Лабораторна робота №1

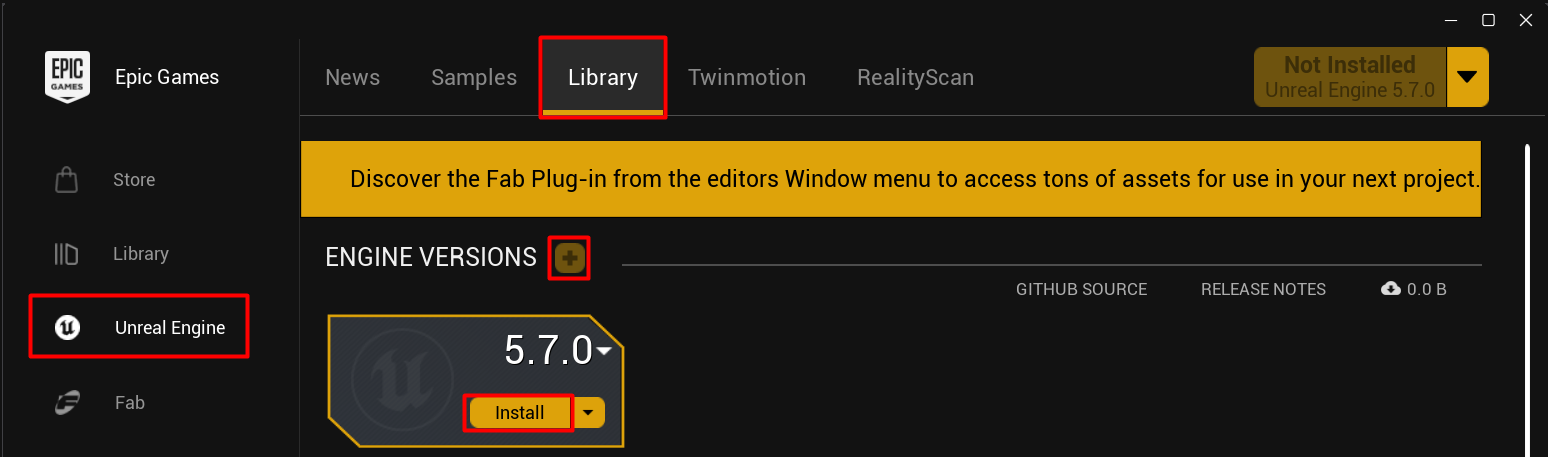
**ПОЧАТОК РОБОТИ З UNREAL ENGINE 5**

**Мета:** набуття практичних навичок роботи з рушієм Unreal Engine 5.

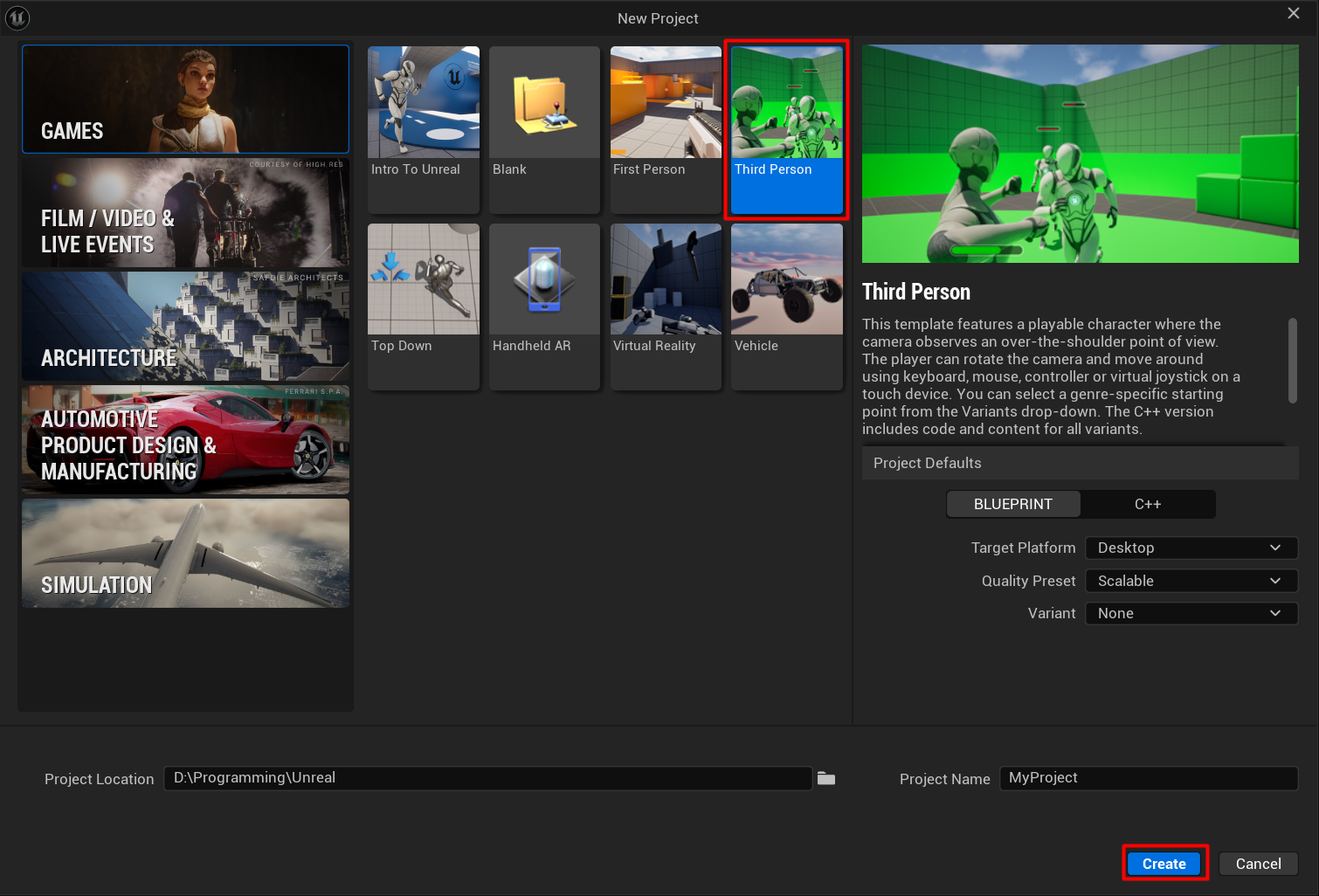
**Хід роботи**

На даний момент *Epic Games* та *Visual Studio Code* вже встановлені у систему.

Відкриваємо *Epic Games Launcher* та встановлюємо *Unreal Engine 5*. Для цього переходимо до вкладки *Unreal Engine*, на ній відкриваємо вкладку *Library*. Далі натискаємо на іконку «+» у полі Engine Versions та натискаємо кнопку «*Install*» (рис. 1).

  
Рисунок 1 – Встановлення Unreal Engine 5

Запускаємо Unreal Engine та створюємо *Third Person* проект (рис. 2).

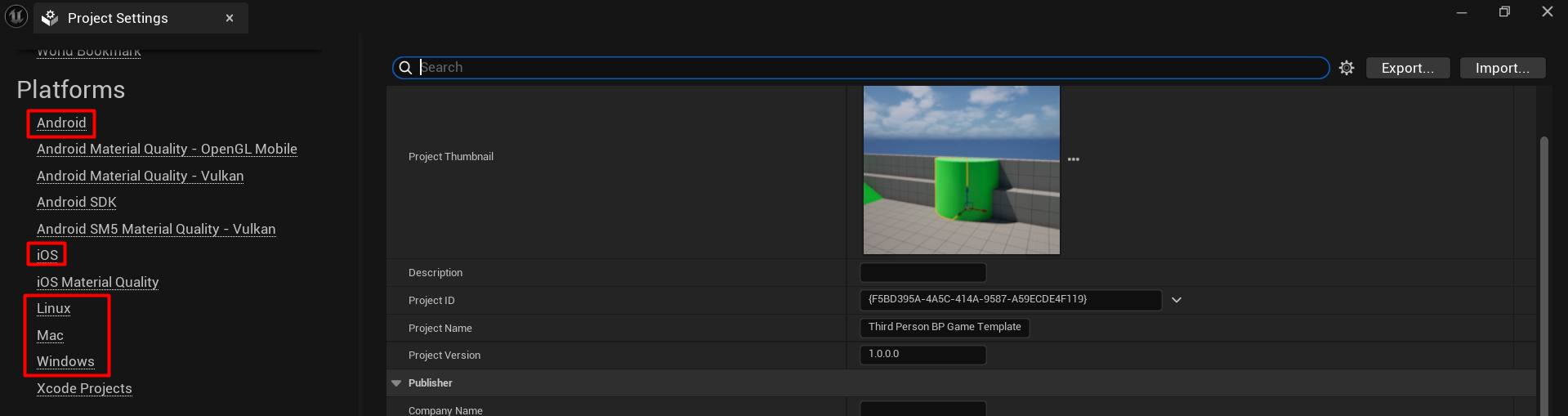
  
Рисунок 2 – Створення проекту Unreal Engine 5

**Blueprint** – це візуальна скриптова мова, вбудована в Unreal Engine. Вона дозволяє створювати ігрову логіку за допомогою вузлів та зв’язків, без написання коду.

У порівнянні з *Blueprint*, клас C++ надає доступ до розширеного функціоналу рушія. Також класи C++ мають більшу продуктивність, особливо при важких обчисленнях.

**Raytracing** (трасування променів) – це метод рендерингу, при якому світлові промені симулюються для створення фотореалістичного освітлення, тіней та відбиття.

У налаштуваннях проекту користувач може обрати одну з наступних платформ: Android, iOS, Linux, MacOS, Windows (рис. 3).

  
Рисунок 3 – Платформи на вибір

**Starter Content** – це набір базових ресурсів, який включає: матеріали, статичні сітки, частинки, звуки, демонстраційні карти.

Сценарій з активним стартовим контентом пришвидшує прототипування гри, дозволяє одразу тестувати логіку та візуальні ефекти та взагалі зручний для навчання.

Сценарій без активного стартового контенту – це по суті пустий, чистий проект, який потребує завантаження або створення власних ресурсів.

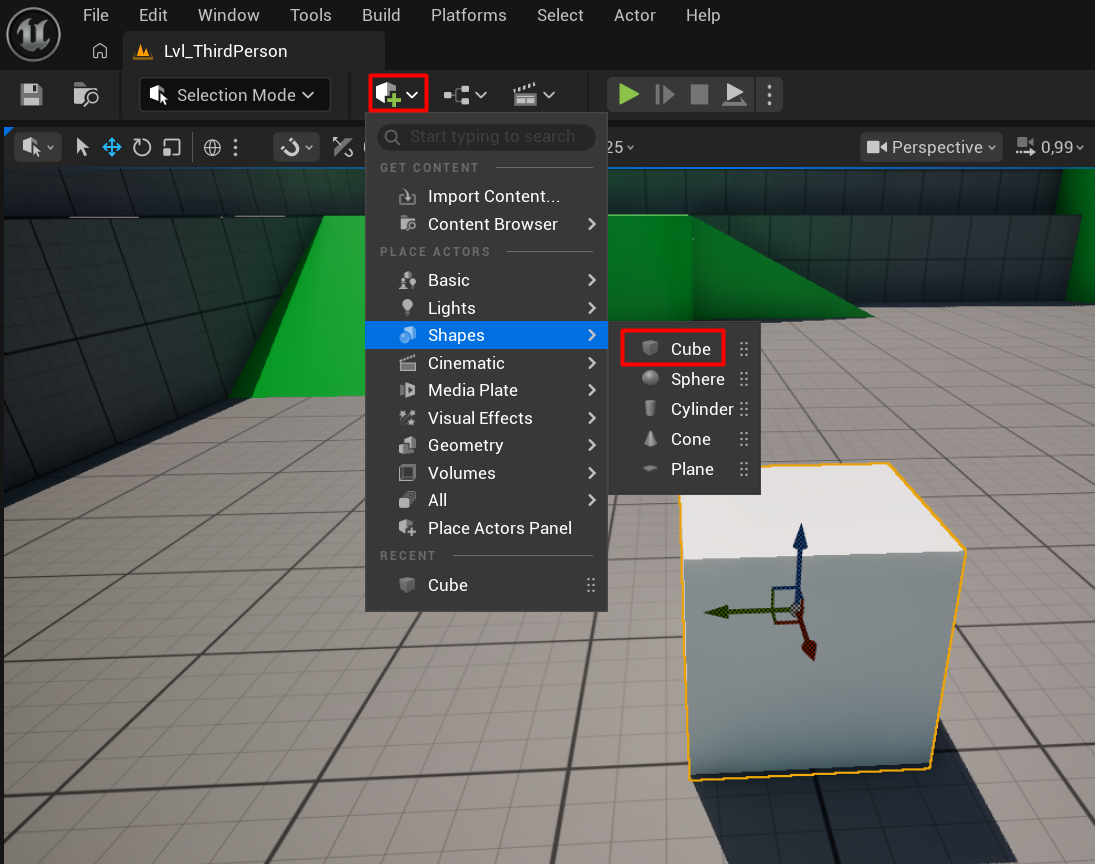
Інтерфейс складається з наступних частин:

1. Рядок меню. Використовується для доступу до команд і функцій редактора.
2. Головна панель інструментів. Містить ярлики для деяких із найпоширеніших інструментів і редакторів у Unreal Engine, а також ярлики для входу в режим відтворення та розгортання проекту на інших платформах.
3. Видове вікно рівня. Відображає вміст рівня.
4. Content Drawer. Відкриває панель вмісту, з якої можна отримати доступ до всіх ресурсів проекту.
5. Нижня панель інструментів. Містить ярлики для командної консолі, журналу виводу, діагностики та функцій похідних даних.
6. Outliner. Відображає ієрархічне дерево всього вмісту рівня.
7. Details. Відображає властивості обраного актора, наприклад його трансформацію, статичну сітку, параметри матеріалу та фізики.

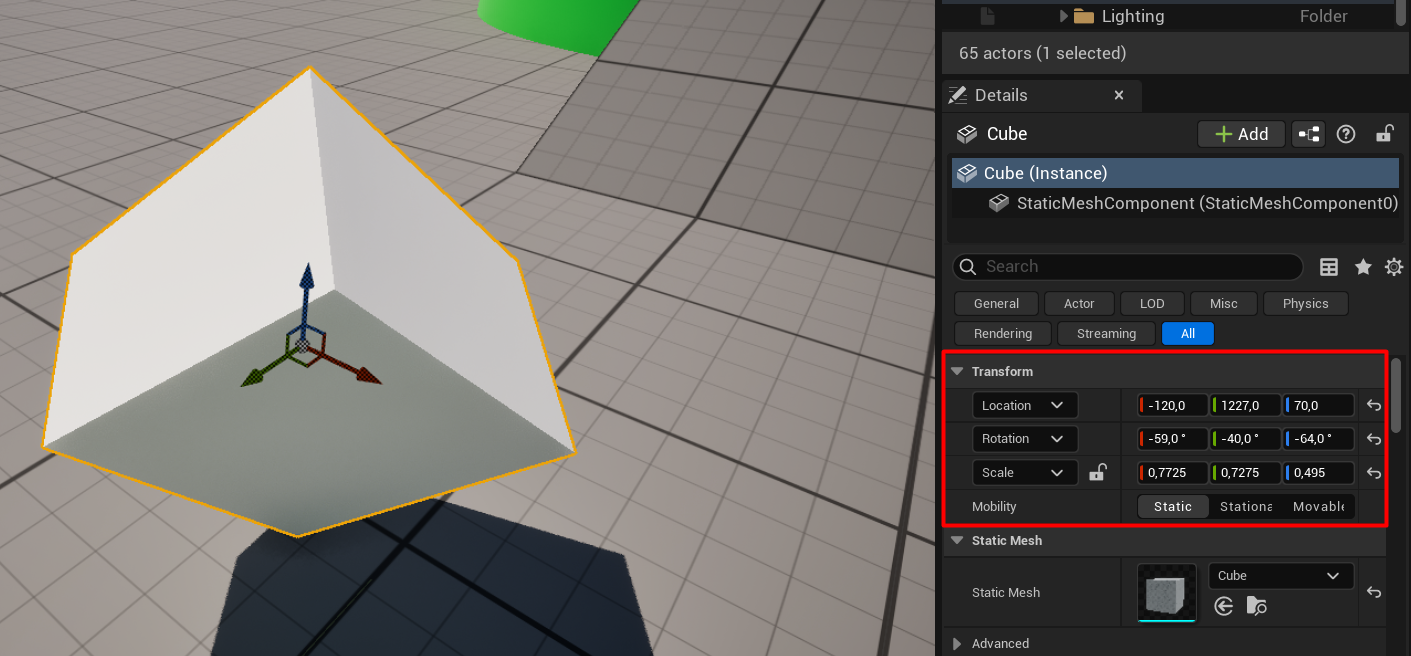
На рисунку 4 подано скріншот інтерфейсу з позначеннями його блоків.

  
Рисунок 4 – Інтерфейс Unreal Engine 5

Для додавання нових форм необхідно використати інструмент *Place Actor*. Обираємо куб та розміщуємо його на мапі (рис. 5).

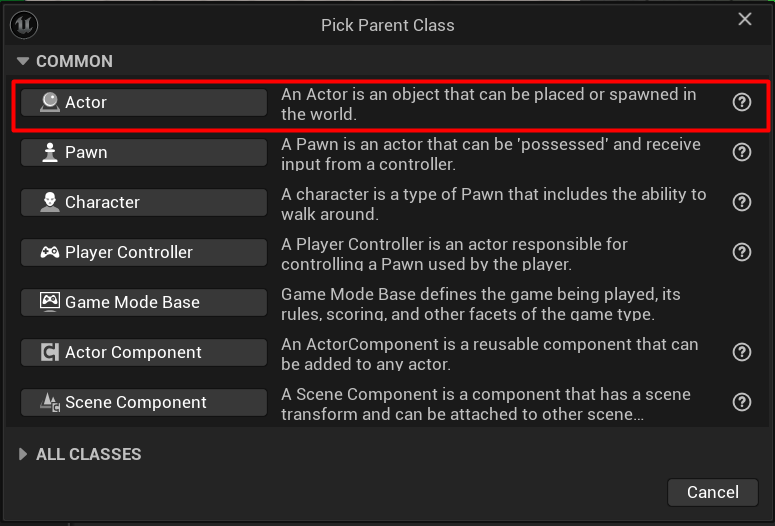
  
Рисунок 5 – Додавання нових форм

Змінюючи значення компонента Transform можна маніпулювати розміщенням актора на сцені, його масштабом та обертанням (рис. 6).

  
Рисунок 6 – Маніпулювання акторами

*Blueprint Class* – це об’єктно-орієнтований тип активу Unreal Engine, створений візуально в редакторі замість написання коду. Він визначає новий клас (тип) на основі існуючого C++ або іншого Blueprint батька та зберігається як ассет у пакеті контенту.

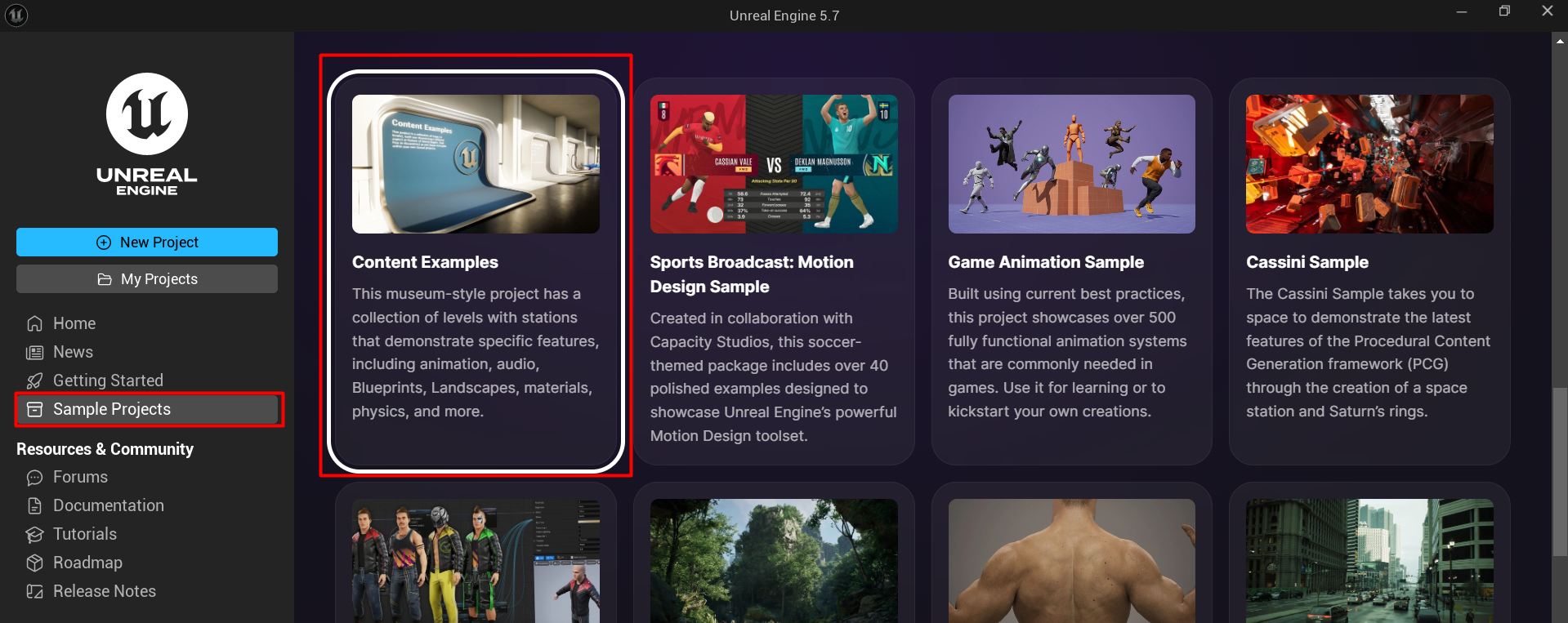
Для додавання *Blueprint Актора* необхідно відкрити *Content Drawer*, натиснути правою кнопкою миші на пусте місце та обрати *Blueprint Class*. У вікні вибору батьківського класу обираємо Actor (рис. 7).

  
Рисунок 7 – Додавання Blueprint Актора

На рисунку 7 видно основні батьківські класи для *Blueprint* *Class*. Описати їх можна наступним чином:

* Actor. Базовий ігровий об’єкт у рівні з трансформом та компонентною структурою, служить контейнером для візуальних та логічних частин та виконує події на кшталт *BeginPlay* та *Tick*.
* Pawn. Спадкоємець *Actor*, що представляє кероване фізичне тіло у світі; призначений для володіння контролером та взаємодії із системами пересування.
* Character. Спеціалізований *Pawn* з *Capsule Collider* та *Character Movement Component* для готових режимів пересування (ходьба, стрибок, біг), зручний для гравців та гуманоїдних NPC.
* Player Controller. Нефізичний клас, що обробляє введення та приймає рішення за гравця, керує Possess/Unpossess, логікою камери та HUD.
* Game Mode Base. Серверний клас правил гри, що відповідає за спаун, правила матчу та вибір класів *PlayerController*, *GameState* тощо, не реплікується на клієнтах як об’єкт сцени.
* Actor Component. Модуль поведінки, що перевикористовується, який додається до *Actor* для розширення функціональності без власного трансформу.
* Scene Component. Компонент із трансформом, що використовується як точка прив’язки всередині *Actor* для розміщення мешів, камер та інших компонентів.

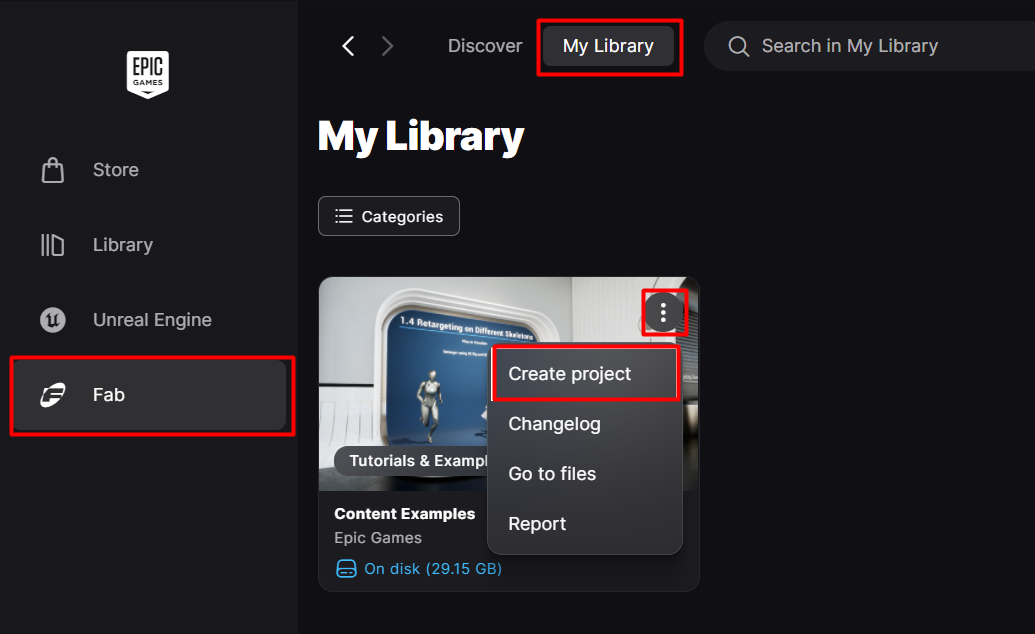
Для інсталювання проекту *Content Examples* необхідно запустити Unreal Engine, відкрити вкладку *Sample Projects* та обрати бажаний проект (рис. 8).

  
Рисунок 8 – Вибір проекту-прикладу

Дані дії відкриють сторінку проекту *Content Examples* на сайті *fab.com*. Далі необхідно натиснути «Add to My Library», щоб додати проект в бібліотеку власних фабів (рис. 9).

  
Рисунок 9 – Додавання проектів до власної бібліотеки

Після цього необхідно відкрити Epic Games та перейти до вікна фабів (*Fab*) на вкладку *My Library* та створити новий проект на основі *Content Examples* (рис. 10).

  
Рисунок 10 – Створення проекту на основі прикладу

*Content Examples* містить велику кількість карт для демонстрації можливостей Unreal Engine (рис. 11). На вибір доступні наступні категорії карт:

1. *AI*. Карти демонстрації роботи зі штучним інтелектом акторів: *AI\_Mass*, *AI\_NavMesh*, *AI\_StateTree*, *AI\_StateTreeAIController*.
2. *Animation*. Карти демонстрації роботи з анімаціями акторів: *Animation\_Basics*, *Animation\_Choosers*, *Animation\_ControlRig*, Animation\_Deformer, *Animation\_Grooms*, *Animation\_IKRig*, *Animation\_MorphTargets*, *Animation\_PhysicsControl*, *Animation\_Retargeting*, *PhysicalAnimation*, *Sequencer*.
3. *Audio*. Карти демонстрації роботи з аудіо ресурсами: *Audio*, *Audio\_Metasounds*.
4. *Blueprint*. Карти демонстрації роботи з *Blueprint Акторами*: *Blueprint\_Advanced*, *Blueprint\_Communication*, *Blueprint\_EditorUtilities*, *Blueprint\_EnhancedInput*, *Blueprint\_Input\_Examples*, *Blueprint\_Mouse\_Interactions*, *Blueprint\_Network*, *Blueprint\_Overview*, *Blueprint\_ProceduralMesh*, *Blueprint\_RenderToTarget*, *Blueprint\_Splines*.
5. *Geometry*. Карти демонстрації роботи з акторами геометрії: *FBX\_Import\_Options*, *Geometry\_ModelingTools*, *Geometry\_Nanite*, *StaticMeshes*.
6. *Level*. Карти демонстрації роботи з картами: *Level\_DataLayers*, *Level\_Landscape*, *Level\_PCG*, *Level\_PostProcessing*, *Level\_Scripting*, *Level\_Streaming*, *Level\_Volumes*, *Level\_WorldPartitionStreaming*.
7. *Lighting*. Карти демонстрації роботи з освітленням: *Lighting\_Realtime*.
8. *Material*. Карти демонстрації роботи з матеріалами: *Material\_Advanced*, *Material\_Decals*, *Material\_Instances*, *Material\_Nodes*, *Material\_ParallaxOcclusionMapping*, *Material\_PivotPainter*, *Material\_PivotPainter2*, *Material\_Properties*, *Material\_Skin\_SubsurfaceRendering*, *Material\_Substrate*.
9. *Misc*. Категорія «Різне»: *Math\_Hall*, *Paper2D*.
10. *Physics*. Карти демонстрації роботи з фізикою: *Physics*, *Physics\_Cloth*, *Physics\_Destruction*.
11. *UI*. Карти демонстрації роботи з інтерфейсом користувача: *UI\_CommonUI*, *UI\_StatePostBuffer*, *UI\_UMG*.
12. *VFX*. Карти демонстрації роботи з візуальними ефектами: *Niagara\_Advanced\_Particles*, *Niagara\_Data\_Channels*, *Niagara\_Fluids*, *Niagara\_Particles*.

Кожна демонстраційна карта мітить описові стенди з текстом (рис. 12) та демонстраційні стенди з акторами та відповідним супровідним пояснювальним текстом (рис. 13).

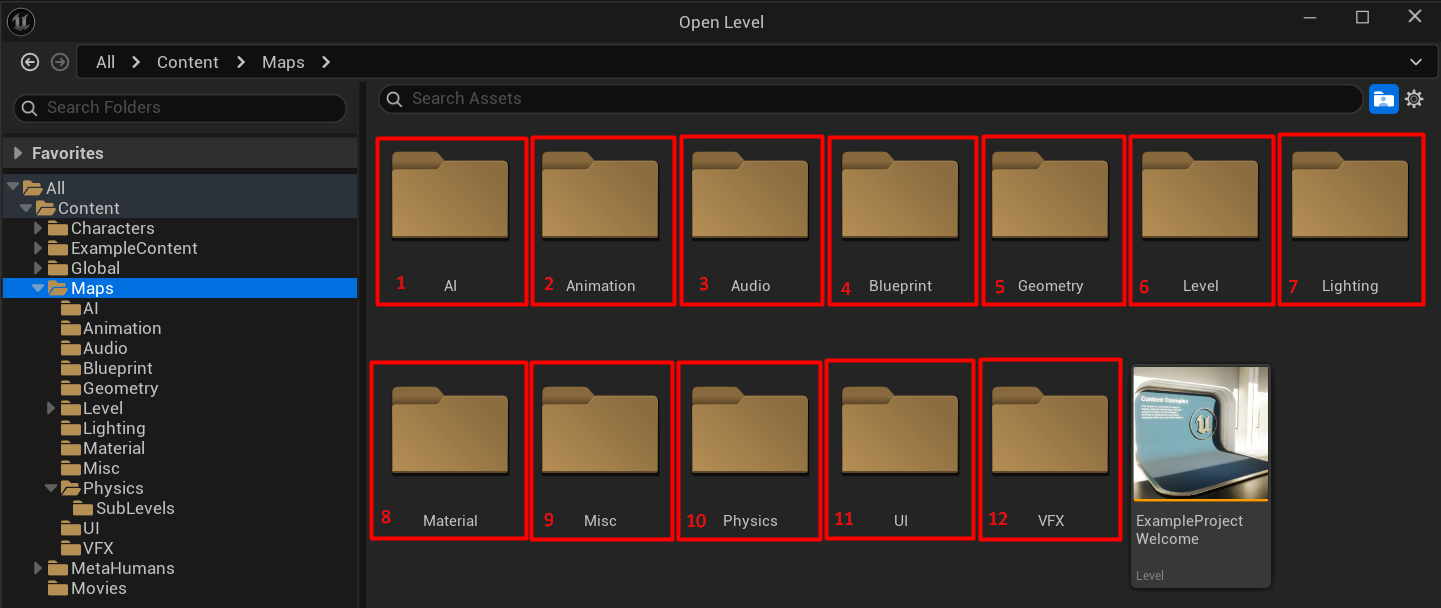
  
Рисунок 11 – Категорії карт проекту Content Examples

  
Рисунок 12 – Описові стенди карт проекту Content Examples

  
Рисунок 13 – Демонстраційні стенди карт проекту Content Examples

**Висновки**

При виконанні лабораторної роботи набуто практичні навички роботи з рушієм Unreal Engine 5. У ході роботи завантажено рушій Unreal Engine 5.7, створено перший проект; отримано навички створення форм та базового маніпулювання ними, завантаження проектів прикладів та створення похідних проектів.