Use Case Name: Beer Advisor

Point of Contact: Lucas Standaert (standl@rpi.edu), Marcelo de Castro (decasm3@rpi.edu), Anna Yaroslaski (yarosa2@rpi.edu), Sam Stouffer (stoufs2@rpi.edu)

Beer Advisor

Summary

Beer is said to be one of the oldest alcoholic drinks created by humans. In fact, we have been creating different types of beers for millennia and, hence, due to the wide variety of beers with different flavor profiles and styles, choosing a beer is not a simple matter. There are also many people who are picky about what they drink, as they may not like certain styles of beer. There can also be the concerns of alcohol content and brewery location, as people want to support their local beer scene. In this context, the Beer Advisor helps people in the difficult task which is finding the perfect beer. The application combines information from different databases in order to find the perfect beer match considering the user's requests.

Goal

The goal of this application is to provide users with beer recommendations by matching their preferences with attributes of commercially available beers.

Requirements

- The system must be able to differ beers between styles, alcohol content, and other key factors.
- It will return a listing of beers with all the information the system can provide on them.
- It will have an extensive catalog of beers obtained from trustworthy data sources.

Scope

The application will only recommend beers and will not include other alcohols such as spirits or wines.

The beer selection and beer attributes used for recommendation will be limited to those provided by the application's data sources: beer.db and opendatasoft.

The application will recommend beers based on location, alcohol content, style, IBU, color, original gravity (OG), season and basic ingredients and will not include any additional attributes such as "mouthfeel". Of course, the characteristics on which the recommendation depends, will be used if the information that is available in our resources.

The application does not guarantee recommendations for all combinations. For example, finding an IPA with a low alcohol content may not be feasible. In this scenario, the ontology will recommend similar beverages based upon a set hierarchy, which the user can manipulate if they wish.

Users will have the ability to compare their searches to other similar results that the system has stored.

Stakeholders

- Customers/ Beer drinker.
- Beer distributors.
- Beer stores.
- Breweries.
- Bars.
- Beer Databases.

Actors

Primary: Customer/Beer drinker – This is the person that will benefit directly from the ontology. For them, the ontology will assist in finding beer that they desire in a more efficient manner.

Primary: Beer Store – Owners of beer stores could use the ontology as a way to get more information of beers in their area they may not know about, to further widen their selection.

Primary: Bars – Bars have a similar interaction to that of beer stores, using the ontology to expand their selection.

Secondary: OpenBeerDB

Secondary: beer.db

Secondary: opendatasoft

Secondary: beerkb - An ontology provided to us by Elisa Kendall that we will be reusing in our ontology.

Secondary: brewery – While breweries won't be accessing the ontology for purchases, the beers they make will affect what beers are stored inside the ontology.

Secondary: beer distributors – This is similar to breweries, as the beers that are shipped by distributors will also affect the ontologies database.

Triggers

The trigger to this application is the user launching the Beer Advisor application and performing a search for a beer with a determined set of characteristics. Examples of triggers are listed below.

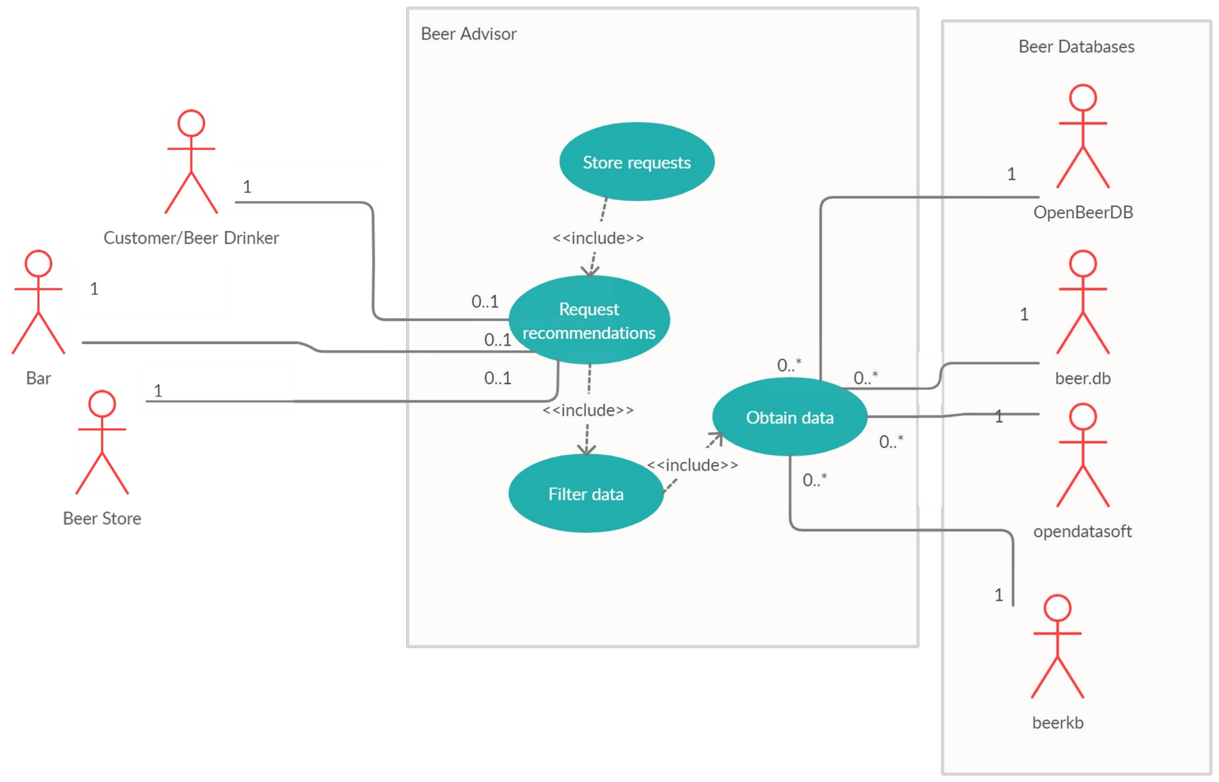
- A customer is unsure of what beers are available in their local area, or what local breweries there are.
- A customer wants to find a new style of beer to try.
- A customer wants to find a new brand of a style they already drink.
- A bar is looking for new brands to expand their current selection.
- A store is looking for new brands to expand their current selection.

Pre-conditions

It is assumed that the user specifies the type of beer and for that the user will need to provide the characteristics he is looking for, such as beer type, alcohol content, season and location.

The application should be web-based and, therefore, accessible from any device. It is also assumed that the application can perform access to beer and breweries databases.
Post-conditions The ontology will contain a plethora of different beers differentiated by type, alcohol content, location, and more. Users will be able to query the ontology for what they desire but a primary actor will have no permanent effect on the system. Therefore, there is no need to save information after the user is satisfied with the recommendation provided by the tool.

Use Case Diagram



This diagram is meant to be an overview of how the system for beer advisor works. Inside of our subject, essence of beer advisor, are the two main operations the system will have. The first operation is the ability to request recommendations. Each user will submit one recommendation based upon our guidelines, which will provide them with a list of beers. These recommendations will be completed by one of these actors, who are our primary actors.

The second action that can be completed is on the back end, obtaining data from our different databases. In this scenario, we will obtain 0 or many different beers from these databases, while for the purpose of this ontology we will only be accessing the data from each database once. This diagram shows a rough draft of what we believe our ontology will look like.

Inside of this are two new actions. The first is the ability to store requests, which can later be used inside of request recommendations. After a user has their results returned, they can ask for a list of similar requests to plug into the system, which come from the stored requests.

The second feature is the connection between requesting recommendations and obtaining data, filtering the data. Inside of this will be a set of rules and requests for look up, but also providing information in the case of poor query results, i.e. if the system is unable to return a satisfactory list without using similarto relationships.

Usage Scenarios

Scenario 1

A man has just moved to Troy and is looking for a new beer. However he's a little picky on what he likes. He looks for beers locally, as he wants to support local microbrews, and mainly drinks american lagers and pilsners. He loads up Beer Advisor to help him look for a new beer to drink. He looks through the types and selects american lagers first. He then selects pilsners as his second choice. He then moves onto the location, and selects Troy, New York. After selecting both of his options he enters his search and is provided a list of different lagers and pilsners from troy breweries. Now he knows what options he has and can purchase the beer he wants.

Scenario 2

A bar in Pittsburgh is looking to expand it's selection of beers, and wants to know what local brands there are. However he wants a selection of winter beers, which are usually darker beers, and wants to cap the alcohol content at 7%. He opens Beer Advisor and begins the selection process. He first selects his region, Northeast, and then selects the season he is looking for. After this, he selects beers that have an alcohol content from 5% to 6% and 6% to 7%. After this, the Beer Advisor provides him the names and locations of the beers, as well as the names of the breweries so he can contact them for direct distribution.

Scenario 3

A user has just finished his search on Beer Advisor, and has found a nice list of beers to go and try out. However, he feels he may have limited his search a bit too much, and wants to see how he can expand his list. He notices that Beer Advisor can offer him a new set of queries, from prior users, that can expand his selection. He goes through the suggestions, adding more styles and adding a new location to expand his current selection. From this modification he comes out with a much larger selection of beer, and feels much happier with his results.

Scenario 4

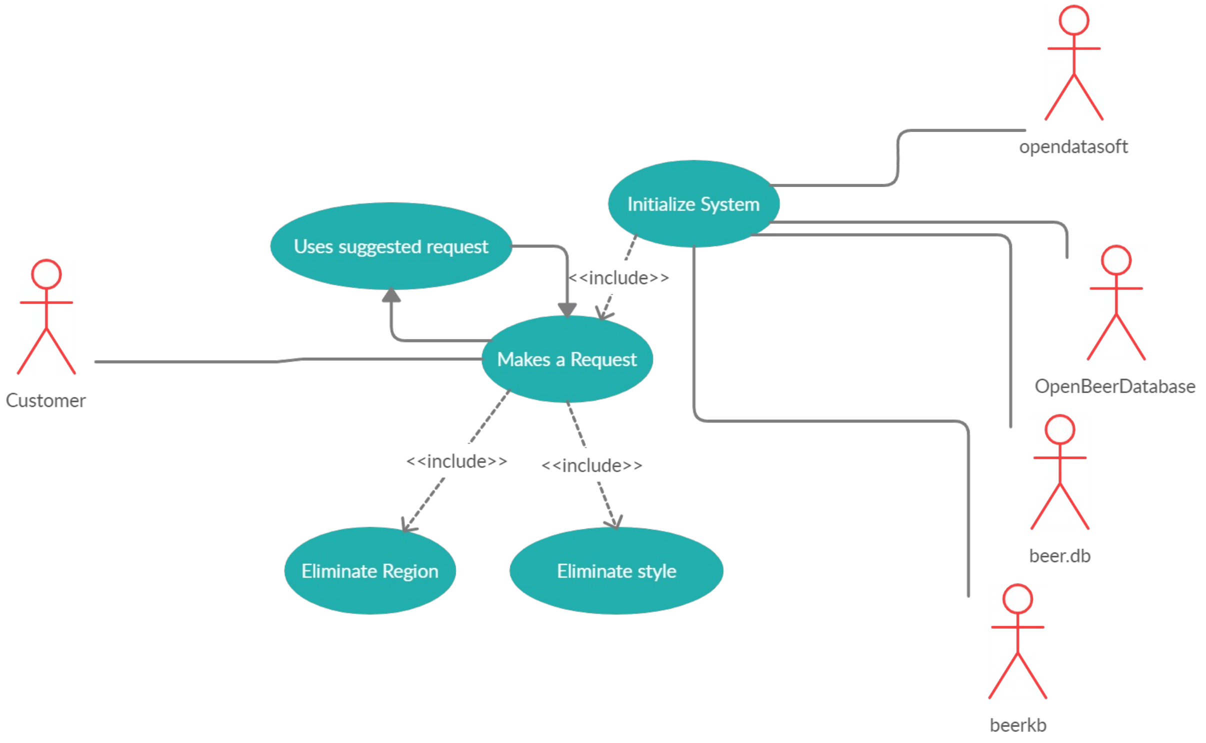
A user lives in a rural area populated with small towns and doesn't know of any local breweries. However they still want to try and stay close to home with their selection, so they load up Beer Advisor and search for breweries close to their hometown. After entering their query, Beer Advisor starts to search. Seeing how there are no breweries in their selected location, it expands its search to the entire state, and finds a multitude of breweries. It returns a list of beers with these breweries to the user, who can now find local breweries they may not have known about.

Flow of Events

Basic Flow of Events

1. User loads up Beer Advisor.
2. User selects a region.
3. User inputs a city.
4. User inputs a season.
5. User inputs a type.
6. System takes these criteria and searches for appropriate items.
7. First the system eliminates any regions that don't match.
8. Then the system eliminates cities that don't match.
9. Next it eliminates seasons that don't match.
10. Finally it narrows down the search to the specific type of beer.
11. System returns items in a list with all information available.

Basic Flow Diagram:



This diagram represents our secondary basic flow. In this diagram, after the system has initialized, a user will make their request, eliminating the beer by style and region. After this the system will return a list and prompt the user to see if they want to make a new request based off of a suggested request. This will then do another query, and add this to the user's current list. This will continue until the user feels satisfied with their choices and leaves the system.

Alternate flow 1

1. User loads up Beer Advisor.
2. User selects a region.
3. User inputs a season.
4. User inputs a type.
5. System takes these criteria and searches for appropriate items.
6. First the system eliminates any regions that don't match.
7. Then the system eliminates cities that don't match.
8. Next it eliminates seasons that don't match.
9. Finally it narrows down the search to the specific type of beer.
10. System returns items in a list with all information available.
11. User requests a list of related searches
12. User selects new request
13. System adds beers from new request to previous list
14. Repeat 12-14 until user is satisfied with their list
15. User exits the system

Alternate flow 2

1. User loads up Beer Advisor
2. User inputs a type
3. User inputs an alcohol content
4. The specific combination of type and alcohol content is impossible, so the beer cannot be found
5. The system searches for similar beers.
6. The system returns a message saying that no beers exist but presents the user with the list of similar beers.
7. The system then prompts the user for a new query
8. The user exits the system

Alternative flow 3

1. User loads up Beer Advisor
2. User selects a region
3. User inputs a type
4. User inputs an alcohol content
5. The specific combination of type and alcohol content cannot be found for that location
6. The system searches for combination in similar locations
7. The system returns a message saying that no beers exist but presents the user with the list of beers from similar locations
8. The system then prompts the user for a new query
9. The user exits the system

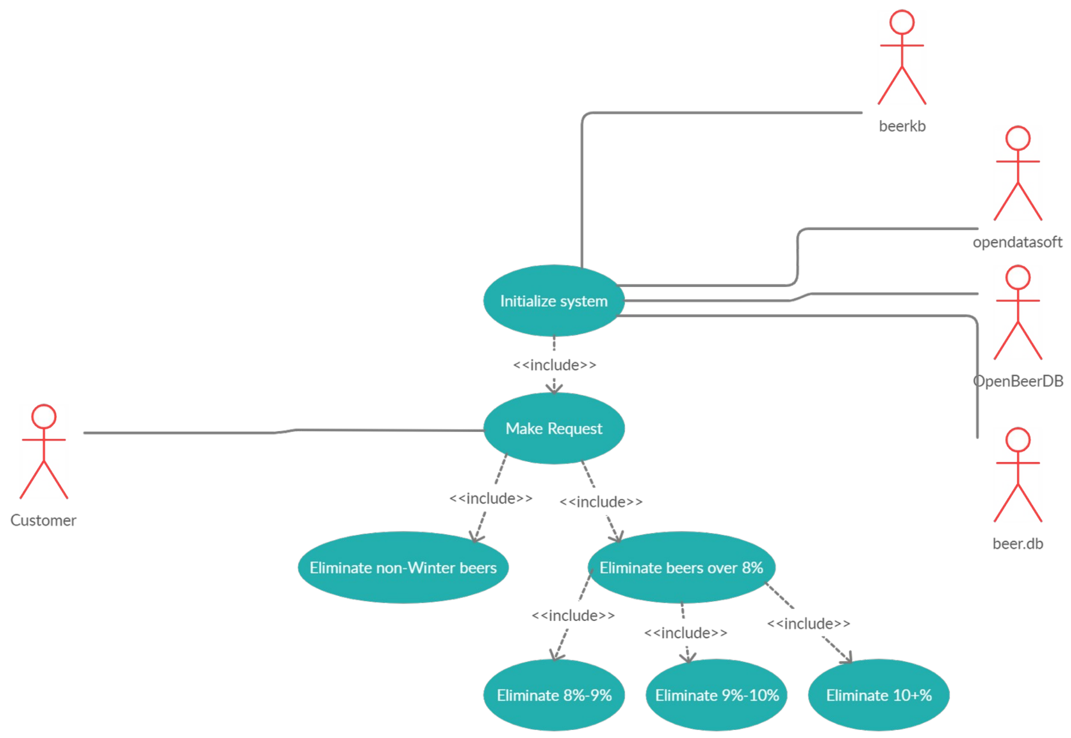
Preliminary Competency Questions

What is a winter beer that is under 8% alcohol content?

Answer: Winter beers are a darker beer with fuller flavor and malted styles. They are also more full bodied. Some classic examples include stouts, Russian imperial stouts, porters, and ales. Of these, ales, stouts, and porters will be under 8% on average. However, Russian imperial stouts average at around 9%, so the list will mainly include ales, stouts, and porters, and a few Russian imperial stouts.

In order to complete this first question the ontology should be able to obtain a list of beers, separate them by seasons, and then limit the alcohol content of said beer. The main focus is also upon the region, as many beers may not be available to the customer given where they are from.

Diagram

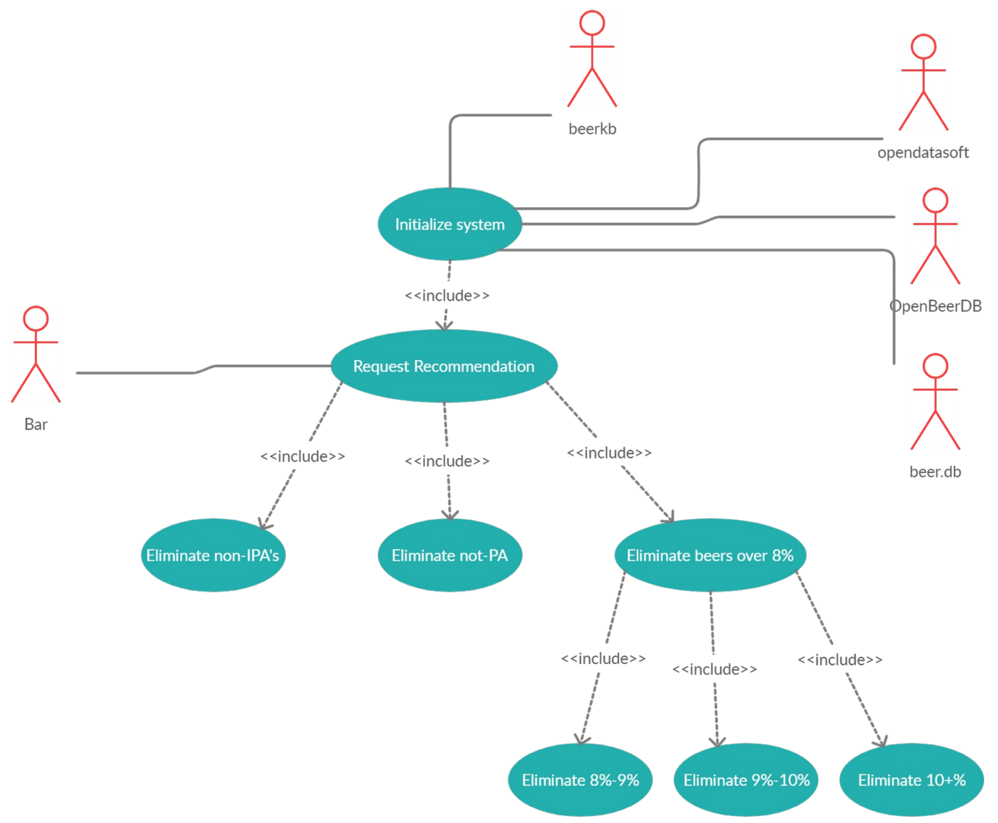


This diagram showcases how the system will process this request. Upon initializing the system, it will access it's database to obtain any necessary or new information. After this, the user will make a request, in this scenario eliminating beers non-winter beers and beers over 8 percent. In order to store the alcohol content, beers will be put into bracketed groups. These brackets will go in one percent increments from 5 to 10 percent. Anything above 10 will be grouped together and anything below five will be grouped together.

What is a brewery in Pennsylvania that makes IPA's under 8%?

Answer: A personal favorite brewery of mine is Helltown Brewing, located in Mt Pleasant, PA. The company makes a whole line of IPA's that meet this category, including the Buffy, Salem, and Rapture IPA's.

For many people, supporting their local breweries is important. When you move to a new area you may not have any idea what breweries are local to you and what kinds of beer they make. This question is aimed at targeting that part of the ontology. By combining information from different databases, the ontology should be able to filter out the location and region first, then remove the incorrect types of beer, reporting back the list.



Diagram

This diagram illustrates how the system will interact with the user to find their result. Similar to the diagram above, the system first initializes and obtains the necessary data. From here, the user will input their request and the system will obtain the proper results. In this case the system will first eliminate any beer that is not an IPA, and then eliminate any beer that is not from Pennsylvania. It will then complete a similar process to eliminate any beers over 8%, providing the bar owner with the proper result.

What is an IPA that is 5% or below?

Answer: The average alcohol by volume (ABV) for an IPA is around 5.5-7%. An IPA that is around this alcohol content may be very hard to find and the ontology may not yield results for this search. In this scenario, it will instead recommend a similar style of beer to an IPA. An example would be a recommendation of a pale ale, which is similar to an IPA but has a lower alcohol content, or a session IPA, a hybrid between an IPA and a session beer.

In order to get this answer, the ontology will first obtain data from our current resources, including an ontology provided to us by the instructors. Beers will then be placed into their proper categories based on different style guidelines. These guidelines will form the basis for our inferences of a beer's type based on the factors listed above. For example, IPA's are on average between 5.5-7% abv, so this becomes a part of the inference guidelines..

After the system attempts to query it's database, it will find no results. The ontology will then check a relationship called similarTo, utilizing beer as its criteria. It will then access a hierarchy which checks similarity. As a default, it will search for beers that fall under a similar subclass.. In this scenario, it will find pale ale and session IPA as similar beers, and return them..

I really like New Belgium's IPA's, what other beers have people searched for from New Belgium?

Answer: New Belgium is a Colorado based brewery that makes a multitude of different beers. One example answer would be Fat Tire Amber Ale, a very light beer that is easy to drink. So the application can have Fat Tire Amber Ale as output.

This answer will leverage the ontologies ability to relate user searches. In order to get this result, the application will go through stored previous searches from that user, see the beers that are from New Belgium and check which ones are similar to the current search. The ontology would be used to match this information, and the application will return a list of beers to the user, sorted in a way that prioritizes the best match..

Is there a stout made by a local brewery in Idyllwild, California?

Answer: Idyllwild is a fairly distant city that only has a well known brew pub. However, depending upon the season, they may not be making stouts. The system would respond with breweries that are nearby, in the state of california, that do make stouts, such as Stone Brewing company.

In order to obtain this answer, the ontology will first query it's database for any porters that exist within the stated city. The decision about looking for porters is because we can set similarTo in the ontology, and we know that stouts are similar to porters. In addition to that, if stouts are similar to other types of beers, the application can look for those as well. If there is no other beer similar to stout, the application can trace beers that are similar to porters. If this fails, it will then use a relationship between breweries, known as nearTo, and will continue until it finds a sufficient amount of results. Once this is done, it will return the results to the user.

Resources

Data	Type	Characteristics	Description	Owner	Source
OpenBeerDB	Open Database		Open database containing information on available beer		https://openbeerdb.com/
beer.db	Web Service	Open public domain beer	Beer based database that is	beer.db	https://openbeer.gith

		database	open to public domain		ub.io/
Open beer database	Web API		A database containing beers by style, category, country, brewer, and address	opendatasoft	https://data.opendatasoft.com/explore/dataset/open-beer-database%40public-us/table/
beerkb	Beer ontology		Existing beer database with	Elisa Kendell	http://www.cs.umd.edu/projects/plus/SHOE/onts/beer1.0.html

All Diagrams created at creately.com.