MyHealthAvatar

A Demonstration of 4D Digital Avatar Infrastructure for Access of Complete Patient Information

Project acronym: MyHealthAvatar

Deliverable No. 3.6
Report on the Review of Open Source APIs for MHA

Grant agreement no: 600929

Dissemination Level		
PU	Public	**X**
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

COVER AND CONTROL PAGE OF DOCUMENT	
Project Acronym:	MyHealthAvatar
Project Full Name:	A Demonstration of 4D Digital Avatar Infrastructure for Access of Complete Patient Information
Deliverable No.:	D3.6
Document name:	Report on the review of open source APIs for MHA
Nature (R, P, D, O)[1]	R
Dissemination Level (PU, PP, RE, CO)[2]	PU
Version:	1
Actual Submission Date:	10/03/2015
Editor: Institution: E-Mail:	Xia Zhao University of Bedfordshire xia.zhao@beds.ac.uk

ABSTRACT:
This deliverable investigates the proof of market of having open source APIs for the MyHealthAvatar and reports the design and implementation of the APIs according to the user requirements. The security implementation for the API is also discussed.

KEYWORD LIST:
MyHealthAvatar APIs, OAuth

[1] **R**=Report, **P**=Prototype, **D**=Demonstrator, **O**=Other

[2] **PU**=Public, **PP**=Restricted to other programme participants (including the Commission Services), **RE**=Restricted to a group specified by the consortium (including the Commission Services), **CO**=Confidential, only for members of the consortium (including the Commission Services)

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n^{o} 600929.

MODIFICATION CONTROL			
Version	**Date**	**Status**	**Author**
0.1	26/01/2015	Draft	Xia Zhao
0.2	10/02/2015	Draft	Xia Zhao
0.3	25/02/2015	Draft	Xia Zhao
0.4	06/03/2015	Draft	Xia Zhao, Zhikun Deng
1	06/03/2015	Final	Xia Zhao

List of contributors

- Xia Zhao (BED)
- Zhikun Deng (BED)
- Feng Dong (BED)

Contents

1 Executive Summary

MyHealthAvatar follows recommendations from relevant VPH activities on “Digital Patient”. MyHealthAvatar architectural platform will be designed as an integrated facility that allows multiple functionalities rather than just a data storage facility as in the previous attempts. Its distinctive features include:

- ICT utilities to support data collection with minimal user input, including web information extraction, mobile apps, etc.
- ICT toolbox to support clinical decisions by using simulation models and by using visual analytics.
- Data and model repositories to provide rich resources of data and models
- Ontology and RDF repositories to support data search and reasoning.
- A cloud based ICT architecture that allows the access of data from a range of different sources, and integration of the repositories, the toolbox and the ICT utilities.
- A local cloud solution to support the computing requirement for the avatars without remote data transfer.
- A proof of market on open sources for MyHealthAvatar APIs.

The 4D Avatar proposed within the MyHealthAvatar project is intended to be a citizen’s lifelong companion a complete digital representation of his/her health-related data, including both medical records and social or lifestyle information, to create a comprehensive picture of the person in the context of their individualised healthcare. All of the data will be useful at different levels in assisting health decisions to be made.

This deliverable covers the activities for Tasks T3.6 and Milestone MS064 and presents the first stage MyHealthAvatar API implementation (version 2).

2 Introduction

Within MyHealthAvatar, collecting, accessing, managing and possibly sharing healthcare related data are not only important to individuals who can manage their own health, but also important for clinicians and other healthcare workers for patient monitoring and providing suitable in-time care. In order for potential applications to use the data collected by MyHealthAvatar, it is essential to provide accessible APIs which can expose and manage data.

2.1 Structure of this document

This document reviews the potential applications of MyHealthAvatar API and describes the implementation of the current version of API.

The structure of this document is organised as follows. Section 3 reviews the API requirements within the MyHealthAvatar project, which is generated mainly from *D9.1 Definition of the demos*. Section 4 describes the implementation of the available MyHealthAvatar APIs. Section 5 presents the security implementation aspect for the API. Section 6 concludes the report and outlines the future work.

3 Applications of MyHealthAvatar APIs

The applications of MyHealthAvatar APIs (see Figure 1) include the four high end clinical demos defined within the project in *D9.1 Definition of the demos* (i.e. DIAB-EME, CHF, OST and NEPH), the MyHealthAvatar demonstration web application and any other potential third-party applications. This section briefly reviews the requirements of MyHealthAvatar APIs for the project clinical demos and MyHealthAvatar web demonstration application.

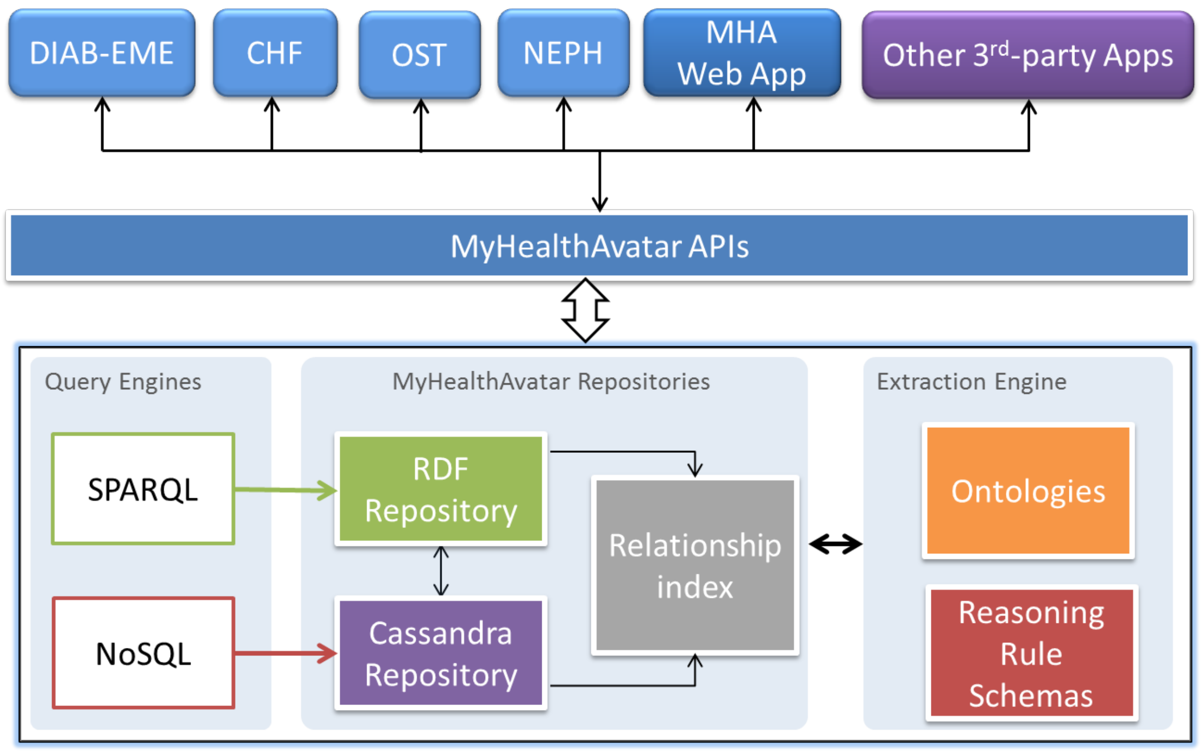


Figure 1: MyHealthAvatar APIs with potential applications.

3.1 *APIs Requirements for DIAB-EME clinical demo*

The Diabetes and Emergency demo (DIAB-EME) requires the management of and access to user's demographic data (e.g. age, ethnicity, gender, weight, etc.), activity data (e.g. location, movements and step counts), diet, environment, health survey data (e.g. quality of life, mood, alcohol, smoking, family diabetes history, etc.), clinical test data (e.g. blood pressure, glucose level) and other medical history (e.g. known allergies, sensitivity to drugs, etc.). The access of these data will be provided through appropriate MyHealthAvatar APIs.

The DIAB-EME demo may require MyHealthAvatar to provide APIs to handle the results of the risk assessment models in order to reflect to user's self-health management. These set of APIs will be developed when the implementation of this DIAB-EME demo scenario provides clearer assessment results structures.

3.2 APIs Requirements for CHF clinical demo

The Congestive Heart Failure (CHF) demo requires the MyHealthAvatar API to mainly provide the management of and access to user's demographic data (e.g. age, gender, height and weight, BMI), activity data (e.g. location, movements and step counts), diet, environment, health survey data (e.g. quality of life, mood, alcohol, smoking, etc.), clinical test data (e.g. blood pressure, pulse, blood test results). CHF will handle the external data linkage to electronic health records and clinical hospital information system repository. The summary of these external data may also be exposed in MyHealthAvatar APIs.

The CHF demo may require the MyHealthAvatar API to provide functions to handle the monitoring alarm information and the results of risk assessment.

3.3 APIs Requirements for OST clinical demo

The Osteoarthritis (OST) demo requires the MyHealthAvatar API to provide access to all the health related data including user's demographic data (e.g. age, gender, height and weight), activity data (e.g. movements, type of activities, step counts), nutrition, diet, environment, health survey data (e.g. quality of life, mood, alcohol, smoking, etc.), clinical test data, medical history. OST will connect to image server to access user's medical imaging data when necessary.

The OST demo may require the MyHealthAvatar to provide functions for presenting the result of risk analysis and guidance for daily physical exercises, diet and medication if necessary.

3.4 APIs Requirements for NEPH clinical demo

The Nephroblastoma-Wilms Tumor (NEPH) demo mainly requires the MyHealthAvatar API to provide the access to user's demographic data (e.g. age, gender, height and weight, BMI) and relevant medical history data via the link to the external hospital system. NEPH will directly access to clinical trial management system, medical imaging data, and hospital information system when necessary.

3.5 APIs Requirements for MyHealthAvatar Web Application

MyHealthAvatar Web Application (http://myhealthavatar.org/) serves as the first-party application for manage, access and visualise MyHealthAvatar data. It requires the MyHealthAvatar API to provide functions for access and manage all the data stored in the data repository including user's demographic data, nutrition and diet, environment, quality of life, mood, alcohol, smoking, clinical test results, medical history, data sharing, dairy and planning, etc. Section 4 provides the current implementation of APIs for MyHealthAvatar data. Some of the function will be shared by the clinical demos such as diary and planning.

4 Implementation of MyHealthAvatar APIs

The implementation of MyHealthAvatar APIs focuses on the providing functions for potential applications to access and manage data stored in the data repository implemented in WP6. This section summarises the already implemented APIs (version 2) for MyHealthAvatar. The online document is available at http://myhealthavatar.org/api/doc/v2. The APIs consume and produce data in JSON format.

The APIs are designed into two main groups according to the MyHealthAvatar data structures: one group is personalised user oriented APIs which handles data on specific MyHealthAvatar user; the other is general APIs which manage non-personalised MyHealthAvatar data.

4.1 APIs for Personalised MyHealthAvatar Data

4.1.1 APIs for User Daily Activities

Daily activity API includes activity daily summary, activity segments and full summary of activities for the user. Daily activity includes the following data.

- date: data generated date, in format yyyy-MM-dd
- local_id: id in MyHealthAvatar in format of uuid
- summary:
    - source: currently one of fitbit, moves, withings
    - steps: step count for the activity (if applicable)
    - duration: duration of the activity in minutes (if applicable)
    - distance: distance of the activity in metres (if applicable)
    - calories: calories burn for the activity in kcal (if applicable) on top of the BMR/Idle burn
    - calories_bmr: calories burn at BMR/Idle in kcal (if applicable)
    - calories_out: calories out total in kcal (if applicable)
    - elevation: elevation in metres (if applicable)
    - soft_activity_minutes: duration of soft/light activities in minutes (if applicable)
    - moderate_activity_minutes: duration of moderate/fair activities in minutes (if applicable)
    - intense_activity_minutes: duration of intense/very-active activites in minutes (if applicable)
    - sedentary_minutes: duration of sedentary in minutes (if applicable)
    - activities:
        - activity: specific name of the activity
        - activity_group: a more generic activity supergroup to which the action belongs to (if applicable)
        - steps: step count for the activity (if applicable)
        - duration: duration of the activity in minutes (if applicable)
        - distance: distance of the activity in metres (if applicable)

            - calories: calories burn for the activity in kcal (if applicable) on top of the BMR/Idle burn

- segments:
    - source: currently only moves
    - type: currently one of move, place or off (off segments indicate that Moves tracking had stopped for some reason)
    - start_time: segment start time in yyyy-MM-dd HHmmssZ format
    - end_time: segment end time in yyyy-MM-dd HHmmssZ format
    - place:
        - local_id: place uudi id in MyHealthAvatar (if applicable)
        - name: name of the place (if applicable)
        - type: one of
            - unknown: the place has not been identified
            - home: the place is labeled as home
            - school: the place is labeled as school
            - work: the place is labeled as work
            - user: the place has been manually named
            - foursquare: the place has been identified from foursquare
        - location:
            - lat: latitude coordinate as number
            - lon: longitude coordinate as number
        - foursquare_id: foursquare venue id (if applicable)
        - foursquare_category_ids: foursquare category ids for the place (if available)
    - activities:
        - activity: specific name of the activity
        - activity_group: a more generic activity supergroup to which the action belongs to (if applicable)
        - steps: step count for the activity (if applicable)
        - duration: duration of the activity in minutes (if applicable)
        - distance: distance of the activity in metres (if applicable)
        - calories: calories burn for the activity in kcal (if applicable) on top of the BMR/Idle burn
        - start_time: activity start time in yyyy-MM-dd HHmmssZ format
        - end_time: activity end time in yyyy-MM-dd HHmmssZ format
        - track_points:
            - time: time of the tracking in yyyy-MM-dd HHmmssZ format
            - lat: latitude coordinate as number
            - lon: longitude coordinate as number

Activity daily summary API has the following methods. Method 1) shows all the daily activity summary for the specific user; Method 2) shows the daily activity summary for the specific user from the given date; Method 3) shows the daily activity summary for the specific user until the given date; Method 4) shows the daily activity summary for the specific user in the given date range; Method 5) shows the daily activity summary for the specific user on the given date.

1) `[GET] http://myhealthavatar.org/mha/api/v2/user/activities/summary`
2) `[GET] http://myhealthavatar.org/mha/api/v2/user/activities/summary?from={yyyy-mm-dd}`
3) `[GET] http://myhealthavatar.org/mha/api/v2/user/activities/summary?to={yyyy-mm-dd}`
4) `[GET] http://myhealthavatar.org/mha/api/v2/user/activities/summary?from={yyyy-mm-dd}&to={yyyy-mm-dd}`
5) `[GET] http://myhealthavatar.org/mha/api/v2/user/activities/summary?date={yyyy-mm-dd}`

Below is a sample response from the activity daily summary API.

```
[
  {
    "date": "2014-09-12",
    "source": "fitbit",
    "steps": 0,
    "calories": 230,
    "calories_bmr": 1913,
    "calories_out": 2143,
    "elevation": 48.77,
    "soft_activity_minutes": 0,
    "moderate_activity_minutes": 0,
    "intense_activity_minutes": 0,
    "sedentary_minutes": 1166,
    "activities": [
      {
        "activity": "Treadmill, 0% Incline",
        "duration": 1097,
        "distance": 2040,
        "steps": 3783,
        "calories": 230
      }
    ]
  },
  {
    "date": "2014-09-12",
    "source": "withings",
    "distance": 5783.6,
    "steps": 7065,
    "calories": 242.61,
    "elevation": 32.2,
    "soft_activity_minutes": 40,
    "moderate_activity_minutes": 30,
    "intense_activity_minutes": 2
  },
  {
    "date": "2012-12-12",
    "source": "moves",
    "calories_bmr": 1785,
    "activities": [
      {
        "activity": "walking",
        "activity_group": "walking",
        "duration": 3333,
        "distance": 3333,
        "steps": 3333,
        "calories": 300
      },
      {
        "activity": "walking_on_treadmill",
        "activity_group": "walking",
        "duration": 270,
        "distance": 0,
        "steps": 303,
        "calories": 30
      },
      {
        "activity": "transport",
        "activity_group": "transport",
        "duration": 1124,
        "distance": 8443,
        "steps": 0,
        "calories": 0
      },
      {
        "activity": "underground",
        "activity_group": "transport",
        "duration": 1003,
        "distance": 8058,
        "steps": 0,
        "calories": 0
      },
      {
        "activity": "zumba",
        "duration": 570,
        "distance": 0,
        "steps": 0,
        "calories": 200
      }
    ]
  }
]
```

Activity segment APIs include detailed information on activity information (e.g. activity type, start and end time of the activity, longitude and latitude of the track points of the activity if available. The following methods are implemented. The *from*, *to*, and *date* parameters have the same meaning as in activity daily summary APIs, so Method 1)-5) work in the similar way to the activity daily summary APIs to return relevant activity segment data to the specific user. Method 6) returns activity segments data for the specific user from the given start time; Method 7) returns activity segments data for the specific user from the given time range.

1) `[GET] http://myhealthavatar.org/mha/api/v2/user/activities/segments`
2) `[GET] http://myhealthavatar.org/mha/api/v2/user/activities/segments?from={yyyy-mm-dd}`
3) `[GET] http://myhealthavatar.org/mha/api/v2/user/activities/segments?to={yyyy-mm-dd}`

4) `[GET] http://myhealthavatar.org/mha/api/v2/user/activities/segments?from={yyyy-mm-dd}&to={yyyy-mm-dd}`
5) `[GET] http://myhealthavatar.org/mha/api/v2/user/activities/segments?date={yyyy-mm-dd}`
6) `[GET] http://myhealthavatar.org/mha/api/v2/user/activities/segments?start_time={yyyy-mm-dd hh:mm:ss}`
7) `[GET] http://myhealthavatar.org/mha/api/v2/user/activities/segments?start_time={yyyy-mm-dd hh:mm:ss}&end_time={yyyy-mm-dd hh:mm:ss}`

Below is a sample response from the activity segment APIs.

```
[
  {
    "date": "2012-12-12",
    "source": "moves",
    "type": "place",
    "start_time": "2012-12-11 22:00:00+0000",
    "end_time": "2012-12-12 05:14:30+0000",
    "place": {
      "location": {
        "lat": 55.55555,
        "lon": 33.33333
      },
      "type": "unknown"
    }
  },
  {
    "date": "2012-12-12",
    "source": "moves",
    "type": "move",
    "start_time": "2012-12-12 05:14:30+0000",
    "end_time": "2012-12-12 05:46:17+0000",
    "activities": [
      {
        "activity": "walking",
        "activity_group": "walking",
        "duration": 782,
        "distance": 1251,
        "steps": 1353,
        "calories": 99,
        "start_time": "2012-12-12 05:14:30+0000",
        "end_time": "2012-12-12 05:27:32+0000",
        "track_points": [
          {
            "time": "2012-12-12 05:14:30+0000",
            "lat": 55.55555,
            "lon": 33.33333
          },
          {
            "time": "2012-12-12 05:27:32+0000",
            "lat": 55.55555,
            "lon": 33.33333
          }
        ]
      },
      {
        "activity": "transport",
        "activity_group": "transport",
        "duration": 1124,
        "distance": 8443,
        "steps": 0,
        "calories": 0,
        "start_time": "2012-12-12 05:27:32+0000",
        "end_time": "2012-12-12 05:46:16+0000",
        "track_points": [
          {
            "time": "2012-12-12 05:27:32+0000",
            "lat": 55.55555,
            "lon": 33.33333
          },
          {
            "time": "2012-12-12 05:42:08+0000",
            "lat": 55.55555,
            "lon": 33.33333
          },
          {
            "time": "2012-12-12 05:46:17+0000",
            "lat": 55.55555,
            "lon": 33.33333
          }
        ]
      }
    ]
  }
]
```

The full activity summary APIs have the following methods. Method 1)-5) work in the similar way to activity daily summary. Method 6) accepts JSON data from different sources and stored them into the MyHealthAvatar data repository. This version has three identified data sources: Fitbit, Moves and Withings. More sources can be added in the future version according to application requirements, such as accepting data from manual input or direct mobile devices.

1) `[GET] http://myhealthavatar.org/mha/api/v2/user/activities`
2) `[GET] http://myhealthavatar.org/mha/api/v2/user/activities?from={yyyy-mm-dd}`
3) `[GET] http://myhealthavatar.org/mha/api/v2/user/activities?to={yyyy-mm-dd}`
4) `[GET] http://myhealthavatar.org/mha/api/v2/user/activities?from={yyyy-mm-dd}&to={yyyy-mm-dd}`
5) `[GET] http://myhealthavatar.org/mha/api/v2/user/activities?date={yyyy-mm-dd}`
6) `[POST] http://myhealthavatar.org/mha/api/v2/user/activities?source=[fitbit, moves, withings]`

Below is a sample response from the full activity summary API.

```
[
  {
    "date": "2014-09-12",
    "local_id": "7cfe069f-09ab-4889-91ef-2c8c5544b13a",
    "summary": {
        "source": "fitbit",
        "steps": 0,
        "calories": 230,
        "calories_bmr": 1913,
        "calories_out": 2143,
        "elevation": 48.77,
        "soft_activity_minutes": 0,
        "moderate_activity_minutes": 0,
        "intense_activity_minutes": 0,
        "sedentary_minutes": 1166,
        "activities": [
          {
            "activity": "Treadmill, 0% Incline",
            "duration": 1097,
            "distance": 2040,
            "steps": 3783,
            "calories": 230
          }
        ]
      }
    },
    {
      "date": "2014-09-12",
      "local_id": "8c9adc1d-9453-4180-a750-47ddac842363",
      "summary": {
        "source": "withings",
        "distance": 5783.6,
        "steps": 7065,
        "calories": 242.61,
        "elevation": 32.2,
        "soft_activity_minutes": 40,
        "moderate_activity_minutes": 30,
        "intense_activity_minutes": 2
      }
```

```
  },
  {
    "date": "2012-12-12",
    "local_id": "0fd8cc4d-c0a6-4655-9fa5-c50f5fbd4ed2",
    "summary": {
      "source": "moves",
      "calories_bmr": 1785,
      "activities": [
        {
          "activity": "walking",
          "activity_group": "walking",
          "duration": 3333,
          "distance": 3333,
          "steps": 3333,
          "calories": 300
        },
        {
          "activity": "walking_on_treadmill",
          "activity_group": "walking",
          "duration": 270,
          "distance": 0,
          "steps": 303,
          "calories": 30
        },
        {
          "activity": "transport",
          "activity_group": "transport",
          "duration": 1124,
          "distance": 8443,
          "steps": 0,
          "calories": 0
        },
        {
          "activity": "underground",
          "activity_group": "transport",
          "duration": 1003,
          "distance": 8058,
          "steps": 0,
          "calories": 0
        },
        {
          "activity": "zumba",
          "duration": 570,
          "distance": 0,
          "steps": 0,
          "calories": 200
        }
      ]
    },
    "segments": [
      {
        "source": "moves",
        "type": "place",
        "start_time": "2012-12-11 22:00:00+0000",
        "end_time": "2012-12-12 05:14:30+0000",
        "place": {
          "location": {
            "lat": 55.55555,
            "lon": 33.33333
          },
          "type": "unknown"
        }
      },
      {
        "source": "moves",
        "type": "move",
        "start_time": "2012-12-12 05:14:30+0000",
        "end_time": "2012-12-12 05:46:17+0000",
          "activities": [
            {
              "activity": "walking",
              "activity_group": "walking",
              "duration": 782,
              "distance": 1251,
              "steps": 1353,
              "calories": 99,
              "start_time": "2012-12-12 05:14:30+0000",
              "end_time": "2012-12-12 05:27:32+0000",
              "track_points": [
                {
                  "time": "2012-12-12 05:14:30+0000",
                  "lat": 55.55555,
                  "lon": 33.33333
                },
                {
                  "time": "2012-12-12 05:27:32+0000",
                  "lat": 55.55555,
                  "lon": 33.33333
                }
              ]
            },
            {
              "activity": "transport",
              "activity_group": "transport",
              "duration": 1124,
              "distance": 8443,
              "steps": 0,
              "calories": 0,
              "start_time": "2012-12-12 05:27:32+0000",
              "end_time": "2012-12-12 05:46:16+0000",
              "track_points": [
                {
```

```
                "time": "2012-12-12 05:27:32+0000",
                "lat": 55.55555,
                "lon": 33.33333
              },
              {
                "time": "2012-12-12 05:42:08+0000",
                "lat": 55.55555,
                "lon": 33.33333
              },
              {
                "time": "2012-12-12 05:46:17+0000",
                "lat": 55.55555,
                "lon": 33.33333
              }
            ]
          }
        ]
      },
      {
        "source": "moves",
        "type": "place",
        "start_time": "2012-12-12 05:46:17+0000",
        "end_time": "2012-12-12 08:00:51+0000",
        "place": {
          "location": {
            "lat": 55.55555,
            "lon": 33.33333
          },
          "type": "unknown"
        },
        "activities": [
          {
            "activity": "walking_on_treadmill",
            "activity_group": "walking",
          "duration": 270,
          "distance": 0,
          "steps": 303,
          "calories": 30
        }
      ]
    },
    {
      "source": "moves",
      "type": "move",
      "start_time": "2012-12-12 08:00:51+0000",
      "end_time": "2012-12-12 08:07:15+0000",
      "activities": [
        {
          "activity": "walking",
          "activity_group": "walking",
          "duration": 384,
          "distance": 421,
          "steps": 488,
          "calories": 99,
          "start_time": "2012-12-12 08:00:51+0000",
          "end_time": "2012-12-12 08:07:15+0000",
          "track_points": [
            {
              "time": "2012-12-12 08:00:51+0000",
              "lat": 55.55555,
              "lon": 33.33333
            },
            {
              "time": "2012-12-12 08:07:15+0000",
              "lat": 55.55555,
              "lon": 33.33333
            }
          ]
          }
        ]
      },
      {
        "source": "moves",
        "type": "place",
        "start_time": "2012-12-12 08:07:15+0000",
        "end_time": "2012-12-12 09:05:30+0000",
        "place": {
          "local_id": "b834fd40-c48f-48bb-a1e4-46561dd843f4",
          "name": "test",
          "location": {
            "lat": 55.55555,
            "lon": 33.33333
          },
          "type": "foursquare",
          "foursquare_id": "4df0fdb17d8ba370a011d24c",
          "foursquare_category_ids": [
            "4bf58dd8d48988d125941735"
          ]
        },
        "activities": [
          {
            "activity": "walking",
            "activity_group": "walking",
            "duration": 40,
            "distance": 18,
            "steps": 37,
            "calories": 99,
            "start_time": "2012-12-12 08:12:15+0000",
            "end_time": "2012-12-12 08:12:55+0000",
            "track_points": [
              {
                "time": "2012-12-12 08:12:15+0000",
                "lat": 55.55555,
                "lon": 33.33333
              },
              {
                "time": "2012-12-12 08:12:55+0000",
                "lat": 55.55555,
                "lon": 33.33333
              }
            ]
          }
        ]
      },
      {
        "source": "moves",
        "type": "move",
        "start_time": "2012-12-12 09:05:30+0000",
        "end_time": "2012-12-12 09:11:29+0000",
        "activities": [
          {
            "activity": "walking",
            "activity_group": "walking",
            "duration": 358,
            "distance": 493,
            "steps": 441,
            "calories": 99,
            "start_time": "2012-12-12 09:05:30+0000",
            "end_time": "2012-12-12 09:11:28+0000",
            "track_points": [
              {
                "time": "2012-12-12 09:05:31+0000",
                "lat": 55.55555,
                "lon": 33.33333
              },
              {
```

```
            "time": "2012-12-12 09:05:36+0000",
            "lat": 55.55555,
            "lon": 33.33333
          },
          {
            "time": "2012-12-12 09:09:47+0000",
            "lat": 55.55555,
            "lon": 33.33333
          },
          {
            "time": "2012-12-12 09:10:17+0000",
            "lat": 55.55555,
            "lon": 33.33333
          },
          {
            "time": "2012-12-12 09:11:29+0000",
            "lat": 55.55555,
            "lon": 33.33333
          }
        ]
      }
    ]
  },
  {
    "source": "moves",
    "type": "place",
    "start_time": "2012-12-12 09:11:29+0000",
    "end_time": "2012-12-12 13:36:38+0000",
    "place": {
      "location": {
        "lat": 55.55555,
        "lon": 33.33333
      },
      "type": "unknown"
```

```
        },
        "activities": [
          {
            "activity": "zumba",
            "duration": 570,
            "distance": 0,
            "steps": 0,
            "calories": 200
          }
        ]
      },
      {
        "source": "moves",
        "type": "move",
        "start_time": "2012-12-12 13:36:38+0000",
        "end_time": "2012-12-12 14:07:44+0000",
        "activities": [
          {
            "activity": "underground",
            "activity_group": "transport",
            "duration": 1003,
            "distance": 8058,
            "steps": 0,
            "calories": 0,
            "start_time": "2012-12-12 13:36:38+0000",
            "end_time": "2012-12-12 13:53:21+0000",
            "track_points": [
              {
                "time": "2012-12-12 13:36:38+0000",
                "lat": 55.55555,
                "lon": 33.33333
              },
              {
                "time": "2012-12-12 13:53:22+0000",
              "lat": 55.55555,
              "lon": 33.33333
            }
          ]
        },
        {
          "activity": "walking",
          "activity_group": "walking",
          "duration": 862,
          "distance": 1086,
          "steps": 1257,
          "calories": 99,
          "start_time": "2012-12-12 13:53:22+0000",
          "end_time": "2012-12-12 14:07:44+0000",
          "track_points": [
            {
              "time": "2012-12-12 13:53:22+0000",
              "lat": 55.55555,
              "lon": 33.33333
            },
            {
              "time": "2012-12-12 13:54:02+0000",
              "lat": 55.55555,
              "lon": 33.33333
            },
            {
              "time": "2012-12-12 14:07:44+0000",
              "lat": 55.55555,
              "lon": 33.33333
            }
          ]
        }
      ]
    },
      {
        "source": "moves",
        "type": "place",
        "start_time": "2012-12-12 14:07:44+0000",
        "end_time": "2012-12-12 21:27:30+0000",
        "place": {
          "location": {
            "lat": 55.55555,
            "lon": 33.33333
          },
          "type": "unknown"
        }
      }
    ]
  }
]
```

4.1.2 APIs for User Diary

User diary API handles the following data in the current version implementation.

- Local_id: id in MyHealthAvatar in format of uuid
- text: main title of the diary
- start_time: start time in yyyy-MM-dd HHmmssZ format
- end_time: end time in yyyy-MM-dd HHmmssZ format
- location: the location information of the diary record
- tags: key word tags for the diary record
- details: the detail information of the diary record
- valueItem: value specified items (if available)

The user diary API has the following methods. Method 1) accepts diary input in JSON format and stores/updates it in the data repository. Method 2) obtains all the diary records for the authenticated user. Method 3) obtains a diary record by given time for the authenticated user. Method 4) obtains a list of diary records for the authenticated user from the given time until now. Method 5) obtains a list of diary records for the authenticated user by the given time range. Method 6) deletes a diary record by its local id.

```
1) [POST] http://myhealthavatar.org/mha/api/v2/user/diary
2) [GET] http://myhealthavatar.org/mha/api/v2/user/diary
```

3) `[GET] http://myhealthavatar.org/mha/api/v2/user/diary?time=[yyyy-MM-dd'T'HH:mmssZ]`
4) `[GET] http://myhealthavatar.org/mha/api/v2/user/diary?from=[yyyy-MM-dd'T'HH:mmssZ]`
5) `[GET] http://myhealthavatar.org/mha/api/v2/user/diary?from=[yyyy-MM-dd'T'HH:mmssZ]&to=[yyyy-MM-dd'T'HH:mmssZ]`
6) `[POST] http://myhealthavatar.org/mha/api/v2/user/diary/delete?id=[local_id]`

Below is a sample response for the GET API methods.

```
[
 {
   "local_id": "bd9b13dc-6f1a-4711-880c-ee6991d9db7e",
   "text": "Car travel for project meeting",
   "start_time":"2014-02-27 09:15:00+0000",
   "end_time":"2014-02-27 11:10:00+0000",
   "location":"Luton to Sheffield via M1",
   "tags":null,
   "details":"Luton to Sheffield via M1",
   "valueItems":null
 }
]
```

4.1.3 APIs for User Body Measurements

User Body measurement APIs handle the following data in the current version implementation.

- local_id: id in MyHealthAvatar in format of uuid
- source: currently one of withings or manual
- time: time of the measurement added in format of yyyy-MM-dd HH:mm:ssZ
- height: in cm (if available)
- weight: in kg (if available)
- fat_free_mass: in kg (if available)
- fat ratio: in % (if available)
- fat_mass_weight: in kg (if available)
- spo2: in % (if available)
- bmi: Body Mass Index (if available)
- total_cholesterol: in mg/dl (if available)
- hdl_cholesterol: in mg/dl (if available)
- triglyceride: in mg/dl (if available)
- blood_pressure_systolic: in mmHg (if available)

- blood_pressure_diatolic: in mmHg (if available)
- pulse: heart pulse (if available)
- fasting_glucose_level: in mg/dl (if available)
- left_vent_hypertrophy: true or false (if available)

The APIs have the following methods. The GET methods 1)-5) work in a similar way to the activity daily summary API described in section 4.1.1 to obtain measurement data for specific user in full, or given date ranges. The POST method 6) accepts measurement input in JSON format and stored them into MyHealthAvatar data repository.

1) `[GET] http://myhealthavatar.org/mha/api/v2/user/measurements`
2) `[GET] http://myhealthavatar.org/mha/api/v2/user/measurements?from={yyyy-mm-dd}`
3) `[GET] http://myhealthavatar.org/mha/api/v2/user/measurements?to={yyyy-mm-dd}`
4) `[GET] http://myhealthavatar.org/mha/api/v2/user/measurements?from={yyyy-mm-dd}&to={yyyy-mm-dd}`
5) `[GET] http://myhealthavatar.org/mha/api/v2/user/measurements?date={yyyy-mm-dd}`
6) `[POST] http://myhealthavatar.org/mha/api/v2/user/measurements?source=[*]`

Below is a sample response for the GET API methods.

```
[
  {
    "local_id": "ab56cd75-972a-423f-bd91-ff21f5c929b3",
    "source": "manual",
    "update_time": "2012-12-12 09:11:29+0000",
    "height": 165,
    "weight": 50,
    "fat_free_mass": 652,
    "fat_ratio": 178,
    "fat_mass_weight": 14125,
    "spo2": 98,
    "bmi": 22,
    "total_cholesterol": 180,
    "hdl_cholesterol": 45,
    "triglyceride": 45,
    "blood_pressure_systolic": 110,
    "blood_pressure_diatolic": 75,
    "pulse": 78,
    "fasting_glucose_level": 45,
    "left_vent_hypertrophy": false
  }
]
```

Below is a sample request for the POST method.

```
{
  "update_time": "2012-12-12 09:11:29+0000",
  "height": 165,
  "weight": 50,
  "fat_free_mass": 652,
  "fat_ratio": 178,
  "fat_mass_weight": 14125,
  "spo2": 98,
  "bmi": 22,
  "total_cholesterol": 180,
  "hdl_cholesterol": 45,
  "triglyceride": 45,
  "blood_pressure_systolic": 110,
  "blood_pressure_diatolic": 75,
  "pulse": 78,
  "fasting_glucose_level": 45,
  "left_vent_hypertrophy": false
}
```

4.1.4 APIs for General User Health Data

The general user health data API handle the following survey-style data.

- local_id: id in MyHealthAvatar in format of uuid
- update_date: update date time in format of yyyy-MM-dd HH:mm:ssZ
- smoking: true-smoking, false-not smoking (if applicable)
- alcohol: 0-never, 1-monthly or less, 2-two or four times a month, 3-two or three times per week, 4-four or more times a week (if applicable)
- diabetes: true-have diabetes, false-not have diabetes (if applicable)
- parental_diabetes: true-parents have diabetes, false-parents do not have diabetes (if applicable)

- parental_hypertension: true-parents have hypertension, false-parents do not have hypertension (if applicable)
- prior_cardiovascular: true-have cardiovascular disease before, false-not have cardiovascular disease before (if applicable)
- physical_activity: 0-poor, 1-fair, 2-good, 3-very good, 4-excellent (if applicable)
- mood: 0-poor, 1-fair, 2-good, 3-very good, 4-excellent (if applicable)
- social_engagement: 0-poor, 1-fair, 2-good, 3-very good, 4-excellent (if applicable)
- entertainment: 0-poor, 1-fair, 2-good, 3-very good, 4-excellent (if applicable)

The general user health data API contains the following methods. The GET methods 2)-6) work in a similar way to the activity daily summary API described in section 4.1.1 to obtain general user health data for specific user in full, or given date ranges. The POST method 1) accepts general user health data input in JSON format and stored them into MyHealthAvatar data repository.

1) `[POST] http://myhealthavatar.org/mha/api/v2/user/general`
2) `[GET] http://myhealthavatar.org/mha/api/v2/user/general`
3) `[GET] http://myhealthavatar.org/mha/api/v2/user/general?from={yyyy-mm-dd}`
4) `[GET] http://myhealthavatar.org/mha/api/v2/user/general?to={yyyy-mm-dd}`
5) `[GET] http://myhealthavatar.org/mha/api/v2/user/general?from={yyyy-mm-dd}&to={yyyy-mm-dd}`
6) `[GET] http://myhealthavatar.org/mha/api/v2/user/general?date={yyyy-mm-dd}`

Below is a sample response for the GET methods of the API.

```
[
  {
    "local_id": "188207b4-a4e1-4ee8-9ba5-d7211c405539",
    "update_time": "2012-12-12 09:11:29+0000",
    "smoking": false,
    "alcohol": 0,
    "diabetes": false,
        "parental_diabetes": false,
        "parental_hypertension": false,
        "prior_cardiovascular": false,
        "physical_activity": 3,
        "mood": 3,
        "social_engagement": 3,
        "entertainment": 3
  }
]
```

Below is a sample request for the POST method of the API.

```
{
  "update_time": "2012-12-12 09:11:29+0000",
  "smoking": false,
  "alcohol": 0,
  "diabetes": false,
  "parental_diabetes": false,
  "parental_hypertension": false,
  "prior_cardiovascular": false,
  "physical_activity": 3,
  "mood": 3,
  "social_engagement": 3,
  "entertainment": 3
}
```

4.1.5 APIs for User Profile

The user profile API handles the following data.

- lastupdate: last update time in yyyy-MM-dd HHmmssZ format
- local_id: mha id of the user profile
- personal_information:
    - basic:
        - lastupdate: last update time in yyyy-MM-dd HHmmssZ format
        - given_name: given names of the user
        - surname: surname of the user
        - gender: male, female or not specified
        - dob: date of birth in yyyy-MM-dd format
        - occupation: occupation information
        - contact
            - email: a list of email
            - phone: a list of phone numbers
            - mobile: a list of mobile numbers
            - address_home: a list of home addresses include street, city, state/county/province, country and postcode
            - address_work: a list of work addresses include street, city, state/county/province, country and postcode
    - private_contacts
        - lastupdate: last update time in yyyy-MM-dd HHmmssZ format
        - relationship: relationship to the user

        - user_id: if the user is a current user in MHA, an id of the user is given
        - contact: if the user is not a current user in MHA, the contact detail is given
    - medical_contacts
        - lastupdate: last update time in yyyy-MM-dd HHmmssZ format
        - place_id: if the institution is in MHA, an id of the place is given
        - institute_name: if the institution is not in MHA, the name and address of the institution are given
        - address: address of the insitution.
        - user_id: if the user is a current user in MHA, an id of the user is given
        - contact: if the user is not a current user in MHA, the contact detail is given
- insurance:
    - lastupdate: last update time in yyyy-MM-dd HHmmssZ format
    - insurance_number: the insurance reference number
    - insurance_contact: the id of the insurance contact
- allergies:
    - lastupdate: last update time in yyyy-MM-dd HHmmssZ format
    - agent: the allergy agent name
    - description: the descirption of the allergy symptoms
    - onset_time: onset time in yyyy-MM-dd HHmmssZ format
- diagnosis:
    - lastupdate: last update time in yyyy-MM-dd HHmmssZ format
    - description: the description of the diagnosis
    - problem_id: the IDC10 code system id e.g.: 2.16.840.1.113883.6.3
    - onset_time: onset time in yyyy-MM-dd HHmmssZ format
- medications:
    - lastupdate: last update time in yyyy-MM-dd HHmmssZ format
    - medicine_name: the name of the medicine
    - medicine_code: the IDC10 code system id e.g.: 2.16.840.1.113883.6.3
    - dose_quantity: the quantity of each dose
    - dose_unit: the unit of each dose, e.g.: tablets, mg
    - dose_frequency_value: the frequency of the dose
    - dose_frequency_unit: the frequency unit of the dose, e.g.: min, h
    - start_time: the start time of the medication in yyyy-MM-dd HHmmssZ format
    - end_time: the end time of the medication in yyyy-MM-dd HHmmssZ format
- vital_signs
    - lastupdate: last update time in yyyy-MM-dd HHmmssZ format
    - description: the descirption of the vital signs
    - vital_sign_code: LOINC code, e.g. 9279-1
    - value: the value of the vital sign
    - unit: the unit of the vital sign
    - date_measured: the time of the vital sign is measured in in yyyy-MM-dd HHmmssZ format

The user profile API contains the following methods. Method 1) accepts user profile in full or partial set as JSON format and stores/updates them into the MyHealthAvatar data repository. Method 2) obtains the full profile of the authenticated user. Method 3)-8) obtains the user profile for the authenticated user by sections including personal information, insurance, allergy, diagnosis, medication and vital sign.

1) [POST] http://myhealthavatar.org/mha/api/v2/user/full_profile
2) [GET] http://myhealthavatar.org/mha/api/v2/user/full_profile
3) [GET] http://myhealthavatar.org/mha/api/v2/user/personal_information
4) [GET] http://myhealthavatar.org/mha/api/v2/user/insurance
5) [GET] http://myhealthavatar.org/mha/api/v2/user/allergy
6) [GET] http://myhealthavatar.org/mha/api/v2/user/diagnosis
7) [GET] http://myhealthavatar.org/mha/api/v2/user/medication
8) [GET] http://myhealthavatar.org/mha/api/v2/user/vital_sign

Below is a sample request for the POST method of the API.

```
{
  "lastupdate": "2012-12-11 22:00:00+0000",
  "given_name": "John",
  "surname": "Smith",
  "gender": "male",
  "dob": "1970-10-10",
  "occupation": "Lecturer",
  "email": "xx@example.com",
  "phone": "004401234567890",
  "mobile": "004401234567890",
  "address_home": [
    {
      "street": "20 Efg",
      "city": "Abcd",
      "state": "Bedfordshire",
      "country": "United Kingdom",
      "postcode": "LU1 4JU"
    }
  ],
  "address_work": [
    {
        "street": "12 Abc",
        "city": "Abcd",
        "state": "Bedfordshire",
        "country": "United Kingdom",
        "postcode": "LU1 3JU"
      }
    ],
    "private_contacts": [
      {
        "user_id": "ab56cd75-972a-423f-bd91-ff21f5c929b3",
        "relationship": "guardian"
      },
      {
        "user_id": "ab56cd75-972a-423f-bd91-ff21f5c929b4",
        "relationship": "spouse"
      },
      {
        "user_id": "ab56cd75-972a-423f-bd91-ff21f5c929b5",
        "relationship": "friend"
      },
      {
        "relationship": "friend",
        "given_name": "Peter",
        "surname": "Smith",
        "contact": {
          "email": "xx@example.com",
          "phone": "004401234567890",
          "mobile": "004401234567890"
        }
      }
    ],
    "medical_contacts": [
      {
        "place_id": "eb56cd75-972a-423f-bd91-ff21f5c929b5",
      "user_id": "eb56cd75-972a-423f-bd91-ff21f5c929b4",
      "contact_type": "cardiologist"
    },
    {
      "institution_name": "Luton&Dustable Hospital",
        "street": "30 Abc",
        "city": "Abcd",
        "state": "Bedfordshire",
        "country": "United Kingdom",
        "postcode": "LU2 3JU",
      "user_id": "eb56cd75-972a-423f-bd91-ff21f5c929b4",
      "contact_type": "cardiologist"
    },
    {
      "institution_name": "Luton&Dustable Hospital",
        "street": "30 Abc",
        "city": "Abcd",
        "state": "Bedfordshire",
        "country": "United Kingdom",
        "postcode": "LU2 3JU",
      "given_name": "Peterson",
      "surname": "Smith",
      "contact": {
        "email": "xx@example.com",
        "phone": "004401234567890",
        "mobile": "004401234567890"
      },
      "contact_type": "cardiologist"
    }
  ],
  "insurance": [
    {
      "insurance_number": "X12345",
      "insurance_contact": {
          "name": "ABC",
          "street": "30 Abc",
          "city": "Abcd",
          "state": "Bedfordshire",
          "country": "United Kingdom",
          "postcode": "LU2 3JU",
          "phone": [
            "1234567890",
            "0987654321"
          ]
        }
      }
    ],
    "allergies": [
    {
          agent: "eggs",
          description: "bird-fanciers' lung",
          onset_date: "2012-08-29"
    }
    ],
    "diagnosis": [
      {
        "problem_id": "2.16.840.1.113883.6.3",
        "description": "abc",
        "onset_time": "2012-08-29 22:00:00+0000"
      }
    ],
    "medications": [
      {
        "medicine_name": "abcd",
        "medicine_code": "2.16.840.1.113883.6.3",
        "dose_quantity": 1.5,
        "dose_unit": "ml",
        "dose_frequency_value": 4,
      "dose_frequency_unit": "hour",
      "start_time": "2012-08-29 22:00:00+0000",
      "end_time": "2012-08-31 22:00:00+0000"
    }
  ],
  "vital_signs": [
    {
      "description": "abcde",
      "vital_sign_code": "9279-1",
      "value": 20.5,
      "unit": "min",
      "date_measured": "2012-08-31 22:00:00+0000"
    }
  ]
}
```

Below is a sample response from the GET methods of the API.

```
{
  "lastupdate": "2012-12-22 22:00:00+0000",
  "local_id": "817c2199-9fd9-497c-8d9f-d8559206244d",
  "personal_information": {
    "basic": {
      "lastupdate": "2014-11-03 10:43:43+0000",
      "given_name": "John",
      "surname": "Smith",
      "gender": "male",
      "dob": "1970-10-10",
      "occupation": "Lecturer",
      "contact": {
        "email": [
          "xx@example.com"
        ],
        "phone": [
          "004401234567890"
        ],
        "mobile": [
          "004401234567890"
        ],
        "address_home": [
          {
            "street": "20 Efg",
            "city": "Abcd",
            "state": "Bedfordshire",
            "country": "United Kingdom",
            "postcode": "LU1 4JU"
          }
        ],
        "address_work": [
          {
            "street": "12 Abc",
            "city": "Abcd",
            "state": "Bedfordshire",
            "country": "United Kingdom",
            "postcode": "LU1 3JU"
          }
        ]
      }
    },
    "private_contacts": [
      {
        "lastupdate": "2014-11-03 10:43:43+0000",
        "relationship": "friend",
        "user_id": "ab56cd75-972a-423f-bd91-ff21f5c929b5"
      },
      {
        "lastupdate": "2014-11-03 10:43:43+0000",
        "relationship": "guardian",
        "user_id": "ab56cd75-972a-423f-bd91-ff21f5c929b3"
    },
    {
      "lastupdate": "2014-11-03 10:43:43+0000",
      "relationship": "spouse",
      "user_id": "ab56cd75-972a-423f-bd91-ff21f5c929b4"
    },
    {
      "lastupdate": "2014-11-03 10:43:43+0000",
      "relationship": "friend",
      "contact": {
        "lastupdate": null,
        "given_name": "Peter",
        "surname": "Smith",
        "contact": {
          "email": [
            "xx@example.com"
          ],
          "phone": [
            "004401234567890"
          ],
          "mobile": [
            "004401234567890"
          ]
        }
      }
    }
  ],
  "medical_contacts": [
    {
      "lastupdate": "2014-11-03 10:43:43+0000",
      "institution_name": "Luton&Dustable Hospital",
      "address": {
        "street": "30 Abc",
        "city": "Abcd",
        "state": "Bedfordshire",
        "country": "United Kingdom",
        "postcode": "LU2 3JU"
      },
      "user_id": "eb56cd75-972a-423f-bd91-ff21f5c929b4",
      "contact_type": "cardiologist"
    },
    {
      "lastupdate": "2014-11-03 10:43:43+0000",
      "place_id": "eb56cd75-972a-423f-bd91-ff21f5c929b5",
      "user_id": "eb56cd75-972a-423f-bd91-ff21f5c929b4",
      "contact_type": "cardiologist"
    },
    {
      "lastupdate": "2014-11-03 10:43:43+0000",
      "institution_name": "Luton&Dustable Hospital",
      "address": {
        "street": "30 Abc",
        "city": "Abcd",
        "state": "Bedfordshire",
        "country": "United Kingdom",
        "postcode": "LU2 3JU"
      },
      "contact_type": "cardiologist",
      "contact": {
        "lastupdate": null,
        "given_name": "Peterson",
        "surname": "Smith",
        "contact": {
          "email": [
            "xx@example.com"
          ],
          "phone": [
            "004401234567890"
          ],
          "mobile": [
            "004401234567890"
          ]
        }
      }
    }
  ]
},
"insurance": [
  {
    "lastupdate": "2012-12-20 22:00:00+0000",
    "insurance_number": "X12346",
    "insurance_contact": "9b02540d-7343-489a-8040-86fc423a7d34"
  },
  {
    "lastupdate": "2014-11-03 10:43:43+0000",
    "insurance_number": "X12345",
    "insurance_contact": "71a0e199-d1a3-4284-b7f3-5a5bde120c43"
  }
],
"allergies": [
  {
    "lastupdate": "2012-12-22 22:00:00+0000",
    "agent": "eggs",
    "description": "bird-fanciers' lung",
    "onset_date": "2012-08-29"
  },
  {
    "lastupdate": "2014-11-03 10:43:43+0000",
    "agent": "eggs",
    "description": "bird-fanciers' lung",
    "onset_date": "2012-08-29"
  }
  ],
  "diagnosis": [
    {
      "lastupdate": "2014-11-03 10:43:43+0000",
      "description": "abc",
      "problem_id": "2.16.840.1.113883.6.3",
      "onset_time": "2012-08-29 23:00:00+0100"
    }
  ],
  "medications": [
    {
      "lastupdate": "2014-11-03 10:43:43+0000",
      "medicine_name": "abcd",
      "medicine_code": "2.16.840.1.113883.6.3",
      "dose_quantity": 1.5,
      "dose_unit": "ml",
      "dose_frequency_value": 4,
      "dose_frequency_unit": "hour",
      "start_time": "2012-08-29 23:00:00+0100",
      "end_time": "2012-08-31 23:00:00+0100"
    }
  ],
  "vital_signs": [
    {
      "lastupdate": "2014-11-03 10:43:43+0000",
      "description": "abcde",
      "vital_sign_code": "9279-1",
      "value": 20.5,
      "unit": "min",
      "date_measured": "2012-08-31 23:00:00+0100"
    }
  ]
```

4.1.6 APIs for Sharing

The sharing API enables the authenticated user to become a follower or a followee of other MyHealthAvatar user, so they can share information. The users are identified by user name/id in these APIs. The API contains the following methods. Methods 1) and 2) allow the authenticated user to send s request to become a follower or followee of another MyHealthAvatar user. Methods 3) and 4) allow the authenticated user to accept a request to become a follower or followee of another MyHealthAvatar user. Method 5) and 6) update the last check time of the follower viewing the shared authenticated user's data. Method 7) and 8) allow the authenticated user to delete a follower or a followee. Method 9) and 10) returns a list of followers and followees of the specific user, respectively. Method 11) and 12) returns a list of follower and followee request of the specific user, respectively.

```
1)  [POST]
    http://myhealthavatar.org/mha/api/v2/user/followers/request?user_name=**
2)  [POST]
    http://myhealthavatar.org/mha/api/v2/user/followees/request?user_name=**
3)  [POST]
    http://myhealthavatar.org/mha/api/v2/user/followers/accept?user_name=**
4)  [POST]
    http://myhealthavatar.org/mha/api/v2/user/followees/accept?user_name=**
5)  [POST]
    http://myhealthavatar.org/mha/api/v2/user/followers/update?user_name=**
6)  [POST]
    http://myhealthavatar.org/mha/api/v2/user/followees/update?user_name=**
7)  [POST]
    http://myhealthavatar.org/mha/api/v2/user/followers/delete?user_name=**
8)  [POST]
    http://myhealthavatar.org/mha/api/v2/user/followees/delete?user_name=**
9)  [GET] http://myhealthavatar.org/mha/api/v2/user/followers
10) [GET] http://myhealthavatar.org/mha/api/v2/user/followees
11) [GET] http://myhealthavatar.org/mha/api/v2/user/followers/request
12)  [GET] http://myhealthavatar.org/mha/api/v2/user/followees/request
```

4.2 APIs for General MyHealthAvatar Data

4.2.1 APIs for Vital Sign Codes

Vital sign code API contains the following data.

- code: LOINC code
- description: description of the code
- units: units of the code
- lastupdate: last update time in yyyy-MM-dd HHmmssZ format

The available vital sign codes are pre-added into the data repository. The API provides the GET method to obtain all available codes.

```
[GET] http://myhealthavatar.org/mha/api/v2/vital_sign_codes
```

Below is a sample response from the API.

```
[
  {
    "code": "8306-3",
    "description": "BODY HEIGHT^LYING",
    "units": "m, cm",
    "lastupdate": "2014-11-03 14:43:07+0000"
  },
  {
    "code": "2710-2",
    "description": "OXYGEN SATURATION",
    "units": "%",
    "lastupdate": "2014-11-03 14:43:07+0000"
  },
  {
    "code": "8462-4",
    "description": "INTRAVASCULAR SYSTOLIC",
    "units": "mm[Hg]",
    "lastupdate": "2014-11-03 14:43:07+0000"
  },
  {
    "code": "8867-4",
    "description": "HEART RATE",
    "units": "/min",
    "lastupdate": "2014-11-03 14:43:07+0000"
  },
  {
    "code": "8480-6",
    "description": "INTRAVASCULAR SYSTOLIC",
    "units": "mm[Hg]",
    "lastupdate": "2014-11-03 14:43:07+0000"
  },
  {
    "code": "8287-5",
    "description": "CIRCUMFRENCE.OCCIPITAL-FRONTAL (TAPE MEASURE)",
    "units": "m, cm",
    "lastupdate": "2014-11-03 14:43:07+0000"
  },
  {
    "code": "9279-1",
    "description": "RESPIRATION RATE",
    "units": "/min",
    "lastupdate": "2014-11-03 14:43:07+0000"
  },
  {
    "code": "8302-2",
    "description": "BODY HEIGHT (MEASURED)",
    "units": "m, cm",
    "lastupdate": "2014-11-03 14:43:07+0000"
  },
  {
    "code": "3141-9",
    "description": "BODY WEIGHT (MEASURED)",
    "units": "kg, g",
    "lastupdate": "2014-11-03 14:43:07+0000"
  },
  {
    "code": "8310-5",
    "description": "BODY TEMPERATURE",
    "units": "cel",
    "lastupdate": "2014-11-03 14:43:07+0000"
  }
]
```

4.2.2 APIs for General Insurance Contact

General Insurance Contact API handles the following data. local_id: id of the insurance contact in the MHA system.

- name: name of the insurance company
- address: address of the insurance contact
- phone: contact phone numbers

The data are obtained either while users providing relevant information into the user profile or pre-loaded into the data repository. Two GET methods are implemented to obtain the stored insurance contact information. Method 1) returns all available contact information; Method 2) return the specific contact information by id.

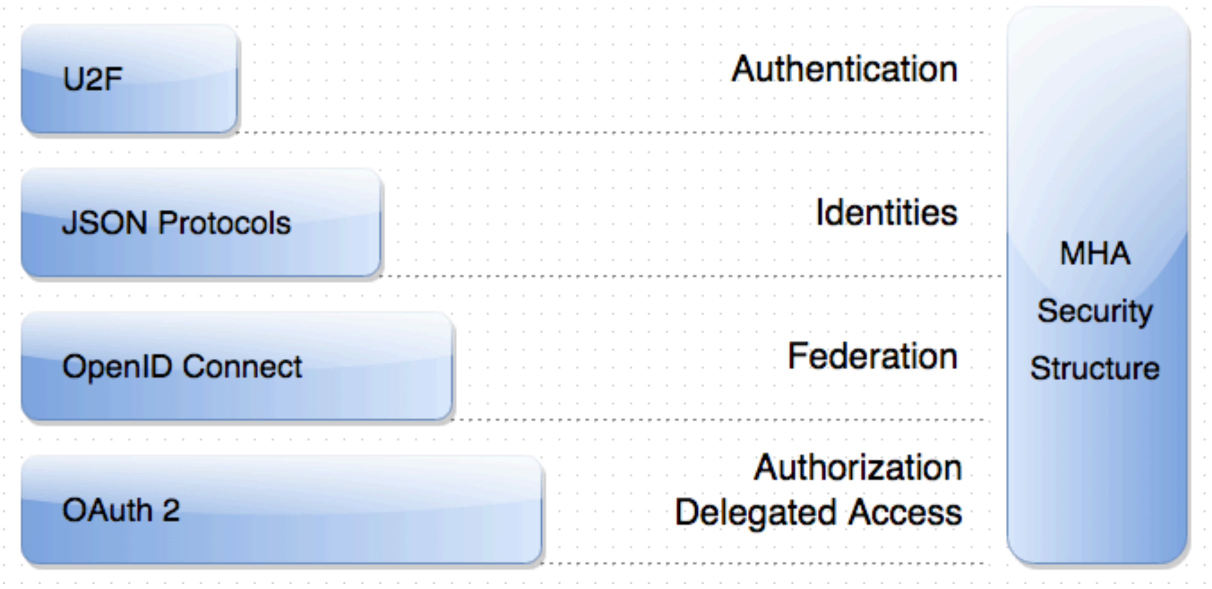
1) `[GET] http://myhealthavatar.org/mha/api/v2/insurance_contact`
2) `[GET] http://myhealthavatar.org/mha/api/v2/insurance_contact?id=[uuid]`

Below is a sample response from the API.

```
[
  {
    "local_id": "9dec4b05-1aff-4b9a-b872-29824a5a6ca1",
    "name": "ABC",
    "address": {
      "street": "30 Abc",
      "city": "Abcd",
      "state": "Bedfordshire",
      "country": "United Kingdom",
      "postcode": "LU2 3JU"
    },
    "phone": [
      "0987654321",
      "1234567890"
    ]
  },
  {
    "local_id": "22be8498-bba4-4756-afa4-f7559cd99945",
    "name": "ABC",
    "address": {
      "street": "30 Abc",
      "city": "Abcd",
      "state": "Bedfordshire",
      "country": "United Kingdom",
      "postcode": "LU2 3JU"
    },
    "phone": [
      "0987654321",
      "1234567890"
    ]
  }
]
```

5 Security of MyHealthAvatar APIs

The MHA security is adopting current cutting edge technologies, and guided by Neo-security stack, which could be illustrated as following graph:



For authentication, MHA API uses MHA web application as identity provider, and would propose Universal Secord Factor (U2F) protocol to enhance the current password infrastructure of MHA web application. A strong second factor would augment the current security system, free application like Google Authenticator on Android mobile devices could serve as the second factor.

Recently a list of JSON based protocols are formalized for identification, which are lightweight compared to related XML protocols. MHA will take the trend into consideration, and adopt the protocols when they are mature enough for the purpose. The protocols includes JSON Web Token (JWT) which guide the way to encode token in JSON format, JSON Web Signature (JWS) for digital signatures and JSON Web Key (JWK) for symmetric and asymmetric encryptions. The JSON format create shorter string which could be used in HTTP(s) headers and query strings.

OpenID Connect is effectively the third edition of OpenID, which is not compatible with previous versions due to the complete rewrite atop of OAuth 2 protocol. OpenID Connect achieves higher Level of Assurance (LoA) compared to other similar protocols (namely SAML and WS-Federation). OpenID is to be adopted by MHA API and MHA web application to support federation and Single Sign On (SSO).

OAuth 2 is essential foundation for MHA API services, which provides following benefits:

- Delegated access to third-parties applications (e.g. MHA mobile application)
- Avoid users to share credential information to third-parties
- Allow users to revoke access of third-parties

Currently, MHA API mainly supports the grant types of: Authorization Code and Implicit. Third parties need to apply for ClientId and ClientSecret, which are reviewed, issued and monitored by MHA administration team. Following key information is stored for third parties:

- Third-parties contact email, website
- Permission to third-parties (read, write, etc.)
- Call-back URL for third-parties

HTTPS should always use in MHA API interactions instead of HTTP, which protect users' information from leaking out. MHA API suggests and requests third-parties application to also use cutting-edge security mechanisms to protect access token and information retrieved from MHA platform, third-parties should notify MHA API that any infringement of their system so that MHA API would able to revoke the client's access to prevent further impact on users' security.

MHA API will test, review and apply any security patches from upstream (OS, dependencies libraries, etc.), and aims to processing requests in timely fashion, with accurate, complete and authorized response (to API callers). ISO/IEC standards will be referenced when dealing with security and privacy part of the system, which includes but not limits to ISO/IEC 27002:2013 for information security and 27018:2014 for personal identifiable information (PII) protection.

6 Conclusion

This report has presented the MyHealthAvatar demonstration requirements on the API and the current version 2 implementation of the MyHealthAvatar API. The security mechanism over the API has been implemented to ensure that the user data are authorised and the client applications are authenticated before data being accessed.

The implementation of the MyHealthAvatar at this stage has been focusing on the management and use of the data in the data repository. The next stage implementation will include the functions for the semantic data repository. More APIs may be required when the implementation requirements of the demos become clearer. The whole API implementation and evaluation work will continue through the end project to support the successful deployment of the four high end clinical demos and the overall MyHealthAvatar demonstration platform.

Appendix 1 – Abbreviations and acronyms

APIs	Application Interfaces
CHF	Congestive Heart Failure
DIAB-EME	Diabetes and Emergency
HTTP	The Hypertext Transfer Protocol
HTTPS	HTTP over Secure Socket Layer
JSON	JavaScript Object Notation
MHA	MyHealthAvatar
OST	Osteoarthritis
REST	Representation State Transfer