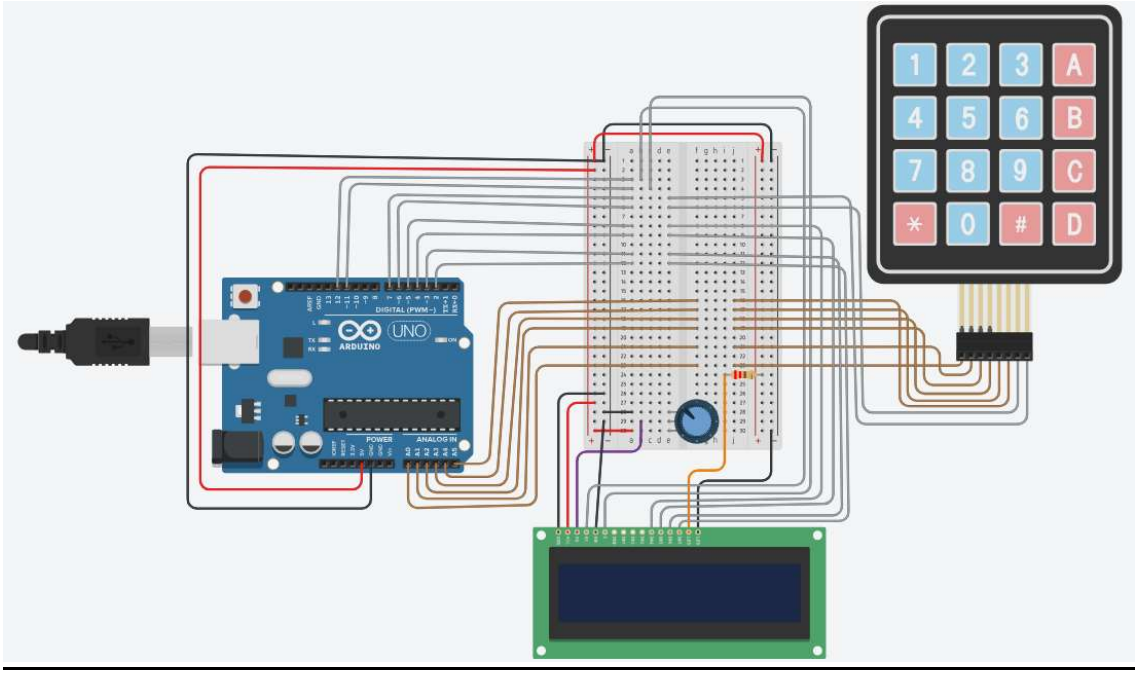
7. Mapping In Keypad

mapping refers to the process of linking the physical arrangement of the keys on the 4x4 matrix keypad to specific characters (like ‘1’,’2’,’3’,’A’) in software using a 2D array.

The keypad has 4 rows and 4 columns, making a total of 16 keys. When a key is pressed, a connection is made between a specific row and column. However, the microcontroller does not know which character that row–column combination represents unless we map it manually.

---

8. Circuit Diagram

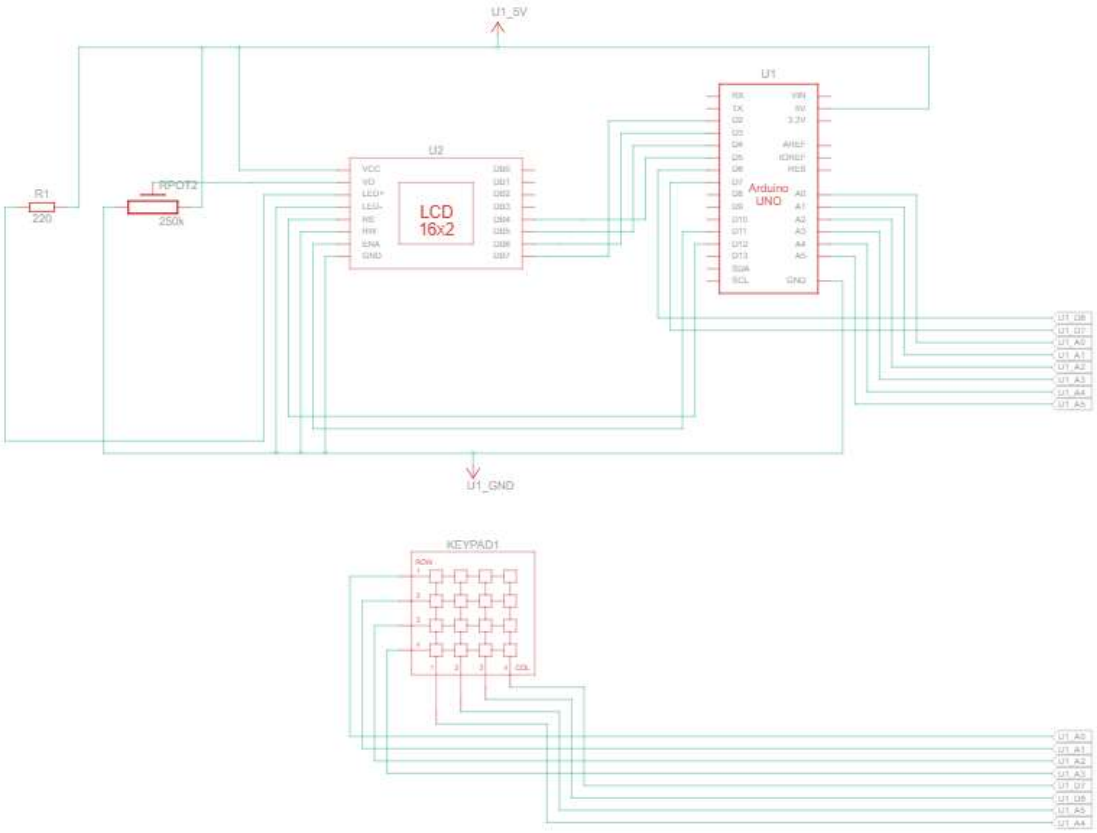


---

9. Layout(Tinkercad Simulation)

In the Tinkercad Simulation:

- Use breadboard to mount the Potentiometer.
- Use breadboard to mount the Resister.



10.Code With Explaination

```
#include <LiquidCrystal.h>              // Library to control the LCD
#include <Keypad.h>         // Library to interface with the 4x4 matrix keypad

// LCD pin configuration (RS, E, D4, D5, D6, D7)

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

// Define number of rows and columns of the keypad

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

// Define the key values based on row and column position

char keys[ROWS][COLS] = {
 {'1','2','3','A'},
 {'4','5','6','B'},
 {'7','8','9','C'},
 {'*','0','#','D'}
};
// Connect keypad row and column pins to Arduino

byte rowPins[ROWS] = {A0, A1, A2, A3};   // Row pins connected to analog
pins
byte colPins[COLS] = {A4, A5, 6, 7};     // Column pins connected to
analog/digital pins

// Initialize keypad object

Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

void setup() {
 lcd.begin(16, 2);       // Initialize 16x2 LCD
 lcd.print("Press a Key:");                  // Display default message on first line
}
void loop() {
 char key = keypad.getKey();                       // Check if a key is pressed

 if (key) {                                        // If a key is pressed (non-zero value)
   lcd.setCursor(0, 1);                  // Move cursor to second line, first column
   lcd.print("Key : ");                           // Display label
   lcd.print(key);                             // Display the pressed key
 }
}
```

- **Justification**
  - #include <LiquidCrystal.h>: Imports the library needed to control the LCD screen.
  - #include <Keypad.h>: Imports the library to interface with the matrix keypad.
  - LiquidCrystal lcd(12, 11, 5, 4, 3, 2);: Initializes the LCD with Arduino pins connected to RS, E, D4–D7.
  - char keys[ROWS][COLS]: A 2D array to map each row and column combination to a specific character (used for scanning).
  - rowPins[] and colPins[]: Arrays that define the actual Arduino pins connected to the keypad rows and columns.
  - Keypad keypad = ...: Creates the keypad object using the defined mapping and pin configurations.
  - lcd.begin(16, 2): Starts the LCD with 16 columns and 2 rows.
  - lcd.print("Press a Key:"): Displays a message on the screen when the program starts.
  - keypad.getKey() checks continuously for key presses.
  - If a key is detected, it's shown using lcd.setCursor() and lcd.print().

11.Conclusion

In this project, we successfully built a simple system using an Arduino Uno, a 4x4 keypad, and a 16x2 LCD. The system takes input from the keypad and shows the pressed key on the LCD screen. This helped us learn how to connect and use input (keypad) and output (LCD) devices with Arduino.

We used two important libraries – Keypad and LiquidCrystal – to make our work easier and to handle keypad scanning and LCD display. The project taught us important concepts like how a keypad works, how to map keys, and how to give real-time feedback to users.

This project is a good starting point for building more advanced systems like password locks, entry control panels, or simple menu systems. It helped improve our understanding of embedded systems, programming, and electronics basics in a fun and practical way.