**Outreach Mobile Web App High Level Design**

Draft 1.1

Saravanan Shanmugam

Last Updated: 21-Nov-14

**Abstract**

This document describes the scope, high level design, database mappings and API specifications for the Outreach mobile web application.

Contents

2 Version History 4

3 References 4

4 Abbreviations and Acronyms 4

5 Assumptions, Issues and Risks 5

5.1 Assumptions 5

5.2 Issues 5

5.3 Risks 5

6 Introduction 6

6.1 Scope 6

6.2 Out of Scope 6

7 Solution Design 6

7.1 Architecture 6

7.1.1 Outreach Mobile UI 7

7.1.2 Outreach Mobile Web Service 7

7.2 Security Model 7

7.2.1 OAuth 2.0 Implementation 7

7.3 Static Model 8

7.4 Dynamic Design 8

7.4.1 E2E Flow 8

7.5 Database Design 8

7.6 Interface Specification and data mappings 10

7.6.1 Generic Response Type 10

7.6.2 Login Service 10

7.6.3 Forgot Password Service 10

7.6.4 Client Search Service 11

7.6.5 Add Client Service 13

7.6.6 Add Client Information Service 15

7.6.7 Add Disability / Health Condition Service 17

7.7 Deployment Design 19

8 Volumetric 19

9 Backup Strategy 19

10 Implementation Strategy 19

10.1 Technology Selection 19

# Version History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Description** |
| **Draft 1.** | Nov-2014 | Innoppl | Initial Draft |

Table 1: Version History

# References

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No.** | **Version** | **Source** | **Author** | **Document Name** |
|  | v1.1 | First draft | Innoppl | Outreach\_SRS\_v1.1.pdf  *(Software Requirements Specification Document)* |
|  | N/A | Client Supplied Product | Innoppl | ApprovedDesigns.zip  *(Mobile Phone Screen Mockups)* |
|  | v1 | Client Supplied Product | Saravanan  Innoppl | OutreachMobileAppAPISpecification.docx  *(Initial draft API Specification)* |
|  | N/A | Client Supplied Product | Innoppl | Database\_conceptual\_diagram.jpeg  *(Database )* |
|  | N/A | Client Supplied Product | Innoppl | erd.png |
|  | V4.0 | www.hudhdx.info | N/A | HMIS\_Logical\_Model.pdf |

Table 2: References

# Abbreviations and Acronyms

|  |  |
| --- | --- |
| **Abbreviation/Acronym** | **Description** |
| **UI** | User Interface |
| **N/A** | Not Available |
| **E2E** | End to End |
| **HTML** | HyperText Markup Language |
| **HTTP** | HyperText Transfer Protocol |
| **SSL** | Secure Socket Layer |
| **ERD** | Entity Relationship Diagram |
| **HUD** | U.S. Department of Housing and Urban Development |
| **HMIS** | Homeless Management Information Systems |
| **SRS** | Software Requirements Specification |
| **API** | Application Program Interface |
| **AJAX** | Asynchronous Javascript for XML |

Table 3: Abbreviations and Acronyms

# Assumptions, Issues and Risks

## Assumptions

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No.** | **Assumption** | **Comments** | **Status** |
|  | The layout specified in ApprovedDesigns.zip [Ref.2.] will be used for both phone and tablet resolutions. Desktop layout is not in the current scope. | Based on discussion held with Saravanan, it is agreed that the layout provided will be used for both phone and tablet resolution. Desktop layout is not in our current scope. | Closed |
|  |  |  |  |
|  |  |  |  |

Table 4: Assumptions

## Issues

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No.** | **Assumption** | **Comments** | **Status** |
|  |  |  |  |
|  |  |  |  |

Table 5: Issues

## Risks

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No.** | **Assumption** | **Comments** | **Status** |
|  |  |  |  |
|  |  |  |  |

Table 6: Risks

# Introduction

## Scope

* All use cases specified in Outreach\_SRS\_v1.1.pdfdocument.
* The layout design is based on the approved design mockup. The same layout will be used for phone and tablet.
* Outreach Mobile Web application Web Services design, implementation and deployment.
* Outreach UI design, Implementation and deployment.

## Out of Scope

* Layout for desktop is not in the current scope of work.
* Database Design

# Solution Design

## Architecture

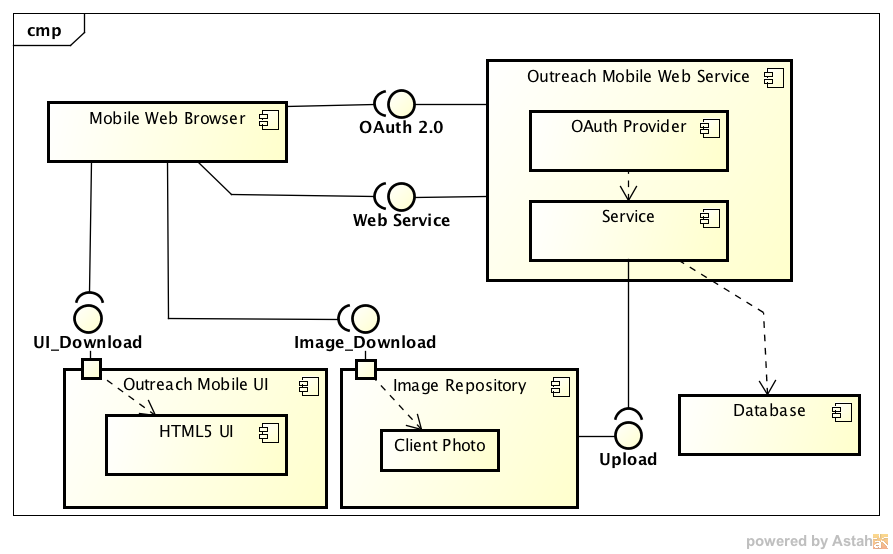


Figure 1: Outreach Mobile Web UML Component Diagram

The above figure (Figure 1) provides a high level detail of components in the Outreach mobile web application.

### Outreach Mobile UI

Approved

### Outreach Mobile Web Service

To be developed.

## Security Model

The authentication and authorization is handled using the [[1]](#footnote-2)OAuth 2.0 mechanism. OAuth 2.0 is an open authorization protocol specification defined by IETF OAuth WG (Working Group) which enables applications to access each other’s data.

Oauth 2.0 is selected because of the following fundamental considerations:

* The entire application is developed as RESTful webservices and accessed by the Outreach HTML 5 based UI via Ajax. Since it is stateless in nature, the mechanisms of securing these services are different from standard web application where it is easily handled by session management, but in the case of REST, no session can be maintained as the calling point may or may not be a web browser.
* The RESTful services will be accessed by 3rd party applications.
* The application can be accessed from native/hybrid mobile applications.

### OAuth 2.0 Implementation

Since the core functionalities are implemented as RESTful web services and accessed using Ajax from HTML 5 web pages, we will be using the OAuth 2.0 generic Implicit Grant mechanism. The following Figure 2gives a high-level overview of the OAuth 2.0 implicit grant mechanism. Since the access token is sent as part of the HTTP response, it is visible over the wire in Internet and so the transport has to be secured using SSL.

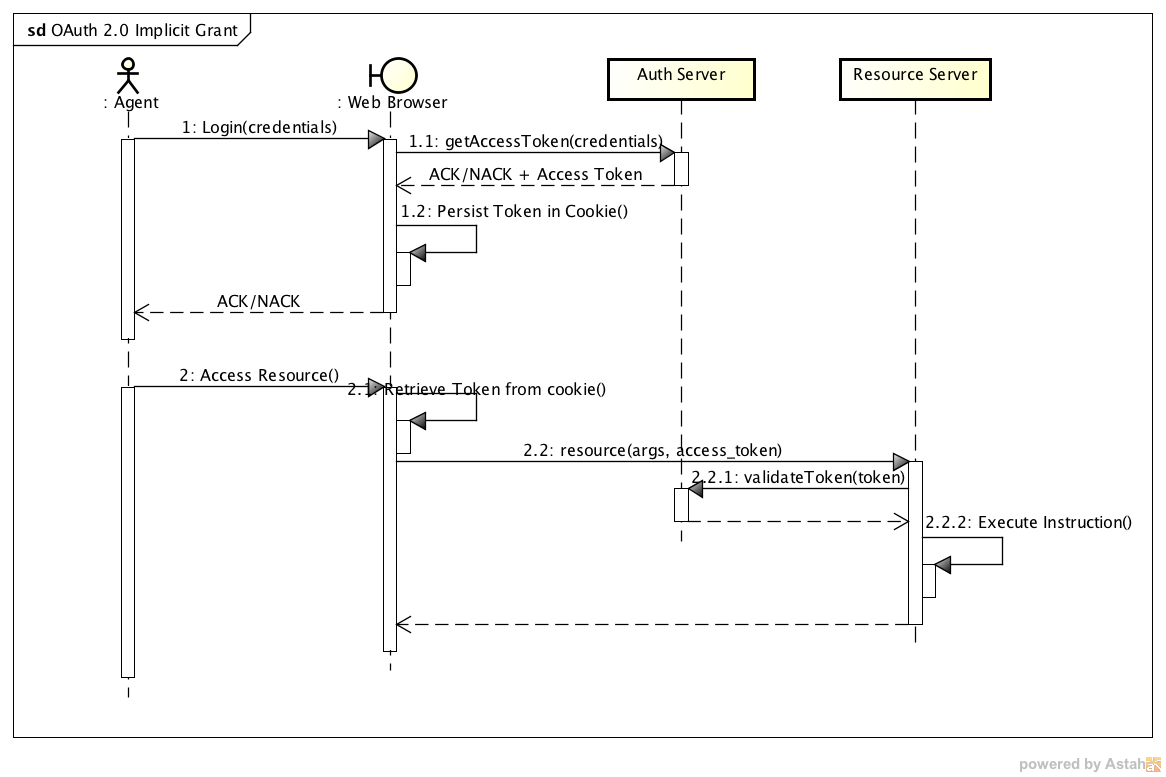


Figure 2: OAuth 2.0 Implicit Grant UML Sequence Diagram

## Static Model

## Dynamic Design

### E2E Flow

## Database Design

Ref.5.

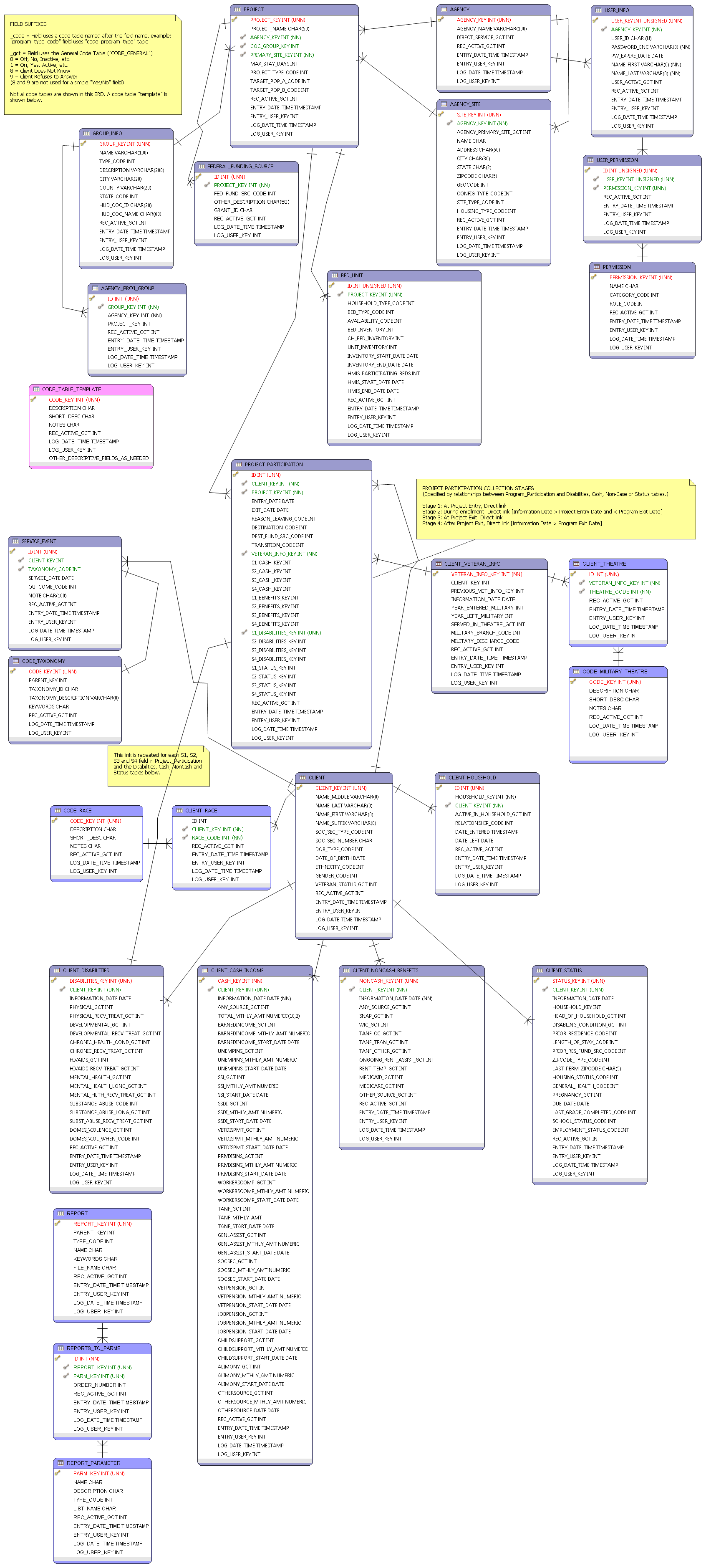


Figure 3: ERD

## Interface Specification and data mappings

### Generic Response Type

All response to the below APIs will have the following structure.

{

"response\_header": {

"code": 0,

"message": "",

"total\_count": 234

},

"body": [

Array of JSON Objects

]

}

The code field identifies whether the response is success or failure. If the value of code is 0 then it is a success else it is a failure. The following is a sample failure response. Only response\_header is sent for failure scenarios.

{

"response\_header": {

"code": 1001,

"message": "No records found",

}

}

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Mandatory(M)/Optional(O)** |
| **response\_header** | A generic response header | M |

Table 7: Generic Response Header

### Login Service

All web services are authenticated and authorized using the OAuth 2.0 implicit grant flow. Please refer section 6.2 for the OAuth 2.0 implicit grant flow.

#### Data Mapping

### Forgot Password Service

#### Web Service Specification

|  |  |
| --- | --- |
|  |  |
| END POINT | /login |
| HTTP METHOD | POST |
| TRANSPORT | SSL |
| AUTHENTICATION | NONE |
| AUTHORIZATION | NONE |
| REQUEST |  |
| SUCCESS RESPONSE |  |
| FAILURE RESPONSE |  |

#### Data Mapping

### Client Search Service

#### UI Screen

|  |  |
| --- | --- |
| Figure 4: Client Search Screen | Figure 5: Client Search Results Screen |

#### Web Service Specification

|  |  |
| --- | --- |
|  |  |
| END POINT | /client/search |
| HTTP METHOD | POST |
| TRANSPORT | SSL |
| AUTHENTICATION | TOKEN BASED AUTHENTICATION |
| HTTP HEADERS | Content-Type:application/json  Accept:application/json |
| AUTHORIZATION |  |
| REQUEST | {  "first\_name": "john",  "last\_name": "smith",  "client\_key": "Cl123",  "ssn": "123456789",  "age\_range": "18,25"  } |
| SUCCESS RESPONSE | HTTP 200 OK  {  "response\_header": {  "code": 0,  "total\_count": 234  },  "body": [  {  "id": "1",  "name": "jason",  "client\_code": "cl123",  "image": "<Image URL>",  "age": "19"  },  {  "id": "3",  "name": "sdfon",  "client\_code": "cl123",  "image": "<Image URL>",  "age": "39"  }  ]  } |
| FAILURE RESPONSE | HTTP 200 OK  {  "response\_header": {  "code": 1001,  "message": "No Records Found"  }  } |

#### Request Data Mapping

| **Field Name** | **Required?** | **Data Type** | **DB Table** | **DB Column** |
| --- | --- | --- | --- | --- |
| **first\_name** | N | String | CLIENT | NAME\_FIRST |
| **last\_name** | N | String | CLIENT | NAME\_LAST |
| **client\_key** | N | String | ? | ? |
| **ssn** | N | String | CLIENT | SOC\_SEC\_NUMBER |
| **age\_range** | N | String | CLIENT | DATE\_OF\_BIRTH  (Age needs to be calculated based on DOB) |

Table 8: Client Search Request Data Mapping

#### Response Data Mapping

| **Field Name** | **Null (Y/N)** | **Data Type** | **DB Table** | **DB Column** |
| --- | --- | --- | --- | --- |
| **id** | N | String | CLIENT | CLIENT\_KEY |
| **name** | N | String | CLIENT | ? |
| **client\_code** | ? | ? | ? | ? |
| **image** | Y | String | ? | ? |
| **age** | Y | Date | CLIENT | DATE\_OF\_BIRTH  (Age needs to be calculated based on DOB) |

Table 9: Client Search Response Data Mapping

7401395034

9940479752 – Palanivel

### Add Client Service

#### UI Screen

|  |  |
| --- | --- |
| Figure 6: Consent Screen | Figure 7: Add/Edit Client Screen |

#### Web Service Specification

|  |  |
| --- | --- |
|  |  |
| END POINT | /client |
| HTTP METHOD | POST |
| TRANSPORT | SSL |
| AUTHENTICATION | TOKEN BASED AUTHENTICATION |
| AUTHORIZATION |  |
| HTTP HEADERS | Content-Type:application/json  Accept:application/json |
| REQUEST | {  "client\_code": "cl123",  "fname": "Malik",  "lname": "john",  "gender": "john",  "data\_quality": "1",  "ssn": "123456789",  "ssn\_quality": "2",  "dob": "12/03/2014",  "dob\_type": "2",  "race": "1",  "ethnicity": "1",  "image": "image.png"  } |
| SUCCESS RESPONSE | HTTP 200 OK  {  "response\_header": {  "code": 0  },  "body": [  {  "id": "1",  "name": "jason",  "image": "Image URL",  "age": "19"  }  ]  } |
| FAILURE RESPONSE | Error Codes will be |

#### Request Data Mapping

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Required?** | **Data Type** | **DB Table** | **DB Column** |
| **client\_code** | Y | String | ? | ? |
| **first\_name** | N | String | CLIENT | NAME\_FIRST |
| **last\_name** | N | String | CLIENT | NAME\_LAST |
| **client\_key** | N | String | ? | ? |
| **ssn** | N | String | CLIENT | SOC\_SEC\_NUMBER |
| **age\_range** | N | String | CLIENT | DATE\_OF\_BIRTH  (Age needs to be calculated based on DOB) |

Table 10: Add Client Request Data Mapping

#### Response Data Mapping

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Null (Y/N)** | **Data Type** | **DB Table** | **DB Column** |
| **id** | N | String | CLIENT | CLIENT\_KEY |
| **name** | N | String | CLIENT | ? |
| **image** | Y | String | ? | ? |
| **age** | Y | Date | CLIENT | DATE\_OF\_BIRTH  (Age needs to be calculated based on DOB) |

Table 11: Add Client Response Data Mapping

### Add Client Information Service

#### UI Screen

|  |
| --- |
| 7-client-information.png  Figure 8: Add Client Information Screen |

#### Web Service Specification

|  |  |
| --- | --- |
|  |  |
| END POINT | /client\_info |
| HTTP METHOD | POST |
| TRANSPORT | SSL |
| AUTHENTICATION | TOKEN BASED AUTHENTICATION |
| AUTHORIZATION |  |
| HTTP HEADERS | Content-Type:application/json  Accept:application/json |
| REQUEST | {  "client\_id": 123,  "veteran\_status": 1,  "disabling\_condition": 1,  "residence\_prior": 12,  "residence\_prior\_length\_of\_stay": 1,  "months\_homeless\_past\_three\_years": 333  } |
| SUCCESS RESPONSE | HTTP 200 OK  {  "response\_header": {  "code": 0  }  } |
| FAILURE RESPONSE |  |

#### Request Data Mapping

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Required?** | **Data Type** | **DB Table** | **DB Column** |
| **client\_id** | Y | Numeric | CLIENT | CLIENT\_KEY |
| **veteran\_status** | Y | Numeric | CLIENT | VETERAN\_STATUS |
| **disabling\_condition** | Y | Numeric | CLIENT\_STATUS | DISABLING\_CONDITION\_GCT |
| **residence\_prior** | Y | Numeric | CLIENT\_STATUS | PRIOR\_RESIDENCE\_CODE |
| **residence\_prior\_length\_of\_stay** | Y | Numeric | CLIENT\_STATUS | LENGTH\_OF\_STAY\_CODE |
| **months\_homeless\_past\_three\_years** | Y | Numeric | ? | ? |

Table 12: Add Client Information Request Data Mapping

#### Response Data Mapping

Body is not sent. Status Code 0 indicates success.

### Add Disability / Health Condition Service

#### UI Screen

|  |  |
| --- | --- |
| 8-1-disabilities-health-conditions.png  Figure 9: Disabilities | 8-2-disabilities-health-conditions.png  Figure 10: Physical Disability |
| 8-3-disabilities-health-conditions.png  Figure 11: Developmental Disability | 8-4-disabilities-health-conditions.png  Figure 12: Chronic Health Condition |
| 8-5-disabilities-health-conditions.png  Figure 13: Mental Health Problem | 8-6-disabilities-health-conditions.png  Figure 14: Substance Abuse Problem |

#### Web Service Specification

|  |  |
| --- | --- |
|  |  |
| END POINT | /disability\_health |
| HTTP METHOD | POST |
| TRANSPORT | SSL |
| AUTHENTICATION | TOKEN BASED AUTHENTICATION |
| AUTHORIZATION |  |
| HTTP HEADERS | Content-Type:application/json  Accept:application/json |
| REQUEST | {  "client\_id": 123,  "veteran\_status": 1,  "disabling\_condition": 1,  "residence\_prior": 12,  "residence\_prior\_length\_of\_stay": 1,  "months\_homeless\_past\_three\_years": 333  } |
| SUCCESS RESPONSE | HTTP 200 OK  {  "response\_header": {  "code": 0  }  } |
| FAILURE RESPONSE |  |

#### Request Data Mapping

Disability Type 5 = Physical Disability 6 = Developmental Disability 7 = Chronic Health Condition 8 = HIV/AIDS 9 = Mental Health Problem 10 = Substance Abuse

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Required?** | **Data Type** | **DB Table** | **DB Column** |
| **client\_id** | Y | Numeric | CLIENT | CLIENT\_KEY |
| **veteran\_status** | Y | Numeric | CLIENT | VETERAN\_STATUS |
| **disabling\_condition** | Y | Numeric | CLIENT\_STATUS | DISABLING\_CONDITION\_GCT |
| **residence\_prior** | Y | Numeric | CLIENT\_STATUS | PRIOR\_RESIDENCE\_CODE |
| **residence\_prior\_length\_of\_stay** | Y | Numeric | CLIENT\_STATUS | LENGTH\_OF\_STAY\_CODE |
| **months\_homeless\_past\_three\_years** | Y | Numeric | ? | ? |

Table 13: Add Client Information Request Data Mapping

#### Response Data Mapping

Body is not sent. Status Code 0 indicates success.

## Deployment Design

# Volumetric

# Backup Strategy

# Implementation Strategy

## Technology Selection

All tools and technologies that we use are either Open Source or available for free.

| **Business Functionality** | **Technology** | **Reason** |
| --- | --- | --- |
| **Solution Platform** | Java7 / JEE6 | The recommendation solution involves a highly scalable architecture with concurrency support. This is feasible with Java. The other alternatives are PHP & .Net.  PHP does not support concurrency. And we will end up with scalability issues when trying to update huge number of book records.  Microsoft solutions are expensive to implement and scale. |
| **Web Services** | Spring MVC Framework | Spring is the de-facto standard for implementing scalable web services and it considerably reduces time.  Easy to test |
| **OAuth Provider** | Spring OAuth Provider | Since we are going to use Spring Framework, Spring OAuth Provider is the default choice as it is easy to integrate and supports OAuth 2.0 specification. |
| **Automated Testing** | Spring Testing Framework | Easy to test by deploying to web servers automatically. |
| **Build Automation** | Maven | Maven is mature and is most widely used build automation tool. |
| **Database Access** | Hibernate JPA | Database agnostic, scalable and high performance ORM engine. Highly compatible with Spring Framework. |
| **Database** | MySQL |  |
| **JEE Web Server** | Jetty | Simple, Lightweight and scalable web server. |
| **Java Version** | JDK 7.0 | Latest stable version of Java |
| **JEE Version** | JEE 6 | Latest stable version of Java Enterprise Edition |

Table 14: Recommendation Service Technology Selection

**<<END OF DOCUMENT>>**

1. For more information about OAuth 2.0 please refer [https://tools.ietf.org/html/draft-ietf-oauth-v2-30#section-1.3.2](https://tools.ietf.org/html/draft-ietf-oauth-v2-30" \l "section-1.3.2) [↑](#footnote-ref-2)