DANIYAL MIRZA

Engine Programmer / Tech Lead

mail@daniyalmirza.com • linkedin.com/in/daniyalmirza • daniyalmirza.com • Irvine, CA

Engine programmer with more than a decade in games, working at the layer the rest of the studio builds on: memory and framerate optimization, Xbox console development, and in-editor tooling, on a heavily modified Unreal branch kept current across major versions. Now leads tech and rendering on an unannounced Obsidian project while still carrying a full engineering load.

LANGUAGES	C++ · Python · C# · PowerShell
ENGINES	Unreal Engine (UE4 to UE5) · Unity
ENGINE AND PLATFORM	Engine upgrades · Xbox console development · Platform certification · Middleware integration · Editor and pipeline tooling · Memory and framerate optimization
UNREAL SYSTEMS	World Partition and HLOD · Level streaming · Chaos physics · Recast and Detour navigation · EQS · Niagara · Unreal Insights · Cooker and Zen
BUILD AND CI	Horde · Jenkins · TeamCity · Perforce · Git
PROCESS	Jira · Confluence

EXPERIENCE

Obsidian Entertainment

Jan 2020 - Present
Engine Programmer / Tech Lead
Irvine, CA

- Own engine-level integration of third-party middleware: Wwise, Triton acoustics and Simplygon.
- Drive memory and framerate optimization across projects, profiling to locate regressions and landing the fixes at the engine and asset layers.
- Provide the DevOps support testing and iteration run on, through CI moves from Jenkins to TeamCity to Horde, so build problems do not become content-team downtime.

Unannounced Title 2025 - Present, Tech Lead, tech and rendering

- Tech lead for tech and rendering on the project, with two engine programmers, a rendering engineer and two technical artists as direct reports, while carrying a full senior engineering load.
- Wrote the team's programmer onboarding path, and occasionally interview engineering candidates.

RUNTIME PERFORMANCE

- Built a loading-screen system that raises streaming throughput while the screen is black, cutting load times between areas on Xbox Series S from 9.8 s to 5.1 s.
- Reworked navmesh streaming inside the engine's navigation system, taking its worst frame from 33 ms to 1.7 ms.
- Reworked AI spawn-point queries in the engine's EQS so one long search can no longer starve the rest, taking one search from 281 ms of work to 7.9 ms.

MEMORY

- Ran the Xbox Series S memory budget program and reclaimed roughly 400 MB with a ray tracing BLAS residency policy in the renderer.
- Found HLOD layers that never streamed out and virtual texture pool settings that had silently done nothing since UE 5.4, and fixed both.

ENGINE UPGRADES

- Carried the project's heavily modified engine from Unreal 5.3 through 5.5, 5.6 and 5.7.
- Ported about 26 targeted Epic fixes into the studio's engine fork between upgrades, declining larger rewrites where the risk outweighed the fix.

BUILD AND COOK

- Enabled incremental cooking for project code, cutting a no-change incremental cook from ~2h23m to ~8m.
- Cut editor startup into the project's largest map from 170 s to 65 s.

STABILITY

- Work the auto-filed crash queue across engine, renderer, editor, middleware and game code, with roughly 100 crash fixes in 2026.

TOOLS

- Built an Unreal Insights query server for live and recorded traces; its first use found a 189 second background build occupying every worker thread in the editor.
- Built editor tools and debug view modes for art and design, including a bulk placement swap tool and actor coloring views that work in packaged builds.

Avowed shipped 2025, Xbox Series X|S, PC

- Owned the engine upgrade path from Unreal 4.20 to 5.3, including the UE4 to UE5 transition taken mid-production, integrating each release into a heavily modified branch and absorbing the fallout for every other discipline.
- Worked Xbox console development through to ship, clearing the compliance failures and cert-blocking defects that gate submission.

- Built and maintained the in-editor tooling design, art and animation used daily, cutting the cost of a content iteration.
- Maintained the project's build system, keeping cook, package and distribution pipelines healthy for a team shipping to two platforms at once.

The Rogue Initiative
Jul 2016 - Jan 2020
Systems Programmer
Canoga Park, CA

- Owned build maintenance and back-end engineering across every studio project, keeping Jenkins CI and the release pipeline running.
- Ran asset-management and runtime optimization work across projects.

Unannounced Client Demo 2019

- Designed the API bridging the engine to a client's proprietary library, exchanging JSON payloads and routing state through GameplayTags so behavior could be rebound without code changes.
- Wrote the Python tooling that converted the client's spreadsheets into engine data tables, holding in-game systems in parity with a server-authoritative source that kept moving.

RedpillVR 2019: Authored interpolative prediction for networked location and animation data, hiding link latency in replicated character movement rather than letting it surface as visible correction.

Unannounced Internal Project 2018, third-person action

- Wrote reusable plugin-based modules for game management, including a save/load system that moved serialization onto a dedicated I/O thread to keep the game thread clear of disk latency.
- Built editor tooling that surfaced pipeline-specific asset management operations inside the editor for the art and animation teams.
- Designed humanoid AI on Unreal's Behavior Tree, Perception and EQS systems, with custom listeners driving decisions off animation state; cover selection and prioritized search patterns among the resulting behaviors.
- Implemented camera locking and prototyped melee and ranged monster behavior prioritized on player position.

Crowe shipped 2017, VR, Oculus Rift, HTC Vive, Tech Lead

- Tech lead on a shipped VR title, owning platform services, UI, game-management and analytics modules, runtime optimization, and CI-backed build management.
- Implemented the bow and arrow the protagonist's kit is built around, a two-handed draw and release, with guns and other projectiles as secondary weapons. A designer tuned its art effects and haptics.
- Carried a hands-on engineering load alongside the lead role: hand presence and grab, world-space UI, and level layout.

Trailer, The Drowned Armory youtu.be/h85oMFKponw · Trailer, Survival Arena mode youtu.be/1u2X_v6wrFU

Possession
Sep 2015 - May 2016
Lead AI Engineer (Unreal), RTS/FPS hybrid, shipped 2016

- Architected the base codebase and designed the RTS AI, threading decision work behind Behavior Trees layered over state machines.
- Built the system joining the strategic RTS layer to the first-person layer, which is the connection a hybrid of the two depends on.

Wildstyle Basketball (Easley-Dunn Productions)
Jul 2015 - Jan 2016
Systems and Gameplay Engineer (Unity), 3x3 street basketball, mobile

- Built a StatSheet-driven data system that let designers drive gameplay behavior from spreadsheets without engineering in the loop.
- Worked directly on the stat delivery architecture and built gameplay modules alongside the AI team.

Tata Consultancy Services (Client: Royal Skandia) *Associate Systems Engineer*
Jul 2011 - May 2012

- Built the Corporate Actions module of a financial management web application in Java and JSP against an Oracle backend.

PERSONAL PROJECTS

Nightspire Engine 2026 - Present, browser as the shipping platform

A game engine written from scratch that treats the browser as a first class shipping target rather than a demo target: a C++20 core compiled to WebAssembly through Emscripten, rendering on WebGPU with no WebGL2 fallback, with Lua 5.4 statically linked as the gameplay scripting language.

- Built around a browser-shaped thread topology: a host thread that owns the DOM and must never block, a game thread running as a worker pthread that owns Lua, a render thread holding the WebGPU device, and a task pool sized from the concurrency the browser reports. Threads need cross origin isolation, so the serving headers are part of the engine rather than of the deployment.
- Instrumentation is part of the definition of done, not a later pass. No system ships without log coverage and tracing on its hot paths through NightInsights, an in-engine tracer that exports Chrome Trace format, with memory tagged per subsystem and held at zero steady-state growth.
- Eleven phases in: an archetype and SoA entity model with component reflection and render extraction, a forward lit pass over WebGPU, a Lua runtime with hot reload, and Dear ImGui tooling with an outliner and inspector. Modules are layered in a strict dependency order from Core up through PAL, RHI, Renderer and World.
- CMake and Ninja against a pinned Emscripten SDK, driven by PowerShell so one set of scripts works on Windows, macOS and Ubuntu. CI builds the wasm target on Ubuntu and Windows, runs a Playwright smoke test in Chromium, and deploys the demo only on green.

Live demo demo.nightspireengine.com

Constrained Dynamics 2015, mass-spring simulation

A deformable cube in C and OpenGL: a lattice of springs holding the cube at its rest shape, deforming on contact with a boundary, and resolving those collisions through the spring system rather than alongside it.

Engine Programming v1.5 2015, USC coursework, third-person arcade shooter

- Implemented low-level networking on the TRIBES model, including a Ghost Manager to replicate object state out to connected clients.
- Wrote a custom physics engine, and an animation system that blended partial and additive animations together programmatically.

Demo video, 1:05 youtube.com/watch?v=9-y0t25nqzc

Engine Programming v1.0 2014, USC coursework, third-person action game: Built the core engine systems behind a single level of gameplay: programmed animation, physics, a particle system, and the game structure holding them together.

EDUCATION

Master of Science, Computer Science Viterbi School of Engineering, University of Southern California 2016

Bachelor of Technology, Computer Science Aligarh Muslim University, Aligarh, India 2011