# Solution for Consumer Health Data Aggregator Challenge-Technical Design Document

**Version 1.0**

HealthViewX
Payoda Inc.

## Contents

# 1. Introduction

## 1.1. Purpose

The purpose of this document is to outline the technical design of the FIT+ consumer application and provide an overview for the FIT+ app implementation.

Its main purpose is to -

- Provide the link between the Functional Specification and the detailed Technical Design documents
- Detail the functionality which will be provided by each component or group of components and show how the various components interact in the design
- Provide a basis for the FIT+ app's detailed design and development

This document is not intended to address installation and configuration details of the actual implementation.

## 1.2. Scope

The Application Design outlined in this document builds upon the scope defined in the Requirements phase.

## 1.3. Audience

The intended audiences for this document are Stakeholders, the project development teams, technical architects, database designers, testers and vendors.

## 1.4. Acronyms, Abbreviations, Terms and Definitions

Please refer to Appendix A for a list of all acronyms and abbreviations.

## 1.5. Document Organization

This document is organized into the following sections:

| | |
|---|---|
| Introduction | Provides information related to this document (e.g. purpose, term definitions etc.) |
| Design Overview | Describes the approach, architectural goals and constraints, Guiding principles, Design patterns used in design and development |
| Topology Diagram | Describes the various system components and the integration between them |
| Application Architecture | Describe the application architecture in terms of different layers of application. Description of the presentation layer, business layer, data access layer and resource layer and their relationship to each other. |
| Assumption and Constraints | Details various assumptions made during design and development of the Online Screening tool |
| Appendix A | Describes Acronyms, Abbreviations, Terms and Definitions |
| Appendix B | Lists all products and tools used in design and development |

# 2. Design Overview

## 2.1. Approach

This document is created and extended in multiple phases over the course of the project -

- *Requirements Phase* - During the Requirements Phase, the initial version of this document is created, describing the candidate architecture to be validated in the System Design Phase.

- *System Design Phase* - During the System Design phase, the Evolutionary Prototype is created and this document is finalized by establishing a sound architectural foundation for the Construction Phase.

- *Construction Phase* - During the Construction Phase, this document is not expected to change radically; it is mainly updated to reflect changes in any interface definitions.

- *Transition / Training Phase* - During the Transition/Training Phase, no further additions or modifications are made to this document.

## 2.2. Architectural Goals and Constraints

The overall architecture goals of the system is to provide a highly available and scalable FIT+ application for provider, to understand what programs or services are available and to determine if they are potentially eligible for those services.

The FIT+ application can be used in many ways, few are- *ƒ*

1. Patient can connect & configure with multiple wearable health devices
2. Patient can set goals on top of the connected health devices
3. Provide a bridge where patient and physician can communicate and share the health data status
4. Provider video conferencing and text interface between patient and physician
5. Alert frequency can be set by patient in case of critical reading.

A key Architectural goal is to leverage industry best practices for designing and developing a scalable, enterprise wide application. To meet this goal, the design of the FIT+ application will be based on the “GOF” (creational, structural, behavioral, visualization) patterns as well as the industry HIPAA standard development guidelines for building the FIT+ application.

## 2.3. Guiding Principles

Guiding principles provide a foundation upon which to develop the target architecture for the FIT+ application, in part by setting the standards and measures that the tool must satisfy. These in turn drive design principles that can be used to validate the design and ensure that it is aligned with Healthcare's overall Architecture, Design Principles and Standards. Some of the guiding principles that will be followed during the FIT+ application design and development are outlined below.

### 2.3.1. Scalable

Scalability is the ability of the platform to scale both up and down to support varying numbers of users or transaction volumes. The application should be able to scale horizontally (by adding more servers) or vertically (by increasing hardware capacity or software efficiency).

### 2.3.2. Flexible

Flexibility is the ability of the application to adapt and evolve to accommodate new requirements without affecting the existing operations. This relies on a modular architecture, which isolates the complexity of integration, presentation, and business logic from each other in order to allow for the easy integration of new technologies and processes within the application.

### 2.3.3. Standards-Based

Portal services will comply with established industry standards. The standards-compliance will not only apply to application development but also to design, platform/infrastructure and other parts of the FIT+ application. Examples of standards include HTML, XML, HL7, FHIR, HIPAA Security, PHI ...etc.

## 2.4. Design Patterns

Design patterns are elements of reusable object oriented software. A design pattern catalog is a repository of design patterns. Use of such patterns makes the design of an application transparent. These patterns have been used successfully by developers in their respective fields, and therefore, the pros and cons of the pattern (as well as implementation issues) are known beforehand. All design patterns are reusable and can be adapted to particular contexts.

## 2.5. Design Principles

Best practices and design principles will be applied in two main areas –

1. Presentation Services to individual desktops, mobile apps (Android & iOS) should be uncoupled and should adhere to design principle of HTML, iOS & Android.

2. Business Rules should be encoded within the application development framework,

   A. Business rules will need to be separated from the presentation and database frameworks
   B. Server applications are based on event-based systems. Complex server side event cascades will need to be supported.
   C. Standard frameworks for encoding business rules and events will need to be used.
   D. Adoption of a component based framework needs to be considered to promote reuse of information objects.
   E. Data encryption has to be followed across data storage and in motion
   F. WS Trust security standards need to be followed.

# 3. Topology Diagram

The diagram below provides an illustration of the System Architecture along with various system components that will be used in architecting the Online Screening Tool -

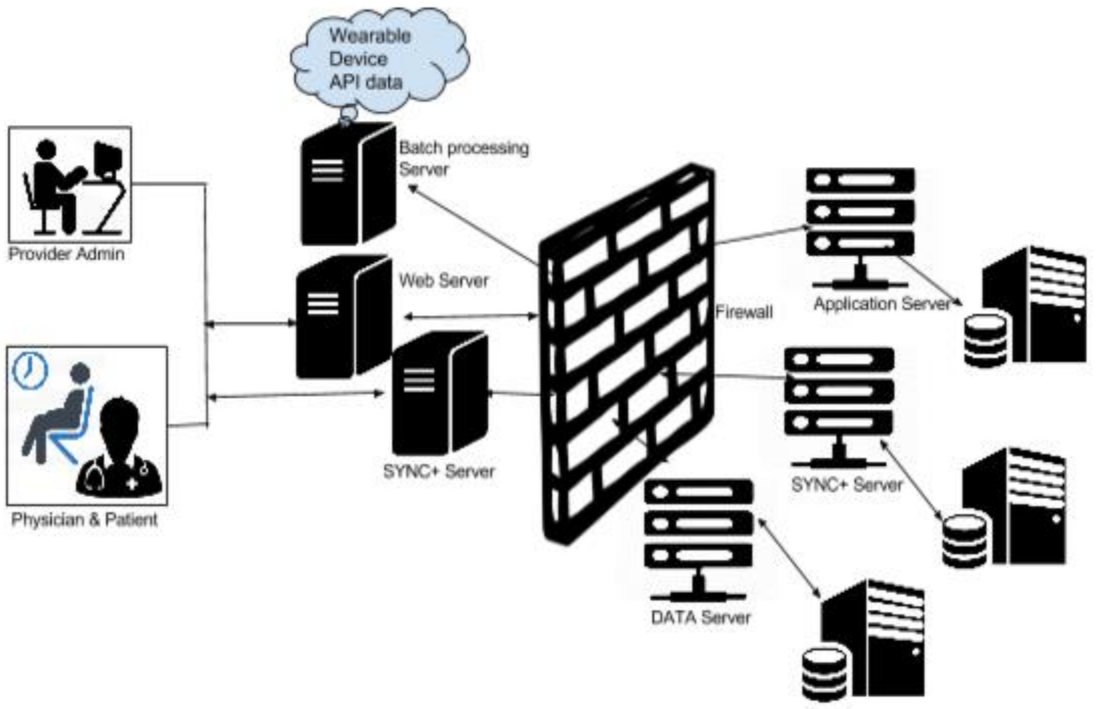


Fig 1: Topology Diagram

Interaction of software components along with its responsibilities is explained below -

**Web Server –** Web server is responsible for serving web pages, mostly HTML pages, via the HTTP protocol to clients. The Web server sends out web pages in response to requests from browsers. A page request is generated when a client clicks a link on a web page in the browser. For secure communication we use HTTPs protocol and all the REST APIs will be accessed via Web Server.

**XMPP Server –** XMPP server is responsible for serving chat, mostly XMPP queries, via the XMPP protocol to clients. The XMPP server sends out XML in response to requests from clients.

**Chat & Application Server** – Chat & Application server combined hosts the SYNC+ application and hosts the business logic and the business model classes of applications. It serves requests for dynamic HTTP / HTTPs web pages / API from Web servers also XMPP request from Chat server.

**Chat & Application Database –** SYNC+ database stores the application configuration, user store, roles, groups, permissions, features, chat transcripts, files reference, audit trails of SYNC+ application in relational format.

**HTTP / HTTPs** - Hyper Text Transport Protocol is the communication protocol used to connect to servers on the World Wide Web. The primary function of HTTP is to establish a connection with a Web server and transmit HTML pages to the user's browser. For security reason all the communication where user data is manipulated will use HTTPs connection.

**ORM –** This creates, in effect, a "virtual object database" that can be used from within the programming language. The ORM interaction lets you encode access request statements in structured query language (SQL) that are then passed to the program that manages the database. It returns the results through a similar interface.

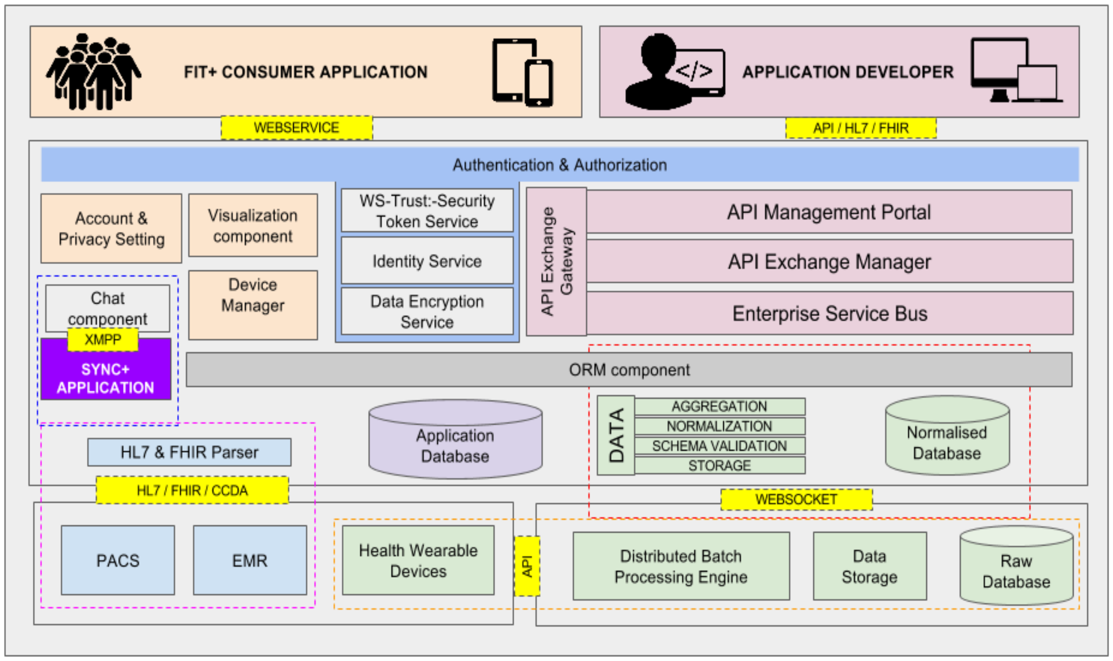
**XML** - A programming language/specification developed by the W3C, for organizing and tagging elements of a document so that the document can be transmitted and interpreted between applications and organizations.

**JSON** - JSON (JavaScript Object Notation) is a lightweight data-interchange format. A collection of name/value pairs. In various languages, this is realized as an object, record, struct, dictionary, hash table, keyed list, or associative array. These are universal data structures. Virtually all modern programming languages support them in one form or another.

## 4. Application Architecture

Application architecture defines the various components and their interactions in context of a whole system. Application architecture is the critical software that bridges the architectural gap between the application server and the application’s business logic, thereby eliminating the complexities and excessive costs of constructing, deploying and managing distributed enterprise applications.

The FIT+ architecture can be represented in the following layers illustrated by the diagram below:



SYNC+ application will have a layered application architecture which provides some of the key features below –

1. **STRUCTURE:** Organizing applications along business-level boundaries and not technical boundaries
2. **SPEED & FLEXIBILITY:** Making application changes through configuration and not programming
3. **CONTROL:** Modifying, extending or overwriting any architectural element.
4. **REUSE:** Achieving greater reusability and integration by loosely coupling application logic to infrastructure.

At a conceptual level, they represent distinct and cohesive aggregations of functionality. SYNC+ application design is based on a tiered approach. "A tier is a logical partition of the separation of concerns of the system. Each tier is assigned its unique responsibility in the system. We view each tier as logically separated from one another. Each tier is loosely coupled with the adjacent tier."

**Presentation Layer:** Client interfaces for SYNC+ application are SMART mobiles & tablets, Desktops and Laptops. Mobile supported Operating Systems are Android and iOS. The presentation component has been built on Objective C for iOS and JAVA for Android and Portal will be developed based on node JS, JavaScript and bootstrap framework. There are different actor’s involve Portal admin, Physician & Patient. Portal Admin will use Laptop/Desktop. Physician & Patient will use Mobile application.

**Authentication and Authorization Layer:** All the request from the user will be validated for Authentication and Authorization. The authentication technique will happen based on the auth_token, which will be generated while the user has authenticated successfully. Auth_token will have expiry time, once expired client need to refresh the token and get new auth_token for further access. Authorization will be done once successfully authenticated, each features will be tied to a Policy based on the user role. Each time the policy will be checked and responded with right access. All the request will be on top of Web Service, which uses the concept of a Security Token Service (STS) - a web service that issues security tokens as defined in the WS-Security specification.

**Data Encryption Service/Standard:** PHI data is in rest and motion are sensibly encrypted with AES encryption algorithm. All the transaction / communication which happens from the server are through HTTPs secure protocols. XMPP protocol communication will use SSL/Transport Layer Security (TLS) for sending and receiving messages.

**Identity Service:** The user base layer which will have information about the user and it provides tighten security layer such as OTP / MFA. Authentication and Authorization are well handled by Identity service layer.

**PHI Access & Approval Component:** All the data are validated against each request to check whether the data is PHI and it’s secured.

**Dashboard & Visualization Component** : The reports & widget’s are micro services in which all the data are fetched from DB and combined as a single service which gives more speed in delivering the data with the help of caching mechanism.

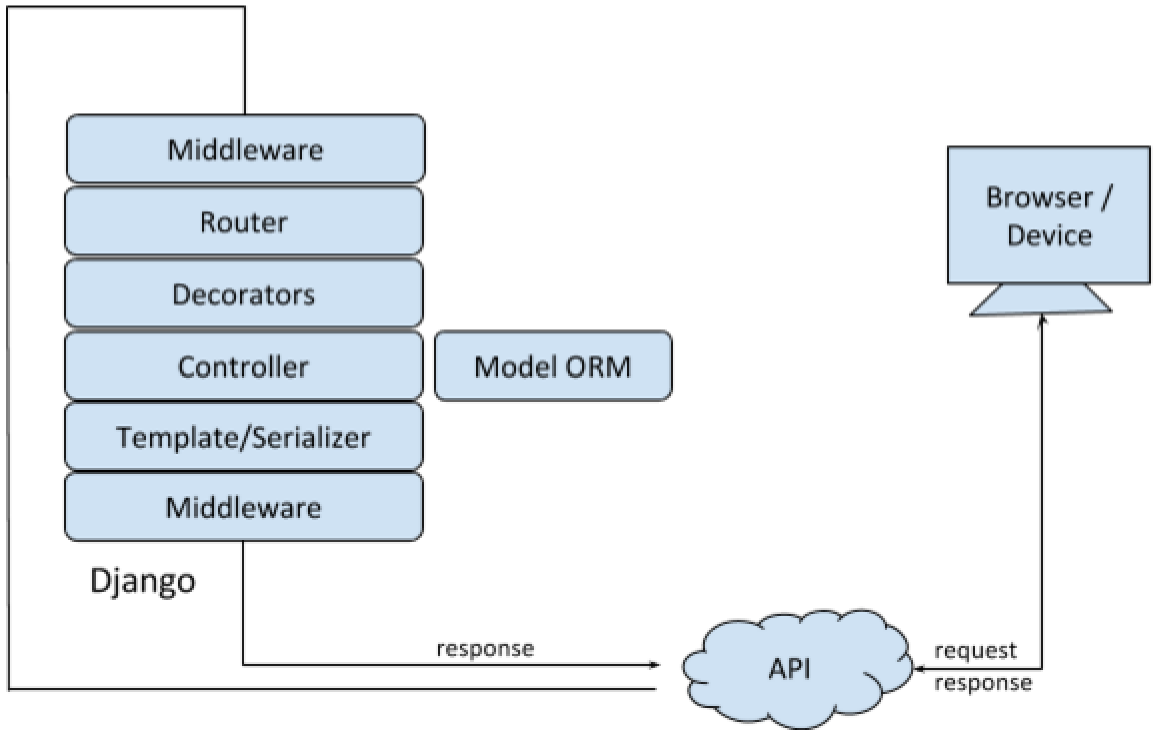
**EMPI Component:** EMPI component will interact with multiple EMR system and find the patient uniqueness across EMRs. There are a wide variety of matching algorithms that have been used in practice but they are classified into two categories:

- **deterministic algorithms** that search for an exact match between attributes
- **probabilistic algorithms** that search for an approximate match between two records

**Distributed Batch Processing Engine:** The component will take care of the external parties APIs according to the Rate Limit. The time-to-live of device data is handled asynchronously to collect and store in Raw-data format. This component is highly scalable by adding nodes (workers) to the system.

**Generic Parser Component:** The parser component will help parsing FHIR, API, and HL7 & C-CDA document. Parser will provide FIT+ data format which both FIT+ & SYNC+ application can understand.

Architecture of Middleware API flow,



## 4. Assumption and Constraints

While the guiding principles establish the general values that the target architecture should consider, a number of assumptions were made about both the infrastructure and general direction for technology.

Assumptions and Constraints:

1. User Acceptance Test / System Acceptance Test Environment will be available for performing Usability testing, User Acceptance testing, Installation & Configuration testing and Performance Testing

2. General Architecture principles based on past experiences and HealthViewX Best practices & methodologies will be used in designing the solution

3. The basic TCP/IP (HTTP,HTTPs, XMPP, SFTP) protocol will be the only one used to access the application The web browser will be the primary client used by employees and public users

## 5. Appendix A: Acronyms, Abbreviations, Terms and Definitions

| | |
|---|---|
| API | Application Program Interface |
| GOF | Gang of Four |
| EMR | Electronic Medical Record |

| | |
|---|---|
| FHIR | Fast Healthcare Interoperability Resources |
| HL7 | Health Level-7 |
| CDA | Clinical Document Architecture |
| C-CDA | Consolidated-CDA |
| EMPI | Enterprise Master Patient Index |
| ORM | Object-relational mapping |
| HTTP | Hypertext Transfer Protocol |
| XMPP | Extensible Messaging and Presence Protocol |
| TLS | Transport Layer Security |
| XML | Extensible Markup Language |
| JSON | JavaScript Object Notation |
| WS | Web Service |

# 6. Appendix B: Lists all products and tools used in design and development

| # | Components | Open Source | Ver | License | License Contract Link |
|---|---|---|---|---|---|
| 1 | Identity Server | WSO2 | 5.0.0 | Apache License, Version 2. | http://wso2.com/licenses |
| 2 | API Manager | WSO2 | 1.9.1 | Apache License, Version 2. | http://wso2.com/licenses |
| 3 | Enterprise Service Bus | WSO2 | 4.9.0 | Apache License, Version 2. | http://wso2.com/licenses |

|  |  |  |  |  |  |
|---|---|---|---|---|---|
| 6 | Database | PostgreSQL | 9.3.7 | PostgreSQL License | http://www.postgresql.org/about/licence/ |
| 7 | | MySQL | 5.6 | GPL | http://www.mysql.com/about/legal/licensing/oem/ |
| 10 | Backend Framework | Django | 1.8.3 | BSD license | https://github.com/django/django/blob/master/LICENSE |
| 11 | Chat Server | Ejabberd | 15.0.6 | GPL 2 or later + permission to link with OpenSSL | https://www.ejabberd.im/licenses |
| 12 | Frontend Framework | Yahoo Mojito | 0.2.0 | BSD license | https://github.com/yahoo/mojito/blob/master/LICENSE.txt |
| 13 | Frontend Library | jQuery | 1.11.3 | MIT license | https://jquery.org/license/ |
| 15 | UI Framework | Bootstrap | 3.3.5 | MIT license | https://github.com/twbs/bootstrap/blob/master/LICENSE |