Presentation

BattleBlade

CSN-254 Project

PRESENTED TO

Prof Sandeep Kumar Garg

PRESENTED BY

Group-4

CONTENTS

TEAM DETAILS AND CONTRIBUTION

MEET SINDHAV
22114053
meet_js@cs.iitr.ac.in
+91-7984221079

Contributions:

Worked on gameplay design, modifications, character design and UI of the game. Also, integrated audio and visual assets for better user experience.

MOHAMMED HAAZIQ JAMAL
22114055
mohammed_hj@cs.iitr.ac.in
+91-8126061554

Contributions:

Controller application development specifically in establishing connection between game and controller via web sockets and worked on controller's UI changes.

MUNDADA ADITYA MAHESH
22114058
mundada_am@cs.iitr.ac.in
+91-7720079142

Contributions:

Working in game mechanics, implementing attack and movement features, adding special abilities to players in the game and mapping websocket data recieved to required game data.

NAYAN RATAN KAKADE
22114060
nayan_rk@cs.iitr.ac.in
+91-7420936972

Contribution :

Developing the welcome page and IPAddress key page of the controller application, working on websocket connection on controller side, and integration of game and controller.

POLASA HARIHARAN
22114068
polasa_h@cs.iitr.ac.in
+91-7670839921

Contribution :

Controller application front-end development (including UI design) and integration with web-sockets, game mechanics design.

ROOPAM TANEJA
22125030
roopam_t@cs.iitr.ac.in
+91-9810788629

Contribution :

Adding websocket functionality to game controls while adding necessary features for ensuring seamless integration with app. Also contributed in adding characters and designing different screens in the game.

INTRODUCTION

Experience the thrill of multiplayer gaming without the need for expensive consoles or specialized equipment! Introducing “BATTLEBLADE“ – our innovative game coupled with a smartphone controller. It is designed to deliver high-end console-style entertainment directly to your PC and smartphone. With our specially crafted smartphone controller app integrated with our game, everyone can now enjoy an engaging multiplayer without having to spend a fortune.

FEATURES

Multiplayer Gameplay

The system enables two-player gameplay where smartphones serve as controllers.

Real-Time Connection

WebSockets for real-time interaction between the game server and smartphones.

Character Selection

Users can select characters from a grid displayed on the screen.

Various attack types

Different attacks and superpowers are available to players.

CODE SNIPPETS

Connection through Websockets

Websockets and asyncio libraries of python are utilised to achieve this functionality.

```python
async def websocket_server(websocket, path):
    print('connection established')
    global player1_connected
    global player2_connected
    global custom_data

    if trigger.is_set():
        return

    asyncio.create_task(end_connection(websocket))

    try:
        async for message in websocket:

            if is_valid_json(message) == 0:
                await websocket.send(message)
                return

            if trigger.is_set():
                return

            event = json.loads(message)
            player = event.get("player")
            key = event.get("control")
            print(f"{player} : {key}")
            if player == 1:
                player1_connected = True
            elif player == 2:
                player2_connected = True

            if select_screen:
                move_on_grid(player, key)

            elif run and not select_screen:
                custom_data = np.array(
                    [
                        (key == 1) & (player == 1),
```

Encryption of IP Address

Variant Beaufort of Vigenère cipher is used for encoding and decoding the IP address.

```
def encrypt(plain_text, key, alphabet):
    # Variant Beaufort cipher - A variant of Vigenere Cipher : encryption

    cipher_text = ""
    key_index = 0

    for char in plain_text:
        if char in alphabet:
            shift = alphabet.index(key[key_index % len(key)])
            cipher_text += alphabet[(alphabet.index(char) - shift) % len(alphabet)]
            key_index += 1
        else:
            cipher_text += char

    return cipher_text


def decrypt(cipher_text, key, alphabet):
    # Variant Beaufort cipher - A variant of Vigenere Cipher : decryption

    plain_text = ""
    key_index = 0

    for char in cipher_text:
        if char in alphabet:
            shift = alphabet.index(key[key_index % len(key)])
            plain_text += alphabet[(alphabet.index(char) + shift) % len(alphabet)]
            key_index += 1
        else:
            plain_text += char

    return plain_text
```

Handling Connections on Controller App

```dart
//Websocketpart//
late Timer _reconnectTimer;
late WebSocketChannel _channel;

@override
void initState() {
  super.initState();
  _connectWebSocket();
  _startReconnectTimer();
}
void _connectWebSocket() {
String ipaddress = decryptedText;
print(ipaddress);
try {
  _channel = WebSocketChannel.connect(
    Uri.parse('ws://$ipaddress:8765'),
  );
} catch (e) {
  print('Error connecting to WebSocket: $e');

}

void _startReconnectTimer() {
  _reconnectTimer = Timer.periodic(Duration(seconds: 30), (_) {
    _channel.sink.close();
    _connectWebSocket();
  }); // Timer.periodic
}

void _sendControl(int player, int control) {
  print("pressed 1");

  final message = jsonEncode({
    'player': player,
    'control': control,
  });
  _channel.sink.add(message);
}

void _startSendingRepeatedly(int player, int control) {
  print("pressed 2");
      _reconnectTimer = Timer.periodic(const Duration(milliseconds: 10), (_) {
    _sendControl(widget.player, widget.control);
  }); // Timer.periodic
}

void _stopSending() {
  _reconnectTimer.cancel();
}
```

Controller App UI

```dart
class homepage extends StatelessWidget {
@override
Widget build(BuildContext context) {

  double deviceWidth = MediaQuery.of(context).size.width;

  return Scaffold(
    backgroundColor: Color.fromARGB(255, 75, 62, 80),
    body: Container(
      child: Column(
        mainAxisAlignment: MainAxisAlignment.start,
        crossAxisAlignment: CrossAxisAlignment.center,
        children: [
          Align(
            alignment: Alignment.centerRight,
            child: Container(
              height: deviceWidth *
                  0.9,
              width: deviceWidth * 1.0,
              decoration: BoxDecoration(
                image: DecorationImage(
                  image: AssetImage(
                      'C:\\Users\\dell\\StudioProjects\\untitled\\build\\flutter_assets\\top.png'),  // AssetImage
                  fit: BoxFit.fill,
                ), // DecorationImage
              ), // BoxDecoration
            ), // Container
          ), // Align
          Container(
            decoration: BoxDecoration(
              gradient: LinearGradient(
                begin: Alignment.topLeft,
                end: Alignment.bottomRight,
                colors: [
                  Color.fromARGB(255, 56, 49, 68), // Top-left color
                  Color.fromARGB(255, 49, 44, 60), // Bottom-left color
                  Color.fromARGB(255, 59, 51, 70), // Top-right color
                  Color.fromARGB(255, 53, 46, 63), // Bottom-right color
                ],
              ), // LinearGradient
            ), // BoxDecoration
            child: Column(
              children: [
                Container(
                  child: Text(
                    "BATTLEBLADE",
                    style: TextStyle(
                      color: Color.fromARGB(255, 207, 128, 86),
                      fontSize: 40,
                      fontWeight: FontWeight.bold,
                      fontFamily: GoogleFonts.silkscreen().fontFamily,
                    ), // TextStyle
                  ), // Text
                  margin: EdgeInsets.only(bottom: 10.0, top: 10.0),
```

DPAD Buttons UI

```dart
Widget build(BuildContext context) {
  return Container(
    margin: EdgeInsets.fromLTRB(0, 30, 20, 10),
    height: containerSize - 40, width: containerSize - 40,
    decoration: BoxDecoration(
      shape: BoxShape.circle, color: Colors.white.withOpacity(0.7),
      boxShadow: [
        BoxShadow(
          color: Colors.black.withOpacity(0.5), offset: Offset(-1, 1),
          blurRadius: 4, spreadRadius: 2,
        ), // BoxShadow
        BoxShadow(
          color: Colors.white.withOpacity(0.3),
          offset: Offset(1, -1),
          blurRadius: 5, spreadRadius: -4,
        ), // BoxShadow
      ],
    ), // BoxDecoration
    child: Column(
      mainAxisAlignment: MainAxisAlignment.spaceEvenly,
      children: [
        Row(
          mainAxisAlignment: MainAxisAlignment.spaceEvenly,
          children: [
            DPadButton(
              index: 0,
              imagePath:'C:\\Users\\dell\\StudioProjects\\untitled\\build\\flutter_assets\\up.png',
              player: 1, control: 3,
            ), // DPadButton
          ],
        ), // Row
        Row(
          mainAxisAlignment: MainAxisAlignment.spaceEvenly,
          children: [
            DPadButton(
                index: 1, imagePath:'C:\\Users\\dell\\StudioProjects\\untitled\\build\\flutter_assets\\left.png', player: 1, control: 1,), // DPadButton
            SizedBox(width: 40),
            DPadButton(
                index: 2, imagePath:'C:\\Users\\dell\\StudioProjects\\untitled\\build\\flutter_assets\\right.png', player: 1, control: 2,), // DPadButton
          ],
        ), // Row
        Row(
          mainAxisAlignment: MainAxisAlignment.spaceEvenly,
          children: [
            DPadButton(
                index: 3, imagePath:'C:\\Users\\dell\\StudioProjects\\untitled\\build\\flutter_assets\\down.png', player: 1, control: 25,), // DPadButton
          ],
        ), // Row
      ],
    ), // Column
  ); // Container
}
```

```python
def attack(self, target,attack_type):
  attacking_rect = pygame.Rect(self.rect.centerx - (2 * self.rect.width * self.flip), self.rect.y, 2 * self.rect.width, self.rect.height - 5)
  if attack_type==1:
        if self.attack_cooldown == 0:
        #execute attack
            self.attacking = True
            self.attack_sound.play()
            attacking_rect = pygame.Rect(self.rect.centerx - (2 * self.rect.width * self.flip), self.rect.y, 2 * self.rect.width, self.rect.height - 5)
            if attacking_rect.colliderect(target.rect) and not target.invincible:
              target.health -= 5
              target.hit = True
  elif attack_type == 2 and self.attack2_cooldown == 0:
        if self.attack_cooldown == 0:
        #execute attack
            self.attacking = True
            self.attack_sound.play()
            attacking_rect = pygame.Rect(self.rect.centerx - (2 (parameter) target: Any .flip), self.rect.y, 2 * self.rect.width, self.rect.height - 5)
            self.attack2_cooldown = 1500
            if attacking_rect.colliderect(target.rect) and not target.invincible:
              target.health -= 20
              target.hit = True
```

Algorithm for Gameplay Mechanics

```python
if self.attack2_cooldown>0:
  self.attack2_cooldown-=1
#can only perform other actions if not currently attacking
if self.attacking == False and self.alive == True and round_over == False:
  #check player 1 controls
  if self.player == 1:
    #movement
    if custom_data[0]:
      dx = -SPEED
      self.running = True
    if custom_data[1]:
      dx = SPEED
      self.running = True
    #jump
    if custom_data[2] and self.jump == False:
      self.vel_y = -35
      self.jump = True
    #attack
    if custom_data[3] or custom_data[4]:

      #determine which attack type was used
      if custom_data[3]:
        self.attack_type = 1
      if custom_data[4]:
        self.attack_type = 2
      self.attack(target,self.attack_type)
    if custom_data[10] and self.supuse==0 and self.health<50:
      self.health=50
      self.health_boost=True
      self.health_boost_endtime=pygame.time.get_ticks() + self.health_boost_duration
      self.supuse=1
    if custom_data[12] and self.supuse==0:
      self.invincible=True
      self.invincible_end_time = pygame.time.get_ticks() + self.invincible_duration
      self.supuse=1
```

GAME SNAPSHOTS

Home Screen


BATTLEBLADE
GET READY FOR THE ULTIMATE BATTLE
ENTER CODE TO CONNECT
ppgefvrmzolt

Selection Screen

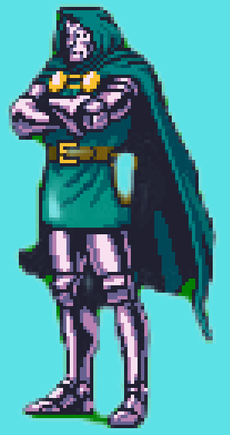
Both players browse and select their respective characters.

CHARACTERS

PSYDUCK

SUPERMAN

BATMAN



DOOM

BROLY

WIZARD

WOLVERINE

WARRIOR

Game Screen

Multiple backgrounds are provided to enhance gaming experience.

Game Screen

Multiple backgrounds are provided to enhance gaming experience.

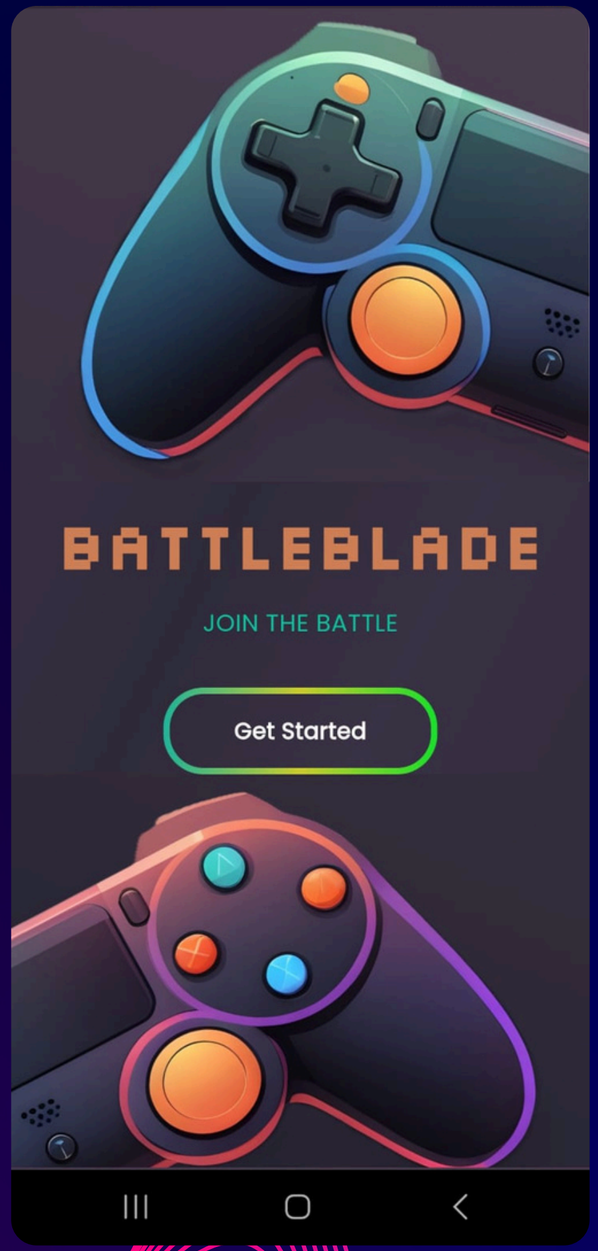

P1: 1
P2: 2

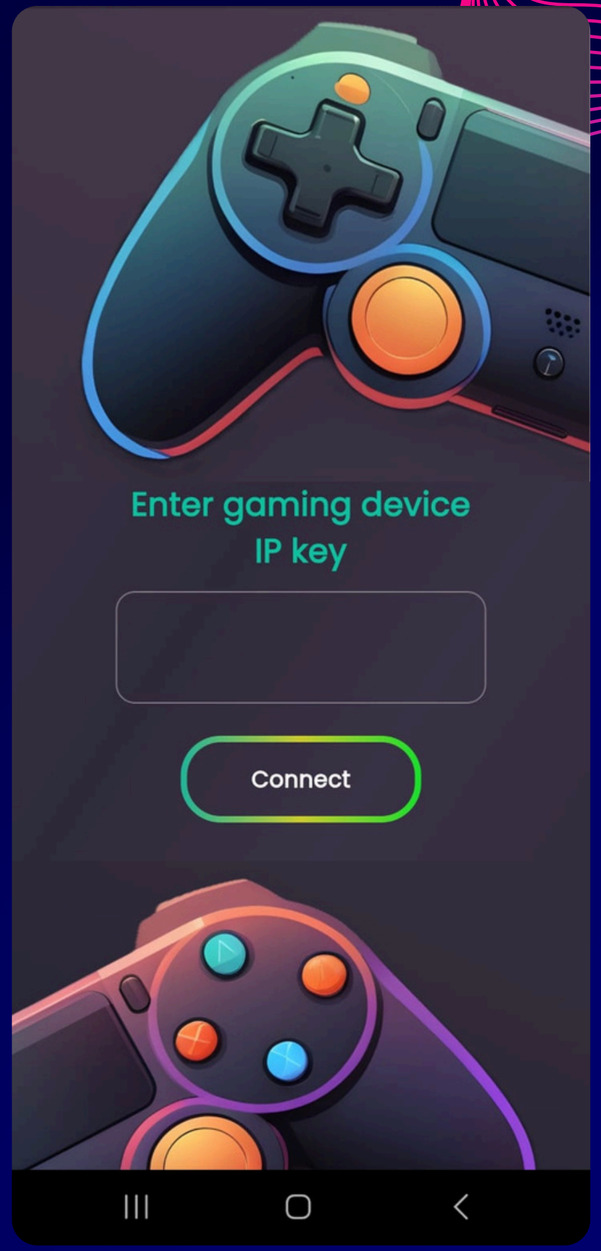

BattleBlade
P1: 0
P2: 2

APP SNAPSHOTS

HOME

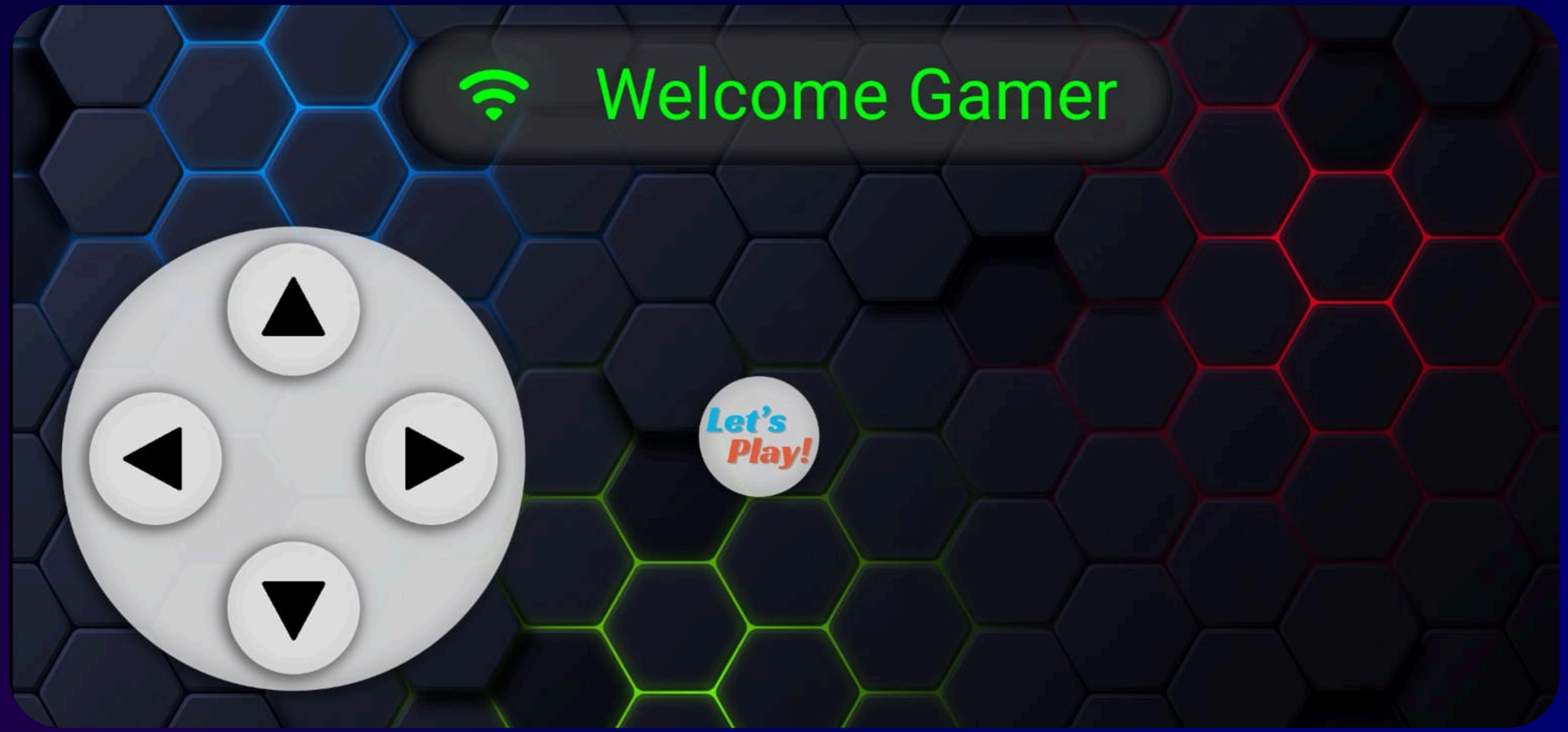
This is the app’s landing page. The key displayed on the game screen must be entered here for establishing a connection





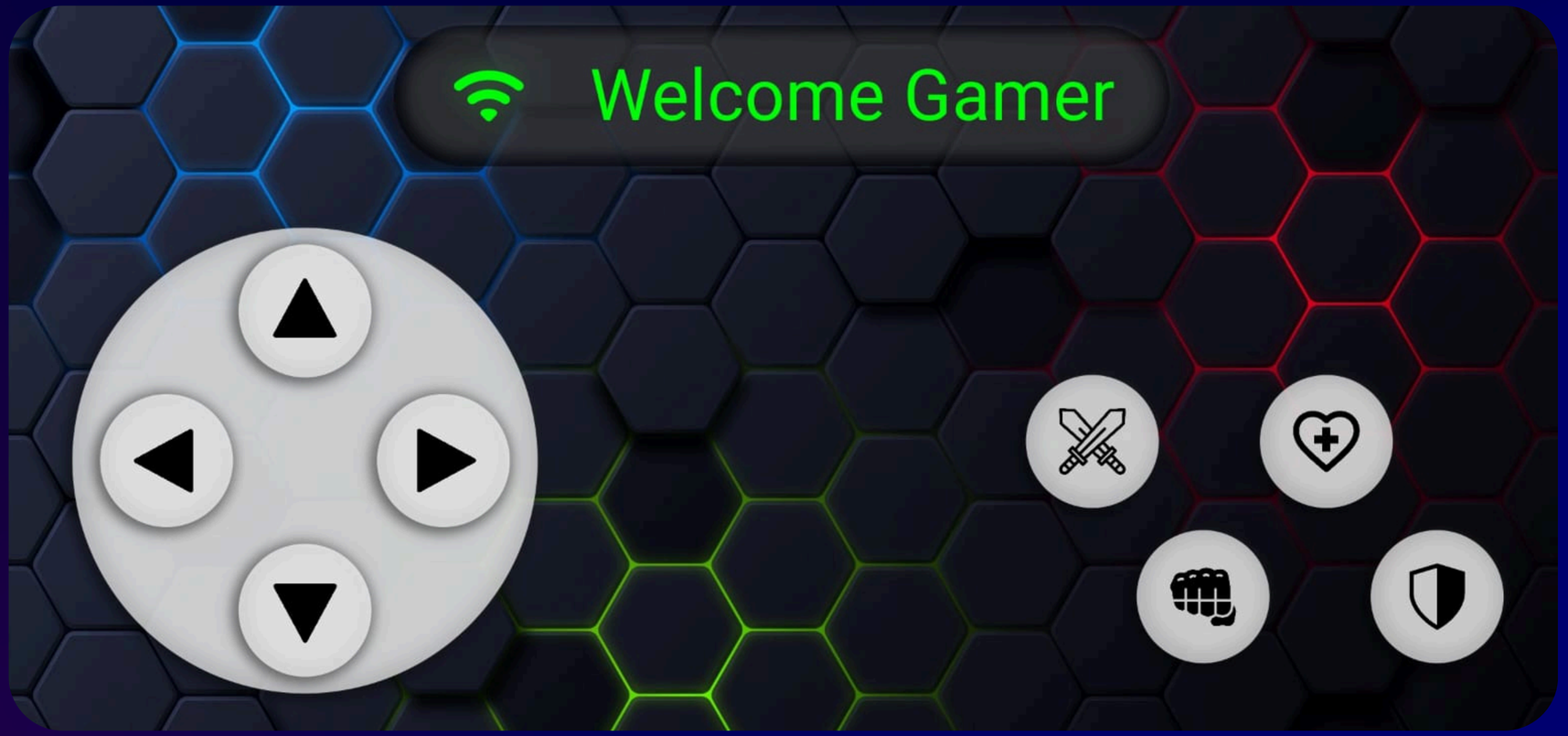
SELECTION PAGE

Navigate character menu using DPAD buttons and lock a character by clicking on ‘Let’s Play’ button



CONTROLLER PAGE

Each button is mapped to specific actions in the game



ACKNOWLEDGMENT

We would like to express our immense gratitude to all the individuals who have helped and supported us throughout the project. We are thankful to our course instructor, Prof. Sandeep Kumar for his support from initial advice, and encouragement, till the completion of this project. We would also like to thank all the Teaching Assistants (TAs) who were always present in our practical sessions for assistance. A special acknowledgement goes to our classmates who helped us by exchanging interesting ideas and sharing their valuable experiences.

Thank You