SEG3102 - S. Somé Due: 09/10/14

Deliverable #2

Online Property Rental System

Team 15: Sean Floyd, Isaac Shannon (6778524, 6709038)

Table of Contents

Table of Figures

Executive Summary

In this document, we will develop analysis-level use case realizations of a subset of the Online Property Rental system. We will do so by developing a key abstraction model, provide UML Interaction Diagrams and a design class diagram.

The intended readers are the OPR customers, who will hire the team to build their web application.

This document contains the key abstractions, which provides a key abstraction model for the system as a UML class. It also contains the contract descriptions of the system responsibilities. We will also provide use case realizations, which will be in the form of UML Interaction Diagrams. Finally, we included a Design Class diagram, which provides an analysis-level class diagram showing methods, dependencies and navigability.

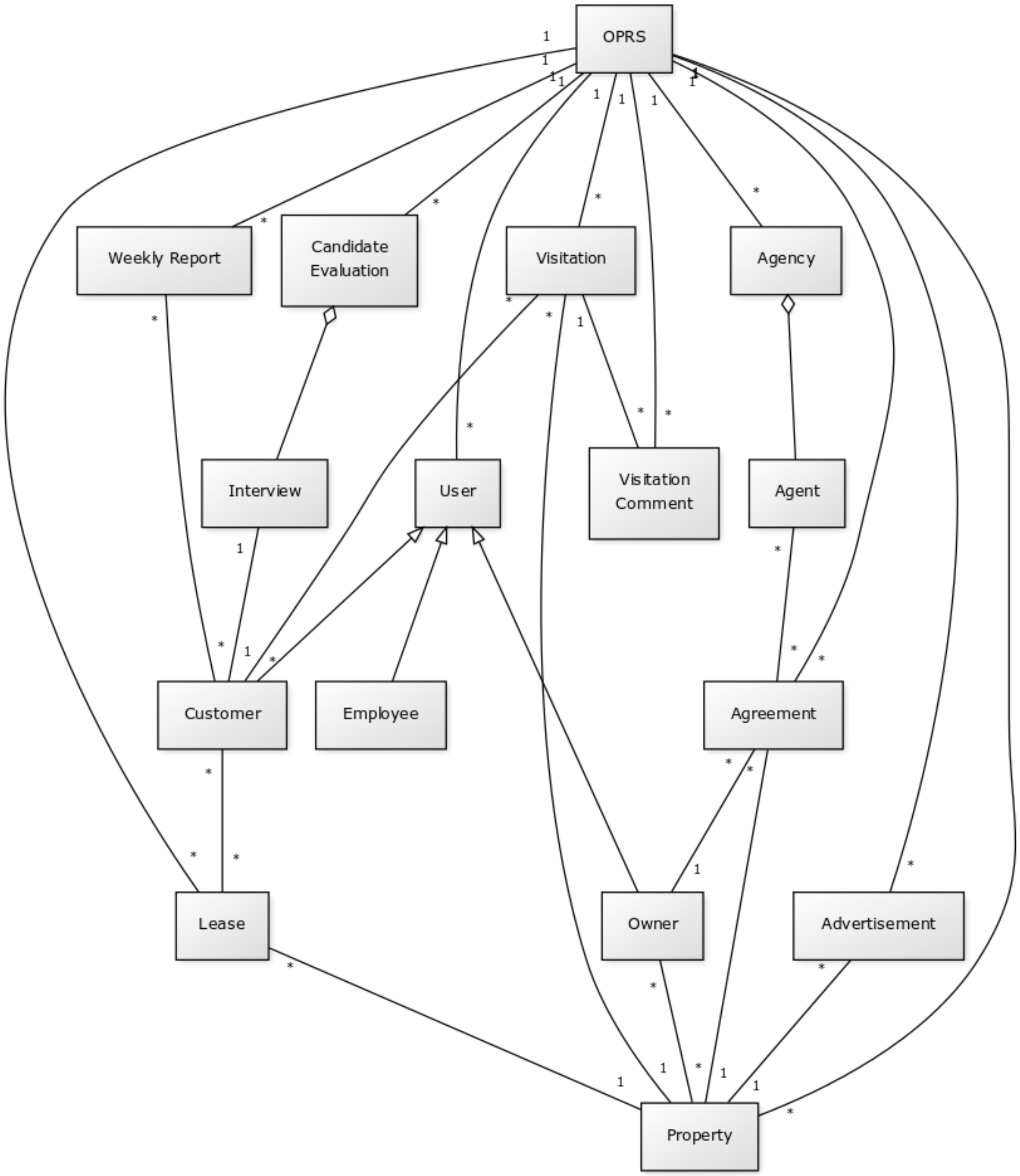
Description of Team & Roles

The team will consist of Sean Floyd and Isaac Shannon.
Sean wrote the document, answered questions 4, 5, 6, 7

Assumptions

No assumptions were made.

Key Abstractions



Contract Descriptions of the System Responsibilities

Name: //login (credentials)
Cross References: Use case: Login
Preconditions:

- There is an existing instance of OPR.
- User is a non-null instance of userAccount.

Post Conditions:

- no change to domain objects

Name: //logout()
Cross References: Use case: Logout
Preconditions:

- There is an existing instance of OPR.
- user is a non-null instance of userAccount.
- user is logged into userAccount.

Post-Conditions:

- no change to domain objects

Name: //new account (userInfo)
Cross References: Use case**:** Create Account
Preconditions:

- There is an existing instance of OPR.

Post-Conditions:

- If the user information was complete and user not registered
  - A new instance of user was created.
  - The attributes of user were initialized with userInfo.
  - User is associated to OPRS.
  - an identifier is associated to user.
  - A creation time is associated to user.

Name: //view account (accountId)
Cross References: Use Case: View Account
Preconditions:

- There is an existing instance of OPR.
- User is a non-null instance of userAccount.
- User is logged into userAccount.

Postconditions:

- no change to domain objects

Name: //view visiting list(user)
Cross References: Use Case: View Visiting List, Remove From Visiting List, Rent Property
Preconditions:

- There is an existing instance of OPR.
- User is a non-null instance of userAccount.
- User is logged into userAccount.

Postconditions:

- no change to domain objects

Name: //new property(owner, propertyInfo)
Cross References: Use case: Add Property
Preconditions:

- There is an existing instance of OPR.
- Owner is a non-null instance of ownerAccount.
- Owner is logged into ownerAccount.

Post-Conditions:

- If the user information is complete
    - A new instance of property was created.
    - The attributes of property were initialized with the propertyInfo.
    - Property is associated to owner.
    - Property is associated to OPRS.
    - An identifier is associated to property.

Name: //owner list properties(account)
Cross References: User Case: Owner View Properties, Update Property, Delete Property
Preconditions:

- There is an existing instance of OPR.
- Owner is a non-null instance of ownerAccount.
- Owner is logged into ownerAccount.

Postconditions:

- no change to domain objects

Name: //update property(property, propertyInfo)
Cross References: Use Case: Update Property
Preconditions:

- There is an existing instance of OPR.
- owner is a non-null instance of Owner.
- owner is logged into ownerAccount.
- owner has 1+ listed properties with OPRS
- property is a non-null instance of Property.

Postconditions:

- changed attributes of property were modified from propertyInfo

Name: //view properties()
Cross References: Use Case: View Properties
Preconditions:

- There is an existing instance of OPR.

Postconditions:

- no changes to domain objects

Name: //search properties(input)
Cross References: Use Case: Search Properties, View Properties
Preconditions:

- There is an existing instance of OPR

Postconditions:

- no changes to domain objects

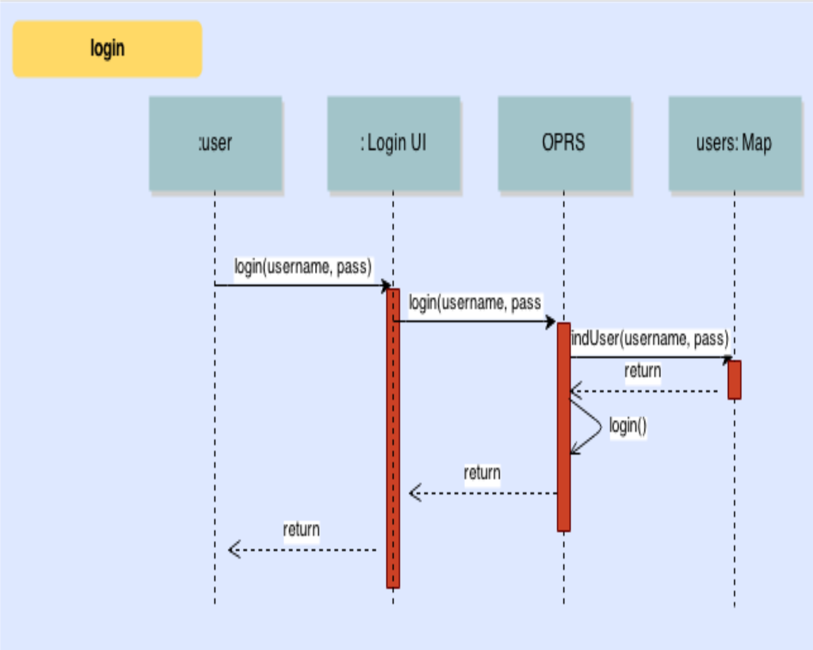
Name: //add to visiting list(aProperty, aCustomer)
Cross References: Use Case: View Properties, Add To Visiting List
Preconditions:

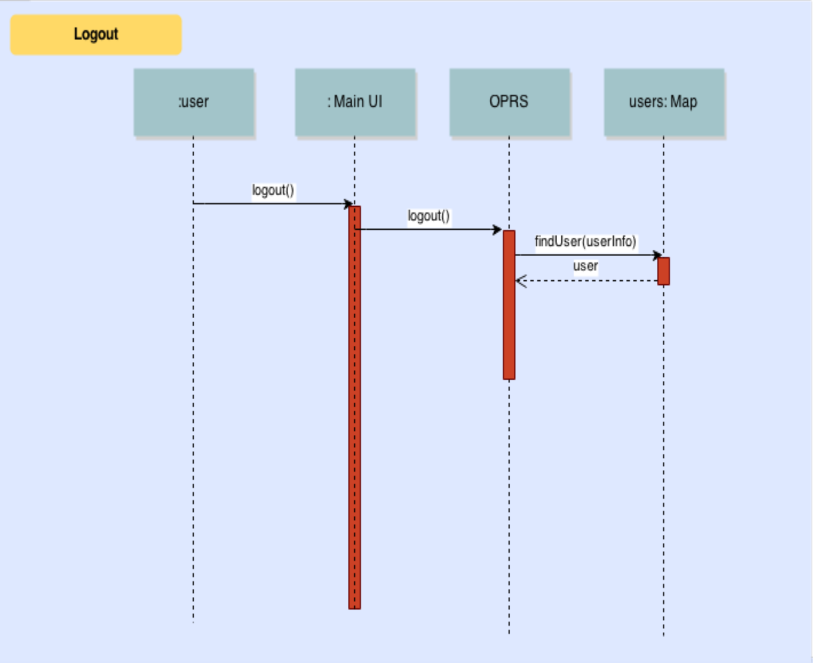
- There is an existing instance of OPRS
- aCustomer is a non-null instance of Customer
- aCustomer is logged into OPRS
- aProperty is a non-null instance of Property

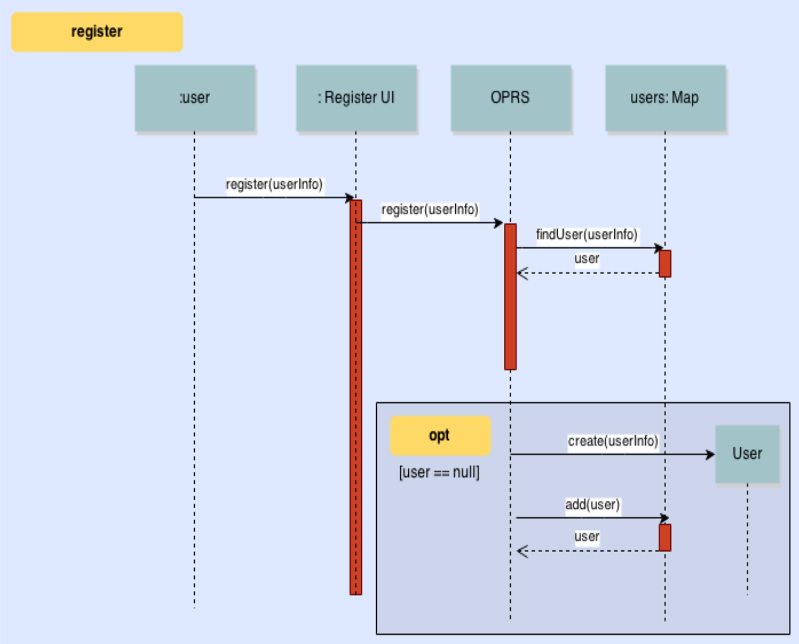
Postconditions:

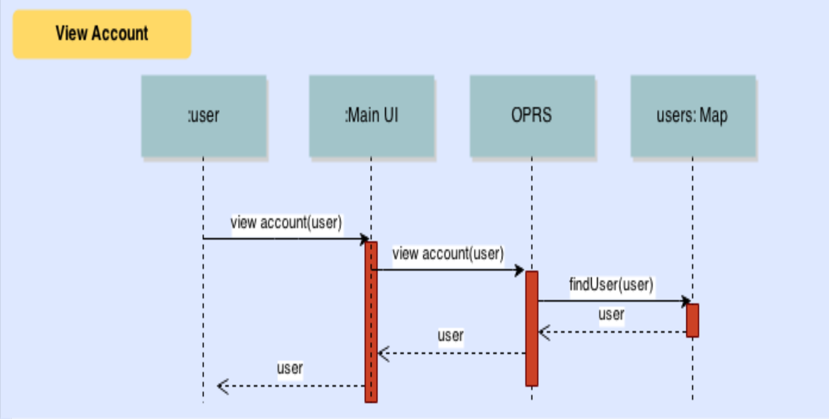
- a new instance of Visiting aVisit is created
- the attributes of aVisit were initialized from aProperty
- aVisit was associated to aCustomer

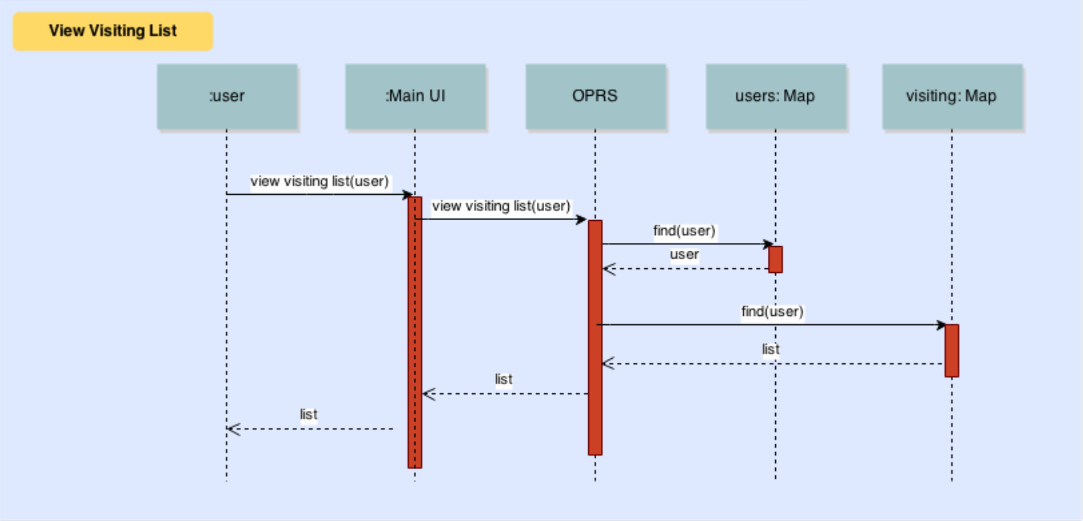
Use Case Realizations

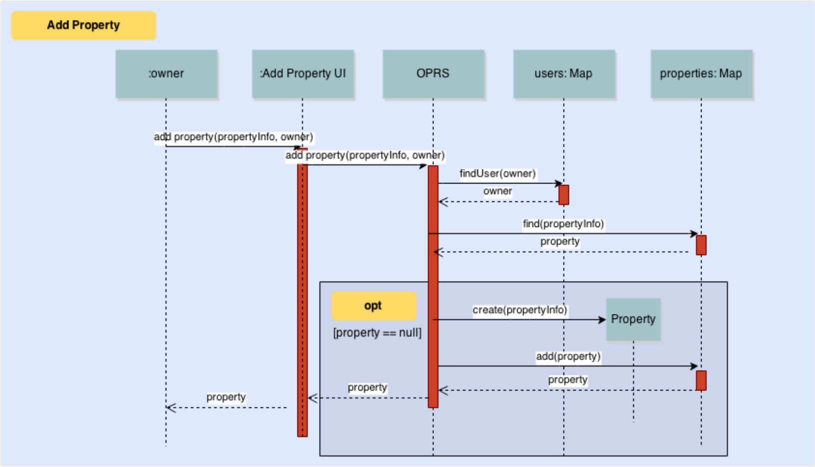


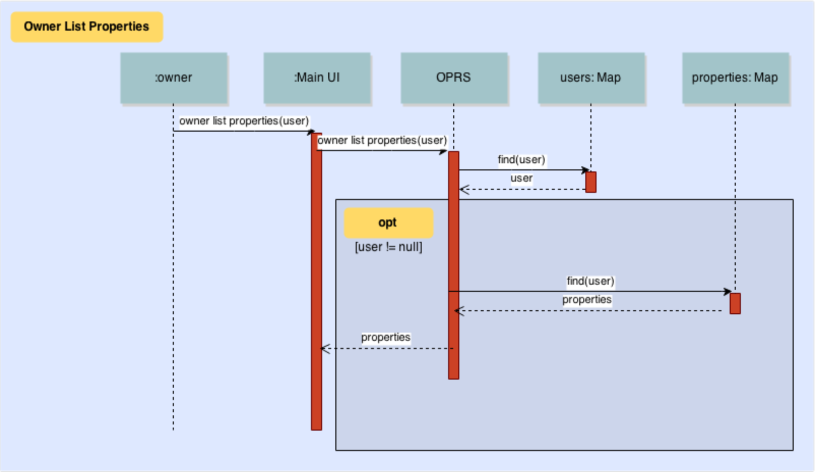


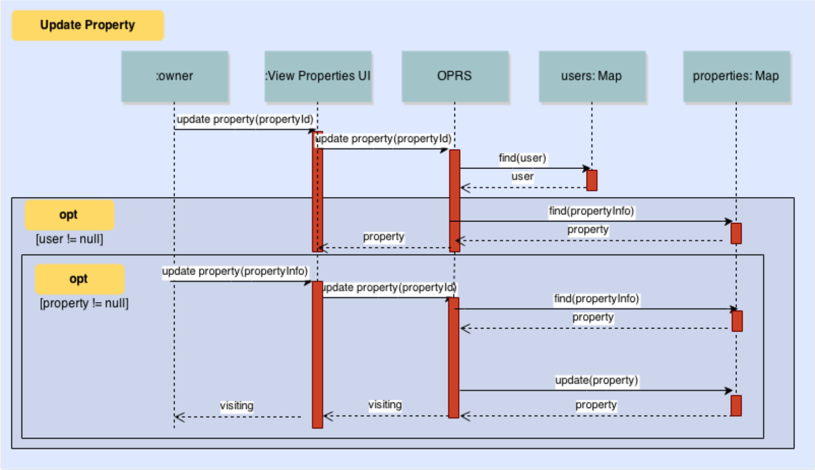


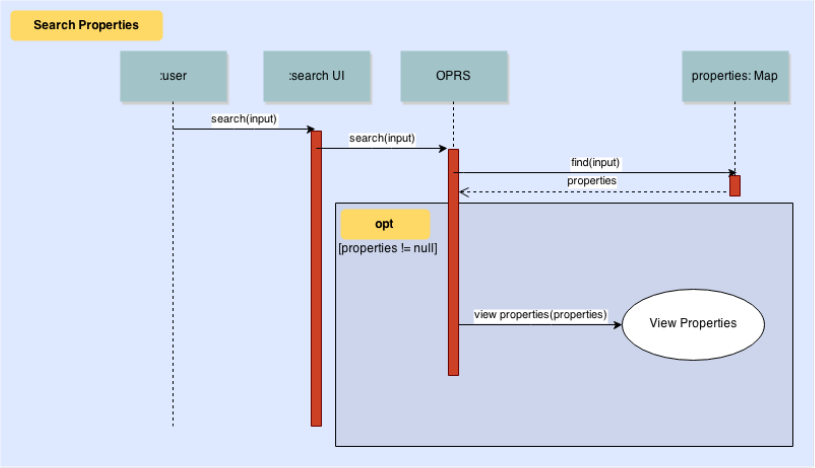


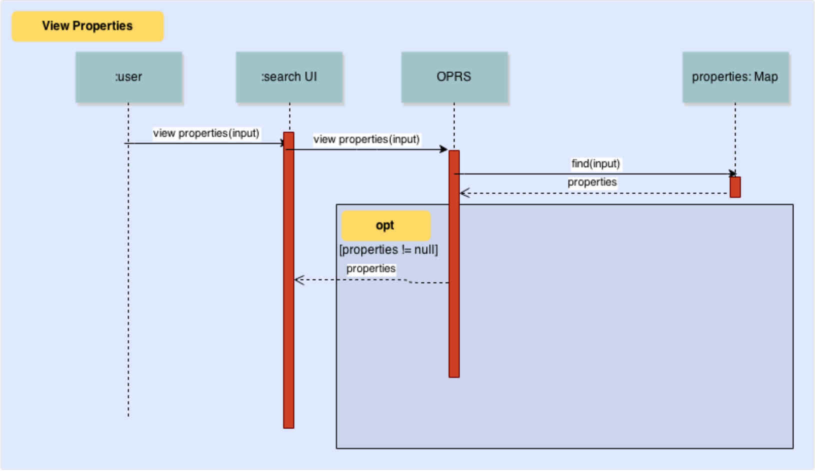


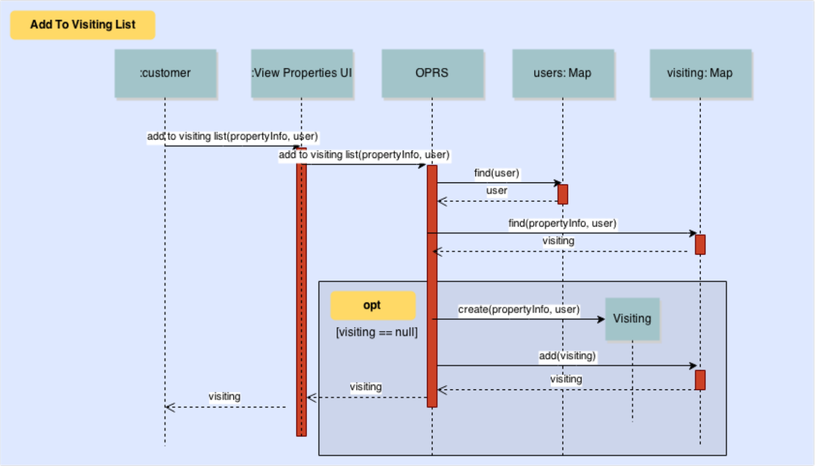

Add Property
:owner
:Add Property UI
OPRS
users: Map
properties: Map
add property(propertyInfo, owner)
add property(propertyInfo, owner)
findUser(owner)
owner
find(propertyInfo)
property
opt
[property == null]
create(propertyInfo)
Property
add(property)
property
property
property


Owner List Properties
:owner
:Main UI
OPRS
users: Map
properties: Map
owner list properties(user)
owner list properties(user)
find(user)
user
opt
[user != null]
find(user)
properties
properties


Update Property
:owner
:View Properties UI
OPRS
users: Map
properties: Map
update property(propertyId)
update property(propertyId)
find(user)
user
opt
[user != null]
find(propertyInfo)
property
property
opt
[property != null]
update property(propertyInfo)
update property(propertyId)
find(propertyInfo)
property
update(property)
visiting
visiting
property

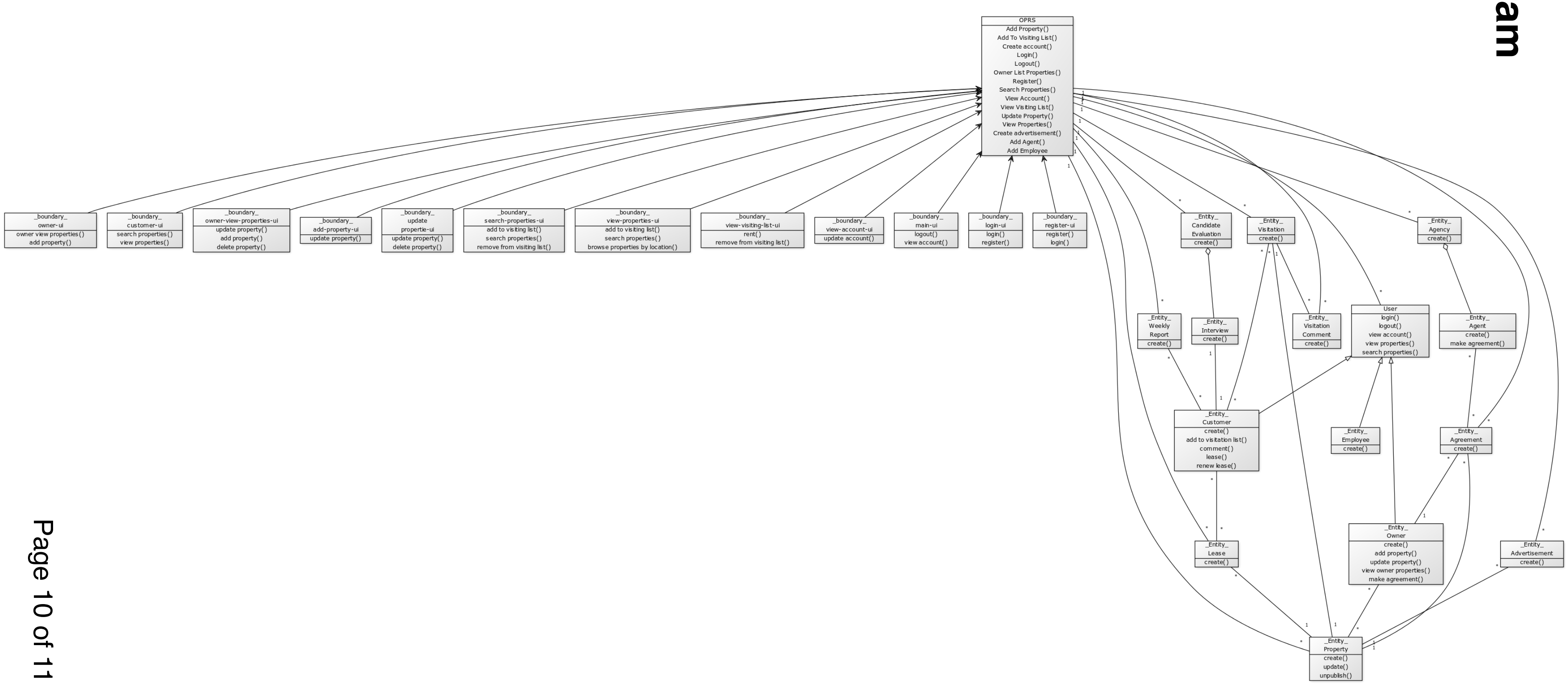

Search Properties
:user
:search UI
OPRS
properties: Map
search(input)
search(input)
find(input)
properties
opt
[properties != null]
view properties(properties)
View Properties


View Properties
:user
:search UI
OPRS
properties: Map
view properties(input)
view properties(input)
find(input)
properties
opt
[properties != null]
properties


Add To Visiting List
:customer
:View Properties UI
OPRS
users: Map
visiting: Map
add to visiting list(propertyInfo, user)
add to visiting list(propertyInfo, user)
find(user)
user
find(propertyInfo, user)
visiting
opt
[visiting == null]
create(propertyInfo, user)
Visiting
add(visiting)
visiting
visiting
visiting

Design Class diagram

http://www.yuml.me/e78c9daa



References

1. yuml.me
   accessed November 12th to create UML design class diagram and key abstractions
2. http://www.umlet.com/
   accessed October 14th 2014 to create UML diagram
3. http://www.dreamhomesource.com
   accessed October 14th 2014. Team logo taken from this website.
4. www.draw.io
   accessed November 12th 2014 to create use case realizations.

Appendix

A. Initial Requirements
B. Use Cases
C. Modalities
D. Deliverable1.pdf

Appendices are not included.