

Token Management System

Final Project Report

Subject Code: CSE411

Section: 03

**Submitted To:**  **Submitted By:**

Md. Mohsin Uddin Kafi Khan

Senior Lecturer ID: 2017-2-60-078

Department of CSE Nazifa Bhuiyan

East West University ID: 2017-2-60-138

Nahida Sultana

ID: 2017-2-60-139

Date of Submission

7/1/2021

TABLE OF CONTENTS

Topic Page

Introduction and Motivations 3

Challenges 4

Component Diagram 4

Requirements 5 - 6

Non-functional Requirements 5

Functional Requirements 5 - 6

UML Use Case Diagrams 6 - 8

UML Class Diagrams 9

UML Activity and Sequence Diagrams 10 - 19

UML Activity Diagrams 10 - 12

UML Sequence Diagrams 13 - 15

Entity Relationship Diagram 16

Conclusion 16

User Interface Screenshot 17 - 22

Token Management System

**Introduction:** Every organization has processes that require the involvement of various personnel from the organization’s hierarchy. Such processes may take weeks or months to process and the personnel associated with the process usually does not have any convenient system to keep track of the process. As a result, they often have to submit to the solicitation; which is inconvenient for all the parties involved. Our team introduces a web application that assigns a token to each process and makes it convenient for the users to track the progression of a process. Keeping adaptability, security, and robustness in mind, we have devised a Token management system for East West University. Furthermore, with its easy-to-use interface, it can be tweaked to be used in any organization to manage its complex procedures with the click of a few buttons. This paper contains the Software Requirement Specification (SRS) of the aforementioned system.

*Token: An unique identification of a request.*

**Motivation:** Our primary objective is to make the process such as changing sections, dropping semester, etc. easy for both students and the administrative bodies involved by computerizing the system through the introduction of a web application. The motivations behind the project are as follows:

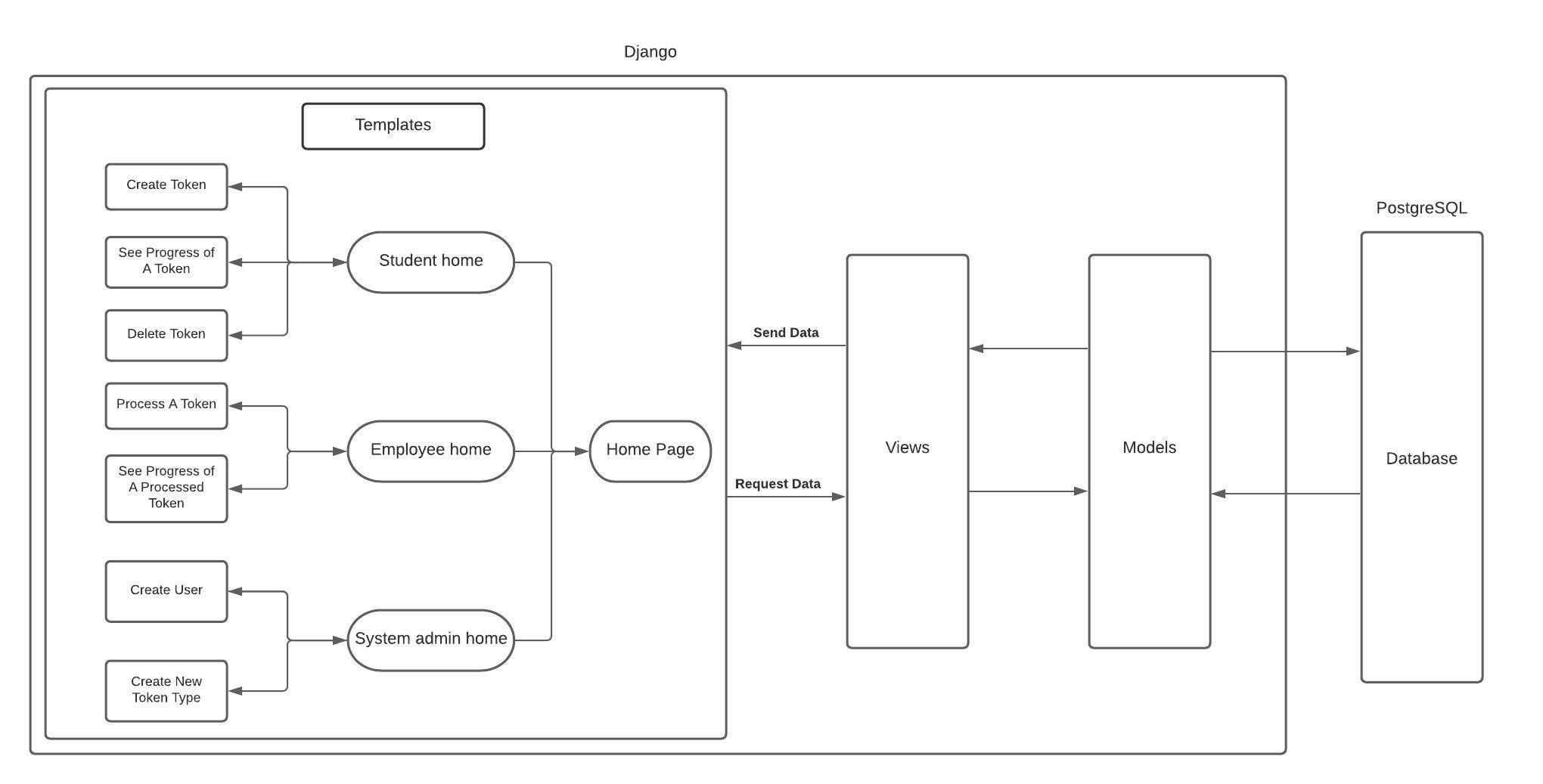
* Usually, students have to reach out to several administrative bodies such as their advisor, the chairperson, and course instructors to change sections or drop a semester. Which is both inconvenient and time consuming for the students.
* In continuation of the last point, this too is a burden on the university personnel; as they are often preoccupied with more important responsibilities.
* As the impact of Coronavirus permeated throughout the world, East West University too faced its fair share of disruption. On July 1st, 2020 East West University submitted to conducting online classes. This set out not to be a cut and dry process. As Coronavirus struck the financial state of the students, the applications for semester drop and financial help became more ubiquitous and a thing of greater importance. However, there was no system in motion for the officials to manage the aforementioned applications in an organized manner. As a result, the inbox of the officials became more cluttered with emails from students. This was, and still is, incredibly inconvenient for the involved parties.

Due to the aforementioned reasons, we do feel like there is a void of a management system that can be filled by our Token Management System.

**Challenges:**

* To get an accurate description of the required processes that need to be implemented, we will need to conduct a survey and require the opinions of the involved university employees, which is not feasible under these circumstances.
* Most of the members of our team are not used to working with Django framework. So, they have to go through a learning curve to acquire proficiency with these frameworks.
* In extension to the previous point, making Django, JS, and HTML interact with each other is difficult. None of us have worked with it before.
* We lack the provision of a management team or a client team that could provide us valuable feedback throughout the implementation process.
* We lack test subjects that could give us feedback about the usability of the application and point out the bugs.
* We have only 2.5 months to build a full-fledged application.
* As we do not have working experience in this field, ensuring the safety of the users’ information is a vital challenge.

**Component Diagram:**



**Non-functional Requirements:**

* The system should have the ability to adapt to changes, namely, new processes, or changes in the processes. Since the application will be used by non-technical users, not having this functionality would render the application pointless.
* The application should have easy to use, intuitive, and non-technical interface, so anyone can use it without a learning curve.
* The application should be able to search among thousands of tokens instantaneously.
* Since our application will deal with students’ personal information, it is essential to have strong security.
* The data of the users should not be used for any other purposes.

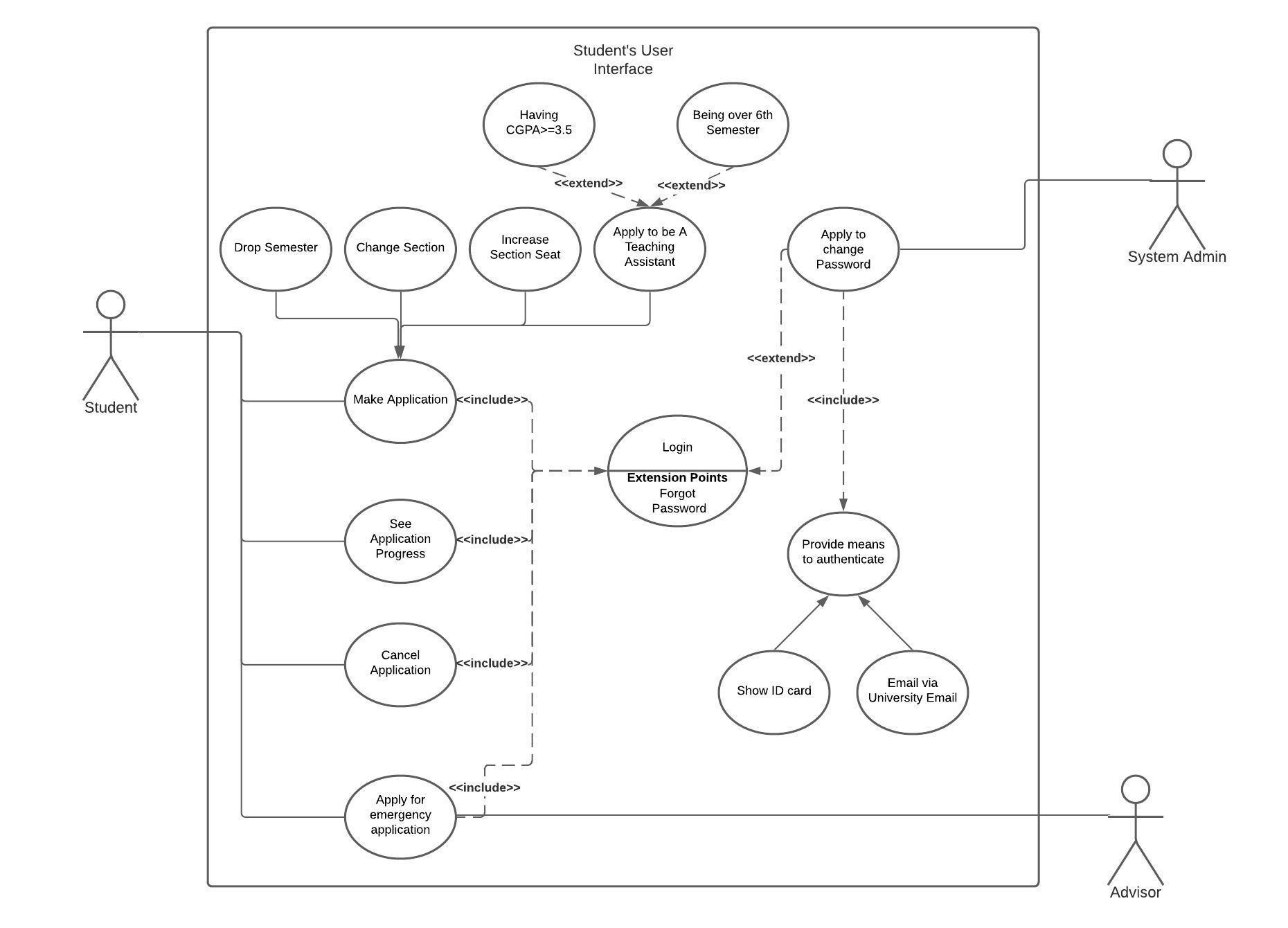
**Functional Requirements:**

* The system should have the ability to show the progress of a process.
  + Scenario: When a student is applying for a change of section, he will see when it reaches the advisor, the chairperson, and the course instructors.
  + Scenario: When a student is applying for a semester drop, he will be able to see whether it is under consideration or not.
* The system will have three levels of users:
  + System administrator
    - Input: username and password
    - Output: Search tokens, redirect tokens, change the course of a token, and add or delete new processes.
  + User from the organization
    - Input: username and password
    - Output: process tokens, redirect tokens, and reject a token.
  + User outside of the organization (students):
    - Input: username and password
    - Output: create a token, see the progress of a token.
  + Scenario: A student needs to change a section; so, he writes an application and the system automatically assigns a token with it. Throughout the entire process, the student will be able to see where his token resides.
* A user should not be able to create multiple requests for the same cause.
  + Scenario: A student requests a section change. After a few days, he gets impatient and tries to make another request in the hopes of a faster response. The system will retrain him from making such redundant requests.
* Users from the organization should be able to process a token and redirect it to another person as they see fit.
  + Scenario: An advisor sees a request, and notices that the chairperson is overwhelmed by an unprocessed token. So, the advisor may redirect the token to the course instructors.
  + Scenario: should an advisor deem a request invalid, they may discard it and the student will notice that their request is discarded.
  + Scenario: if an advisor feels like a request requires immediate attention, they may flag it as urgent and attach their comment with the token.
* System administrators should have the ability to add/modify/delete a process.
  + Scenario: The administrative bodies are considering providing financial support to students on request. A system administrator should be able to integrate this process into the system.
* Users from the organization should have the ability to undo a certain action.
  + Scenario: A user just marked a token as important and passed it to the next user. However, after considering the situation he feels like the person it was sent to is way too busy to handle a urgent important request. So, he may retrieve the token and redirect it.
* The system should have a strong authentication system in place. To do so, we aim to implement dual-layer security. Firstly, a user's password will be encrypted by Django and saved in the database and then upon log-in request, it would be authenticated by the Django server.
  + Scenario: When a user tries to submit improper data by changing the HTML code, it would be sent to Django. The passed information may not be proper in Django’s terms. It may discard it and notify the user.

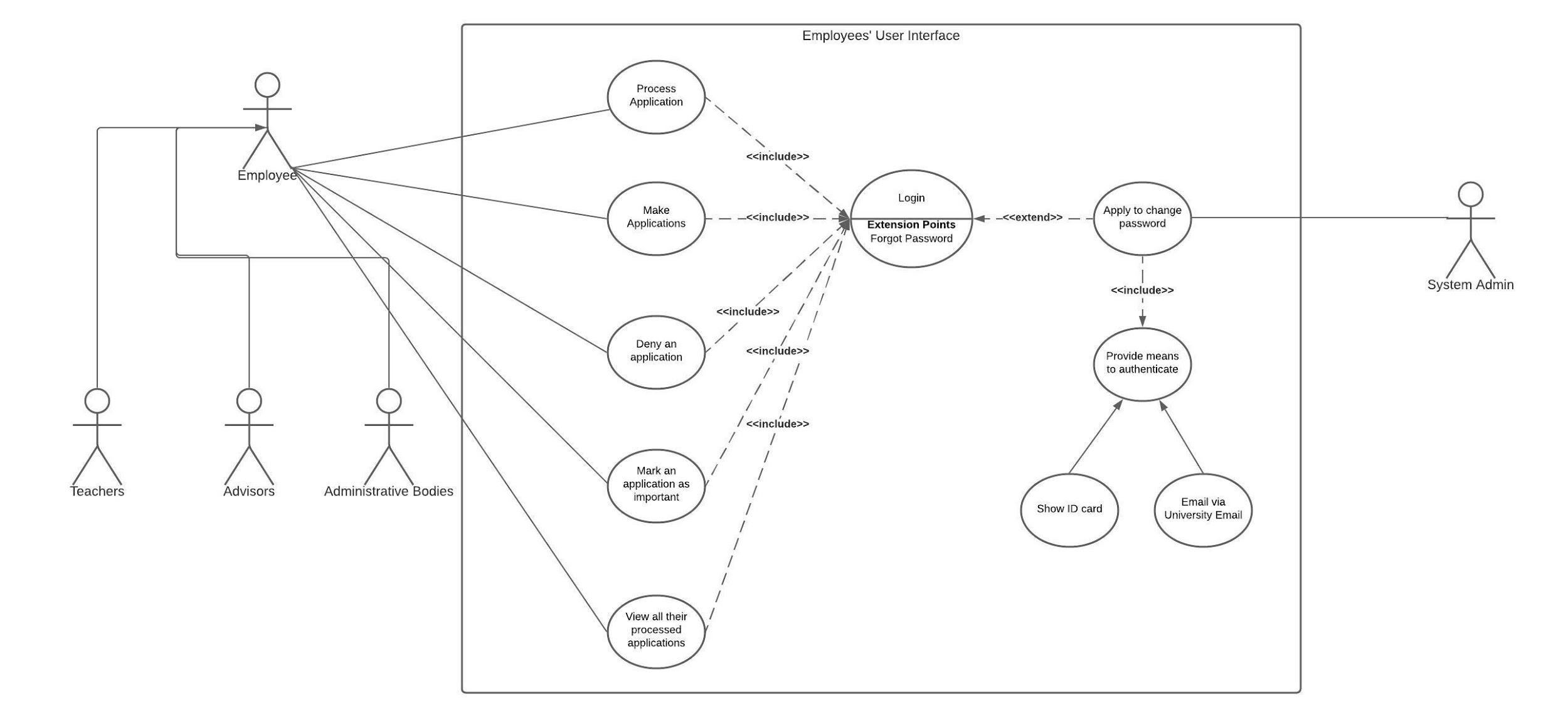
**UML Usecase Diagrams:**

Diagrams included:

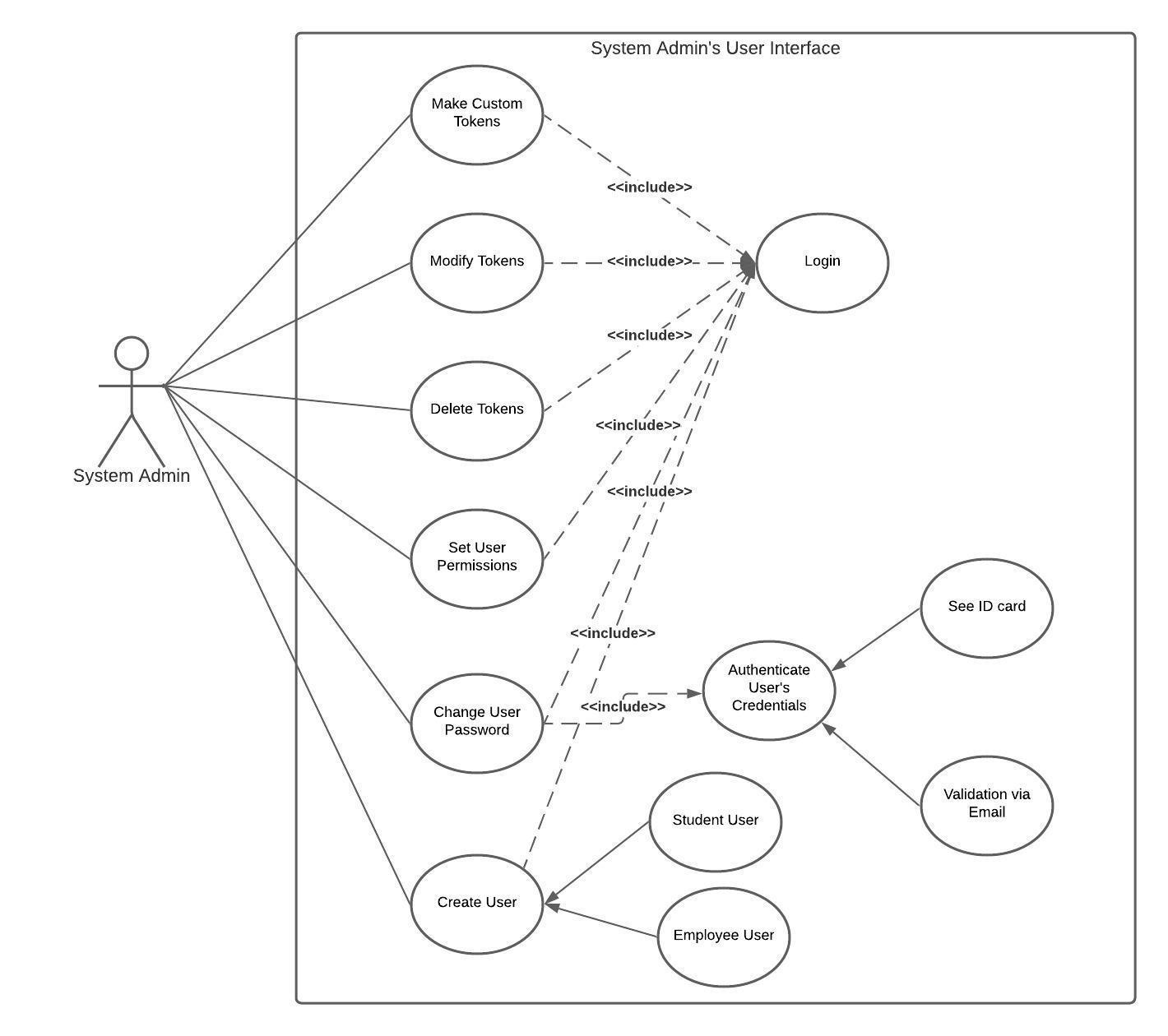
1. Students’ user interface
2. Employees’ user interface
3. System admins’ user interface



*Fig-2: Students’ user interface*

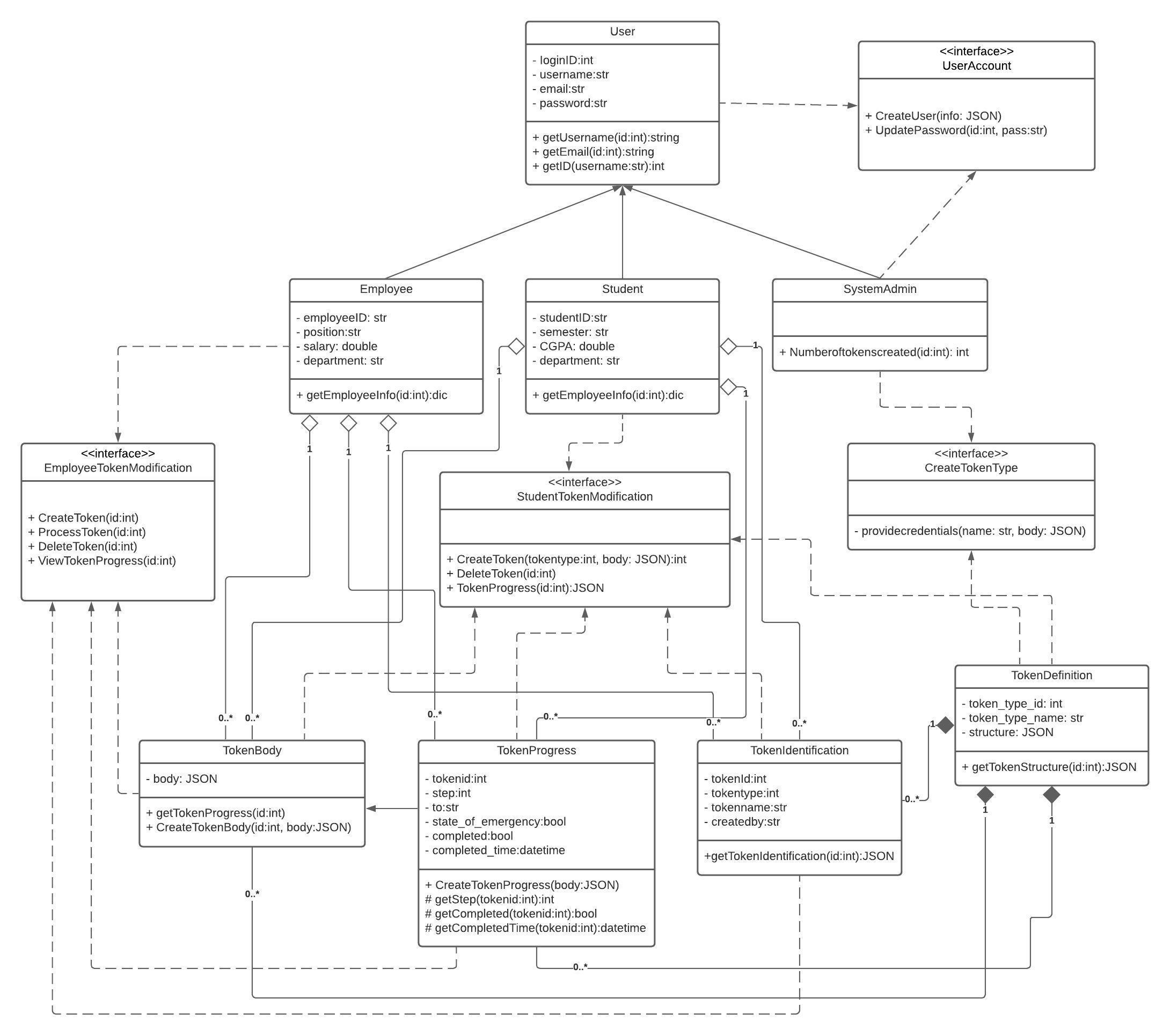


*Fig-3: Employees’ user interface*



*Fig-4: System admins’ user interface*

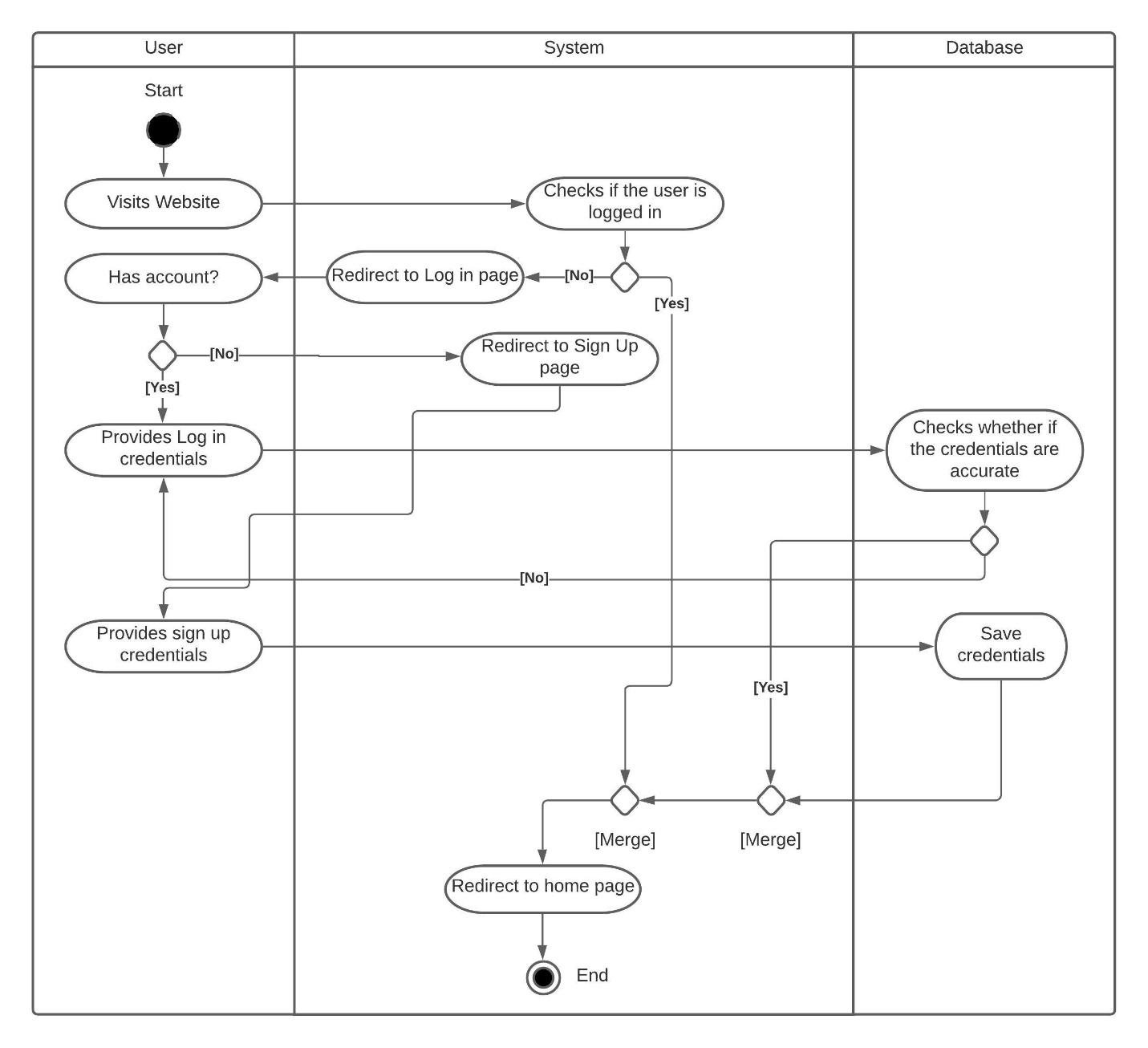
**UML Class Diagram:**

****

*Fig-5: Class Diagram*

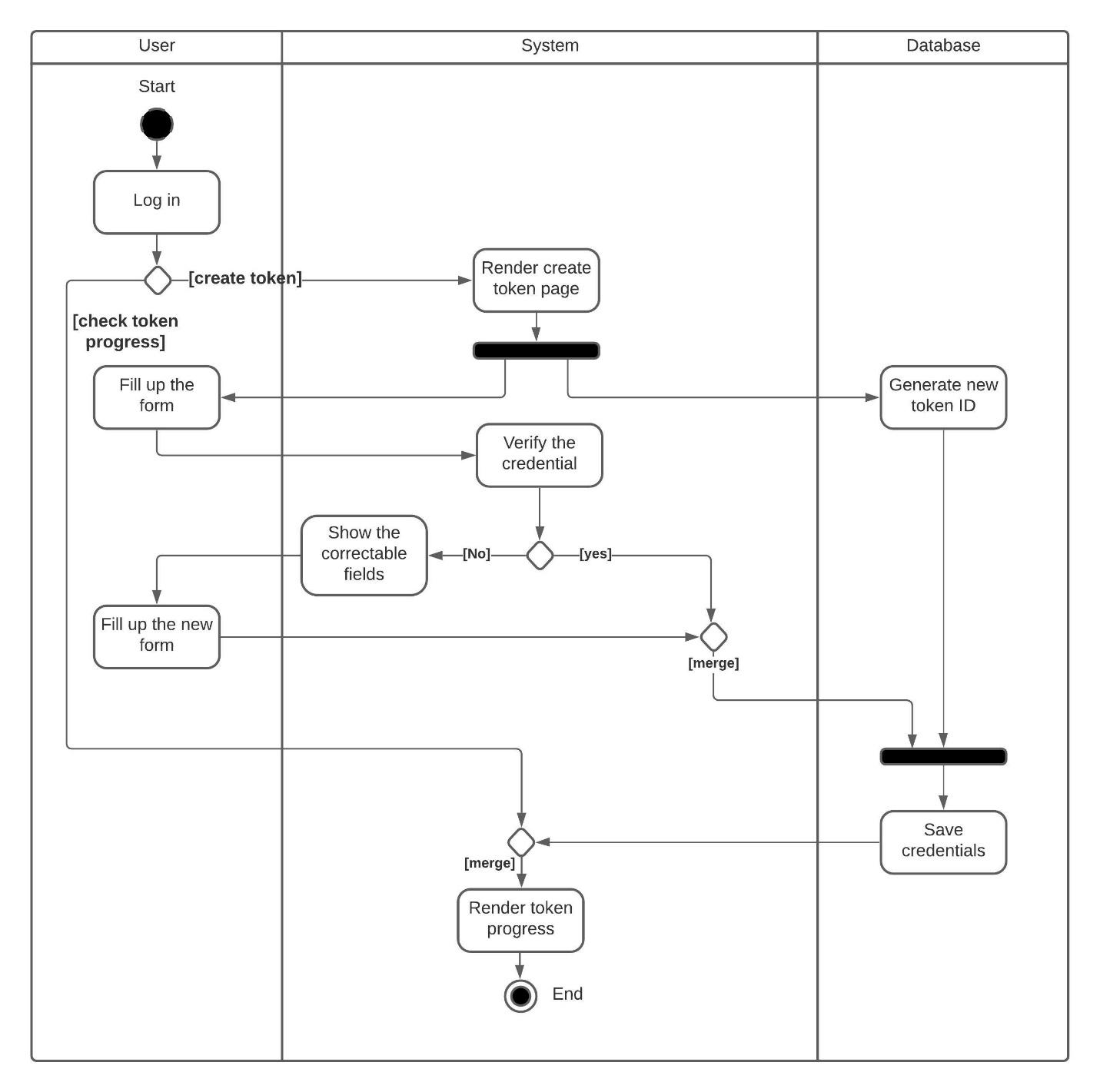
**UML Activity Diagram:**

Scenario: When a user visits our web application, the system checks whether the user is already logged in or not. If they are logged in, they are automatically directed to the home page of the application of their user type. If they are not logged in, the system redirects them back to the log-in page. If they have an account, they can provide credentials and log into the system. If they do not have an account, they are redirected to the sign-up page. Upon filling the form, if the system deems them to be appropriate, the system saves the credentials into the database and redirects them into their home page.



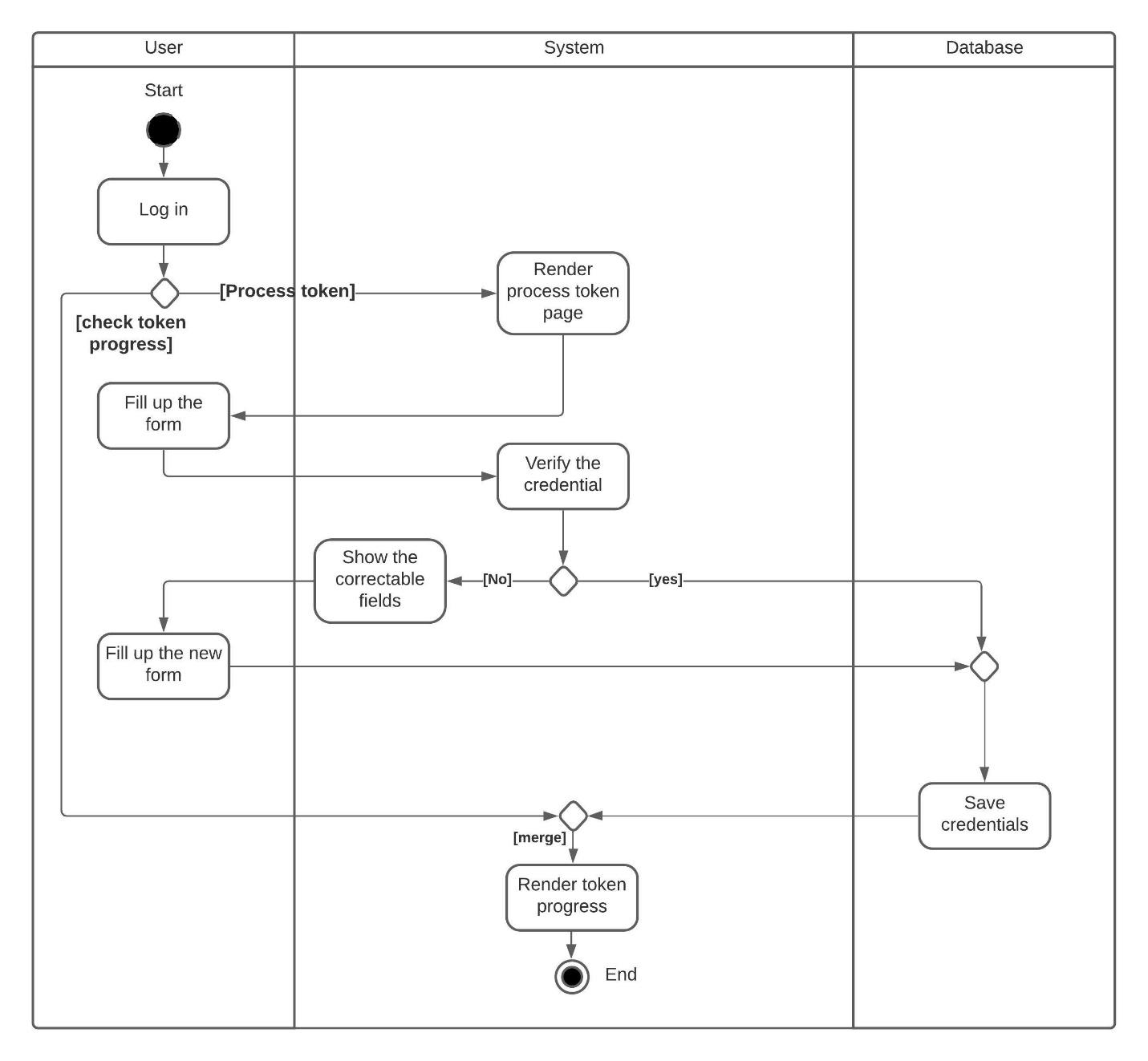
*Fig-6: Activity Diagram for Log in*

**Scenario:** When a student logs into the system, he has options to create a new token or check the progress of the previously created tokens. Furthermore, if he wants to check previously created tokens, the system will direct them to the token progress page. However, he wants to create a new token, he will be directed to create the token page. When he fills up the form and submits it, the system will verify the credentials. If the credentials are accurate, the system saves it into the database and redirects him to the token progress page. However, if the credentials are not up to the mark, the system renders a correctable field and asks the user to resubmit the form.



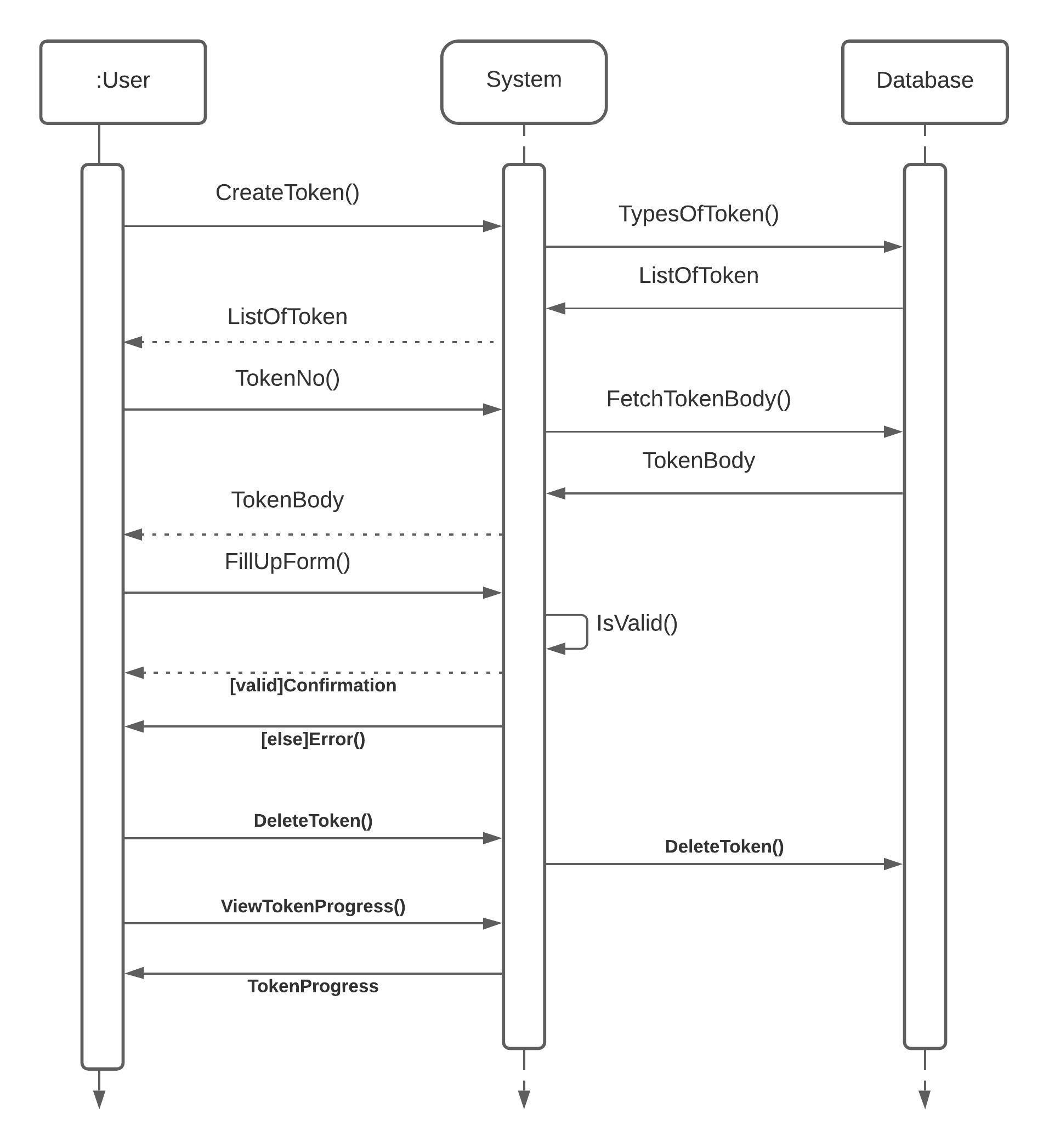
*Fig-7: Activity Diagram for students' user interface*

**Scenario:** When an employee visits our website, he is greeted with the option to process some tokens or check the progress of a processed token. If he chooses to check the progress of a previously processed token, the system will render the token progress page. Furthermore, if he chooses to process a token, the system will communicate with the database and return the appropriate form. Upon filling in the form, the system will check whether the credentials are following the token template. If it is, the system will save the response into the database. If there was a problem with the filled-up form, the system will render a correctable form for the user to fill up. Upon completion, the user will be redirected to the token progress page.

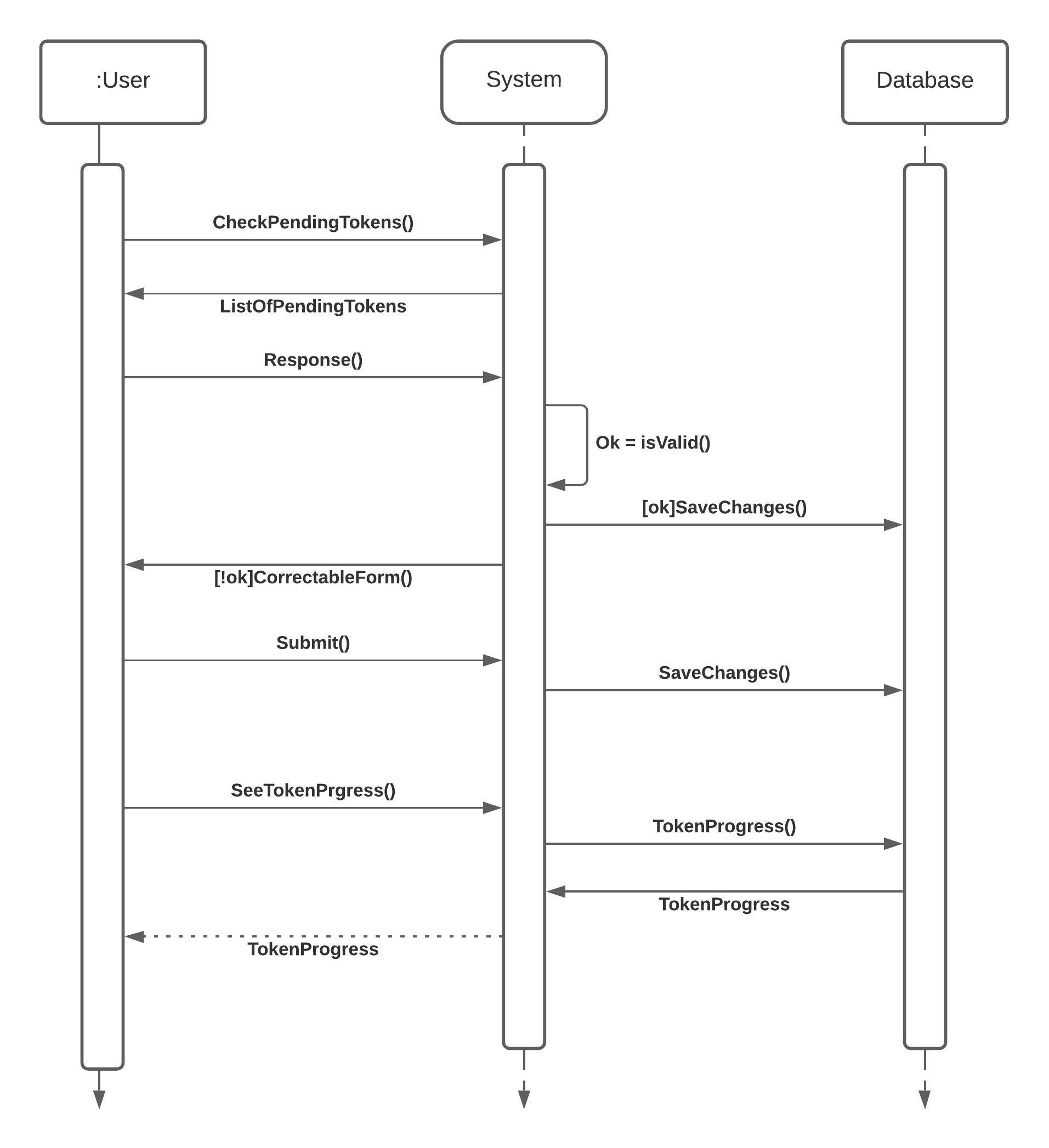


*Fig-8: Activity diagram for employees' user interface.*

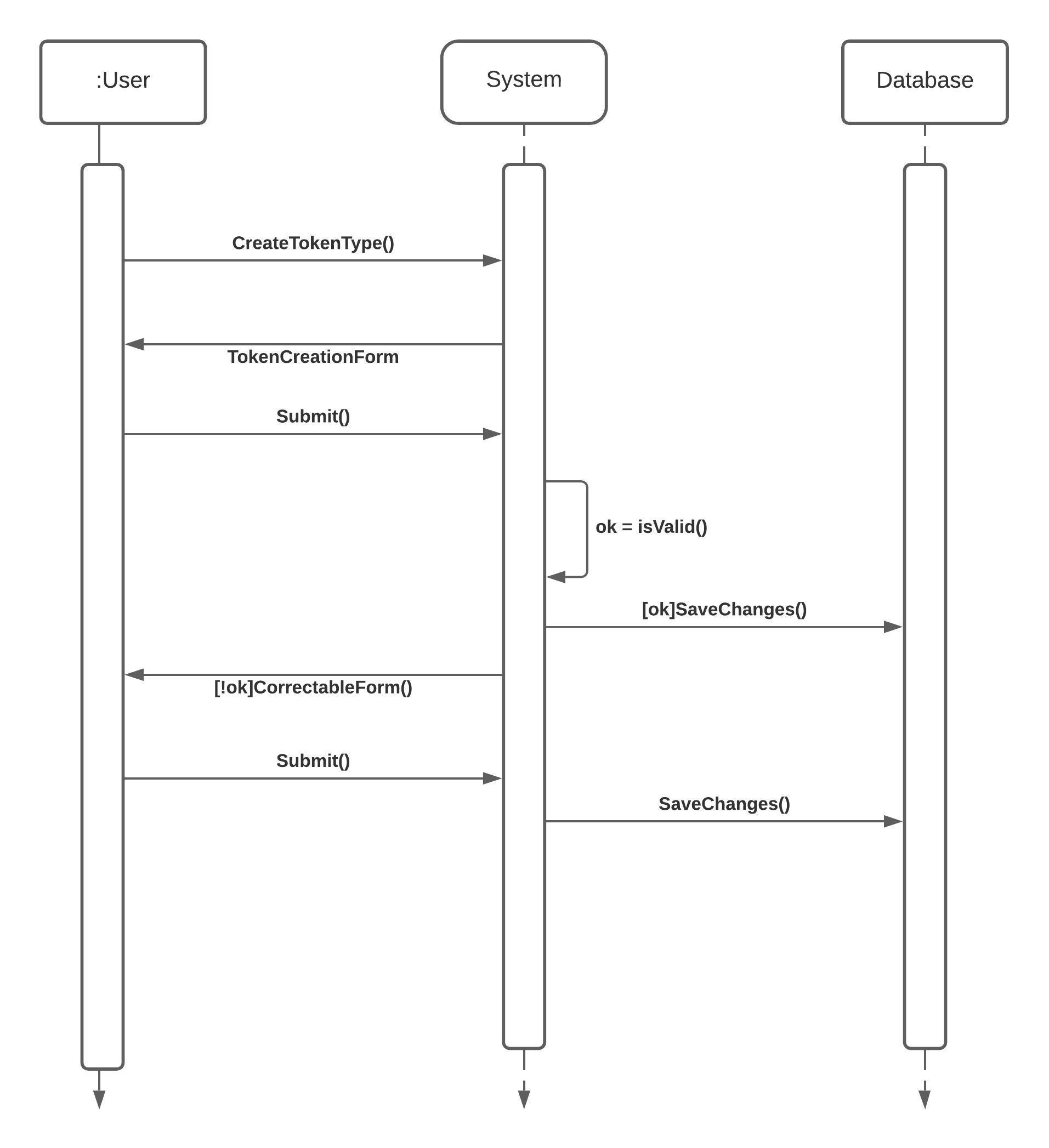
**UML Sequence Diagram:**



*Fig-9: Sequence diagram for user login.*

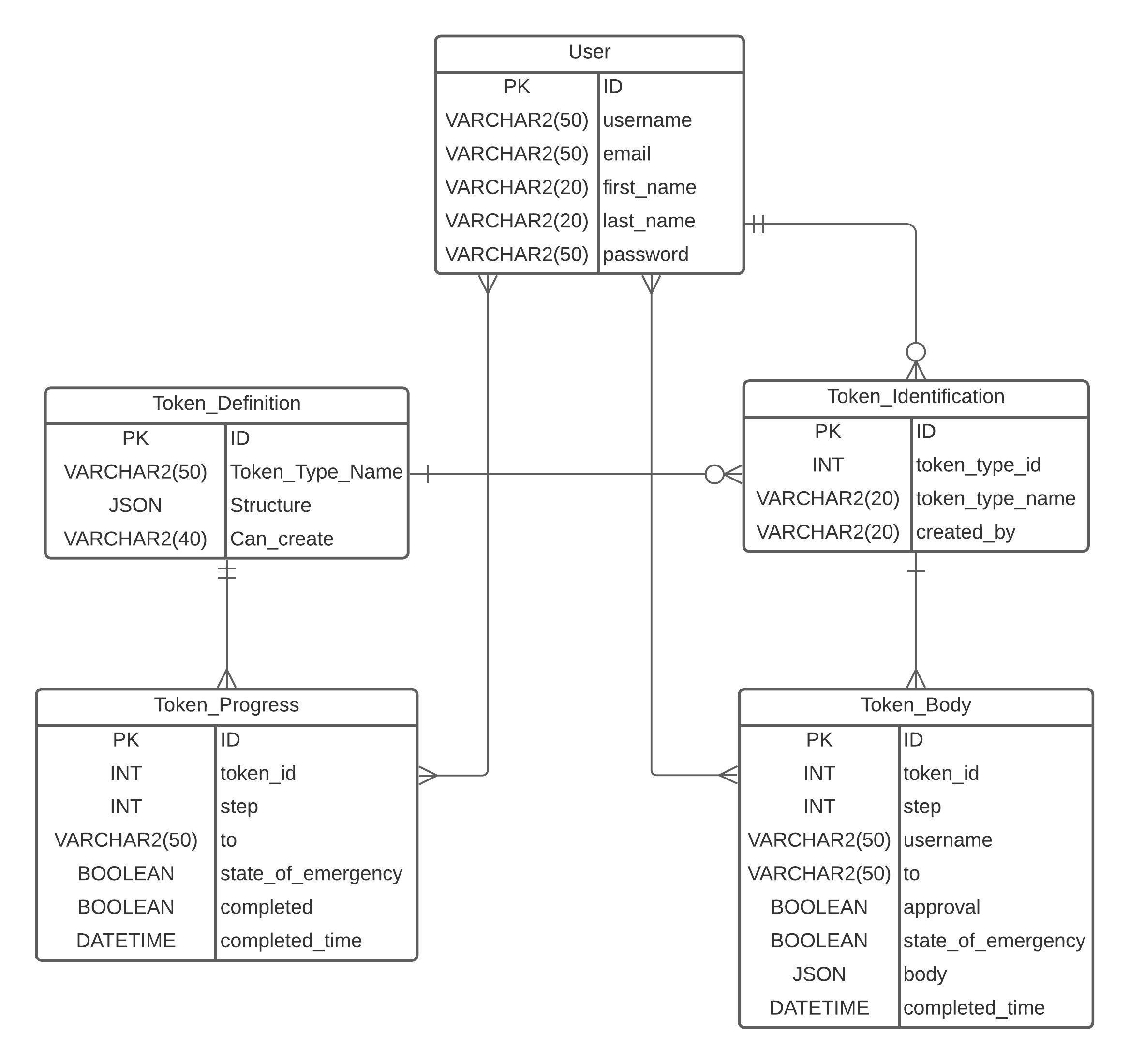


*Fig-10: Sequence diagram for employees' user interface.*



*Fig-11: Sequence diagram for system admins' user interface.*

**Entity Relationship Diagram:**

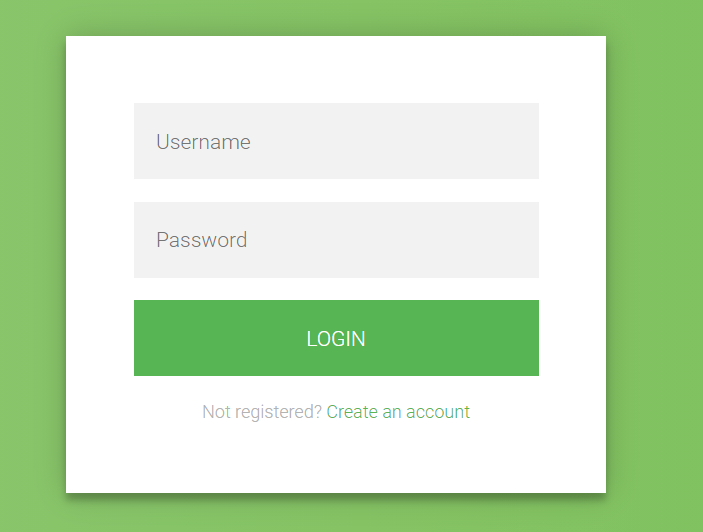


*Fig-12: ER diagram of the database.*

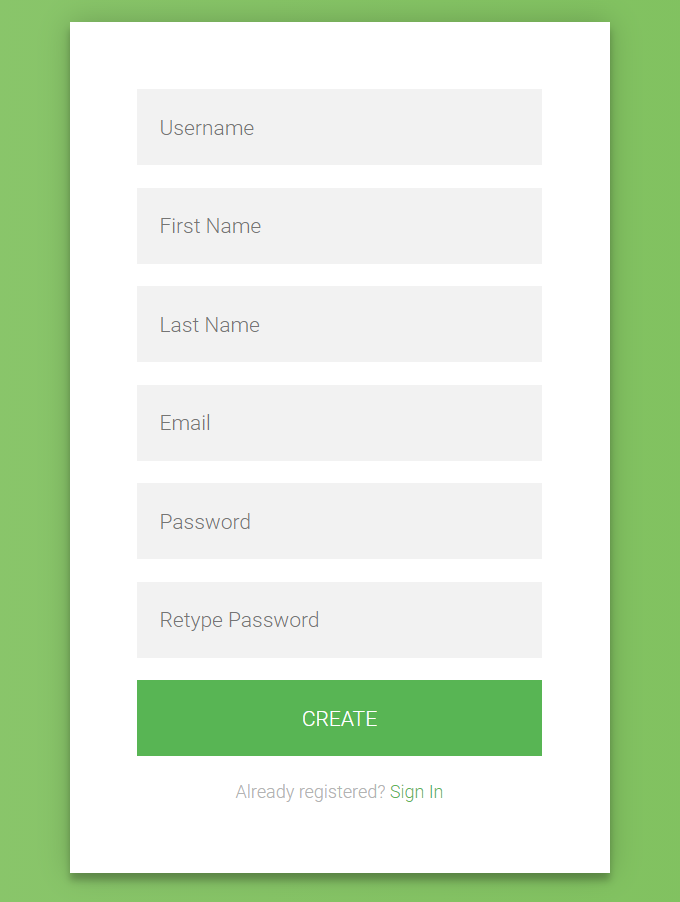
Conclusion: Regardless of the incredibly narrow time frame, we did manage to put together a full-fledged web application for token management. It is able to meet all the functional and non-functional requirements. However, due to time constraints, we failed to implement a few of the features that we thought would be useful. Moreover, our application does have minute bugs that we failed to solve due to time constraints. Furthermore, our application is not well tested in a real environment. So, testing it in a real environment may introduce unforeseen bugs.

**User Interface**

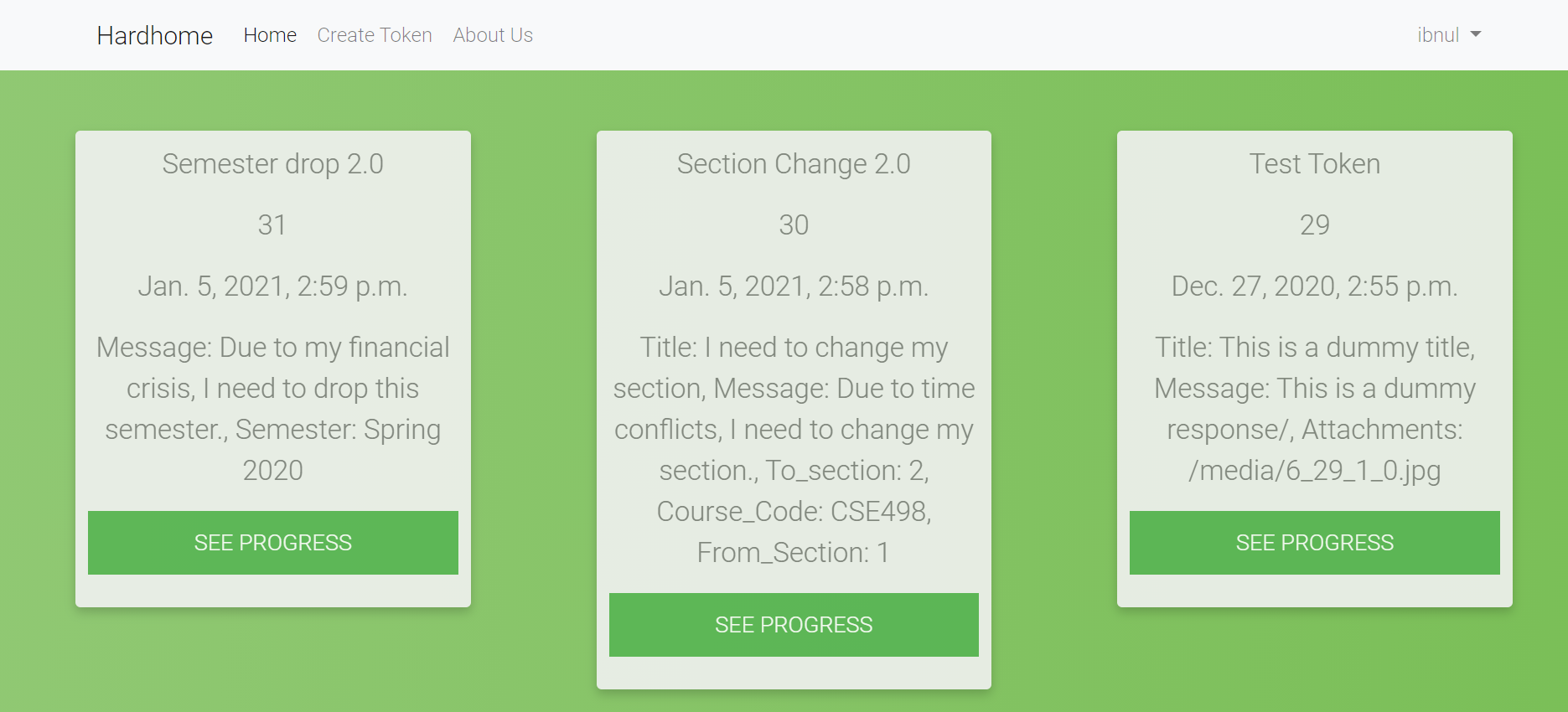
Log in interface:



Sign up interface:



Student home interface:



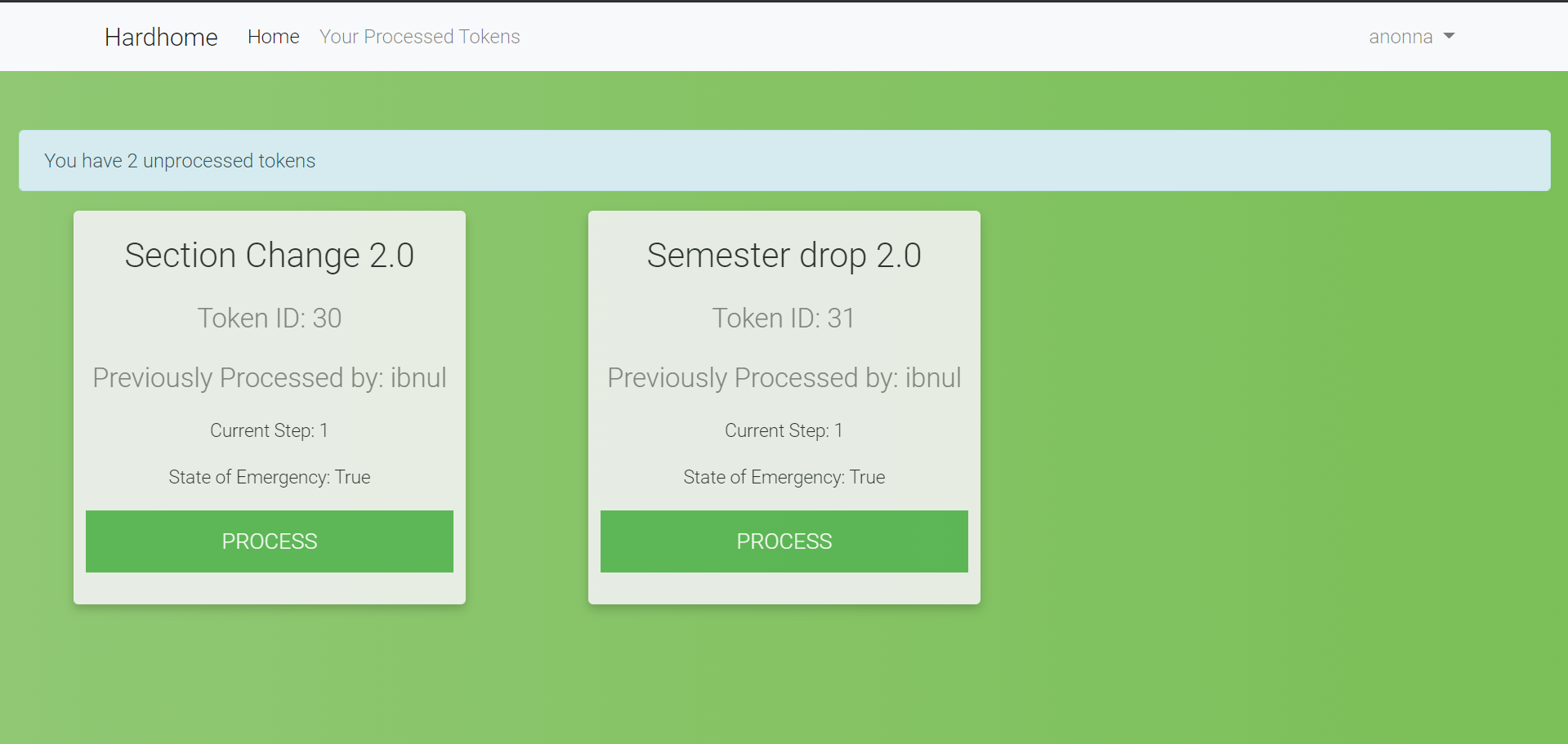
Students’ create token interface



Student’s token progress interface:



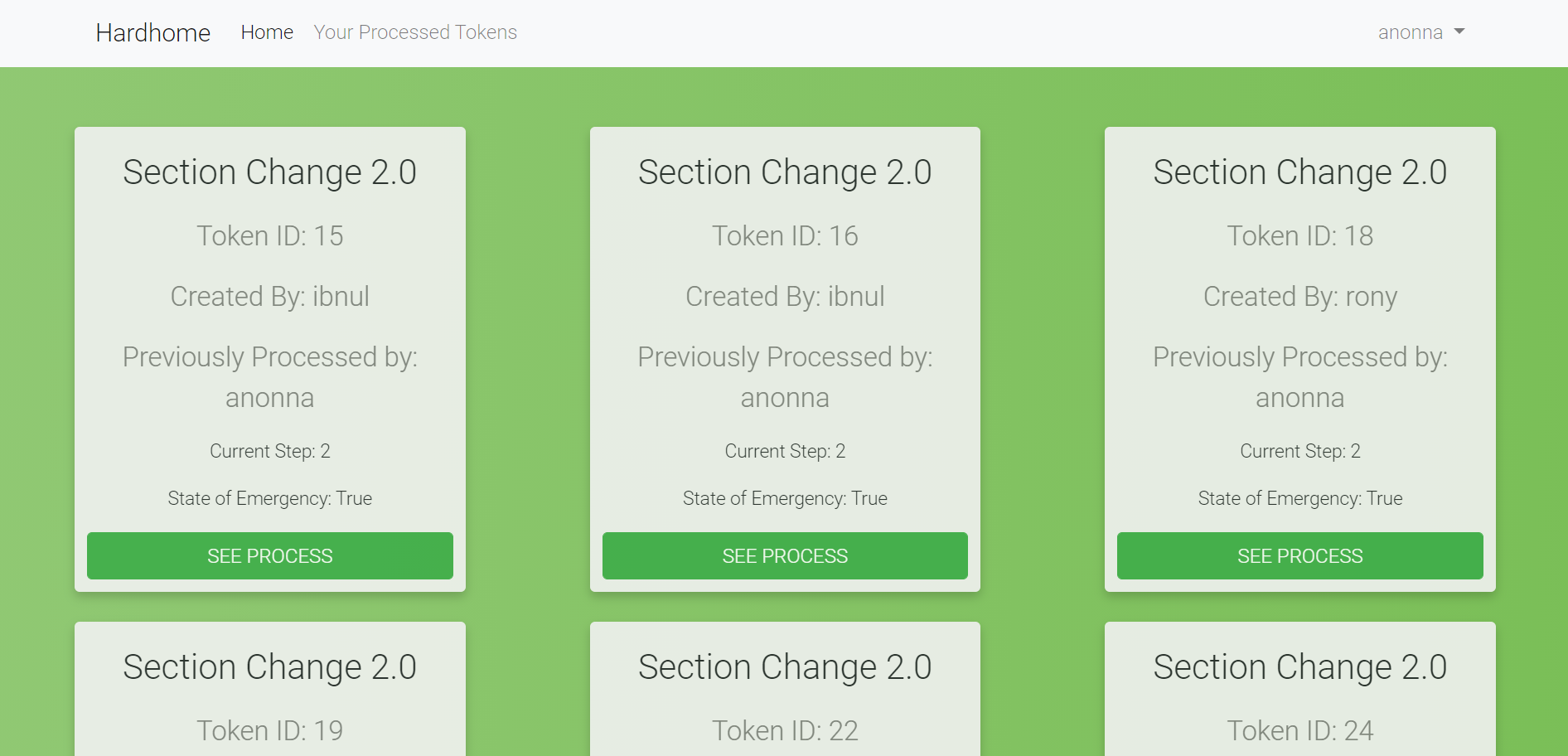
Employees’ home interface:



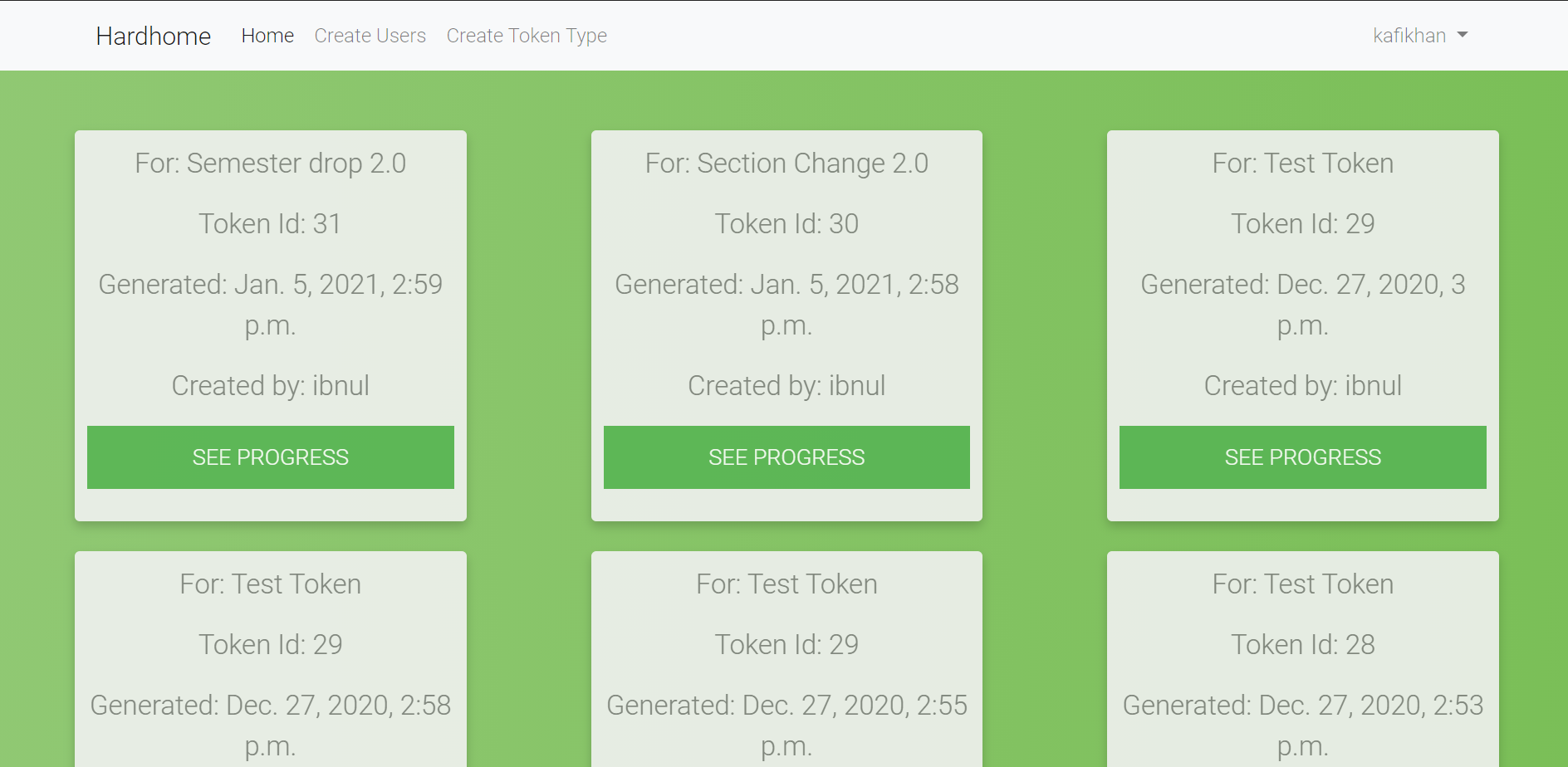
Employees’ process token interface:



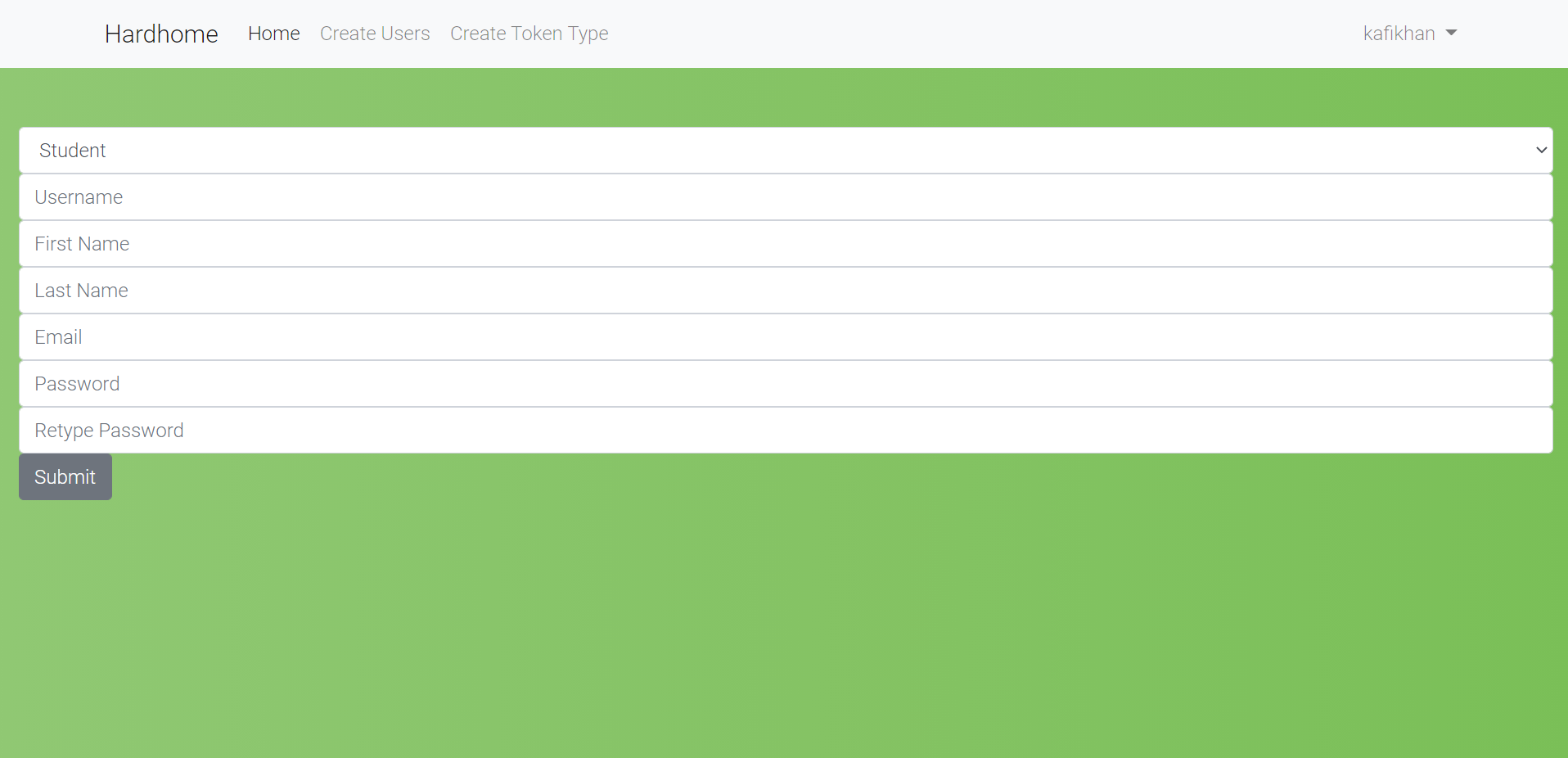
Employees’ processed token interface:



System admins home interface:



System admins create user interface:



System admins create token type interface:

