GUNADARMA UNIVERSITY

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

Animetacchi Website Development with Django and Ajax

Name	:	Alvian Dwi Kurnianto
Student ID Number	:	10112667
Department	:	Information System
Supervisor	:	Dr. Setia Wirawan

This Submitted to The Faculty of Computer Science and Information Technology

Gunadarma University

In Partial Fulfillment of The Requirements

For Undergraduate Degree

JAKARTA

2016

Statement of Originality and Publication

Here by:

Name	:	**Alvian Dwi Kurnianto**
Student ID Number	:	**10112667**
Title of Undergraduate Thesis	:	**ANIMETACCHI WEBSITE DEVELOPMENT WITH DJANGO AND AJAX**
Session Date	:	**April 09, 2016**
Passing Date	:	**April 09, 2016**

States that the writings are the result of my own work and may be published entirely by Gunadarma University. All quotations in any form have been following the rules and ethics that applicable. All copyrights of logos and products mentioned in this book are the property of their respective rights holders, unless otherwise noted. Regarding the content and writing is the responsibility of authors, not Gunadarma University. Hence this statement is made actually with full awareness.

Jakarta, April 09, 2016

(Alvian Dwi Kurnianto)

Validation Page

Advisory Comitee

Session Date: April 09, 2016

No.	Name	Position
1.	Dr. Setia Wirawan	Supervisor
2.		Member
3.		Member
4.		Member

Board of Examines

Passing Date: April 09, 2016

No.	Name	Position
1.		Chairman
2.		Secretary
3.		Member
4.		Member
5.		Member
6.		Member

Acknowledge by,

Supervisor

(Dr. Setia Wirawan)

Department Head of Bachelor's
Defense Examination

(Dr. Edi Sukirman, SSi, MM)

Abstract

Alvian Dwi Kurnianto, 10112667
ANIMETACCHI WEBSITE DEVELOPMENT WITH DJANGO AND AJAX.

Keywords: Ajax, Django, Website

(xiii + 93 + appendix)

Website development today showed significant improvements. There are many types of websites, each specializing in a particular type of content and its use and classified into several types. Website Animetacchi including the three types of websites such as content within the site. The website also can be used to earn an income. This is evidenced by a study reveal that the reviews made by online users have a significant impact on buying behavior. To build a good website, it takes some criteria. Based on these criteria, the author also wanted to create a site attractive to users. The author will create a website using the Django web framework and Ajax on jQuery. The formulation of the problem is how to make a website of Animetacchi using Django framework and Ajax on jQuery. Users can view a list of anime and manga, give ratings and reviews on anime and manga, make a list of watch anime and reading manga, as well as interacting with other users. In making the web-based application, it takes a research method to solve problems. The scientific method used in this research is the development of software engineering methods SDLC. Website Animetacchi has been successfully implemented and successfully deployed in alviandjango.pythonanywheere.com. On this website, users can see the anime, manga, characters, and voice. Users can be identified as a registered user, unregistered, and admin. Registered users can create watch lists and reading as well as provide feedback to the site, as well as interact with other users.

Bibliography (1994:2016)

Preface

All praise and gratitude to the Almighty God, Allah SWT, for all of the blessing that have been given to the human in the entire world, and also shalawat and salam may everlastingly be upon to the Muslim's adoration and shining model, namely prophet Muhammad SAW. Thus, with all the grants, this "Animetacchi Website Development with Django and Ajax" can be finished. This research is intended to fulfill the require ments to finish study on the undergraduate degree of Information System Department in Gunadarma University. The completion of this thesis is also because of the support and encouragement from many person. It is not possible to name everyone here; thanks go to all of them. However, the special acknowledgment will be dedicated to:

1. Prof. Dr. E. S. Margianti, SE., MM, as Rector of Gunadarma University.
2. Prof. Dr. Rernat. Achmad Benny Mutiara, SSi, SKom., as Dean of Faculty of Computer Science and Information Technology.
3. Dr. Setia Wirawan, SKom., MMSi, as Head of Information System Department also as author's supervisor. For his patience, guidance, support, and encouragements during the research.
4. Drs. Haryanto, MMSi, as Director of SarMag (Sarjana Magister) Program.
5. Dr. Edi Sukirman, MM, as Head of Bachelor's Defense Examination.
6. Remi Senjaya, ST., MMSi, as Secretary of SarMag (Sarjana Magister) Program.
7. Author's beloved Father (Casiyan), Author's Mother (Sri Minarsih), Author's Brother (Aris) and his wife (Elly), who always provide motivation, advice, support, and encourage the author to complete this undergraduate thesis.

8. Author's mentor Mr. Wahyudi, who introduce python and django to author. Without him, author might not been mastering django.

9. Author's future wife Dyah Ayu Sukmawati, who always provide love, motivation, advice, support, and help to complete this undergraduate thesis. Without her, this thesis might not been done.

10. Two django girls of SMSI03 Anindita Ayu Pratiwi and Firda Maryana, who provide so much help to author on writing this thesis. Without them, this thesis might not been finished.

11. Author's Manager at coassets Dawin Widjaja who allow author to take leave.

12. To all SMSI-03, Citra Dewi Larasati, Kanigara Putra Loegiman, Muhammad Fariz Habibie, Nurul Rita Mustika, Otto, Ria Yuliastama.

13. To all team members of author's team. Find the Mentor team, Finhacks team, and Freelance team.

14. All parties can not mention who helped this thesis; thank you for all the help and advice.

This thesis still needs a lot of improvement. There are still a lot of mistakes in the writing of this thesis which makes it far from the perfection. However, the criticism and suggestion are indispensable to make the better research in the future. Hopefully, this research can bring some advantages to the readers.

Jakarta, April 09, 2016

(Alvian Dwi Kurnianto)

Contents

Statement of Originality and Publication ii

Validation Page iii

Abstract iv

Preface v

Contents vii

List of Figures xi

List of Tables xiii

1 INTRODUCTION 1

1.1 Background 1
1.2 Problem Statement 3
1.3 Scope of The Research 3
1.4 Research Objective 3
1.5 Research Methodology 3
1.6 Systematic Writing 4

2 LITERATURE REVIEW 6

2.1 System Development 6
2.1.1 Software Process 6
2.1.2 Rapid Application Development 8
2.2 Website 8
2.2.1 Types of Websites 9
2.3 Anime 10
2.4 Manga 11
2.5 Navigation Structure 12

2.5.1 Linear Navigation Structure 12
2.5.2 Nonlinear Navigation Structure 12
2.5.3 Hirarchical Navigation Structure 13
2.5.4 Composite Navigation Structure 13
2.6 Unified Modeling Language 14
2.6.1 Use-Case Diagram 14
2.6.1.1 Actor 15
2.6.1.2 Activity Diagram 16
2.7 Python 16
2.8 Django 16
2.9 HTML 18
2.10 Javascript 19
2.11 AJAX 19
2.12 MySQL 20
2.13 Testing 20
2.13.1 Black Box Testing 21

3 ANALYSIS AND DESIGN 22
3.1 Research Methodology 22
3.2 System Requirement Analysis 23
3.2.1 Animetacchi Website Overview 23
3.2.2 Functional Requirement 23
3.2.3 Non-functional Requirement 24
3.2.3.1 Hardware Specifications 24
3.2.3.2 Software Specifications 24
3.3 Design 24
3.3.1 Navigation Structure 24
3.3.1.1 Admin Navigation Structure 25
3.3.1.2 User Navigation Structure 25
3.3.2 UML Design 26
3.3.2.1 Use Case Diagram 26
3.3.2.2 Activity Diagram 29
3.3.3 Database Design 38
3.3.3.1 “Member” Table Design 38
3.3.3.2 “Genre” Table Design 39
3.3.3.3 “Anime” Table Design 39
3.3.3.4 “Manga” Table Design 40
3.3.3.5 “Watchlist” Table Design 40

3.3.3.6 “Readinglist” Table Design 40
3.3.3.7 “Character” Table Design 41
3.3.3.8 ”CharacterManga” Table Design 41
3.3.3.9 “VoiceCharacter” Table Design 41
3.3.3.10 “Daisukilist” Table Design 41
3.3.3.11 “Daikirailist” Table Design 42
3.3.3.12 “Chart” Table Design 42
3.3.3.13 “CommentAnime” Table Design 43
3.3.3.14 “CommentManga” Table Design 43
3.3.3.15 “UserProfile” Table Design 43
3.3.4 Interface Design 44
3.3.4.1 Home page for member and nonmember . . 44
3.3.4.2 Sign in page 44
3.3.4.3 Sign Up page 47
3.3.4.4 Forgot Password page 47
3.3.4.5 Confirmation page 51
3.3.4.6 Anime List Page 51
3.3.4.7 Manga List Page 51
3.3.4.8 Anime Details page 54
3.3.4.9 Manga Details Page 54
3.3.4.10 Character Detail page 55
3.3.4.11 Voice Actor Page 55
3.3.4.12 User Profile Page 57

4 IMPLEMENTATION AND TESTING 59
4.1 Implementation 59
4.1.1 Website Implementation 59
4.1.1.1 Home Page Implementation 59
4.1.1.2 Sign In Page 60
4.1.1.3 Sign Up Page 61
4.1.1.4 Forgot Password Page 63
4.1.1.5 Confirmation Page 65
4.1.1.6 Anime List Page 69
4.1.1.7 Manga List Page 71
4.1.1.8 Detail of Anime Page 71
4.1.1.9 Detail of Manga Page 73
4.1.1.10 Character Detail Page 73
4.1.1.11 Voice Actor Page 75

4.1.1.12 User Page 75
4.1.2 Database Implementation 78
4.2 Testing 83

5 CONCLUSION **90**
5.1 Conclusion 90
5.2 Future Work 90

Bibliography **91**

Appendix **94**

Appendix B Listing Program **96**

Appendix Output **205**

List of Figures

2.1 Rapid Application Development Phases [17] 8
2.2 Linear Navigation Structure [15] 12
2.3 Nonlinear Navigation Structure [15] 13
2.4 Hierarchical Navigation Structure [15] 13
2.5 Composite Navigation Structure [15] 13
2.6 Use-case diagram syntax [9] 15
2.7 Use-case Diagram 15
2.8 Django Layers [25] 18

3.1 Stage of Research 22
3.2 Admin Naviagtion Structure 25
3.3 User Navigation Structure 27
3.4 Use Case Diagram 27
3.5 Administrator Activity Diagram For Log In 29
3.6 Activity Diagram for Administrator Activities 30
3.7 Administrator Activity Diagram for Log Out 30
3.8 Activity Diagram to New Member Sign Up 31
3.9 Activity Diagram to Login Member 32
3.10 Activity Diagram for Members on Dashboard 32
3.11 Activity Diagram for Members on Anime 33
3.12 Activity Diagram for Members Post Comments on Anime . . . 34
3.13 Activity Diagram for Non Member Users on Anime 35
3.14 Activity Diagram for Members on Manga 36
3.15 Activity Diagram for Non Member Users on Manga 37
3.16 Activity Diagram on Chart Page 38
3.17 Home For Members Page Views 45
3.18 Home For Non Member Page Views 46
3.19 Sign In Page Views 47
3.20 Sign Up Page Views 48

3.21 Forgot Password Page views 49
3.22 Forgot Password Reset Done Page Views 49
3.23 Form Forgot Password Page Views 50
3.24 Password Reset Complete Page views 50
3.25 Confirmation Page Views 51
3.26 Confirmation Expired Page Views 52
3.27 Confirmation Check Email Page Views 52
3.28 Anime List Page Views 53
3.29 Manga List Page Views 53
3.30 Anime Detail Page Views 54
3.31 Details Manga Page Views 55
3.32 Character Detail Page Views 56
3.33 Voice Actor Page Views 56
3.34 User Profile Feed Page Views 57
3.35 User Profile Feed Page Views 58

4.1 Home Page 59
4.2 Home Page (News Section) 60
4.3 Sign In Page 62
4.4 Sign Up Page 62
4.5 Forgot Password Page 65
4.6 Password Reset Done Page 65
4.7 Password Reset Page 66
4.8 Password Reset Complete Page 66
4.9 Password Reset Unseccessful 66
4.10 Confirmation Page 67
4.11 Confirmation Expired Page 67
4.12 Check Email Page 69
4.13 Anime List Page 69
4.14 Manga List Page 72
4.15 Detail of Anime Page 72
4.16 Detail of Manga Page 74
4.17 Character Detail Page 74
4.18 Voice Actor Page 75
4.19 User Page 75
4.20 User Profile Feed Page 76
4.21 User Profile for Anime Library 76
4.22 User Profile for Manga Library 77

List of Tables

3.1 “Member” Table Design 39
3.2 “Genre” Table Design 39
3.3 “Anime” Table Design 39
3.4 “Manga” Table Design 40
3.5 “Watchlist” Table Design 40
3.6 “Readinglist” Table Design 41
3.7 “Character” Table Design 41
3.8 ”CharacterManga” Table Design 41
3.9 “VoiceCharacter” Table Design 42
3.10 “Daisukilist” Table Design 42
3.11 “Daikirailist” Table Design 42
3.12 “Chart” Table Design 43
3.13 “CommentAnime” Table Design 43
3.14 “CommentManga” Table Design 43
3.15 “UserProfile” Table Design 44

4.1 Black Box Testing Results 84

Chapter 1

INTRODUCTION

1.1 Background

Website development today showed significant improvements. This is evidenced by the presence of some statistical data, as quoted from netcraft.com, until March 2016 shows the response of 1,003,887,790 sites. This is the second time the number of sites has reached more than one billion since September 2014 but was down in November 2014 and April 2015. In the meantime, ording to zakon.org, there are increasing number of domain name registration, internet hosts, and the growth of the WWW that can be described in the appendix.

Based on the Cisco Visual Networking Index: Forecast and Methodology (2014-2019), annual global IP traffic will surpass the zettabyte (1000 exabytes) threshold in 2016, and the two zettabyte threshold in 2019. Global IP traffic will reach 1.1 zettabytes per year or 88.4 exabytes (one billion gigabytes) per month in 2016. By 2019, global IP traffic will pass a new milestone figure of 2.0 zettabytes per year, or 168.0 exabytes per month. Global IP traffic has increased more than fivefold in the past 5 years, and will increase nearly threefold over the next 5 years. Overall, IP traffic will grow at a compound annual growth rate (CAGR) of 23 percent from 2014 to 2019. Therefore, the author create a web-based application site.

Website has various types that can be divided into two categories, namely static and interactive. Interactive site is part of the Web 2.0 community and allows for interactivity between the site owner and site visitors. Static sites serve or capture information but does not allow engagement with the user directly. There are many types of websites, each specializing in a particular type of content and its use and classified into several types. Classification of

sites belonging to the category of the website that the author, among others, Site Review, Rating Site, Social Network Sites, and Social Bookmarking Sites. Review Site allows people to give a review about a product or service. Rating Site is a site designed to assess or select people, content, or other things. Social Network Site cited from the Journal of Computer-Mediated Communication, is a service-based website that allows individuals to (1) build a profile of public or semi-public limited system, (2) share the lists to users with various connections, and (3) view and traverse a list of the user's own or other property contained in the system. While Social Bookmarking Site is a site where users share content from the Internet, assess and comment on a content. Website Animetacchi including the three types of websites such as content within the site, such as anime, manga, character and voice actor can be given a vote, entered into the list of what people liked or disliked, commented, and shared to social media such as Facebook. Animetacchi addition users can also view user profile itself and other users in the system.

The website also can be used to earn an income. This is evidenced by a study conducted by comScore and The Kelsey Group reveal that the reviews made by online users have a significant impact on buying behavior. The results of the study quoted from this comscore.com examining the impact offline sales based on their reviews online for various services, such as restaurants, hotels, travel, legal, medical, automotive and home. Nearly one in four Internet users (24%) reported using online reviews prior to paying for a service offline.

To build a good website, it takes some criteria as quoted from smithandjonesadvertising.com: proper design, consistency, efficiency and organization. One of the tools that can be used to create a website with these criteria is Django. According to Moustakis et al. in a research journal that calculates the interest rate criterion at a site by using Analytical Hierarchy Process resulted in the conclusion that there are nine criteria in order of importance, include relevance, usefulness, reliability, specialization, architecture, navigability, efficiency, layout, animation. Meanwhile, to create a website with these criteria, it can be created by using jQuery. Based on these criteria, the author also wanted to create a site attractive to users. The author will create a website using the Django web framework and Ajax on jQuery.

1.2 Problem Statement

In this research, the formulation of the problem is how to make a website of Animetacchi using Django framework and Ajax on jQuery. In the website, users can view a list of anime and manga, give ratings and reviews on anime and manga, make a list of watch anime and reading manga, as well as interacting with other users.

1.3 Scope of The Research

This research is limited by several limits. On this website, users can see the anime, manga, characters, and voice. Users can be identified as a registered user, unregistered, and admin. Registered users can create watch lists and provide feedback to the site, as well as interact with other users.

1.4 Research Objective

The purpose of this research is to make a web-based application of Animeacchi with Django web framework and Ajax on jQuery where users can see a list of anime and manga, give ratings and reviews on anime and manga, create a list of anime and manga, as well as interact with other users.

1.5 Research Methodology

In making the web-based application, it takes a research method to solve problems. The scientific method used in this research is the development of software engineering methods SDLC. Here are the stages of research methods will be used:

1. Literature Review

   This part of methodology collected all information and references of research such as article, journal, books and other sources of internet related to this undergraduate thesis discussion.

2. Analysis and Design

   This procedure discusses about problem analysis, stages of research, stages of analysis and stages of design.

3. Design

   Design software aims to give an idea on what are the features that will be done and how it looks, covering the design of the use case diagram, activity diagram design, design mockups, and design ORM diagram to the database.

4. Testing and Implementation

   This procedure of SDLC discusses about code implementation based on application design, testing and get results of applications that have been implemented.

1.6 Systematic Writing

The purpose of this systematic writing is to simplify the writing. In this research, the author divides into five chapters. Each chapter contains a different subject. Subject is described as follows:

1. CHAPTER I: Introduction

   This chapter describes the background issues that underlie the writing of research, formulation of the problem, problem definition, the purpose of writing of the research to be achieved, methods of research and writing systematic research.

2. CHAPTER 2: Literature Review

   This chapter describes the things that support and connect with creating web applications animetacchi include websites, system development, Anime, Manga, navigation structure, UML, Python, Django, HTML, Javascript, AJAX, MySQL, and testing methodology.

3. CHAPTER 3: Analysis and Design

   This chapter discusses the analysis phase of functional and non-functional requirements, and design a website.

4. CHAPTER IV: Implementation and Testing

   This chapter describes the stages of the creation of applications ranging from implementation and testing phase of the application.

5. CHAPTER V: Conclusion

This chapter is the last chapter containing conclusions from the whole discussion, the next step for the research, along with suggestions to develop and expectations addressed to all parties interested in the writing of this research.

Chapter 2

LITERATURE REVIEW

2.1 System Development

Software Development Life Cycle (SDLC) describes the methodology by which the development of any software takes place. Before SDLC the process of developing the software was taken as informal activity with no formal rules and standards. This may lead to the various problems such as delay in development, cost overrun, and low software quality. The introduction of the SDLC gives the precise standard and the steps for the development of the software. SDLC overcome all the problems which are there before the introduction of the SDLC [23].

2.1.1 Software Process

A process is a collection of activities, actions, and tasks that are performed when some work product is to be created. An activity strives to achieve a broad objective (e.g., communication with stakeholders) and is applied regardless of the application domain, size of the project, complexity of the effort, or degree of rigor with which software engineering is to be applied. An action (e.g., architectural design) encompasses a set of tasks that produce a major work product (e.g., an architectural design model). A task focuses on a small, but well-defined objective (e.g., conducting a unit test) that produces a tangible outcome [18].

A process framework establishes the foundation for a complete software engineering process by identifying a small number of framework activities that are applicable to all software projects, regardless of their size or complexity. In addition, the process framework encompasses a set of umbrella

activities that are applicable across the entire software process. A generic process framework for software engineering encompasses five activities [18]:

1. Communication

   Before any technical work can commence, it is critically important to communicate and collaborate with the customer (and other stakeholders. The intent is to understand stakeholders' objectives for the project and to gather requirements that help define software features and functions.

2. Planning

   Any complicated journey can be simplified if a map exists. A software project is a complicated journey, and the planning activity creates a "map" that helps guide the team as it makes the journey. The map—called a software project plan—defines the software engineering work by describing the technical tasks to be conducted, the risks that are likely, the resources that will be required, the work products to be produced, and a work schedule.

3. Modeling

   Whether you're a landscaper, a bridge builder, an aeronautical engineer, a carpenter, or an architect, you work with models every day. You create a "sketch" of the thing so that you'll understand the big picture—what it will look like architecturally, how the constituent parts fit together, and many other characteristics. If required, you refine the sketch into greater and greater detail in an effort to better understand the problem and how you're going to solve it. A software engineer does the same thing by creating models to better understand software requirements and the design that will achieve those requirements.

4. Construction

   This activity combines code generation (either manual or automated) and the testing that is required to uncover errors in the code.

5. Deployment

   The software (as a complete entity or as a partially completed increment) is delivered to the customer who evaluates the delivered product and provides feedback based on the evaluation.

2.1.2 Rapid Application Development

RAD is a process which accelerates the cycle of development of an application. The phases are similar to waterfall but the 'chunks' are smaller. The emphasis in this model is on fast iterations through the cycle. Prototypes are designed, developed and evaluated with users, involving them in the process and correcting the design. The model is particularly suited to projects in rapidly changing environments where the team needs to adapt to different situations [11]. It is possible to develop quality products faster, thus valuabe resources can be saved. This methodology consists of the following three phases [17]:

1. Requirement Planning
2. RAD Design Workshop
3. Implementation

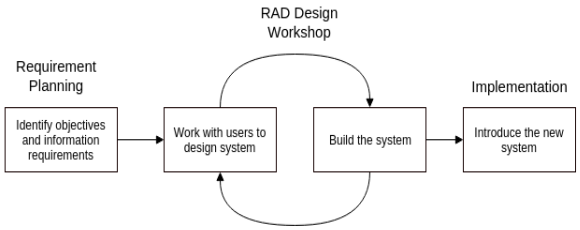


Figure 2.1 – Rapid Application Development Phases [17]

2.2 Website

The World-Wide Web was developed to be a pool of human knowledge, which would allow collaborators in remote sites to share their ideas and all aspects of a common project [2]. The Web is a part of the Internet, a group of interconnected computer networks that spans the globe. Web servers deliver content of many kinds, much of it connected to other content via hyperlinks and therefore referred to as hypertext. Most of these documents are written in a simple markup language called HTML, about which we will have much more to say. But web servers aren't limited to publishing HTML documents; they can deliver almost any digital content you care to envision [26].

The website was originally a dish service information using the concept of hyperlink, which allows internet surfer or user perform search information on the internet. The information presented by the web using the concept of multimedia, information can be presented using many media, such as text, images, animations, sounds, or movies.

The website has several supporting elements, namely the domain name (Domain Name / URL Uniform Resource Locator) and website hosting. Domain is the name or address that is uniquely assigned to identify the name of the server on a computer network or the Internet. The domain name is also known as a union of a web site. The domain name serves to help Internet users when accessing a page with access to the server without having to remember complicated IP addresses known to the row of numbers. Hosting can be thought of as a storage device capable of storing a variety of data, images, text and other files. Hosting itself has a diverse price depending on capacity and features offered.

2.2.1 Types of Websites

It is possible to categorize most Websites into the following types [14]:

1. Simple Intro Page

   In this case, the Website is just a “glorified” business card. It is a “static”, one page site, which provides a little introduction about the business and the contact information.

2. Electronic Brochure

   This type of Website provides more information about the company and the products or services. However, it resembles an electronic brochure in that the content never really changes. Statistics show that majority of Websites fall under this category.

3. Keeps Fresh Content

   This type of Website is a step above the “Electronic Brochure” Website in that the site owners take some effort to change the content on a periodic basis. Majority of such sites have a “News” or “Events” section that provides the latest happenings at the organization. The more elaborate sites will have frequent “facelifts” for the site to keep a fresh “look and feel”. However, these sites are still considered “static” in that they do not allow the Web visitors to interact with the organization in any way.

The most they allow would be a ”contact form” or “request for more information” etc.

4. Allows Interaction

   This type of Website is considered “dynamic” in the sense that it allows its Web members to perform “Online Services”. The Web visitors are encouraged to become Web members in order to gain access to the “members only” areas of the website. The login section and other services are database driven and provide the Web users with predefined services that are common to all Members.

5. ntegrated with Business Processes

   This type of Website provides the most functionality because the “front-end” services are integrated with the “back-end” databases. In this case, when a Web member logs-in, the Website provides them with a menu of services that are specific for them. The site “recognizes” the user who has logged in and then “customizes” the Online Services that the user can perform.

2.3 Anime

Anime is the Japanese word for animation regardless of country of origin. Anime is usually used in an English context to refer especially to the Japanese animation style [3]. Anime has existed before the World War II, with the first anime television series having been in the 1960s [24]. Western animation studios, like Disney, may base some their movies from children's books and fairy tales (Little Mermaid, Beauty and the Beast) however they are changed so they become child-friendly and always have happy endings. Fairy tales have been around for hundred, if not thousands of years, and began as an oral form of storytelling by adults. The fairy tales where never a genre intended for children. In the fifteenth century fairy tales began publishing them in print directed mainly to the adult readers [27]. Most of the fairy tales are grim and have sad endings. For example; In ‘The Little Mermaid’, originally written by Danish author Hans Christian Andersen, the mermaid's tongue was cut out by the witch in exchange for becoming human, as opposed to having the voice magically taken way in the Disney version. The endings are completely different, in the original; the prince married another

princess leaving the mermaid heartbroken. She is told that she is able to escape death and return to the sea if she kills the prince, not being able to kill the man she loves the mermaid returns to the shore where she dies, turning into sea foam. In the Disney version there is an epic battle where the witch is killed and the mermaid marries the prince and lives happily ever after.

Japanese animation studios are wellknown for being dramatic and mythical. They are also frequently drawn to stories with grave undertones or even to the realms of tragedy and epics. When doing a movie based on children's fiction, they often bring out its more mature themes or else reimagine it and giving their worlds complex subtexts [6]. In 1975 Toei Animation made their own version of the little mermaid, Anderusen Dōwa Ningyo hime or 'Hans Christian Andersen's The Little Mermaid'. Although whimsical, it is closer to Andersen's original story, especially the ending, than the later Disney version.

2.4 Manga

Manga in the Japanese word for comics, but like with anime, manga in used in English to refer especially to the Japanese comic style [3]. Comics in all shapes from comic strips to graphic novels, is sequential art. Sequential art is a narrative created from images, with or without texts. It is difficult to identify exactly when manga first emerged, for there are different theories. One theory claims that it began with sequential art in Japan, with the creation of scrolls of illustrations by Buddhist monks in the twelfth century. The most famous example is 'Chōjū-giga' or Animal Scrolls, which feature a lengthy sequence of expressive and humorous scenes of various animals such as moneys, foxes, rabbits and toads acting out the activities and pastime of clergy members and the nobility [4]. Another theory claims that first manga is referred to doodles in a sketchbook by Japanese artist Hokusai in the 1800s and were referred as "whimsical sketches" or "light-hearted pictures". As soon as modern printing was enabled in Japan, various comics were published, like the European-style satirical cartoons and the American newspaper strip comics. It was after World War II that Japanese manga industry started to rise. Televisions were not common in Japan until the 1950s, movies were expensive so comics were a more presentable source of entertainment because it was cheap. As televisions gained popularity, the bigger publishing companies introduced magazines, turned popular manga titles into anime and sold toys [24].

Bearing this in mind, the word “manga” has come to have two meanings outside Japan. Some use it to designate Japanese “comics,” the socio-cultural objects, and often the industry and community surrounding them. However, others use “manga” to name this visual language itself — loosely conceived of as an “aesthetic style”. Since the conflation of these ideas can be confusing and inappropriate, in this piece “manga” will be used in the first sense — to designate a socio-cultural artifact — while referring to the system of graphic expression as “Japanese Visual Language” or JVL. While JVL is the graphic system of communication, “manga” is the socio-cultural context in which it appears most [7].

2.5 Navigation Structure

Navigation structure is a plot that is used in applications that will be made. Before preparing multimedia applications in software, determining the flow to be used in applications that are made should be done first. The basic form of navigation structures commonly used in the manufacturing process there are four kinds of multimedia applications, namely linear navigation structure, non-linear, hierarchy, and composite [15].

2.5.1 Linear Navigation Structure

Linear navigation structure is a structure that has a series of stories sequentially. This structure displays layer one by one sequentially, according to the rules [15].

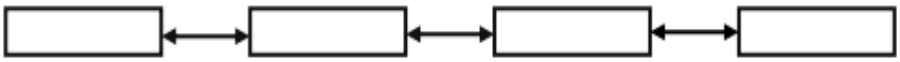


Figure 2.2 – Linear Navigation Structure [15]

2.5.2 Nonlinear Navigation Structure

Non-linear navigation structure (not sequential) is a development of linear navigation structure, only in this structure is allowed to make branching. Branching on nonlinear structure is different from branching in a hierarchical structure. In non-linear structure, position of all the pages are the same, so it is not known a master or slave page [15].

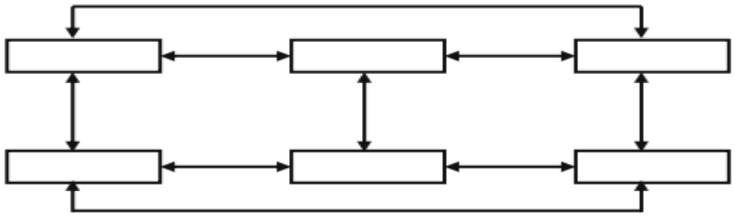


Figure 2.3 – Nonlinear Navigation Structure [15]

2.5.3 Hirarchical Navigation Structure

Hierarchy navigation structure is often called branched navigation structure, which is a branching structure that relies on the data or images for display on a layer with a certain criteria. Display on the main menu called the master page (the main page), it has a page called branching slave page (page support) and if selected will be the second page, and so on [15].

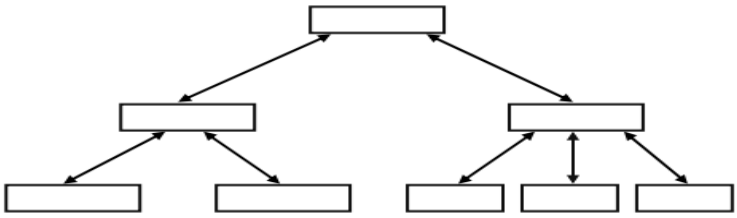


Figure 2.4 – Hierarchical Navigation Structure [15]

2.5.4 Composite Navigation Structure

Composite navigation structure is a combination of the previous structure and also called the free navigation structure; the point is if a display requires branching then made branching. This structure is most widely used in the creation of multimedia applications [15].

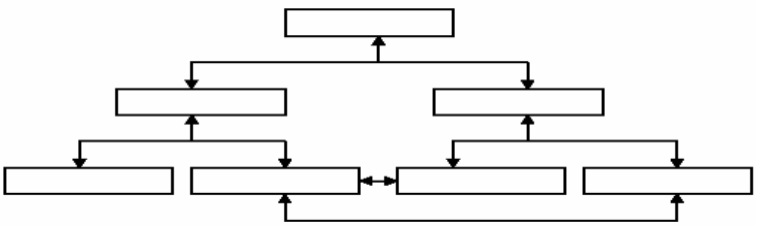


Figure 2.5 – Composite Navigation Structure [15]

2.6 Unified Modeling Language

The Unified Modeling Language (UML) is a general-purpose visual modeling language that is used to specify, visualize, construct, and document the artifacts of a software system. It captures decisions and understanding about systems that must be constructed. It is used to understand, design, browse, configure, maintain, and control information about such systems. The modeling language is intended to unify past experience about modeling techniques and to incorporate current software best practices into a standard approach [22].

The UML captures information about the static structure and dynamic behavior of a system. A system is modeled as a collection of discrete objects that interact to perform work that ultimately benefits an outside user. The static structure defines the kinds of objects important to a system and to its implementation, as well as the relationships among the objects. The dynamic behavior defines the history of objects over time and the communications among objects to accomplish goals. Modeling a system from several separate but related viewpoints permits it to be understood for different purposes [22].

2.6.1 Use-Case Diagram

Use case diagram is one of the UML diagrams that is used to represent the functionality of the system. It shows the functionality that the system will provide and the users who will communicate with the system to use that functionality. The use case diagram includes the nota-tions such as actors, use cases, communication association and the system or subsystem boundary. The use case diagrams provide interactions between roles known as actors and system to achieve a certain goal. Human or external system both are assumed as actors in definition of use case diagrams. Use cases define function nality of the system from a users' perspective and are used to document the system scope. Usually, use cases are used at a higher level in systems engineering as compared to their use in software engineering. Syntax use-case diagram can be seen in Figure 2.6.

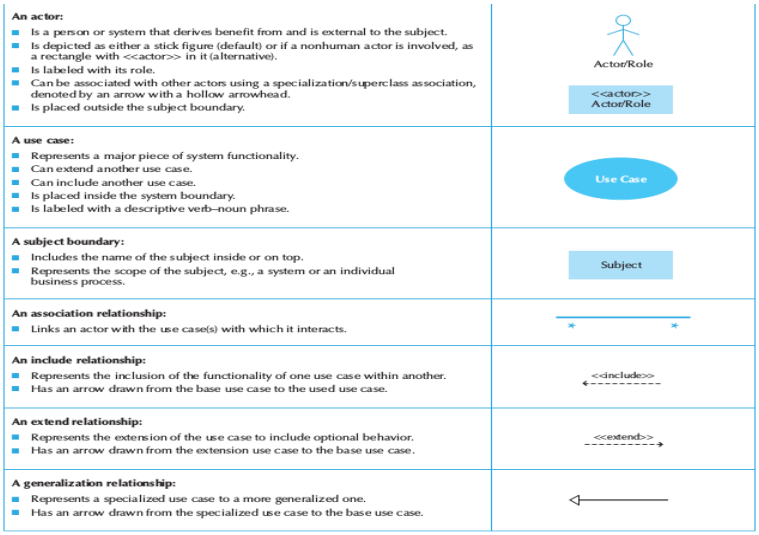


Figure 2.6 – Use-case diagram syntax [9]

2.6.1.1 Actor

An actor is an idealization of a role played by an external person, process, or thing interacting with a system, subsystem, or class. An actor characterizes the interactions that a class of outside users may have with the system. Different users may be bound to the same actor and therefore represent multiple instances of the same actor definition [22].

Each actor participates in one or more use cases. It interacts with the use case by exchanging messages. The internal implementation of an actor is not relevant in the use case; an actor may be characterized sufficiently by a set of attributes that define its state. An actor is drawn as a small stick person with the name below it [22].

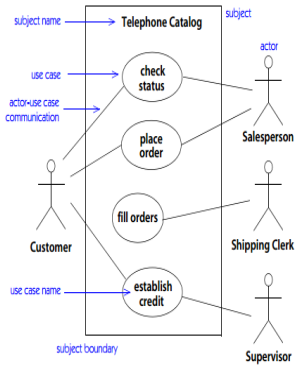


Figure 2.7 – Use-case Diagram

2.6.1.2 Activity Diagram

An activity is operation sequence from start to end the system done and per activity can be transformed on data or process [1]. Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In UML, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control [8].

2.7 Python

Python is a high level programming language (high level language) developed by Guido van Rossum in 1989 and introduced the first time in 1991. Python was born of the desire to simplify the programmer to complete tasks quickly. Python is designed to provide ease very remarkable for programmers both in terms of time efficiency, and ease of program development and in terms of compatibility with the system. Python can be used to create standalone program and programming scripts [20].

Implementation Python under an open source license that makes Python can be used and distributed freely, even for commercial purposes. Python provides libraries for desktop development, and the development of educational applications. PyBiblio an application for education related to many different sources. In addition, to software development, gaming and dimensional graphics and programming computer network. In addition to the support of the web and internet Python, Python provides support for programming level lower computer network, such as network interface socket.

2.8 Django

Django is a high-level Python web application framework designed to support the development of dynamic websites, web applications, and web services. It is designed to promote rapid development and clean, pragmatic design and build high-performing, elegant web application quickly. The main advantages of Django [10]:

1. Tight Integration between Components

First of all, Django provides a set of tightly integrated components; all of these components have been developed by the Django team themselves. Django was originally developed as an in-house framework for managing a series of news-oriented websites. Later its code was released on the Internet and the Django team continued its development using the Open Source model. Because of its roots, Django's components were designed for integration, reusability and speed from the start.

2. Object-Relational Mapper

   Django's database component, the Object-Relational Mapper (ORM), provides a bridge between the data model and the database engine. It supports a large set of database systems, and switching from one engine to another is a matter of changing a configuration file. This gives the developer great flexibility if a decision is made to change from one database engine to another.

3. Clean URL Design

   he URL system in Django is very flexible and powerful; it lets you define patterns for the URLs in your application, and define Python functions to handle each pattern. This enables developers to create URLs that are both user and search engine friendly.

4. Automatic Administration Interface

   Django comes with an administration interface that is ready to be used. This interface makes the management of your application's data a breeze. It is also highly flexible and customizable.

5. Advanced Development Environment

   In addition, Django provides a very nice development environment. It comes with a lightweight web server for development and testing. When the debugging mode is enabled, Django provides very thorough and detailed error messages with a lot of debugging information. All of this makes isolating and fixing bugs very easy.

6. Multi Lingual Support

   Django supports multi-lingual websites through its built-in internationalization system. This can be very valuable for those working on web-

sites with more than one language. The system makes translating the interface a very simple task.

The concepts of Django are similiar enough to a Model View Controller design, but Django is described as a Model Template View (MTV) framework. The Django models are modified slightly to better represent the structure of web applications. The View layer includes business logic and request handling, Templates determine the appearance of the websitehe, and the Model and object relational mapping layer defines data and relationships that are stored in the database [25].

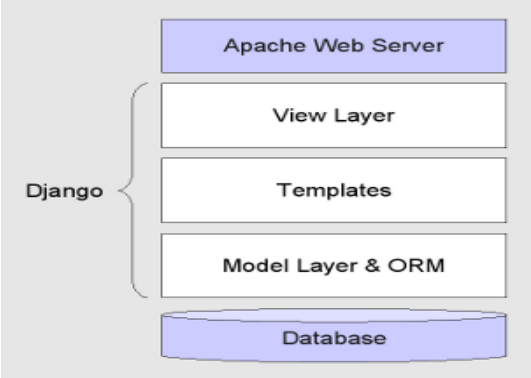


Figure 2.8 – Django Layers [25]

2.9 HTML

HTML is pure text file that can be created with any text editor. HTML is the language for describing the structure of Web pages. HTML gives the means to [19]:

- Publish online documents with headings, text, tables, lists, photos, etc.
- Retrieve online information via hypertext links, at the click of button.
- Design forms for conducting transactions with remote services, for use in searching for information, making reservations, ordering products, etc.
- Include spread-sheets, video clips, sound clips and other systems directly in their documents.

Nevertheless, until now HTML still standing strong as the basis of web languages like PHP, ASP, JSP and others. In fact, in general, the majority of web

sites on the Internet was still using HTML as the primary technology [8]. A basic HTML document consists of four sections defined by four sets of elements. For example, here is the code HTML (saved with the extension .htm or .html) [5].

```
1 <html>
2 <head>
3 <title> My Website </title>
4 </head>
5 <body> How to learn html </body>
6 </html>
```

2.10 Javascript

JavaScript is not related to Java, although it is a curly-brace language like Java, C#, C++, and many other programming languages. JavaScript is related to ECMAScript, however. Ecma International, working with other organizations, created this standardized scripting language in the ECMA-262 specification and ISO/IEC 16262. The language is widely used for client-side scripting on the web, and you commonly see several well-known variations of ECMAScript such as JavaScript, JScript, and ActionScript. The current release is ECMAScript Edition 5.1 and most common browsers support ECMAScript Edition 5.0 or newer [12].

2.11 AJAX

Rousseaux and Lhoste [21] described that with the faster evolution in IT tools rapid software prototyping is a best way to meet the demands of users instantly. With a classical software development cycle it is difficult to develop an application so fast which normally takes several months for development. By using RAD functionalities and interfaces can be revised with the passage of time. The term “Ajax” first used in a piece of written which is now online. Ajax is a latest way for the applications of web said by Jesse James Garrett in 2005 for the first time. Garrett named this phrase for the architecture and makes it equal like the new production group of web Applications. These new productions can be Google maps.

Ajax cannot be said as a actual language of programming. This is an advanced design model. It is made up of a lot of connected technologies and

thoughts. With the use of Ajax in applications of web we can get back data to the server asynchronously. Ajax helps in retrieve of data with no intervention on display. It refreshes the webpage without reloading it again. Some technologies implicated in Ajax are XHTML, CSS, XML, XSLT and JavaScript. Mostly web sites use AJAX. Like Facebook, Netvibes, Flickera and Google maps. This technology brought more flexibility and interactivity in Web-Pages. It adds functionalities like desktop to the web sites. Its aim is to almost finish the gaps between web and desktop [21].

2.12 MySQL

MySQL is the world's most popular open source database software, with over 100 million copies of its software downloaded or distributed throughout it's history. MySQL is a Relational Database Management System (RDBMS) that runs as a server providing multi-user access to a number of database. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of prop- rietary agreements. MySQL is a key part of LAMP (Linux, Apache, MySQL, PHP/ Perl/ Python), the fast-growing open source enterprise software stack [19].

The main features from MySQL are [19]:

1. Written in C and C++.
2. Working in a variety of platforms (eg Mac OS X, Solaris, Sun OS, Unix, Novel Netware, Windows, etc).
3. Provide transactions and no transactions storage engine.
4. The server is available as a separate program to use on the network client or server environment.
5. MySQL has a library that can be embedded in systems that run itself, so that the system can be used on computers that do not has network.

2.13 Testing

Software testing is the process of verification and validation whether a software system or program meets the requirements of business and technical requirements that drive the design and development and how it works

as expected and also identify critical errors that are classified based on the level severity in systems that must be corrected [**?**]. Software testing is a technique often used to verify and validate the quality of the software. Software testing is a procedure for executing a program or system in order to find fault.

2.13.1 Black Box Testing

Black box testing is also called as functional testing, a functional testing technique that designs test cases based on the information from the specification. Black box testing not concern with the internal mechanisms of a system, these are focus on the outputs generated in response to selected inputs and execution conditions. In black box testing, the software tester should not (or does not) have access to the internal source code itself [16]

Advantages of using Black Box Testing are [13]:

1. Efficient for large code segment.
2. Tester perception is very simple.
3. Users perspective are clearly separated from developers perspective (programmer and tester are independent of each other).
4. Quicker test case development.

Chapter 3

ANALYSIS AND DESIGN

3.1 Research Methodology

In conducting research and making this application, Rapid Application Development methodology is used. The research methodology refers to the model in Figure 3.1. The reseach begins with analysis of requirements phase which includes functional requirements and nonfunctional requirements. The second phase is design the application. After designing the application, the next phase is implementing the design and testing the application using Black Box testing.

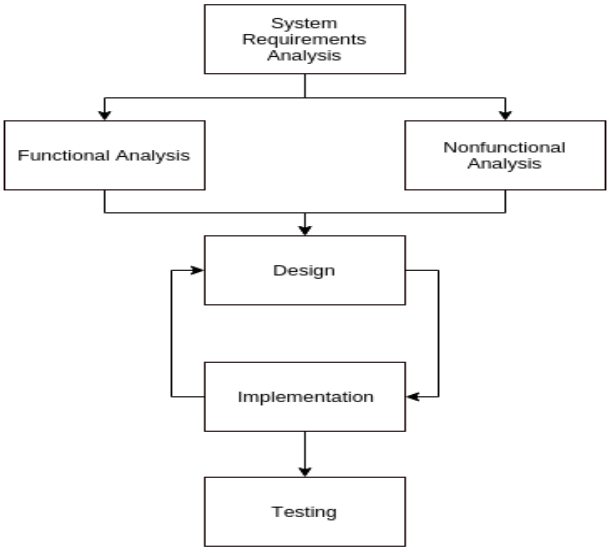


Figure 3.1 – Stage of Research

3.2 System Requirement Analysis

Animetacchi is a website where user can gives a review, rating of anime and manga. This website created by using mysql database, Python programming language and Django Framework. The main idea of the website is the anime and manga with details such as a synopsis, plot, number of episode , date of delivery, the anime characters and voice actors the anime character can be seen by the users. Besides anime, manga was included in this website. Manga is a comic, so there is no voice actor. The main focus of the website is a flexible interaction and a simple interface. Client does not restrict the technology used for front-end, but the technology used should suitable with the current trends. So ajax that are on jquery as a front-end interaction technology used was choosed.

3.2.1 Animetacchi Website Overview

Animetacchi is a website where user could give a review, rating of anime and manga. This website created by using mysql database, Python programming language and Django Framework.

3.2.2 Functional Requirement

Functional requirement analysis contains exposure of features that will be implemented into the application. These features include:

1. As a web-based application.
2. Users can be divided into members and non-members.
3. Non-member users can register before, member users can login to the application.
4. Member users and non members can view the entire contents of the website except user profiles for non member users.
5. Member users also can created a favorites lists, readinglist and watchlist, make comments, look for comments, sort comments based on the time of making comments or the amount of like, leave a like on the comment, give a rating, and view personal profiles.
6. Admin can input anime, manga, characters and voice actors.

3.2.3 Non-functional Requirement

Non-functional requirement analysis has performanced to know the specication required to make its system that consists of hardware and software.

3.2.3.1 Hardware Specifications

The specification of the computer used to make this website are:

1. Notebook: AXIOO Neon RNE
2. Processor: intel Core i7
3. RAM: 4 GB
4. Hardisk: 1 TB

3.2.3.2 Software Specifications

The tools used for making this website are:

1. Operating System : Linux Ubuntu 14.04 LTS
2. Programming Language: Python 2.7.6, HTML, Javascript, and CSS
3. Framework: Django Python 1.7 and Jauery
4. Database: Mysql
5. Editor: Geany dan Sublime 3
6. Browser: Google Chrome 47

3.3 Design

The system design includes navigation structure, Unified Modeling Language (UML) design, interface design, and database design.

3.3.1 Navigation Structure

The navigation structure is used to describe the application flow. This website is using composite navigation structure type. The structure of the navigation on this website can be seen in the following figure.

3.3.1.1 Admin Navigation Structure

Besides user navigation structure, there is an admin navigation structure that describes the Administrator page. After administrators Log In, the Site Administrator page will appear . On that page there are three modules. There are Animetacchi module, Authorization & Authentication Module, and Ratings module. Each module has several links that show the models and the models that link will show a list of the items each models. Log Out can be done directly in whole pages of Administrators. Animetacchi module consist of 14 models presented alphabetically, including Anime, Manga Character, Character, Chart, Comment Anime, Manga Comment, Daikirailist, Daisukilist, Genre, Manga, Member, Readinglist, User Profile, and Watchlist. Authentication & Authorization modules consist of Groups and Users models. As for Ratings consist of Score and Voices models. Here is the navigation structure for the Administrator.

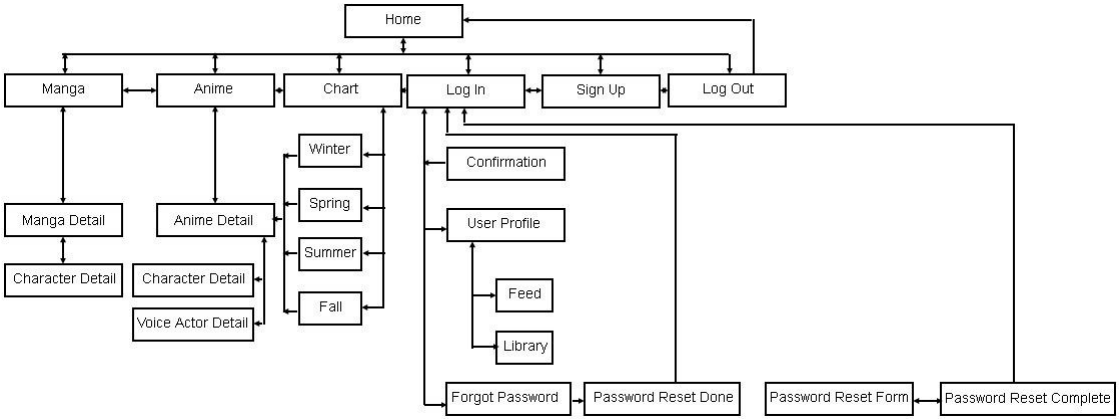


Figure 3.2 – Admin Naviagtion Structure

3.3.1.2 User Navigation Structure

First, user access this website, then there will be five menu and the Log Out menu was hidden in the user's profile photo that have been logged in. The five menu are Manga, Anime, Chart, Log In and Sign Up. Users who are not logged, able to access all the menus, but can not access the User Profile pages, feeds, and Library. Users who access the Manga page that contain of Manga list in the website, can access the Details Manga page that contain a description of the manga and can access the Character Detail page that contain of the description of the characters involved in the manga. Users who access the anime page that contain of anime list in the website,

can access the details Anime page that contain a description of the anime, can access the character detail page that contain of the description of the characters involved in the manga, and can access the voice actor details page that contains of the voiced characters bio involved in the anime. Users who access the Chart menu, can access four different pages. There are winter, spring, summer and fall. In that four page, there are a link to the anime detail page. In addition, users who have Sign Up will received an email containing a link to the confirmation page, and then in the confirmation page there is a link to access the login page. There are also Forgot Password page that can be accessed through the Log In page. After filling the email address on the page, users will lead to a Password Reset Done page. After sending the email address, the user receives an email from the system will direct the user to the password reset form page to fill a new password. After the filling process was completed, users will lead to a password reset complete page. User navigation structure can bee seen in Figure 3.3.

3.3.2 UML Design

The design of Unified Modeling Language (UML) is used as a modeling tool to explain the proposed website system.

3.3.2.1 Use Case Diagram

Use case diagrams describe the case, the actors and the relations between cases and actors. Use Case Diagram of the system can be seen in Figure 3.3 below.

There are three actors that can use this system, non member, member, and administrator. This website uses Log In system, so non member users have to sign up first, then the system will automatically send an email confirmation to the email address of the users. After confirms, the user can Log In and can do another things. Here is a list of roles and capabilities that can be performed by actors.

1. Non Member

   Non Members are users who have not yet signed up to the system. The actor is only able to seen the website without any restriction, and registration to the system. Registration was done to make non member users can be the member users who has a wide range of roles that are not owned by the Non Member.

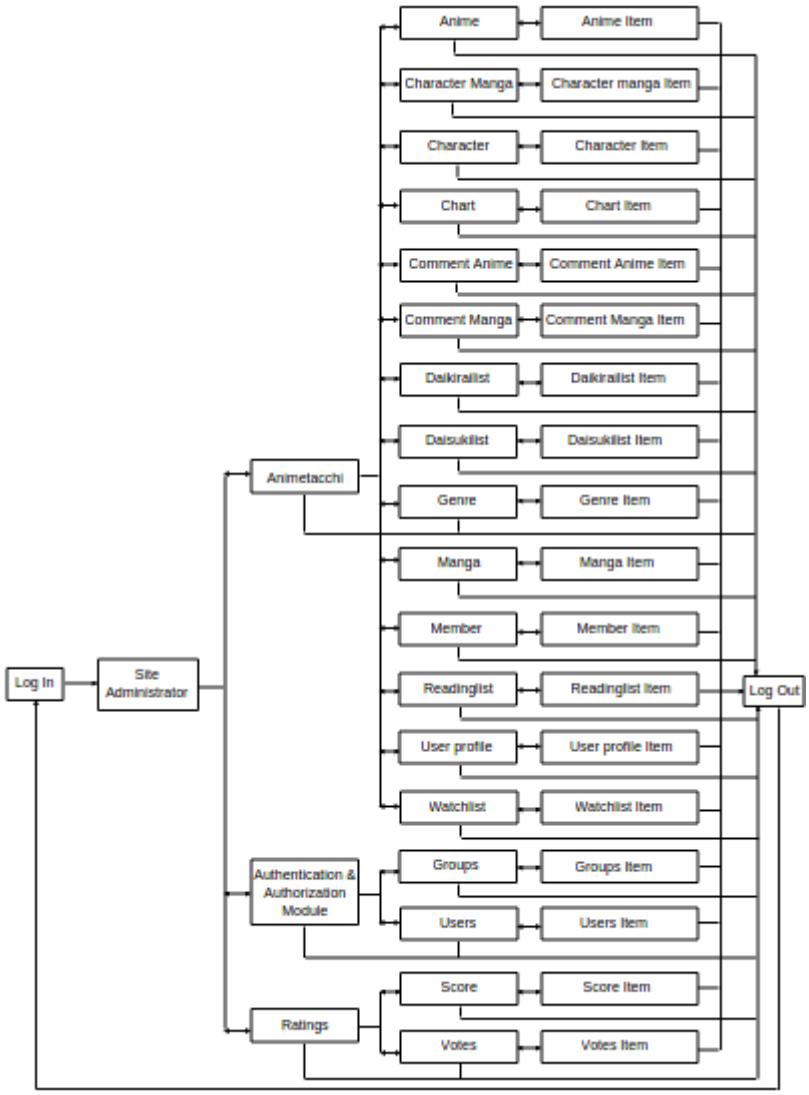


Figure 3.3 – User Navigation Structure

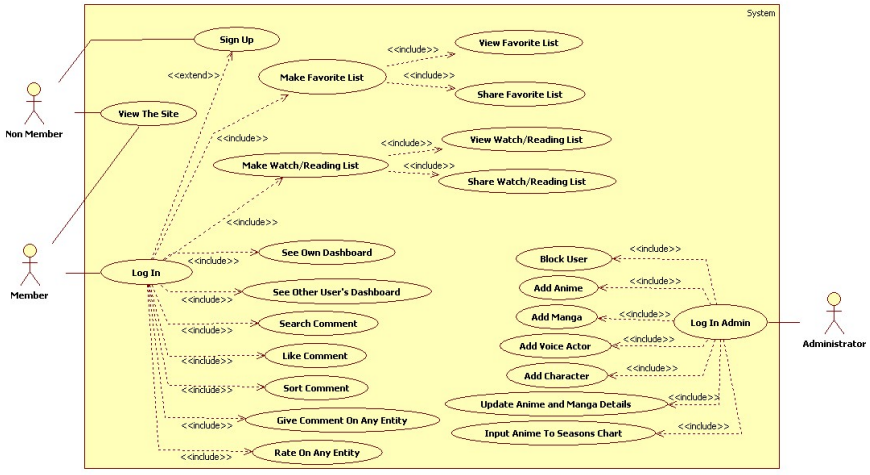


Figure 3.4 – Use Case Diagram

2. Member

   Members are users who have registered in the system that has several roles, among others.

   - Log In

   Members who have registered can Log In using the username and password that have been registered prior to entry into the system.

   - Make Favorite List

   Members can create a Favorite List which lists the character and voice actor users like, so that the member can add character and voice actor, and can see the list created and distributed the favorite list to Facebook.

   - Make Watch/Reading List

   Members can create a Watch / Reading List anime / manga that will be seen, so that the member is able to add the anime / manga, melhat list that has been created, and share to Facebook watch list.

   - See Own Profile

   Members can see personal data the users themselves.

   - See Other User's Profile

   Members can see personal data of other users.

   - Search Comment

   Member has the right look for any comments or feedback on anime or manga.

   - Like Comment

   Members will also be entitled to like comments or feedback from other users.

   - Sort Comment

   Members can sort comments to match the user's desire.

   - Give Comment

   On Any Entity Members can comment on manga, anime, character, or voice actors available.

   - Rate Entity

   Members are able to give an assessment of the existing entities in the system, namely anime, manga, character and voice actor.

- Administrator is authorized to block the user, perform the update details of anime and manga, anime and can enter into seasons chart.

3.3.2.2 Activity Diagram

Activity diagrams explained the process and activities in the system. The picture below showed the activity diagram for Animetacchi website.

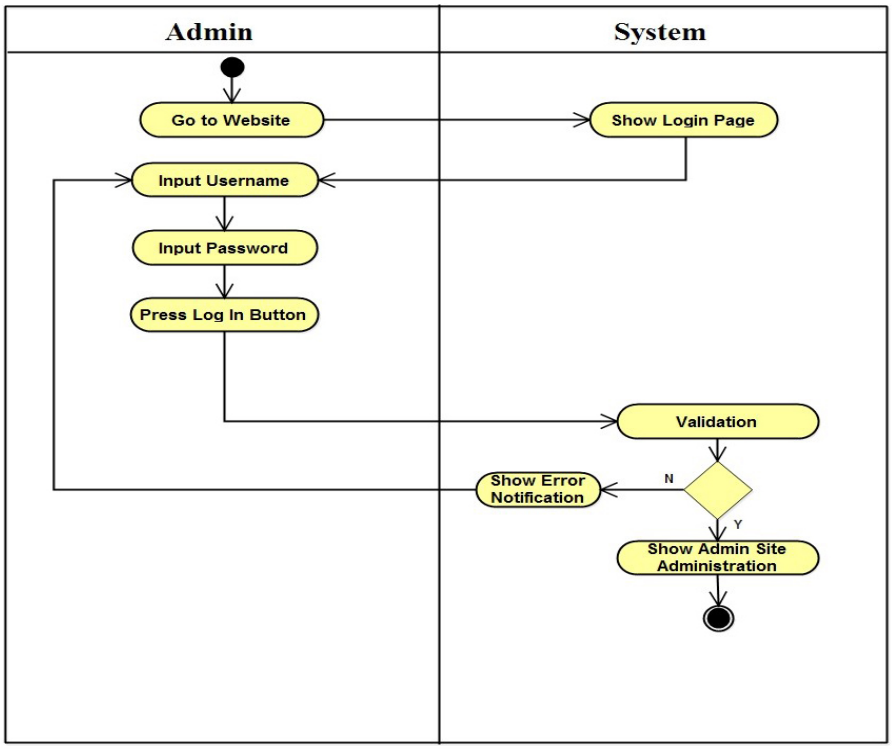


Figure 3.5 – Administrator Activity Diagram For Log In

In the Activity diagram above illustrates the current Administrator activity Log In. First Administrator open the website, then the system shows the Log In page. Then fill Administrator Username, Password, and press the button Log In. The system will process the data and generate the conditions, if the data filled Administrator true, then Log In the process successfully. But if the process fails, the Administrator must repeat activities from entering the Username, Password and press the button Log In.

After doing admin Log In, the first activity is doing is selecting modules from a page that has served the system. Admin then select Sub Module (Models). To add items to the models, admin pressing the Add Item, after the admin confronted conditions, if you choose to continue to add items, then the system will display a page Add Object so that the administrator can add objects. If the admin chose the opposite condition, then the Admin

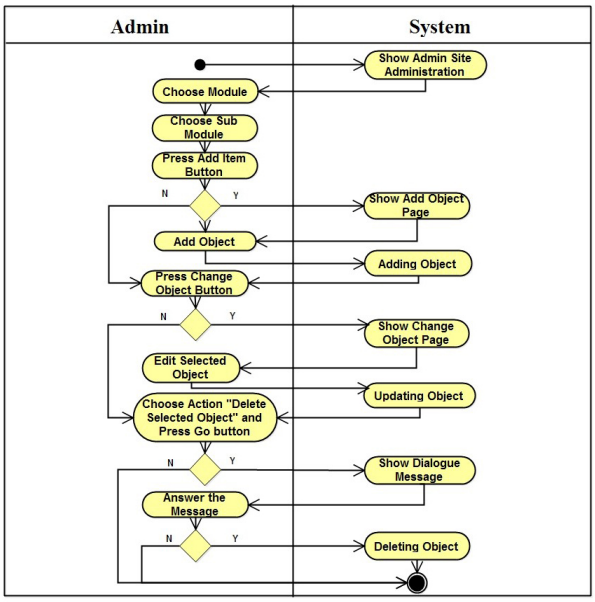


Figure 3.6 – Activity Diagram for Administrator Activities

can choose to press the button to change the object, and admin are also faced with the condition. If administrators want to continue the process of changing the object, the system will display the page Change Object so Admins can add new objects and update the system object. However, if the admin cancel the process, he can select actions such as deleting the selected object and pressing the Go button. If Admin select the action, he will be faced with the challenges that Admin sure you want to continue the action with the first response to messages from the system. If Admin resume action to delete the object, then the system will remove the object, but if the admin cancel deleting an object, admin activity ends.

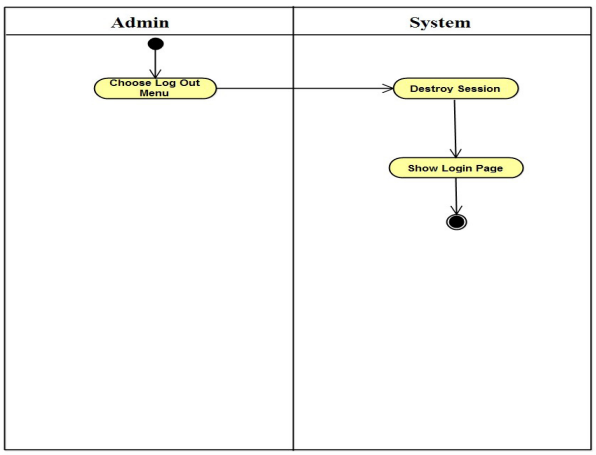


Figure 3.7 – Administrator Activity Diagram for Log Out

Activity diagram above illustrates the process that starts from the admin Log Out choose Log Out menu, then the system will Destroy Session, and then displays Log In admin page.

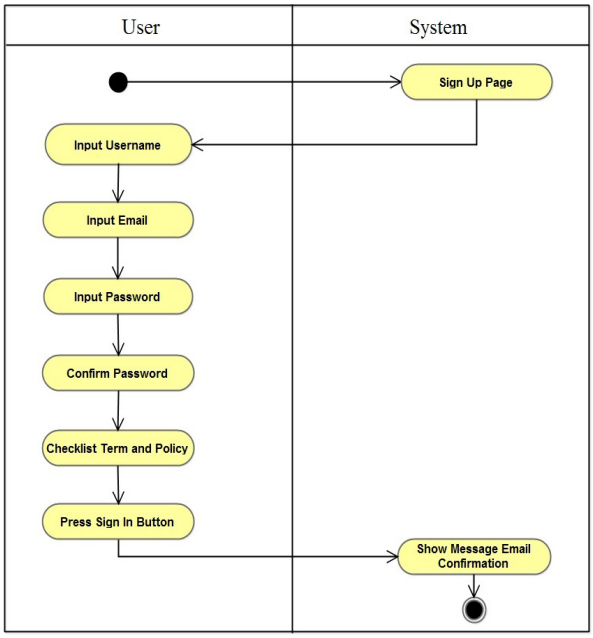


Figure 3.8 – Activity Diagram to New Member Sign Up

Activity Diagram above explains Sign Up user activity that originated from the user selects the menu Sign Up, then the system displays the Sign Up page. After that the user can enter data themselves include Username, Email, Password, Confirm password, Checklist Term and Policy, and then press the Sign In button. After making the process of sending user data, the system will display a confirmation email message.

Activity Log In the member begins when the user opens a website, through the system displays the main page of the website. After that the user can select the Log In, and then the system displays the Log In page. If a user forgets the password that has been created, the user can select the Forgot Password link, then the system displays the Forgot Password page. If the user does not forget the password, the user can fill in the form email and password, then press the Sign In button. after that, the system performs user validation. If the validation fails, the system will menampilkna error warning, but if successful, the system will display the Dashboard.

Once the user is on the Dashboard, the user can select the Library, the system will display a page Anime Library, but if the user mmemilih Manga Library menu, the system will display a page Manga Library.

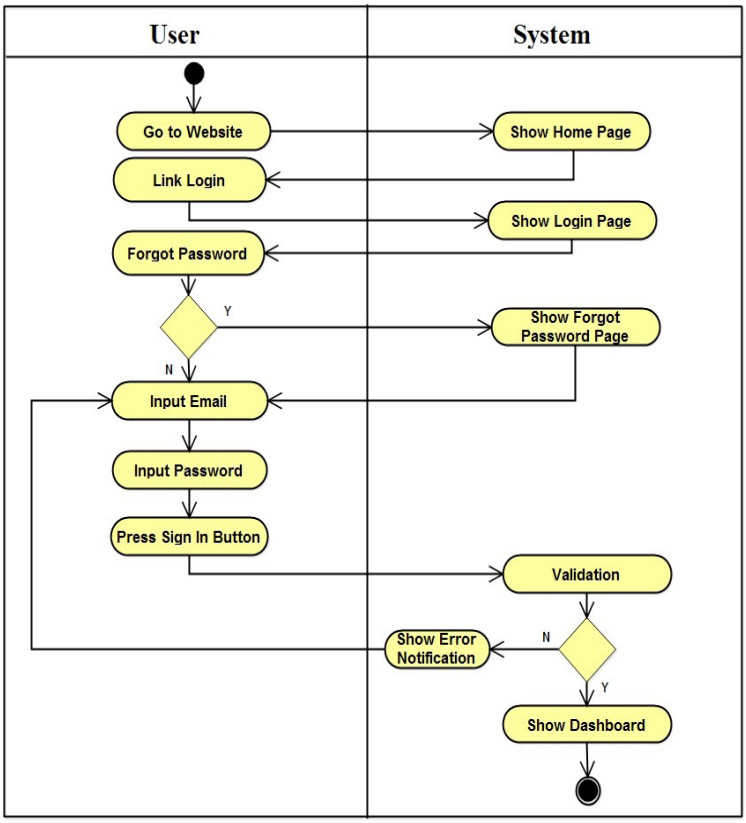


Figure 3.9 – Activity Diagram to Login Member

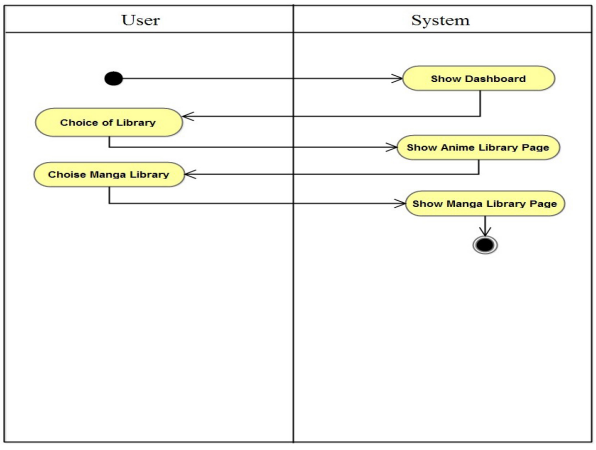


Figure 3.10 – Activity Diagram for Members on Dashboard

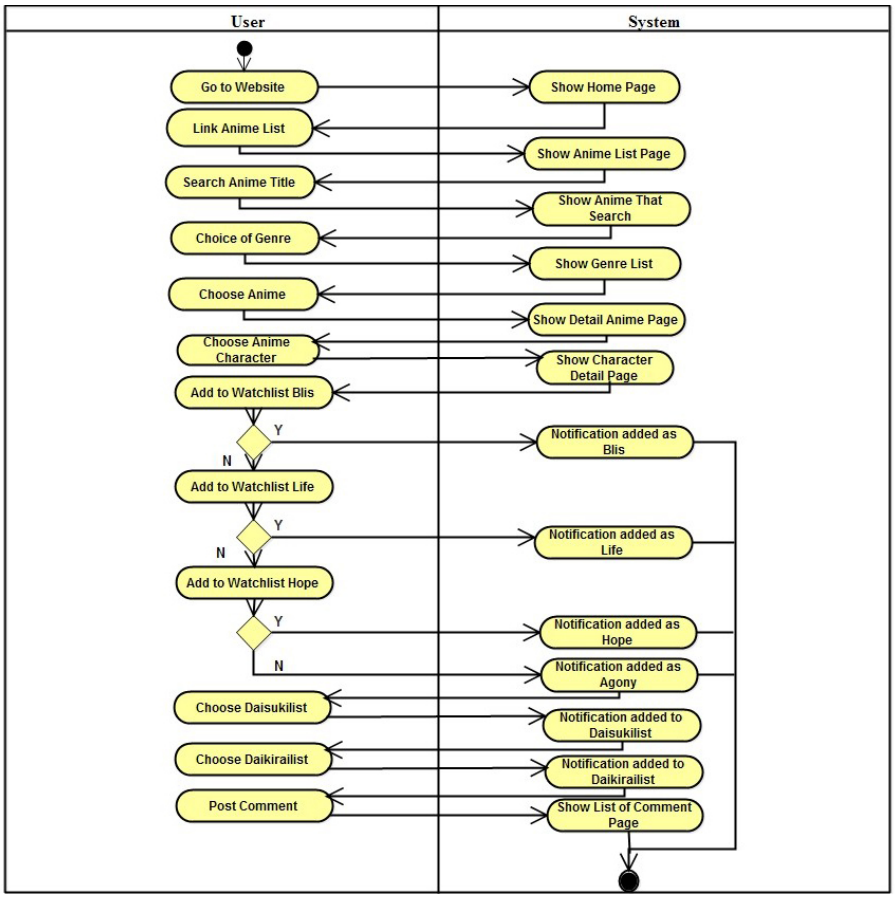


Figure 3.11 – Activity Diagram for Members on Anime

Activities performed at Anime registered members starting from opening a website and the system displays the main page. On the main page select menu Anime registered member, then the system displays the page Anime List. If the member did a search site, the system will display the search results Anime. If the user selects some of the genre, then the system displays a selection of Anime based on genre. If members choose one of the anime, the system will show the Details page Anime. If members choose one character in the anime is selected, then the system displays the page Character Detail page. If the user wants to add Anime to Watchlist Bliss, then the system displays a notification that has been added to the Watchlist Anime Bliss, if not, the user can add the category Anime Watchlist others seperty Agony, Life, or Hope. If the member wants to add Anime to Daisukilist, the system will display a notification that the process is successful. A similar thing happens when users add Anime to Daikirailist. If the member would like to comment on Anime selected, then the system displays a list of comments on the page.

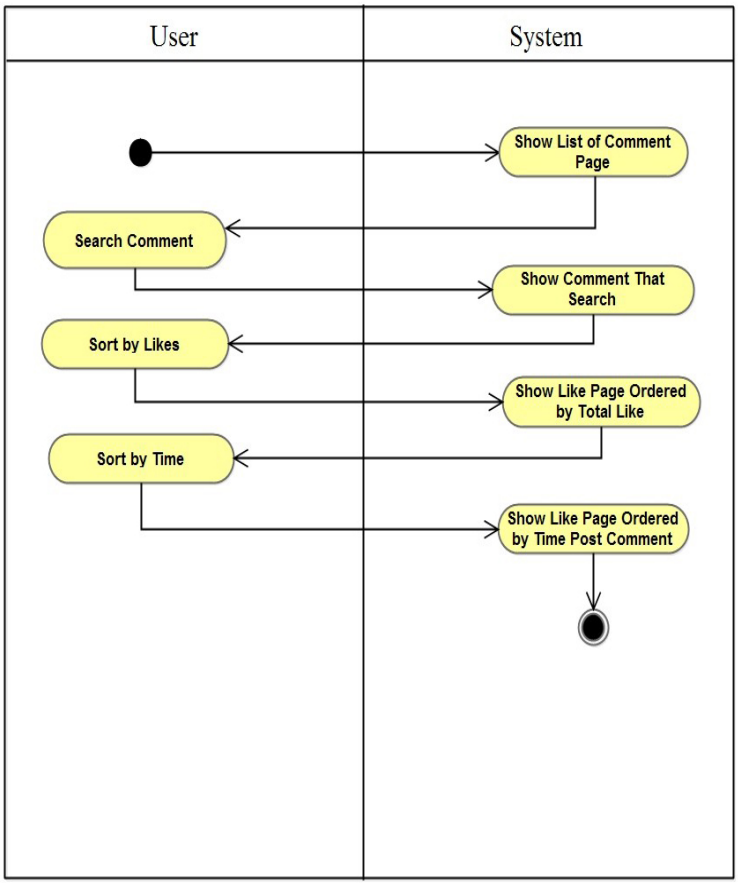


Figure 3.12 – Activity Diagram for Members Post Comments on Anime

Activities members when commenting divided into a number of other

activities. If the member is finished commenting, the system displays a list of the comments page. Members can search for comments, then the system displays the results sought comment. Members will also be able to sort the comments Based on the number of likes, then the system sort the comments. Members can also sort comments based on time, then the system will display a page containing comments ordered by time.

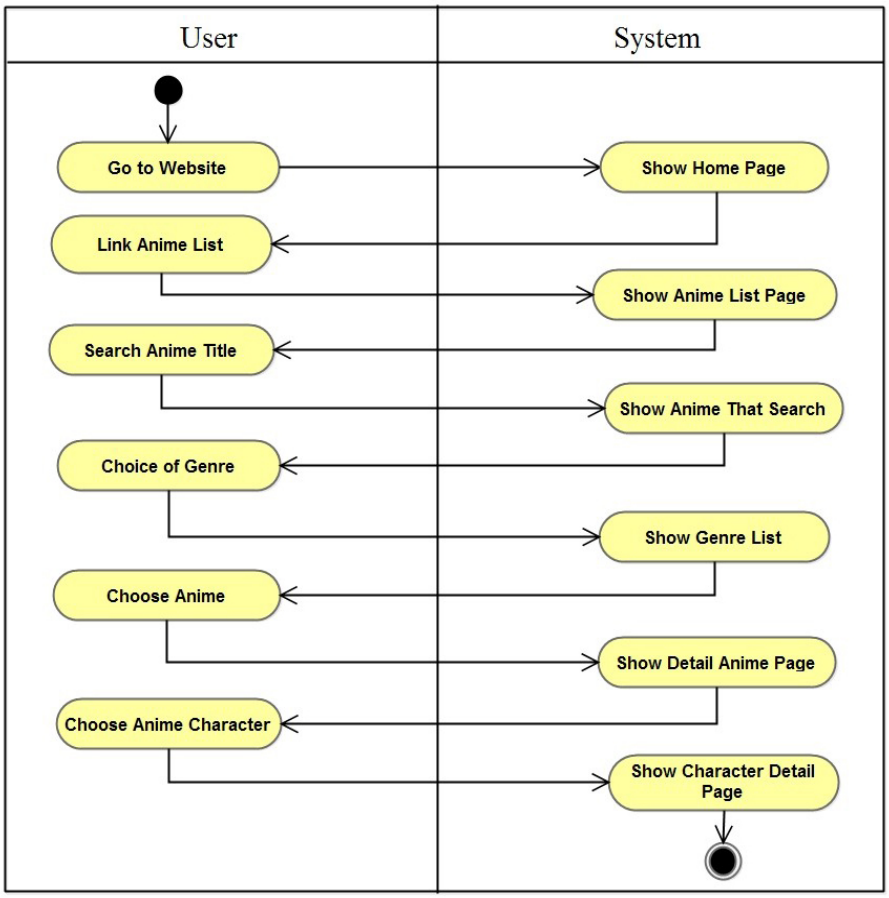


Figure 3.13 – Activity Diagram for Non Member Users on Anime

For users who are not registered at Anime activity stems from opening the website, and the system will display the main page of the website. The user can select multiple menu Anime then the system displays the page Anime List. Users are still able to do a search site, searching for Anime by genre, Anime choose to see the details, and choose anime character to look into details of the character.

Activities undertaken registered members on Manga starts from opening a website and the system displays the main page. On the main page select menu Manga registered member, then the system displays the page Manga List. If the member did a search Manga, the system will display the search

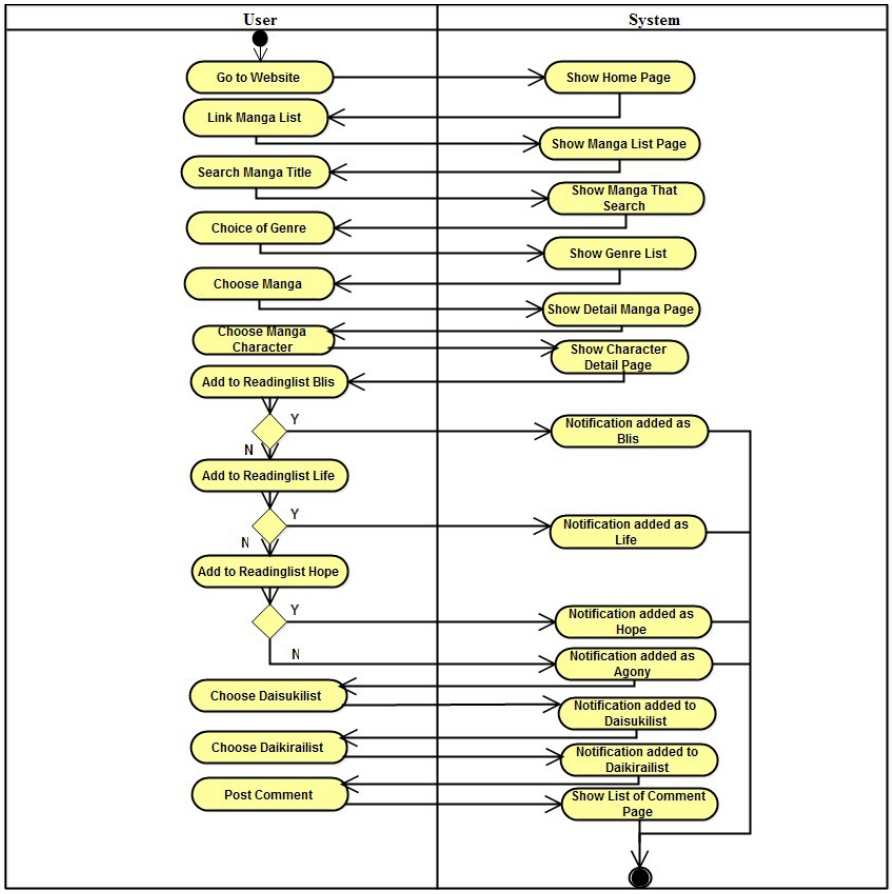


Figure 3.14 – Activity Diagram for Members on Manga

results Manga. If the user selects multiple genres, then the system displays a manga based on the selection of genre. If members choose one of the Manga, the system will show the Details page Manga. If members choose one character in the manga is selected, then the system displays the page Character Detail page. If the user wants to add Manga Readinglist to Bliss, then the system displays a notification that Manga has been added to Readinglist Bliss, if not, the user can add Manga on Readinglist other categories seperty Agony, Life, or Hope. If the member wants to add to Daisukilist Manga, the system will display a notification that the process is successful. A similar thing happens when a user adds to Daikirailist Manga. If the member would like to comment on Manga elected, then the system displays a list of comments on the page.

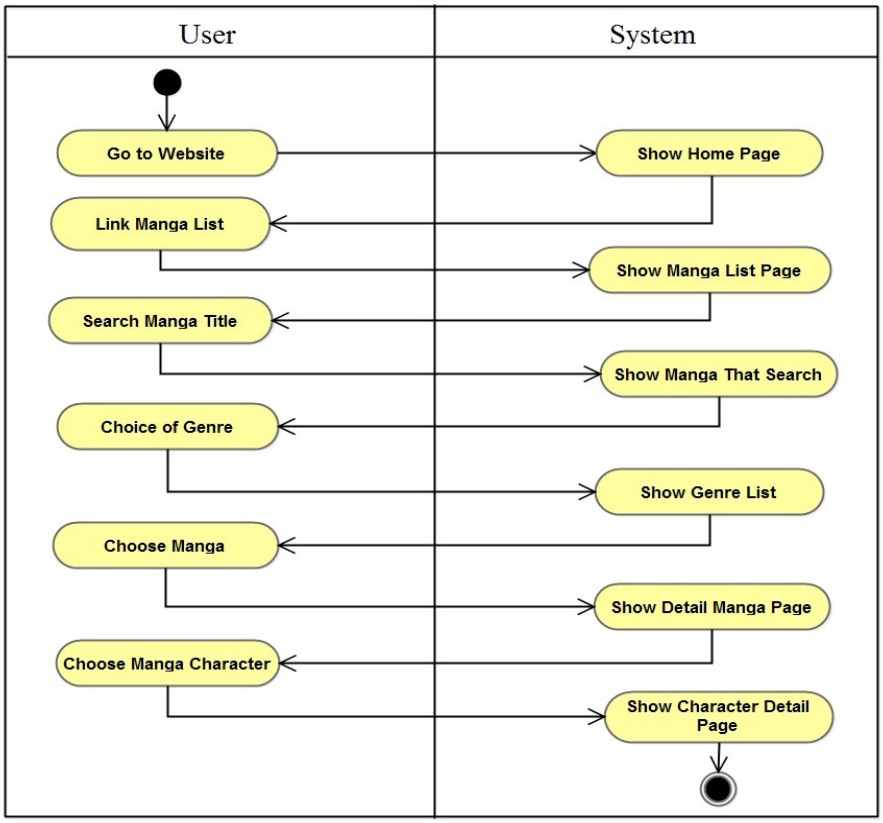


Figure 3.15 – Activity Diagram for Non Member Users on Manga

For users who are not registered activity in Manga stems from opening the website, and the system will display the main page of the website. The user can select multiple menu Manga then the system displays the page Manga List. Users will still be able to search Manga, searching by genre Manga, Manga chose to see the details, and choose a Manga character to look into

details of the character.

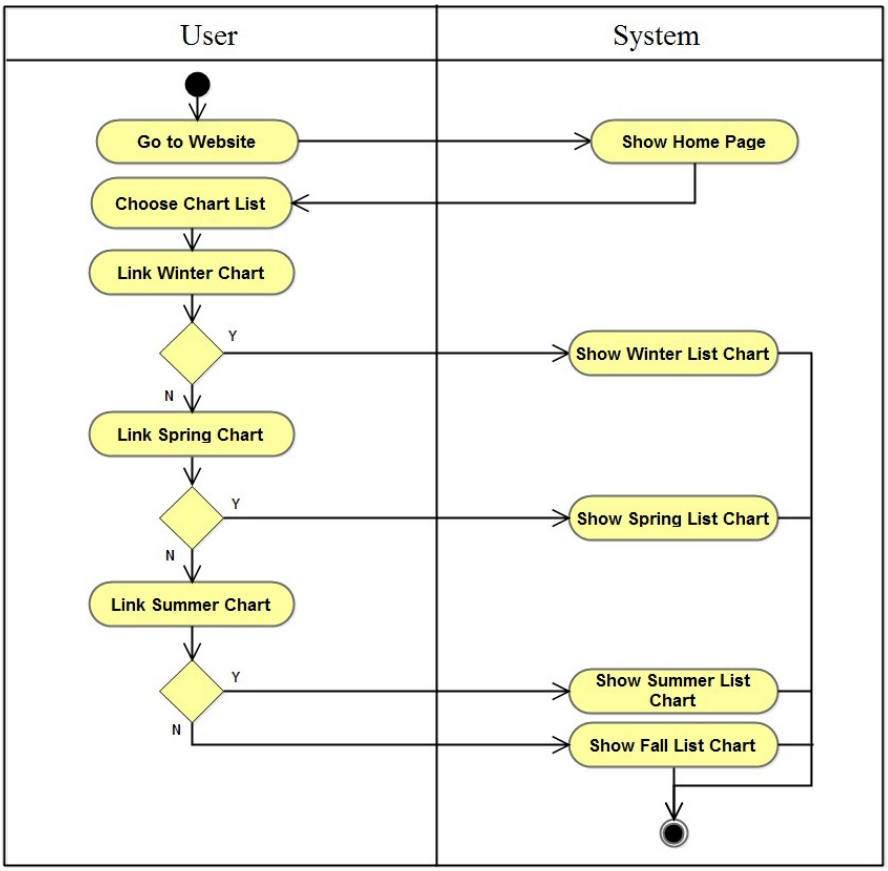


Figure 3.16 – Activity Diagram on Chart Page

Users who have not registered and members can choose Chart menu. After the system displays the main page, users can choose a chart based on four distinct seasons, including Winter, Spring, Summer, and Fall, the system will show the chart page selected by the user to suit the season.

3.3.3 Database Design

In building a a website, database is needed to store the data which related to the application. The database design is a structure that will be used in this website.

3.3.3.1 “Member” Table Design

Table "Member" consists of eight fields, including "user", "mname", "m_cover", "m_picture", "m_bio", "m_tagline", "blocked", and "seq_members". Field "user" is a foreign key derived from the User model.

Table 3.1 – "Member" Table Design

No	Field Name	Field Type	Note
1	user	ForeignKey	From User model
2	m_name	CharField	max_length=50, Null=True
3	m_cover	ImageField	Null=True, Blank=True
4	m_picture	ImageField	Null=True, Blank=True
5	m_bio	TextField	Null=True, Blank=True
6	m_tagline	CharField	max_length=50, Null=True
7	blocked	BooleanField	Default=False
8	seq_members	IntegerField	Null=True

3.3.3.2 "Genre" Table Design

Table "Genre" consists of one field, namely "genre_type" with a maximum length of 30 characters.

Table 3.2 – "Genre" Table Design

No	Field Name	Field Type	Note
1	genre_type	CharField	max_length=30, Null=True

3.3.3.3 "Anime" Table Design

Table "Anime" consists of ten fields, among others "a_name", "a_genre", "a_cover", "a_displaypic", "a_synopsys", "a_airedstart", "a_airedend", "save_added", "slug" , "Slug" field is used to store and generate valid URLs for dynamically created web pages. It is auto-generated. "a_genre" are fields pliers have a relationship many-to-many to the model of genre, so it has a lot of meaning can Anime has many genres.

Table 3.3 – "Anime" Table Design

No	Field Name	Field Type	Note
1	a_name	CharField	max_length=50, Null=True
2	a_genre	ManyToManyField	From Genre model
3	a_cover	ImageFiled	Null=True, Blank=True
4	a_displaypic	ImageFiled	Null=True, Blank=True
5	a_synopsys	TextField	Null=True, Blank=True
6	a_airedstart	DateField	Null=True, Blank=True
7	a_airedend	DateField	Null=True, Blank=True
8	save_added	DateTimeField	auto_now=True
9	slug	SlugField	Null=True, Blank=True
10	seq_anime	IntegerField	NullTrue, Blank= True

3.3.3.4 "Manga" Table Design

Table "Manga" is composed of ten fields, among other "name", "genre", "cover", "displaypic", "synopsys", "airedstart", "airedend", "save_added", "slug". "Genre" is a tang field has a relation many-to-many to the model of genre, so it has a lot of meaning can Anime has many genres.

Table 3.4 – "Manga" Table Design

No	Field Name	Field Type	Note
1	name	CharField	max_length=50, Null=True
2	genre	ManyToManyField	From Genre model
3	cover	ImageFiled	Null=True, Blank=True
4	displaypic	ImageFiled	Null=True, Blank=True
5	ynopsys	TextField	Null=True, Blank=True
6	airedstart	DateField	Null=True, Blank=True
7	airedend	DateField	Null=True, Blank=True
8	save_added	DateTimeField	auto_now=True
9	slug	SlugField	Null=True, Blank=True
10	seq_anime	IntegerField	NullTrue, Blank= True

3.3.3.5 "Watchlist" Table Design

Table "Watchlist" consists of four fields, among other "members", "watchlist", "status", and "seq_watchlist". "Members" is a foreign key Members of the model, while the "watchlist" is a foreign key of the model Anime.

Table 3.5 – "Watchlist" Table Design

No	Field Name	Field Type	Note
1	members	ForeignKey	From Members model, Null=True, Blank=True
2	watchlist	ForeignKey	From Anime model, Null=True, Blank=True
3	status	CharField	max_length=10, Null=True
4	seq_watchlist	IntegerField	Null=True

3.3.3.6 "Readinglist" Table Design

Table "Reading List" consists of four fields, among other "members", "readinglist", "status", and "seq_watchlist". "Members" is a foreign key of the model Members, while "readinglist" is a foreign key of the model Manga.

Table 3.6 – “Readinglist” Table Design

No	Field Name	Field Type	Note
1	members	ForeignKey	From Members model, Null=True, Blank=True
2	readinglist	ForeignKey	From Manga model, Null=True, Blank=True
3	status	CharField	max_length=10, Null=True
4	seq_watchlist	IntegerField	Null=True

3.3.3.7 “Character” Table Design

Table "Character" is composed of four fields, among other "name", "picture", "anime", and "synopsys".

Table 3.7 – “Character” Table Design

No	Field Name	Field Type	Note
1	name	CharFiled	max_length=50, Null=True
2	picture	ImageField	Null=True, Blank=True
3	anime	ForeignKey	From Anime model, Null=True, Blank=True
4	synopsys	TextFiled	Null=True, Blank=True

3.3.3.8 ”CharacterManga” Table Design

Table "Manga Character" consists of four fields, among other "name", "picture", "manga" and "synopsys".

Table 3.8 – ”CharacterManga” Table Design

No	Field Name	Field Type	Note
1	name	CharFiled	max_length=50, Null=True
2	picture	ImageField	Null=True, Blank=True
3	manga	ForeignKey	From Manga model, Null=True, Blank=True
4	synopsys	TextFiled	Null=True, Blank=True

3.3.3.9 “VoiceCharacter” Table Design

Table "Voice Character" consists of four fields, among other "name", "picture", "character", and "synopsys".

3.3.3.10 “Daisukilist” Table Design

Table "Daisukilist" consists of six fields, among other "members", "anime", "manga", "character", "character_manga", and "voice_character". Table

Table 3.9 – "VoiceCharacter" Table Design

No	Field Name	Field Type	Note
1	name	CharFiled	max_length=50, Null=True
2	picture	ImageField	Null=True, Blank=True
3	character	ManyToManyField	Null=True, Blank=True
4	synopsys	TextFiled	Null=True, Blank=True

"Daisukilist" useful for accommodating Anime, Manga, Character of Anime, Manga and Character of Anime Voice Actor preferred members.

Table 3.10 – "Daisukilist" Table Design

No	Field Name	Field Type	Note
1	members	ForeignKey	From Members model
2	anime	ManyToManyField	Null=True, Blank=True
3	manga	ManyToManyField	Null=True, Blank=True
4	character	ManyToManyField	Null=True, Blank=True
5	character_manga	ManyToManyField	Null=True, Blank=True
6	voice_character	ManyToManyField	Null=True, Blank=True

3.3.3.11 "Daikirailist" Table Design

Table "Daikirailist" consists of six fields, among other "members", "anime", "manga", "character", and "character_manga". Table "Daisukilist" useful for accommodating Anime, Manga, Character of Anime and Manga Character of the hated members.

Table 3.11 – "Daikirailist" Table Design

No	Field Name	Field Type	Note
1	members	ForeignKey	From Members model
2	anime	ManyToManyField	Null=True, Blank=True
3	manga	ManyToManyField	Null=True, Blank=True
4	character	ManyToManyField	Null=True, Blank=True
5	character_manga	ManyToManyField	Null=True, Blank=True

3.3.3.12 "Chart" Table Design

Table "Chart" consists of three fields, among others, "season", "anime", and "year".

Table 3.12 – "Chart" Table Design

No	Field Name	Field Type	Note
1	seasons	CharField	max_length=10
2	anime	ManyToManyField	Null=True, Blank=True
3	year	IntegerField	Null=True, Blank=True

3.3.3.13 "CommentAnime" Table Design

Table "CommentAnime" consists of six fields, among others "comment", "user", "anime", "added", "edited", and "votes". This table to accommodate comments for Anime.

Table 3.13 – "CommentAnime" Table Design

No	Field Name	Field Type	Note
1	comment	TextField	
2	user	ForeignKey	From Members model, Null=True, Blank=True
3	anime	ForeignKey	From Anime model, Null=True, Blank=True
4	added	DateTimeField	Null=True, Blank=True
5	edited	DateTimeField	Null=True, Blank=True
6	votes	VotableManager()	

3.3.3.14 "CommentManga" Table Design

Table "Comments Manga" consists of six fields, among others "comment", "user", "manga", "added", "edited", and "votes". This table is used to store comments against Manga.

Table 3.14 – "CommentManga" Table Design

No	Field Name	Field Type	Note
1	comment	TextField	
2	user	ForeignKey	From Members model, Null=True, Blank=True
3	manga	ForeignKey	From Manga model, Null=True, Blank=True
4	added	DateTimeField	Null=True, Blank=True
5	edited	DateTimeField	Null=True, Blank=True
6	votes	VotableManager()	

3.3.3.15 "UserProfile" Table Design

Table "User Profile" consists of three fields, among others, "user", "activation key", "key_expires". "User" shows the relationship one-to-one against a model Member, so that one member can only have one account.

Table 3.15 – "UserProfile" Table Design

No	Field Name	Field Type	Note
1	user	OneToOneField	From User Model
2	activation_key	CharField	max_length=40, Blank=True
3	key_expires	DateTimeField	default=datetime.now

3.3.4 Interface Design

Interface design is needed in the process of making an application in order to facilitate the process of making the website.

3.3.4.1 Home page for member and nonmember

Home page is the first page that will appear when opening a website. In this application there are two types of the home page, which is the home page for users who are already a member, and the home page for users who are not registered (Non Member), so there are some menu located at the top navigation bar of a design that is similar to, among others link to the home, menu Anime, Games, and the Chart menu will feature a choice of four season, including Winter, Spring, Summer, and Fall. On the home page for the member users who have logged in, then it will only display three main menus and a user frofil photo. If the profile photo is clicked, it will display the menu View Profile and Sign Out. Shown in Figure 3.1 there is a great picture. Images displayed shaped slider image will look to fill the pages with some excerpts from some anime. In the bottom image slider there is news of the latest anime featuring the main image of anime reported, the genre of anime and anime juduk. At the bottom there is a footer section that displays the year of manufacture and the copyright of the manufacturer's website. Section navigation bar and footer sections will always be displayed on each page with the same format. The following home page designs for users Member and Non Member users.

3.3.4.2 Sign in page

Sign in page for users who have registered in the system to enter the system. On this page there is a background image that fills the screen. Then in the middle of the page there is a box that contains a form for entering your email address and password for users who have registered for the submit button, and the forgot password link. In addition, for users who are not

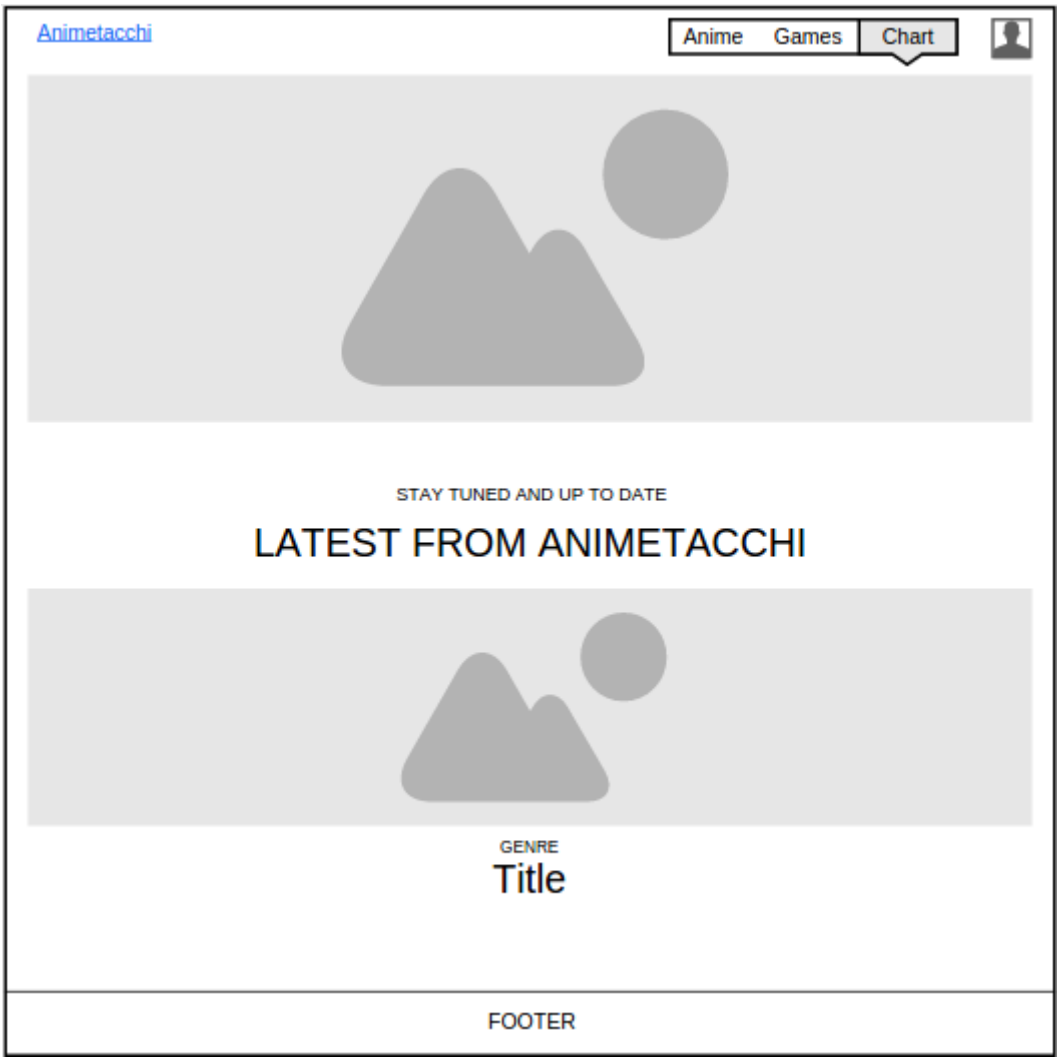


Figure 3.17 – Home For Members Page Views

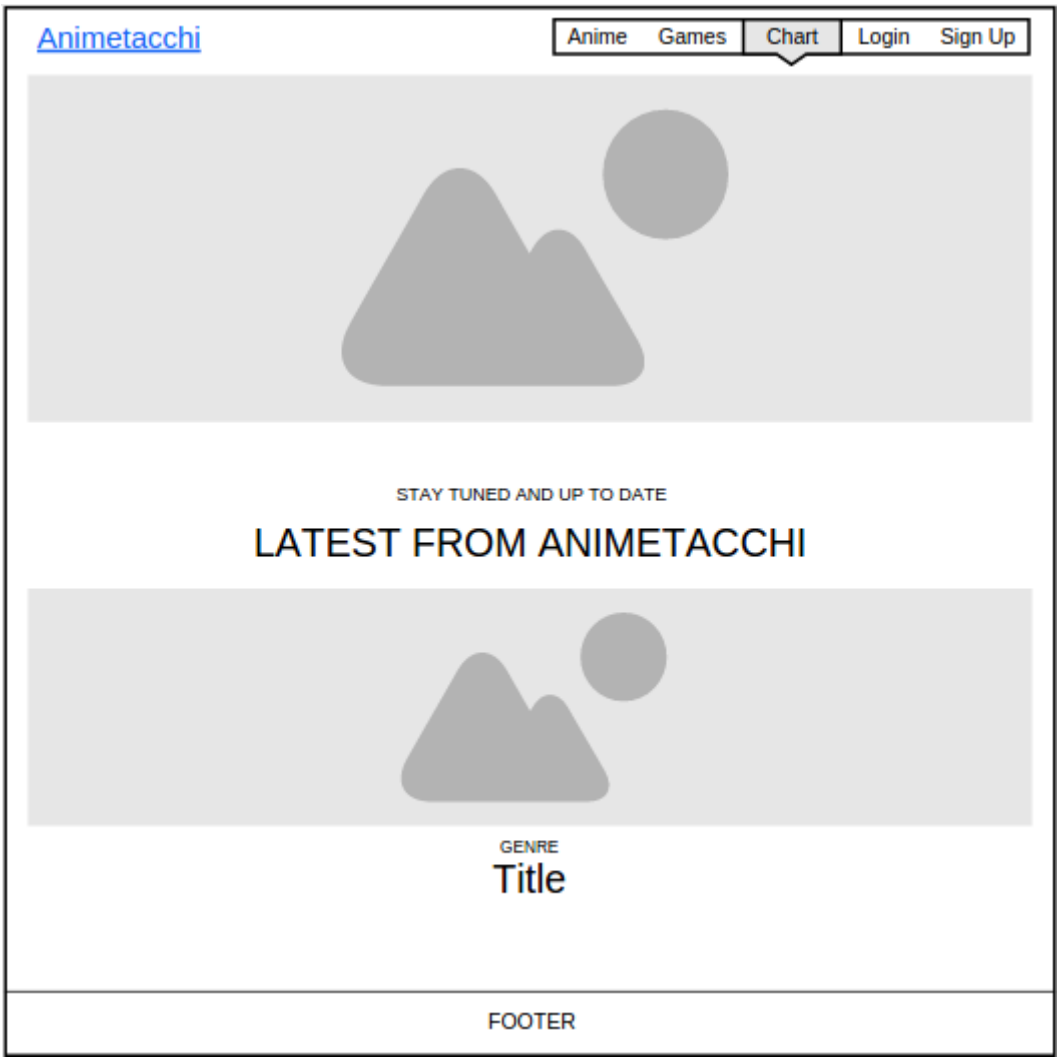


Figure 3.18 – Home For Non Member Page Views

registered in the system but already opened this page, provided the key that will go to the Sign Up page. The following page designs Sign In.

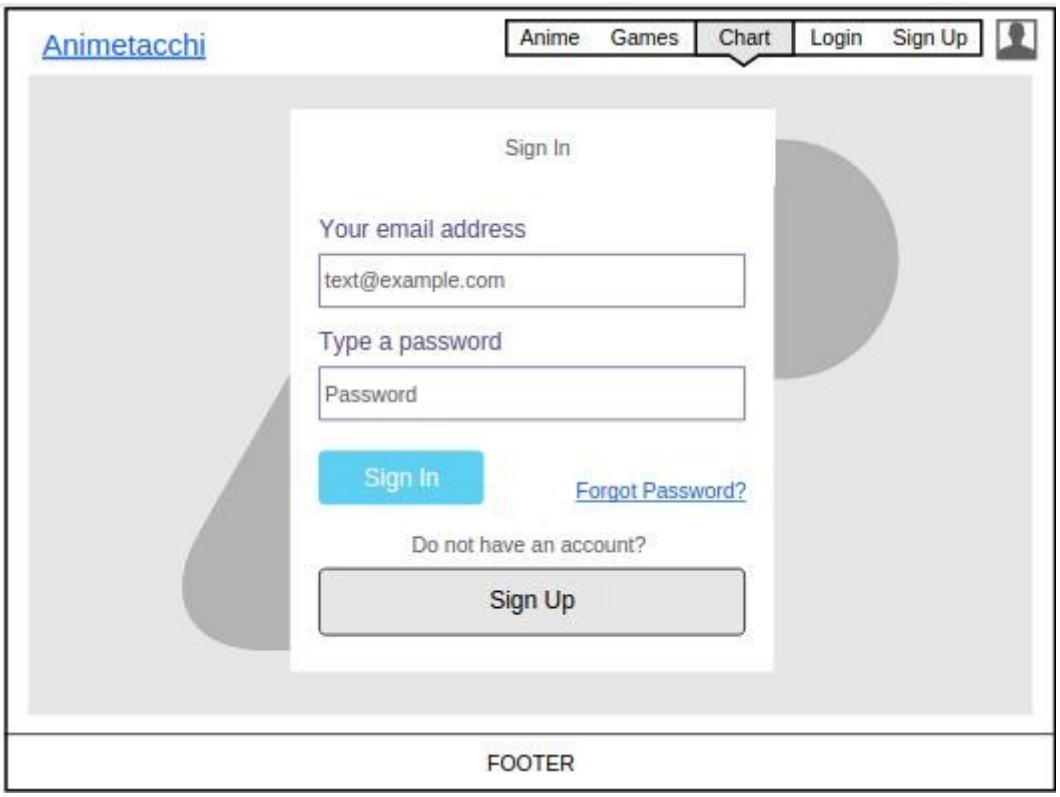


Figure 3.19 – Sign In Page Views

3.3.4.3 Sign Up page

Sign Up page for users who have not registered in the system. On this page there is a background image that fills the screen. Then in the middle of the page there is a box that contains a form to enter data such as username, email address and password and confirm the password for the user who has registered, checkboxes to approve policies and regulations, and a button to submit. In addition, for users who are already registered in the system but already opened this page, provided the key that will go to the Sign In page. The following page designs Sign Up.

3.3.4.4 Forgot Password page

Forgot Password page for users who forgot his password and wants to change the password. On this page there is a background image that fills the screen. Then in the middle of the page there is a box that contains a form to enter data and email address to the submit button. In addition, for users

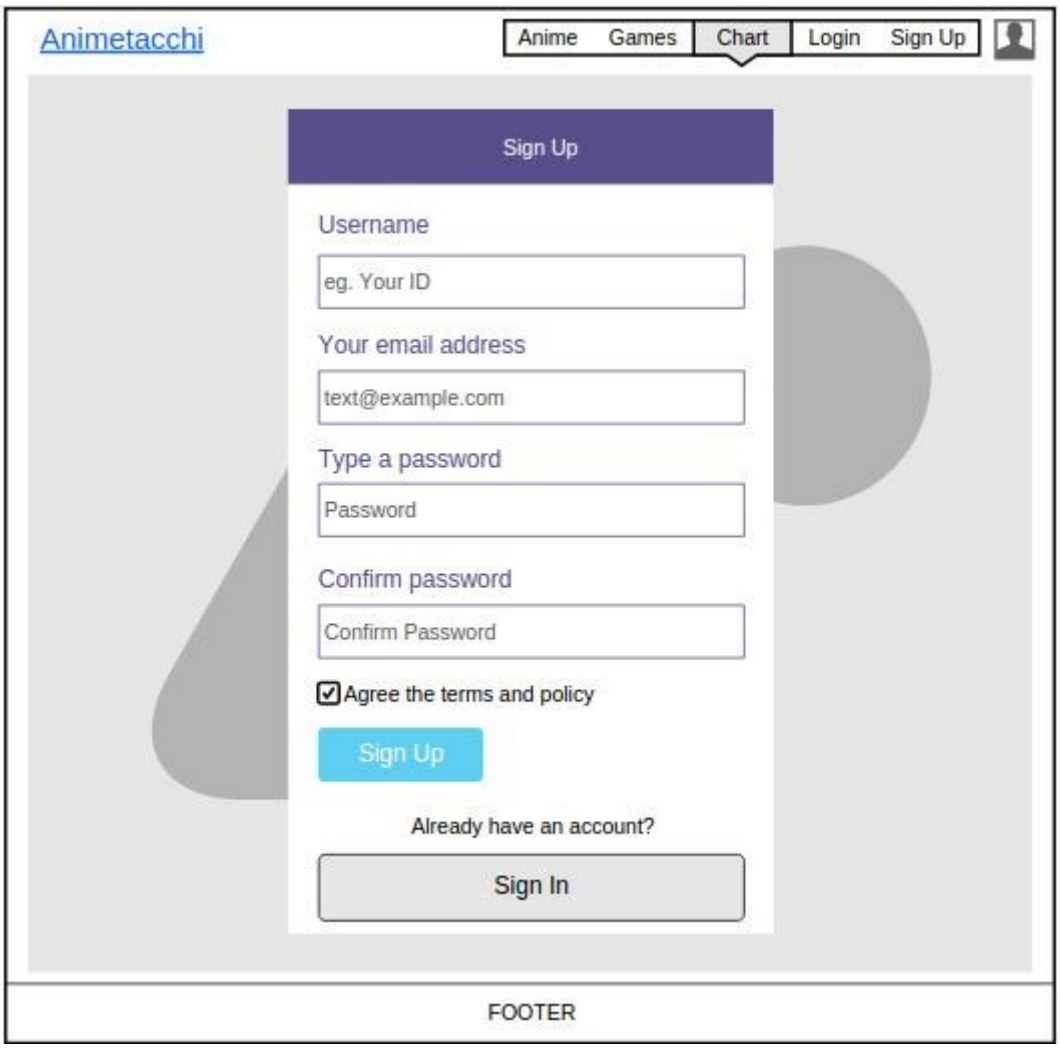


Figure 3.20 – Sign Up Page Views

who want to cancel changing the password contained Cancel button. The following page designs Forgot Password.

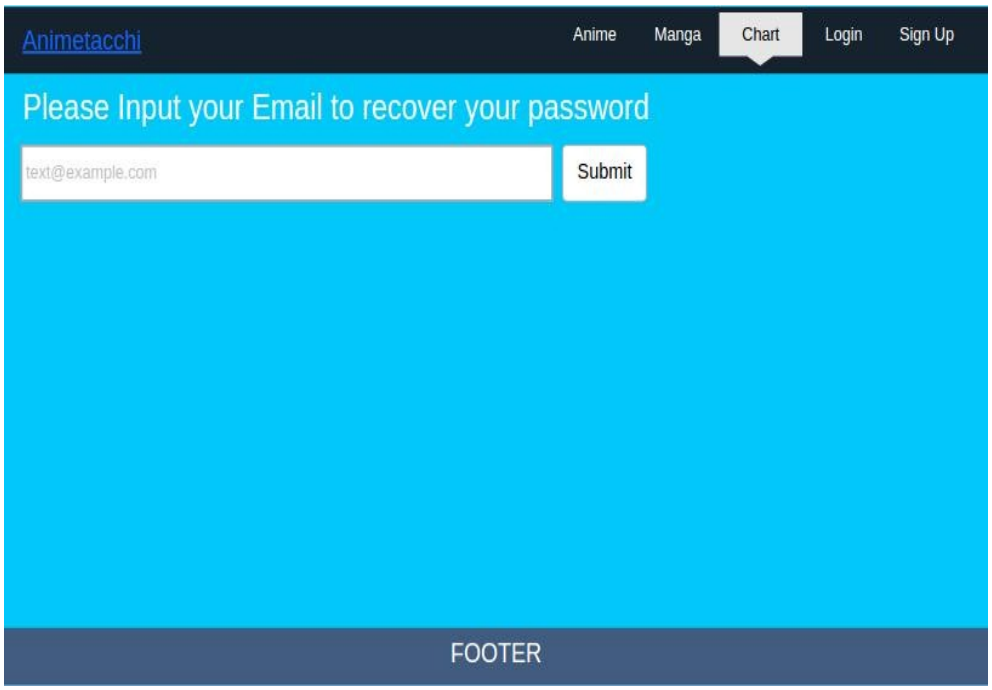


Figure 3.21 – Forgot Password Page views

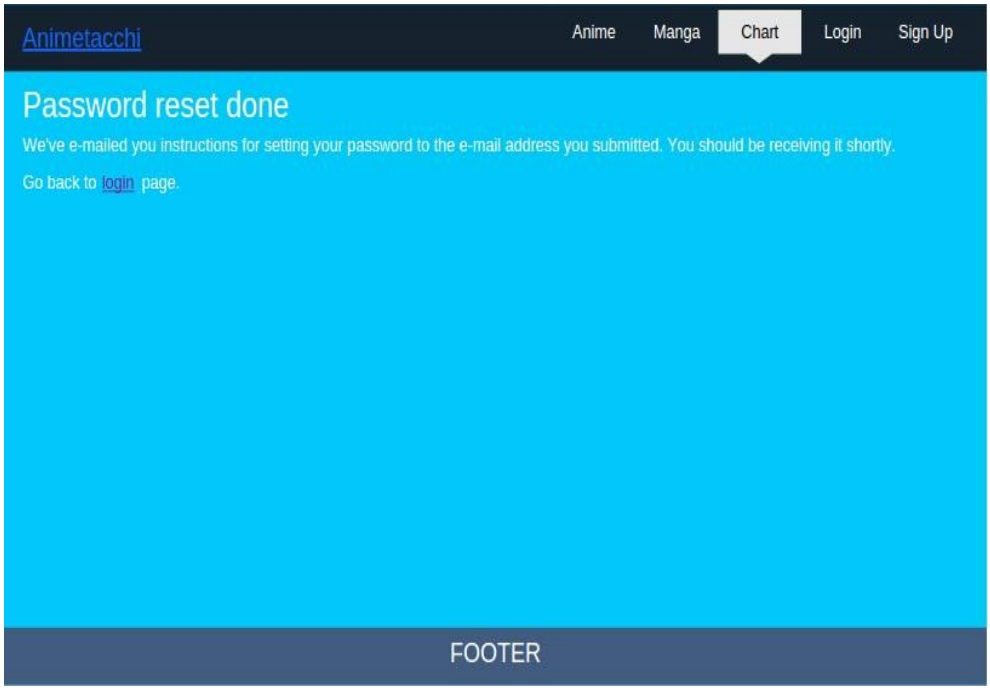


Figure 3.22 – Forgot Password Reset Done Page Views

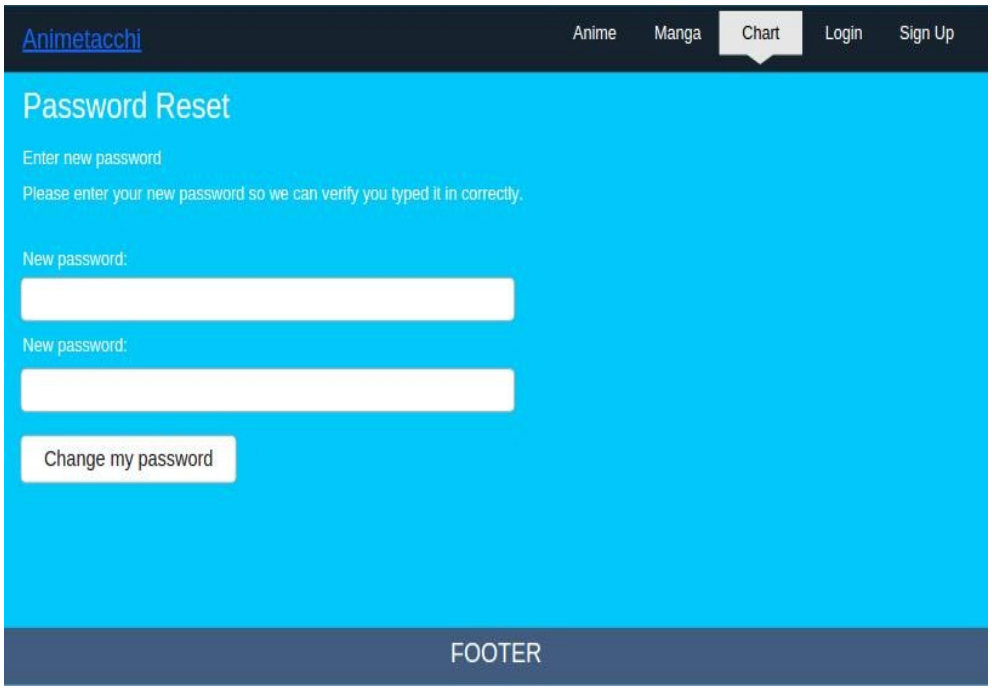


Figure 3.23 – Form Forgot Password Page Views

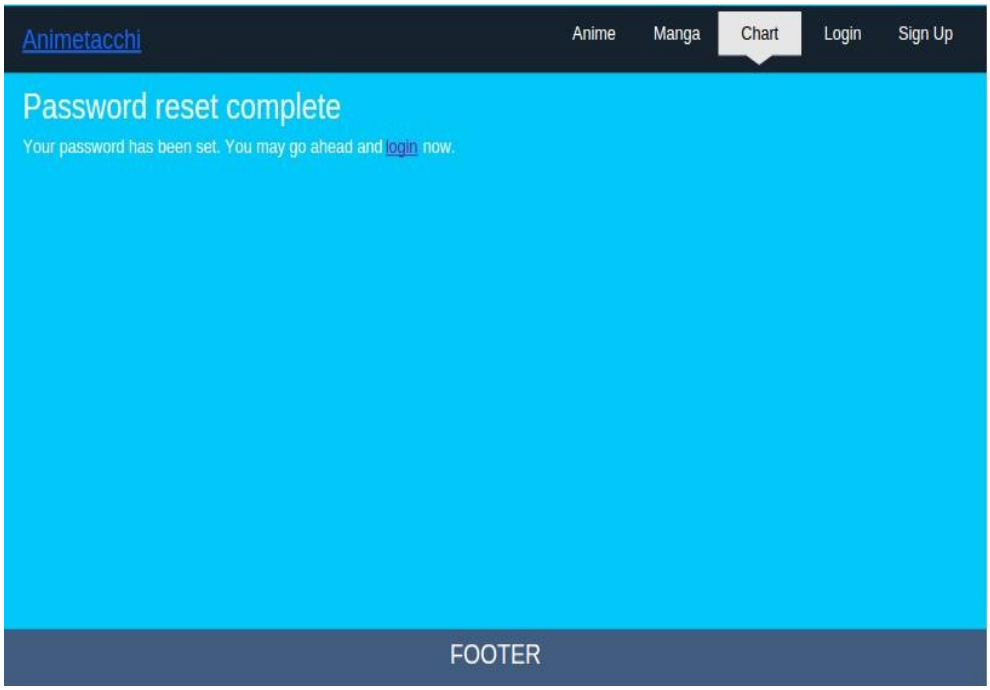


Figure 3.24 – Password Reset Complete Page views

3.3.4.5 Confirmation page

Confirmation page for users who have received an email from the system. On this page there are writings that indicate that the registration process has been completed and the user can login via the link provided. The following Confirmation page designs.

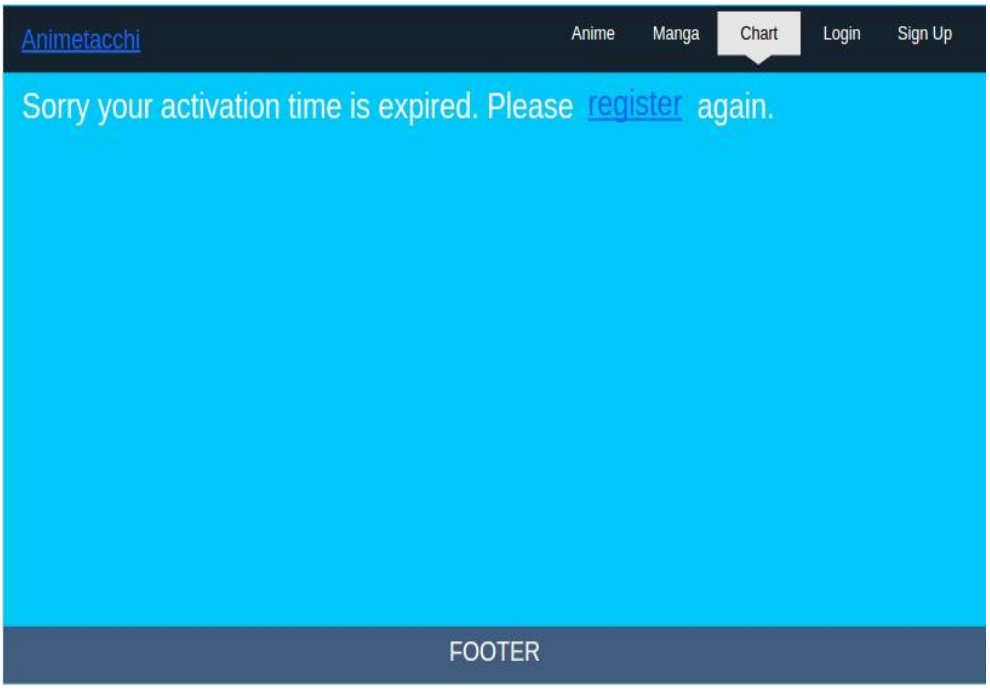


Figure 3.25 – Confirmation Page Views

3.3.4.6 Anime List Page

Anime List page is used to display a list of anime contained in this website is based on an existing genre. On this page there is a search input to perform a search Anime. Then under the search input there are some checkboxes that are used to filter based on the anime genre. In addition, on the right side there is a row of pictures anime cover and its title each successive horizontally. At the bottom of the list of anime sequence, there is a small square that acts as a button that shows the page number from the list of anime and direction of the arrow to the next page. The following page designs Anime List.

3.3.4.7 Manga List Page

Manga List page is used to display a list of anime contained in this website is based on an existing genre. The design of this page resembles the

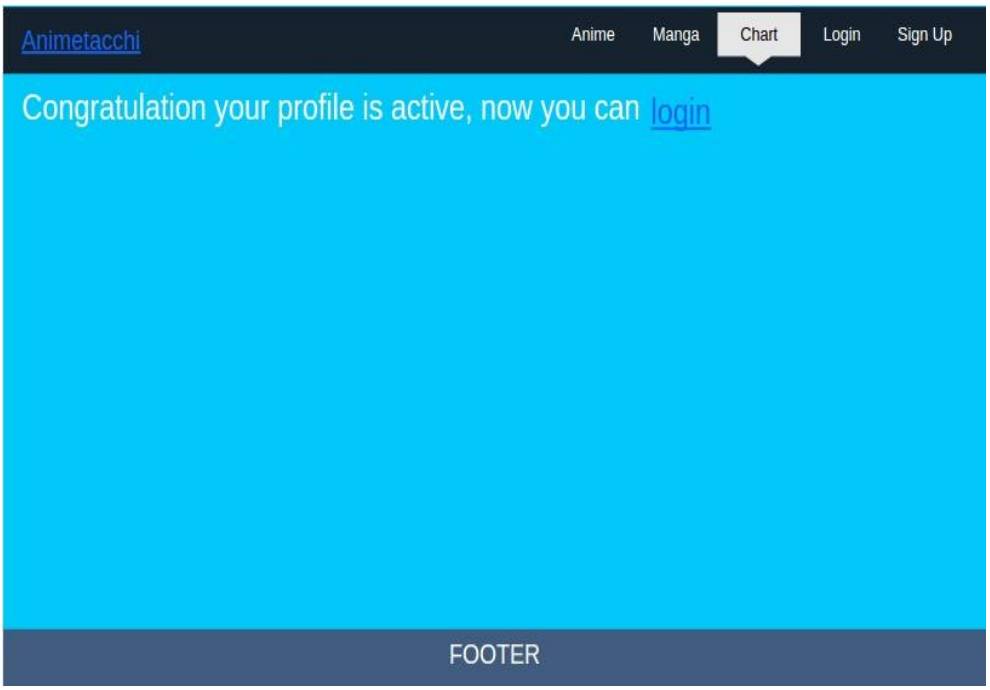


Figure 3.26 – Confirmation Expired Page Views

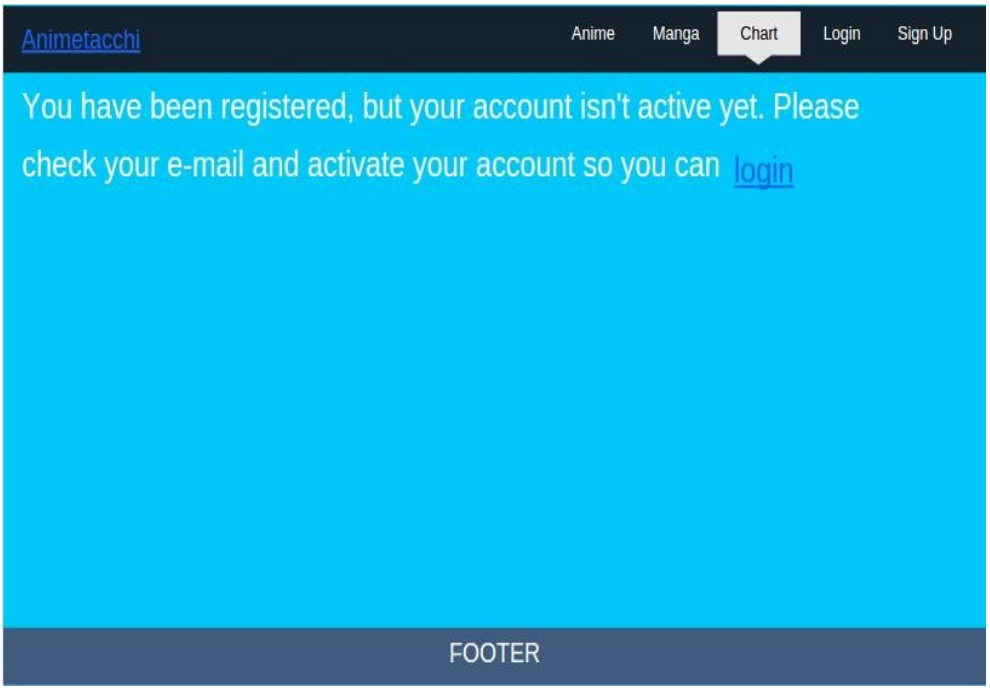


Figure 3.27 – Confirmation Check Email Page Views

Figure 3.28 – Anime List Page Views

design of the page Anime List. The following page designs Manga List.

Figure 3.29 – Manga List Page Views

3.3.4.8 Anime Details page

Details Anime page is used to display a description of the anime contained in this website has been selected by the user. On this page there is a display image that extends beyond the page to display a picture of the anime. Below the display image, there is a cover image depicting the cover from recent anime. On the side of the cover image contains the title of anime, anime synopsis, genre-genre of anime, star ratings to give judgment on anime, and three buttons. Button the first to add anime to watchlist. The watchlist selection combo box in the form of four categories, namely Bliss, Life, Agony and Hope. Then under the description of the anime, there is a list of characters that appear in the anime. Moreover, in the right part of the description of the anime, there is an area for comments, like comments, and view comments. The following Anime Detail page designs.

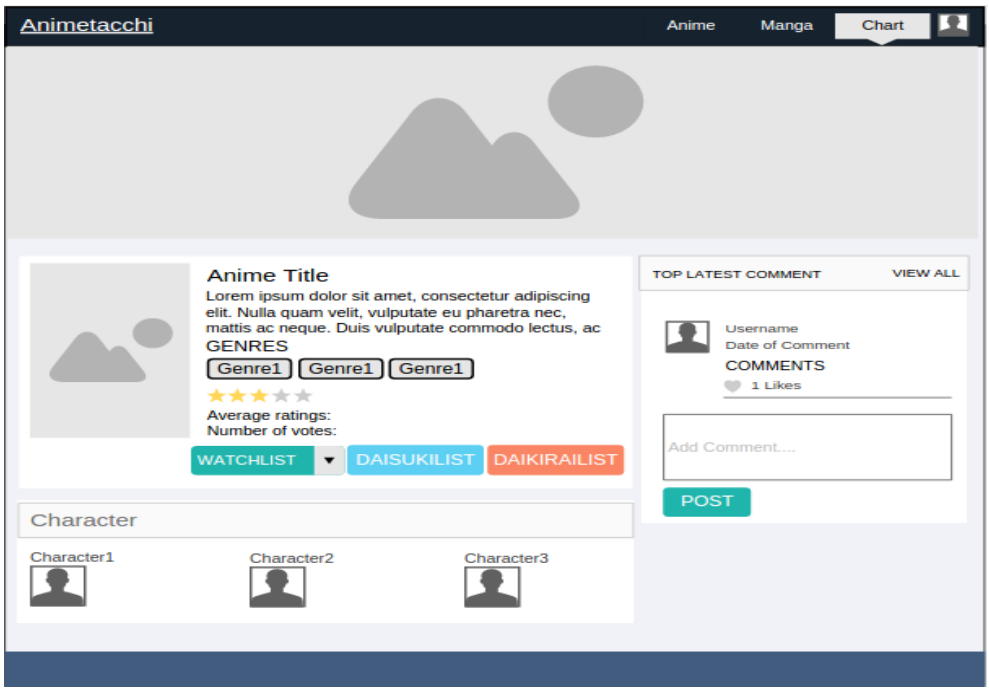


Figure 3.30 – Anime Detail Page Views

3.3.4.9 Manga Details Page

Manga Details page is used to display a description of mangoes contained in this website has been selected by the user. Content on this page is similar to the design of Anime Detail page. The following page designs Anime List.

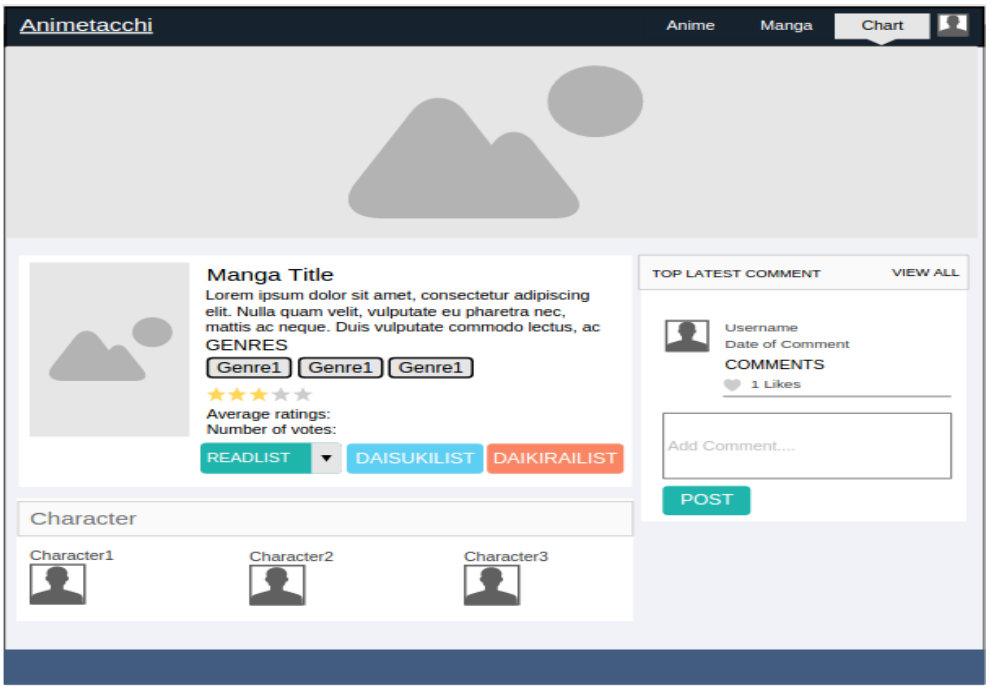


Figure 3.31 – Details Manga Page Views

3.3.4.10 Character Detail page

Character Detail page is used to display the character description contained in this website has been selected by the user. On this page there are cover image featuring a picture of your character. In addition to the cover image, there is the name of a character, a link that shows the origin of the good character of the manga and anime, character synopsis, star ratings to give judgment on the character, and two buttons. The first to add a button to daisukilist anime. The watchlist selection combo box in the form of four categories, namely Bliss, Life, Agony and Hope. Then under the description of the anime, there is a list of characters that appear in the anime. Moreover, in the right part of the description of the anime, there is an area for comments, like comments, and view comments. The following draft Character Detail page.

3.3.4.11 Voice Actor Page

Actor Voice page is used to display a description of actor voice contained in this website has been selected by the user. On this page of components and design exactly the Character Detail page designs. The following page designs Voice Actor.

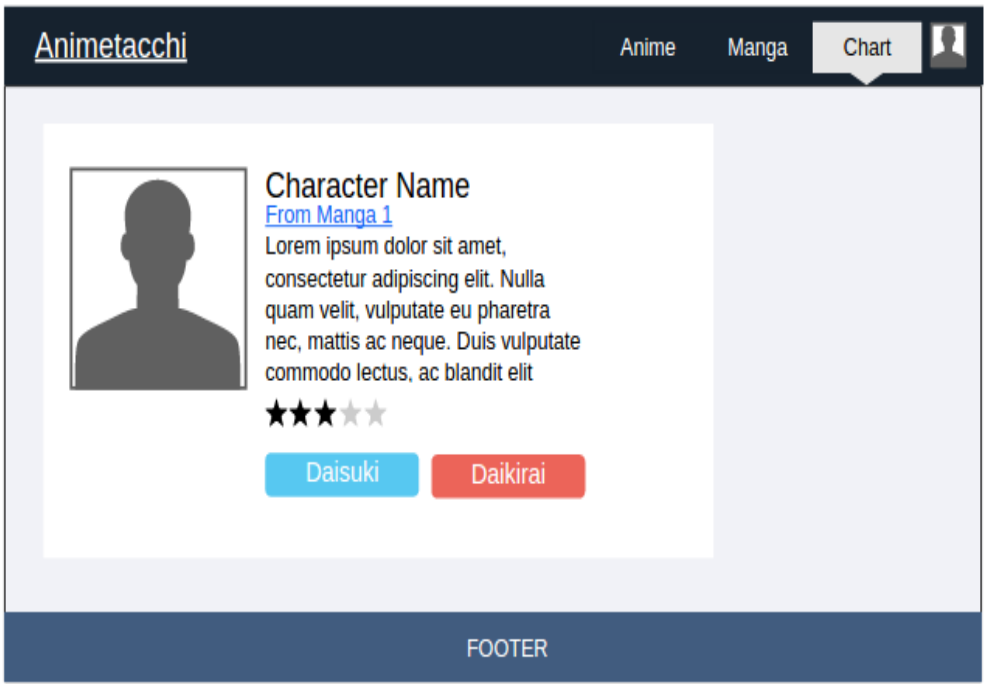


Figure 3.32 – Character Detail Page Views

Figure 3.33 – Voice Actor Page Views

3.3.4.12 User Profile Page

User Profile page is used to display the details of user activity, both adding to the list daisukilist, daikiralist, watchlist for anime and manga readlist list and timeline On this page there are two options, namely feed and library. Feed page displays a list daisukilist, daikirailist list, and timeline. Timeline shows the manga or anime that recently added watchlist / readinglist. While the library page shows the watchlist of anime and manga, written readinglist of users displayed in tabular form. Here's what the User Profile page for page feed and library.

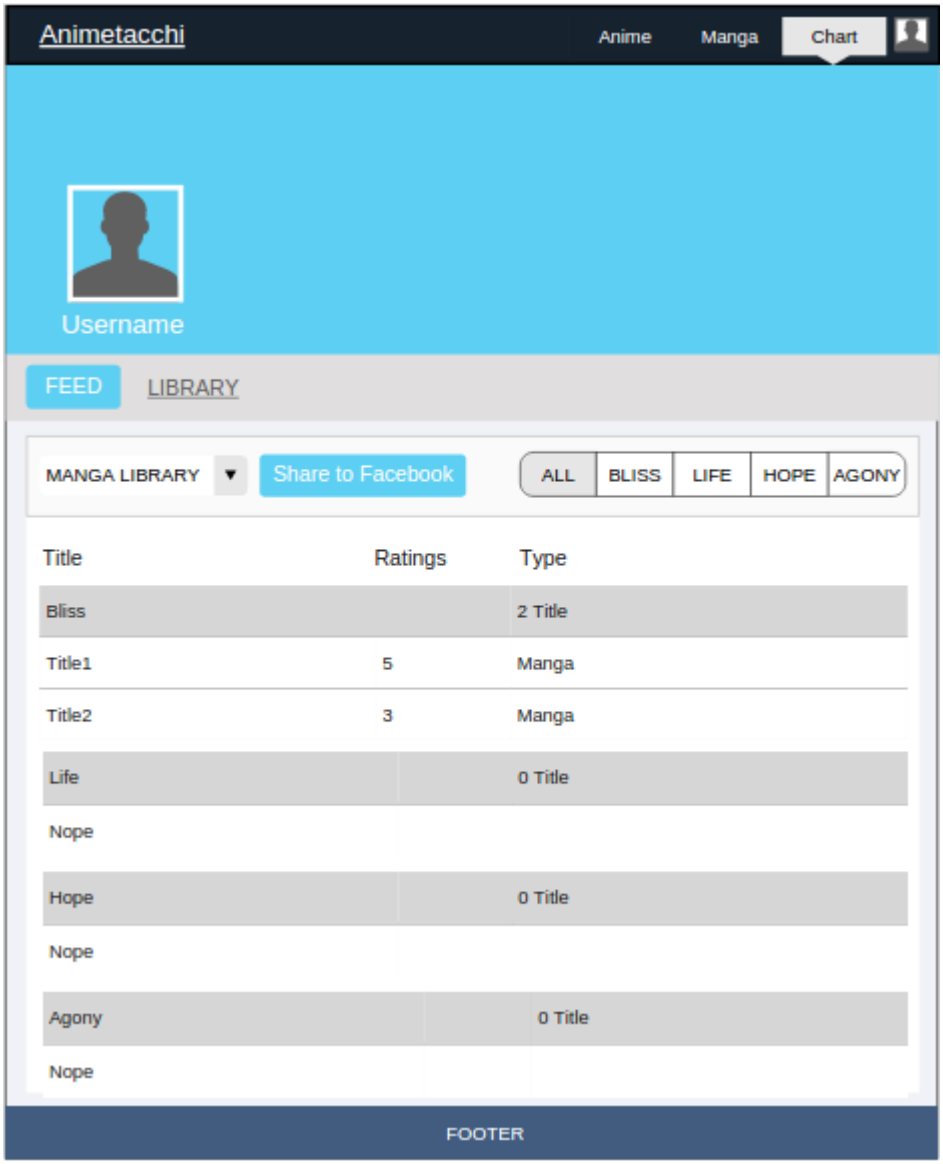


Figure 3.34 – User Profile Feed Page Views

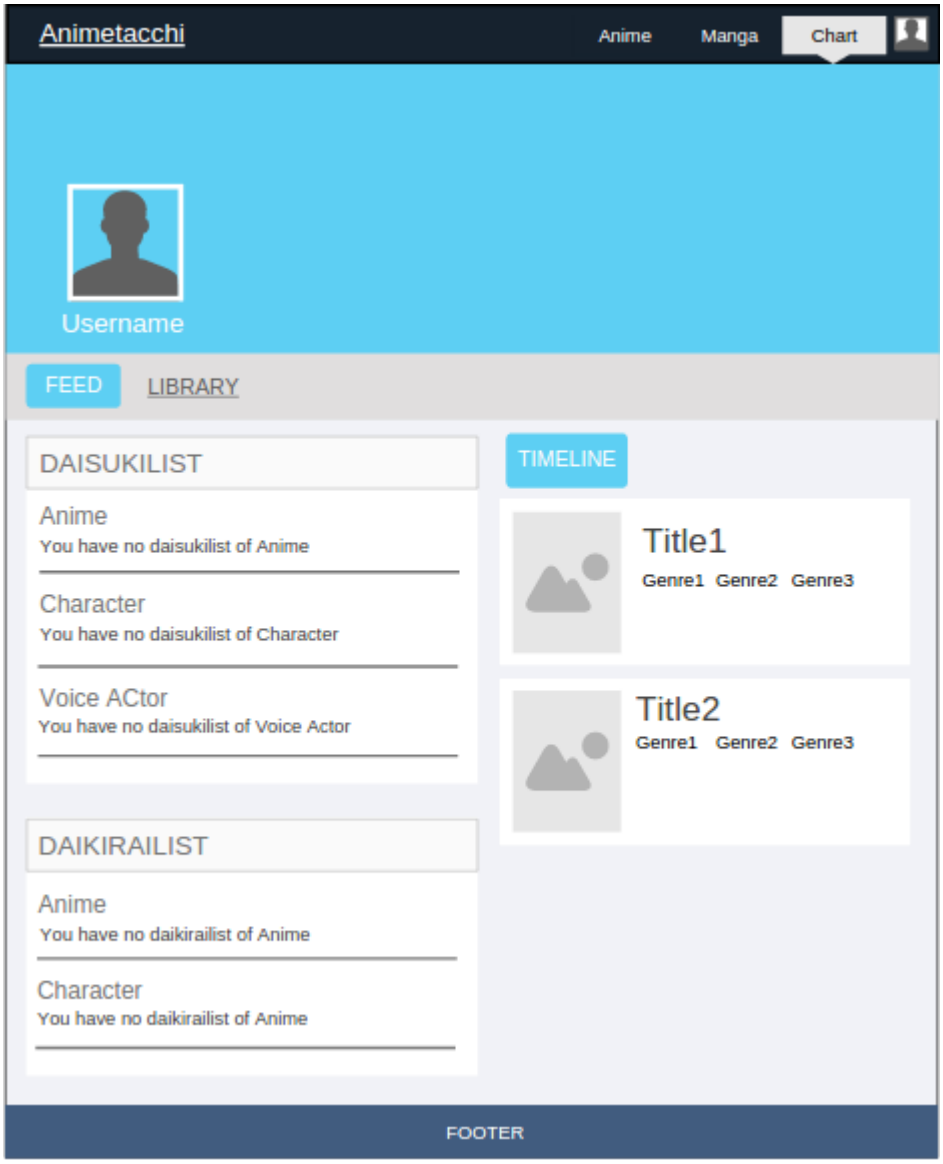


Figure 3.35 – User Profile Feed Page Views

Chapter 4

IMPLEMENTATION AND TESTING

4.1 Implementation

When the application was designed in the previous chapter, the application must be implemented. The web-based application named Animetacchi can be accessed on http://alviandjango.pythonanywhere.com. The application display consists of two parts, admin page and user page.

4.1.1 Website Implementation

Website implemantation explained about the phases after designing the application, implementation. Implementation is done by following the design that explained in Analysis and Design chapter.

4.1.1.1 Home Page Implementation

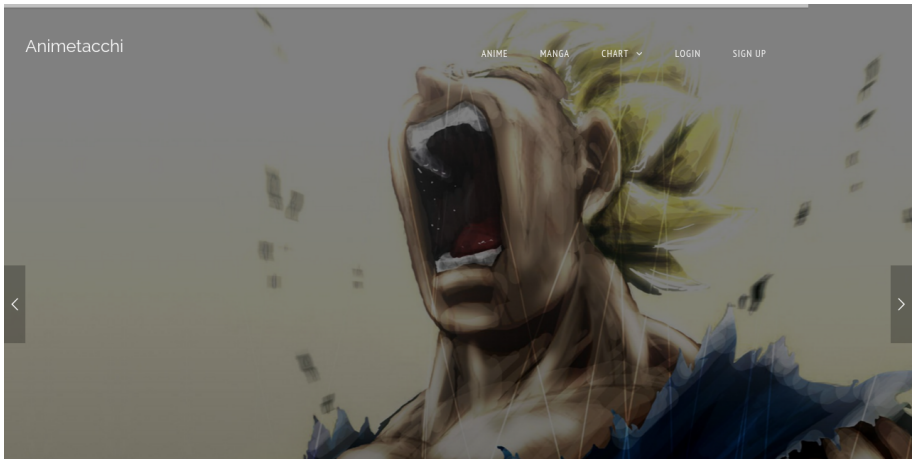


Figure 4.1 – Home Page

Home page is the first page user see when visiting the website. Member and non-member can see this page because log in is not required. The interface of Home page can be seen in Figure 4.1 and Figure 4.2.

This snippet represents the Home page:

```
1 def home(request):
2     anime_data = Anime.objects.all()[:3]
3     index = News.objects.all()
4     return render_to_response('index.html',locals(),
          context_instance=RequestContext(request))
```

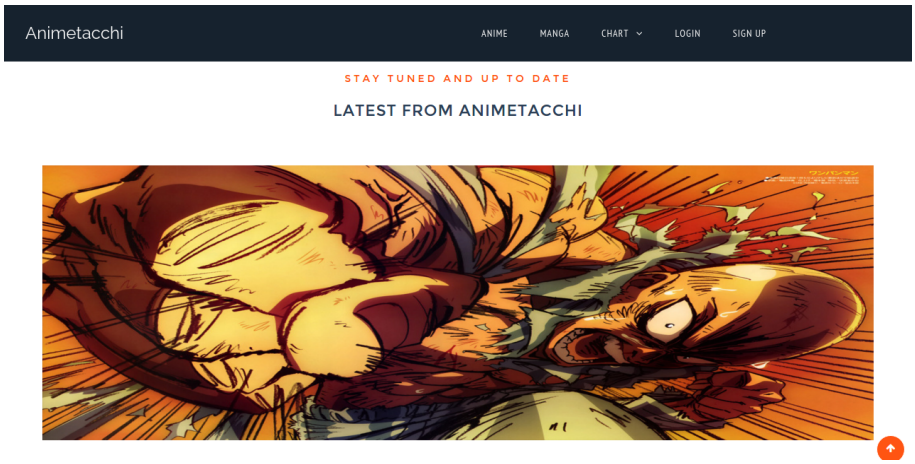


Figure 4.2 – Home Page (News Section)

4.1.1.2 Sign In Page

Sign in page is a page to login to the application. Username and password must be entered correctly, so member can successfully sign in to the application. If usermane and password are wrong, there will be an error message displayed and member must login again.

This snippet represents the Sign In page:

```
1 def signin(request):
2     if request.POST:
3         try:
4             email = request.POST.get('email')
5             password = request.POST.get('password')
6             email = User.objects.get(email=email)
7             cek_auth = auth.authenticate(username=email.
                  username, password=password)
8             auth.login(request, cek_auth)
```

```
9
10          #Doesn't Exist
11          except User.DoesNotExist:
12              message_error = email +" does not exist, please
                    register first."
13              return render_to_response('signin.html', locals
                    (), context_instance=RequestContext(request))
14
15          #Not Match Email and Password
16          except Exception, err:
17              message_match = "The email and password that you
                     entered don't match."
18              return render_to_response('signin.html', locals
                    (), context_instance=RequestContext(request))
19
20          #Active User Filter
21          members = Members.objects.get(user=request.user)
22          if request.user.is_active is False:
23              message_active = "Sorry, your ID is not active."
24              return render_to_response('signin.html',locals()
                    ,context_instance=RequestContext(request))
25          elif members.blocked is True:
26              message_active = "Sorry, your ID is blocked."
27              return render_to_response('signin.html',locals()
                    ,context_instance=RequestContext(request))
28
29          message_success = 'success'
30          member = Members.objects.get(user=request.user)
31          return render_to_response('signin.html',locals(),
                context_instance=RequestContext(request))
32
33      return render_to_response('signin.html',locals(),
            context_instance=RequestContext(request))
```

4.1.1.3 Sign Up Page

This page is intended for users who want to be a member of Animetacchi. User must fill the username, email address, and password as a requirement to create an account. The interface of Sign Up page can be seen in Figure

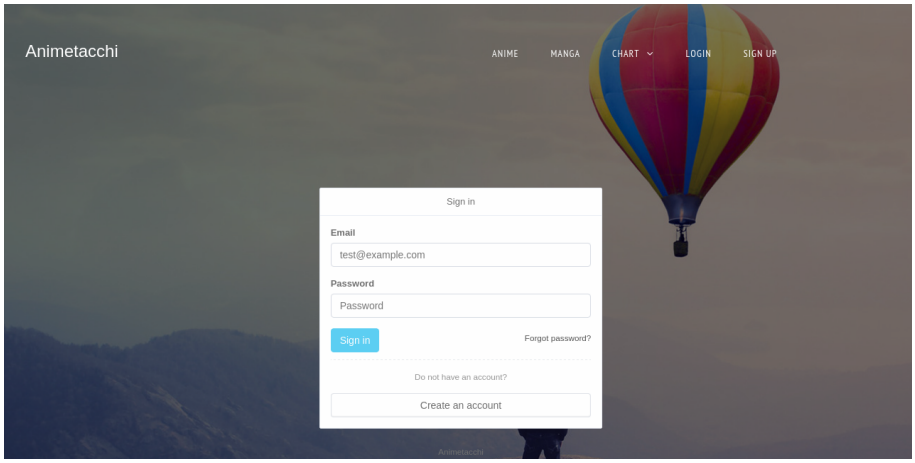


Figure 4.3 – Sign In Page

4.4.

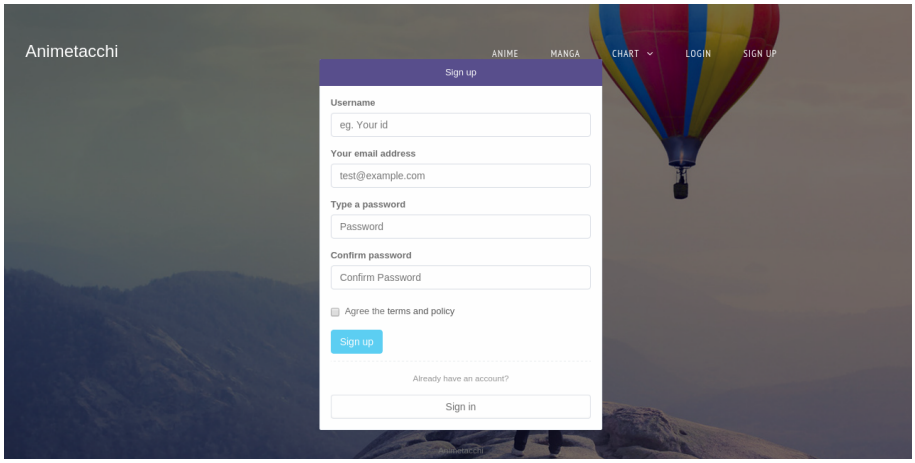


Figure 4.4 – Sign Up Page

Snippet for representing the Sign Up page:

```
1  def signup(request):
2      if request.POST.get('action') == 'signup':
3          username = request.POST.get('uname')
4          email = request.POST.get('email')
5          password = request.POST.get('passwd')
6          if User.objects.filter(username=username):
7              json_data = {'alert': 'Sorry, username ' +
                   username + ' has been used!!', 'val': False}
8              return HttpResponse(simplejson.dumps(json_data),
                       content_type="application/json")
9          if User.objects.filter(email=email):
10             json_data = {'alert': 'Sorry, email ' + email +
```

```
                ' has been used!!', 'val': False}
11              return HttpResponse(simplejson.dumps(json_data),
                    content_type="application/json")
12          usr = User.objects.create_user(username=username,
                email=email,password=password)
13          usr.is_active = False # not active until he opens
                activation link
14          usr.save()
15          member = Members()
16          member.user = usr
17          member.m_name = username
18          member.save()
19          activation_key=str(uuid.uuid4())
20          new_profile = UserProfile(user=usr, activation_key=
                activation_key)
21          new_profile.save()
22          host=request.META['HTTP_HOST']
23          email_subject = 'Account confirmation'
24          email_body = "Hey {}, thanks for signing up. To
                activate your account, click this link within \
25          48 hours http://{}/confirm/{}".format(username, host
                , activation_key)
26          send_mail(email_subject, email_body, 'be-py@alviandk
                .com',[email], fail_silently=False)
27          json_data = {'alert': 'email_subject Success!!'}
28          return HttpResponse(simplejson.dumps(json_data),
                content_type="application/json")
29      return render_to_response('signup.html',locals(),
            context_instance=RequestContext(request))
```

4.1.1.4 Forgot Password Page

When member forgot their password, member can visit the Forgot Password page by selecting the "Forgot Password?" link on Sign In page. Member must input their email to recover their password. The process of recovering the password can be seen in Figure 4.5, Figure 4.6, Figure 4.7, Figure 4.8, and Figure 4.9.

Snippet for representing the Forgot Password page:

```
1 @never_cache
```

```
2  def forgot_password(request):
3      if not request.user.is_authenticated():
4          return return_reset_password(request)
5      else:
6          return HttpResponseRedirect("/")
7
8  def return_reset_password(request):
9      return password_reset(request, template_name='User/
           forgot.html', \
10                     email_template_name='User/forgot_email.
                           html', \
11                     post_reset_redirect='/password_reset/done/
                           ',
12                     )
13
14 def cust_password_reset_done(request):
15     if not request.user.is_authenticated():
16         return password_reset_done(request,  template_name='
               User/password_reset_done.html')
17     else:
18         return HttpResponseRedirect("/")
19
20
21 @never_cache
22 def cust_password_reset_confirm(request, uidb36=None, token=
       None):
23     if not request.user.is_authenticated():
24         return password_reset_confirm(request, uidb36=uidb36
               , token=token, \
25         template_name='User/password_reset_confirm.html', \
26         post_reset_redirect='/reset/done/')
27     else:
28         return HttpResponseRedirect("/")
29
30
31 def cust_password_reset_complete(request):
32     if not request.user.is_authenticated():
33         return password_reset_complete(request,
34         template_name='User/password_reset_complete.html')
35     else:
```

```
36          return HttpResponseRedirect("/")
```

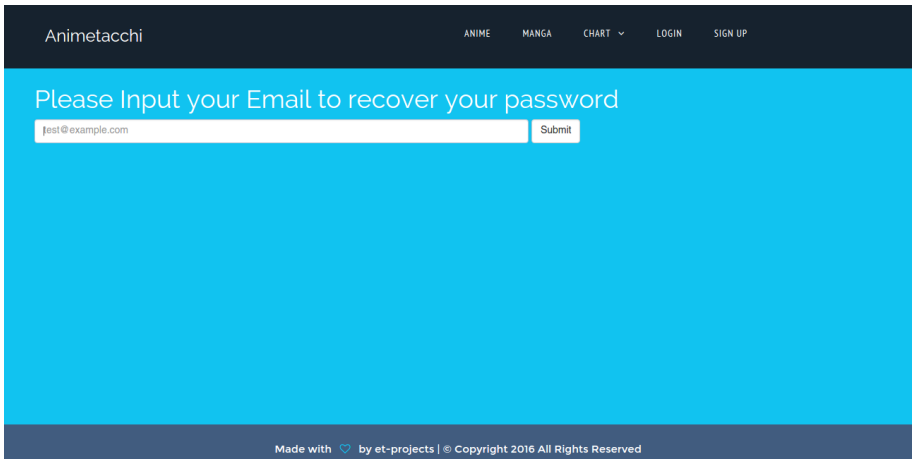


Figure 4.5 – Forgot Password Page

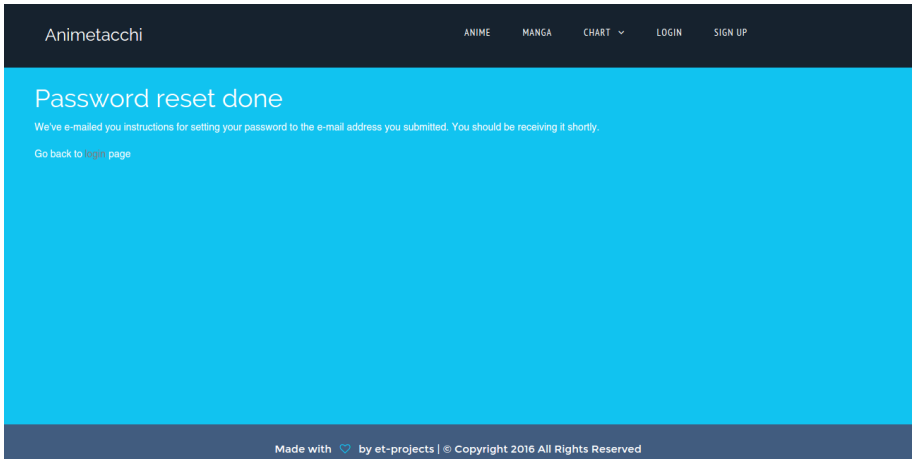


Figure 4.6 – Password Reset Done Page

4.1.1.5 Confirmation Page

This page appears when user selecting the link given to their email address for activating their new account. If user does not click the activation link within 48 hours since they register, the account will be deleted from database and user must create an account again to be a member. Confirmation page can be seen in Figure 4.10.

Figure 4.11 shows a page when user click the activation link on their email, but the activation time is expired (more than 48 hours).

When user already registered, does not mean that their account is active yet. User must check their email to activate their account, so they can be a

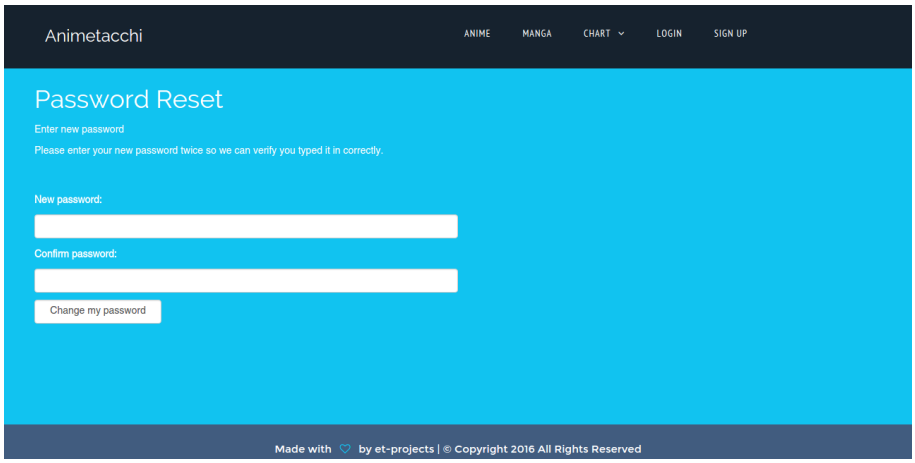


Figure 4.7 – Password Reset Page

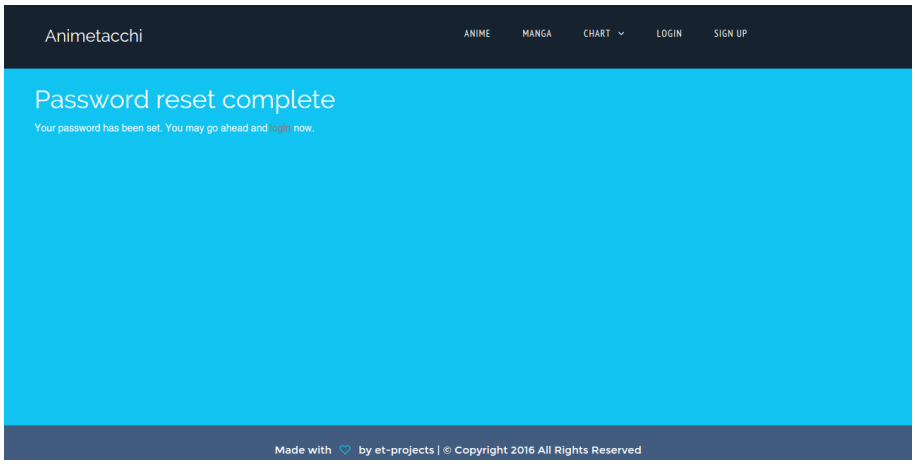


Figure 4.8 – Password Reset Complete Page

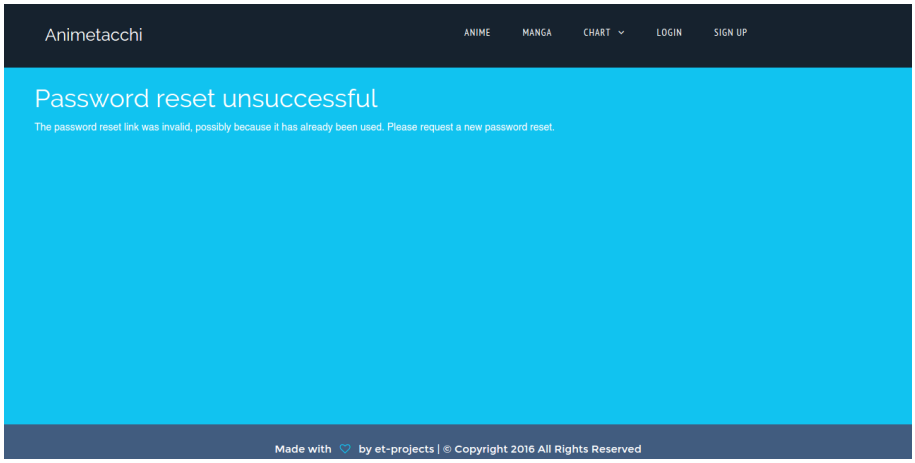


Figure 4.9 – Password Reset Unseccessful

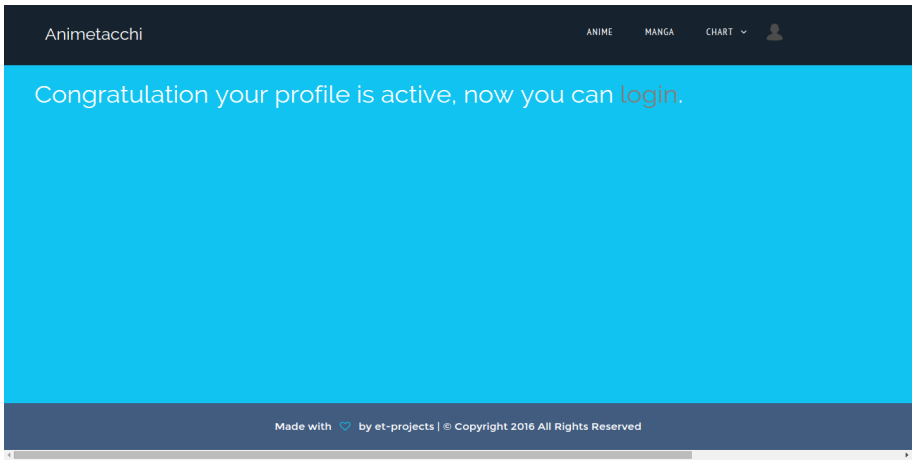


Figure 4.10 – Confirmation Page

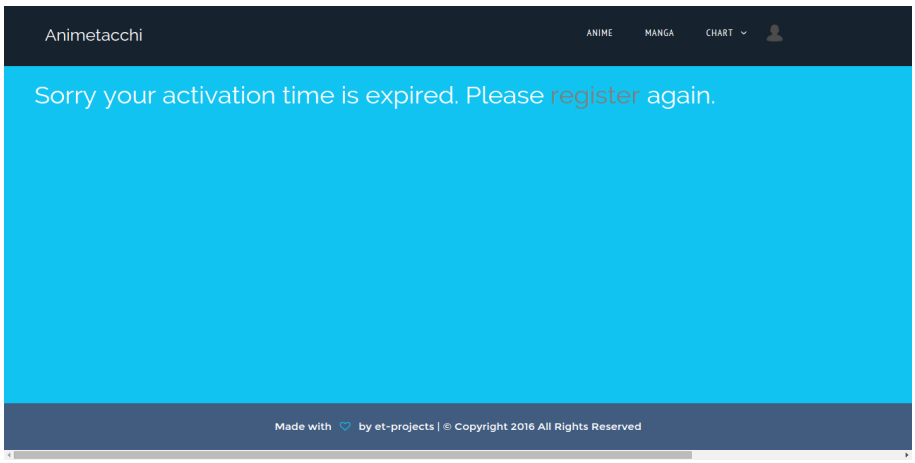


Figure 4.11 – Confirmation Expired Page

member of Animetacchi and login. Figure 4.12 represents the Check Email page.

Snippet to represent the account confirmation process:

```
1  def check_email(request):
2      return render_to_response('User/check-email.html',locals
           (),context_instance=RequestContext(request))
3
4
5  def register_confirm(request, activation_key):
6      #check if user is already logged in and if he is
           redirect him to some other url, e.g. home
7      if request.user.is_authenticated():
8          HttpResponseRedirect(reverse('home'))
9
10     # check if there is UserProfile which matches the
           activation key (if not then display 404)
11     try:
12         user_profile = UserProfile.objects.get(
               activation_key=activation_key)
13     except:
14         return HttpResponseRedirect(reverse('home'))
15
16     #check if the activation key has expired, if it hase
           then render confirm_expired.html
17     if user_profile.key_expires + timedelta(days=1) <
           timezone.now():
18         #user_profile.user.delete()
19     return render_to_response('User/confirm_expired.html',
           locals(),context_instance=RequestContext(request))
20     #if the key hasn't expired save user and set him as
           active and render some template to confirm activation
21     user = user_profile.user
22     user.is_active = True
23     user.save()
24     return render_to_response('User/confirm.html',locals(),
           context_instance=RequestContext(request))
```

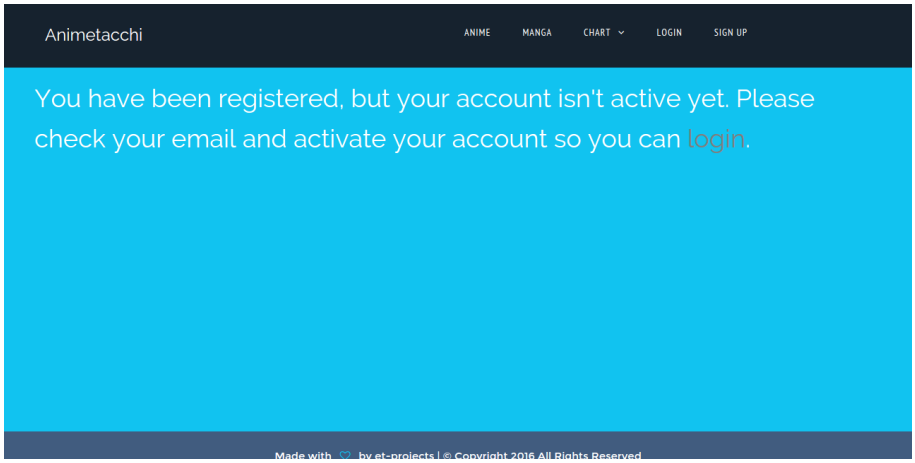


Figure 4.12 – Check Email Page

4.1.1.6 Anime List Page

Anime List page contains the list of anime available in Animetacchi. Figure 4.13 shows the interface of Anime List page.

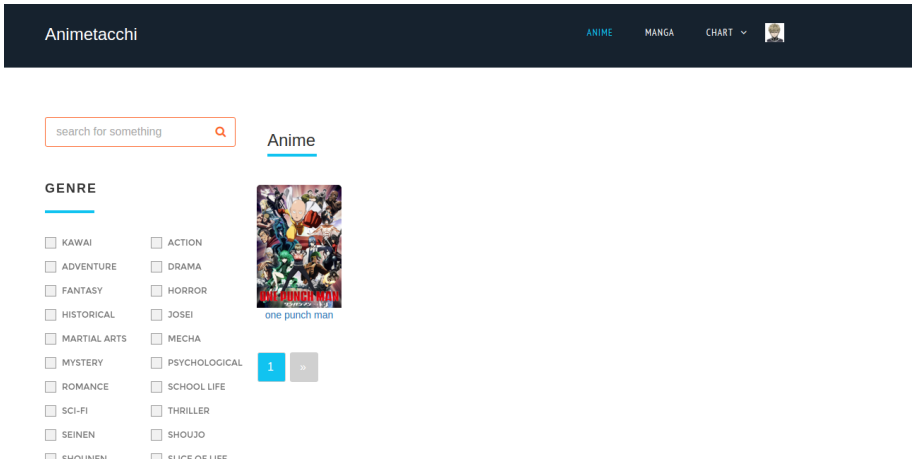


Figure 4.13 – Anime List Page

Snippet to represent the Anime List page:

```
1  def anime(request):
2      #Search Function
3      if request.POST.get('action') == 'search':
4          search_val = request.POST.get('search_value')
5          anime_data = Anime.objects.filter(a_name__contains=
               search_val).order_by('a_name')
6
7          paginator = Paginator(anime_data, 10)
8          try:
```

```
9               page = int(request.GET.get('page'))
10              if not page:
11                  page = 1
12          except Exception:
13              page = 1
14
15          anime_data = paginator.page(page).object_list
16          current_page = paginator.page(page)
17          number = current_page.number
18
19          if request.GET.get('pages_ajax') is not None :
20              pass
21
22          return render_to_response('anime_data.html',locals()
                ,context_instance=RequestContext(request))
23
24      #Sorting Function
25      if request.POST.get('action') == 'checkbox':
26          anime_check = json.loads(request.POST.get('
                anime_check'))
27          if len(anime_check) > 0:
28              anime_data = Anime.objects.filter(a_genre__in=
                    anime_check)#.distinct('a_name')
29          else:
30              anime_data = Anime.objects.all().order_by('
                    a_name')
31
32          paginator = Paginator(anime_data, 10)
33          try:
34              page = int(request.GET.get('page'))
35              if not page:
36                  page = 1
37          except Exception:
38              page = 1
39
40          anime_data = paginator.page(page).object_list
41          current_page = paginator.page(page)
42          number = current_page.number
43
44          if request.GET.get('pages_ajax') is not None :
```

```
45              pass
46
47          return render_to_response('anime_data.html',locals()
                ,context_instance=RequestContext(request))
48
49      #Default Data
50      genre = Genre.objects.all()
51      anime_data = Anime.objects.all().order_by('a_name')
52      paginator = Paginator(anime_data, 10)
53      try:
54          page = int(request.GET.get('page'))
55          if not page:
56              page = 1
57      except Exception:
58          page = 1
59
60      anime_data = paginator.page(page).object_list
61      current_page = paginator.page(page)
62      number = current_page.number
63
64      if request.GET.get('pages_ajax') is not None :
65          return render_to_response('anime_data.html', locals
                (), context_instance=RequestContext(request))
66
67      return render_to_response('anime.html',locals(),
            context_instance=RequestContext(request))
```

4.1.1.7 Manga List Page

Manga List page contains the list of manga available in Animetacchi. Figure 4.14 shows the interface of Manga List page.

The snippet to represent the Manga List page is similar to the coding of Anime List page, only the anime variable is replaced with manga.

4.1.1.8 Detail of Anime Page

This page contains the detail of each anime. On the Detail of Anime page, there is a poster of anime, the title of anime, genres, ratings, cast, and the comments about the anime. Member can add the anime to their favourite list (Daisuki list) or their dislike list (Daikirai list). Member can also to add

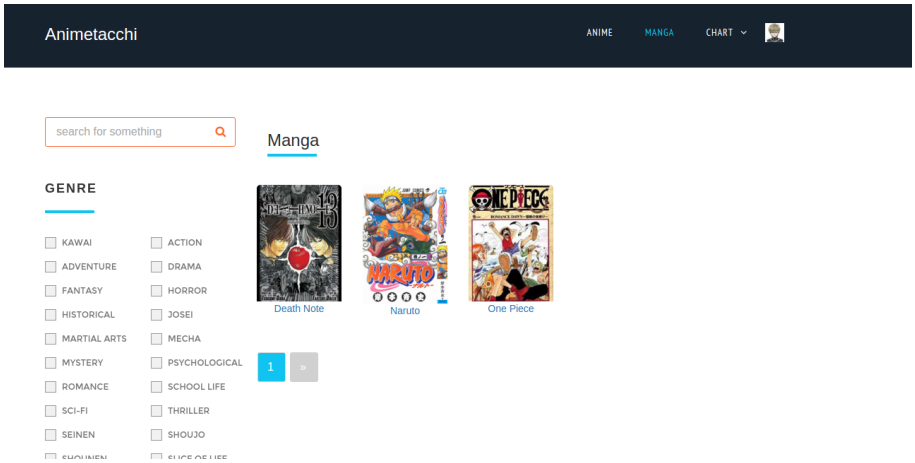


Figure 4.14 – Manga List Page

the anime to their watch list. Figure 4.15 is an interface of Detail of Anime page.

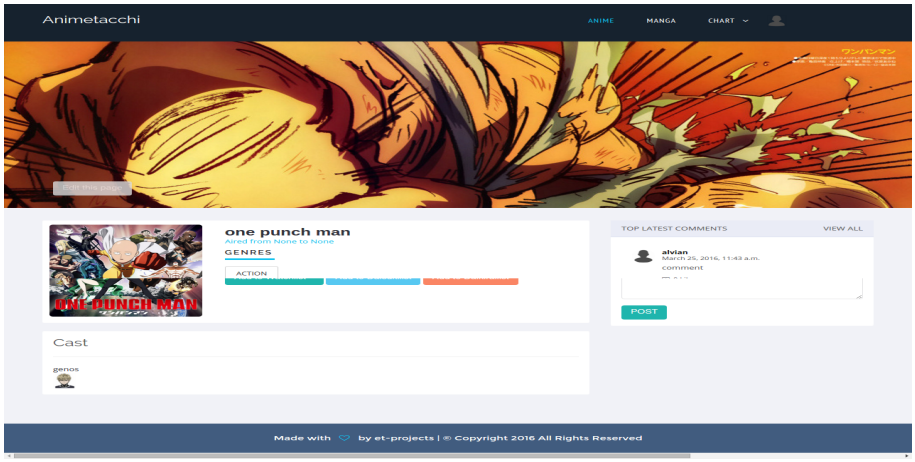


Figure 4.15 – Detail of Anime Page

Snippet to represent the Detail of Anime page:

```
1 def anime_details(request,slug):
2     anime_data = Anime.objects.get(slug=slug)
3     commentanimes = CommentAnime.objects.filter(anime=
          anime_data).annotate(likes=Count('votes')).order_by('
          -likes','-edited','-added')
4
5     #c.votes.up(request.user)
6     try:
7     members = Members.objects.get(user=request.user)
8     except:
9         pass
```

```
10      try:
11          Daisukilist.objects.get(anime=anime_data, members=
                members)
12          daisuki = True
13      except:
14          daisuki = False
15      try:
16
17          Daikirailist.objects.get(anime=anime_data, members=
                members)
18          daikirai = True
19      except:
20          daikirai = False
21      try:
22          watchlist=WatchList.objects.get(members=members,
                watchlist=anime_data)
23      except:
24          watchlist = False
25      return render_to_response('anime_details.html',locals(),
            context_instance=RequestContext(request))
```

4.1.1.9 Detail of Manga Page

This page contains the detail of each manga. On the Detail of Manga page, there is a poster of manga, the title of manga, description, genres, ratings, cast, and the comments about the anime. Member can add the anime to their favourite list (Daisuki list) or their dislike list (Daikirai list). Member can also to add the anime to their reading list. Figure 4.16 is an interface of Detail of Manga page.

The snippet to represent the Detail of Manga page is similar to the coding of the Detail of Anime page, only the anime variable is replaced with manga.

4.1.1.10 Character Detail Page

Character Detail page contains the picture and detail description about the specific character. Member can give the rating to the character and add them to Daisuki list or Daikirai list. Figure 4.17 shows the interface of Character Detail page.

The snippet to represent the Character Detail page is similar to the coding

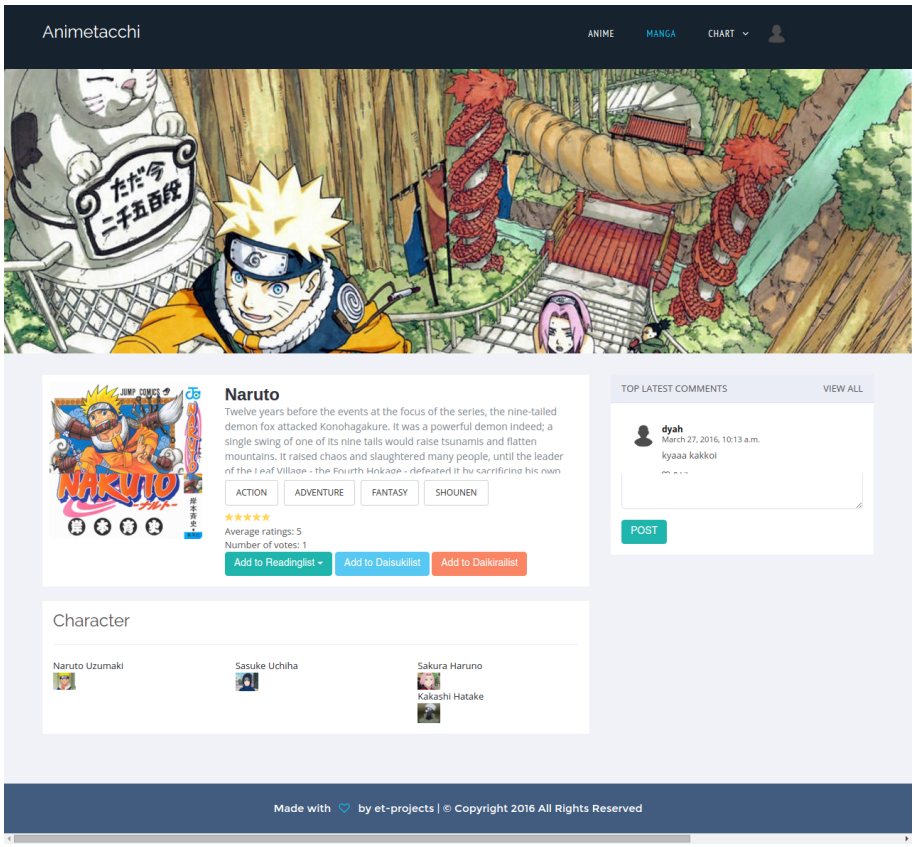


Figure 4.16 – Detail of Manga Page

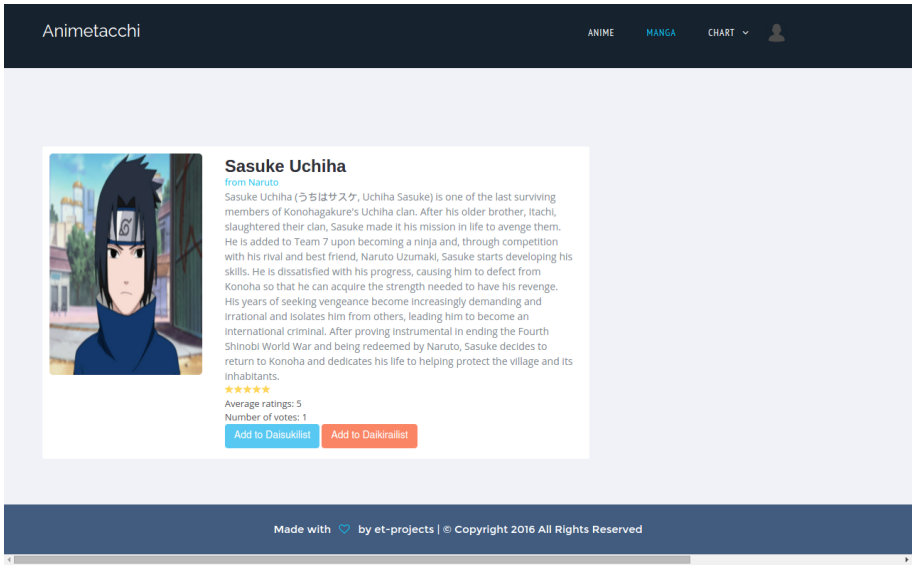


Figure 4.17 – Character Detail Page

of the Anime List page, only the anime variable is replaced with character.

4.1.1.11 Voice Actor Page

Voice Actor page contains name of the character dubber, their photo, and the description about the actor. Member can give a rating to the actor and add them to Daisuki list and Daikirai list. Figure 4.18 shows the interface of Voie Actor page.

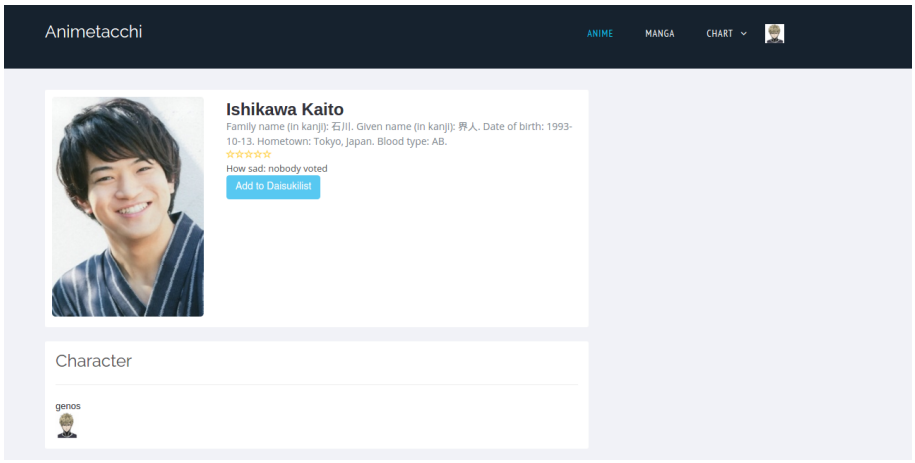


Figure 4.18 – Voice Actor Page

The snippet to represent the Voice Actor page is similar to the coding of the Anime List page, only the anime variable is replaced with voice actor.

4.1.1.12 User Page

User page is a page that gives information about member of Animetacchi. Figure 4.19 is a capture of User page.

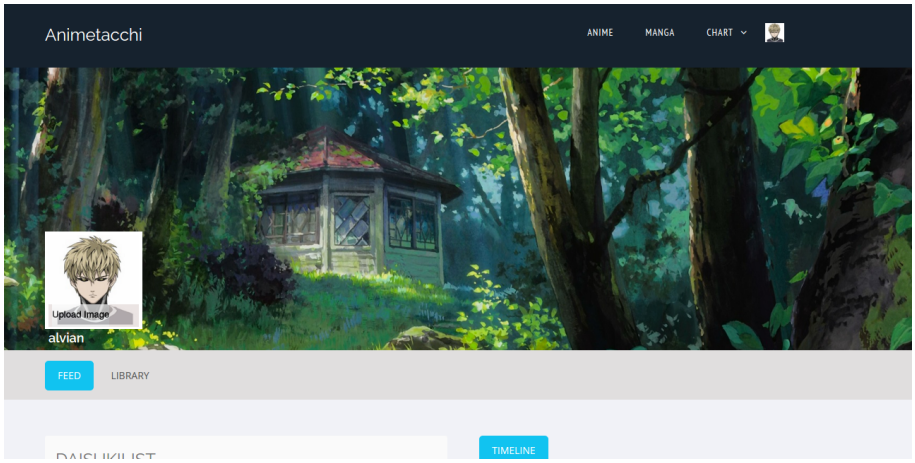


Figure 4.19 – User Page

Figure 4.20 below is a capture of User Profile Feed page.

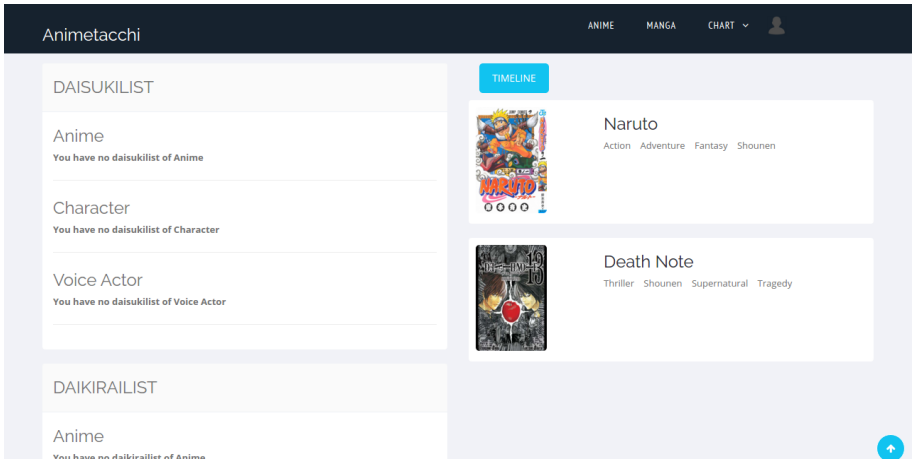


Figure 4.20 – User Profile Feed Page

Figure 4.21 shows a capture of Anime Library in User Profile.

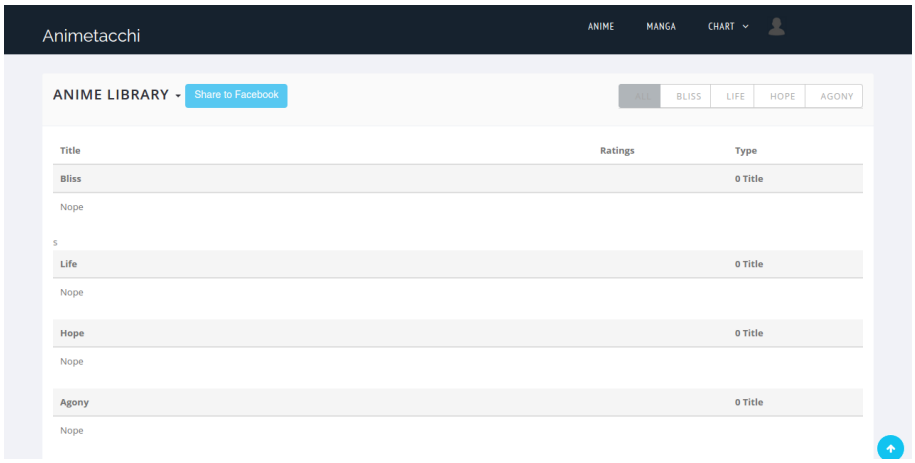


Figure 4.21 – User Profile for Anime Library

Figure 4.22 shows a capture of Manga Library in User Profile.
Snippet to represent User Profile and Library page:

```
1  def dashboard(request, username):
2      if request.POST.get('action') == 'saveAbout':
3          countData = About.objects.all().count()
4          aboutField = request.POST.get('about')
5          seq_about = request.POST.get('seq_about')
6          memberId = request.POST.get('member')
7          member = Members.objects.get(id=memberId)
8          aboutData = About()
9          if seq_about is not None:
```

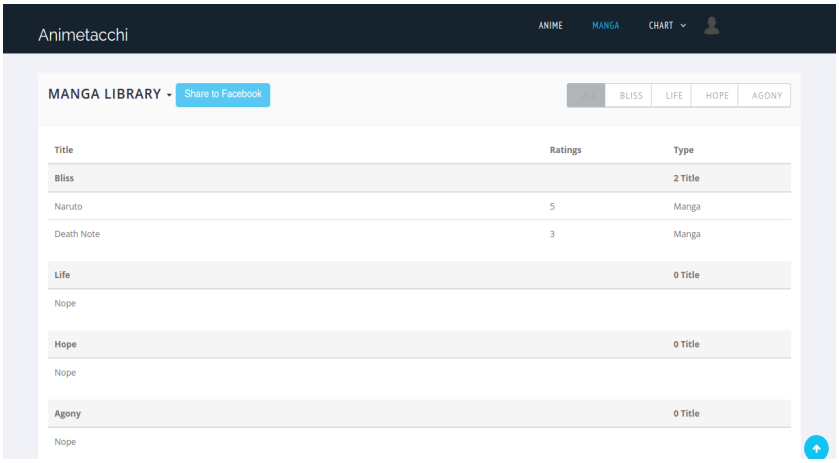


Figure 4.22 – User Profile for Manga Library

```
10              aboutData = About.objects.get(seq_about=
                    seq_about)
11              aboutData.member = member
12              aboutData.about = aboutField
13              aboutData.seq_about = seq_about
14              aboutData.save()
15              return render_to_response('User/dashboard.html',
                    locals(),context_instance=RequestContext(
                    request))
16          aboutData.member = member
17          aboutData.about = aboutField
18          aboutData.seq_about = countData
19          aboutData.save()
20          return render_to_response('User/dashboard.html',
                locals(),context_instance=RequestContext(request)
                )
21
22      if request.user.is_authenticated():
23          member = Members.objects.get(user=request.user)
24          usr = User.objects.get(username=username)
25          if request.user.username != username:
26              return HttpResponseRedirect(reverse('users',
                    kwargs={'username': username}))
27          about_data = About.objects.filter(member=member.id)
28          return render_to_response('User/dashboard.html',
                locals(),context_instance=RequestContext(request)
                )
```

```
29      return HttpResponseRedirect(reverse('home'))
30  def library(request,username):
31          usr = User.objects.get(username=username)
32          member = Members.objects.get(user=usr)
33          bliss = WatchList.objects.filter(members=member,
                status='Bliss')
34          agony = WatchList.objects.filter(members=member,
                status='Agony')
35          hope = WatchList.objects.filter(members=member,
                status='Hope')
36          life = WatchList.objects.filter(members=member,
                status='Life')
37          host=request.META['HTTP_HOST']
38          about_data = About.objects.filter(member=member.id)
39          return render_to_response('User/library.html',locals
                (),context_instance=RequestContext(request))
```

4.1.2 Database Implementation

Database implementation using Django web framework is helped by the script in acchi/models.py. First, the database needs to be set on a file animetacchi/settings.py. The following pieces of coding to database settings:

```
1  DATABASES = {
2          'default': {
3                      'ENGINE': 'django.db.backends.sqlite3',
4                      'NAME': os.path.join(PROJECT_PATH, 'db.
                           sqlite3'),
5                  }
6  }
```

Table which will be created on the database represented by the names of the class defined in the file models.py. Fields that are on the table, defined on each attribute class. The following snippet of models.py is used as database tables and fields:

```
1  class Members(models.Model):
2      user = models.OneToOneField(User,related_name='members')
3      m_name = models.CharField(max_length=50, null=True)
4      m_cover = models.ImageField(upload_to='MembersCover',
           null=True, blank=True)
```

```
5       m_picture = models.ImageField(upload_to='MembersPicture'
            , null=True, blank=True)
6       m_bio = models.TextField(null=True, blank=True)
7       m_tagline = models.CharField(max_length=50, null=True)
8       blocked = models.BooleanField(default=False)
9       seq_members = models.IntegerField(null=True)
10      def __unicode__(self):
11          return unicode(self.user.username)
12  class Genre(models.Model):
13      genre_type = models.CharField(max_length=30, null=True )
14      def __unicode__(self):
15          return unicode(self.genre_type)
16  class Anime(models.Model):
17      a_name = models.CharField(max_length=50, null=True)
18      a_genre = models.ManyToManyField(Genre)
19      a_cover = models.ImageField(upload_to='AnimeCover', null
            =True, blank=True)
20      a_displaypic = models.ImageField(upload_to='AnimePicture
            ', null=True, blank=True)
21      a_synopsys = models.TextField(null=True, blank=True)
22      a_airedstart = models.DateField(null=True, blank=True)
23      a_airedend = models.DateField(null=True, blank=True)
24      save_added = models.DateTimeField(auto_now=True)
25      slug = models.SlugField(null=True, blank=True)
26      seq_anime = models.IntegerField(null=True, blank=True)
27      def save(self, *args, **kwargs):
28          self.slug = slugify(self.a_name)
29          super(Anime, self).save(*args, **kwargs)
30      def __unicode__(self):
31          return unicode(self.a_name)
32  class Manga(models.Model):
33      name = models.CharField(max_length=50, null=True)
34      genre = models.ManyToManyField(Genre)
35      cover = models.ImageField(upload_to='MangaCover', null=
            True, blank=True)
36      displaypic = models.ImageField(upload_to='MangaPicture',
             null=True, blank=True)
37      synopsys = models.TextField(null=True, blank=True)
38      airedstart = models.DateField(null=True, blank=True)
39      airedend = models.DateField(null=True, blank=True)
```

```
40      save_added = models.DateTimeField(auto_now=True)
41      slug = models.SlugField(null=True, blank=True)
42      seq_manga = models.IntegerField(null=True, blank=True)
43      def save(self, *args, **kwargs):
44          self.slug = slugify(self.name)
45          super(Manga, self).save(*args, **kwargs)
46      def __unicode__(self):
47         return unicode(self.name)
48  class WatchList(models.Model):
49      members = models.ForeignKey('Members', null=True, blank=
            True)
50      watchlist = models.ForeignKey('Anime', null=True, blank=
            True)
51      status = models.CharField(max_length=10, choices=STATUS,
             null=True)
52      seq_watchlist = models.IntegerField(null=True)
53      def __unicode__(self):
54          return unicode(self.members)
55  class ReadingList(models.Model):
56      members = models.ForeignKey('Members', null=True, blank=
            True)
57      readinglist = models.ForeignKey('Manga', null=True,
            blank=True)
58      status = models.CharField(max_length=10, choices=STATUS,
             null=True)
59      seq_watchlist = models.IntegerField(null=True)
60      def __unicode__(self):
61          return unicode(self.members)
62  class Character(models.Model):
63      name = models.CharField(max_length=50, null=True)
64      picture = models.ImageField(upload_to='CharacterPicture'
            , null=True, blank=True)
65      anime = models.ForeignKey('Anime', null=True, blank=True
            )
66      synopsys = models.TextField(null=True, blank=True)
67      def __unicode__(self):
68          return (self.name)
69  class CharacterManga(models.Model):
70      name = models.CharField(max_length=50, null=True)
```

```
71      picture = models.ImageField(upload_to='CharacterPicture'
            , null=True, blank=True)
72      manga = models.ForeignKey('Manga', null=True, blank=True
            )
73      synopsys = models.TextField(null=True, blank=True)
74      def __unicode__(self):
75          return (self.name)
76  class VoiceCharacter(models.Model):
77      name = models.CharField(max_length=50, null=True)
78      picture = models.ImageField(upload_to='VoiceActorPicture
            ', null=True, blank=True)
79      character = models.ManyToManyField('Character', null=
            True, blank=True)
80      synopsys = models.TextField(null=True, blank=True)
81      def __unicode__(self):
82          return (self.name)
83  class Daisukilist(models.Model):
84      members = models.ForeignKey('Members')
85      anime = models.ManyToManyField(Anime,null=True, blank=
            True )
86      manga = models.ManyToManyField(Manga,null=True, blank=
            True )
87      character = models.ManyToManyField(Character,null=True,
            blank=True)
88      character_manga = models.ManyToManyField(CharacterManga,
            null=True, blank=True)
89      voice_character = models.ManyToManyField(VoiceCharacter,
            null=True, blank=True)
90      def __unicode__(self):
91          return ('{}-daisuki'.format(self.members))
92  class Daikirailist(models.Model):
93      members = models.ForeignKey('Members')
94      anime = models.ManyToManyField(Anime,null=True, blank=
            True )
95      manga = models.ManyToManyField(Manga,null=True, blank=
            True )
96      character = models.ManyToManyField(Character,null=True,
            blank=True)
97      character_manga = models.ManyToManyField(CharacterManga,
            null=True, blank=True)
```

```
98      def __unicode__(self):
99          return ('{}-daikirai'.format(self.members))
100 class Chart(models.Model):
101     season = models.CharField(max_length=10, choices=SEASON,
            null=True)
102     anime = models.ManyToManyField(Anime,null=True, blank=
            True )
103     year = models.IntegerField(null=True, blank=True)
104     class Meta:
105         ordering = ['year']
106     def __unicode__(self):
107         return ('{}-{}'.format(self.season, self.year))
108 class CommentAnime(models.Model):
109     comment = models.TextField()
110     user = models.ForeignKey('Members', null=True, blank=
            True)
111     anime = models.ForeignKey('Anime', null=True, blank=True
            )
112     added = models.DateTimeField(auto_now=True,null=True,
            blank=True)
113     edited = models.DateTimeField(null=True, blank=True)
114     votes = VotableManager()
115     class Meta:
116         ordering = ['-edited', '-added']
117     def __unicode__(self):
118         return ('{}-{}'.format(self.user, self.comment[:20])
                )
119 class CommentManga(models.Model):
120     comment = models.TextField()
121     user = models.ForeignKey('Members', null=True, blank=
            True)
122     manga = models.ForeignKey('Manga', null=True, blank=True
            )
123     added = models.DateTimeField(auto_now=True,null=True,
            blank=True)
124     edited = models.DateTimeField(null=True, blank=True)
125     votes = VotableManager()
126     class Meta:
127         ordering = ['-edited', '-added']
128     def __unicode__(self):
```

```
129         return ('{}-{}'.format(self.user, self.comment[:20])
                )
130 class UserProfile(models.Model):
131     user = models.OneToOneField(User)
132     activation_key = models.CharField(max_length=40, blank=
            True)
133     key_expires = models.DateTimeField(default= datetime.now
            )
134     def __str__(self):
135         return self.user.username
136 class RequestAnime(models.Model):
137     content = models.TextField()
138     user = models.ForeignKey('Members', null=True, blank=
            True)
139     anime = models.ForeignKey('Anime', null=True, blank=True
            )
140     added = models.DateTimeField(auto_now=True,null=True,
            blank=True)
141     class Meta:
142         ordering = ['-added']
143     def __unicode__(self):
144         return ('{}-{}'.format(self.user, self.added))
145 ratings.register(Anime, score_range=(1, 5), form_class=
    VoteForm)
146 ratings.register(Manga, score_range=(1, 5), form_class=
    VoteForm)
147 ratings.register(Character, score_range=(1, 5), form_class=
    VoteForm)
148 ratings.register(CharacterManga, score_range=(1, 5),
    form_class=VoteForm)
149 ratings.register(VoiceCharacter, score_range=(1, 5),
    form_class=VoteForm)
```

4.2 Testing

After the application is implemented, the application must be tested before eventually used. In the testing phase, the author conducted a Black Box Testing and the results were described in the following Table

Table 4.1 – Black Box Testing Results

No	Test Scenario	Test Case	Expected Results	Testing Results	Conclusion
1	Empty all fields log in data (username and password). Then select the Login button	Username: (Empty) Password: (Empty)	The system will deny access to log in and display the alert message to log in first.	As Expected	Valid
2	Fill the admin username correctly but the password incorrectly. Then select the Log in button	Username: admin01 (True) Password: 4321 (False)	System will deny the log in process and display the alert message to check the username and password entered.	As Expected	Valid
3	Fill the admin username and password correctly. Then select the Log in button	Username: admin01 (True) Password: 1234 (True)	Successfully log in and go directly to Admin Page	As Expected	Valid

No	Test Scenario	Test Case	Expected Results	Testing Results	Conclusion
4	Select a Models and select an Add button	All fields are empty	System will deny to add the object and display an alert message to correct the errors	As Expected	Valid
5	Empty all the fields, then select the Save button	All fields are filled correctly	The system will add and save the new Models object	As Expected	Valid
6	Select a Models object and select Change. Edit the data then select the Save button	Test the Save button after editing	The system removes the selected object from database	As Expected	Valid
7	Empty all fields login data (email and password), then select the Login button	Email and Password: (Empty)	The System will deny access to log on and display the alert message	As Expected	Valid

No	Test Scenario	Test Case	Expected Results	Testing Results	Conclusion
8	Fill the user email correcctly but the password incorrectly, then select the Login button	Email: alviandk@gmail.com (True) Password: 34y12 (False)	The System will deny access to log on and display the alert message	As Expected	Valid
9	Fill the user email and password correctly, then select the Login button.	Email: alviandk@gmail.com (True) Password: (True)	Successfully log in and go drectly to the home page	As Expected	Valid
10	Select Anime or Character or Voice Actor, then select the “Add to Daisukilist”	Make favourite list	The Anime or Character or Voice Actor successfully added to favourite list and the system display the “Added to Daisukilist” notification	As Expected	Valid

No	Test Scenario	Test Case	Expected Results	Testing Results	Conclusion
11	Select Anime or Character, then select the "Add to Daikirailist" button	Make dislike list	The Anime or Character successfully added to dislike list and the system display "Added to your daikirailist" notification	As Expected	Valid
12	Select Anime then select the "Add to Watchlist" button	Add an Anime to Watch list	The Anime successfully added to watch list and display the notification	As Expected	Valid
13	Select Manga, then select the "Add to Readinglist" button	Add a Manga to Reading list	The Manga successfully added to reading list and display the notification	As Expected	Valid
14	Select the user icon then select the Dashboard menu	See own profile	The user dashboard page will appear and user can see their profile	As Expected	Valid
15	Select the other user's name	See other user profile	The other user profile appears	As Expected	Valid

No	Test Scenario	Test Case	Expected Results	Testing Results	Conclusion
16	Fill the Seacrh comments box and type the comments user want to search, then select the search icon	Seacrh comments	The results of searched comments from other users appear	As Expected	Valid
17	Select “View all” then select the “Sort by likes” button or “Sort by time” button	Sort comments	Comments will appear by sort category	As Expected	Valid
18	Select the love button in other user’s comment	Likes the comments	Successfully like the comment and the ammount of likes will increase	As Expected	Valid
19	Type a comment in comment box, then select the “POST” button	Give comment on any entity	Successfully post a comment and the comment appears in the bottom of the comment list	As Expected	Valid

No	Test Scenario	Test Case	Expected Results	Testing Results	Conclusion
20	Select Manga or Anime then select the stars	Rate the entity.	Successfully rate the entity and the “Rated” notification appears	As Expected	Valid
21	Select the user icon and select the dashboard menu. Select the LIBRARY button, choose the library (Anime or Manga), then select the “Share to Facebook” button	Share the library to user’s Facebook	The Facebook Authorization page will appears and after selecting the “Post to Facebook” button, the library will be shared	As Expected	Valid

Chapter 5

CONCLUSION

5.1 Conclusion

Website Animetacchi has been successfully implemented and successfully deployed in Alvian django.pythonanywheere.com. On this website, users can see the anime, manga, characters, and voice. Users can be identified as a registered user, unregistered, and admin. Registered users can create watch lists and reading as well as provide feedback to the site, as well as interact with other users.

The main tools used to implement this website is the Django web framework that uses the Python programming language, and Ajax on jQuery.

5.2 Future Work

For further development, there are some suggestions that can be applied by adding some functionality:

1. Users can import a list of anime and manga from myanimelist sites.
2. Users can leave a request to make changes to the detail of anime and manga.
3. The site can be fitted advertisment so that website owners can generate revenue.

Bibliography

[1] Somayeh Azizi. *Modeling UML2 Activity Diagram by Using Graph Transformation Systems and Abstract State Machine*. 2011.

[2] Tim Berners-Lee, Robert Cailiau, Ari Luotonen, Henrik Frystyk Nielsen, and Arthur Secret. The world-wide web. *Communications of The ACM*, 37(8), August 1994.

[3] Rosa Bjork Blondal. Anime as an adaptation. *Thesis for a BA Degree in Japanese Language and Culture*, January 2014.

[4] R. E. Brenner. *Understanding Manga and Anime*. Greenwood Publishing Group, Inc, Westport, CT, 2007.

[5] David R. Brooks. *An Introduction to HTML and JavaScript: for Scientist and Engineers*. Springer-Verlag London Limited, 2007.

[6] D. Casvallaro. *Anime and The Art of Adaptation*. McFarland & Company, Inc, North California, 2010.

[7] Neil Cohn. Japanese visual language: The structure of manga. 2007.

[8] Frederick Constantianus and Bernard Renaldy Suteja. Analisa dan desain sistem bimbingan tugas akhir berbasis web dengan studi kasus fakultas teknologi informasi. *Jurnal Informatika UKM*, 2005.

[9] Alan Dennis, Barbara Haley Wixom, and David Tegarde. *Systems Analysis & Design With UMLVersion 2.0 (An-Object Oriented Approach)*. John Wiley & Sons, (2nd edition) edition, 2005.

[10] Ayman Hourieh. *Learning Website Development with Django*. Packt Publishing Ltd., (1st edition) edition, April 2008.

[11] Nick Jenkins. *A Software Testing Primer: An Introduction to Software Testing*. 2008.

[12] Glenn Johnson. *Programming In HTML5 With JavaScript And CSS3 (Training Guide)*. Microsoft Press, 2013.

[13] Mohd. Ehmer Khan and Farmeena Khan. A comparative study of white box, black box and grey box testing techniques. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 2012.

[14] Gunnit Khurana and Raman Khurana. Website design methodology. 2004.

[15] A. Mahendra. Struktur navigasi. Online, August 2009. Available: http://oke.or.id/wp-content/plugins/downloads-manager/upload/Struktur[Accessed March 29, 2016].

[16] Srinivas Nidhra and Jagruthi Dondeti. Black box and white box testing techniques. *International Journal of Embedded Systems and Applications (IJESA)*, 2012.

[17] Agustinus Noertjahyana. Studi analisis rapid development sebagai salah satu alternatif metode pengembangan perangkat lunak. *Jurnal Informatika 3*, 2:pp. 74–79., 2002.

[18] Roger S. Pressman. *Software Engineering: A Practitioner'sApproach*. Mc Graw-Hill, New York, (7th edition) edition, 2010.

[19] Muhammad Huda Rabbani. Website classification system using keyword method and term frequency-inverse document frequency (td-idf) weighting method. *Gunadarma University*, 2015.

[20] R.H.Sianipar and Hamzan Wadi. *Pemrogramman Python Teori dan Implementasi*. Informatika, 2015.

[21] F. Rousseaux and K. Lhoste. Rapid software prototyping using ajax and google map api. in advances in computer-human interactions. *Advances in Computer-Human Interactions, ACHI'09, Second International Conferences on IEEE,* pages pp. 317–323., 2009.

[22] James Rumbaugh, Ivar Jacobson, and Grady Booch. *The Unified Modeling Language Reference Manual*. Addison-Wesley, 2005.

[23] Munish Saini and Kuljit Kaur. A review of open source software development life cycle models. *International Journal of Multimedia and Ubiquituous Engineering*, 2014.

[24] J. Thompson. *Manga, The Complete Guide*. Del Rey Books, an imprint of The Random House, Inc, New York, 2007.

[25] Ian Ward. excess.org: Django 1.1 talk text. Online, 2009. Available: https://excess.org/article/2009/05/django-1-1-talk-text. [Accessed March 29, 2016].

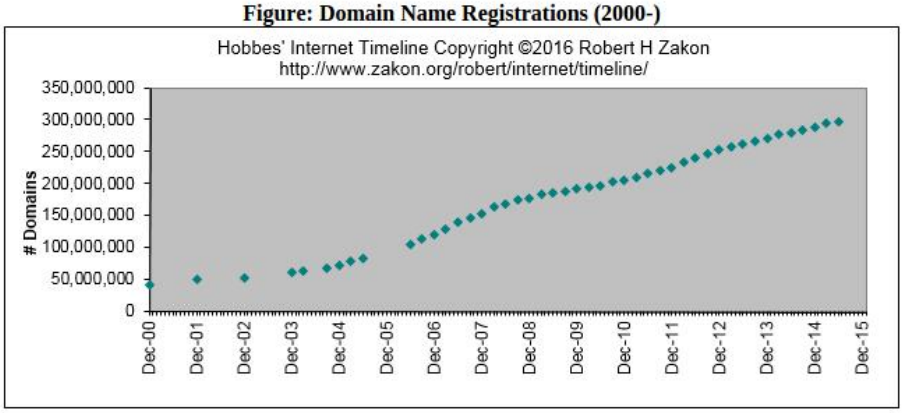
[26] Jeffrey Zeldman. *Taking Your Talent to the Web: A Guide for the Transitioning Designer*. New Riders Publishing, Indiana, (1st edition) edition, May 2001.

[27] J. Zipes. *The Enchanted Screen: The Unknown History of Fairy-Tale Films*. Routledge, New York, 2011.

Appendix

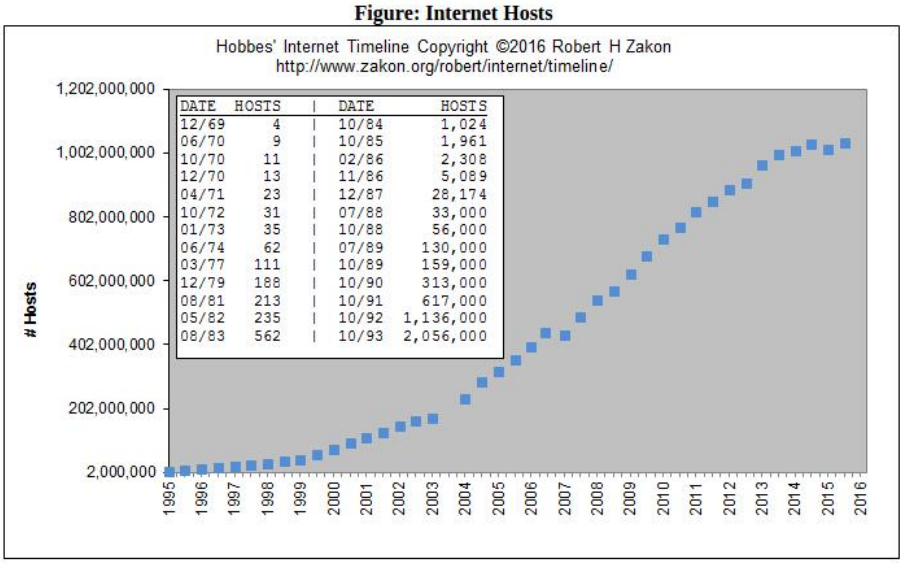
- Domain Growth

Figure: Domain Name Registrations (2000-)



- Internet Hosts Growth

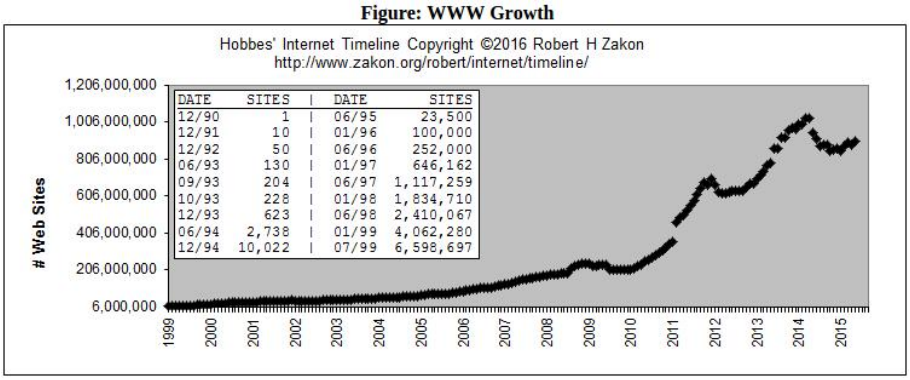
Figure: Internet Hosts



DATE	HOSTS	DATE	HOSTS
12/69	4	10/84	1,024
06/70	9	10/85	1,961
10/70	11	02/86	2,308
12/70	13	11/86	5,089
04/71	23	12/87	28,174
10/72	31	07/88	33,000
01/73	35	10/88	56,000
06/74	62	07/89	130,000
03/77	111	10/89	159,000
12/79	188	10/90	313,000
08/81	213	10/91	617,000
05/82	235	10/92	1,136,000
08/83	562	10/93	2,056,000

- WWW Growth

Figure: WWW Growth



DATE	SITES	DATE	SITES
12/90	1	06/95	23,500
12/91	10	01/96	100,000
12/92	50	06/96	252,000
06/93	130	01/97	646,162
09/93	204	06/97	1,117,259
10/93	228	01/98	1,834,710
12/93	623	06/98	2,410,067
06/94	2,738	01/99	4,062,280
12/94	10,022	07/99	6,598,697

- Percentage of Viewers who Buy the Reviewed Services

Purchase Behavior Subsequent to Online Review Consultation October 12-18, 2007 Source: comScore, Inc./The Kelsey Group	
Service	**Percent of Review Viewers Subsequently Making a Purchase of Stated Service**
Restaurant	41%
Hotels	40%
Travel	27%
Automotive	24%
Home	19%
Medical	14%
Legal	8%

Appendix B

Listing Program

settings.py

```
# Django settings for animetacchi project.

import os
DEBUG = True
TEMPLATE_DEBUG = DEBUG
PROJECT_PATH = os.path.dirname(os.path.dirname(__file__))

ADMINS = (
    # ('Your Name', 'your_email@example.com'),
)

MANAGERS = ADMINS

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql_psycopg2',
          # Add 'postgresql_psycopg2', 'mysql', 'sqlite3'
          or 'oracle'.
        'NAME': 'animetacchi',                      # Or
          path to database file if using sqlite3.
        # The following settings are not used with sqlite3:
        'USER': 'recruit',
        'PASSWORD': 'recruit',
        'HOST': 'localhost',                      # Empty
          for localhost through domain sockets or
          '127.0.0.1' for localhost through TCP.
```

```
24          'PORT': '',                      # Set to empty
                string for default.
25      }
26  }
27
28  # Hosts/domain names that are valid for this site; required
        if DEBUG is False
29  # See https://docs.djangoproject.com/en/1.5/ref/settings/#
        allowed-hosts
30  ALLOWED_HOSTS = ['*']
31
32  # Local time zone for this installation. Choices can be
        found here:
33  # http://en.wikipedia.org/wiki/List_of_tz_zones_by_name
34  # although not all choices may be available on all operating
         systems.
35  # In a Windows environment this must be set to your system
        time zone.
36  TIME_ZONE = 'America/Chicago'
37
38  # Language code for this installation. All choices can be
        found here:
39  # http://www.i18nguy.com/unicode/language-identifiers.html
40  LANGUAGE_CODE = 'en-us'
41
42  SITE_ID = 1
43
44  # If you set this to False, Django will make some
        optimizations so as not
45  # to load the internationalization machinery.
46  USE_I18N = True
47
48  # If you set this to False, Django will not format dates,
        numbers and
49  # calendars according to the current locale.
50  USE_L10N = True
51
52  # If you set this to False, Django will not use timezone-
        aware datetimes.
53  USE_TZ = False
```

```
54
55 MEDIA_ROOT = os.path.join(PROJECT_PATH, 'media')
56 MEDIA_URL = '/media/'
57 STATIC_ROOT = os.path.join(PROJECT_PATH, 'static')
58 STATIC_URL = '/static/'
59 ADMIN_MEDIA_PREFIX = '/static/admin/'
60
61 # CAPTCHA SETTINGS
62 CAPTCHA_FONT_PATH = os.path.join(PROJECT_PATH, 'courier.ttf'
       )
63 CAPTCHA_FONT_SIZE = 34
64
65 # Additional locations of static files
66 STATICFILES_DIRS = (
67     # Put strings here, like "/home/html/static" or "C:/www/
          django/static".
68     # Always use forward slashes, even on Windows.
69     # Don't forget to use absolute paths, not relative paths
          .
70 )
71
72 # List of finder classes that know how to find static files
       in
73 # various locations.
74 STATICFILES_FINDERS = (
75     'django.contrib.staticfiles.finders.FileSystemFinder',
76     'django.contrib.staticfiles.finders.AppDirectoriesFinder
          ',
77 #     'django.contrib.staticfiles.finders.
       DefaultStorageFinder',
78     'compressor.finders.CompressorFinder',
79 )
80
81 # Make this unique, and don't share it with anybody.
82 SECRET_KEY = '#rj!&#1rml6u81l!lm8w1$u_1=+(e1gd88i!51irap(
       b_omv8k'
83
84 # List of callables that know how to import templates from
       various sources.
85 TEMPLATE_LOADERS = (
```

```
86      'django.template.loaders.filesystem.Loader',
87      'django.template.loaders.app_directories.Loader',
88       #'django.template.loaders.eggs.Loader',
89  )
90
91  MIDDLEWARE_CLASSES = (
92      #'animetacchi.middleware.DefaultMiddleware',
93      'django.middleware.common.CommonMiddleware',
94      'django.contrib.sessions.middleware.SessionMiddleware',
95      'django.middleware.csrf.CsrfViewMiddleware',
96      'django.contrib.auth.middleware.AuthenticationMiddleware
            ',
97      'django.contrib.messages.middleware.MessageMiddleware',
98      'responsive.middleware.DeviceInfoMiddleware',
99      # Uncomment the next line for simple clickjacking
            protection:
100     # 'django.middleware.clickjacking.
            XFrameOptionsMiddleware',
101 )
102
103 #SOCIAL_AUTH_USER_MODEL = 'numflo.CustomUser'
104
105 LOGIN_URL = '/signin/'
106 ROOT_URLCONF = 'animetacchi.urls'
107 SOCIAL_AUTH_NEW_ASSOCIATION_REDIRECT_URL = '/'
108 SOCIAL_AUTH_LOGIN_REDIRECT_URL = '/'
109 #SOCIAL_AUTH_USER_MODEL = 'numflo.Student'
110
111 TEMPLATE_CONTEXT_PROCESSORS = (
112     'django.contrib.auth.context_processors.auth',
113     'django.core.context_processors.i18n',
114     'django.core.context_processors.request',
115     'django.core.context_processors.media',
116     'django.core.context_processors.static',
117     'responsive.context_processors.device_info'
118 )
119
120 DEFAULT_BREAKPOINTS = {
121     'phone': 480,
122     'tablet': 767,
```

```
123     'desktop': None,
124 }
125
126 TEMPLATE_DIRS = (
127     os.path.join(PROJECT_PATH, 'templates')
128 )
129
130 # Python dotted path to the WSGI application used by Django's runserver.
131 WSGI_APPLICATION = 'animetacchi.wsgi.application'
132
133 EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'
134 EMAIL_USE_TLS = True
135 EMAIL_HOST = 'medusa.hideserver.net'
136 EMAIL_PORT = 465
137 EMAIL_HOST_USER = 'be-py@alviandk.com'
138 EMAIL_HOST_PASSWORD = 'be-pyp4ssword'
139 DEFAULT_FROM_EMAIL = 'Default Web <be-py@alviandk.com>'
140
141
142 # compression setup
143 COMPRESS_ENABLED = True
144 COMPRESS_CSS_FILTERS = ['compressor.filters.css_default.
    CssAbsoluteFilter', 'compressor.filters.cssmin.
    CSSMinFilter']
145 COMPRESS_URL = MEDIA_URL
146 COMPRESS_ROOT = MEDIA_ROOT
147 COMPRESS_PARSER = 'compressor.parser.LxmlParser'
148
149 DJANGO_WYSIWYG_FLAVOR = "tinymce_advanced"
150
151 INSTALLED_APPS = (
152     'django.contrib.auth',
153     'django.contrib.contenttypes',
154     'django.contrib.sessions',
155     'django.contrib.sites',
156     'django.contrib.messages',
157     'django.contrib.staticfiles',
158     'django.contrib.humanize',
```

```
159     # Uncomment the next line to enable the admin:
160     'django.contrib.admin',
161     'django_wysiwyg',
162     'compressor',
163     'animetacchi',
164     'captcha',
165     'south',
166     'social_auth',
167     'responsive',
168     'easy_thumbnails',
169     'import_export',
170     'djcelery',
171     'service',
172     'mathfilters',
173 
174     #eksternal app
175     'ratings',
176     'vote',
177     # Uncomment the next line to enable admin documentation:
178     # 'django.contrib.admindocs',
179 )
180 
181 AUTHENTICATION_BACKENDS = (
182     'social_auth.backends.twitter.TwitterBackend',
183     'social_auth.backends.facebook.FacebookBackend',
184     'social_auth.backends.google.GoogleOAuthBackend',
185     'social_auth.backends.google.GoogleOAuth2Backend',
186     'social_auth.backends.google.GoogleBackend',
187     'social_auth.backends.yahoo.YahooBackend',
188     'social_auth.backends.browserid.BrowserIDBackend',
189     'social_auth.backends.contrib.linkedin.LinkedinBackend',
190     'social_auth.backends.contrib.disqus.DisqusBackend',
191     'social_auth.backends.contrib.livejournal.
          LiveJournalBackend',
192     'social_auth.backends.contrib.orkut.OrkutBackend',
193     'social_auth.backends.contrib.foursquare.
          FoursquareBackend',
194     'social_auth.backends.contrib.github.GithubBackend',
195     'social_auth.backends.contrib.vk.VKOAuth2Backend',
196     'social_auth.backends.contrib.live.LiveBackend',
```

```
197     'social_auth.backends.contrib.skyrock.SkyrockBackend',
198     'social_auth.backends.contrib.yahoo.YahooOAuthBackend',
199     'social_auth.backends.contrib.readability.
            ReadabilityBackend',
200     'social_auth.backends.contrib.fedora.FedoraBackend',
201     'social_auth.backends.OpenIDBackend',
202     'django.contrib.auth.backends.ModelBackend',
203 )
204
205 TWITTER_CONSUMER_KEY          = ''
206 TWITTER_CONSUMER_SECRET       = ''
207 FACEBOOK_APP_ID               = '1542844709278416'
208 FACEBOOK_API_SECRET           = '
        d98458a1cdde0ca88534df8923bebfbb'
209 LINKEDIN_CONSUMER_KEY        = ''
210 LINKEDIN_CONSUMER_SECRET     = ''
211 ORKUT_CONSUMER_KEY           = ''
212 ORKUT_CONSUMER_SECRET        = ''
213
214 GOOGLE_CONSUMER_KEY          = ''
215 GOOGLE_CONSUMER_SECRET       = ''
216 GOOGLE_OAUTH2_CLIENT_ID      = '797637730491-
        c86l4kib8vnai52cocldpuffavuh59e5.apps.googleusercontent.
        com'
217 GOOGLE_OAUTH2_CLIENT_SECRET  = 'Kzjf1HtJu8TkVQ_TnqGKOhZw'
218
219 FOURSQUARE_CONSUMER_KEY      = ''
220 FOURSQUARE_CONSUMER_SECRET   = ''
221 VK_APP_ID                    = ''
222 VK_API_SECRET                = ''
223 LIVE_CLIENT_ID               = ''
224 LIVE_CLIENT_SECRET           = ''
225 SKYROCK_CONSUMER_KEY         = ''
226 SKYROCK_CONSUMER_SECRET      = ''
227 YAHOO_CONSUMER_KEY           = ''
228 YAHOO_CONSUMER_SECRET        = ''
229 READABILITY_CONSUMER_SECRET  = ''
230 READABILITY_CONSUMER_SECRET  = ''
231
232 SOUTH_MIGRATION_MODULES = {
```

```
233     'easy_thumbnails': 'easy_thumbnails.south_migrations',
234     'djcelery': 'djcelery.south_migrations',
235 }
```

admin.py

```
1  from animetacchi.models import *
2  from django.contrib import admin
3  from django import forms
4
5  class NewsAdmin(admin.ModelAdmin):
6      list_display = ['news','index_cover','article']
7      search_fields = ['news']
8
9  class MembersAdmin(admin.ModelAdmin):
10     list_display = ['user', 'm_name', 'm_cover', 'm_picture'
           , 'm_bio', 'm_tagline', 'seq_members']
11     search_fields = ['m_name', 'm_tagline']
12
13 class GenresAdmin(admin.ModelAdmin):
14     list_display = ['genre_type']
15     search_fields = ['genre_type']
16
17
18 class AnimeAdmin(admin.ModelAdmin):
19     list_display = ['a_name', 'a_displaypic', 'a_synopsys',
          'a_airedstart', 'a_airedend']
20     search_fields = ['a_name', 'a_displaypic', 'a_synopsys',
           'a_airedstart', 'a_airedend']
21
22 #et-projectsAdmin
23 admin.site.register(News, NewsAdmin)
24 admin.site.register(Members, MembersAdmin)
25 #admin.site.register(Friendship)
26 admin.site.register(Genre, GenresAdmin)
27 admin.site.register(About)
28 admin.site.register(Character)
29 admin.site.register(VoiceCharacter)
30 admin.site.register(Anime, AnimeAdmin)
31 admin.site.register(WatchList)
```

```
34 admin.site.register(Daisukilist)
35 admin.site.register(Daikirailist)
36 admin.site.register(CommentAnime)
37 admin.site.register(Chart)
38
39 admin.site.register(Manga)
40 admin.site.register(ReadingList)
41 admin.site.register(CharacterManga)
42 admin.site.register(CommentManga)
43
44 admin.site.register(UserProfile)
45 admin.site.register(RequestAnime)
```

models.py

```
1
2
3 #-*- encoding=UTF-8 -*-
4 from django.db import models
5 from django.contrib.auth.models import User
6 from django.utils.encoding import smart_str, smart_unicode
7 from django.conf import settings
8 import os
9 import re
10 import json
11 from datetime import date, datetime, timedelta
12 from django.template.defaultfilters import slugify
13
14 from django.db.models import Q
15 from django.utils import timezone
16 from django.utils.safestring import mark_safe
17 from django.utils.html import strip_tags
18 from django.core.exceptions import ValidationError
19 from service.tasks import send_notification
20 from django.core.urlresolvers import reverse
21 from django.template.defaultfilters import slugify
22
23 from ratings.handlers import ratings
24 from ratings.forms import VoteForm
25
26 from vote.managers import VotableManager
27
```

```
28
29 class News(models.Model):
30     news = models.TextField(null=True, blank=True)
31     index_cover = models.ImageField(upload_to='IndexCover',
           null=True, blank=True)
32     article = models.TextField(null=True, blank=True)
33     seq_news = models.IntegerField(null=True,blank=True)
34
35     def __unicode__(self):
36         return unicode(self.news)
37
38
39 class Members(models.Model):
40     user = models.ForeignKey(User)
41     m_name = models.CharField(max_length=50, null=True)
42     m_cover = models.ImageField(upload_to='MembersCover',
           null=True, blank=True)
43     m_picture = models.ImageField(upload_to='MembersPicture'
           , null=True, blank=True)
44     m_bio = models.TextField(null=True, blank=True)
45     m_tagline = models.CharField(max_length=50, null=True)
46     blocked = models.BooleanField(default=False)
47     seq_members = models.IntegerField(null=True)
48
49     def __unicode__(self):
50        return unicode(self.user.username)
51 '''
52 class Friendship(models.Model):
53     user = models.ForeignKey(User)
54     friend = models.ForeignKey(User, null=True, blank=True)
55     seq_friendship = models.IntegerField(null=True)
56
57     def __unicode__(self):
58         return unicode(self.user.username)
59 '''
60 class About(models.Model):
61     member = models.ForeignKey('Members', null=True, blank=
           True)
62     about = models.TextField(null=True, blank=True)
63     location = models.TextField(null=True, blank=True)
```

```
64      seq_about = models.IntegerField(null=True)
65
66      def __unicode__(self):
67          return unicode(self.member)
68
69
70  class Genre(models.Model):
71      genre_type = models.CharField(max_length=30, null=True )
72
73      def __unicode__(self):
74          return unicode(self.genre_type)
75
76
77  class Anime(models.Model):
78      GENRE = (('Action', 'Action'), ('Adventure', 'Adventure'
            ), ('Anime Influenced', 'Anime Influenced'), ('Comedy
            ', 'Comedy'), ('Dementia', 'Dementia'), ('Demons', '
            Demons'), ('Doujinshi', 'Doujinshi'), ('Drama', '
            Drama'), ('Ecchi', 'Ecchi'), ('Fantasy', 'Fantasy'),
            ('Game', 'Game'), ('Gender Bender', 'Gender Bender'),
             ('Gore', 'Gore'), ('Harem', 'Harem'), ('Historical',
             'Historical'), ('Horror', 'Horror'), ('Kids', 'Kids'
            ), ('Magic', 'Magic'), ('Mahou Shoujo', 'Mahou Shoujo
            '), ('Mahou Shounen', 'Mahou Shounen'), ('Martial
            Arts', 'Martial Arts'), ('Mecha', 'Mecha'), ('
            Military', 'Military'), ('Music', 'Music'), ('Mystery
            ', 'Mystery'), ('Parody', 'Parody'), ('Police', '
            Police'), ('Psychological', 'Psychological'), ('
            Racing', 'Racing'), ('Romance', 'Romance'), ('Samurai
            ', 'Samurai'), ('School', 'School'), ('Sci-Fi', 'Sci-
            Fi'), ('Shoujo Ai', 'Shoujo Ai'), ('Shounen Ai', '
            Shounen Ai'), ('Slice of Life', 'Slice of Life'), ('
            Space', 'Space'), ('Sports', 'Sports'), ('
            Supernatural', 'Supernatural'), ('Super Power', '
            Super Power'), ('Thriller', 'Thriller'), ('Vampire',
            'Vampire'))
79
80      a_name = models.CharField(max_length=50, null=True)
81      a_genre = models.ManyToManyField(Genre)
```

```
82      a_cover = models.ImageField(upload_to='AnimeCover', null
            =True, blank=True)
83      a_displaypic = models.ImageField(upload_to='AnimePicture
            ', null=True, blank=True)
84      a_synopsys = models.TextField(null=True, blank=True)
85      a_airedstart = models.DateField(null=True, blank=True)
86      a_airedend = models.DateField(null=True, blank=True)
87      save_added = models.DateTimeField(auto_now=True)
88      slug = models.SlugField(null=True, blank=True)
89      seq_anime = models.IntegerField(null=True, blank=True)
90
91      def save(self, *args, **kwargs):
92          self.slug = slugify(self.a_name)
93          super(Anime, self).save(*args, **kwargs)
94
95      def __unicode__(self):
96          return unicode(self.a_name)
97
98  class Manga(models.Model):
99      GENRE = (('Action', 'Action'), ('Adventure', 'Adventure'
            ), ('Anime Influenced', 'Anime Influenced'), ('Comedy
            ', 'Comedy'), ('Dementia', 'Dementia'), ('Demons', '
            Demons'), ('Doujinshi', 'Doujinshi'), ('Drama', '
            Drama'), ('Ecchi', 'Ecchi'), ('Fantasy', 'Fantasy'),
            ('Game', 'Game'), ('Gender Bender', 'Gender Bender'),
             ('Gore', 'Gore'), ('Harem', 'Harem'), ('Historical',
             'Historical'), ('Horror', 'Horror'), ('Kids', 'Kids'
            ), ('Magic', 'Magic'), ('Mahou Shoujo', 'Mahou Shoujo
            '), ('Mahou Shounen', 'Mahou Shounen'), ('Martial
            Arts', 'Martial Arts'), ('Mecha', 'Mecha'), ('
            Military', 'Military'), ('Music', 'Music'), ('Mystery
            ', 'Mystery'), ('Parody', 'Parody'), ('Police', '
            Police'), ('Psychological', 'Psychological'), ('
            Racing', 'Racing'), ('Romance', 'Romance'), ('Samurai
            ', 'Samurai'), ('School', 'School'), ('Sci-Fi', 'Sci-
            Fi'), ('Shoujo Ai', 'Shoujo Ai'), ('Shounen Ai', '
            Shounen Ai'), ('Slice of Life', 'Slice of Life'), ('
            Space', 'Space'), ('Sports', 'Sports'), ('
            Supernatural', 'Supernatural'), ('Super Power', '
            Super Power'), ('Thriller', 'Thriller'), ('Vampire',
```

```
        'Vampire'))
100
101     name = models.CharField(max_length=50, null=True)
102     genre = models.ManyToManyField(Genre)
103     cover = models.ImageField(upload_to='MangaCover', null=
            True, blank=True)
104     displaypic = models.ImageField(upload_to='MangaPicture',
             null=True, blank=True)
105     synopsys = models.TextField(null=True, blank=True)
106     airedstart = models.DateField(null=True, blank=True)
107     airedend = models.DateField(null=True, blank=True)
108     save_added = models.DateTimeField(auto_now=True)
109     slug = models.SlugField(null=True, blank=True)
110     seq_manga = models.IntegerField(null=True, blank=True)
111
112
113     def save(self, *args, **kwargs):
114         self.slug = slugify(self.name)
115         super(Manga, self).save(*args, **kwargs)
116
117     def __unicode__(self):
118        return unicode(self.name)
119
120
121 class WatchList(models.Model):
122     STATUS = (('Bliss', 'Bliss'), ('Life', 'Life'),
123             ('Hope', 'Hope'), ('Agony', 'Agony'),)
124     members = models.ForeignKey('Members', null=True, blank=
            True)
125     watchlist = models.ForeignKey('Anime', null=True, blank=
            True)
126     status = models.CharField(max_length=10, choices=STATUS,
             null=True)
127     seq_watchlist = models.IntegerField(null=True)
128
129     def __unicode__(self):
130         return unicode(self.members)
131
132 class ReadingList(models.Model):
133     STATUS = (('Bliss', 'Bliss'), ('Life', 'Life'),
```

```
134             ('Hope', 'Hope'), ('Agony', 'Agony'),)
135     members = models.ForeignKey('Members', null=True, blank=
            True)
136     readinglist = models.ForeignKey('Manga', null=True,
            blank=True)
137     status = models.CharField(max_length=10, choices=STATUS,
             null=True)
138     seq_watchlist = models.IntegerField(null=True)
139
140     def __unicode__(self):
141         return unicode(self.members)
142
143
144 class Character(models.Model):
145     name = models.CharField(max_length=50, null=True)
146     picture = models.ImageField(upload_to='CharacterPicture'
            , null=True, blank=True)
147     anime = models.ForeignKey('Anime', null=True, blank=True
            )
148     synopsys = models.TextField(null=True, blank=True)
149
150     def __unicode__(self):
151         return (self.name)
152
153 class CharacterManga(models.Model):
154     name = models.CharField(max_length=50, null=True)
155     picture = models.ImageField(upload_to='CharacterPicture'
            , null=True, blank=True)
156     manga = models.ForeignKey('Manga', null=True, blank=True
            )
157     synopsys = models.TextField(null=True, blank=True)
158
159     def __unicode__(self):
160         return (self.name)
161
162 class VoiceCharacter(models.Model):
163     name = models.CharField(max_length=50, null=True)
164     picture = models.ImageField(upload_to='VoiceActorPicture
            ', null=True, blank=True)
```

```
165     character = models.ManyToManyField('Character', null=
            True, blank=True)
166     synopsys = models.TextField(null=True, blank=True)
167 
168     def __unicode__(self):
169         return (self.name)
170 
171 
172 class Daisukilist(models.Model):
173     members = models.ForeignKey('Members')
174     anime = models.ManyToManyField(Anime,null=True, blank=
            True )
175     manga = models.ManyToManyField(Manga,null=True, blank=
            True )
176     character = models.ManyToManyField(Character,null=True,
            blank=True)
177     character_manga = models.ManyToManyField(CharacterManga,
            null=True, blank=True)
178     voice_character = models.ManyToManyField(VoiceCharacter,
            null=True, blank=True)
179 
180     def __unicode__(self):
181         return ('{}-daisuki'.format(self.members))
182 
183 
184 class Daikirailist(models.Model):
185     members = models.ForeignKey('Members')
186     anime = models.ManyToManyField(Anime,null=True, blank=
            True )
187     manga = models.ManyToManyField(Manga,null=True, blank=
            True )
188     character = models.ManyToManyField(Character,null=True,
            blank=True)
189     character_manga = models.ManyToManyField(CharacterManga,
            null=True, blank=True)
190 
191     def __unicode__(self):
192         return ('{}-daikirai'.format(self.members))
193 
194 class Chart(models.Model):
```

```
195     SEASON = (('winter', 'winter'), ('spring', 'spring'),
196               ('summer', 'summer'), ('fall', 'fall'),)
197
198     season = models.CharField(max_length=10, choices=SEASON,
            null=True)
199     anime = models.ManyToManyField(Anime,null=True, blank=
            True )
200     year = models.IntegerField(null=True, blank=True)
201
202     class Meta:
203         ordering = ['year']
204
205     def __unicode__(self):
206         return ('{}-{}'.format(self.season, self.year))
207
208
209 class CommentAnime(models.Model):
210     comment = models.TextField()
211     user = models.ForeignKey('Members', null=True, blank=
            True)
212     anime = models.ForeignKey('Anime', null=True, blank=True
            )
213     added = models.DateTimeField(auto_now=True,null=True,
            blank=True)
214     edited = models.DateTimeField(null=True, blank=True)
215     votes = VotableManager()
216
217     class Meta:
218         ordering = ['-edited', '-added']
219
220     def __unicode__(self):
221         return ('{}-{}'.format(self.user, self.comment[:20])
                )
222
223 class CommentManga(models.Model):
224     comment = models.TextField()
225     user = models.ForeignKey('Members', null=True, blank=
            True)
226     manga = models.ForeignKey('Manga', null=True, blank=True
            )
```

```
227     added = models.DateTimeField(auto_now=True,null=True,
            blank=True)
228     edited = models.DateTimeField(null=True, blank=True)
229     votes = VotableManager()
230
231     class Meta:
232         ordering = ['-edited', '-added']
233
234     def __unicode__(self):
235         return ('{}-{}'.format(self.user, self.comment[:20])
                )
236
237
238 class UserProfile(models.Model):
239
240
241     user = models.OneToOneField(User)
242     activation_key = models.CharField(max_length=40, blank=
            True)
243     key_expires = models.DateTimeField(default= datetime.now
            )
244
245     def __str__(self):
246         return self.user.username
247
248 class RequestAnime(models.Model):
249     content = models.TextField()
250     user = models.ForeignKey('Members', null=True, blank=
            True)
251     anime = models.ForeignKey('Anime', null=True, blank=True
            )
252     added = models.DateTimeField(auto_now=True,null=True,
            blank=True)
253
254
255     class Meta:
256         ordering = ['-added']
257
258     def __unicode__(self):
259         return ('{}-{}'.format(self.user, self.added))
```

```
260
261
262
263 ratings.register(Anime, score_range=(1, 5), form_class=
      VoteForm)
264 ratings.register(Manga, score_range=(1, 5), form_class=
      VoteForm)
265 ratings.register(Character, score_range=(1, 5), form_class=
      VoteForm)
266 ratings.register(CharacterManga, score_range=(1, 5),
      form_class=VoteForm)
267 ratings.register(VoiceCharacter, score_range=(1, 5),
      form_class=VoteForm)
```

urls.py

```
1  urls.py
2
3  from django.conf.urls import patterns, include, url
4  from django.conf import settings
5  # Uncomment the next two lines to enable the admin:
6  from django.contrib import admin
7  admin.autodiscover()
8  from django.conf import settings
9
10 # Uncomment the next two lines to enable the admin:
11 # from django.contrib import admin
12 # admin.autodiscover()
13
14 urlpatterns = patterns('',
15     # Examples:
16     url(r'^$', 'animetacchi.views.home', name='home'),
17     url(r'^about/$', 'animetacchi.views.about', name='about'
         ),
18     url(r'^services/$', 'animetacchi.views.services', name='
         services'),
19
20     # account
21     url(r'^logout/$', 'animetacchi.views.logout', name='
         logout'),
22     url(r'^signin/$', 'animetacchi.views.signin', name='
         signin'),
```

```
23      url(r'^signup/$', 'animetacchi.views.signup', name='
            signup'),
24      url(r'^confirm/(?P<activation_key>[a-zA-Z\-\_0-9\: ]+)/$
            ', 'animetacchi.views.register_confirm', name='
            confirm'),
25      url(r'^check-email/$', 'animetacchi.views.check_email',
            name='check_email'),
26
27      # chart
28      url(r'^chart/(?P<season>[a-zA-Z\-\_0-9]+)/$', '
            animetacchi.views.anime_chart', name='anime_chart'),
29
30      # anime
31      url(r'^anime/$', 'animetacchi.views.anime', name='anime'
            ),
32      url(r'^anime/(?P<slug>[a-zA-Z\-\_0-9\: ]+)/$', '
            animetacchi.views.anime_details', name='anime_details
            '),
33      url(r'^anime/(?P<slug>[a-zA-Z\-\_0-9\: ]+)/reviews/$', '
            animetacchi.views.anime_reviews', name='anime_reviews
            '),
34      url(r'^anime/(?P<slug>[a-zA-Z\-\_0-9\: ]+)/reviews/add/$
            ', 'animetacchi.views.anime_reviews_add', name='
            anime_reviews_add'),
35      url(r'^anime/(?P<slug>[a-zA-Z\-\_0-9\: ]+)/character/(?P
            <id>\d+)/$', 'animetacchi.views.anime_character',
            name='anime_character'),
36      url(r'^anime/voice/(?P<id>\d+)/$', 'animetacchi.views.
            anime_voice', name='anime_voice'),
37
38      url(r'^daikirai/$', 'animetacchi.views.daikirai', name='
            daikirai'),
39      url(r'^daisuki/$', 'animetacchi.views.daisuki', name='
            daisuki'),
40      url(r'^watchlist/$', 'animetacchi.views.watchlist', name
            ='watchlist'),
41      url(r'^readinglist/$', 'animetacchi.views.readinglist',
            name='readinglist'),
42
43
```

```
44
45      # manga
46      url(r'^manga/$', 'animetacchi.views.manga', name='manga'
            ),
47      url(r'^manga/(?P<slug>[a-zA-Z\-\_0-9\: ]+)/$', '
            animetacchi.views.manga_details', name='manga_details
            '),
48      url(r'^manga/(?P<slug>[a-zA-Z\-\_0-9\: ]+)/reviews/$', '
            animetacchi.views.manga_reviews', name='manga_reviews
            '),
49      url(r'^manga/(?P<slug>[a-zA-Z\-\_0-9\: ]+)/reviews/add/$
            ', 'animetacchi.views.manga_reviews_add', name='
            manga_reviews_add'),
50      url(r'^manga/(?P<slug>[a-zA-Z\-\_0-9\: ]+)/character/(?P
            <id>\d+)/$', 'animetacchi.views.manga_character',
            name='manga_character'),
51
52      # user
53      url(r'^dashboard/(?P<username>[a-zA-Z\-\_0-9]+)/$', '
            animetacchi.views.dashboard', name='dashboard'),
54      url(r'^users/(?P<username>[a-zA-Z\-\_0-9]+)/$', '
            animetacchi.views.users', name='users'),
55      url(r'^users/(?P<username>[a-zA-Z\-\_0-9]+)/library/$',
            'animetacchi.views.library', name='library'),
56      url(r'^users/(?P<username>[a-zA-Z\-\_0-9]+)/library/
            manga/$', 'animetacchi.views.library_manga', name='
            library_manga'),
57
58      # Uncomment the next line to enable the admin:
59      url(r'^admin/', include(admin.site.urls)),
60
61      #ratings
62      url(r'^ratings/', include('ratings.urls')),
63      url(r'^ratings/average/(?P<entity>[a-zA-Z\-\_0-9\: ]+)
            /(?P<id>[a-zA-Z\-\_0-9\: ]+)/', 'animetacchi.views.
            ratings_average', name='ratings_average'),
64
65      #likes
66      url(r'^likes/(?P<entity>[a-zA-Z\-\_0-9\: ]+)/', '
            animetacchi.views.likes', name='likes'),
```

```
67      url(r'^show_likes/(?P<entity>[a-zA-Z\-\_0-9\: ]+)/(?P<id
            >[a-zA-Z\-\_0-9\: ]+)/', 'animetacchi.views.
            show_likes', name='show_likes'),
68
69      url(r'^sort_comment/$', 'animetacchi.views.sort_comment'
            , name='sort_comment'),
70      url(r'^sort_comment_manga/$', 'animetacchi.views.
            sort_comment_manga', name='sort_comment_manga'),
71
72      url(r'^search_comment/$', 'animetacchi.views.
            search_comment', name='search_comment'),
73
74      url(r'^request_edit/$', 'animetacchi.views.request_edit'
            , name='request_edit'),
75
76  )
77  if settings.DEBUG:
78      urlpatterns += patterns('',
79          url(r'^media/(?P<path>.*)$', 'django.views.static.
                serve', {
80              'document_root': settings.MEDIA_ROOT,
81          }),
82          url(r'^static/(?P<path>.*)$', 'django.views.static.
                serve', {
83              'document_root': settings.STATIC_ROOT,
84          }),
85  )
```

```
1   views.py
2
3   # -*- coding: utf-8 -*-
4   from django.core.cache import cache
5   from django.views.decorators.cache import cache_page
6   from django.contrib.auth.models import User, check_password
7   from django.contrib.auth.forms import UserCreationForm
8   from django.shortcuts import get_object_or_404,
        render_to_response
9   from django.template import Context, Template,
        RequestContext
10  from django.template.defaultfilters import slugify
```

```
11 from django.template import Template as template_django
12 from django.core import serializers
13 from django.core import urlresolvers
14 from django.core.mail import send_mail, EmailMessage
15 from django.http import HttpResponse, HttpResponseRedirect,
       Http404
16 from django.core.urlresolvers import reverse
17 from django.utils.html import strip_tags
18 from django.contrib import auth
19 from django.contrib import messages
20 from django.conf import settings
21 from django.core.paginator import Paginator, InvalidPage,
       EmptyPage, PageNotAnInteger
22 from django.views.decorators.csrf import csrf_exempt
23 from django.utils.encoding import smart_str, smart_unicode
24 from django.contrib.auth.decorators import login_required
25 from django.contrib.auth import authenticate, login
26 from django.db.models import Q, Count
27 from django.shortcuts import render
28 from django.core.exceptions import ObjectDoesNotExist
29
30 import re
31 import string
32 import pycountry
33 import os
34 import commands
35 import json
36 import whois
37 import uuid
38 import urllib2
39 import urllib
40 import string
41 import calendar
42
43 import random
44 import base64
45 import hashlib
46 import hmac
47 import simplejson
48 import time as times
```

```
49
50
51 from captcha.fields import CaptchaField
52 from datetime import date, datetime, timedelta, time
53 from django.utils.timezone import utc
54
55 from animetacchi.models import *
56 from service.tasks import send_email_task, send_sms_task,
       send_notification
57
58 from restclient import POST
59
60 def email_send_with_api(subject, to, msg, fr=None):
61   if not fr:
62     fr = 'Animetacchi'
63   url= 'http://www.tokowebku.com/api/email_sender/'
64   form_fields = {'from': fr, 'subject': subject, 'to': to, '
       msg': msg}
65   result = POST(url, async=False, params=form_fields)
66   return result
67
68 def home(request):
69     #substract_date = datetime.now() - timedelta(days=3)
70     #anime_data = Anime.objects.filter(save_added__gte=
         substract_date)
71     anime_data = Anime.objects.all()[:3]
72     index = News.objects.all()
73     return render_to_response('index.html',locals(),
         context_instance=RequestContext(request))
74
75 def about(request):
76     if request.POST.get('action') == 'saveAbout':
77         aboutField = request.POST.get('about')
78         memberField = request.POST.get('member')
79         aboutData = About()
80         aboutData.member = memberField
81         aboutData.about = aboutField
82         aboutData.save()
83         return render_to_response('User/dashboard.html',
             locals(),context_instance=RequestContext(request)
```

```
            )
84      return render_to_response('about.html',locals(),
            context_instance=RequestContext(request))
85
86  def services(request):
87      return render_to_response('services.html',locals(),
            context_instance=RequestContext(request))
88
89  def signin(request):
90      if request.POST:
91          try:
92              email = request.POST.get('email')
93              password = request.POST.get('password')
94              email = User.objects.get(email=email)
95              cek_auth = auth.authenticate(username=email.
                    username, password=password)
96              auth.login(request, cek_auth)
97
98          #Doesn't Exist
99          except User.DoesNotExist:
100             message_error = email +" does not exist, please
                    register first."
101             return render_to_response('signin.html', locals
                    (), context_instance=RequestContext(request))
102
103         #Not Match Email and Password
104         except Exception, err:
105             message_match = "The email and password that you
                     entered don't match."
106             return render_to_response('signin.html', locals
                    (), context_instance=RequestContext(request))
107
108         #Active User Filter
109         members = Members.objects.get(user=request.user)
110         if request.user.is_active is False:
111             message_active = "Sorry, your ID is not active."
112             return render_to_response('signin.html',locals()
                    ,context_instance=RequestContext(request))
113         elif members.blocked is True:
114             message_active = "Sorry, your ID is blocked."
```

```
115             return render_to_response('signin.html',locals()
                    ,context_instance=RequestContext(request))
116
117         message_success = 'success'
118         member = Members.objects.get(user=request.user)
119         return render_to_response('signin.html',locals(),
                context_instance=RequestContext(request))
120
121     return render_to_response('signin.html',locals(),
            context_instance=RequestContext(request))
122
123
124 def logout(request):
125     auth.logout(request)
126     return HttpResponseRedirect(reverse('home'))
127
128 def signup(request):
129     if request.POST.get('action') == 'signup':
130         username = request.POST.get('uname')
131         email = request.POST.get('email')
132         password = request.POST.get('passwd')
133         if User.objects.filter(username=username):
134             json_data = {'alert': 'Sorry, username ' +
                        username + ' has been used!!', 'val': False}
135             return HttpResponse(simplejson.dumps(json_data),
                        content_type="application/json")
136         if User.objects.filter(email=email):
137             json_data = {'alert': 'Sorry, email ' + email +
                        ' has been used!!', 'val': False}
138             return HttpResponse(simplejson.dumps(json_data),
                        content_type="application/json")
139         usr = User.objects.create_user(username=username,
                email=email,password=password)
140         usr.is_active = False # not active until he opens
                activation link
141         usr.save()
142         member = Members()
143         member.user = usr
144         member.m_name = username
145         member.save()
```

```
146         activation_key=str(uuid.uuid4())
147
148         new_profile = UserProfile(user=usr, activation_key=
                activation_key)
149         new_profile.save()
150         host=request.META['HTTP_HOST']
151         email_subject = 'Account confirmation'
152         #email_body = "Hey {}, thanks for signing up. To
                activate your account, click this link within \
153         #48 hours http://{}/confirm/{}".format(username,
                host, activation_key)
154         send_mail(email_subject, email_body, 'be-py@alviandk
                .com',[email], fail_silently=False)
155         email_send_with_api(email_subject, email, email_body
                , fr=None)
156         json_data = {'alert': 'email_subject Success!!'}
157         return HttpResponse(simplejson.dumps(json_data),
                content_type="application/json")
158     return render_to_response('signup.html',locals(),
            context_instance=RequestContext(request))
159
160 def check_email(request):
161     return render_to_response('User/check-email.html',locals
            (),context_instance=RequestContext(request))
162
163
164 def register_confirm(request, activation_key):
165     #check if user is already logged in and if he is
            redirect him to some other url, e.g. home
166     if request.user.is_authenticated():
167         HttpResponseRedirect(reverse('home'))
168
169     # check if there is UserProfile which matches the
            activation key (if not then display 404)
170     try:
171         user_profile = UserProfile.objects.get(
                activation_key=activation_key)
172     except:
173         return HttpResponseRedirect(reverse('home'))
174
```

```
175     #check if the activation key has expired, if it hase
            then render confirm_expired.html
176     if user_profile.key_expires + timedelta(days=1) <
            timezone.now():
177         user_profile.user.delete()
178         return render_to_response('User/confirm_expired.html
                ',locals(),context_instance=RequestContext(
                request))
179     #if the key hasn't expired save user and set him as
            active and render some template to confirm activation
180     user = user_profile.user
181     user.is_active = True
182     user.save()
183     return render_to_response('User/confirm.html',locals(),
            context_instance=RequestContext(request))
184
185 def anime(request):
186     #Search Function
187     if request.POST.get('action') == 'search':
188         search_val = request.POST.get('search_value')
189         anime_data = Anime.objects.filter(a_name__contains=
                search_val).order_by('a_name')
190
191         paginator = Paginator(anime_data, 10)
192         try:
193             page = int(request.GET.get('page'))
194             if not page:
195                 page = 1
196         except Exception:
197             page = 1
198
199         anime_data = paginator.page(page).object_list
200         current_page = paginator.page(page)
201         number = current_page.number
202
203         if request.GET.get('pages_ajax') is not None :
204             pass
205
206         return render_to_response('anime_data.html',locals()
                ,context_instance=RequestContext(request))
```

```
207
208     #Sorting Function
209     if request.POST.get('action') == 'checkbox':
210         anime_check = json.loads(request.POST.get('
                anime_check'))
211         if len(anime_check) > 0:
212             anime_data = Anime.objects.filter(a_genre__in=
                    anime_check).distinct('a_name')
213         else:
214             anime_data = Anime.objects.all().order_by('
                    a_name')
215
216         paginator = Paginator(anime_data, 10)
217         try:
218             page = int(request.GET.get('page'))
219             if not page:
220                 page = 1
221         except Exception:
222             page = 1
223
224         anime_data = paginator.page(page).object_list
225         current_page = paginator.page(page)
226         number = current_page.number
227
228         if request.GET.get('pages_ajax') is not None :
229             pass
230
231         return render_to_response('anime_data.html',locals()
                ,context_instance=RequestContext(request))
232
233     #Default Data
234     genre = Genre.objects.all()
235     anime_data = Anime.objects.all().order_by('a_name')
236     paginator = Paginator(anime_data, 10)
237     try:
238         page = int(request.GET.get('page'))
239         if not page:
240             page = 1
241     except Exception:
242         page = 1
```

```
243
244     anime_data = paginator.page(page).object_list
245     current_page = paginator.page(page)
246     number = current_page.number
247
248     if request.GET.get('pages_ajax') is not None :
249         return render_to_response('anime_data.html', locals
            (), context_instance=RequestContext(request))
250
251     return render_to_response('anime.html',locals(),
        context_instance=RequestContext(request))
252
253 def manga(request):
254     #Search Function
255     if request.POST.get('action') == 'search':
256         search_val = request.POST.get('search_value')
257         manga_data = Manga.objects.filter(a_name__contains=
            search_val).order_by('name')
258
259         paginator = Paginator(manga_data, 10)
260         try:
261             page = int(request.GET.get('page'))
262             if not page:
263                 page = 1
264         except Exception:
265             page = 1
266
267         manga_data = paginator.page(page).object_list
268         current_page = paginator.page(page)
269         number = current_page.number
270
271         if request.GET.get('pages_ajax') is not None :
272             pass
273
274         return render_to_response('manga_data.html',locals()
            ,context_instance=RequestContext(request))
275
276     #Sorting Function
277     if request.POST.get('action') == 'checkbox':
```

```
278         manga_check = json.loads(request.POST.get('
                manga_check'))
279         if len(manga_check) > 0:
280             manga_data = Manga.objects.filter(a_genre__in=
                    manga_check).distinct('name')
281         else:
282             manga_data = Manga.objects.all().order_by('name'
                    )
283 
284         paginator = Paginator(manga_data, 10)
285         try:
286             page = int(request.GET.get('page'))
287             if not page:
288                 page = 1
289         except Exception:
290             page = 1
291 
292         manga_data = paginator.page(page).object_list
293         current_page = paginator.page(page)
294         number = current_page.number
295 
296         if request.GET.get('pages_ajax') is not None :
297             pass
298 
299         return render_to_response('manga_data.html',locals()
                ,context_instance=RequestContext(request))
300 
301     #Default Data
302     genre = Genre.objects.all()
303     manga_data = Manga.objects.all().order_by('name')
304     paginator = Paginator(manga_data, 10)
305     try:
306         page = int(request.GET.get('page'))
307         if not page:
308             page = 1
309     except Exception:
310         page = 1
311 
312     manga_data = paginator.page(page).object_list
313     current_page = paginator.page(page)
```

```
314     number = current_page.number
315
316     if request.GET.get('pages_ajax') is not None :
317         return render_to_response('manga_data.html', locals
                (), context_instance=RequestContext(request))
318
319     return render_to_response('manga.html',locals(),
            context_instance=RequestContext(request))
320
321 def anime_details(request,slug):
322     anime_data = Anime.objects.get(slug=slug)
323     commentanimes = CommentAnime.objects.filter(anime=
            anime_data).annotate(likes=Count('votes')).order_by('
            -likes','-edited','-added')
324
325     #c.votes.up(request.user)
326     try:
327     members = Members.objects.get(user=request.user)
328     except:
329         pass
330     try:
331         Daisukilist.objects.get(anime=anime_data, members=
                members)
332         daisuki = True
333     except:
334         daisuki = False
335     try:
336
337         Daikirailist.objects.get(anime=anime_data, members=
                members)
338         daikirai = True
339     except:
340         daikirai = False
341     try:
342         watchlist=WatchList.objects.get(members=members,
                watchlist=anime_data)
343     except:
344         watchlist = False
345     return render_to_response('anime_details.html',locals(),
            context_instance=RequestContext(request))
```

```
346
347 def manga_details(request,slug):
348     manga_data = Manga.objects.get(slug=slug)
349     commentmangas = CommentManga.objects.filter(manga=
        manga_data).annotate(likes=Count('votes')).order_by('
        -likes','-edited','-added')
350     try:
351     members = Members.objects.get(user=request.user)
352     except:
353         pass
354     try:
355         Daisukilist.objects.get(manga=manga_data, members=
            members)
356         daisuki = True
357     except:
358         daisuki = False
359     try:
360
361         Daikirailist.objects.get(manga=manga_data, members=
            members)
362         daikirai = True
363     except:
364         daikirai = False
365     try:
366         watchlist=ReadingList.objects.get(members=members,
            readinglist=manga_data)
367     except:
368         watchlist = False
369
370     return render_to_response('manga_details.html',locals(),
        context_instance=RequestContext(request))
371
372
373 def anime_character(request,slug,id):
374     anime = Anime.objects.get(slug=slug)
375     character = Character.objects.get(id=id)
376     try:
377     members = Members.objects.get(user=request.user)
378     except:
379         pass
```

```
380     try:
381         Daisukilist.objects.get(character=character, members
                =members)
382         daisuki = True
383     except:
384         daisuki = False
385     try:
386         Daikirailist.objects.get(character=character,
                members=members)
387         daikirai = True
388     except:
389         daikirai = False
390 
391     return render_to_response('anime_character.html',locals
            (),context_instance=RequestContext(request))
392 
393 def manga_character(request,slug,id):
394     manga = Manga.objects.get(slug=slug)
395     character = CharacterManga.objects.get(id=id)
396     try:
397     members = Members.objects.get(user=request.user)
398     except:
399         pass
400     try:
401         Daisukilist.objects.get(character_manga=character,
                members=members)
402         daisuki = True
403     except:
404         daisuki = False
405     try:
406         Daikirailist.objects.get(character_manga=character,
                members=members)
407         daikirai = True
408     except:
409         daikirai = False
410 
411     return render_to_response('manga_character.html',locals
            (),context_instance=RequestContext(request))
412 
413 def anime_voice(request,id):
```

```
414
415     voice = VoiceCharacter.objects.get(id=id)
416     try:
417     members = Members.objects.get(user=request.user)
418     except:
419         pass
420     try:
421         Daisukilist.objects.get(voice_character=voice,
                members=members)
422         daisuki = True
423     except:
424         daisuki = False
425     return render_to_response('anime_voice.html',locals(),
            context_instance=RequestContext(request))
426
427 def anime_reviews(request,slug):
428     anime_data = Anime.objects.get(slug=slug)
429     try:
430     member = Members.objects.get(user=request.user)
431     except:
432     pass
433     commentanimes = CommentAnime.objects.filter(anime=
            anime_data).annotate(likes=Count('votes')).order_by('
            -likes','-edited','-added')
434
435
436     return render_to_response('anime_reviews.html',locals(),
            context_instance=RequestContext(request))
437
438 @login_required
439 def anime_reviews_add(request,slug):
440     anime_data = Anime.objects.get(slug=slug)
441     member = Members.objects.get(user=request.user)
442
443
444     if request.method == "POST":
445     comment=CommentAnime.objects.create(comment=request.POST
            .get('comment'), user=member, anime=anime_data)
446     comment.save()
```

```
447     return HttpResponseRedirect(reverse('anime_reviews',
            kwargs={'slug':anime_data.slug}))
448
449     return render_to_response('anime_reviews_add.html',
            locals(),context_instance=RequestContext(request))
450
451
452 def manga_reviews(request,slug):
453     manga_data = Manga.objects.get(slug=slug)
454     try:
455     member = Members.objects.get(user=request.user)
456     except:
457     pass
458     commentmangas = CommentManga.objects.filter(manga=
            manga_data).annotate(likes=Count('votes')).order_by('
            -likes','-edited','-added')
459
460
461
462     return render_to_response('manga_reviews.html',locals(),
            context_instance=RequestContext(request))
463
464
465 @login_required
466 def manga_reviews_add(request,slug):
467     manga_data = Manga.objects.get(slug=slug)
468     member = Members.objects.get(user=request.user)
469
470
471     if request.method == "POST":
472     comment=CommentManga.objects.create(comment=request.POST
            .get('comment'), user=member, manga=manga_data)
473     comment.save()
474     return HttpResponseRedirect(reverse('manga_reviews',
            kwargs={'slug':manga_data.slug}))
475
476     #return render_to_response('manga_reviews_add.html',
            locals(),context_instance=RequestContext(request))
477
478
```

```
479 def anime_chart(request,season):
480     try:
481         anime_data = Chart.objects.get(season=season)
482     except:
483       anime_data =[]
484     return render_to_response('anime_chart.html',locals(),
          context_instance=RequestContext(request))
485
486
487 def users(request,username):
488     if request.user.username == username:
489         return HttpResponseRedirect(reverse('dashboard',
              kwargs={'username': username}))
490     try:
491         user = User.objects.get(username=username)
492     except:
493         return HttpResponseRedirect(reverse('home'))
494     member = Members.objects.get(user=user)
495     return render_to_response('User/dashboard.html',locals()
          ,context_instance=RequestContext(request))
496
497
498 def dashboard(request, username):
499     if request.POST.get('action') == 'saveAbout':
500         countData = About.objects.all().count()
501         aboutField = request.POST.get('about')
502         seq_about = request.POST.get('seq_about')
503         memberId = request.POST.get('member')
504         member = Members.objects.get(id=memberId)
505         aboutData = About()
506         if seq_about is not None:
507             aboutData = About.objects.get(seq_about=
                  seq_about)
508             aboutData.member = member
509             aboutData.about = aboutField
510             aboutData.seq_about = seq_about
511             aboutData.save()
512             return render_to_response('User/dashboard.html',
                  locals(),context_instance=RequestContext(
                  request))
```

```
513             aboutData.member = member
514             aboutData.about = aboutField
515             aboutData.seq_about = countData
516             aboutData.save()
517             return render_to_response('User/dashboard.html',
                    locals(),context_instance=RequestContext(request)
                    )
518
519         if request.user.is_authenticated():
520             member = Members.objects.get(user=request.user)
521             usr = User.objects.get(username=username)
522             if request.user.username != username:
523                 return HttpResponseRedirect(reverse('users',
                        kwargs={'username': username}))
524             about_data = About.objects.filter(member=member.id)
525             return render_to_response('User/dashboard.html',
                    locals(),context_instance=RequestContext(request)
                    )
526         return HttpResponseRedirect(reverse('home'))
527
528 def library(request,username):
529
530
531
532             usr = User.objects.get(username=username)
533             member = Members.objects.get(user=usr)
534             bliss = WatchList.objects.filter(members=member,
                    status='Bliss')
535             agony = WatchList.objects.filter(members=member,
                    status='Agony')
536             hope = WatchList.objects.filter(members=member,
                    status='Hope')
537             life = WatchList.objects.filter(members=member,
                    status='Life')
538             host=request.META['HTTP_HOST']
539             about_data = About.objects.filter(member=member.id)
540             return render_to_response('User/library.html',locals
                    (),context_instance=RequestContext(request))
541
542
```

```
543 def library_manga(request,username):
544
545
546
547         usr = User.objects.get(username=username)
548         member = Members.objects.get(user=usr)
549         bliss = ReadingList.objects.filter(members=member,
                status='Bliss')
550         agony = ReadingList.objects.filter(members=member,
                status='Agony')
551         hope = ReadingList.objects.filter(members=member,
                status='Hope')
552         life = ReadingList.objects.filter(members=member,
                status='Life')
553         host=request.META['HTTP_HOST']
554
555         about_data = About.objects.filter(member=member.id)
556         return render_to_response('User/library-manga.html',
                locals(),context_instance=RequestContext(request)
                )
557
558
559
560 def daikirai(request):
561     #Search Function
562
563     if request.method=="POST":
564
565         username = request.POST.get('username')
566         user = User.objects.get(username=username)
567         members = Members.objects.get(user=user)
568
569         try:
570             daikirai = Daikirailist.objects.get(members=
                    members)
571         except:
572             daikirai = Daikirailist.objects.create(members=
                    members)
573         try:
```

```
574             daisuki = Daisukilist.objects.get(members=
                    members)
575         except:
576             daisuki = Daisukilist.objects.create(members=
                    members)
577 
578         if request.POST.get('anime_id'):
579       anime_id = request.POST.get('anime_id')
580       anime = Anime.objects.get(id=int(anime_id))
581       daikirai.anime.add(anime)
582       daisuki.anime.remove(anime)
583         elif request.POST.get('manga_id'):
584             manga_id = request.POST.get('manga_id')
585             manga = Manga.objects.get(id=int(manga_id))
586             daikirai.manga.add(manga)
587             daisuki.manga.remove(manga)
588         elif request.POST.get('character_id'):
589             character_id = request.POST.get('character_id')
590             character = Character.objects.get(id=int(
                    character_id))
591             daikirai.character.add(character)
592             daisuki.character.remove(character)
593         elif request.POST.get('character_manga_id'):
594             character_manga_id = request.POST.get('
                    character_manga_id')
595             character_manga = CharacterManga.objects.get(id=
                    int(character_manga_id))
596             daikirai.character_manga.add(character_manga)
597             daisuki.character_manga.remove(character_manga)
598 
599         daikirai.save()
600         daisuki.save()
601 
602 
603         return HttpResponse("success")
604 
605 def daisuki(request):
606     #Search Function
607 
608     if request.method=="POST":
```

```
609
610         username = request.POST.get('username')
611         user = User.objects.get(username=username)
612         members = Members.objects.get(user=user)
613
614         try:
615             daikirai = Daikirailist.objects.get(members=
                members)
616         except:
617             daikirai = Daikirailist.objects.create(members=
                members)
618         try:
619             daisuki = Daisukilist.objects.get(members=
                members)
620         except:
621             daisuki = Daisukilist.objects.create(members=
                members)
622
623         if request.POST.get('anime_id'):
624       anime_id = request.POST.get('anime_id')
625       anime = Anime.objects.get(id=int(anime_id))
626       daisuki.anime.add(anime)
627       daikirai.anime.remove(anime)
628         elif request.POST.get('manga_id'):
629             manga_id = request.POST.get('manga_id')
630             manga = Manga.objects.get(id=int(manga_id))
631             daisuki.manga.add(manga)
632             daikirai.manga.remove(manga)
633         elif request.POST.get('character_id'):
634             character_id = request.POST.get('character_id')
635             character = Character.objects.get(id=int(
                character_id))
636             daisuki.character.add(character)
637             daikirai.character.remove(character)
638         elif request.POST.get('character_manga_id'):
639             character_manga_id = request.POST.get('
                character_manga_id')
640             character_manga = CharacterManga.objects.get(id=
                int(character_manga_id))
641             daisuki.character_manga.add(character_manga)
```

```
642             daikirai.character_manga.remove(character_manga)
643         elif request.POST.get('voice_id'):
644             voice_id = request.POST.get('voice_id')
645             voice = VoiceCharacter.objects.get(id=int(
                    voice_id))
646             daisuki.voice_character.add(voice)
647
648
649         daikirai.save()
650         daisuki.save()
651
652
653         return HttpResponse("success")
654
655 def watchlist(request):
656     #Search Function
657
658     if request.method=="POST":
659
660         username = request.POST.get('username')
661         user = User.objects.get(username=username)
662         members = Members.objects.get(user=user)
663         anime_id = request.POST.get('anime_id')
664         anime = Anime.objects.get(id=int(anime_id))
665
666         try:
667             watchlist = WatchList.objects.get(members=
                    members, watchlist=anime)
668         except:
669             watchlist = WatchList.objects.create(members=
                    members, watchlist=anime)
670
671         watchlist.status=request.POST.get('status')
672
673         watchlist.save()
674
675         return HttpResponse("success")
676
677 def readinglist(request):
678     #Search Function
```

```
679
680     if request.method=="POST":
681
682         username = request.POST.get('username')
683         user = User.objects.get(username=username)
684         members = Members.objects.get(user=user)
685         manga_id = request.POST.get('manga_id')
686         manga = Manga.objects.get(id=int(manga_id))
687
688         try:
689             readinglist = ReadingList.objects.get(members=
                members,readinglist=manga)
690         except:
691             readinglist = ReadingList.objects.create(members
                =members, readinglist=manga)
692
693         readinglist.status=request.POST.get('status')
694
695         readinglist.save()
696
697         return HttpResponse("success")
698
699 def ratings_average(request, entity, id):
700
701     if entity == 'anime':
702         entities = Anime.objects.get(id=int(id))
703     elif entity == 'manga':
704         entities = Manga.objects.get(id=int(id))
705     elif entity == 'character':
706         entities = Character.objects.get(id=int(id))
707     elif entity == 'character_manga':
708         entities = CharacterManga.objects.get(id=int(id))
709     elif entity == 'voice':
710         entities = VoiceCharacter.objects.get(id=int(id))
711
712     return render_to_response('ratings_average.html',locals
        (),context_instance=RequestContext(request))
713
714 def likes(request, entity):
715
```

```
716     if request.method=="POST":
717         username = request.POST.get('username')
718         id = request.POST.get('id')
719         user = User.objects.get(username=username)
720         members = Members.objects.get(user=user)
721         if entity == 'anime':
722             try:
723                 entities = CommentAnime.objects.get(id=int(
                        id))
724             except:
725                 entities = CommentAnime.objects.create(id=
                        int(id))
726         elif entity == 'manga':
727             try:
728                 entities = CommentManga.objects.get(id=int(
                        id))
729             except:
730                 entities = CommentManga.objects.create(id=
                        int(id))
731
732         voted = entities.votes.exists(user)
733         if voted:
734             entities.votes.down(user)
735         else:
736             entities.votes.up(user)
737         voted = entities.votes.exists(user)
738         total_votes = entities.votes.count()
739
740         json_data = {'voted': voted, 'total_votes':
                total_votes}
741
742         return HttpResponse(simplejson.dumps(json_data),
                content_type="application/json")
743
744 def show_likes(request, entity, id):
745
746     username = request.user
747     user = User.objects.get(username=username)
748     members = Members.objects.get(user=user)
749     print user
```

```
750     if entity == 'anime':
751         entities = CommentAnime.objects.get(id=int(id))
752 
753     elif entity == 'manga':
754         entities = CommentManga.objects.get(id=int(id))
755 
756     voted = entities.votes.exists(user)
757 
758     json_data = {'voted': voted}
759 
760     return HttpResponse(simplejson.dumps(json_data),
            content_type="application/json")
761 
762 
763 def sort_comment(request):
764 
765 
766     if request.method=="POST":
767 
768         by = request.POST.get('by')
769         anime_id = request.POST.get('anime_id')
770         anime_data = Anime.objects.get(id=int(anime_id))
771 
772         if by == 'likes':
773             commentanimes = CommentAnime.objects.filter(
                    anime=anime_data).annotate(likes=Count('votes
                    ')).order_by('-likes','-edited','-added')
774         elif by == 'time':
775       commentanimes = CommentAnime.objects.filter(anime=
            anime_data).order_by('-edited','-added')
776 
777         return render_to_response('sort-comment.html',locals
            (),context_instance=RequestContext(request))
778 
779 def sort_comment_manga(request):
780 
781 
782     if request.method=="POST":
783 
784         by = request.POST.get('by')
```

```
785         manga_id = request.POST.get('manga_id')
786         manga_data = Manga.objects.get(id=int(manga_id))
787 
788         if by == 'likes':
789             commentmangas = CommentManga.objects.filter(
                    manga=manga_data).annotate(likes=Count('votes
                    ')).order_by('-likes','-edited','-added')
790         elif by == 'time':
791       commentmangas = CommentManga.objects.filter(manga=
            manga_data).order_by('edited','added')
792 
793         return render_to_response('sort-comment-manga.html',
              locals(),context_instance=RequestContext(request)
              )
794 
795 def search_comment(request):
796 
797     if request.method=="POST":
798 
799         if request.POST.get('anime_id'):
800       anime_id = request.POST.get('anime_id')
801       search_val = request.POST.get('search')
802       anime_data = Anime.objects.get(id=int(anime_id))
803       commentanimes = CommentAnime.objects.filter(anime=
            anime_data, comment__icontains=search_val).order_by
            ('-edited','-added')
804       if not commentanimes:
805         return HttpResponse("<h4>Sorry no data</h4>")
806       return render_to_response('sort-comment.html',locals()
            ,context_instance=RequestContext(request))
807         elif request.POST.get('manga_id'):
808       manga_id = request.POST.get('manga_id')
809       search_val = request.POST.get('search')
810       manga_data = Manga.objects.get(id=int(manga_id))
811       commentmangas = CommentManga.objects.filter(manga=
            manga_data, comment__icontains=search_val).order_by
            ('-edited','-added')
812       if not commentmangas:
813         return HttpResponse("<h4>Sorry no data</h4>")
```

```
814         return render_to_response('sort-comment-manga.html',
                locals(),context_instance=RequestContext(request))
815
816
817 def request_edit(request):
818
819     if request.method=="POST":
820
821         if request.POST.get('anime_id'):
822         anime_id = request.POST.get('anime_id')
823         synopsis = request.POST.get('synopsis')
824         start = request.POST.get('synopsis')
825         end = request.POST.get('end')
826         content = "synopsis: {} \nstart: {} \nend: {}".format(
                synopsis, start, end)
827         anime = Anime.objects.get(id=int(anime_id))
828         username = request.POST.get('username')
829         user = User.objects.get(username=username)
830         members = Members.objects.get(user=user)
831         requestedit= RequestAnime.objects.create(content=
                content, user=members, anime=anime)
832         requestedit.save()
833         return render_to_response('sort-comment.html',locals()
                ,context_instance=RequestContext(request))
```

```
1 anime.html
2
3 {% extends 'base_menu.html' %}
4
5 {% block css %}
6     <link href="{{ MEDIA_URL }}css/nouislider.css" rel="
          stylesheet">
7     <link href="{{ MEDIA_URL }}css/nouislider.pips.css" rel=
          "stylesheet">
8     <link href="{{ MEDIA_URL }}css/bootstrap-reset.css" rel=
          "stylesheet">
9
10    <link href="{{ MEDIA_URL }}css/bootstrap.min.css" rel="
          stylesheet">
11    <style>
```

```
12          .anime-title:after {
13              content: '';
14              position: absolute;
15              left: 15px;
16              bottom: 0px;
17              width: 70px;
18              height: 4px;
19              background: #11c3f0;
20          }
21      </style>
22
23  {% endblock %}
24  {% block js %}
25
26      <script>
27          $(document).ready(function(){
28              $('[data-toggle="tooltip"]').tooltip();
29          });
30          function reloadajax(varpage,id){
31              $.ajax({
32                  type:"GET",
33                  url:"{% url 'anime' %}",
34                  data : {
35                      'pages_ajax': 'page_ajax',
36                      'page': varpage,
37                      'id':id
38                  },
39                  success:function(response){
40                      $('#anime_data').html(response);
41                  }
42              });
43          }
44          function search(){
45              search_value = $('input[name=search]').val();
46              $.ajax({
47                  type: "POST",
48                  cache: false,
49                  url: "{% url 'anime' %}",
50                  data: {
51                      'action': 'search',
```

```
52                      'search_value': search_value,
53                      'csrfmiddlewaretoken': '{{csrf_token}}'
54                  },
55                  success:function(response){
56                      $('#anime_data').html(response);
57                  }
58              });
59          }
60
61          function checkbox(){
62              var anime_check = [];
63              $('input[name=animes_check]:checked').each(
                    function(){
64                  anime_check.push($(this).val());
65              });
66              $.ajax({
67                  type: "POST",
68                  cache: false,
69                  url: "{% url 'anime' %}",
70                  data: {
71                      'action': 'checkbox',
72                      'anime_check': JSON.stringify(
                            anime_check),
73                      'csrfmiddlewaretoken': '{{csrf_token}}'
74                  },
75                  success:function(response){
76                      $('#anime_data').html(response);
77                  }
78              });
79          }
80      </script>
81  {% endblock %}
82
83  {% block body%}
84
85      <div class="sidebar-page shop-page">
86
87          <div class="auto-container">
88
89              <div class="row clearfix">
```

```
90
91                  <!--Left Content-->
92                 <div class="col-lg-9 col-md-8 col-sm-6 col-xs
                       -12 pull-right">
93
94                   <!--Products Section-->
95                     <section class="products-section">
96
97                         <div class="shop-filters category-
                               filters">
98                          <div class="clearfix">
99                              <!--Filter Outer-->
100                             <div class="col-lg-10 col-md
                                    -6 col-sm-6 col-xs-12
                                    pull-left">
101                              <div class="anime-title"
                                     >
102                               <h3>Anime</h3>
103                              </div>
104                             </div>
105                             <!--<div class="col-lg-2 col
                                    -md-6 col-sm-6 col-xs-12
                                    pull-right pull-bottom">
106                              <div class="btn-group"
                                     style="margin-top:20
                                     px;">
107                                   <button type="
                                          button" class
                                          ="btn btn-
                                          default
                                          filter-btn
                                          dropdown-
                                          toggle" data-
                                          toggle="
                                          dropdown"
                                          aria-haspopup
                                          ="true" aria-
                                          expanded="
                                          false">
                                          GENDER :  <
```

```
                                        span class="
                                        filter-val">
                                        MALE</span> &
                                        ensp;<span
                                        class="fa fa-
                                        angle-down"
                                        ></span></
                                        button>
108                                 <ul class="
                                        dropdown-menu
                                        " id="
                                        dropDownId">
109                                     <li><a href=
                                            "#">Male
                                            </a></li>
110                                     <li><a href=
                                            "#">
                                            Female</a
                                            ></li>
111                                     <li><a href=
                                            "#">All</
                                            a></li>
112                                 </ul>
113                             </div>
114                         </div>-->
115
116                     </div>
117                 </div>
118                 <div id="anime_data">
119                     <div class="row clearfix">
120                         {% for i in anime_data %}
121
122                         <!--Column-->
123                         <article class="product-item
                                col-lg-2 col-md-6 col-sm
                               -12 col-xs-12">
124                             <div class="inner-box
                                tooltips" style="
                                width:129;height:187
                                px;"
```

```
125                                                 data-toggle="
                                                        tooltip" data-
                                                        placement="top"
                                                        title="{{i.
                                                        a_name}}">
126                                                 <figure class="image
                                                        " style="border-
                                                        radius:5px;"><a
                                                        href="{% url '
                                                        anime_details'
                                                        slug=i.slug %}"><
                                                        img
127                                                         src="{{
                                                                MEDIA_URL
                                                                }}{{i.
                                                                a_displaypic
                                                                }}" alt="
                                                                "></a>
128                                                     <div class="
                                                            product-
                                                            content"
129                                                           style="
                                                                white-
                                                                space:
                                                                nowrap;
                                                                text-
                                                                overflow
                                                                :
                                                                ellipsis
                                                                ;
                                                                overflow
                                                                : hidden
                                                                ;">
130                                                             <a href="{%
                                                                url '
                                                                anime_details
                                                                ' slug=i.
                                                                slug %}"
                                                                >{{i.
                                                                a_name
```

```
                                                    }}</a></
                                                    div>
131                                     </figure>
132
133
134                                 </div>
135                             </article>
136                             {% endfor %}
137
138                         </div>
139
140                         <!-- Centered Pagination -->
141                         <div class="centered-pagination
                                text-left">
142                             <ul>
143                                 {% for i in paginator.
                                        page_range %}
144                                     <li><a {% if i ==
                                            number %}class="
                                            active"{% endif%}
                                             onclick="
                                            reloadajax('{{i
                                            }}')" href="#">{{
                                            i}}</a></li>
145                                 {% endfor %}
146                                 {% if current_page.
                                        has_next %}
147                                     <li><a onclick="
                                            reloadajax('{{
                                            current_page.
                                            next_page_number
                                            }}')" href="#">&
                                            raquo;</a></li>
148                                 {% else %}
149                                     <li class="disabled"
                                            ><a href="#">&
                                            raquo;</a> </li>
150                                 {% endif %}
151                                 <!--<a href="#" class="
                                        active">1</a></li>
```

```
152                                 <li><a href="#">2</a></
                                        li>
153                                 <li><a href="#">3</a></
                                        li>
154                                 <li><a href="#" >.... </a
                                        ></li>
155                                 <li><a href="#">10</a></
                                        li>
156                                 <li><a href="#">11</a></
                                        li>
157                                 <li><a href="#">12</a></
                                        li>-->
158
159                             </ul>
160                         </div>
161                     </div>
162
163                 </section>
164
165             </div><!--End Left Content-->
166
167
168             <!--Sidebar-->
169             <div class="col-lg-3 col-md-4 col-sm-6 col-
                    xs-12 pull-left">
170                 <aside class="sidebar">
171
172                     <!-- Search Form -->
173                     <div class="widget search-form">
174                             <div class="form-group">
175                                 <input type="search"
                                        name="search"
                                        placeholder="search
                                        for something">
176                                 <button onclick="search
                                        ()" name="submit"><
                                        span class="fa fa-
                                        search"></span></
                                        button>
177                             </div>
```

```
178
179                         </div>
180
181                         <!-- Choose Brand -->
182                         <div class="widget choose-brand
                                checkbox-filters two-column wow
                                fadeInUp" data-wow-delay="0ms"
                                data-wow-duration="1500ms">
183                              <div class="sidebar-title"><h3>
                                     Genre</h3></div>
184                              <div class="row clearfix">
185                                   {% for i in genre %}
186                                   <div class="col-lg-6 col-md
                                          -12 col-sm-12 column">
187                                        <div class="check-item"
                                             ><input onclick="
                                             checkbox()" name="
                                             animes_check" type="
                                             checkbox" id="{{i.
                                             genre_type}}" value="
                                             {{i.id}}"><label
188                                                  class="checkbox-
                                                     label" for="
                                                     {{i.
                                                     genre_type}}"
                                                      style="white
                                                     -space:
                                                     nowrap;text-
                                                     overflow:
                                                     ellipsis;
                                                     overflow:
                                                     hidden;"><
                                                     span class="
                                                     icon-box"><
                                                     span
189                                                  class="uncheck"
                                                     ></span> <
                                                     span class="
                                                     fa fa-check
                                                     check"></span
```

```
                                                    ></span>{{i.
                                                    genre_type
                                                    }}</label>
                                        </div>

                                    </div>
                                    {% endfor %}
                                </div>

                            </div>

                        </aside>


                    </div>
                    <!--Sidebar-->

                </div>
            </div>

    </div>

{% endblock %}
```

```
anime_details.html

{% extends 'base_menu.html' %}
{% load ratings_tags %}

{% block css %}

    <link href="{{ MEDIA_URL }}css/style2.css" rel="
        stylesheet">
    <link href="{{ MEDIA_URL }}css/bootstrap-reset.css" rel=
        "stylesheet">


    <link href="{{ MEDIA_URL }}css/style-responsive.css" rel
        ="stylesheet" />
```

```
13      <link href="{{ MEDIA_URL }}css/anime-style.css" rel="
            stylesheet" />
14
15  {% endblock %}
16  {% block js %}
17  <script type="text/javascript">
18
19          function daikirai(anime_id){
20
21              $.ajax({
22                  type: "POST",
23                  cache: false,
24                  url: "{% url 'daikirai' %}",
25                  data: {
26                      'username': "{{request.user.username}}",
27                      'anime_id': anime_id,
28                      'csrfmiddlewaretoken': '{{csrf_token}}'
29                  },
30                  success:function(response){
31                      $('#daikirai').hide();
32                      $('#daisuki').show();
33                      $.notify("Added to your daikirailist", "
                            danger");
34                      setTimeout('RedirectUpdate()', 2000);
35                  }
36              });
37          }
38
39      function daisuki(anime_id){
40
41              $.ajax({
42                  type: "POST",
43                  cache: false,
44                  url: "{% url 'daisuki' %}",
45                  data: {
46                      'username': "{{request.user.username}}",
47                      'anime_id': anime_id,
48                      'csrfmiddlewaretoken': '{{csrf_token}}'
49                  },
50                  success:function(response){
```

```
51                      $('#daisuki').hide();
52                      $('#daikirai').show();
53                      $.notify("Added to your daisukilist", "
                            info");
54                      setTimeout('RedirectUpdate()', 2000);
55                  }
56              });
57          }
58
59          function watchlist(anime_id, status){
60
61              $.ajax({
62                  type: "POST",
63                  cache: false,
64                  url: "{% url 'watchlist' %}",
65                  data: {
66                      'username': "{{request.user.username}}",
67                      'anime_id': anime_id,
68                      'status': status,
69                      'csrfmiddlewaretoken': '{{csrf_token}}'
70                  },
71                  success:function(response){
72                      $('#watchlist').html(status+'<span class
                            ="caret"></span>');
73
74                      $.notify("Added to your watchlist as "+
                            status, "default");
75                      setTimeout('RedirectUpdate()', 2000);
76                  }
77              });
78          }
79
80
81
82      </script>
83  {% endblock %}
84
85  {% block body%}
86      <!--Main Slider-->
87      {% with anime_data as i %}
```

```
88          <section class="page-title anime-cover" style="
                background-image: url({{ MEDIA_URL }}{{i.a_cover
                }});">
89              {% if request.user.is_authenticated %}
90              <div class="auto-container">
91                  <div class="row clearfix">
92                      <div class="col-lg-6 col-xs-12" style="
                            position:absolute;bottom:30px;">
93                          <div style="top:5px; padding-left:15
                                px;">
94                              <button class="btn" data-toggle=
                                    "modal" data-target="#edit-
                                    page" style="opacity:0.8;
                                    color:white;">Edit this page
                                    </button>
95                          </div>
96                      </div>
97                  </div>
98              </div>
99              {% endif %}
100         </section>
101
102         <!-- Modal -->
103         <div class="modal fade" id="edit-page" tabindex="-1"
                role="dialog" aria-labelledby="myModalLabel">
104           <div class="modal-dialog" role="document">
105             <div class="modal-content">
106               <div class="modal-header">
107                 <button type="button" class="close" data-
                        dismiss="modal" aria-label="Close"><span
                        aria-hidden="true">&times;</span></button
                        >
108                 <h4 class="modal-title" id="myModalLabel">
                        Edit Anime</h4>
109               </div>
110
111               <div class="modal-body">
112                 Synopsis
113                 <textarea id="synopsis-edit" class="form-
                        control" rows="7">{{i.a_synopsys}}</
```

```
                    textarea>
114                 <br>
115                 <div class="form-inline">
116                   <div class="form-group">
117                     <label for="exampleInputName2">Aired
                            from </label>
118                     <input id="start-edit" type="text" value
                            ="{{i.a_airedstart|date:'
                            SHORT_DATE_FORMAT'}} " class="form-
                            control" id="exampleInputName2"
                            placeholder="Jane Doe">
119                   </div>
120                   <div class="form-group">
121                     <label for="exampleInputEmail2">to</
                            label>
122                     <input id="end-edit" type="text" value="
                            {{i.a_airedend|date:'
                            SHORT_DATE_FORMAT'}}" class="form-
                            control" id="exampleInputEmail2"
                            placeholder="jane.doe@example.com">
123                   </div>
124                 </div>
125
126
127
128               </div>
129               <div class="modal-footer">
130                 <button id="request-edit" type="button"
                        class="btn btn-default" data-dismiss="
                        modal">Request edit</button>
131               </div>
132             </div>
133           </div>
134         </div>
135
136
137     <div class="sidebar-page">
138         <div class="auto-container">
139             <div class="row clearfix" style="margin-top:-20
                    px;">
```

```
140                 <div class="col-lg-8 col-md-8 col-sm-12 col-
                        xs-12" id="formAnimes">
141                     <div class="row">
142                         <div class="col-md-12">
143                             <section class="panel">
144                                 <div class="panel-body">
145                                     <div class="col-md-4 col
                                            -sm-6 col-xs-12"
                                            style="margin:-5px 0
                                            0 -20px;">
146                                         <!--Image-->
147                                         <figure class="image
                                                ">
148                                             <a href="#"
                                                    class="
                                                    lightbox-
                                                    image"
149                                                 title="{{i.
                                                        a_name}}"
                                                    ><img src=
                                                    "{{
                                                    MEDIA_URL
                                                    }}{{i.
                                                    a_displaypic
                                                    }}" alt=""
                                                     width="
                                                    215"
                                                    height="
                                                    311" style
                                                    ="border-
                                                    radius:5px
                                                    ;"></a>
150                                         </figure>
151                                     </div>
152
153                                     <div class="col-md-8 col
                                            -sm-6 col-xs-12">
154                                         <!--Content-Column
                                                -->
```

```
155                                 <div class="cont-
                                        column">
156                                     <div class="
                                            inner-box">
157                                         <h3 class="
                                                anime-
                                                title"><a
                                                href="#"
                                                >{{i.
                                                a_name}}</
                                                a></h3>
158
159                                     <div class="item-
                                            cats subtitle"
                                            >Aired from {{
                                            i.a_airedstart
                                            }} to {{i.
                                            a_airedend}}
                                        </div>
160                                     <div class="
                                            description
                                            anime-desc" >
161                                         {{i.
                                                a_synopsys
                                                }}
162                                     </div>
163                                     <div class="
                                            sidebar">
164                                         <div class="
                                                popular-
                                                tags">
165                                             <div class
                                                    ="
                                                    sidebar
                                                    -title"
                                                    >
166                                                 <h2
                                                        class
                                                        ="
                                                        anime
```

```
                                        -tags
                                        ">
                                        Genres
                                        </h2>
168                                   </div>
169                                   <div
                                          class
                                          ="
                                          tags"
                                          >
170                                   {% for
                                          genre
                                           in i
                                          .
                                          a_genre
                                          .all
                                          %}
171                                     <a
                                            href
                                            ="#
                                            "
                                            >{{
                                            genre
                                            .
                                            genre_type
                                            }}</
                                            a>
172                                   {%
                                          endfor
                                           %}
173                                   </div>
174                                   </div>
175                                 </div>
176                               {% if request.
                                  user.
                                  is_authenticated
                                   %}
177
178                 {% get_rating_form for i as
                       rating_form %}
```

```
179                     <form id="rate-form">
180                       {% csrf_token %}
181                       {{ rating_form }}
182
183                     </form>
184                     <div id="ratings-average">
185                       {% get_rating_score for i as score
                              %}
186                       {% if score %}
187                         Average ratings: {{ score.
                              average|floatformat }}<br>
188                         Number of votes: {{ score.
                              num_votes }}<br>
189                       {% else %}
190                         How sad: nobody voted {{ film
                              }}<br>
191                       {% endif %}
192                     </div>
193
194
195                     <div class="btn-group">
196                      <button id="watchlist" data-toggle=
                           "dropdown" class="btn btn-
                           primary dropdown-toggle" type="
                           button">
197                           {% if watchlist %} {{watchlist.
                                status}} {% else %}Add to
                                Watchlist {% endif %} <span
                                class="caret"></span></button
                                >
198                       <ul role="menu" class="dropdown-
                            menu">
199                         <li><a href="#watchlist" onclick
                              ="watchlist({{i.id}},'Bliss')
                              ">Bliss</a></li>
200                         <li><a href="#watchlist" onclick
                              ="watchlist({{i.id}},'Life')"
                              >Life</a></li>
201                         <li><a href="#watchlist" onclick
                              ="watchlist({{i.id}},'Hope')"
```

```
                    >Hope</a></li>
202                 <li><a href="#watchlist" onclick
                        ="watchlist({{i.id}},'Agony')
                        ">Agony</a></li>
203               </ul>
204             </div><!--btn-group-->
205             <div class="btn-group">
206               <button {% if daisuki %} style="
                      display:none;" {% endif %}  id=
                      "daisuki" class="btn btn-info"
                      onclick="daisuki({{i.id}})"
                      type="button">
207                 Add to Daisukilist </button>
208             </div><!--btn-group-->
209
210             <div id="daikirai" class="btn-group"
                    >
211               <button {% if daikirai %} style="
                      display:none;" {% endif %}
                      class="btn btn-danger" onclick=
                      "daikirai({{i.id}})" type="
                      button">
212                 Add to Daikirailist </button>
213             </div><!--btn-group-->
214           {% endif %}
215
216                                 </div>
217                             </div>
218                         </div>
219
220                     </div>
221                 </section>
222             </div>
223           </div>
224           <div class="row">
225               <div class="col-md-12">
226                   <section class="panel">
227                       <div class="panel-body">
228       <div class="cont-column">
229         <div class="inner-box">
```

```
230                     <h3>Cast</h3><hr/>
231                     <div class="row">
232
233                                             {% for
                                                   character
                                                    in i.
                                                   character_set
                                                   .all %}
234                         {% for voice in character.
                               voicecharacter_set.all %}
235                         <div class="col-md-4">
236                           <strong>
237                             <a href ="{% url 'anime_voice'
                                    id=voice.id %}">
238                               {{voice}}
239                             </a>
240                           </strong>
241                           <br> as <a href ="{% url '
                                anime_character' slug=i.slug
                                id=character.id %}">
242                           {{character.name}}</a>
243                           {% if character.picture %}
244                             <img class="media-object
                                  avatar avatar-sm" src="{{
                                  MEDIA_URL }}{{character.
                                  picture}}" alt="">
245                           {% else %}
246                             <img class="media-object
                                  avatar avatar-sm" src="{{
                                  MEDIA_URL }}MembersPicture/
                                  pic.png" alt="">
247                           {% endif %}
248                         </div>
249                         {% empty %}
250                         <div class="col-md-4">
251                           <a href ="{% url '
                                anime_character' slug=i.slug
                                id=character.id %}">
252                           {{character.name}}</a>
253                           {% if character.picture %}
```

```
254                             <img class="media-object
                                    avatar avatar-sm" src="{{
                                    MEDIA_URL }}{{character.
                                    picture}}" alt="">
255                           {% else %}
256                             <img class="media-object
                                    avatar avatar-sm" src="{{
                                    MEDIA_URL }}MembersPicture/
                                    pic.png" alt="">
257                           {% endif %}
258
259                         </div>
260                         {% endfor %}
261                       {%endfor%}
262                                       </div>
263                                   </div>
264                               </div>
265                           </section>
266                       </div>
267                   </div>
268               </div>
269
270               <div class="col-lg-4 col-md-4 col-sm-12 col-
                      xs-12">
271                   <section class="panel panel-default
                          post-comments">
272                       <div class="panel-heading">
273                 Top Latest Comments
274                 <div class="pull-right">
275                   <a href="{% url 'anime_reviews' slug=i.slug
                        %}">View All </a>
276
277                       </div>
278                   </div>
279                   <div id="panel-comments" class="
                          panel-body">
280                 {% if commentanimes %}
281                 {% for review in commentanimes|slice:":3" %}
282                 <div class="media p15">
```

```
283                 <a class="pull-left" href="{% url 'users'
                        username=review.user.user.username %}">
284                   {% if review.user.m_picture %}
285                     <img class="media-object avatar avatar-
                            sm" src="{{ MEDIA_URL }}{{review.user
                            .m_picture}}" alt="">
286                   {% else %}
287                     <img class="media-object avatar avatar-
                            sm" src="{{ MEDIA_URL }}
                            MembersPicture/pic.png" alt="">
288                   {% endif %}
289                 </a>
290
291                 <div class="comment">
292                   <div class="comment-author h6 no-m">
293                     <a href="{% url 'users' username=review.
                            user.user.username %}"><b>{{review.
                            user.user.username}}</b></a>
294                   </div>
295                   <div class="comment-meta small">
296                     {{review.added}}
297                   </div>
298                   <p>
299                     {{review.comment}}
300                   </p>
301
302                   <p class="small">
303                     <a id="heart-{{review.id}}" class="mr10"
                            ><i class="fa fa-heart-o mr5"></i><
                            span id="like-{{review.id}}">{{review
                            .votes.count}} </span> Likes</a>
304                   </p>
305                   <script>
306
307                     $.get("{% url 'show_likes' entity='
                            anime' id=review.id %}", function(
                            data, status){
308
309                       if (data.voted){
```

```
310                         $('#heart-{{review.id}}').addClass
                                ('text-danger');
311                       }
312                     });
313
314                 </script>
315                 <hr>
316               </div>
317             </div>
318             {% endfor %}
319                         {% endif %}
320
321             {% if request.user.is_authenticated %}
322               <form method="POST" action="{% url '
                      anime_reviews_add' slug=i.slug %}">
323                                 <div class="form-group">
324                  {% csrf_token %}
325                                     <textarea class="
                                        form-control"
                                        rows="3"
                                        placeholder="Add
                                        Comment..." name=
                                        "comment"></
                                        textarea>
326                                 </div>
327                                 <button type="submit"
                                        class="btn btn-
                                        primary">POST</button
                                        >
328                             </form>
329                         {% endif %}
330                     </div>
331
332
333
334                 </section>
335
336               </div>
337             </div>
338         </div>
```

```
339     </div>
340
341
342   <script type="text/javascript">
343         $(document).ready(function() {
344       $('#id_score').addClass('rating');
345       $('#id_score').attr('type', 'hidden');
346     });
347
348      $("#rate-form").click(function() {
349                 rate();
350           });
351
352           function rate(){
353
354             $.ajax({
355                 type: "POST",
356                 cache: false,
357                 url: "{% url 'ratings_vote' %}",
358                 data: {
359           'honeypot': $('#id_honeypot').val(),
360           'security_hash': $('#id_security_hash').val(),
361           'timestamp': $('#id_timestamp').val(),
362           'key': $('#id_key').val(),
363           'object_pk': $('#id_object_pk').val(),
364           'content_type': $('#id_content_type').val(),
365           'score': $('#id_score').val(),
366                     'csrfmiddlewaretoken': '{{csrf_token}}'
367                 },
368                 success:function(response){
369           $.get("{% url 'ratings_average' entity='anime' id=
                 i.id %}", function(data, status){
370
371             $('#ratings-average').html(data)
372           });
373                     $.notify("Rated "+ status, "default");
374                     setTimeout('RedirectUpdate()', 2000);
375                 }
376             });
377         }
```

```
378
379         $("#panel-comments").on('click', 'a[id^=heart-]',
                function(){
380       var id = $(this).attr('id').split('-')[1];
381       $.ajax({
382                 type: "POST",
383                 cache: false,
384                 url: "{% url 'likes' entity='anime' %}",
385                 data: {
386           'username': '{{request.user}}',
387           'id': id,
388                     'csrfmiddlewaretoken': '{{csrf_token}}'
389                 },
390                 success:function(response){
391           voted=response.voted
392                     $('#like-'+id).text(response.total_votes
                          );
393                     if (voted){
394             $('#heart-'+id).addClass('text-danger');
395             $.notify("Liked", "info");
396           }
397           else{
398             $('#heart-'+id).removeClass('text-danger');
399             $.notify("Unliked", "default");
400           }
401
402                     setTimeout('RedirectUpdate()', 2000);
403
404                 }
405           });
406
407     });
408
409         $("#request-edit").click(function() {
410             synopsis=$('#synopsis-edit').val();
411             start=$('#start-edit').val();
412             end=$('#end-edit').val();
413             $.ajax({
414                 type: "POST",
415                 cache: false,
```

```
416                 url: "{% url 'request_edit' %}",
417                 data: {
418             'username': '{{request.user}}',
419             'anime_id': '{{i.id}}',
420                         'synopsis': synopsis,
421                         'start': start,
422                         'end': end,
423                         'csrfmiddlewaretoken': '{{csrf_token}}'
424                 },
425                 success:function(response){
426             $.notify("Request sent", "default");
427                         setTimeout('RedirectUpdate()', 2000);
428                 }
429               });
430
431             });
432   </script>
433   {% endwith %}
434 {% endblock %}
```

index.html

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8">
5      <title>Animetacchi</title>
6      <!-- Stylesheets -->
7      <link href="{{ MEDIA_URL }}css/bootstrap.css" rel="
           stylesheet">
8      <link href="{{ MEDIA_URL }}css/revolution-slider.css"
           rel="stylesheet">
9      <link href="{{ MEDIA_URL }}css/style.css" rel="
           stylesheet">
10     <!-- Responsive -->
11     <meta http-equiv="X-UA-Compatible" content="IE=edge">
12     <meta name="viewport" content="width=device-width,
           initial-scale=1.0, maximum-scale=1.0, user-scalable=0
           ">
```

```
15      <link href="{{ MEDIA_URL }}css/responsive.css" rel="
            stylesheet">
16      <!--[if lt IE 9]>
17      <script src="http://html5shim.googlecode.com/svn/trunk/
            html5.js"></script><![endif]-->
18      <!--[if lt IE 9]>
19      <script src="{{ MEDIA_URL }}js/respond.js"></script><![
            endif]-->
20      <!-- Jquery Js -->
21      <script src="{{ MEDIA_URL }}js/jquery.min.js"></script>
22      <!-- ReadMore Js -->
23      <script src="{{ MEDIA_URL }}js/read-more.js"></script>
24      <style>
25          .morecontent span {
26              display: none;
27          }
28          .more-link {
29              display: block;
30          }
31      </style>
32  </head>
33
34  <body class="theme-orange">
35  <div class="page-wrapper">
36
37      <!-- Preloader -->
38      <div class="preloader"></div>
39
40      <!-- Main Header / Sticky Header-->
41      <header class="main-header sticky-header">
42
43          <!-- Header Lower -->
44          <div class="header-lower">
45
46                  <!--Outer Box-->
47                  <div class="outer-box clearfix">
48
49                      <!-- Logo -->
50                      <div style="position:absolute;padding-
                            top:28px;">
```

```
51                          <a href="{% url 'home' %}"><span><h3
                                style="color:#fff;line-height:1
                                em;">Animetacchi</h3></span></a>
52                      </div>
53
54                      <!--Search Btn
55                      <div class="search-box-btn"><span class=
                            "icon fa fa-search"></span></div>
56              -->
57                      <!-- Main Menu -->
58                      <nav class="main-menu">
59
60                          <div class="navbar-collapse collapse
                                clearfix">
61                          {% include 'base/navigation.html
                                ' %}
62                          </div>
63                      </nav><!-- Main Menu End-->
64
65                  </div>
66          </div><!-- Header Lower End-->
67
68      </header><!--End Main Header -->
69
70      <!--Main Slider-->
71      <section class="main-slider">
72
73          <div class="tp-banner-container">
74              <div class="tp-banner">
75                  <ul>
76
77                          <li data-transition="fade" data-
                                slotamount="1" data-masterspeed="
                                1000"
78                              data-thumb="{{ MEDIA_URL }}
                                    IndexCover/dragon-ball-z.jpg"
                                     data-saveperformance="off"
79                              data-title="Awesome Title Here">
80                              <img src="{{ MEDIA_URL }}
                                    IndexCover/dragon-ball-z.jpg"
```

```
                                alt="" data-bgposition="
                                center top"
81                                data-bgfit="cover" data-
                                    bgrepeat="no-repeat">
82
83
84                          </li>
85
86                          <li data-transition="slideup" data-
                                slotamount="1" data-masterspeed="
                                1000"
87                              data-thumb="{{ MEDIA_URL }}
                                    IndexCover/op.jpg" data-
                                    saveperformance="off"
88                              data-title="Awesome Title Here">
89                              <img src="{{ MEDIA_URL }}
                                    IndexCover/op.jpg" alt=""
                                    data-bgposition="center top"
90                                  data-bgfit="cover" data-
                                        bgrepeat="no-repeat">
91
92
93                          </li>
94                          <li data-transition="slideup" data-
                                slotamount="1" data-masterspeed="
                                1000"
95                              data-thumb="{{ MEDIA_URL }}
                                    images/main-slider/5.jpg"
                                    data-saveperformance="off"
96                              data-title="Awesome Title Here">
97                              <img src="{{ MEDIA_URL }}images/
                                    main-slider/5.jpg" alt=""
                                    data-bgposition="center top"
98                                  data-bgfit="cover" data-
                                        bgrepeat="no-repeat">
99
100
101                         </li>
102
103                 </ul>
```

```
104
105                 <div class="tp-bannertimer"></div>
106             </div>
107         </div>
108     </section>
109
110
111     <!--Sidebar Page-->
112     <div class="sidebar-page">
113         <div class="auto-container">
114             <div class="row clearfix">
115
116                 <!--Content Side-->
117                 <div class="col-lg-12 col-md-12 col-sm-12
                        col-xs-12">
118                     <section class="blog-container centered-
                            blog text-center">
119
120                         <!--Section Title-->
121                         <div class="sec-title text-center">
122                             <div class="sub-title-theme
                                    theme_color">STAY TUNED AND
                                    UP TO DATE</div>
123                             <h3 class="bigger-title">Latest
                                    from Animetacchi</h3>
124                         </div>
125
126                         {% for i in anime_data %}
127                         <!--Blog Post-->
128                         <div class="blog-post wow fadeInLeft
                                " data-wow-delay="0ms" data-wow-
                                duration="1500ms">
129                             <article class="column-inner">
130                                 <figure class="image-box">
131                                     <a href="{% url '
                                            anime_details' slug=i
                                            .slug %}"><img src="
                                            {{ MEDIA_URL }}{{i.
                                            a_cover}}" alt=""
                                            title="Blog" width="
```

```
                    1171" height="389"></
                    a>
132             </figure>
133             <div class="lower-part">
134                 <div class="post-info">
135                     {% for genre in i.
                            a_genre.all  %}
136                         <input type="
                                hidden" value
                                ="{{genre.
                                genre_type}}"
                                >
137                         <span class="
                                bullet">&bull
                                ;</span> <a
                                href="
                                javascript:;"
                                >{{genre.
                                genre_type|
                                upper}}</a>
138                     {% endfor %}
139                     <!--<span class="
                            bullet">&bull;</
                            span> <a href="#"
                            >DESIGN,
140                     MEDIA, ILLUSTRATION
                            </a> <span class=
                            "bullet">&bull;</
                            span> <a href="#"
                            >BY JONATHAN
141                     DOE</a> <span class=
                            "bullet">&bull;</
                            span> <a href="#"
                            >31 COMMENTS</a
                            >-->
142                 </div>
143                 <div class="post-title"
                        ><h3><a href="{% url
                        'anime_details' slug=
                        i.slug %}">{{i.a_name
```

```
                                        }}</a>
144                                    </h3></div>
145                                    <div class="post-text
                                        more "> {{i.
                                        a_synopsys}}
146                                    </div>
147                                </div>
148                            </article>
149                        </div>
150                        {% endfor %}
151                    </section>
152                </div>
153            </div>
154        </div>
155    </div>
156
157    <!-- Client Testimonials -->
158    <section class="client-testimonials style-one rounded-
           nav orange-theme">
159      <div class="auto-container">
160
161          <!--Section Title-->
162            <div class="sec-title text-center">
163                <h2><span class="quote-icon fa fa-quote-
                       right"></span></h2>
164            </div>
165
166            {% if index %}
167                <div class="testimonial-slider-full">
168                    {% for i in index %}
169                    <article class="slide">
170
171                        <div class="text">{{i.news}}</div>
172                        <!--div class="author-info"></div-->
173                    </article>
174                    {% endfor %}
175
176                </div>
177            {% endif %}
178        </div>
```

```
179     </section>
180
181     <!--Main Footer-->
182     <footer class="main-footer">
183
184         <!--Footer Bottom-->
185       <div class="footer-bottom">
186             <div class="auto-container">
187                 <!--Copyright-->
188                 <div class="copyright">Made with  <span
                        class="theme_color fa fa-heart-o"></span
                       >  by et-projects | &copy; Copyright
                        2016 All Rights Reserved</div>
189             </div>
190         </div>
191
192     </footer>
193
194
195 </div>
196 <!--End pagewrapper-->
197
198 <!--Scroll to top-->
199 <div class="scroll-to-top"><span class="fa fa-arrow-up"></
    span></div>
200
201
202 <!--Search Popup-->
203 <div id="search-popup" class="search-popup">
204   <div class="close-search theme-btn"><span class="flaticon-
      close7"></span></div>
205   <div class="popup-inner">
206
207       <div class="search-form">
208           <form method="post" action="index.html">
209               <div class="form-group">
210                   <fieldset>
211                         <input type="search" class="form-
                              control" name="search-input"
                              value="" placeholder="Search Here
```

```
                                " required >
212                            <input type="submit" value="Search
                                   Now!" class="theme-btn">
213                        </fieldset>
214                    </div>
215                </form>
216
217                <br>
218                <h3>Recent Search Keywords</h3>
219                <ul class="recent-searches">
220                    <li><a href="#">Business</a></li>
221                    <li><a href="#">Web Development</a></li>
222                    <li><a href="#">SEO</a></li>
223                    <li><a href="#">Logistics</a></li>
224                    <li><a href="#">Freedom</a></li>
225                </ul>
226
227            </div>
228
229        </div>
230    </div>
231
232    <script src="{{ MEDIA_URL }}js/jquery.js"></script>
233    <script src="{{ MEDIA_URL }}js/bootstrap.min.js"></script>
234    <script src="{{ MEDIA_URL }}js/jquery.mCustomScrollbar.
       concat.min.js"></script>
235    <script src="{{ MEDIA_URL }}js/revolution.min.js"></script>
236    <script src="{{ MEDIA_URL }}js/bxslider.js"></script>
237    <script src="{{ MEDIA_URL }}js/owl.js"></script>
238    <script src="{{ MEDIA_URL }}js/mixitup.js"></script>
239    <script src="{{ MEDIA_URL }}js/jquery.fancybox.pack.js"></
       script>
240    <script src="{{ MEDIA_URL }}js/owl.js"></script>
241    <script src="{{ MEDIA_URL }}js/wow.js"></script>
242    <script src="{{ MEDIA_URL }}js/script.js"></script>
243    </body>
244    </html>
```

```
1   signin.html
2
```

```
3  {% extends 'base_home.html' %}
4
5  {% block css %}
6      <link rel="stylesheet" href="{{ MEDIA_URL }}signin/css/
           app.v2.css" type="text/css"/>
7      <link rel="stylesheet" href="{{ MEDIA_URL }}signin/css/
           formValidation.css" type="text/css" cache="false"/>
8
9  {% endblock %}
10 {% block js %}
11
12     <script type="text/javascript">
13         $(document).ready(function() {
14             {% if message_error %}
15                 $.notify("{{message_error}}", "error");
16             {% endif %}
17             {% if message_match %}
18                 $.notify("{{message_match}}", "info");
19             {% endif %}
20             {% if message_active %}
21                 $.notify("{{message_active}}", "warn");
22             {% endif %}
23             {% if message_success %}
24                 $.notify("{{message_success}}", "error");
25                 $(function(){
26                     window.location="{% url 'users' username
                           =request.user %}";
27                 })
28             {% endif %}
29         });
30
31
32     </script>
33
34 {% endblock %}
35
36 {% block body %}
37
38 <body>
```

```
39      <div style="background-image: url(/media/signin/image/bg
            .jpg);">
40          <div style="background-color: rgba(0,0,0,0.5);">
41              <p style="padding-top:220px;"></p>
42              <section id="content" class="m-t-lg wrapper-md
                    animated fadeInDown">
43                  <!--<a class="nav-brand" href="{% url 'home
                        '%}">Animetacchi</a>-->
44
45                  <div class="row m-n">
46                      <div class="col-md-4 col-md-offset-4 m-t
                            -lg">
47                          <section class="panel">
48                              <header class="panel-heading
                                    text-center"> Sign in</header
                                    >
49                              <form method="POST" action="{%
                                    url 'signin' %}" class="panel
                                    -body" id="defaultForm">{%
                                    csrf_token %}
50                                  <div class="form-group"><
                                        label class="control-
                                        label">Email</label>
51                                      <input type="email" name
                                            ="email" placeholder=
                                            "test@example.com"
                                            class="form-control">
52                                  </div>
53                                  <div class="form-group"><
                                        label class="control-
                                        label">Password</label>
54                                      <input type="password"
                                            name="password"
                                            placeholder="Password
                                            "
55                                                  class="form-
                                                        control">
56                                  </div>
57                                  <a href="#" class="pull-
                                        right m-t-xs">
```

```
58                                      <small>Forgot password
                                            ?</small>
59                                  </a>
60                                  <button type="submit" class=
                                        "btn btn-info">Sign in</
                                        button>
61                                  <div class="line line-dashed
                                        "></div>
62                                  <p class="text-muted text-
                                        center">
63                                      <small>Do not have an
                                            account?</small>
64                                  </p>
65                                  <a href="{% url 'signup' %}"
                                         class="btn btn-white btn
                                        -block">Create an account
                                        </a>
66                              </form>
67                          </section>
68                      </div>
69                  </div>
70              </section>
71              <!-- footer -->
72              <footer id="footer">
73                  <div class="text-center padder clearfix">
74                      <p>
75                          <small>Animetacchi<br>&copy; 2016</
                                small>
76                      </p>
77                  </div>
78              </footer>
79          </div>
80      </div>
81  </body>
82
83  {% endblock %}
```

```
1   signup.html
2
3   {% extends 'base_home.html' %}
```

```
4
5  {% block css %}
6      <link rel="stylesheet" href="{{ MEDIA_URL }}signin/css/
           app.v2.css" type="text/css"/>
7      <link rel="stylesheet" href="{{ MEDIA_URL }}signin/css/
           formValidation.css" type="text/css" cache="false"/>
8
9
10
11 {% endblock %}
12 {% block js %}
13     <script type="text/javascript">
14         function submit(type) {
15             var uname = $('input[name=uname]').val();
16             var email = $('input[name=email]').val();
17             var passwd = $('input[name=password]').val();
18             var agree = $('#agree').is(':checked');
19             if (agree == false){
20                 document.getElementById("agree").focus();
21                 return
22             }
23             $.ajax({
24                 type: "POST",
25                 cache: false,
26                 url: "{% url 'signup' %}",
27                 data: {
28                     'action': type,
29                     'uname': uname,
30                     'email': email,
31                     'passwd': passwd,
32                     'csrfmiddlewaretoken':'{{csrf_token}}'},
33                 success:function(response){
34                     if (response.val == false){
35                         $.notify(response.alert, "info");
36                     } else {
37                         $.notify(response.alert, "success");
38                         setTimeout('RedirectUpdate()', 2000)
                               ;
39                     }
40                 }
```

```
41              });
42          }
43
44          function RedirectUpdate() {
45              window.location = "{% url 'check_email' %}"
46          }
47
48
49      </script>
50
51  {% endblock %}
52
53  {% block body %}
54
55  <body>
56      <div style="background-image: url(/media/signin/image/bg
            .jpg);">
57          <div style="background-color: rgba(0,0,0,0.5);">
58              <p style="padding-top:120px;"></p>
59              <section id="content" class="m-t-lg wrapper-md
                    animated fadeInDown">
60                  <!--<a class="nav-brand" href="{% url 'home
                        '%}">Animetacchi</a>-->
61
62                  <div class="row m-n">
63                      <div class="col-md-4 col-md-offset-4 m-t
                            -lg">
64                          <section class="panel">
65                              <header class="panel-heading bg
                                    bg-primary text-center"> Sign
                                     up</header>
66                              <form class="panel-body" id="
                                    defaultForm">
67                                  <div class="form-group"><
                                        label class="control-
                                        label">Username</label>
68                                      <input type="text" name=
                                            "uname" placeholder="
                                            eg. Your id" class="
                                            form-control">
```

```
69                                  </div>
70                                  <div class="form-group"><
                                        label class="control-
                                        label">Your email address
                                        </label>
71                                       <input type="email" name
                                            ="email" placeholder=
                                            "test@example.com"
72                                               class="form-
                                                    control">
73                                  </div>
74                                  <div class="form-group"><
                                        label class="control-
                                        label">Type a password</
                                        label>
75                                       <input type="password"
                                            name="password"
                                            placeholder="Password
                                            "
76                                               class="form-
                                                    control">
77                                  </div>
78                                  <div class="form-group"><
                                        label class="control-
                                        label">Confirm password</
                                        label>
79                                       <input type="password"
                                            name="confirmPassword
                                            " placeholder="
                                            Confirm Password"
80                                               class="form-
                                                    control">
81                                  </div>
82                                  <div class="form-group form-
                                        horizontal">
83                                       <div class="checkbox">
84                                            <label>
85                                                 <input type="
                                                      checkbox" id=
                                                      "agree" name=
```

```
                                                    "agree">
                                                    Agree the <a
                                                    href="#">
                                                    terms
86                                                      and policy</a>
87                                                  </label>
88                                              </div>
89                                          </div>
90                                          <div onclick="submit('signup
                                                ')">
91                                              <button class="btn btn-
                                                    info">Sign up</button
                                                    >
92                                          </div>
93                                          <div class="line line-dashed
                                                "></div>
94                                          <p class="text-muted text-
                                                center">
95                                              <small>Already have an
                                                    account?</small>
96                                          </p>
97                                          <a href="{% url 'signin' %}"
                                                class="btn btn-white btn
                                                -block">Sign in</a></form
                                                >
98                                      </section>
99                                  </div>
100                             </div>
101                         </section>
102                         <!-- footer -->
103                         <footer id="footer">
104                             <div class="text-center padder clearfix">
105                                 <small>Animetacchi<br>&copy; 2016</small
                                        >
106                             </div>
107                         </footer>
108                     </div>
109                 </div>
110             </body>
111
```

```
112 <!-- / footer -->
113
114 {% endblock %}
```

```
1  dashboard.html
2
3  {% extends 'base_menu.html' %}
4
5  {% block css %}
6
7      <link href="{{ MEDIA_URL }}css/style2.css" rel="
           stylesheet">
8      <link href="{{ MEDIA_URL }}css/bootstrap-reset.css" rel=
           "stylesheet">
9
10     <link href="{{ MEDIA_URL }}css/style-responsive.css" rel
           ="stylesheet" />
11     <style type="text/css">
12         .proPic {
13             width:138px;
14             height:138px;
15             float: left;
16             no-repeat;
17             border:5px solid #fff;
18         }
19         .proPic .redO{
20             position:relative;
21             background:#E2D8D7;
22             font-weight: 700;
23             top: -30px;
24             opacity:0.7;
25             padding: 0 0 0 5px;
26         }
27     </style>
28 {% endblock %}
29 {% block js %}
30     <script type="text/javascript">
31         function aboutEdit(){
32             $('#aboutFile').hide();
33             $('#aboutEdit').show();
```

```
34              $('#aboutEdita').hide();
35              $('#aboutSave').show();
36          }
37          function submit(type,seq){
38              var about = $('#aboutForm').val();
39              var member = $('input[name=memberUser]').val();
40              var seq_about = $('input[name=seq_about]').val()
                    ;
41              $.ajax({
42                  type: "POST",
43                  cache: false,
44                  url: "{% url 'users' username=member.m_name
                        %}",
45                  data: {
46                      'action': type,
47                      'seq_about':seq_about,
48                      'about': about,
49                      'member': member,
50                      'csrfmiddlewaretoken':'{{csrf_token}}'},
51                  success:function(response){
52                      window.location = "{% url 'users'
                            username=member.m_name %}"
53                  }
54              });
55          }
56      </script>
57  {% endblock %}
58
59  {% block body%}
60      <!--Main Slider-->
61      <section class="page-title"
62               style="background-image: url({{ MEDIA_URL }}{{
                     member.m_cover}}); width:100%;height:400px;"
                     >
63          <div class="auto-container">
64              <div class="row clearfix">
65                  <div class="col-lg-6 col-xs-12" style="
                        position:absolute;bottom:30px;">
66                      <div class="proPic">
67                          <a href="#">
```

```
68              {% if member.m_picture %}
69              <img src="{{MEDIA_URL}}{{member.m_picture}}"
                    alt="MembersPictureAnimetacchi"
70                                      width="128" height=
                                            "128">
71                          {% else %}
72                          <img src="{{MEDIA_URL}}
                                MembersPicture/pic.png" alt="
                                MembersPictureAnimetacchi"
73                                      width="128" height=
                                            "128">
74                          {% endif %}
75                      </a>
76
77                      {% if member.user == request.user %}
78              <!--<div class="redO"><a href="#" class="fa fa
                    -camera"> Update Picture</a></div>-->
79              <h2 style="padding-left:15px;"> {{member.
                    m_name}}</h2>
80                      {% else %}
81
82              <h2 style="padding-left:15px;" > {{member.
                    m_name}}</h2>
83
84                      {% endif %}
85
86
87
88                  <div style="top:5px; padding-left:15px;"
                        ></div>
89
90
91                  </div>
92                  <figure class="image">
93                  </figure>
94
95              </div>
96
97          </div>
98      </div>
```

```
99      </section>
100
101     <!-- Page Title -->
102     <section style="height:70px;background-color: #E0DEDE"
            class="page-title" >
103       <div class="auto-container">
104           <div class="row clearfix">
105                 <div class="tabs-box">
106                     <!-- Tab Buttons -->
107                     <div class="tab-buttons anim-3-all
                            clearfix" style="padding-left:15px;">
108                         <a href="{% url 'users' username=
                                member.user.username %} " class="
                                tab-btn {% if not 'library' in
                                request.META.PATH_INFO %} active
                                {% endif %}">Feed</a>
109                         <a href="{% url 'library' username=
                                member.user.username %}" class="
                                tab-btn {% if 'library' in
                                request.META.PATH_INFO %} active
                                {% endif %}">Library</a>
110
111                     </div>
112                 </div>
113
114
115           </div>
116       </div>
117     </section>
118    {% block dashboard_body %}
119     <div class="sidebar-page">
120         <div class="auto-container">
121             <div class="row clearfix" >
122                 <div class="col-sm-6" id="formAbout">
123                     <div class="row">
124                         <div class="col-md-12">
125                             <!--
126                             <section class="panel">
127                                 <header class="panel-heading
                                        ">
```

```
128                                 <h3>About
129                                 <span class="tools pull-
                                        right">
130                                     <div onclick="
                                            aboutEdit()">
131                                         <a href="
                                                javascript:;"
                                                 id="
                                                aboutEdita"
                                                class="fa fa-
                                                pencil"></a>
132                                     </div>
133                                     <div onclick="submit
                                            ('saveAbout')">
134                                         <button style="
                                                display:none;
                                                " id="
                                                aboutSave"
                                                class="btn
                                                btn-default
                                                btn-xs">Save
135                                         </button>
136                                         <input type="
                                                hidden" name=
                                                "memberUser"
                                                value="{{
                                                request.
                                                get_full_path
                                                }}"/>
137                                         {% for i in
                                                about_data %}
138                                             <input type=
                                                    "hidden"
                                                    name="
                                                    seq_about
                                                    " value="
                                                    {{i.
                                                    seq_about
                                                    }}"/>
139                                         {% endfor %}
```

```
140                                     </div>
141                                 </span>
142                                 </h3>
143                             </header>
144                             <div class="panel-body" id="
                                    aboutFile" >
145                                 {% for i in about_data
                                        %}{{i.about}}{%
                                        endfor %}
146
147                             </div>
148                             <div style="display:none;"
                                    id="aboutEdit" class="
                                    panel-body" >
149                                 <textarea class="form-
                                        control ckeditor" id=
                                        "aboutForm" rows="3"
                                        style="margin-top: 0
                                        px; margin-bottom: 0
                                        px; height: 100px;"
                                        placeholder="Tell us
                                        about yourself. :3"
                                        >{% for i in
                                        about_data %}{{i.
                                        about}}{% endfor %}</
                                        textarea>
150
151                             </div>
152                         </section>-->
153
154                     </div>
155                 </div>
156             </div>
157
158             <div class="col-sm-6">
159                 <div class="row">
160                     <div class="col-md-12">
161                         <!--<section class="panel">
162
163                             <div class="panel-body">
```

```
164                                     <div class="form-group">
165                                         <textarea class="
                                                form-control"
                                                rows="3"
                                                placeholder="
                                                Write something
                                                ..."></textarea>
166                                     </div>
167                                     <button type="submit"
                                            class="btn btn-
                                            primary">POST</button
                                            >
168
169                                 </div>
170                             </section>-->
171
172                         </div>
173                     </div>
174                 </div>
175             </div>
176             <div class="row clearfix">
177                 <div class="col-sm-6">
178                     <div class="row">
179                         <div class="col-md-12">
180                             <section class="panel">
181                                 <header class="panel-heading
                                        ">
182                                     <h3>Daisukilist</h3>
183
184                                 </header>
185                                 <div class="panel-body">
186                                     {% for daisuki in member
                                            .daisukilist_set.all
                                            %}
187                                     <h3>Anime</h3>
188                                     <div class="row">
189                 {% for anime in daisuki.anime.all %}
190                   <div class="col-md-4 col-sm-6 col-xs
                        -12">
191                     <!--Image-->
```

```
192                     <figure class="image">
193                       <a href="{% url 'anime_details'
                              slug=anime.slug %}" class="
                              lightbox-image"
194                              title="{{anime.a_name}}"><img
                                 src="{{ MEDIA_URL }}{{anime.
                                 a_displaypic}}" alt="" width
                                ="137" height="199" style="
                                 border-radius:5px;"></a>
195                     </figure>
196                   </div>
197
198                 {% empty %}
199                   <div class="col-md-12">
200                     <strong>You have no daisukilist of
                           Anime</strong>
201                   </div>
202                 {% endfor %}
203               </div>
204               <hr>
205               <h3>Character</h3>
206               <div class="row">
207                 {% for character in daisuki.character.
                      all %}
208
209                   <div class="col-md-4 col-sm-6 col-xs
                        -12">
210                     <!--Image-->
211                     <figure class="image">
212                       <a href="{% url 'anime_character'
                              slug=character.anime.slug id=
                              character.id %}" class="
                              lightbox-image"
213                              title="{{character.name}}"><img
                                  src="{{ MEDIA_URL }}{{
                                 character.picture}}" alt=""
                                 width="137" height="199"
                                 style="border-radius:5px;"
                                 ></a>
214                     </figure>
```

```
215                     </div>
216
217                   {% empty %}
218                     <div class="col-md-12">
219                       <strong>You have no daisukilist of
                              Character</strong>
220                     </div>
221                   {% endfor %}
222                 </div>
223                 <hr>
224                 <h3>Voice Actor</h3>
225                 <div class="row">
226                   {% for voice in daisuki.voice_character.
                        all %}
227
228                     <div class="col-md-4 col-sm-6 col-xs
                          -12">
229                       <!--Image-->
230                       <figure class="image">
231                         <a href="{% url 'anime_voice' id=
                              voice.id %}" class="lightbox-
                              image"
232                           title="{{voice.name}}"><img src
                                ="{{ MEDIA_URL }}{{voice.
                                picture}}" alt="" width="137
                                " height="199" style="border
                                -radius:5px;"></a>
233                       </figure>
234                     </div>
235                   {% empty %}
236                     <div class="col-md-12">
237                       <strong>You have no daisukilist of
                              Voice Actor</strong>
238                     </div>
239                   {% endfor %}
240                 </div>
241                 <hr>
242
243                 {% empty %}
244                   <h3>Anime</h3>
```

```
245                     <div class="row">
246                       <div class="col-md-12">
247                         <strong>You have no daisukilist of
                                Anime</strong>
248                       </div>
249                     </div>
250                     <hr>
251                     <h3>Character</h3>
252                     <div class="row">
253                       <div class="col-md-12">
254                         <strong>You have no daisukilist of
                                Character</strong>
255                       </div>
256                     </div>
257                     <hr>
258                     <h3>Voice Actor</h3>
259                     <div class="row">
260                       <div class="col-md-12">
261                         <strong>You have no daisukilist of
                                Voice Actor</strong>
262                       </div>
263                     </div>
264                     <hr>
265                                 {% endfor %}
266                             </div>
267                         </section>
268
269                     </div>
270                 </div>
271                 <div class="row">
272                     <div class="col-md-12">
273                         <section class="panel">
274                             <header class="panel-heading
                                    ">
275                                 <h3>Daikirailist</h3>
276
277                             </header>
278                             <div class="panel-body">
279                                 {% for daikirai in
                                        member.
```

```
                                    daikirailist_set.all
                                    %}
280                                 <h3>Anime</h3>
281                                 <div class="row">
282                 {% for anime in daikirai.anime.all %}
283                   <div class="col-md-4 col-sm-6 col-xs
                          -12">
284                     <!--Image-->
285                     <figure class="image">
286                       <a href="{% url 'anime_details'
                              slug=anime.slug %}" class="
                              lightbox-image"
287                              title="{{anime.a_name}}"><img
                                 src="{{ MEDIA_URL }}{{anime.
                                 a_displaypic}}" alt="" width
                                 ="137" height="199" style="
                                 border-radius:5px;"></a>
288                     </figure>
289                   </div>
290
291                   {% empty %}
292                     <div class="col-md-12">
293                       <strong>You have no daikirailist of
                             Anime</strong>
294                     </div>
295                   {% endfor %}
296                 </div>
297                 <hr>
298                 <h3>Character</h3>
299                 <div class="row">
300                   {% for character in daikirai.character.
                        all %}
301
302                     <div class="col-md-4 col-sm-6 col-xs
                          -12">
303                       <!--Image-->
304                       <figure class="image">
305                         <a href="{% url 'anime_character'
                                slug=character.anime.slug id=
                                character.id %}" class="
```

```
                                lightbox-image"
306                             title="{{character.name}}"><img
                                    src="{{ MEDIA_URL }}{{
                                character.picture}}" alt=""
                                width="137" height="199"
                                style="border-radius:5px;"
                                ></a>
307                         </figure>
308                       </div>
309
310                     {% empty %}
311                       <div class="col-md-12">
312                         <strong>You have no daikirailist of
                                Character</strong>
313                       </div>
314                     {% endfor %}
315                   </div>
316                   <hr>
317
318
319                   {% empty %}
320                     <h3>Anime</h3>
321                     <div class="row">
322                       <div class="col-md-12">
323                         <strong>You have no daikirailist of
                                Anime</strong>
324                       </div>
325                     </div>
326                     <hr>
327                     <h3>Character</h3>
328                     <div class="row">
329                       <div class="col-md-12">
330                         <strong>You have no daikirailist of
                                Character</strong>
331                       </div>
332                     </div>
333                     <hr>
334
335                                 {% endfor %}
336                             </div>
```

```
337                             </section>
338
339                         </div>
340                     </div>
341                 </div>
342                 <div class="col-sm-6">
343                     <div class="row">
344                         <div class="col-md-12">
345                             <div class="tabs-box">
346                                 <!-- Tab Buttons -->
347                                 <div class="tab-buttons anim
                                        -3-all clearfix" style="
                                        padding-left:15px;">
348                                     <a  class="tab-btn
                                            active ">Timeline</a>
349
350                                 </div>
351                             </div>
352
353
354                             {% for i in member.watchlist_set
                                    .all %}
355                             <section class="panel">
356                                 <div class="panel-body">
357                                  <div class="col-md-4 col-sm
                                        -6 col-xs-12" style="
                                        margin:-5px 0 0 -20px;">
358                                     <!--Image-->
359                                     <figure class="image">
360                                 <a href="#" class="lightbox-image
                                    " title="{{i.watchlist.a_name
                                    }}"><img src="{{ MEDIA_URL
                                    }}{{i.watchlist.a_displaypic}}
                                    " alt="" width="100" height="
                                    150" style="border-radius:5px;
                                    "></a>
361                                     </figure>
362                                         </div>
363
```

```
364                                     <div class="col-md
                                            -8 col-sm-6 col-
                                            xs-12">
365                                      <!--Content-Column
                                            -->
366                                       <div class="cont-
                                              column">
367                                        <div class="
                                                inner-box">
368                                    <h3 class="anime-title"
                                          ><a href="#">{{i.
                                          watchlist.a_name}}</
                                          a></h3>
369                                          <ul class="
                                                list-inline
                                                ">
370                                          {% for genre
                                                in i.
                                                watchlist.
                                                a_genre.all
                                                 %}
371                                            <li href="#"
                                                 >{{genre.
                                                 genre_type
                                                 }}</li>
372                                          {% endfor %}
373                                          </ul>
374                                         </div>
375                                        </div>
376                                       </div>
377                                     </div>
378                                   </section>
379                                   {% endfor %}
380
381                                {% for i in member.
                                      readinglist_set.all %}
382                                   <section class="panel">
383                                    <div class="panel-body"
                                        >
```

```
384                                 <div class="col-md-4 col-sm
                                        -6 col-xs-12" style="
                                        margin:-5px 0 0 -20px;">
385                                     <!--Image-->
386                                     <figure class="
                                            image">
387                             <a href="#" class="lightbox-image"
                                    title="{{i.readinglist.name}}"><
                                    img src="{{ MEDIA_URL }}{{i.
                                    readinglist.displaypic}}" alt=""
                                    width="100" height="150" style="
                                    border-radius:5px;"></a>
388                                     </figure>
389                                 </div>
390
391                                 <div class="col-md
                                        -8 col-sm-6 col
                                        -xs-12">
392                                     <!--Content-
                                            Column-->
393                                     <div class="cont
                                            -column">
394                                         <div class="
                                                inner-box">
395                                     <h3 class="anime-title"
                                        ><a href="#">{{i.
                                        readinglist.name}}</
                                        a></h3>
396                                             <ul class="
                                                list-
                                                inline">
397                                             {% for genre
                                                 in i.
                                                readinglist
                                                .genre.
                                                all %}
398                                             <li href="#
                                                ">{{
                                                genre.
                                                genre_type
```

```
                                                                }}</li>
399                                                            {% endfor %}
400                                                           </ul>
401                                                          </div>
402                                                         </div>
403                                                        </div>
404                                                    </div>
405                                                </section>
406                                            {% endfor %}
407                                        </div>
408                                    </div>
409                                </div>
410                            </div>
411                        </div>
412                    </div>
413                    {% endblock %}
414
415
416                    {% endblock %}
```

```
1  library.html
2
3  {% extends 'User/dashboard.html' %}
4    {% block dashboard_body %}
5      <div class="sidebar-page" style="padding: 50px 0px 200px
           ;">
6          <div class="auto-container">
7              <div class="row clearfix" >
8                  <div class="col-sm-6" id="formAbout">
9                      <div class="row">
10                         <div class="col-md-12">
11
12
13                         </div>
14                     </div>
15                 </div>
16
17
18             </div>
19
```

```
20        <div class="row clearfix">
21          <div class="col-sm-12">
22            <div class="row">
23                        <div class="col-md-12">
24                            <section class="panel">
25                                <header class="panel-heading
                                    ">
26                                    <div class="default-
                                        title">
27                        <span role="presentation" class="
                            dropdown">
28                          <a id="drop4" href="#" class="dropdown
                              -toggle" data-toggle="dropdown"
29                            role="button" aria-haspopup="true"
                                aria-expanded="false">
30                            {% block caret %} Anime Library {%
                                endblock %}<span class="caret"></
                                span>
31                          </a>
32                          <ul id="menu1" class="dropdown-menu"
                              aria-labelledby="drop4">
33                            <li><a href="{% url 'library'
                                username=member.user.username
                                %}">Anime Library</a></li>
34                            <li><a href="{% url 'library_manga
                                ' username=member.user.username
                                %}">Manga Library</a></li>
35
36                                                  </ul>
37
38                        </span>
39                                                {% if request.user.
                                                    is_authenticated
                                                    %}
40                                                  <a href="https
                                                      ://www.
                                                      facebook.com/
                                                      sharer/sharer
                                                      .php?u=http%3
                                                      A//{{host|
```

```
                                                    urlencode}}/
                                                    users/{{
                                                    request.user.
                                                    username}}/
                                                    library/"
                                                    target="
                                                    _blank" >
41                                                   <button
                                                        class="
                                                        btn btn-
                                                        info">
                                                        Share to
                                                        Facebook
                                                        </button>
42                                                  </a>
43                                              {% endif %}
44
45
46                                      <span class="tools pull-
                                            right">
47
48                              <div class="btn-row">
49                                <div class="btn-group nav-tabs" role="
                                      tablist" data-toggle="buttons">
50                                  <label id="opt1" class="btn btn-
                                        default active"  role="presentation
                                        ">
51                                    <input type="radio" name="options"
                                        id="option1" autocomplete="off"
                                        checked>
52                                    <a href="#all" aria-controls="all"
                                        role="tab" data-toggle="tab">All
                                        </a>
53                                  </label>
54                                  <label id="opt2" class="btn btn-
                                        default">
55                                    <input type="radio" name="options"
                                        id="option2" autocomplete="off">
56                                    <a href="#bliss" aria-controls="
                                        bliss" role="tab" data-toggle="
```

```
                tab">Bliss</a>
57            </label>
58            <label id="opt3" class="btn btn-
                default">
59              <input type="radio" name="options"
                  id="option3" autocomplete="off">
60              <a href="#life" aria-controls="life"
                   role="tab" data-toggle="tab">
                  Life</a>
61            </label>
62            <label id="opt4" class="btn btn-
                default">
63              <input type="radio" name="options"
                  id="option3" autocomplete="off">
64              <a href="#hope" aria-controls="hope"
                   role="tab" data-toggle="tab">
                  Hope</a>
65            </label>
66            <label id="opt5" class="btn btn-
                default">
67              <input type="radio" name="options"
                  id="option3" autocomplete="off">
68              <a href="#agony" aria-controls="
                  agony" role="tab" data-toggle="
                  tab">Agony</a>
69            </label>
70          </div>
71          <script>
72
73            $('#opt1').click(function() {
74              $( '#opt1 a' ).trigger( "click" )
                    ;
75            });
76            $('#opt2').click(function() {
77              $( '#opt2 a' ).trigger( "click" )
                    ;
78            });
79            $('#opt3').click(function() {
80              $( '#opt3 a' ).trigger( "click" )
                    ;
```

```
81                          });
82                          $('#opt4').click(function() {
83                              $( '#opt4 a' ).trigger( "click" )
                                    ;
84                          });
85                          $('#opt5').click(function() {
86                              $( '#opt5 a' ).trigger( "click" )
                                    ;
87                          });
88
89                      </script>
90                  </div>
91
92
93                                  </span>
94                                  </div>
95                                 </header>
96
97              {% block panel %}
98                                 <div class="panel-body tab-
                                       content">
99
100                 <div class="table-responsive tab-pane
                        active" id="all" role="tabpanel">
101                   <table class="table">
102                     <thead>
103                       <tr>
104                         <th>Title</th>
105                         <th></th>
106                         <th>Ratings</th>
107                         <th>Type</th>
108                       </tr>
109                     </thead>
110                     {% include 'User/table/bliss.html' %}
111                   </table>
112                   <table class="table">
113                     {% include 'User/table/life.html' %}
114                   </table>
115                   <table class="table">
116                     {% include 'User/table/hope.html' %}
```

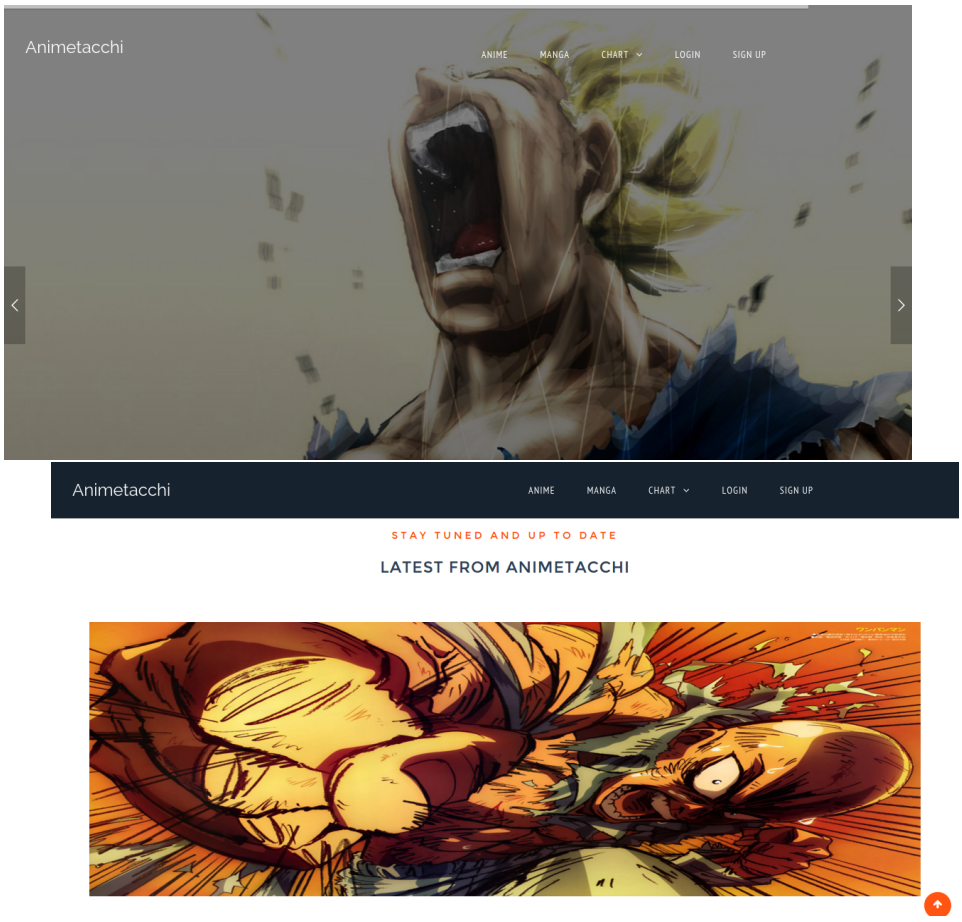
```
117                     </table>
118                     <table class="table">
119                       {% include 'User/table/agony.html' %}
120                     </table>
121                   </div><!--"table-responsive"-->
122                   <div class="table-responsive tab-pane" id=
                        "bliss" role="tabpanel">
123                     <table class="table">
124                       <thead>
125                         <tr>
126                           <th>Title</th>
127                           <th></th>
128                           <th>Ratings</th>
129                           <th>Type</th>
130                         </tr>
131                       </thead>
132                       {% include 'User/table/bliss.html' %}
133                     </table>
134
135                   </div><!--"table-responsive"-->
136                   <div class="table-responsive tab-pane" id=
                        "life" role="tabpanel">
137                     <table class="table">
138                       <thead>
139                         <tr>
140                           <th>Title</th>
141                           <th></th>
142                           <th>Ratings</th>
143                           <th>Type</th>
144                         </tr>
145                       </thead>
146                       {% include 'User/table/life.html' %}
147                     </table>
148
149                   </div><!--"table-responsive"-->
150                   <div class="table-responsive tab-pane" id=
                        "hope" role="tabpanel">
151                     <table class="table">
152                       <thead>
153                         <tr>
```

```
154                             <th>Title</th>
155                             <th></th>
156                             <th>Ratings</th>
157                             <th>Type</th>
158                           </tr>
159                         </thead>
160                         {% include 'User/table/hope.html' %}
161                       </table>
162
163                     </div><!--"table-responsive"-->
164                     <div class="table-responsive tab-pane" id=
                          "agony" role="tabpanel">
165                       <table class="table">
166                         <thead>
167                           <tr>
168                             <th>Title</th>
169                             <th></th>
170                             <th>Ratings</th>
171                             <th>Type</th>
172                           </tr>
173                         </thead>
174                         {% include 'User/table/agony.html' %}
175                       </table>
176
177                     </div><!--"table-responsive"-->
178
179
180
181                                 </div><!--"panel-body"-->
182                                 {% endblock %}
183                               </section>
184                             </div>
185                           </div>
186       </div>
187     </div> <!--end row clearfix -->
188       </div>
189   </div>
190
191
192   {% endblock %}
```

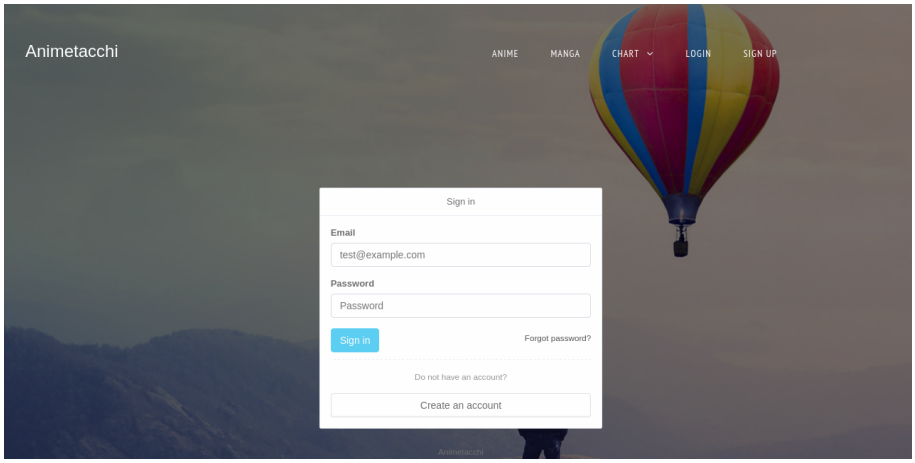
Appendix

Output

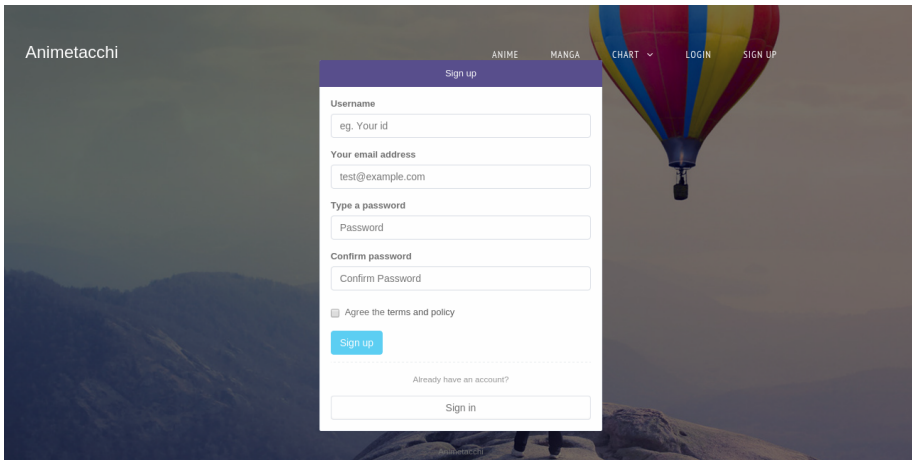
- Home Page



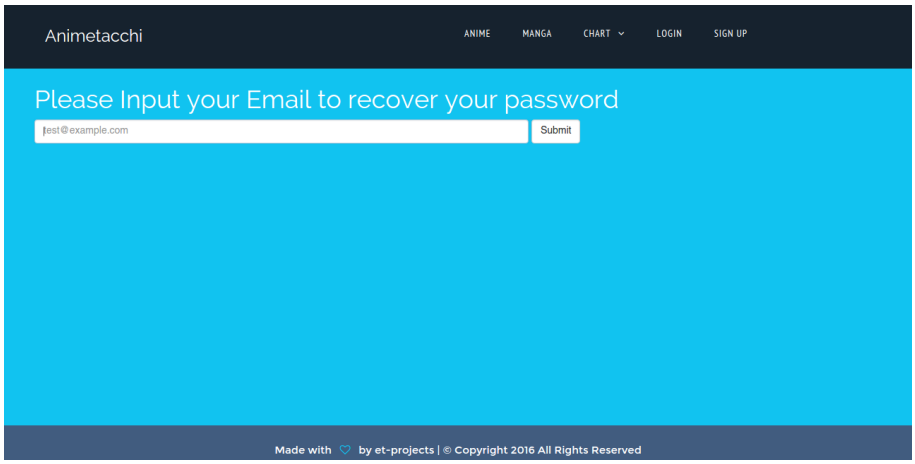
- Sign In Page

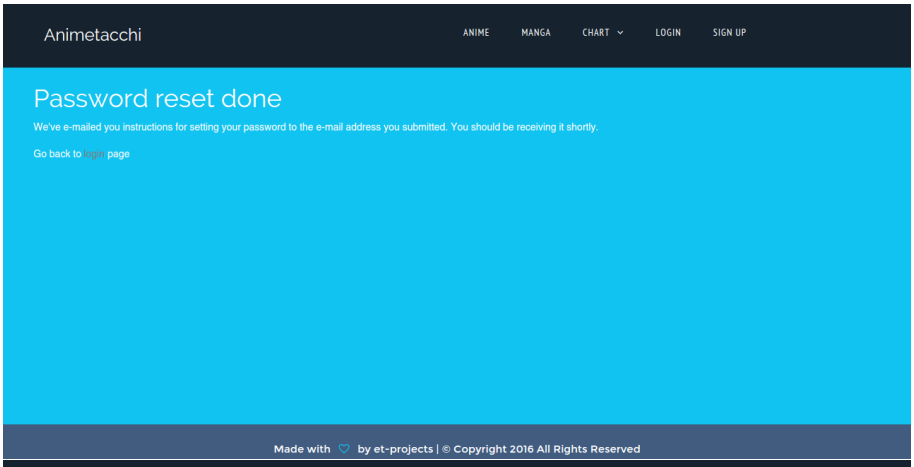


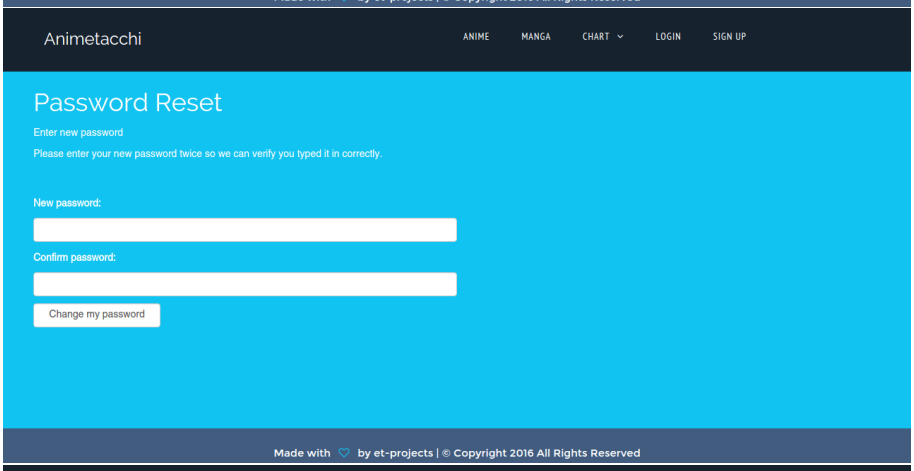
- Sign Up Page

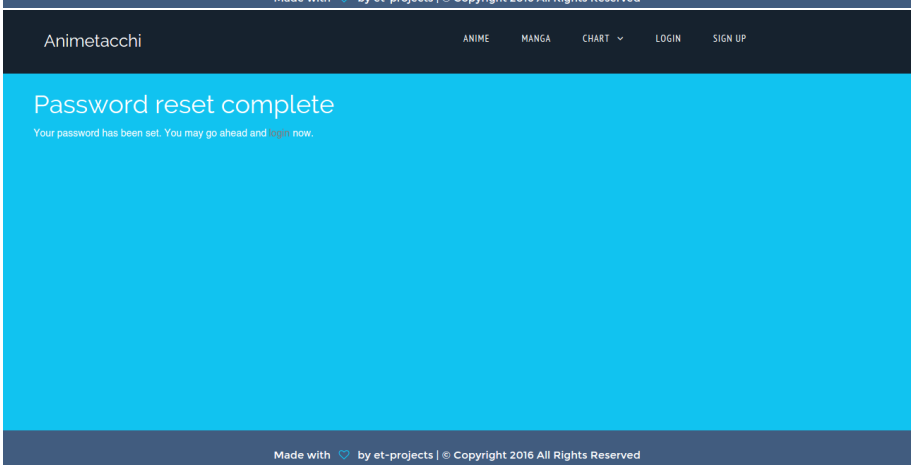


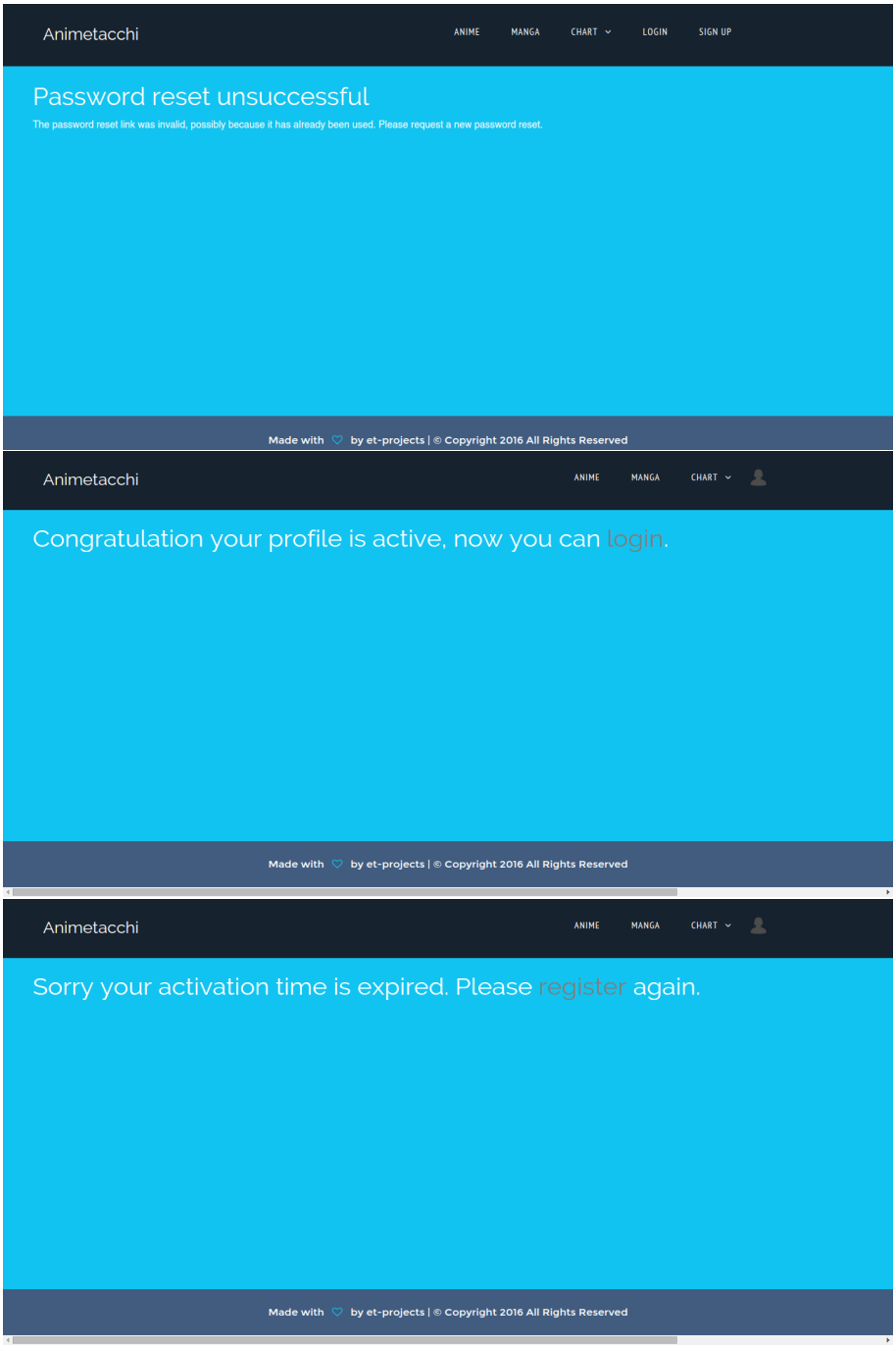
- Forgot Password Page

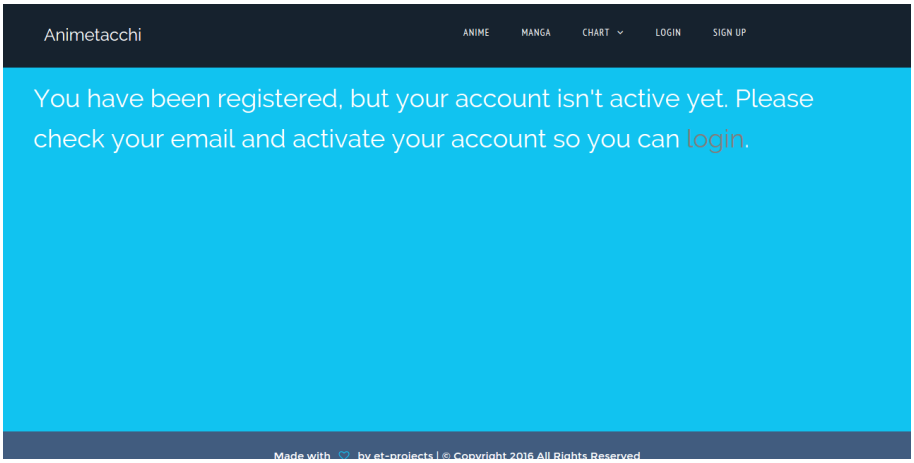



Animetacchi
ANIME
MANGA
CHART
LOGIN
SIGN UP
Password reset done
We've e-mailed you instructions for setting your password to the e-mail address you submitted. You should be receiving it shortly.
Go back to login page
Made with by et-projects | © Copyright 2016 All Rights Reserved

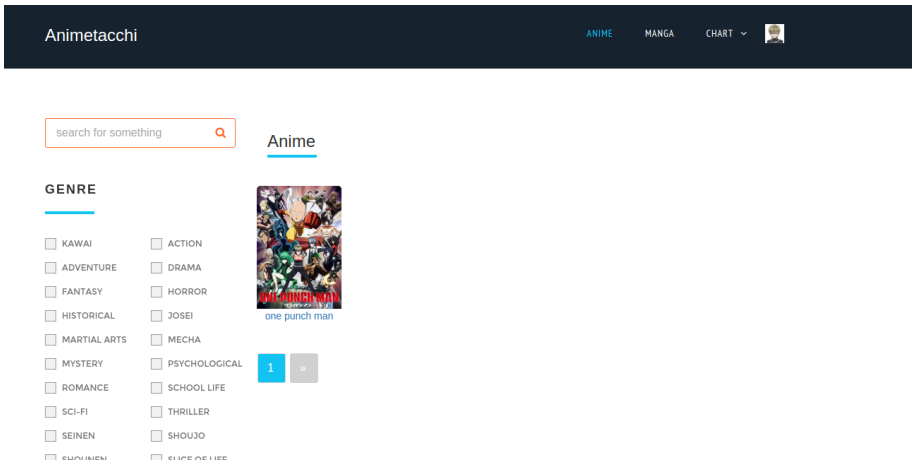

Animetacchi
ANIME
MANGA
CHART
LOGIN
SIGN UP
Password Reset
Enter new password
Please enter your new password twice so we can verify you typed it in correctly.
New password:
Confirm password:
Change my password
Made with by et-projects | © Copyright 2016 All Rights Reserved


Animetacchi
ANIME
MANGA
CHART
LOGIN
SIGN UP
Password reset complete
Your password has been set. You may go ahead and login now.
Made with by et-projects | © Copyright 2016 All Rights Reserved

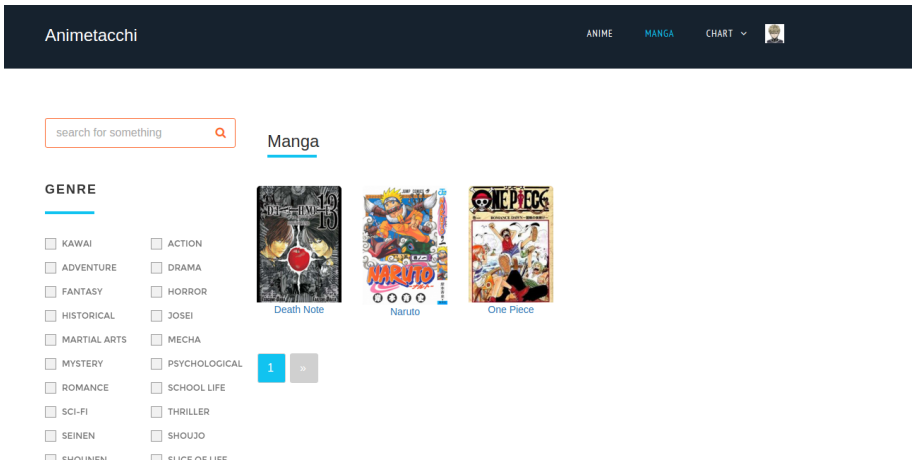

Animetacchi
ANIME
MANGA
CHART
LOGIN
SIGN UP
Password reset unsuccessful
The password reset link was invalid, possibly because it has already been used. Please request a new password reset.
Made with by et-projects | © Copyright 2016 All Rights Reserved
Animetacchi
ANIME
MANGA
CHART
Congratulation your profile is active, now you can login.
Made with by et-projects | © Copyright 2016 All Rights Reserved
Animetacchi
ANIME
MANGA
CHART
Sorry your activation time is expired. Please register again.
Made with by et-projects | © Copyright 2016 All Rights Reserved



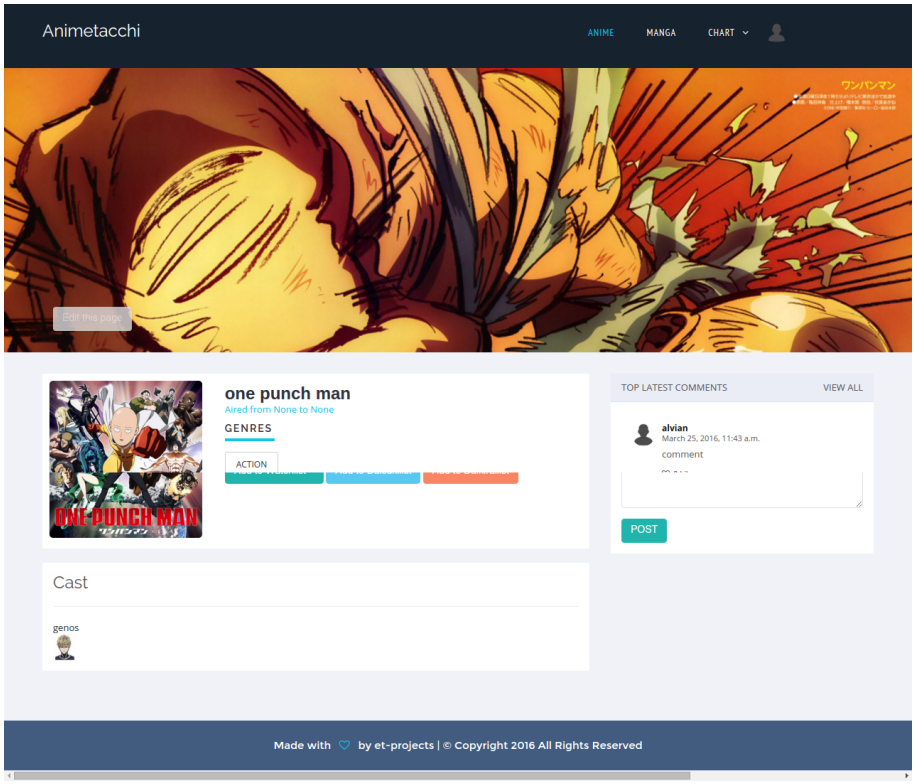
- Anime List Page



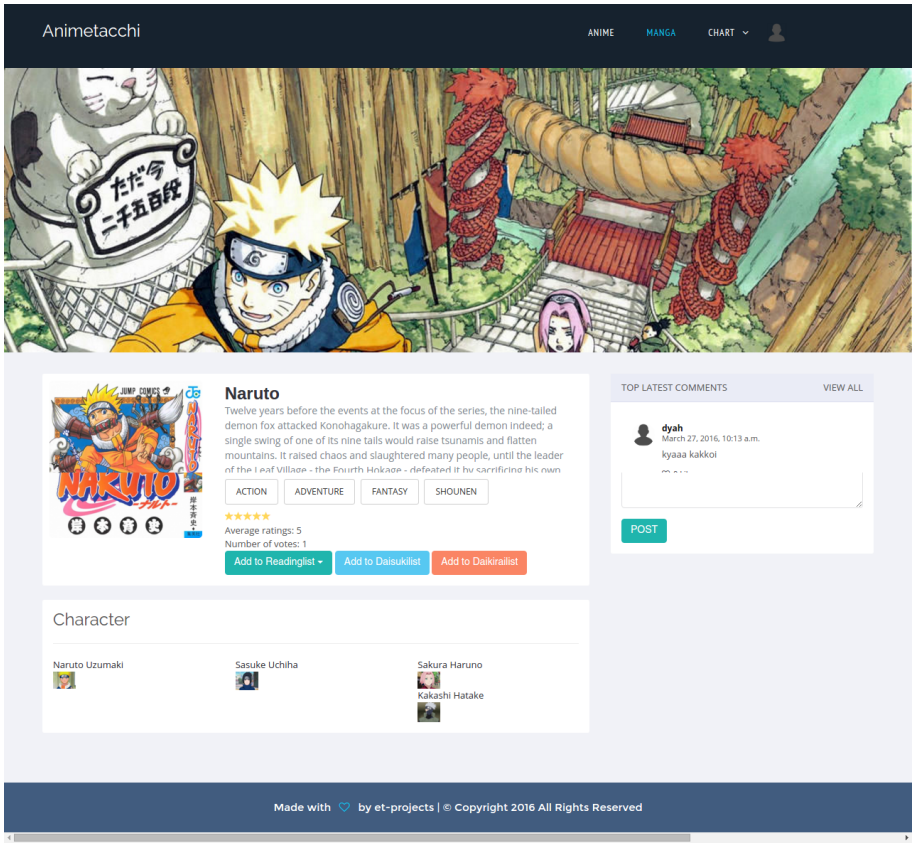
- Manga List Page



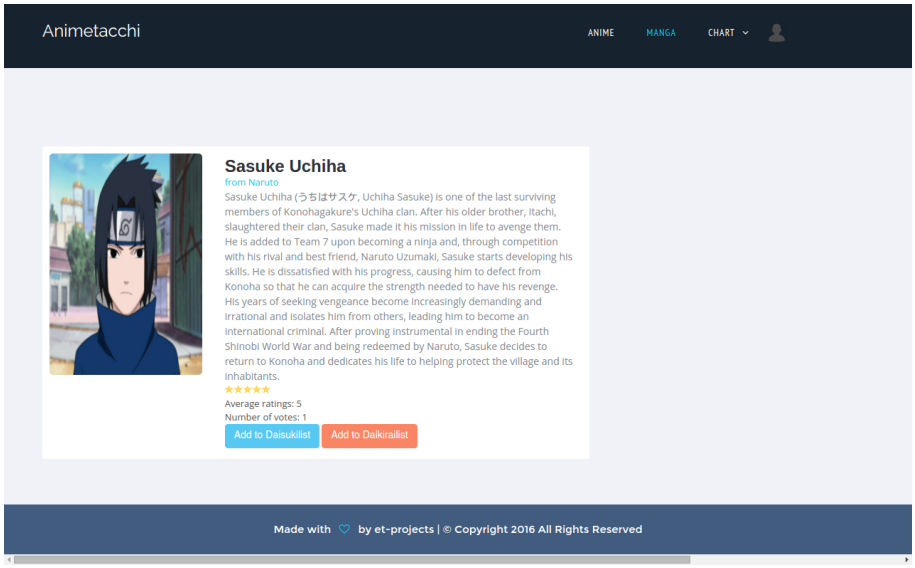
- Detail of Anime Page



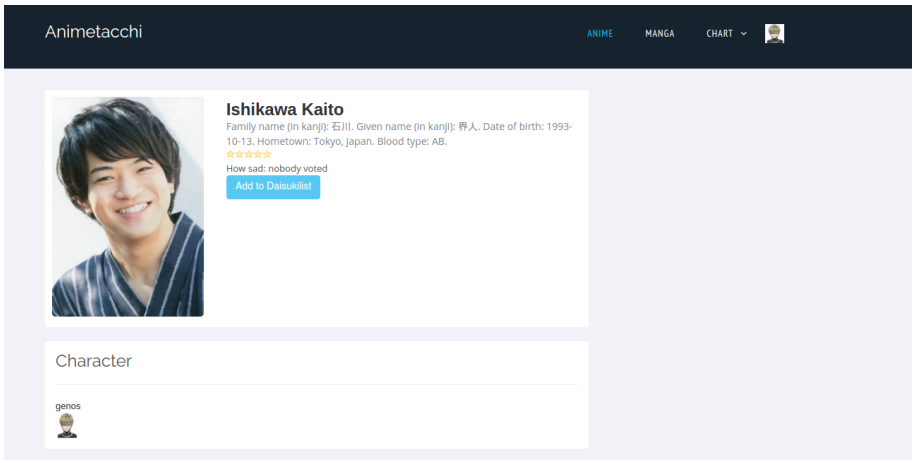
- Detail of Manga Page



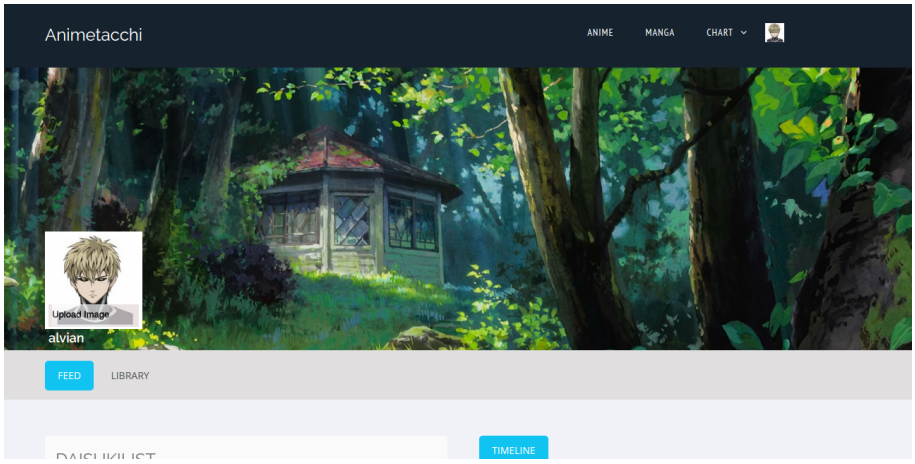
- Character Detail Page



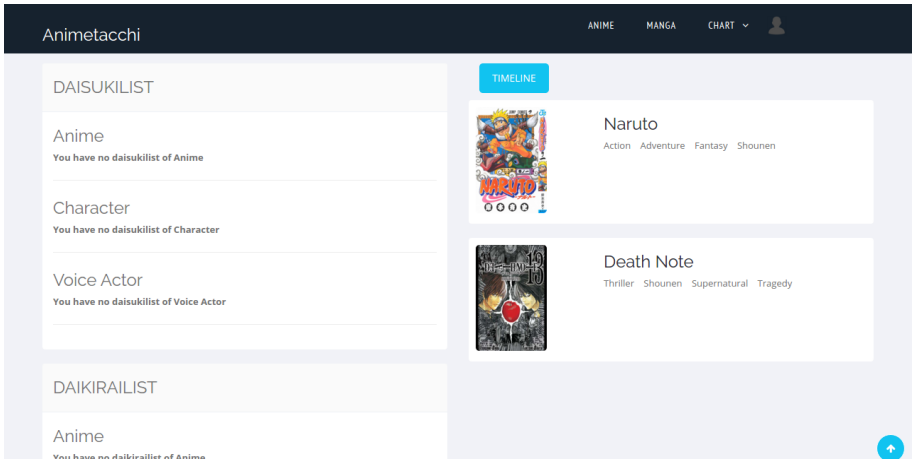
- Voice Actor Page



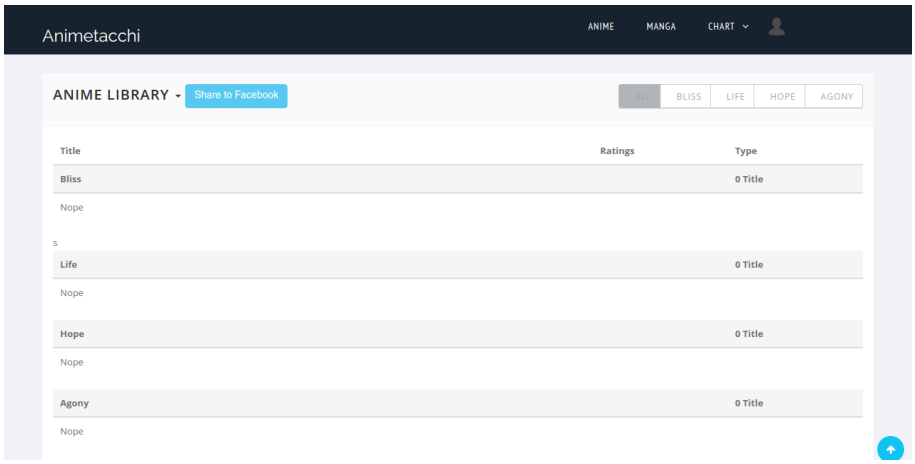
- User Profile Page



- User Profile Feed Page



- Anime Library



- Manga Library