



Bilkent University

Department of Computer Engineering

CS319 Term Project

Fall 2022

Section 1

Group 1B

Analysis Report

Group members:

Ayşe Kelleci 21902532

Yusuf Doğan 21702970

Zeynep Hanife Akgül 22003356

Kardelen Ceren 22003017

Melisa Tanrıku 21703437

Instructor: Eray Tüzün

Contents

1. Introduction	3
2. Proposed System	3
2.1. Functional Requirements	3
2.2. Non-functional Requirements	7
2.3. System Models	7
2.3.1. Use-Case Model	7
2.3.2. Object and Class Models	18
2.3.3. Dynamic Models	20
2.3.3.1. Activity Diagrams	20
2.3.3.2. State Diagrams	22
2.3.4. User Interface	23
2.3.4.1. Common Interfaces	23
2.3.4.2. Student Interfaces	28
2.3.4.3. Coordinator Interfaces	31
3. References	35

1 Introduction

Currently, Bilkent students leaving for a foreign university through the Erasmus program need to email their coordinators frequently for the course selection and approval process. Instead, we propose a new website in order to decrease the workload of coordinators, make it easier for both students and coordinators to track their files, and increase communication and information sharing without emails. Through this website, the placement of students to their preferred universities is done automatically, the students may choose their courses, the coordinators may approve or reject them, the relevant documents may be generated and signed, and students and coordinators may message each other or create posts in the forum that all users can see and reply to. Overall, the goal is to create a user-friendly website where outgoing students and coordinators can access and handle all their Erasmus-related tasks.

2 Proposed System

2.1 Functional Requirements

User Types

There will be three user types on the site. “Student” users refer to students who applied for the Erasmus program. “Coordinator” users refer to Erasmus department coordinators in Bilkent who are responsible for evaluating the courses the students choose during their mobility period (i.e. the semester(s) they go to the foreign university). “Board member” users are the faculty board members who approve the course transfers along with the coordinator. Board members are very restricted in what they can do or see on the site in an effort to decrease the process’ complexity.

Login

All students applying for Erasmus have accounts that are created by the system administrator. Students will be given an initial password paired with their Bilkent emails, which they will be advised to change after they log in. Coordinator and board member accounts will be created by the administrator in the same way. If a user forgets their password, they can change it through their email.

Profile

All users will have a profile page through which they can change their contact information and passwords. They can also choose whether to display their information on their public profile.

Forum

There will be a forum feature accessible through the main page. The forum will be used to create posts that all students and coordinators can see and reply to. For example, a student may post their questions about the visa process for a certain country and other students who will go or who have gone to the same country may help by replying.

The posts' subject lines will be listed on the forum page, as well as the person who created the post, the last person who replied to the post, and the number of replies to the post. After they click on the post, users can see the main text of the post and the replies it received. Through the same page, they can write a reply and by clicking on the name of the people who sent a reply, users can access their profiles.

FAQ

FAQ (frequently asked questions) page will be available on the site. While students will have no authority to change the FAQ, the coordinator will be able to add questions along with answers to the FAQ or remove questions from this part. FAQ is necessary to give a general understanding of the Erasmus process to students and answer their most common questions without emails.

Messages

Messaging is a way of direct communication between the coordinator and students. On the left side of the messaging page, there will be a list of people chronologically ordered from the most recent person to the first person that the user messaged. New messages will appear here as well. Users can access other users' profiles from that list. To continue an existing chat, users can click one of those messages from the list. To start a new chat, the user will click the "new chat" button and select a person from a list of users, which they can search or filter. The coordinator and students are able to start messaging anytime with any student. While chatting with

a person, on the upper side of the page, the coordinator can see some information about the student like name, surname, phone number, email, status, and mobility period. While students are chatting with other students, they can see their name, applied university, and phone number.

Users will be able to filter messages. This feature aims to reduce the amount of time spent on searching messages. Users can filter by name, the university applied for Erasmus, or the country and sort alphabetically or chronologically.

Choosing courses

Students have to add courses that they will take in their Erasmus semester. The Erasmus Application system aims to enable students to conduct this process in an efficient way. Course-related operations can be accessed from the main page under the “Courses” title. Students will see the number of courses that they are planning to take with course name, id (such as CS319), ECTS credit, and their equivalent courses in Bilkent. The total amount of ECTS credits will be at the bottom of the list. At the bottom of the page, there will be two buttons, one of which is “add previously approved courses”. Through this button, students will be able to view and choose courses that were previously accepted by Bilkent. The other button, “add a new course”, will be used if the student wants to take a course that was not previously approved by Bilkent. If the new course is the equivalent of an elective course in Bilkent, the coordinator will be notified to either approve or reject the course. However, if the new course will be taken instead of a mandatory course in Bilkent, the instructor of the said course will be sent an email by the system and the instructor’s response will be shown to the coordinator, which will then either approve or reject the course. If the coordinators reject a course, they may choose to include a reason. In addition, courses with low credit can be merged to be equivalent with courses that have high credit, which requires the coordinator’s approval as well.

To-do List

Students and coordinators will have a to-do list on their homepage. The content of this for students is a list of actions to be done in the Erasmus process. Also, students can remove tasks from the list and add new tasks to their to-do list at any time. The

coordinator's to-do list will have a list of waiting for course approvals. Likewise, a coordinator can remove a task or add a new task to their list. When a task is done, it will be moved to the completed items list on the main page.

Notification

All users have a notification panel. Notifications can be accessed from the main page of the site. Students and coordinators can be notified about new messages, upcoming events, approvals/rejections, new answers to their posts in the forum, and successful document uploading/downloading.

File Handling

In the Erasmus application process, students deal with some documents such as the pre-approval form, the learning agreement, or the course transfer form. Our site will enable students and coordinators to generate such documents using their planned courses, stored on the site. In addition, they will be able to download these documents in .doc or .pdf formats and upload their own documents at any time. They will also be able to upload an image of their signature, which the site will automatically add to the generated files. Moreover, if a change is made to a file, it will optionally be highlighted to make it easier to spot differences.

Student list

The coordinator will be able to access the list of all students. Students will be displayed with their information and clicking on their name will direct the page to their profile. There will be a button next to each student's name that will direct the coordinator to the message page for that student. The coordinator can see the responsible coordinator for a student on the list. Students may be filtered based on their applied university, mobility period, etc.

Status

On the student's main page, their current status will be shown. The status is the current step in the student's Erasmus process, such as waiting for approval from the coordinator, uploading a specific file, and so on. The status of students can also be seen from their profile by the coordinator.

2.2 Non-functional Requirements

Reducing Unnecessary Interaction

Coordinators responsible for Erasmus are exposed to many irrelevant questions apart from their profession. As a solution to this issue, firstly, we will include a FAQ page (frequently asked questions) on our site. Secondly, there will be a forum where all students can ask or answer questions. The forum will help students to find the answers to their questions before those questions arrive at the instructor.

Extendibility

In this system, Object Oriented Model will be utilized and the architecture of the system that we designed all with models and diagrams will allow developers to extend and update it later on when required. Moreover, in order to ensure maintainability, the code will be properly commented and documented.

Usability

The site will have an intuitive user interface to make it easy for users to understand the fundamentals of the program since the site will keep interacting with users throughout its life. For this purpose, it will have a clear and easily understandable user interface. The site will meet users' demands by putting all the necessary features enabling easy information sharing together.

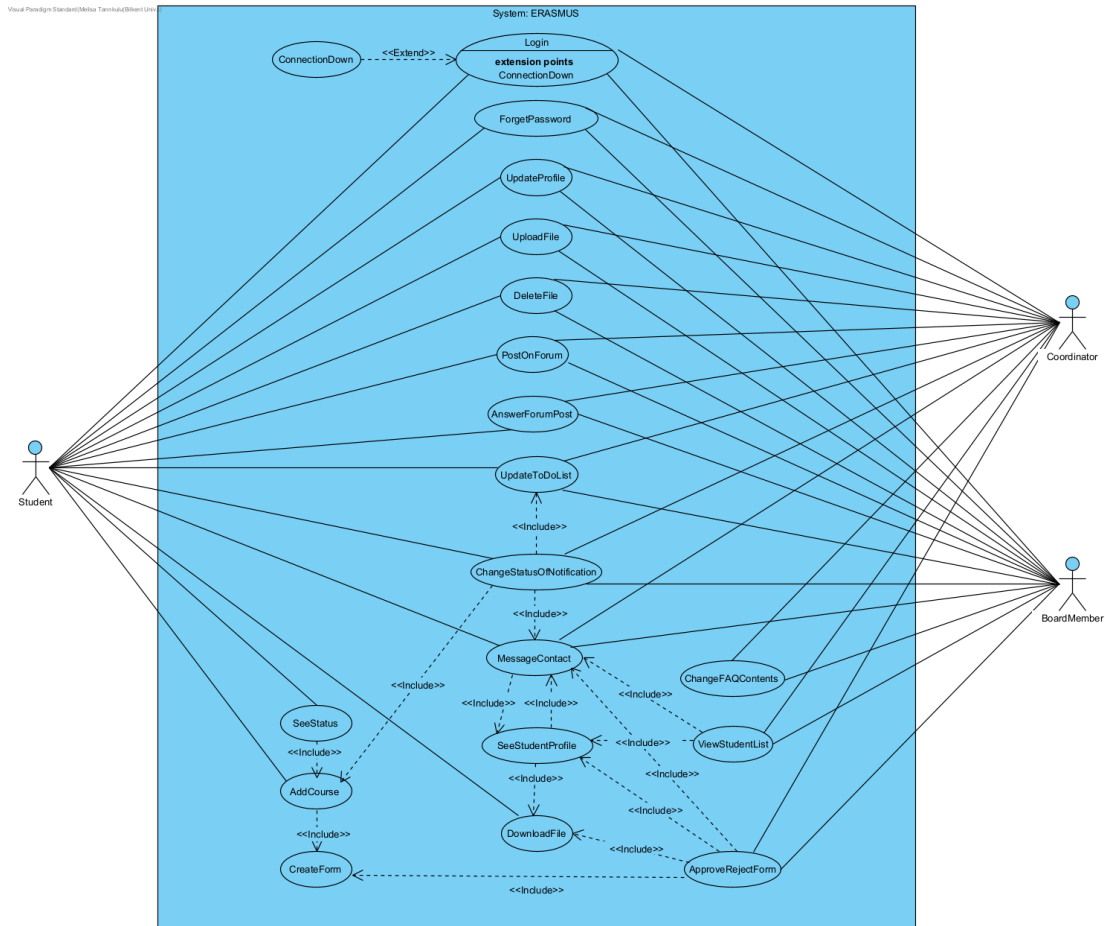
Reliability

All personal information (password, GPA, messages etc.) about users will be kept secret. Students also have the option to share information about themselves to other students like phone number, email address, university, mobility period, and other information.

2.3 System Models

2.3.1 Use-Case Model

The use case diagram given above shows the general structure of our web-based Erasmus Application. There are 19 use cases. ERASMUS refers to the system behind our website.



Use Case # 1

1. **Name:** Login
2. **Participating actors:** Student, Coordinator, BoardMember
3. **Entry condition:**
 - The ACTOR has an existing account in the system.
4. **Exit condition:**
 - The ACTOR gains access to the system to use any functionality within.
5. **Flow of events:**
 1. The ACTOR enters their Bilkent ID and password.
 2. ERASMUS compares the given login information with the ones in the user database. If they match, the ACTOR is authenticated. Otherwise, the terminal rejects the authentication attempt.
6. **Special/quality requirements:**
 - ERASMUS connects to the server within 30 seconds.

Use Case # 2

1. **Name:** ForgotPassword
2. **Participating actors:** Student, Coordinator, BoardMember

3. **Entry condition:**
 - The ACTOR has an existing account in the system.
4. **Exit condition:**
 - The ACTOR's password changes.
5. **Flow of events:**
 1. The ACTOR activates the "Forgot Password" function of their terminal.
 2. ERASMUS responds by requesting the ACTOR's Bilkent ID.
 3. The ACTOR enters their Bilkent ID.
 4. ERASMUS responds by sending a message including a password reset link to the email, which belongs to the given Bilkent ID account.
 5. The ACTOR clicks on the link and enters a new password on the directed page and submits it.
 6. ERASMUS updates the database.

Use Case # 3

1. **Name:** UpdateToDoList
2. **Participating actors:** Student, Coordinator, BoardMember
3. **Entry condition:**
 - The ACTOR is logged into ERASMUS.
4. **Exit condition:**
 - The ACTOR changes/updates the to-do list.
5. **Main Flow of events:**
 1. The ACTOR activates the "To-Do List" function of their terminal.
 2. ERASMUS responds by presenting the to-do list and done list content of the ACTOR.
 3. The ACTOR activates the "Add New To-Do" function of their terminal.
 4. ERASMUS responds by presenting a form to the ACTOR. The form includes the name of the to-do item, and a placeholder for the link.
 5. The ACTOR completes the form by specifying minimally the name of the to-do item and submits the form.
 6. ERASMUS receives the form, updates the to-do list and displays the updated list.
6. **Alternative Flow of events:**
 1. The ACTOR wants a to-do list item to be marked as done.
 1. The ACTOR selects a to-do item from the list and specifies it as done.
 2. ERASMUS responds by moving the to-do item to the done list, and displaying the updated lists.
 2. The ACTOR wants a to-do list item to be flagged.
 1. The ACTOR selects a to-do item from the list and specifies it as flagged.
 2. ERASMUS responds by marking the selected to-do item as flagged and displays the updated list.
 3. The ACTOR wants a to-do list item to be deleted.

1. The ACTOR selects a to-do item from the list and specifies it as to be deleted.
2. ERASMUS responds by deleting the item from the to-do list, and displays the updated list.
4. The ACTOR wants a done list item to be moved to the to-do list.
 1. The ACTOR selects a done list item and specifies it as to-do item.
 2. ERASMUS responds by moving the done list item to the to-do list and displays the updated lists.

Use Case # 4

1. **Name:** ChangeStatusOfNotification
2. **Participating actors:** Student, Coordinator, BoardMember
3. **Entry condition:**
 - The ACTOR is logged into ERASMUS.
4. **Exit condition:**
 - The ACTOR changes the status of a notification.
5. **Main Flow of events:**
 1. The ACTOR activates the “Notifications” function of their terminal.
 2. ERASMUS responds by presenting the content of the notifications list of the ACTOR.
 3. The ACTOR selects a notification and specifies it as read.
 4. ERASMUS updates the database, responds by marking the notification as read and displays the updated list.
6. **Alternative Flow of events:**
 1. The ACTOR wants to delete a notification.
 1. The ACTOR selects a notification and specifies it as to be deleted.
 2. ERASMUS updates the database, responds by deleting the notification from the list and displays the updated list.
 2. The ACTOR wants to flag a notification.
 1. The ACTOR selects a notification and specifies it as flagged.
 2. ERASMUS updates the database, responds by marking the notification as flagged and displays the updated list.
 3. The ACTOR wants to be directed to the page associated with the notification.
 1. The ACTOR clicks on a notification linkage.
 2. ERASMUS responds by directing the ACTOR to AddCourse, MessageContact, or UpdateToDoList use cases according to the notification.

Use Case # 5

1. **Name:** MessageContact
2. **Participating actors:** Student, Coordinator, BoardMember
3. **Entry condition:**
 - The ACTOR is logged into ERASMUS.

4. Exit condition:

- The ACTOR sends a message to a contact.
- The contact receives a notification about the message.

5. Main Flow of events:

1. The ACTOR activates the “Messages” function of their terminal.
2. ERASMUS responds by presenting the list of contacts that the ACTOR has received/sent any message from/to.
3. The ACTOR selects a contact from the list.
4. ERASMUS responds by presenting the message content with the selected contact and the information (full name, phone number, e-mail, etc.) of that contact.
5. The ACTOR reads the message(s) they received from that contact, and responds with another message.
6. ERASMUS updates the database, sends the message to the contact and displays the updated message content.

6. Alternative Flow of events:

1. The ACTOR wants to create a new chat.
 1. The ACTOR activates the “New Chat” function of their terminal.
 2. ERASMUS responds by displaying the list of possible contacts to create a new chat.
 3. The ACTOR selects a contact from the list.
 4. ERASMUS responds by displaying an empty chat with the selected contact, and the information (full name, phone number, e-mail, etc.) of that contact.
 5. The ACTOR sends a message to the contact.
 6. ERASMUS updates the database, sends the message to the contact and displays the updated message content.

7. Special/quality requirements:

- At any point during the flow of events, this use case can include the SeeStudentProfile use case. The SeeStudentProfile is initiated when the COORDINATOR or BOARDMEMBER clicks on the STUDENT’s profile link.

Use Case # 6

1. Name: UpdateProfile

2. Participating actors: Student, Coordinator, BoardMember

3. Entry condition:

- The ACTOR is logged into ERASMUS.

4. Exit condition:

- The ACTOR updates their profile.

5. Main Flow of events:

1. The ACTOR activates the “Update Profile” function of their terminal.
2. ERASMUS responds by displaying the contents of the ACTOR’s profile (full name, email, ID, etc.).
3. The ACTOR activates the “Change Password” function of their terminal.

4. ERASMUS responds by presenting a form to the ACTOR. The form requires the ACTOR to enter their old password, and the new password.
 5. The ACTOR fills the form and submits it.
 6. ERASMUS updates the database.
- 6. Alternative Flow of events:**
1. The ACTOR wants to change their email.
 1. The ACTOR activates the “Change Email” function of their terminal.
 2. ERASMUS responds by presenting a form to the ACTOR. The form requires the ACTOR to enter their new email.
 3. The ACTOR fills the form and submits it.
 4. ERASMUS updates the database.
 2. The ACTOR wants to change their preferences.
 1. The ACTOR activates the “Change Preferences” function of their terminal.
 2. ERASMUS responds by presenting their preferences to the ACTOR.
 3. The ACTOR updates their preferences.
 4. ERASMUS updates the database.

· Use Case # 7

1. **Name:** UploadFile
2. **Participating actors:** Student, Coordinator, BoardMember
3. **Entry condition:**
 - The ACTOR is logged into ERASMUS.
4. **Exit condition:**
 - The ACTOR uploads a new file.
5. **Flow of events:**
 1. The ACTOR activates the “Upload File” function of their terminal.
 2. ERASMUS responds by presenting a form for the ACTOR to upload their file.
 3. The ACTOR uploads the file to the system.
 4. ERASMUS updates the database and displays the updated file list.

Use Case # 8

1. **Name:** DownloadFile
2. **Participating actors:** Student, Coordinator, BoardMember
3. **Entry condition:**
 - The ACTOR is logged into ERASMUS.
4. **Exit condition:**
 - The ACTOR downloads a new file.
5. **Flow of events:**
 1. The ACTOR activates the “Download File” function of their terminal.
 2. ERASMUS responds by presenting the list of the user’s files.
 3. The ACTOR selects a file to download.

4. ERASMUS enables the download process.

Use Case # 9

1. **Name:** DeleteFile
2. **Participating actors:** Student, Coordinator, BoardMember
3. **Entry condition:**
 - The ACTOR is logged into ERASMUS.
4. **Exit condition:**
 - The ACTOR deletes a file.
5. **Flow of events:**
 1. The ACTOR activates the “Delete File” function of their terminal.
 2. ERASMUS responds by presenting the list of the user’s files.
 3. The ACTOR selects a file to delete.
 4. ERASMUS updates the database, enables the delete process and displays the updated file list.

Use Case # 10

1. **Name:** ChangeFAQContents
2. **Participating actors:** Coordinator, BoardMember
3. **Entry condition:**
 - The ACTOR is logged into ERASMUS.
4. **Exit condition:**
 - The ACTOR changes the contents of the FAQ.
5. **Main Flow of events:**
 1. The ACTOR activates the “FAQ” function of their terminal.
 2. ERASMUS responds by presenting the FAQ content.
 3. The ACTOR selects an item and specifies it as to be deleted.
 4. ERASMUS updates the database, responds by deleting the selected item and displays the updated FAQ list.
6. **Alternative Flow of events:**
 1. The ACTOR wants to add a new item to the FAQ list.
 1. The ACTOR activates the “FAQ” function of their terminal.
 2. ERASMUS responds by presenting the FAQ content.
 3. The ACTOR activates the “Post” function of their terminal.
 4. ERASMUS responds by presenting a form to the ACTOR. The form includes the question and the corresponding answer.
 5. The ACTOR fills out the form and submits it.
 6. ERASMUS updates the database, updates the database and displays the updated post list.

Use Case # 11

1. **Name:** PostOnForum
2. **Participating actors:** Student, Coordinator, BoardMember
3. **Entry condition:**

- The ACTOR is logged into ERASMUS.
- 4. **Exit condition:**
 - The ACTOR creates a new post on the forum.
- 5. **Flow of events:**
 1. The ACTOR activates the “Forum” function of their terminal.
 2. ERASMUS responds by presenting the Forum content.
 3. The ACTOR activates the “New Post” function on their terminal.
 4. ERASMUS responds by presenting a form to the ACTOR. The form includes the subject and the content of the post.
 5. The ACTOR fills the form and submits it.
 6. ERASMUS receives the form, updates the database and the forum content, and displays it.

Use Case # 12

1. **Name:** AnswerForumPost
2. **Participating actors:** Student, Coordinator, BoardMember
3. **Entry condition:**
 - The ACTOR is logged into ERASMUS.
4. **Exit condition:**
 - The ACTOR sends a comment on a post on the forum.
5. **Flow of events:**
 1. The ACTOR activates the “Forum” function of their terminal.
 2. ERASMUS responds by presenting the Forum content.
 3. The ACTOR selects a post.
 4. ERASMUS responds by presenting the post content.
 5. The ACTOR activates the “Answer” function on their terminal.
 6. ERASMUS responds by presenting a form to the ACTOR. The form includes the content of the answer.
 7. The ACTOR fills the form and submits it.
 8. ERASMUS receives the form, updates the database and the post content, and displays it.

Use Case # 13

1. **Name:** ViewStudentList
2. **Participating actors:** Coordinator, BoardMember
3. **Entry condition:**
 - The ACTOR is logged into ERASMUS.
4. **Exit condition:**
 - The ACTOR views the student list.
5. **Main Flow of events:**
 1. The ACTOR activates the “Student List” function of their terminal.
 2. ERASMUS responds by presenting the student list to the actor.
6. **Alternative Flow of events:**
 1. The ACTOR wants to send a message to the student.
 1. The ACTOR selects a student from the list to send a message.

2. ERASMUS responds by invoking the MessageContact use case.
2. The ACTOR wants to send a message to the coordinator of a student.
 1. The ACTOR selects a student's coordinator to send a message.
 2. ERASMUS responds by invoking the MessageContact use case.
3. The ACTOR wants to see the profile of a student.
 1. The ACTOR selects a student to see their profile.
 2. ERASMUS responds by invoking the SeeStudentProfile use case.

Use Case # 14

1. **Name:** SeeStudentProfile
2. **Participating actors:** Coordinator, BoardMember
3. **Entry condition:**
 - The ACTOR is logged into ERASMUS.
4. **Exit condition:**
 - The ACTOR views a student's profile.
5. **Main Flow of events:**
 1. The ACTOR activates the "See Student Profile" function of their terminal.
 2. ERASMUS responds by displaying the student's information (email, phone number, university/mobility period, status and ID).
6. **Alternative Flow of events:**
 1. The ACTOR wants to send a message to the student.
 1. The ACTOR activates the "Message Contact" function on their terminal.
 2. ERASMUS responds by invoking the MessageContact use case.
 2. The ACTOR wants to see student's courses.
 1. The ACTOR activates the "Student's Courses" function of their terminal.
 2. ERASMUS responds by displaying the list of courses the student is taking.
 3. The ACTOR wants to see the student's files.
 1. The ACTOR activates the "See Student's Files" function of their terminal.
 2. ERASMUS responds by displaying the list of files the student uploaded.
 3. The ACTOR selects a file to download.
 4. ERASMUS responds by invoking the DownloadFile use case.

Use Case # 15

1. **Name:** AddCourse
2. **Participating actors:** Student
3. **Entry condition:**

- STUDENT is logged into ERASMUS.
4. **Exit condition:**
- STUDENT adds a new course.
 - COORDINATOR is informed about the Course Approval form.
5. **Main Flow of events:**
1. STUDENT activates the “Add Course” function of their terminal.
 2. ERASMUS responds by presenting the list of the courses by the coordinate and their information (course name, course ID, ECTS credits, the course equivalent to it according to Bilkent courses, ECTS total credits) STUDENT added, which are approved by COORDINATOR and BOARDMEMBER.
 3. STUDENT activates the “Add Previously Approved Courses” function of their terminal.
 4. ERASMUS responds by presenting the list of previously approved courses.
 5. STUDENT selects a course they want to add.
 6. ERASMUS responds by adding the selected course to the course list.
 7. STUDENT submits the final course list.
 8. ERASMUS updates the database and responds by invoking CreateForm use case.
6. **Alternative Flow of events:**
1. STUDENT wants to add a new course.
 1. STUDENT activates the “Add Course” function of their terminal.
 2. ERASMUS responds by presenting the list of the courses and their information (course name, course ID, ECTS credit, the course equivalent to it according to Bilkent courses, total ECTS credits of the courses) STUDENT added.
 3. STUDENT activates the “Add New Course” function of their terminal.
 4. ERASMUS responds by presenting a form to the ACTOR. The form includes the name, ID, ECTS credit, the equivalent, and the link of the course.
 5. STUDENT fills the form and submits it.
 6. ERASMUS receives the form, updates the course list and displays it.
 7. STUDENT submits the final course list.
 8. ERASMUS updates the database and responds by invoking CreateForm use case.
 2. STUDENT wants to merge courses.
 1. STUDENT selects the courses to be merged and activates the “Merge Course” function of their terminal.
 2. ERASMUS responds by merging the courses.
 3. STUDENT submits the final course list.
 4. ERASMUS updates the database and responds by invoking CreateForm use case.

1. **Name:** CreateForm
2. **Participating actors:** Student, Coordinator, BoardMember
3. **Entry condition:**
 - The ACTOR is logged into ERASMUS.
4. **Exit condition:**
 - A form with given information is created.
5. **Main Flow of events:**
 1. ERASMUS creates the Course Approval form with the given course list and updates the database.
6. **Alternative Flow of events:**
 1. COORDINATOR or BOARDMEMBER approved/rejected the Course Approval form.
 1. ERASMUS updates the database, updates the Course Approval form with the COORDINATOR/BOARDMEMBER's response and informs STUDENT.

Use Case # 17

1. **Name:** Approve/RejectForm
2. **Participating actors:** Coordinator, BoardMember
3. **Entry condition:**
 - The ACTOR is logged into ERASMUS.
4. **Exit condition:**
 - The ACTOR approved/rejected the course approval form.
 - STUDENT is informed about the status of the file.
5. **Main Flow of events:**
 1. The ACTOR activates the "Course Approval" function of their terminal.
 2. ERASMUS responds by displaying the course information (course name, course ID, ECTS credit, the equivalent of the course and the course link) and the form created by the system.
 3. The ACTOR activates the "Download File" function of their terminal.
 4. ERASMUS responds by invoking the DownloadFile use case.
 5. The ACTOR views the form and approves or rejects the form accordingly.
 6. ERASMUS invokes the CreateForm use case to update the Course Approval form.
6. **Alternative Flow of events:**
 1. The ACTOR wants to see the profile of the STUDENT.
 1. The ACTOR activates the "Course Approval" function of their terminal.
 2. ERASMUS responds by displaying the course information (course name, course ID, ECTS credit, the equivalent of the course and the course link) and the form created by the system.
 3. The ACTOR activates the "See Student Profile" function of their terminal.
 4. ERASMUS responds by invoking the SeeStudentProfile use case.

2. The ACTOR wants to send a message to the student.
 1. The ACTOR activates the “Course Approval” function of their terminal.
 2. ERASMUS responds by displaying the course information (course name, course ID, ECTS credit, the equivalent of the course and the course link) and the form created by the system.
 3. The ACTOR activates the “Message Student” function of their terminal.
 4. ERASMUS responds by invoking the MessageContact use case.

Use Case # 18

1. **Name:** SeeStatus
2. **Participating actors:** Student
3. **Entry condition:**
 - STUDENT is logged into ERASMUS.
4. **Exit condition:**
 - STUDENT is notified about their status.
5. **Flow of events:**
 1. STUDENT activates the “See Status” function of their terminal.
 2. ERASMUS responds by displaying the status of the student.
 3. STUDENT selects their status to be directed to the related page.
 4. ERASMUS responds by invoking AddCourse use case.

Use Case # 19

6. **Name:** ConnectionDown
7. **Participating actors:** Student, Coordinator, BoardMember
8. **Entry condition:**
 - This use case extends the Login use case. It is initiated by the system whenever the network connection between the ACTOR and the system is lost.
9. **Exit condition:**
 - The ACTOR is logged out from ERASMUS.
10. **Flow of events:**
 1. STUDENT is logged out from the system by ERASMUS and it invokes the Login use case.

2.3.2 Object and Class Models

The class diagram given above shows the general structure of our web-based Erasmus Application. There are 18 classes.

previously if a coordinator approves them. They can merge two or more courses if their credit is lower than required.

University Class: University class includes general information about universities and the student list for selected universities.

EmailSender Class: When adding new courses, it may be required to get approval from the coordinator of the equivalent course. In this circumstance, students can send an email to the course coordinator by using this class.

Document class: Document class is used for creating or uploading new documents. Users can also sign documents

Chat Class: Chat class is holding chats between users (coordinators and students.)

Message Class: Message class is for sending messages.

Notification class: Notification class is used to notify users about new updates, info, or requirements.

ToDo Class: ToDo class holds users' todos. Users can mark todos as done or add new ones.

FAQ Class: The FAQ class can be modified by coordinators. Coordinators add a new FAQ and their answers.

Question Class: The question class is a subclass of the FAQ.

User Controller: The User class is used for user management

Forum Class: Forum is a class for asking a question to the public (all Erasmus students and coordinators), and everybody can answer that.

Post Class: Post class is used for adding a new post to the forum.

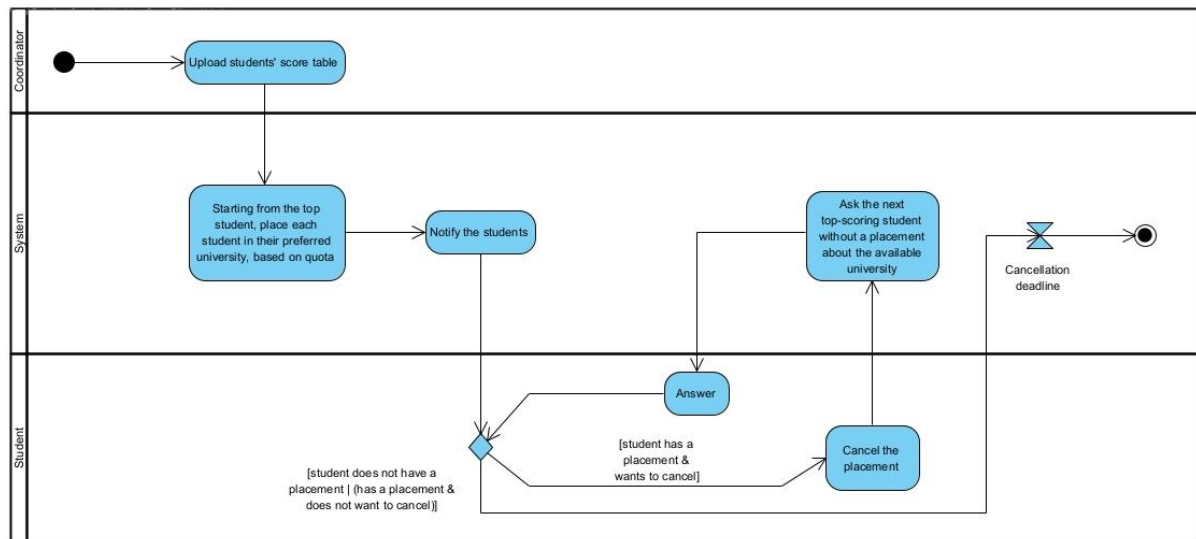
Response Class: Response class is used for replying to posts or previous responses.

2.3.3 Dynamic Models

2.3.3.1 Activity Diagrams

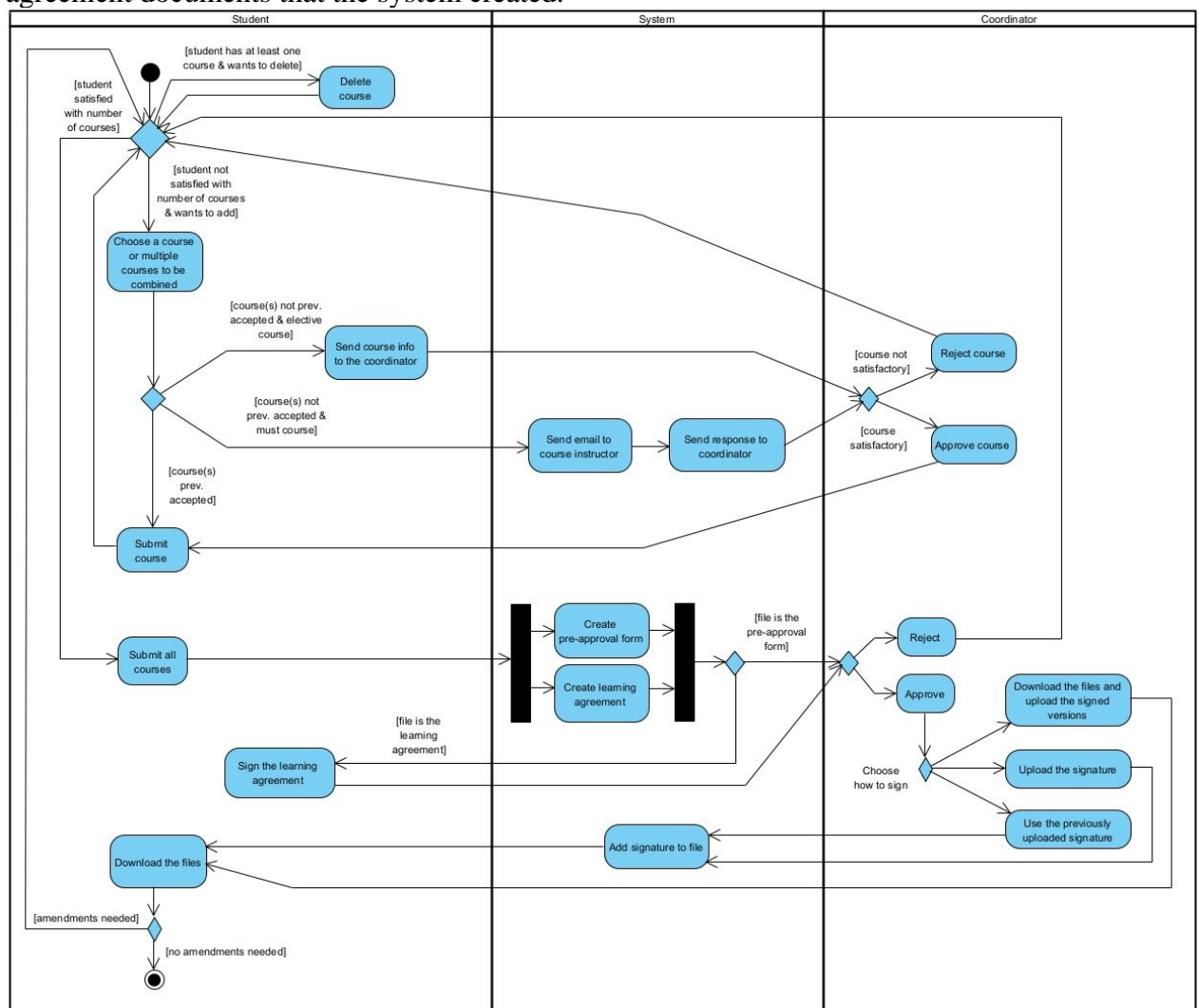
Student placement activity diagram:

The diagram below shows the student placement process according to students' scores and preferences.



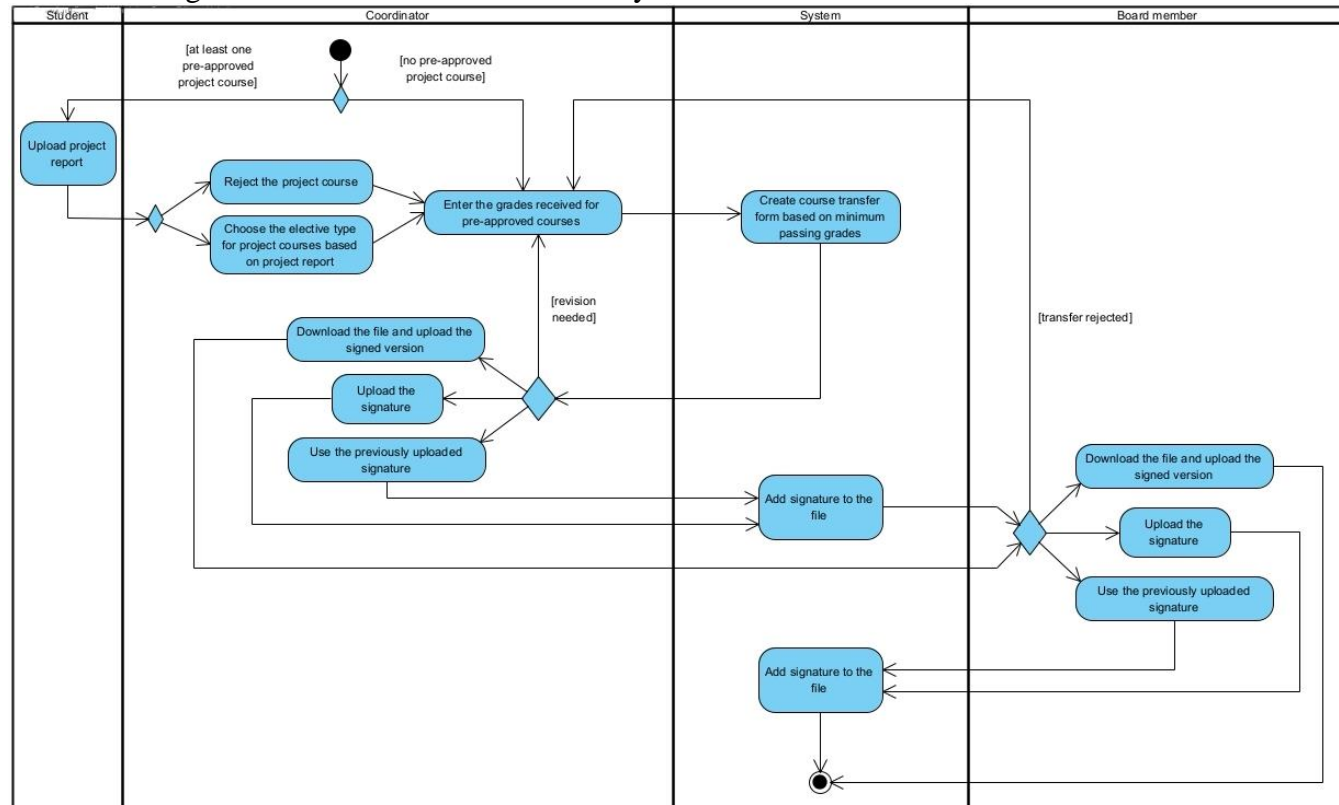
Course choosing activity diagram:

The diagram below shows the process of students choosing courses for their mobility period, coordinators evaluating them, and signing the pre-approval and learning agreement documents that the system created.



Course transfer activity diagram:

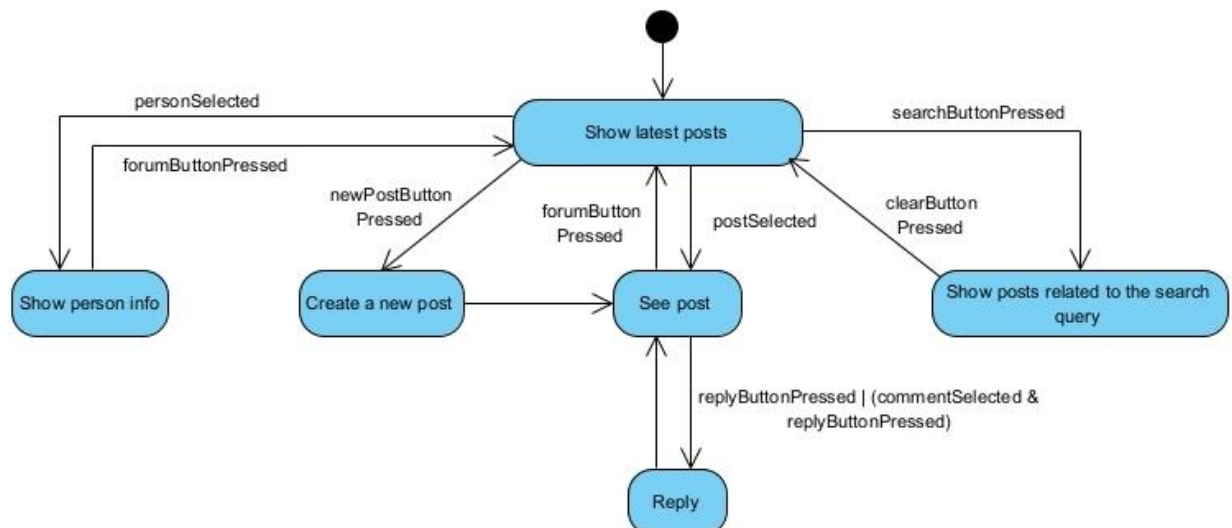
The diagram below shows the process in which, after the mobility period, the courses students took get transferred to Bilkent University.



2.3.3.2 State Diagrams

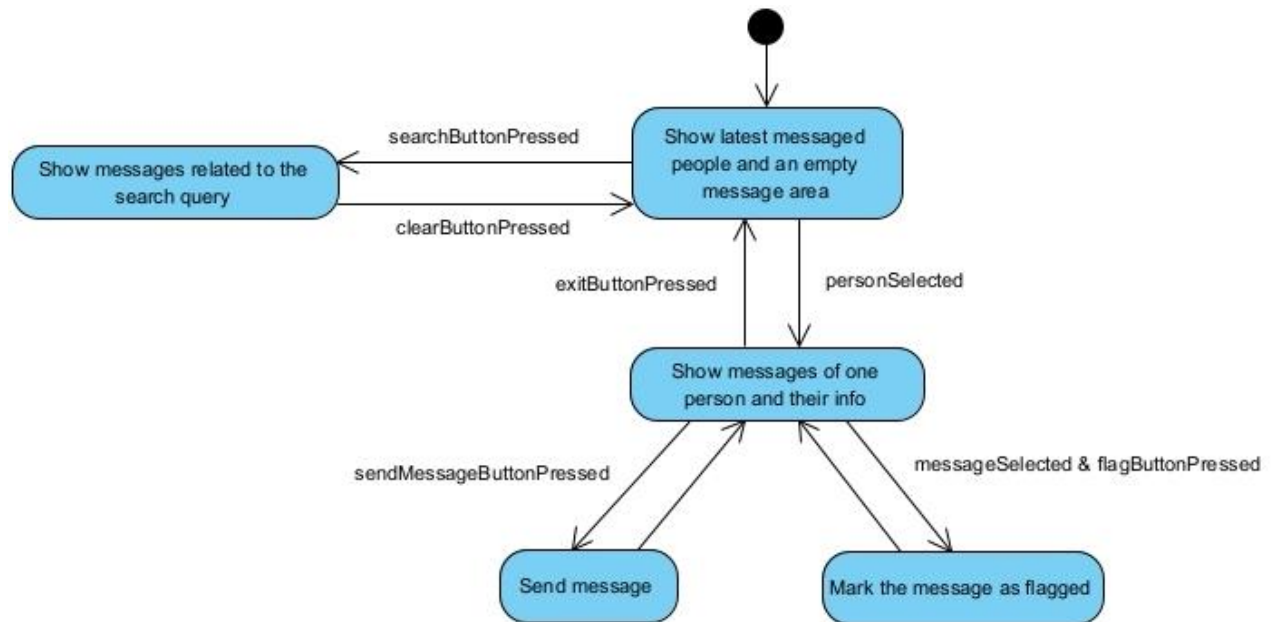
Forum state diagram:

The diagram below indicates what the forum page may show according to the pressed buttons.



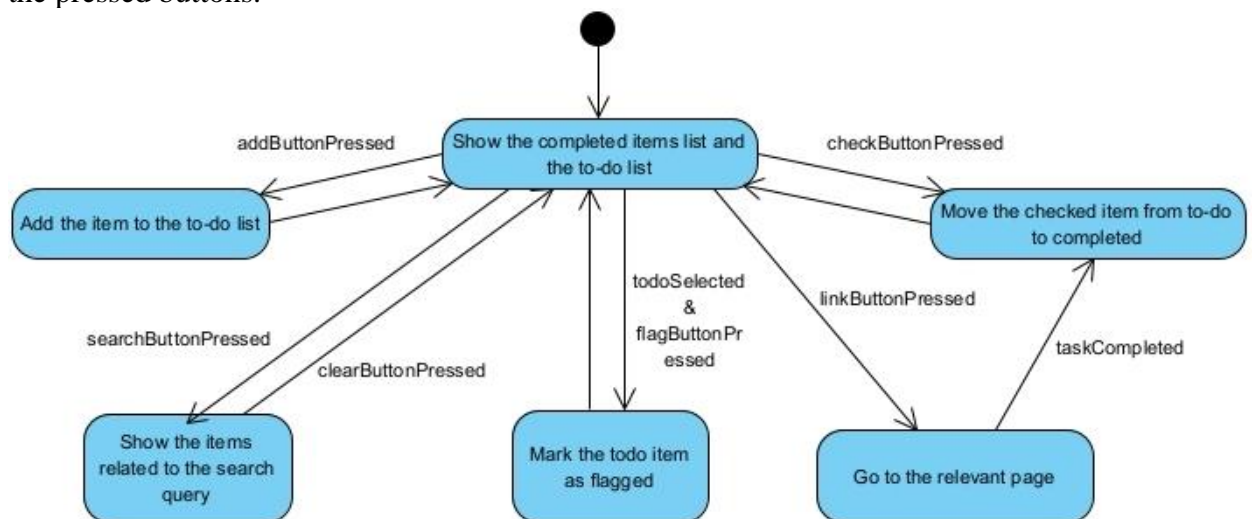
Message system state diagram:

The diagram below shows what the messages page may do according to the pressed buttons.



To-do list state diagram:

The diagram below shows what the to-do list on the main page may do according to the pressed buttons.

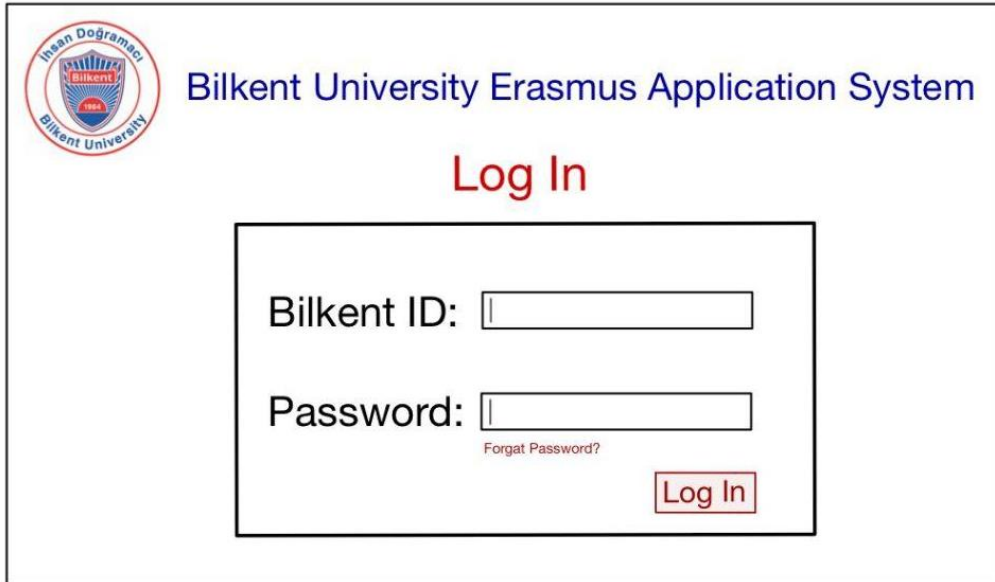


2.3.4 User Interface

2.3.4.1 Common Interfaces

Common interfaces are used by both users. For these pages, the only noticeable change between users is on the menu to the left. Coordinators have a “student list” menu button whereas students have a “courses” button.

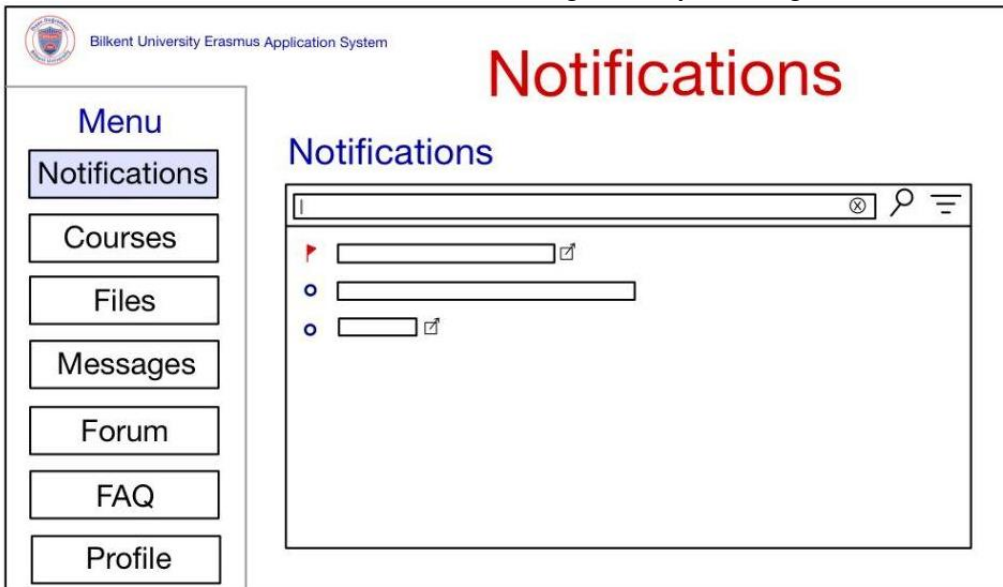
Log-In Page



The screenshot shows the login page of the Bilkent University Erasmus Application System. In the top left corner is the Bilkent University logo, which includes the text "İhsan Doğramacı Bilkent University". To the right of the logo, the text "Bilkent University Erasmus Application System" is displayed in blue. Below this, the words "Log In" are written in a large, bold, red font. A central white box with a black border contains the login fields: "Bilkent ID:" followed by a text input field, and "Password:" followed by a text input field. Below the password field is a small red link that says "Forgot Password?". To the right of the password field is a red "Log In" button.

Notifications


Users can select notifications to delete or flag them by clicking on the circles.



The screenshot shows the notifications page of the Bilkent University Erasmus Application System. At the top left is the Bilkent University logo and the text "Bilkent University Erasmus Application System". The word "Notifications" is written in a large, bold, red font. On the left side, there is a vertical menu with the title "Menu" and several buttons: "Notifications" (highlighted with a blue border), "Courses", "Files", "Messages", "Forum", "FAQ", and "Profile". The main content area is titled "Notifications" in blue. It features a search bar at the top with a magnifying glass icon and a close icon. Below the search bar, there is a list of notifications. Each notification entry consists of a small red flag icon, a text input field, and a small square icon with a checkmark. The first notification has a red flag icon, the second has a red circle icon, and the third has a red circle icon.

Files

Through this page, users can access their files and download them.

 Bilkent University Erasmus Application System

Files

Menu

Notifications

Courses

Files

Messages

Forum


FAQ

Profile

Filename.pdf	View	Download	<input type="checkbox"/>
Filename.pdf	View	Download	<input type="checkbox"/>

Upload File

Upload File

 Bilkent University Erasmus Application System

Files

Menu

Notifications

Courses

Files

Messages

Forum

FAQ

Profile


Drag Your File Here

Choose File

Submit

Forum

Any keyword can be searched to find a particular post on the forum. Posts can also be filtered by their subjects, posters, and dates.

 Bilkent University Erasmus Application System

Forum

Menu

Notifications

Courses

Files

Messages

Forum

FAQ


Profile

Subject	Started By	Last Post	Replies
Subject	Name Surname	Name Surname	3
Subject	Name Surname	Name Surname	1

New Post

Viewing Post

All replies to a particular post are aligned in the same line. A reply is indented a little right of the replied text.

 Bilkent University Erasmus Application System

Forum

Menu

Notifications

Courses

Files

Messages

Forum

FAQ

Profile

Subject

Name Surname

Date

Name Surname

Date

Name Surname

Date

Answer

New Post or Replying to a Post

Although the interfaces are quite similar, the subject option is not shown if a user is replying to a post.

The screenshot shows the 'Forum' page of the Bilkent University Erasmus Application System. On the left is a 'Menu' sidebar with buttons for Notifications, Courses, Files, Messages, Forum (highlighted), FAQ, and Profile. The main content area is titled 'Forum' in red. It contains a 'Subject:' label followed by a text input field. Below this is a large text area for the post content. At the bottom right of the main area is a red 'Post' button.

Viewing Own Profile

According to whether the user is a student or a coordinator, the information displayed may change. Along with viewing information, users also can change their password, and mail address. From the preferences, they can choose to share their specific information like their phone number, mail address with other users.

The screenshot shows the 'Profile' page of the Bilkent University Erasmus Application System. On the left is a 'Menu' sidebar with buttons for Notifications, Courses, Files, Messages, Forum, FAQ, and Profile (highlighted). The main content area is titled 'Profile' in red. It features a user profile card with a circular icon placeholder and the text 'Name Surname'. Below the card are input fields for 'Email', 'ID', and 'University'. To the right of these fields is a box containing three links: 'Change Password', 'Change Mail', and 'Preferences'.

2.3.4.2 Student Interfaces

Main Page

Status shows the current state of the Erasmus process. For example, a student may be waiting for approval from the coordinator or the student may be expected to upload a specific file etc.

The Main Page interface features a sidebar menu on the left with options: Notifications, Courses, Files, Messages, Forum, FAQ, and Profile. The main content area is titled 'Main Page' in red. It includes a 'Status' section with a text box displaying 'Waiting for approval from the coordinator'. Below this are two sections: 'To Do' and 'Done', each containing a list of tasks with checkboxes and a search bar. An 'Add To Do' button is located at the bottom of the 'To Do' section.

Courses

The Courses interface features a sidebar menu on the left with options: Notifications, Courses (highlighted), Files, Messages, Forum, FAQ, and Profile. The main content area is titled 'Courses' in red. It includes a section titled 'Current Courses Taken' with a table showing columns for Course Name, Course ID, Ects, and Taken Instead of. Below the table is an 'Ects Total' field. An 'Add Course' section contains two buttons: 'Add previously approved course' and 'Add new course'.

Course Name	Course ID	Ects	Taken Instead of

Add Previously Approved Course

Currently taken courses's list is shown above for every course adding process.

**Bilkent University Erasmus Application System**

Courses

Menu

Notifications

Courses

Files

Messages

Forum

FAQ

Profile

Current Courses Taken


Course Name	Course ID	Ects	Taken Instead of

Ects Total:

Approved Course List

Course Name	Course ID	Ects	Can Be Taken Instead of
			<input type="button" value="Select"/>
			<input type="button" value="Select"/>
			<input type="button" value="Select"/>

Add New Course

**Bilkent University Erasmus Application System**

Courses

Menu

Notifications

Courses

Files

Messages

Forum

FAQ

Profile

Current Courses Taken

Course Name	Course ID	Ects	Taken Instead of


Ects Total:

Course Information

Course Name	Course ID	Ects	Taken Instead of	Course Info Link

Merge Courses

Multiple courses can be merged into one to cover a Bilkent course. Students can give required information of courses in required areas.

**Bilkent University Erasmus Application System**

Courses

Menu

Notifications

Courses

Files

Messages

Forum

FAQ

Profile

Current Courses Taken

Course Name	Course ID	Ects	Taken Instead of

Ects Total:

Taken Instead of:


Course Name	Course ID	Ects	Course Info Link

Ects Total:

Submit

Messages

Students can only see other students' names, and other information if the students allowed sharing.

**Bilkent University Erasmus Application System**

Messages

Menu

Notifications

Courses

Files

Messages

Forum

FAQ

Profile

Name Surname >

Name Surname >

Name Surname >

Name Surname >

Name Surname >

New Chat

Name Surname

University

Phone No

Email

30

Frequently Asked Questions

Plus signs extend the questions to display their answers and the minus signs remove the answers from the view.

The screenshot shows the 'Frequently Asked Questions' page. On the left is a 'Menu' sidebar with buttons for Notifications, Courses, Files, Messages, Forum, FAQ (highlighted), and Profile. The main content area has a title 'Frequently Asked Questions' in red. Below the title is a search bar with a magnifying glass icon and a list icon. The content area contains three question entries: 'A question text example 1' with a plus sign, 'Question 2' with a minus sign and an 'Answer Text' input field below it, and 'Question 3' with a plus sign. A vertical scrollbar is on the right side of the content area.

2.3.4.3 Coordinator Interfaces

Main Page

Regarding options are displayed when To Do's or Done's chosen to be flagged, deleted or marked as done.

The screenshot shows the 'Main Page' interface. On the left is a 'Menu' sidebar with buttons for Notifications, Student List, Files, Messages, Forum, FAQ, and Profile. The main content area has a title 'Main Page' in red. Below the title are two panels: 'To Do' and 'Done'. The 'To Do' panel contains a list of three items, each with a red flag icon and a checkbox. The 'Done' panel contains a list of three items, each with a blue circle icon and a checkbox. Below these panels is an 'Add To Do' button. A vertical scrollbar is on the right side of the content area.

Student List

Student names have two links, one directing to the message interface and the other to the profile. The coordinator link directs to the message interface only.

The screenshot shows the 'Student List' page. On the left is a 'Menu' with buttons for Notifications, Student List (highlighted), Files, Messages, Forum, FAQ, and Profile. The main area has a title 'Student List' in red. Below it is a table with columns: Name, Status, University, and Coordinator. The 'Name' column has a search icon and a link icon. The 'Coordinator' column also has a link icon. The table is currently empty.

Name	Status	University	Coordinator
------	--------	------------	-------------

Messages

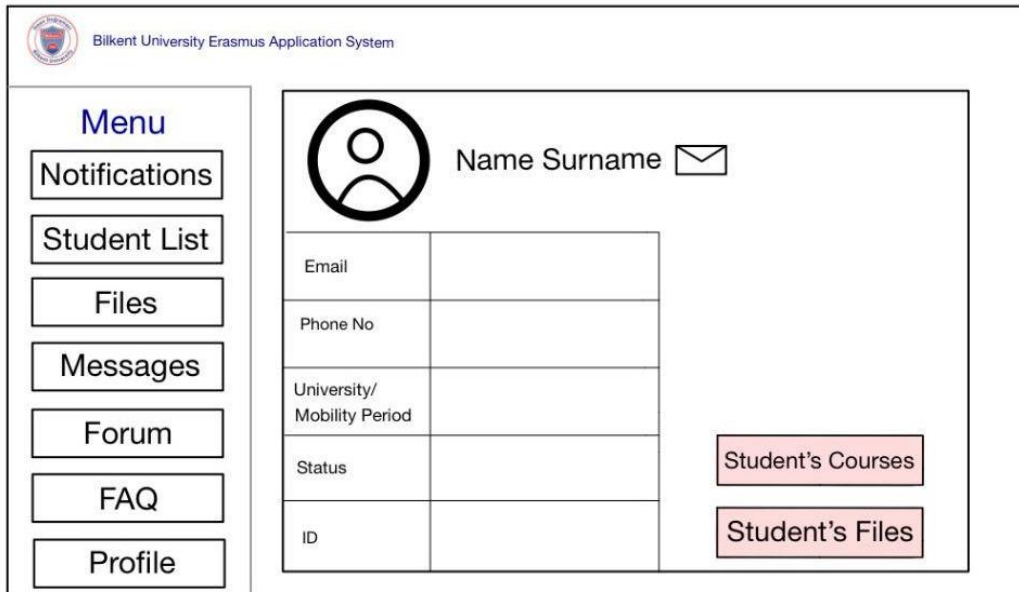
Coordinators can see every information submitted on a student profile in messaging view.

The screenshot shows the 'Messages' page. On the left is a 'Menu' with buttons for Notifications, Student List, Files, Messages (highlighted), Forum, FAQ, and Profile. The main area has a title 'Messages' in red. Below it is a list of messages on the left, each with a profile icon and a 'Name Surname' label. To the right of the list is a detailed view of a message, showing a profile icon and a table with fields: Name Surname, University, Phone No, Status, Email, and Mobility Period. Below the table is a message body with a pink bubble on the left and a green bubble on the right. At the bottom is a text input field and a green send button.

Name Surname	University
Name Surname	Status
Name Surname	Mobility Period

Viewing Student's Profile

Courses waiting to be approved and all courses taken by the student can be seen in student's courses

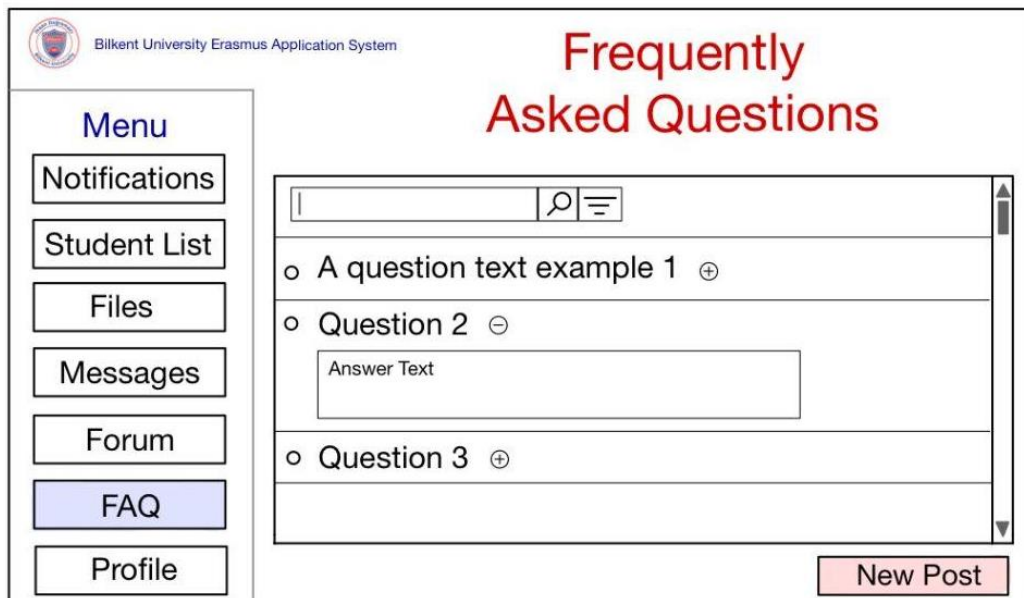


The screenshot shows the Bilkent University Erasmus Application System interface. On the left is a 'Menu' with buttons for Notifications, Student List, Files, Messages, Forum, FAQ, and Profile. The main area displays a student's profile with a circular icon, the text 'Name Surname' with an email icon, and a table with fields for Email, Phone No, University/Mobility Period, Status, and ID. To the right of the table are two buttons: 'Student's Courses' and 'Student's Files'.

Email	
Phone No	
University/ Mobility Period	
Status	
ID	

Frequently Asked Questions

A coordinator can remove a frequently asked question by marking the circles and choosing to delete.



The screenshot shows the Bilkent University Erasmus Application System interface for the 'Frequently Asked Questions' section. The 'FAQ' button in the left menu is highlighted. The main area has a title 'Frequently Asked Questions' and a search bar. Below the search bar is a list of questions, each with a circle icon and a plus/minus sign. The second question, 'Question 2', is expanded, showing an 'Answer Text' input field. A 'New Post' button is located at the bottom right.

Frequently Asked Questions

○ A question text example 1 +


○ Question 2 -

Answer Text

○ Question 3 +

New Post

Posting a Frequently Asked Question

 Bilkent University Erasmus Application System

Frequently Asked Questions

Question:

Post

Menu

Notifications

Student List

Files

Messages


Forum

FAQ

Profile

Course Approval

Coordinators will be able to see submitted courses, course lists or merged courses and can reject or approve them along with an optional comment to be received by the student.

 Bilkent University Erasmus Application System

Course Information

Course Name	Course ID	Ects	Taken Instead of	Course Info Link

Approve

Reject

Menu

Notifications

Student List




Files

Messages

Forum

FAQ

Profile

 Name Surname  

3 References

- [1] Object-Oriented Software Engineering, Using UML, Patterns, and Java, 2nd Edition, by Bernd Bruegge and Allen H. Dutoit, Prentice-Hall, 2004, ISBN: 0-13-047110-0.
- [2] “Use Case Diagram Notations Guide - Visual Paradigm.” Visual Paradigm Community Circle, May 11, 2018. <https://circle.visual-paradigm.com/docs/uml-and-sysml/use-case-diagram/use-case-diagram-notations-guide/>.
- [3] Cysneiros, Luiz M. and Eric Yu. "Non-functional requirements elicitation." Perspectives on software (n.d.): 2004.