Project Sigma
Implementation Model Document
version: 1.2

Project Manager:	Imtiaz Ahmed 0444588
Requirement Analyst:	Kuchimanchi Lakshmi Prasanna 0433913
Software Designer:	Joonas Maksimainen 0372184
Developer:	Vitezslav Kriz 0457494
Designer and Developer:	Eduard Telezhnikov 0460339
Software Tester:	Juho Juvani 044472

Revision History			
Date	Version	Description	Author
29/10/15	1.0	Implementation Model Overview Iteration 1 and 2	Vitezslav Kriz
1/11/15	1.1	Iteration 3	Vitezslav Kriz
11/12/15	1.2	Defects repair	Vitezslav Kriz

Implementation Model Document

Table of Contents

1 Introduction

1.1 Purpose

This document provides a detailed plan for integration within each iteration.

1.2 Scope

This document applies to all iterations of the time Sigma project. The content may vary and grow following the project evolution.

1.3 Definitions, Acronyms, Abbreviations

Build – In this project build is actual version of source code consist of several files.

1.4 References

Software Architecture Document, Team 2, SWQPO, LUT

Glossary, Team 2, SWQPO, LUT

1.5 Overview

This document shows structure of packages in Application Sigma their relation. For each iteration, components and packages are identified. Iterations contains information how to test each build.

2 Implementation Model Overview

Project Sigma use existing library Django for language Python. These approach creates only one executable which is responsible for running whole server application.

2.1 Implementation Model Structure

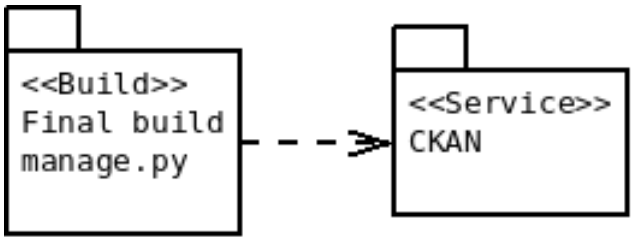

Figure 1: Implementation Model - Top Level Package

Running application depends on a functional service of CKAN system. CKAN can run independently of Django server, but Django system doesn't work appropriate without working CKAN connection via API. CKAN model is responsible for connecting to CKAN on Django side of system.

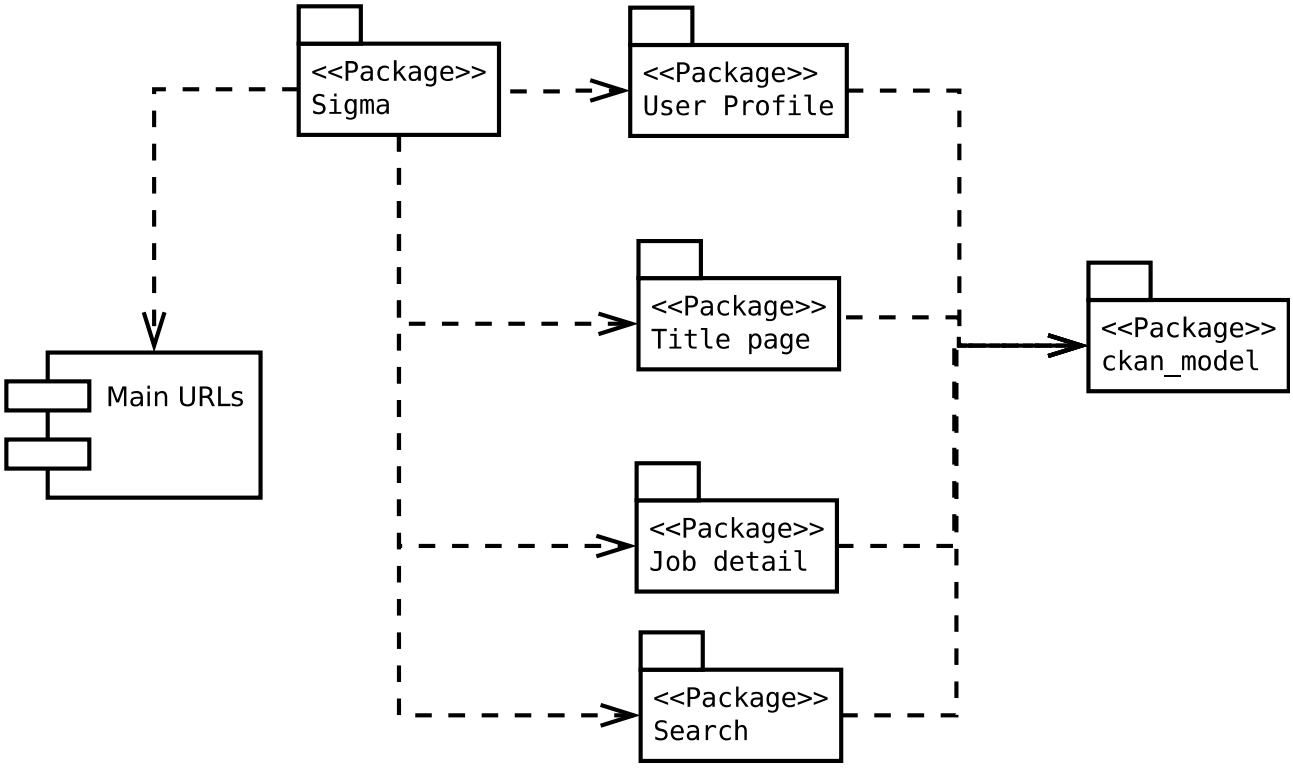

Figure 2: Implementation Model - Overview (Level 1)

2.1.1 Packages detailed description

Packages User Profile, Title Page, Job detail and Search are packages based on Django generated code based on MVC pattern. These package share same structure as shown on figure 3.

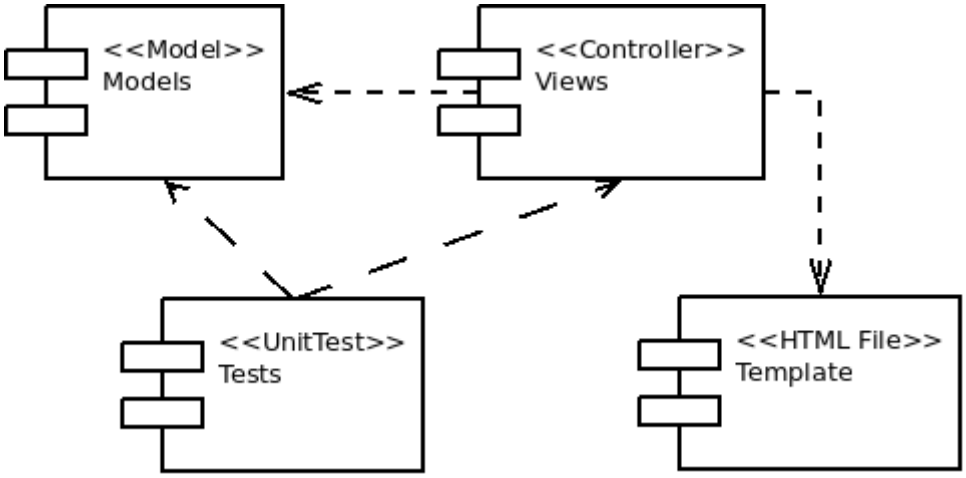

Figure 3: Implementation Model - Search Package

3 Iteration 1

First iteration should shows working tools selected to this project. Connection to CKAN database will works in this iteration.

3.1 Components & Subsystems

3.1.1 Components and subsystems to be implemented

During this iteration, functional prototype of Django server must be implemented. Following packages must be implemented:

1. Django side
    - Sigma settings
        - Urls
    - Search page
        - Student search
    - Title page
2. CKAN side
    - Search by students skills and University

- List of used tags

Django and CKAN side can be done independently on each other. CKAN dependency should be replaced by developing stub package which provide only static data with same structure.

3.2 Specific dependency diagram

In this iteration are two builds. First one shows working Django server prototype without connection to CKAN system. Second build use CKAN connection. On figure 4 you can see both builds second one use gray color.

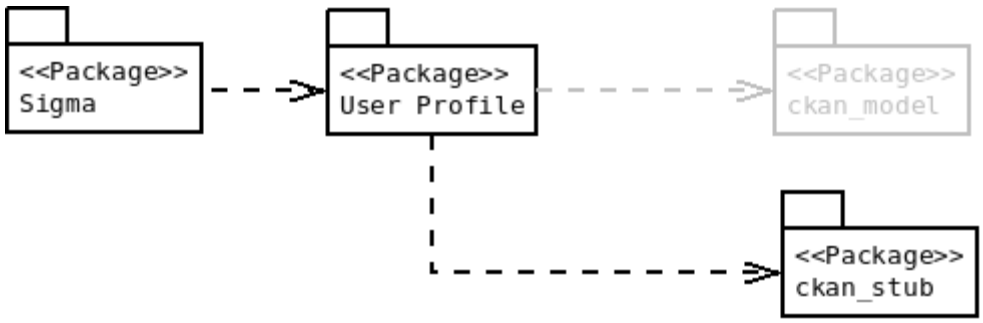

Figure 4: Django Prototype - Build

3.3 Evaluation criteria

Both builds work and show data in same meaning. Executing of these separate builds is done by changing execution parameter. CKAN version are to show data saved in CKAN database, which can be accessed by public website.

4 Iteration 2

In this iterations must be implemented most of function for getting and showing data. Adding and editing data aims only to Student's profile and work.

These packages must be implemented:

1. Authorization
    - Sign up
    - Sign in
2. Student's profile
    - Add school work

    - Add CV
3. Showing
    - Show student's detail
    - Search Job
    - Show Job Detail
    - Show Event

Student's profile can be implemented only after working authorization. Showing package is independent on other packages.

4.1.1 Specific dependency diagram

There is only one build, which depend on CKAN system. CKAN stub shouldn't be used anymore.

4.2 Evaluation criteria

Build will be testable against requirements specified in Requirements document. It is obvious that a lot of requirements will not be satisfied. After evaluation of build of this iterations, there should be list of done requirements, requirements with defects and list of these defects and missing requirements.

5 Iteration 3

This iteration aims to editing and adding data from company point of view. Packages which should be implemented:

- Job detail
- extended User Profile for company
- Moderator and Administrator

5.1 Build

System is deployed on cbuserver and run 24/7 as a service.

5.2 Evaluation criteria

Each use case can be testable. Recruiters can add Job offer, edit it. Students can find this added Job Offer. All functionality from previous iteration should be work as well. Administrator and Moderators now can do properly their roles.