SquareDesk Design Document

CSCI E-97 Assignment 3

SquareDesk Renter Service API

Date: October 30, 2014

Author: Philip Lin

Reviewer(s): S Alexander Zaman

Introduction

Sites and applications exist that allow people to share their homes, apartments, or other physical spaces that they have with others. This document describes a space sharing application of this sort called SquareDesk. The concept of SquareDesk is to provide a means for individuals to rent out places that they have as office space. The providers can make money from their unused space, and people looking for a place to work have an alternative workplace to traditional office spaces, bookstores, coffee shops or their own homes.

This document describes the implementation of the SquareDesk application – the definition of an API for modeling users who are renters, a search engine, and a scheduling service for the SquareDesk application.

Overview

Office space providers need a way to list their available spaces for rent, and renters need a way to search for, reserve, and pay for office spaces that suit their working needs and budget. SquareDesk provides a solution for this by giving providers a way to list their spaces with details about the space, and giving users a search engine to find spaces by dates/times needed, price, and other criteria. Additionally, SquareDesk has a secure system in place to create reservations and take care of payments.

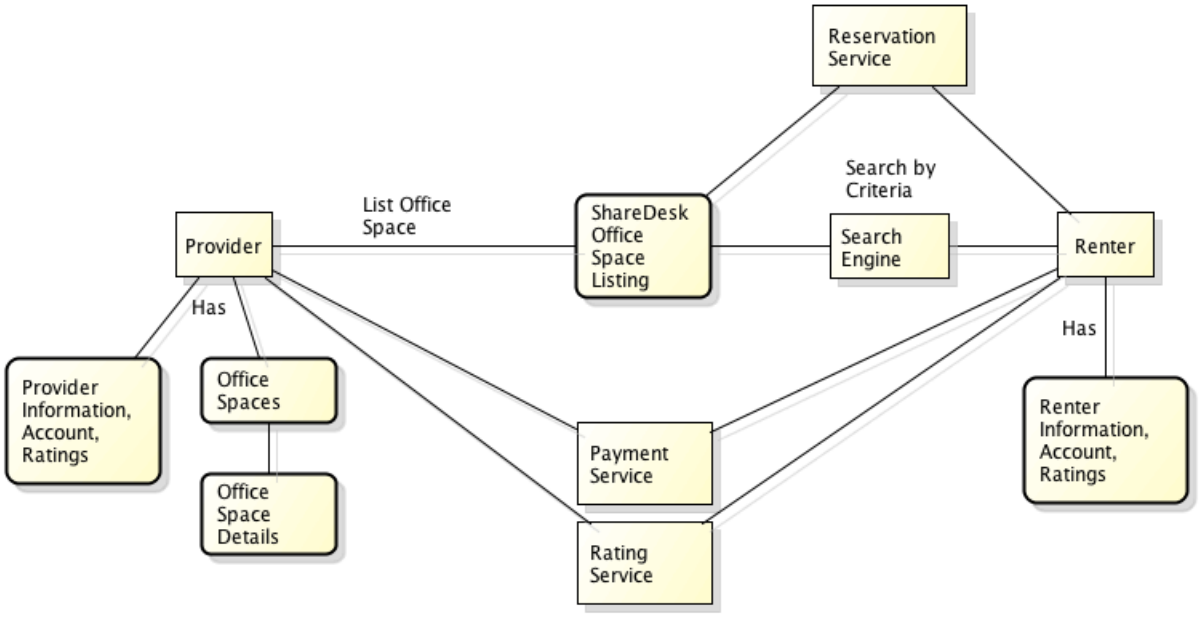

Figure: SquareDesk structure overview.

Since SquareDesk is the service that facilitates transactions between providers and renters, a small commission is charged for each payment made on a space reservation.

Demand for this service will exist because SquareDesk makes the whole process of listing/finding a space and making a monetary transaction easy, convenient, and fast.

Profits can be sustained as long as transactions continue to occur on the site, and the money earned from those transactions exceeds the costs needed to host and maintain the site.

Requirements

The Renter Service API should support the following functions:

1. Create, read, update, and delete instances of Renter.

   People who wish to rent office spaces register with SquareDesk, and an instance of Renter is created for each registered renter. Individual renter instances are created, retrieved, updated, and deleted by a manager within the SquareDesk application. Each Renter instance has a set of attributes and associations that contain data about the renter, which can be read and updated after the instance is created or when necessary.

   An authentication token is used to authenticate requestor and control access for the renter manager.

2. Support specification of the Renter details.

   A Renter is registered with the site and includes these properties and associations that can be accessed and updated: Identifier (GUID), Name, Contact Info, Picture, Ratings, and Account.

3. Support searching for and scheduling bookings for office spaces based on Renter search criteria.

   A Renter uses a search engine to search for available office spaces to rent. The search engine uses a set of pre-defined search criteria to search for spaces based on user input. For an office space to be returned in a search, it must match all the criteria the renter specifies. Renters can view and modify all the criteria the search engine defines. The criteria include these options: Preferred Features (any number of features can be searched), Location, Facility Type and Room, Minimum Average Rating, Start Date, End Date, and Cost. A scheduling service is used to reserve office spaces for a specified starting and ending date. An office space may only be reserved if it is available for the dates requested, or in other words if no other bookings that overlap the requested period exist.

Use Cases

The SquareDesk application supports 2 primary use cases:

1. Provider

Office space providers go to the SquareDesk site, register themselves and their available spaces, and provide details about themselves as well as their available spaces. SquareDesk then connects them with prospective renters, and the provider will receive rent payments from the renter minus the 10% commission SquareDesk gets for each transaction. Providers can then rate the renters they work with.

2. Renter

Renters looking for an office space go to the SquareDesk web site, and search for office spaces using a specific set of search criteria. SquareDesk then finds the available spaces that meet the renter's criteria. The renter chooses the space that best fits their needs, and reserves and pays for the space through the SquareDesk payment system. Renters can rate the providers as well as the office space.

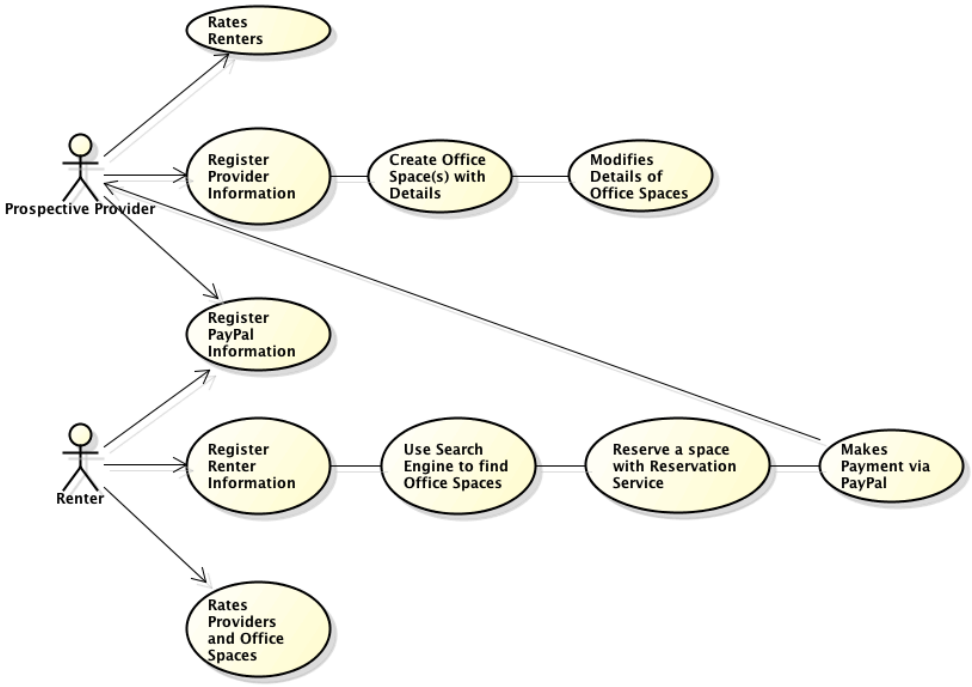

Figure: Use Case Diagram.

Implementation

Class Diagram

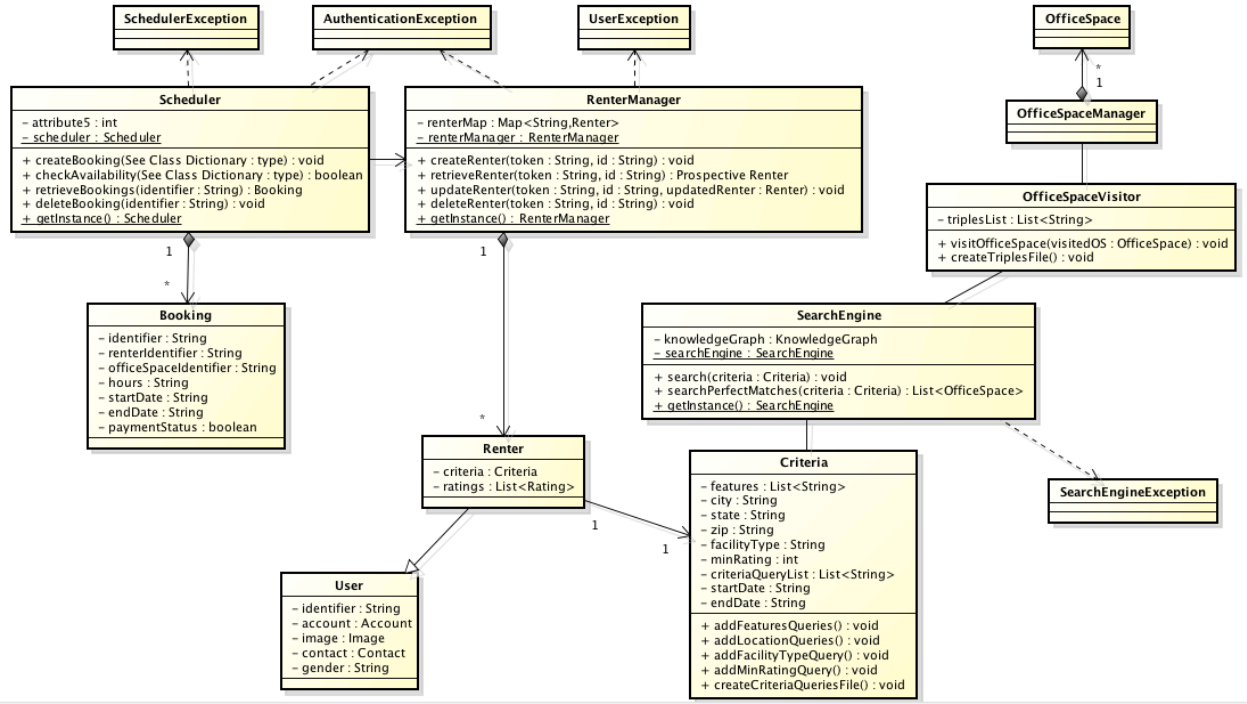

Figure: Class Diagram.

Class Dictionary

This section specifies the class dictionary for the ShareDeskOffice Space Provider API. The classes should be defined within the package “cscie97.asn2.sharedesk.provider”.

Booking

The Booking class is used to define an object that contains information about an office space reservation between a renter and a provider.

Properties

Property Name	Type	Description
identifier	String	Private identifier for the booking.
renterIdentifier	String	Private renter this booking is for.
officeSpaceIdentifier	String	Private office space this booking is for.
hours	String	Private hours of the day for the booking.
startDate	String	Private start date of the booking.
endDate	String	Private end date of the booking.
paymentStatus	boolean	Private indicator of payment status for the booking.

Criteria

The Criteria class is used to define an object that contains search criteria for looking up available office spaces using the search engine. The criteria are stored as a list of different String queries to be executed against the knowledge graph of the search engine.

Methods

Method Name	Signature	Description
addFeaturesQueries	() : void	Adds queries for desired features to the list of criteria queries.
addLocationQueries	() : void	Adds queries for desired location information to the list of criteria queries.
addFacilityTypeQuery	() : void	Adds query for desired facility type to the list of criteria queries.
addMinRatingQuery	() : void	Adds query for desired minimum rating to the list of criteria queries.
createCriteriaQueriesFile	() : void	Creates text file used to import renter criteria queries to check against the knowledge graph.

Properties

Property Name	Type	Description
features	List<String>	Private list of all desired features of the office space.
city	String	Private city desired for the office space.
state	String	Private state desired for the office space.
zip	String	Private zip desired for the office space.
facilityType	String	Private facility type desired for the office space.
minRating	double	Private minimum rating desired for the office space.
startDate	String	Private start date desired for the office space.
endDate	String	Private end date desired for the office space.

OfficeSpaceVisitor

The OfficeSpaceVisitor class defines a visitor pattern object that visits individual office space instances and uses the office space attributes to get and store triples to add to the knowledge graph of the search engine.

Methods

Method Name	Signature	Description
visitOfficeSpace	(visitedOS : OfficeSpace) : void	Visits the office space instance and adds triples to a list to be used to add triples to the search engine knowledge graph.
createTriplesFile	() : void	Creates N-Triples file used to import attribute Triples of the office space into the search engine knowledge graph.

Properties

Property Name	Type	Description
triplesList	List<String>	Private List of all triples obtained using the attributes of all OfficeSpace instances visited.

RenterManager

The RenterManager class manages a map of all the Renter objects, as well as the create, retrieve, update, and delete operations for instances of Renter. Only one instance of RenterManager is used in the program (singleton pattern).

Methods

Method Name	Signature	Description
createRenter	(token : String, id : String) : void	Creates a new Renter object, as well as the objects that are fields of or associated with Renter.
retrieveRenter	(token : String, id : String) : Renter	Returns the Renter object identified by the input identifier from the map of renters
updateRenter	(token : String, id : String, updatedRenter : Renter) : void	Updates the Renter object identified by the input identifier from the map of renters using a new renter DTO.
deleteRenter	(token : String, id : String) : void	Deletes the Renter object identified by the input identifier from the map of renters.
getInstance	() : RenterManager	Static method that returns the singleton instance of the RenterManager.

Properties

Property Name	Type	Description
renterMap	Map<String, Renter>	Private Map maintaining a list of all registered instances of users who are renters.
renterManager	RenterManager	Private static single instance of RenterManager.

Renter

The Renter class represents a Renter with certain attributes as defined in the User class, as well as an association class called Criteria that describes the queries to be used in the search engine to find office spaces.

Properties

Property Name	Type	Description
criteria	Criteria	Private Criteria object that is used to define search criteria for looking up available office spaces in the form of String queries.
ratings	List<Rating>	Private List maintaining a list of all ratings received for an office space from renters who have rented it.

Associations

Association Name	Type	Description
user	User	Parent class of Renter, with definitions for attributes of both Renter and Provider objects.

Scheduler

The Scheduler class is responsible for creating and managing reservations for a renter to rent an office space that a provider has listed, if that office space is currently available. The Scheduler can also remove reservations when they are over or no longer needed. Only one instance of Scheduler is used in the program (singleton pattern).

Methods

Method Name	Signature	Description
createBooking	(token : String, id : String,	Creates a Booking object if all conditions

	renterIdentifier : String, officeSpaceIdentifier : String, hours : String, startDate : String, endDate : String) : void	for creating a booking are met. All fields of the booking are required to be set upon creation.
checkAvailability	(token : String, officeSpaceIdentifier : String, startDate : String, endDate : String) : boolean	Checks the availability of an office space for start and end dates through the office space manager. Returns a boolean indicating the availability.
retrieveBooking	(identifier : String) : void	Retrieves a Booking object from the Map of current bookings.
deleteBooking	(identifier : String) : void	Deletes a Booking object from the Map of current bookings.
getInstance	() : Scheduler	Static method that returns the singleton instance of the Scheduler.

Properties

Property Name	Type	Description
bookingsMap	Map<String, Booking>	Private Map of all Booking objects in the ShareDesk system.
scheduler	Scheduler	Private static single instance of Scheduler.

SchedulerException

Exception class for handling errors that originate from the scheduler.

Methods

Method Name	Signature	Description
getExceptionDetails	() : void	Returns details associated with the exception.

Search Engine

The Search Engine class leverages the KnowledgeGraph API to implement a method of searching for office spaces using any number of input String queries. Only one instance of SearchEngine is used in the program (singleton pattern).

Methods

Method Name	Signature	Description
search	(criteria : Criteria) : void	Uses the visitor to add triples from all office spaces to the knowledge graph, and uses the criteria to create and execute queries against the knowledge graph, printing out results of each individual criteria query.
searchPerfectMatches	(criteria : Criteria) : List<OfficeSpace>	Returns a list of the office spaces that match all the input criteria from the renter.
getInstance	() : SearchEngine	Static method that returns the singleton instance of the SearchEngine.

Properties

Property Name	Type	Description
knowledgeGraph	String	Private unique non-mutable identifier that is used to identify individual office spaces. Format is 8 digit number.
searchEngine	SearchEngine	Private static single instance of the SearchEngine.

SearchEngineException

Exception class for handling errors that originate from the search engine.

Methods

Method Name	Signature	Description
getExceptionDetails	() : void	Returns details associated with the

		exception.

User

The User class represents a SquareDesk user with certain attributes. Users can be further abstracted into Providers and Renters.

Properties

Property Name	Type	Description
identifier	String	Private unique non-mutable identifier that is used to identify individual office spaces. Format is 8 digit number.
gender	String	Private unique non-mutable identifier that is used to specify user gender.
account	Account	Private object used to define the account information of the renter, including PayPal account number.
contact	Contact	Private object used to define contact information, which is the name, phone, and email of the renter.
image	Image	Private object used to define the profile image of the renter, with the image in URI form.

UserException

Exception class for handling errors with the CRUD operations for users (providers and renters).

Methods

Method Name	Signature	Description
getExceptionDetails	() : void	Returns details associated with the exception.

Implementation Details

CRUD Operations for Renter Objects

The create, retrieve, update, and delete operations for individual instances of Renters are defined in the RenterManager class.

When a Renter object is initially created with the create method, these objects associated with Users are also created in the constructor:

Identifier

Account

Image

Contact

List of Ratings

Criteria

These objects created in the constructor initially contain empty fields, but are updated with actual information by using class setters and the retrieve and update methods of the manager classes.

In this way, correct information can be added to instances of a Renter object once that instance has been created.

Sequence Diagram for Criteria-based Search

The process of using the search engine to obtain a list of the available office spaces that fit a certain criteria is shown in the sequence diagram below. A renter will set the criteria to be used in the search engine, which manifests itself as a criteria object. The criteria object passes a list of queries to the knowledge graph of the search engine, which had previously loaded triples describing all available office space objects prior to the search. The queries are checked against the triples, and all office space objects that match all the triples are returned as the results of the search.

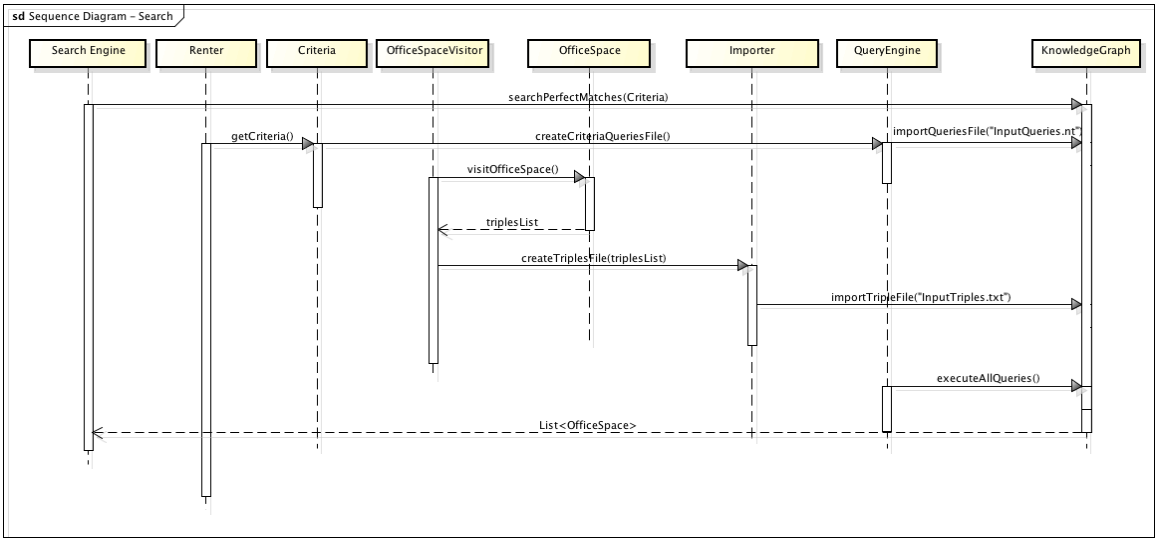

Figure: Sequence Diagram for Criteria-based Search.

Sequence Diagram for Booking Spaces via the Scheduler

The process of booking an office space is done through the scheduler service. The scheduler checks the availability of the office space in the system, as each office space has a field that keeps track of the dates that is booked for. If an office space is available to be booked, the scheduler can create a new booking if it has the renter, office space, dates, hours, and payment status ready to create the booking. All fields of a booking must be set when creating a booking.

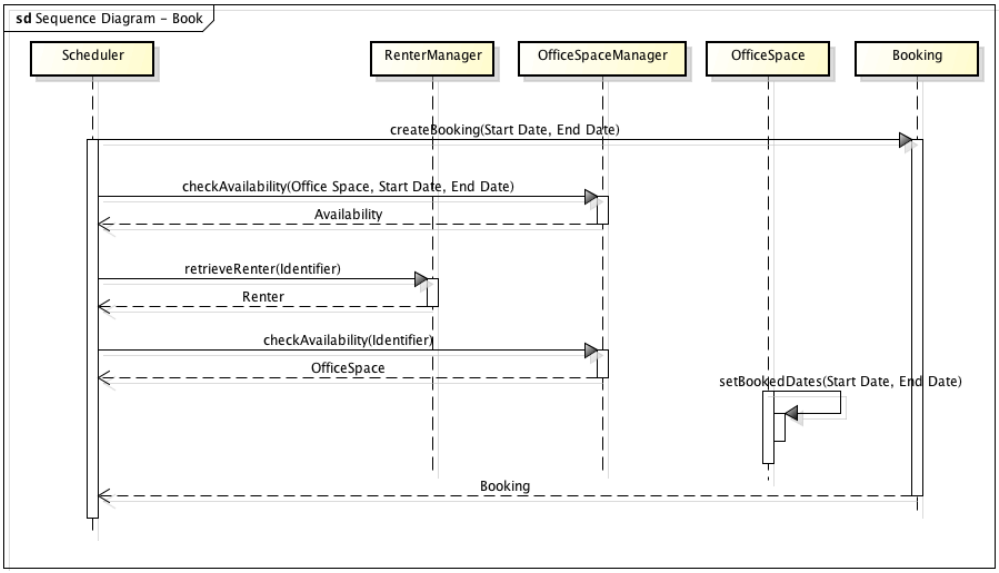

Figure: Sequence Diagram for Booking Office Spaces.

Testing

A test driver class called TestDriver, which includes a main method, is used to test the functionality of the API. The test driver should be able to create sample Renter instances as well as retrieve, update, and delete them as well.

The test driver should create different Criteria instances that are associated with Renters. These criteria will be tested in the search engine to ensure that the proper office spaces that match the criteria are being returned.

Also, the Scheduler service should be able to create and delete Booking instances for renters renting office spaces.

Exceptions should also be handled for authentication errors, as well as errors when creating, retrieving, updating, or deleting Renter objects as well as Booking objects. These errors happen when objects don't exist when called for, or when attributes of objects are improperly edited or accessed. Exceptions with the KnowledgeGraph are defined in the KnowledgeGraph API.

The TestDriver class should be defined within the package "cscie97.asn3.test".

Risks

Because of the in-memory implementation, the number of Renters and Bookings is limited by the memory allocated to the JVM. Renters and Bookings also must be implemented and updated properly as to avoid making references to objects or fields that don't exist, something that can be checked before any operations are actually done. Authentication must be done in order to ensure that any changes to the system are authorized and account information is correct and usable.