Team 7

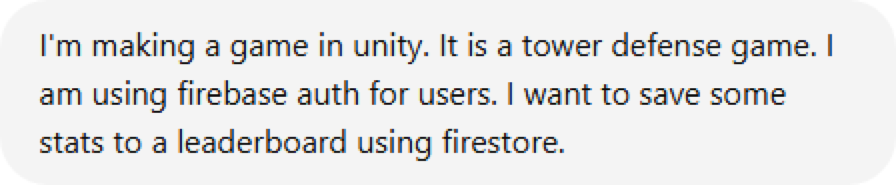
AI Contribution

Info:

Model used: ChatGPT o4-mini-high
Goal: To create a leaderboard functionality

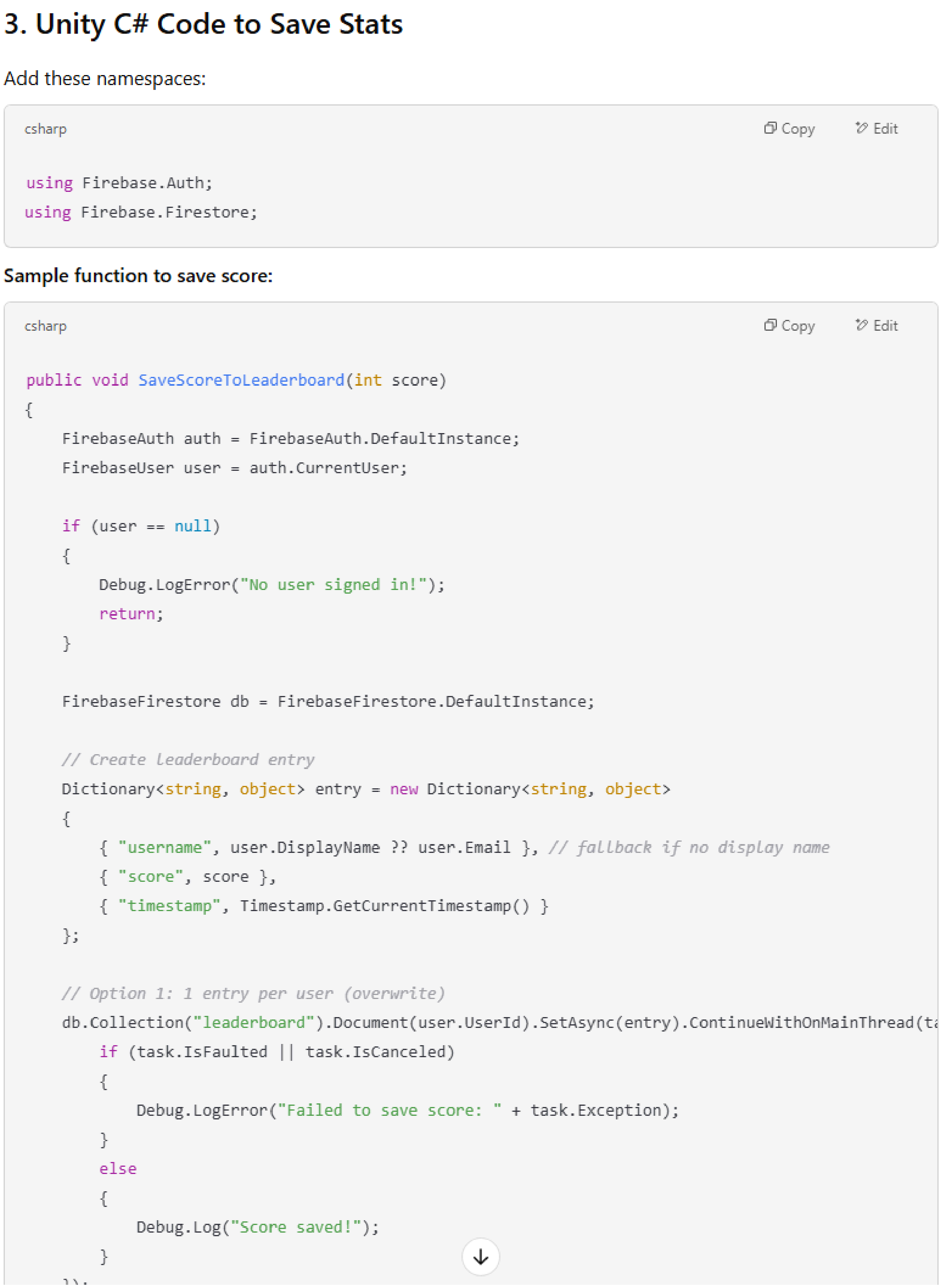
Part 1:

I started with describing to ChatGPT what I was doing, what tools I was using, and what I wanted from it. Below is my initial prompt



I'm making a game in unity. It is a tower defense game. I am using firebase auth for users. I want to save some stats to a leaderboard using firestore.

It returned with a few steps regarding what I have to do to configure firebase, and some code that would be useful to implement. I made sure the first few steps were done, and proceeded to the third step, testing the save of stats.



3. Unity C# Code to Save Stats

Add these namespaces:

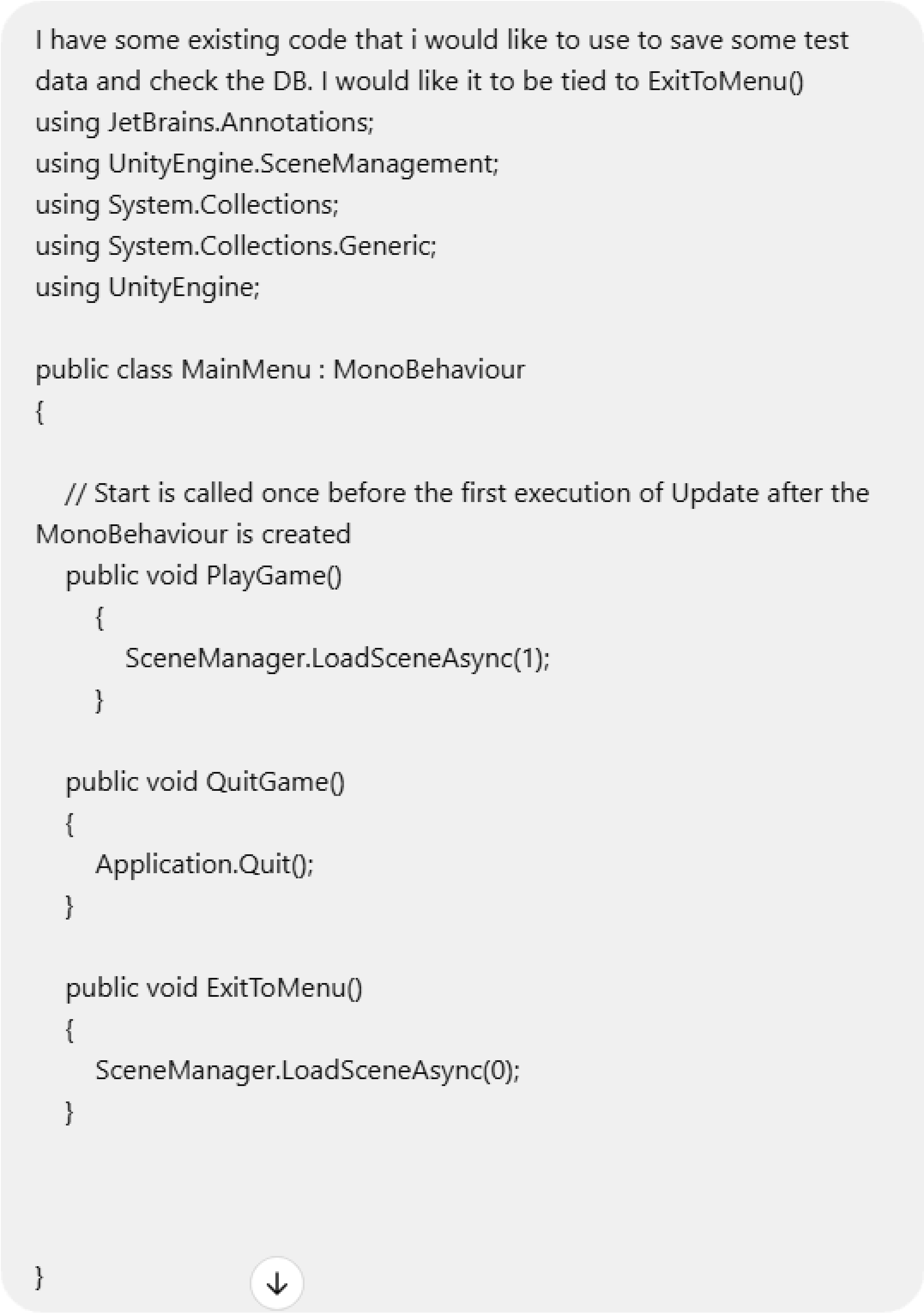
```csharp
using Firebase.Auth;
using Firebase.Firestore;
```

Sample function to save score:

```csharp
public void SaveScoreToLeaderboard(int score)
{
    FirebaseAuth auth = FirebaseAuth.DefaultInstance;
    FirebaseUser user = auth.CurrentUser;

    if (user == null)
    {
        Debug.LogError("No user signed in!");
        return;
    }

    FirebaseFirestore db = FirebaseFirestore.DefaultInstance;

    // Create leaderboard entry
    Dictionary<string, object> entry = new Dictionary<string, object>
    {
        { "username", user.DisplayName ?? user.Email }, // fallback if no display name
        { "score", score },
        { "timestamp", Timestamp.GetCurrentTimestamp() }
    };

    // Option 1: 1 entry per user (overwrite)
    db.Collection("leaderboard").Document(user.UserId).SetAsync(entry).ContinueWithOnMainThread(ta
        if (task.IsFaulted || task.IsCanceled)
        {
            Debug.LogError("Failed to save score: " + task.Exception);
        }
        else
        {
            Debug.Log("Score saved!");
        }
    });
```

Team 7

Since I have existing code where I want to test this, I explained that and provided the code I wanted it to use for testing. Below is the described prompt.



I have some existing code that i would like to use to save some test data and check the DB. I would like it to be tied to ExitToMenu()

```
using JetBrains.Annotations;
using UnityEngine.SceneManagement;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MainMenu : MonoBehaviour
{

    // Start is called once before the first execution of Update after the MonoBehaviour is created
    public void PlayGame()
        {
            SceneManager.LoadSceneAsync(1);
        }

    public void QuitGame()
    {
        Application.Quit();
    }

    public void ExitToMenu()
    {
        SceneManager.LoadSceneAsync(0);
    }



}
```

It responded with the following code.

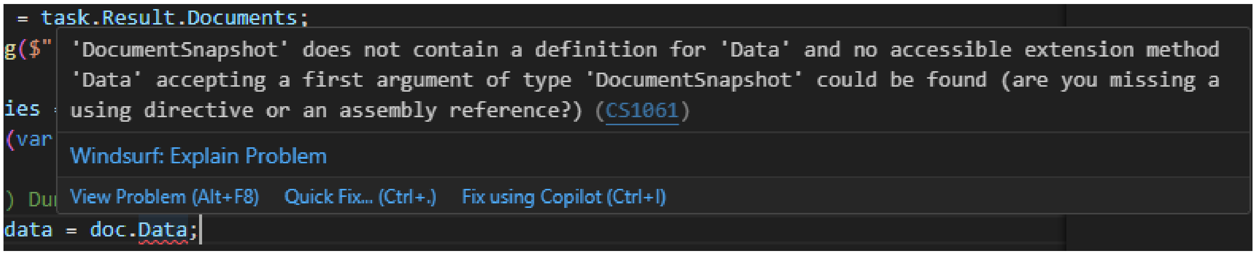
```
using UnityEngine;
using UnityEngine.SceneManagement;
using Firebase.Auth;
using Firebase.Firestore;
using Firebase.Extensions;   // for ContinueWithOnMainThread
using System.Collections.Generic;

public class MainMenu : MonoBehaviour
{
    // Called by your "Exit to Menu" button
    public void ExitToMenu()
    {
        SaveTestData();                       // <-- enqueue our write
        SceneManager.LoadSceneAsync(0);       // <-- then go back to menu
    }

    void SaveTestData()
    {
        var auth = FirebaseAuth.DefaultInstance;
        var user = auth.CurrentUser;
        if (user == null)
        {
            Debug.LogError("[Firestore] no user signed in, skipping write.");
            return;
        }

        var db = FirebaseFirestore.DefaultInstance;

        // build a small test payload
        var testDoc = new Dictionary<string, object>
        {
            { "username", user.DisplayName ?? user.Email },
            { "testValue", Random.Range(0, 1000) },
            { "createdAt", Timestamp.GetCurrentTimestamp() }
        };

        // write under a "testData" collection using auto-ID
        db.Collection("testData")
          .AddAsync(testDoc)
          .ContinueWithOnMainThread(task =>
        {
            if (task.IsFaulted || task.IsCanceled)
```

```
                Debug.LogError($"[Firestore] failed to write test data:
{task.Exception}");
            else
                Debug.Log("[Firestore] test data written, ID = " +
task.Result.Id);
        });
    }

    // ... your other methods unchanged
    public void PlayGame()
    {
        SceneManager.LoadSceneAsync(1);
    }

    public void QuitGame()
    {
        Application.Quit();
    }
}
```

I added this code to my script, and there were some errors. With some digging I found that I had downloaded the dependencies for firebase storage instead of what I told ChatGPT to use (Firestore). I updated the dependencies, and the errors went away. When I executed the ExitToMenu() button, I checked the DB and found that it had worked.

I added the data and collection names that I actually wanted, and tested successfully. Next, I wanted to read that data, and populate a leaderboard. So I constructed a prompt and asked ChatGPT the best way to do it.

I Then wrestled with it for about two hours because the code it was giving me kept giving me errors like below



```
 = task.Result.Documents;
'DocumentSnapshot' does not contain a definition for 'Data' and no accessible extension method 'Data' accepting a first argument of type 'DocumentSnapshot' could be found (are you missing a using directive or an assembly reference?) (CS1061)
data = doc.Data;
```

I found that It was using slightly wrong functions like using data.Count when it needed to use task.Result.Count. I was able to give ChatGPT the error messages and figure out what was going wrong most of the time. Other times I had to manually debug.

Conclusion

When I finished the feature I felt more comfortable using ChatGPT as a developing companion. I definitely could see the scope of ChatGPT was an issue because it didn’t have access to my codebase. I had to very deliberately explain what I wanted to do where and how, because it didn't know the constraints beforehand. If an AI agent has access to the codebase and can make informed decisions based on those factors I could see it working much better. One thing that also came to mind while doing this was security concerns. If this agent has access to my codebase who knows who else does, you have to trust the engineers of the LLM to keep your data secure.