**Place Class**

String name

double location

String address

String description

String[] tags

Road road

double estTimeCommitment

Place(String name, double location, String address, String description, String[] tags, Road road, double estTimeCommitment)

this.name = name

this.location = location

this.description = description

this.tags = tags

this.road = road

estTimeCommitment = null

setName(String name)

this.name = name

getName()

return name

setLocation(double location)

this.location = location

getLocation()

return location

setDescription(String description)

this.description = description

getDescription()

return description

getTags():

return tags

setTags(String[] tags)

this.tags = tags

getAddress()

return address

setAddress(String address)

this.address = address

getEstTimeCommitment()

return estTimeCommitment

setEstTimeCommitment(double timeCommitment)

estTimeCommitment = timeCommitment

associateAddressCoordinate(address, Place)

Map.get(address) //hashmap, address maps to a Place

setRoad(Road road)

this.road = road

getRoad()

return road

public void destroy()

destroy object

**Road Class**

String name

double distanceOnRoad

int trafficEstimate

int speedLimit

Road connectsTo

Road(String name, double distanceOnRoad, int trafficEstimate, int speedLimit, Road connectsTo)

this.name = name

this.distanceOnRoad = distanceOnRoad

this.trafficEstimate = trafficEstimate

this.speedLimit = speedlimit

this.connectsTo = connectsTo

timeLeft = distanceOnRoad / speedLimit + trafficEstimate

setTime()

timeLeft = distanceOnRoad / speedLimit + trafficEstimate

getTime()

return time

setName(String name)

this.name = name

getName()

return name

setDistanceOnRoad(int distanceOnRoad)

this.distanceOnRoad = distanceOnRoad

getDistanceOnRoad()

return distanceOnRoad

setTrafficEstimate(int trafficEstimate)

this.trafficEstimate = trafficEstimate

getTrafficEstimate()

return trafficEstimate

setSpeedLimit(int speedLimit)

this.speedLimit = speedLimit

getSpeedLimit()

return speedLimit

getConnectsTo()

return connectsTo

setConnectsTo(Road connectsTo)

this.connectsTo = connectsTo

destroy()

destroy object

**Navigation Class** //client class, doesn’t need to be instantiated

Place destination

int timeEstimate

double distance

Road[] route

setRoute(Road[] newRoute)

route = newRoute

timeEstimate = 0

distance = 0

for each road in route

timeEstimate += road.getTime()

distance += road.getDistanceOnRoad()

getRoute()

return route

setTimeEstimate(int timeEstimate)

this.timeEstimate = timeEstimate

getTimeEstimate()

return timeEstimate

getDistance()

return distance

setDistance(double distance)

this.distance = distance

setDestination(Place newDestination)

destination = newDestination

getDestination()  
 return destination

destroy()

destroy object

**Map Class**

Place[] places

int clientRenderDistance

String region

double[] regionalCoordinates

\\Determines effectively the distance between points on the grid of the map,

Scaled up or down dependent on urban density.

double endpointCoordinate

getEndpointCoordinate()

return endpointCoordinate

setEndpointCoordinate(double coord)

endpointCoordinate = coord

populateMap(Place[] places, locationTags[] tags)

for each Place in places

if places distance from client < clientRenderDistance

render place

searchPlaces(String searchedFor)

for each place in places

if place == searchedFor

return place

displayUserLocation()

render client.getProfilePicture() at client.getLocation()

displayRoads()

for each road in navigation.getRoute()

render road

generateRegionalCoordinates(region)

regionalCoordinates = region

createCoordinateAlley(location.clientuser, location.place)

create a route of roads between the location of the client and specified place

Navigation(route)

avgCoordinates(double coordinate1, double coordinate2)

return coordinate1 / coordinate2

regionLookup(String client.location)

return region of client

destroy()

destroy object

**Class Client Abstract**

//not to be confused with Navigation class that acts as a client for the user to interact with, this is a class to store user information

String firstName

String lastName

String dateOfBirth

String username

String password

double contactPhone

String emailAddress

int renderDistancePreference

file profilePicture

double location

locationPermissions: Boolean

verifyLogin(String username, String password, String emailAddress)

return this.username == username and this.password == password and this.emailAddress = emailAddress

getFirstName()

return firstName

setFirstName(String newName)

this.firstName = newName;

getLastName()

return lastName

setLastName(String newName)

this.lastName = newName

getDateOfBirth()

return dateOfBirth

setDateOfBirth(String newDate)

dateOfBirth = newDate

getUsername()

return username

setUsername(String newUsername)

username = newUsername

//No get password, shouldn't be accessed outside this class

setPassword(String oldPassword, String newPassword):

if oldPassword == password

password = newPassword

getContactPhone()

return contactPhone

setContactPhone(double newPhone)

contactPhone = newPhone

getEmailAddress()

return emailAddress

setEmailAddress(String newAddress)

emailAddres = newAddress

getProfilePicture()

return profilePicture

setProfilePicture(file newPicture)

profilePicture = newPicture

getLocation()

return location

setLocation(double newLocation)

location = newLocation

getRenderDistancePreference()

return renderDistancePreference

setRenderDistancePreference(int distance)

renderDistancePreference = distance

getLocationPermissions(): Boolean

return locationPermissions

setLocationPermissions(boolean permission)

locationPermissions = permission

destroy()

destroy object

**Class** **ClientBusiness extends Client**

int operationHours

String businessName

String businessType

String address

double businessPhone

clientBusiness(int operationHours, String bussinessName, String type, String address, double phone)

this.operationHours = operationHours

this.bussinessName = bussinessName

businessType = type

this.address = address

businessPhone = phone

setBusinessName(String name)

businessName = name

getBusinessName()

return businessName

getBusinessType()

return businessType

setBusinessType(String type)

businessType = type

getOperationHours()

return operationHours

setOperationHours(int hours)

operationHours = hours

getBusinessAddress(): String

return businessAddress

setBusinessAddress(String address)

businessAddress = address

getBusinessPhone(): double

return businessPhone

setBusinessPhone(double phone)

businessPhone = phone

destroy()

destroy object

**Class** **ClientUser extends Client**

Boolean visibility

ClientUser(Boolean visibility)

this.visibility = visibility

getVisibility(): Boolean

return visbility

setVisbility(Boolean visbility)

this.visibility = visibility

destroy()

destroy object

**Class Event extends Place**

String eventName

String[]eventTags

Boolean inviteOnly

ClientUser[] guestList

Boolean ageRestricted

event(String name, String[] tags, Boolean inviteOnly, ClientUser[] guests, Boolean restricted)

eventName = name

eventsTags = tags

this.inviteOnly = inviteOnly

guestlist = guests

ageRestricted = restricted

getEventName()

return eventName

setEventName(String name)

eventName = name

getEventTags()

return eventTags

setEventTags(String[] tags)

eventTags = tags

getGuestList()

return guestList

setGuestList(ClientUser[] guests)

guestlist = guests

getAgeRestricted()

return ageRestricted

setAgeRestricted(Boolean restricted)

ageRestricted = restricted