Cody Van De Mark's Side Projects

Evolution Moon (Steam Release) -
Solo engineer
https://store.steampowered.com/app/1748740/Evolution_Moon_Warfare/

Evolution Moon was a small competitive online multiplayer game I wrote over the course of two months using Unreal engine 4. I was the only engineer on the project. I worked with artist Ben Mauro to envision and create this game.

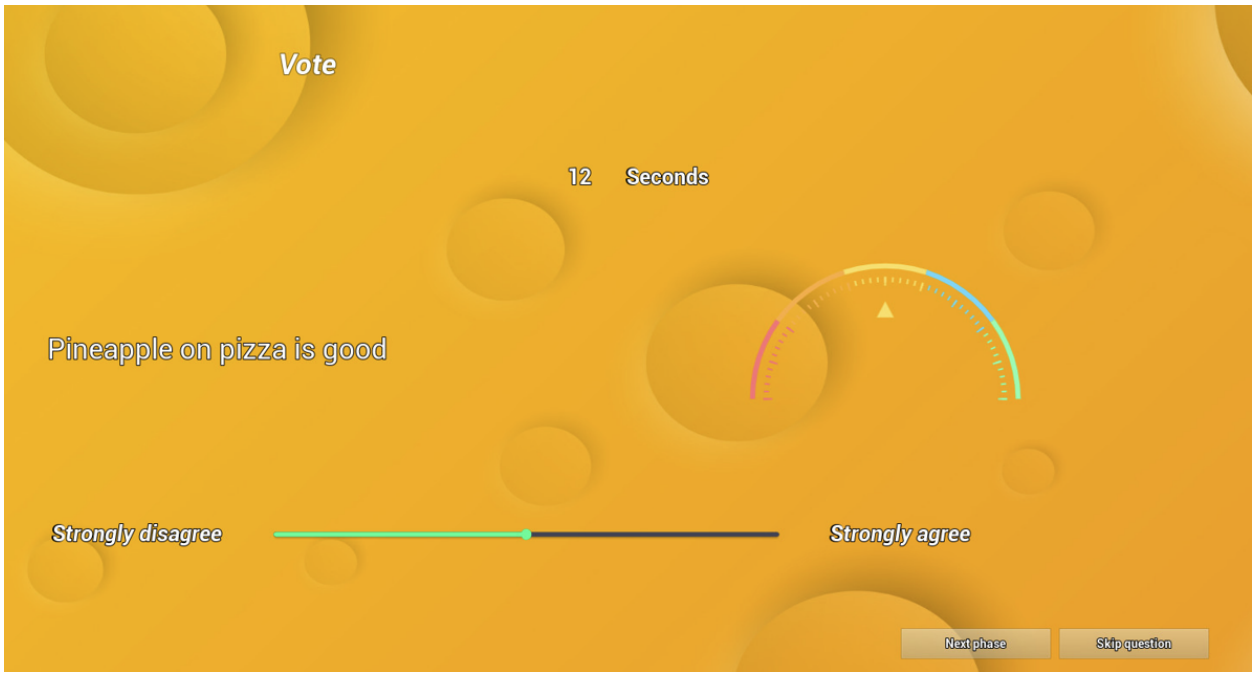
Table Trenches (Android & iOS Release) -
Multiplayer architect & engineer
https://play.google.com/store/apps/details?id=com.db.tabletrenches.release&hl=en_US&gl=US
https://www.dbcreations.studio/table-trenches

Table Trenches is an online competitive multiplayer augmented reality game released on Android and iOS. It is part of the Google Play Pass. The game is written in C# using the Unity engine. I did all of the network and multiplayer code. Two other engineers worked on gameplay while my focus was on all of the networking, data syncing and server/client states. Additionally, I worked on localization and various gameplay state features.

Don’t Get Rejected
Solo engineer
https://www.youtube.com/watch?v=ldMbneq59KQ

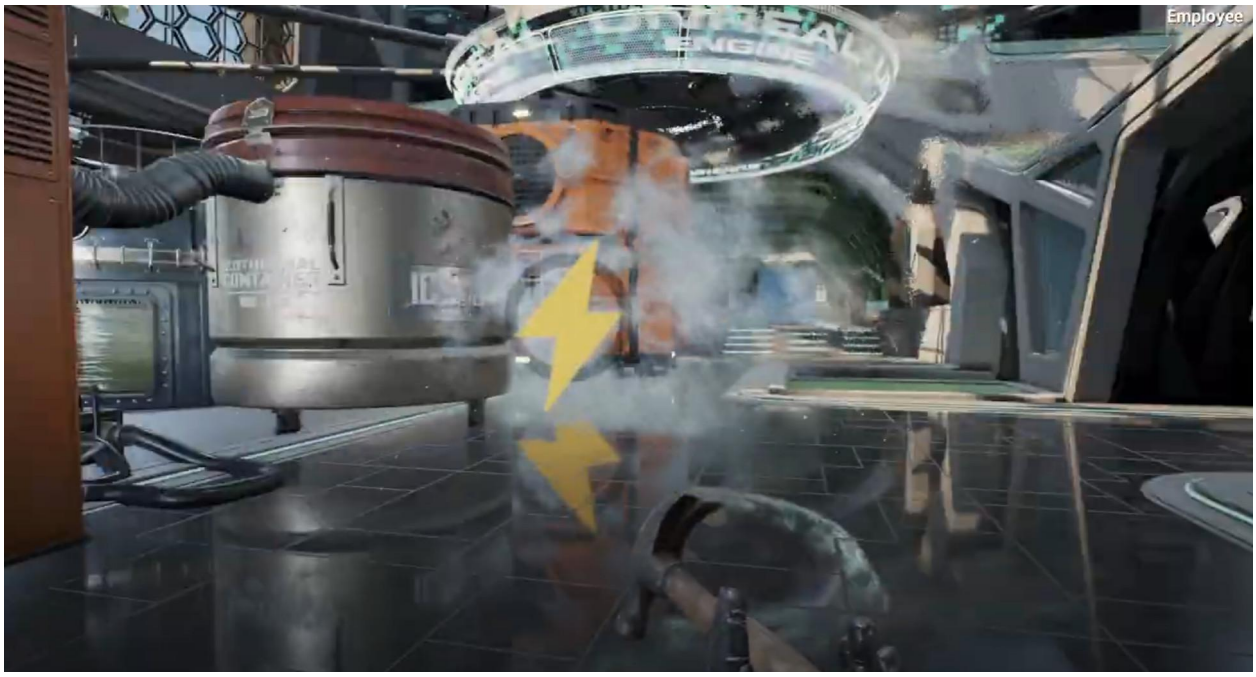
Don’t Get Rejected is a new UE5 game I have been working on. It is a procedural office-based horror game with liminal spaces and non-euclidean geometry. The goal is to make your way to your interview by navigating a procedural environment of strange horrors.

Hot Controversy
Solo engineer



Hot Controversy is an early game concept I am working on in UE4. It is a party game to debate controversial topics with your friends. The development is very early but active. All of the UI work is temporary at this point, but the core game loop is there.

Multiplayer networked repair game
Solo engineer



This is a networked multiplayer proof of concept I was working on in Unreal Engine 4. Machines in the world randomly break and players need to go repair them. Everything is synced. Players can work together to repair machines or fix them separately. One player could start fixing a machine and then go to another machine while a second player finishes repairing the first machine. All of the data is kept in a server-side state. Players sync the minimum necessary to keep optimization high.

VR Shadows and Materials
Solo engineer
Custom shadows and materials in UE4 made with pixel shaders in order to lightening the rendering load for VR. Neither of these examples use translucency or post-processing. Doing this in the pixel shader cuts out some extra steps of rendering and overdraw from translucency.

https://www.youtube.com/watch?v=086E6X-snZU (Shadows done in the pixel shader without translucence or post process)

https://www.youtube.com/watch?v=m01hIJ5raG4 (Material blending done in pixel shader without translucency or post process)

VR Mars Drone
Partner Engineer
Side project in UE4 to create a VR experience where you could drive a robot on the surface of Mars, but the user also has a drone that it can fly with an in-app VR tablet. The VR tablet lets you fly a drone around and see what it sees, as well as has a virtual touch-screen for interaction.

https://www.youtube.com/watch?v=0pVTogu-14s

VR Basketball
Solo engineer
Project put together as a VR demo for when I was a faculty member at RIT. It's basketball in VR for free-throwing.

https://www.youtube.com/watch?v=Zx4Sa_H3Dns

VR Audio Visualizer
Solo engineer
VR audio visualizer I made for the Frameless Labs VR symposium. It takes in real-time audio data from various songs that are playing and maps it to a sphere of triangles acting as speakers in the VR world. It also allows the user to interact with many objects in the environment, change the time of day/skybox, etc.

https://www.youtube.com/watch?v=Ng77i0gBsI0

https://www.rit.edu/framelesslabs/symposium-2021

Desolate Cosmos (Unreal 4 Tech Art & Performance Tuning)
Solo engineer
Desolate Cosmos was a side project I did to specifically learn some tech art techniques and how to optimize performance for Unreal 4. Work was in LODs, HLODs, proxy meshes, cubemaps for reflections, level streaming, custom materials, custom skyboxes, working with game A.I., and a lot of just general perf tuning.

https://www.youtube.com/watch?v=roaUF3Cyk-M (fly-through of environment)

https://renardchien.github.io/DesolateCosmos-PostMortem/ (post-mortem discussing techniques)

https://forums.unrealengine.com/community/released-projects/1497120-desolate-cosmos-demo (release info with a variety of images)

Unreal 4 Networking and Physics Playback
Partner Engineer
I did all of the networking for this project. We were trying to make a sloth soccer game with full online networking. Never got a chance to finish it, but the idea is it uses cubic hermite splines & interpolation to replay the physics state from the server back to clients.The tricky part was that clients had to be able to see instant results so the physics state is cheated on the clients to make it look like it is reacting instantly while it interpolates data from the server. The idea is to combine two splines of movement together.

https://www.youtube.com/watch?v=1GeCTdZTUY4

Unreal 4 Animation Prototyping Tool
Solo engineer
This is a tool I made for artists working in Unreal. It quickly allows artists to drag and drop animation points onto 3d assets. It has a few different animation algorithms they can choose from. That way with just a few mouse clicks they can have an object in the scene and animating actively.

https://www.youtube.com/watch?v=N3PthNHeyGs

Unreal 4 Custom Layered Camera Interface
Solo engineer for this feature, one of several engineers for the game as a whole

Built for the game “Do You Copy?”. This is a layered camera setup I put together because we needed 3d UI that the user could interact with while the world still rendered behind it. This uses two cameras with one acting as a mask. The first camera renders the scene while the second camera only renders the object the user wants to inspect in 3d, and then layers it on top of the first cameras output. Controls are then forwarded between the two cameras when needed.

https://www.youtube.com/watch?v=_HKfWggs4Qs

Unreal 4 Non-Euclidean Geometry
Solo engineer
Uses a number of cameras rendering their view to a flat plane that can be applied to other objects. Based on the angle of the user, it will only render which cameras are actually necessary for performance reasons. Lighting that is “passed through” from these other portals is faked by programming dynamic lights that are timed to lighting events from the other cameras. Similarly, a water material is applied to the ground on one side to create the appearance that the rain is “passing through”.

https://www.youtube.com/watch?v=NM4OvjloNng (video of the system)

Unreal 4 Procedural Meshes & Deformation
Solo engineer
Side project to work with procedural generation of meshes in UE4 along with a tool to allow for deformation of meshes during runtime. Uses a simple ray trace to hit an object, detect that point on the mesh, gather all of the vertices within a range and move them.

https://www.youtube.com/watch?v=zJy8SwE9Dww (Mesh deformation tool)
https://www.youtube.com/watch?v=hqYCG30vtdQ (Sphere mesh generation & deformation)

Unreal 4 Saving & Loading Levels/Objects from a Server
Solo engineer
Tool designed for creating custom levels in Unreal 4 and distributing them. The server is a Node.js server with a MongoDB database. The client is all an Unreal 4 app. The user can download levels (though simple in the demo) and load them in. Alternatively, they can also save their own copy of the level with changes back to the server. There’s also a web UI for editing individual properties of a level.

https://www.youtube.com/watch?v=FnTkR_GlVQQ (Web interface for editing individual objects in a level)

https://www.youtube.com/watch?v=u1RPGOR1OQ0 (Downloading levels on the fly from a server)

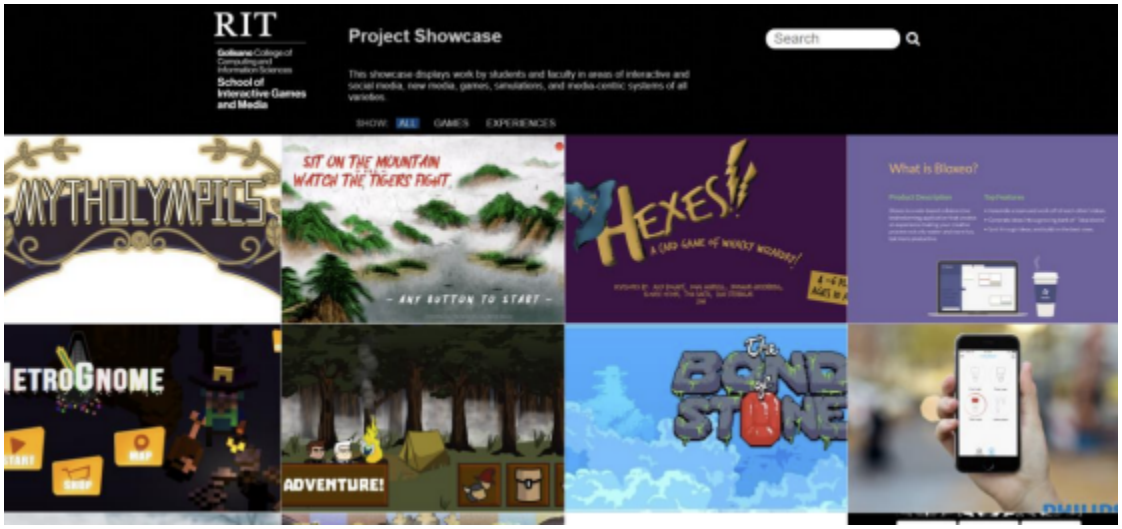
DriftwoodRP (Real-Time Web Server for Dungeons & Dragons)
One of two engineers - was responsible for backend, database and real-time networking

DriftwoodRP was an open source web CMS for Dungeons & Dragons. Though I made this in 2012, it is similar to modern tools like Roll20 or Fantasy Grounds. It provided a collaborative space for people to create, manage & play Dungeons & Dragons. Service had an account system, permission system, content management system, game streaming via websockets with HTML5 canvas, live chat, image uploading and more. Unfortunately, I could not afford the expenses of maintaining it.

Client was HTML5, CSS3, Javascript, Canvas, Websockets.
Server was Node.js with MongoDB, Websockets and a REST api.

RIT's Interactive Games & Media Gallery
Solo engineer
Gallery I made for students at the Rochester Institute of Technology to show off their work. The requirements on this project were restrictive. All of the HTML, CSS and JS are my own, but in order for it to work within RIT's servers, I was restricted to a limited number of files (including HTML, CSS and JS) and it all had to be running in an embedded web page. As a result, the codebase is a bit limited, but it does run well and works on many different browsers/devices. Adding content is merely editing a JSON file with the new projects to add.

http://people.igm.rit.edu/portfolios/Gallery.html



Just Press Play (Web-based Achievement Tracking System)
Backend & database engineer
Just Press Play was a Microsoft sponsored web-based achievement system for students that acts as a game to encourage students to become more involved in academics & social interactions. I worked in HTML5, CSS, Javascript, PHP & SQL. Initially I was a developer, but then started working on the expanded system architecture.

https://www.microsoft.com/en-us/research/project/just-press-play/
https://www.rit.edu/showcase/index.php?id=168

JS Raytracing (3d rendered to 2d canvas)
Solo engineer
This was a personal education project I did in order to improve my understanding of raytracing in 3d space. The original raytracer example is linked below. I followed the initial tutorial, then added a lot of features in order to solidify my understanding. I added features to move cameras, move light sources, change field of view, change render depth, etc.

https://s3.amazonaws.com/cvmstatic/Raytracer/raytracer.html (my customized version)
https://tmcw.github.io/literate-raytracer/index.html (original raytracing tutorial)