# Table and Chairs

## 1 Introduction

One of the steps of the hiring process here in Zuru Tech, is the implementation and delivery of a project in Unreal Engine that gives us an idea of your tech skills and how you write code.

In this document, you will find all the information needed to complete and deliver the project successfully, which will consist of the generation of a simple set of geometry along with its manipulation in real-time.

## 2 Delivery

Once you receive this document, and choose a starting date with your HR contact, you will have 2 weeks of time to complete the assignment; once finished you should provide these files:

1. A ***.zip*** archive, containing the working Unreal Engine project only with the needed files/folder (you should avoid sending ***Intermediate, Binaries, Saved, …*** folders and the C***++*** solution).
2. A ***.zip*** archive, containing the same project from the previous step, compiled and ready to be executed. The platform depends on the one where you have developed the assignment.
3. If useful, add a document (**Markdown**) describing the solution you proposed.

Note that the first 2 points are **mandatory**, and without them the phase will not start correctly. Also **you should not publish the project** in any public online services, or share it with anyone outside Zuru Tech.

In addition, code from other projects already available online should be clearly motivated. Although it's possible to resort on snippets from referenced websites, please consider that most of the code must be original to allow a honest and fair evaluation.

## 3 How is it evaluated

After you have delivered the project, it will be assigned to one of our developers for the evaluation phase, in which it will be considered:

* Code Quality: how the code is organized inside the folder, comments and any coding standard adopted.
* Performance: how your solution behaves in terms of execution time and memory.
* Clever: how your solution makes use of any particular algorithm in order to solve the problem.
* Patterns/Architecture: which pattern you have used and for what purpose, and how the various components of your project are put together.
* Bonus: given by the implementation of the optional points.

## 4 Requirements

As stated before, the objective of this assignment is to develop a project which procedurally generates a **table** (by means of Unreal UProceduralMeshComponent) and places the correct number of **chairs** around it. In addition, it should allow the user to select and change the parameters that are used to perform the procedural generation.

Aside from these base steps you will also find a set of **Optional Requirements**: developing them in the proper way will grant some extra points during the evaluation phase (see “How is it evaluated” section, for more details).

You have total freedom in the actual implementation (Architecture, class used, etc.), as long as it implements what is required.

### 4.1 Terms

A ***table*** is defined as a geometry that has:

* A squared tabletop with **width**, **length,** and **height** parameters
* 4 Legs with a fixed width and length, and a variable **height** parameter.

A ***chair*** is defined as a geometry that has:

* 4 Legs with a fixed width and length, and a variable **height** parameter.
* A back with a fixed width and length, and a variable **height** parameter.
* A seat with **width**, **length,** and **height** parameters

For each side of the ***table*** a certain number of ***chairs*** (with the same parameters) should be spawned based on:

* The length of the table side
* The width of the chair seat
* The **spacing** parameter between each chair

### 4.2 Mandatory

The solution that you will provide must have the following points to be considered valid:

* The application must display a text where indication on controls and base application operation are provided.
* The user should be able to place a ***table*** (with chairs) by clicking on a point in the plane. Every time that a ***table*** and ***chairs*** should be spawned, their parameters should be randomized in order to have a different layout every time.
* For every generated ***table***, a number of ***chairs*** should be spawned on each side of it. The number of ***chairs*** can vary based on the rules that you find at the end of this section.
* If the user clicks on a ***table***, instead of the plane, it will be selected with clear visual feedback.
* If the user clicks on an already selected ***table***, it will be deselected.
* When a ***table*** is selected a UI/Widget should be displayed on screen, allowing the user to change the variable parameters of the procedural generation of the ***table*** and ***chair*** instances. The changes should be then reflected on the objects.

### 4.3 Optional

After you have implemented the mandatory requirements, you can go the extra mile and work on these optional points. Note that the more you develop, the more points you will have.

* The user should be able to select more than one ***table*** at the same time.
* When one or more ***tables*** are selected, the UI should allow changing the parameters for every selected object, meaning that the user can set the same parameters for each instance of an object***.***
* A ***table*** should be spawned only if it does not collide with any other table group in the scene. Feedback must be provided to the user if the spawn failed.

## 5 What is provided

With this document you will also receive a starting project, which contains:

* A Player Controller and Third Person Camera Pawn to move around the scene.
* A simple scene which contains a plane and the base lighting
* A C++ Module in which you must place your code.

## 6 Useful Links