# System and Software Architecture Description (SSAD)

**Image Processing Platform**

**Team 4**

|  |  |  |
| --- | --- | --- |
| **Name** | **First Role** | **Second Role** |
| Hao Wu | Requirements Engineer | Software Architect |
| Junran Liu | Operational Concept Engineer | Software Architect |
| Meiyi Yang | Project Manager | Life Cycle Plan |
| Vinny DeGenova | IIV & V | Quality Focal Point |
| Xiangchen Zhao | Life Cycle Plan | Prototyper |
| Xinhui Liu | Feasibility Analyst | Operational Concept Engineer |
| Yifan Liu | Prototyper | Requirements Engineer |

**12/01/2016**

# Version History

| Date | Author | Version | Changes made | Rationale |
| --- | --- | --- | --- | --- |
| 10/04/16 | HAO WU | 1.0 | * Original template for use with Instructional ICM-Sw v1.0 | * Initial draft for use with Instructional ICM-Sw v1.0 |
| 10/14/16 | HAO WU | 1.1 | * Modify the architecture | * For DC package |
| 12/01/16 | HAO WU | 2.0 | * Complete Technology-Independent Model * Complete Technology-Specific System Design * Complete Architectural Styles, Patterns and Frameworks | * For As Built package |

# Table of Contents

System and Software Architecture Description (SSAD) i

Version History i

Table of Contents ii

Table of Tables iii

Table of Figures iv

1. Introduction 1

1.1 Purpose of the SSAD 1

1.2 Status of the SSAD 1

2. System Analysis 2

2.1 System Analysis Overview 2

2.2 System Analysis Rationale 7

3. Technology-Independent Model 8

3.1 Design Overview 8

3.2 Design Rationale 12

4. Technology-Specific System Design 13

4.1 Design Overview 13

4.2 Design Rationale 17

5. Architectural Styles, Patterns and Frameworks 18

# Table of Tables

[Table 1: Actors Summary 2](#_Toc468656211)

[Table 2: Artifacts and Information Summary 3](#_Toc468656212)

[Table 3: Process Description 4](#_Toc468656213)

[Table 4: Typical Course of Action 5](#_Toc468656214)

[Table 5: Alternate Course of Action 5](#_Toc468656215)

[Table 6: Exceptional Course of Action 5](#_Toc468656216)

[Table 7: Process Description 5](#_Toc468656217)

[Table 8: Typical Course of Action 5](#_Toc468656218)

[Table 9: Alternate Course of Action 6](#_Toc468656219)

[Table 10: Exceptional Course of Action 6](#_Toc468656220)

[Table 11: Process Description 6](#_Toc468656221)

[Table 12: Typical Course of Action 6](#_Toc468656222)

[Table 13: Alternate Course of Action 6](#_Toc468656223)

[Table 14: Exceptional Course of Action 7](#_Toc468656224)

[Table 19: Hardware Component Description 9](#_Toc468656225)

[Table 20: Software Component Description 10](#_Toc468656226)

[Table 22: Design Class Description 10](#_Toc468656227)

[Table 23: Hardware Component Description 14](#_Toc468656228)

[Table 24: Software Component Description 14](#_Toc468656229)

[Table 26: Design Class Description 14](#_Toc468656230)

[Table 27: Architectural Styles, Patterns, and Frameworks 18](#_Toc468656231)

# Table of Figures

[Figure 1: System Context Diagram 2](#_Toc468656232)

[Figure 2: Artifacts and Information Diagram 3](#_Toc468656233)

[Figure 3a: Process Diagram 4](#_Toc468656234)

[Figure 4: Conceptual Domain Model 8](#_Toc468656235)

[Figure 5: Hardware Component Class Diagram 8](#_Toc468656236)

[Figure 6: Software Component Class Diagram 9](#_Toc468656237)

[Figure 7: Deployment Diagram 9](#_Toc468656238)

[Figure 9: Design Class Diagram 10](#_Toc468656239)

[Figure 10: Robustness Diagram 12](#_Toc468656240)

[Figure 11: Sequence Diagram 12](#_Toc468656241)

[Figure 12: Hardware Component Class Diagram 13](#_Toc468656242)

[Figure 13: Software Component Class Diagram 13](#_Toc468656243)

[Figure 14: Deployment Diagram 13](#_Toc468656244)

[Figure 16: Design Class Diagram 14](#_Toc468656245)

[Figure 17a: Process Realization Diagram 16](#_Toc468656246)

### Introduction

#### Purpose of the SSAD

This document defines comprehensive architecture for the Image Processing Platform(IPP) system. It explicates different aspects of the system by using charts and diagram. The aim of the document is to characterize the details of the whole system, include software components and their functions.

#### Status of the SSAD

This is the final version of the SSAD document and we already completed the system in details. This document helps others to understand the basic frame of our system and give an overview of the system. This document also explains why we design the system like this and how the idea came up.

### System Analysis

#### System Analysis Overview

The IPP can help users to mine specific images from various content elements without acquiring specific knowledge. It identifies same topic images which are similar with the images user uploaded. This gives the possibility to users to monitor image data transferring on the internet and detect the violent elements from the data. Meanwhile, as more images are fetched by system, the accuracy of detection will be improved.

##### System Context

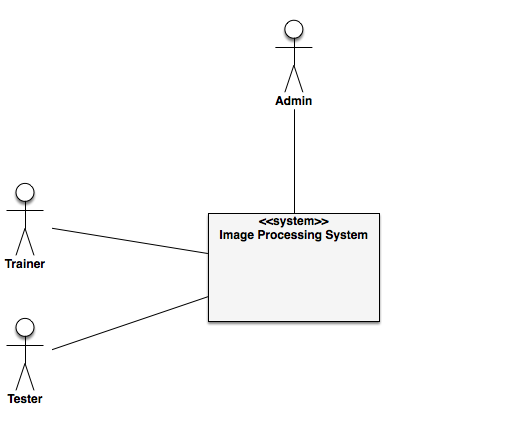


Figure 1: System Context Diagram

Table 1: Actors Summary

| **Actor** | **Description** | **Responsibilities** |
| --- | --- | --- |
| Trainer | User of the system | Upload images from local and start the model training process |
| Tester | User of the system | Upload image from local and execute the image recognition process. |
| Administrator | Administrator of the system | Control and monitor the system |

##### Artifacts & Information

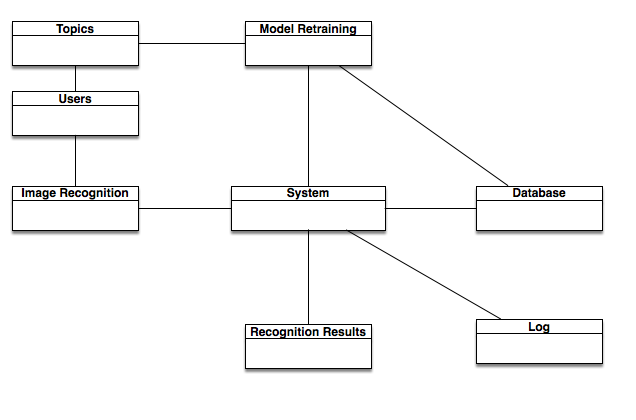


Figure 2: Artifacts and Information Diagram

Table 2: Artifacts and Information Summary

|  |  |
| --- | --- |
| **Artifact** | **Purpose** |
| Image Recognition | To recognize image’s category. |
| Model Training | The system update model with uploaded images. |
| Topics | Topic is necessary for Image Recognition |
| Database | To save model and Image to database |
| System | To process and analysis images from user |
| Users | To provide user a function to upload images |
| Recognition Results | To recognize images |
| Log | To inform administrator system condition. |

##### Behavior

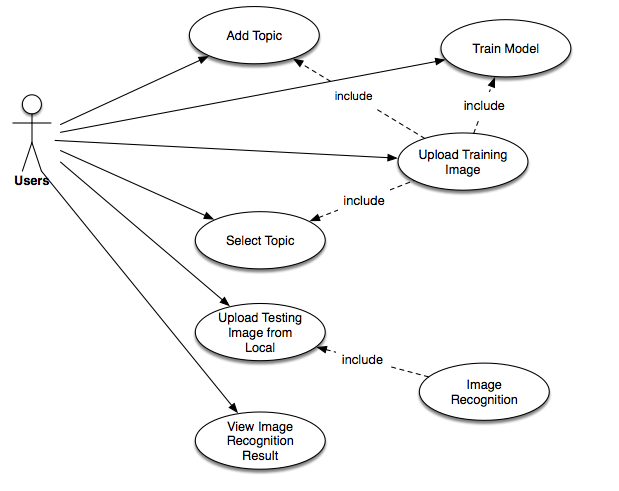


Figure 3a: Process Diagram

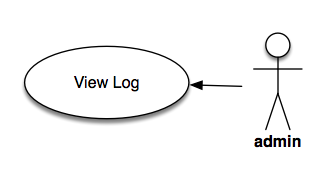


Figure 3b: Process Diagram

1. Table 3 is the process description of uploading training image. Table 4, Table 5, Table 6 talk about the uploading process in detail, the different ways to upload training images and exceptional course of action.

Table 3: Process Description

|  |  |
| --- | --- |
| **Identifier** | Upload Training Images |
| **Purpose** | Upload a set of image to server |
| **Requirements** | WC\_4147,WC\_4109,WC\_4040 |
| **Development Risks** | The system need to identify the format of each images and the system can handle multiple images uploading at same time. |
| **Pre-conditions** | Users need to upload proper format images and have good network conditions. |
| **Post-conditions** | User trained model |

Table 4: Typical Course of Action

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Trainer clicks upload button | System provides interface to user to upload images |
| **2** | Trainer clicks submit button | System starts to receive images |

Table 5: Alternate Course of Action

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** |  |  |
| **2** |  |  |

Table 6: Exceptional Course of Action

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Trainer clicks submit button | if there is no image uploaded or the image is improper, the system will inform user and redirect to the uploading page and show the error to user |

1. Table 7 gives the description of selecting a topic. Table 8, Table 9, Table 10 talk about the selecting process in detail, the different ways to select topic and exceptional course of action.

Table 7: Process Description

|  |  |
| --- | --- |
| **Identifier** | Add Topic |
| **Purpose** | denominate a classifier |
| **Requirements** | WC\_4147 |
| **Development Risks** | This process needs to insure that users input appropriate characters and inform users the error information when they failure to meet the input requirement |
| **Pre-conditions** | Users need to users input appropriate characters |
| **Post-conditions** | User train model |

Table 8: Typical Course of Action

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Trainer clicks add topic button | System provides interface to user to input the topic name. |
| **2** | Trainer clicks save button | System adds a topic and saves it in the databse |

Table 9: Alternate Course of Action

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** |  |  |

Table 10: Exceptional Course of Action

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Trainer clicks save button | if the string user typed in is not match the requirement, system will inform the error information to users. |

1. Table 11 gives the description of selecting a topic. Table 12, Table 13, Table 14 talk about the training process in detail, the different ways to start process and exceptional course of action.

Table 11: Process Description

|  |  |
| --- | --- |
| **Identifier** | Model Training |
| **Purpose** | Activate the training process to generate a specific classifier and update model. |
| **Requirements** | WC\_4107 |
| **Development Risks** | This process needs to apply machine learning algorithm to generate a corresponding classifier on model. |
| **Pre-conditions** | Users need to upload a set of images in advance and already assigned a topic to the images. |
| **Post-conditions** | Users train model |

Table 12: Typical Course of Action

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Trainer clicks training button | System checks the images conditions and preprocess the images for smaller size. |
| **2** | Trainer view progress button | System returns a feedback (eg. a progress bar) |

Table 13: Alternate Course of Action

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** |  |  |

Table 14: Exceptional Course of Action

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Trainer clicks training button | Users don’t upload the images or assign a topic. |

1. Table 11 gives the description of selecting a topic. Table 12, Table 13, Table 14 talk about the training process in detail, the different ways to start process and exceptional course of action.

##### Modes of Operation

The IPP has two main modes of operation:

1. Training Model mode: This mode is generated by machine learning algorithm and images uploaded by trainer。
2. Image Recognition mode: This mode is used by testers. These users can just use the ready-made model instead of uploading images to generate a new model.

#### System Analysis Rationale

Most of our users are non-technical persons so that we need design a very simple interface and noticeable entrance. Given that trainers may don’t have enough images to generate classifier, we provide extra function which is uploading image from Instagram to let trainer upload same topic images as many as possible. We also allow trainers to use other trainer’s topic to update the topic. That’s will help them to improve their classifier’s accuracy.

### Technology-Independent Model

#### Design Overview

##### System Structure

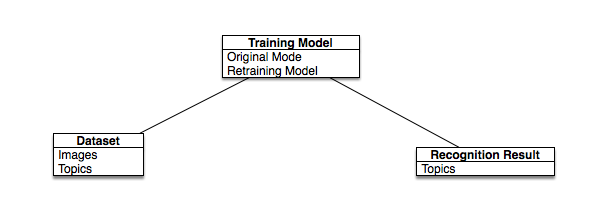


Figure 4: Conceptual Domain Model

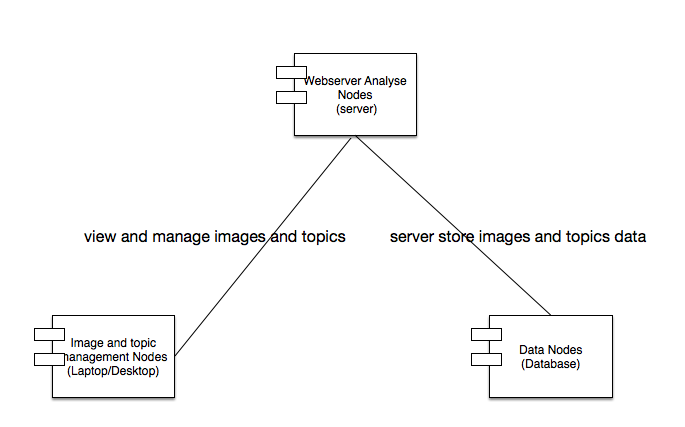


Figure 5: Hardware Component Class Diagram

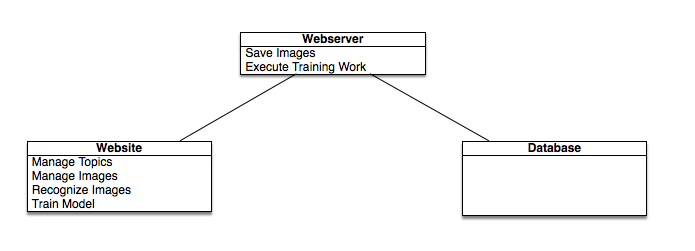


Figure 6: Software Component Class Diagram

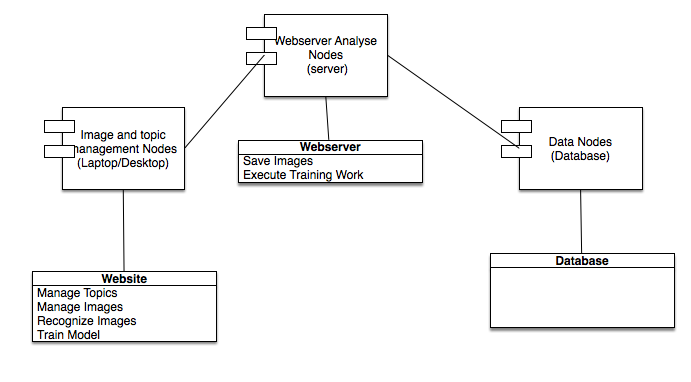


Figure 7: Deployment Diagram

Table 19: Hardware Component Description

|  |  |
| --- | --- |
| **Hardware Component** | **Description** |
| Image and Topic Management Nodes | Image and Topic Management Nodes are the computers/laptops that the users use to upload images and add or modify topics in the system. |
| Webserver Analyze Nodes | Webserver Analyze Nodes are the server which run the webserver. They process all the training tasks and testing tasks |
| Data Nodes | Data Nodes is database service run on the computer/server |

Table 20: Software Component Description

|  |  |
| --- | --- |
| **Software Component** | **Description** |
| Website | Website software component is the interface provided by us. It gives the view to user to let them view the content and upload their dataset and testing Image. |
| Webserver | Webserver software component is used to process the requests and receive the images uploaded by users. The core image classification algorithm is contained in the webserver. |
| Database | Database software component stores some useful data for reusing them. |

##### Design Classes

###### <Classes n>

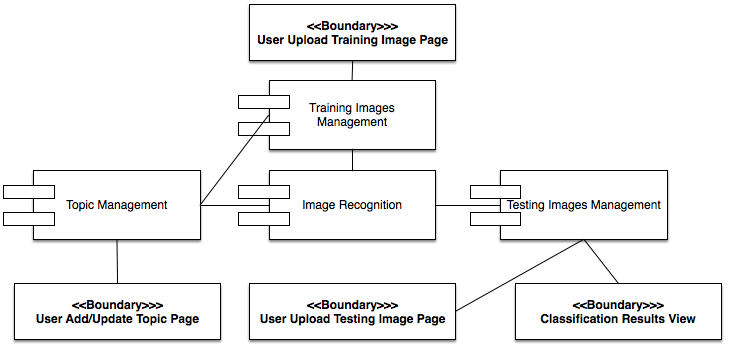


Figure 9: Design Class Diagram

Table 22: Design Class Description

|  |  |  |
| --- | --- | --- |
| **Class** | **Type** | **Description** |
| User Add/Update Topic Page | Boundary | The user can add a new topic or update the topic they select to scale up or update the new training model. |
| Topic Management | Component | This component directs user to the topic management page |
| Image Recognition | Component | This component execute image recognition algorithm and create a new model after training process is finished |
| Training Images Management | Component | This component directs user to the a image management page basic on which topic user choose. |
| User Uploading Training Image Page | Boundary | This page gives the uploading image function to user. |
| User Upload Testing Image Page | Boundary | This page gives the uploading image and choosing model function to user. |
| Training Images Management | Component | This component execute image recognition algorithm to detect images. |
| Classification Result Page | Boundary | This pages show the testing result to user. |

##### Process Realization

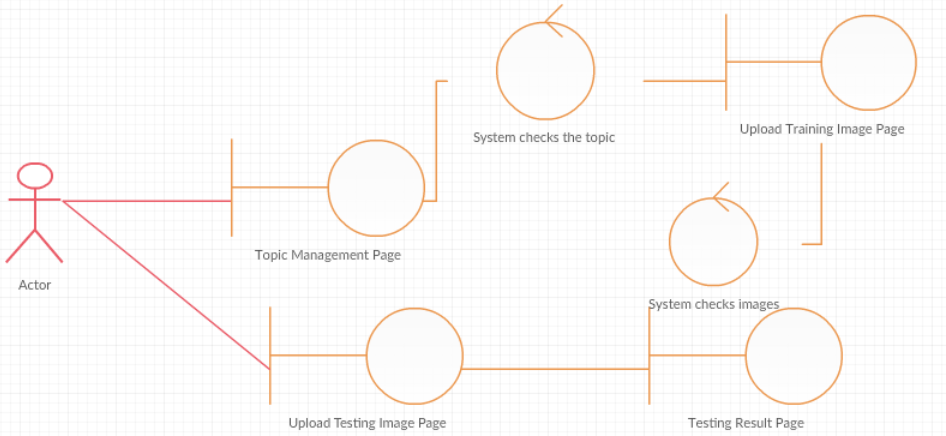


Figure 10: Robustness Diagram

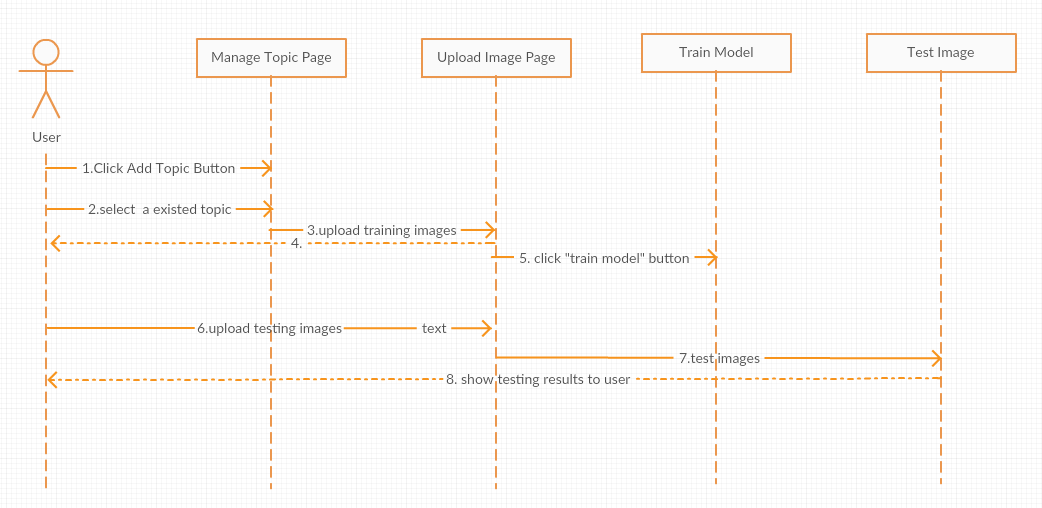


Figure 11: Sequence Diagram

#### Design Rationale

For the sake of applying our image recognition algorithm into webserver, we need provide some interface to user to let them add new classification and upload correspond images to enlarge the model. Considering most of our users have only little experience computer skill, our process flow should be simple to operate. So we facilitate the process into two main step, one is adding or selecting a topic and the other on is uploading images. The system should be easy to every user. In testing part, we want user can access our image recognition algorithm easily. So what users need to do is just uploading several testing images. After click the testing button, we will give back the final result to users.

### Technology-Specific System Design

#### Design Overview

##### System Structure



Figure 12: Hardware Component Class Diagram

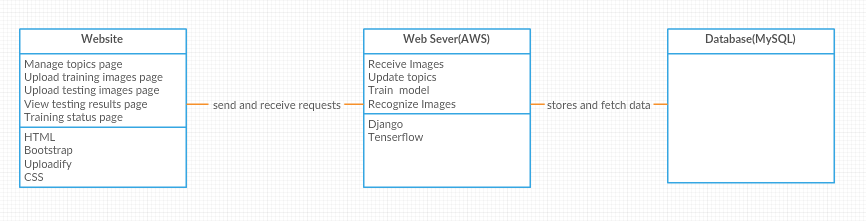


Figure 13: Software Component Class Diagram

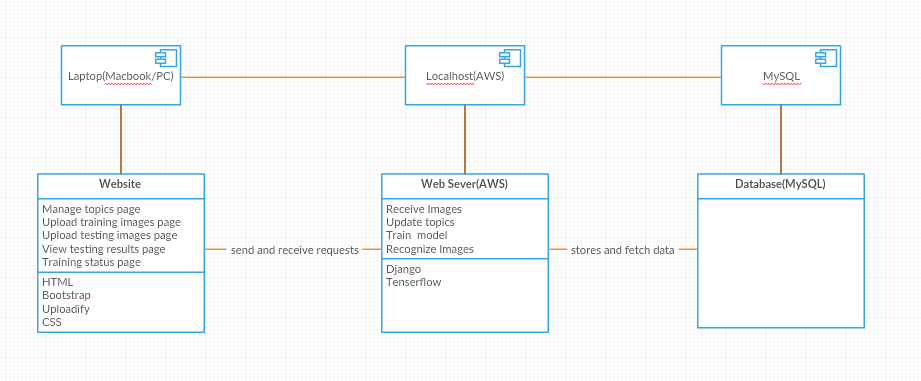


Figure 14: Deployment Diagram

Table 23: Hardware Component Description

|  |  |
| --- | --- |
| **Hardware Component** | **Description** |
| Laptop(MacBook/PC) | Users use any laptop can access our website and use our system. |
| LocalHost(AWS) | This component is a webserver which running on AWS. The webserver listen and process the requests from users |
| MySQL | We use MySQL database to store useful data. |

Table 24: Software Component Description

|  |  |
| --- | --- |
| **Software Component** | **Description** |
| Website | The Website component uses HTML, CSS, javascript, Uploadify and Bootstrap. |
| Web Server(AWS) | The Web Server component is deployed on the AWS and use Django web framework and python3 language. |
| Database(MySQL) | The Database component save the image files and their information for next retraining. |

##### Design Classes

###### <Classes n>

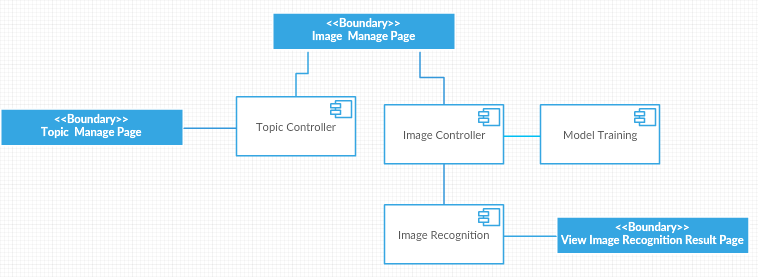


Figure 16: Design Class Diagram

Table 26: Design Class Description

|  |  |  |
| --- | --- | --- |
| **Class** | **Type** | **Description** |
| Topic Manage Page | Boundary | Users use this page to add a new category or update the category |
| Image Manage Page | Boundary | Users use this page to upload training or testing images to server and train a new model or recognize images |
| View Image Recognition Result Page | Boundary | Users can view the recognition result through this page |
| Topic Controller | Component | Topic Controller component collect topic information input by users and redirects to Image Manage page. |
| Image Controller | Component | This controller component allows users uploading multiple images through Image Manage Page. In addition, the component will send the image files to server. |
| Model Training | Component | This component will run the learning algorithm, process the images and analyze them to build a new model. |
| Image Recognition | Component | This component will classify the images and pass the results to the View Image Recognition Result Page. |

##### Process Realization

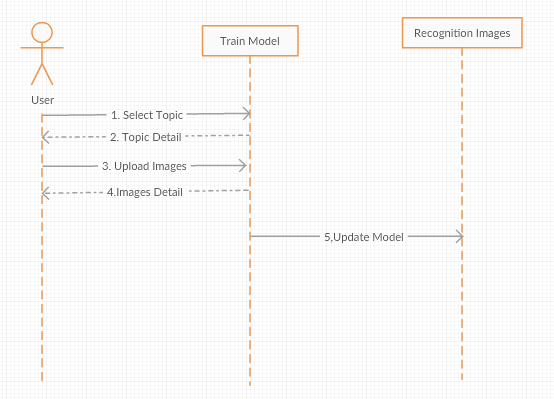


Figure 17a: Process Realization Diagram

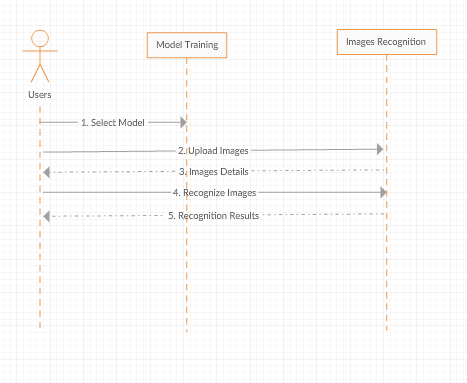


Figure 17b: Process Realization Diagram

#### Design Rationale

Our overall architecture is very simple, but very efficient. We use django as our entire web framework and connect the server to database. The reason why we choose django is because most of us have some experience on python language and we can develop our system quickly and efficiently. Our client provides suggests us to deploy our system onto AWS, so we choose to run our web server on the AWS. We use Tensorflow as our machine learning algorithm because it uses advanced algorithm to classify images efficiently. Tensorflow also has high accuracy rate on image recognition compared with other similar algorithm.

### Architectural Styles, Patterns and Frameworks

Table 27: Architectural Styles, Patterns, and Frameworks

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Benefits, Costs, and Limitations** |
| Facade pattern | The learning algorithm integrated on the web server is very complicated, so what we choose façade pattern to help our development. The second reason we choose façade pattern is that we have many third library, such as bootstrap, uploadify, tensorflow, the façade pattern can reduce the dependencies of these third library on our system to make our system easier to modify. | The benefit of using façade pattern is reducing the difficulty of modifying the system when we need to substitute component in the system. The cost and limitation is that we need spend lots of time to code many interface-related code. |
| MVC | MVC(Model-View-Controller) is the most common software design pattern. We choose MVC design pattern because we are familiar with it and the we framework we choose is also completely support MCV. | The benefits of using MVC is that we can easily divide the entire development work into every team member instead of considering how to integrate everyone’s work. |