Design Document

Date: Thu Oct 30 22:43:38 EDT 2014
Author: Vinod Halaharvi
Part 2 of 3 part SquareDesk application

Introduction

This application is called SquareDesk. This application is akin to AirBnB and other office sharing sites on the Internet. In this app, people can make extra money by renting a part (or whole) of their house as office Space. People who are looking to provide office space can login online and list their facility for rent and people who are looking to rent the office space can also login and search for the office space near by and choose to rent the one they like.

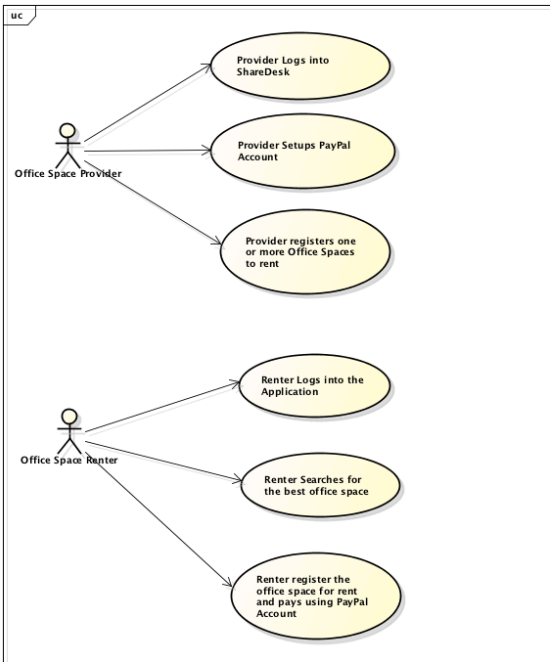
Overview

- SquareDesk is a new service that allows people to rent out their home as office space and make additional income by renting out portions of their home as office space.
- The job of the SquareDesk is to make it easy for people to register and list their homes as office space.
- Provider simply navigates to the SquareDesk site, registers, provides details about the space they have for rent, and SquareDesk does the rest.
- Renter goes to SquareDesk web site and search for office space based on various search criteria and selects the officespace he likes the best. He then books this office space.
- As a commission SquareDesk takes (10%) of what Provider makes.
- Both Providers and Renters can rate each other

Requirements

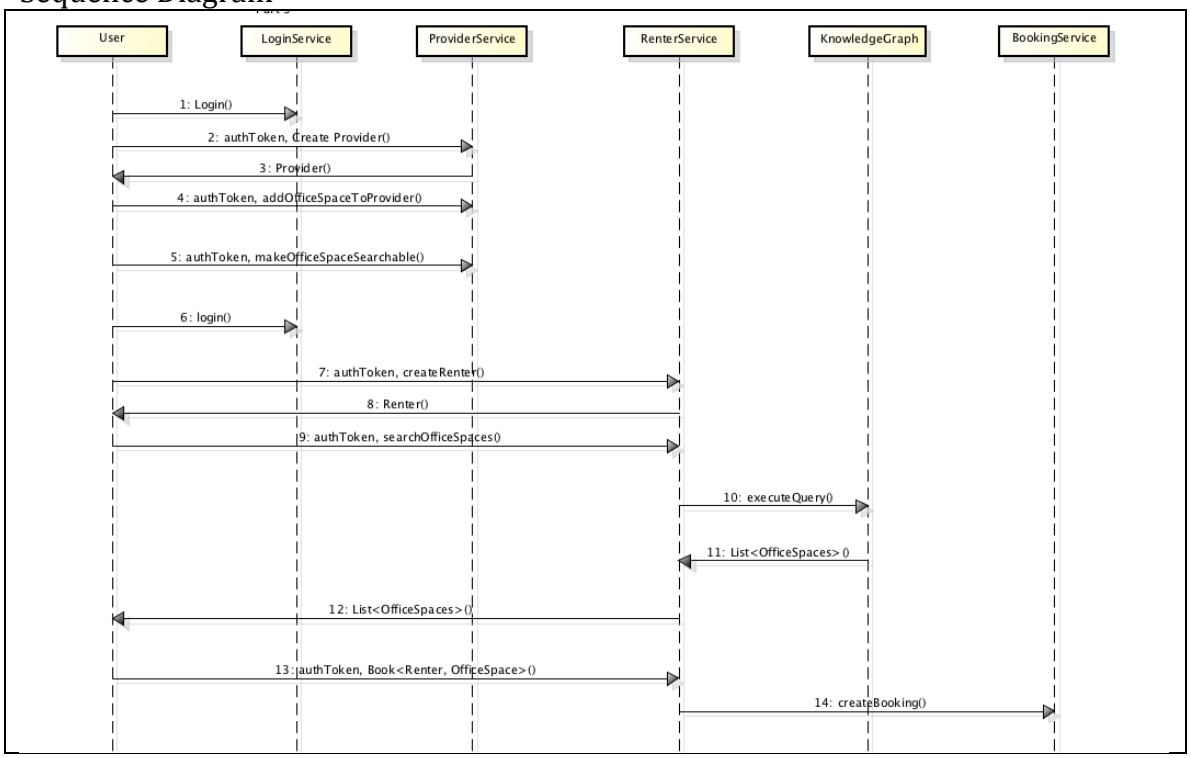
- SquareDesk has 3 key objects Providers, OfficeSpaces and Renters
- Create, read, update instances of Office Space Provider
- Create, read, update instances of Office Space
- Create, read, update instances of Renter
- Support specification of OfficeSpace, Office Provider and Renter details
- There should be a place holder for authentication token, Authentication is part of sprint 3
- OfficeSpace, OfficeSpace Provider, Renter, all have unique names in the system
- Support searching for office space based on Renter search criteria

Use Cases



Implementation

Sequence Diagram



Process flow and Sequence Diagram: The following steps, as detailed below will explain the steps shown in the sequence diagram. The user could be both a renter and a provider, but for convenience of this example we consider two different users, one renter and another provider. The user logs in and provides proper credentials to the login service. If the credentials are valid the login service will provide user a unique authentication token. The log in service is not part of the current sprint. We assume that the user has provided proper credentials and the login service has provided back the user, a valid authentication token. The user will have to provide this token to the various service APIs to request a service. Without a valid token, the user request to the API services will be ignored and AccessException will be raised.

Provider creation process and KnowledgeGraph: Knowledge graph, as seen in the first assignment is responsible to store associations between various "searchable" objects. This interface is kept as is and adjustments have been made to some classes to make this search happen. We will explain this in this section. We have seen in the previous sprint how user creates a provider and office spaces and how the office spaces are added to the provider. In this sprint, after the provider adds an office space, he makes this office space searchable. Their is a 'isSearchable()' method in the Provider class which acts as a flag to indicate if this officeSpace is searchable. When the Provider makes the officeSpace searchable, that point on, all the CRUD operations on the officeSpaces/Provider will be reflected in the KnowledgeGraph. That way the service API state is in sync with the KnowledgeGraph. No other explicit synchronization is needed.

Renter, Provider and Profile objects: Profile is an abstract class that refactors the common functionality of provider and renter objects. Both the renter and provider objects have ratings, they both have to be authenticated before making requests to the service API's. They both have contact information and etc. It makes sense to abstract both the renter and Provider object using a comman class. The Profile object provides this abstraction. Both the Provider and Renter classes extend from this class. We are currently not using the 'User' class, as was brought up in the discussion to act as a composition class for both Provider and Renter. User class might make more sense during the authentication and authorization sprint and so is left out from the current sprint. The RenterService and ProviderService API use the individual renter and provider objects respectively, since there is no one "User" object yet.

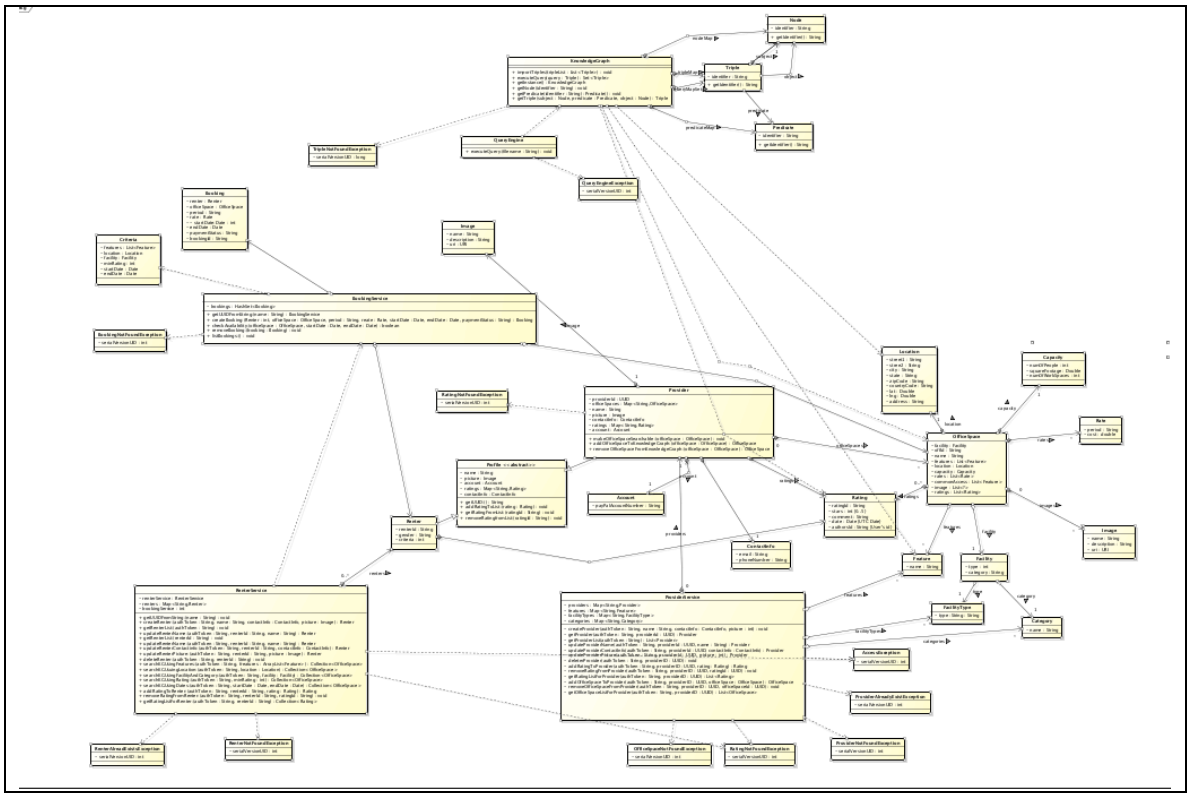
Renter, Provider object creation using singleton ProfileFactory: Renter and Provider Service APIs use singleton ProfileFactory class to create the Renter and Provider objects. The first argument to the createProfile() method distinguishes the renter creation from the provider creation.

Renter, RenterService comparison with Provider and ProviderService: The Renter object is very much similar to the Provider object that we created in the assignment 2. Just like provider object, the renter object has a unique system generated id, which is unique with in a JVM. The hash mapping of the Renter Service

API class is very similar to the Provider Service class. So please refer to the ProviderService API section for complete details.

Renter object creation process, OfficeSpace search and Booking process: A user looking to rent an office space, will first request the renter service API to create a renter profile. The renter service creates this profile and returns the user with a renter object. The user will search for an office space in the system through renter service. The renter service in turn uses the KnowledgeGraph search engine to perform the search. The result of this search is an ArrayList of officeSpaces, which are returned to the Renter. The renter then requests the renter service to perform a booking on its behalf. Booking Service API is responsible for keeping the rental booking state with in the system. The renter service API requests the Booking service to create a booking for the given renter and officeSpace combination. There are other attributes that are associated at this granularity of association. Among other attributes, the Booking Service keeps track of startDate and endDate of the rental period. The retail service can query the Booking service directly for the availability of the officeSpaces, which fall in the range or endDate - startDate without having to contact the KnowledgeGraph. There is one search abstraction however, that the renter can make use of. The renter can create a search criteria using 'Criteria' object. This object has various properties to keep track of search criteria, for example, location, facility, category, startDate, endDate etc. The renter could provide this object directly to the renter service to find the officeSpaces that meet all the search criteria. There is an implicit AND during the search.

Class Diagram



Class Dictionary

(Previous assignment ProviderService Sumary, Please Skip if not relevant) ProviderService class is the entry point to the application. ProviderService API has associations to Provide class. This association is stored in ‘providers’ property of the ProviderService class. The provider’s property is a 'Map' data structure, which maps from String to a Provider’s object. This String is a global identifier, which is used to uniquely identify a provider in the system.

There is an assumption made that provider's name is unique in the system. The global identifier is derived off of the provider's name. Though practically provider's real name might not be unique, the system can enforce the provider to create a unique username. Here the name property refers to that “unique name”.

The provider service class also holds reference to features, facility type and category objects. These objects follow a Flyweight pattern so there is only one instance of each object. For example there is only one feature object for a feature text like 'WIFI'. All the other objects share these object references that uses WIFI. The same applies to facility types and the categories. Provider service API class has lot of operations. Provider service class has operations to create a provider, to update a provider or delete a provider or get provider information based on global unique identifier. Provider service API has operations to add ratings for a given provider or an office space.

The workflow is as follows: the client requests the provider service API to create provider by giving Provider information like name, contact information and image along with the authorization token. The provider service class authorizes the token then creates a new provider based on this information. It also creates a globally unique provider ID for the provider. This globally unique provider ID is stored in the providerId attribute of the provider object and also inserted in the 'providers' property map of ProviderService object. The return value of create operation is a reference to provider object. The client can then use provider object's reference to get the provider ID and use the provided ID to do CRUD operations on the provider in future. The same applies to the ratings and OfficeSpace operations. However there is no natural key for the ratings and office space classes. Though the synthetic keys offId and the ratingId can be obtained from their respective objects. Using these 'id' references, CRUD operations can be performed on those objects. Please refer to the class diagram for a clearer visual on the object relations.

Account

Account class is used to store PayPal account information of the Provider. More details will emerge in part 2 of this project. This is a placeholder for the current sprint

Property Name	Type	Description
payPalAccountNumber	String	Private Pay Account number for the Provider Account to accept payment from the renter

ContactInfo

Class to store Contact Information of Provider. The Provider API should validate email and phoneNumber before storing the information in to those attributes.

Property Name	Type	Description
email	String	Private field to store email of the Provider. Email class to abstracts the Provider's email.
phoneNumber	String	Private field to store Provider's phone number. PhoneNumber class abstracts Provider's PhoneNumber

Provider

Provider class is used to store Provider's information. Each Provider in the system has a unique identifier. Provider class holds the association to OfficeSpace class. The Association between Provider and OfficeSpace class is one of Composition. So if the

Provider object is deleted, all the associated OfficeSpace objects should be deleted as well, with few notable exceptions - some objects are created as singleton objects. FacilityType and Feature and Category objects are singleton. OfficeSpace object shares these objects. If a feature or facility type is not present in the list, then the Provider creates a new facility type or feature. A Map data structure is used to store these objects and so no code updates are needed. Please check for various associations under the OfficeSpace class dictionary for more detail.

Property Name	Type	Description
contactInfo	ContactInfo	Private association to ContactInfo class
providerId	String	Private unique identifier for Provider. This is system generated id and is unique across the JVM
name	String	Private name of the provider
picture	Image	private association to Image class
account	Account	Private association to Account class
officeSpaces	Map<String, OfficeSpace>	Private zero to many associations to OfficeSpace class.
ratings	Map<String, Rating>	Private zero to many associations to Rating

OfficeSpace

An Office Space is the area within the home or garage that is available for rent as office space. An office space has an unique identifier and a name. An office space includes Features, a Location, Capacity information, Images, Common Access, Rates, and Facility Type

Properties

Property Name	Type	Description
offId	String	Private unique identifier for Provder. This is system generated id and is unique across the JVM
name	String	Private name of the OfficeSpace
location	Location	Private association to Location class
capacity	Capacity	Private association to Capacity class
facility	Facility	Private association to Facility class
features	List<Feature>	Private associations to Feature class. This is 0 to much association. Feature class is instantiation by ProviderService

		and maintained by it. This class uses a Flyweight design pattern. For each feature text there is only one object and all the classes share that object. When the office-Space is deleted that object is not deleted. So the association type is not that of composition.
rates	List<Rate>	Private association to Rate class. One to may association.
commonAccess	List<Feature>	Private field to store features that are common for this office space. This features will thus be shared by multiple renters who rent this office space
images	List<Image>	Private association to Image class.
ratings	List<Rating>	Private zero to many association to Rating

Location

This class stores the location information of the OfficeSpace. Both the address information and latitude and Longitude information is stored as a part of the location information. Latitude and Longitude is derived from the address information of the office space. Latitude and Longitude information will be used for searching purposes to calculate the distance related metrics. Provider Service API should make proper validation checks before inserting data in to this class.

Properties

Property Name	Type	Description
street1	String	Private field to store street address 1
street2	String	Private field to store street address 2
city	String	Private field to store city
state	String	Private field to store state
zipCode	String	Private field to store zip code
address	String	Private field to store address
countryCode	String	Private field to store country code
lat	Double	Private field to store latitude information
lng	Double	Private field to store longitude information

Image

Image class is used to store the name of an Image, its description and URI path. The other classes to store Image information use this class

Properties

Property Name	Type	Description
name	String	Private filed to store name associated with this image
description	String	Private field to store description of the image
uri	URI	Private field to store Uniform resource Identifier of the Image

Rating

There are ratings for both Office Space Provider and the Office Space renter. This class stores Rating information.

Properties

Property Name	Type	Description
stars	int	Private field to store stars
comment	String	Private field to store comment from the author
date	Date	Private field to store date that this comment was created
authorsId	String	Private field to store author of the comment
ratingId	String	Private unique identifier for Provider. This is system generated id and is unique across the JVM

Rate

Each office space includes one or more Rates. A Rate specifies a period and cost. For example a Rate may specify a period of one day, and a cost of $10. Note that all transactions are performed using US dollars. Multiple Rates can be associated with an office space, for example, a weekly rate can be offered in addition to a daily rate.

Properties

Property Name	Type	Description

period	String	Private field to store the Rental period
cost	Double	Private field to store the Rental cost for the period specified in the period field

Capacity

The Capacity defines the size of the office space. There is a single capacity object per office space. The capacity defines the number of people, the square footage, and number of workspaces

Properties

Property Name	Type	Description
numOfPeople	int	Private field to store number of people that can fit into this facility
squareFootage	Double	Private field to store square footage of this facility
numOfWorkSpaces	int	Private field to store number of work space of this facility

FacilityType

The Facility Type defines the type of office space, either a home or garage. If a home, the facility type can be further categorized as an office, kitchen table, or dining room, or some other location within the house. The categorization of the space within the home should be extensible, to support a new location, for example, back porch, or attic.

Property Name	Type	Description
type	type: String	Private field to store the type of the Facility. Either home or Garage

Category

This class store information about category with in the home facility. For example office, kitchen table, or dining room, or some other location within the house

Property Name	Type	Description

name	String	Private field to store the category information.

Facility

Facility class holds associations to category and FacilityType class.

Property Name	Type	Description
category	String	Private field to store the category of Facility.
type	String	Private field to store the type of the Facility. Either home or Garage

ProviderService

Provisioning of Office providers is managed using a Service API. This service will provide one of three services that make up the SquareDesk application. The 2 other APIs will be specified later. ProviderService is a static class and so has a single object in the system. This object is responsible in creating, updating and reading from Provider object.

Property Name	Type	Description
officeSpaces	officeSpaces:Map<String, OfficeSpace>	Private field to store associations to OfficeSpace. This is a hash map. The Map is from Provider's name to Provider's object. So the Provider Service API will lookup the Provider's object by presenting Provider's name. Firstly the name is converted to UUID and then a lookup is performed in this Map to get Provider's object. This data structure has to be kept in sync with Provider's name changes. The assumption made is that the Provider name is unique in the system. It is not reasonable to think that Provider's real name is unique, but if that is the case a "username" will be used which will be unique. This name corresponds to the username of the provider.

ratings	ratings:Map<String, Rating>	Private association to Rating table. Here also there is a UUID, but there is not natural key. Each rating is given a unique system generated id.
providers	providers :Map<String, Provider>	Private field to store list of all the providers. This is a Map data structure , where the key is uniquely generated provider Id and the value if Provider objects. A lookup is performed using the ProviderId to get Provider object. This providerId is the same field that's also present in Provider. ProviderId class. When Provider object is deleted, then this mapping has to delete from this Map also.
features	features :Map<String, Feature>	Private field to store list of all the features. Feature class uses flyweight design pattern and there is only one object of Feature class per feature text. All other class objects used
facilityTypes	facilityTypes :Map<String, FacilityType>	Description
categories	categories :Map<String, Category>	Description

Method Name	Signature	Description
updateProviderName	updateProviderName(String authToken, String providerId, String providerId, String name):Provider	Update the provider name if authorization token is valid. Authorization is handled in part 2 of this project
deleteProvider	deleteProvider(String authToken, String providerId):void, String providerId)	Delete the provider if authorization token is valid. Authorization is handled in part 2 of this project
addRatingToProvider	addRatingToProvider(String authToken, String providerId, String providerId, Rating rating)	Add rating to the provider if authorization token is valid.

		Authorization is handled in part 2 of this project
removeRatingFromProvider	,removeRatingFromProvider(String authToken, String providerId, String providerId, String ratingId)	Remove rating from provider if authorization token is valid. Authorization is handled in part 2 of this project

Operations

There are few exception classes also that only have a single property serviceVersionUID, which is used for serialization and de-serialization purposes. Following below are the Exception classes.
OfficeSpaceNotFoundException
RatingNotFoundException
ProviderNotFoundException
ProviderAlreadyExistException
AccessException
RatingNotFoundException

AccessException

Property Name	Type	Description
serialVersionUID	long	Used internally by JVM for serialization and de-serialization purposes

Account

Property Name	Type	Description
payPalAccountNumber	String	Private Pay Account number for the Provider Account to accept payment from the renter

Booking

Booking class is mainly to associate a renter with the office space. There are other attributes as well the give additional information for this association, for example start date, end date, rate, location, etc.

Property Name	Type	Description

renter	Renter	Private field to store renter object
officeSpace	OfficeSpace	Private field to store officeSpace object
period	String	Private field to store period of rental agreement
rate	Rate	Private field to store rate corresponding to the period of rental agreement
startDate	Date	Private field to store start date of the rental agreement
endDate	Date	Private field to store end date of the rental agreement
paymentStatus	String	Private field to store payment status on the rental agreement. For example, “paid”, “due” etc.
bookingId	String	Unique id in the system to identify booking. This is system generated using Java’s UUID feature

BookingAlreadyExistsException

Property Name	Type	Description
serialVersionUID	long	Used internally by JVM for serialization and de-serialization purposes

BookingNotFoundException

Property Name	Type	Description
serialVersionUID	long	Used internally by JVM for serialization and de-serialization purposes

BookingService

This class holds associations for bookings object. Each booking object turn holds association from renter to OfficeSpace. This way we can navigate from a renter to the officespaces the renter rents.

Property Name	Type	Description
bookings	HashSet<Booking>	Private association to booking objects. This is HashSet. We are overriding the ‘equals()’ and

		'hashCode()' method from the parent Object class. So we don't need a HashMap like we did in the 'renters' and 'providers' association objects. This will simplify the implementation a little bit
Method Name	Signature	Description
getUUIDFromString	(name:String):BookingService	Public method to get Unique Identifier for a Booking
createBooking	(Renter,officeSpace:OfficeSpace,period:String,reate:Rate,startDate:Date,endDate:Date,paymentStatus:String):Booking	Public method to create a booking.
checkAvailability	(officeSpace:OfficeSpace,startDate:Date,endDate:Date):boolean	Public method to check if the booking for the current officespace is available for those dates.
removeBooking	(booking:Booking):void	Public method to remove booking from the 'bookings' objects
listBookings	():void	Public method to list all the bookings currently available in the system

Criteria

Helper class to bundle a OfficeSpace search criteria. An implicit 'AND' is applied during search on the fields that this class has and the resultant OfficeSpaces are returned to the renter.

Property Name	Type	Description
features	List<Feature>	Private field to store features association objects.
location	Location	Private field to store location.
facility	Facility	Private field to store to store facility
minRating	int	Private field to store minRating. When minRating is used as a search criteria in the KnowledgeGraph, all the office spaces that have minimum average ratings of minRating will be returned. There is a minAverageRating field in the OfficeSpace class to store that information.

		That information has to be updated each time a rating is added to the OfficeSpace object.
startDate	Date	Private field to store startDate of the search criteria
endDate	Date	Private field to store endDate of the search criteria

ImportException

Property Name	Type	Description
serialVersionUID	long	Used internally by JVM for serialization and de-serialization purposes

KnowledgeGraph

Main Singleton class responsible for holding search related information. This class is not changed from that of Assignment 1. Same Triple and Node relations are kept intact. Even the data types.

Property Name	Type	Description
nodeMap	Map<String,Node>	Private association to store Node information in the KnowledgeGraph
predicateMap	Map<String,Node>	Private association to store Predicate information in the KnowledgeGraph
tripleMap	Map<String,Triple>	Private association to store Triple information in the KnowledgeGraph
queryMapSet	Map<String,Set<Triple>>	Private association to store denormalized Query mappings in the KnowledgeGraph. The query mappings are denormalize to provide O(1) running time. Though this would mean O(n^2) Memory
getTripleSet	(Set<Triple>,Triple):Set<Triple>	Public method to get Triple set from the Knowledge Graph for the input Triple.
addTriple	(triple:Triple):void	Public method to add a Triple to KnowledgeGraph
removeTriple	(triple:Triple):void	Public method to remove a

		Triple to KnowledgeGraph
executeQuery	(triple:Triple):Set<Triple>	Public method to execute query and return List<OfficeSpace> back to the renter.

Node

Property Name	Type	Description
identifier	String	Private identifier to store Node identifier

OfficeSpace (Continued)

Please see above for more attributes for OfficeSpace class. The new properties have been added as a part of this sprint.

Property Name	Type	Description
avgRating	Double	Private field to store averate ratings of this office space
isSearchable	():boolean	Private field to store if this officespace is searchable
setSearchable	(searchable:boolean)	Public method to make this OfficeSpace searchable

Predicate

Prediate class to store Predicate information.

Property Name	Type	Description
identifier	String	Description

Provider (Continued)

The following below are added to the Provider class for this sprint. Please see above for more information.

Method Name	Signature	Description
makeOfficeSpaceSearchable	(officeSpace:OfficeSpace):void	Public method to make officeSpace searchable
addOfficeSpaceToKnowledgeGraph	(officeSpace:OfficeSpace):OfficeSpace	Public method to add OfficeSpace to KnowledgeGraph
removeOfficeSpaceFromKnowledgeGraph	(officeSpace:OfficeSpace):OfficeSpace	Public method to remove OfficeSpace from

		KnowlegeGraph

QueryEngine

Class that runs the Query on the KnowledgeGraph.

Property Name	Type	Description
executeQuery	(filename:String):void	Public method to execute Query and return List<OfficeSpaces>

QueryEngineException

Property Name	Type	Description
serialVersionUID	long	Description

Renter

Renter class to perform renter operations - renter object communicated with RenterService API to perform rental operations.

Property Name	Type	Description
renterId	String	Private field to store renterId
gender	String	Private field to store gender
criteria	Criteria	Private field to store criteria

RenterAlreadyExistException

Property Name	Type	Description
serialVersionUID	long	Description

RenterNotFoundException

Property Name	Type	Description
serialVersionUID	long	Description

RenterService

This class is akin to ProviderService and provides rental related service.

Property Name	Type	Description
renters	Map<String,Renter>	Description
bookingService	BookingService	Description
getUUIDFromString	(name:String),String	Public method to get UUID from string

Method Name	Signature	Description
createRenter	(authToken:String,name:String,contactInfo:ContactInfo,picture:Image):Renter	Public method to create a renter
getRenterList	(authToken:String)	Public method to to list renters
updateRenterName	(authToken:String,renterId:String,name:String):Renter	Public method to update renter name
getRenterList	(renterId:String),Collection<Renter>	Public method to list renters
updateRenterName	(authToken:String,renterId:String,name:String):Renter	Public method to update renter name
updateRenterContactinfo	(authToken:String,renterId:String,contactInfo:ContactInfo):Renter	Public method to renter's contact information
updateRenterPicture	(authToken:String,renterId:String,picture:Image):Renter	Public method to renter's picture
deleteRenter	(authToken:String,renterId:String):void	Public method to delete renter
searchKGUsingFeatures	(authToken:String,freatures:ArrayList<Feature>):Collection<OfficeSpace>	Public method to search for officespaces using features
searchKGUsuingLocation	(authToken:String,location:Location):Collection<OfficeSpace>	Public method to search for officespaces using location
searchKGUsingFacilityAndCategory	(authToken:String,facility:Facility):Collection<OfficeSpace>	Public method to search for officespaces using facility and category
searchKGUsingRating	(authToken:String,minRating:int):Collection<OfficeSpace>	Public method to search for officespaces using Rating
searchKGUsingDates	(authToken:String,startDate:Date,endDate:Date):Collection<OfficeSpace>	Public method to search for officespaces using start and endDates
addRatingToRenter	(authToken:String,renterId:String,rating:Rating):Rating	Public method to add a rating
removeRatingFromRenter	(authToken:String,renterId:String,ratin	Public method to

	gId:String):Void	remove a rating
getRatingListForRenter	(authToken:String,renterId:String):Collection<Rating>	Public method to list ratings

Triple

Association between Node and Predicate

Property Name	Type	Description
subject	Node	Private field to store subject
predicate	Node	Private field to store predicate
object	Node	Private field to store object
identifier	Node	Private field to store identifier

TripleNotFoundException

Property Name	Type	Description
serialVersionUID	long	Used internally by JVM for serialization and de-serialization purposes

Profile

Abstract class for Provider and Renter

Property Name	Type	Description
getUUID	():String	Private field to store getUUID information
addRatingToList	(rating:Rating)	Public method to add Rating
getRatingFromList	(ratingId:String)	Public mehod to get Ratings
removeRatingfromList	(ratingId:String)	Public method to remove Ratings
contactInfo	ContactInfo	Private field to store ContactInfo
name	String	Private field to store name
picture	Image	Private field to store Picture
account	Account	Private field to store Account
ratings	Map<String,Rating>	Private field to store

		Ratings

Risks: 1) No DTO pattern has been applied yet, so RenterService, ProviderService and BookingService will all return the actual objects to the client. The client can modify these objects, which will break the encapsulation. The last sprint should address this concern.
2) KnowlegeGraph holds all combinations of associations in memory and has order polynomial memory requirement.

Testing: Testing has been made more modular in this sprint. There is a TestBaseDriver class, which is the Base class for other TestDriver classes. RenterService, ProviderService, BookingService, Renter, Provider all have their separate TestDriver classes. There is a single 'TestDriver.java' file where the 'main' function is defined that calls the other test classes.

REFERENCE and CREDIT

1) SnakeYaml is used for parsing renter.yaml and provider.yaml files.

2) Eclipse software features and plugins like JAutodoc are used for code implementation and documentation