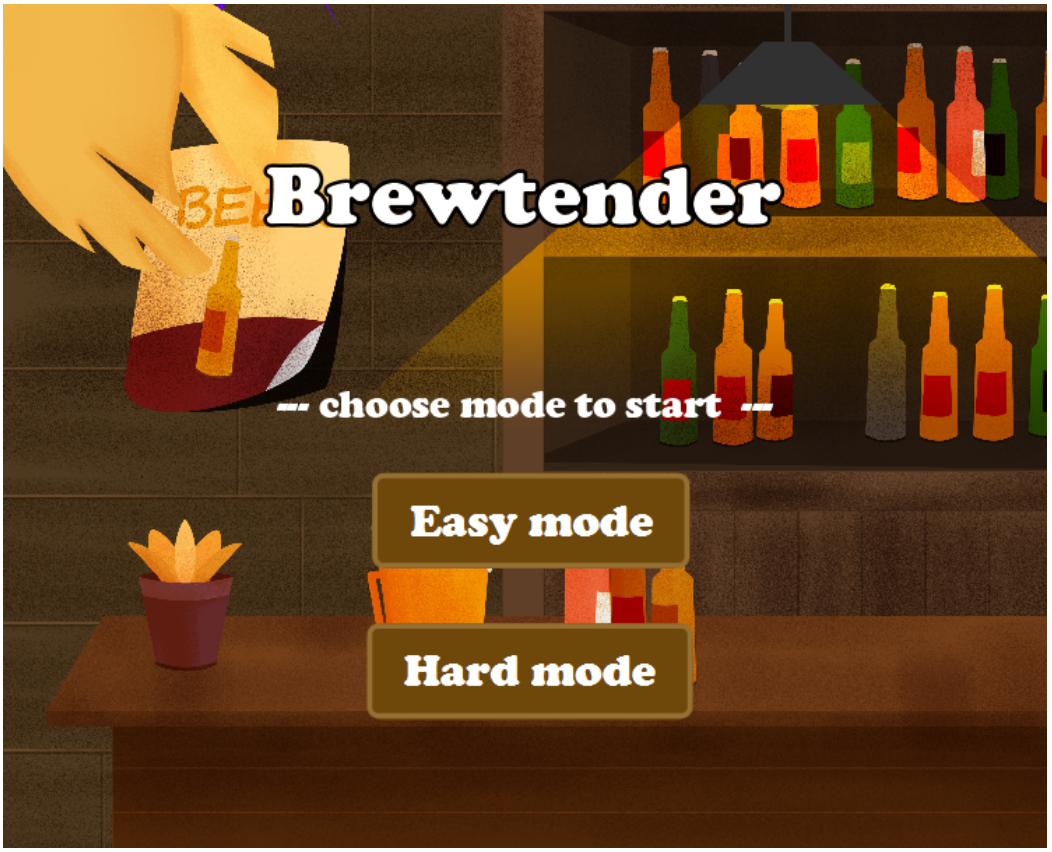
Brewtender



Created by

Krissanapat Chatwattana 6431303921
Chakkrit Jongkraijak 6431306821

2110215 Programming Methodology
Semester 2 Year 2021
Chulalongkorn University

Brewtender

1. Introduction

Brewtender is inspired by Genshin impact's bartender event. In the game, you are assigned as the barista. Your duty is to serve a drink to customers. The game's objective is to send orders as ordered to get the score as high as possible.

2. Game Controls

The main menu is on the first page. This page contains two buttons that are "Easy mode" and "Hard mode" buttons. "Easy mode" will lead you to the game that has easy customer orders. and "Hard mode" will lead to complicated orders.

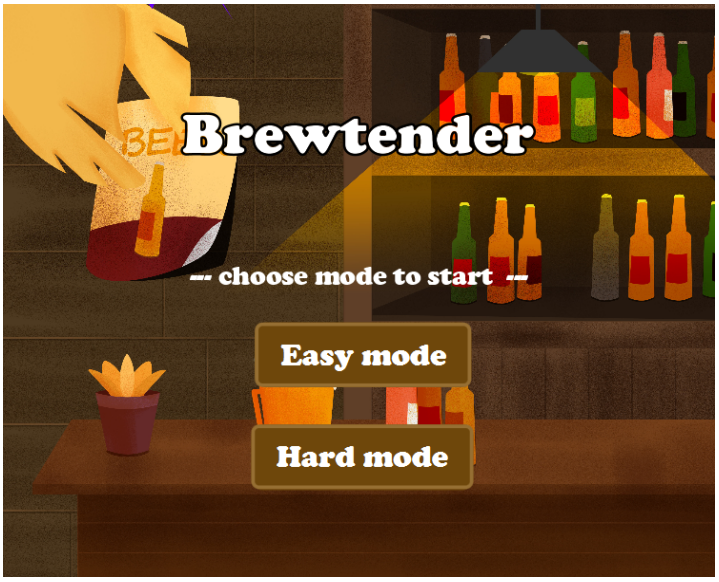


Figure 1 : Start screen

Game Screen

After clicking any button, it will lead you to the game screen.

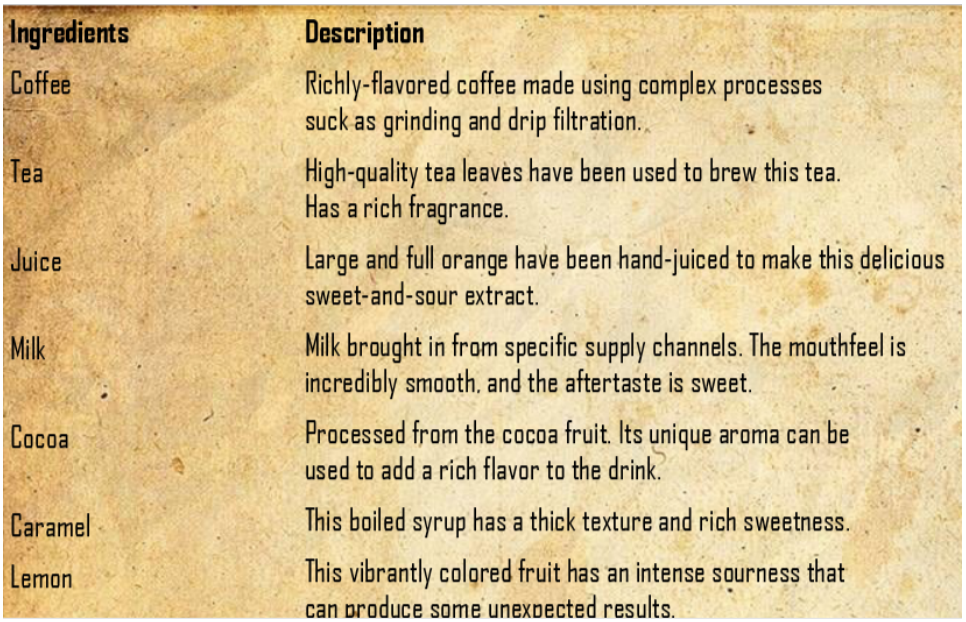
Figure 2 : Game screen

when you click the ‘Start’ button, the game will start.

Figure 3 : Start game screen

when the game starts, the timer will start counting down.
You have only 5 minutes per round. You have to pick the ingredients to make the menu that customers request in the ‘Order’ bar. When you click on the ingredients buttons, they will change the border to green and show the number that you’ve picked. And you also have to pick the size of glass in the ‘Size’ bar. then, click on the ‘Serve’ button to serve. If you want to clear the ingredients that you’ve picked, you can do it by clicking the ‘Clear’ button. you have 3 times to try before the customer will be angry and skip the order.

You can see the recipe of the drinks by clicking the ‘Recipes’ button and it will show you this window.



Ingredients	Description
Coffee	Richly-flavored coffee made using complex processes such as grinding and drip filtration.
Tea	High-quality tea leaves have been used to brew this tea. Has a rich fragrance.
Juice	Large and full orange have been hand-juiced to make this delicious sweet-and-sour extract.
Milk	Milk brought in from specific supply channels. The mouthfeel is incredibly smooth, and the aftertaste is sweet.
Cocoa	Processed from the cocoa fruit. Its unique aroma can be used to add a rich flavor to the drink.
Caramel	This boiled syrup has a thick texture and rich sweetness.
Lemon	This vibrantly colored fruit has an intense sourness that can produce some unexpected results.

when the time is run out. It will show the score page with your grade and scores. You can click ‘New game’ to restart the game or click ‘Quit’ to close the window.

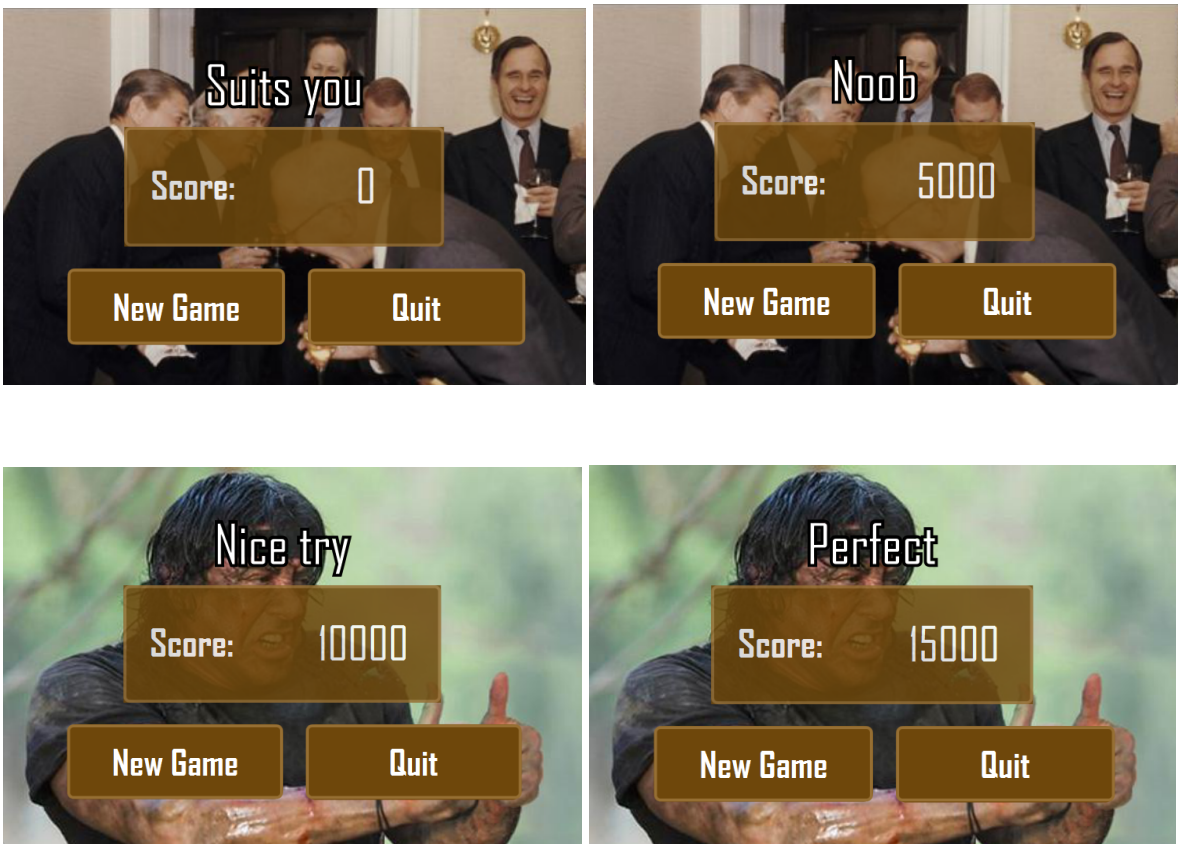
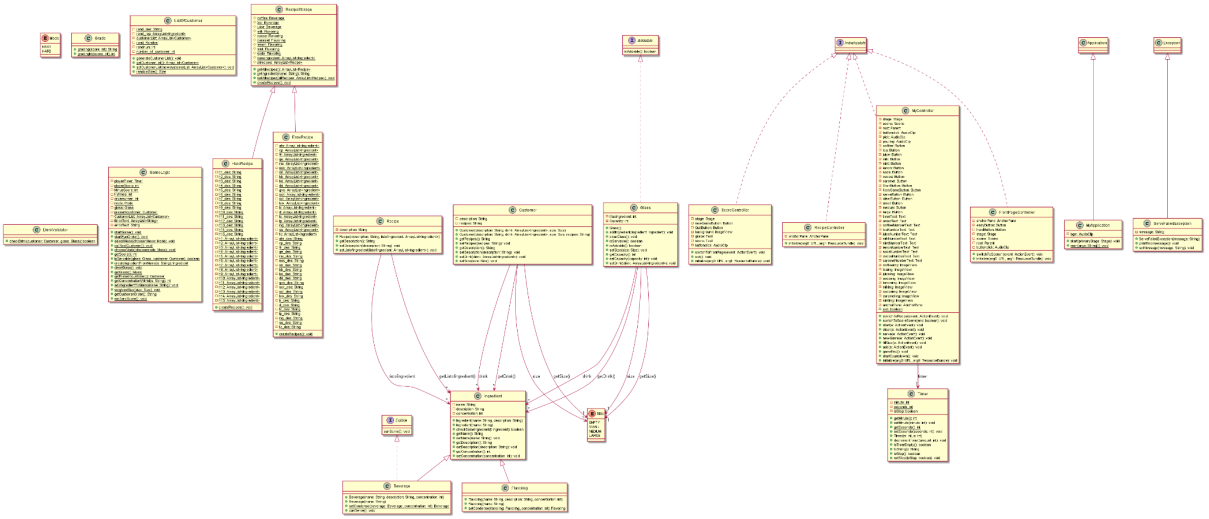


Figure 4 : Score screen

3. UML diagrams



4. Class Details

1. Package component

1.1 class Ingredient

This calss provide base information about ingredient.

1.1.1 Field

Name	Description
- String name	name of the ingredient
- String description	description of the ingredient
- int concentration	number of the ingredient - set value to 1

1.1.2 Constructor

Name	Description
+ Ingredient(String name,String description)	Initialize fields name and description
+ Ingredient(String name)	Initialize name and set description to an empty String

1.1.3 Method

Name	Description
+ boolean checkSameIngredient (Ingredient i)	check if 2 ingredients are the same by checking the name and concentration, return false if none.
+ getter & setter of each field	

1.2 class Beverage extends Ingredient implements Edible

This class present Ingredient in type of beverage. The ingredient that must add to make a edible drink.

1.2.1 Constructor

Name	Description
+ Beverage(String name,String description,int concentration)	-Initialize fields
+ Beverage(String name)	-Initialize name

1.2.2 Method

Name	Description
+ static Beverage setCondense(Beverage beverage,int concentration)	-create new Beverage class with the name of the given Beverage class -set its concentration -return created class
+ void canServe()	return nothing

1.3 class Flavoring extends Ingredient

This class present Ingredient in type of flavoring that cause flavor to the drink

1.3.1 Constructor

Name	Description
+ Flavoring(String name,String description,int concentration)	-Initialize fields
+ Flavoring(String name)	-Initialize name

1.3.2 Method

Name	Description
+ static Flavoring setCondense(Flavoring beverage,int concentration)	-create new Flavoring class with the name of the given Flavoring class -set its concentration -return created class

1.4 **Interface Edible**

This interface present that the class is edible

Name	Description
+ void canServe()	

2. Package component.recipe

2.1 class Recipe

This class provide base information about recipe.

2.1.1 Field

Name	Description
- String description	description of the recipe.
- ArrayList<Ingredient> listofingredient	list of ingredient in recipe.

2.1.2 Constructor

Name	Description
+ Recipe(String description,ArrayList<Ingredient> listOfIngredient)	initialize fields

2.1.3 Method

Name	Description
+ getter & setter of each field	

2.2 class RecipeStorage

This class present the base storage of all recipes

2.2.1 Field

Name	Description
# static Beverage coffee	create Beverage class with -name “Coffee” -description "Richly-flavored coffee made using complex processes suck as grinding and drip filtration." -concentration 1
# static Beverage tea	create Beverage class with -name “Tea” -description "High-quality tea leaves have been used to brew this tea. Has a rich fragrance." -concentration 1
# static Beverage juice	create Beverage class with -name “Juice” -description "Large and full orange have been hand-juiced to make this delicious sweet-and-sour extract." -concentration 1
# static Flavoring milk	create Beverage class with -name “Milk” -description "Milk brought in from specific supply channels. The mouthfeel is incredibly smooth, and the aftertaste is sweet." -concentration 1

# static Flavoring cocoa	create Beverage class with -name “Cocoa Paste” -description "Processed from the cocoa fruit. Its unique aroma can be used to add a rich flavor to the drink." -concentration 1
# static Flavoring caramel	create Beverage class with -name “Caramel” -description "This boiled syrup has a thick texture and rich sweetness." -concentration 1
# static Flavoring lemon	create Beverage class with -name “Lemon” -description "This vibrantly colored fruit has an intense sourness that can produce some unexpected results." -concentration 1
# static Flavoring mint	create Beverage class with -name “Mint” -description "Freshly picked leaves that can be used to add a cooling taste." -concentration 1
# static Flavoring soda	create Beverage class with -name “Soda” -description "A liquid that bubbles over. It has no taste, but its mouthfeel packs a punch" -concentration 1
# static ArrayList<Ingredient>	create an ArrayList of the given fields

baseingredient	coffee,tea,juice,milk,mint cocoa,caramel,lemon, soda
# static ArrayList<Recipe> allrecipes	an ArrayList of recipe

2.2.2 Method

Name	Description
+ static String getIngredient(String name)	find the ingredient in baseingrediet that have the same name as the given String name -return that ingredient description
+ static createRecipes()	return nothing
+ getter & setter of allrecipes	

2.3 class EasyRecipe extends RecipeStorage

This class contains the recipes of easy mode customer order.

2.3.1 Field

Name	Description
- static ArrayList<Ingredient> atn	ArrayList of coffee that setCondense to 3
- static ArrayList<Ingredient>	ArrayList of

cp	coffee,milk,caramel
- static ArrayList<Ingredient> fr	ArrayList of coffee setCondense to 2 and soda
- static ArrayList<Ingredient> ge	ArrayList of coffee setCondense to 2 and milk
- static ArrayList<Ingredient> ma	ArrayList of coffee,milk,cocoa
- static ArrayList<Ingredient> nss	ArrayList of coffee and milk setCondense to 2
- static ArrayList<Ingredient> sn	ArrayList of coffee,tea,milk
- static ArrayList<Ingredient> bb	ArrayList of juice,soda,mint
- static ArrayList<Ingredient> bs	ArrayList of juice setCondense to 2 and lemon
- static ArrayList<Ingredient> dd	ArrayList of juice setCondense to 2 and soda
- static ArrayList<Ingredient> gvs	ArrayList of juice setCondense to 3
- static ArrayList<Ingredient> sck	ArrayList of juice setCondense to 2 and milk
- static ArrayList<Ingredient> scl	ArrayList of juice and milk setCondense to 2
- static ArrayList<Ingredient> bw	ArrayList of tea,milk,mint
- static ArrayList<Ingredient> b	ArrayList of tea and milk setCondense to 2
- static ArrayList<Ingredient> d	ArrayList of tea and juice setCondense to 2

- static ArrayList<Ingredient> lc	ArrayList of tea,milk,cocoa
- static ArrayList<Ingredient> lp	ArrayList of tea,milk,caramel
- static ArrayList<Ingredient> mg	ArrayList of tea setCondense to 3

- static ArrayList<Ingredient> sa	ArrayList of tea setCondense to 2 and milk
- static ArrayList<Ingredient> tb	ArrayList of tea setCondense to 2 and lemon
- static String atn_des	set to Athenaeum
- static String cp_des	set to Caramel Pinecone
- static String fr_des	set to Foamy Reef
- static String gedes	set to Golden Eden
- static String ma_des	set to Moonlit Alley
- static String nss_des	set to Night of Swirling Stars
- static String sn_des	set to Stroke of Night
- static String bbn_des	set to Barbatos' Boon
- static String bs_des	set to Birch Sap
- static String dd_des	set to Dawning Dew
- static String gvs_des	set to Gray Valley Sunset
- static String sck_des	set to Snow-Covered Kiss
- static String scl_des	set to Sweet Cider Lake
- static String bw_des	set to Boreal Watch
- static String b_des	set to Brightcrown

- static String d_des	set to Dusk
- static String lc_des	set to Laughter and Cheer
- static String lp_des	set to Love Poem

- static String mg_des	set to Misty Garden
- static String sa_des	set to Scholar's Afternoon
- static String tb_des	set to Tart Brilliance

2.3.2 Method

Name	Description
+ static void createRecipes()	-create Recipe by using pair of ArrayList and String in the field with the same variable name -create ArrayList of all Recipes and set that ArrayList by using setAllrecipes() of super class *(such as Recipe atn_reps = new Recipe(atn_des,atn);)

2.4 **class HardRecipe** extends RecipeStorage

This class contains the recipes of hard mode customer order.

2.4.1 Field

Name	Description
- static String h1_des	set to "I want Brightcrown with more cooling and sour taste";
- static String h2_des	set to "I want Dusk with more fizzy"
- static String h3_des	set to "I want Athenaeum with much more sweetness"
- static String h4_des	set to "I want Caramel Pinecone with more cooling and more sweet"
- static String h5_des	set to "I want Foamy Reef with more sour and sweetness"
- static String h6_des	set to "I want Golden Eden with more smooth taste"
- static String h7_des	set to "I want Moonlit Alley with much more unique aroma"
- static String h8_des	set to "I want Night of Swirling Stars with more cooling and smooth taste"
- static String h9_des	set to "I want Stroke of Night with much more rich flavor"
- static String h10_des	set to "I want Birch Sap with more rich fragrance"

- static String h11_des	set to "I want Barbatos' Boon with more sour taste"
- static String h12_des	set to "Give me the menu that have tea and sour taste"
- static String h13_des	set to "Give me the menu that have Juice that's fizzy"
- static String h14_des	set to "Give me the menu that have tea with smooth taste"
- static String h15_des	set to "Give me the menu that have juice with much more smooth taste"
- static ArrayList<Ingredient> h1	ArrayList of milk setCondense to 2 and millk,tea,lemon
- static ArrayList<Ingredient> h2	ArrayList of juice set condense to 2 and tea,soda
- static ArrayList<Ingredient> h3	ArrayList of coffee setCondense to 3 and caramel set Condense to 2
- static ArrayList<Ingredient> h4	ArrayList of caramel setCondense to 2 and mint ,coffee,milk

- static ArrayList<Ingredient> h5	ArrayList of coffee setCondense to 2 and soda,lemon,caramel
- static ArrayList<Ingredient> h6	ArrayList of coffee setCondense to 2 and milk setCondense to 2
- static ArrayList<Ingredient> h7	ArrayList of coffee,milk,cocoa setCondense to 3
- static ArrayList<Ingredient> h8	ArrayList of milk setCondense to 3 and coffee,mint
- static ArrayList<Ingredient>	ArrayList of coffee,tea,milk,cocoa

h9	setCondense to 2
- static ArrayList<Ingredient> h10	ArrayList of juice setCondense to 2 and lemon,tea
- static ArrayList<Ingredient> h11	ArrayList of juice,soda,mint,lemon
- static ArrayList<Ingredient> h12	ArrayList of tea set Condense to 2 and lemon
- static ArrayList<Ingredient> h13	ArrayList of juice setCondense to 2 and soda
- static ArrayList<Ingredient> h14	ArrayList of tea setCondense to 2 and milk
- static ArrayList<Ingredient> h15	ArrayList of milk setCondense to 3 and juice

2.4.2 Method

Name	Description
+ static void createRecipes()	-create Recipe by using pair of ArrayList and String in the field with the same variable name -create ArrayList of all Recipes and set that ArrayList by using setAllrecipes() of super class *(such as Recipe h1_reps = new Recipe(h1_des,h1);)

3. Package container

3.1 Interface Addable

This class use to define what class is addable

3.1.1 Method

Name	Description
+ boolean isAddable()	check if that class is addable

3.2 class Glass implements Addable

This class present glass to fill in with the Ingredient

3.2.1 Field

Name	Description
- int MaxIngredient	MaxIngredient that the glass can add. -set to 5
- int Capacity	present capacity of the glass
- ArrayList<Ingredient> drink	ArrayList of Ingredient
- Size size	size of the glass -set size to Size.EMPTY

3.2.2 Constructor

Name	Description
+ Glass()	-Initialize the ArrayList drink -set drink -set capacity to 0

3.2.3 Method

Name	Description
+ void addIngredient(Ingredient ingredient)	-if capacity is less than MaxIngredient. check if there are ingredient in glass that same as added ingredient. if it has the same ingredient set the concentration of that ingredient plus 1 if it doesn't have the same ingredient add that ingredient into ArrayList drink -set new drink -set new capacity
+ void clearGlass()	-Initialize new ArrayList drink -set it capacity to 0
+ boolean isServable()	check if the drink is servable

	by -checking if there are Ingredient instanceof Edible in ArrayList drink
+ isAddable()	-return if the capacity is not equals to MaxIngredient
+ getter & setter of each field	

3.3 enum Size

This enum present size of the glass

3.3.1 enum

EMPTY,SMALL,MEDUIM,LARGE

4. Package exception

4.1 class ServeFailedException extends Exception

4.1.1 Fields

Name	Description
- String message	message

4.1.2 Constructor

Name	Description
+ ServeFailedException (String message)	initialize fields

4.1.3 Method

Name	Description
+ void printErrormessage()	print error message
+ setter of each field	

5. Package GUI

5.1 class MyApplication extends Application

5.1.1 Field

Name	Description
- static AudioClip bgm	AudioClip of main BGM

5.1.2 Method

Name	Description
+ void start(Stage primaryStage) throws Exception	Start game with primaryStage

+ static void main(String [] args)	

5.2 class MyController implements Initializable

5.2.1 Fields

Name	Description
- Stage stage	game page stage
- Scene scene	game page scene
- Parent root	root of game page
- AudioClip buttonclick	AudioClip of button click
- AudioClip pick	AudioClip of pick
- AudioClip pouring	AudioClip of pouring
- Button coffee,tea,juice,milk,mint,lemon,soda,cocoa,caramel,StartButton,NewGameButton,serveButton,clearButton,small,medium,large	buttons in game page
- Text timerText,orderText,coffeeNumberText,teaNumberText,juiceNumberText,milkNumberText,mintNumberText,lemonNumberText,sodaNumberText,cocoaNumberText,caramelNumberText	texts in game page
- ImageView coffeeimg,teaimg,juiceimg,sodaimg,lemonimg,milkimg,cocoaimg,caramelimg,mintim	images in game page

g	
- AnchorPane anchorPane	anchor pane
+ Timer timer	timer for counting down
+ static volatile boolean exit	to check if time runs out

5.2.2 Method

Name	Description
+ void switchToRecipe (ActionEvent event) throws IOException	Start game with primaryStage
+ void switchToScoreScene(boolean end) throws IOException	switch to score scene
+ void start(ActionEvent e)	start the game
+ void clear(ActionEvent e)	clear picked ingredients and glass size
+ void serve(ActionEvent e)	serve the order
+ void newGame(ActionEvent e)	restart a new game
+ void fillSize(ActionEvent e)	set the size of glass
+ void add(ActionEvent e)	add ingredients
+ void gameEnd()	end the game
+ void startCountdown()	start count down timer
+ void initialize(URL arg0, ResourceBundle arg1)	

5.3 class FrontPageController implements Initializable

5.3.1 Fields

Name	Description
- AnchorPane anchorPane	anchor pane
- Button easyButton,hardButton	buttons in front page
- Stage stage	front page stage
- Scene scene	front page scene
- Parent root	root of front page
- AudioClip buttonclick	AudioClip of button click

5.3.2 Method

Name	Description
+ void switchToScene1 (ActionEvent event) throws IOException	switch to game page
+ void initialize(URL arg0, ResourceBundle arg1)	

5.4 class RecipeController implements Initializable

5.4.1 Fields

Name	Description

- AnchorPane anchorPane	anchor pane

5.4.2 Method

Name	Description
+ void initialize(URL arg0, ResourceBundle arg1)	

5.5 class ScoreController implements Initializable

5.5.1 Fields

Name	Description
- Stage stage	score page stage
- Button newGameButton,QuitButton	butttons in score page
- ImageView background	background image
- AudioClip buttonclick	AudioClip of button click
- Text grade,score	texts in score page

5.5.2 Method

Name	Description
+ void switchToFrontPage (ActionEvent event) throws	switch to front page

IOException	
+ void exit()	exit the window
+ void initialize(URL arg0, ResourceBundle arg1)	

6. Package logic

6.1 class Customer

6.1.1 Field

Name	Description
- String description	description of customer
- ArrayList<Ingredient> drink	array list of drink
- Size size	size
- String recipes	recipes

6.1.2 Constructor

Name	Description
+ Customer(String description,ArrayList<Ingredient> drink,Size size)	initialize fields
+ Customer(String description,ArrayList<Ingredient> drink,Size size,String recipes)	initialize fields

6.1.3 Method

Name	Description
+ getter & setter of each field	

6.2 class DrinkValidator

6.2.1 Method

Name	Description
+ static boolean checkDrink(Customer customer,Glass glass) throws ServeFailedException	return true if the glass contains al least 1 beverage and not an empty class and glass is addable if glass size empty but servable throws ServeFailedException("Fill your glass size!!") if glass not empty but not servable throws ServeFailedException("Add at least 1 beverage!!") if both thorws ServeFailedException("Add at least 1 beverage and Fill

	your glass size!!")

6.3 class GameLogic

This class present all logic of the game

6.3.1 Field

Name	Description
- static Timer playerTimer	timer of player
- static int playerScore	player score
- static int MinusScore	minus score when add wrong ingredient
- static int trytimes	number of times that can try
- static int orderrunner	use to run index of customer
- static Mode mode	mode of the game
- static Glass glass	empty glass in the game
- static Customer presentcustomer	represent present customer order
- static ArrayList<Customer> CustomerList	represetn list of customer
- ArrayList<String> ErrorText	list of encourage text alert when added wrong ingredient
- String errorText	error text alert when serve failed

6.3.2 Method

Name	Description
+ static void startGame()	-set playerScore to 0 -set MinusScore to 0 -set trytimes to 3 -set orderrunner to 0 -create new class glass -call function startOrder()
+ static void startRunOrder()	check what mode is -createRecipes by mode -generateCustomerList and set CustomerList to that list -set presentcustomer to CustomerList index 0
+ static void selectMode(Mode choosenMode)	set mode to choosenMode
+ static void callNextCustomer()	-set trytimes to 3 -set orderrunner plus 1 -set present customer to CustomerList is orderrunner index
+ static void choosenMode(Mode choosenMode)	set mode to choosenmode
+ static int getScore()	set playerScore
+ static boolean isServable(Glass glass,Cutomer customer)	check if the glass is servable by checking the drink

	-if it’s correct plus player score 500 - MinusScore -set MinusScore to 0 -call function callNextCustomer -if it’s not correct check what the error is. -if trytimes is 0 set alert to Wrong Ingredient -set errortext to "Customer get upset. Why you so noob man." -set MinusScore to 0 -call function callNextCustomer() if trytimes > 0 -set errortext random of ArrayList ErrorText -set Minus plus 100
+ static Ingredient createIngredientFromName(String s)	-create Ingredient from given String name -return class that same name as given String
+ static void clearGlass()	- create new glass class
+ static Glass getGlass()	-return glass
+ static Customer getPresentcustomer()	- return presentCustomer
+ static int	-return the concentration of

getConcentrationWitdId(String s)	the Ingredient in glass that has same name as the given String
+ static void addIngredientToGlass(String name)	-create Ingredient from name and add to glass
+ static void setglassSize(Size size)	-set size of the glass
+ static String getCustomerOrder()	-return String in form presentcustomer.getDescription()+" , Size "+presentcustomer.getSize()
+ static void setZeroScore()	-set playerScore to 0

6.4 class Grade

6.4.1 Method

Name	Description
+ static String grading(int score)	return grade of score in String
+ static int gradingInt(int score)	return grade of score in int

6.5 class ListOfCustomer

6.5.1 Field

Name	Description
- static String rand_des	random description
- static ArrayList<Ingredient> rand_rep	random recipes
- static ArrayList<Customer> customerList	customerlist or random decription and random recipes
- static Random rand	call function Random
- static int randnum	random index
- static int number_of_customer	number of customer

6.5.2 Method

Name	Description
+ static void generateCustomerList()	to generate customer list
+ static Size randomSize()	return random size
+ getter & setter of each field	

6.6 class Timer

6.6.1 Field

Name	Description
- static int minute	minute in timer
- static int seconds	second in timer
- boolean isStop	check if timer is stop

6.6.2 Constructor

Name	Description
+ Timer(int m, int s)	initialize fields

6.6.3 Method

Name	Description
+ void decrementTimer(int amount)	decrease second of timer
+ boolean isTimerEmpty()	check if timer is empty
+ String toString()	return string of timer
+ getter & setter of each field	

6.7 enum Mode

This class present the mode of the game

6.7.1 enum

HARD,EASY;