Test Plan Document

Author: Hamad Almarri (SID 200294236)

University Of Regina

Final Year Project (2014-2015)

Project Name: Adala

Supervisor: Abdul Bais

# 1. Introduction

The purpose of this test plan is to ensure the quality of the software where the plan will cover unit testing for Adala’s back-end units (i.e. java classes) and will cover integration test, which applied on the front-end pages where every test examine the flow of pages and also the input validation. This plan follows 80%-20% rule, which is about test 80% of the software using 20% of resources, and the resources we have at this level is time excluding budget.

# 2. System Overview

The system is a database-driven architected where it has back-end classes for controlling and connecting web pages as front-end with the database. The core of the system is the CRUD functionality where the entries are stored and restored from the database. The second part that to be tested is the flow of managing the investigative cases from creating them, going through assigning people to work on them, and update the cases. The last part for testing is the security, authentication, and authorization of system functionality, and this part includes users’ login and users’ roles. The last part is the artificial neural network where it has been used in generating offender profiles from crime scenes.

# 3. Test Items (Functions)

## 3.1. Entity Beans Testing

Critical entity beans will be tested to ensure no invalid data will be stored into the database. The units that are tested are the ones that need to be verified or validated before being stored, but the basic getters and setters will not be tested. The entities are showed below in hierarchy tree: -

├── entries

│   ├── Address.java

│   ├── Conveyance.java

│   ├── Person.java

│   ├── PersonName.java

│   ├── PhysicalCharacteristic.java

│   ├── ThreatAssessment.java

│   ├── contacts

│   │   ├── CellPhone.java

│   │   ├── Contact.java

│   │   ├── Email.java

│   │   └── Telephone.java

└──AccessManager.java

## 3.2. Investigative Case Management Testing

A web test tool will be used to test the workflow of the investigative case management from initiated a case till closing it. The test will include check points assertions for assigning investigators, accepting or rejecting the requests, and the notifications that are send by the actions on the investigative cases. The scenario that will be tested in this plan is the adding new person to the system scenario.

## 3.3. System Security and Access Control Testing

This testing is not purely a security test. It doesn’t include SQL, or HTML injections checks. It’s only testing the user login functionality, users’ authentication, and users’ authorization. The same web test tool will be used for this part too.

## 3.4. Artificial Neural Network Testing

ANN needs to be trained with training set, and in order to make training phase successful, ANN need to be tuned. The ANN tuning will be accomplished with using jUnit to ensure that all outputs are close to optimal.

# 4. Feature To Be Tested

The things that this test plan is planned for are mainly the inputs **validation** to preserve the consistency of stored data such that no birth date should be stored that is a future date from the time the data is entered, and the type of inputs where no numbers in persons’ names etc. The next part is the **cases management** where assigning investigators to cases with proper notifications should work properly. The third portion is the **system security**.

# 5. Feature Not To Be Tested

The browsers’ compatibility and different client operating system will not be tested. As long as the web pages are working on the developer browser (which is Firefox) and on the developer machine/s which are MacOS v10.9.5 and Debian Jessie on ThinkPad laptop, the software will be consider is working well at the time being. However, browsers’ compatibility should be tested on future testing plans along with different mobiles/tablets screens. The reason of not doing this test is the lack of resources and time at this point.

# 6. Approach

## 6.1. Tools

* jUnit4 for testing back-end classes and entities. <http://junit.org>
* Webtest from canoo. <http://webtest.canoo.com>

## 6.2. Training

The tools are easy to use and they do not require training. Their documentations are clear and they have nice examples to start with.

## 6.3. Methods of Testing

Black box method will be used in this plan following three test design methods, which are: -

* Equivalent Classes
* Boundary Value Analysis
* Cause and Effect Graph
* For section 9.2. the steps are designed for webtest tool where it will run the steps automatically and show results of which of the tests and steps pass/failed

# 7. Item Pass/Fail Criteria

* 80% of unit tests be completed
* 80% of unit tests pass
* 85% of web tests pass

# 8. Suspension Criteria And Resumption Requirements

If more than 40% of unit tests fails, then no need to continue other tests unless these bugs are fixed. At least 60% of unit tests pass to continue doing web tests. However, the unit tests must pass 80% at the end of this testing phase.

# 9. Test Cases

## 9.1. Unit tests

### Class: entities.entries.Person

Methods under test: -

* setDateOfBirth
* setBirthPlace
* setGender

### setDateOfBirth

Assuming current date is March 5th, 2015

Rules:

* Date must be less than or equal to current date
* (Date – current date) >= 150 (max age is 150 years old)

|  |  |  |  |
| --- | --- | --- | --- |
| TC# | Input | expected output | method |
| 1 | 21/05/1991 | valid | E.C. |
| 2 | 02/11/1200 | error | E.C. |
| 3 | 04/04/2016 | error | E.C. |
| 4 | 05/03/2015 | valid | BVA |
| 5 | 04/03/2015 | valid | BVA |
| 6 | 06/03/2015 | error | BVA |
| 7 | 05/03/1865 | valid | BVA |
| 8 | 04/03/1865 | error | BVA |
| 9 | XX/XX/1864 | valid | BVA |

### setBirthPlace

Rules: -

* Valid
  + Must be just text value
  + White spaces are ok
  + 25 character or less is ok
  + More than or equal 3
* Invalid
  + Digits
  + Special characters
  + Larger than 25 characters
  + Null

|  |  |  |  |
| --- | --- | --- | --- |
| TC# | input | expected output | method |
| 1 | Regina | valid | E.C. |
| 2 | Al Hasa | valid | E.C. |
| 3 | paris | valid | E.C. |
| 4 | Regina (the queen city) | error | E.C. |
| 5 | abcabcabcabcabcabcabcabcab | error | BVA |
| 6 | abcabcabcabcabcabcabcabca | valid | BVA |
| 7 | abcabcabcabcabcabcabcabc | valid | BVA |
| 8 | Al | error | BVA |
| 9 | T | error | BVA |
| 10 | Tai | valid | BVA |
| 11 | 123 | error | E.C. And BVA |
| 12 | 1@34 | error | E.C. And BVA |
| 13 | null | error | E.C. |
| 14 | Toronto 3215 | error | E.C. |
| 15 | Ottawa &%^ | error | E.C. |
| 16 | @#$ | error | E.C. |
| 17 | ! | error | E.C. |

### setGender

Rules:

* Valid
  + Strictly either "Male" or "Female" Case is not sensitive
* Invalid
  + Anything else not "Male" or "Female"
  + Null

|  |  |  |  |
| --- | --- | --- | --- |
| TC# | input | expected output | method |
| 1 | Male | valid | E.C. |
| 2 | male | valid | E.C. |
| 3 | Female | valid | E.C. |
| 4 | female | valid | E.C. |
| 5 | M | error | E.C. |
| 6 | m | error | E.C. |
| 7 | F | error | E.C. |
| 8 | f | error | E.C. |
| 9 | null | error | E.C. |
| 10 | Male Female | error | E.C. |
| 11 | Female Male | error | E.C. |
| 12 | "" | error | E.C. |
| 13 | 123 | error | E.C. |
| 14 | #$%# | error | E.C. |
| 15 | f e m a l e | error | E.C. |
| 16 | m a l e | error | E.C. |

### Class: entities.entries.PersonName

Methods under test: -

* setFirstName
* setLastName

Rules: -

|  |  |  |
| --- | --- | --- |
|  | Conditions |  |
| **input** | **valid** | **invalid** |
| First Name | ASCII from a-z and A-Z, and >= 2 Char | ASCII other than a-z and A-Z and not blank and < 2 char |
| Last Name | ASCII from a-z and A-Z, and >= 2 Char | ASCII other than a-z and A-Z and not blank and < 2 char |
| Both | Spaces are ok |  |

### setFirstName

|  |  |  |  |
| --- | --- | --- | --- |
| **TC#** | **input** | **expected output** | **method** |
| 1 | Joe | valid | E.C and BVA |
| 2 | •ώ€β·°¯ | error | E.C |
| 3 | null | error | E.C |
| 4 | Ji | valid | BVA |
| 5 | ASCII #33-44 | error | E.C |
| 6 | ASCII #0-31 | error | E.C |
| 6 | ASCII #46-64 | error | E.C |
| 7 | ASCII #91-96 | error | E.C |
| 8 | ASCII #123-127 | error | E.C |

### setLastName

|  |  |  |  |
| --- | --- | --- | --- |
| **TC#** | **input** | **expected output** | **method** |
| 1 | Al Marri | valid | E.C |
| 2 | •ώ€β·°¯ | error | E.C |
| 3 | null | error | E.C |
| 4 | Li | valid | BVA |
| 5 | ASCII #33-44 | error | E.C |
| 6 | ASCII #0-31 | error | E.C |
| 6 | ASCII #46-64 | error | E.C |
| 7 | ASCII #91-96 | error | E.C |
| 8 | ASCII #123-127 | error | E.C |

### Class: entities.entries.Conveyance

Methods under test: -

* setVehicleIdentificationNumber
* setLicensePlateNumber
* setLicensePlateState
* setLicensePlateYear

### setVehicleIdentificationNumber

Rules: -

* Valid
  + Digits and chars are ok
  + 25 character or less is ok
  + Number of chars >= 5
* Invalid
  + Special characters
  + White spaces
  + Larger than 25 characters
  + Null

|  |  |  |  |
| --- | --- | --- | --- |
| TC# | input | expected output | method |
| 1 | 1G1BNUCH6GX181007 | valid | E.C. |
| 2 | ABCABCABCABC | valid | E.C. |
| 3 | 12332112332112321 | valid | E.C. |
| 4 | abcabcabcabcabcabcabcabcab | error | BVA |
| 5 | abcabcabcabcabcabcabcabca | valid | BVA |
| 6 | Abcabcabcabcabcabcabcab3 | valid | BVA |
| 7 | T6OP0 | valid | BVA |
| 8 | AS2V | error | BVA |
| 9 | T6#P0 | error | E.C. And BVA |
| 10 | null | error | E.C. |
| 11 | $AK4545F&$ | error | E.C. |
| 12 | IOCD345 DF3 | error | E.C. |

### setLicensePlateNumber

* Valid
  + Digits and chars are ok
  + Exactly = 6
  + First three are digits
  + Last three are chars
* Invalid
  + Special characters
  + White spaces
  + null

|  |  |  |  |
| --- | --- | --- | --- |
| TC# | input | expected output | method |
| 1 | 109IZT | valid | E.C. |
| 2 | ABCABCABCABC | error | BVA |
| 3 | 12332112332112321 | error | BVA |
| 4 | 1A1A1A | error | E.C. |
| 5 | ABC123 | error | E.C. |
| 6 | AS2V | error | BVA |
| 7 | T6# P0 | error | E.C. And BVA |
| 8 | null | error | E.C. |

### setLicensePlateState

* Valid
  + Only two chars
* Invalid
  + Special characters
  + White spaces
  + null
  + Digits

|  |  |  |  |
| --- | --- | --- | --- |
| TC# | input | expected output | method |
| 1 | SK | valid | E.C. |
| 2 | O | error | BVA |
| 3 | ONT | error | BVA |
| 4 | 12 | error | E.C. |
| 5 | P& | error | E.C. |
| 6 | null | error | E.C. |

### setLicensePlateYear

Assuming current date is March 5th, 2015

* Valid
  + Date must be less than or equal to current date
  + (Date – current date) >= 150

|  |  |  |  |
| --- | --- | --- | --- |
| TC# | input | expected output | method |
| 1 | 01/03/2005 | valid | E.C. |
| 2 | 11/02/1200 | error | E.C. |
| 3 | 04/04/2016 | error | E.C. |
| 4 | 05/03/2015 | valid | BVA |
| 5 | 04/03/2015 | valid | BVA |
| 6 | 06/03/2015 | error | BVA |
| 7 | 05/03/1865 | valid | BVA |
| 8 | 04/03/1865 | error | BVA |
| 9 | XX/XX/1864 | valid | BVA |

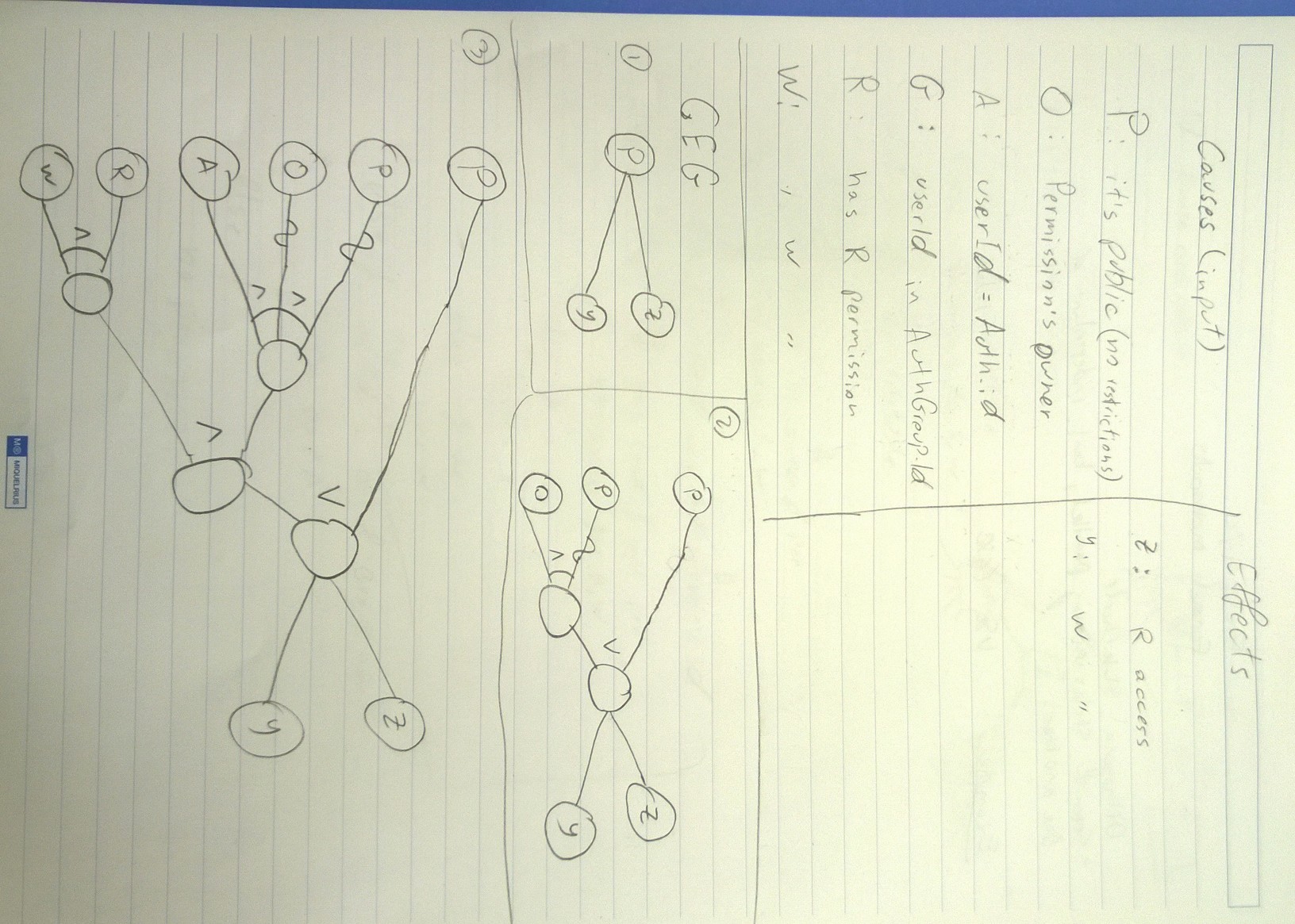
### Class: security. AccessManager

Methods under test: -

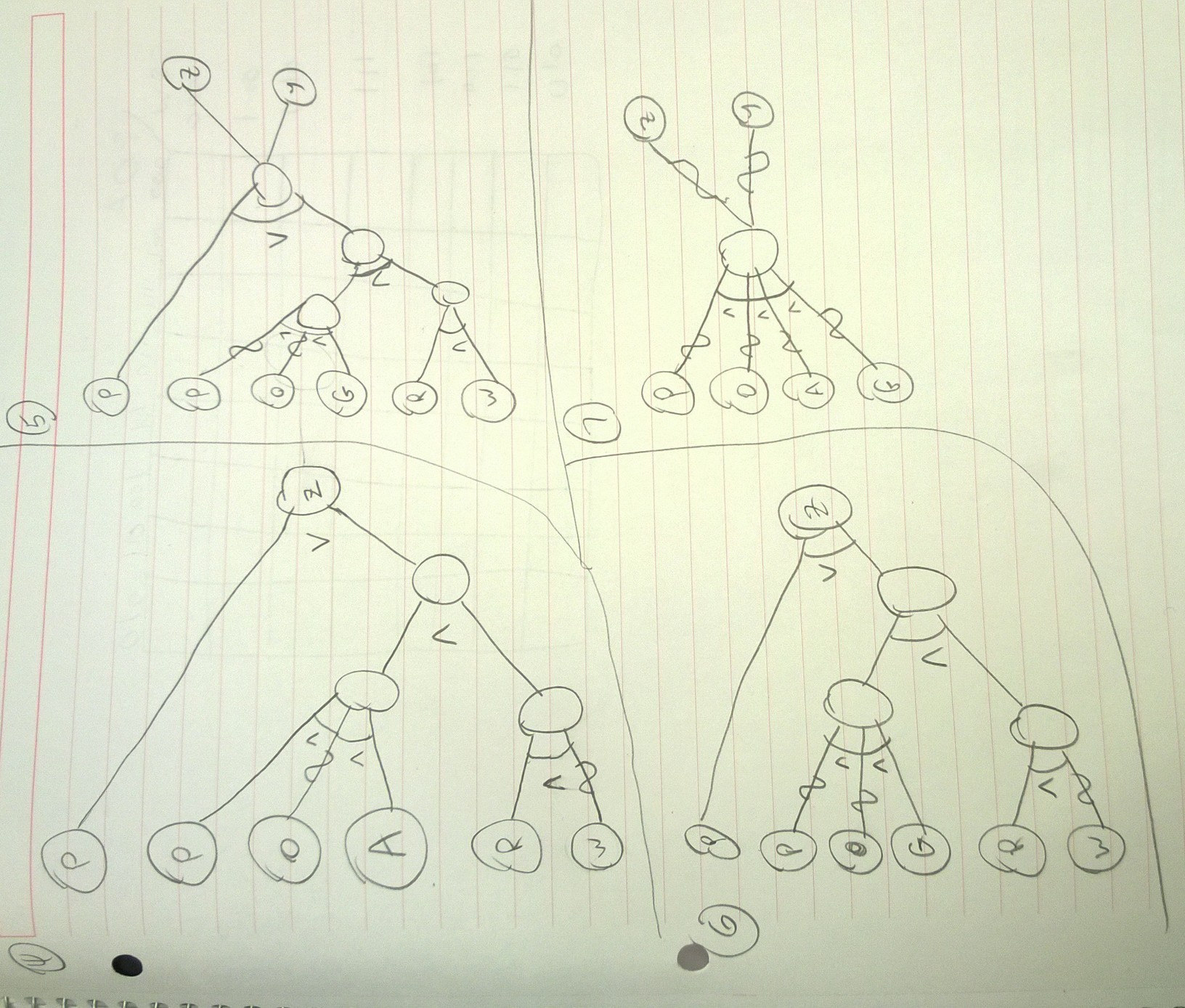
* manage

### manage

Cause and Effect Graph 1: -



Cause and Effect Graph 2: -



Decision table: -

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| P | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| O | X | 1 | 0 | 0 | 0 | 0 | 0 |
| A | X | X | 1 | 1 | 0 | 0 | 0 |
| G | X | X | X | X | 1 | 1 | 0 |
| R | X | X | 1 | 1 | 1 | 1 | X |
| W | X | X | 1 | 0 | 1 | 0 | X |
| Z | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| Y | 1 | 1 | 1 | 0 | 1 | 0 | 0 |

Test Cases: -

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **inputs** |  |  |  |  |  | **expected outputs** | |
| **TC#** | P | O | A | G | R | W | read granted | write granted |
| **1** | **1** | X | X | X | X | X | 1 | 1 |
| **2** | 0 | 1 | X | X | X | X | 1 | 1 |
| **3** | 0 | 0 | 1 | X | 1 | 1 | 1 | 1 |
| **4** | 0 | 0 | 1 | X | 1 | 0 | 1 | 0 |
| **5** | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| **6** | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| **7** | 0 | 0 | 0 | 0 | X | X | 0 | 0 |

### Class: entities.entries.Address

### setCity()

Rules are same as **setBirthPlace** in Person class, and same test cases.

### setProvince()

Rules are same as **setLicensePlateState** in Conveyance class, and same test cases.

### setCountry()

Rules are same as **setCitizenship** in Person class, and same test cases.

### setPostalCode()

Rules: -

* Valid
  + Digits and chars are ok
  + Exactly = 6
  + First three are digits
  + Last three are chars
  + One space only
  + Format: “L#L #L#” (where “L” is a litter and “#” is a digit)
* Invalid
  + Special characters
  + null

|  |  |  |
| --- | --- | --- |
| TC # | Inputs | Expected outputs |
| 1 | S4S 0A2 | Nothing |
| 2 | $4S 0B0 | ERROR |
| 3 | P5L 3T33 | ERROR |
| 4 | I6N 5S | ERROR |
| 5 | S4S0A2 | ERROR |
| 6 | null | ERROR |
| 7 | 123 123 | ERROR |
| 8 | ABC ABC | ERROR |

Class: entities.entries.PhysicalCharacteristic

Rules: -

* buildCharacteristics must be in { "Plump", "Stocky", "Overweight", "Fat", "Slim", "Trim", "Skinny", "Buff", "Well built" }
* heightCharacteristics must be in { "Short", "Tall", "Petite", "Average height" }
* complexionCharacteristics must be in { "Dark", "Light", "Fair", "Olive", "Pale", "Tan", "Pimply", "Freckles", "Spots", "Pimples" }
* hairCharacteristics must be in { "Blond", "Fair", "Red", "Brown", "Black", "Grey", "White", "Long", "Short", "Curly", "Frizzy", "Straight", "Bald", "Receding" }

Class: entities.entries. ThreatAssessment

Rules: -

threatAssessmentLevel must be in { "low", "medium", "high" }

Class: entities.entries.contacts. CellPhone

setContent() should accept only this format: “+# (###) ####” as Canadian national format

Class: entities.entries.contacts.Email

setContent() must be matched with regex of The Official Standard: RFC 5322

(?:[a-z0-9!#$%&'\*+/=?^\_`{|}~-]+(?:\.[a-z0-9!#$%&'\*+/=?^\_`{|}~-]+)\*

| "(?:[\x01-\x08\x0b\x0c\x0e-\x1f\x21\x23-\x5b\x5d-\x7f]

| \\[\x01-\x09\x0b\x0c\x0e-\x7f])\*")

@ (?:(?:[a-z0-9](?:[a-z0-9-]\*[a-z0-9])?\.)+[a-z0-9](?:[a-z0-9-]\*[a-z0-9])?

| \[(?:(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.){3}

(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?|[a-z0-9-]\*[a-z0-9]:

(?:[\x01-\x08\x0b\x0c\x0e-\x1f\x21-\x5a\x53-\x7f]

| \\[\x01-\x09\x0b\x0c\x0e-\x7f])+)

\])

Class: entities.entries.contacts.Telephone

setContent() should accept only this format: “+# (###) ####” as Canadian national format

## 9.2. Integration tests

### 9.2.1. Adding new person test (main scenario of adding new person)

|  |  |  |
| --- | --- | --- |
|  | **Steps** | **Pass Criteria** |
| 1 | Open home page /index.xhtml | Check text “Persons” |
| 2 | Click on link “Persons” /management/entries/person/listPersons.xhtml | Check text “Persons List” |
| 3 | Click on “Add Person”  /management/entries/person/addPersonInformation.xhtml | Check text “First name” |
| 4 | Fill up the form as shown below: -   * First name: Hamad * Last name: Al Marri * Date of birth: 23-07-1984 * Birth place: Al Hasa * Gender: select -> Male * Citizenship: Saudi Arabia * Upload an image for photo |  |
| 5 | Click “Save & Continue” button | Check text “Type of id” |
| 6 | Fill up the two fields with   * Id #: 9640028 * Type of id: SGI |  |
| 7 | Click “Save & Continue” button | Check text “Contact” |
| 8 | Fill up the two fields with   * Type: Cell phone * Contact: +13069990000 |  |
| 9 | Click “Save & Continue” button | Check text “Vehicle” |
| 10 | Fill up the fields with   * Vehicle identification number: 1G1NB69H6GX181227 * License plate number: 146IXY * License plate state: SK * License plate year: 2015 * Make: CHEV CAPRIC * Model: LV * Year: 1986 * Color: RED * Style: FOOR DOOR CAR |  |
| 11 | Click “Save & Continue” button | Check text “Modus” |
| 12 | Fill up the fields with   * Modus operandi: right handed * NCIC fingerprint Classification: AA * Alias names and monikers: the monster * Scars, marks, and tattoos: abu antar mark |  |
| 13 | Click “Save & Continue” button | Check text “Description” |
| 14 | Write anything in description field and choose any thing for physical characteristics |  |
| 15 | Click “Save & Finish” button | Check text “Persons List” |

### 9.2.2. Assigning investigators to a case

|  |  |  |
| --- | --- | --- |
|  | **Steps** | **Pass Criteria** |
| 1 | Open home page /index.xhtml | Check text “Cases” |
| 2 | Click on link “Cases” /management/police/investigativeCase/listInvestigativeCases.xhtml | Check text “Investigative Cases List” |
| 3 | Click on “Add New Case”  /management/police/investigativeCase/addInvestigativeCase.xhtml | Check text “Investigative Case Information” |
| 4 | Fill up Description with any text and pick any future due date |  |
| 5 | Click on button Save | Check text “Investigative Cases List” |
| 6 | Click on link of the created case | Check text “Status” |
| 7 | Click on button “Send Request to Investigator” |  |
| 8 | Write in investigator name field “m., Cannon” |  |
| 9 | Click button “Send” |  |
| 10 | Open home page /index.xhtml | Check text “Notifications” |
| 11 | Click on link “Notifications” /management/police/police/notification/listNotifications.xhtml | Check text “Sent” And at the same row check text “m. Connon” |

### 9.2.3. Security authorization

|  |  |  |
| --- | --- | --- |
|  | **Steps** | **Pass Criteria** |
| 1 | Open home page /index.xhtml | Check text “Cases” |
| 2 | Open login page /security/login.xhtml | Check text “username” |
| 3 | Fill text input with   * Username: James * Password: &32L@$t21 |  |
| 4 | Click on button “Login” |  |
| 5 | Open admin page /admin/admin.xhtml | Check text “access denied” |

## 9.3. ANN tests

By using jUnit we are able to tune the ANN for best results that are close to optimum

The method is to initialize the ANN model and to setup the training set, which is already set in file “crimeScene\_OffenderProfileViewExport.txt". By using the class trainer “OffenderProfileGeneratorTrainer” the ANN will be ready to be trained. Set errorDeltaAccepted = 0.3 since we want to ensure that the output form ANN for each node must be either 0 or 1. We don’t want to have a value such as 0.5, which will be ambiguous. For this reason we need to ensure the maximum off value is 0.3 from the boundaries (i.e. 0 = [0.0, 0.3] and 1 = [0.7 , 1.0]). The configuration of ANN is recommended to be: -

* 36 input nodes since the crime scene has 36 attributes
* 24 hidden nodes, which is 36 \* 2/3.
* 21 output nodes since the offender profile has 21 attributes
* Learning rate = 0.52
* Momentum = 0.15
* Number of passes = 50,000

Theses settings/configurations seems to be good enough for the current state of the system. They may to be changed in future to satisfy new data sets.

The test cases are 36, and the dependencies can be shown in the class “FillUpDatabaseWithCrimeScenesAndOffenderProfiles”.