University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: ____1____

Date: ___2024-09-30_____

Group Number: ____15____

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Kenny Niu	37151198	y7r6p	kennyn172@gmail.com
Camille Pureza	72136310	p1y7c	camillepureza@gmail.com
Irene Chang	85567402	d0r9y	irecha@student.ubc.ca

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address, and then let us assign you to a TA for your project supervisor.)

In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia

CPSC 304 – M1 Project Proposal

Brief Project Description

What is the domain of the application? Describe it

In the distant future, exorbitant housing prices in Vancouver normalized groups of 20 or more UBC graduates to pool their money together to afford a home and, unfortunately, deal with cramped and chaotic living conditions. One such resident is a brilliant software engineer who believes Agile is the best thing ever created. They decided to implement a version of Jira at home, which would make everyone's lives much easier. In general, our application is the first iteration of Jira where users can manage their house inventory, household-related responsibilities, and finances.

What aspects of the domain are modeled by the database?

Living with multiple people makes it difficult to track everyone's responsibilities and items. Our database will model the essential components (e.g. food/item inventory, agreements, and finances) of living in a peaceful, fair, and organized household. Of course, an important entity to represent is the household members, which will interact with most of the other entities modeled in our database. We also include the possibility of people having pets. If someone were to have a pet, it should be their responsibility to keep things clean and care for it. We want to integrate the relationship between the food/item inventory and managing finances in our model. In a household, you want to ensure that shared items are paid for fairly between people. Not to mention, utility bills and rent should be split equally. Everyone is also responsible for maintaining the house; it's common for people to enforce chore duties on specific people. We include the management of task/chore duty to track that people are doing their fair share and that the house is always well-maintained! However, if people have disputes or complaints about certain things, we want to model a way to handle these situations.

Database Specification

What functionality will the database provide? I.e. What kinds of things will people using the database be able to do?

As mentioned previously our application has three main functions: storing food/item inventory, managing household member agreements/settlements, and tracking finances. When there is food expiring or certain items are running low (such as toilet paper or soap), our application will alert users of these items and suggest users to add them to their shopping list or create new tasks. For example, it can recommend a new task of throwing out all the expired foods. To enforce equal responsibilities among household members, any user can organize who is assigned to each task and track their progress. Finally, our application will manage the entire expenditure of the house and send reminders of when upcoming bills are due.

Application Platform

What platform will the final project be on? What is your application technology stack?
The Back-End and database portion of our project will use Oracle, SQL (as query language) and Javascript (as programming language). We will be using the React framework (using JSX language) for our GUI or Front-End components and vanilla CSS for component-styling.

ER Diagram

A note about artificial keys:

- For the People entity, although unlikely, it is still possible for two people to have the same name and birthday.
- For the Complaint entity, the Details attribute is not a good key candidate because there could be several complaints with the same details, and there is a chance that the details could be multiple sentences long.
- For the Room entity, it is possible for multiple rooms to have the same square footage and be on the same floor level.
- For the Account entity, expenses change frequently, and people could have the same budget.
- For the Expense entity, it is possible that in a single day, multiples of the same item could be purchased.