System and Software Architecture Description (SSAD)

FlowerSeeker

Team 04

Celia Chen, Project Manager
Jason Tan, Life Cycle Planner
Xiaoran Huang, Feasibility Analyst
Chenghao Yang, Operational Concept Engineer
Ruiwen Tang, Requirements Engineer
Xian Li, System/Software Architect
Clifford Rhyne, IV&V

4/25/15

Version History

Date	Author	Version	Changes made	Rationale
10/12/14	Celia Chen, Li	1.0	Updated section 1 and 2 with use cases and diagrams produced by Visual Paradigm	Draft FC Package
10/19/14	CC	2.0	Updated artifacts based on the current backend Minor adjustment on use cases	FC Package
11/20/14	CC	3.0	Updated section 3, 4, and 5. Minor adjustments on use cases	Draft DC Package
12/03/14	CC	3.1	Minor adjustments on sequence diagrams, classes.	DC Package
12/07/14	CC	4.0	Minor adjustments based on the reviews from ARB	DC Package
2/10/15	CC	4.1	Minor adjustments based on the reviews from ARB	Draft Rebaseline
4/25/15	CC, JT	5.0	Update database design	ASBUILT Package

Table of Contents

Table of Tables

Table of Figures

1. Introduction

1.1 Purpose of the SSAD

The purpose of this SSAD is to record the results of the use case analysis and design of FlowerSeeker system. The programmers and developers of our team can use this SSAD as reference to the architecture of the system that we are developing. During the development of this system, our team will be faithful to the structure of the system described in SSAD.

1.2 Status of the SSAD

This version contains the first draft of the system's architecture's diagram and descriptions. The process of designing architecture is based on the current agreed win conditions.

Version 2.0 contains the updated system architecture diagram based on the previous meeting with the client on adjustment of the scope of the project. Since the whole team will be involved in developing the backend as long as the frontend, the artifacts and information diagram has been modified based on the current system.

Version 3.0 contains all the information needed for all sections, including class diagrams and tables and sequence diagrams.

Version 3.1 contains all updated information.

2. System Analysis

2.1 System Analysis Overview

The purpose of our system Flower Seeker is to provide an online marketplace for both customers and florists. On Flower Seeker, customers can search for flowers based on style, price, florist rating, occasion and so on and communicate with local florists directly. On the other hand, it also provides a marketplace for florists to easily upload the product details of their flowers and sell them online. Customers can also customized their bouquet bundles and make their own designs through communicating with florists. Flower Seeker let the customers and florists get an opportunity and provide a platform to communicate with each other directly and build their business relationship based on that.

2.1.1. System Context

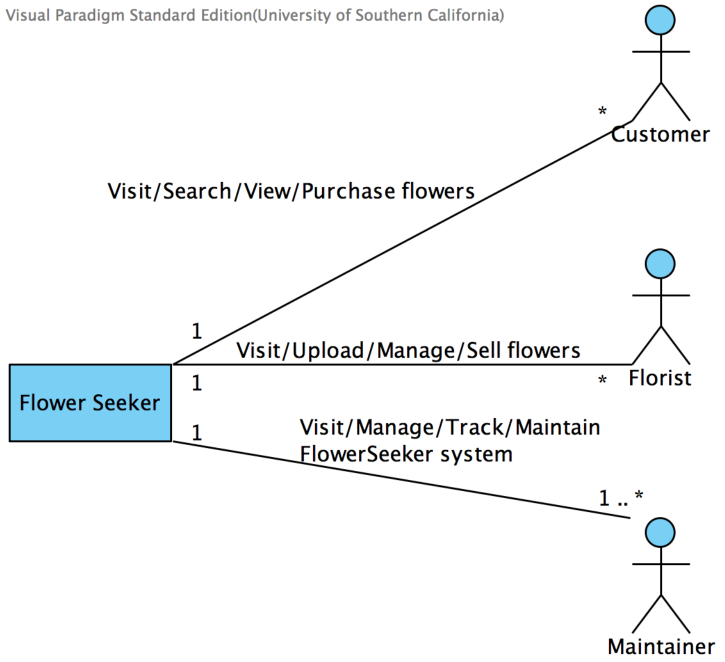
Visual Paradigm Standard Edition(University of Southern California)
Customer
*
Visit/Search/View/Purchase flowers
1
Visit/Upload/Manage/Sell flowers
Florist
*
1
Flower Seeker
1
Visit/Manage/Track/Maintain FlowerSeeker system
1 .. *
Maintainer

Figure 1: System Context Diagram

Actor	Description	Responsibilities
Florist	Florist is a user who registers as a flower shop owner on FlowerSeeker and sells flowers through FlowerSeeker.	Create and manage profile Upload and manage flower catalog Upload information about orders (Making the bouquet, out for delivery, etc.)
Customer	Customer is a user who registers as a customer on FlowerSeeker and purchases flowers through FlowerSeeker from florists.	Create and manage profile Provide reviews to the purchased product (Optional) Place orders Manage/Track orders
Maintainer	Maintainer is a managerial staff of the website who manages and tracks all user activities.	Monitor all user activities on Flower Seeker website

Table 1: Actors Summary

2.1.2 Artifacts & Information

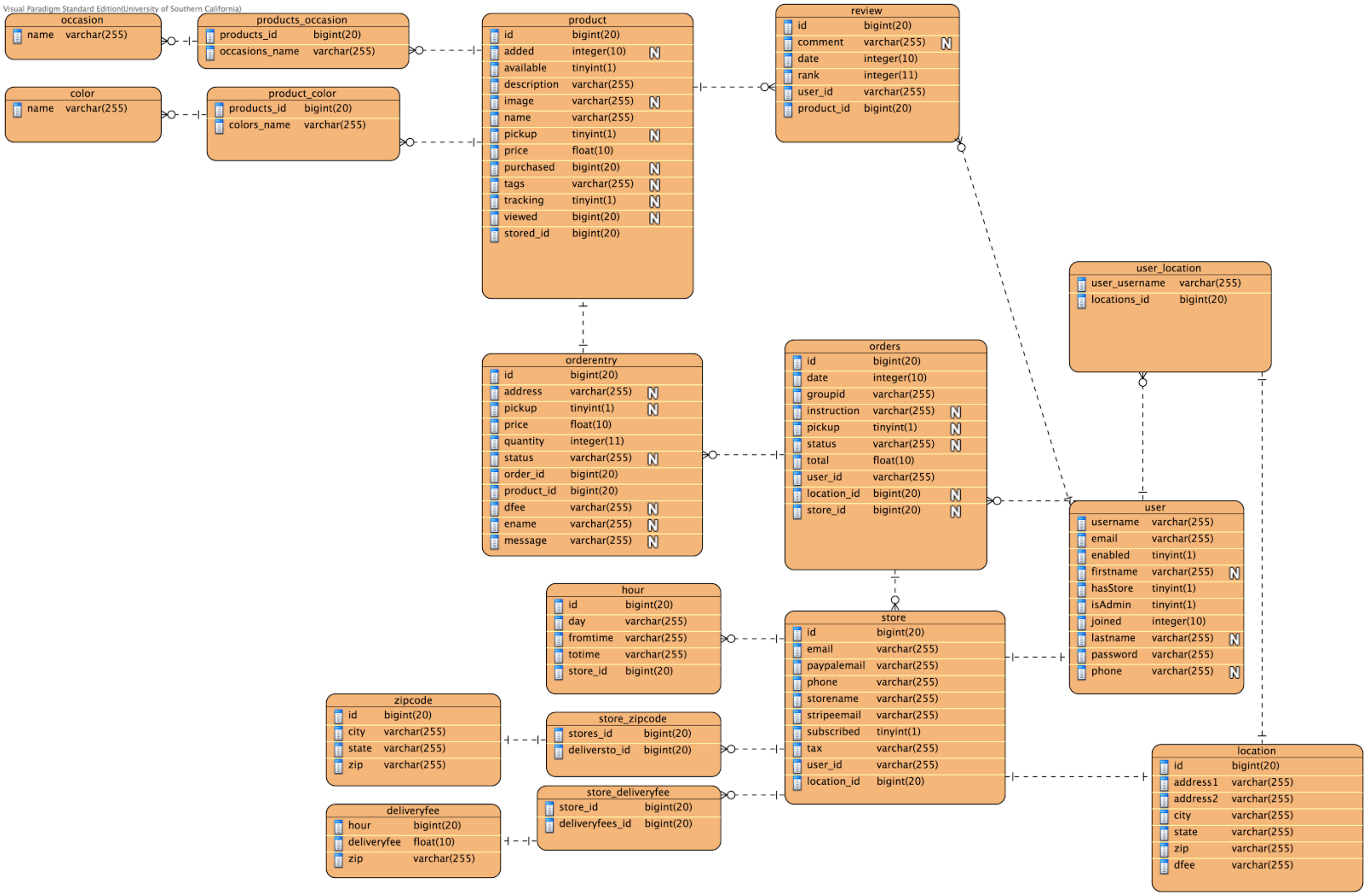

Figure 2: Artifacts and Information Diagram

Artifact	Purpose
Product	Stores product information, including id, added, available, description, image, name, pickup, price, purchased, tags, tracking, viewed, store_id
Products Occasion	Stores product occasions including products_id, occasions_name
Occasion	Stores occasion name
Product color	Stores product color information, including product_id, color_name

Color	Stores color name
Review	Stores review information including id, comment, date, rank, user_id, product_id
Orderentry	Stores orderentry information including id, address, pickup, price, quantity, status, order_id, product_id, dfee, ename, message
Orders	Stores orders information including id, date, groupid, instruction, pickup, status, total, user_id, location_id, store_id
User Location	Stores user_location information including user_username, locations_id
Hour	Stores hour information including id, day, fromtime, totime, store_id
Store	Stores store information including id, email, paypalemail, phone, storename, stripeemail, subscribed, tax, user_id, location_id
User	Stores user information including username, email, enabled, firstname, hasStore, isAdmin, joined, lastname, password, phone
Location	Stores location information including id, address1, adress2, city, state, zip, dfee
Zipcode	Stores zipcode information including id, city, state, zip
Store zipcode	Stores zipcode information including stores_id and deliversto_id
Delivery Fee	Stores Store Delivery Fee information including hour, deliveryfee, zip
Store Delivery Fee	Stores Delivery Fee information including store_id, deliveryfees_id

Table 2: Artifacts and Information Summary

2.1.3 Behavior

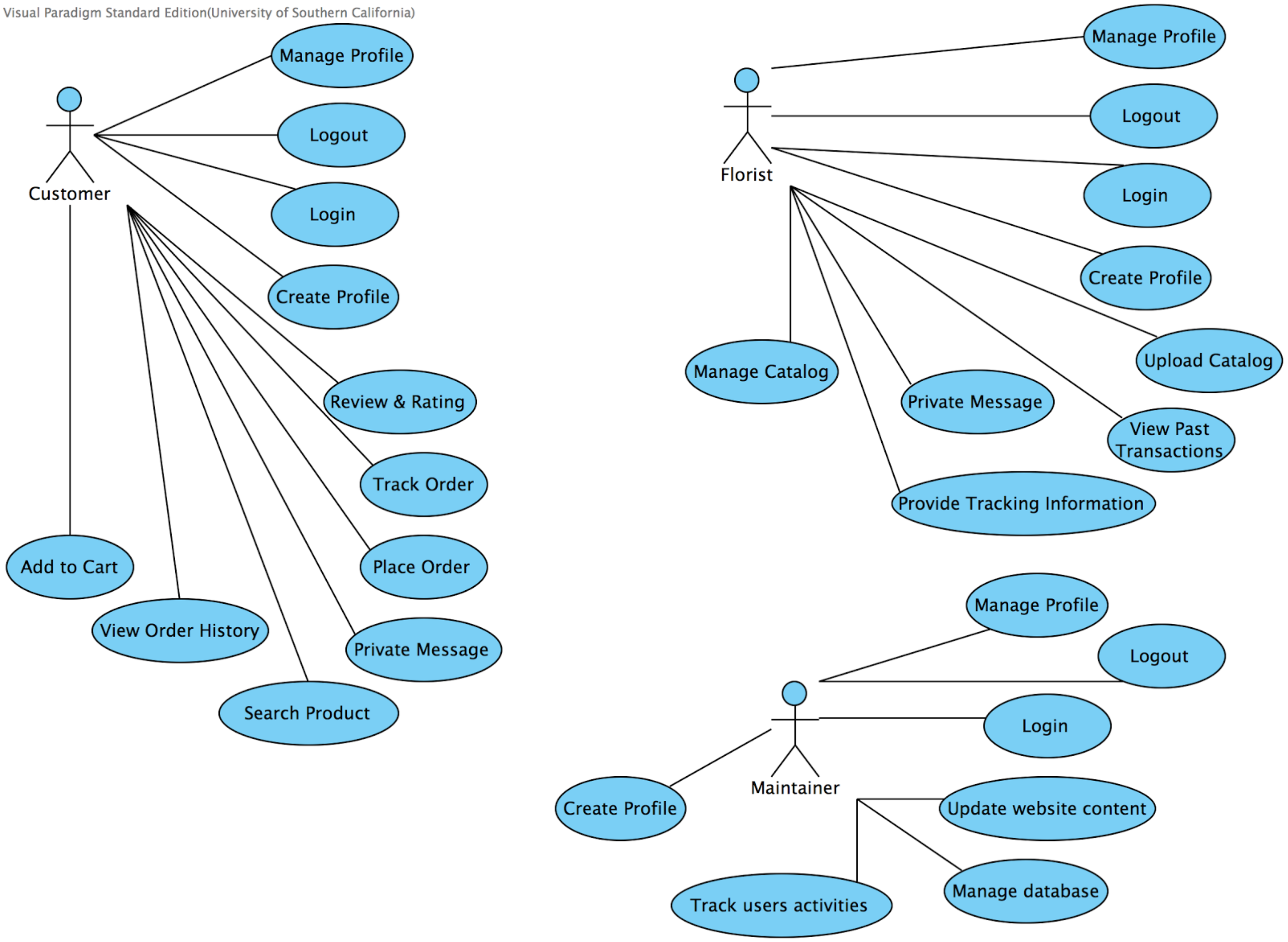

Figure 3: Process Diagram

2.1.3.1 Capability Login

2.1.3.1.1 Process Login

Process Description: Login

Identifier	UC-1 Login
Purpose	To check to see whether a user has been logged into the FlowerSeeker system and if the system can authenticate the user. If yes, the system redirects the user to the correct page based on the role (florist, customer, or maintainer).
Requirements	None
Development Risks	None
Pre-conditions	The user is on the login page.

Post-conditions	If the user is authorized, the system will redirect the user to the corresponding page according to the role. Otherwise, the system will display an error message.

Typical Course of Action: Login Successfully

Seq#	Actor's Action	System's Response
1	Go to Login Page	
2	Enter Username and Password, click the Login button	
3		Pre-check to see if the username and password are empty
4		Send username and password to the authentication checking back-end system
5		Authorize user and redirect the user to the corresponding page according to the role.

Alternate Course of Action: Empty username and/or password

Seq#	Actor's Action	System's Response
1	Go to Login Page	
2	Click the Login button with either empty username and/or password	
3		Pre-check to see if the username and password are empty
4		Display an error message on the current Login page

Alternate Course of Action: Login Failed

Seq#	Actor's Action	System's Response
1	Go to Login Page	
2	Click the Login button with either empty username and/or password	
3		Pre-check to see if the username and password are empty
4		Send username and password to the authentication checking back-end system
5		Display an error message on the current Login page

2.1.3.1.2 Process Logout

Process Description: Logout

Identifier	UC-2 Logout
Purpose	To log the current user out of the system and close the session
Requirements	None
Development Risks	None
Pre-conditions	The user is logged in to the system.
Post-conditions	The session is closed, username and password fields are cleared.

Typical Course of Action: Logout Successfully

Seq#	Actor's Action	System's Response
1	Click on Logout button	
2		Close the current session, display a successful message.

2.1.3.2 Capability Profile Management

2.1.3.2.1 Process Create Profile

Process Description: Create Profile

Identifier	UC-3 Create Profile
Purpose	To allow user create profile on FlowerSeeker according to their roles (florist, customer, and maintainer).
Requirements	WC_3471, WC_3470
Development Risks	None
Pre-conditions	The user is on the create profile page.
Post-conditions	The system stores information to the database and redirects the user to his/her corresponding user page.

Typical Course of Action: Create Successfully

Seq#	Actor's Action	System's Response
1	Go to Create Profile page	
2	Fill out required information and click Create button	
3		Pre-check if any of the required fields are empty
4		Send all information to database and create an entry for this user.

5		The system displays a successful message and redirects user to his/her corresponding user page

Alternate Course of Action: Empty required information fields

Seq#	Actor's Action	System's Response
1	Go to Create Profile page	
2	Fill out required information and click Create button	
3		Pre-check if any of the required fields are empty
4		Display an error message on the current Create Profile page

Alternate Course of Action: Duplicated email address/username

Seq#	Actor's Action	System's Response
1	Go to Create Profile page	
2	Fill out required information and click Create button	
3		Pre-check if any of the required fields are empty
4		Send all information to database and check for existing entry with the given email address or username
		Display an error message on the current Create Profile page

2.1.3.2.2 Process Manage Profile

Process Description: Manage Profile

Identifier	UC-4 Manage Profile
Purpose	To allow user manage profile on FlowerSeeker according to their roles (florist, customer, and maintainer).
Requirements	None
Development Risks	None
Pre-conditions	The user is logged in to the system and on the profile page.
Post-conditions	The system updates information about this user on the database and redirects the user to his/her corresponding user page.

Typical Course of Action: Update Successfully

Seq#	Actor's Action	System's Response

1	Click on Edit Profile button	
2	Fill out required information and click submit button	
3		Pre-check if any of the required fields are empty
4		Send all information to database and update the entry for the user.
5		The system displays a successful message and updates the current user page

Alternate Course of Action: Empty required information fields

Seq#	Actor's Action	System's Response
1	Click on Edit Profile button	
2	Fill out required information and click submit button	
3		Pre-check if any of the required fields are empty
4		Display an error message on the current Profile page

2.1.3.3 Capability Searching Function

2.1.3.3.1 Process Search Product

Process Description: Search Product

Identifier	UC-5 Search Product
Purpose	To allow customers to search flowers according to the needs.
Requirements	WC_3341, WC_3338
Development Risks	None
Pre-conditions	The customer is logged in to the system and on the product page.
Post-conditions	The system retrieves information from the database based on the keywords and updates product page.

Typical Course of Action: Search Successfully

Seq#	Actor's Action	System's Response
1	Go to product page	
2	Select desired price range, occasion, color, type, or location information and click Search button	

3		Send all information to database and retrieve products that match those conditions.
4		Display all matched products on a product page.

Alternate Course of Action: No Match Product found

Seq#	Actor's Action	System's Response
1	Go to Product page	
2	Select desired price range, occasion, color, type, or location information and click Search button	
3		Send all information to database and retrieve products that match those conditions.
4		Display a message that no match product found.

2.1.3.4 Capability Order Placement

2.1.3.4.1 Process Add to Cart

Process Description: Add to Cart

Identifier	UC-6 Add to Cart
Purpose	To allow customer to add products to shopping cart.
Requirements	None
Development Risks	None
Pre-conditions	The customer is logged in to the system and on the product page.
Post-conditions	Product is added to the shopping cart and system updates the number of products in the shopping cart on the current page.

Typical Course of Action: Add to Cart Successfully

Seq#	Actor's Action	System's Response
1	Drag product to shopping cart	
2		Check database to see if it's available.
3		Upload product information into cache and update the number of products in the shopping cart on the current page

Alternate Course of Action: Product Out of Stock

Seq#	Actor's Action	System's Response
1	Drag product to shopping cart	
2		Check database to see if it's available.
3		Display an error message on the current page saying the product is out of stock.

2.1.3.4.2 Process Place Order

Process Description: Place Order

Identifier	UC-7 Place Order
Purpose	To allow customer place order.
Requirements	WC_3352, WC_3343, WC_3337
Development Risks	None
Pre-conditions	The customer is logged in to the system, successfully added product to the cart and is currently on the checkout page.
Post-conditions	The system displays "Order is placed" message.

Typical Course of Action: Order Successfully Placed

Seq#	Actor's Action	System's Response
1	Fill in shipping information, payment information and click Place Order button	
2		Pre-check to see if any of the required information is empty
3		Send the payment information to payment system to verify
4		Approve transaction. Display a successful message and redirect to the order information page.

Alternate Course of Action: Information Missing

Seq#	Actor's Action	System's Response
1	Fill in shipping information, payment information and click Place Order button	
2		Pre-check to see if any of the required information is empty
3		Display an error message on the current page saying required fields are missing.

Alternate Course of Action: Place Order Failed

Seq#	Actor's Action	System's Response
1	Fill in shipping information, payment information and click Place Order button	
2		Pre-check to see if any of the required information is empty
3		Send the payment information to payment system to verify
4		Disapprove the transaction. Display an error message and redirect to payment information page.

2.1.3.5 Capability Order History

2.1.3.5.1 Process View Order History

Process Description: View Order History

Identifier	UC-8 View Order History
Purpose	To allow customer view order history.
Requirements	WC_3359
Development Risks	None
Pre-conditions	The customer is logged in to the system and is currently on the profile page.
Post-conditions	The system displays all the order history on the order history page of the user.

Typical Course of Action: Successfully show order history

Seq#	Actor's Action	System's Response
1	Click on the order history button	
2		The system retrieves all the order this user made and displays the result on the page.

Alternate Course of Action: No Order Found

Seq#	Actor's Action	System's Response
1	Click on the order history button	

2		The system retrieves all the order this user made and found no order accociated to the user.
3		Display a message on the current page saying no order found under this user.

2.1.3.5.2 Process View past transactions

Process Description: View Past Transactions

Identifier	UC-9 View Past Transactions
Purpose	To allow florist view past orders and transactions
Requirements	WC_3358
Development Risks	None
Pre-conditions	The florist is logged in to the system, and is currently on the order history page.
Post-conditions	The system displays all the past transactions.

Typical Course of Action: Successfully view past transactions

Seq#	Actor's Action	System's Response
1	Go to Order History page	
2		System retrieves all order history of the florist.

Alternate Course of Action: No orders

Seq#	Actor's Action	System's Response
1	Go to Order History page	
2		System retrieves no order associated with this florist.
3		Display a message saying no orders found.

2.1.3.6 Capability Order Tracking

2.1.3.6.1 Process Track Order

Process Description: Track Order

Identifier	UC-10 Track Order
Purpose	To allow customer track order.
Requirements	WC_3342
Development Risks	None

Pre-conditions	The customer is logged in to the system, an order has been placed and accepted by the florist, and is currently on the order page.
Post-conditions	The system displays the progress of the order.

Typical Course of Action: Successfully track order

Seq#	Actor's Action	System's Response
1	Click on the track order button	
2		The system retrieves the order status and displays on the current page.

2.1.3.6.1 Process Provide Tracking Information

Process Description: Provide Tracking Information

Identifier	UC-11 Provide Tracking Information
Purpose	To allow florist update status of order so that the customers can use those information to track their orders.
Requirements	WC_3342
Development Risks	None
Pre-conditions	The florist is logged in to the system, received and accepted an order and is currently on the order page.
Post-conditions	The system displays the updated order page.

Typical Course of Action: Successfully update order status

Seq#	Actor's Action	System's Response
1	Go to order page	
2	Update information about order status	
3		Send information to database and update the entry in the order table.
4		Display a successful message

2.1.3.7 Capability Review and Rating

2.1.3.7.1 Process Review and Rating

Process Description: Review and Rating

Identifier	UC-12 Review and Rating
Purpose	To allow customer view and also leave review and rate on the purchased product and florist.
Requirements	WC_3346, WC_3345
Development Risks	None

Pre-conditions	The customer is logged in to the system and is currently on any of the purchased product page.
Post-conditions	The system displays the review and rating on the product page.

Typical Course of Action: Successfully review and rate product

Seq#	Actor's Action	System's Response
1	Click on the review button	
2	Leave review and rating, and click on submit button	
3		Pre-check if the length of the review is more than 20 characters
4		Store the review and rating information into the database
5		Update the page with the review and rating information

Alternate Course of Action: Review is too short

Seq#	Actor's Action	System's Response
1	Click on the review button	
2	Leave review and rating, and click on submit button	
3		Pre-check if the length of the review is more than 20 characters
4		Display an error message saying the review is too short

2.1.3.8 Capability Catalog Management

2.1.3.8.1 Upload Catalog

Process Description: Upload Catalog

Identifier	UC-13 Upload Catalog
Purpose	To allow florist upload flower catalog.
Requirements	WC_3339
Development Risks	None
Pre-conditions	The florist is logged in to the system and is currently on the upload catalog page.
Post-conditions	The system displays the uploaded catalog page.

Typical Course of Action: Successfully uploaded catalog

Seq#	Actor's Action	System's Response

1	Go to the upload page	
2	Fill in the information of product and click the upload button	
3		Pre-check if any required information is missing and the size of the catalog is within limit
4		Send information to the database and display the product to the current page.

Alternate Course of Action: Missing Information

Seq#	Actor's Action	System's Response
1	Go to the upload page	
2	Fill in the information of product and click the upload button	
3		Pre-check if any required information is missing and the size of the catalog is within limit
4		Display an error message saying the required information is missing

Alternate Course of Action: Catalog exceed limit

Seq#	Actor's Action	System's Response
1	Go to the upload page	
2	Fill in the information of product and click the upload button	
3		Pre-check if any required information is missing and the size of the catalog is within limit
4		Display an error message saying the size of the catalog exceeded limit

2.1.3.8.2 Process Manage Catalog

Process Description: Manage Catalog

Identifier	UC-14 Manage Catalog
Purpose	To allow florist manage flower catalog.
Requirements	None
Development Risks	None

Pre-conditions	The florist is logged in to the system and is currently on the catalog page.
Post-conditions	The system displays the updated catalog page.

Typical Course of Action: Successfully manage catalog

Seq#	Actor's Action	System's Response
1	Go to any catalog page	
2	Update information of the catalog and click the upload button	
3		Pre-check if any required information is missing and the size of the catalog is within limit
4		Send information to the database and display the updated catalog page.

Alternate Course of Action: Missing Information

Seq#	Actor's Action	System's Response
1	Go to any catalog page	
2	Update information of the catalog and click the update button	
3		Pre-check if any required information is missing and the size of the catalog is within limit
4		Display an error message saying the required information is missing

Alternate Course of Action: Catalog exceed limit

Seq#	Actor's Action	System's Response
1	Go to any catalog page	
2	Update information of the catalog and click the update button	
3		Pre-check if any required information is missing and the size of the catalog is within limit
4		Display an error message saying the size of the catalog exceeded limit

2.1.3.9 Capability Private Message

2.1.3.9.1 Process Private Message (Customer)

Process Description: Private Message (Customer)

Identifier	UC-15 Private Message (Customer)
Purpose	To allow customer communicate with florist
Requirements	WC_3351
Development Risks	None
Pre-conditions	The customer is logged into the system and is currently on the florist profile page.
Post-conditions	The system displays a successfully contacted florist message.

Typical Course of Action: Successfully communicate with florist

Seq#	Actor's Action	System's Response
1	Go to florist profile page and click on Chat with Me button	
2	Leave any comment/concerns to the florist	
3		System delivers message to florist and display a successful message to customer.

2.1.3.9.2 Process Private Message (Florist)

Process Description: Private Message (Florist)

Identifier	UC-16 Private Message (Florist)
Purpose	To allow florist communicate with customer
Requirements	WC_3351
Development Risks	None
Pre-conditions	The florist is logged into the system and is currently on the dashboard page.
Post-conditions	The system displays a successfully contacted customer message.

Typical Course of Action: Successfully communicate with florist

Seq#	Actor's Action	System's Response
1	Go to florist dashboard page and open an unread message from customer	

2	Reply to the message and click send	
3		System delivers message to customer and display a successful message to florist.

2.1.4 Modes of Operation

The system will operate in only one mode, so there is no additional information concerning modes of operations provided.

2.2 System Analysis Rationale

The users of the system include florists, customers, and maintainers. These users require authentication in order to access the system with various permissions and functionalities based on the type of user.

3. Technology-Independent Model

3.1 Design Overview

3.1.1 System Structure

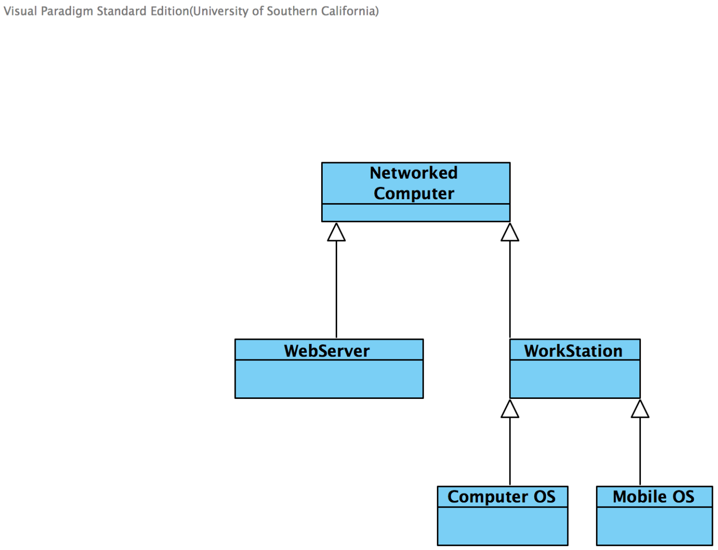
Visual Paradigm Standard Edition(University of Southern California)
Networked Computer
WebServer
WorkStation
Computer OS
Mobile OS

Figure 4: Hardware Component Class Diagram

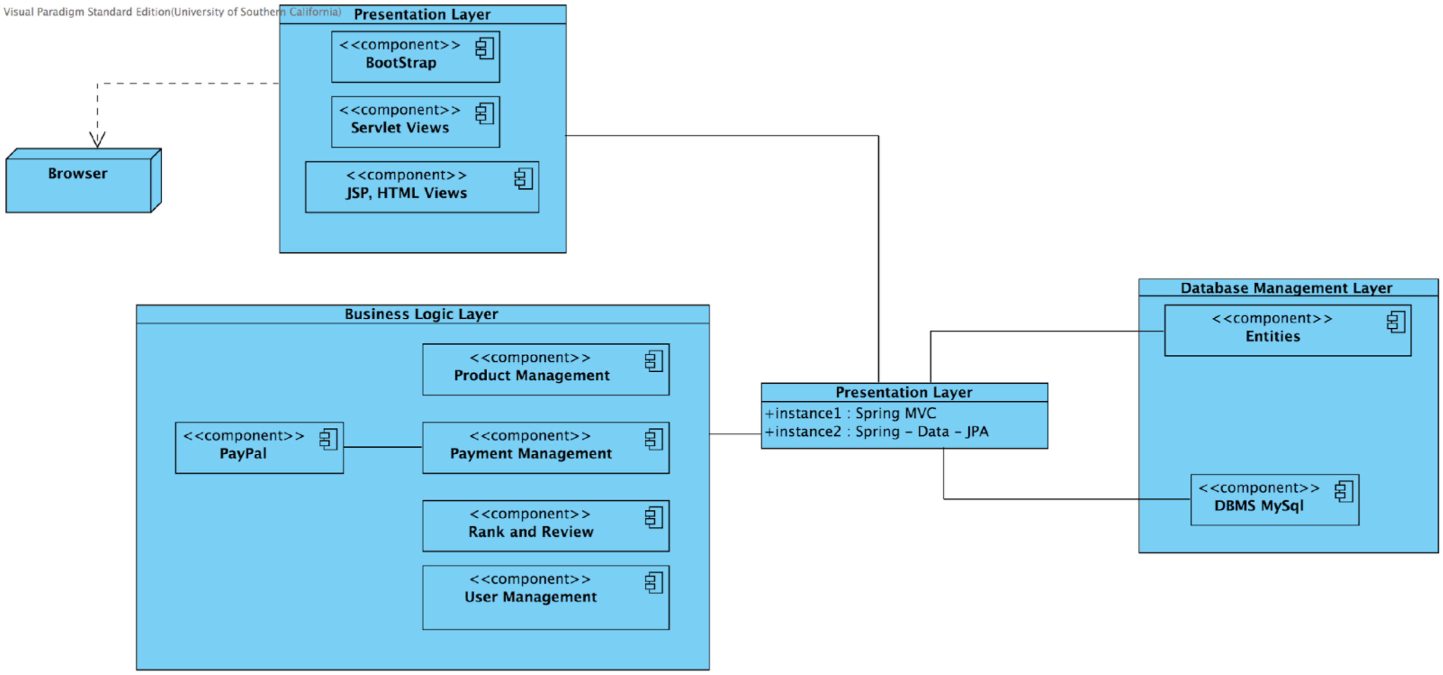

Figure 5: Software Component Class Diagram

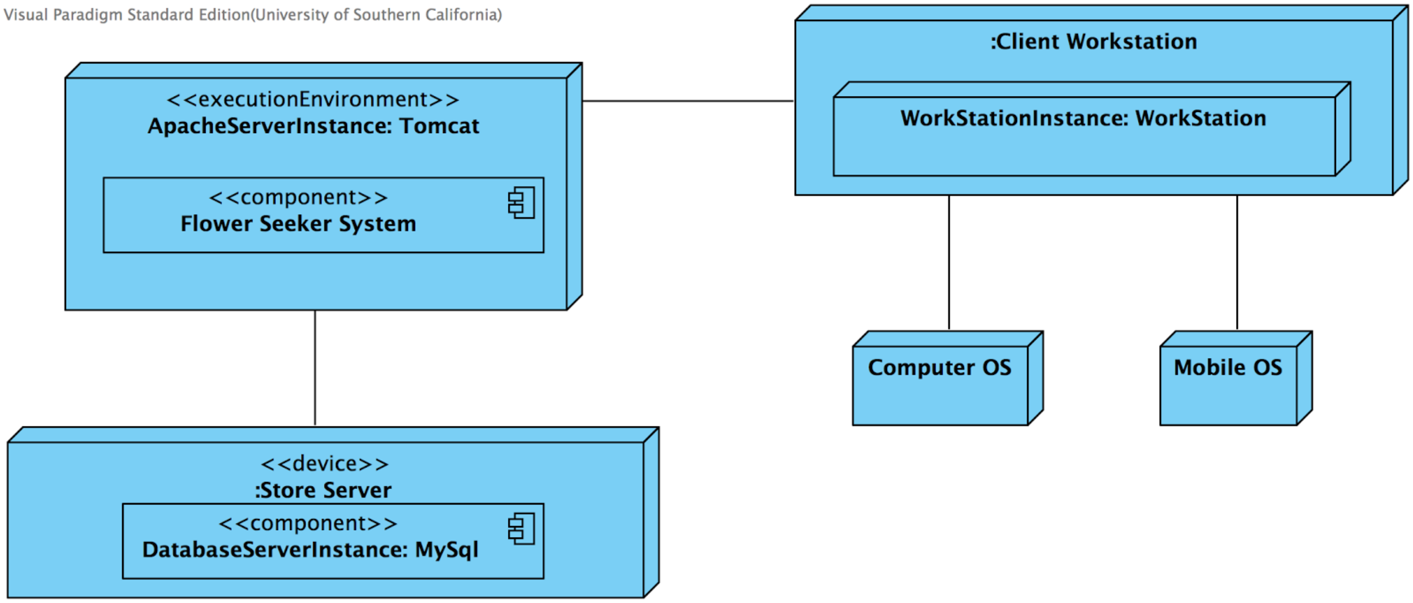
Visual Paradigm Standard Edition(University of Southern California)
<<executionEnvironment>>
ApacheServerInstance: Tomcat
<<component>>
Flower Seeker System
:Client Workstation
WorkStationInstance: WorkStation
Computer OS
Mobile OS
<<device>>
:Store Server
<<component>>
DatabaseServerInstance: MySql

Figure 6: Deployment Diagram

Hardware Component	Description
Web Server	FlowerSeeker's system deploys on a web server.
Work Station	FlowerSeeker's users' mobile devices/computers in which they use the system on a web browser

Computer OS	Users that use any computer to browse FlowerSeeker (Windows OS, Mac OS, Linux OS)
Mobile OS	Users that use any mobile phone to browse FlowerSeeker (Windows Phone, IPhone, Android)
Networked Computer	Networked computer

Table 3: Hardware Component Description

Software Component	Description
Presentation Layer	This layer consists of all the GUI of the system and serves to display the user interface, manage user interaction between users' actions and business logic layer. For example, when a customer types in words into the search box, this layer will pass the typed information to the business logic layer.
Business Logic Layer	This layer is the bridge between the presentation layer and the data management layer. It serves to execute functions to process data from the presentation layer and also to store and retrieve data from the data management layer. For example, when the customer clicks add to cart button, it will trigger some functions within this layer to add the product to the shopping cart. And this product will be recorded into the shopping cart table in the data management layer.
Data Management Layer	This layer consists of all the tables in the database. Entity classes correspond to tables in the database and DAO classes contain all the operating class for the entities.

Table 4: Software Component Description

3.1.2 Design Classes

3.1.2.1 User Management

There are 3 different users of FlowerSeeker: Florist, Customer and Maintainer. Classes have been designed for florists and customers in the current version.

3.1.2.1.1 Florist Management

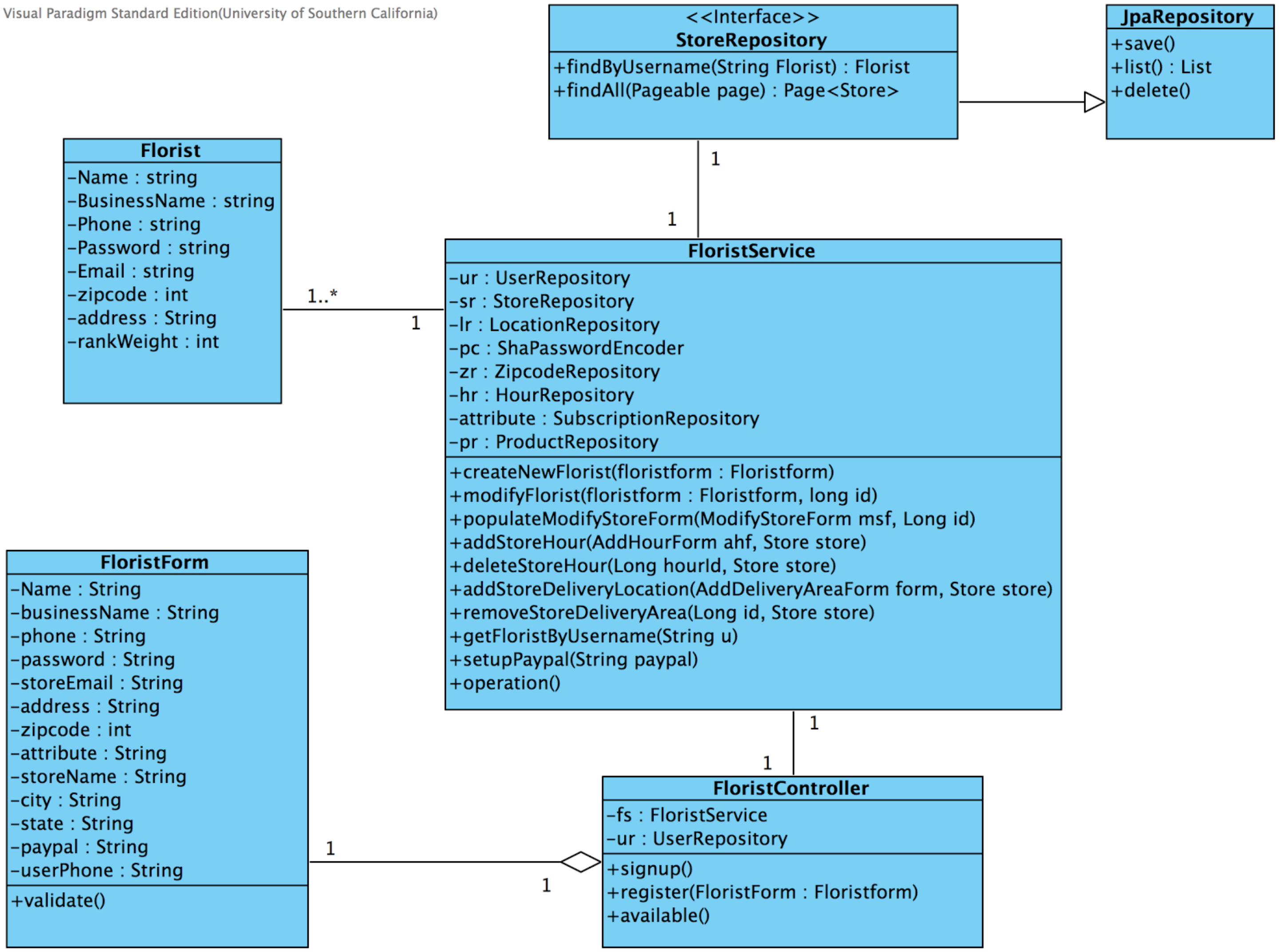

Figure 7. Florist Management Class Diagram

Class	Type	Description
Florist	Entity	It represents florist information.
FloritsForm	Controller	It contains the information in a form format used to render GUI in JSP.
FloristController	Controller	Control events between GUI and FloristService.
FloristService	Controller	Florist business logic class
FloristRepository	Controller	Interface
JpaRepository	Controller	It allows to use java JPA for serialization of Florist model

Table 5: Florist Management Class Description

3.1.2.1.2 Customer Management

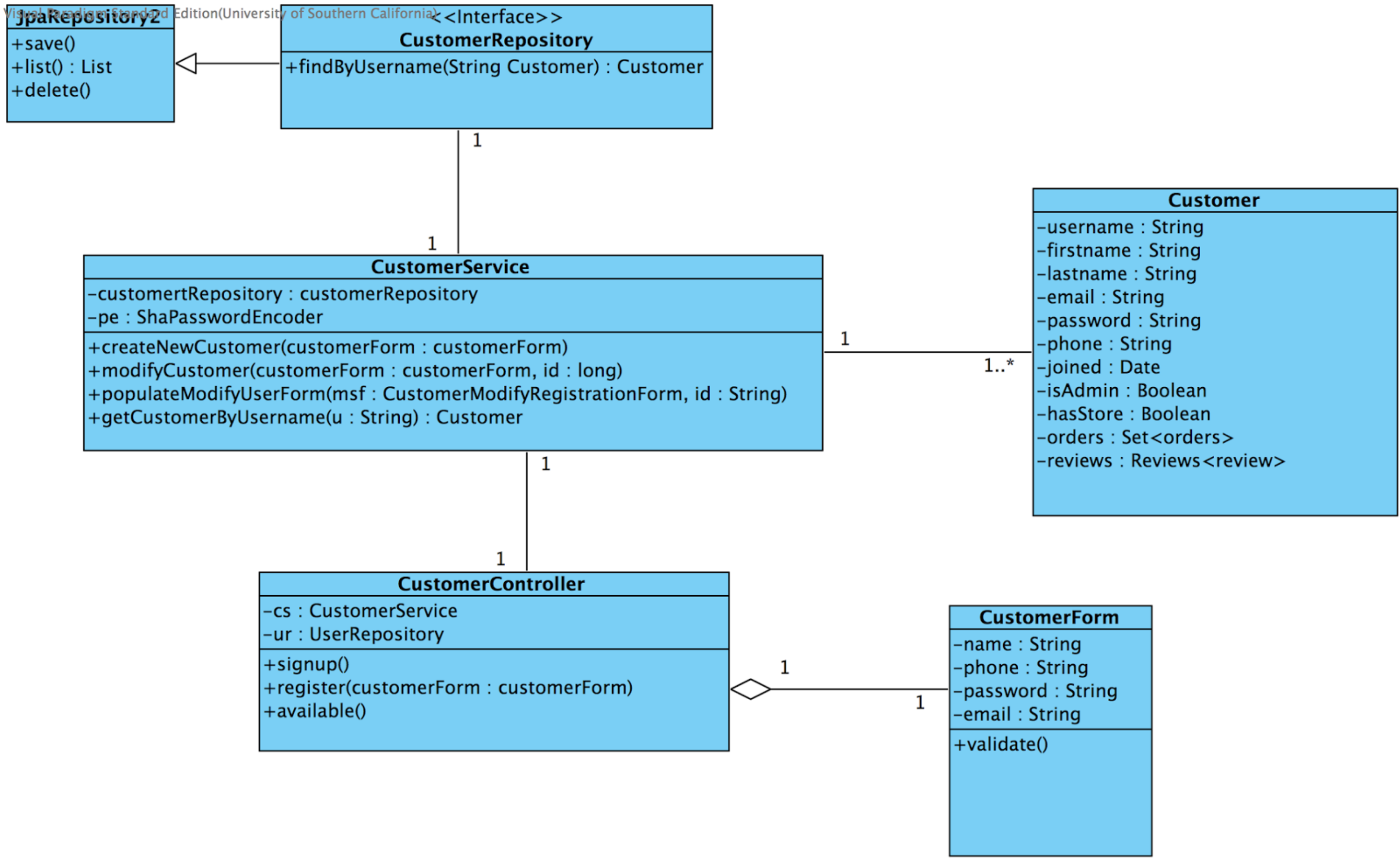

Figure 8. Customer Management Class Diagram

Class	Type	Description
Customer	Entity	It represents customer information.
Customer Form	Controller	It contains the information in a form format used to render GUI in JSP.
CustomerController	Controller	Control events between GUI and CustomerService.
CustomerService	Controller	Customer business logic class
CustomerRepository	Controller	Interface
JpaRepository	Controller	It allows to use java JPA for serialization of Customer model

Table 6: Customer Management Class Description

3.1.2.2 Product Management

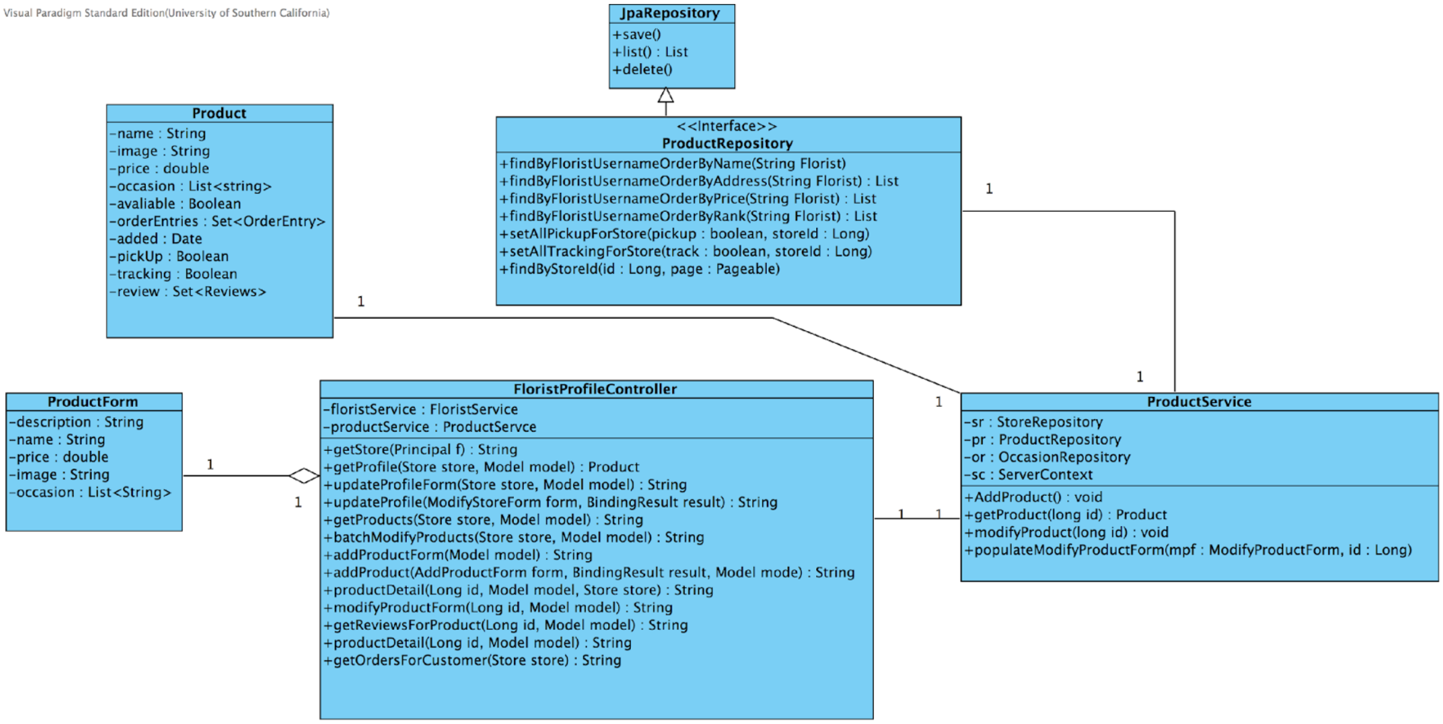
Visual Paradigm Standard Edition(University of Southern California)
JpaRepository
+save()
+list() : List
+delete()
Product
-name : String
-image : String
-price : double
-occasion : List<string>
-avaliable : Boolean
-orderEntries : Set<OrderEntry>
-added : Date
-pickUp : Boolean
-tracking : Boolean
-review : Set<Reviews>
<<Interface>>
ProductRepository
+findByFloristUsernameOrderByName(String Florist)
+findByFloristUsernameOrderByAddress(String Florist) : List
+findByFloristUsernameOrderByPrice(String Florist) : List
+findByFloristUsernameOrderByRank(String Florist) : List
+setAllPickupForStore(pickup : boolean, storeId : Long)
+setAllTrackingForStore(track : boolean, storeId : Long)
+findByStoreId(id : Long, page : Pageable)
1
1
1
1
ProductForm
-description : String
-name : String
-price : double
-image : String
-occasion : List<String>
1
1
FloristProfileController
-floristService : FloristService
-productService : ProductServce
+getStore(Principal f) : String
+getProfile(Store store, Model model) : Product
+updateProfileForm(Store store, Model model) : String
+updateProfile(ModifyStoreForm form, BindingResult result) : String
+getProducts(Store store, Model model) : String
+batchModifyProducts(Store store, Model model) : String
+addProductForm(Model model) : String
+addProduct(AddProductForm form, BindingResult result, Model mode) : String
+productDetail(Long id, Model model, Store store) : String
+modifyProductForm(Long id, Model model) : String
+getReviewsForProduct(Long id, Model model) : String
+productDetail(Long id, Model model) : String
+getOrdersForCustomer(Store store) : String
1
1
ProductService
-sr : StoreRepository
-pr : ProductRepository
-or : OccasionRepository
-sc : ServerContext
+AddProduct() : void
+getProduct(long id) : Product
+modifyProduct(long id) : void
+populateModifyProductForm(mpf : ModifyProductForm, id : Long)

Figure 9: Product Management Class Diagram

Class	Type	Description
Product	Entity	It represents product information.
Product Form	Controller	It contains the information in a form format used to render GUI in JSP
Product Controller	Controller	Control events between GUI and Product Service
Product Service	Controller	Product business logic class
Product Repository	Controller	Interface
JpaRepository	Controller	It allows using java JPA for serialization of Product model.

Table 7: Product Management Class Description

3.1.2.3 Payment Management

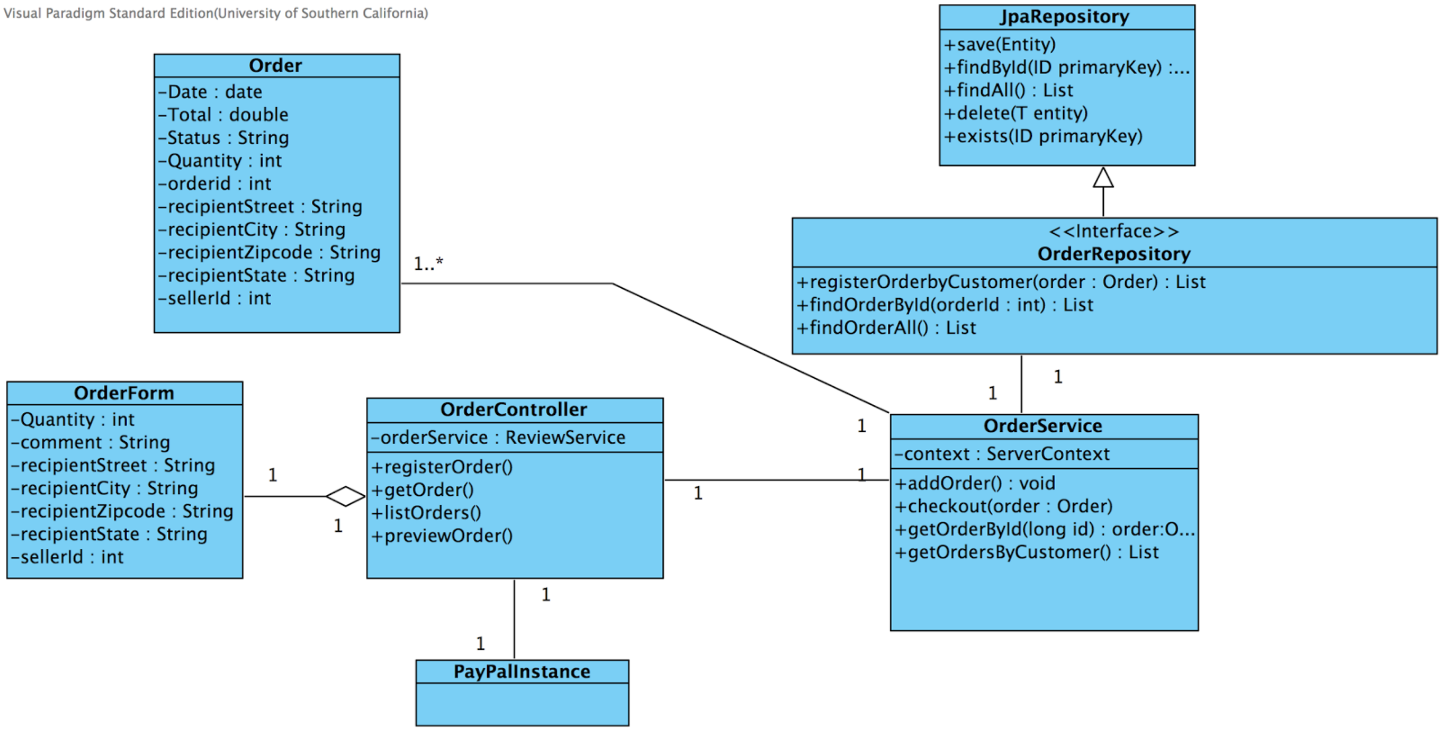

Figure 10: Payment Management Class Diagram

Class	Type	Description
Order	Entity	It represents order information.
OrderDao	Controller	Interface
OrderDaoImp	Controller	Implements OrderDao
PaymentManagementAction	Controller	It validates all inputs and call on corresponding services.
PayPal	Boundary	NDI
PreviewCheckoutPage	Boundary	Web GUI

Table 8: Payment Management Class Description

3.1.2.3 Review and Rating Management

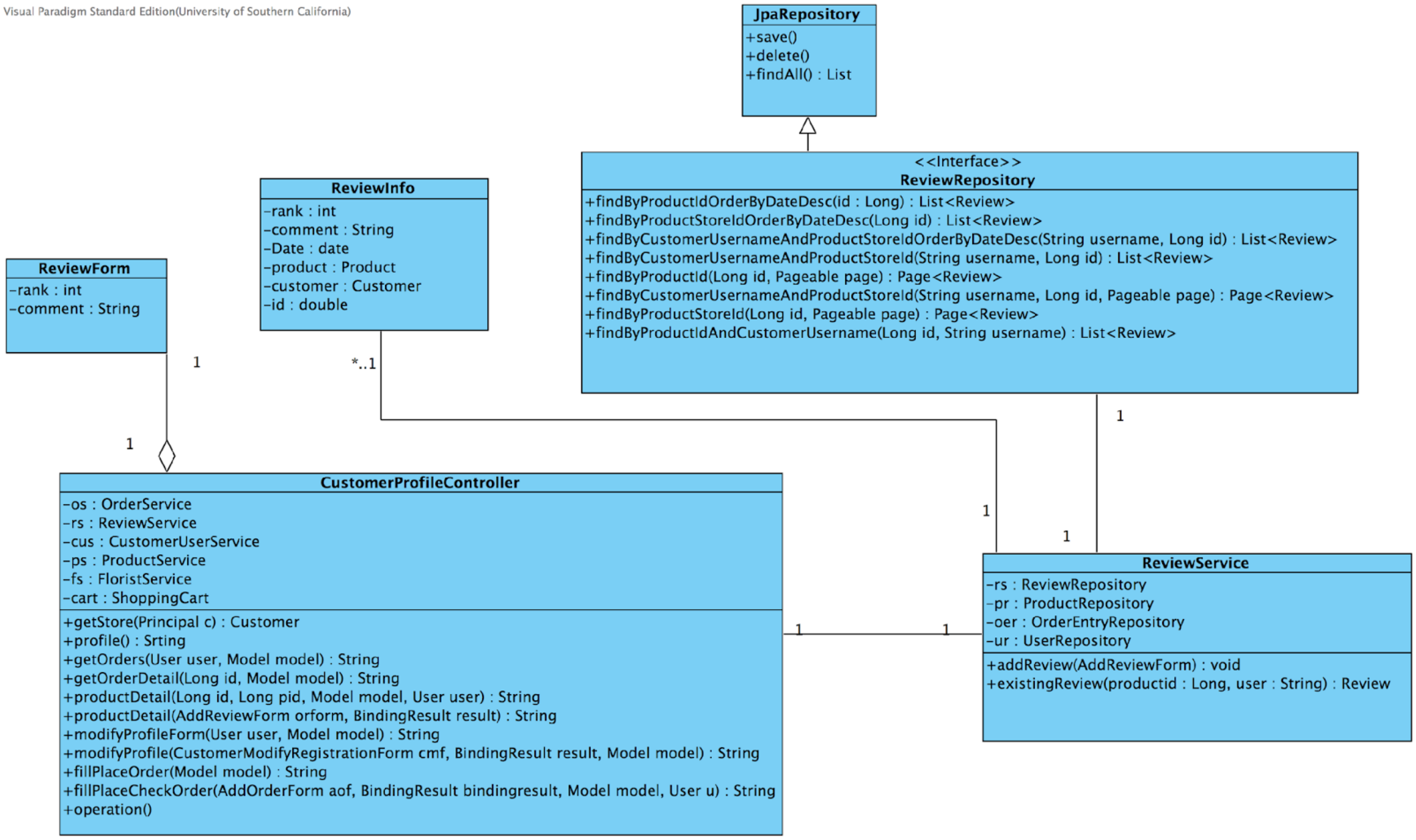
Visual Paradigm Standard Edition(University of Southern California)
JpaRepository
+save()
+delete()
+findAll() : List
<<Interface>>
ReviewRepository
+findByProductIdOrderByDateDesc(id : Long) : List<Review>
+findByProductStoreIdOrderByDateDesc(Long id) : List<Review>
+findByCustomerUsernameAndProductStoreIdOrderByDateDesc(String username, Long id) : List<Review>
+findByCustomerUsernameAndProductStoreId(String username, Long id) : List<Review>
+findByProductId(Long id, Pageable page) : Page<Review>
+findByCustomerUsernameAndProductStoreId(String username, Long id, Pageable page) : Page<Review>
+findByProductStoreId(Long id, Pageable page) : Page<Review>
+findByProductIdAndCustomerUsername(Long id, String username) : List<Review>
ReviewInfo
-rank : int
-comment : String
-Date : date
-product : Product
-customer : Customer
-id : double
ReviewForm
-rank : int
-comment : String
1
*..1
1
1
1
1
CustomerProfileController
-os : OrderService
-rs : ReviewService
-cus : CustomerUserService
-ps : ProductService
-fs : FloristService
-cart : ShoppingCart
+getStore(Principal c) : Customer
+profile() : Srting
+getOrders(User user, Model model) : String
+getOrderDetail(Long id, Model model) : String
+productDetail(Long id, Long pid, Model model, User user) : String
+productDetail(AddReviewForm orform, BindingResult result) : String
+modifyProfileForm(User user, Model model) : String
+modifyProfile(CustomerModifyRegistrationForm cmf, BindingResult result, Model model) : String
+fillPlaceOrder(Model model) : String
+fillPlaceCheckOrder(AddOrderForm aof, BindingResult bindingresult, Model model, User u) : String
+operation()
1
1
1
ReviewService
-rs : ReviewRepository
-pr : ProductRepository
-oer : OrderEntryRepository
-ur : UserRepository
+addReview(AddReviewForm) : void
+existingReview(productid : Long, user : String) : Review

Figure 11: Review and Raking Management Class Diagram

Class	Type	Description
Review	Entity	It represents review and ranking information.
ReviewForm	Controller	It contains the information in a form format used to render GUI in JSP
ReviewController	Controller	Control events between GUI and Review Services
ReviewService	Controller	Review business logic class
ReviewRepository	Controller	Interface
JpaRepository	Controller	It allows to use java JPA for serialization of Product model

Table 9: Review and Raking Management Class Description

3.1.3 Process Realization

Sequence diagrams have been generated for the high-risk and prioritized features including payment, search and order tracking.

3.1.3.1 Payment

FlowerSeeker uses PayPal classic API to process payments and handle transactions between customers and florist.

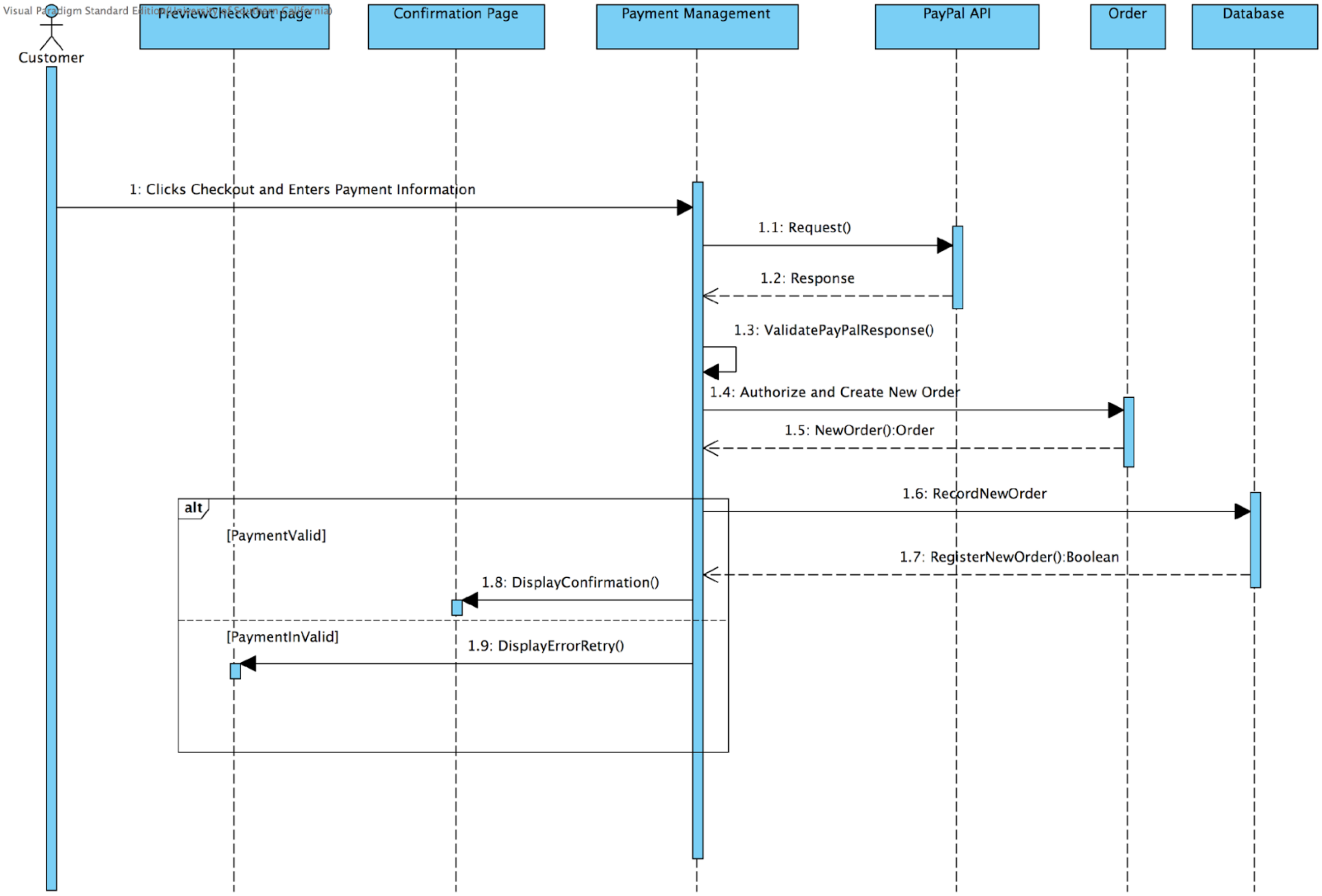

Figure 12. Payment Sequence Diagram

3.1.3.2 Search

Customers are able to search product by name, price, rating, location and occasions.

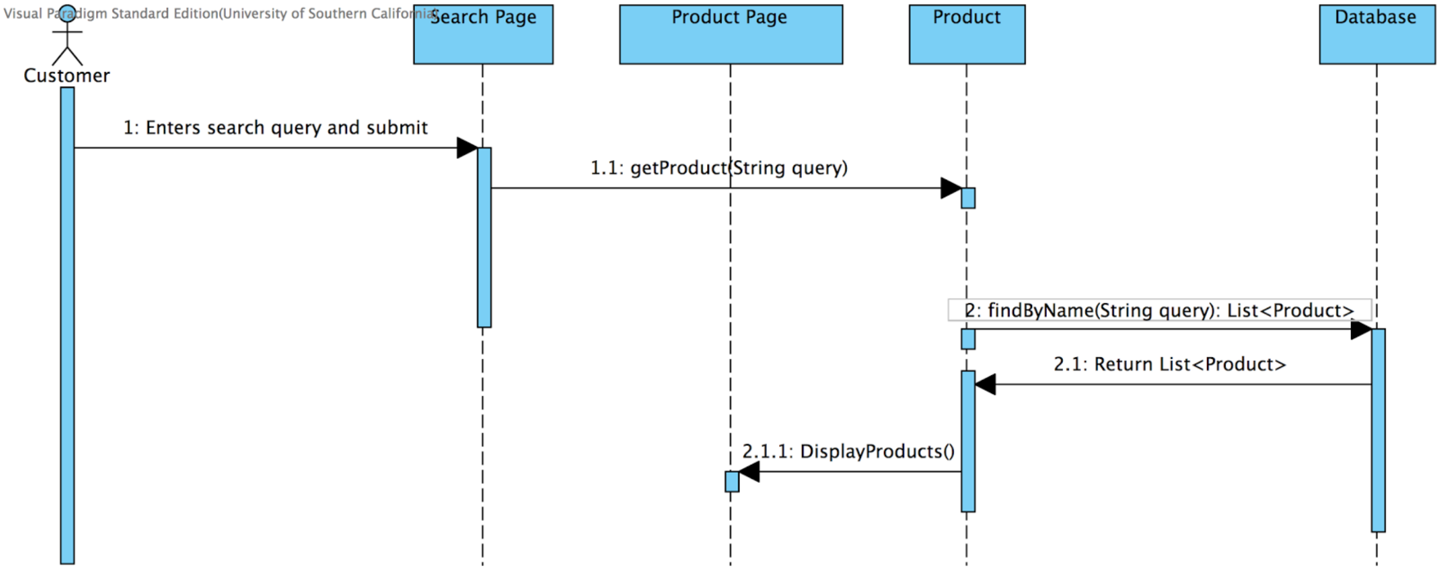
Visual Paradigm Standard Edition(University of Southern California)
Customer
Search Page
Product Page
Product
Database
1: Enters search query and submit
1.1: getProduct(String query)
2: findByName(String query): List<Product>
2.1: Return List<Product>
2.1.1: DisplayProducts()

Figure 13. Search By Name Sequence Diagram

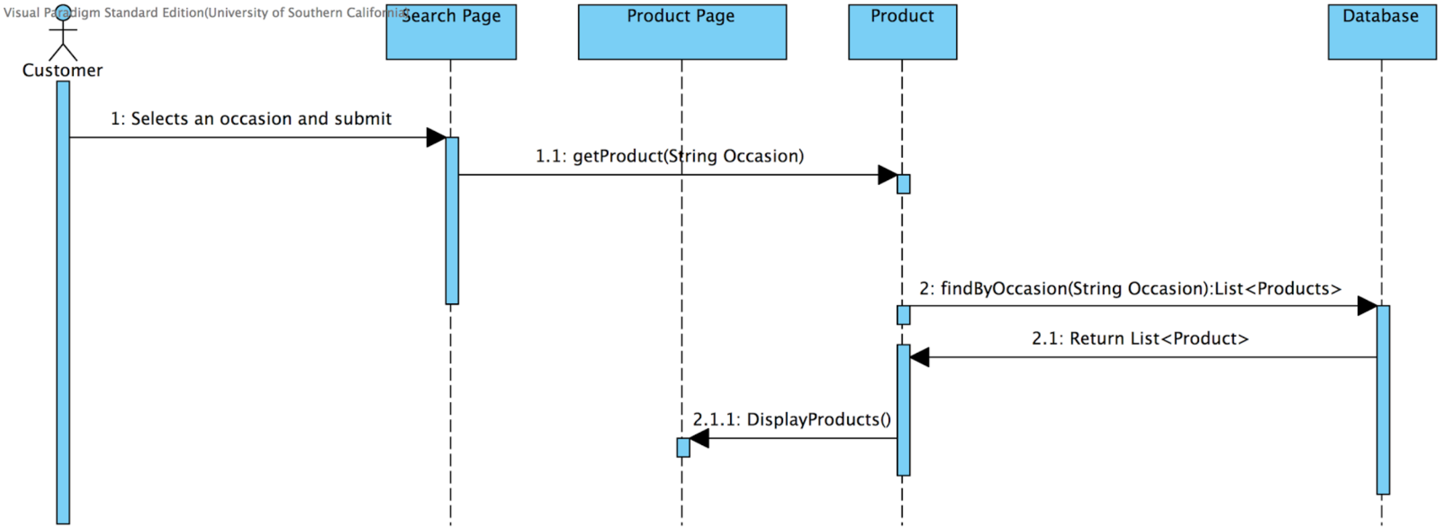
Visual Paradigm Standard Edition(University of Southern California)
Customer
Search Page
Product Page
Product
Database
1: Selects an occasion and submit
1.1: getProduct(String Occasion)
2: findByOccasion(String Occasion):List<Products>
2.1: Return List<Product>
2.1.1: DisplayProducts()

Figure 14. Search By Occasion Sequence Diagram

3.1.3.3 Order Tracking

Florists can update the status of the order after the order has been placed and customers can track the status of any of their orders.

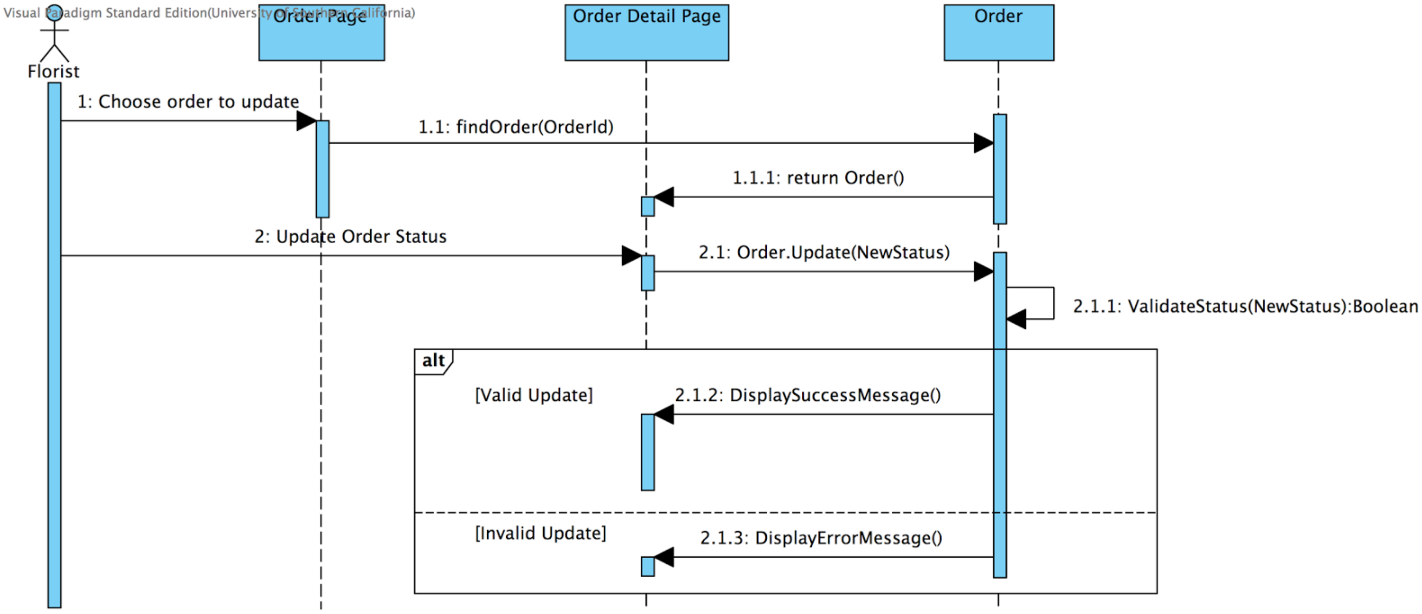
Visual Paradigm Standard Edition(University of Southern California)
Florist
Order Page
Order Detail Page
Order
1: Choose order to update
1.1: findOrder(OrderId)
1.1.1: return Order()
2: Update Order Status
2.1: Order.Update(NewStatus)
2.1.1: ValidateStatus(NewStatus):Boolean
alt
[Valid Update]
2.1.2: DisplaySuccessMessage()
[Invalid Update]
2.1.3: DisplayErrorMessage()

Figure 15. Florist update order status

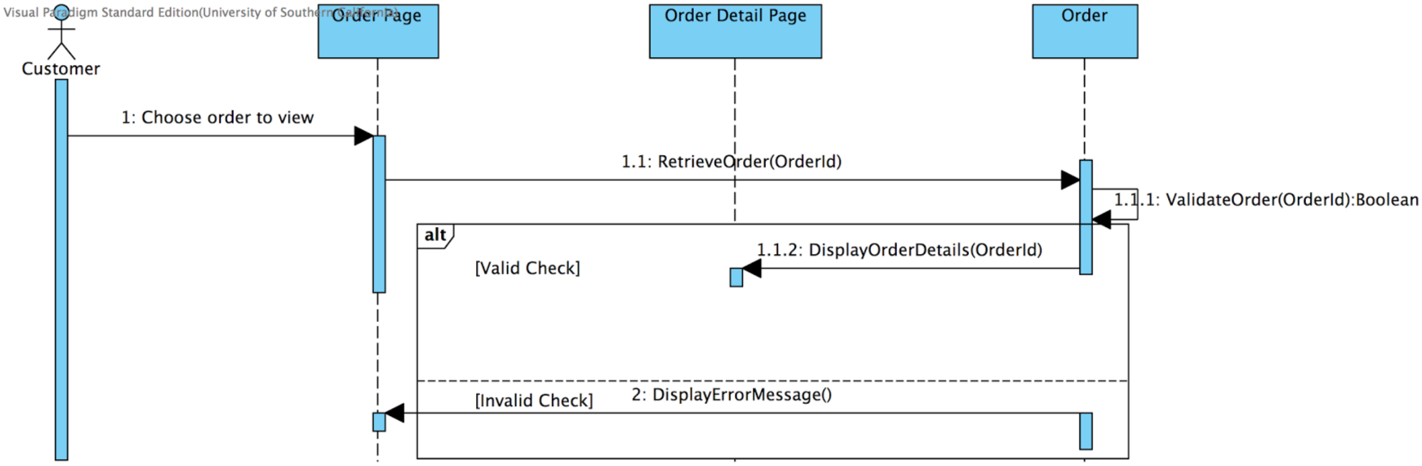
Visual Paradigm Standard Edition(University of Southern California)
Customer
Order Page
Order Detail Page
Order
1: Choose order to view
1.1: RetrieveOrder(OrderId)
1.1.1: ValidateOrder(OrderId):Boolean
alt
[Valid Check]
1.1.2: DisplayOrderDetails(OrderId)
[Invalid Check]
2: DisplayErrorMessage()

Figure 16. Customer track order status

3.2 Design Rationale

Since FlowerSeeker is a web application and users will be accessing the website on their own devices or computers, it would make sense for us to separate the presentation tier and the rest. In the client-server architecture, the client handles both the UI and the business logic tier and that makes the client “fat” [1]. A fat client requires heavy traffic with the server; therefore, it is more difficult to use over slower network connections such as Internet and wireless. So by adding the middle tier to separate the presentation and business logic, the client is only handling the UI and it makes the client “thinner”. Therefore, FlowerSeeker system uses 3-tier architecture to ensure scalable and good performance.

4. Architectural Styles, Patterns and Frameworks

Name	Description	Benefits, Costs, and Limitations
3-tier architecture	There are 3 tiers for the architecture of FlowerSeeker: A front-end web server, an application level that process content, and a back-end database that stores all information.	Benefits: 1. It allows any of the three tiers to be changed or updated independently. 2. It can be run on different platforms. 3. It provides good security since the middle layer protects the database layer. Limitations: 1. Since it has more complex structure, it is difficult to setup and maintain. 2. Since there exists physical separation of business logic functions and database, it may moderately affect the performance. Costs: Sever(s) for application server and database server.
PayPal	PayPal provides a defined API that handles payment processes.	Benefits: 1. It provides sandbox for developers to test payment modules in a secure way. 2. It has detailed documentation and it is easy to use. Costs: Since we are using PayPal classic API, the cost is free. However, we will discuss with the client to see if it’s necessary to upgrade it to a Pro version, which provides more payment options but with a monthly fee. Limitations: 1. It needs both the customers and florists to have a PayPal account. 2. Since we are using the free version, the service we receive is limited. For example, we will not

		be able to receive any credit card payments.
JPA	JPA enforces object oriented programming style. Allowing communication between objects and databases by replacing direct persistence communication between object-persistence entities.	Benefits: 1. Implementation of database management objects becomes easier and less coupled. Cost: Free Limitations: 1. JPA only deal with objects, if the system needs to retrieve simple data from a table, the entire object must be constructed and passed.
Spring Framework	Spring framework that helps to build robust enterprise java applications, it provides different components for specific task such as database management.	Benefits: 1. Allows to decouple object on the system and provides several useful framework for database and view layers. 2. Easy to test Cost: Free Limitations: 1. Huge learning curve

Table 10: Architectural Styles, Patterns, and Frameworks

Reference:

[1] http://en.wikipedia.org/wiki/Fat_client