

# Game Analysis

## Delivery 1

Hang Xue, Oriol Via, Marta Llurba



# Introduction

- Code Structure
- Analyze data
  - User Activity
  - Player Demographics
  - Purchases
  - User Engagement
  - Session analysis



# Code Structure

```
using System;
using System.Collections;
using UnityEngine;
using UnityEngine.Networking;

public class DataTransmission : MonoBehaviour
{
    uint currentUserId;
    uint currentSessionId;
    uint currentPurchaseId;

    private void OnEnable()
    {
        Simulator.OnNewPlayer += HandleNewPlayer;
        Simulator.OnNewSession += HandleNewSession;
        Simulator.OnEndSession += HandleEndSession;
        Simulator.OnBuyItem += HandleBuyItem;
    }

    private void OnDisable()
    {
        Simulator.OnNewPlayer -= HandleNewPlayer;
        Simulator.OnNewSession -= HandleNewSession;
        Simulator.OnEndSession -= HandleEndSession;
        Simulator.OnBuyItem -= HandleBuyItem;
    }

    private void HandleNewPlayer(string name, string country, DateTime date)
    {
        StartCoroutine(UploadPlayer(name, country, date));
    }

    private void HandleNewSession(DateTime date)
    {
        StartCoroutine(UploadStartSession(date));
    }

    private void HandleEndSession(DateTime date)
    {
        StartCoroutine(UploadEndSession(date));
    }

    private void HandleBuyItem(int item, DateTime date)
    {
        StartCoroutine(UploadItem(item, date));
    }
}
```

```
IEnumerator UploadPlayer(string name, string country, DateTime date)
{
    WWWForm form = new WWWForm();
    form.AddField("name", name);
    form.AddField("country", country);
    form.AddField("date", date.ToString("yyyy-MM-dd HH:mm:ss"));

    using (UnityWebRequest www = UnityWebRequest.Post("https://citilumnes.upc.es/-hangx/Player_Data.php", form))
    {
        yield return www.SendWebRequest();

        if (www.result != UnityWebRequest.Result.Success)
        {
            UnityEngine.Debug.LogError("Player data upload failed: " + www.error);
        }
        else
        {
            string answer = www.downloadHandler.text.Trim(new char[] { '\u0000', '\u0001', '\u0002', '\u0003', '\u0004', '\u0005', '\u0006', '\u0007', '\u0008', '\u0009', '\u000A', '\u000B', '\u000C', '\u000D', '\u000E', '\u000F', '\u0010', '\u0011', '\u0012', '\u0013', '\u0014', '\u0015', '\u0016', '\u0017', '\u0018', '\u0019', '\u001A', '\u001B', '\u001C', '\u001D', '\u001E', '\u001F', '\u0020', '\u0021', '\u0022', '\u0023', '\u0024', '\u0025', '\u0026', '\u0027', '\u0028', '\u0029', '\u002A', '\u002B', '\u002C', '\u002D', '\u002E', '\u002F', '\u0030', '\u0031', '\u0032', '\u0033', '\u0034', '\u0035', '\u0036', '\u0037', '\u0038', '\u0039', '\u003A', '\u003B', '\u003C', '\u003D', '\u003E', '\u003F', '\u0040', '\u0041', '\u0042', '\u0043', '\u0044', '\u0045', '\u0046', '\u0047', '\u0048', '\u0049', '\u004A', '\u004B', '\u004C', '\u004D', '\u004E', '\u004F', '\u0050', '\u0051', '\u0052', '\u0053', '\u0054', '\u0055', '\u0056', '\u0057', '\u0058', '\u0059', '\u005A', '\u005B', '\u005C', '\u005D', '\u005E', '\u005F', '\u0060', '\u0061', '\u0062', '\u0063', '\u0064', '\u0065', '\u0066', '\u0067', '\u0068', '\u0069', '\u006A', '\u006B', '\u006C', '\u006D', '\u006E', '\u006F', '\u0070', '\u0071', '\u0072', '\u0073', '\u0074', '\u0075', '\u0076', '\u0077', '\u0078', '\u0079', '\u007A', '\u007B', '\u007C', '\u007D', '\u007E', '\u007F', '\u0080', '\u0081', '\u0082', '\u0083', '\u0084', '\u0085', '\u0086', '\u0087', '\u0088', '\u0089', '\u008A', '\u008B', '\u008C', '\u008D', '\u008E', '\u008F', '\u0090', '\u0091', '\u0092', '\u0093', '\u0094', '\u0095', '\u0096', '\u0097', '\u0098', '\u0099', '\u009A', '\u009B', '\u009C', '\u009D', '\u009E', '\u009F', '\u00A0', '\u00A1', '\u00A2', '\u00A3', '\u00A4', '\u00A5', '\u00A6', '\u00A7', '\u00A8', '\u00A9', '\u00AA', '\u00AB', '\u00AC', '\u00AD', '\u00AE', '\u00AF', '\u00B0', '\u00B1', '\u00B2', '\u00B3', '\u00B4', '\u00B5', '\u00B6', '\u00B7', '\u00B8', '\u00B9', '\u00BA', '\u00BB', '\u00BC', '\u00BD', '\u00BE', '\u00BF', '\u00C0', '\u00C1', '\u00C2', '\u00C3', '\u00C4', '\u00C5', '\u00C6', '\u00C7', '\u00C8', '\u00C9', '\u00CA', '\u00CB', '\u00CC', '\u00CD', '\u00CE', '\u00CF', '\u00D0', '\u00D1', '\u00D2', '\u00D3', '\u00D4', '\u00D5', '\u00D6', '\u00D7', '\u00D8', '\u00D9', '\u00DA', '\u00DB', '\u00DC', '\u00DD', '\u00DE', '\u00DF', '\u00E0', '\u00E1', '\u00E2', '\u00E3', '\u00E4', '\u00E5', '\u00E6', '\u00E7', '\u00E8', '\u00E9', '\u00EA', '\u00EB', '\u00EC', '\u00ED', '\u00EE', '\u00EF', '\u00F0', '\u00F1', '\u00F2', '\u00F3', '\u00F4', '\u00F5', '\u00F6', '\u00F7', '\u00F8', '\u00F9', '\u00FA', '\u00FB', '\u00FC', '\u00FD', '\u00FE', '\u00FF' });
            if (uint.TryParse(answer, out uint parsedId) && parsedId > 0)
            {
                currentUserId = parsedId;
                CallbackEvents.OnAddPlayerCallback.Invoke(currentUserId);
            }
            else
            {
                UnityEngine.Debug.LogError("Invalid user ID received: " + answer);
            }
        }
    }
}
```

```
IEnumerator UploadStartSession(DateTime date)
{
    WWWForm form = new WWWForm();
    form.AddField("user_ip", currentUserId.ToString());
    form.AddField("start_session", date.ToString("yyyy-MM-dd HH:mm:ss"));
    string url = "https://citilumnes.upc.es/-hangx/Session_Data.php";
    UnityWebRequest www = UnityWebRequest.Post(url, form);
    yield return www.SendWebRequest();

    if (www.result != UnityWebRequest.Result.Success)
    {
        string answer = www.downloadHandler.text.Trim(new char[] { '\u0000', '\u0001', '\u0002', '\u0003', '\u0004', '\u0005', '\u0006', '\u0007', '\u0008', '\u0009', '\u000A', '\u000B', '\u000C', '\u000D', '\u000E', '\u000F', '\u0010', '\u0011', '\u0012', '\u0013', '\u0014', '\u0015', '\u0016', '\u0017', '\u0018', '\u0019', '\u001A', '\u001B', '\u001C', '\u001D', '\u001E', '\u001F', '\u0020', '\u0021', '\u0022', '\u0023', '\u0024', '\u0025', '\u0026', '\u0027', '\u0028', '\u0029', '\u002A', '\u002B', '\u002C', '\u002D', '\u002E', '\u002F', '\u0030', '\u0031', '\u0032', '\u0033', '\u0034', '\u0035', '\u0036', '\u0037', '\u0038', '\u0039', '\u003A', '\u003B', '\u003C', '\u003D', '\u003E', '\u003F', '\u0040', '\u0041', '\u0042', '\u0043', '\u0044', '\u0045', '\u0046', '\u0047', '\u0048', '\u0049', '\u004A', '\u004B', '\u004C', '\u004D', '\u004E', '\u004F', '\u0050', '\u0051', '\u0052', '\u0053', '\u0054', '\u0055', '\u0056', '\u0057', '\u0058', '\u0059', '\u005A', '\u005B', '\u005C', '\u005D', '\u005E', '\u005F', '\u0060', '\u0061', '\u0062', '\u0063', '\u0064', '\u0065', '\u0066', '\u0067', '\u0068', '\u0069', '\u006A', '\u006B', '\u006C', '\u006D', '\u006E', '\u006F', '\u0070', '\u0071', '\u0072', '\u0073', '\u0074', '\u0075', '\u0076', '\u0077', '\u0078', '\u0079', '\u007A', '\u007B', '\u007C', '\u007D', '\u007E', '\u007F', '\u0080', '\u0081', '\u0082', '\u0083', '\u0084', '\u0085', '\u0086', '\u0087', '\u0088', '\u0089', '\u008A', '\u008B', '\u008C', '\u008D', '\u008E', '\u008F', '\u0090', '\u0091', '\u0092', '\u0093', '\u0094', '\u0095', '\u0096', '\u0097', '\u0098', '\u0099', '\u009A', '\u009B', '\u009C', '\u009D', '\u009E', '\u009F', '\u00A0', '\u00A1', '\u00A2', '\u00A3', '\u00A4', '\u00A5', '\u00A6', '\u00A7', '\u00A8', '\u00A9', '\u00AA', '\u00AB', '\u00AC', '\u00AD', '\u00AE', '\u00AF', '\u00B0', '\u00B1', '\u00B2', '\u00B3', '\u00B4', '\u00B5', '\u00B6', '\u00B7', '\u00B8', '\u00B9', '\u00BA', '\u00BB', '\u00BC', '\u00BD', '\u00BE', '\u00BF', '\u00C0', '\u00C1', '\u00C2', '\u00C3', '\u00C4', '\u00C5', '\u00C6', '\u00C7', '\u00C8', '\u00C9', '\u00CA', '\u00CB', '\u00CC', '\u00CD', '\u00CE', '\u00CF', '\u00D0', '\u00D1', '\u00D2', '\u00D3', '\u00D4', '\u00D5', '\u00D6', '\u00D7', '\u00D8', '\u00D9', '\u00DA', '\u00DB', '\u00DC', '\u00DD', '\u00DE', '\u00DF', '\u00E0', '\u00E1', '\u00E2', '\u00E3', '\u00E4', '\u00E5', '\u00E6', '\u00E7', '\u00E8', '\u00E9', '\u00EA', '\u00EB', '\u00EC', '\u00ED', '\u00EE', '\u00EF', '\u00F0', '\u00F1', '\u00F2', '\u00F3', '\u00F4', '\u00F5', '\u00F6', '\u00F7', '\u00F8', '\u00F9', '\u00FA', '\u00FB', '\u00FC', '\u00FD', '\u00FE', '\u00FF' });
            if (uint.TryParse(answer, out uint parsedId) && parsedId > 0)
            {
                currentSessionId = parsedId;
                CallbackEvents.OnNewSessionCallback.Invoke(currentSessionId);
            }
            else
            {
                UnityEngine.Debug.LogError("Invalid session ID received: " + answer);
            }
        }
    }
    else
    {
        UnityEngine.Debug.LogError("Session start data upload failed: " + www.error);
    }
}

IEnumerator UploadEndSession(DateTime date)
{
    WWWForm form = new WWWForm();
    form.AddField("user_ip", currentUserId.ToString());
    form.AddField("end_session", date.ToString("yyyy-MM-dd HH:mm:ss"));
    form.AddField("session_id", currentSessionId.ToString());
    string url = "https://citilumnes.upc.es/-hangx/Class_Session_Data.php";
    UnityWebRequest www = UnityWebRequest.Post(url, form);
    yield return www.SendWebRequest();

    if (www.result == UnityWebRequest.Result.Success)
    {
        CallbackEvents.OnEndSessionCallback.Invoke(currentSessionId);
    }
    else
    {
        UnityEngine.Debug.LogError("Session end data upload failed: " + www.error);
    }
}
```

```
IEnumerator UploadItem(int item, DateTime date)
{
    WWWForm form = new WWWForm();
    form.AddField("item", item.ToString());
    form.AddField("user_ip", currentUserId.ToString());
    form.AddField("session_id", currentSessionId.ToString());
    form.AddField("buy_date", date.ToString("yyyy-MM-dd HH:mm:ss"));

    UnityWebRequest www = UnityWebRequest.Post("https://citilumnes.upc.es/-hangx/Purchase_Data.php", form);
    yield return www.SendWebRequest();

    if (www.result != UnityWebRequest.Result.Success)
    {
        UnityEngine.Debug.LogError("Purchase data upload failed: " + www.error);
    }
    else
    {
        string answer = www.downloadHandler.text.Trim(new char[] { '\u0000', '\u0001', '\u0002', '\u0003', '\u0004', '\u0005', '\u0006', '\u0007', '\u0008', '\u0009', '\u000A', '\u000B', '\u000C', '\u000D', '\u000E', '\u000F', '\u0010', '\u0011', '\u0012', '\u0013', '\u0014', '\u0015', '\u0016', '\u0017', '\u0018', '\u0019', '\u001A', '\u001B', '\u001C', '\u001D', '\u001E', '\u001F', '\u0020', '\u0021', '\u0022', '\u0023', '\u0024', '\u0025', '\u0026', '\u0027', '\u0028', '\u0029', '\u002A', '\u002B', '\u002C', '\u002D', '\u002E', '\u002F', '\u0030', '\u0031', '\u0032', '\u0033', '\u0034', '\u0035', '\u0036', '\u0037', '\u0038', '\u0039', '\u003A', '\u003B', '\u003C', '\u003D', '\u003E', '\u003F', '\u0040', '\u0041', '\u0042', '\u0043', '\u0044', '\u0045', '\u0046', '\u0047', '\u0048', '\u0049', '\u004A', '\u004B', '\u004C', '\u004D', '\u004E', '\u004F', '\u0050', '\u0051', '\u0052', '\u0053', '\u0054', '\u0055', '\u0056', '\u0057', '\u0058', '\u0059', '\u005A', '\u005B', '\u005C', '\u005D', '\u005E', '\u005F', '\u0060', '\u0061', '\u0062', '\u0063', '\u0064', '\u0065', '\u0066', '\u0067', '\u0068', '\u0069', '\u006A', '\u006B', '\u006C', '\u006D', '\u006E', '\u006F', '\u0070', '\u0071', '\u0072', '\u0073', '\u0074', '\u0075', '\u0076', '\u0077', '\u0078', '\u0079', '\u007A', '\u007B', '\u007C', '\u007D', '\u007E', '\u007F', '\u0080', '\u0081', '\u0082', '\u0083', '\u0084', '\u0085', '\u0086', '\u0087', '\u0088', '\u0089', '\u008A', '\u008B', '\u008C', '\u008D', '\u008E', '\u008F', '\u0090', '\u0091', '\u0092', '\u0093', '\u0094', '\u0095', '\u0096', '\u0097', '\u0098', '\u0099', '\u009A', '\u009B', '\u009C', '\u009D', '\u009E', '\u009F', '\u00A0', '\u00A1', '\u00A2', '\u00A3', '\u00A4', '\u00A5', '\u00A6', '\u00A7', '\u00A8', '\u00A9', '\u00AA', '\u00AB', '\u00AC', '\u00AD', '\u00AE', '\u00AF', '\u00B0', '\u00B1', '\u00B2', '\u00B3', '\u00B4', '\u00B5', '\u00B6', '\u00B7', '\u00B8', '\u00B9', '\u00BA', '\u00BB', '\u00BC', '\u00BD', '\u00BE', '\u00BF', '\u00C0', '\u00C1', '\u00C2', '\u00C3', '\u00C4', '\u00C5', '\u00C6', '\u00C7', '\u00C8', '\u00C9', '\u00CA', '\u00CB', '\u00CC', '\u00CD', '\u00CE', '\u00CF', '\u00D0', '\u00D1', '\u00D2', '\u00D3', '\u00D4', '\u00D5', '\u00D6', '\u00D7', '\u00D8', '\u00D9', '\u00DA', '\u00DB', '\u00DC', '\u00DD', '\u00DE', '\u00DF', '\u00E0', '\u00E1', '\u00E2', '\u00E3', '\u00E4', '\u00E5', '\u00E6', '\u00E7', '\u00E8', '\u00E9', '\u00EA', '\u00EB', '\u00EC', '\u00ED', '\u00EE', '\u00EF', '\u00F0', '\u00F1', '\u00F2', '\u00F3', '\u00F4', '\u00F5', '\u00F6', '\u00F7', '\u00F8', '\u00F9', '\u00FA', '\u00FB', '\u00FC', '\u00FD', '\u00FE', '\u00FF' });
        if (uint.TryParse(answer, out uint parsedId) && parsedId > 0)
        {
            currentPurchaseId = parsedId;
            CallbackEvents.OnItemBuyCallback.Invoke();
        }
        else
        {
            UnityEngine.Debug.LogError("Invalid purchase ID received: " + www.downloadHandler.text);
        }
    }
}
```



# Code Structure

```
IEnumerator UploadPlayer(string name, string country, DateTime date)
{
    WWWForm form = new WWWForm();

    form.AddField("Name", name);
    form.AddField("Country", country);
    form.AddField("Date", date.ToString("yyyy-MM-dd HH:mm:ss"));

    UnityWebRequest www = UnityWebRequest.Post("https://citmalumnes.upc.es/~hangx/Player_Data.php", form);
    yield return www.SendWebRequest();

    if (www.result != UnityWebRequest.Result.Success)
    {
        UnityEngine.Debug.Log(www.error);
    }
    else
    {
        UnityEngine.Debug.Log("Player data uploaded successfully.");
        uint.TryParse(www.downloadHandler.text, out currentUserId);
        CallbackEvents.OnAddPlayerCallback.Invoke(currentUserId);
    }
}
```

```
IEnumerator UploadItem(int item, DateTime date)
{
    WWWForm form = new WWWForm();
    form.AddField("Item", item);
    form.AddField("User_ID", currentUserId.ToString());
    form.AddField("Session_ID", currentSessionId.ToString());
    form.AddField("Buy_Date", date.ToString("yyyy-MM-dd HH:mm:ss"));

    UnityWebRequest www = UnityWebRequest.Post("https://citmalumnes.upc.es/~hangx/Purchase_Data.php", form);
    yield return www.SendWebRequest();

    if (www.result != UnityWebRequest.Result.Success)
    {
        UnityEngine.Debug.Log(www.error);
    }
    else
    {
        UnityEngine.Debug.Log("Purchase data uploaded successfully.");
        uint.TryParse(www.downloadHandler.text, out currentPurchaseId);
        CallbackEvents.OnItemBuyCallback.Invoke();
    }
}
```

```
IEnumerator UploadSession(DateTime date, bool sessionStart)
{
    WWWForm form = new WWWForm();
    UnityWebRequest www;

    if (!sessionStart)
    {
        form.AddField("User_ID", currentUserId.ToString());
        form.AddField("Start_Session", date.ToString("yyyy-MM-dd HH:mm:ss"));

        www = UnityWebRequest.Post("https://citmalumnes.upc.es/~hangx/Session_Data.php", form);

        yield return www.SendWebRequest();

        if (www.result != UnityWebRequest.Result.Success)
        {
            UnityEngine.Debug.Log(www.error);
        }
        else
        {
            UnityEngine.Debug.Log("Session data uploaded successfully.");
            uint.TryParse(www.downloadHandler.text, out currentSessionId);
            CallbackEvents.OnNewSessionCallback.Invoke(currentSessionId);
        }
    }
    else
    {
        form.AddField("User_ID", currentUserId.ToString());
        form.AddField("End_Session", date.ToString("yyyy-MM-dd HH:mm:ss"));
        form.AddField("Session_ID", currentSessionId.ToString());

        www = UnityWebRequest.Post("https://citmalumnes.upc.es/~hangx/Close_Session_Data.php", form);

        yield return www.SendWebRequest();
        CallbackEvents.OnEndSessionCallback.Invoke(currentSessionId);
    }
}
```

# Code Structure

```
1 <?php
2 $servername = "localhost:3306";
3 $username = "hangx";
4 $password = "7vGPx2sehzBY";
5 $database = "hangx";
6
7 // Create connection
8 $conn = new mysqli($servername, $username, $password, $database);
9
10 // Check connection
11 if ($conn->connect_error) {
12     die("Connection failed: " . $conn->connect_error);
13 }
14 ?>
```

```
1 <?php
2 include 'db_connect.php';
3
4 $name = $_POST["Name"];
5 $country = $_POST["Country"];
6 $date = $_POST["Date"];
7
8 error_log("Received player data: Name={$name}, Country={$country}, Date={$date}");
9
10 $stmt = $conn->prepare("INSERT INTO 'Players' ('Name', 'Country', 'Date') VALUES (?, ?, ?)");
11 $stmt->bind_param("sss", $name, $country, $date);
12
13 if ($stmt->execute()) {
14     echo $conn->insert_id;
15 } else {
16     error_log("Error in Player_Data.php: " . $stmt->error);
17     echo "Error: " . $stmt->error;
18 }
19
20 $stmt->close();
21 $conn->close();
22 ?>
```

```
1 <?php
2 include 'db_connect.php';
3
4 $userId = $_POST["User_ID"];
5 $sessionId = $_POST["Session_ID"];
6 $itemId = $_POST["Item"];
7 $buyDate = $_POST["Buy_Date"];
8
9 error_log("Received purchase data: User_ID={$userId}, Session_ID={$sessionId}, Item={$itemId}, Buy_Date={$buyDate}");
10
11 $stmt = $conn->prepare("INSERT INTO 'Purchases' ('userId', 'sessionId', 'itemId', 'buyDate') VALUES (?, ?, ?, ?)");
12 $stmt->bind_param("iiis", $userId, $sessionId, $itemId, $buyDate);
13
14 if ($stmt->execute()) {
15     echo $conn->insert_id;
16 } else {
17     error_log("Error in Purchase_Data.php: " . $stmt->error);
18     echo "Error: " . $stmt->error;
19 }
20
21 $stmt->close();
22 $conn->close();
23 ?>
```

```
1 <?php
2 include 'db_connect.php';
3
4 $userId = $_POST["User_ID"];
5 $startSession = $_POST["Start_Session"];
6
7 error_log("Received session start data: User_ID={$userId}, Start_Session={$startSession}");
8
9 $stmt = $conn->prepare("INSERT INTO 'Sessions' ('userId', 'startSession') VALUES (?, ?)");
10 $stmt->bind_param("is", $userId, $startSession);
11
12 if ($stmt->execute()) {
13     echo $conn->insert_id;
14 } else {
15     error_log("Error in Session_Data.php: " . $stmt->error);
16     echo "Error: " . $stmt->error;
17 }
18
19 $stmt->close();
20 $conn->close();
21 ?>
```

```
1 <?php
2 include 'db_connect.php';
3
4 $sessionId = $_POST["Session_ID"];
5 $endSession = $_POST["End_Session"];
6
7 error_log("Received end session data: Session_ID={$sessionId}, End_Session={$endSession}");
8
9 $stmt = $conn->prepare("UPDATE 'Sessions' SET 'endSession' = ? WHERE 'sessionId' = ?");
10 $stmt->bind_param("si", $endSession, $sessionId);
11
12 if ($stmt->execute()) {
13     if ($stmt->affected_rows > 0) {
14         //echo "Session closed successfully";
15         echo $endSession;
16     } else {
17         error_log("No session updated in Close_Session_Data.php");
18         echo "No session updated";
19     }
20 } else {
21     error_log("Error in Close_Session_Data.php: " . $stmt->error);
22     echo "Error: " . $stmt->error;
23 }
24
25 $stmt->close();
26 $conn->close();
27 ?>
```

# Key Findings

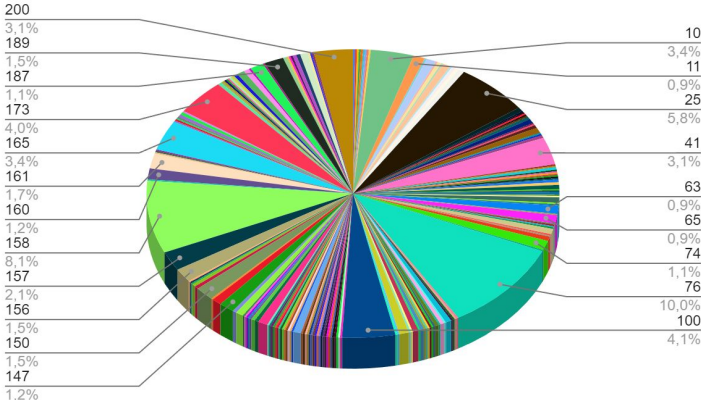
## User Activity

- Number of sessions per user

userId	num_sessions
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1
10	22
11	6
12	1
13	5
14	1
15	2
16	1
17	4
18	1
19	1
20	1
21	2
22	1
23	1
24	1
25	38

userId	num_sessions
76	65
158	53
25	38
100	27
173	26
10	22
165	22
41	20
200	20
157	14
161	11
156	10
189	10
150	10
160	8
147	8
74	7
187	7
11	6
63	6
65	6
13	5
97	5
148	5
137	5

```
1 SELECT
2     userId,
3     COUNT(sessionId) AS num_sessions
4 FROM Sessions
5 GROUP BY userId
6 ORDER BY num_sessions DESC
```





# Key Findings

## User Activity

- Average session duration

**avg\_session\_duration**

56135.7603

```
1 SELECT
2     AVG(TIMESTAMPDIFF(SECOND, startSession, endSession))
3     AS avg_session_duration
4 FROM Sessions
5 WHERE endSession IS NOT NULL
```

# Key Findings

## User Activity

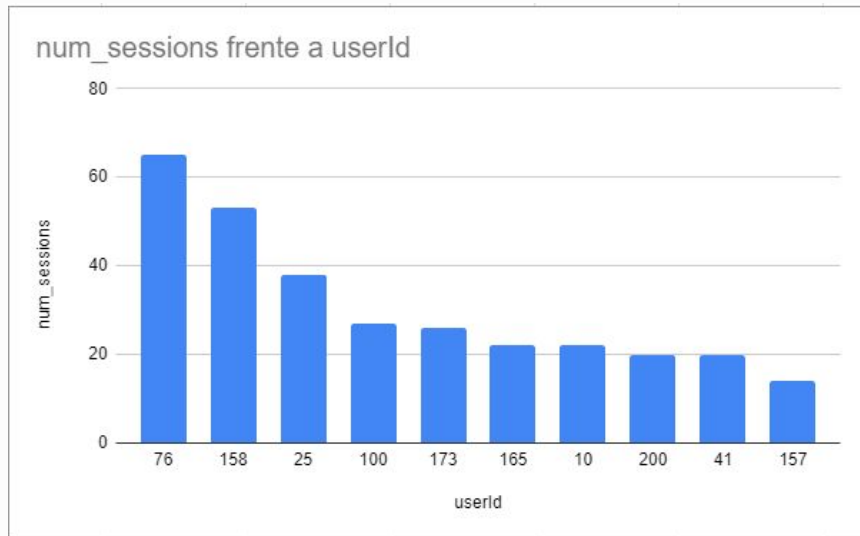
- Most active user (based on session count)

```

1 SELECT
2     userId,
3     COUNT(sessionId) AS num_sessions
4 FROM Sessions
5 GROUP BY userId
6 ORDER BY num_sessions DESC
7 LIMIT 10

```

userId	num_sessions ▼ 1
76	65
158	53
25	38
100	27
173	26
165	22
10	22
200	20
41	20
157	14





# Key Findings

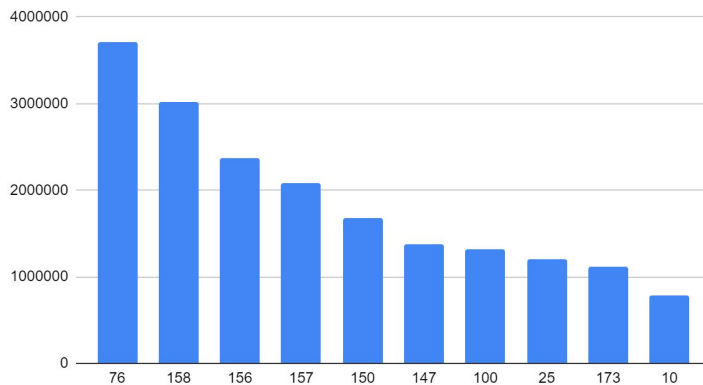
## User Activity

- Most active user (based on total session count)
- session count)

userId	total_duration_seconds
76	3704646
158	3013221
156	2377038
157	2075993
150	1682930
147	1373431
100	1313306
25	1204719
173	1110863
10	788861

```
1 SELECT
2     userId,
3     SUM(TIMESTAMPDIFF(SECOND, startSession, endSession))
4 AS total_duration_seconds
5 FROM Sessions
6 GROUP BY userId
7 ORDER BY total_duration_seconds DESC
8 LIMIT 10
```

Histograma





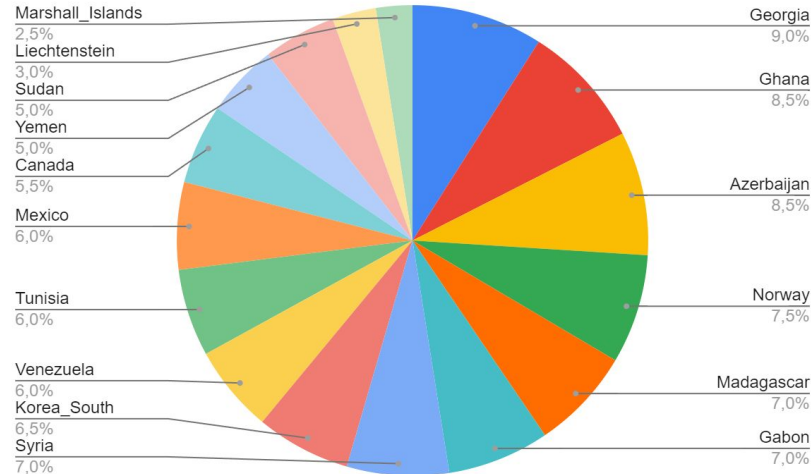
# Key Findings

## Player Demographics

- Distribution of players by country

Country	num_players ▾ 1
Georgia	18
Ghana	17
Azerbaijan	17
Norway	15
Madagascar	14
Gabon	14
Syria	14
Korea_South	13
Venezuela	12
Tunisia	12
Mexico	12
Canada	11
Yemen	10
Sudan	10
Liechtenstein	6
Marshall_Islands	5

```
1 SELECT
2     Country,
3     COUNT(userId) AS num_players
4 FROM Players
5 WHERE userId > 0
6 GROUP BY Country
7 ORDER BY num_players DESC
```





# Key Findings

## Player Demographics

- New player acquisition rate

new_players	acquisition_rate
200	0.5495

```
1 SELECT
2     COUNT(userId) AS new_players,
3     COUNT(userId) / DATEDIFF(MAX(Date), MIN(Date)) AS
4     acquisition_rate
5 FROM Players
```



# Key Findings

## Purchases

- Average purchase value

num_items	num_purchases	average_purchase_value
5	65	13.0000

```
1 SELECT
2     COUNT(DISTINCT p.itemId) AS num_items,
3     COUNT(p.purchaseId) AS num_purchases,
4     COUNT(p.purchaseId) / COUNT(DISTINCT p.itemId) AS
5     average_purchase_value
6 FROM Purchases p
```

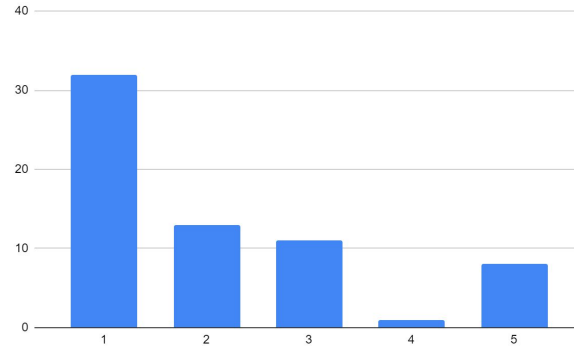
# Key Findings

## Purchases

- Popular items

itemId	num_purchases
1	32
2	13
3	11
5	8
4	1

```
1 SELECT
2     itemId,
3     COUNT(purchaseId) AS num_purchases
4 FROM Purchases
5 GROUP BY itemId
6 ORDER BY num_purchases DESC
```





# Key Findings

## User Engagement

- Conversion rate from sessions to purchases

num_sessions	num_purchases	conversion_rate
653	65	0.0995

```
1 SELECT
2     COUNT(DISTINCT s.sessionId) AS num_sessions,
3     COUNT(DISTINCT p.purchaseId) AS num_purchases,
4     COUNT(DISTINCT p.purchaseId) / COUNT(DISTINCT
5     s.sessionId) AS conversion_rate
6 FROM Sessions s
7 LEFT JOIN Purchases p ON s.sessionId = p.sessionId
```



# Key Findings

## User Engagement

- Retention rate (how many users return for multiple sessions)

num_returning_users	num_users_on_first_session	retention_rate
31	31	1.0000

```
1 SELECT
2     COUNT(DISTINCT r1.userId) AS num_returning_users,
3     COUNT(DISTINCT r2.userId) AS
4     num_users_on_first_session,
5     COUNT(DISTINCT r1.userId) / COUNT(DISTINCT r2.userId)
6     AS retention_rate
7 FROM Sessions r1
8 JOIN Sessions r2 ON r1.userId = r2.userId
9     AND r1.sessionId <> r2.sessionId
10    AND r1.startSession < r2.startSession
```



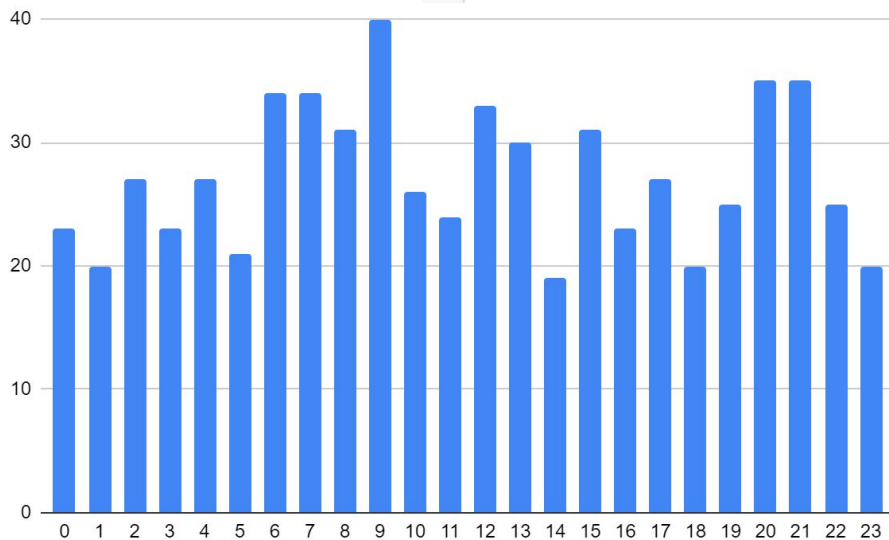
# Key Findings

## Session analysis

- Peak usage hours

hour_of_day	num_sessions
0	23
1	20
2	27
3	23
4	27
5	21
6	34
7	34
8	31
9	40
10	26
11	24
12	33
13	30
14	19
15	31
16	23
17	27
18	20
19	25
20	35
21	35
22	25
23	20

```
1 SELECT
2     HOUR(startSession) AS hour_of_day,
3     COUNT(sessionId) AS num_sessions
4 FROM Sessions
5 GROUP BY hour_of_day
6 ORDER BY `hour_of_day` ASC LIMIT 24
```







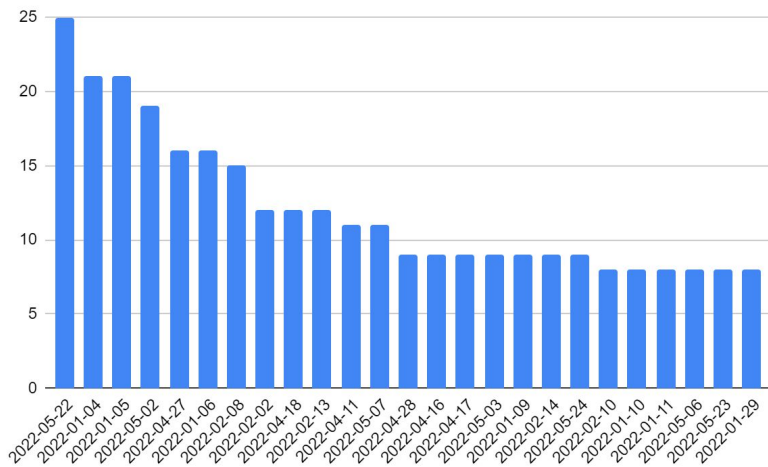
# Key Findings

## Session analysis

- Peak usage days

```
1 SELECT
2     DATE(startSession) AS day,
3     COUNT(sessionId) AS num_sessions
4 FROM Sessions
5 GROUP BY day
6 ORDER BY `num_sessions` DESC LIMIT 25
```

day	num_sessions ▾ 1
2022-05-22	25
2022-01-04	21
2022-01-05	21
2022-05-02	19
2022-04-27	16
2022-01-06	16
2022-02-08	15
2022-02-02	12
2022-04-18	12
2022-02-13	12
2022-04-11	11
2022-05-07	11
2022-04-28	9
2022-04-16	9
2022-04-17	9
2022-05-03	9
2022-01-09	9
2022-02-14	9
2022-05-24	9
2022-02-10	8
2022-01-10	8
2022-01-11	8
2022-05-06	8
2022-05-23	8
2022-01-29	8





# Key Findings

## Session analysis

- Average time between sessions

**avg\_time\_between\_sessions**

248833.2176

```
1 SELECT
2     AVG(TIMESTAMPDIFF(SECOND, s1.endSession,
3         s2.startSession)) AS avg_time_between_sessions
4 FROM Sessions s1
5 JOIN Sessions s2 ON s1.userId = s2.userId
6     AND s1.sessionId <> s2.sessionId
7     AND s1.endSession < s2.startSession
```