REST API Description

Bancho

- GET:
  - api/person/id (returns the person with the given ID)
  - api/person/byphone/phone (returns the person with the specified phone)
  - api/person/byhobby/countPeople/hobby (returns the number of people with the specified hobby)
  - api/person/byhobby/hobby (returns the person with the specified hobby)
  - api/address/id (returns the address with the given ID)
  - api/company/id (returns the company with the given ID)
  - api/company/byphone/phone (returns the company with the specified phone)
  - api/company/emplAbove/num (returns a list of companies who have more than the specified number of employees)
  - api/hobby/id (returns the hobby with the given ID)
  - api/phone/id (returns the phone with the given ID)
- POST (all post methods create a new object; call must include the JSON object to be created, returns the created object with assigned ID to it):
  - api/person
  - api/address
  - api/company
  - api/hobby
  - api/phone
- PUT (all put methods change an existing object; call must include the JSON object to be edited; returns the edited object):
  - api/person
  - api/address
  - api/company
  - api/hobby
  - api/phone
- DELETE (all delete methods delete an existing object with the specified ID; returns the deleted object):
  - api/person/id
  - api/address/id
  - api/company/id
  - api/hobby/id
  - api/phone/id

Entity classes

Bancho

We chose to first create the Entity classes and then use them to generate the tables. We did that together on my computer. We decided to use the singe table strategy to handle inheritance because it is the default way to do it and we already had some experience with it.

Test Data Generator

Bancho

I created a Test Data Generator class that populates the tables in the database with random information.

In order to work, it relies on the fact that the City Info populating script has been executed first. The class generates 100 people (ID's 1 to 100) and 100 companies (ID's 101 to 200).

Database Facades

Bancho

I worked on the Info Entity facade. Apart from the CRUD methods, I also created the getPersonByPhone(). The way it works is that it looks for an Info Entity with the specified phone number. If it doesn't find anything, it throws a PersonNotFoundException(). If it does find something, it checks whether is is a person or a company. If it is the latter, it throws a PhoneDoesNotBelongToPersonException().

Alex

I made the CityInfoFacade and I also worked on the Info Entity Facade adding the getCompanyByPhone method and getPeopleWithHobby.The exceptions which may be thrown are CompanyNotFoundException and PhoneDoesNotBelongToCompanyException..

Martin

After some pair-programming of entity classes, setting up Git-hub and creating our own database connections and persistence units, we have decided that we will split facades into five different java classes, so each of us can work at the same time on different classes. My responsibility was to make PhoneFacade and AddressFacade. While programing them I didnt have any serious problems since we have practised it a lot during last Fridays exercises. Simoultaneously with this part I also created some exception classes as PhoneNotFoundException, HobbyNotFoundException as well as AdressNotFoundException.

JUnit and RESTAssured Tests

Bancho

Due to lack of time, we only implemented one test, using each framework, that only test a limited amount of methods. In order for the JUnit test to work, the Test Data Generator needs to be executed first.

Deploying on OpenShift

Bancho

Since we don't have a script that populates the tables, but a Java class instead, I made it, so that every time the server is launched on OpenShift it uses that file to populate the tables with Random data.

How to test the system

Since as previously mentioned, we do not have extensive test cases, the best way to test the system would be via Postman, using the REST API provided.

Javascript

Alex

I worked on the javascript and html files. Unfourtunately,
Admin.js is containing [ADD a PERSON,GET a PERSON, EDIT a PERSON and DELETE a PERSON (CRUD).] but only add and get functions work. Working on the admin took time because I had the Unsupported media type Error. It occurred because I sending the data of the inputs properly. But soon enough I made a person object from the inputs and then parse it to a json.

peopleWithHobby.js—[GET List of People with hobby. Here I had a problem with the database and the query altough it seemed right.]

These files have not been completely finished due tue lack of time. It could have been much better if we had a day more for the front-end.

FindACompany.js – It sends the input field information in url using get method and It changes and adds the information to a table.

Martin
For a web pages we have decided to use a bootstrap main reasons for that were that we are familliar with the enviroment and we have also saved a lot of time. My part of the front end side of the project was index.html,main.js and some minor details on the other webpages (e.g. nav buttons). In my javascript file I use a function where I request the getPersonByPhone() through the ajax. While coding this part a had a small difficulty with 500 error code. The problem was solved by deleting redundant method(which was fogotten that is there) in api. Source of the problem was, that we had two methods which used same path(CA2/api/person/byphone/{number}) so server was confused which one to choose. I have consulted the whole process of coding main.js with my team mate Bancho Petrov.

Security

Alex
We wanted to add security constraints for the admin page as well as login confirmation (Basic authentication) but we chose to spend our time in the things that we think are harder. Afterall we all knew how to add user roles and basic authentication.

REST APIs

Martin
As a next step after the facades and exceptions I started to work on PhoneApi and HobbyApi. In a hobby api I faced a problem with an deleteHobby() . After reading output both from postman and netbeans we thought the problem is located in API but after a while without a results we decided to consult our problem with a teacher who found the problem in a HobbyFacade, the faulty part was a usage of two different entity managers in one method. After this minor error I didnt face any other problems in my part of the API.
Alex
I spent time doing the PersonApi, more specifically the getPeopleWIthHobby and getCountOfPeopleWithHobby. Although it retrieves an int and we do not use it...

I also did the method getCompanyByPhone in the CompanyApi.I did not have problems with the Api's as we practiced it a lot during the SP exercises.

Overall experiences during the proccess

Martin

At the beginning we had the same problems with Git-hub as we faced in previous CA. After few commits we checked graphs of commits on a github.com and we could see that some of the collaborators didnt have any commits counted although we committed several times. The root of the problem was that in the Git-hub applications on our machines we didnt set up our emails. Since then every commit was counted but previous ones were lost forever (they can be manually checked in a project history).

Collaboration with other team members

Martin

From my point of view our collaboration was on a very good level, everyone did what he had to and in a stated time. We didnt have any problems with meetings or splitting the work between three of us and last but not least when someone needed help because he was under time pressure, everyone was willing to give a hand and actually helped him. Considering those facts in my opinion evereyone should get full points for his part even we know our project has some problems.

Error responses and JSON format

Martin

AddressNotFoundException()

Occurrence: getAddress() -returned when requested adress does not exist
deleteAddress()-returned when requested adress does not exist
editAdress() - returned when requested adress does not exist
Code: 404
Message: Address with requested id not found

CompanyNotFoundException()

Occurrence: getCompany()-returned when requested company does not exist
deleteCompany()-returned when requested company does not exist
editCompany()- returned when requested company does not exist
getCompanyByPhone()-returned when requested company does not exist
Code: 404
Message: The Company you requested has not been found

HobbyNotFoundException()

Occurrence: getHobby()-returned when requested hobby does not exist
deleteHobby()-returned when requested hobby does not exist
editHobby()- returned when requested hobby does not exist
Code: 404
Message: There is no person with such hobby

PersonNotFoundException()

Occurrence: getPerson()-returned when requested person does not exist
deletePerson()-returned when requested person does not exist
editPerson()- returned when requested person does not exist
getPersonByPhone()-getPerson()-returned when requested person does not exist
Code: 404
Message: There is no person with the info

PhoneNotFoundException()

Occurrence: getPhone() -returned when requested phone does not exist
deletePhone()-returned when requested phone does not exist

Code: 404
Message: Phone with requested id not found

PhoneDoesNotBelongToCompanyException()

Occurrence: getCompanyByPhone()-returned when the input phone number is not connected to any of the companies
Code: 400
Message: There is no company with the provided phone

PhoneDoesNotBelongToPersonException()

Occurrence: getPersonByPhone()-returned when the input phone number is not connected to anyone
Code: 400
Message: There is no person with the provided phone

NonUniqueResultException()

Occurrence: getPersonByPhone()-Returned when we ask for a Person by Phone and we have more than one result
getCompanyByPhone()-Returned when we ask for a company by a Phone and we have more than one result
Code: 404
Message: NonUniqueResultException

Json format

Martin

Function: Find person

Method: getPersonByPhone(@PathParam("phone") String phone)
http://localhost:8080/CA2/api/person/byphone/{phone}

Function: Find Company by size

getCompaniesEmplAbove(@PathParam("num") String num)
http://localhost:8080/CA2/api/company/{numberOfEmployees}

Find a company

getCompanyByPhone(@PathParam("phone") String phone)
http://localhost:8080/CA2/api/company/byphone/{phone}

Find people with hobby

getPeopleWithHobby(@PathParam("hobby") String hobby)
http://localhost:8080/CA2/api/person/byhobby/{hobby}

Link to GitHub
https://github.com/bancho22/CA2