Software Design Document

for

Wat Duang Dee Web Application

Version 1.0 approved

Prepared by sb2store group

Tortrakul Chathep 552115027  
Nitikorn Samutphet 552115034  
Watcharanan Thongma 552115064  
Sukirt Sithichaisawad 552115075  
Suriya Sasujit 552115077

April 21, 2015

Table of Contents

Table of Contents 1

Revision History 1

1. User Requirement Specification 2

2. User Requirement Specification with System Requirement Specification 4

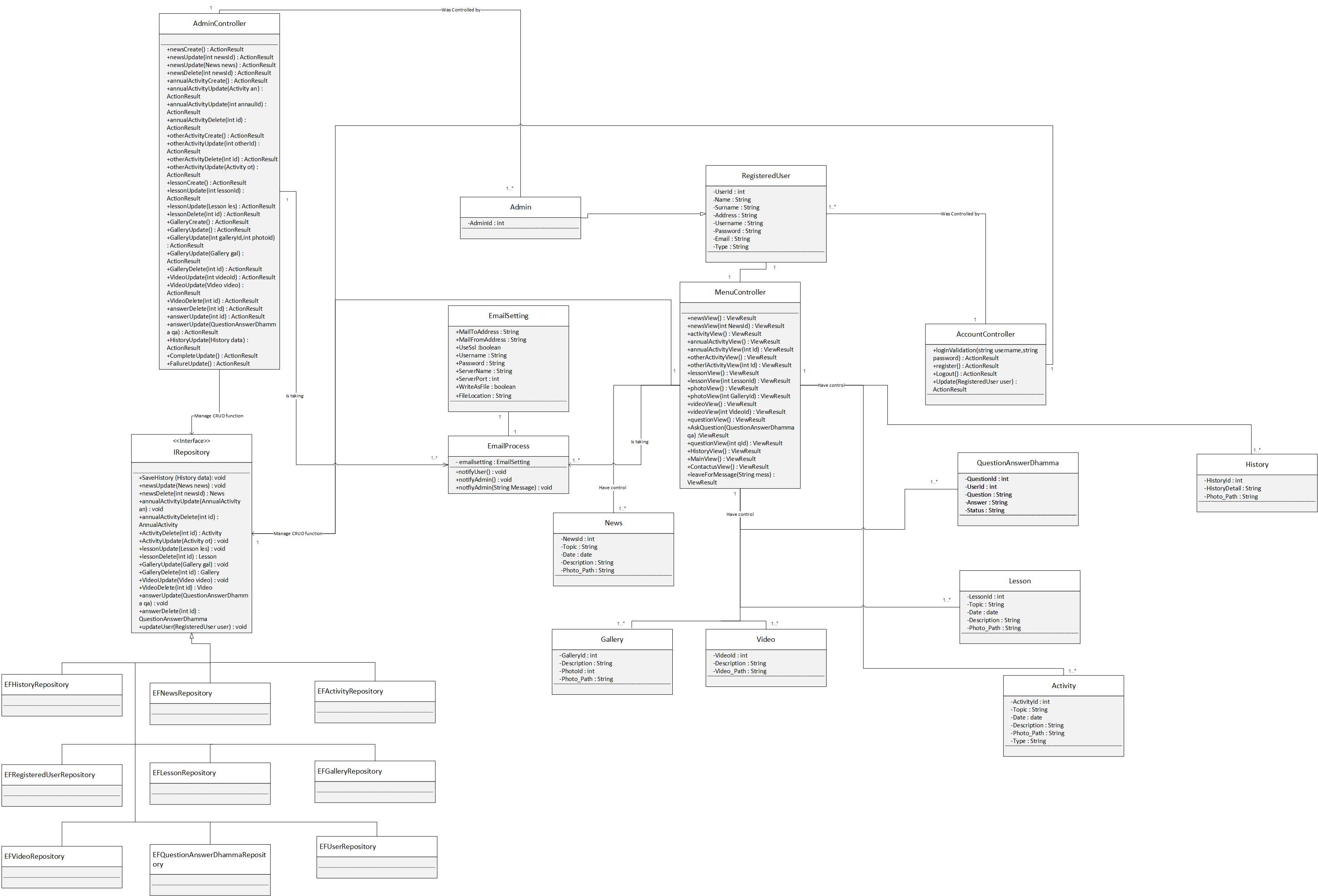
3. Use Case 26

**3.1 Use Case Diagram 26**

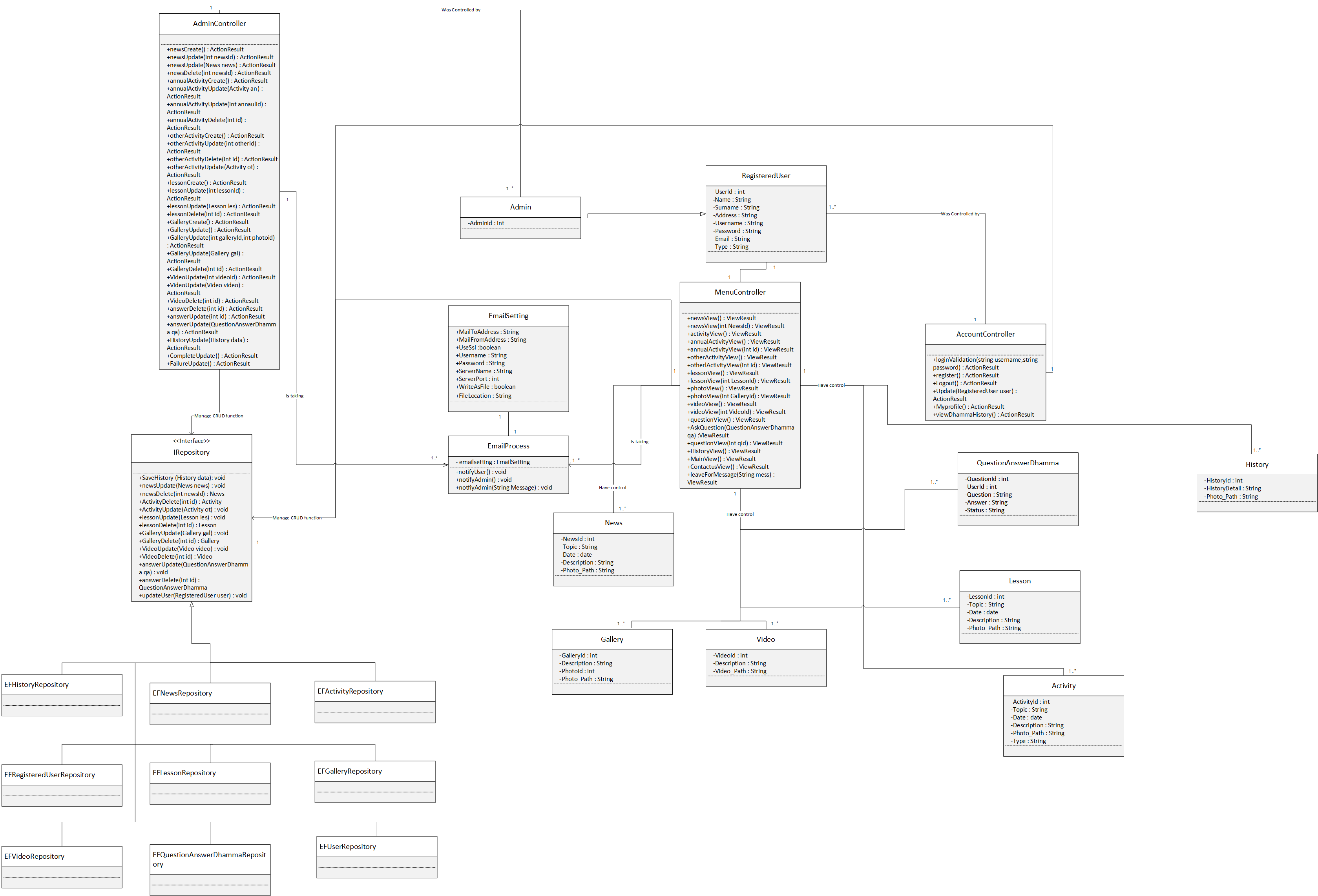
**3.2 Use Case Description and Activity Diagram 29**

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Sb2store group. | 22-04-2015 | Create | 1.0 |
|  |  |  |  |

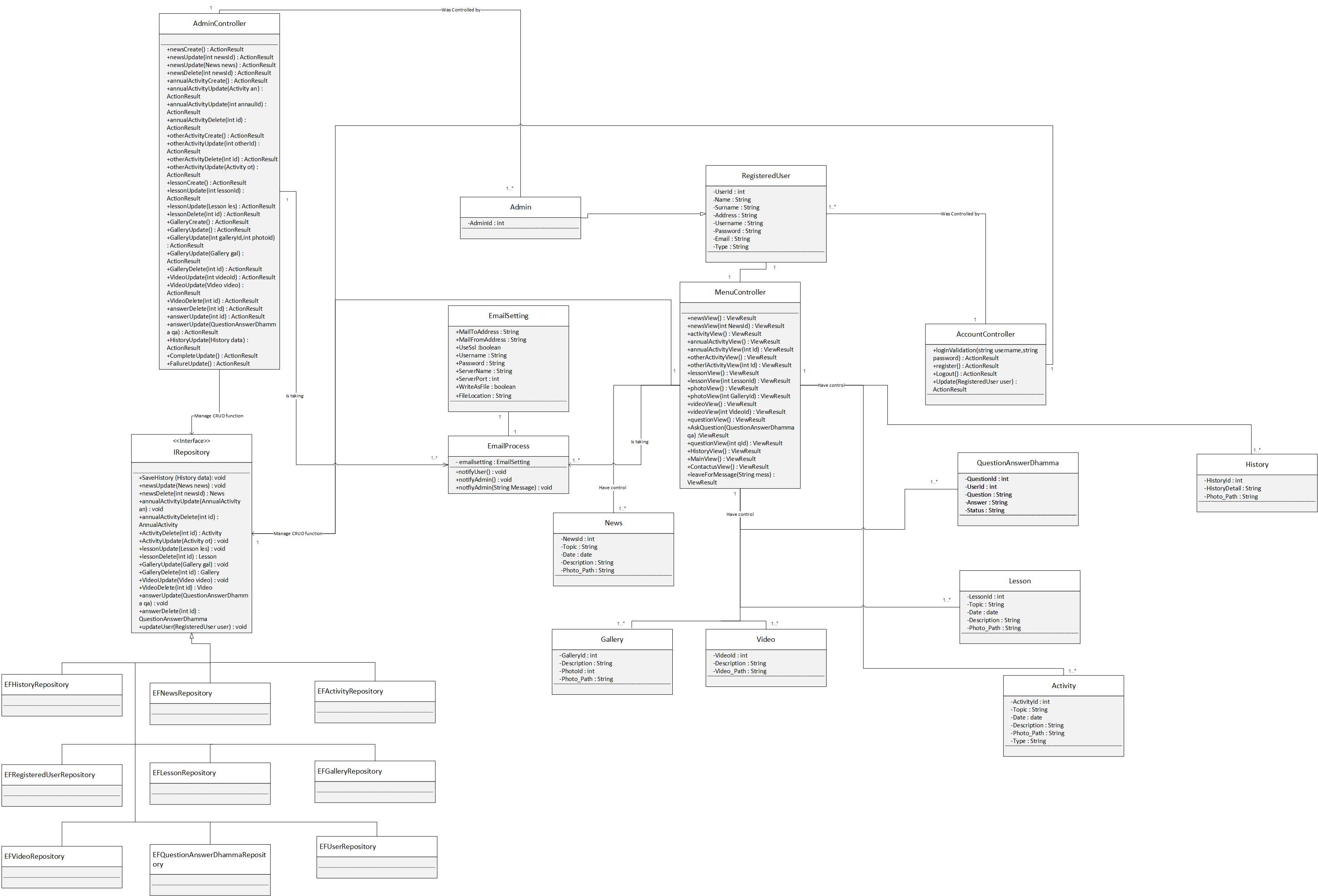
**1. Class Diagram**

**Figure 47 Class Diagram (CD-01) for the entire system of Wat Duang Dee web application**

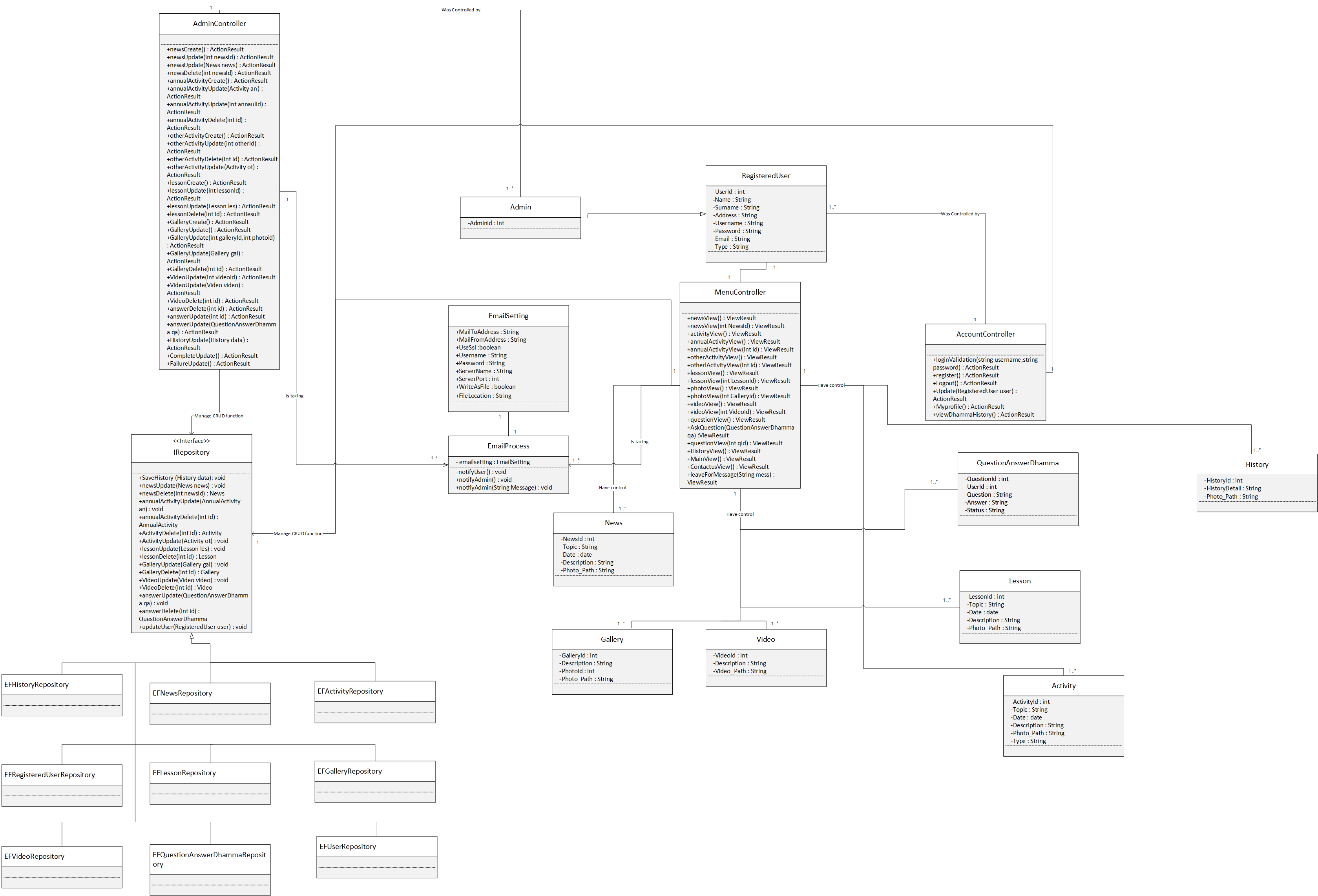


**Figure 48**

**Expand the first part of Class Diagram (CD-01) of Wat Duang Dee web application**



**Figure 49 expand the second part of Class Diagram (CD-01) of Wat Duang Dee web application**



**Figure 50 expand the third part of Class Diagram (CD-01) of Wat Duang Dee web application**

- Class Diagram (CD-01) was designed by using Mode-View-Controller concept with an entity framework for manage a database.

**2. The method design.**

**2.1 MenuController Class**



**Figure 51 MenuController class**

This class is a controller class that manage all of the view and models that is the main function of the Wat Duang Dee web application system

The following user requirement specification which covered in this class.

**URS-03:** The users can view a main page.

**URS-04:** The users can view a temple’s annual activity.

**URS-05:** The users can view a temple’s other activity.

**URS-06:** The users can view a temple’s history.

**URS-07:** The registered users can ask the dhamma question.

**URS-08:** The registered users can view the dhamma answer.

**URS-09:** The users can view the temple’s photo gallery.

**URS-10:** The users can view the temple’s video.

**URS-11:** The users can view the temple’s map.

**URS-12:** The users can view a contact us page.

**URS-13:** The users can access the temple’s Facebook page.

**URS-41:** The users can view the temple’s new page.

**URS-42:** The users can view the photo in a gallery in the temple’s photo gallery.

**URS-43:** The registered users can leave a message for the temple.

**URS-44:** The users can view a dhamma lesson.

**Method name:** newsView

public ViewResult newsView () {}

**Description:** This method is will direct users to the news page.

**Returns:** returnView();

**Method name:** newsView

public ViewResult newsView (int NewsId) {}

**Description:** This method is will direct users to see each of news.

**Returns:** returnView();

**Method name:** activityView

public ViewResult activityView () {}

**Description:** This method is will direct users to main activity page.

**Returns:** returnView();

**Method name:** annualActivityView

public ViewResult annualActivityView () {}

**Description:** This method is will direct users to the annual activity page.

**Returns:** returnView();

**Method name:** annualActivityView

public ViewResult annualActivityView(int Id) {}

**Description:** This method is will direct users to each of annual activity.

**Returns:** returnView();

**Method name:** otherActivityView

public ViewResult otherActivityView() {}

**Description:** This method is will direct users to the other activity page.

**Returns:** returnView();

**Method name:** otherActivityView

public ViewResult otherActivityView(int Id) {}

**Description:** This method is will direct users to each of other activity.

**Returns:** returnView();

**Method name:** lessonView

public ViewResult lessonView() {}

**Description:** This method is will direct users to the dhamma lesson page

**Returns:** returnView();

**Method name:** lessonView

public ViewResult lessonView(int LessonId) {}

**Description:** This method is will direct users to each dhamma lesson.

**Returns:** returnView();

**Method name:** photoView

public ViewResult photoView() {}

**Description:** This method is will direct users to the photo gallery page

**Returns:** returnView();

**Method name:** photoView

public ViewResult photoView(int GalleryId) {}

**Description:** This method is will direct users to see the photo in a gallery

**Returns:** returnView();

**Method name:** videoView

public ViewResult videoView() {}

**Description:** This method is will direct users to the video gallery page

**Returns:** returnView();

**Method name:** videoView

public ViewResult videoView(int VideoId) {}

**Description:** This method is will direct users to see each video in a gallery

**Returns:** returnView();

**Method name:** questionView

public ViewResult questionView() {}

**Description:** This method is will direct users to the Dhamma Q/A page

**Returns:** returnView();

**Method name:** AskQuestion

public ViewResult AskQuestion(QuestionAnswerDhamma qa) {}

**Description:** This method is will direct users to ask the dhamma in the Dhamma Q/A page

**Returns:** returnView(qa);

**Method name:** questionView

public ViewResult questionView(int qId) {}

**Description:** This method is will direct users to see each dhamma question and answered in a Dhamma Q/A page

**Returns:** returnView();

**Method name:** HistoryView

public ViewResult HistoryView() {}

**Description:** This method is will direct users to the history page

**Returns:** returnView();

**Method name:** MainView

public ViewResult MainView() {}

**Description:** This method is will direct users to the main page

**Returns:** returnView();

**Method name:** ContactusView

public ViewResult ContactusView () {}

**Description:** This method is will direct users to the contact us page

**Returns:** returnView();

**Method name:** ContactusView

public ViewResult ContactusView(String mess) {}

**Description:** This method is will received a message of users from the contact us page

**Returns:** returnView();

**2.2 AccountController Class**



**Figure 52 AccountController class**

This class is a controller class that manage all of the view and models that is about the account of users in the Wat Duang Dee web application system

The following user requirement specification which covered in this class.

**URS-39:** The registered users can update their user profile.

**URS-40:** The registered users can view their dhamma question history.

**URS-01:** The registered users can log in to the system.

**URS-02:** The registered users can log out from the system.

**URS-14:** The users can register to be a registered user.

**Method name:** loginValidation

public ViewResult loginValidation(string username,string password){}

**Description:** This method is will received the username and password all validate wheatear the username and password is match or not.

**Returns:** returnView();

**Method name:** register

public ViewResult register(){}

**Description:** This method is will provide the registration method for users.

**Returns:** returnView();

**Method name:** Logout

public ViewResult Logout (){}

**Description:** This method is will destroy the authorization of the registered users.

**Returns:** returnView();

**Method name:** Update

public ViewResult Update(RegisteredUser user){}

**Description:** This method is will edit the profile of the registered users and send it to update in an entity frameworl.

**Returns:** returnView();

**Method name:** Myprofile

public ViewResult Myprofile(){}

**Description:** This method is will direct the users to their my profile page.

**Returns:** returnView();

**Method name:** viewDhammaHistory

public ViewResult viewDhammaHistory(){}

**Description:** This method is will direct the users to their dhamma history page.

**Returns:** returnView();

**2.3 AdminController Class**



**Figure 53 AdminController class**

This class is a controller class that manage all of the view and models that is support the administrator to manage all of the information in the Wat Duang Dee web application system

The following user requirement specification which covered in this class.

**URS-15:** The administrator users can update temple’s history.

**URS-16:** The administrator users can update the temple’s news.

**URS-17:** The administrator users can add the temple’s news.

**URS-18:** The administrator users can delete the temple’s news.

**URS-19:** The administrator users can update the temple’s annual activity.

**URS-20:** The administrator users can add the temple’s annual activity.

**URS-21:** The administrator users can delete the temple’s annual activity.

**URS-22:** The administrator users can update the temple’s other activity.

**URS-23:** The administrator users can add the temple’s other activity.

**URS-24:** The administrator users can delete the temple’s other activity.

**URS-25:** The administrator users can update the temple’s photo gallery.

**URS-26:** The administrator users can add the temple’s photo gallery.

**URS-27:** The administrator users can delete the temple’s photo gallery.

**URS-28:** The administrator users can add a photo in the gallery of the temple’s photo gallery.

**URS-29:** The administrator users can delete a photo in the gallery of the temple’s photo gallery.

**URS-30:** The administrator users can add a video on the temple’s video gallery.

**URS-31:** The administrator users can delete a video on the temple’s video gallery.

**URS-32:** The administrator users can update temple’s dhamma lesson.

**URS-33:** The administrator users can add temple’s dhamma lesson.

**URS-34:** The administrator users can delete temple’s dhamma lesson.

**URS-35:** The administrator users can approve the user’s dhamma question.

**URS-36:** The administrator users can answer the user’s dhamma question.

**URS-37:** The administrator users can edit the answer of the user’s dhamma question.

**URS-38:** The administrator users can delete the user’s dhamma question.

**Method name:** newsCreate

public ViewResult newsCreate(){}

**Description:** This method is will direct the administrator users to create a news of temple.

**Returns:** returnView();

**Method name:** newsUpdate

public ViewResult newsUpdate(int newsId){}

**Description:** This method is will direct the administrator users to edit a news of temple.

**Returns:** returnView(News);

**Method name:** newsUpdate

public ViewResult newsUpdate(News news){}

**Description:** This method is will direct edited a news of temple to save in entity framework.

**Returns:** return RedirectToAction("Main");

**Method name:** newsDelete

public ViewResult newsDelete(int newsId){}

**Description:** This method is will direct a news of temple to delete in entity framework.

**Returns:** return RedirectToAction("Main");

**Method name:** annualActivityCreate

public ViewResult annualActivityCreate(){}

**Description:** This method is will direct the administrator users to create an annual activity.

**Returns:** returnView();

**Method name:** annualActivityUpdate

public ViewResult annualActivityUpdate(int annaulId){}

**Description:** This method is will direct the administrator users to edit an annual activity.

**Returns:** returnView(Activity);

**Method name:** annualActivityUpdate

public ViewResult annualActivityUpdate(Activity an){}

**Description:** This method is will direct edited an annual activity to save in entity framework.

**Returns:** return RedirectToAction("Main");

**Method name:** annualActivityDelete

public ViewResult annualActivityDelete(int id){}

**Description:** This method is will direct an annual activity of temple to delete in entity framework.

**Returns:** return RedirectToAction("Main");

**Method name:** otherActivityCreate

public ViewResult otherActivityCreate(){}

**Description:** This method is will direct the administrator users to create other activity.

**Returns:** returnView();

**Method name:** otherActivityUpdate

public ViewResult otherActivityUpdate(int otherId){}

**Description:** This method is will direct the administrator users to edit other activity.

**Returns:** returnView(Activity);

**Method name:** otherActivityUpdate

public ViewResult otherActivityUpdate(Activity ot){}

**Description:** This method is will direct edited other activity to save in entity framework.

**Returns:** return RedirectToAction("Main");

**Method name:** otherActivityDelete

public ViewResult otherActivityDelete(int id){}

**Description:** This method is will direct other activity of temple to delete in entity framework.

**Returns:** return RedirectToAction("Main");

**Method name:** lessonCreate

public ViewResult lessonCreate(){}

**Description:** This method is will direct the administrator users to create lesson.

**Returns:** returnView();

**Method name:** lessonUpdate

public ViewResult lessonUpdate(int lessonId){}

**Description:** This method is will direct the administrator users to edit lesson.

**Returns:** returnView(Lesson);

**Method name:** lessonUpdate

public ViewResult lessonUpdate(Lesson les){}

**Description:** This method is will direct edited lesson to save in entity framework.

**Returns:** return RedirectToAction("Main");

**Method name:** lessonDelete

public ViewResult lessonDelete(int id){}

**Description:** This method is will direct lesson to delete in entity framework.

**Returns:** return RedirectToAction("Main");

**Method name:** GalleryCreate

public ViewResult GalleryCreate(){}

**Description:** This method is will direct the administrator users to create gallery.

**Returns:** returnView();

**Method name:** GalleryUpdate

public ViewResult GalleryUpdate(int galleryId,int photoid){}

**Description:** This method is will direct the administrator users to update a gallery and photo.

**Returns:** returnView(Gallery);

**Method name:** GalleryUpdate

public ViewResult GalleryUpdate(Gallery gal){}

**Description:** This method is will direct gallery to save in entity framework.

**Returns:** return RedirectToAction("Main");

**Method name:** GalleryDelete

public ViewResult GalleryDelete(int id){}

**Description:** This method is will direct gallery to delete in entity framework.

**Returns:** return RedirectToAction("Main");

**Method name:** VideoUpdate

public ViewResult VideoUpdate(int videoId){}

**Description:** This method is will direct the administrator users to update/upload a video.

**Returns:** returnView(Video);

**Method name:** VideoUpdate

public ViewResult VideoUpdate(Video video){}

**Description:** This method is will direct video to save in entity framework.

**Returns:** return RedirectToAction("Main");

**Method name:** VideoDelete

public ViewResult VideoDelete(int id){}

**Description:** This method is will direct video to delete in entity framework.

**Returns:** return RedirectToAction("Main");

**Method name:** answerUpdate

public ViewResult answerUpdate(int id){}

**Description:** This method is will direct the administrator users to update an answer of dhamma question.

**Returns:** returnView(qa);

**Method name:** answerUpdate

public ViewResult answerUpdate(QuestionAnswerDhamma qa){}

**Description:** This method is will direct updated answer of dhamma question to save in entity framework.

**Returns:** return RedirectToAction("Main");

**Method name:** answerDelete

public ViewResult answerDelete(int id){}

**Description:** This method is will direct questin and answer to delete in entity framework.

**Returns:** return RedirectToAction("Main");

**Method name:** HistoryUpdate

public ViewResult HistoryUpdate(History data){}

**Description:** This method is will direct updated history to save in entity framework.

**Returns:** return RedirectToAction("Main");

**Method name:** CompleteUpdate

public ViewResult CompleteUpdate(){}

**Description:** This method is will direct a completion message to users.

**Returns:** return View();

**Method name:** FailureUpdate

public ViewResult FailureUpdate(){}

**Description:** This method is will direct an error message to users.

**Returns:** return View();

**2.4 IRepository Interface**



**Figure 54 IRepository interface**

This interface is a repository class that manage all of the data with an entity framework that is support the administrator to manage all of the information in the Wat Duang Dee web application system

**Method name:** SaveHistory

public void SaveHistory(History data){}

**Description:** This method is will direct updated history to save in database.

**Returns:** void

**Method name:** newsUpdate

public void newsUpdate(News news){}

**Description:** This method is will direct updated news to save in database.

**Returns:** void

**Method name:** newsDelete

public News newsDelete(int newsId){}

**Description:** This method is will direct news to delete in database.

**Returns:** return (news);

**Method name:** ActivityUpdate

public void ActivityUpdate(Activity ot){}

**Description:** This method is will direct the activity to save in database.

**Returns:** void

**Method name:** ActivityDelete

public Activity ActivityDelete(int id){}

**Description:** This method is will direct the activity to delete in database.

**Returns:** return(ot);

**Method name:** LessonUpdate

public void LessonUpdate(Lesson les){}

**Description:** This method is will direct the dhamma lesson to save in database.

**Returns:** void

**Method name:** LessonDelete

public Activity LessonDelete(int id){}

**Description:** This method is will direct the dhamma lesson to delete in database.

**Returns:** return(les);

**Method name:** GalleryUpdate

public void GalleryUpdate(Gallery gal){}

**Description:** This method is will direct the gallery to save in database.

**Returns:** void

**Method name:** GalleryDelete

public Gallery GalleryDelete(int id){}

**Description:** This method is will direct the gallery to delete in database.

**Returns:** return(gal);

**Method name:** VideoUpdate

public void VideoUpdate(Video video){}

**Description:** This method is will direct the video to save in database.

**Returns:** void

**Method name:** VideoDelete

public Video VideoDelete(int id){}

**Description:** This method is will direct the video to delete in database.

**Returns:** return(gal);

**Method name:** answerUpdate

public void answerUpdate(QuestionAnswerDhamma qa){}

**Description:** This method is will direct the question and answer dhamma to save in database.

**Returns:** void

**Method name:** answerDelete

public QuestionAnswerDhamma answerDelete(int id){}

**Description:** This method is will direct the question and answer dhamma to delete in database.

**Returns:** return(qa);

**Method name:** updateUser

public void updateUser(RegisteredUser user){}

**Description:** This method is will direct the user profile to save in database.

**Returns:** void

**2.4 EmailProcess Class**



**Figure 55 EmailProcess class**

This class is a class for handle the smtp email and send to notify all of users that involve with.

**Method name:** notifyUser

public void notifyUser(){}

**Description:** This method is will direct the message to all of registered users by email.

**Returns:** void

**Method name:** notifyAdmin

public void notifyUser(){}

**Description:** This method is will direct the message to all of administrator users by email.

**Returns:** void

**Method name:** notifyUser

public void notifyUser(String message){}

**Description:** This method is will direct the specific message to all of administrator users by email.

**Returns:** void

**3. Sequence Diagram**

**3.1 Sequence Diagram (SC-01) response to UC-01 and URS-03**



**Figure 56 Sequence Diagram (SC-01)**

**3.2 Sequence Diagram (SC-02) response to UC-02 and URS-05**

 **Figure 57 Sequence Diagram (SC-02)**

**3.3 Sequence Diagram (SC-03) response to UC-03 and URS-44**

 **Figure 58 Sequence Diagram (SC-03)**

**3.4 Sequence Diagram (SC-04) response to UC-04 and URS-04**

**Figure 59 Sequence Diagram (SC-04)**

**3.5 Sequence Diagram (SC-05) response to UC-05 and URS-06**

 **Figure 60 Sequence Diagram (SC-05)**

**3.6 Sequence Diagram (SC-06) response to UC-06 and URS-09**

 **Figure 61 Sequence Diagram (SC-06)**

**3.7 Sequence Diagram (SC-07) response to UC-07 and URS-10**

**Figure 62 Sequence Diagram (SC-07)**

**3.8 Sequence Diagram (SC-08) response to UC-08 and URS-11**



**Figure 63 Sequence Diagram (SC-08)**

**3.9 Sequence Diagram (SC-09) response to UC-09 and URS-12**

**Figure 64 Sequence Diagram (SC-09)**

**3.10 Sequence Diagram (SC-10) response to UC-10 and URS-13**

 **Figure 65 Sequence Diagram (SC-10)**

**3.11 Sequence Diagram (SC-11) response to UC-11 and URS-14**



**Figure 66 Sequence Diagram (SC-11)**

**3.12 Sequence Diagram (SC-12) response to UC-12 and URS-41**



**Figure 67 Sequence Diagram (SC-12)**

**3.13 Sequence Diagram (SC-13) response to UC-13 and URS-42**

 **Figure 68 Sequence Diagram (SC-13)**

**3.14 Sequence Diagram (SC-14) response to UC-14 and URS-01**

 **Figure 69 Sequence Diagram (SC-14)**

**3.15 Sequence Diagram (SC-15) response to UC-15 and URS-02**

 **Figure 70 Sequence Diagram (SC-15)**

**3.16 Sequence Diagram (SC-16) response to UC-16 and URS-07**

 **Figure 71 Sequence Diagram (SC-16)**

**3.17 Sequence Diagram (SC-17) response to UC-17 and URS-08**

 **Figure 72 Sequence Diagram (SC-17)**

**3.18 Sequence Diagram (SC-18) response to UC-18 and URS-39**

 **Figure 73 Sequence Diagram (SC-18)**

**3.19 Sequence Diagram (SC-19) response to UC-19 and URS-40**