Bilkent University
Department of Computer Engineering

CS319 Object-Oriented Software Engineering

BILTERN: Summer Training Course Management Application

Project Analysis Report

- Cenk Merih Olcay 22002408
- Arshia Bakhshayesh 22001468
- Enes Bektaş 22002401
- Faruk Uçgun 22003016
- Faaiz Khan 22001476

Instructor: Eray Tüzün
Teaching Assistant(s): Muhammad Umair Ahmed and Yahya Elnouby

Contents

Requirements Report

BILTERN: Summer Training Course Management

1 Introduction

Every student at Bilkent University is obligated to perform two summer trainings as part of their student program. This course should not only be seen as a requirement towards a degree, though, as its inherent value goes far beyond that. These trainings prepare students for their professional life after they graduate from Bilkent. Along the way, these students gain precious information, not only related to their fields, but also related to professionalism.

As such, it is vital that every piece of this puzzle can do their respective jobs easily and efficiently. Our project aims to do exactly that, providing features that would help everyone.

Students can upload and re-upload reports without any hassle, and make the necessary changes thanks to our built-in notification and feedback system. Teaching assistants can perform surface-level evaluation and provide feedback with regards to formatting, etc. Graders/Instructors can use the platform to read and review reports as well as provide feedback on the reports with everything that needs changing, and ultimately grading them. Coordinators can manage the graders easily as well as look at the statistics of the course and monitor its progress as a whole. The secretary can manage the course and its variables directly through the platform itself.

Our project aims to include the following features that contain many needed quality of life improvements:

- Report uploading and viewing
- Report feedbacks
- Requesting report reuploads/resubmissions from the company/student
- Notification system to alert each member to whatever they need to know
- Access to previous feedbacks

2 Current System

As a team, we attended the information session held by Department Chair Prof. Selim Aksoy, a stakeholder in this project. Selim Hoca gave us a presentation describing the current system in that session. The current system is a composition of Moodle, Email, and Google Drive. Moodle is the platform used for report upload and storage. Information traffic is mainly

maintained through email. And the drive is used for providing previews and feedback.

This combination of three platforms already makes the course process very cumbersome for any role in the system.

- The Moodle page for the course is opened using the information extracted from the Stars registration system. The Moodle page is managed by Instructors to open assignment and announcement folders.
- Most communication related to the course for notifications, updates, and other information requires a separate email service. It is highly ineffective due to a lack of automation.
- Plagiarism for the report is checked in Moodle through the integrated Turnitin system.
- TAs pre-evaluate reports by downloading reports from Moodle before the Instructor Evaluation.
- During the evaluation, the Instructor fills in all the information through a pdf or a word program. Evaluation forms from companies come inside a sealed envelope and are given to instructors. And Part A is filled manually by an Instructor according to the information inside this envelope. An instructor may need to fill in completed parts of the form (most possibly Part A) repeatedly for the iterations of Part B.
- Automation for generating feedback, forms, and extracting course summaries and statistics is unavailable. Feedback is provided through Google Drive; an Undergraduate has to view feedback from a different platform.
- To track report information, such as how many iterations it has been, and to access previous iterations/feedback requires a web surf through emails and Moodle course pages for graders or Students to find Google Drive links unique for every iteration.
- The system lacks the functionality to provide course summaries and statistics for reports, Undergraduate success/fail rates, completion of report stages, and collection of ABET information for courses that might be necessary for accreditation programs.
- Undergrads can't be aware of any updates on their reports' status (grade finalized or another iteration is required) unless they receive an email from their instructors.
- For different departments or even courses, procedures to achieve course goals may deviate from each other. Standardization for EEE, CS, IE, and ME Departments is also missing.

3 Proposed System

3.1 Non-functional Requirements

3.2.1 Usability

- The project is designed to improve the current internship report system. Hence, the system has a user-friendly interface that leaves no place for confusion.
- Each user has dashboards for easy access to frequently used services and information.
- Using the student list page, it's easy to access any student, reports related to them, stages and actions that can be taken at those stages.
- Course coordinators, Secretaries can view course statistics.
- The reports can be selected and downloaded altogether for ease of use.
- Reports and their feedback can be viewed on the browser.
- Different colors are widely used to indicate similar and different functionalities.
- Only web application will be available as stated by the course instructor and there won't be a mobile application.

3.2.2 Safety

- The system only includes email and id information related to a user
- Students will not be able to see the information related to other students, they will only know their grader and teaching assistant.
- In the system, user passwords will be encrypted and stored that way.
- Students will enter the system after the secretary creates accounts for them, so students not taking XX-299 or XX-399 courses will not be able to enter the system.

3.2.3 Maintainability

- Since the program will be object oriented, it will be easier to make changes to the existing code.
- Github is used to create new branches when testing new functionalities so that they don't affect the main code.
- Class structure will be formed in a way that line number is reduced.
- Adding comments to the code will also make it more readable.
- We will make use of polymorphism, encapsulation to reduce coupling and increase reusability.

3.2.4 Performance

- In order not to stall the users, the reaction time for any action taken will be at most 1.5 seconds.
- The system support up to 1000 users at a time.
- Low bandwidth will not prevent users from taking actions, it might slow the process down.

3.2 Pseudo Requirements

1. The programming language chosen for the backend side must support Object Oriented Programming (OOP).
2. Project must be implemented as a web based application.
3. Source code of the project should be trackable on Github.

3.3 System Models

3.4.1 Use-case Model

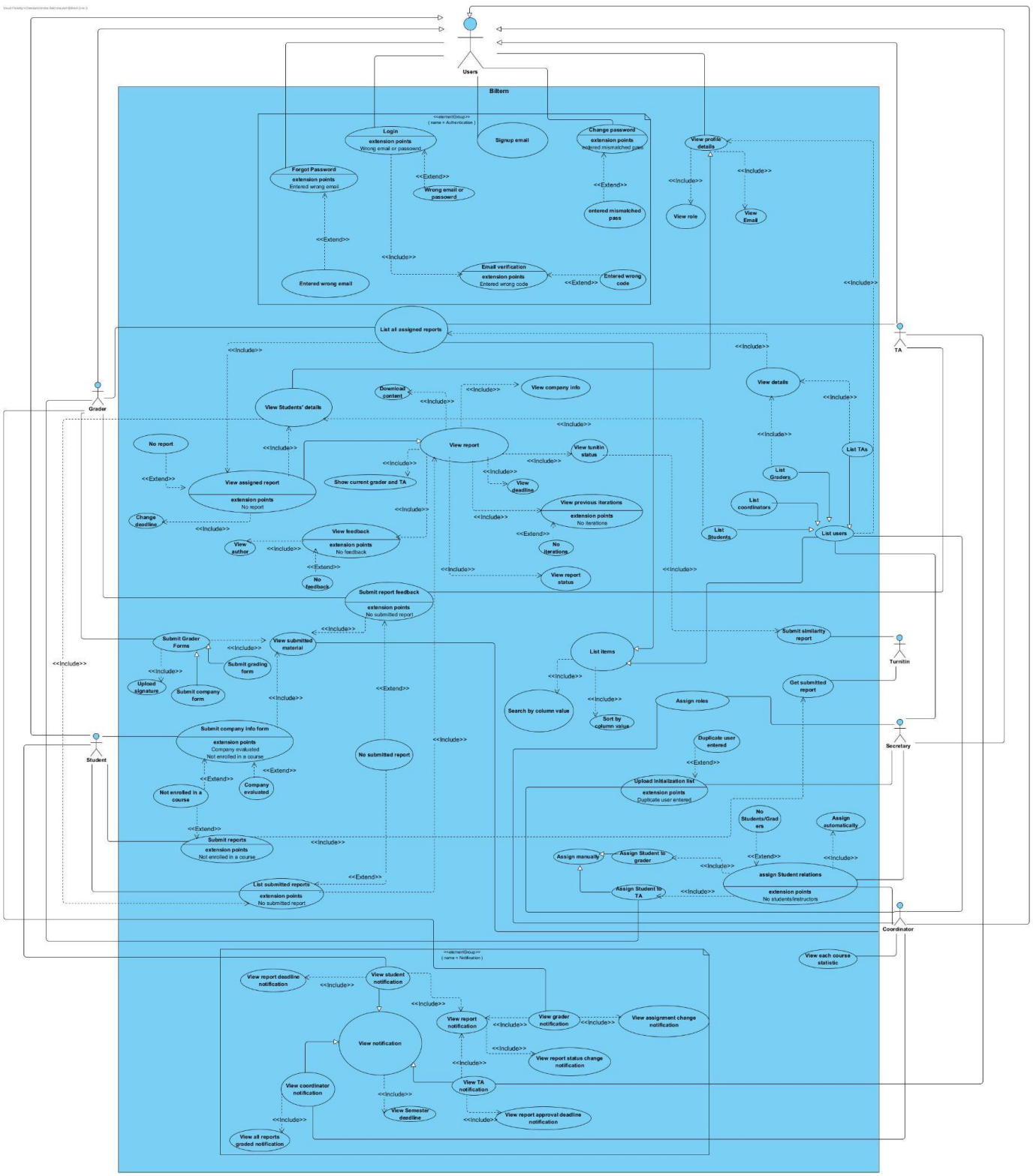

Fig. 1: Biltern Use-case Diagram

For better quality: https://drive.google.com/file/d/15mN1GCfoi6Oafv8fm50PdTnj1NSAGK7I/view?usp=sharing

1. Login

1. Name: Login
2. Participating Actor: All users
3. Entry Condition: Opening the app through a web browser.
4. Exit Condition: The user logs in or the user exits the page
5. Flow of Events:
    5.1. The user decides to log in.
    5.2. The user fills in the email and password bars on the login page.
    5.3 If the user enters the password or email incorrectly or the account does not exist
        5.3.1 "Invalid Email or Password" error message is displayed.
    5.4 Else the user logs in successfully.
        5.5.1 A verification mail containing a code is sent to the entered email address

2. Wrong email or password

1. Name: Wrong email or password
2. Participating Actor: All users
3. Entry Condition: Extends login. It is invoked whenever the email and password do not match or the email does not exist in the system.
4. Exit Condition: Logging in
5. Flow of Events:
    5.1. Wrong credentials are put in
    5.2. The user sees an error message when entering the state

3. Email verification

1. Name: Email verification
2. Participating Actor: All users
3. Entry Condition: After known login credentials are entered and the login button is clicked
4. Exit Condition: Entering the correct code or going back to the login page
5. Flow of Events:
    5.1. Correct credentials are put in
    5.2. User is prompted to enter the verification code sent to their email

5.3 user clicks the submit code button
5.3 If the code is wrong
5.3.1 Code error use case is entered
5.4 If the code is correct
5.4.1 User logs in to their account

4. Code error

1. Name: Code error
2. Participating Actor: All users
3. Entry Condition: extends login. Initiated when wrong code is entered
4. Exit Condition: Entering the wrong code or going back to the login page
5. Flow of Events:
5.1. Wrong code is entered
5.2. A wrong code error message is displayed
5.3 Resend code button is displayed

5. Forgot password

1. Name: Forgot password
2. Participating Actor: All users
3. Entry Condition: Clicking on the "Forgot password" button on the login page
4. Exit Condition: Changing password or clicking on "Return to login page"
5. Flow of Events:
5.1. Forgot password button is clicked
5.2. The user fills in the email field
5.3 If the entered email is not from Bilkent
5.3.1 "Invalid Bilkent email" error message is displayed.
5.4 Else a change password mail is sent to the entered email
5.6 The user returns to the login page

6. Non-existing email

1. Name: Non-existing email

2. Participating Actor: All users
3. Entry Condition: Extends forgot password. Initiated whenever an unknown email is entered
4. Exit Condition: Entering an existing email
5. Flow of Events:
    5.1. Unknown email is put in
    5.2. The user sees an error message regarding the unknown email

7. Change password

1. Name: Change password
2. Participating Actor: All users
3. Entry Condition: Change password button is clicked in the settings tab when logged in
4. Exit Condition: Choosing another tab or signing out
5. Flow of Events:
    5.1. User clicks on the change password button
    5.2. User is prompted to enter the current password and the new password and repeat the new password.
    5.3. If the current password is wrong
        5.3.1. Mismatch pass extended case is entered
    5.4. Else if the new password and new repeated password fields do not match
        5.4.1 User is prompted to match the passwords
    5.5. Else the user clicks on the change password button
    5.6. The password of that user account is changed

8. View profile details

1. Name: View profile details
2. Participating Actor: All users
3. Entry Condition: Click on the view account detail tab
4. Exit Condition: Clicking on any other tab or logging out
5. Flow of Events:
    5.1. User clicks on the account detail tab
    5.2. User views account email and role

9. List all assigned reports

1. Name: List all assigned reports
2. Participating Actor: Instructor, TA, Coordinator, Secretary
3. Entry Conditions:

3.1 Instructor and TA click on the view reports tab

3.2 Coordinator and Secretary click on the TA or instructor profiles from another list

4. Exit Condition: Clicking on any other tab or clicking on go to the previous page button

5. Flow of Events:

5.1. Lists all the reports of a TA/grader

5.2. Will show nothing if the specified owner does not have any reports assigned

5.3. Each report is clickable

5.4. **Inherits** all of the Luist items use case capabilities

10. View assigned report

1. Name: View assigned report
2. Participating Actor: Instructor, TA, Coordinator, Secretary
3. Entry Condition: Clicked on a report
4. Exit Condition: Clicking on go to previous page button or clicking on any other tab
5. Flow of Events:

5.1. User clicks on a report from a list

5.2. User can view the corresponding student's details

5.3. **Inherits** the same behavior from the View report

11. Change deadline

1. Name: Change deadline
2. Participating Actor: Instructor, TA, Coordinator, Secretary
3. Entry Condition: Clicked on set deadline on the view report page
4. Exit Condition: Closing the select deadline pop-up or setting a new deadline
5. Flow of Events:

5.1. User clicks on the Change deadline button

5.2. User is prompted to choose a date using a pop-up

5.3. User then clicks the confirm button

5.4. The deadline is changed and the user is brought back to the View Report page

12. View student details

1. Name: View student details
2. Participating Actor: Instructor, TA, Coordinator, Secretary
3. Entry Condition:

3.1. Clicked on a student name from a list

3.2. Clicked on the view student details button from the view report page

4. Exit Condition: Clicking on go to previous page button or clicking on any other tab

5. Flow of Events:

5.1. **Inherits** the same behavior from View profile info

5.2. Lists all student's reports

13. View report

1. Name: View assigned report
2. Participating Actor: All users
3. Entry Condition: Clicked on a report
4. Exit Condition: Clicking on go to the previous page button or clicking on any other tab
5. Flow of Events:

5.1. User clicks on a report from a list

5.2 User views current grader and TA

5.3 User views the current deadline

5.4 User views the Turnitin status

5.5 User can download the report

14. View feedback

1. Name: View Feedback
2. Participating Actor: All users
3. Entry Condition: Clicked on the view feedback button in the view report page
4. Exit Condition: Clicking on go to previous page button or clicking on any other tab
5. Flow of Events:

5.1. User clicks on the view feedback button

5.2 User views the feedback given to the report

5.3 User views the author of the feedback and the date of the feedback

5.4 User can download feedback

15. View previous iterations

1. Name: View previous iterations
2. Participating Actor: All users

3. Entry Condition: Clicked on previous iterations button
4. Exit Condition: Clicking on go to previous page button or clicking on any other tab
5. Flow of Events:
    5.1. User clicks on the previous iterations button
    5.2 User views a list of previous report versions that have received feedback
    5.3 User views the date of upload and feedback and author of the feedback

16. View company info

1. Name: View company info
2. Participating Actor: All users
3. Entry Condition: Clicked on company info button in the view report page
4. Exit Condition: Clicking on go to previous page button or clicking on any other tab
5. Flow of Events:
    5.1. User clicks on the company info button
    5.2 User views the company representative's email, full name, and company name

17. Submit instructor forms

1. Name: Submit instructor forms
2. Participating Actor: Grader
3. Entry Condition: Clicking on grade button on the view report page
4. Exit Condition: Clicking on go to previous page button or clicking on any other tab or submitting the form
5. Flow of Events:
    5.1. User clicks on the Grade button
    5.2 User fills in the grading form
    5.3 User can upload a new signature or use the previously used one
    5.4 User submits the grading form
    5.5 User can download the submitted form
    5.6 User returns to the view report page

18. Submit feedback

1. Name: Submit feedback
2. Participating Actor: Grader, TA

3. Entry Condition: Clicking on provide feedback button on the view assigned reports page
4. Exit Condition: Clicking on go to previous page button or clicking on any other tab or submitting a feedback
5. Flow of Events:
    5.1. User clicks on the provide feedback button
    5.2 If the corresponding report has not been uploaded
        5.2.1 the button is not clickable
    5.3 Else the user is prompted to upload a pdf or word file
    5.4 User selects the file from the pop up menu and uploads it
    5.5 User submits the feedback by clicking submit
    5.6 User is returned to the view submitted reports page

19. View submitted material

1. Name: View submitted material
2. Participating Actor: Grader, Undergraduate
3. Entry Condition: Clicking on Submitted material button on the main page
4. Exit Condition: Clicking on go to previous page button or clicking on any other tab
5. Flow of Events:
    5.1. User clicks on the submitted material tab
    5.2 User views a list of all submitted material
    5.3 User can view each submitted material and download

20. Submit Company info form

1. Name: Submit company info form
2. Participating Actor: Undergraduate
3. Entry Condition: Clicking on submit company info form on the view report page
4. Exit Condition: Clicking on go to previous page button or clicking on any other tab
5. Flow of Events:
    5.1. User clicks on the submit company form button
    5.2 If the state of the report has passed "awaiting company evaluation"
        5.2.1 The button is not clickable

    5.3 Else user fills in the company info form (full name, email, company name, address)
    5.4 User clicks on submit button and the form is submitted

5.5 User returns to the view report page

21. Submit reports

1. Name: Submit reports
2. Participating Actor: Undergraduate
3. Entry Condition: Clicking on the submit report button in the view report page
4. Exit Condition: Clicking on go to previous page button or clicking on any other tab or clicking on the submit button
5. Flow of Events:
    - 5.1 If the user is not enrolled in a course
        - 5.1.1 The view report page will not be accessible through the report list as it will be empty
    - 5.1. Else User clicks on the submit reports button
    - 5.2 User uploads the report in pdf of word format
    - 5.3 User clicks on submit and the report is submitted
    - 5.4 The user is returned to the view report page

22. List submitted reports

1. Name: List submitted reports
2. Participating Actor: Undergraduate
3. Entry Condition: Clicking on the reports tab of the main page
4. Exit Condition: Clicking on go to previous page button or clicking on any other tab
5. Flow of Events:
    - 5.1. User clicks on the reports tab
    - 5.2 if they are not enrolled in a course
        - 5.2.1 an empty list is shown
    - 5.2 Else user views a list of their reports

23. View details

1. Name: View details
2. Participating Actor: Secretary, Department Coordinator
3. Entry Condition: Clicking on a Grader or a TA from a list
4. Exit Condition: Clicking on go to previous page button or clicking on any other tab
5. Flow of Events:
    - 5.1. User clicks on the TA item or grader item
    - 5.2 User lists the assigned reports to the TA/Grader

24. List TAs / List graders

1. Name: List TAs/Graders
2. Participating Actor: **Inherits** from list users
3. Entry Condition: Clicking on the list of users tab
4. Exit Condition: Clicking on go to previous page button or clicking on any other tab
5. Flow of Events:
    5.1. **Inherits** behaviors from list users
    5.2 User can view each TA and grader account details by clicking on it

25. List coordinators

1. Name: List TAs/Graders
2. Participating Actor: **Inherits** from list users
3. Entry Condition: Clicking on the list of users tab and choosing the coordinator users tab
4. Exit Condition: Clicking on go to previous page button or clicking on any other tab
5. Flow of Events:
    5.1. **Inherits** behaviors from list users
    5.2 User views a list of coordinators

26. List students

1. Name: List students
2. Participating Actor: **Inherits** from list users
3. Entry Condition: Clicking on the list of users tab and choosing the student users tab
4. Exit Condition: Clicking on go to previous page button or clicking on any other tab
5. Flow of Events:
    5.1. **Inherits** behaviors from list users
    5.2 User views a list of students
    5.3 User can view each student's account detail by clicking on it

27. List user

1. Name: List users
2. Participating Actor: Coordinator, Secretary
3. Entry Condition: Clicking on the List of users tab

4. Exit Condition: Clicking on go to previous page button or clicking on any other tab
5. Flow of Events:
    5.1. **Inherits** behaviors from List items
    5.2 User can view the account detail of each account listed by clicking on it

28. List items

1. Name: Lister
2. Participating Actor: all users
3. Entry Condition: viewing a list
4. Exit Condition: Clicking on go to previous page button or clicking on any other tab
5. Flow of Events:
    5.1. User can search by column value
    5.2 User can sort the list by column value

29. Assign roles

1. Name: Assign roles
2. Participating Actor: Coordinator, Secretary
3. Entry Condition: Clicking on the assign roles tab
4. Exit Condition: Clicking on go to the previous page button or clicking on any other tab
5. Flow of Events:
    5.1. User enters a user's email
    5.2 User chooses a role to assign to that email
    5.3 User clicks on assign role button
    5.4 A success message is shown
    5.5 Role is assigned

30. Upload initialization list

1. Name: Upload initialization list
2. Participating Actor: Coordinator, Secretary
3. Entry Condition: click on account initialization tab
4. Exit Condition: Clicking on go to previous page button or clicking on any other tab
5. Flow of Events:
    5.1. User clicks on upload

5.2 User chooses an excel file from their own device through a pop-up menu

5.3 User clicks on submit

5.4 a success message is shown

5.5 accounts are initialized

31. Manual assignment

1. Name: Manual assignment
2. Participating Actor: Coordinator, Secretary
3. Entry Condition: click on assign relations tab
4. Exit Condition: Clicking on go to previous page button or clicking on any other tab
5. Flow of Events:

5.1. User clicks on assign student relations tab

5.2 User lists all TAs, Graders, and instructors using list users

5.3 User chooses whether to assign to TA or to assign to Grader or visa versa

5.4 User clicks on the assign button

5.5 a success message is shown

5.6 accounts are assigned

32. Automatic assignment

1. Name: automatic assignment
2. Participating Actor: Coordinator, Secretary
3. Entry Condition: click on assign relations tab
4. Exit Condition: Clicking on go to previous page button or clicking on any other tab
5. Flow of Events:

5.1. User clicks on assign student relations tab

5.2 User clicks on automatic assignment button

5.3 User is prompted whether they are sure about this decision

5.4 User clicks yes

5.5 all students are assigned equally to the TAs and Graders

5.5 a success message is shown

33. View each course statistic

1. Name: automatic assignment
2. Participating Actor: Coordinator
3. Entry Condition: click on statistics tab

4. Exit Condition: Clicking on go to previous page button or clicking on any other tab
5. Flow of Events:
    5.1 User views the statistics of each course they are in charge of categorized by the status of reports

34. View notification

1. Name: View Notification
2. Participating Actor: All users
3. Entry Condition: Click on the notifications tab
4. Exit Condition: Clicking on the close tab button or clicking on any other tab
5. Flow of Events:
    5.1 User sees personalized notification for the reports and the users they are in charge of handling.
    5.2 User views important deadline notifications which are related to them

3.4.2 Object and Class Model

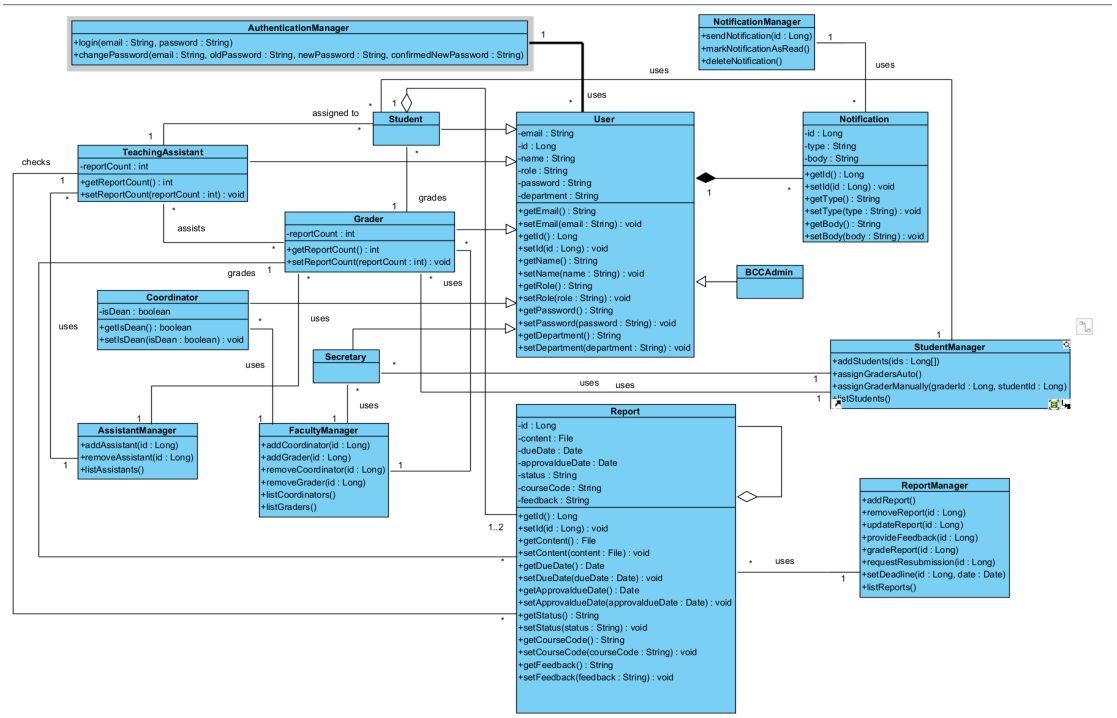

Fig. 2: Biltern Class Diagram

User: User is an abstract superclass, containing the 5 different types of users of Biltern. The class contains common information about all the users, specifically their names, emails, roles, departments, and passwords. The user class is inherited by 5 other subclasses: Undergraduate, TeachingAssistant, Grader, Coordinator, and Secretary. Each user has access to general capabilities, like logging in, changing their password, and managing their notifications.

Student: Student is a subclass that inherits the User class. An object of this class will have all the information present in the user class, as well as arrays for company and student reports, an assigned teaching assistant, and an assigned grader. Apart from the general capabilities, an object of this class will also be able to manage their reports: adding, updating, removing, etc.

TeachingAssistant: TeachingAssistant is a subclass that inherits the User class. An object of this class will have all the information present in the user class, as well as arrays for company and student reports that are assigned to be reviewed by them, an assigned grader, and an array of assigned Students. Apart from the general capabilities, an object of this class will also be able to manage reports: providing feedback, requesting resubmission, etc.

Grader: Grader is a subclass that inherits the User class. An object of this class will have all the information present in the user class, as well as arrays for company and student reports that are assigned to be reviewed by them, arrays of assigned teaching assistants and Students, and a file containing their electronic signature. Apart from the general capabilities, an object of this class will also be able to manage reports: providing feedback, requesting resubmission, grading reports, etc. Graders are also able to manage their teaching assistants: adding, removing, etc.

Coordinator: Coordinator is a subclass that inherits the User class. An object of this class will have all the information present in the user class, as well as a simple boolean attribute to differentiate the dean from the other coordinators. Apart from the general capabilities, an object of this class will also be able to manage faculty: adding and removing graders, and manage students: assigning graders to them automatically, and manually, as well as adding students to the course or removing them from it.

Secretary: Secretary is a subclass that inherits the User class. An object of this class will have all the information present in the user class, but nothing more as no more information is required from the secretary. Apart from the general capabilities, an object of this class will also be able to manage faculty:

adding and removing graders/coordinators, and manage students: assigning graders to them automatically, and manually, as well as adding students to the course or removing them from it.

Report: Report is an abstract superclass, containing the 2 different types of reports used for summer trainings. The class contains common information about all the reports, specifically the emails of the students that submit them, the corresponding course for each report, the last date of submission for the report, and an array of feedbacks from teaching assistants and/or graders. The report class is inherited by 2 other subclasses: CompanyReport and StudentReport. Each Student must submit both of these reports to be eligible for grading. These reports can be added, removed, updated, graded, and so on by the users of Biltern.

AuthenticationManager: Users use this class to login to Biltern. This class also provides a functionality to reset a user's password in case they forget it.

NotificationManager: This class allows users to manage their notifications. This includes sending them to other users, marking them as read, and deleting them.

ReportManager: This class allows Students, teaching assistants, and graders to view and manage reports. The aforementioned users are able to add, remove, update, grade, and list reports with the help of this class

AssistantManager: This class allows graders to manage their teaching assistants. Graders can add, remove, and list assistants using this class.

FacultyManager: This class allows coordinators and secretaries to add/remove graders/coordinators. This class also contains the functionality to list every single member of the faculty in a certain department.

StudentManager: This class provides coordinators and secretaries to add, and remove students to and from the course. Students can be added using an excel file containing all of their required attributes, after which, they are able to login. The aforementioned users may also use this class to list all students taking the course, as well as view statistics regarding the training.

3.4.3 Dynamic Models

3.4.3.1 State Diagrams

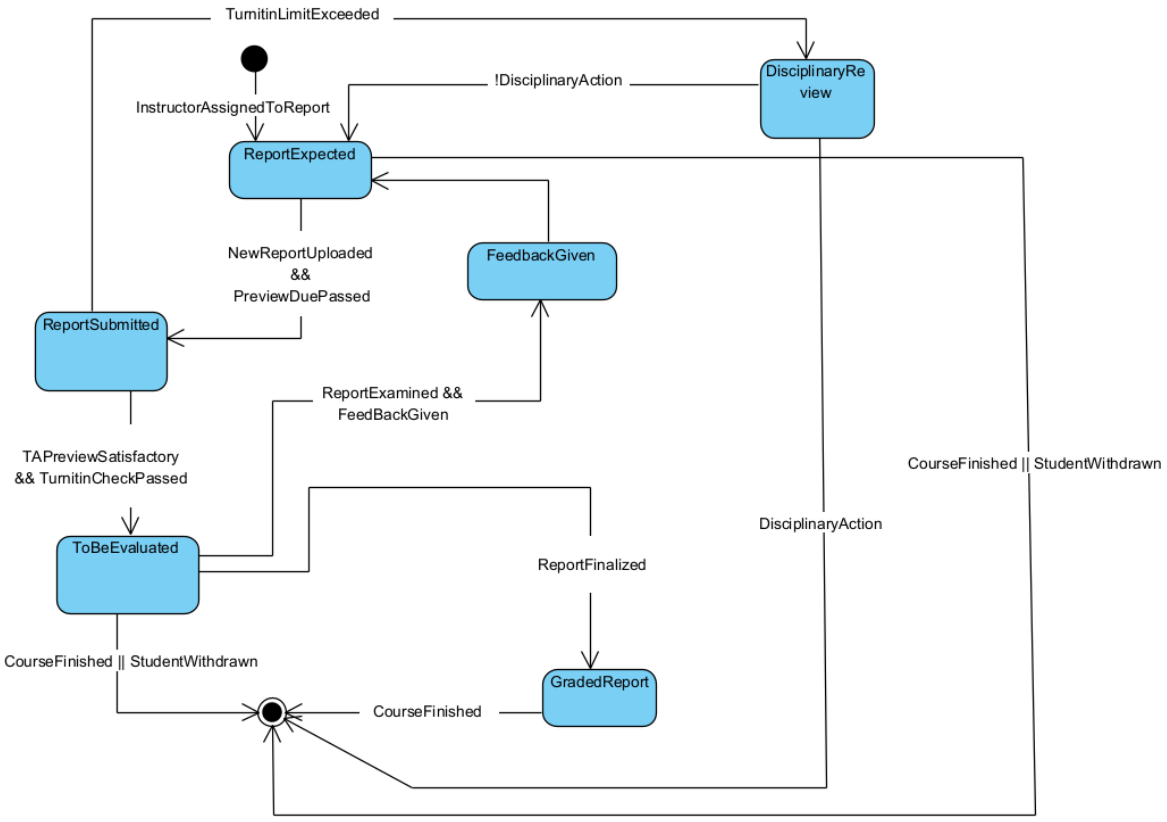

Fig. 3: Instructor’s Report Evaluation State Diagram

This state diagram shows the report evaluation event from an instructor's perspective. The Instructor gets notified when the semester starts, and a report is uploaded and passes the preview feedback stage. Suppose feedback is not given for a long time (more than the Department Coordinator's decision). In that case, the Department Coordinator gets notified, and they can decide to provide extra time for the feedback stage. Otherwise, if feedback is given on time, based on the report's status (if another iteration is demanded), the Instructor's role on the report can be completed or continued.

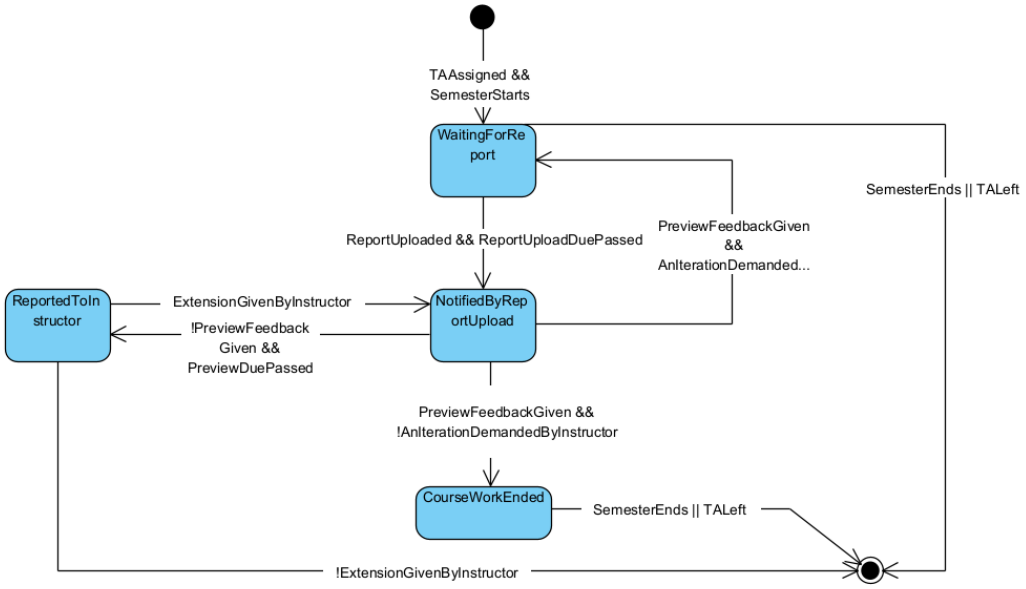

Fig. 4: TA Report Evaluation State Diagram

This state diagram shows the report preview stage for a TA. This diagram shows parallelism with the previous diagram. TA waits for the semester to begin, for Students to upload a report, and for the due date for report upload to pass. After that TA must provide preview feedback for the report till the deadline set by the Instructor. Otherwise, they get reported to the Instructor for that particular report preview, and the Instructor can decide whether to give extra time for the preview. After a successful preview within due dates, the TAs role can continue if another iteration is demanded for the report.

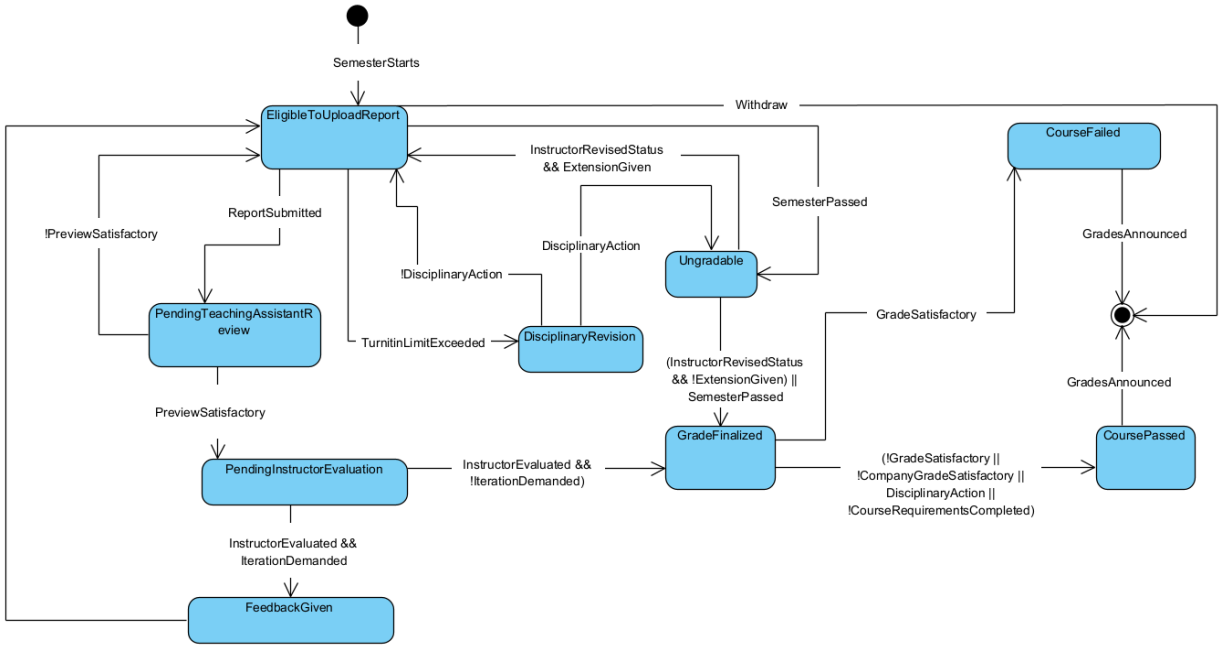

Fig. 5: Undergraduate Report Evaluation State Diagram

This state diagram explains the course status for an Student. When the Undergraduate is assigned to a course and the semester begins, the Undergraduate's status is pending a report upload. After the training report is uploaded, if the Turnitin check is over a certain limit, that report requires a preview from the Instructor for possible disciplinary action; otherwise Undergraduate should upload a new report resolving the Turnitin issues. After a report upload, the Undergraduate propagates to the next stage for preview and evaluation. If the due date is passed, it's up to the Instructor to penalize the Undergraduate or give a failing grade. Otherwise, again the Instructor can finalize the Undergraduate's grade or demand another iteration with feedback. In the case of an Undergraduate without a report uploaded at the end of the semester, the Undergraduate's failing grade is given by the Instructor.

3.4.3.2 Activity Diagrams

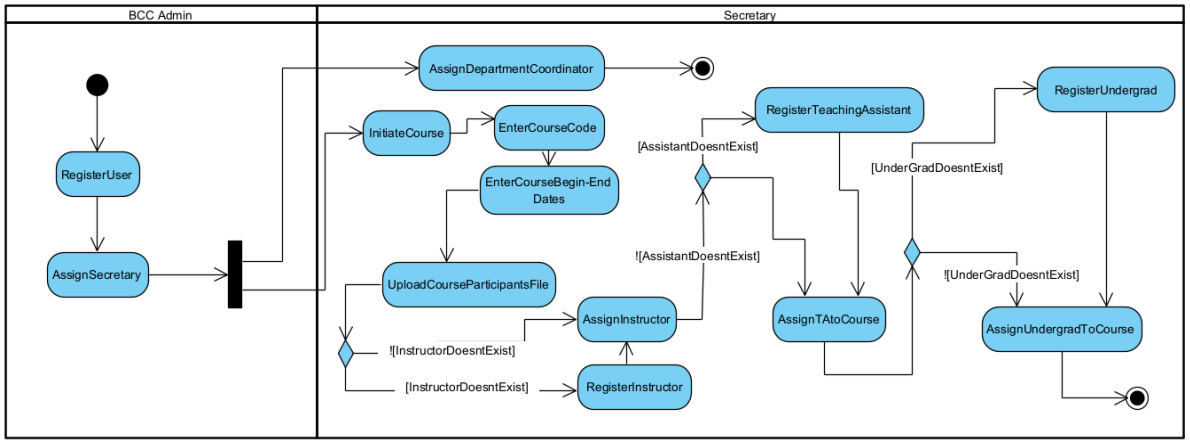

Fig. 6: Course Registration Activity Diagram

This activity diagram illustrates the event of initiation of a course from scratch. BCC Admin assigns the first secretary, and the secretary decides to initiate a course. Then the secretary will enter the necessary course details and upload a list of participants to register and assign users. If a user for any role TA, Instructor, or Undergraduate is missing, the system will first register that user to the system, notify the user through email, and then assign the user to that particular role.

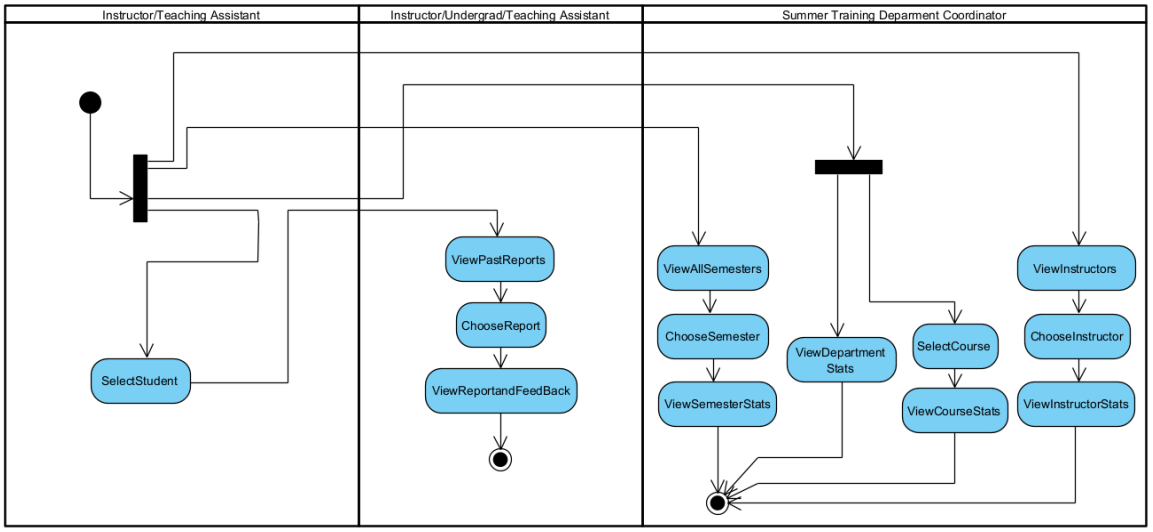

Fig. 7: Progress-Statistic View Activity Diagram

This activity diagram defines the capabilities of users for progress tracking. After selecting an Student, an instructor or TA can view the undergrad's past

reports and feedback, whereas Students can only see their past reports and feedback. Progress tracking works differently for a Department Coordinator; instead of a report, they can view ABET, grades, and other stats specific to a semester, ongoing or a completed course, or an instructor.

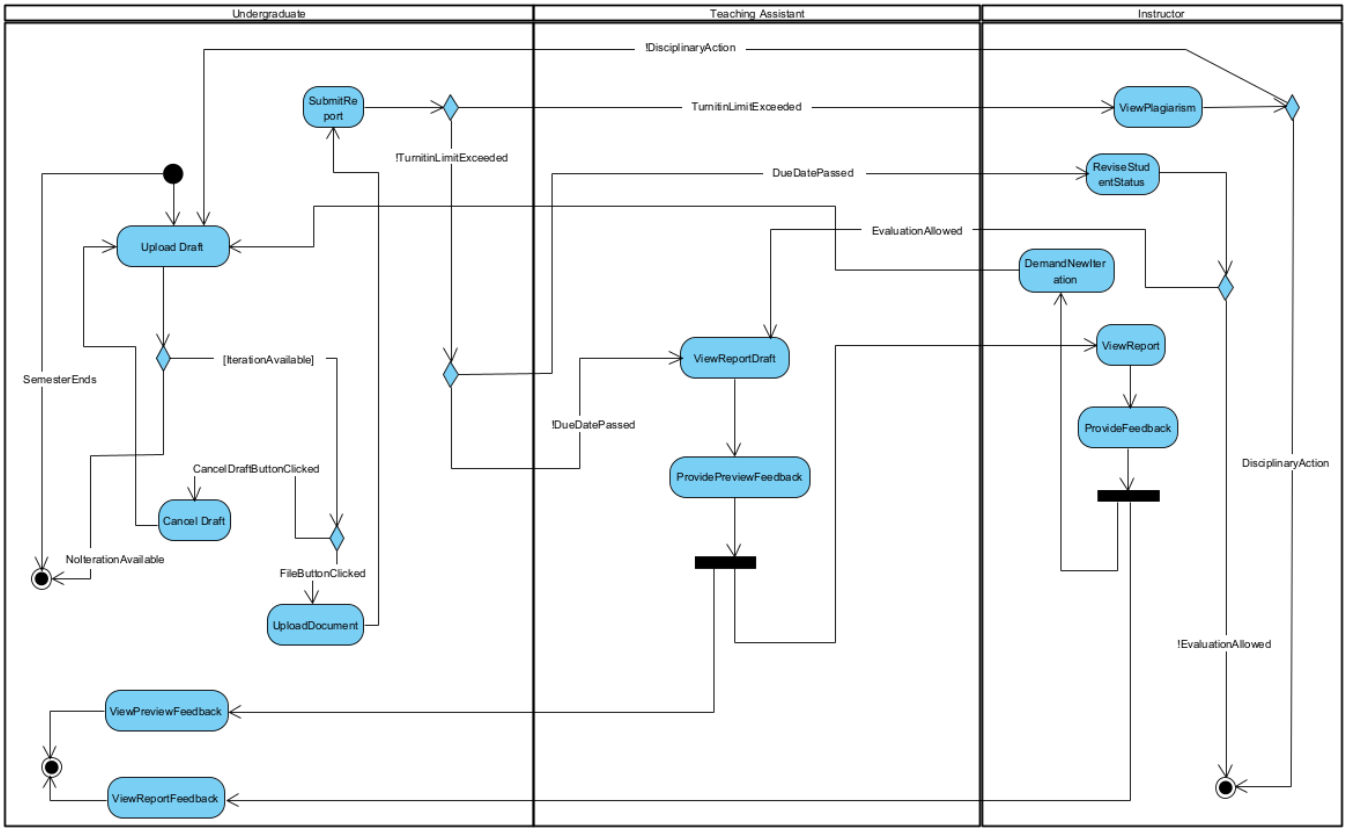

Fig. 8: Course Assignment Activity Diagram

This activity diagram shows the flow of a report evaluation event. It starts with the Undergraduate's report upload; the report can be uploaded anytime. However, if the due date is passed before the upload, the Instructor views the situation and decides on the evaluation. In cases when the file exceeds the Turnitin limit, Undergraduates are either expected to upload a new report or fail the course directly with disciplinary action based on Instructor's decision, so it does not proceed to evaluation. For a report to be evaluated, it first goes under TA preview and feedback. After that, the Undergraduate can view the preview feedback and report and move to the next stage for Instructor evaluation. If the Instructor decides to finalize the report, the event ends; otherwise, if the Instructor demands another iteration, the Undergraduate can view the Instructor's feedback and upload another report.

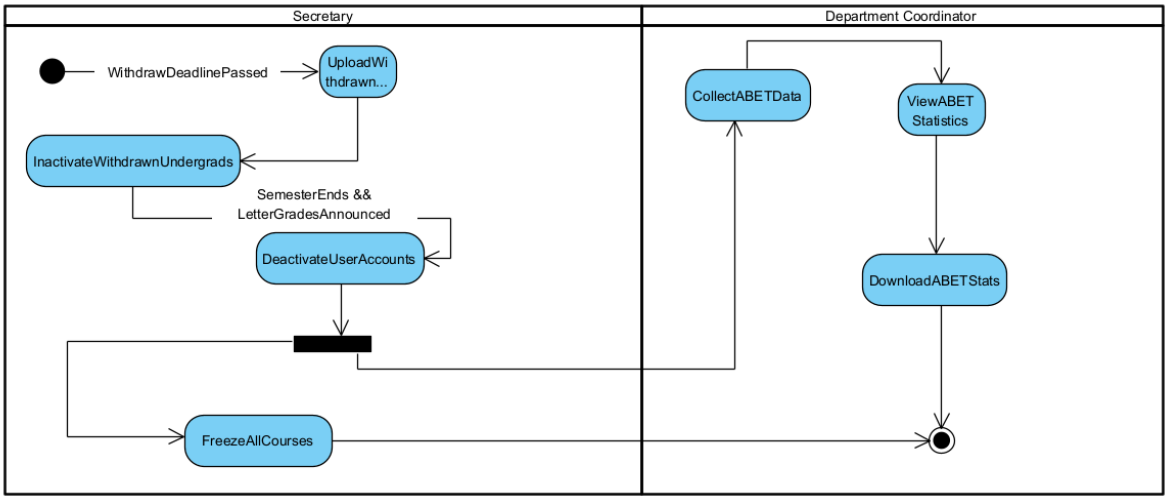

Fig. 9: Course Completion Activity Diagram

This activity diagram describes the event of course withdrawal and completion. After the semester begins and the withdrawal deadline is passed, the secretary uploads the list (Excel file with columns for Students' personal information and their course codes) of Students that have withdrawn from the courses. Then, the course dashboard becomes invisible for Students by accepting to make these accounts inactive for the courses they drop (since an Undergraduate can also take both summer training courses). Also, after the completion of the semester with letter grades being announced, the secretary inactive Undergraduate and Instructor accounts from further activity and collects ABET data necessary for the accreditation. It becomes visible to a Department Coordinator for their related department courses.

3.4.4 User Interface

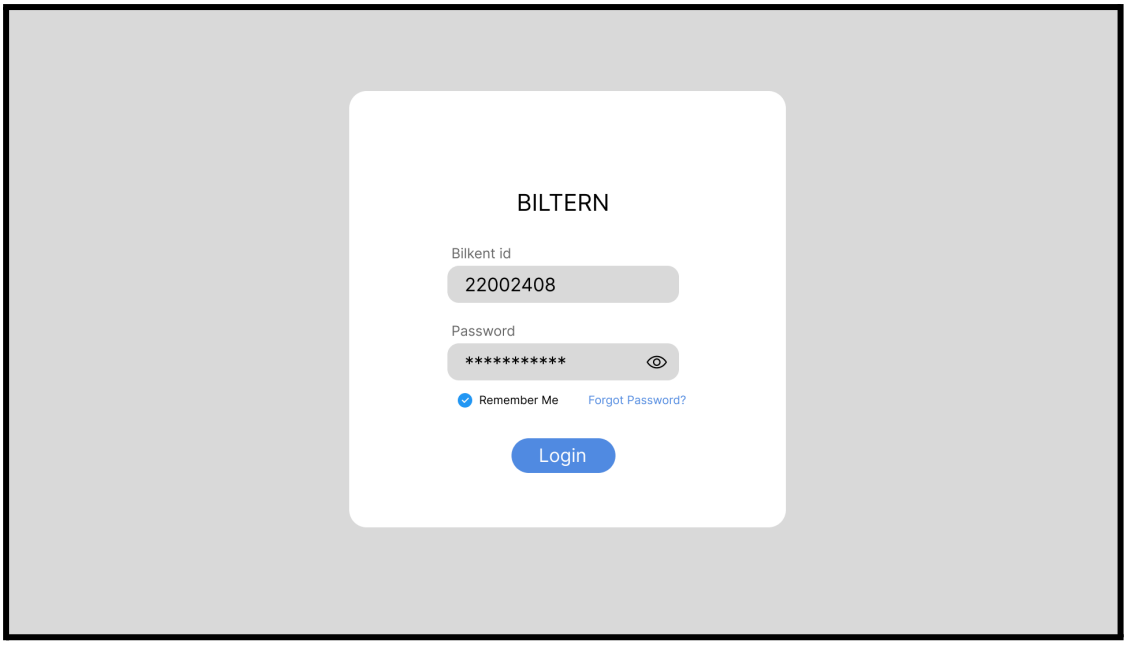

Fig. 10: Login Screen

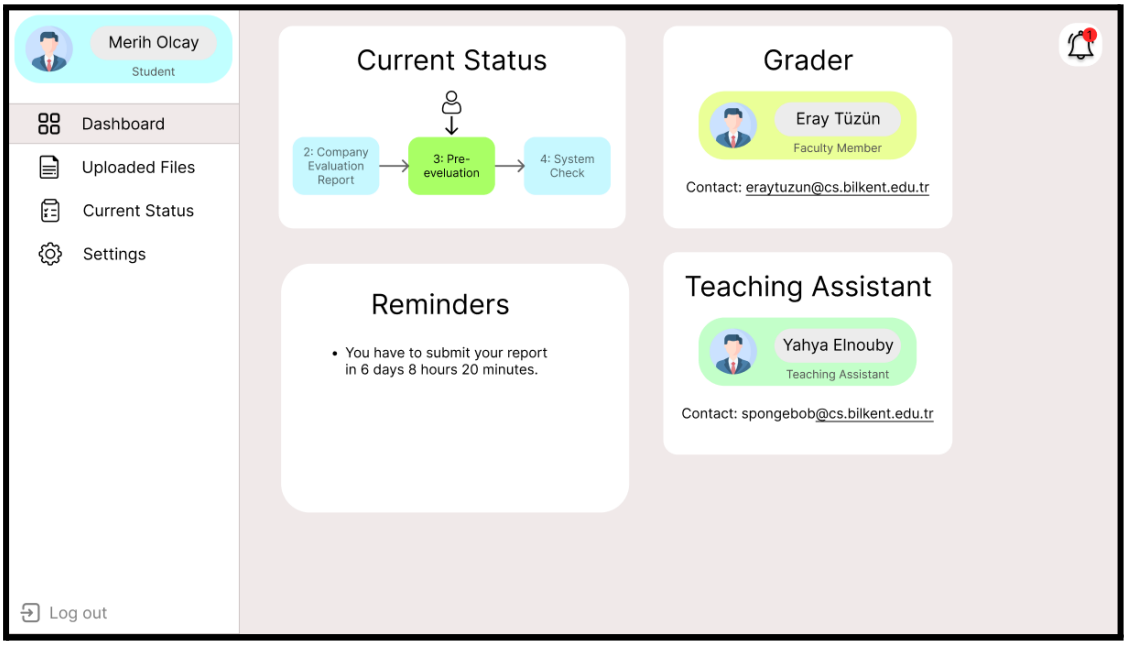

Fig. 11: Dashboard Page for Student

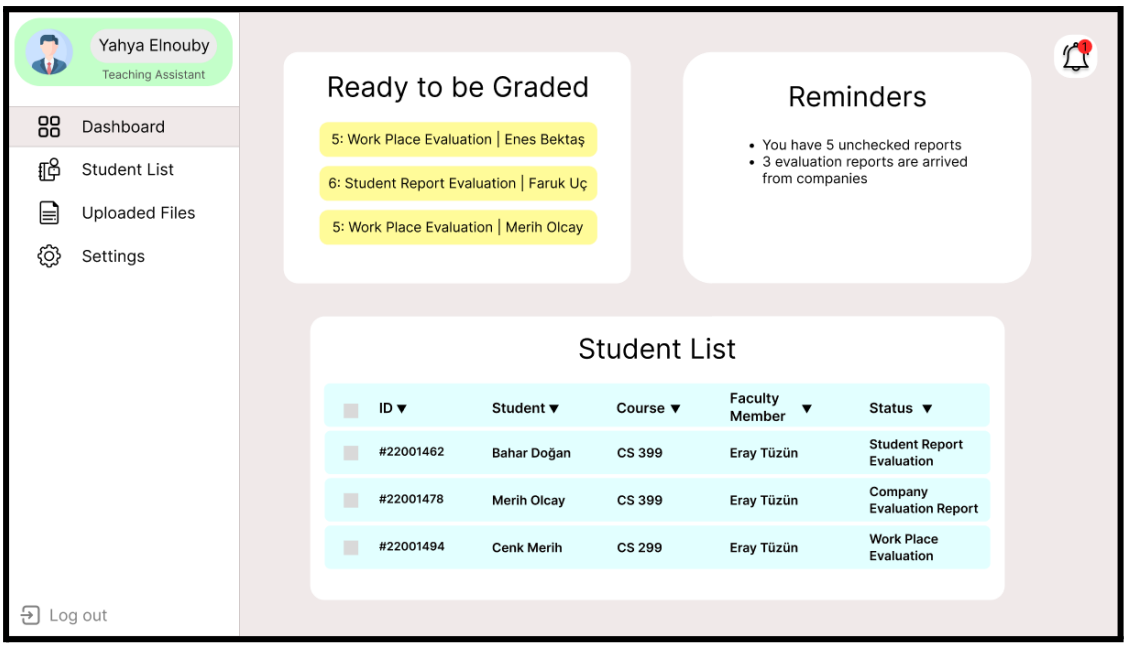

Fig. 12: Dashboard Page for Teaching Assistant

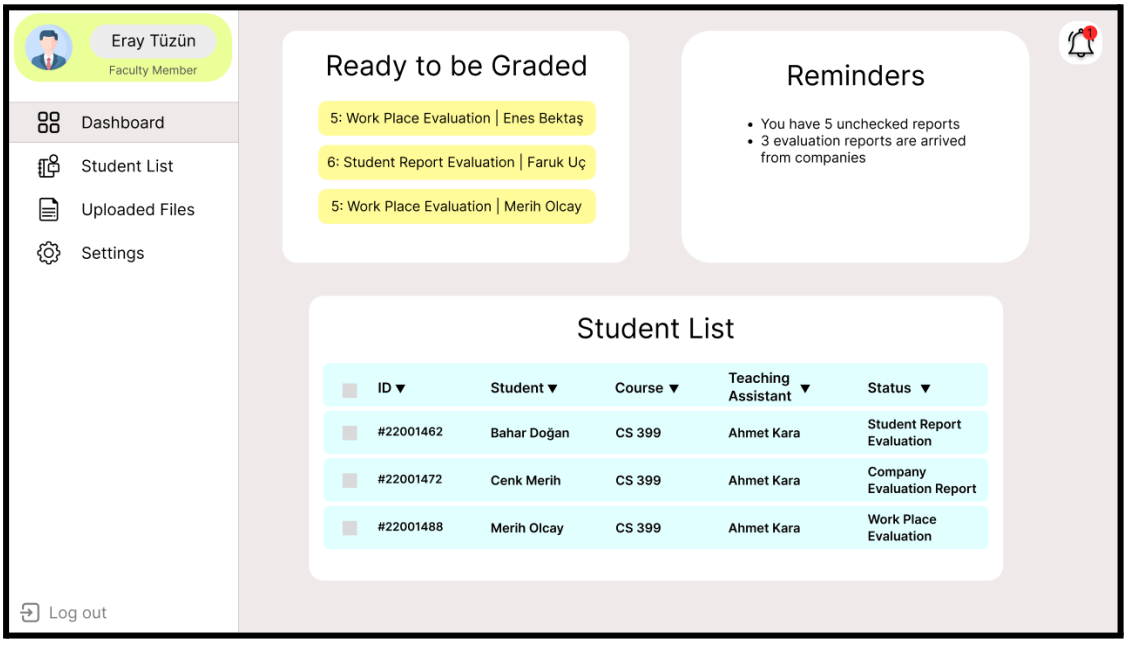

Fig. 13: Dashboard Page for Faculty Member

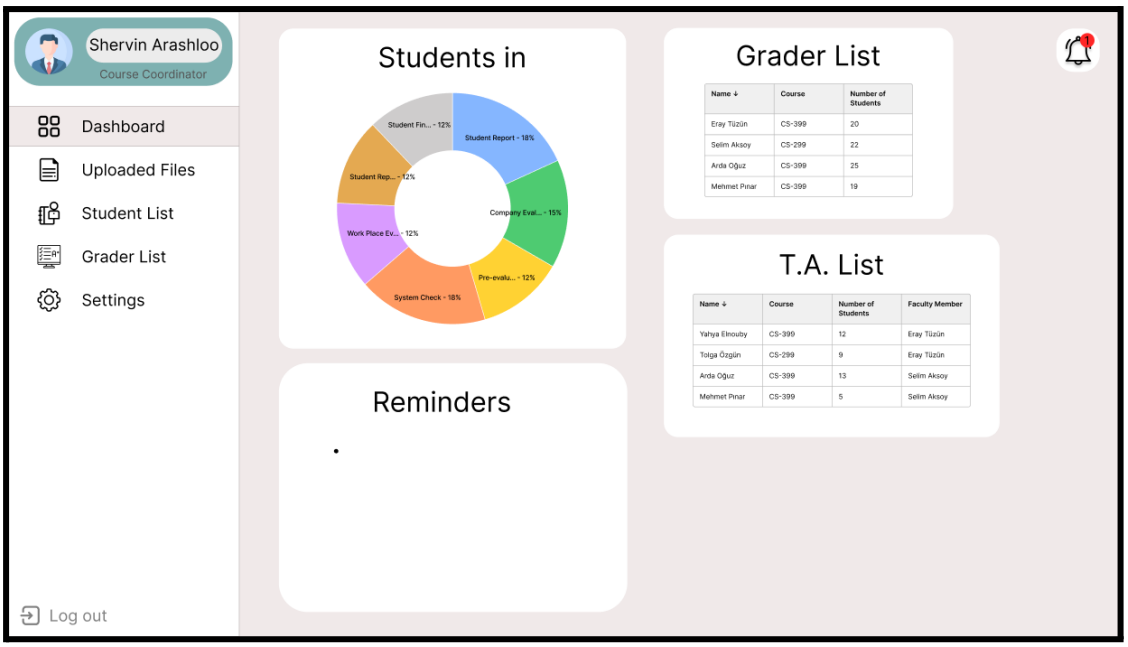

Fig. 14: Dashboard Page for Course Coordinator

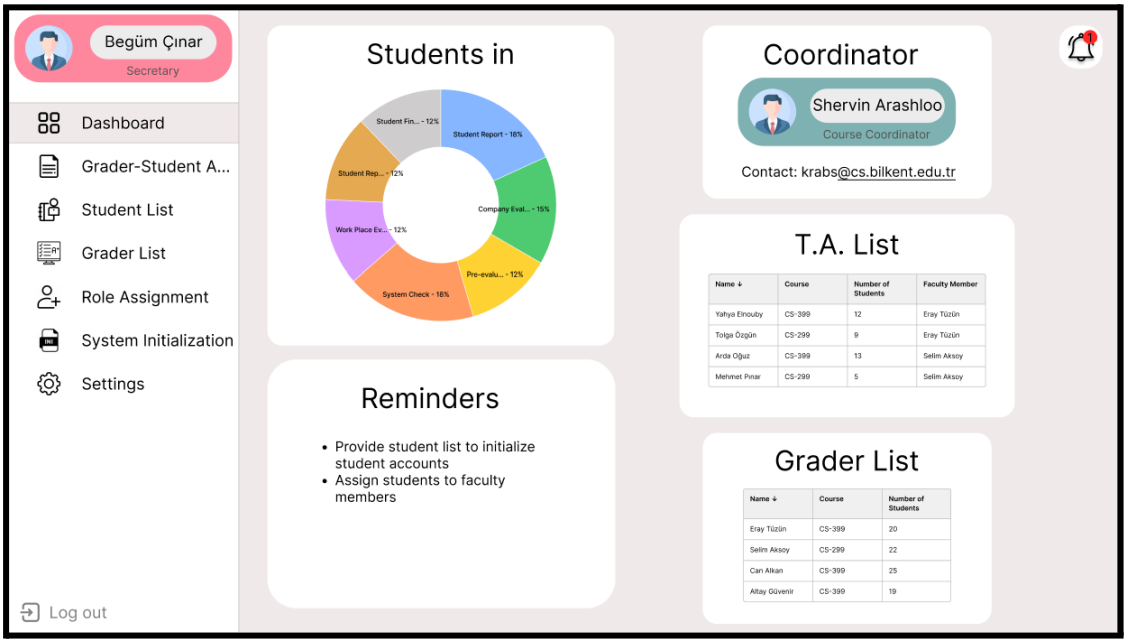

Fig. 15: Dashboard Page for Secretary

Fig. 16: Dashboard Page for BCC Admin

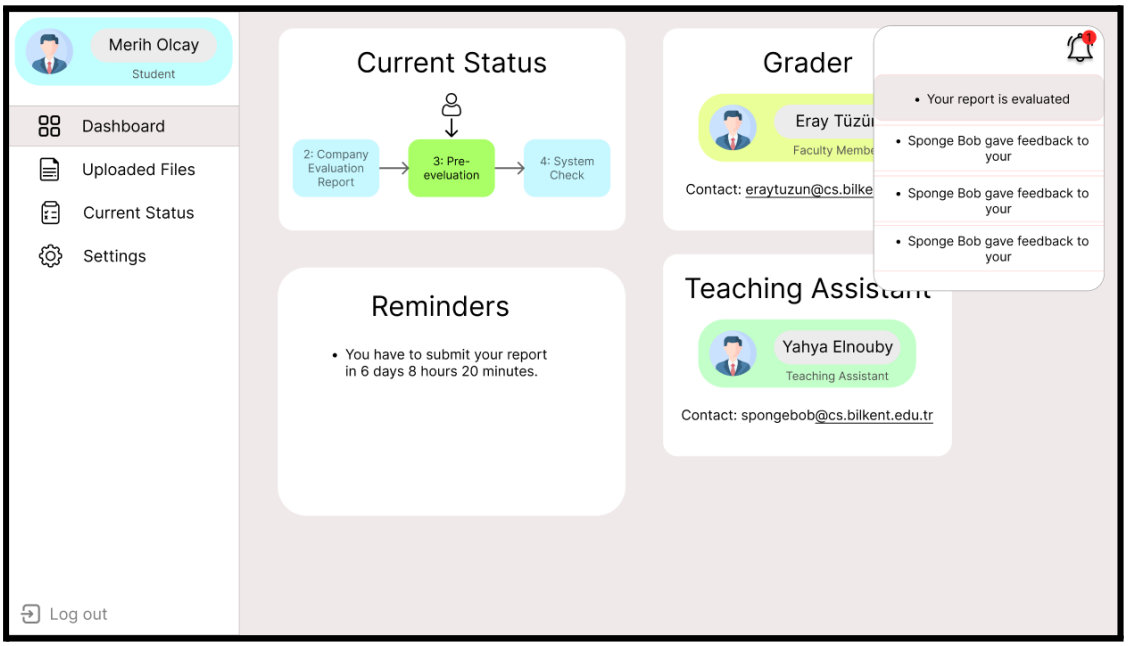

Fig. 17: Notifications

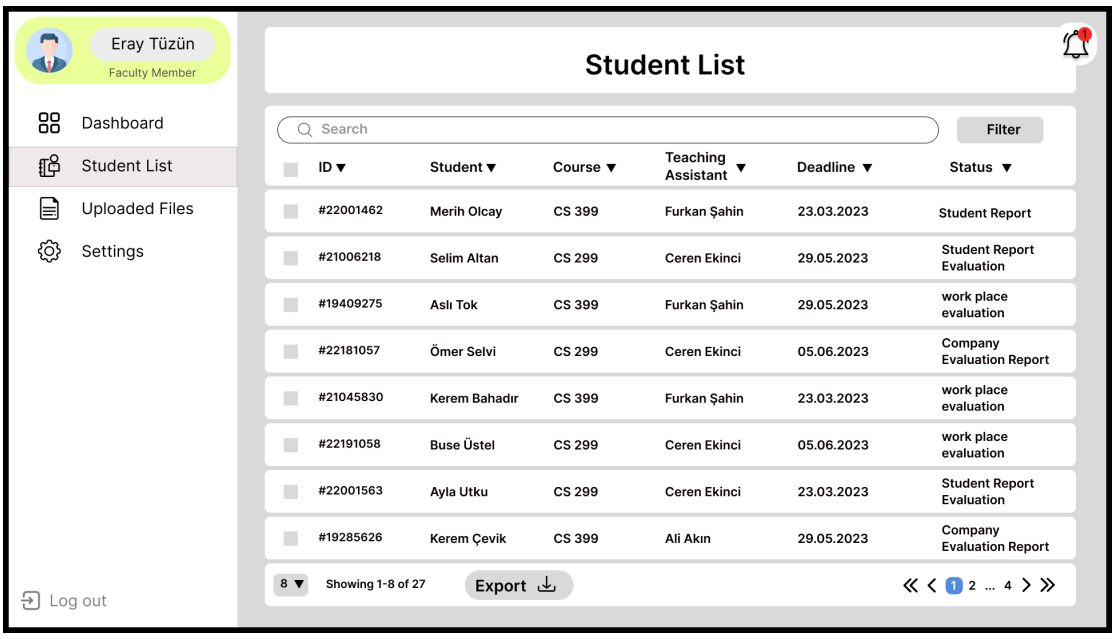

Fig. 18: Student List Page for Faculty Member

Fig. 19: Student List Page for Faculty Member

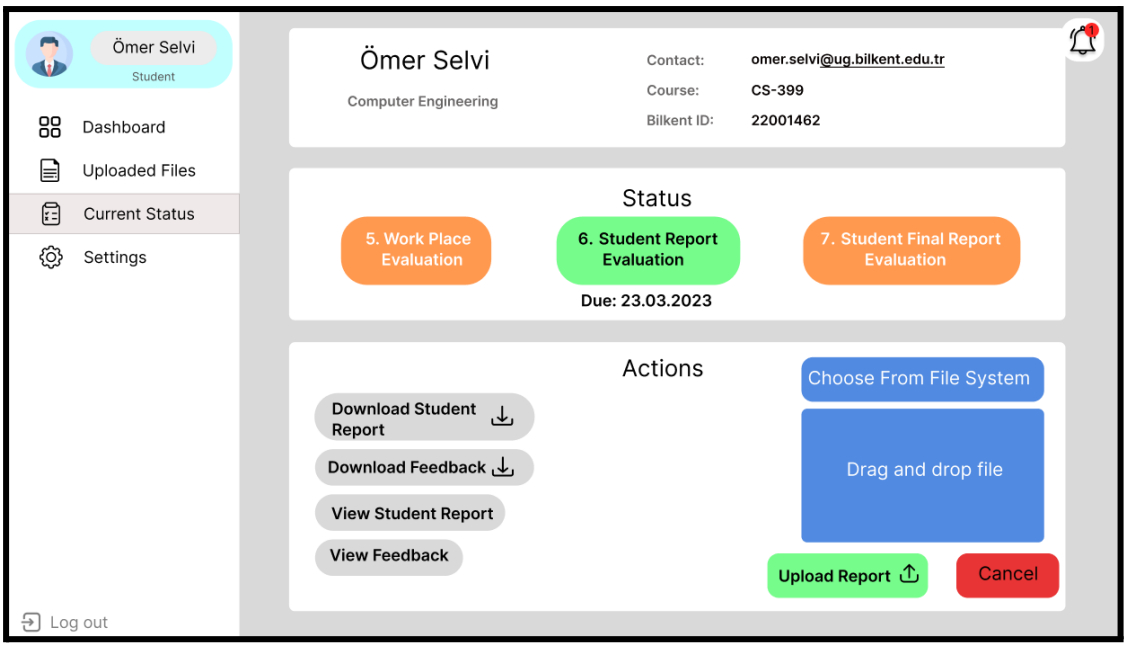

Fig. 20: Undergraduate Status Page for Student

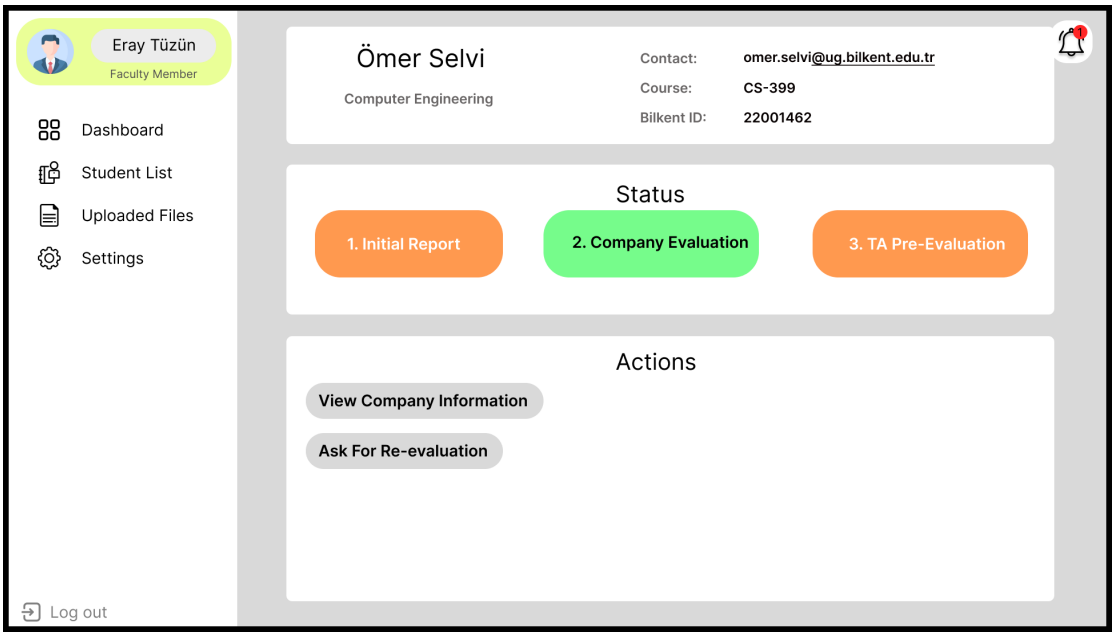

Fig. 21: Undergraduate Status Page Example for Faculty Member

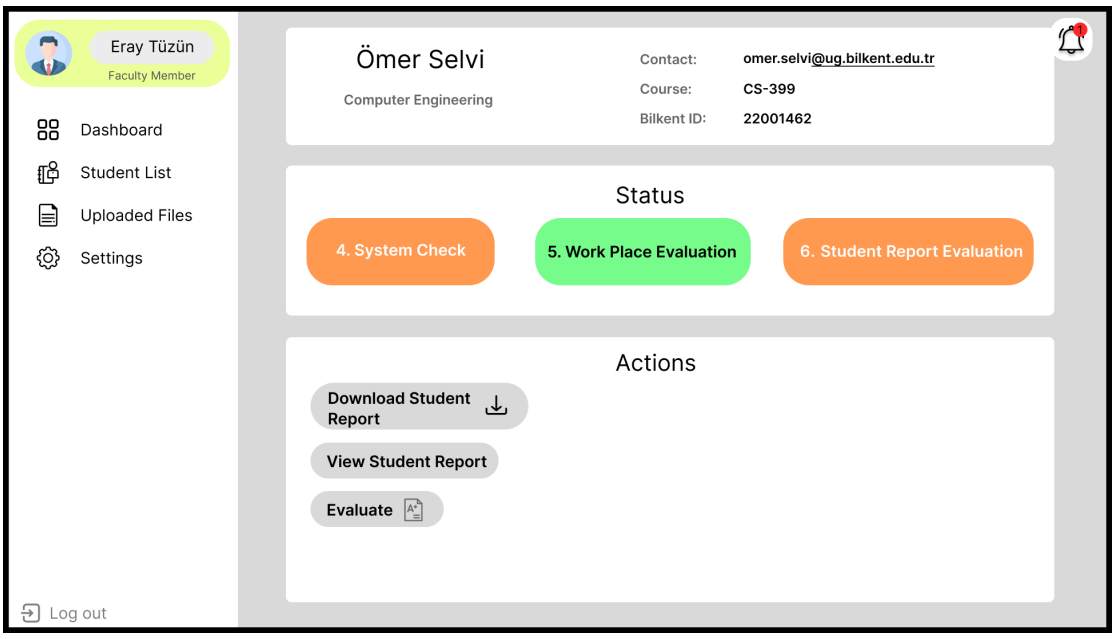

Fig. 22: Undergraduate Status Page Example for Faculty Member

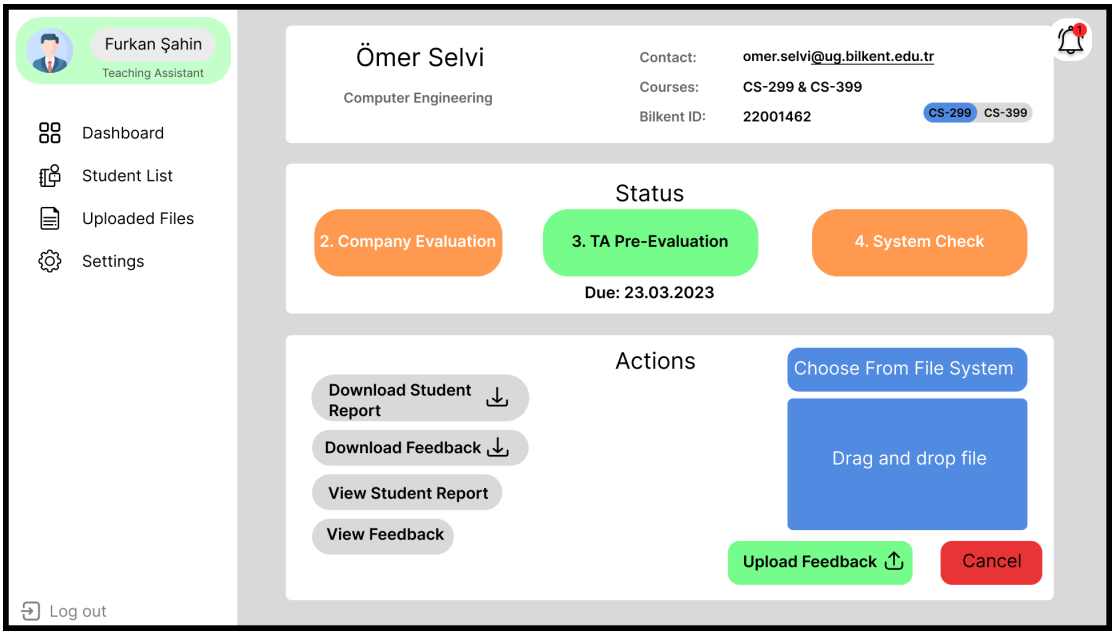

Fig. 23: Undergraduate Status Page Example for Teaching Assistant

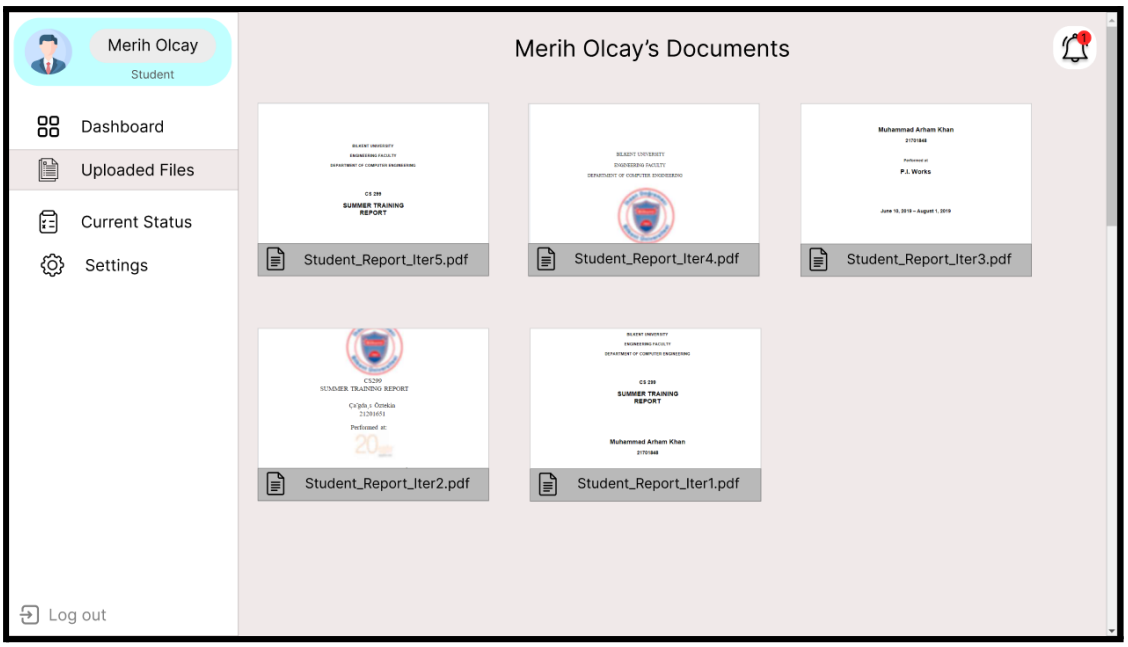

Fig. 24: Uploaded Files Page for Undergraduate

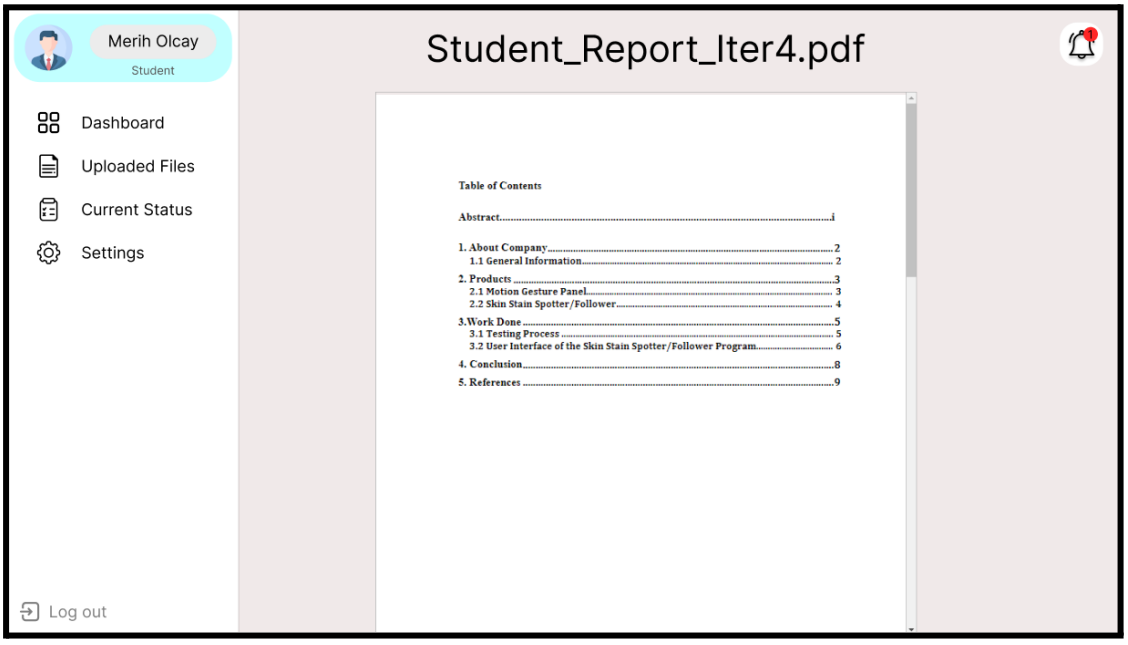

Fig. 25: Display File Page for Undergraduate

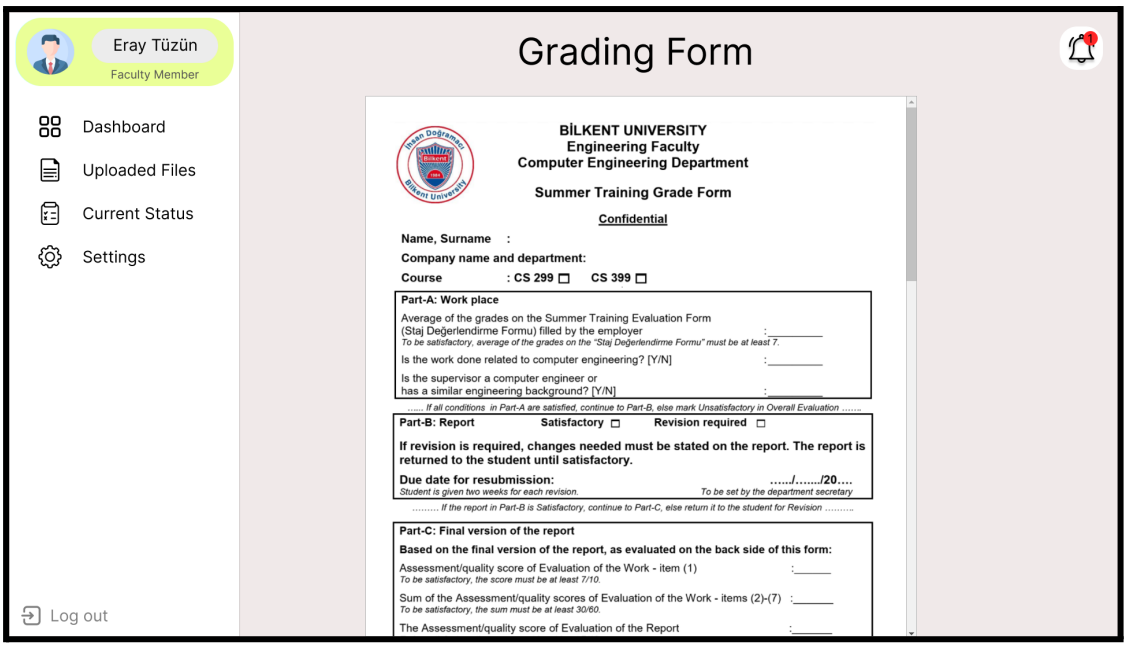

Fig. 26: Grading Form Page for Faculty Member

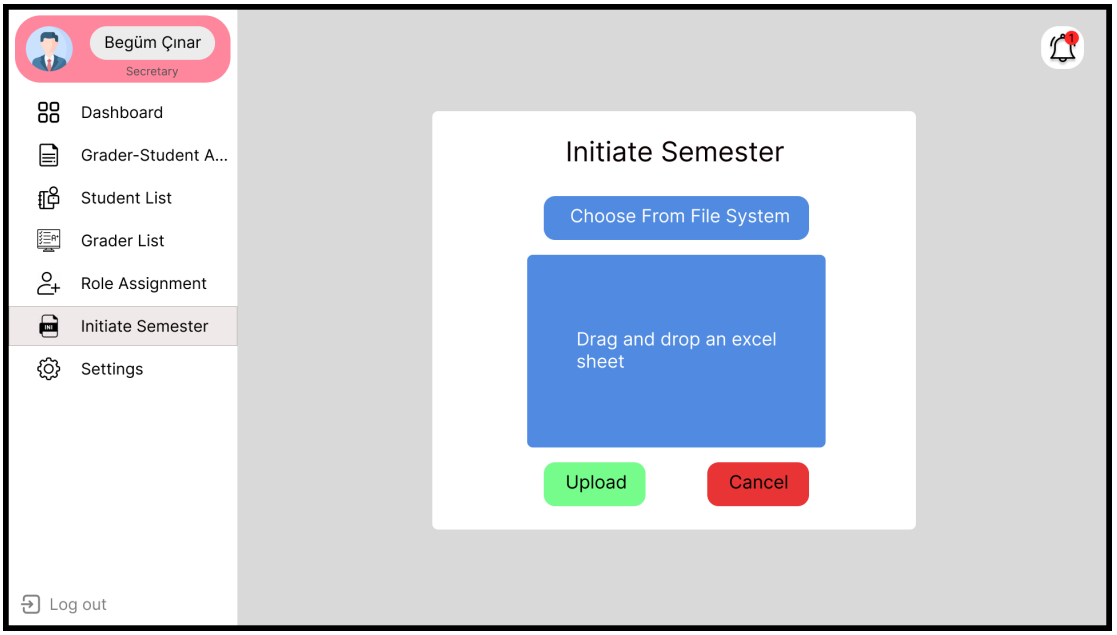

Fig. 27: Semester Initialization page for Secretary

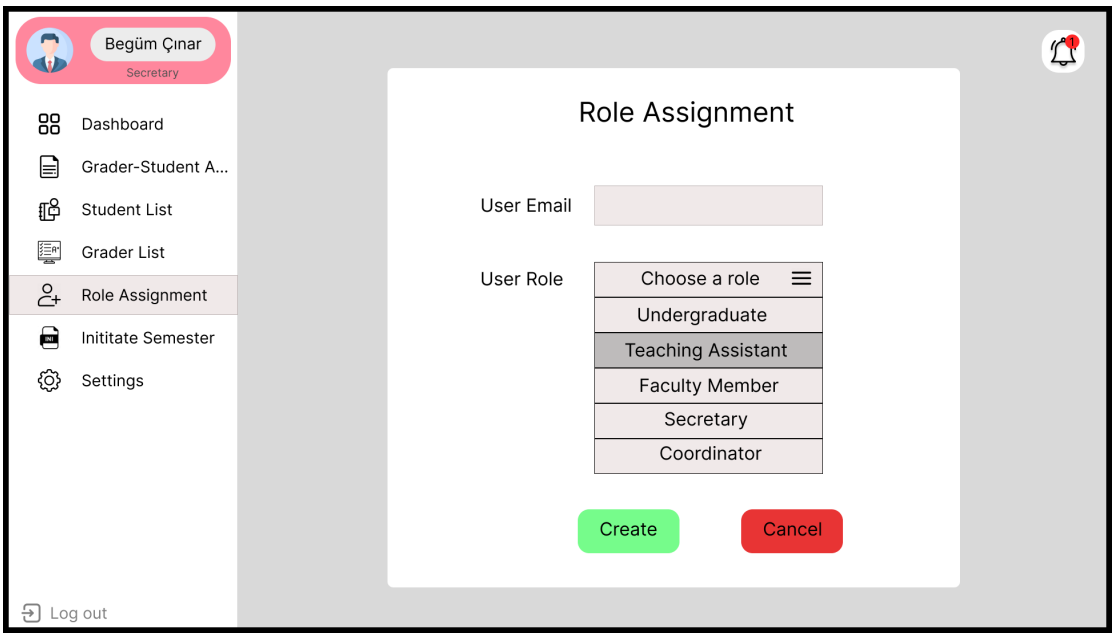

Fig. 28: Role Assignment Page for Secretary

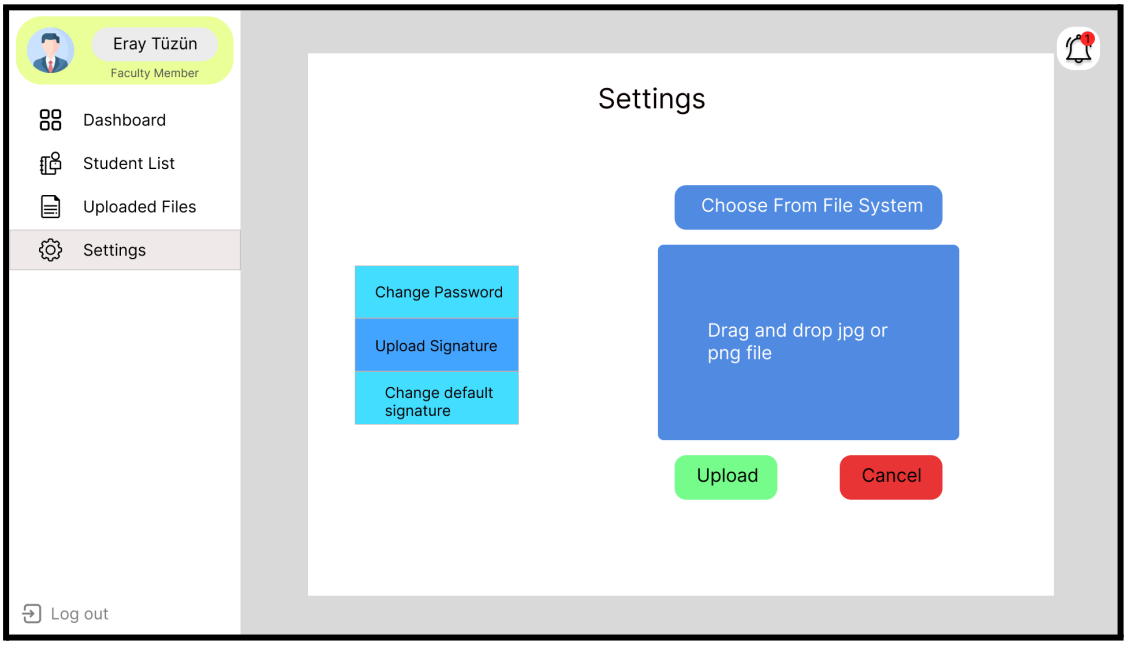

Fig. 29: Settings Page for Faculty Member

4 Improvement Summary

All the Undergraduate classes have been renamed to Student.
The object and class model has been updated to be more detailed, and establish missing links between classes. As well as some additions to the way the entity classes are set up.

User Interface:

- System initialization page has been renamed as semester initialization for clarification.

- Uploaded files page has been changed, now it works as an archive for a student's documents.

- Namings in the mockups have been changed to make them more realistic.

- Report status names are numbered to clarify the order of the stages.

- Some of the stages are renamed for further clarification, like workplace evaluation stage, student final report evaluation etc.

- Only instructors can set deadlines now, not the teaching assistants.

Use case:

-System boundary was added

-Actors were moved out of the system and were placed at a distinguishable place out of the system boundary.

-Turnitin and its use cases as an external system were connected to their respective use cases

-Attribute-like use cases were turned into words (mainly notification use cases)

-Include relation labels were placed closer to their respective vectors

Dynamic Models

- Complex expressions in state diagrams were reduced with the addition of new states.

- Some of the transitions in state diagrams have been modified to provide a clearer picture of state transitions.

- Multiple outgoing edges from actions in activity diagrams had violations, they were solved through use of fork nodes

- User Role Activation-Deactivation State Diagram was removed because the state of user roles are already obvious and it was redundant to show.

- Some parts in state diagrams that were not really representing the processes in the application domain are either changed or directly removed.

5 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.