Last Updated: June 7th, 2013
Last Updated By: Collin Styles

Design Use Cases

Wine Application: Wine-Know

Terroir Integrated Technology Specialists, T.I.T.S.

Shahrzad Gustafson	Project Manager
Collin Styles	Algorithm Specialist
Tim Chi	Software Architect
Janne Rapakko	Software Development Lead
Jagtej Sodhi	Software Development Lead
Jaysen Parmar	User Interface Specialist
Ankit Agarwal	Senior System Analyst
Ricardo Cruell	Subject Matter Expert
Kyle So	Quality Assurance Lead
Trent Russell	Database Specialist

Terroir Integrated Technology Specialists
Date Compiled: April 26, 2013

Table of Contents

Design Use Case Key

Description: A summary of what that specific design use case does

Desired Outcome: What is supposed to happen

User Goals: What the user wants to accomplish in this action

Dependent Use Cases: If the use case is dependent on any previous ones to work

Involved Requirements: The corresponding requirements for the Use Case

Details:

Priority – The urgency of the use case to the application

0 - Completed

1- Highest Priority, Vital to Application

2- High Priority

3- Low Priority, Implemented if given time

Progress- The current phase the use case is in

Planning – Use Case is currently being considered and planned

Designing- Use Case is being laid-out and designed with team

Developing- Use Case is currently being implemented

Testing – Use Case is in test phase, being test with the team and potential users

Completed – Use Case is fully implemented into application and currently functioning

Deferred- Development of use case is currently on hold

Test Phase – The current testing phase the use case is in

Planned - Testing will commence when development is completed

In Progress - Currently testing with team

Complete - Test phase complete, Use Case currently functional

Deferred - Development of use case is currently on hold, therefore testing is as well

Pre-Conditions:

Actions that have to be true, before this Design Use Case executes

Post-Conditions:

The outcome of the Design Use Case being completed

Trigger:
How the Design Use Case begins (or how it is "triggered")

Workflow:
The system steps involved in the Design Use Case

Alternate Paths:
Other paths the system can take if they decide not to complete the Design Use Case

Terms

Random Wine – A wine that is completely random. The system randomly selects a wine (no user input)

Specific Wine – A specific wine is one that is recommended by the system based off of some specifications, which were inputted by the user. The option the user has in selecting a specific wine is either by color, varietal or year.

Food Items - When the term 'Food Items' is used, it includes: Cured Meat, Red Meat, White Meat, Rich Fish, Fish, Bread, Roasted Vegetables, Vegetables, and Sweets

Type of Dish – When 'kind of dish' or 'dish' is used, it includes: Red Meat, White Meat, Fish, Starches and Vegetarian

Varietals – When the user is given the option to choose a wine by varietal this refers to 40 different varietals. Some examples of these varietals include: Merlot, Cabernet Sauvignon, Chardonnay, Zinfandel, Pinot Noir, Syrah/Shiraz, Sauvignon Blanc, Riesling, Viognier

Wine Glass Rating – A wine glass rating is the same thing as a 5 start rating, instead wine glasses are used. Below is an example of a '4 Wine Glasses'

Design Use Case #1: Recommend Random Wine

Description:

This Design Use Case outlines the ability for the user to be recommended a Random wine

Desired Outcome:

A random wine is recommended by the system

User Goals:

The user does not know what type of wine they want to be recommend so they would like to see what random wine the app will recommended

Dependent Design Use Cases:

None

Involved Requirements:

SR 01 (Recommend Random Wine)

Details:

Priority – 0

Progress Status – Completed

Test Phase – Completed

Pre-Conditions:

The system must be on the sip page (Sip.java)

Post-Conditions:

The system shall suggest a random wine and will display the wine to the user. The system shall display the name of the wine, the varietal, the year, the color

Trigger:

From the "Sip" activity, the user presses the "Random Wine" button.

Workflow:

1. The "randomWineClicked" activity is launched from the "Sip" activity.
2. RandomWineClicked calls "getRandomWine()".

3. getRandomWine generates a random wine ID and returns the corresponding wine from the database.
4. randomWineClicked displays the wine and it's information on the screen.

Alternate Paths:

None

Design Use Case #2: Recommend Specific Wine: Red

Description:

This design use case outlines the ability for the user to be recommended a Specific Red Wine. The user has in mind a wine they want based off of either: Color, Varietal or Descriptor and in this case they have a specific color in mind, Red.

Desired Outcome:

The user shall be recommended a red wine.

User Goals:

The user wants to be recommended a red wine

Dependent Design Use Cases:

None

Involved Requirements:

SR 02

Details:

Priority – 0

Progress Status – Completed

Test Phase – Completed

Pre-Conditions:

They user must have select 'Specific' from the 'Sip page'

Post-Conditions:

The system shall recommend and display to the user a red wine

Trigger:

From the "SpecificWine" activity, the user presses "Color" and then "Red" on the next screen.

Workflow:

1.The "SpecificRedWine" activity is launched from the "SpecificWine" activity.
2. SpecificRedWine calls the database method "getRedWine()".
3. getRedWine() returns all red wines in the database.
4. SpecificRedWine generates a random ID and obtains the wine corresponding to that ID.
5. SpecificRedWine outputs the information (e.g., name, varietal, etc.) for that wine.

Alternate Paths:

None

Design Use Case #3: Recommend Specific Wine: White

Description:

This use case outlines the ability for the user to be recommended a Specific White Wine. The user has in mind a wine they want based off of either: Color, Varietal or Descriptor and in this case they have a specific color in mind, White

Desired Outcome:

The user shall be recommended a white wine

User Goals:

The user wants to be recommended a white wine

Dependent Design Use Cases:

None

Involved Requirements:

SR 03

Details:

Priority – 0

Progress Status – Completed

Test Phase – Completed

Pre-Conditions:

They user must have select 'Specific' from the 'Sip page'

Post-Conditions:

The system shall recommend and display (name, color, varietal, and year) to the user a white wine

Trigger:

From the "SpecificWine" activity, the user presses "Color" and then "White" on the next screen.

Workflow:

1. The "SpecificWhiteWine" activity is launched from the "SpecificWine" activity.
2. SpecificWhiteWine calls the database method "getWhiteWine()".
3. getRedWine() returns all white wines in the database.
4. SpecificWhiteWine generates a random ID and gets the wine corresponding to that ID.
5. SpecificWhiteWine outputs the information (e.g., name, varietal, etc.) for that wine.

Alternate Paths:

None

Design Use Case #4: Recommend Specific Wine: Varietal

Description:

This use case outlines the ability for the user to be recommended a Specific Wine by Varietal. The user has in mind a wine they want based off of either: Color, Varietal or Year and in this case they have a specific Varietal in mind

Desired Outcome:

The user shall be recommended a wine based off of the varietal they specify

User Goals:

The user wants to be recommended a wine based off of a specific varietal

Dependent Design Use Cases:

None

Involved Requirements:

SR 04

Details:

Priority – 0

Progress Status – Completed

Test Phase – Completed

Pre-Conditions:

They user must have select 'Specific' from the 'Sip page'

Post-Conditions:

The system shall recommend and display (name, color, varietal, year and review) to the user a wine that is of the varietal they indicated

Trigger:

From the "SpecificWine" activity, the user presses the "Varietal" button.

The user then presses the button corresponding to the varietal that they have in mind.

Workflow:

1. The "VarietalClicked" activity is launched from the "SpecificWine" activity.
2. The "VarietalSelected" activity is launched and a varietal is passed in via an Intent.
3. VarietalSelected calls "getWinesByVarietal()", passing the varietal as a parameter.

4. getWinesByVarietal() returns all wines of that varietal in the database.
5. VarietalSelected generates a random ID and gets the wine corresponding to that ID
6. VarietalSelected outputs the information (e.g., name, varietal, etc.) for that wine.

Alternate Paths:

None

Design Use Case #5: Recommend Specific Wine: Descriptor

Description:

This use case outlines the ability for the user to be recommended a Specific Wine by Year The user has in mind a wine they want based off of either: Color, Varietal or Descriptor and in this case they have a specific Descriptorin mind

Desired Outcome:

The user shall be recommended a wine based off of the desciptorr they specify

User Goals:

The user wants to be recommended a wine based off of a specific descriptor

Dependent Use Cases:

None

Involved Requirements:

SR 05

Details:

Priority – 0

Progress Status – Completed

Test Phase – Completed

Pre-Conditions:

They user must have select 'Specific' from the 'Sip page'

Post-Conditions:

The system shall recommend and display (image of wine label, name, color, varietal, year and review) to the user a wine that is of the varietal they indicated

Trigger:

From the "SpecificWine" activity, the user presses the "Descriptor" button.

The user then presses the button corresponding to the descriptor that they have in mind.

Workflow:

1. The "DescriptorClicked" activity is launched from the "SpecificWine" activity.

2. The “DescriptorSelected” activity is launched and a descriptor is passed in via an Intent.
3. DescriptorSelected calls “getWinesByDescriptor()”, passing the descriptor as a parameter.
4. getWinesByDescriptor() returns all wines in the database that have that descriptor.
5. DescriptorSelected generates a random ID and gets the wine corresponding to that ID
6. DescriptorSelected outputs the information (e.g., name, varietal, etc.) for that wine.

Alternate Paths:

None

Design Use Case #6: Wine Pairings

Description:

This use case outlines the ability for the user to be recommended a wine that goes with a specific food item (See Terms)

Desired Outcome:

The user shall be suggested a wine that goes with a specific food item that they have indicated

User Goals:

The user wants to be suggested a wine that pairs nicely with the food item they indicated

Dependent Design Use Cases:

None

Involved Requirements:

SR 06

Details:

Priority – 1

Progress Status – In progress

Test Phase – In progress

Pre-Conditions:

The user shall be on the 'Eat' page

Post-Conditions:

The system shall suggest a wine that goes with the specific food item the user had indicated

Trigger:

From the "Eat" activity, the user presses the "Pairing" button.

The user then presses the button corresponding to the type of meal that they have in mind.

Workflow:

1. The "PairingClicked" activity is launched from the "Eat" activity.
2. The "DisplayPairingWine" activity is launched and a number (corresponding to the type of food pressed) is passed in the Intent.
3. DisplayPairingWine calls "getWines()", passing in the food type as a parameter.
4. getWines() returns a list of all types of wines (e.g., "bold red wine", "dessert wine", etc.) that pair well with that food type.

5. DisplayPairingWine generates a random ID and gets the wine type corresponding to that ID.
6. DisplayPairingWine calls "getWinesByVarietal" on the varietal randomly selected earlier.
7. GetWinesByVarietal returns a list of all wines in the database of that varietal.
8. DisplayPairingWine generates a random ID and gets the wine corresponding to that ID.
9. DisplayPairingWine outputs the information (e.g., name, varietal, etc.) for that wine.

Alternate Paths:

None

Design Use Case #7: Recipes

Description:
This use case outlines the ability for the user to view different recipes, based on a type of dish (See Terms), that uses wine as an ingredient

Desired Outcome:
The user shall be suggested a specific recipe that satisfies both the type of dish the user selected and the recipe incorporates wine as one of the ingredients

User Goals:
The user wants to view a recipe that satisfies the type of dish they want to cook and also that the recipe incorporates wine into the dish

Dependent Design Use Cases:
None

Involved Requirements:
SR 07

Details:
Priority – 0
Progress Status – Completed
Test Phase – Completed

Pre-Conditions:
The user shall be on the 'Eat' page

Post-Conditions:
The system shall recommend and display a recipe that uses satisfies the type of dish the user wanted and also uses wine as an ingredient

Trigger:
From the "Eat" activity, the user presses the "Recipes" button.

The user then presses the button corresponding to the type of food that they have in mind.

Workflow:
1. The "RecipeClicked" activity is launched from the "Eat" activity.
2. "onClick()" is called when the user presses on one of the food type buttons.

3. onClick() checks which button was pressed and sets the visibility of the recipe corresponding to that food type to “visible”. It also set the visibility of all other recipes to “invisible”.

<u>Alternate Paths:</u>
None

Design Use Case #8: Give Rating

Description:

This use case outlines the ability for the user to give a rating of 1-5 'Wine Glasses' (5 being the best) of a specific wine

Desired Outcome:

The user shall be able to rate their wine on a scale of 1-5 'Wine Glasses' and then be given confirmation and an option to give another rating, or view the ratings

User Goals:

The user wants to rate a wine they have had before

Dependent Design Use Cases:

None

Involved Requirements:

SR 08

Details:

Priority – 0

Progress Status – Completed

Test Phase –Completed

Pre-Conditions:

The user shall be on the 'Rating' page

Post-Conditions:

The system shall be updated with the users rating of the specific wine they chose

Trigger:

1. From the "ReviewClicked" activity, the user presses the "Give A Rating" button.
2. The user then presses the button corresponding to the wine that they want to give a review for.
3. The user then selects a rating from the options presented.

Workflow:

1. The "GiveRatingClicked" activity is launched from the "ReviewClicked" activity.
2. GiveRatingClicked generate ten random wines and displays them on-screen.

3. When the user selects a wine, GiveRatingClicked displays a set of options for rating (e.g., 1 wine glass, 2 wine glasses, etc.).
4. When the user selects a rating, GiveRatingClicked calls the "setRating" database method and passes in the ID of the wine that was selected.
5. setRating() updates the database with the new rating value.
6. GiveRatingClicked prints that the update was successful and asks the user if they want. to rate another wine or view their rated wines.

Alternate Paths:

None

Design Use Case #9: View Ratings

Description:
This use case outlines the ability for the user to view the rating of wines they have rated.

Desired Outcome:
The user shall be able to view the ratings of all the wines they have rated.

User Goals:
The user wants to view the wine ratings

Dependent Design Use Cases:
None

Involved Requirements:
SR 09

Details:
Priority – 0
Progress Status – Completed
Test Phase –Completed

Pre-Conditions:
The user shall be on the 'Rating' page

Post-Conditions:
The system shall display to the user a list of wine names and their corresponding rating

Trigger:
From the "ReviewClicked" activity, the user presses the "View Ratings" button.

Workflow:
1. The "GiveRatingClicked" activity launches the "ViewRatingClicked" activity.
2. ViewRatingClicked calls the "getRatedWines()" database method.
3. getRatedWines() returns a list of wines that the user has rated.
4. ViewRatingClicked formats and outputs the list of wines returned by getRatedWines().

Alternate Paths:
None

Design Use Case #10: View Wine Terms

Description:

This use case outlines the ability for the user to view the list of all common wine terms

Desired Outcome:

The user shall be able to view the list of wine terms that they can select to choose a definition for

User Goals:

The user wants to view the list of wine terms to see if a word intrigues them or if they have a specific word in mind they can see if they word is there so they can view the definition for it

Dependent Design Use Cases:

None

Involved Requirements:

SR 10

Details:

Priority – 0

Progress Status – Completed

Test Phase –Completed

Pre-Conditions:

None

Post-Conditions:

The system shall display to the user a list of all the wine terms

Trigger:

From any screen, the user presses the "Terms" button.

Workflow:

1. The "TermsClicked" activity is launched from wherever the user currently is.
2. TermsClicked outputs a scrollable list of buttons, each labelled with a unique wine term.

Alternate Paths:

None

Design Use Case #11: View Definition of Wine Term

Description:
This use case outlines the ability for the user to view the definition of a term from the list of wine terms

Desired Outcome:
The user shall be able to view the definition of a wine term from the list of wine terms

User Goals:
The user wants to view the list of wine terms to see if a word intrigues them or if they have a specific word in mind they can see if they word is there so they can view the definition for it

Dependent Design Use Cases:
None

Involved Requirements:
SR 11

Details:
Priority – 0
Progress Status – Completed
Test Phase –Completed

Pre-Conditions:
UC 10 – View Terms:

Post-Conditions:
The system shall display to the user the definition that goes with the terms they selected

Trigger:
From the "TermsClicked" activity, the user presses the button for a specific wine term.

Workflow:
1. The "onClick()" method for "TermsClicked" is launched when the user presses a button.
2. onClick() checks which button was pressed and fetches the string which contains the definition of that term.
3. onClick() outputs the definition that was looked up earlier.

Alternate Paths:
None

Design Use Case #12: View Random Facts

Description:

This use case outlines the ability for the user to view random facts about wine

Desired Outcome:

The user shall be able to view random facts about wine

User Goals:

The user wants to view random facts about wine

Dependent Design Use Cases:

None

Involved Requirements:

SR 12

Details:

Priority – 0

Progress Status – Completed

Test Phase –Completed

Pre-Conditions:

None

Post-Conditions:

The system shall display to the user a random fact about wine

Trigger:

From the "FactsClicked" page, the user presses the image of the man.

Workflow:

1. The "FactsClicked" activity generates a random ID.
2. FactsClicked looks up the fact corresponding to that ID.
3. FactsClicked displays the random fact in the speech bubble.

Alternate Paths:

None

Design Use Case #13: Get Help

Description:
This use case outlines the ability for the user to get help on how to use Wine-Know

Desired Outcome:
The user shall be able to get help on how to use Wine-Know

User Goals:
The user wants to get help on how to use Wine-Know

Dependent Design Use Cases:
None

Involved Requirements:
SR 13

Details:
Priority – 0
Progress Status – Completed
Test Phase –Completed

Pre-Conditions:
None

Post-Conditions:
The system shall display a help sheet that describes in more detail what 'Sip', 'Terms', 'Rate', 'Facts' and 'Eat' refer to

Trigger:
From anywhere, the user presses the "?" button in the top-right corner of the screen.

Workflow:
1. The "Help" activity is launched from wherever the user currently is.
2. Help displays a list of descriptions of the app's main features.

Alternate Paths:
None

Design Use Case #14: Go Home

Description:

This use case outlines the ability for the user to return to the home page

Desired Outcome:

The user shall be able to return to the home page

User Goals:

The user wants to return to the home page

Dependent Design Use Cases:

None

Involved Requirements:

SR 14

Details:

Priority – 0

Progress Status – Completed

Test Phase –Completed

Pre-Conditions:

None

Post-Conditions:

The system shall return the user to the home page

Trigger:

From anywhere, the user presses the house-shaped button in the top-left corner of the screen.

Workflow:

1. From wherever the user currently is, the "MainActivity" activity is launched.

Alternate Paths:

None