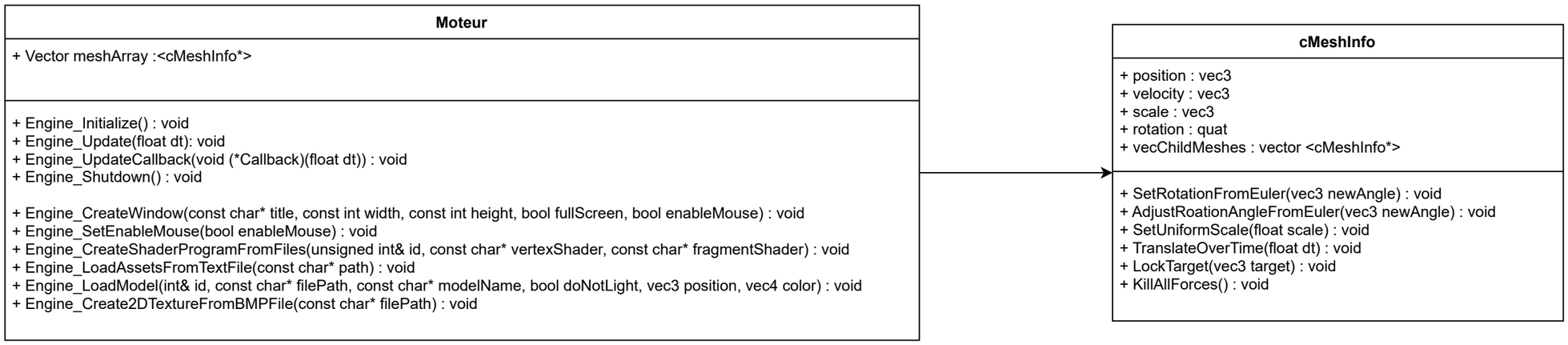
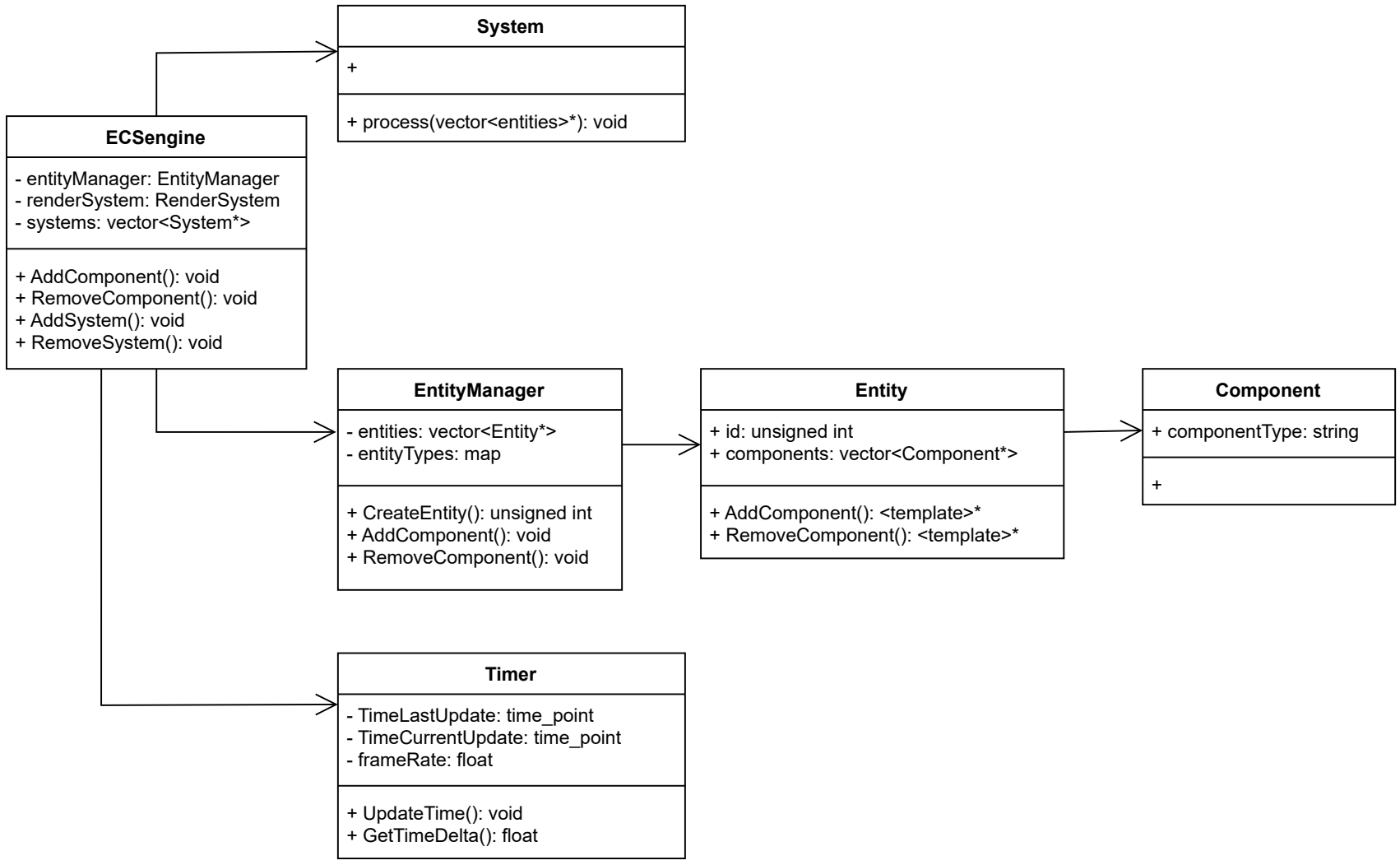
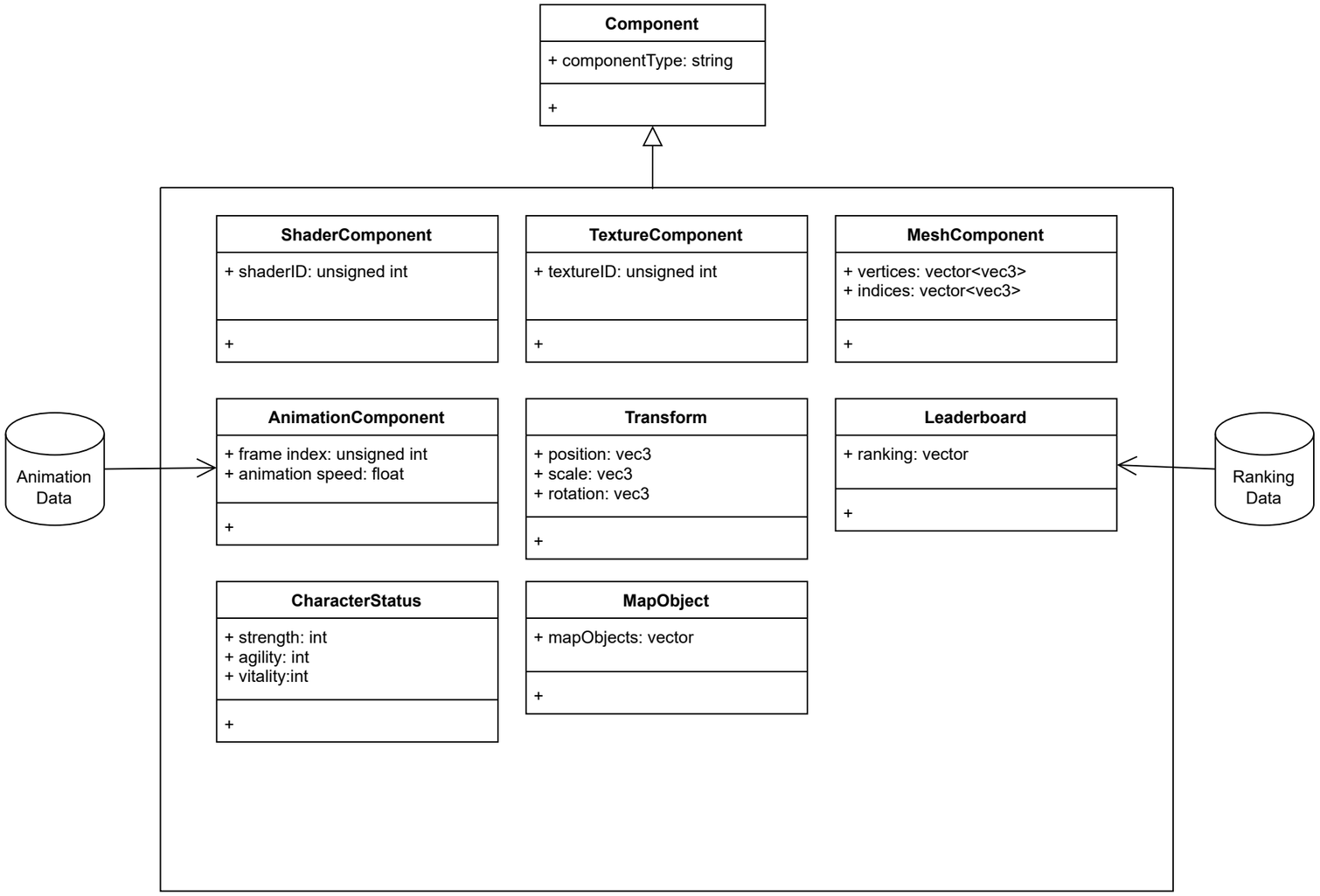

Moteur
+ Vector meshArray :<cMeshInfo*>
+ Engine_Initialize() : void
+ Engine_Update(float dt): void
+ Engine_UpdateCallback(void (*Callback)(float dt)) : void
+ Engine_Shutdown() : void
+ Engine_CreateWindow(const char* title, const int width, const int height, bool fullScreen, bool enableMouse) : void
+ Engine_SetEnableMouse(bool enableMouse) : void
+ Engine_CreateShaderProgramFromFiles(unsigned int& id, const char* vertexShader, const char* fragmentShader) : void
+ Engine_LoadAssetsFromTextFile(const char* path) : void
+ Engine_LoadModel(int& id, const char* filePath, const char* modelName, bool doNotLight, vec3 position, vec4 color) : void
+ Engine_Create2DTextureFromBMPFile(const char* filePath) : void
cMeshInfo
+ position : vec3
+ velocity : vec3
+ scale : vec3
+ rotation : quat
+ vecChildMeshes : vector <cMeshInfo*>
+ SetRotationFromEuler(vec3 newAngle) : void
+ AdjustRoationAngleFromEuler(vec3 newAngle) : void
+ SetUniformScale(float scale) : void
+ TranslateOverTime(float dt) : void
+ LockTarget(vec3 target) : void
+ KillAllForces() : void


System
+
+ process(vector<entities>*): void
ECSengine
- entityManager: EntityManager
- renderSystem: RenderSystem
- systems: vector<System*>
+ AddComponent(): void
+ RemoveComponent(): void
+ AddSystem(): void
+ RemoveSystem(): void
EntityManager
- entities: vector<Entity*>
- entityTypes: map
+ CreateEntity(): unsigned int
+ AddComponent(): void
+ RemoveComponent(): void
Entity
+ id: unsigned int
+ components: vector<Component*>
+ AddComponent(): <template>*
+ RemoveComponent(): <template>*
Component
+ componentType: string
+
Timer
- TimeLastUpdate: time_point
- TimeCurrentUpdate: time_point
- frameRate: float
+ UpdateTime(): void
+ GetTimeDelta(): float


Component
+ componentType: string
+
ShaderComponent
+ shaderID: unsigned int
+
TextureComponent
+ textureID: unsigned int
+
MeshComponent
+ vertices: vector<vec3>
+ indices: vector<vec3>
+
Animation Data
AnimationComponent
+ frame index: unsigned int
+ animation speed: float
+
Transform
+ position: vec3
+ scale: vec3
+ rotation: vec3
+
Leaderboard
+ ranking: vector
+
Ranking Data
CharacterStatus
+ strength: int
+ agility: int
+ vitality:int
+
MapObject
+ mapObjects: vector
+

System

+

+ process(vector<entities>*): void

ShaderSystem

+ shaderManager: cShaderManager
+ shaderID: unsigned int

+ GetShaderManager(): ShaderManager
+ Process(Vector<entities>, float dt): void
+ CreateShaderProgram(unsigned int& id, char* vertShader, char* fragShader): void
+ Shutdown()

RenderSystem

+ systemName: string
+ window: Window
+ camera: Camera
+ vaoManager: cVAOManager
+ animationManager: AnimationManager

+ CreateWindow(const char* title, const int width, const int height, bool fullScreen): void

+ Initialize(const char* title, const int width, const int height, bool fullScreen): void

+ LoadMesh(std::string fileName, std::string modelName, sModelDrawInfo& plyModel, unsigned int shaderID): bool

+ Process(Vector<entities>, float dt): void

+ Shutdown()

MeshSystem

+ systemName: String
+ vaoManager: cVAOManager
+ textureManager: cBasicTextureManager

+ GetVAOManager(): cVAOManager

+ Process(Vector<entities>, float dt): void

+ LoadMesh(std::string fileName, std::string modelName, sModelDrawInfo& plyModel, unsigned int shaderID): bool

+ SetTexturePath(const char* filePath): void

+ Load2DTexture(unsigned int& textureID, const char* filePath)

+ LoadCubeMapTexture(unsigned int& textureID,std::string cubeMapName, std::string posX_fileName, std::string negX_fileName,std::string posY_fileName, std::string negY_fileName,std::string posZ_fileName, std::string negZ_fileName, bool bIsSeamless, std::string& errorString): void

+ Shutdown()

PhysicsSystem

InputSystem

JsonSystem