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://citmalumnes.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[] { '\uFEFF', '\u200B', ' ', '\t', '\r', '\n' });
            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_ID", currentUserId.ToString());
    form.AddField("Start_Session", date.ToString("yyyy-MM-dd HH:mm:ss"));

    string url = "https://citmalumnes.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[] { '\uFEFF', '\u200B', ' ', '\t', '\r', '\n' });
        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_ID", currentUserId.ToString());
    form.AddField("End_Session", date.ToString("yyyy-MM-dd HH:mm:ss"));
    form.AddField("Session_ID", currentSessionId.ToString());

    string url = "https://citmalumnes.upc.es/~hangx/Close_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_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.LogError("Purchase data upload failed: " + www.error);
    }
    else
    {
        string answer = www.downloadHandler.text.Trim(new char[] { '\uFEFF', '\u200B', ' ', '\t', '\r', '\n'
        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

```php
<?php
$servername = "localhost:3306";
$username = "hangx";
$password = "7vGPx2sehzBY";
$database = "hangx";

// Create connection
$conn = new mysqli($servername, $username, $password, $database);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
?>
```

```php
<?php
include 'db_connect.php';

$name = $_POST["Name"];
$country = $_POST["Country"];
$date = $_POST["Date"];

error_log("Received player data: Name={$name}, Country={$country}, Date={$date}");

$stmt = $conn->prepare("INSERT INTO `Players`(`Name`, `Country`, `Date`) VALUES (?, ?, ?)");
$stmt->bind_param("sss", $name, $country, $date);

if ($stmt->execute()) {
    echo $conn->insert_id;
} else {
    error_log("Error in Player_Data.php: " . $stmt->error);
    echo "Error: " . $stmt->error;
}

$stmt->close();
$conn->close();
?>
```

```php
<?php
include 'db_connect.php';

$userId = $_POST["User_ID"];
$sessionId = $_POST["Session_ID"];
$itemId = $_POST["Item"];
$buyDate = $_POST["Buy_Date"];

error_log("Received purchase data: User_ID={$userId}, Session_ID={$sessionId}, Item={$itemId}, Buy_Date={$buyDate}");

$stmt = $conn->prepare("INSERT INTO `Purchases`(`userId`, `sessionId`, `itemId`, `buyDate`) VALUES (?, ?, ?, ?)");
$stmt->bind_param("iiis", $userId, $sessionId, $itemId, $buyDate);

if ($stmt->execute()) {
    echo $conn->insert_id;
} else {
    error_log("Error in Purchase_Data.php: " . $stmt->error);
    echo "Error: " . $stmt->error;
}

$stmt->close();
$conn->close();
?>
```

```php
<?php
include 'db_connect.php';

$userId = $_POST["User_ID"];
$startSession = $_POST["Start_Session"];

error_log("Received session start data: User_ID={$userId}, Start_Session={$startSession}");

$stmt = $conn->prepare("INSERT INTO `Sessions`(`userId`, `startSession`) VALUES (?, ?)");
$stmt->bind_param("is", $userId, $startSession);

if ($stmt->execute()) {
    echo $conn->insert_id;
} else {
    error_log("Error in Session_Data.php: " . $stmt->error);
    echo "Error: " . $stmt->error;
}

$stmt->close();
$conn->close();
?>
```

```php
<?php
include 'db_connect.php';

$sessionId = $_POST["Session_ID"];
$endSession = $_POST["End_Session"];

error_log("Received end session data: Session_ID={$sessionId}, End_Session={$endSession}");

$stmt = $conn->prepare("UPDATE `Sessions` SET `endSession` = ? WHERE `sessionId` = ?");
$stmt->bind_param("si", $endSession, $sessionId);

if ($stmt->execute()) {
    if ($stmt->affected_rows > 0) {
        //echo "Session closed successfully";
        echo $endSession;
    } else {
        error_log("No session updated in Close_Session_Data.php");
        echo "No session updated";
    }
} else {
    error_log("Error in Close_Session_Data.php: " . $stmt->error);
    echo "Error: " . $stmt->error;
}

$stmt->close();
$conn->close();
?>
```

Key Findings

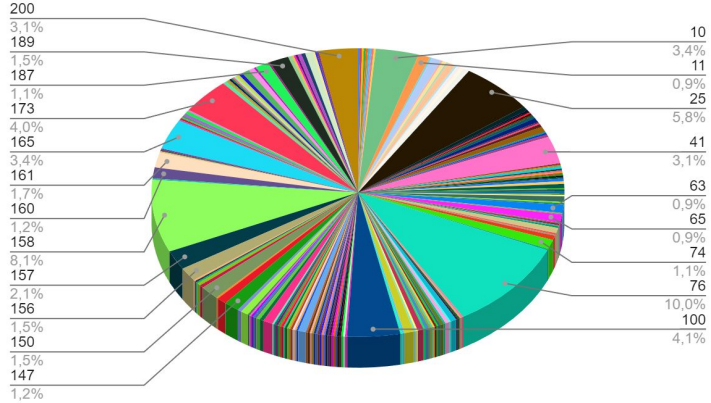
User Activity

- Number of sessions per user

```sql
SELECT
    userId,
    COUNT(sessionId) AS num_sessions
FROM Sessions
GROUP BY userId
ORDER BY num_sessions DESC
```

userId ▲ 1	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 ▼ 1
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



Key Findings

User Activity

- Average session duration

avg_session_duration
56135.7603

```sql
SELECT
    AVG(TIMESTAMPDIFF(SECOND, startSession, endSession))
AS avg_session_duration
FROM Sessions
WHERE endSession IS NOT NULL
```

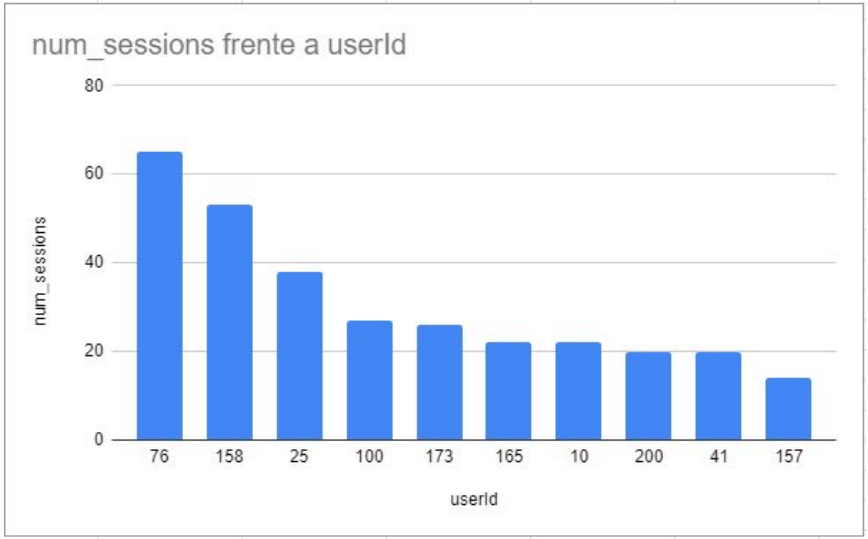
Key Findings

User Activity

- Most active user (based on session count)

```sql
SELECT
    userId,
    COUNT(sessionId) AS num_sessions
FROM Sessions
GROUP BY userId
ORDER BY num_sessions DESC
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



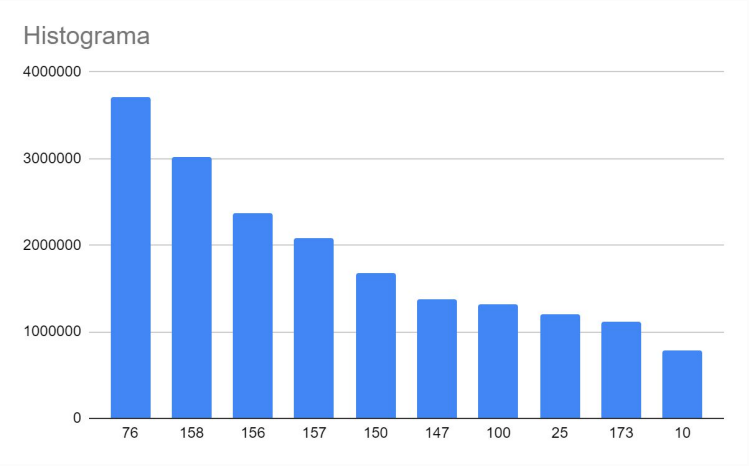
Key Findings

User Activity

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

```sql
SELECT
    userId,
    SUM(TIMESTAMPDIFF(SECOND, startSession, endSession))
AS total_duration_seconds
FROM Sessions
GROUP BY userId
ORDER BY total_duration_seconds DESC
LIMIT 10
```

userId	total_duration_seconds ▼ 1
76	3704646
158	3013221
156	2377038
157	2075993
150	1682930
147	1373431
100	1313306
25	1204719
173	1110863
10	788861



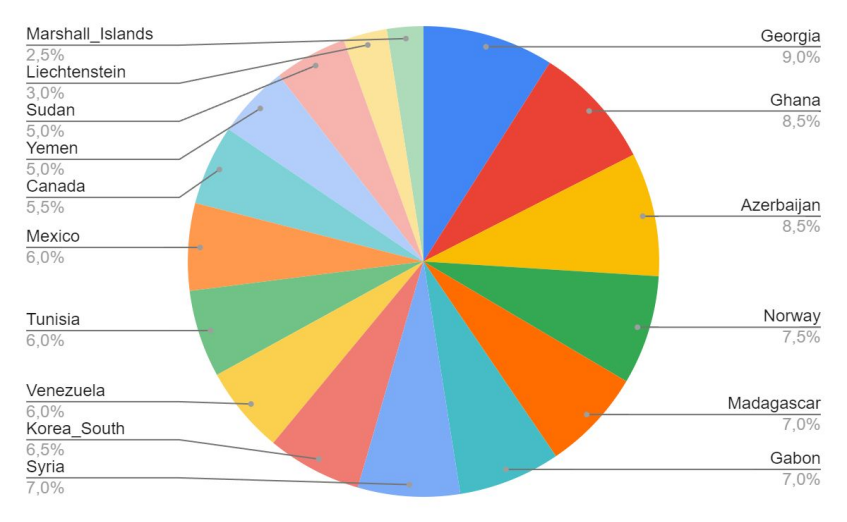
Key Findings

Player Demographics

- Distribution of players by country

```sql
SELECT
    Country,
    COUNT(userId) AS num_players
FROM Players
WHERE userId > 0
GROUP BY Country
ORDER BY num_players DESC
```

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



Key Findings

Player Demographics

- New player acquisition rate

new_playerss	acquissition_rate
200	0.5495

```sql
SELECT
    COUNT(userId) AS new_players,
    COUNT(userId) / DATEDIFF(MAX(Date), MIN(Date)) AS
acquisition_rate
FROM Players
```

Key Findings

Purchases

- Average purchase value

num_itemss	num_purchases	average_purchase_value
5	65	13.0000

```sql
SELECT
    COUNT(DISTINCT p.itemId) AS num_items,
    COUNT(p.purchaseId) AS num_purchases,
    COUNT(p.purchaseId) / COUNT(DISTINCT p.itemId) AS
average_purchase_value
FROM Purchases p
```

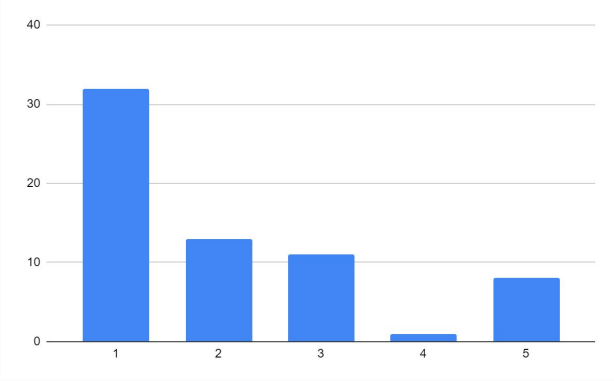
Key Findings

Purchases

- Popular items

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

```sql
SELECT
    itemId,
    COUNT(purchaseId) AS num_purchases
FROM Purchases
GROUP BY itemId
ORDER BY num_purchases DESC
```



Key Findings

User Engagement

- Conversion rate from sessions to purchases

num_sessionss	num_purchases	conversion_rate
653	65	0.0995

```sql
SELECT
    COUNT(DISTINCT s.sessionId) AS num_sessions,
    COUNT(DISTINCT p.purchaseId) AS num_purchases,
    COUNT(DISTINCT p.purchaseId) / COUNT(DISTINCT
s.sessionId) AS conversion_rate
FROM Sessions s
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

```sql
SELECT
    COUNT(DISTINCT r1.userId) AS num_returning_users,
    COUNT(DISTINCT r2.userId) AS
num_users_on_first_session,
    COUNT(DISTINCT r1.userId) / COUNT(DISTINCT r2.userId)
AS retention_rate
FROM Sessions r1
JOIN Sessions r2 ON r1.userId = r2.userId
    AND r1.sessionId <> r2.sessionId
    AND r1.startSession < r2.startSession
```

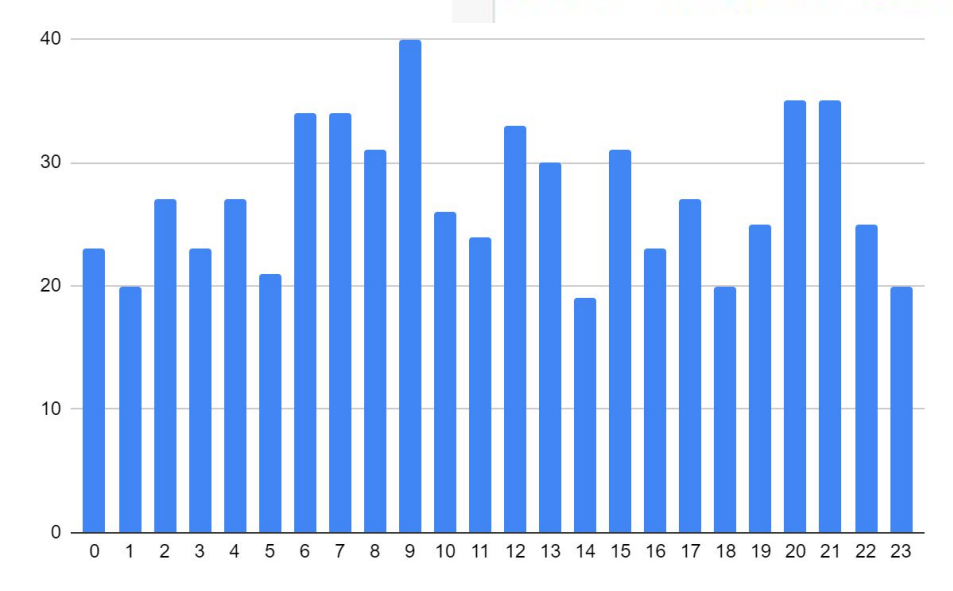
Key Findings

Session analysis

- Peak usage hours

```sql
SELECT
    HOUR(startSession) AS hour_of_day,
    COUNT(sessionId) AS num_sessions
FROM Sessions
GROUP BY hour_of_day
ORDER BY `hour_of_day` ASC LIMIT 24
```

hour_of_day ▲ 1	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

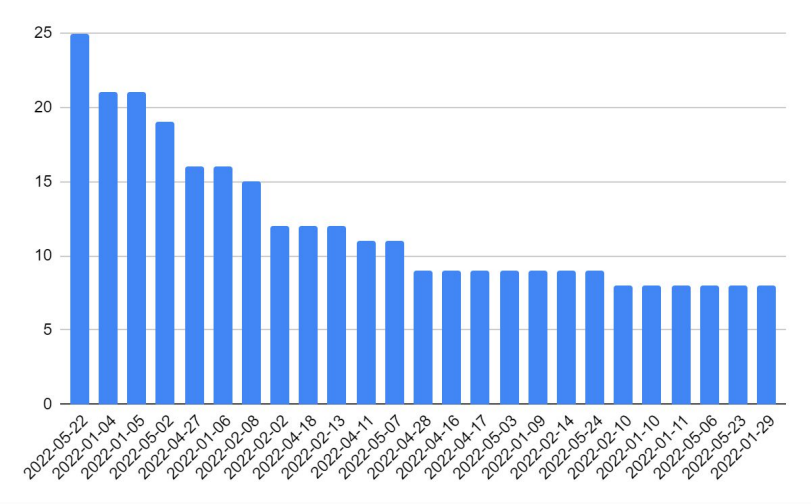


Key Findings

Session analysis

- Peak usage days

```sql
SELECT
    DATE(startSession) AS day,
    COUNT(sessionId) AS num_sessions
FROM Sessions
GROUP BY day
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

```sql
SELECT
    AVG(TIMESTAMPDIFF(SECOND, s1.endSession, s2.startSession)) AS avg_time_between_sessions
FROM Sessions s1
JOIN Sessions s2 ON s1.userId = s2.userId
    AND s1.sessionId <> s2.sessionId
    AND s1.endSession < s2.startSession
```