# Short Description

# Functionalities

## Functionalities to be implemented by Cole

## Functionalities to be implemented by Garrett

## Functionalities to be implemented by Grant

1. Automatic Database Population – Generate API Output

## Functionalities to be implemented by Nate

## Functionalities to be implemented by Zach

1. Meal Plan Generation – Food Waste Reduction Algorithm

One of the core appeals of the Overcooked application is its potential to reduce food left over after a week of cooking. We intend to accomplish this by creating meal plans with recipes that overlap. For example, if Recipe A uses half a carton of vegetable stock, and Recipe B also uses half a carton of vegetable stock, then Recipes A and B have a degree of overlap. By creating meal plans with heavily overlapped recipes, the amount of ingredients left over after each meal plan will be reduced. We believe that this emphasis on reducing food waste, as well as our algorithm for achieving this goal, make Overcooked a unique meal planning application.

For each recipe, a ‘leftover score’ can be calculated based on the amount and type of required ingredients not consumed by the recipe. Ingredients will be weighted (wasted meat, for instance, will have a higher leftover score than wasted potatoes). The leftover score (l) for a recipe (r) consisting of the set of ingredients and quantities {(i0, q0), …, (in-1, qn-1)} is defined as

Where d(ij) is the smallest discrete unit of ingredient (ij) that can be purchased, s(ij) is the minimum natural number such that s(ij) \* d(ij) > qj, andW(ij) is the weight associated with ingredient ij. In future versions of the application, weights could be determined by users to minimize the waste of certain ingredients, but for now the weights will be determined by the best judgement of the team members. When selecting single recipes, the user will be presented with the recipes that have the lowest leftover score (in addition to meeting the requirements specified by the user such as dietary restrictions).

The leftover score for two recipes can be calculated by creating a union of the sets of ingredients required and adding the quantities of each ingredient to obtain the (ingredient, quantity) set {(i0, q0)1, …, (in-1, qn-1)1}. The leftover score for this set is calculated using the equation above. Each recipe will contain an additional field with the identifiers of the 3 recipes it pairs best with. The lower the leftover score generated from the combined ingredient set of the two recipes, the better the recipes are said to pair. When a user requests a meal plan with N recipes, N//2 optimally matched pairs of recipes are selected, and N%2 single recipes with the lowest individual leftover scores will be selected. This approach avoids a slow and computationally expensive combinatorial search for optimal pairings when the user requests a meal plan. The tradeoff is the additional storage space required for the 3 best pairings for each recipe. Additionally, this algorithm has the potential to reduce food waste, but does not necessarily produce an optimal solution.

1. Automatic Database Population – Parse API Output

* Not implemented yet
* The output from querying the API needs to be transformed into a consistent format for the database
* The buddy recipes and leftover scores for newly added recipes must be updated