MODEL CURRICULUM
Mobile Game Developer

Sector:	Media and Entertainment
Sub-Sector:	Animation and Gaming
Occupation:	Mobile Game Asset Creation
REF ID:	**MES/Q0510**
NSQF Level:	4
Version:	1.0

Certificate

Table of Contents

Program Overview

Brief Job Description

- Perform preproduction activities (Analyze Game Art)
- Knowledge OOP Languages
- Game Design Principles
- Knowledge about different Game Engines
- Creating Simple Game Interface
- Create simple 2D games
- Design characters
- Maintain workplace health and safety
- Design gameplay mechanics, levels, and user interfaces.
- Write and maintain game design documentation.
- Collaborate with artists, programmers, and writers to ensure cohesive game development.
- Develop and implement game features, such as characters, quests, and dialogue.
- Test and iterate on gameplay to ensure quality and fun.
- Work with producers to manage project timelines, budgets, and resources.
- Stay up-to-date with industry trends, technologies, and player preferences.

Personal Attributes:

- Creativity and imagination.
- Storyline writing skills
- Attention to detail and analytical skills.
- Time management and organization.
- Teamwork and leadership.
- Adaptability and continuous learning. Knowledge of modelling, texturing, rigging, and animation techniques.
- Problem-solving skills and the ability to debug and optimize graphics performance issues.
- Excellent communication and collaboration skills to work effectively with multidisciplinary teams.

This curriculum is structured as per **NSQF Level 4** guidelines with a **450-hour** program, ensuring a structured flow from fundamentals to advanced animation techniques.

Mobile Game Developer
CURRICULUM / SYLLABUS

This program is aimed at training candidates for the job of a "**Mobile Game Developer**", in the "Media and Entertainment" Sector/Industry and aims at building the following key competencies amongst the learner.

Training Delivery Plan	
Program Name	**Mobile Game Developer**
Qualification Pack Name & Ref. ID	**MES/Q2501**
Version No.	**1.0**
Pre-Requisite License or Training	A) 10+2 Pass and Diploma in Animation OR NSQF Level-3 Certification in a relevant field B) Acquaintance with designing and animation
Minimum Job Entry Age	18 years
Minimum Duration of the Course	450 hrs
Training Outcomes	Upon completion of the **Mobile Game Developer Course**, learners will be able to: 1. Develop simple 2D mobile games using game engines like Unity. 2. Apply C# programming and OOP concepts to create gameplay features. 3. Design user interfaces and integrate basic game assets. 4. Test, debug, and optimize games for mobile platforms. 5. Collaborate with teams and maintain game documentation. 6. Follow workplace safety norms and demonstrate professional behavior.

Version Control & Collaboration (Training Delivery Plan / Annexure)

Version Control & Collaboration Policy

- All projects must use **Git** with **Git LFS** for large assets.
- **Branching model**:
    - Main (stable, always releasable)

  - Feature/<name> (short-lived, per task)
  - Release/vX.Y (before production)
  - Hotfix/<name> (urgent fixes)
- **Pull Requests** mandatory for merging to main; must include at least **one reviewer approval** and pass automated build/test checks.
- **Issue tracking**: every PR links to an issue ID in GitHub/Jira for traceability.
- **CI/CD**: integrate with **Unity Cloud Build** or **GitHub Actions** to:
  - Run unit tests and lint checks
  - Build Android AAB / iOS IPA
  - Upload builds for testers
- All **secrets** (keystore passwords, API keys) stored in secure environment variables, not in code.

Applicable National Occupational Standards (NOS)

Compulsory NOS:

- MES/ 00101: Game Fundamentals and Planning
- MES/ 00102: Programming with C#
- MES/ 00103: Game Design Basics and Introduction to Game Engines
- MES/ 00104: Game Design and Prototyping
- MES/ 00105: Polishing and Deployment
- MES/ 00106: Maintain Workplace Health & Safety

NOS and Module Details

S.No.	Module	Theory (h)	Practical (h)	Total (h)
1	Game Fundamentals & Planning	40	10	50
2	Programming with C#	40	60	100
3	Game Design basics & intro to engines	35	50	85
4	Game Design & Prototyping	25	75	100
5	Polishing & Deployment	30	65	95
6	Maintain Workplace Health & Safety	5	15	20
	Total	**175**	**275**	**450**

Credits	Total Notional Hours	Hours per Credit
15	**450**	**1 Credit = 30 hrs (NCrF)** **ABC Compliant**

Performance Criteria (PC) and Knowledge & Understanding per Module
Module 1: Game Fundamentals and Planning

Terminal Outcomes:
Upon completing the course, student will be able to:
1. Design and Develop Games: Create a fully functional game using Unity game engine, applying game design principles and project planning techniques.
2. Apply Game Development Process: Understand and apply the game development process, including concept development, game design, prototyping, testing, and iteration.
3. Use Unity Game Engine: Effectively use Unity game engine to create 2D or 3D games, including setting up scenes, objects, physics, and animations.
4. Collaborate with Team Members: Work effectively in a team environment, communicating ideas, and collaborating on game development projects.
5. Create Game Design Documents: Write clear and effective game design documents, outlining game mechanics, art style, and technical requirements.
6. Develop Problem-Solving Skills: Apply problem-solving skills to overcome technical challenges and creative blocks in game development.
7. Apply Project Planning Techniques: Plan and manage game development projects, including setting goals, prioritizing tasks, and meeting deadlines.

Performance Criteria (PC)

- PC1: Game development overview
- PC2: Game design principles, and project planning
- PC3: Game engines (Unity) introduction and setup

Module	Key Artefacts (Evidence Outputs)	Linked Performance Criteria (PCs)	Notes for AA Verification
Game Fundamentals & Planning	Game Design Document (GDD) – Scope triangle (features, cost, time) – User stories / backlog items	PC1: Game development overview PC2: Game design principles & project planning.	Demonstrates planning & documentation skills

Knowledge and Understanding
A. Organizational Context

- KA1: Team collaboration, communication
- KA2: Game concept and idea development, writing game design documents.

B. Technical Skills

- KB1: Basic Game Plan understanding.
- KB2: Knowledge about basic gaming application design, idea and planning a basic game.

C. Core Generic and Professional Skills

- KC1: Developing an eye for detail and creativity in game design.
- KC2: Understanding client requirements and feedback incorporation.

Module 2: Programming with C#

Terminal Outcomes:
Upon completing this course, students will be able to:
1. Write Basic C# Programs: Write simple C# programs using variables, data types, operators, control structures, and functions.
2. Apply Object-Oriented Programming Concepts: Apply object-oriented programming (OOP) concepts such as classes, objects, inheritance, polymorphism, and encapsulation to create reusable code.
3. Use C# Collections and Data Structures: Use C# collections and data structures such as arrays, lists, dictionaries, and sets to store and manipulate data.
4. Handle Errors and Exceptions: Handle errors and exceptions in C# programs using try-catch blocks and throw exceptions when necessary.
5. Develop Game-Related Logic: Develop game-related logic using C# programming, including game loops, event handling, and collision detection.
6. Create and Manipulate Game Objects: Create and manipulate game objects using C# classes and objects, including properties, methods, and events.
7. Apply Debugging Techniques: Apply debugging techniques to identify and fix errors in C# programs.

Performance Criteria (PC)

- PC1: Familiarize with industry-standard gaming designing programming languages and software (C#, GoDot, Defold etc.,).
- PC2: Learn workspace fundamentals of computers, Introduction to programming languages; C# programming with arrays, functions, control structures.
- PC3: Understanding C# with basic programs.

Module	Key Artefacts (Evidence Outputs)	Linked Performance Criteria (PCs)	Notes for AA Verification
Programming with C#	C# katas (loops, collections, events) Physics/collision mini-demo	PC2: Learn workspace fundamentals PC3: Understanding C# with basic programs	Shows practical coding competency

Knowledge and Understanding
A. Organizational Context

- KA1: Industry expectations for C# and programming.
- KA2: Different OOPs concepts required for mobile game development.

B. Technical Skills

- KB1: C# Programming with understanding the concepts.
- KB2: Understanding the generation of output using programs.

C. Core Generic and Professional Skills

- KC1: Problem-solving using C# to generate Game ideas.
- KC2: Effective logic management required for game design.

Module 3: Game Design Basics and Introduction to Game Engines

Terminal Outcomes

Upon completing this course, students will be able to:

1. Design Engaging Games: Create game designs that incorporate rules, interaction principles, and player motivation, demonstrating an understanding of game design basics.

2. Apply Game Engine Fundamentals: Understand and apply game engine fundamentals, including scenes, components, assets, and prefabs, to create game environments.

3. Script Game Logic: Use C# programming to script game logic, including interactions, behaviors, and events, in a game engine.

4. Understand Game Loop Mechanics: Understand the game loop and its role in game development, including update, render, and input handling.

5. Design User Interfaces: Design intuitive and engaging user interfaces for games, including HUDs, menus, and other interactive elements.

6. Balance Game Mechanics: Apply game balancing principles to create engaging and challenging gameplay experiences.

7. Create Interactive Game Objects: Create interactive game objects using prefabs, components, and scripts, demonstrating an understanding of game object manipulation.

Performance Criteria (PC)

- PC1: Rules and interaction principles of game design.
- PC2: Understanding game objects like Scene, Components, Assets, prefabs.
- PC3: Game Scripting using C#.
- PC4: Understanding Game loop, Player Motivation, User interface designs, Game balancing.

Module	Key Artefacts (Evidence Outputs)	Linked Performance Criteria (PCs)	Notes for AA Verification
Game Design Basics & Intro to Engines	Unity scene file with prefabs. Input system demo. UI elements (menus/HUD) Basic state machine logic	PC1: Rules & interaction principles PC2: Game objects (scenes, assets, prefabs) PC3: Game scripting in C# PC4: UI design, game balancing	Connects design theory to working Unity artefacts

Knowledge and Understanding

A. Organizational Context

- KA1: Industry trends in animation Game development.
- KA2: Legal and copyright considerations in Mobile Game production.

B. Technical Skills

- KB1: applicability of C# in developing and designing of Scene using C#.
- KB2: Integrating physics-based animation tools.
- KB3: Concept of creating storytelling and scene designing.
- KB4: Deciding the target audience.

- HB5: Prototyping and Feedback mechanisms

C. Core Generic and Professional Skills

- KC1: Working collaboratively in a team environment.
- KC2: Analyzing and refining Game sequences for improvements.
- KC3: Analyzing scene requirements, scene environments, and challenges
- KC4: Ideas of camera angles, lighting, state machines, layers, pipeline, and tags.

Module 4: Game Design and Prototyping

Terminal Outcomes:

1. Develop Simple Game Scenes: Create simple game scenes using C# programming, incorporating essential gameplay elements such as movement, interaction, or combat.

2. Create and Refine Prototypes: Quickly create and refine prototypes to explore different game ideas, focusing on core mechanics and gameplay elements.

3. Design and Test Gameplay Mechanics: Design and test gameplay mechanics, including movement, interaction, and combat, to create engaging gameplay experiences.

4. Create Playable Prototypes: Create playable prototypes to test core mechanics and identify areas for improvement.

5. Build Polished Game Sections: Build polished, representative sections of games to demonstrate their potential and showcase gameplay features.

6. Sketch and Test Game Ideas: Sketch and test game ideas on paper or digital drawing tools, iterating on designs and refining concepts.

7. Create Interactive Prototypes: Create interactive prototypes using game engines, tools, or programming languages, demonstrating an understanding of game development principles.

Performance Criteria (PC)

- PC1: Development of Simple game Scene using C#.
- PC2: Quickly creating and refining prototypes to explore different ideas.
- PC3: Focusing on the essential gameplay elements, such as movement, interaction, or combat.
- PC4: Creating a simple, playable version of the game to test core mechanics.
- PC5: Building a polished, representative section of the game to demonstrate its potential.
- PC6: Sketching and testing game ideas on paper or digital drawing tools.
- PC7: Creating interactive prototypes using game engines, tools, or programming languages.

Module	**Key Artefacts (Evidence Outputs)**	**Linked Performance Criteria (PCs)**	**Notes for AA Verification**
Game Design & Prototyping	Playable gray-box prototype. Telemetry logs (player actions, errors)	PC1: Simple game scene using C# PC2: Quick prototyping PC4: Playable core mechanics PC5: Polished demo section	Allows AA to validate iteration + player feedback

Knowledge and Understanding

A. Organizational Context

- KA1: Understanding project timelines and deadlines.
- KA2: Working with clients and understanding industry standards for project delivery.

B. Technical Skills

- KB1: Gathering player feedback and iterating on the prototype to improve gameplay.
- KB2: Ensuring the prototype can be expanded or modified to accommodate new features.
- KB3: Designing prototypes to be reusable, reducing waste and saving resources.
- KB4: Exporting Game in various formats for mobile like android and iOS and Web application

C. Core Generic and Professional Skills

- KC1: Presentation and communication skills for portfolio showcasing.
- KC2: Adaptability and continuous learning for career growth in animation.

Module 5: Polishing and Deployment

Terminal Outcomes:

Upon completing this course, students will be able to:

1. Identify and Resolve Technical Issues: Identify and resolve technical issues, glitches, and errors in game development, ensuring a stable and smooth gameplay experience.

2. Balance Gameplay Mechanics: Adjust gameplay mechanics, difficulty levels, and progression to create a smooth and engaging experience for players.

3. Refine Game Presentation: Refine graphics, sound effects, music, and overall presentation to create an immersive and polished game experience.

4. Optimize User Interface: Streamline menus, HUD elements, and overall player interaction to create an intuitive and user-friendly interface.

5. Ensure Cross-Platform Compatibility: Ensure the game runs smoothly on various hardware configurations and platforms, including PC, consoles, mobile devices, and web platforms.

6. Prepare Game for Release: Prepare the game for release, including testing, quality assurance, and deployment.

7. Conduct Thorough Testing: Conduct thorough testing to ensure stability, quality, and overall player satisfaction.

8. Create Promotional Materials: Create trailers, screenshots, and promotional materials to generate buzz and excitement for the game.

9. Maintain Post-Launch Engagement: Provide updates, patches, and DLC to maintain player engagement, address feedback, and ensure the game remains relevant.

Performance Criteria (PC)

- PC1: Identifying and resolving technical issues, glitches, and errors.
- PC2: Adjusting gameplay mechanics, difficulty levels, and progression to create a smooth experience.
- PC3: Refining graphics, sound effects, music, and overall presentation.
- PC4: Streamlining menus, HUD elements, and overall player interaction
- PC5: Ensuring the game runs smoothly on various hardware configurations.
- PC6: Preparing the game for release on PC, consoles, mobile devices, or web platforms.
- PC7: Conducting thorough testing to ensure stability and quality.
- PC8: Creating trailers, screenshots, and promotional materials to generate buzz.
- PC9: Providing updates, patches, and DLC to maintain player engagement and address feedback.

Module	Key Artefacts (Evidence Outputs)	Linked Performance Criteria (PCs)	Notes for AA Verification
Polishing & Deployment	Profiler screenshots (optimization evidence) Build pipeline exports (AAB for Android, IPA for iOS) Store-ready metadata (title, screenshots, description)	PC1: Technical issue resolution PC2: Gameplay balancing PC3: Refining presentation PC5: Cross-platform builds PC8: Promotional materials	Demonstrates industry-standard release readiness

Knowledge and Understanding

A. Organizational Context

- KA1: Testing Game designed.
- KA2: Arranging and connecting various levels involved in game projects.
- KA3: Adding appropriate Lights, Sounds for the game.
- KA4: Designing and finalizing User interface for the game.
- KA5: Testing the game in multiple platforms and ensuring the running capability in multiple platforms and with minimal hardware requirements.
- KA6: Preparing finalized game for different platforms.

B. Technical Skills

- KB1: Testing the game on multiple platforms after finalizing.
- KB2: Creating promotional materials for launching the game.
- Testing final releases of the game.

C. Core Generic and Professional Skills

- KC1: Conducting thorough testing to ensure stability and quality.
- KC2: Creating trailers, screenshots, and promotional materials to generate buzz.
- KC3: Releasing the game through digital storefronts, physical copies, or publisher partnerships.
- KC4: Releasing the game through digital storefronts, physical copies, or publisher partnerships.

Monetization & Compliance (Module 5: Polishing & Deployment)

Monetization & Compliance (Intro)

- **In-App Purchases (IAPs):** implement consumables, non-consumables, and subscriptions using **Google Play Billing** and **Apple StoreKit**. Validate receipts server-side.
- **Advertising:** integrate SDKs (AdMob, Unity Ads, etc.) responsibly; enable COPPA/GDPR compliance modes as needed.

- **Analytics & Crash Reporting:** use **Firebase Crashlytics** for crash logs, performance dashboards, and release stability.
- **Privacy & Compliance:**
  - **COPPA:** if app is child-directed, obtain verifiable parental consent before data collection.
  - **GDPR:** provide clear privacy policy, consent flows, and user rights (access, erase, withdraw).
  - **API Key Hygiene:** never hard-code keys; use secure storage and rotate when necessary.

Module 6: Maintain Workplace Health & Safety

Terminal Outcomes:

Upon completing this course, students will be able to:

1. Comply with health, safety, and security policies applicable to game/software development labs.
2. Apply safe working practices including ergonomics for desk setup (DSE compliance), screen-time management, and prevention of repetitive strain injuries.
3. Implement data security and cyber safety practices including access controls for repositories, API key hygiene, and safe handling of credentials.
4. Understand government and organizational norms relating to health, safety, and digital security.
5. Identify risks and hazards in a game development environment (electrical, fire, network/data breaches) and apply mitigation measures.
6. Respond appropriately to emergencies and recommend improvements in workplace safety and security.

Performance Criteria (PC)

1. PC1: Follow ergonomic and workstation safety guidelines for extended coding/design work.
2. PC2: Apply secure practices in version control systems (Git repos, access controls).
3. PC3: Manage sensitive assets (API keys, SDK licenses) securely.
4. PC4: Understand and follow lab-specific health and safety SOPs.

Module	Key Artefacts (Evidence Outputs)	Linked Performance Criteria (PCs)	Notes for AA Verification
M6: Workplace Health & Safety (Game/Software Lab)	WHS/Cyber checklist (backups, repo access controls, API key hygiene, ergonomics compliance)	PC1: Follow ergonomic & safety guidelines. PC2: Data security/cyber safety. PC3: First aid & emergency response	Provides tangible compliance evidence

Knowledge and Understanding

A. Organizational Context

- KA1: Safety and security regulations for **game/software labs**.
- KA2: Ethical and professional conduct in collaborative software projects.

B. Technical Skills

- KB1: Ergonomic use of hardware (PCs, VR/AR devices, mobile testing rigs).
- KB2: Secure repo management, backup/recovery strategies, and credential management.

C. Core Generic and Professional Skills

- KC1: Risk assessment in a digital/game development environment.
- KC2: Effective teamwork in maintaining a safe and secure lab environment.

Trainer Requirement and Certification

Trainer Requirements:

- Minimum qualification: **B.Tech/BE (CSE/IT) or B.Sc./BCA in Computer Science/IT**.
- Must demonstrate a **portfolio of Unity/C# prototypes**.
- Minimum **3 years' experience in shipped prototypes** or equivalent industry work.
- **Certified ToT (Training of Trainers)** under NSQF guidelines.

Trainer Certification:

- Must have completed **NSQF-certified ToT program**.
- Must have relevant project experience in **mobile game development using Unity/C#**.

Assessor Requirements and Certification

Assessor Requirements:

- Minimum **5 years of experience** in mobile game development (Unity/C#).
- Must have **shipped apps/prototypes** in the mobile gaming domain.
- Should be a **certified assessor** under NSQF guidelines.
- Must demonstrate familiarity with the **Awarding Body's Assessment Agency SOP**.

Assessor Certification:

- Certified assessor under NSQF / NCVET.
- Demonstrated experience in evaluating game development projects (Unity/C#).

Assessment Strategy

Assessment Strategy and Weightage

- Theory and practical assessment will be conducted in a 30:70 ratio.
- Evaluations will include written exams, practical assignments, and viva sessions.
- **Project marks** shall be drawn primarily from **Module 4 (Game Design & Prototyping – playable prototype)** and **Module 5 (Polishing & Deployment – release-ready build + metadata)**.
- Evidence artefacts must be submitted per module for external verification.

Passing Rule

To qualify for certification, the candidate must score at least 70% in aggregate across Theory, Practical, Project, and Viva mapped to all modules/NOS.

Assessment Governance

"Summative assessments and certification shall be conducted by the NCVET-recognized Awarding Body through NCVET-recognized Assessment Agencies. Internal juries shall be used only for formative assessments. All practical assessments will be 100% video-recorded as per SSC/Awarding Body Standard Operating Procedures (SOP)."

Module / NOS	Theory Evidence	Practical Evidence	Project Evidence	Viva Evidence
M1: Game Fundamentals & Planning	Written test on concepts, design principles	Paper GDD, scope triangle, backlog	–	Oral defense of design choices
M2: Programming with C#	Code snippets (loops, OOP)	Live coding demo, mini collision project	–	Q&A on debugging & OOP
M3: Game Design Basics & Engines	Short answers on rules & interaction	Unity scene with prefabs, UI, state machine	–	Walkthrough of design rationale
M4: Game Design & Prototyping	MCQs on prototyping concepts	Gray-box prototype build	**Playable prototype with telemetry logs**	Viva on iteration & feedback
M5: Polishing & Deployment	Written test on optimization & deployment	Lab demo of profiler use, build pipeline	**Release-ready build (AAB/IPA) + store metadata**	Viva on optimization & release pipeline
M6: Workplace Health & Safety	Written test on WHS & cyber safety	WHS/Cyber checklist (backup, API hygiene)	–	Viva on risk scenarios

NOS	Theory Marks	Practical Marks	Project Marks	Viva Marks	Total Marks	Weightage
Game Fundamentals and Planning	50	-	-	20	70	14%
Programming with C#	20	50	20	10	100	20%
Game Design Basics and Introduction to Game Engines	20	30	40	10	100	20%
Game Design and Prototyping	20	30	40	10	100	20%
Polishing and Deployment	30	30	30	10	100	20%
Workplace Safety	10	20			30	6%
Total	**150**	**160**	**130**	**60**	**500**	**100%**

"All assessment artefacts are mapped to the **Performance Criteria (PCs)** and **Terminal Outcomes (TOs)** defined under each module. The assessor will verify artefacts against PCs during external evaluation."

Glossary of Key Terms

- **Game Development**
  - **2D/3D Game:** A game with two-dimensional or three-dimensional graphics.
  - **Asset:** A reusable piece of content, such as images, audio, or 3D models.
  - **Collision Detection**: The process of detecting when objects intersect or collide.
  - **Game Engine:** A software framework for building games, providing tools and libraries.
  - **Level Design**: The process of creating game levels, including layout, obstacles, and challenges.
- **Programming**
  - **API (Application Programming Interface):** A set of rules and protocols for interacting with software components.
  - **C++:** A programming language commonly used for game development.
  - **C#**: The official and primary programming language used for Unity game development.
  - **Java:** A programming language used for Android game development.
  - **JavaScript:** A programming language used for web-based game development.
- **Graphics and Animation**
  - **Animation:** The process of creating the illusion of movement using static images or 3D models.
  - **GPU (Graphics Processing Unit):** A hardware component responsible for rendering graphics.
  - **Particle Effects:** Visual effects, such as explosions or fire, created using small images or particles.
  - **Shader:** A program that runs on the GPU, controlling the appearance of 3D objects.
  - **Sprite:** A 2D image or animation used in games.
- **Audio**
  - **Audio Asset:** Sound effects, music, or voiceovers used in a game.
  - **Audio Implementation:** The process of integrating audio assets into a game.
  - **FMOD:** A popular audio middleware solution for games.
  - **Sound Design:** The process of creating and editing audio assets for games.
  - **Wwise:** A popular audio middleware solution for games.
- **Testing and Optimization**
  - **Benchmarking:** Measuring the performance of a game on different hardware configurations.
  - **Bug Fixing:** Identifying and resolving issues or bugs in a game.
  - **Optimization:** Improving the performance, efficiency, or quality of a game.

- **Profiling:** Analyzing the performance of a game to identify areas for optimization.
- **Testing:** Verifying that a game meets its requirements and works as expected.

- **Platforms**
  - **Android:** A mobile operating system developed by Google.
  - **iOS:** A mobile operating system developed by Apple.
  - **Cross-Platform Development:** Building games that run on multiple platforms.
  - **Mobile Game:** A game designed for mobile devices, such as smartphones or tablets.
  - **PC Game:** A game designed for personal computers.

Annexure

List of Recommended Software and Tools:

- **Game Engines**
  - **Unity:** A popular cross-platform game engine for 2D and 3D games.
  - **Unreal Engine:** A powerful game engine for high-performance, visually stunning games.
  - **Godot Engine:** An open-source game engine for 2D and 3D games.
  - **Construct 3:** A HTML5-based game engine for creating 2D games without coding.
- **Level Design and Mapping**
  - **Tiled:** A free, open-source level editor for creating 2D game levels.
  - **Unity Terrain Tools:** A set of tools for creating and editing terrain in Unity.
  - **Unreal Engine's Terrain Tools:** A set of tools for creating and editing terrain in Unreal Engine.
- **Audio**
  - **FMOD:** A popular audio middleware solution for games.
  - **Wwise:** A popular audio middleware solution for games.
  - **Audacity**: A free, open-source audio editing software.
- **Version Control**
  - **Git:** A popular version control system for tracking changes in code and assets.
- **Programming and Scripting**
  - **Visual Studio Code:** A lightweight, open-source code editor.
  - **C++:** A popular programming language for game development.
  - **C#:** A popular programming language for Unity game development.
- **Testing and Debugging**
  - **Unity Profiler:** A tool for analyzing and optimizing game performance in Unity.
  - **Unreal Engine's Profiler:** A tool for analyzing and optimizing game performance in Unreal Engine.
  - **Debuggers:** Tools for identifying and fixing issues in code.
- **Other Tools**
  - **Project Management Tools:** Tools like Trello, Asana, and Jira for managing game development projects.
  - **Collaboration Tools:** Tools like Slack, Discord, and GitHub for team communication and collaboration.
  - **Game Development Communities:** Online communities like Gamasutra, and Reddit's r/gamedev for connecting with other game developers.

Platform Readiness

Platform Readiness & Release Checklist

Android Build & Deploy

- Generate an upload **keystore** (backup securely, never commit to repo).
- Build and sign an **Android App Bundle (.aab)** for release.
- Upload AAB to **Google Play Console** → enroll in Play App Signing → release to Internal track → promote to Closed/Open/Production tracks.
- Prepare release metadata (app name, descriptions, screenshots, privacy policy URL).
- Run pre-launch report and device tests before publishing.

iOS Build & Deploy

- Enroll in **Apple Developer Program**; create Bundle ID in App Store Connect.
- Manage signing certificates and **provisioning profiles** via Xcode or Apple Developer portal.
- Archive and export **IPA** build in Xcode; distribute via **TestFlight** for testing.
- Upload final build with release notes and metadata for App Store approval.

Device Testing Matrix (example template)

Priority	Platform	Device example	OS versions	Screen res/DPI	GPU class	Test focus
1	Android	Low-end 2GB RAM	Android 9–10	720×1280	Mali/Adreno low	Install/update, cold start
1	Android	Mid-range 4–6GB	Android 11–12	1080×2340	Mid GPU	Performance & UI
1	iOS	iPhone (recent)	Latest iOS	Varying	Apple A-series	TestFlight, feedback
2	Android	Tablet	Recent	1200×1920	Mid	Orientation, multi-window
3	Android	Go/low storage	Older	Small res	Low	Onboarding, storage stress

Reference Books and Learning Materials:

- **Unity Learn** (https://learn.unity.com) – official Unity tutorials & pathways
- **Unity Manual & Scripting API** (https://docs.unity3d.com/Manual/index.html, https://docs.unity3d.com/ScriptReference/)
- **Godot Engine Docs** (https://docs.godotengine.org)
- *Learning C# by Developing Games with Unity* – Harrison Ferrone (modern C# for Unity)

Additional Readings:

Game Development

- **"Game Development with Unity" by Dr. D.S. Shekhawat:** A comprehensive guide to game development with Unity.
- **"Unity Game Development" by K.P. Jayasankar:** A practical guide to game development with Unity.

Unity

- **"Unity in Action" by Dr. D.S. Shekhawat:** A hands-on guide to Unity game development.
- **"Unity Game Development Cookbook" by K.P. Jayasankar:** A collection of recipes and tutorials for Unity game development.

Unreal Engine

- **"Unreal Engine 4 Game Development" by Sandeep Kumar:** A comprehensive guide to game development with Unreal Engine 4.
- **Programming**
- **"C# for Unity Game Development" by K.P. Jayasankar:** A guide to C# programming for Unity game development.