CENG315 Spring 2018-2019
Software Design Description Report

Ceng Manager
(CEM)

By
Ertan Uysal-250201064
Umut Utku Tahan-250201086
Mücahit Turhan-250201065
Kıvanç Ersoy-230201013
Ahmet Öcal-240201114

Table of Contents

Table of Contents **2**

Table of Figures **3**

1 Identification of SDD **4**

1.1 Date of issue and Status 4

1.2 Issuing Organization 4

1.3 Authorship 4

1.4 Change History 4

2 Introduction **4**

2.1 Purpose 4

2.2 Scope 4

2.3 Context 5

2.4 Summary 5

3 References **5**

4 Glossary **5**

5 Body **6**

5.1 Stakeholders and Design Concerns 6

5.2 Design Views and Corresponding Viewpoints 6

5.2.1 Logical Viewpoint 6

5.2.1.1 Design Concern 6

5.2.1.2 UML Class Diagram Description 6

5.2.2 Information Viewpoint 8

5.2.2.1 Design Concern 8

5.2.3 Interface Viewpoint 9

5.2.3.1 Design Concern 9

5.2.3.2 Classes and Method Definitions 9

5.2.4 Interaction Viewpoint 17

5.2.4.1 Design Concern 17

5.2.4.2 Sequence Diagram 17

5.3 Design Rationale 32

Table of Figures

Figure 1 - UML Class Diagram **7**
Figure 2 – ER Diagram **8**
Figure 3 – Login Sequence Diagram **18**
Figure 4 – Add Course Sequence Diagram **19**
Figure 5 – Edit Course Sequence Diagram **20**
Figure 6 – Delete Course Sequence Diagram **21**
Figure 7 – Open Course Sequence Diagram **22**
Figure 8 – Set Weekly Schedule Sequence Diagram **23**
Figure 9 – Import File Sequence Diagram **24**
Figure 10 – Publish Changes Sequence Diagram **25**
Figure 11 – Add Email Information Sequence Diagram **26**
Figure 12 – Edit Email Sequence Diagram **27**
Figure 13 – Delete Email Sequence Diagram **28**
Figure 14 – Send Event as Email Sequence Diagram **29**
Figure 15 – Add Content Manager Sequence Diagram **30**
Figure 16 – Logout Sequence Diagram **31**

1 Identification of SDD

1.1 Date of Issue and Status

Date of Issue: 11.06.2019 23:55
Status: v1.0

1.2 Issuing Organization

The issuing organization of the project is Computer Engineering Department of Izmir Institute of Technology.

1.3 Authorship

The Software Design Description document is written by group members of ThroneTech.

1.4 Change History

First steps of this project started in 13/03/2019 with the preparation of SRS (Software Requirements Specification). Then it was completed in 02/04/2019. After preparation of the SRS, SDD (Software Design Description) preparation process began in 17/04/2019. Then it was completed in 14/05/2019.

2 Introduction

2.1 Purpose

The main purpose of this document is to clarify and visualize the software design and architecture of the project using different viewpoints. It also describes how the use cases will be implement using this design. In a nutshell, this document explains the internal structure of the project to readers.

2.2 Scope

“Ceng manager” is a desktop application which provide control of the computer engineering department of Izmir Institute of Technology. Application can be used by two

different users that are “admin” and “content manager”. Thanks to this application, control of the website is made easy. The changes are reflected on website after the changes made by content manager have been approved by admin.

2.3 Context

The document contains the software design description of the project. Different environments are existed in the system of the project. Therefore, the context of the system defines the relations, dependencies and interactions between system and environments. To explain more understandable, relations and dependencies is explained in the UML class diagram and the interactions is stated in the UML sequence diagram. Also, entities and their relations take parts in ER diagram. The details of the project to be developed is explicated with using diagrams and viewpoints and design information.

2.4 Summary

This documentation is planned for any individual who needs to see how Ceng Manager was structured. All system limitations, functionalities and components incorporated into the system are clarified in detail. By utilizing this documentation, engineers can work together so as to shape the task to the last determination or by utilizing this record, they can broaden the limits of the system for further interest.

3 References

IEEE Std 1016™-2009 (Revision of IEEE Std 1016-1998) IEEE Standard for Information Technology—Systems Design— Software Design Descriptions

4 Glossary

Term	Definition
Users	Content manager and admin
Ceng	Computer Engineering department of Izmir Institute of Technology
Database	Collection of all the information monitored by the system.
Cm	“Content Manager” is the person who is responsible for updating the content of the Ceng.
Cem	Ceng Manager is the name of the application
Admin	The someone who is chosen by management of Ceng is the most authoritative person.

User	Admin and the content managers of this application.
Category	It is a kind of group selection such as freshman, junior, sophomore, senior, master degree and project groups etc.
IYTE CENG WEBSITE	http://ceng.iyte.edu.tr/tr/
ORM	ORM is a technology that handles how objects in object oriented languages correspond to records in relational databases.

5 Body

5.1 Stakeholders and Design Concerns

The stakeholders of the Ceng Manager project are ThroneTech,its advisor Professor Onur Demirors and its assistant Ersin Çine. The main concerns of stakeholders are that the project can satisfy the required needs and can be completed within the specified time which is 12.06.2019.Presentation of the Project will be made in this date.

- The project should be feature friendly that new properties can be adapted
- Software should be safe and secure.
- Database should be simple and handle the project's needs.

5.2 Design Views and Corresponding Viewpoints

5.2.1 Logical Viewpoint:

The logical viewpoint's corresponding design view is to describe the logical structure of the application and define class designs.

5.2.1.1 Design Concern:

Correctly defining the classes and their relationships. For each class we define attributes, visibility of attributes, types of attributes, public methods, input types and returning types of methods. Also we define relationships between classes.

5.2.1.2 UML Class Diagram Description

There are 4 different packages in this project. This approach was made so that different classes with different tasks did not interfere. In the application package, there is only one class that

initiates the CEM. In the pages package, there is one class for every screen due to use of javaFX. And abstract class which name is Page. In the operation package, there are classes where file, email and web operations are performed. In the dataAccess package, due to use of hibertane, every entity in ER diagram have classes. And also we have class where database operations are performed. All of the methods' getters and setters are not shown in the UML Class Diagram.

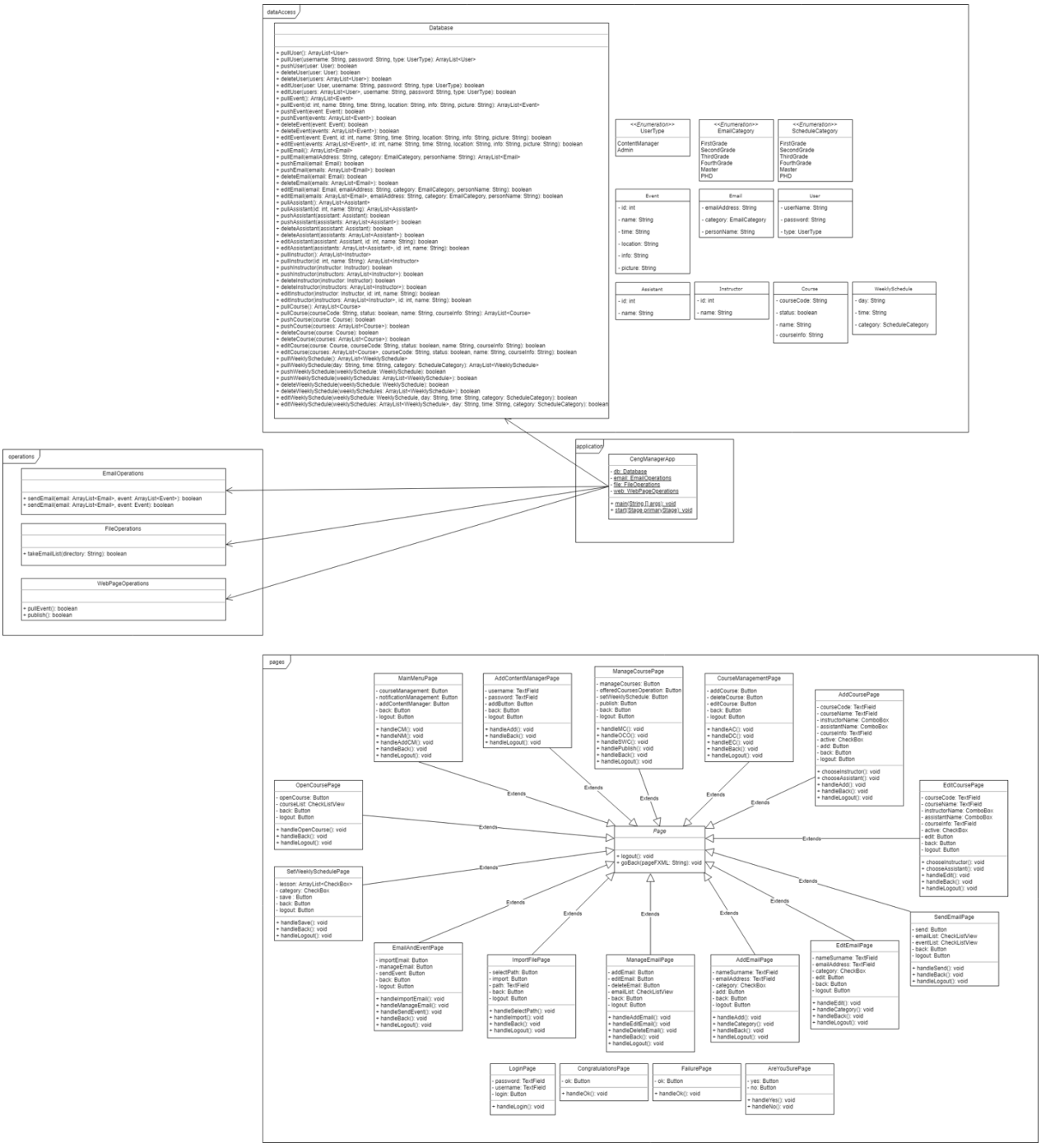

Figure 1- UML Class Diagram

5.2.2 Information Viewpoint:

This viewpoint's corresponding design view is to define how the persistent data stored, designing the persistent data storage.
In here we give entity-relation diagram to explain these concepts.ER diagram is used to described information viewpoint. MySQL database management system will be used for Cem. In order to connect MySQL. "Hibernate" will use ORM tools.

5.2.2.1 Design Concern:

Explanation of the structure of persistent data storage, definition of data content and data access schemes.

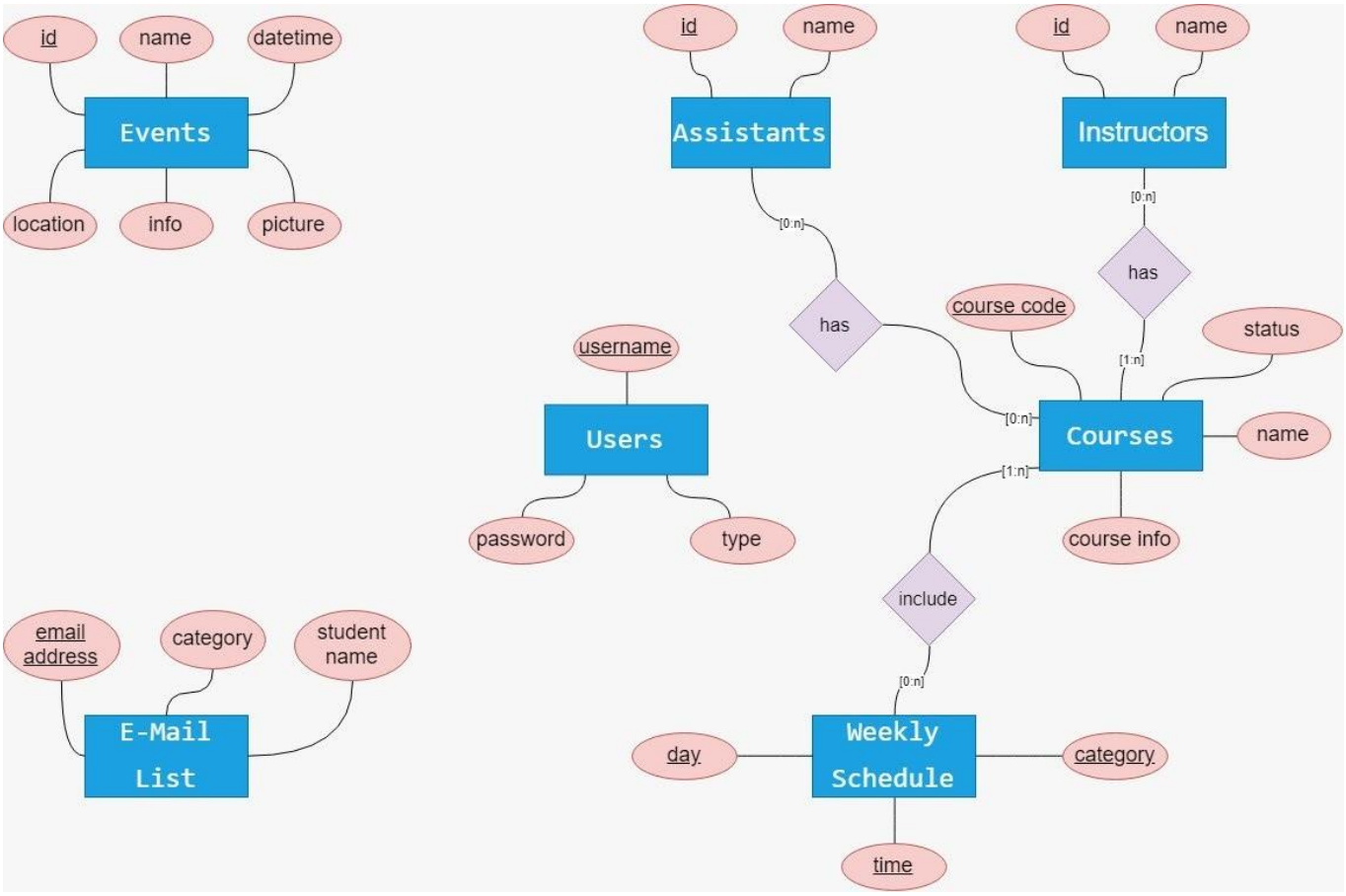

Figure 2- ER Diagram

5.2.3 Interface Viewpoint

This viewpoint's corresponding design view is to explain each methods used in UML, so that it provides interface to ones that want to use those methods. Also reveals how our system will work.
Note that types of inputs and outputs of the methods, visibility of attributes, relations between entities are specified in UML diagram. Here we provide explanations of the methods.

5.2.3.1 Design Concerns:

Designers, programmers, and testers often use design entities that they did not develop. The interface description establishes an agreement among designers, programmers, and testers about how cooperating entities will interact. Each entity interface description should contain everything another designer or programmer needs to know to develop software that interacts with that entity. A clear description of entity interfaces is essential on a multi-person development for smooth integration and ease of maintenance.

5.2.3.2 Classes and Method Definitions

Database:

pullUser:
Returns all existing users in the database.

pullUser:
If you want to get specific users according to username, password, type of the user you can do this with this method. If you just want to get users according to one parameter, then for rest of the parameters enter null.

pushUser:
Push the given user to the database and user gets saved to the database.

deleteUser:
Deletes the user or users according to the specified parameter.

editUser:
According to the specified user or list of users, edits the user according to given properties of the user in the parameters.

pullEvent:

If called without any parameters returns all events from database, for each specified parameter restrics number of returning events according to the properties of event specified in parameters.

pushEvent:

Saves an event or list of events to the database according to the given parameter.

deleteEvent:

Deletes the event or events according to the specified parameter.

editEvent:

According to the specified event or list of events, edits the event according to given properties of the event in the parameters.

pullEmail:

If no parameters are specified return all emails, for each specified parameter restrics number of returning emails according to the properties of email specified in parameters.

pushEmail:

Saves an email or list of emails to the database according to the given parameter.

deleteEmail:

Deletes the email or emails according to the specified parameter.

editEmail:

According to the specified email or list of emails, edits the email according to given properties of the email in the parameters.

pullAssistant:

If no parameters are specified return all assistants, for each specified parameter restrics number of returning assistants according to the properties of assistant specified in parameters.

pushAssistant:

Saves an assistant or list of assistants to the database according to the given parameter.

deleteAssistant:

Deletes the assistant or assistants according to the specified parameter.

editAssistant:

According to the specified assistant or list of assistants, edits the assistant according to given properties of the assistant in the parameters.

pullInstructor:

If no parameters are specified return all instructors, for each specified parameter restrics number of returning instructors according to the properties of instructor specified in parameters.

pushInstructor:

Saves an instructor or list of instructors to the database according to the given parameter.

deleteInstructor:

Deletes the instructor or instructors according to the specified parameter.

editInstructor:

According to the specified instructor or list of instructors, edits the instructor according to given properties of the instructor in the parameters.

pullCourse:

If no parameters are specified return all courses, for each specified parameter restrics number of returning courses according to the properties of course specified in parameters.

pushCourse:

Saves a course or list of courses to the database according to the given parameter.

deleteCourse:

Deletes the course or courses according to the specified parameter.

editCourse:

According to the specified course or list of courses, edits the course according to given properties of the course in the parameters.

pullWeeklySchedule:

If no parameters are specified return all weekly schedules, for each specified parameter restrics number of returning weekly schedules according to the properties of weekly schedule specified in parameters.

pushWeeklySchedule:

Saves a weekly schedule or list of weekly schedules to the database according to the given parameter.

deleteWeeklySchedule:

Deletes the weekly schedule or weekly shedules according to the specified parameter.

editWeeklySchedule:

According to the specified weekly schedule or list of weekly schedules, edits the weekly schedule according to given properties of the weekly schedule in the parameters.

EmailOperations:

sendEmail(ArrayList<Email>,Event):
Send the event to the list of emails.

sendEmail(ArrayList<Email>,ArrayList<Event>):
Send the list of events to all of the emails in the email list.

FileOperations:

takeEmailList(directory):
According to given directory, take specified file in the given directory and save emails in the file to the database.

WebPageOperations:

pullEvent:
From the ceng web page pulls all events and save them to the database.

publish:
Publishes all changes made in the application to the ceng web page.

LoginPage:

handleLogin:
This method handles the event user clicking the Login button.

MainMenuPage:

handleCM:

This method handles the event user clicking the CourseManagement button.

handleNM:

This method handles the event user clicking the NotificationManagement button.

handleAddCM:

This method handles the event user clicking the AddContentManager button.

handleBack:

This method handles the event user clicking the back button.

handleLogout:

This method is used for exit program or switching user

AddContentManagerPage:

handleAdd:

This method handles the event user clicking the Add button.

handleBack:

This method handles the event user clicking the back button.

handleLogout:

This method is used for exit program or switching user

ManageCoursePage:

handleMC:

This method handles the event user clicking the ManageCourses button.

handleOC:

This method handles the event user clicking the OfferedCoursesOperation button.

handleSWC:

This method handles the event user clicking the SetWeeklySchedule button.

handlePublish:

This method handles the event user clicking the Publish button.

handleBack:

This method handles the event user clicking the Back button.

handleLogout:

This method is used for exit program or switching user

CourseManagementPage:

handleAC:

This method handles the event user clicking the AddCourse button.

handleDC:

This method handles the event user clicking the DeleteCourse button.

handleEC:

This method handles the event user clicking the EditCourse button.

handleBack:

This method handles the event user clicking the Back button.

handleLogout:

This method is used for exit program or switching user

AddCoursePage:

chooseInstructor:

This method handles the event user clicking the InstuctorName combo box button.

chooseAssistant:

This method handles the event user clicking the AssistantName combo box button.

handleAdd:

This method handles the event user clicking the Add button.

handleBack:

This method handles the event user clicking the Back button.

handleLogout:

This method is used for exit program or switching user

AreYouSurePage:

handleYes:

This method handles the event user clicking the Yes button.

handleNo:

This method handles the event user clicking the No button.

EditCoursePage:

chooseInstructor:

This method handles the event user clicking the InstructorName combo box button.

chooseAssistant:

This method handles the event user clicking the AssistamtName combo box button.

handleEdit:

This method handles the event user clicking the Edit button.

handleBack:

This method handles the event user clicking the Back button.

handleLogout:

This method is used for exit program or switching user

OpenCoursePage:

handleOpenCourse:

This method handles the event user clicking the OpenCourse button.

handleBack:

This method handles the event user clicking the Back button.

handleLogout:

This method is used for exit program or switching user

SetWeeklySchedulePage:

handleSave:

This method handles the event user clicking the Save button.

handleBack:

This method handles the event user clicking the Back button.

handleLogout:

This method is used for exit program or switching user

EmailAndEventPage:

handleImportEmail:

This method handles the event user clicking the ImportEmail button.

handleManageEmail:

This method handles the event user clicking the ManageEmail button.

handleSendEvent:

This method handles the event user clicking the SendEvent button.

handleBack:

This method handles the event user clicking the Back button.

handleLogout:

This method is used for exit program or switching user

ImportFilePage:

handleSelectPath:

This method handles the event user clicking the SelectPath button.

handleImport:

This method handles the event user clicking the Import button.

handleBack:

This method handles the event user clicking the Back button.

handleLogout:

This method is used for exit program or switching user

ManageEmailPage:

handleAddEmail:

This method handles the event user clicking the AddEmail button.

handleEditEmail:

This method handles the event user clicking the EditEmail button.

handleDeleteEmail:

This method handles the event user clicking the DeleteEmail button.

handleBack:

This method handles the event user clicking the Back button.

handleLogout:

This method is used for exit program or switching user

AddEmailPage:

handleAdd:

This method handles the event user clicking the Add button.

handleCategory:

This method handles the event user clicking the Category check box button.

handleBack:

This method handles the event user clicking the Back button.

handleLogout:

This method is used for exit program or switching user

EditEmailPage:

handleEdit:

This method handles the event user clicking the Edit button.

handleCategory:

This method handles the event user clicking the Category check box button.

handleBack:

This method handles the event user clicking the Back button.

handleLogout:

This method is used for exit program or switching user

SendEmailPage:

handleSend:

This method handles the event user clicking the Send button.

handleBack:

This method handles the event user clicking the Back button.

handleLogout:

This method is used for exit program or switching user

CongratulationsPage:

handleOk:

This method handles the event user clicking the Ok button.

FailurePage:

handleOk:

This method handles the event user clicking the Ok button.

CengManagerApp:

main:

This method calls the launch method and launch calls the start method.

start:

This method starts the application and opens the login page.

Page:

logout:

This method defines logout action for all pages.

goBack:

This method defines going back action for all pages.

5.2.4 Interaction ViewPoint

This viewpoint's corresponding design view is to explain, interaction of the users with the system and the events that occur in this interaction,the functions and classes are indicated.

Sequence diagram is used to describe interaction viewpoint.The whole process is shown from the beginning to the user so that the user can more fully understand the structure of the viewpoint and see which classes and functions he/she uses.

5.2.4.1 Design Concern

This shows relationships between CEM application classes and it helps the developers to determine the route. Sequence diagram is used to show interaction between classes.

5.2.4.2 Sequence Diagram

Login Interaction

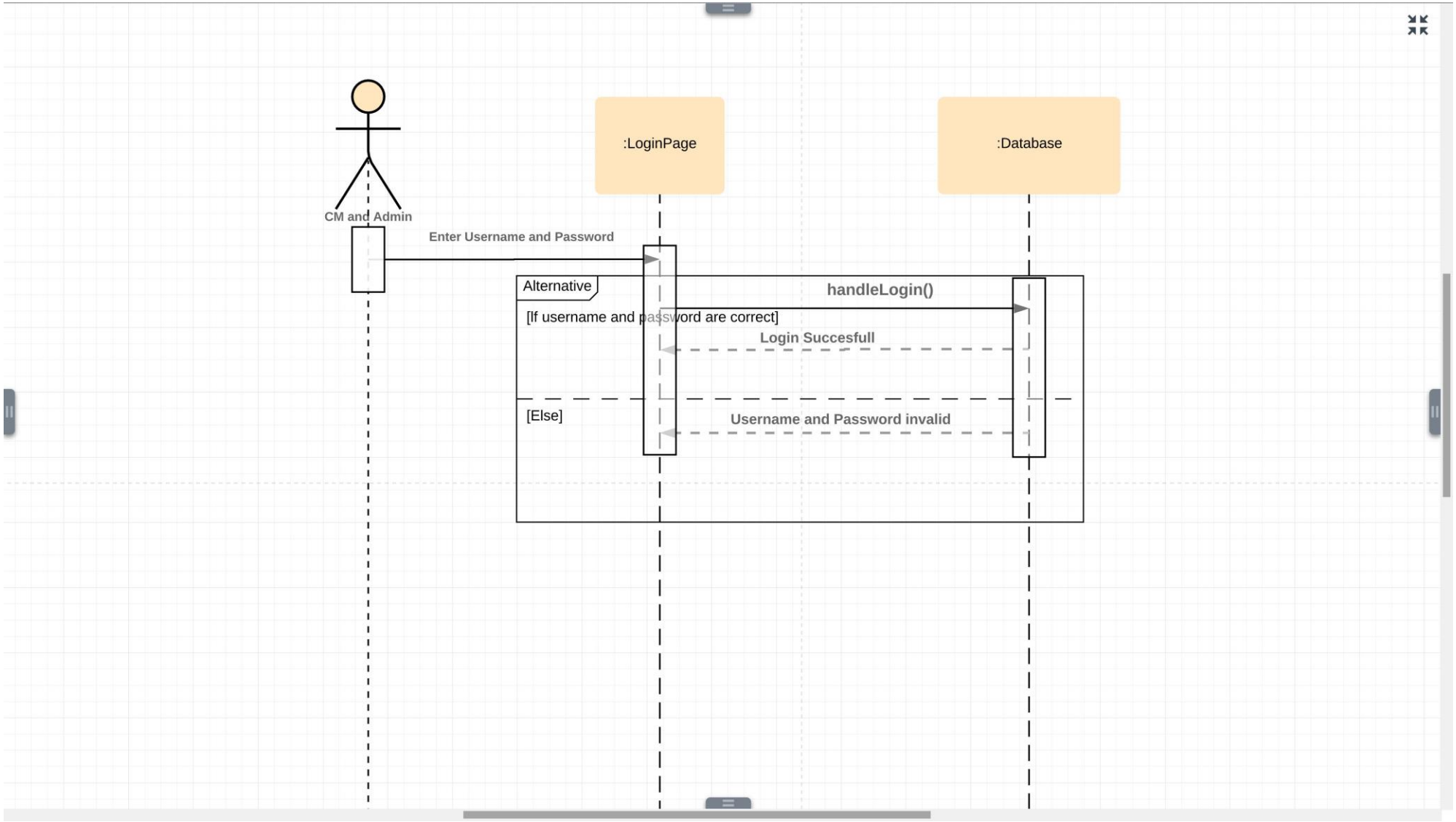

Figure 3-Login Sequence Diagram

Add Course Interaction

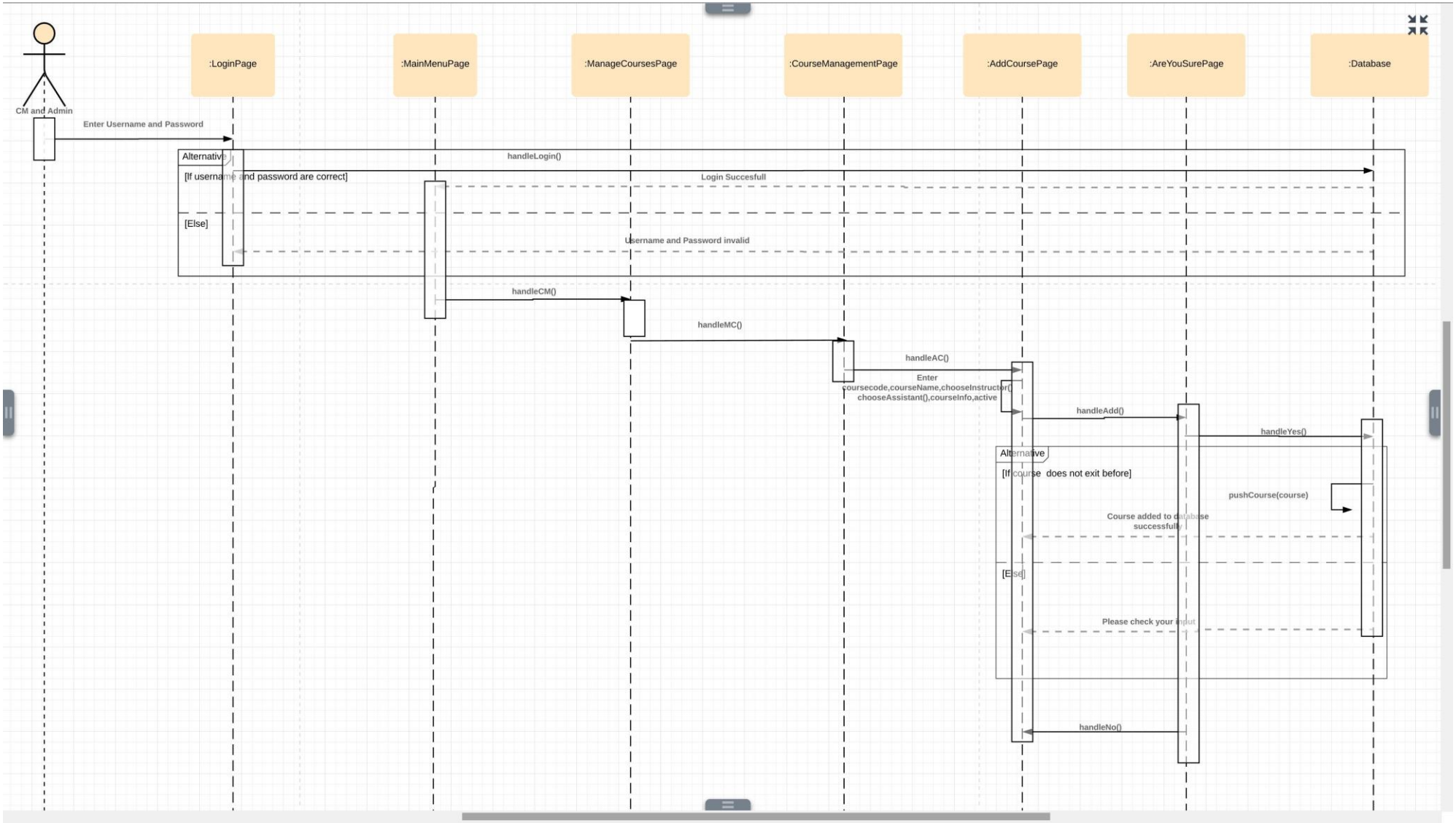

Figure 4-Add Course Sequence Diagram

Edit Course Interaction

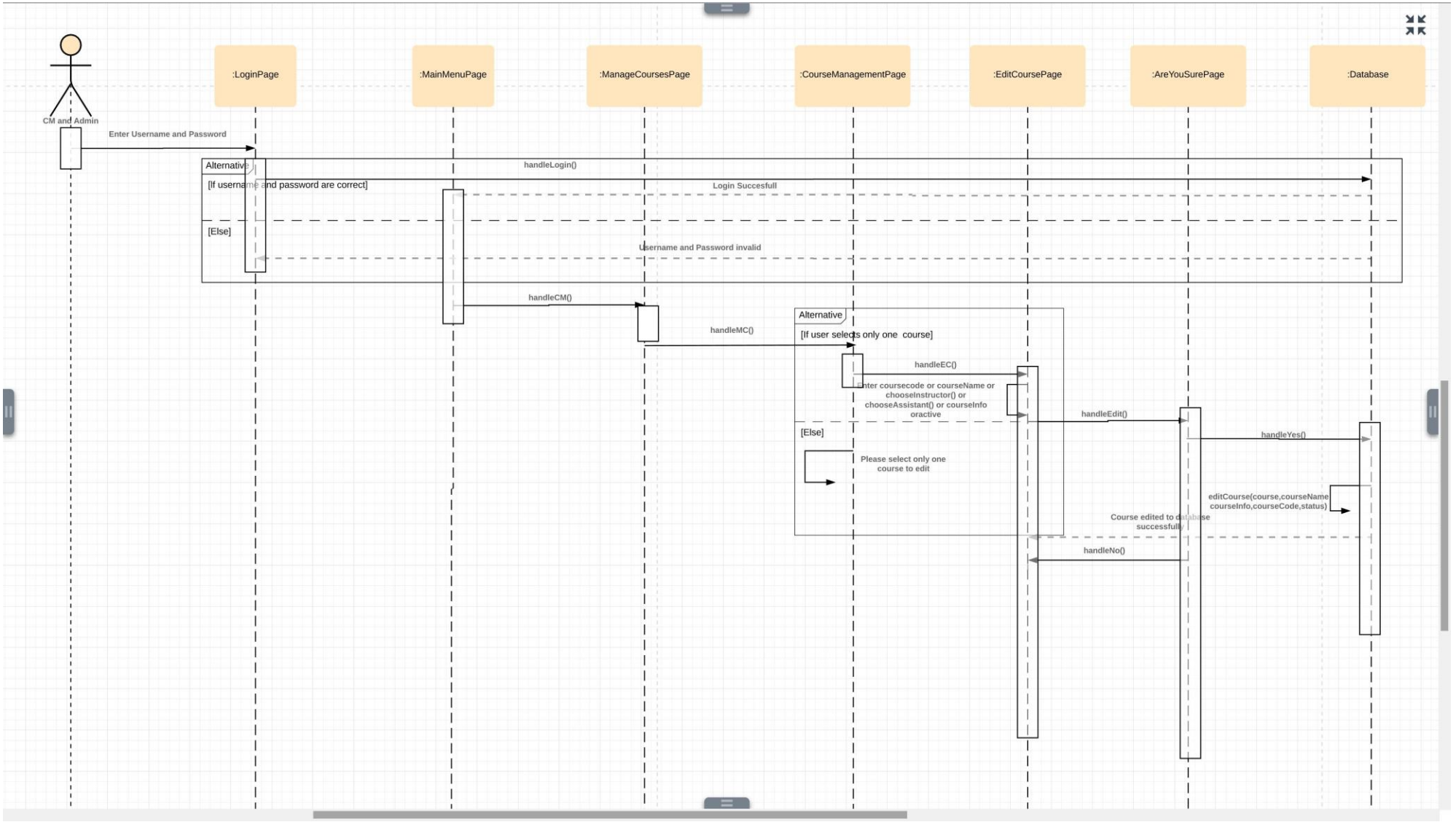

Figure 5-Edit Course Sequence Diagram

Delete Course Interaction

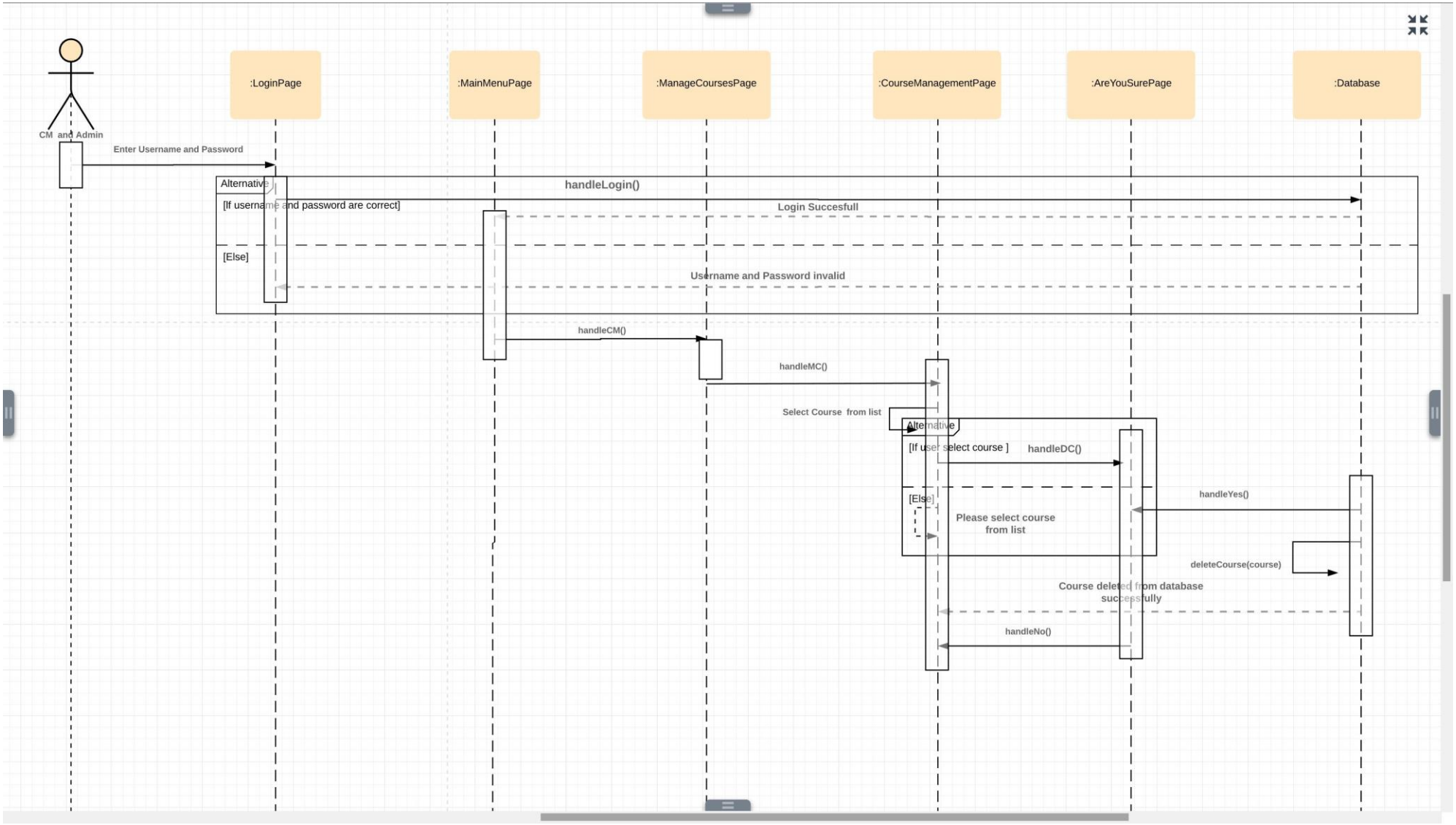

Figure 6-Delete Course Sequence Diagram

Open Course Interaction

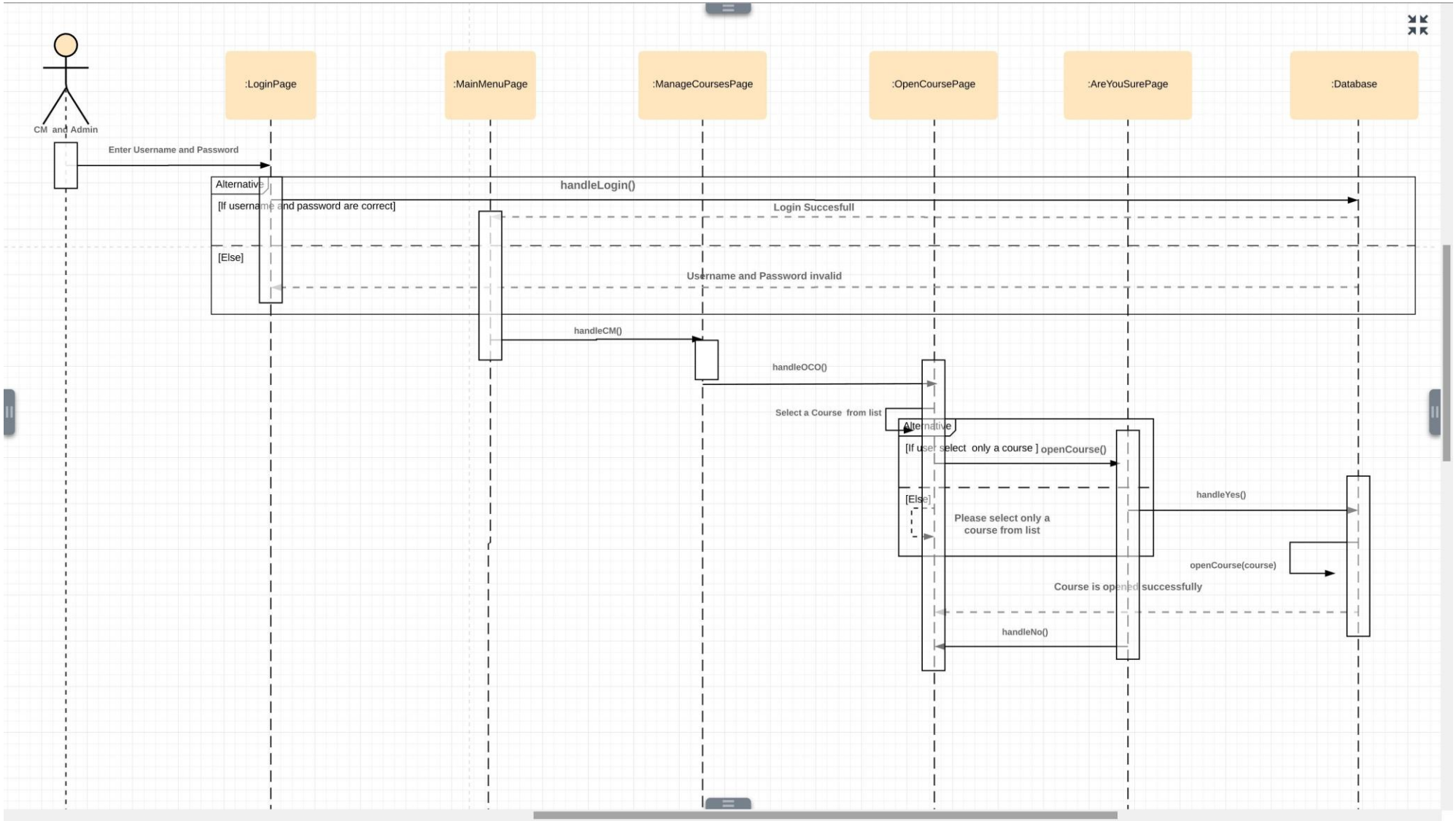

Figure 7-Open Course Sequence Diagram

Set Weekly Schedule Interaction

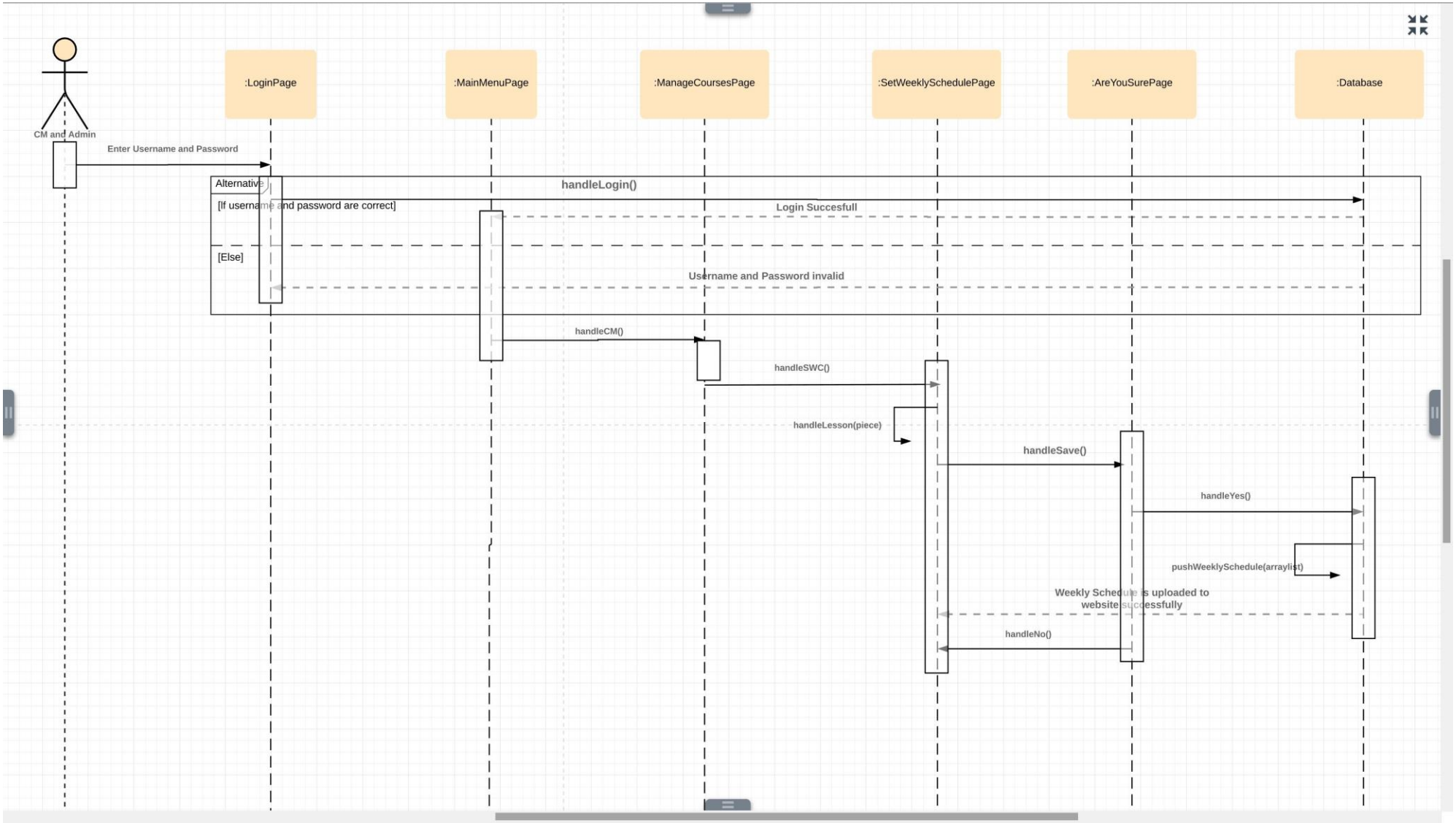

Figure 8- Set Weekly Schedule Sequence Diagram

Import File Interaction

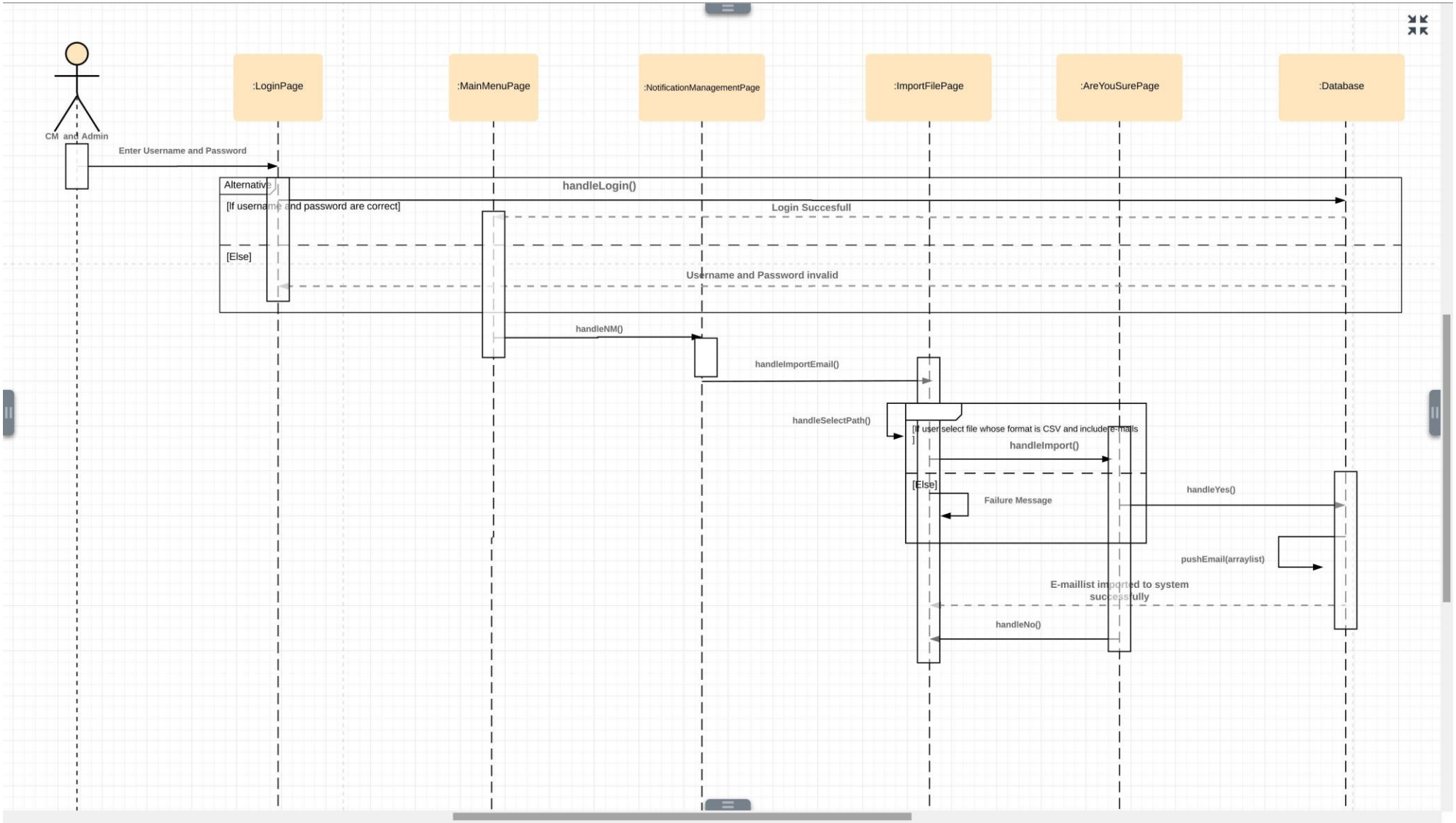

Figure 9-Import File Sequence Diagram

Publish Changes Interaction

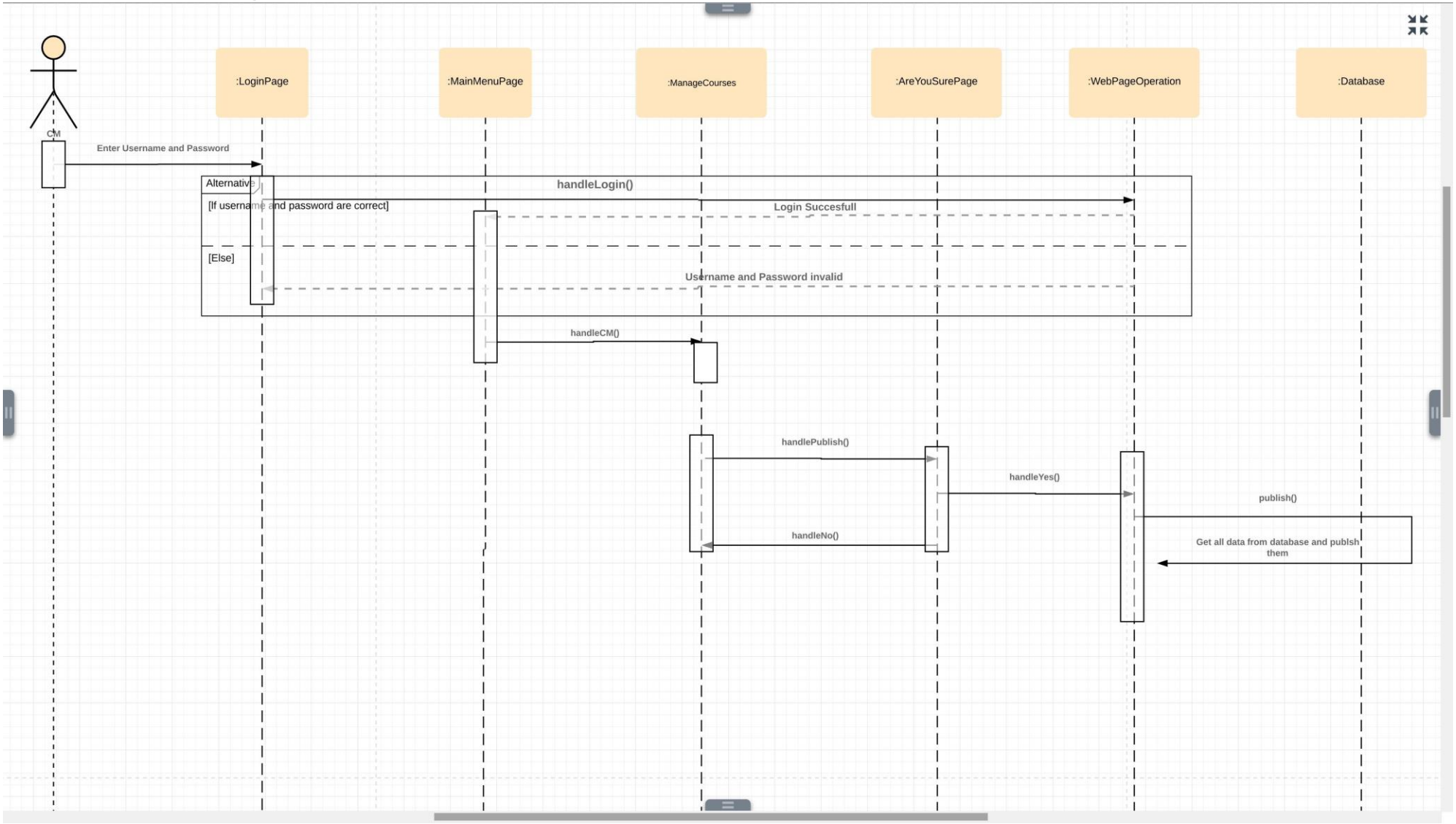

Figure 10-Publish Changes Sequence Diagram

Add Email Information Interaction

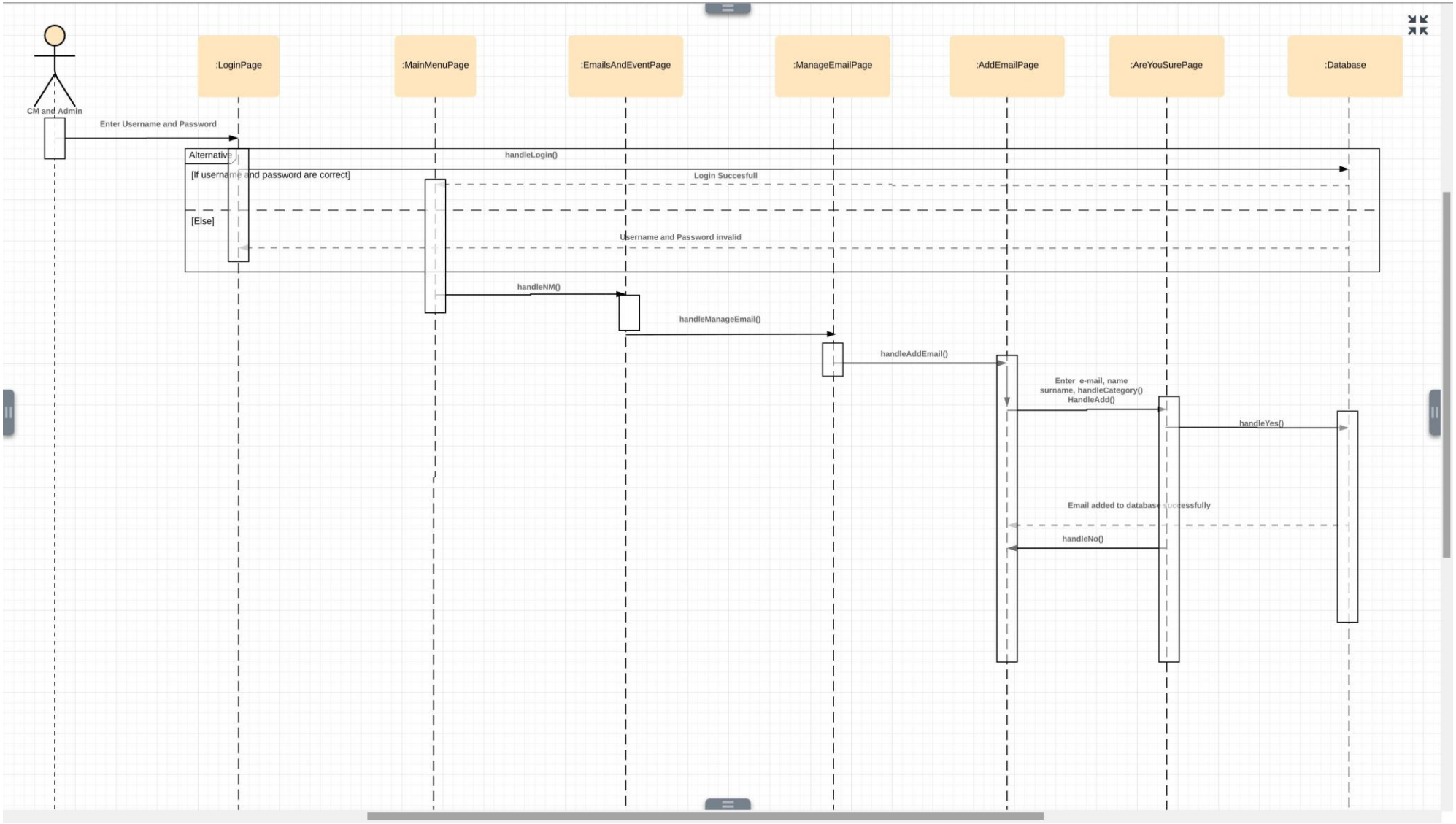

Figure 11-Add Email Information Sequence Diagram

Edit Email Information Interaction

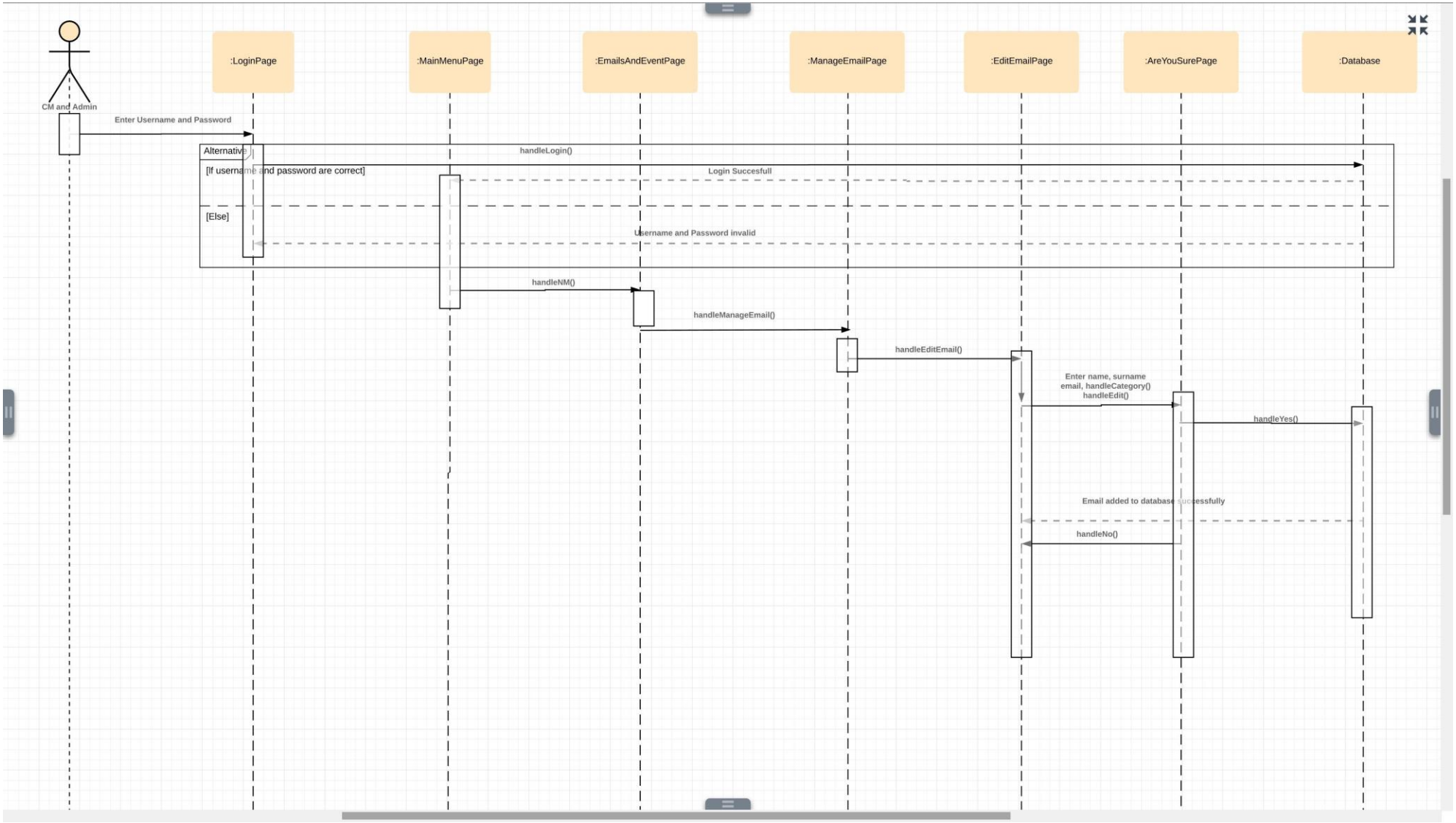

Figure 12-Edit Email Information Sequence Diagram

Delete Email Information Interaction

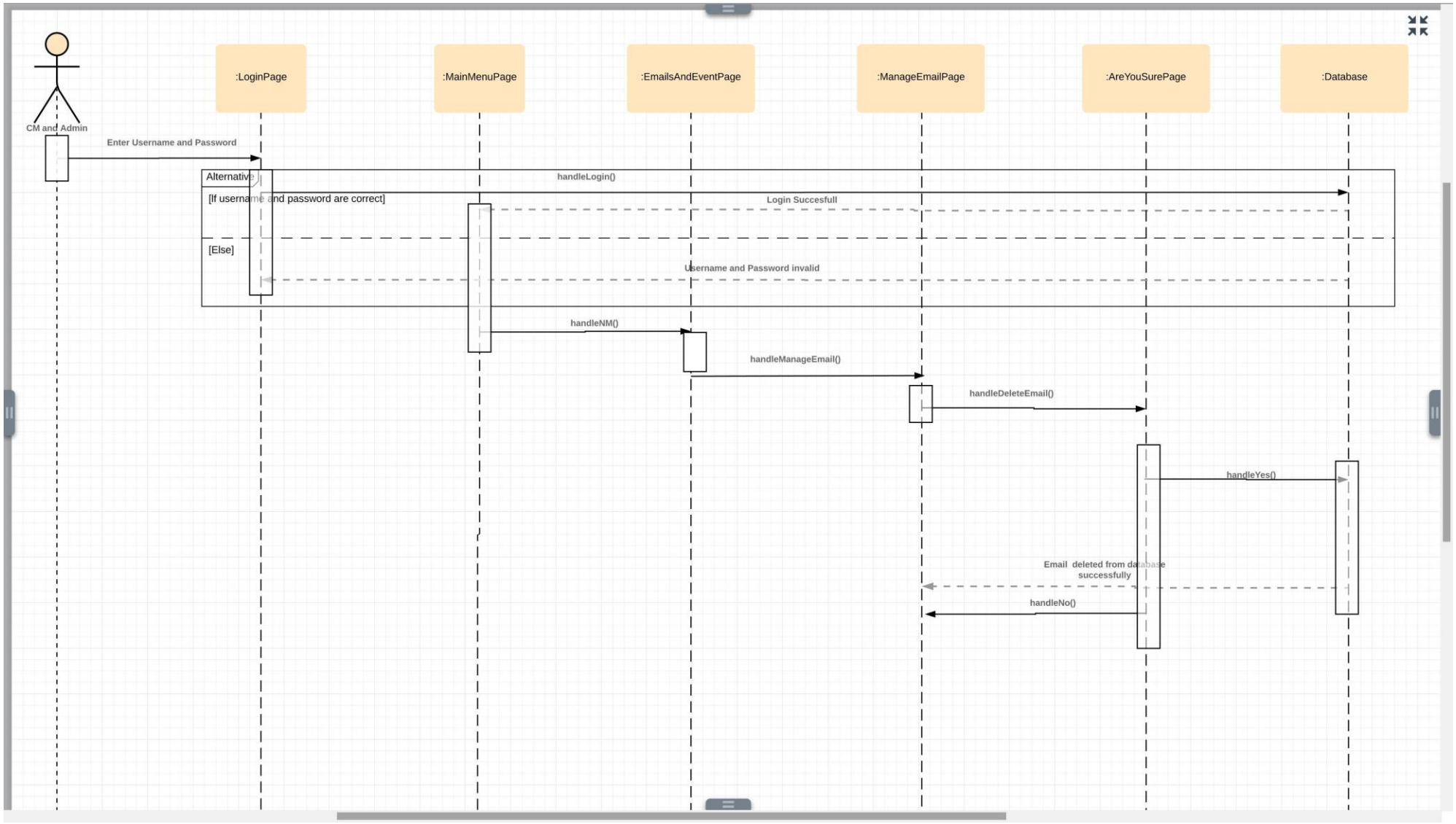

Figure 13-Delete Email Information Sequence Diagram

Send Event As Email Interaction

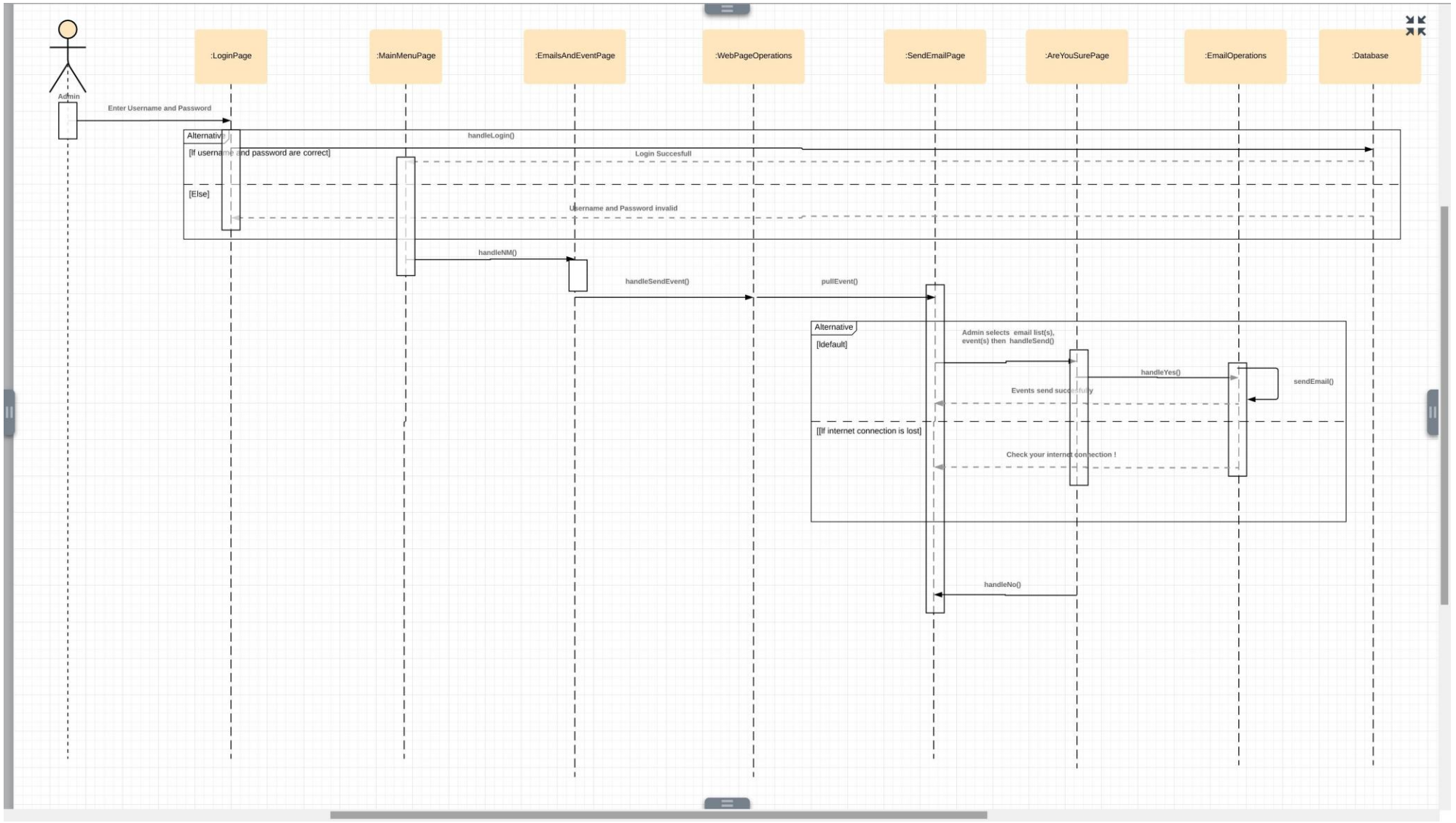

Figure 14-Send Event As Email Sequence Diagram

Add Content Manager Interaction

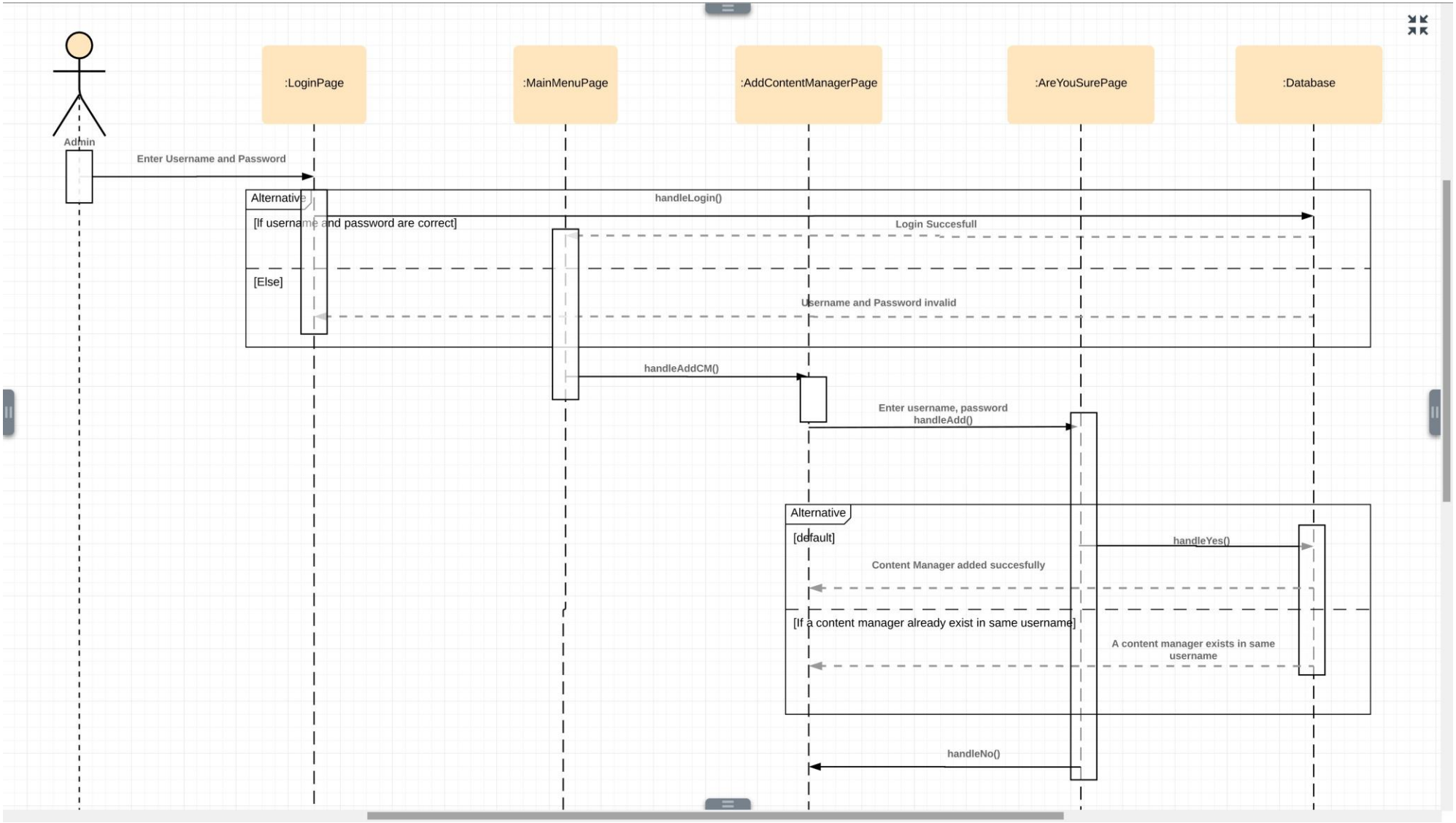

Figure 15-Add Content Manager Sequence Diagram

Logout Interaction

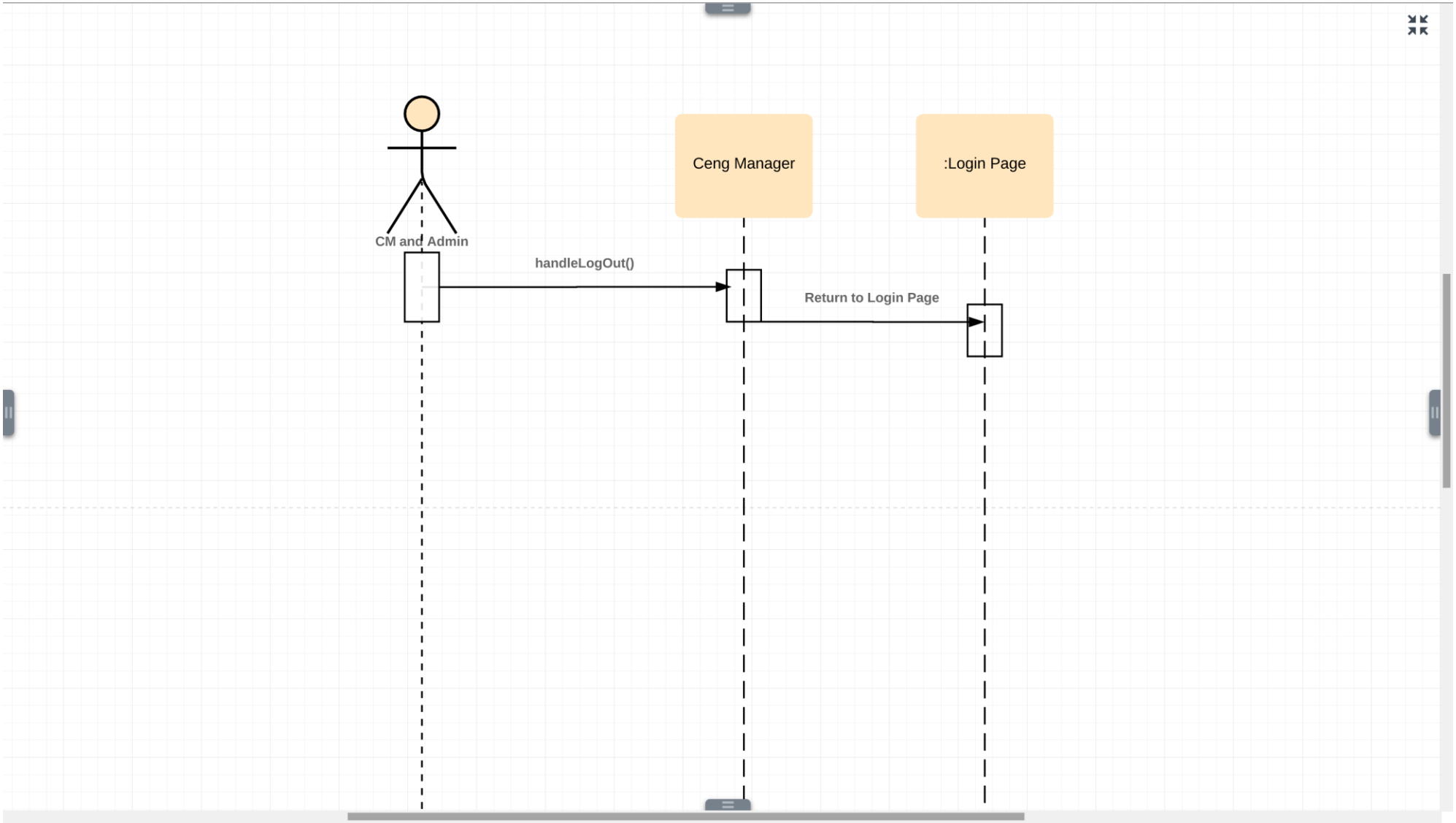

Figure 16-Logout Sequence Diagram

5.3 Design Rationale

The designs are determined according to the software requirement specification report. It can be renewed according to the wishes of the stakeholders and users. Design maked with object-oriented approach in order to maintain the continuity of the project.
ER diagram is used to design the database of the system. UML class diagram shows the interactions between classes and sequence diagram shows object interactions arranged in time sequence.