**University of British Columbia, Vancouver**
Department of Computer Science

# CPSC 304 Project Cover Page

Milestone # 2

Date: October 15, 2024

Group Number: 28

| Name | Student ID | CS Alias (Userid) | Preferred Email Address |
|---|---|---|---|
| Gagenvir Gill | 86152758 | a9g3t | gagenvirg@gmail.com |
| Preston Lai | 22541395 | q1m4a | preston1lai@gmail.com |
| Mave Hur | 55693402 | o3x3e | jmavehur@gmail.com |

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.

# Deliverable 1

Cover page (please see above)

# Deliverable 2

This kitchen management application allows users to organize their ingredients, recipes, and other information for later use. It allows users to find recipes they can make based on what they currently have at home and/or it allows users to know what ingredients of a recipe they are missing. It also helps users in finding new recipes based on those that other users have created while also being able to filter through them based on categories and the user's allergies.

# Deliverable 3

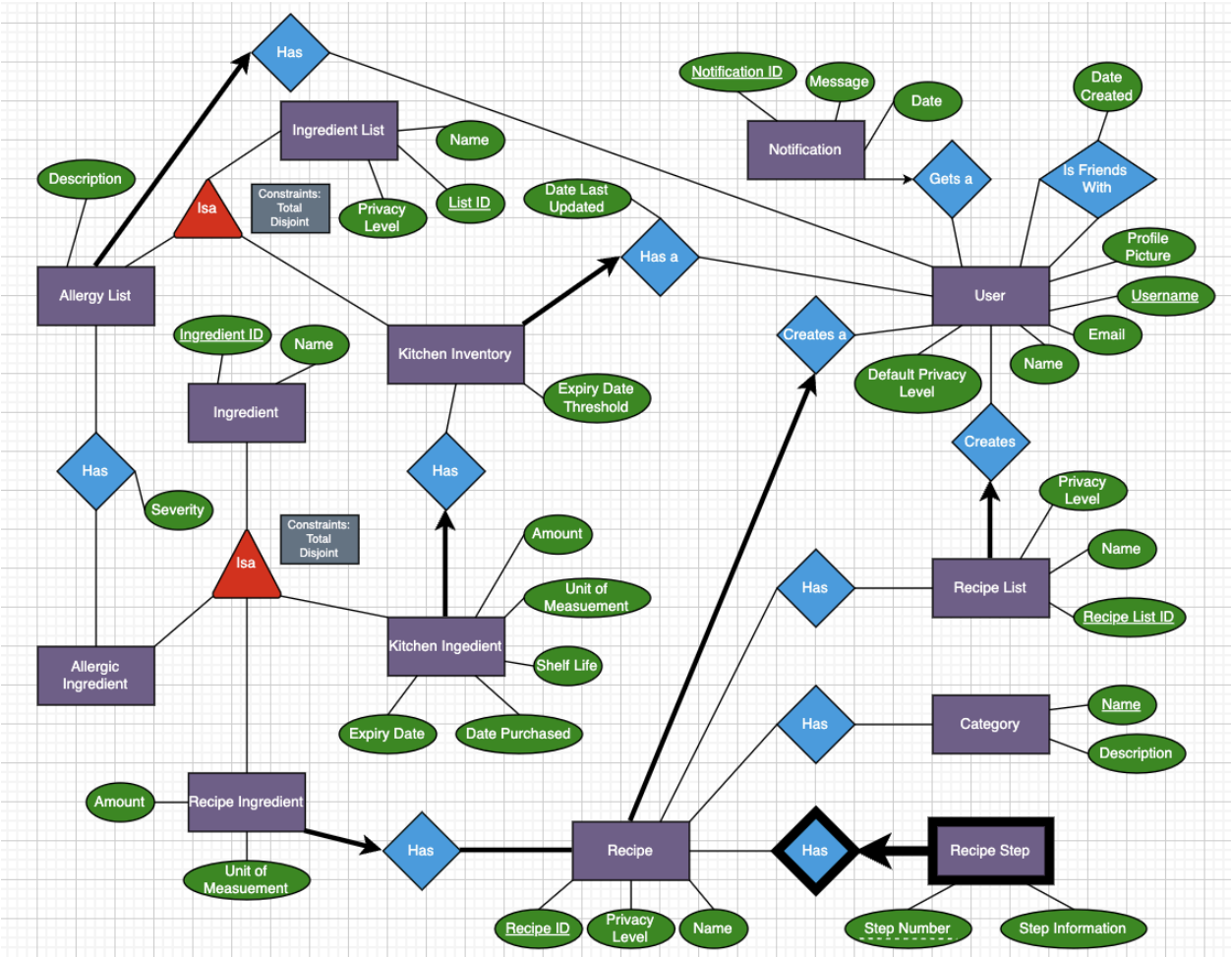
ERD Changes:

- ISA Changes
    - Based on feedback from our TA we added the necessary ISA constraints to our diagram to better display what their function is.
- New Relationship between 2 users: Is Friends With
    - This relationship between 2 users was added as a new feature to the application/database. This feature is the ability for a user to have friends (other users) on the application.
- New Attribute for User: Profile Picture
    - This attribute was added to give each user a profile picture, we added this as it works well with the friends feature we added to our application.
- New Attribute for User: Default Privacy Level
    - This attribute was added to give each user the ability to control what information of theirs is public by default.
- New Attribute for Recipe, Recipe List and Ingredient List: Privacy level
    - This attribute was added as each of these entities is directly linked to a single user (the creator). This gives each user (creator) the ability to decide whether the information is public, private, or available to friends only.
- New Attribute for Kitchen Ingredient and Recipe Ingredient: Unit of Measurement
    - This attribute was added to these entities as they have 'amount' attributes already, but their amount should also be characterized by a unit of measurement (such as grams, liters or cups).
- New Attribute for Allergy List and Allergic Ingredient relationship: Severity
    - This attribute was added as it allows the user to describe how allergic they are to an ingredient.
- New Attribute for Kitchen Inventory: Expiry Date Threshold
    - The Expiry date threshold attribute was added to allow users to tell the application, at what amount of time before an ingredient expires they want to

receive a reminder to use the ingredient. Additionally, these attributes were added based on TA feedback regarding the Ingredient List ISA relationships and how they were redundant, this makes them not redundant.

- New Attribute for Allergy List: Description
    - This attribute was added to allow users to give a bit more context on the list of allergies, as an individual user can have many lists, this allows them to provide some context on each list specifically.
- Moved Attribute from Ingredient List to User Kitchen Inventory relationship: Date Last Updated
    - This attribute was moved as it wasn't necessary to know when the AllergyList was updated (Previously it was an attribute of IngredientList which included both Kitchen Inventory and AllergyList) but it was useful to know when the Kitchen Inventory was updated by the User, so it was added to that relationship as an attribute instead.
- New Attribute for Ingredient List: Name
    - This attribute was added to allow users to differentiate better between their many ingredient lists (either allergy lists or kitchen inventories).

New ERD:



# Deliverable 4

Syntax Keynote

- Attributes in the Primary Key are <u>underlined</u>
- Attributes that are Foreign Keys are **bolded**
- The Attributes names are *italicized*
- When a relationship needs its own table (M-M) the name of the table is supposed to be the relationship name, however, some of the relationship names are repeated or aren't very descriptive. As such, in those cases, the table name will take the format of M1 + relationship name + M2 (example the relationship between recipe and category is RecipeHasCategory)

Kitchen Management Application Oracle SQL DDL Schema

- User(<u>*Username*</u>: VARCHAR2, *ProfilePicture*: BLOB, *Email*: VARCHAR2 (Unique), *Name*: VARCHAR2 (Not Null), DefaultPrivacyLevel VARCHAR2 DEFAULT 'Private' CHECK (DefaultPrivacyLevel IN ('Private', 'Public', 'Friends Only')))

- Friends(<u>***Username1***</u>: VARCHAR2, <u>***Username2***</u>: VARCHAR2, *DateAndTimeCreated*: TIMESTAMP (Not Null))

- Notification(<u>*NotificationID*</u>: NUMBER, *Message*: VARCHAR2 (Not Null), *DateAndTimeSent*: TIMESTAMP (Not Null), ***Username***: VARCHAR2 (Not Null))

- AllergyList(<u>*IngredientListID*</u>: NUMBER, *PrivacyLevel*: DefaultPrivacyLevel VARCHAR2 DEFAULT 'Private' CHECK (DefaultPrivacyLevel IN ('Private', 'Public', 'Friends Only')), *Description*: VARCHAR2, ***Username***: VARCHAR2 (Not Null), *Name*: VARCHAR2 (Default is 'Untitled Allergy List'))

- AllergicIngredient(<u>*IngredientID*</u>: NUMBER, *Name*: VARCHAR2 (Not Null))

- AllergyListHasAllergicIngredient(<u>***IngredientListID***</u>: NUMBER, <u>***IngredientID***</u>: NUMBER, *Severity*: NUMBER(Must be a value from 1-10, Default is 10))

- KitchenInventory(<u>*IngredientListID*</u>: NUMBER, *PrivacyLevel*: DefaultPrivacyLevel VARCHAR2 DEFAULT 'Private' CHECK (DefaultPrivacyLevel IN ('Private', 'Public', 'Friends Only')), *ExpiryDateThreshold*: NUMBER, ***Username***: VARCHAR2 (Not Null), *DateAndTimeLastUpdated*: TIMESTAMP, *Name*: VARCHAR2 (Default is 'Untitled Kitchen Inventory'))

- KitchenIngredient(<u>*IngredientID*</u>: NUMBER, *Name*: VARCHAR2 (Not Null), ***IngredientListID***: NUMBER(Not Null), *Amount*: NUMBER(Not Null, Amount > 0), *UnitOfMeasurement*: VARCHAR2 NOT NULL CHECK (UnitOfMeasurement IN ('piece', 'milliliters', 'liters', 'ounces', 'cups', 'grams', 'pounds', 'kilograms', 'tablespoons',

'teaspoons')), *Shelflife*: NUMBER, *DatePurchased*: TIMESTAMP, *ExpiryDate*: TIMESTAMP)

- RecipeIngredient(*IngredientID*: NUMBER, *Name*: VARCHAR2 (Not Null), ***RecipeID***: NUMBER(Not Null), *Amount*: NUMBER(Not Null, Amount > 0), *UnitOfMeasurement*: VARCHAR2 NOT NULL CHECK (UnitOfMeasurement IN ('piece', 'milliliters', 'liters', 'ounces', 'cups', 'grams', 'pounds', 'kilograms', 'tablespoons', 'teaspoons')))

- Recipe(*RecipeID*: NUMBER, *Name*: VARCHAR2 (Default is 'Untitled Recipe'), DefaultPrivacyLevel VARCHAR2 DEFAULT 'Private' CHECK (DefaultPrivacyLevel IN ('Private', 'Public', 'Friends Only')), ***Username***: VARCHAR2 (Not Null))

- RecipeStep(***RecipeID***: NUMBER, *StepNumber*: NUMBER, *StepInformation*: VARCHAR2 (Not Null))

- Category(*Name*: VARCHAR2, *Description*: VARCHAR2)

- RecipeHasCategory(***RecipeID***: NUMBER, ***Name***: VARCHAR2)

- RecipeList(*RecipeListID*: NUMBER, *Name*: VARCHAR2 (Default is 'Untitled Recipe List'), DefaultPrivacyLevel VARCHAR2 DEFAULT 'Private' CHECK (DefaultPrivacyLevel IN ('Private', 'Public', 'Friends Only')), ***Username***: VARCHAR2 (Not Null))

Things from the ERD we were not able to convey:
- Could not ensure that a Recipe has at least 1 Recipe Ingredient with DDL statements, in the future we would accomplish this with assertions.

# Deliverable 5

**User** Tables Functional Dependencies
- Username -> ProfilePicture, Email, Name, DefaultPrivacyLevel
- Email-> ProfilePicture, Username, Name, DefaultPrivacyLevel

**Friends** Tables Functional Dependencies
- Username1, Username2 -> DateAndTimeCreated
- Username1, DateAndTimeCreated -> Username2
- Username2, DateAndTimeCreated -> Username1

**Notification** Tables Functional Dependencies
- NotificationID -> Message, DateAndTimeSent, Username
- Username, DateAndTimeSent -> Message

**AllergyList** Tables Functional Dependencies
- IngredientListID -> PrivacyLevel, Description, Username, Name
- Username, Name -> IngredientListID

**AllergicIngredient** Tables Functional Dependencies
- IngredientID -> Name

**AllergyListHasAllergicIngredient** Tables Functional Dependencies
- IngredientID, IngredientListID -> Severity

**KitchenInventory** Tables Functional Dependencies
- IngredientListID -> PrivacyLevel, ExpiryDateThreshold, Username, DateAndTimeLastUpdated, Name
- Username, Name -> IngredientListID

**KitchenIngredient** Tables Functional Dependencies
- IngredientID -> Name, IngredientListID, Amount, UnitOfMeasurement, ShelfLife, DatePurchased, ExpiryDate
- DatePurchased, ShelfLife -> ExpiryDate

**RecipeIngredient** Tables Functional Dependencies
- IngredientID -> Name, RecipeID, Amount, UnitOfMeasurement
- RecipeID, Name -> IngredientID

**Recipe** Tables Functional Dependencies
- RecipeID -> Name, PrivacyLevel, Username
- Username, Name -> RecipeID

**RecipeStep** Tables Functional Dependencies
- RecipeID, StepNumber -> StepInformation

**Category** Tables Functional Dependencies
- Name -> Description

**RecipeHasCategory** Tables Functional Dependencies
- No FD's

**RecipeList** Tables Functional Dependencies
- RecipeListID -> Name, PrivacyLevel, Username
- Username, Name ->RecipeListID

# Deliverable 6: Normalization

Syntax Keynote

- Attributes in the Primary Key are underlined
- Attributes that are Foreign Keys are **bolded**
- The Attributes names are *italicized*

**BCNF on User**(*Username*, *ProfilePicture*, *Email*, *Name*, *DefaultPrivacyLevel*)

- As all the FD's have superkeys on the left hand side the relation is already in BCNF.
- No changes on PK, CK, FK.
- **Final Table:**
  User(*Username*: VARCHAR2, *ProfilePicture*: BLOB, *Email*: VARCHAR2 (Unique), *Name*: VARCHAR2 (Not Null), DefaultPrivacyLevel VARCHAR2 DEFAULT 'Private' CHECK (DefaultPrivacyLevel IN ('Private', 'Public', 'Friends Only')))

**BCNF on Friends**(***Username1***, ***Username2***, *DateAndTimeCreated*)

- As all the FD's have superkeys on the left hand side the relation is already in BCNF.
- No changes on PK, CK, FK.
- **Final Table:**
  Friends(***Username1***: VARCHAR2, ***Username2***: VARCHAR2, *DateAndTimeCreated*: TIMESTAMP (Not Null))

**BCNF on Notification**(*NotificationID*, *Message*, *DateAndTimeSent*, ***Username***)

- Notification is NOT in BCNF, for the FD: ***Username***, *DateAndTimeSent -> Message*. This is because the combination of ***Username*** and *DateAndTimeSent* is NOT a superkey for the Notification Relation
- The Closures:
  {*NotificationID*}+ = {*NotificationID*, *Message*, *DateAndTimeSent*, ***Username***}
  {***Username***, *DateAndTimeSent}+ = {**Username**, DateAndTimeSent, Message*}
- Decompose on ***Username***, *DateAndTimeSent -> Message*
  NotificationMessage(***Username***, *DateAndTimeSent*, *Message*)
  Notification(*NotificationID*, ***DateAndTimeSent***, ***Username***)
  NotificationMessage and Notification are now in BCNF
- **Final Tables:**
  NotificationMessage(***Username***: VARCHAR2, *DateAndTimeSent*: TIMESTAMP (Not Null), *Message*: VARCHAR2 (Not Null))
  Notification(*NotificationID*: NUMBER, ***DateAndTimeSent***: TIMESTAMP (Not Null), ***Username***: VARCHAR2 (Not Null))

**BCNF on AllergyList**(*IngredientListID*, *PrivacyLevel*, *Description*, ***Username***, *Name*)

- As all the FD's have superkeys on the left hand side the relation is already in BCNF.
- No changes on PK, CK, FK.

- **Final Table:**
  AllergyList(*IngredientListID*: NUMBER, *PrivacyLevel*: DefaultPrivacyLevel VARCHAR2 DEFAULT 'Private' CHECK (DefaultPrivacyLevel IN ('Private', 'Public', 'Friends Only')), *Description*: VARCHAR2, ***Username***: VARCHAR2 (Not Null), *Name*: VARCHAR2 (Default is 'Untitled Allergy List'))

**BCNF on AllergicIngredient**(*IngredientID*, *Name*)
- As all the FD's have superkeys on the left hand side the relation is already in BCNF.
- No changes on PK, CK, FK.
- **Final Table:**
  AllergicIngredient(*IngredientID*: NUMBER, *Name*: VARCHAR2 (Not Null))

**BCNF on AllergyListHasAllergicIngredient**(***IngredientListID***, ***IngredientID***, *Severity*)
- As all the FD's have superkeys on the left hand side the relation is already in BCNF.
- No changes on PK, CK, FK.
- **Final Table:**
  AllergyListHasAllergicIngredient(***IngredientListID***: NUMBER, ***IngredientID***: NUMBER, *Severity*: NUMBER(Must be a value from 1-10, Default is 10))

**BCNF on KitchenInventory**(*IngredientListID*, *PrivacyLevel*, *ExpiryDateThreshold*, ***Username***, *DateAndTimeLastUpdated*, *Name*)
- As all the FD's have superkeys on the left hand side the relation is already in BCNF.
- No changes on PK, CK, FK.
- **Final Table:**
- KitchenInventory(*IngredientListID*: NUMBER, *PrivacyLevel*: DefaultPrivacyLevel VARCHAR2 DEFAULT 'Private' CHECK (DefaultPrivacyLevel IN ('Private', 'Public', 'Friends Only')), *ExpiryDateThreshold*: NUMBER, ***Username***: VARCHAR2 (Not Null), *DateAndTimeLastUpdated*: TIMESTAMP, *Name*: VARCHAR2 (Default is 'Untitled Kitchen Inventory'))

**BCNF on KitchenIngredient**(*IngredientID*, *Name*, ***IngredientListID***, *Amount*, *UnitOfMeasurement*, *ShelfLife*, *DatePurchased*, *ExpiryDate*)
- KitchenIngredient is NOT in BCNF, for the FD: *DatePurchased*, *ShelfLife* -> *ExpiryDate.*
  This is because the combination of *DatePurchased* and *ShelfLife* is NOT a superkey for the KitchenIngredient Relation.
- The Closures:
  {*IngredientID*}+ = {*IngredientID, Name,* ***IngredientListID****, Amount, UnitOfMeasurement, ShelfLife, DatePurchased, ExpiryDate*}
  {*DatePurchased, ShelfLife*}+ = {*DatePurchased, ShelfLife, ExpiryDate*}.
- Decompose on *DatePurchased*, *ShelfLife -> ExpiryDate*.
  KitchenIngredientPerishableDate(*DatePurchased*, *ShelfLife*, *ExpiryDate*)

KitchenIngredient(*IngredientID*, *DatePurchased*, *ShelfLife*, *Name*, ***IngredientListID***, *Amount*, *UnitOfMeasurement*).
KitchenIngredientPerishableDate and KitchenIngredient are now in BCNF.

- **Final Tables:**
  KitchenIngredientPerishableDate(*DatePurchased*: TIMESTAMP, *ShelfLife*: NUMBER, *ExpiryDate*: TIMESTAMP)
  KitchenIngredient (***DatePurchased***: TIMESTAMP, ***ShelfLife***: NUMBER, *IngredientID*: NUMBER, *Name*: VARCHAR2, ***IngredientListID***: NUMBER(Not Null), *Amount*: NUMBER(Not Null, Amount > 0), *UnitOfMeasurement*: VARCHAR2 NOT NULL CHECK (UnitOfMeasurement IN ('piece', 'milliliters', 'liters', 'ounces', 'cups', 'grams', 'pounds', 'kilograms', 'tablespoons', 'teaspoons')))

**BCNF for RecipeIngredient**(*IngredientID*, *Name*, ***RecipeID***, *Amount*, *UnitOfMeasurement*)

- As all the FD's have superkeys on the left hand side the relation is already in BCNF.
- No changes on PK, CK, FK.
- **Final Table:**
- RecipeIngredient(*IngredientID*: NUMBER, *Name*: VARCHAR2 (Not Null), ***RecipeID***: NUMBER(Not Null), *Amount*: NUMBER(Not Null, Amount > 0), *UnitOfMeasurement*: VARCHAR2 NOT NULL CHECK (UnitOfMeasurement IN ('piece', 'milliliters', 'liters', 'ounces', 'cups', 'grams', 'pounds', 'kilograms', 'tablespoons', 'teaspoons')))

**BCNF for Recipe**(*RecipeID*, *Name*, *PrivacyLevel*, ***Username***)

- As all the FD's have superkeys on the left hand side the relation is already in BCNF.
- No changes on PK, CK, FK.
- **Final Table:**
  Recipe(*RecipeID*: NUMBER, *Name*: VARCHAR2 (Default is 'Untitled Recipe'), DefaultPrivacyLevel VARCHAR2 DEFAULT 'Private' CHECK (DefaultPrivacyLevel IN ('Private', 'Public', 'Friends Only')), ***Username***: VARCHAR2 (Not Null))

**BCNF for RecipeStep**(***RecipeID***, *StepNumber*, *StepInformation*)

- As all the FD's have superkeys on the left hand side the relation is already in BCNF.
- No changes on PK, CK, FK.
- **Final Table:**
  RecipeStep(***RecipeID***: NUMBER, *StepNumber*: NUMBER, *StepInformation*: VARCHAR2 (Not Null))

**BCNF for Category**(*Name*, *Description*)

- As the relation is a Two-attribute relation it is already in BCNF.
- No changes on PK, CK, FK.
- **Final Table:**
  Category(*Name*: VARCHAR2, *Description*: VARCHAR2)

**BCNF for RecipeHasCategory(*RecipeID*, *Name*)**

- As the relation has no FD's, the relation is already in BCNF.
- No changes on PK, CK, FK.
- **Final Table:**
  RecipeHasCategory(***RecipeID***: NUMBER, ***Name***: VARCHAR2)

**BCNF for RecipeList(*RecipeListID*, *Name*, *PrivacyLevel*, *Username*)**

- As all the FD's have superkeys on the left hand side the relation is already in BCNF.
- No changes on PK, CK, FK.
- **Final Table:**
  RecipeList(*RecipeListID*: NUMBER, *Name*: VARCHAR2 (Default is 'Untitled Recipe List'), DefaultPrivacyLevel VARCHAR2 DEFAULT 'Private' CHECK (DefaultPrivacyLevel IN ('Private', 'Public', 'Friends Only')), ***Username***: VARCHAR2 (Not Null))

# Deliverable 7

```
CREATE TABLE Username (
        Username VARCHAR2(50) PRIMARY KEY,
        ProfilePicture BLOB,
        Email VARCHAR2(100) NOT NULL UNIQUE,
        Name VARCHAR2(50) NOT NULL,
        DefaultPrivacyLevel VARCHAR2(20) DEFAULT 'Private' CHECK
        (DefaultPrivacyLevel IN ('Private', 'Public', 'Friends Only')),

);

CREATE TABLE Friends (
        Username1 VARCHAR2(50),
        Username2 VARCHAR2(50),
        DateAndTimeCreated TIMESTAMP NOT NULL,
        PRIMARY KEY (Username1, Username2),
        FOREIGN KEY (Username1) REFERENCES User(Username),
        FOREIGN KEY (Username2) REFERENCES User(Username)
);

CREATE TABLE NotificationMessage (
        Username VARCHAR2(50),
        DateAndTimeSent TIMESTAMP NOT NULL,
        Message VARCHAR2(250) NOT NULL,
        PRIMARY KEY (Username, DateAndTimeSent),
        FOREIGN KEY (Username) REFERENCES User(Username)
);
```

```
CREATE TABLE Notification (
        NotificationID NUMBER PRIMARY KEY,
        DateAndTimeSent TIMESTAMP NOT NULL,
        Username VARCHAR2(50) NOT NULL,
        FOREIGN KEY (Username) REFERENCES NotificationMessage(Username),
        FOREIGN KEY (DateAndTimeSent) REFERENCES
        NotificationMessage(DateAndTimeSent)
);

CREATE TABLE AllergyList (
        IngredientListID NUMBER PRIMARY KEY,
        PrivacyLevel VARCHAR2(20) DEFAULT 'Private' CHECK (PrivacyLevel IN
        ('Private', 'Public', 'Friends Only')),
        Description VARCHAR2(250),
        Username VARCHAR2(50) NOT NULL,
        Name VARCHAR2(50) DEFAULT “Untitled Allergy List”,
        FOREIGN KEY (Username) REFERENCES User(Username)
);

CREATE TABLE AllergicIngredient (
        IngredientID NUMBER PRIMARY KEY,
        Name VARCHAR2(50) NOT NULL,
);

CREATE TABLE AllergyListHasAllergicIngredient (
        IngredientListID NUMBER,
        IngredientID NUMBER,
        Severity NUMBER DEFAULT 10 CHECK (Severity >= 1 AND Severity <= 10),
        PRIMARY KEY (IngredientListID, IngredientID)
        FOREIGN KEY (IngredientListID) REFERENCES AllergyList(IngredientListID)
        FOREIGN KEY (IngredientID) REFERENCES AllergyIngredient(IngredientID)
);

CREATE TABLE KitchenInventory (
        IngredientListID NUMBER PRIMARY KEY,
        PrivacyLevel VARCHAR2(20) DEFAULT 'Private' CHECK (PrivacyLevel IN
        ('Private', 'Public', 'Friends Only')),
        ExpiryDateThreshold NUMBER,
        Username VARCHAR2(50) NOT NULL,
        DateAndTimeLastUpdated TIMESTAMP,
        Name VARCHAR2(50) NOT NULL DEFAULT 'Untitled Kitchen Inventory',
        FOREIGN KEY (Username) REFERENCES User(Username)
);
```

```
CREATE TABLE KitchenIngredientPerishableDate (
        DatePurchased TIMESTAMP,
        ShelfLife NUMBER,
        ExpiryDate TIMESTAMP,
        PRIMARY KEY (DatePurchased, ShelfLife)
);

CREATE TABLE KitchenIngredient (
        DatePurchased TIMESTAMP,
        ShelfLife NUMBER,
        IngredientID NUMBER PRIMARY KEY,
        Name VARCHAR2(50) NOT NULL,
        IngredientListID NUMBER NOT NULL,
        Amount NUMBER NOT NULL CHECK (Amount > 0),
        UnitOfMeasurement VARCHAR2(20) NOT NULL CHECK (UnitOfMeasurement IN
        ('piece', 'milliliters', 'liters', 'ounces', 'cups', 'grams', 'pounds', 'kilograms', 'tablespoons',
        'teaspoons')),
        FOREIGN KEY (IngredientListID) REFERENCES
        KitchenInventory(IngredientListID),
        FOREIGN KEY (DatePurchased, ShelfLife) REFERENCES
        KitchenIngredientPerishableDate(DatePurchased, ShelfLife)
);

CREATE TABLE RecipeIngredient (
        IngredientID NUMBER PRIMARY KEY,
        Name VARCHAR2(50) NOT NULL,
        RecipeID NUMBER NOT NULL,
        Amount NUMBER NOT NULL CHECK (Amount > 0),
        UnitOfMeasurement VARCHAR2(20) NOT NULL CHECK (UnitOfMeasurement IN
        ('piece', 'milliliters', 'liters', 'ounces', 'cups', 'grams', 'pounds', 'kilograms', 'tablespoons',
        'teaspoons')),
        FOREIGN KEY (RecipeID) REFERENCES Recipe(RecipeID)
);

CREATE TABLE Recipe (
        RecipeID NUMBER PRIMARY KEY,
        Name VARCHAR2(50) DEFAULT “Untitled Recipe”,
        PrivacyLevel VARCHAR2(20) DEFAULT 'Private' CHECK (PrivacyLevel IN
        ('Private', 'Public', 'Friends Only')),
        Username VARCHAR2(50) NOT NULL,
        FOREIGN KEY (Username) REFERENCES User(Username)
);
```

```
CREATE TABLE RecipeStep (
        RecipeID NUMBER,
        StepNumber NUMBER,
        StepInformation VARCHAR2(250) NOT NULL,
        PRIMARY KEY (RecipeID, StepNumber),
        FOREIGN KEY (RecipeID) REFERENCES Recipe(RecipeID)
);

CREATE TABLE Category (
        Name VARCHAR2(50) PRIMARY KEY,
        Description VARCHAR2(100)
);

CREATE TABLE RecipeHasCategory (
        RecipeID NUMBER,
        Name VARCHAR2(50),
        PRIMARY KEY (RecipeID, Name),
        FOREIGN KEY (RecipeID) REFERENCES Recipe(RecipeID),
        FOREIGN KEY (Name) REFERENCES Category(Name)
);

CREATE TABLE RecipeList (
        RecipeListID NUMBER PRIMARY KEY,
        Name VARCHAR2(50) DEFAULT “Untitled Recipe List”,
        PrivacyLevel VARCHAR2(20) DEFAULT 'Private' CHECK (PrivacyLevel IN
        ('Private', 'Public', 'Friends Only')),
        Username: VARCHAR2(50) NOT NULL,
        FOREIGN KEY (Username) REFERENCES User(Username)
);
```

# Deliverable 8

**User Table**

```
INSERT INTO User (Username, ProfilePicture, Email, Name, DefaultPrivacyLevel)
VALUES
('Alice', EMPTY_BLOB(), 'alice@gmail.com', 'Alice Person', 'Private'),
('Charlie', HEXTORAW('54657374'), 'charlie@gmail.com', 'Charlie Person', 'Public'),
('Bob', EMPTY_BLOB(), 'bob@gmail.com', 'Bob Person', 'Friends Only'),
('Kevin', HEXTORAW('496D616765'), 'kevin@gmail.com', 'Kevin Person', 'Private'),
('Jason', HEXTORAW('48656C6C6F'), 'image5'), 'jason@gmail.com', 'Jason Person', 'Friends
Only');
```

**Friends Table**

INSERT INTO Friends (Username1, Username2, DateAndTimeCreated) VALUES
('Alice', 'Charlie', TIMESTAMP '2024-01-15 10:00:00'),
('Charlie', 'Bob', TIMESTAMP '2022-01-16 11:30:00'),
('Bob', 'Kevin', TIMESTAMP '2023-01-17 12:15:00'),
('Kevin', 'Jason', TIMESTAMP '2020-01-18 13:45:00'),
('Jason', 'Charlie', TIMESTAMP '2021-01-19 14:00:00');

**NotificationMessage Table**

INSERT INTO NotificationMessage (Username, DateAndTimeSent, Message) VALUES
('Alice', TIMESTAMP '2024-05-19 10:00:03', 'Your potatoes expire in 7 days'),
('Alice', TIMESTAMP '2024-05-19 10:00:04', 'Your lettuce expire in 7 days'),
('Alice', TIMESTAMP '2024-05-19 10:00:05', 'Your rice expire in 7 days'),
('Alice', TIMESTAMP '2024-05-19 10:00:06', 'Your apples expire in 7 days'),
('Charlie', TIMESTAMP '2024-9-19 01:00:03', 'Your chicken expires in 3 days');

**Notification Table**

INSERT INTO Notification (NotificationID, DateAndTimeSent, Username) VALUES
(1, TIMESTAMP '2024-05-19 10:00:03', 'Alice'),
(2, TIMESTAMP '2024-05-19 10:00:04', 'Alice'),
(3, TIMESTAMP '2024-05-19 10:00:05', 'Alice'),
(4, TIMESTAMP '2024-05-19 10:00:06', 'Alice'),
(5, TIMESTAMP '2024-9-19 01:00:03', 'Charlie');

**AlleryList Table**

INSERT INTO AllergyList (IngredientListID, PrivacyLevel, Description, Username, Name) VALUES
(1, 'Private', 'These are the allergies that Alice has', 'Alice', 'Alice's Allergies'),
(2, 'Public', 'These are the allergies that Charlie has', 'Alice', 'Charlie's Allergies'),
(3, 'Public, 'These are the allergies that Bob has', 'Bob', 'Bob's Allergies'),
(4, 'Public', 'These are the allergies that Kevin has', 'Bob', 'Kevin's Allergies'),
(5, 'Friends Only', 'These are the allergies that Jason has', 'Alice', 'Jason's Allergies');

**AllergicIngredient Table**

INSERT INTO AllergicIngredient (IngredientID, Name) VALUES
(1, 'Peanuts'),
(2, 'Shellfish'),
(3, 'Dairy'),
(4, 'Gluten'),
(5, 'Soy');

**AllergyListHasAllergicIngredient Table**

INSERT INTO AllergyListHasAllergicIngredient (IngredientListID, IngredientID, Severity) VALUES
(1, 1, 5),
(1, 2, 3),
(2, 1, 7),
(2, 3, 2),
(3, 4, 8);
(3, 4, 1);
(3, 4, 10);

**KitchenInventory Table**

INSERT INTO KitchenInventory (IngredientListID, PrivacyLevel, ExpiryDateThreshold, Username, DateAndTimeLastUpdated, Name) VALUES
(6, 'Private', 5, 'Charlie', TIMESTAMP '2024-09-09 11:02:03', 'Charlie's First House'),
(7, 'Private', 2, 'Charlie', TIMESTAMP '2024-09-09 01:05:03', 'Charlie's Second House'),
(8, 'Private', 3, 'Charlie', TIMESTAMP '2024-09-09 21:00:03', 'Untitled Kitchen Inventory'),
(9, 'Public', 7, 'Jason', TIMESTAMP '2024-09-20 08:00:33', 'Jason's Kitchen'),
(10,'Friends Only', 10, 'Kevin', TIMESTAMP '2024-09-12 02:01:03', 'Kevin's Kitchen');

**KitchenIngredientPerishableDate Table**

INSERT INTO KitchenIngredientPerishableDate (Date Purchased, ShelfLife, ExpiryDate) VALUES
(TIMESTAMP '2024-01-01 10:00:00', 3, TIMESTAMP '2024-01-04 10:00:00'),
TIMESTAMP '2024-02-10 11:30:00', 6, TIMESTAMP '2024-02-16 11:30:00'(),
(TIMESTAMP '2024-03-15 12:45:00', 9, TIMESTAMP '2024-03-24 12:45:00'),
(TIMESTAMP '2024-04-18 09:00:00', 12, TIMESTAMP '2024-04-30 09:00:00'),
(TIMESTAMP '2024-05-05 08:30:00', 15, TIMESTAMP '2024-05-20 08:30:00');

**KitchenIngredient Table**

INSERT INTO KitchenIngredient (DatePurchased, ShelfLife, IngredientID, Name, IngredientListID, Amount, UnitOfMeasurement) VALUES
(TIMESTAMP '2024-01-01 10:00:00', 3, 1, 'Tomato', 1, 5, 'piece'),
(TIMESTAMP '2024-02-10 11:30:00', 6, 2, 'Olive Oil', 1, 500, 'milliliters'),
(TIMESTAMP '2024-03-15 12:45:00', 9, 3, 'Flour', 2, 1000, 'grams'),
(TIMESTAMP '2024-04-18 09:00:00', 12, 4, 'Chicken Breast', 3, 2, 'pounds'),
(TIMESTAMP '2024-05-05 08:30:00', 15, 5, 'Milk', 4, 1, 'liters');

**RecipeIngredient Table**

INSERT INTO RecipeIngredient (IngredientID, Name, RecipeID, Amount, UnitOfMeasurement) VALUES
(1, 'Apple', 1, 2, 'pounds'),
(2, 'Banana', 2, 10, 'grams'),

(3, 'Salt', 3, 5, 'teaspoons'),
(4, 'Chicken Breast', 4, 10, 'ounces'),
(5, 'Orange Juice', 5, 1, 'cups'),
(6, 'Flour', 1, 300, 'milliliters'),
(7, 'Sugar', 1, 200, 'grams'),
(8, 'Milk', 2, 250, 'milliliters'),
(9, 'Butter', 3, 100, 'grams'),
(10, 'Eggs', 2, 3, 'piece');

**Recipe Table**
INSERT INTO Recipe (RecipeID, Name, PrivacyLevel, Username) VALUES
(1, 'Apple Pie', 'Public', 'Charlie'),
(6, 'Apple Pie', 'Public', 'Charlie'),
(2, 'Banana Milkshake', 'Private', 'Charlie'),
(3, 'Salted Butter', 'Public', 'Jason'),
(4, 'Cooked Chicken', 'Public', 'Alice'),
(5, 'Orange Juice Concoction', 'Friends Only', 'Alice');

**RecipeStep Table**
INSERT INTO RecipeStep (RecipeID, StepNumber, StepInformation) VALUES
(1, 1, 'Preheat oven to 180 degrees.'),
(1, 2, 'Mix flour and sugar into a bowl.'),
(2, 1, ' Heat the milk in a saucepan until warm.'),
(2, 2, 'Whisk eggs into the milk and sugar mix.'),
(3, 1, 'Chop veggies and add in olive oil for 5 minutes.');
(4, 1, 'Boil a pot of water.')
(5, 1, 'Take the Orange Juice out of the fridge.')

**Category Table**
INSERT INTO Category (Name, Description) VALUES
('Dessert', 'sugar food is served after dinner'),
('Appetizer', 'food before the main course'),
('Main Course', 'the main dish'),
('Beverage', 'what you drink during a meal'),
('Salad', 'all veggies');

**RecipeHasCategory Table**
INSERT INTO RecipeHasCategory (RecipeID, Name) VALUES
(1, 'Dessert'),
(2, 'Main Course'),
(3, 'Appetizer'),
(4, 'Beverage'),
(5, 'Salad');

**RecipeList Table**

INSERT INTO RecipeList (RecipeListID, Name, PrivacyLevel, Username) VALUES
(1, 'Healthy Food', 'Public', 'Alice'),
(2, 'Fast Food', 'Friends Only', 'Alice'),
(3, 'Desserts', 'Private', 'Jason'),
(4, 'Vegan Dishes', 'Public', 'Jason'),
(5, 'Protein Food', 'Private', 'Jason');