****

**SPECIALIZED PROJECT**

**SOFTWARE REQUIREMENT SPECIFICATION DOCUMENT**

BUILDING WEBSITE “BOOK REVIEW” ON RUBY ON RAILS PLATFORM INTEGRATED MOBILE

**MENTOR :** Pham An Binh

**CAPSTWO TEAM:** Nguyen Hoang Quy

Phan Quang Tien

Lu Dinh Xuan

Hoang Duc Cuong

**Version: 2.0**

**Team:** T-B

**Date Create:** 02 – Feb – 2017

International School, Duy Tan University (DTU)

**Document Information**

|  |  |
| --- | --- |
| **Project Title** | Book Reviews |
| **Start Date – End Date** | 01/12/2016 to 3/03/2017 |
| **Lead Institution** | International School, Duy Tan University |
| **Project Mentor** | Pham An Binh |
| **Project Manager & Contact detail** | Nguyen Hoang Quy  K19TPM2, International School, Duy Tan University  Mail: quy.iatb@gmail.com  Tel: +0935713176 |
| **Partner Organization** | Duy Tan University |
| **Team member** | Nguyen Hoang Quy  Phan Quang Tien  Lu Dinh Xuan  Hoang Duc Cuong |

**Document Name**

|  |  |
| --- | --- |
| **Document Title** | Software Requirement Specification document |
| **Reporting period** | March, 2017 |
| **Author** | Nguyen Hoang Quy  Phan Quang Tien  Lu Dinh Xuan  Hoang Duc Cuong |
| **Role** | All team members |
| **Filename** | Software Requirement Specification Document – v2.0 |

**Document Revision History**

|  |  |  |
| --- | --- | --- |
| **Date** | **History Change** | **Version** |
| 2/2/2017 | Create document | 1.0 |
| 7/2/2017 | Update Part 2 | 1.1 |
| 19/2/2017 | Update sequence diagram, activity diagram | 1.2 |
| 23/2/2017 | Update class diagram, database design | 1.3 |
| 25/2/2017 | Update 3.1, 3.2, 3.3 | 1.4 |
| 2/3/2017 | Update class diagram | 1.5 |
| 15/3/2017 | Release |  |

*Content*

[**PART I: INTRODUCTION** 7](#_Toc476729710)

[1.1 Purpose 7](#_Toc476729711)

[1.2 Intended Audience and Reading Suggestions 7](#_Toc476729712)

[1.3 Document Scope 7](#_Toc476729713)

[1.4 References 7](#_Toc476729714)

[1.5 Objects and research method 7](#_Toc476729715)

[1.6 General requirements for project 8](#_Toc476729716)

[1.7 About tools 8](#_Toc476729717)

[**PART II: ANALYSIS AND DESIGN SYSTEM** 10](#_Toc476729718)

[2.1 Overview 10](#_Toc476729720)

[2.2 System functionality 10](#_Toc476729721)

[2.3 Environment 10](#_Toc476729722)

[2.4 Actors 10](#_Toc476729723)

[2.5 System Architecture Overview 11](#_Toc476729724)

[2.6 Description and Priority 11](#_Toc476729725)

[2.7 Use case diagram 12](#_Toc476729726)

[2.7.1 Use case overview 12](#_Toc476729727)

[2.7.2 Use case Access 13](#_Toc476729728)

[2.7.3 Use case Register 14](#_Toc476729729)

[2.7.4 Use case Search 15](#_Toc476729730)

[2.7.5 Use case Review 16](#_Toc476729731)

[2.7.6 Use case Feedback 17](#_Toc476729732)

[2.7.7 Use case Mark 18](#_Toc476729733)

[2.7.8 Use case Payment 19](#_Toc476729734)

[2.7.9 Use case Notify 20](#_Toc476729735)

[2.7.10 Use case Request 21](#_Toc476729736)

[2.7.11 Use case Suggest 22](#_Toc476729737)

[2.7.12 Use case Manage Account 23](#_Toc476729738)

[2.7.13 Use case Report 24](#_Toc476729739)

[2.8 Sequence diagram 25](#_Toc476729740)

[2.8.1 Sequence diagram Access 25](#_Toc476729741)

[2.8.2 Sequence diagram Search 25](#_Toc476729742)

[2.8.3 Sequence diagram Register 26](#_Toc476729743)

[2.8.4 Sequence diagram Manage profile 26](#_Toc476729744)

[2.8.5 Sequence diagram Payment 27](#_Toc476729745)

[2.8.6 Sequence diagram Mark 27](#_Toc476729746)

[2.8.7 Sequence diagram Notify 28](#_Toc476729747)

[2.8.8 Sequence diagram Review 28](#_Toc476729748)

[2.8.9 Sequence Manage User 29](#_Toc476729749)

[2.8.10 Sequence diagram Report 29](#_Toc476729750)

[2.9 Activity Diagram 30](#_Toc476729751)

[2.9.1 Activity diagram overview 30](#_Toc476729752)

[2.9.2 Activity diagram Search 31](#_Toc476729753)

[2.9.3 Activity diagram Register 32](#_Toc476729754)

[2.9.4 Activity diagram Sign in 32](#_Toc476729755)

[2.9.5 Activity diagram Manage profile 33](#_Toc476729756)

[2.9.6 Activity diagram Notify 33](#_Toc476729757)

[2.9.7 Activity diagram Payment 34](#_Toc476729758)

[2.10 Class diagram 34](#_Toc476729759)

[**PART III: DESIGN DATABASE AND USER INTERFACE** 35](#_Toc476729760)

[3.1 Database design 35](#_Toc476729762)

[3.2 Database Specific description 35](#_Toc476729763)

[3.3 User interface design 37](#_Toc476729764)

[3.3.1 Home page 37](#_Toc476729765)

[3.3.2 Signup page 39](#_Toc476729766)

[3.3.3 Signin form 41](#_Toc476729767)

[3.3.4 Profile page 42](#_Toc476729768)

[3.3.5 History Review page 43](#_Toc476729769)

[3.3.6 Admin page 44](#_Toc476729770)

[3.3.7 List Book page 46](#_Toc476729771)

[3.3.8 Sidekiq development page 47](#_Toc476729772)

[**PART IV: SETUP AND IMPLEMENT FOR SYSTEM** 48](#_Toc476729773)

[4.1 Setup 48](#_Toc476729775)

[4.2 Implement 50](#_Toc476729776)

[**SUMMARY** 51](#_Toc476729777)

[**REFERENCES** 52](#_Toc476729778)

**PART I: INTRODUCTION**

## Purpose

The purpose of this document is to provide information on my website – Book Reviews. This is a site running on desktop and mobile. This document includes all the functionality that we will build on this project to developers can approach, read and understand what we will do. Also team members can build a product right.

## Intended Audience and Reading Suggestions

This document is written for developers who develop the game. All of this document is very important to read.

## Document Scope

This document has all information about the project’s functions, the diagram describe workflow and designing database, UI for developers.

## References

* Vision and Scope Document
* \_PTTK – CDIO4 - v1.4
* SE06-SRS-V3.0

## Objects and research method

About object:

* Consumers in nation who has interested in about books and enjoy reading.
* The bookstores want to introduce about the new books.
* Technique about Ruby on Rails and Android app.

About research method:

* Learn about Ruby on Rails and Java Android programing to build a website.
* Understanding about Relation Database Management System - MySQL.
* Collecting requirements from users.
* Research tools to build the system.
* Analysis and design system.

## General requirements for project

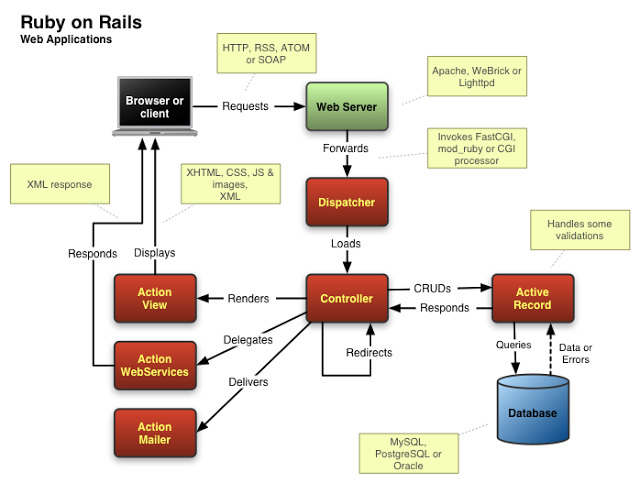
About system:

* Functional
* Manage List Book
* Manage List User
* Register, Sign in
* Manage Category
* Support online
* Review for a book
* Manage Account
* Search
* Non-Functional
* User interface must be friendly, easy to use, easy to customize and change.
* Data must be access quickly, be able to storage good.
* Search for fast and convenient.
* Safety security.

## About tools

* *Ruby on Rails*

Ruby on Rails is a framework written in the Ruby language. It is designed to make programming web applications easier by making assumptions about what every developer needs to get started. It allows you to write less code while accomplishing more than many other languages and frameworks. Experienced Rails developers also report that it makes web application development more fun.

How does it work?

* *MySQL:*

MySQL is a powerful database. It's very good and free of charge. The MySQL™ software delivers a very fast, multi-threaded, multi-user, and robust SQL (Structured Query Language) database server. MySQL Server is intended for mission-critical, heavy-load production systems as well as for embedding into mass-deployed software.

* *AJAX*

AJAX is short for Asynchronous JavaScript and XML) is a set of web development techniques using many web technologies on the client-side to create asynchronous Web applications. With Ajax, web applications can send data to and retrieve from a server asynchronously (in the background) without interfering with the display and behavior of the existing page.

* *Android platform*

Android provides a rich application framework that allows you to build innovative apps and games for mobile devices in a Java language environment. The documents listed in the left navigation provide details about how to build apps using Android's various APIs.

* *Docker and Docker-compose*

Docker is an open-source project that automates the deployment of applications inside software containers.

Docker containers wrap up a piece of software in a complete filesystem that contains everything it needs to run: code, runtime, system tools, system libraries – anything you can install on a server. This guarantees that it will always run the same, regardless of the environment it is running in.

Compose is a tool for defining and running multi-container Docker applications. With Compose, you use a Compose file to configure your application’s services. Then, using a single command, you create and start all the services from your configuration.

Using Compose is basically a three-step process:

1. Define your app’s environment with a Dockerfile so it can be reproduced anywhere.
2. Define the services that make up your app in docker-compose.yml so they can be run together in an isolated environment.
3. Lastly, run docker-compose up and Compose will start and run your entire app.

**PART II: ANALYSIS AND DESIGN SYSTEM**



## Overview

This website makes people come back to the reading culture that is being forgotten slowly. Because it brings books with content, overview, also it is reviewed, rated by so many other people.

## System functionality

* Mark/Unmark
* Rate
* Review
* Request to buy
* Favorites
* Suggest a book
* Payment
* Access
* Manage information
* Manage account
* Send notification

## Environment

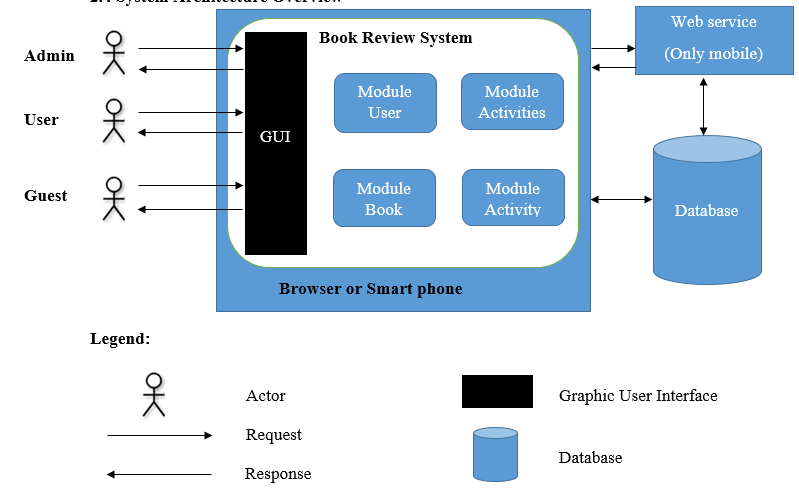
Operating System: Windows XP, 7, 8.1, 10, Linux, Android 4 or above …

Browser: IE 9 or above, Firefox, Safari, Chrome …

## Actors

|  |  |
| --- | --- |
| **Actor** | **Description** |
| Admin | The person who control all activities of the system |
| Users | Member of system who can do basic function on the system. |
| Visitor | The person who just visit and view or search information on website. |

## System Architecture Overview



 **Legend:**

Actor User Graphic Interface

Request Features

Response

Database

Interactive

## Description and Priority

* User can review for a book.
* User can rate for a book.
* User can mark a book is reading or read.
* User can request to buy a book.
* User can suggest a book.
* All of this is important features. Priority: **HIGH**.

## Use case diagram

### Use case overview



|  |  |
| --- | --- |
| Use case ID | UC01 |
| Use case Name | **Đăng nhập** |
| ***Author*** | **Nguyễn Hoàng Quý** |
| Actor:   * Admin, User.   Summary:   * This use case allows the actor access to the system.   Preconditions:   * Admin or user access to website.   PostConditions:   * Have an account of system.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access to the website | System show GUI login | | 2 | Enter username and password |  | | 3 | Click login button | Go to home page. | | |

### Use case Access



|  |  |
| --- | --- |
| Use case ID | UC01 |
| Use case Name | **Access** |
| ***Author*** | **Nguyen Hoang Quy** |
| Actor:   * Admin, User.   Summary:   * This use case allows the actor access to the system followed by the authority of each person.   PreConditions:   * Admin or user must be access to website.   PostConditions:   * Have an account of system.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access to the website | System show GUI login | | 2 | Enter username and password | Require to valid field | | 3 | Click login button | Go to home page. | | |

### Use case Register



|  |  |
| --- | --- |
| Use case ID | UC02 |
| Use case Name | **Register** |
| ***Author*** | **Nguyen Hoang Quy** |
| Actor:   * Visitor   Summary:   * This use case allows the actor register an account and become a member of the system.   PreConditions:   * Visitor must be access to website.   PostConditions:   * Enter all required field.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access to the website | System show GUI login | | 2 | Enter username and password | Require to valid field | | 3 | Click register button | Go to home page. | | |

### Use case Search



|  |  |
| --- | --- |
| Use case ID | UC03 |
| Use case Name | **Search** |
| ***Author*** | **Nguyen Hoang Quy** |
| Actor:   * Admin, User, Visitor   Summary:   * Search information on system.   Preconditions:   * None   PostConditions:   * Display information be found.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Input keyword on search box |  | | 2 | Choose type search |  | | 3 | Click Search | Display all result be found. | | |

### Use case Review



|  |  |
| --- | --- |
| Use case ID | UC04 |
| Use case Name | **Review** |
| ***Author*** | **Nguyen Hoang Quy** |
| Actor:   * Admin, User   Summary:   * User can review a post. * Admin can review a post that is valid or not.   Preconditions:   * Admin or user login success.   PostConditions:   * Posts will be edit by admin.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User click on review function in the post. | Show a field to review for post. | | 2 | Write a review |  | | 3 | Click review button | Submit review to post. | | |

### Use case Feedback



|  |  |
| --- | --- |
| Use case ID | UC05 |
| Use case Name | **Feedback** |
| ***Author*** | **Nguyen Hoang Quy** |
| Actor:   * Admin, User, Visitor   Summary:   * Send a feedback about an error or something wrong to system.   Preconditions:   * None   PostConditions:   * Feedback should be submitted and handled by admin.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Click on the feedback function in home page | Show a field to review for post. | | 2 | Choose a type for feedback |  | | 3 | Write a content |  | | 4 | Submit to send. |  | | |

### Use case Mark



|  |  |
| --- | --- |
| Use case ID | UC06 |
| Use case Name | **Mark** |
| ***Author*** | **Nguyen Hoang Quy** |
| Actor:   * Admin, User   Summary:   * Keep a book is reading or has read.   Preconditions:   * Admin or user login success.   PostConditions:   * The book should be existed.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Mark on a book | Change state of book. | | 2 | Choose book’s state is reading or has read. |  | | 3 | Click on that state. |  | | |

### Use case Payment



|  |  |
| --- | --- |
| Use case ID | UC07 |
| Use case Name | **Payment** |
| ***Author*** | **Nguyen Hoang Quy** |
| Actor:   * Admin, User   Summary:   * Payment system for website.   Preconditions:   * User must be authorized * If log in time out, the system presents a message saying that the log in failed.   PostConditions:   * The request must be handled by admin. * It must be feasible   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Click on function payment in website |  | | 2 | Specifies information of user. |  | | 3 | Checkout |  | | 4 | Handle and response by admin. |  | | |

### Use case Notify



|  |  |
| --- | --- |
| Use case ID | UC08 |
| Use case Name | **Notify** |
| ***Author*** | **Nguyen Hoang Quy** |
| Actor:   * Admin   Summary:   * Send notification to user about something new or alert from website.   Preconditions:   * Admin login success.   PostConditions:   * The user destination must exist.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Choose some user to send |  | | 2 | Write a content of notification |  | | 3 | Submit to send |  | | |

### Use case Request



|  |  |
| --- | --- |
| Use case ID | UC09 |
| Use case Name | **Request** |
| ***Author*** | **Nguyen Hoang Quy** |
| Actor:   * User, Admin   Summary:   * User can send a request for book to admin.   Preconditions:   * User must be authorized. * The book that is requested must be exist.   PostConditions:   * None   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Choose a book to request |  | | 2 | Specifies information of user. |  | | 3 | Submit to send to admin. |  | | 4 | Admin handle request and |  | | |

### Use case Suggest



|  |  |
| --- | --- |
| Use case ID | UC10 |
| Use case Name | **Suggest** |
| ***Author*** | **Nguyen Hoang Quy** |
| Actor:   * Admin   Summary:   * Suggest a book into website   Preconditions:   * Admin must be authorized.   PostConditions:   * None   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Create a book |  | | 2 | Load into system. |  | | 3 | Write a basic review for book |  | | 4 | Submit |  | | |

### Use case Manage Account



|  |  |
| --- | --- |
| Use case ID | UC11 |
| Use case Name | **Manage Account** |
| ***Author*** | **Nguyen Hoang Quy** |
| Actor:   * Admin   Summary:   * User can manage own profile. * Admin can manage list user.   Preconditions:   * Admin must be authorized.   PostConditions:   * None   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Click on avatar or username. |  | | 2 | Edit or update profile |  | | 3 | Submit to change |  | | |

### Use case Report



|  |  |
| --- | --- |
| Use case ID | UC12 |
| Use case Name | **Report** |
| ***Author*** | **Nguyen Hoang Quy** |
| Actor:   * Admin   Summary:   * Create a report for system   Preconditions:   * Admin must be authorized.   PostConditions:   * None   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Click on avatar or username. |  | | 2 | Edit or update profile |  | | 3 | Submit to change |  | | |

## Sequence diagram

### Sequence diagram Access



### Sequence diagram Search



### Sequence diagram Register



### Sequence diagram Manage profile



### Sequence diagram Payment



### Sequence diagram Mark



### Sequence diagram Notify



### Sequence diagram Review



### Sequence Manage User



### Sequence diagram Report



## Activity Diagram

### Activity diagram overview



### Activity diagram Search



### Activity diagram Register



### Activity diagram Sign in



### Activity diagram Manage profile



### Activity diagram Notify



### Activity diagram Payment



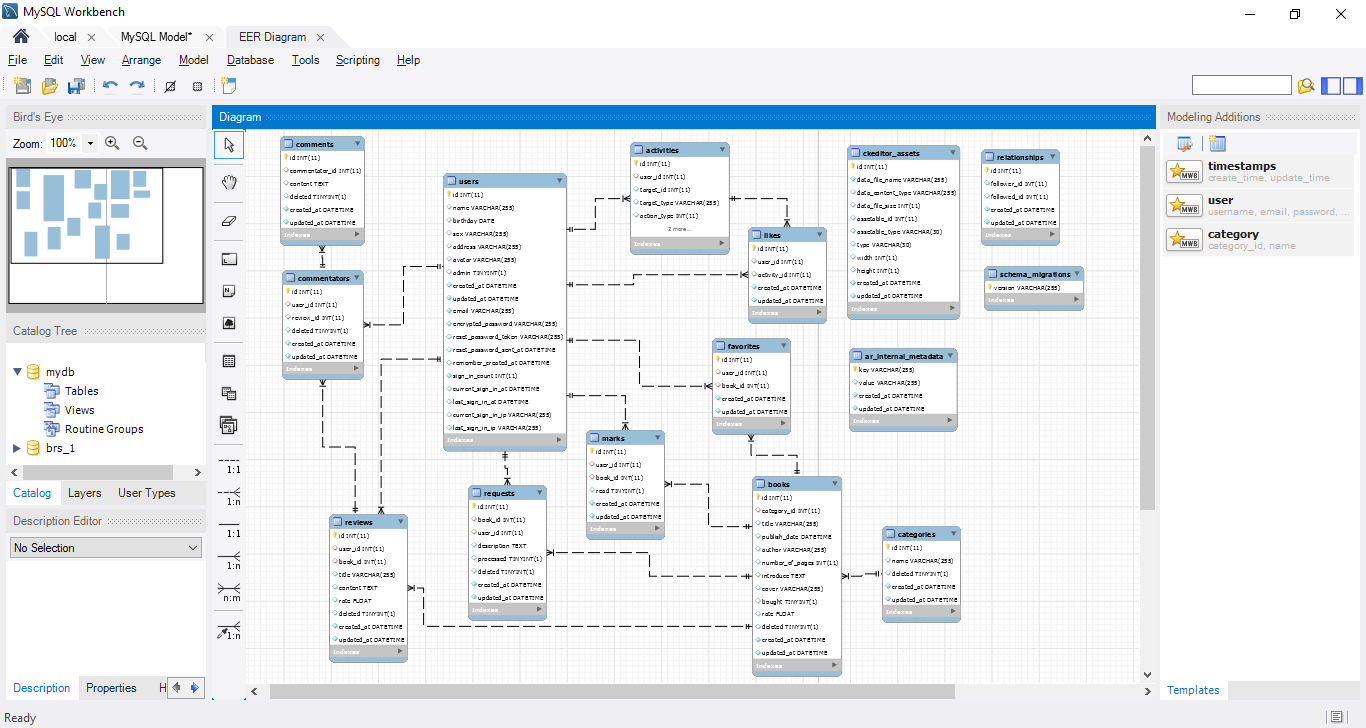
## Class diagram



**PART III: DESIGN DATABASE AND USER INTERFACE**



## Database design



## Database Specific description

* Table users:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Columns** | **Description** | **Type** | **Size** | **Note** |
| `id` |  | Int | 11 | AI PK |
| `name` |  | Varchar | 255 |  |
| `birthday` |  | date |  |  |
| `sex` |  | Varchar | 255 |  |
| `address` |  | Varchar | 255 |  |
| `avatar` |  | Varchar | 255 |  |
| `admin` |  | Tinyint | 1 | 1 = true  0 = false |
| `email` |  | Varchar | 255 |  |
| `encrypted\_password` |  | Varchar | 255 |  |
| `reset\_password\_token` |  | Varchar | 255 |  |
| `sign\_in\_count` |  | Int | 11 |  |
| `current\_sign\_in\_at` |  | datetime |  |  |
| `last\_sign\_in\_at` |  | Datetime |  |  |

* Table books:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Columns** | **Description** | **Type** | **Size** | **Note** |
| `id` |  | Int | 11 | AI PK |
| `category\_id` |  | Int | 11 |  |
| `title` |  | Varchar | 255 |  |
| `publish\_date` |  | Datetime |  |  |
| `author` |  | Varchar | 255 |  |
| `number\_of\_pages` |  | Int | 11 |  |
| `introduce` |  | Text |  |  |
| `cover` |  | Varchar | 255 |  |
| `bought` |  | Tinyint | 1 |  |
| `rate` |  | Float |  |  |
| `deleted` |  | Tinyint | 1 |  |

* Table relationships:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Columns** | **Description** | **Type** | **Size** | **Note** |
| `id` |  | Int | 11 | AI PK |
| `follower\_id` |  | Int | 11 |  |
| `followed\_id` |  | Int | 11 |  |
| `publish\_date` |  | Datetime |  |  |

* Table activities:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Columns** | **Description** | **Type** | **Size** | **Note** |
| `id` |  | Int | 11 | AI PK |
| `user\_id` |  | Int | 11 |  |
| `target\_id` |  | Int | 11 |  |
| `target\_type` |  | Varchar | 255 |  |
| `action\_type` |  | Int | 11 |  |

* Table requests:

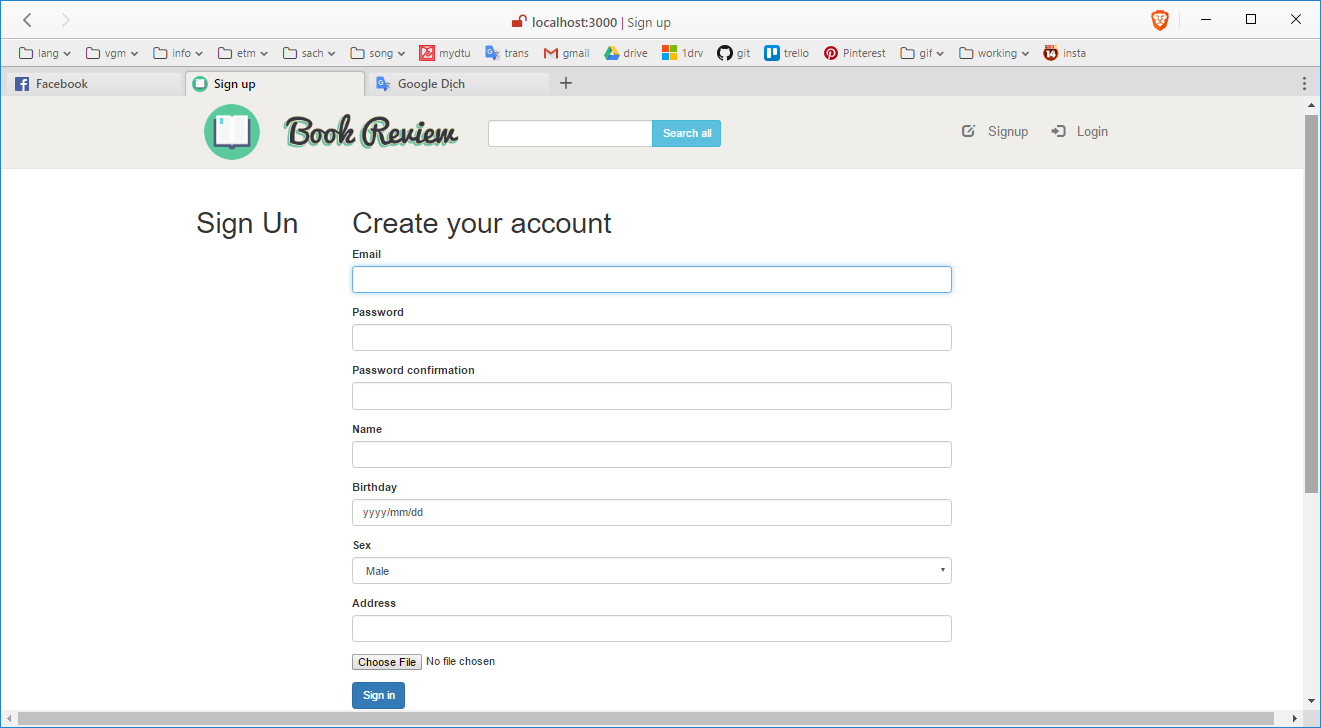
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Columns** | **Description** | **Type** | **Size** | **Note** |
| `id` |  | Int | 11 | AI PK |
| `book\_id` |  | Int | 11 |  |
| `user\_id` |  | Int | 11 |  |
| `description` |  | Text |  |  |
| `processed` |  | Tinyint | 1 |  |
| `deleted` |  | Tinyint | 1 |  |

## User interface design

### C:\Users\iamme\OneDrive\Pictures\Screenshots\2017-02-22 (2).pngHome pageC:\Users\iamme\OneDrive\Pictures\Screenshots\2017-02-22 (1).pngC:\Users\iamme\OneDrive\Pictures\Screenshots\2017-02-22.png

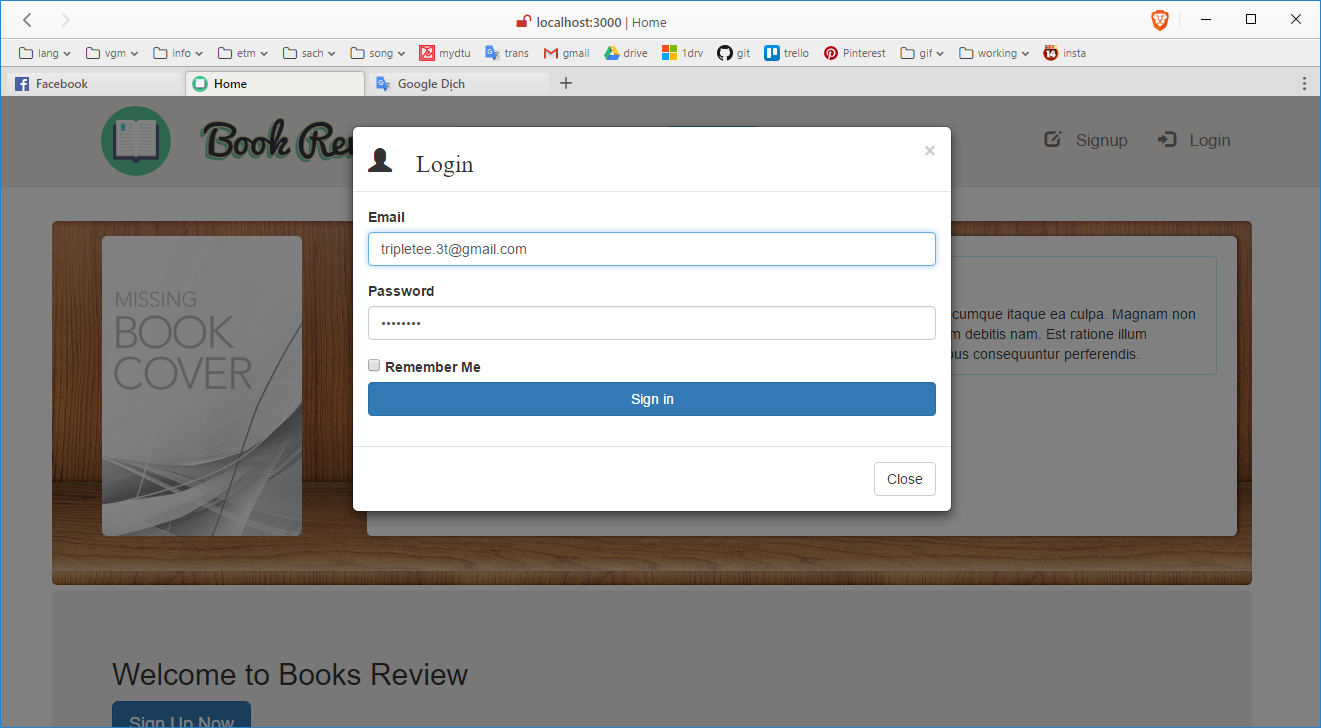
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Screen** | Home page | | | | |
| **Descripton** | | First page when access into website | | | |
| **Access** | | Boot into URL website. | | | |
| **Main content** | | | | | |
| **Object** | | **Type** | **Data** | **Description** | |
| Book Logo | | Image – Link | None | Link to home page. | |
| Book Review Logo | | Image – Link | None | Link to list book page. | |
| Search | | Text – Input | Search in database,  Table POSTS | Input keyword to search | |
| Signup | | Label – Link | None | Link to signup page | |
| Signin | | Label – Link | None | Link to signin form | |
| Slider | | Image – Link | Data in table POSTS | Show a few new books. | |
| Left menu | | Label – Link | Data in table CATEGORIES | Show all category of book. | |
| Paginate | | Button – Link | None | Show books limit for each page. | |
| **Action** | | | | | |
| **Name** | | **Description** | | **Success** | **Fail** |
| Search | | Choose a category, input keyword and submit. | | Display all book of result. | Display flash message error. |
| Rating on book | | Rating for each book by click on stars. | | Set a rate value on book by a user | Display flash message require to signin |
| Name of book | | Display detail of book. | | Go to book detail page | Display flash message error |
| Paginate | | Show books limit for each page. | | Display selected posts on the page | Display flash message page not found |

### Signup page



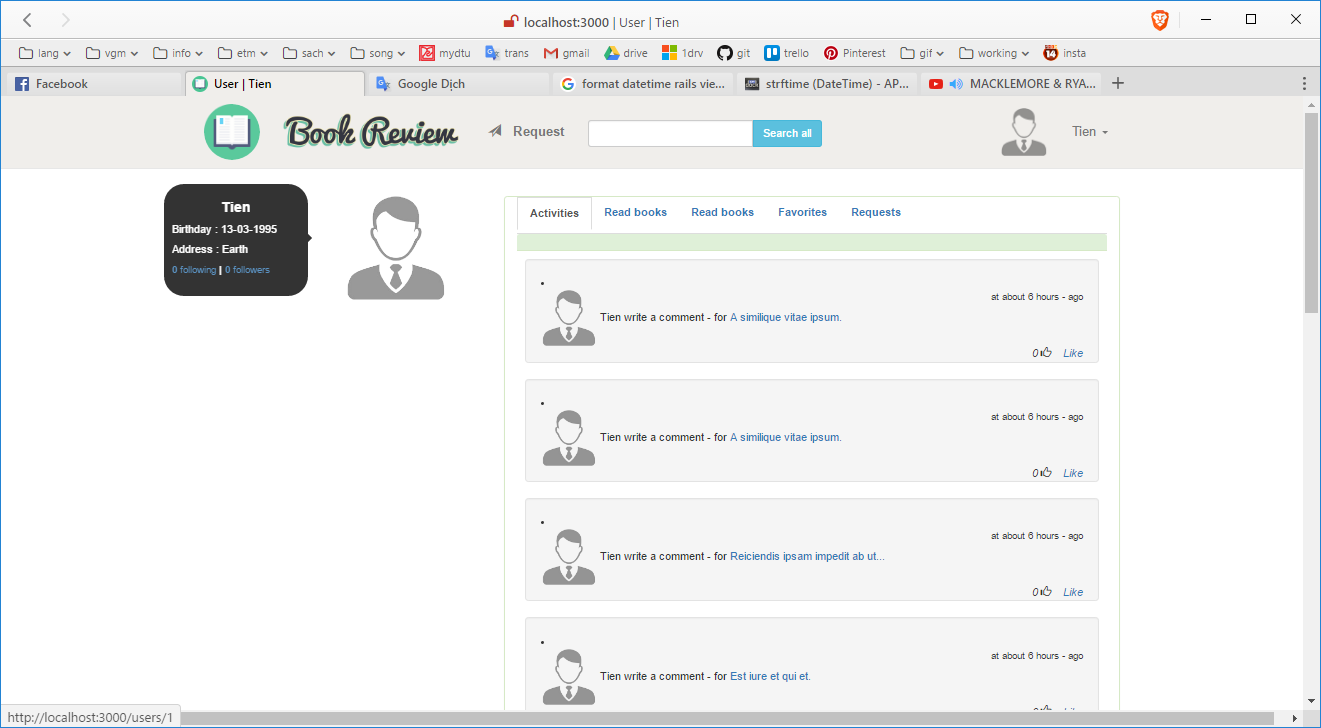
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Screen** | Signup page | | | | |
| **Descripton** | | Page to registry an account. | | | |
| **Access** | | Link on navigation bar of website. | | | |
| **Main content** | | | | | |
| **Object** | | **Type** | **Data** | **Description** | |
| Email | | Text – Input | Field email in table USERS | Email input field | |
| Password | | Text – Input | Field password in table USERS | Password input field | |
| Password confirm | | Text – Input | Field password in table USERS | Password confirm input field | |
| Name | | Text – Input | Field name in table USERS | Name input field | |
| Birthday | | Selectbox – Input | Field birthday in table USERS | Birthday select field | |
| Sex | | Dropdown list – Input | Field sex in table USERS | Email select field | |
| Address | | Text – Input | Field address in table USERS | Address input field | |
| Avatar | | Button – Input | Field avatar in table USERS | Load image file from your PC. | |
| **Action** | | | | | |
| **Name** | | **Description** | | **Success** | **Fail** |
| Submit | | Send a request to create an account to server. | | Login new account and go to home page. | Display flash message error and back to home page. |

### Signin form



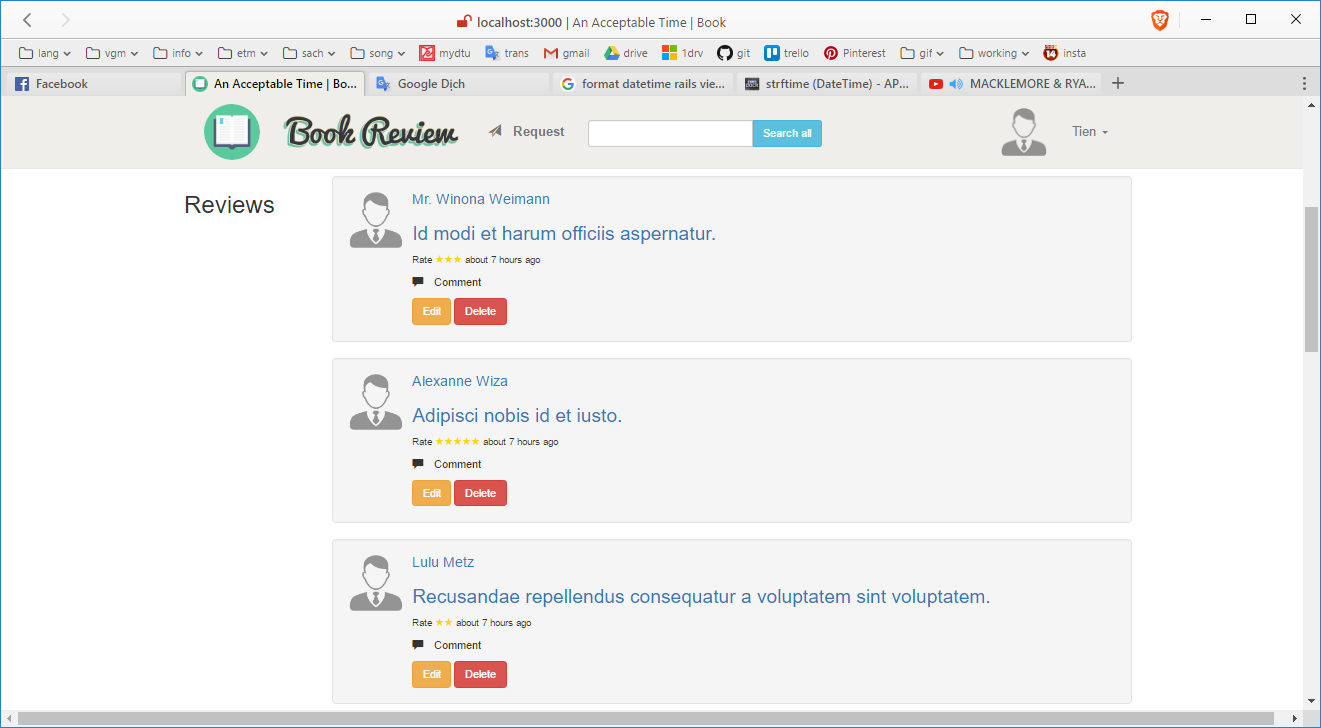
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Screen** | Signin form | | | | |
| **Descripton** | | A pop-up form to sign in an account into system. | | | |
| **Access** | | Link on navigation bar of website. | | | |
| **Main content** | | | | | |
| **Object** | | **Type** | **Data** | **Description** | |
| Email | | Text – Input | Field email in table USERS | Email input field | |
| Password | | Text – Input | Field password in table USERS | Password input field | |
| Remember | | Checkbox | None |  | |
| Signin | | Submit – Button | None | Submit data. | |
| Cancel | | Close – Button | None | Close form and back to current page. | |
| **Action** | | | | | |
| **Name** | | **Description** | | **Success** | **Fail** |
| Submit | | Send a request to signin an account into server. | | Login and go to home page. | Display flash message error |

### Profile page



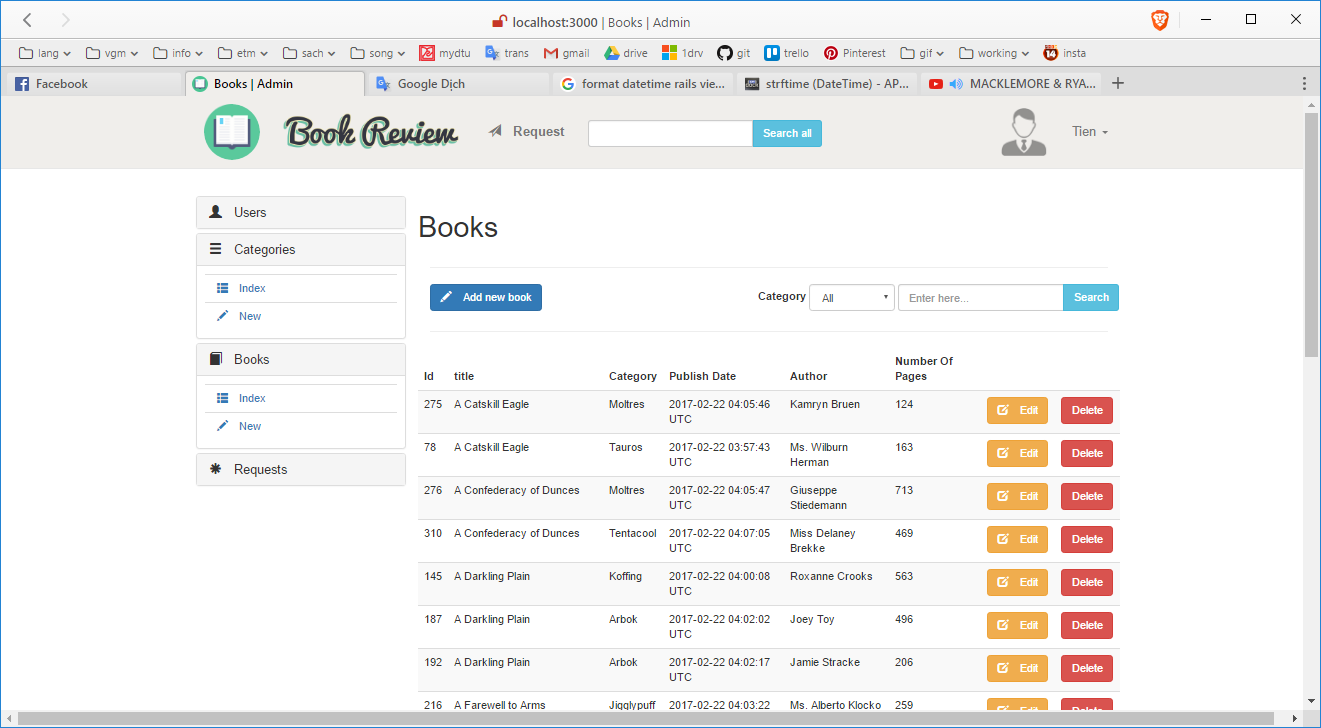
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Screen** | Profile page | | | | |
| **Descripton** | | Manage information and history activity of user. | | | |
| **Access** | | Link on avatar of user. | | | |
| **Main content** | | | | | |
| **Object** | | **Type** | **Data** | **Description** | |
| Name | | Label | Field name in table USERS |  | |
| Birthday | | Label | Field birthday in table USERS |  | |
| Address | | Label | Field address in table USERS |  | |
| Following | | Label – Link | None | Link to list following account. | |
| Follower | | Label – Link | None | Link to list follow account. | |
| Navtab | | Label – Tab |  | History activity | |
| **Action** | | | | | |
| **Name** | | **Description** | | **Success** | **Fail** |
| Submit | | Send a request to signin an account into server. | | Login and go to home page. | Display flash message error |

### History Review page



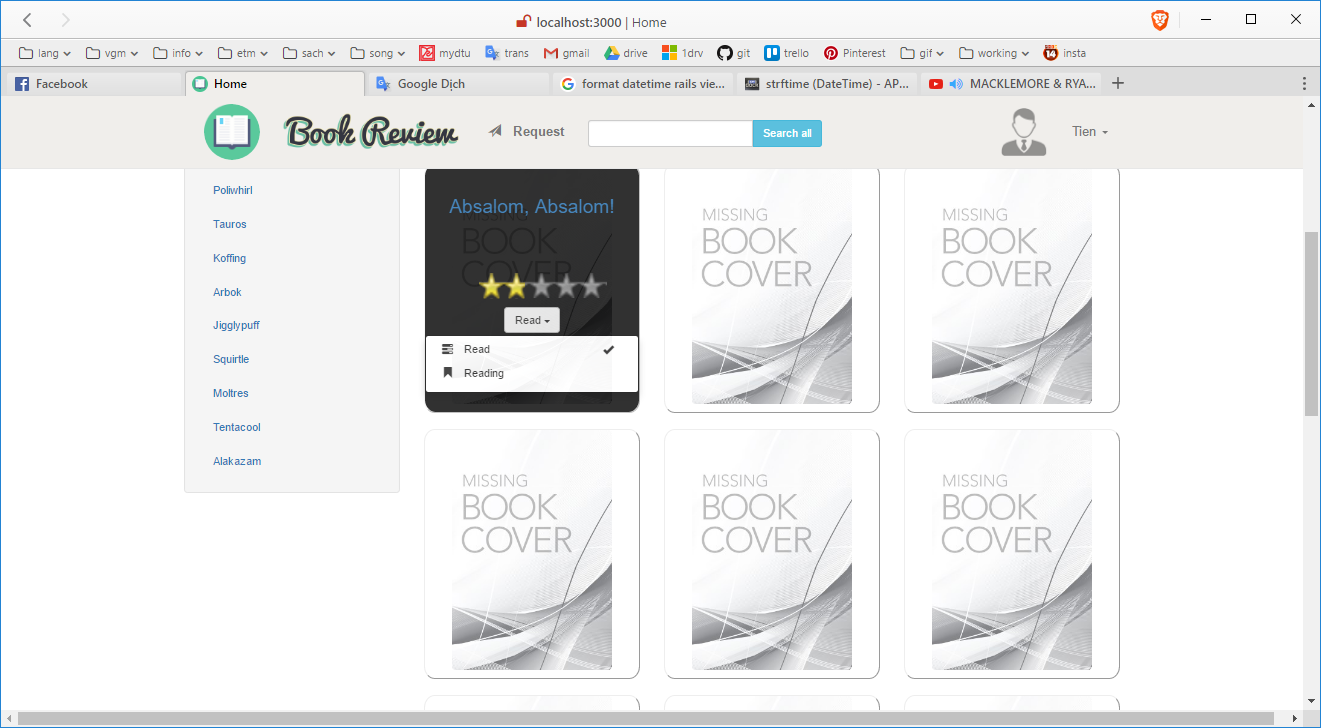
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Screen** | History review page | | | | |
| **Descripton** | | List review of user | | | |
| **Access** | | Link on profile page. | | | |
| **Main content** | | | | | |
| **Object** | | **Type** | **Data** | **Description** | |
| Review | | Label | Field content in table REVIEWS |  | |
| Avatar | | Image – Link | Field avatar in table USERS |  | |
| Comment | | Label – Link | None | Create a comment on review | |
| Edit | | Button – Link | None | Link to edit review. | |
| Delete | | Label – Link | None | Link to delete review. | |
| **Action** | | | | | |
| **Name** | | **Description** | | **Success** | **Fail** |
| Edit | | Go to Edit review page | | Go to edit review page | Display flash message error |
| Delete | | Send request to delete review. | | Display flash message success | Display flash message error |

### Admin page



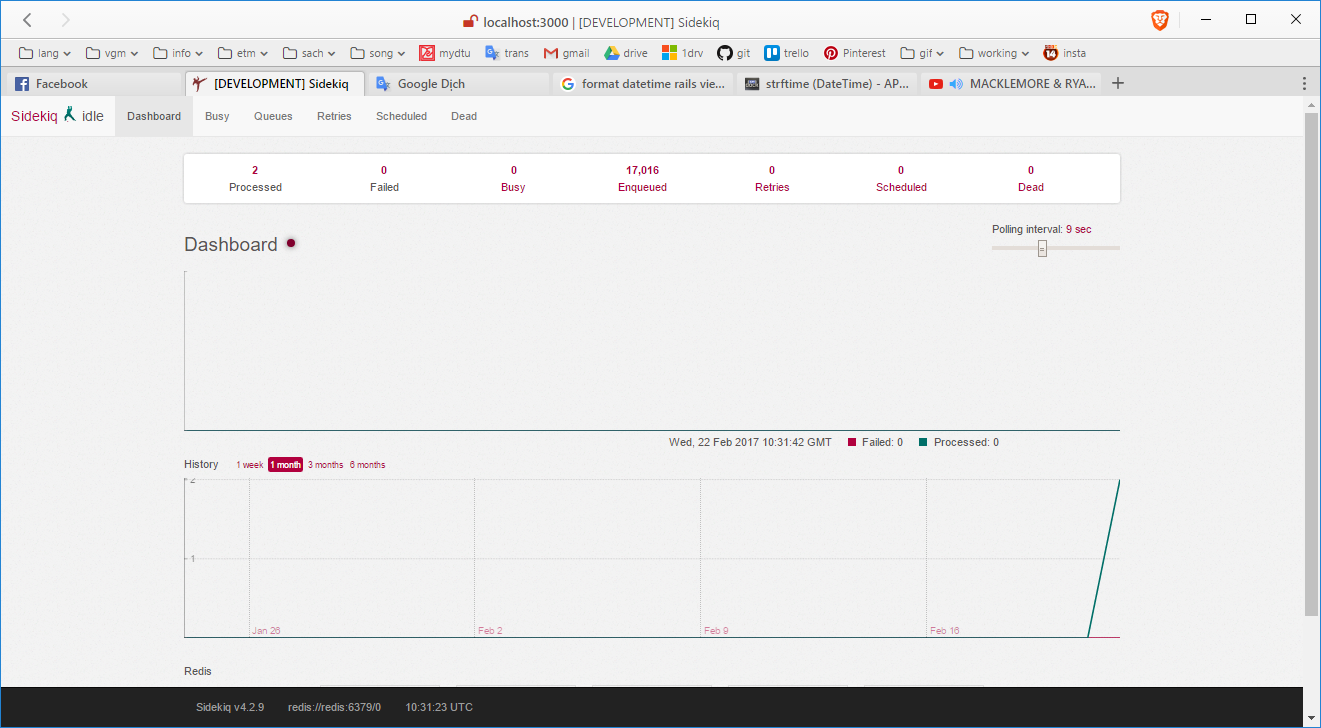
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Screen** | Admin page | | | | |
| **Descripton** | | Manage activity on website in here. | | | |
| **Access** | | Link on name of user. | | | |
| **Main content** | | | | | |
| **Object** | | **Type** | **Data** | **Description** | |
| Menu | | Label | None |  | |
| Search | | Image – Link | None |  | |
| List | | Table | Data from table BOOKS |  | |
| Add | | Button – Link | None | Link to add a book | |
| Edit | | Button – Link | None | Link to edit a book | |
| Delete | | Label – Link | None | Link to delete a book. | |
| **Action** | | | | | |
| **Name** | | **Description** | | **Success** | **Fail** |
| Edit | | Go to Edit book page | | Go to edit book page | Display flash message error |
| Add | | Go to Add a book | | Go to add book page | Display flash message error |
| Delete | | Delete that book. | | Display flash message success | Display flash message error |

### List Book page



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Screen** | List book page | | | | |
| **Descripton** | | List all book in database. | | | |
| **Access** | | Link on navigation bar or search on box. | | | |
| **Main content** | | | | | |
| **Object** | | **Type** | **Data** | **Description** | |
| Cover | | Image | Field cover from table BOOKS |  | |
| Title | | Label – Link | Field title from table BOOKS |  | |
| List book | | Table | Data from table BOOKS |  | |
| Star | | Image – Link | Field rate from table BOOKS |  | |
| Mark | | Button – Link | Field read from table MARK |  | |
| **Action** | | | | | |
| **Name** | | **Description** | | **Success** | **Fail** |
| Rate | | Rating for that book. | | Check rated or unrated | None |
| Mark | | Mark for that book | | Mark as read or reading | None |

### Sidekiq development page



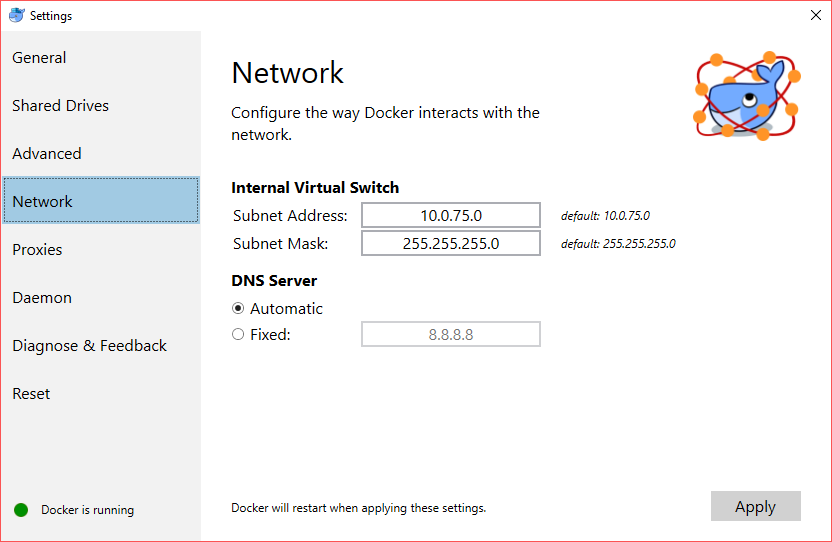
**PART IV: SETUP AND IMPLEMENT FOR SYSTEM**



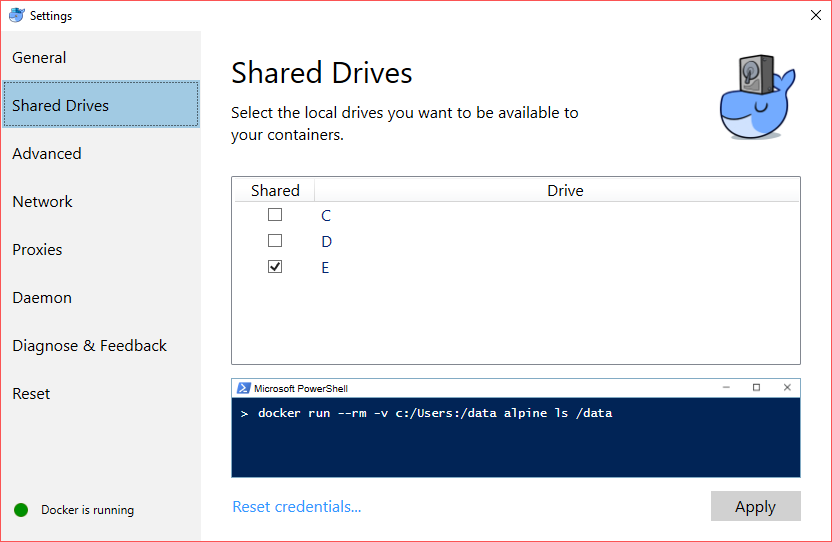
## Setup

* Setup for Windows:

1. Download and install Docker from: <https://www.docker.com/>
2. Setting:



Share your local drives (volumes) with Docker for Windows, so that they are available to your containers



1. Check version

**PS** **C**:\**Users**\**jdoe**> **docker** **--version**

**Docker** **version** 1.12.0, **build** 8**eab29e**, **experimental**

**PS** **C**:\**Users**\**jdoe**> **docker-compose** **--version**

**docker-compose** **version** 1.8.0, **build** **d988a55**

**PS** **C**:\**Users**\**jdoe**> **docker-machine** **--version**

**docker-machine** **version** 0.8.0, **build** **b85aac1**

* Setup for Linux:
  1. Set up the repository

OS requirements: Yakkety 16.10, Xenial 16.04 (LTS), Trusty 14.04 (LTS)

Install packages to allow apt to use a repository over HTTPS:

$ sudo apt-get **install** -y *--no-install-recommends \* apt-transport-https \ ca-certificates \ curl \ software-properties-common

Add Docker’s official GPG key:

$ curl -fsSL https://apt.dockerproject.org/gpg | sudo apt-key add -

Verify that the key ID is 58118E89F3A912897C070ADBF76221572C52609D.

$ apt-key fingerprint 58118E89F3A912897C070ADBF76221572C52609D

Use the following command to set up the stable repository.

$ sudo add-apt-repository \ "deb https://apt.dockerproject.org/repo/ \ ubuntu-$(lsb\_release -cs) \ main"

* 1. Install Docker

Update the apt package index.

$ sudo apt-get **update**

Install the latest version of Docker, or go to the next step to install a specific version. Any existing installation of Docker is replaced.

$ sudo apt-get -y **install** docker-**engine**

* 1. Install Docker-compose

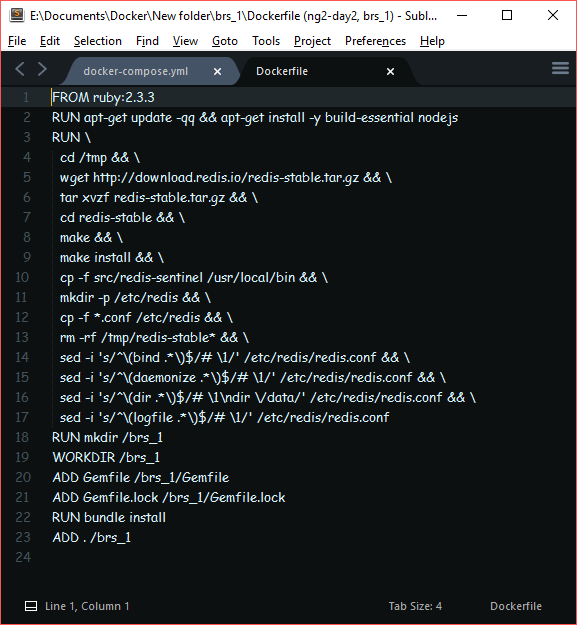
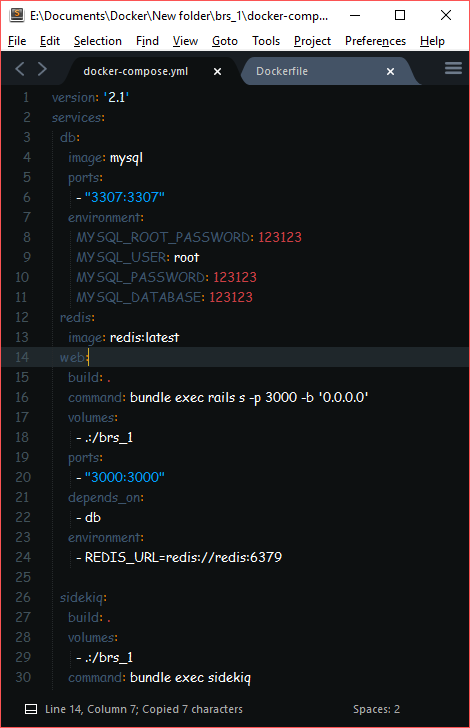
$ curl -L "https://github.com/docker/compose/releases/download/1.11.1/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

* 1. Check version docker

$ docker *–version*

$ docker-compose *--version*

## Implement

1. Check Dockerfle and docker-compose.yml files.
2. Build the project

Run command promt at project folder and using:

$docker build

At the first time build project we need to create database and data for it:

$docker-compose run web rails db:create

$docker-compose run web rails db:migrate

$docker-compose run web rails db:seed

Boot the web with:

$docker-compose up

Finally, open browser and access localhost at port configured in docker-compose.yml file.

**SUMMARY**

1. **Result**

* Understanding about Ruby on Rails and Android platform
* Build a web app with the request is assigned
* Apply Ruby on Rails and Android platform on it

1. **Advantage and disadvantage of project**

* **Advantage:**
  + Build full of features
  + Using docker, docker-compose to build develop environment
  + Using git to manage version of project
  + Build web app with internationalization
* **Disadvantage**

1. **Develop in future**

**REFERENCES**