Summer Training Report Evaluation System

Project Short Name: ***CUMREP*** *(Team: F Yazilim)*

Analysis Report

Yaşar Tatlıcıoğlu - 22003856

Ömer Fırat Bekiroğlu - 22002239

Sabri Eren Dağdelen - 22001764

Anıl Altuncu - 21901880

Emirhan Ay - 22203902

Yamaç Yiğit Ozan - 22003595

Contents

Summer Training Report Evaluation System

1. Introduction

Our project is the Summer Training Report Evaluation System that provides students who completed their summer internship (students taken cs 299 and cs 399 courses), TA's, Department Secretaries, Coordinators and Evaluators a more clear and easier summer training evaluation system. Our aim is to reduce unnecessary, long paperwork and complicated use of a combination of Moodle/Google Drive/Email systems. We aim to design a system that is efficient and effective for evaluation of summer training reports which also can be used smoothly by other departments (engineering).

Current process is extremely tiring for TA's, evaluators, department secretaries and students since nearly every actor must handle three different platforms(Moodle/Google drive/Email). Evaluators, TA's and department secretaries use different platforms to view the input (reports, revised reports, comments etc.) from students and communicate with the students. Thus, the current system confuses actors and leads to an error-prone system. Even if there is no error, dealing with surplus documents from many different platforms is a waste of time and energy for actors. In the end, this process is extremely exhausting for the actors. Our main goal is to reduce these tiring and busy combinations and steps and make them possible all usable in one system.

2. Current System

Our team attended the presentation delivered by Selim Aksoy, a project stakeholder. According to the presentation, system that used to evaluate summer training reports, tasks are done using three different platforms and compellingly. For instance, first TA's download the reports that are submitted through moodle by students, then they have to put it on to Google Drive for the following steps. This clearly shows us how challenging the system commenced. As the system progresses, such tedious steps appear. Apart from that, our observations based on Selim Aksoy's presentation and the current summer training report evaluation procedure are as follows:

- Students submit their reports through moodle

- Assigned TA’s download the reports from moodle and make a quick quality checks (page numbers, table of content, whether every student have uploaded it or not)
- Reports are divided into folders in Google Drive for evaluation
- Department assigns a faculty member to every student as an evaluator of reports. Each faculty member has a folder for assigned reports in Google Drive(cs299-cs399 reports)
- Faculty members evaluate these reports accordingly. Sometimes they give their feedback and required comments directly to the student via Email. Sometimes these notes are shared to the Begüm Çınar (administrative assistant), Begüm Çınar directs these feedback to the students via Email.
- Students revise their reports according to the given feedback and send their revised reports to the Begüm Çınar.
- Begüm Çınar uploads these reports to the Google Drive folder of the assigned faculty member. Also Begüm Çınar sends an email to the faculty members to inform them that revised reports are uploaded into their folder.
- After one or more revisions students send their final summer training reports and then faculty members fill a summer training evaluation/grading form for every student.
- In the end, students whose reports are accepted and satisfactory pass the related course(cs 299 / cs 399)

3. Proposed System

System will allow specified TAs, evaluators and students to access simplified evaluation forms, which will be used at grading and revisions. These reports will be viewable by the students and editable by the evaluators whenever they want**.** The main concept of the application will be on uploading, downloading and giving/receiving feedback on the document by users. At this concept; main purpose of the project, which is making students' and schools' jobs easy, will be conducted by designing a clear, appropriate interface and helper concepts such as student list, notifications of changes in documents etc.

3.1- Actors

There are 4 different actors defined that can interact with the application's system. There will be different interfaces and login information for different actor types. Commonly, they will save their mail address so that they can be contacted. Also, their saved passwords will be used for the protection of their accounts.

- **TA:** TA's main job in the application is to pre-evaluate the reports uploaded by the students and give the feedback -if necessary-. Changes they make in the system such as giving feedback, accepting the report as sufficient etc. will be sent as a mail to the students. Waiting reports of students will be distributed to the TA's automatically by the system. Also, TA's will have a page that shows the waiting reports and forms as a list so they easily access it.
- **Student:** Frankly, this application's main functionality for students is to upload their internship reports and forms that are filled by the company they have their intern in. Students will create their accounts by signing up with their student Id and their passwords. Also, they will login to the system with their saved student Id and password information. Students will be able to upload their reports and forms to the system easily and also see the current situations of their submissions -it has been given feedback, it is accepted etc.-
- **Evaluators:** Evaluators will sign up with their emails and passwords that they created. They will be able to log in with the emails and passwords that they saved to the system. Evaluators will be people that are given the responsibility to evaluate the student reports and the forms from

companies. They will be assigned to the reports by department secretaries.

- **Department Secretaries:** Department secretaries will have 2 main jobs. First is assigning reports to the evaluators and second is giving consent to the reports that are accepted by both the TA’s and Evaluators. There will be a secretary account for each department and it will be able to be used by the department.

3.2- Nonfunctional Requirements

Usability: Main purpose of the application is to ease the process of internship reporting between students and the school. So, it would be pointless if the application is not very handy and useful. Application must offer simplicity and usefulness in various activities that can be done in the application.

- ➢ Easy login and signup structure that doesn’t confuse or bore the user.
- ➢ Clear interface that shows headlines for pages, clear and apparent icons, locations for functional places in the applications.
- ➢ For students; easy way to upload their reports, clear and apparent information about the situation of their reports. Clear feedback information etc.
- ➢ For TA’s; there should be a student list that shows the waiting reports that are assigned to them to be pre-evaluated. They also need a good, useful interface for writing their feedback. Also, they need to have a clean choices that they can easily access, execute -such as accepting the report, giving feedback etc.-
- ➢ For evaluators; easy way to find the students’ reports waiting for them to be evaluated. There should also be a list too for them to see the waiting reports.
- ➢ Department secretaries should easily reach the evaluators, see which evaluator has how many jobs to do in the app etc. so they can easily assign them the reports. Also they should have a page

that shows the list of the reports waiting to be accepted. That would ease their work so much.

Maintainability: It is our goal that the application will be easy to maintain. Clear and well-organized coding is very strict for this application to provide maintainability. Also, basic design principles are so crucial and will be used so that the codes will be globally acceptable, executable and understandable. This way; it will be easily updated, fixed, changed etc.

Security: Users' personal information is going to be kept private. Student's basic data such as their department, Id etc. will be shown but their passwords will be protected. When it is logged in from another device, a mail will be sent to the user so that he/she will be aware of what is going on.

Reliability: Since the internships are important for students and school, there shouldn't be data loss, upload problems in files etc. . By the user's eyes, it should be an absolutely reliable system to use for its purpose. With dynamic structure and possibility to see the current situation of the reports, submissions etc. The application is traceable by users so they can rely on it.

3.3- Functional Requirements

3.3.1- User authentication and authorization: The application will make it possible for users to create accounts with their appropriate information such as their emails, their student-ids etc. and emails will be verified -otherwise no activities will be permitted to the user-. Users will be able to log in to their account by using the verified information they gave in the sign up part. Also, users will be able to change their account information again with sufficient verifications.

3.3.2- User Interface: Even if the different user types have similar interfaces, there are some differences in mostly functionalities. Generally, we target to provide a simple interface that helps users to do their singular activities easily.

Student View: Students will be able to see the files they can download, use etc. while they are conducting their reports and forms. Also they will be able to find the locations they will upload their files to easily in the appropriate pages provided for them.

TA View: Ta's will see the students assigned to them to pre-evaluate their reports in a list. They have kinds of lists such as students that are pre-evaluated, students that are waiting to be pre-evaluated etc. In this way, they will easily find student reports and forms to interact with in the application

Evaluator View: As in the Ta's, evaluators also will have student lists. The main difference between a TA and an evaluator is that they will be assigned to students manually by the department secretary. They will see the students too with kinds of filters such as evaluated, given feedback, waiting to be evaluated etc.

Department Secretary View: Department secretary will have two main functionality in the app which needs two different pages. At one page, they should see reports pre-evaluated by the TAs and waiting to be evaluated by the evaluators. Department secretaries should assign those reports to the evaluators as they wanted. Also they are going to be able to see which evaluator assigned how many reports at the time. That way, they will choose evaluators easier. Another functionality of the department secretary accounts will be accepting the reports that are accepted by a TA and an evaluator. They will also find these reports easily in a page.

3.3.3- Data Input and Output: The application will provide users to upload and download files. Students will be able to upload their reports, forms they have from companies etc. TA's, evaluators and department secretaries will be able to download these reports and also if they want, they can upload their feedback as a file again to the application -also a text space will be available to them in the interface for them to write their feedback-.

3.3.4- Notifications and Messages: Notifications will be sent to users in case of different activities. For example, students will receive a notification when their status of the submissions change. TAs, evaluators, department secretaries will receive a notification when a new report to see comes along to them, also when a report waits too long to be evaluated etc. These notifications will be sent through emails of the users.

3.4- Pseudo Requirements

- The project will be a web application.
- The source codes are going to be traceable on Github.
- The coding must be based on Object Oriented Programming.
- Specifically, back-end developers in the team will use Firebase to develop their product.
- At the front-end side, a combination of HTML-CSS and a well-known Javascript framework react.js will be used. Also, technologies of next.js will be used in these fields of the project.

3.5- System Models

3.5.1- Use Case Model:

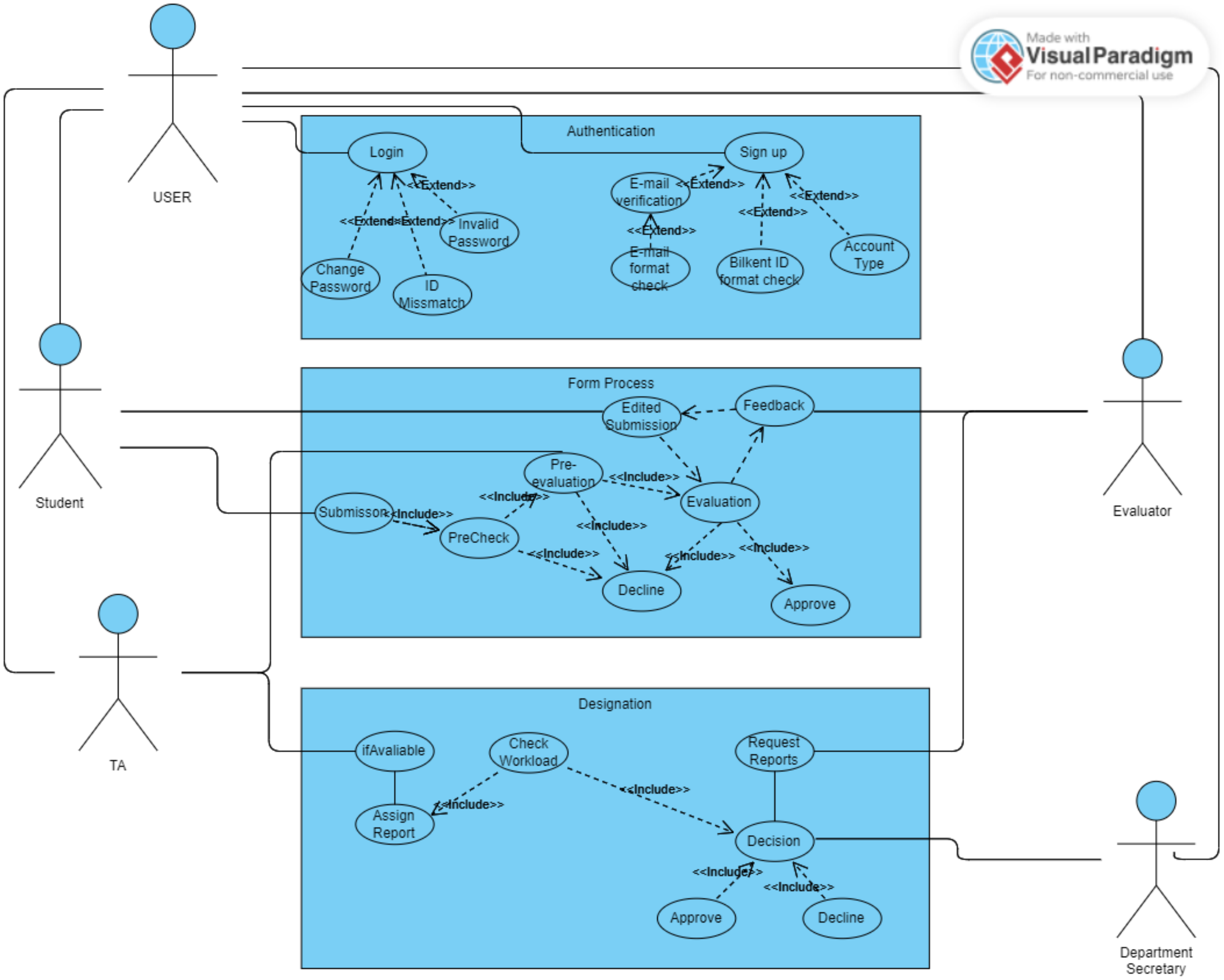


Figure 1: UML Diagram

3.5.2- Use Case Textual Descriptions (Functional Requirements):

Authentication

1. **Name:** Log In
2. **Participating Actor:** USER
3. **Entry Condition:** Opening the web app
4. **Exit Condition:** USER successfully logged in or login process fails.
5. **Flow of Events:**
    - **5.1.** The user decides to log in.
    - **5.2.** The user enters the email and password information.
    - **5.3.** If the user gives incorrect password information.
        - **5.3.1.** "Invalid Password" error message pops up.
    - **5.4.** Else if the user gives incorrect email information.
        - **5.4.1.** "Non-existing User" error message pops up.
    - **5.5.** Else the user logs in successfully.

1. **Name:** Sign Up
2. **Participating Actor:** USER
3. **Entry Condition:** Clicking the "Sign Up" button on the "Log In" page.
4. **Exit Condition:** User successfully signed up or cancels the process.
5. **Flow of Events:**
    - **5.1.** The user clicks the sign up button on the "Log In" page.
    - **5.2.** If an invalid Bilkent University ID is used during sign up.
        - **5.2.1.** "Invalid Bilkent University ID Number" error message pops up.
    - **5.3.** Else if invalid Bilkent University e-mail is entered.
        - **5.3.1.** "Invalid Bilkent University E-mail" error message pops up.
    - **5.4.** Else if account type is not specified by the user.
        - **5.4.1.** "Account Type Must Be Selected" error message pops up.
    - **5.5.** Else the user signs up successfully.
        - **5.5.1.** A verification mail is sent to the given email address.

Form Process

1. **Name:** Submission
2. **Participating Actor:** Student

3. **Entry Condition:** Being a student and uploading the internship form from the “Forms” page.
4. **Exit Condition:** Clicking “Upload” or “Cancel” buttons.
5. **Flow of Events:**

    5.1. The student decides to upload his/her internship form.

    5.2. Through the “Forms” page, he/she clicks the “Submit Internship Form” button.

1. **Name:** Pre-Evaluation
2. **Participating Actor:** TA
3. **Entry Condition:** Being a Teaching Assistant and clicking on one of the forms sent by the students.
4. **Exit Condition:** Approving or declining the form.
5. **Flow of Events:**

    5.1. The Teaching Assistant will see the assigned forms that passed the automatic “Pre-Check”

    5.2. If forms are adequate enough to pass the “Pre-Evaluation” Teaching Assistant direct the forms to “Evaluation” step

    5.3. If forms are not adequate enough Teaching Assistant clicks the “Decline” button

1. **Name:** Edited Submission
2. **Participating Actor:** Student
3. **Entry Condition:** Getting a feedback from the Evaluator
4. **Exit Condition:** Submit edited forms into the system
5. **Flow of Events:**

    5.1. Student will receive feedback on what changes do the forms need

    5.2. Through the “Forms” page, students click the “Submit Edited Internship Form” button. And submit the edited forms.

1. **Name:** Evaluation
2. **Participating Actor:** Evaluator
3. **Entry Condition:** A form that is passed “Pre-Evaluation" step is assigned to the Evaluator
4. **Exit Condition:** Either submit a feedback on the forms or approve the forms
5. **Flow of Events:**

    5.1. Evaluator will see the forms that submitted by a student

    5.2. If the forms are adequate, Evaluator will click “Approve” button

5.3. If the forms need to have changes, Evaluator will click the "Feedback" button

1. **Name:** Feedback
2. **Participating Actor:** Evaluator
3. **Entry Condition:** Evaluator decided to give feedback on the assigned forms
4. **Exit Condition:**Submitting feedback on the system
5. **Flow of Events:**

    5.1. Evaluator will write feedback and upload to the system

Designation

1. **Name:** ifAvailable
2. **Participating Actor:** Teaching Assistant
3. **Entry Condition:** Being a Teaching Assistant
4. **Exit Condition:** The Teaching Assistant is available
5. **Flow of Events:**

    5.1. System automatically checks the Teaching Assistant is available or not

    5.2. If the teaching assistant is available, direct the Teaching Assistant to "Assign Report" step

1. **Name:** Assign Report
2. **Participating Actor:** Teaching Assistant
3. **Entry Condition:** Being a available Teaching Assistant
4. **Exit Condition:** The Teaching Assistant is assigned forms and no longer available
5. **Flow of Events:**

    5.1. System will make "Check Workload" step automatically

    5.2. If the Teaching Assistant' workload is let him/her get new forms, forms will be assigned automatically

1. **Name:** Check Workload
2. **Participating Actor:** Teaching Assistant, Evaluator
3. **Entry Condition:** Being a available Teaching Assistant or an Evaluator who requested new forms

4. **Exit Condition:** Find out the workload of the Teaching Assistant/Evaluator
5. **Flow of Events:**

    5.1. System will automatically count and compare the work done by the Teaching Assistant/Evaluator

1. **Name:** Request Reports
2. **Participating Actor:** Evaluator
3. **Entry Condition:** Being an Evaluator
4. **Exit Condition:** Requesting forms
5. **Flow of Events:**

    5.1. Through the “Forms” page, Evaluator clicks the “Request Form” button.

    5.2. System directs the request to the “Decision” step.

1. **Name:** Decision
2. **Participating Actor:** Department Secretary
3. **Entry Condition:** Being a Department Secretary Account
4. **Exit Condition:** Either “Approve” or “Decline” the request
5. **Flow of Events:**

    5.1. System will make “Check Workload” step automatically

    5.2. Department Secretary will see the request and the workload

    5.3. If Department Secretary decide that the request is appropriate, he/she will click “Approve” button

    5.4. If Department Secretary decide that the request is not appropriate, he/she will click “Decline” button

3.5.3- Object & Class Model:

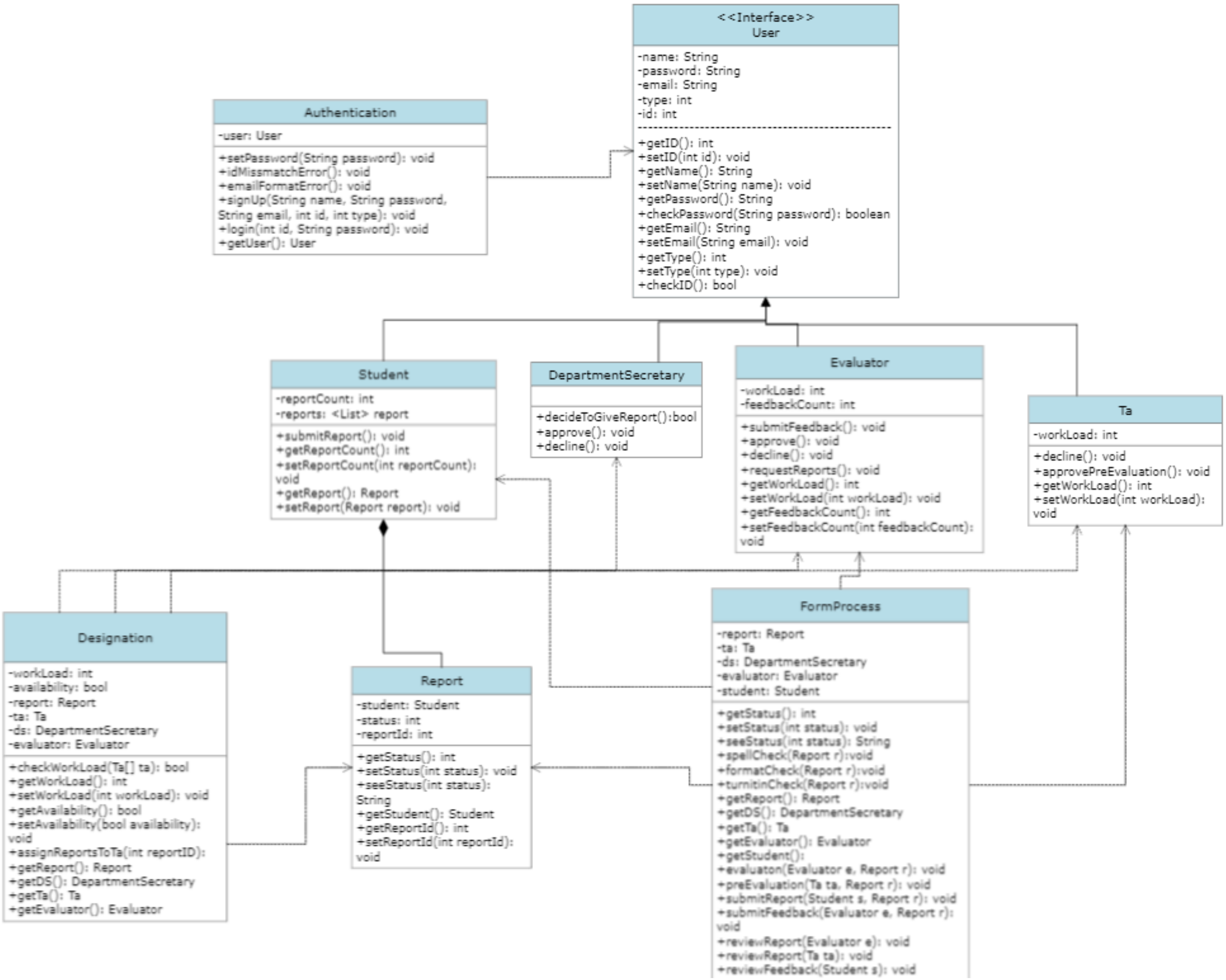

<<Interface>>
User
-name: String
-password: String
-email: String
-type: int
-id: int
+getID(): int
+setID(int id): void
+getName(): String
+setName(String name): void
+getPassword(): String
+checkPassword(String password): boolean
+getEmail(): String
+setEmail(String email): void
+getType(): int
+setType(int type): void
+checkID(): bool
Authentication
-user: User
+setPassword(String password): void
+idMissmatchError(): void
+emailFormatError(): void
+signUp(String name, String password,
String email, int id, int type): void
+login(int id, String password): void
+getUser(): User
Student
-reportCount: int
-reports: <List> report
+submitReport(): void
+getReportCount(): int
+setReportCount(int reportCount):
void
+getReport(): Report
+setReport(Report report): void
DepartmentSecretary
+decideToGiveReport():bool
+approve(): void
+decline(): void
Evaluator
-workLoad: int
-feedbackCount: int
+submitFeedback(): void
+approve(): void
+decline(): void
+requestReports(): void
+getWorkLoad(): int
+setWorkLoad(int workLoad): void
+getFeedbackCount(): int
+setFeedbackCount(int feedbackCount):
void
Ta
-workLoad: int
+decline(): void
+approvePreEvaluation(): void
+getWorkLoad(): int
+setWorkLoad(int workLoad):
void
Designation
-workLoad: int
-availability: bool
-report: Report
-ta: Ta
-ds: DepartmentSecretary
-evaluator: Evaluator
+checkWorkLoad(Ta[] ta): bool
+getWorkLoad(): int
+setWorkLoad(int workLoad): void
+getAvailability(): bool
+setAvailability(bool availability):
void
+assignReportsToTa(int reportID):
+getReport(): Report
+getDS(): DepartmentSecretary
+getTa(): Ta
+getEvaluator(): Evaluator
Report
-student: Student
-status: int
-reportId: int
+getStatus(): int
+setStatus(int status): void
+seeStatus(int status):
String
+getStudent(): Student
+getReportId(): int
+setReportId(int reportId):
void
FormProcess
-report: Report
-ta: Ta
-ds: DepartmentSecretary
-evaluator: Evaluator
-student: Student
+getStatus(): int
+setStatus(int status): void
+seeStatus(int status): String
+spellCheck(Report r):void
+formatCheck(Report r):void
+turnitinCheck(Report r):void
+getReport(): Report
+getDS(): DepartmentSecretary
+getTa(): Ta
+getEvaluator(): Evaluator
+getStudent():
+evaluaton(Evaluator e, Report r): void
+preEvaluation(Ta ta, Report r): void
+submitReport(Student s, Report r): void
+submitFeedback(Evaluator e, Report r):
void
+reviewReport(Evaluator e): void
+reviewReport(Ta ta): void
+reviewFeedback(Student s): void

Figure 2: Class Diagram

3.5.4 Dynamic Models

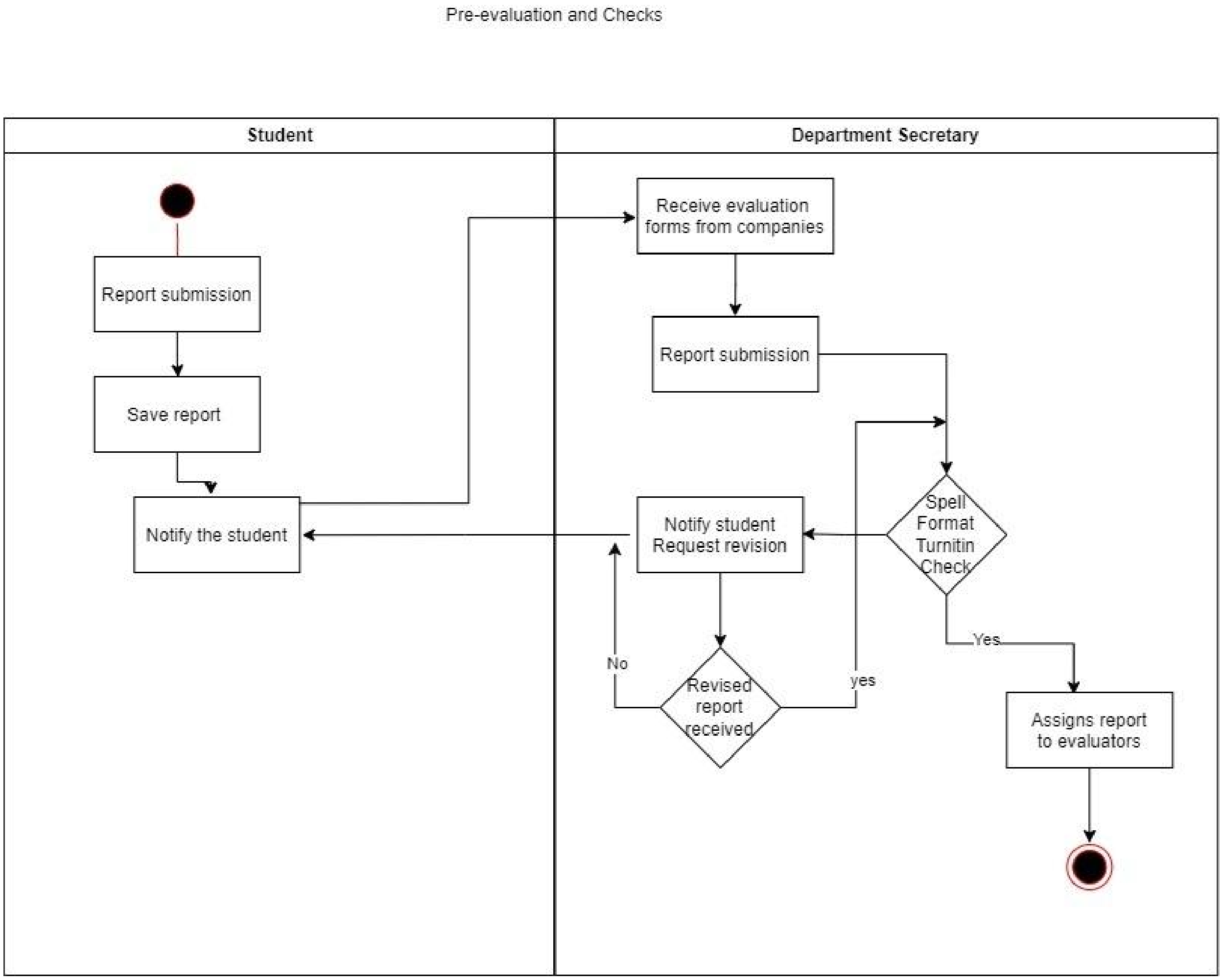

Pre-evaluation and Checks
Student
Department Secretary
Report submission
Save report
Notify the student
Receive evaluation forms from companies
Report submission
Spell Format Turnitin Check
Notify student Request revision
Revised report received
No
yes
Yes
Assigns report to evaluators

Figure 3: Pre-evaluation and Checks Dynamic Model

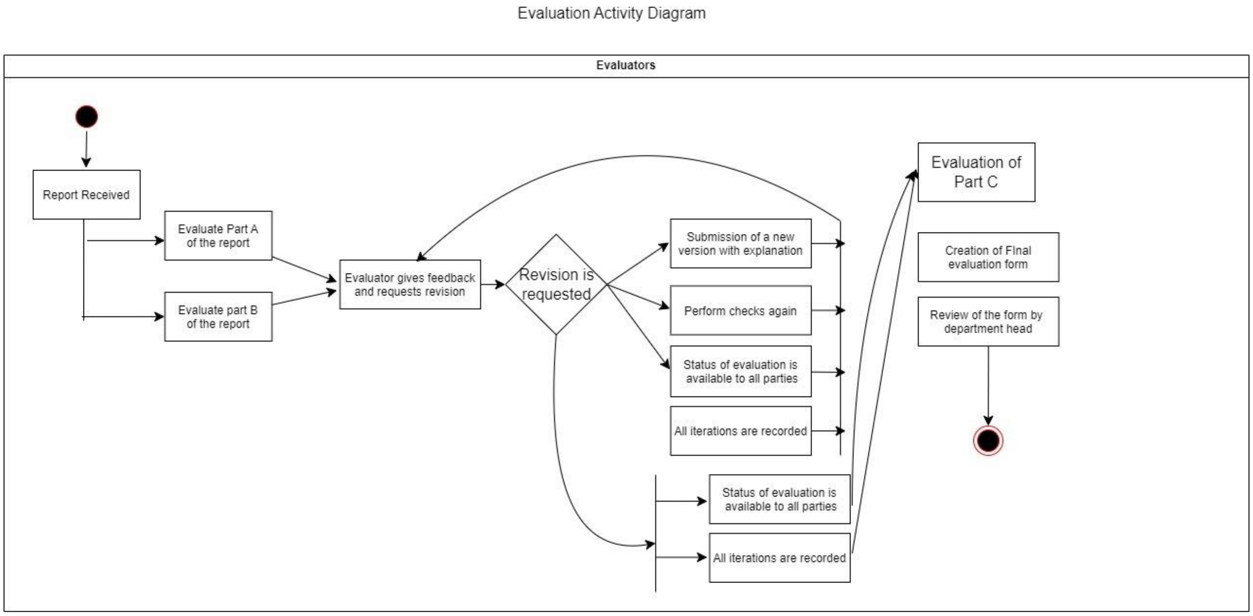

Evaluation Activity Diagram
Evaluators
Report Received
Evaluate Part A of the report
Evaluate part B of the report
Evaluator gives feedback and requests revision
Revision is requested
Submission of a new version with explanation
Perform checks again
Status of evaluation is available to all parties
All iterations are recorded
Status of evaluation is available to all parties
All iterations are recorded
Evaluation of Part C
Creation of FInal evaluation form
Review of the form by department head

Figure 4: Evaluation Activity Dynamic Model

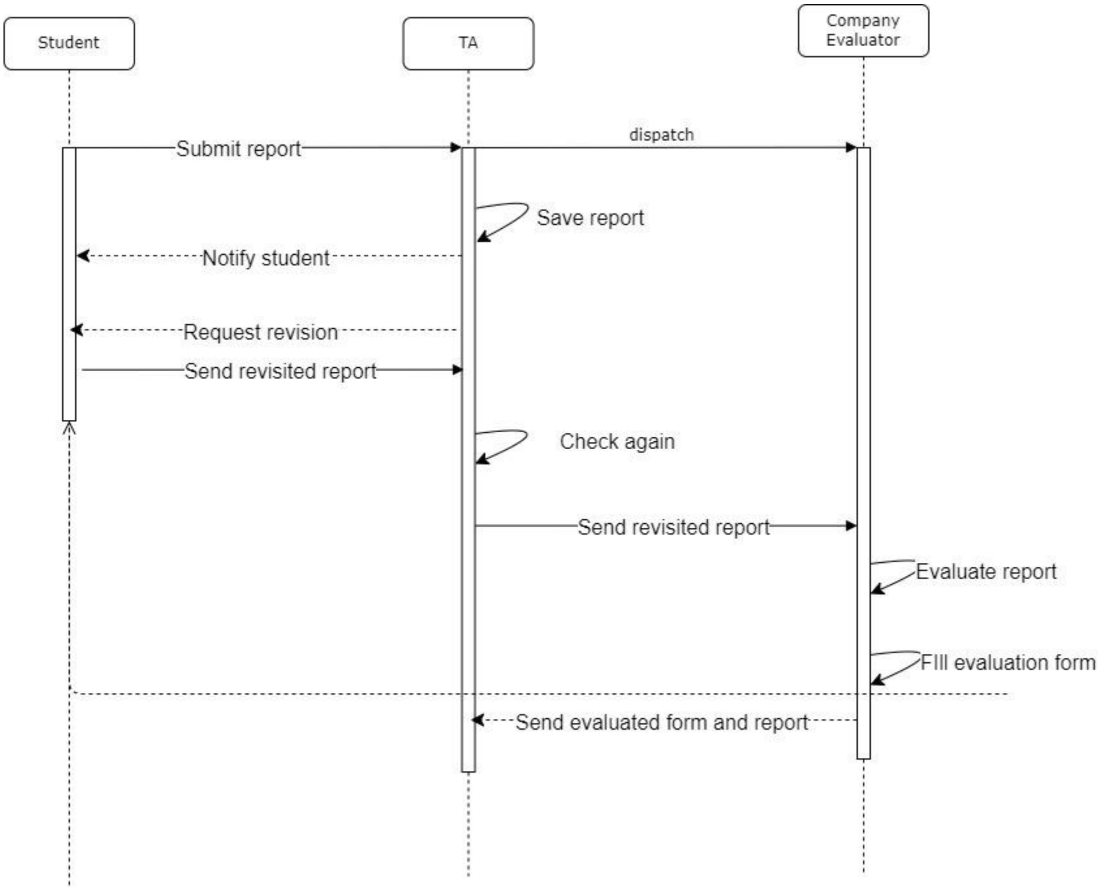


Figure 5: Sequence Diagram for Report Evaluation

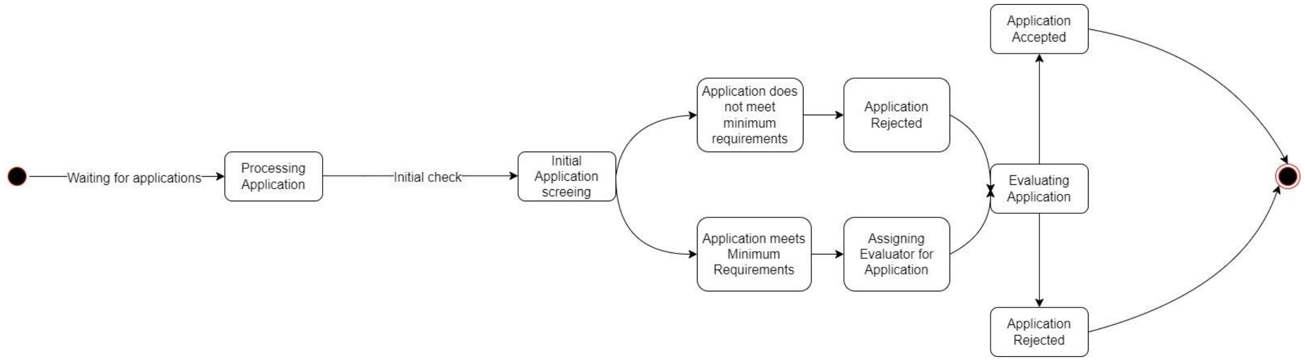
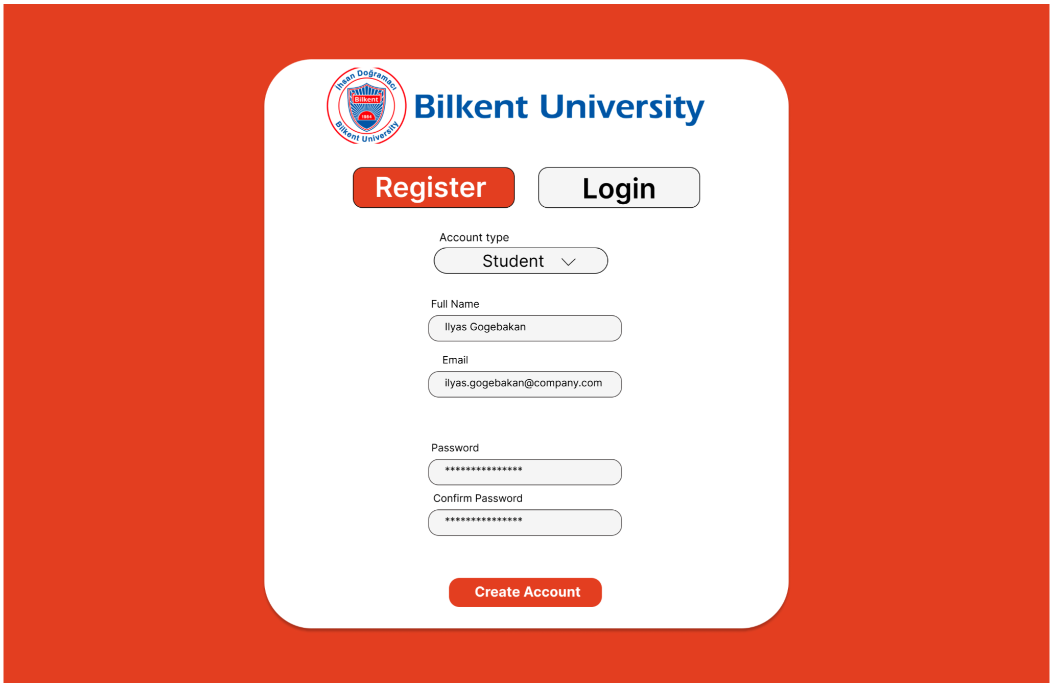

Waiting for applications
Processing Application
Initial check
Initial Application screeing
Application does not meet minimum requirements
Application Rejected
Application meets Minimum Requirements
Assigning Evaluator for Application
Application Accepted
Evaluating Application
Application Rejected

Figure 6: State Diagram of Evaluating Report

3.5.5 User Interface

3.5.5.1 Login and Sign Up Screens



Account type
Student

Full Name
Ilyas Gogebakan

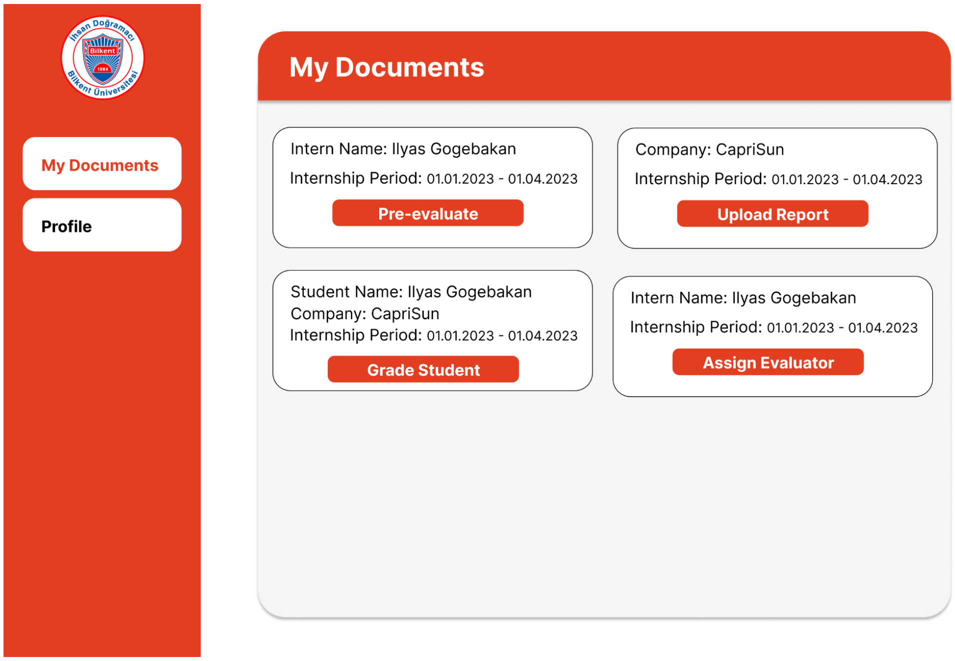
Email
ilyas.gogebakan@company.com

Password
****

Confirm Password
****

Figure 7: Login and Sign Up Screens (Example of Student account)

3.5.5.2 My Documents Page Example for All Types of Accounts



Intern Name: Ilyas Gogebakan
Internship Period: 01.01.2023 - 01.04.2023

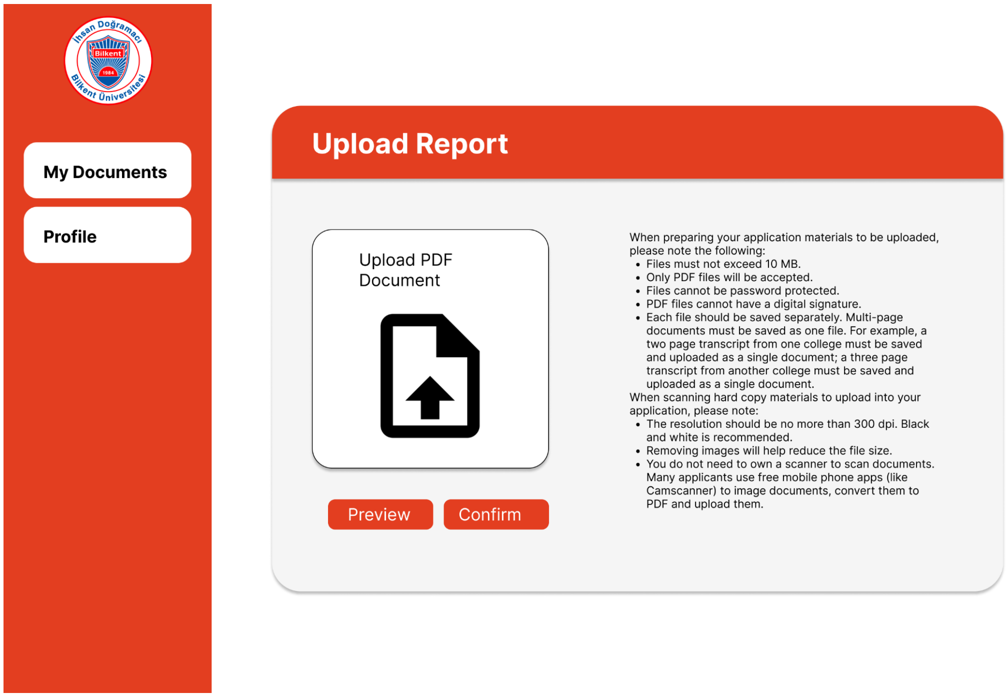
Company: CapriSun
Internship Period: 01.01.2023 - 01.04.2023

Student Name: Ilyas Gogebakan
Company: CapriSun
Internship Period: 01.01.2023 - 01.04.2023

Intern Name: Ilyas Gogebakan
Internship Period: 01.01.2023 - 01.04.2023

Figure 8: My Documents page (Example of three documents for each type of account)

3.5.5.3 Upload Internship Report Screen for Student Accounts



When preparing your application materials to be uploaded, please note the following:

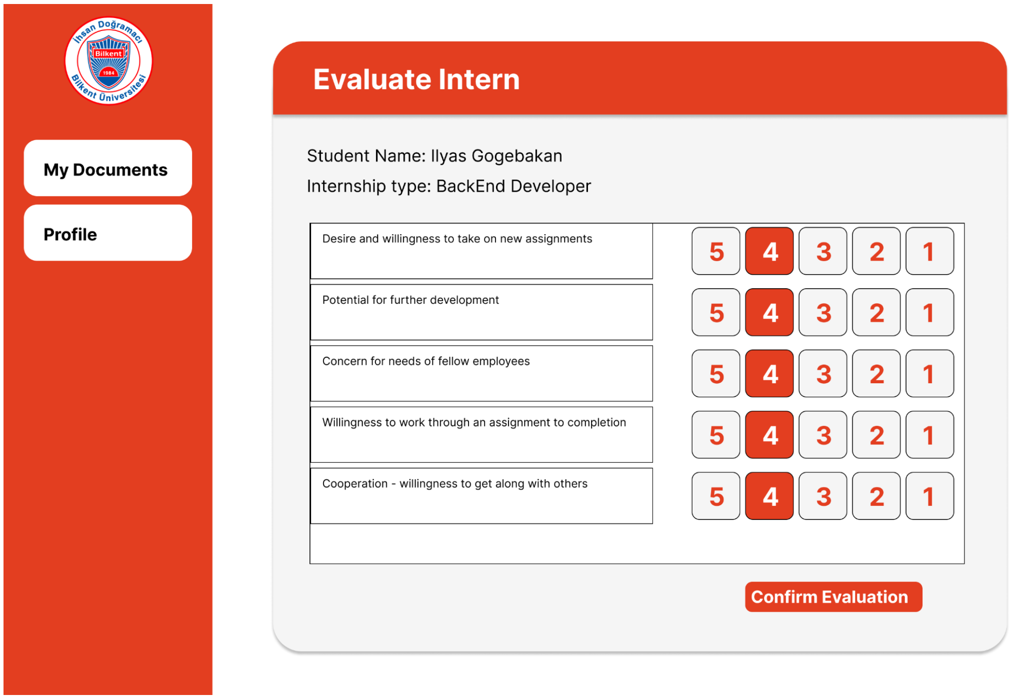
- Files must not exceed 10 MB.
- Only PDF files will be accepted.
- Files cannot be password protected.
- PDF files cannot have a digital signature.
- Each file should be saved separately. Multi-page documents must be saved as one file. For example, a two page transcript from one college must be saved and uploaded as a single document; a three page transcript from another college must be saved and uploaded as a single document.

When scanning hard copy materials to upload into your application, please note:

- The resolution should be no more than 300 dpi. Black and white is recommended.
- Removing images will help reduce the file size.
- You do not need to own a scanner to scan documents. Many applicants use free mobile phone apps (like Camscanner) to image documents, convert them to PDF and upload them.

Figure 10: Upload Report Page

3.5.5.4 Evaluate Interns Page for Company Accounts



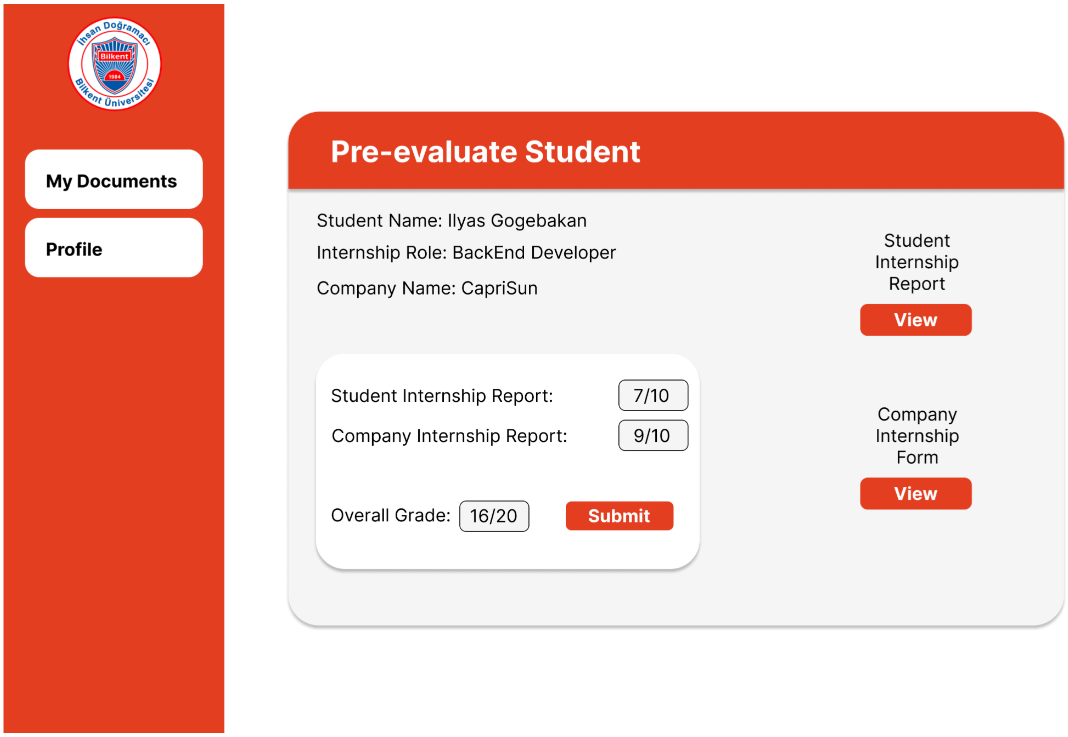
Student Name: Ilyas Gogebakan

Internship type: BackEnd Developer

Desire and willingness to take on new assignments	5 [✓ 4] 3 2 1
Potential for further development	5 [✓ 4] 3 2 1
Concern for needs of fellow employees	5 [✓ 4] 3 2 1
Willingness to work through an assignment to completion	5 [✓ 4] 3 2 1
Cooperation - willingness to get along with others	5 [✓ 4] 3 2 1

Figure 11: Evaluate Intern Page

3.5.5.5 Pre-evaluate Students Page for Evaluator Accounts



Student Name: Ilyas Gogebakan

Internship Role: BackEnd Developer

Company Name: CapriSun

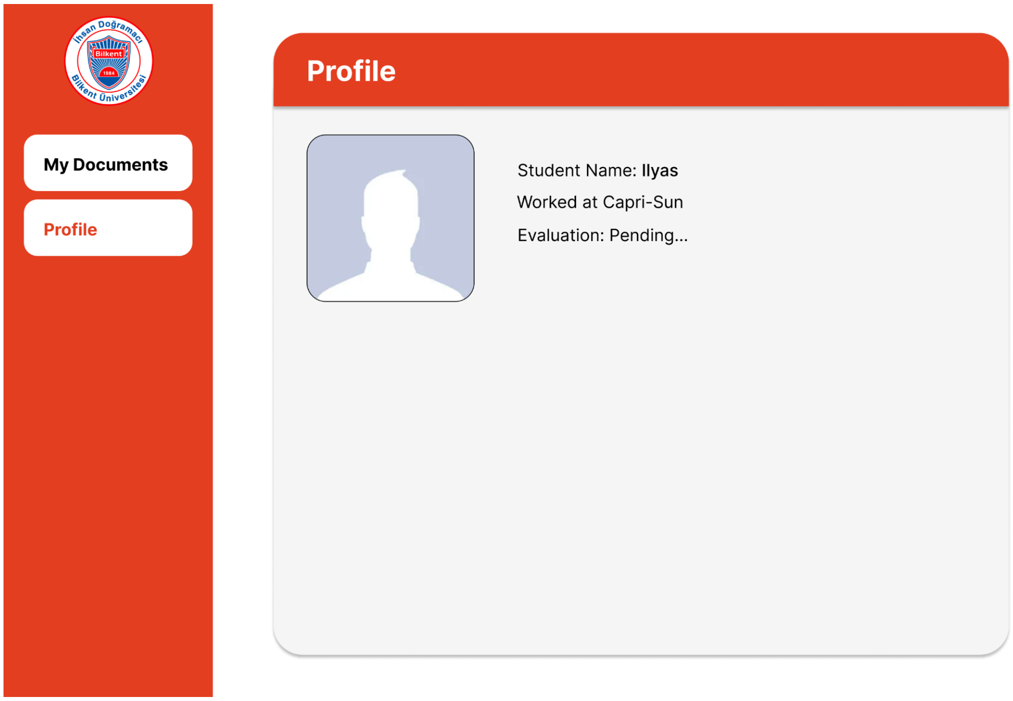
Student Internship Report: 7/10

Company Internship Report: 9/10

Overall Grade: 16/20

Figure 12: Pre-evaluate Student Page

3.5.5.6 Profile Page



Profile

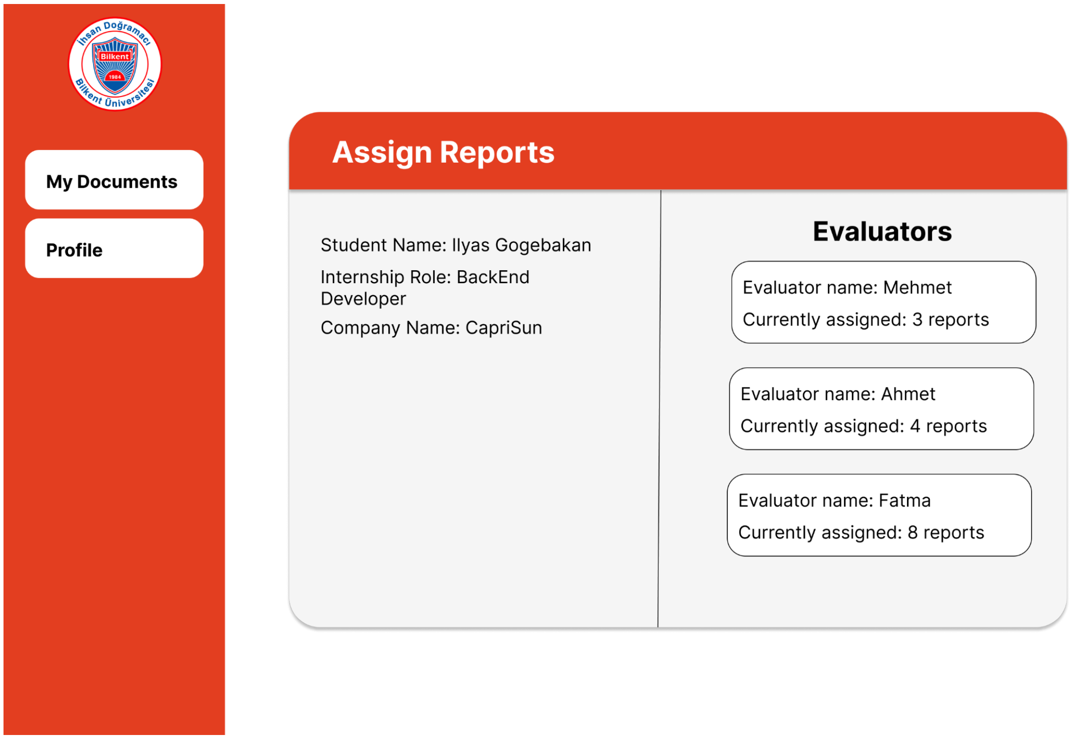
Student Name: **Ilyas**

Worked at Capri-Sun

Evaluation: Pending...

Figure 13: Profile Page (Example of student account)

3.5.5.7 Assign Reports Page



Assign Reports

Student Name: Ilyas Gogebakan

Internship Role: BackEnd Developer

Company Name: CapriSun

Evaluators

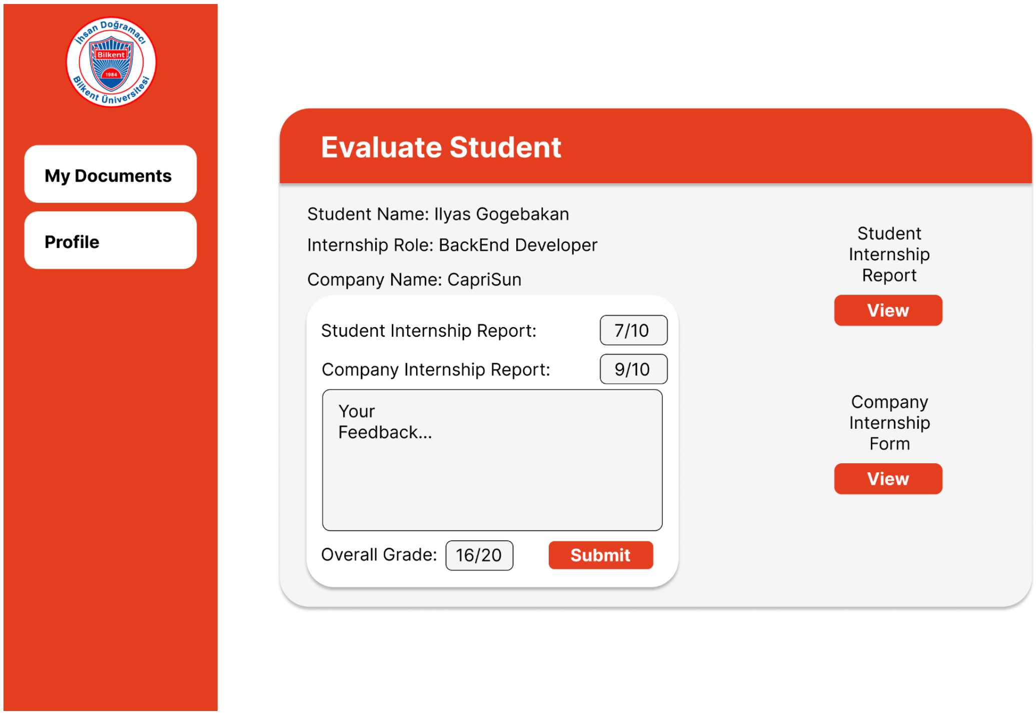
Evaluator name: Mehmet
Currently assigned: 3 reports

Evaluator name: Ahmet
Currently assigned: 4 reports

Evaluator name: Fatma
Currently assigned: 8 reports

Figure 14: Assign Reports (For department secretary)

3.5.5.8 Evaluate Students Page



Student Name: Ilyas Gogebakan

Internship Role: BackEnd Developer

Company Name: CapriSun

Student Internship Report: 7/10

Company Internship Report: 9/10

Overall Grade: 16/20

Figure 15: Evaluate Student (For evaluators)

Note: Some figures only represent one type of account, fields in these might differ depending on account type.

4. References

Designs were made with figma, *figma.com*, accessed 24 March 2023, <https://www.figma.com>.

UML diagram made with VisualParadigm, *online.visual-paradigm.com*, accessed 26 March 2023, <https://online.visual-paradigm.com/>.

Class diagram made with smartdraw, *cloud.smartdraw.com*, accessed 28 March 2023, <https://cloud.smartdraw.com/>.