Developing a CRC Model

CSC207 Winter 2015

Computer Science
UNIVERSITY OF TORONTO

Example

Consider this description of a software system:

> *You are developing a software system to facilitate restaurant reviews. Each restaurant corresponds to a certain price range, neighbourhood, and cuisines it serves. Restaurants that serve alcohol must have a license, which they need to renew every year. The system should also report how long, on average, customers wait for take out in restaurants that offer take-out service. When reviewers leave a review for a restaurant, they must specify a recommendation (Thumbs Up or Thumbs Down) and can also leave a comment. An owner of a restaurant can respond to a review with a comment. All users of the system log in with their username. Users can choose to be contacted by email ...*

To begin

1. In the problem description, underline nouns that may be classes or information that a class may be responsible for storing.
2. Next, from the nouns identified, write down the ones that are potential classes.
3. In the problem description above, circle verbs that may be things that a class is responsible for doing.

Identify nouns

Let's begin by underlining nouns.

> *Each restaurant corresponds to a certain price range, neighbourhood, and cuisines it serves. Restaurants that serve alcohol must have a license, which they need to renew every year. The system should also report how long, on average, customers wait for take out in restaurants that offer take-out service. When reviewers leave a review for a restaurant, they must specify a recommendation (Thumbs Up or Thumbs Down) and can also leave a comment. An owner of a restaurant can respond to a review with a comment. All users of the system log in with their username. Users can choose to be contacted by email and ...*

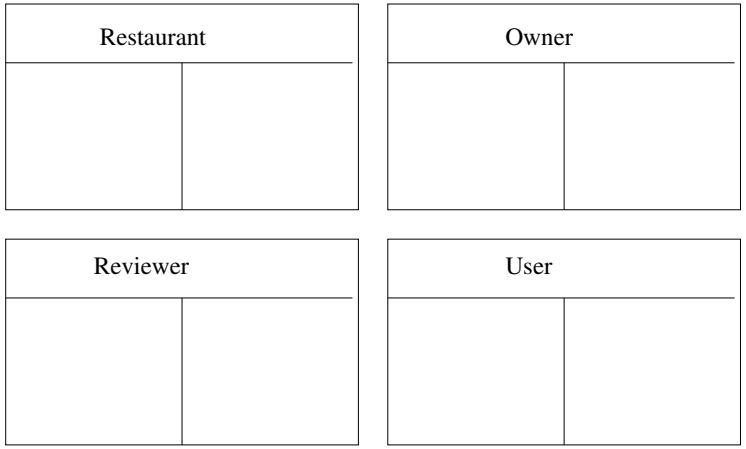
Indentify potential classes

But which ones are the main players?

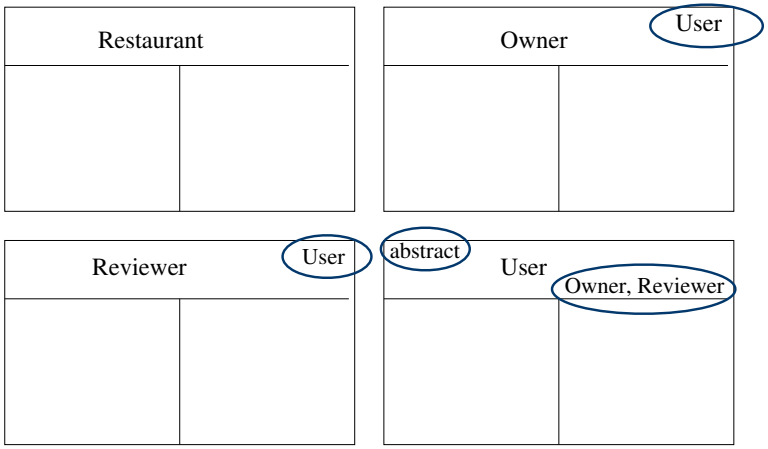
*Each **restaurant** corresponds to a certain price range, neighbourhood, and cuisines it serves. Restaurants that serve alcohol must have a license, which they need to renew every year. The system should also report how long, on average, customers wait for take out in restaurants that offer take-out service. When **reviewers** leave a review for a restaurant, they must specify a recommendation (Thumbs Up or Thumbs Down) and can also leave a comment. An **owner** of a restaurant can respond to a review with a comment. All **users** of the system log in with their username. Users can choose to be contacted by email and ...*

Start building the model

So we begin...



Start to identify inheritance



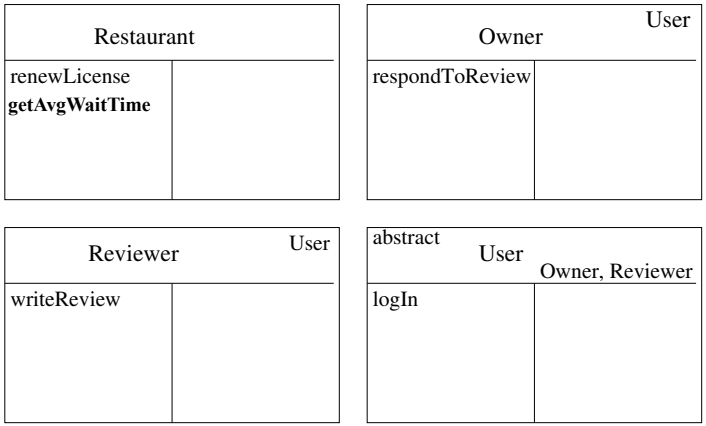
Example

And what do the classes do?

*Each **restaurant** corresponds to a certain price range, neighbourhood, and cuisines it serves. Restaurants that serve alcohol must have a license, which they need to **renew** every year. The system should also **report how long, on average, customers wait for take out** in restaurants that offer take-out service. When **reviewers leave a review** for a restaurant, they must specify a recommendation (Thumbs Up or Thumbs Down) and can also leave a comment. An **owner** of a restaurant can **respond to a review** with a comment. All **users** of the system **log in** with their username. Users can choose to be contacted by email and ...*

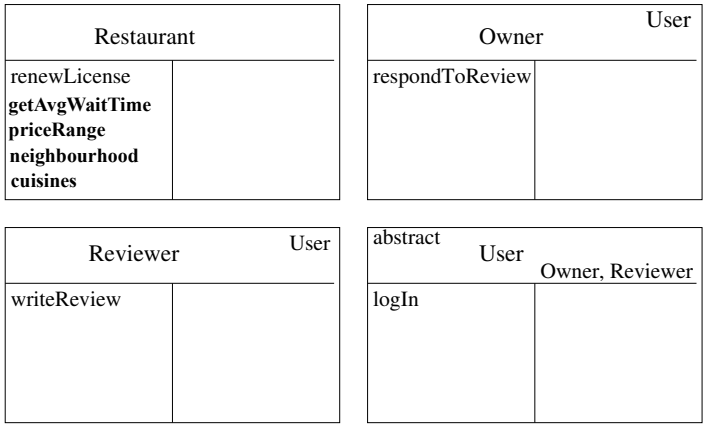
Example

Adding some "what they do" responsibilities...



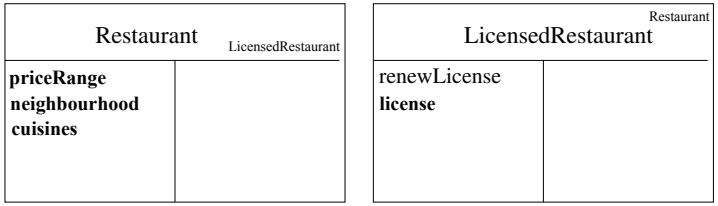
Example

Adding some "what they store" responsibilities for `Restaurant`...



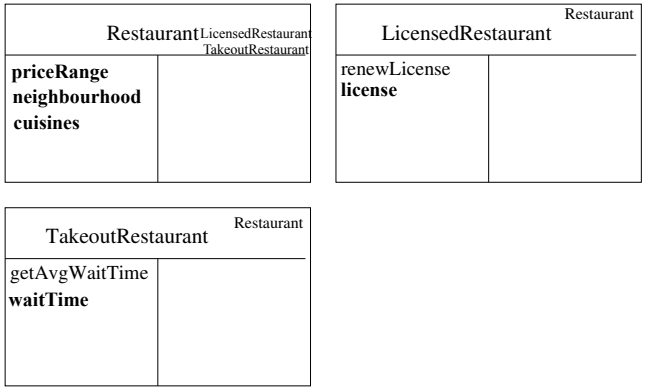
Example

What about the responsibility of storing licenses? But not ALL restaurants have licenses! We need a new type of `Restaurant`. Also, move `renewLicense` responsibility.



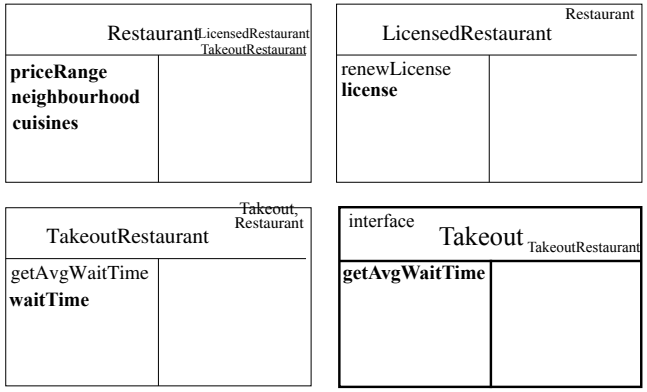
Example

What about the responsibility of storing wait times? Not ALL Restaurants offer takeout! We need a hierarchy.



Example

What if a restaurant is both a `LicensedRestaurant` and a `TakeOutRestaurant`?



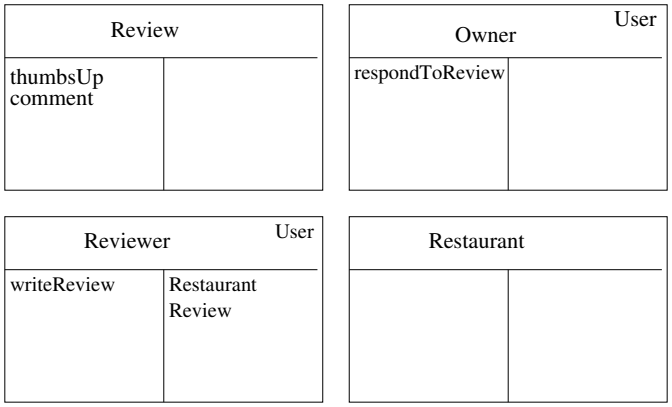
Example

Let's look more closely at reviews.

*Each **restaurant** corresponds to a certain price range, neighbourhood, and cuisines it serves. Restaurants that serve alcohol must have a license, which they need to **renew** every year. The system should also **report how long, on average, customers wait for take out** in restaurants that offer take-out service. When **reviewers leave a review** for a restaurant, they must specify a recommendation (Thumbs Up or Thumbs Down) and can also leave a comment. An **owner** of a restaurant can **respond to a review** with a comment. All **users** of the system **log in** with their username. Users can choose to be contacted by email and ...*

Example

To write a review... it looks like we need a `Review` class.



Example

We have some design decisions to make:

- Does a `Review` know which `Restaurant` it is for?
- Does a `Review` know who wrote it?
- Where do `Review`s live? With a `Restaurant`? With a `Reviewer`?

Example

Make your decisions. Here is one possibility:

Review	
thumbsUp comment reviewer	Reviewer

Owner	User
respondToReview	

Reviewer	User
writeReview	Restaurant Review

Restaurant	
reviews	Review

Example

Let's see if this works...

Scenario: *write a review*.

Scenario walk-through:

To write a review, a `Reviewer` needs to:

- create a `Review`
- provide it to the `Restaurant`
- the `Restaurant` needs to add it

We are missing the last responsibility...

Example

Adding the new responsibility.

Review	
thumbsUp comment reviewer	Reviewer

Owner	User
respondToReview	

Reviewer	User
writeReview	Restaurant Review

Restaurant	
reviews addReview	Review

Exercise -- for practice at home

Continue building the CRC Model by completing a scenario walkthrough for the `respondToReview` scenario.

Now execute a scenario walk-throughs to convince yourself that your design works.

Complete the model by adding all functionality in the description.