**Fridge Project**

Our idea was to develop an application (basic terminal code) in Ruby. The application suggests Japanese recipes that the user can cook, based upon which ingredients are in their refrigerator. Kind of like a ‘smart fridge’. The application also can provide other information such as a URL to the recipe itself, as well as how long the dish will take to make.

1. **The problem and its corresponding solution;**

The problem for the user is that they might not know any Japanese recipes, but might like to make them. So, at the beginning, we made application greet the user and ask if they would like to make some Japanese food. If they decided ‘yes’, we then asked them what food items (ingredients) were in their fridge at the time. Finally, based on the user’s input, we coded our application to provide recipe ideas as well as their URL and cook time.

1. **Development and technical solution;**

First of all, we decided upon a list of recipes (5-7) and then broke the recipes down into their ingredient components. To make this easier to visualise, we decided to do this on a large piece of paper during our brainstorming session. Ultimately, we came up with 7 recipes and 16 ingredients that make up all those recipes. The following is a list of steps in the application

1. We made classes for Fridge, Ingredient and Recipe;
2. We made 1 Fridge object, 16 Ingredient objects and 7 Recipe objects;
3. We gave the Fridge (Owner Name, Ingredients, Recipes, What you Have) attributes;
4. We gave Ingredient the (Name) attribute;
5. We gave Recipe (Name, Ingredients, URL, Minutes) attributes;
6. Then, we made a main menu to greet the user and ask if they want to eat Japanese food. If the answer is No, the program says a farewell message to the user and ends.
7. If the answer is Yes, then the application asks the user if they have the 16 ingredient items by inputting a Y or an N to answer;
8. After looping through all 16 ingredient items, the application stores the Y answer items they have into an array and compares with the array of recipe ingredients;
9. If the application finds matching recipes, it then shows the recipes in a table, with further information such as the URL and cooking time.
10. The application ends.
11. **How did you approach the development?**

Our group consisted of four members, each with varying skill levels at coding in Ruby. Initially, we decided upon an idea, then we got together as a team and started to brainstorm about recipes and structure, according to our knowledge. Everything was written down on paper to avoid confusion and help actualise our idea. When we felt we had appropriate amount of options to meet the projects’ requirements, we discussed the actual division of programming amongst us.

As we are all at different levels of ability, we thought that it was a good idea for each of us to attempt to make a workable application individually, so we could learn by doing and therefore practically understand how to achieve the goals in the project and how to talk about performing them in the presentation.

We all attempted to make versions of the application over the weekend and got together on Monday to compare and discuss the steps that we made in completing the application. Ultimately, we thought that this was a good approach because if we focused too much on delegating different tasks, we might not wholly understand how to make a workable application from beginning-to-end.