# ASSIGNMENT COVERSHEET

**UTS: ENGINEERING & INFORMATION TECHNOLOGY**

| SUBJECT NUMBER & NAME | NAME OF STUDENT(s) (PRINT CLEARLY) | STUDENT ID(s) |
|---|---|---|
| 31927 32998 Application Development with .NET .NET Application Development | *WAI MING RONAN SOH*<br>*Kartikay Singh*<br>*Zain Naqvi* | 25465345<br>14197089 |

| STUDENT EMAIL | STUDENT CONTACT NUMBER |
|---|---|
| waiming.r.soh@student.uts.edu.au<br>kartikay.singh@student.uts.edu.au<br>syedzain.naqvi@student.uts.edu.au | 0423951033<br>0493038173<br>0466713419 |

| NAME OF TUTOR | TUTORIAL GROUP | DUE DATE |
|---|---|---|
| Alexander Minton | 05 | 20/10/2025 |

**ASSESSMENT ITEM NUMBER & TITLE**

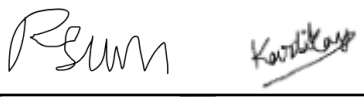
ASSIGNMENT-2: Group Project

☐ I acknowledge that if AI or another nonrecoverable source was used to generate materials for background research and self-study in producing this assignment, I have checked and verified the accuracy and integrity of the information used.
☐ I confirm that I have read, understood and followed the guidelines for assignment submission and presentation on page 2 of this cover sheet.
☐ I confirm that I have read, understood and followed the advice in the Subject Outline about assessment requirements.
☐ I understand that if this assignment is submitted after the due date it may incur a penalty for lateness unless I have previously had an extension of time approved and have attached the written confirmation of this extension.

**Declaration of originality**: The work contained in this assignment, other than that specifically attributed to another source, is that of the author(s) and has not been previously submitted for assessment. I have rewritten any material provided by AI or other nonrecoverable sources and where appropriate acknowledged their contribution. I understand that, should this declaration be found to be false, disciplinary action could be taken and penalties imposed in accordance with University policy and rules. In the statement below, I have indicated the extent to which I have collaborated with others, whom I have named.

No content generated by AI technologies or other sources has been presented as my own work and I have rewritten any text provided by AI or other sources in my own words.

**Statement of collaboration**:

**Signature of student(s)**  **Date 20/10/2025**

## 1. Introduction and Summary of Project

Social Sports Hub is a cross-platform .NET MAUI application designed to help people easily organise and join casual sports games in their local community. The project was developed to address a common social and fitness problem: individuals often want to play sports such as soccer, basketball, or cricket but struggle to find enough players or nearby games. Likewise, organisers frequently face no-shows and inconsistent participation. Social Sports Hub provides a solution that blends convenience, fairness, and community engagement.

The purpose of the project is to promote active lifestyles, social inclusion, and community connection through technology. By combining location-based event discovery, user reliability tracking, and a clean user interface, the app turns casual sports into a collaborative, trustworthy experience. Users can browse available games in their area, register for matches that fit their interests, or create their own events specifying the sport, venue, and capacity.

A key planned innovation of this project was the Honor System, a reputation mechanism that rewards reliable players and discourages no-shows. Users would earn or lose honor points based on attendance, influencing their priority when joining limited-capacity games. However, this system could not be implemented due to time constraints and the complexity of integrating it with other modules. Similarly, the Map View feature  designed to visualise nearby games using the Google Maps API  could not be finalised because Zain encountered hardware limitations and issues integrating code from other contributors into his local environment. Despite these setbacks, the base code for map features exists in the repository and can be added in future updates.

From a technical perspective, the project demonstrates proficiency in core .NET and object-oriented programming principles. It includes the use of classes, generics, LINQ, interfaces, delegates, enumerators, and properties. Entity Framework Core with SQLite provides persistent local storage for users and events. The graphical user interface was built with .NET MAUI to support Android, Windows, and iOS. Multiple interfaces communicate smoothly through MVVM data binding and navigation patterns. Overall, Social Sports Hub achieves its social and technical objectives by demonstrating object-oriented design, effective data management, and a relevant, real-world application scenario.

## 2. Development Approach

### 2.1 Framework and Architecture

Social Sports Hub uses .NET MAUI for the cross-platform interface and Entity Framework Core (EF Core) as the Object–Relational Mapper (ORM).
The app uses a local SQLite database not an external cloud database meaning all data (users, events, honor scores, and stats) is stored persistently on-device through EF Core's local context.

This setup simplifies offline access while still fulfilling the assignment's database requirements (file/database reading and writing). If expanded beyond the assignment scope, the same EF models could easily connect to an external SQL Server or Azure database with minimal configuration changes.

| Layer | Tool / Library | Purpose |
|---|---|---|
| UI / Frontend | .NET MAUI XAML | Cross-platform interface |
| Database | SQLite + Entity Framework Core | Local ORM-based data persistence |
| Mapping | Google Maps API | Displays nearby games with interactive pins |
| Logic / Utilities | LINQ + Lambda Expressions | Filtering and analytics |
| Testing | NUnit | Unit tests for login feature |

## 2.3 Methodology

The development of Social Sports Hub followed a structured, three-phase approach that allowed the team to stay organized and aligned throughout the process.

**Planning and Design:**

We began by brainstorming key features, assigning roles, and sketching UI mock-ups to visualise the user experience. During this stage, we identified the main entities User, SportEvent, Roster, and HonorSystem and defined how they interact within the system. We also mapped out user workflows such as logging in, discovering events, joining games, and tracking participation.

**Implementation:**

Each team member was responsible for specific modules, including login, event management, profile pages, and the map feature. We implemented Entity Framework Core with a repository pattern for data access, ensuring a scalable and maintainable backend. Data binding and observable properties were used to keep the UI reactive and consistent with real-time data changes.

**Testing and Refinement:**

We used NUnit tests to validate key logic, including honor-score updates and event registration. Manual testing was conducted in Visual Studio 2022 (.NET 8) to ensure a smooth user experience across devices. The team refined navigation flow, improved error handling, and validated user inputs to eliminate common bugs and enhance overall stability.

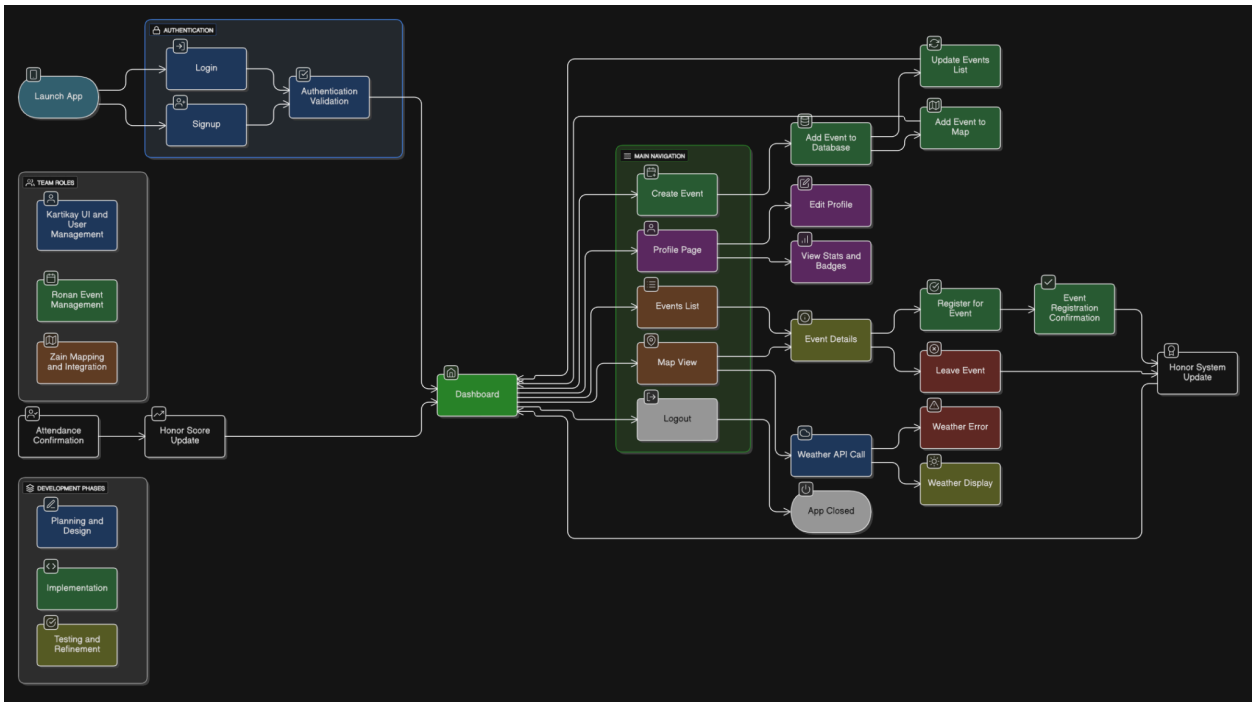
## 2.4 Design Principles

The project was built on solid object-oriented programming principles, ensuring both clarity and flexibility in design.

- **Encapsulation & Abstraction:** Each class only exposes what's necessary, protecting data integrity and simplifying maintenance.
- **Inheritance & Polymorphism:** A base User class was extended to create HostUser and Participant subclasses, promoting code reuse.
- **Interfaces:** Implementing interfaces such as IEventManager and IUserRepository provided flexibility for future scalability.
- **LINQ and Lambda Expressions:** These were used extensively to simplify event filtering, sorting, and querying user data.
- **Delegates and Events:** Event handlers managed map-pin clicks and dynamic UI updates efficiently.
- **Error Handling and Validation:** Exceptions were caught and displayed through intuitive UI alerts, maintaining a seamless user experience.

This structured and modular approach ensured the application met both its functional and technical goals, including multi-interface communication, strong OOP architecture, and robust database connectivity.

## 3. System Flowchart




The System Flowchart Diagram for Social Sports Hub illustrates how data flows between the app's main components  Authentication, Event Management, Weather Integration, and User Interactions. It visually represents the sequence from user login or registration to navigating through the dashboard, creating or joining events, and viewing event details. The flowchart also highlights the interconnection between users, events, and supporting modules like weather updates and honor tracking. Each process is designed to ensure a smooth and interactive experience, where users can easily manage their sports activities and stay informed about upcoming games.

However, due to time and technical constraints, three planned features were not implemented in the final build: the Map View, Weather System, and Honor System. The Map View, developed and tested separately by Zain, was intended to display nearby games and event pins using the Google Maps API, but couldn't be integrated because of hardware limitations. The Weather System, designed to provide real-time updates through the OpenWeather API, was also left out to prioritize core functionality. Lastly, the Honor System, which would have tracked user reliability and attendance, couldn't be completed in time. The map features have existing code (viewable in GitHub "Map" repository) and can be seamlessly integrated in future versions of the app.

**4. Role of Team Members**

This project was completed collaboratively by a team of three students: Kartikay, Ronan, and Zain. Each member specialised in different layers of the app to ensure modularity, integration efficiency, and parallel

| Member | Main Responsibilities | Key Contributions |
|---|---|---|
| **Kartikay Singh** | **UI/UX Design, User Management, Database Setup, Weather Integration, and Testing** | Designed and implemented key interface and functionality components using .NET MAUI XAML. Developed the Login and Signup pages with complete input validation and database integration through Entity Framework Core, connecting to a local SQLite database. Created and structured the Profile Page to display player details and event participation information. Implemented event join functionality, event filtering, and participant management features, allowing users to join or leave events and view host information directly from the interface. Focused on enhancing usability and interaction flow across devices while improving the system's reliability and responsiveness. Although planned integrations such as the Weather API and Honor Score system could not be completed due to time constraints, significant progress was made in refining the core functionality and user experience. |

| | | |
|---|---|---|
| **Wai Ming Ronan Soh** | **Event Management & Logic Layer** | Built the SportEvent and SportEventRoster classes. Developed CRUD functionality for creating, updating, and deleting events using EF Core. Implemented the host's event-management screen ("My Events"), allowing organisers to confirm attendance and manage waitlists. Added validation for duplicate events, time conflicts, and capacity enforcement. Contributed to LINQ queries and NUnit test cases. |
| **Zain Naqvi** | **Mapping & Integration Layer** | Integrated Google Maps API plugin for MAUI to display nearby games. Implemented location detection and automatic pin-creation for each SportEvent. Designed the MapView page that shows all current games, fetching coordinates from SQLite. Built logic for tapping a pin to view game details. |

**Team Workflow and Coordination**

Workflow was discussed during the beginning of implementation where we sorted out the identity and scope of what app we were to build. After we had decided on an idea, we then brainstormed on the functionality we intend to implement in the app. The original scope of the app will showcase object-oriented principles by modeling core entities such as Users, Events, and Notifications as separate classes. The GUI will be developed using .NET (WinForms/WPF/MAUI) and integrated with external libraries for mapping (e.g., Bing Maps or Google Maps API) and real-time notifications. The project will feature at least four main screens: Login/Registration, Event Creation, Event Discovery (map view), and User Dashboard. Then delegated the work amongst ourselves in achieving each part which we used a Github to centralise our code base and workflow.

- Used GitHub for version control and task tracking.
- Weekly meetings to discuss blockers and align features.
- Followed a merge-review policy to maintain code integrity.
- Employed pair programming during database integration and UI-binding sessions.

**Key Collaborative Achievements**

The Social Sports Hub project was a team-driven effort that required consistent communication, version control, and cross-functional coordination. Each team member contributed to different aspects of the application, ensuring that the final product felt cohesive and user-friendly. Some of our key collaborative achievements include:

- **Cross-module integration:** Data from the login and signup modules seamlessly connects to event creation and profile tracking, ensuring a smooth user flow throughout the app.
- **Consistent UI/UX design:** The team maintained a unified color palette, iconography, and layout across all screens, creating a clean and intuitive interface.
- **Collaborative problem-solving:** Regular discussions helped resolve technical challenges, such as database configuration and MAUI platform issues, through collective troubleshooting.
- **Version control coordination:** We used GitHub to manage branches and commits, allowing multiple members to work simultaneously without major conflicts.
- **Adaptive teamwork:** When hardware and time limitations affected the completion of certain features, the team adapted by redistributing tasks and documenting unfinished components for future implementation.

These collaborative efforts not only enhanced the overall quality of the app but also improved our understanding of teamwork and communication within a real-world software development environment.

**Reflection**

The development of Social Sports Hub was a valuable learning experience that strengthened our technical and collaborative skills. Throughout the project, we successfully implemented the core functionality, including user authentication, event creation, and database integration. However, three major features  the Honor System, Map View, and Weather System  were not fully implemented in the final version due to time and technical limitations.

The Honor System, which was intended to track player reliability and reward consistent participation, proved more complex than anticipated. It required detailed logic for tracking attendance, updating scores, and ensuring fairness between users. With more time, we would plan the logic earlier in development and assign dedicated testing time to ensure smooth integration with the user and event modules.

The Map View, meant to display nearby games through the Google Maps API, was tested separately but not merged due to Zain’s hardware difficulties and multiple conflicts when

syncing the code from other contributors. In future projects, we would establish clearer version control practices and allocate additional buffer time for integration testing.

Finally, the Weather System, which was designed to provide live weather updates for events, was deprioritised to focus on essential components. Next time, we would manage our time more effectively by setting development milestones and using agile-style iterations to ensure secondary features like weather and map functionality are implemented progressively.

Overall, the project taught us the importance of collaboration, time management, and flexibility. While some goals remained unfinished, the experience gave us a deeper understanding of real-world software development challenges and how to plan better for future improvements.

**5. Acknowledgments and AI Usage**

Generative AI (ChatGPT, GPT-5) was used only in supportive, non-substitutive ways:

- To set up the blank .NET MAUI project structure, ensuring correct file hierarchy and dependencies (App.xaml, MainPage.xaml, ViewModels folder, etc.).
- To refine front-end XAML layouts, ensuring alignment with the intended color palette and design vision.
- To review minor syntax and debugging suggestions when Visual Studio errors were unclear.

All final logic, data models, and UI implementations were written, tested, and validated by the team. The AI contributions were purely assistive similar to using documentation or a style guide and did not generate or complete major portions of the assignment. Human oversight was applied in every case: code was reviewed, tested, and refactored by the team before inclusion.

**6. References**


- Microsoft Docs. (2025). *Entity Framework Core Documentation.* https://learn.microsoft.com/ef/core
- Microsoft Docs. (2025). *.NET MAUI Fundamentals.* https://learn.microsoft.com/dotnet/maui
- NUnit Documentation. (2025). *Testing in C# with NUnit.* https://nunit.org

- Google Developers. (2025). *Maps SDK for .NET MAUI.* https://developers.google.com/maps
- OpenAI. (2025). *ChatGPT (GPT-5)* – used as a writing assistant and debugging reference.