```dart
import 'dart:async';
import 'dart:convert';
import 'dart:typed_data';
import 'package:flutter/services.dart';

import 'package:flutter/material.dart';
import 'package:flutter_bluetooth_serial/flutter_bluetooth_serial.dart';

class CreateCourse extends StatefulWidget {
  final BluetoothDevice server;

  const CreateCourse({required this.server});

  @override
  _CreateCourse createState() => new _CreateCourse();
}

class _CreateCourse extends State<CreateCourse> {

  BluetoothConnection? connection;

  final TextEditingController textEditingController =
  new TextEditingController();

  final myController1 = TextEditingController();
  final myController2 = TextEditingController();
  final myController3 = TextEditingController();
  final myController4 = TextEditingController();


  bool isConnecting = true;
  bool get isConnected => (connection?.isConnected ?? false);

  bool isDisconnecting = false;

  bool SendingData = false;

  @override
  void initState() {
    super.initState();

    BluetoothConnection.toAddress(widget.server.address).then((_connection) {
      print('Connected to the device');
      connection = _connection;
      setState(() {
        isConnecting = false;
        isDisconnecting = false;
      });

      connection!.input!.listen(_DataReceived).onDone(() {
        if (isDisconnecting) {
          print('Disconnecting locally!');
        } else {
          print('Disconnected remotely!');
        }
```

```
      });
    }).catchError((error) {
      print('Cannot connect, exception occured');
      print(error);
    });
  }

  @override
  void dispose() {
    // Avoid memory leak     if (isConnected) {
      isDisconnecting = true;
      connection?.dispose();
      connection = null;
    }
    myController1.dispose();
    myController2.dispose();
    myController3.dispose();
    myController4.dispose();

    super.dispose();
  }

  @override
  Widget build(BuildContext context) {

    final serverName = widget.server.name ?? "Unknown";
    return Scaffold(
        appBar: AppBar(
            backgroundColor: Colors.green,
            title: (isConnecting
                ? Text('Connecting to ' + serverName + '...', style: TextStyle(color:
Colors.black))
                : isConnected
                ? Text('Live connection with ' + serverName, style: TextStyle(color:
Colors.black))
                : Text('Disconnected from ' + serverName, style: TextStyle(color:
Colors.black)))),
        backgroundColor: Colors.green,
        body: Center(
            child: SingleChildScrollView(
                child: Column(
                    children: <Widget>[
                      Container(
                        child: Text(
                          'Enter the desired PUTR settings here ',
                          style: TextStyle(
                            fontWeight: FontWeight.bold,
                            fontSize: 16,
                            color: Colors.black,
                            letterSpacing: 2,
                          ),
                        ),
                      ),
                      Padding(
                        padding: const EdgeInsets.only(top: 30),
```

```
                      child: Container(
                        width: 350,
                        height: 60,
                        decoration: BoxDecoration(
                            color: Colors.green,
                            borderRadius: BorderRadius.all(Radius.circular(25)),
                            border: Border.all(color: Colors.black)
                        ),
                        child: TextFormField(
                            inputFormatters: [LengthLimitingTextInputFormatter(1)],
                            controller: myController1,
                            textAlign: TextAlign.center,
                            keyboardType: TextInputType.number,
                            decoration: InputDecoration(
                                border: InputBorder.none,
                                labelText: 'Top Left',
                                labelStyle: TextStyle(
                                  color: Colors.black,
                                  fontWeight: FontWeight.bold,
                                  fontSize: 30,
                                  letterSpacing: 3,
                                ),
                                floatingLabelBehavior: FloatingLabelBehavior.never,
                                contentPadding:
                                EdgeInsets.only(left: 15, bottom: 18, top: 11,
right: 15)
                            ),
                            style: TextStyle(
                              color: Colors.black,
                              fontWeight: FontWeight.bold,
                              fontSize: 30,
                              letterSpacing: 3,
                            )
                        ),
                      ),
                    ),
                    Padding(
                      padding: const EdgeInsets.only(top: 30),
                      child: Container(
                        width: 350,
                        height: 60,
                        decoration: BoxDecoration(
                            color: Colors.green,
                            borderRadius: BorderRadius.all(Radius.circular(25)),
                            border: Border.all(color: Colors.black)
                        ),
                        child: TextFormField(
                            inputFormatters: [LengthLimitingTextInputFormatter(1)],
                            controller: myController2,
                            textAlign: TextAlign.center,
                            keyboardType: TextInputType.number,
                            decoration: InputDecoration(
                                border: InputBorder.none,
                                labelText: 'Top Right',
                                labelStyle: TextStyle(
```

```
                                color: Colors.black,
                                fontWeight: FontWeight.bold,
                                fontSize: 30,
                                letterSpacing: 3,
                              ),
                              floatingLabelBehavior: FloatingLabelBehavior.never,
                              contentPadding:
                              EdgeInsets.only(left: 15, bottom: 18, top: 11,
right: 15)
                          ),
                          style: TextStyle(
                            color: Colors.black,
                            fontWeight: FontWeight.bold,
                            fontSize: 30,
                            letterSpacing: 3,
                          )
                      ),
                    ),
                  ),
                  Padding(
                    padding: const EdgeInsets.only(top: 30),
                    child: Container(
                      width: 350,
                      height: 60,
                      decoration: BoxDecoration(
                          color: Colors.green,
                          borderRadius: BorderRadius.all(Radius.circular(25)),
                          border: Border.all(color: Colors.black)
                      ),
                      child: TextFormField(
                          inputFormatters: [LengthLimitingTextInputFormatter(1)],
                          controller: myController3,
                          textAlign: TextAlign.center,
                          keyboardType: TextInputType.number,
                          decoration: InputDecoration(
                              border: InputBorder.none,
                              labelText: 'Bottom Left',
                              labelStyle: TextStyle(
                                color: Colors.black,
                                fontWeight: FontWeight.bold,
                                fontSize: 30,
                                letterSpacing: 3,
                              ),
                              floatingLabelBehavior: FloatingLabelBehavior.never,
                              contentPadding:
                              EdgeInsets.only(left: 15, bottom: 18, top: 11,
right: 15)
                          ),
                          style: TextStyle(
                            color: Colors.black,
                            fontWeight: FontWeight.bold,
                            fontSize: 30,
                            letterSpacing: 3,
                          )
                      ),
```

```
                    ),
                  ),
                  Padding(
                    padding: const EdgeInsets.only(top: 30),
                    child: Container(
                      width: 350,
                      height: 60,
                      decoration: BoxDecoration(
                          color: Colors.green,
                          borderRadius: BorderRadius.all(Radius.circular(25)),
                          border: Border.all(color: Colors.black)
                      ),
                      child: TextFormField(
                          inputFormatters: [LengthLimitingTextInputFormatter(1)],
                          controller: myController4,
                          textAlign: TextAlign.center,
                          keyboardType: TextInputType.number,
                          decoration: InputDecoration(
                              border: InputBorder.none,
                              labelText: 'Bottom Right',
                              labelStyle: TextStyle(
                                color: Colors.black,
                                fontWeight: FontWeight.bold,
                                fontSize: 30,
                                letterSpacing: 3,
                              ),
                              floatingLabelBehavior: FloatingLabelBehavior.never,
                              contentPadding:
                              EdgeInsets.only(left: 15, bottom: 18, top: 11,
right: 15)
                          ),
                          style: TextStyle(
                            color: Colors.black,
                            fontWeight: FontWeight.bold,
                            fontSize: 30,
                            letterSpacing: 3,
                          )
                      ),
                    ),
                  ),
                  Padding(
                      padding: const EdgeInsets.only(top: 70)
                  ),
                  Row(
                    mainAxisAlignment: MainAxisAlignment.center,

                    children: <Widget>[
                      RaisedButton(
                        elevation: 5.0,
                        onPressed: ()
                        {


_sendData(myController1.text+myController2.text+myController3.text+myController4.text
);
```

```
                        },
                        shape: RoundedRectangleBorder(

                          borderRadius: BorderRadius.circular(30.0),
                          side: BorderSide(color: Colors.black),
                        ),
                        color: Colors.green,
                        child: Text(
                          'Set',
                          style: TextStyle(
                            fontWeight: FontWeight.bold,
                            fontSize: 30,
                            color: Colors.black,
                            letterSpacing: 3,
                          ),
                        ),
                      ),
                    ],
                  ),

                ]
              )
          )

      )

  );
}

void _DataReceived(Uint8List data) {
  // Allocate buffer
  int backspacesCounter = 0;
  data.forEach((byte) {
    if (byte == 8 || byte == 127) {
      backspacesCounter++;
    }
  });
  Uint8List buffer = Uint8List(data.length - backspacesCounter);
  int bufferIndex = buffer.length;

  //backspace control
  backspacesCounter = 0;
  for (int i = data.length - 1; i >= 0; i--) {
    if (data[i] == 8 || data[i] == 127) {
      backspacesCounter++;
    } else {
      if (backspacesCounter > 0) {
        backspacesCounter--;
      } else {
        buffer[--bufferIndex] = data[i];
      }
    }
  }
```

```
    String dataString = String.fromCharCodes(buffer);
    if(dataString == "done")
      {
        SendingData = false;
      }
    print(dataString);

  }

  void _sendData(String text) async {
    text = text.trim();
    textEditingController.clear();

    if (text.length > 0) {
      try {
        connection!.output.add(Uint8List.fromList(utf8.encode(text + "\r\n")));
        await connection!.output.allSent;

      } catch (e) {
        // Ignore error, but notify state
        setState(() {});
      }
    }
  }
}
```