TILES PILES PROTOTYPE TECHNICAL DOCUMENT

# TASK A

In order to reproduce the desired mechanics, the first thing I set up was the logical grid on the floor. As asked, it’s a 4x4 grid. Since this was a test for a fast prototype as a programmer, I decided not to use Unity Animations to move objects: instead all the movement of the ingredients is done by code, detecting the correct position for movement. Code is set up both for Android devices that for Windows, since I decided to create first a working project on Unity, and then adding the input methods for mobile phones.

Main Mechanics:

* Ingredient can be moved only on adjacent cells with at least 1 ingredient on it.
* Ingredient groups rotate when moved.
* Victory is achieved when there are no ingredients left on the table, and all the ingredient are placed in one cell between 2 bread pieces.

Level Creation:

* Level creation is done by the Grid Manager: on the inspector in Unity of the Floor object, you can choose which ingredient is spawned on each cell, to allow different levels creation.

Classes and Structs:

* GridManager: class that controls all the mechanics of the game
* GridCell: struct that contains several different elements of the single cell of the grid
* IngredientController: class that detects when the moving ingredient/s is/are close to his/their destination

Methods:

* DetectInput: checks if player input is on a movable ingredient and his destination
* CalculateMovement: calculates the correct destination if reachable, and performs all the logic behind adding/removing ingredients on the logic matrix
* MovePiece: performs the ingredients movement
* SetNewParent: sets the correct parent when ingredients are stacked on a single cell
* CheckForVictory: performs a check on the logic matrix to see if victory condition is achieved
* ReloadLevel: reloads the level at the start state

# TASK B

New and update UI:

Buttons:

* StartGame: starts the game with the current level
* ReloadLevel: Save and Reloads the actual level with the same ingredients layout
* LoadLevel + Input Field: loads the level saved at the desired index position
* SaveLevel + Input Fiel: saves the current level at the desired index
* SetNewParent: sets the correct parent when ingredients are stacked on a single cell
* NewLevel: Generates a new random level of the desired difficulty
* + Button: Increases difficulty (max 16)
* - Button: Lowers difficulty (max 4)