BILKENT UNIVERSITY
COMPUTER SCIENCE DEPARTMENT
CS 319 OBJECT-ORIENTED SOFTWARE ENGINEERING
DELIVERABLE 2 ITERATION 2
SPRING 2025
GROUP 2 SECTION 1
08.04.2025

Full Name	ID
Perhat Amanlyyev	22201007
Emiralp İlgen	22203114
Anıl Keskin	22201915
İlmay Taş	22201715
Simay Uygur	22203328

Contents

1. Non-Functional Requirements..... 3
1.1 Performance Requirements..... 3
1.2 Usability and Accessibility Requirements..... 3
1.3 Security Requirements..... 4
1.4 Reliability and Availability Requirements..... 4
1.5 Scalability and Maintainability Requirements..... 4
1.6 Interoperability and Integration Requirements..... 5
1.7 Data Integrity and Validation Requirements..... 5
1.8 Logging, Auditability, and Compliance Requirements..... 5
1.9 Notification System Requirements..... 6
1.10 Proctoring and Scheduling Requirements..... 6
1.11 Data Import and Export Requirements..... 6
2. Class Diagram..... 7
3. State Machine Diagrams..... 8
4. Activity Diagrams..... 10
5. Sequence Diagram..... 11
6. Mockups..... 13

1. Non-Functional Requirements

1.1 Performance Requirements

- The system shall ensure that critical operations (e.g., login, schedule viewing, data retrieval) are complete within **≤2 seconds** under standard load conditions (≤500 concurrent users).
- Chronological sorting, filtering, and dashboard rendering operations shall be completed within **≤2 seconds** for optimal usability.
- The system must scale to handle **up to 1000 concurrent users** during peak periods using **Tomcat connection pooling** and **Spring Boot microservices**.
- To ensure timely and efficient information delivery, real-time updates must be implemented using WebSockets or Server-Sent Events (SSE), enabling immediate communication between client and server. Immediate communication ensures fast, accurate, and reliable user interactions — making the system responsive, trustworthy, and resource-efficient.
- The system shall employ **lazy loading** and **pagination** for large datasets (e.g., notifications, logs). It leads to less data is transferred initially, benefiting users on:
    - faster page loads, because only subset of data is fetched initially;
    - better smooth-scrolling and on-demand loading preventing usage of large datasets;
- React frontend shall utilize **code splitting** (e.g., **React.lazy**, **Suspense**) to improve initial load times by loading only necessary components and deferring unnecessary components for future usages. This approach increases page speed, decreases time for interaction.

1.2 Usability and Accessibility Requirements

- The system shall feature a user-friendly **React-based UI** compatible with desktop and mobile devices.
- The user interface shall comply with **WCAG 2.1 AA accessibility standards**, including keyboard navigation support and sufficient contrast ratios.
- Session termination must securely clear all session data (e.g., cookies, local storage) to prevent unauthorized access.
- All dashboards and views shall be role-specific (e.g., TAs, instructors, admins) with consistent design principles.
- React state management (e.g., **Redux** or **Context API**) shall minimize full-page reloads, ensuring seamless user experience.

- Critical user workflows (e.g., approving overrides, submitting swap requests) shall be reachable within **≤3 clicks** from the dashboard.

1.3 Security Requirements

- The system shall implement **Role-Based Access Control (RBAC)** using **Spring Security** with **JWT tokens** stored in **HTTP-only cookies** to mitigate XSS vulnerabilities.
- **Two-Factor Authentication (2FA)** shall be mandatory for administrative users.
- Passwords shall comply with the organization's policy: minimum **12 characters**, with at least **1 uppercase letter**, **1 number**, and **1 special character**.
- Passwords shall be hashed using **bcrypt** with a minimum cost factor of **12**.
- All sensitive data (e.g., IBANs, schedules, exam details) shall be encrypted using **AES-256** both at rest and in transit (via **TLS 1.3**).
- CSRF protection shall be enforced on all relevant endpoints.
- Users shall be logged out after **15 minutes** of inactivity.
- API communications between frontend and backend shall be restricted to **HTTPS** and secure headers (e.g., **CORS**, **Strict-Transport-Security**).

1.4 Reliability and Availability Requirements

- The system shall maintain **≥99.9% uptime**, corresponding to a maximum downtime of **8.76 hours per year**.
- Backend services shall utilize **automatic failover mechanisms** (e.g., Kubernetes readiness/liveness probes).
- Daily **automated backups** shall be performed to enable rapid recovery.
- Bulk operations (e.g., TA imports, mass notifications) shall include **rollback mechanisms** to ensure data consistency and integrity.
- The frontend shall support **offline access** (where applicable) via service workers or **IndexedDB** for caching critical data.

1.5 Scalability and Maintainability Requirements

- The system shall be designed to support an increase from **500 to 2000 concurrent users** with minimal architectural changes.
- The backend shall implement **HikariCP connection pooling** to optimize database connections.
- Spring Boot services shall follow a **layered architecture (Controller-Service-Repository)** to facilitate maintainability and modularity.

- Frontend components (built with **TypeScript**) shall achieve **≥90% unit test coverage**.
- Backend services shall achieve **≥80% test coverage** for unit and integration tests.
- All APIs shall comply with **OpenAPI 3.0** for ease of integration and future extensibility.

1.6 Interoperability and Integration Requirements

- The system shall integrate with the university's **Single Sign-On (SSO)** via **OAuth2**.
- The system shall interoperate with the university's **SMTP servers** (using **STARTTLS**) for sending transactional emails (e.g., password resets, notifications).
- The backend shall expose **RESTful APIs** adhering to **OpenAPI 3.0** standards.
- External systems and future modules shall be able to communicate via these standardized APIs.

1.7 Data Integrity and Validation Requirements

- All user input forms shall be subject to completeness checks and validation before submission.
- IBAN fields shall be validated using standard checksum algorithms.
- The system shall validate assignment inputs against schedule conflicts in under 2 second to ensure timely processing.
- Optimistic locking techniques shall be used to prevent race conditions in TA assignment workflows.
- Data imports (e.g., bulk TA lists) shall undergo **schema validation** and reject invalid entries.

1.8 Logging, Auditability, and Compliance Requirements

- All critical actions (e.g., swap requests, approvals, leave submissions) shall be logged with metadata such as **timestamp**, **user ID**, and **IP address**.
- Logs shall be retained for a minimum of **7 years** to comply with audit and regulatory requirements.
- Log data shall be centralized using **Logback** or integrated with an external system (e.g., **ELK stack**).

1.9 Notification System Requirements

- Notifications shall be delivered in real-time or near real-time using **WebSockets** or **SSE**.
- The Dean's Office shall be notified immediately of critical events (e.g., override approvals).
- Notification lists shall support pagination and filtering.
- Swap and override request responses must generate automatic notifications to relevant TAs and instructors.

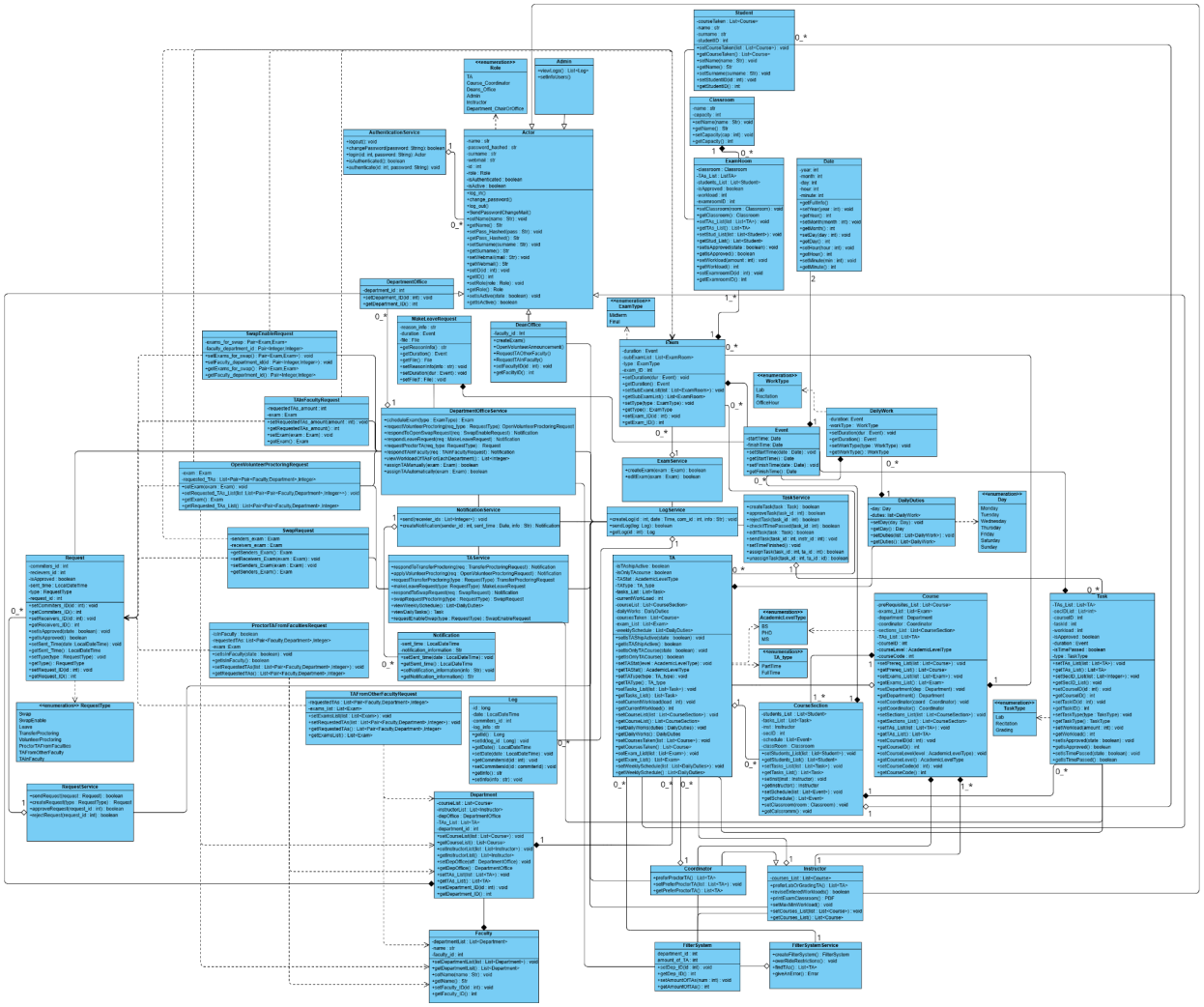
1.10 Proctoring and Scheduling Requirements

- Proctoring, swap, and override requests must be submitted no later than **6 hours** before the exam.
- The system shall enforce non-overlapping exam times and locations at both application and database levels.
- The system shall maintain consistency between leave data and assignment records, ensuring synchronization within 5 seconds of an update.
- The system shall provide real-time dashboard access with data refresh intervals not exceeding 3 seconds.

1.11 Data Import and Export Requirements

- The system shall complete bulk import operations of up to 10,000 records within 10 seconds.
- Data imports must validate schema compliance and reject invalid or incomplete entries.
- Rollback functionality shall be available for bulk operations affecting critical datasets.

2. Class Diagram



Link for the [Class Diagram]

3. State Machine Diagrams

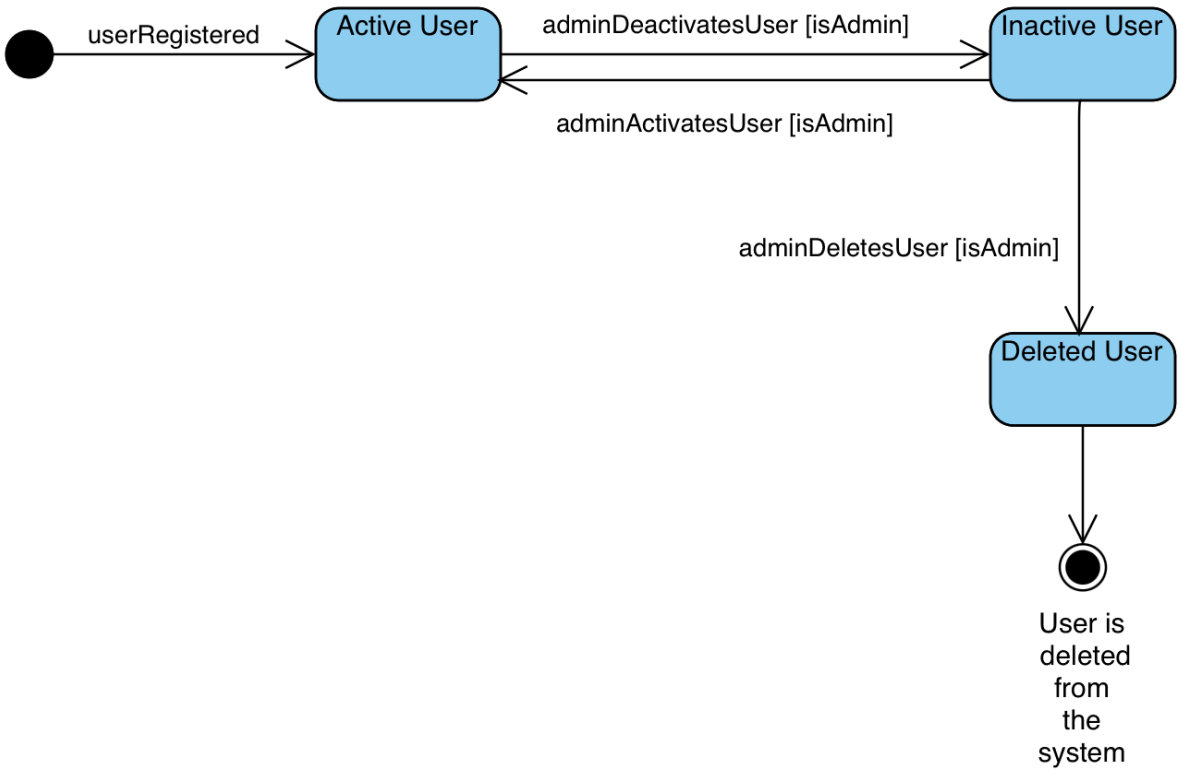


Diagram 1: State Machine Diagram of a User

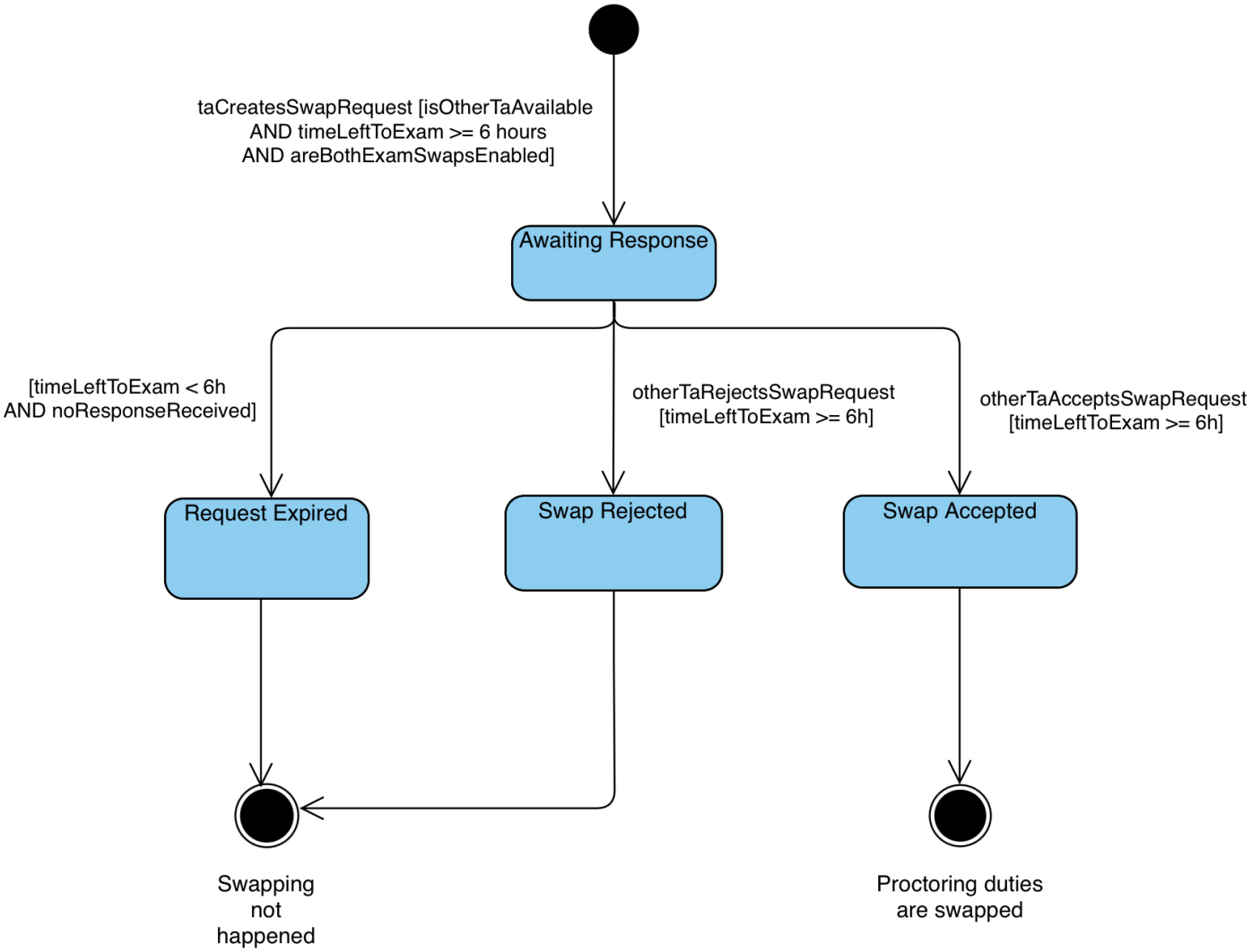


Diagram 2: State Machine Diagram of a Proctoring Swap Request between Teaching Assistants

Link for all of the state machine diagrams: [State Diagrams]

4. Activity Diagrams

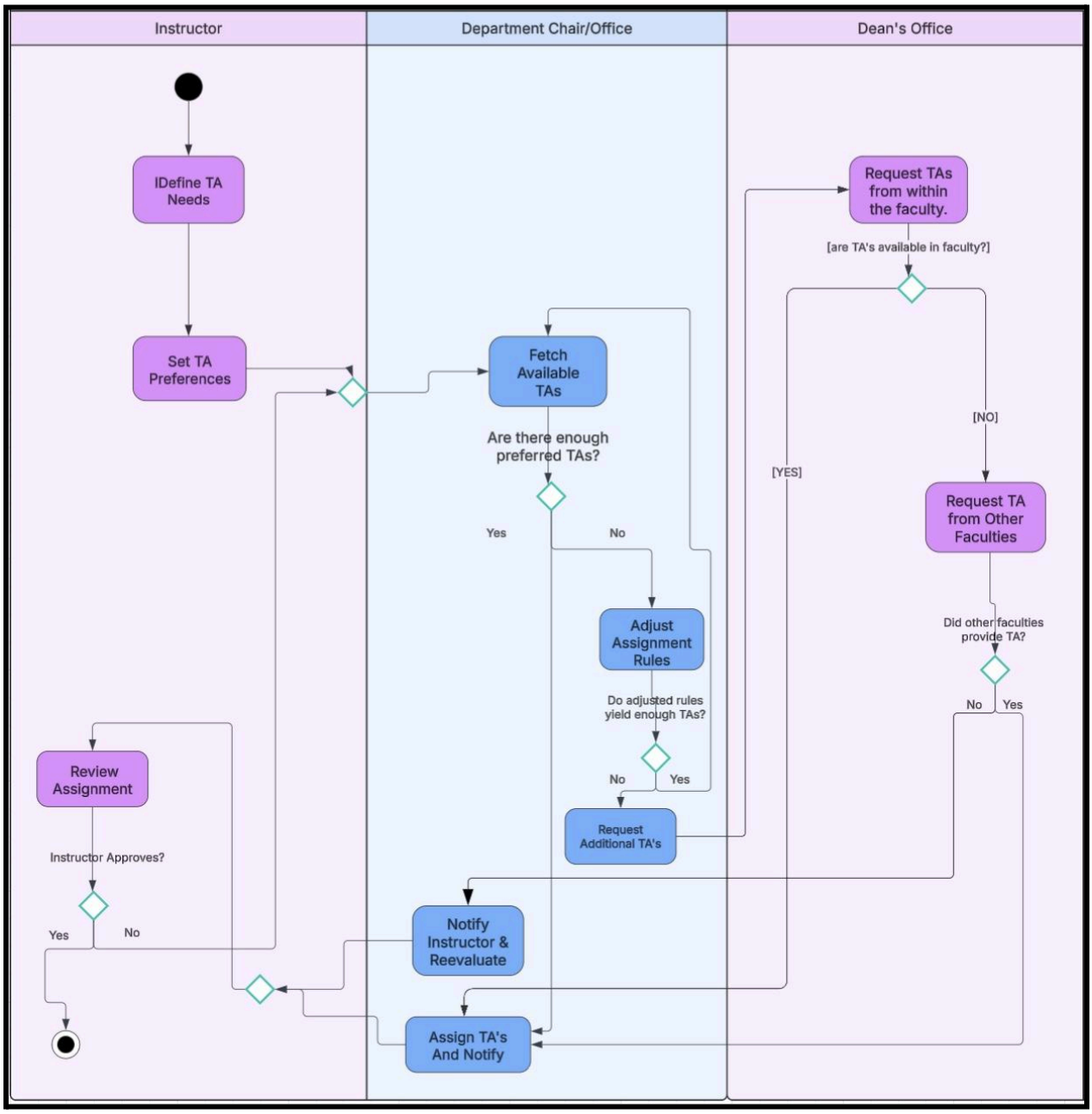


Diagram 1: The activity diagram for automatically assigning the TA process

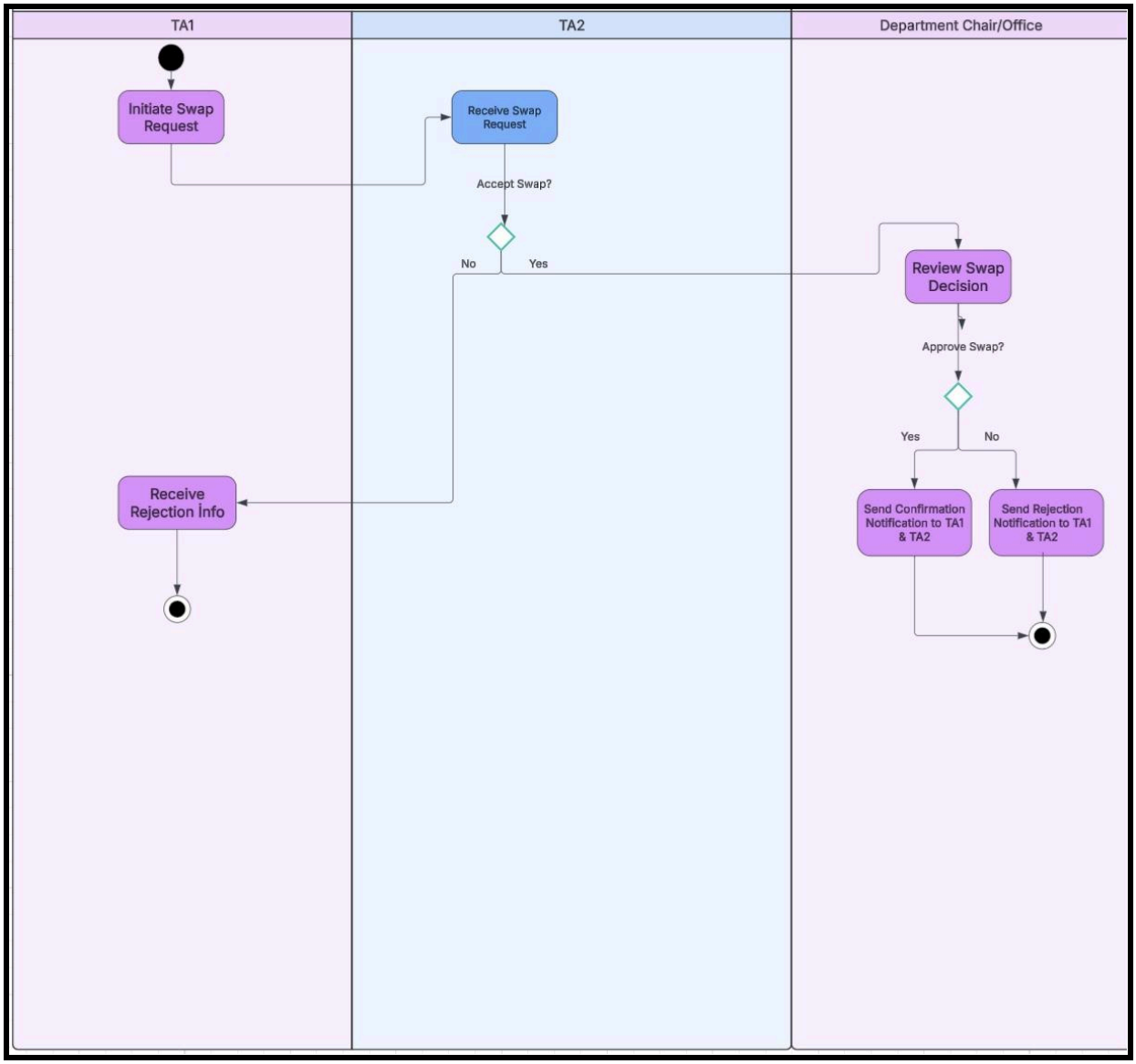


Diagram 2: Activity Diagram for TA-to-TA Swap Request with Department Chair Approval

Link for Activity Diagrams : [Activity Diagrams]

5. Sequence Diagram

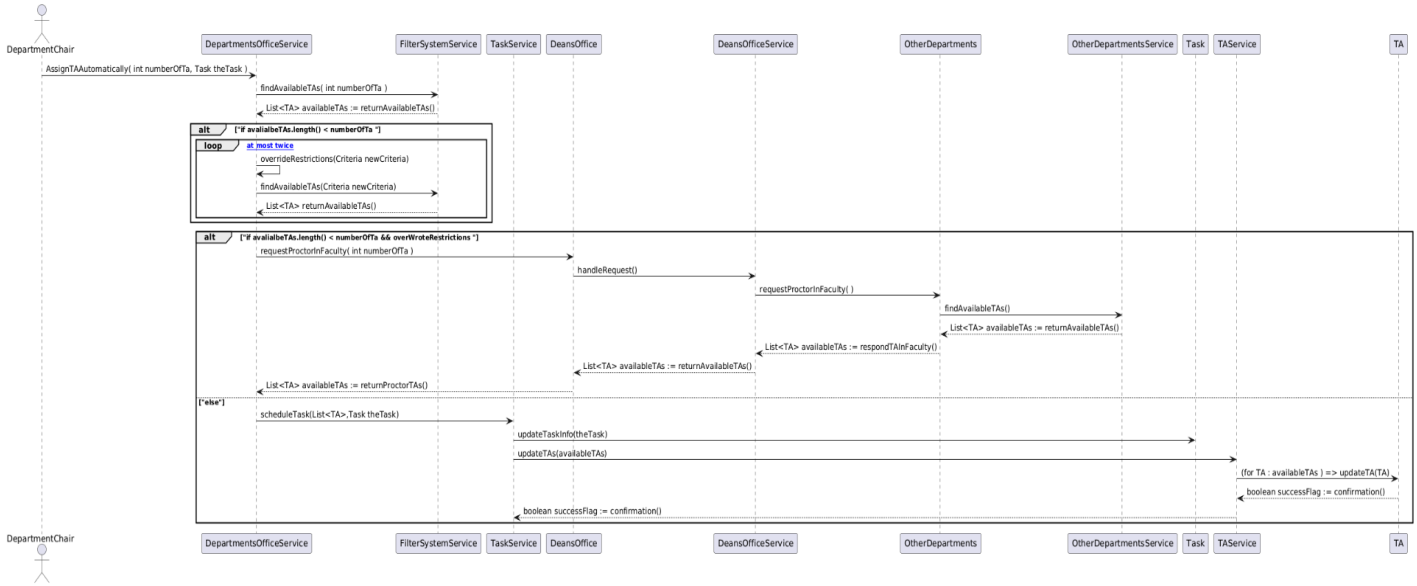


Diagram 1 : Sequence diagram of AssignTAAutomatically
Link for Sequence Diagram: [Sequence Diagram]

6. Mockups

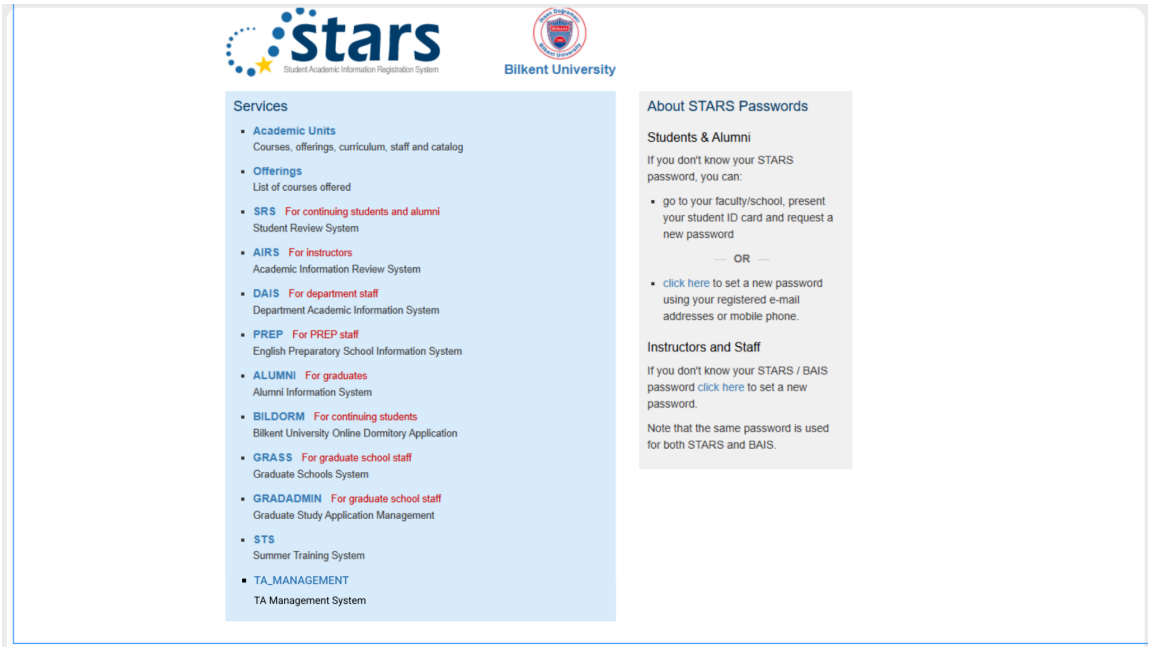


Figure 1: First Screen

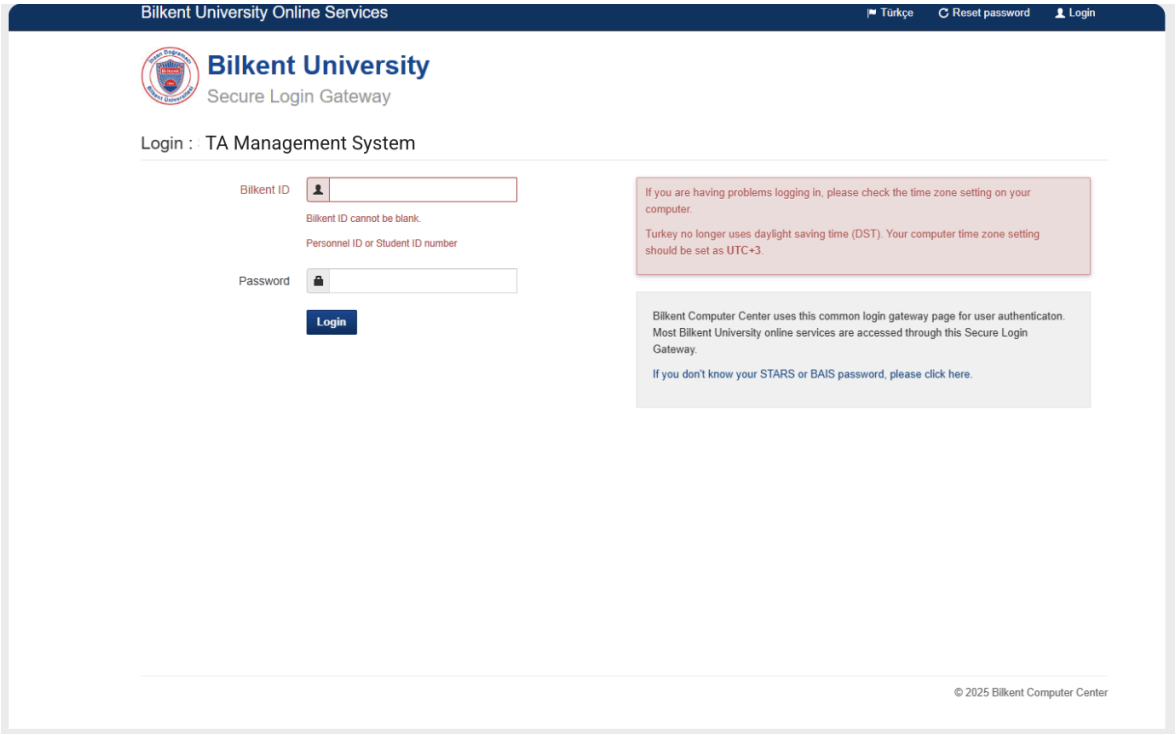


Figure 2: Login Page of the TA Management System

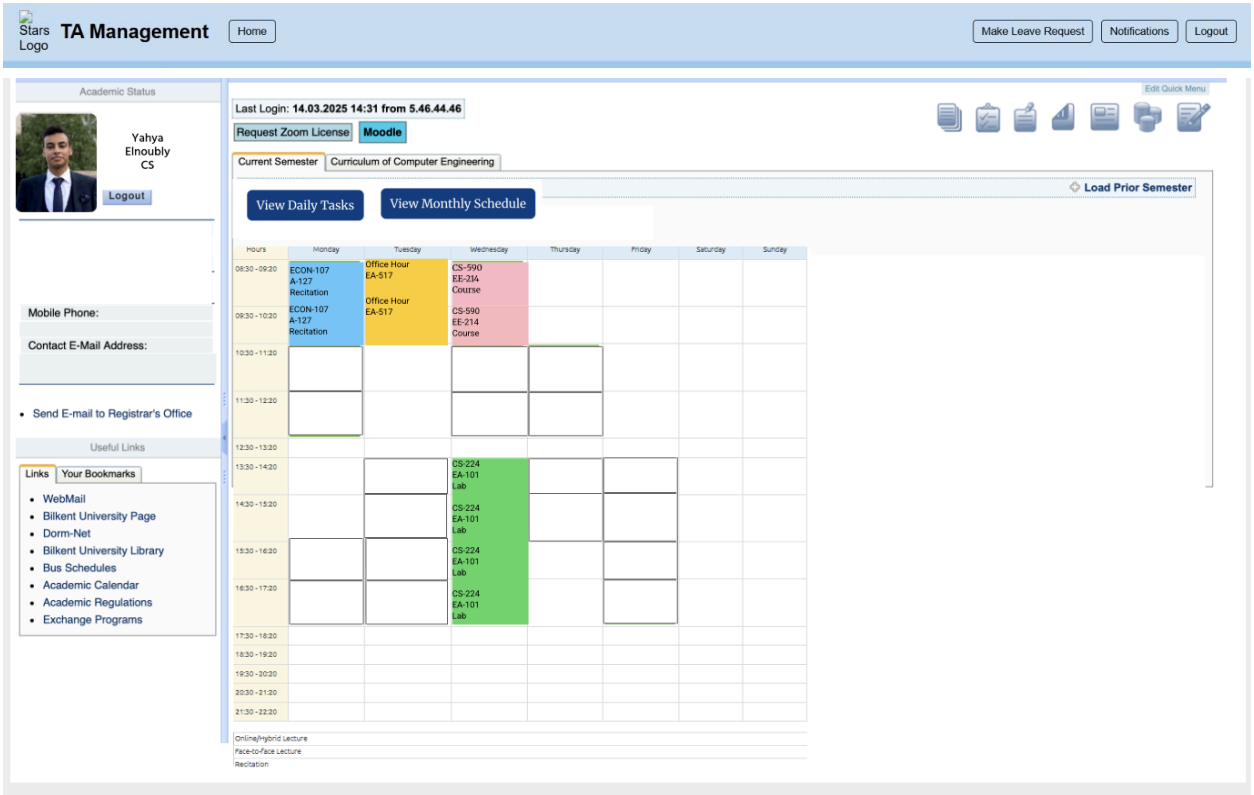


Figure 3: TA Main Page

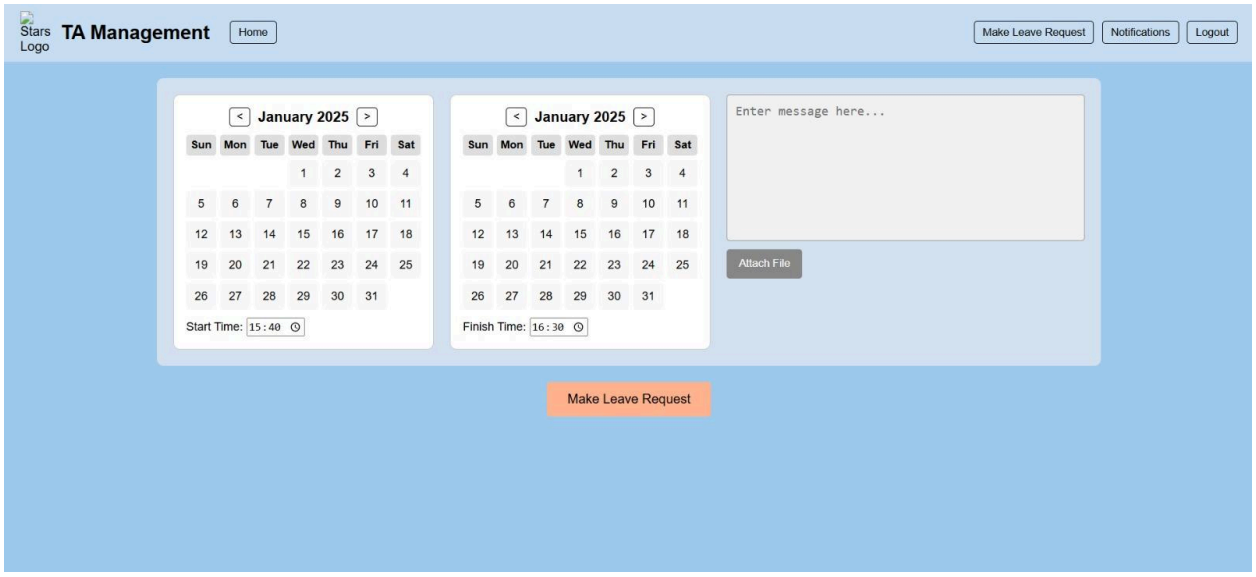


Figure 4: Leave Request Form Page of TA

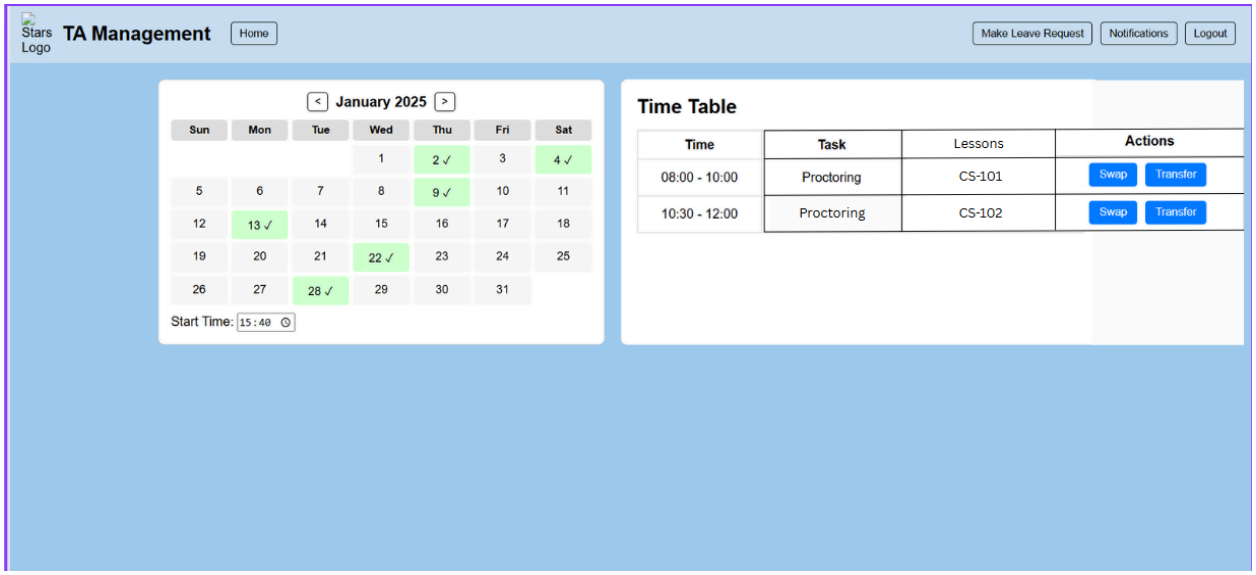


Figure 5: Monthly Schedule of TA

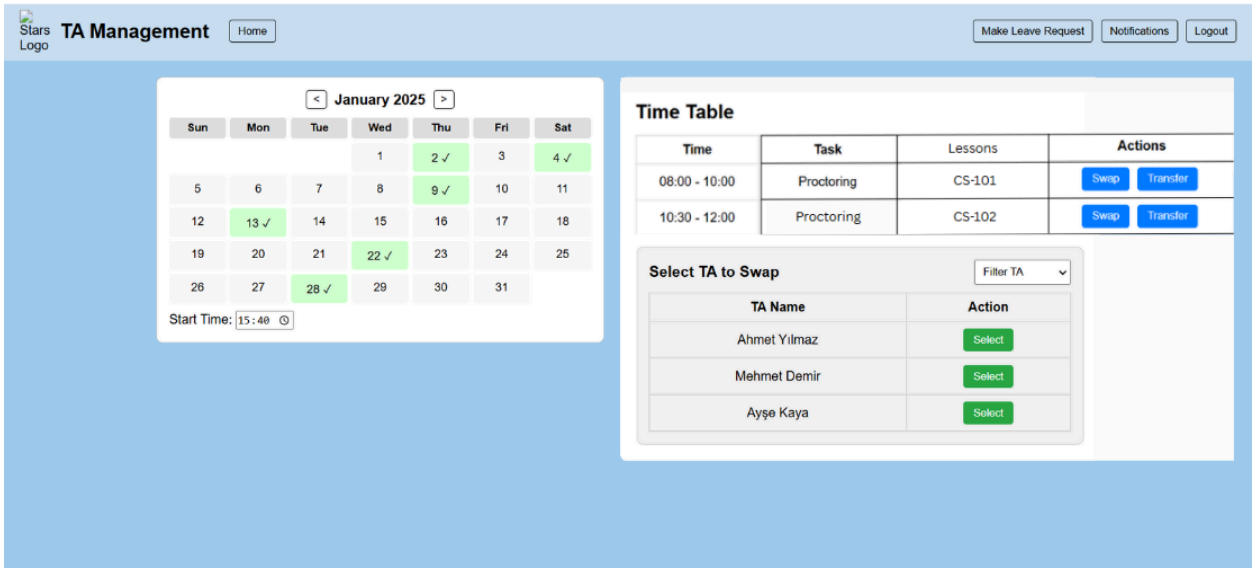


Figure 6: Monthly Schedule of TA

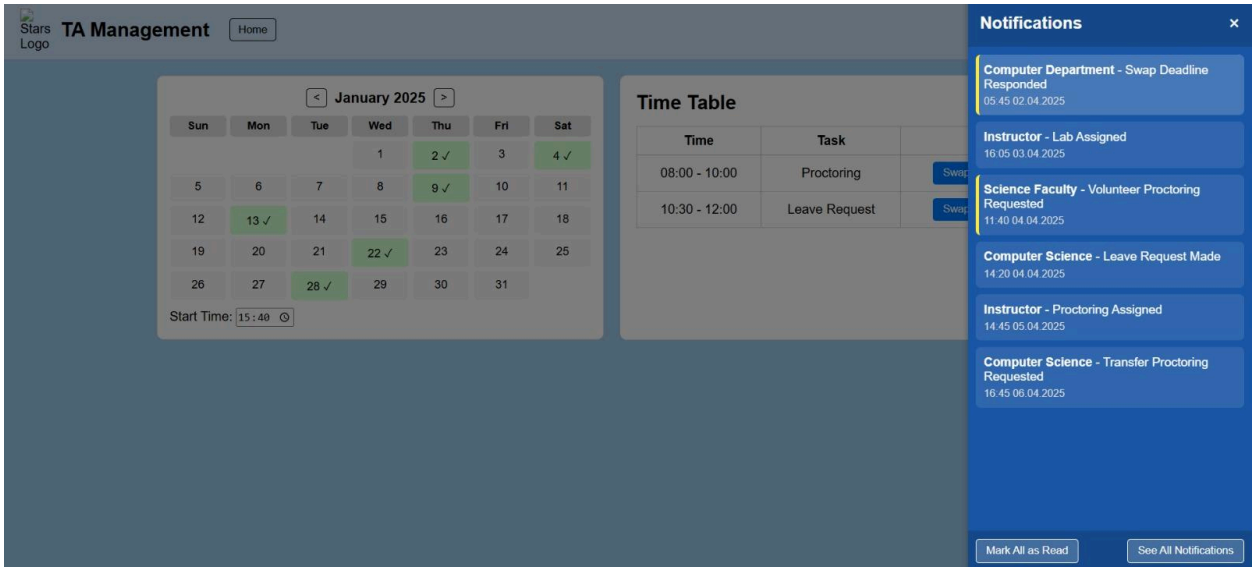


Figure 7: TA Notification Box

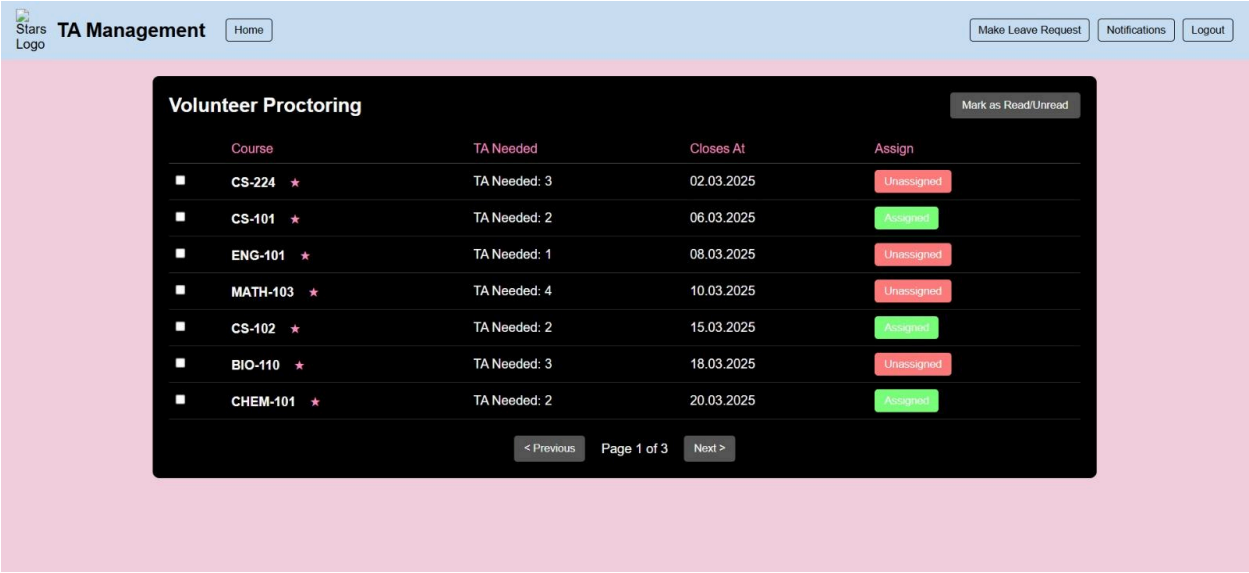


Figure 8: Volunteer Proctoring Announcements List for TA

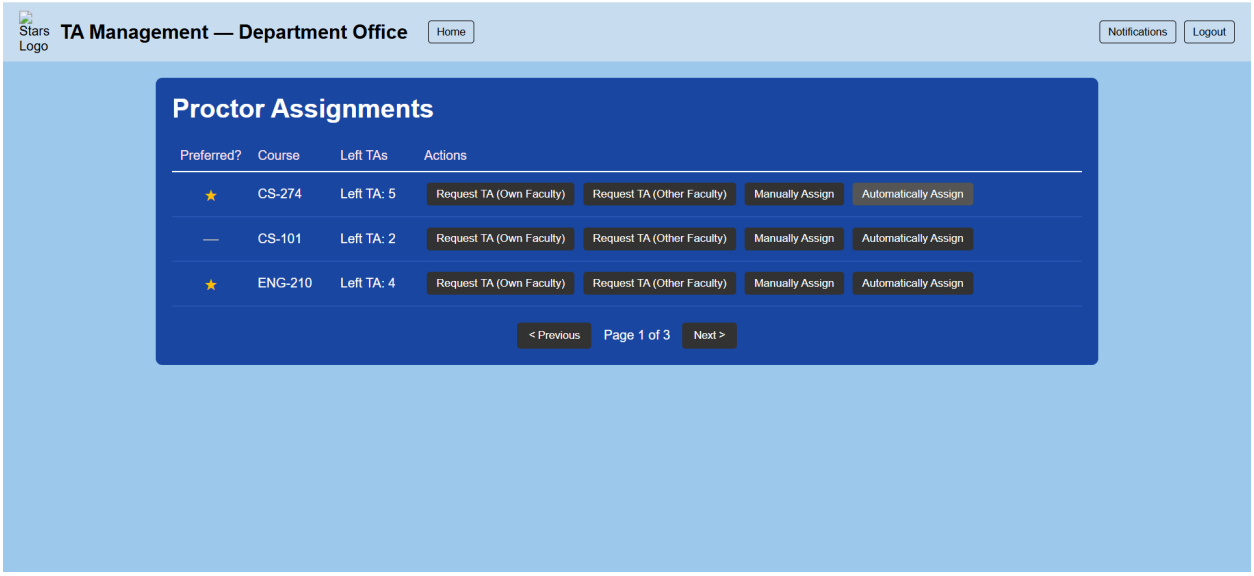


Figure 9: Proctoring Assignments List Page for the Department's Office

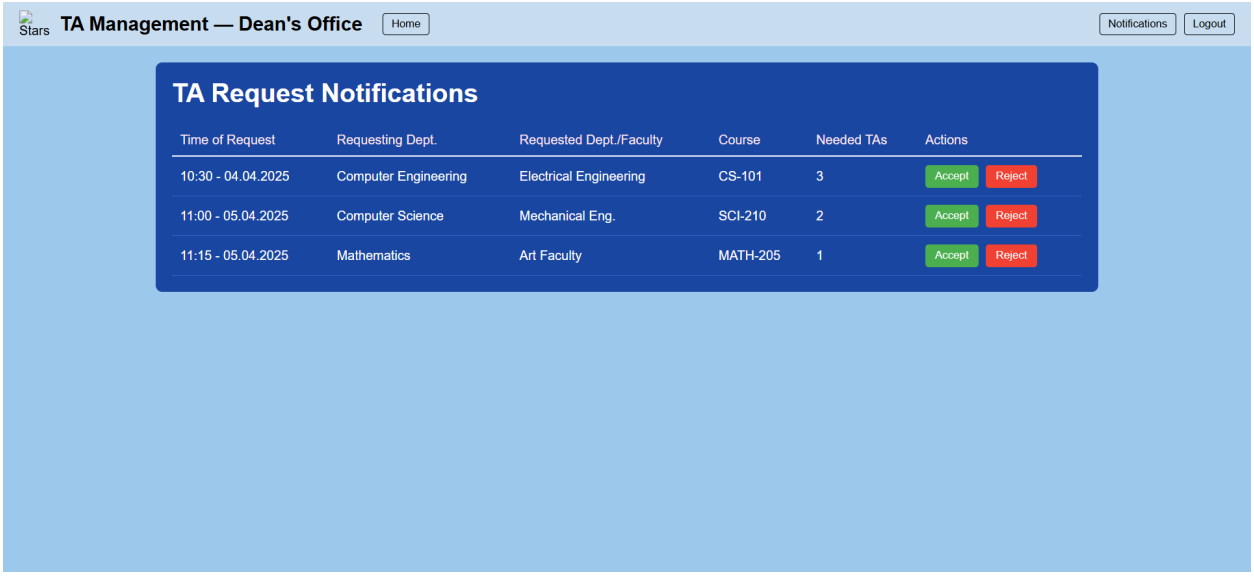


Figure 10: All TA Proctoring Requests by Departments' Offices (TA from Other Faculty and TA from Own Faculty) List Page for the Dean's Office

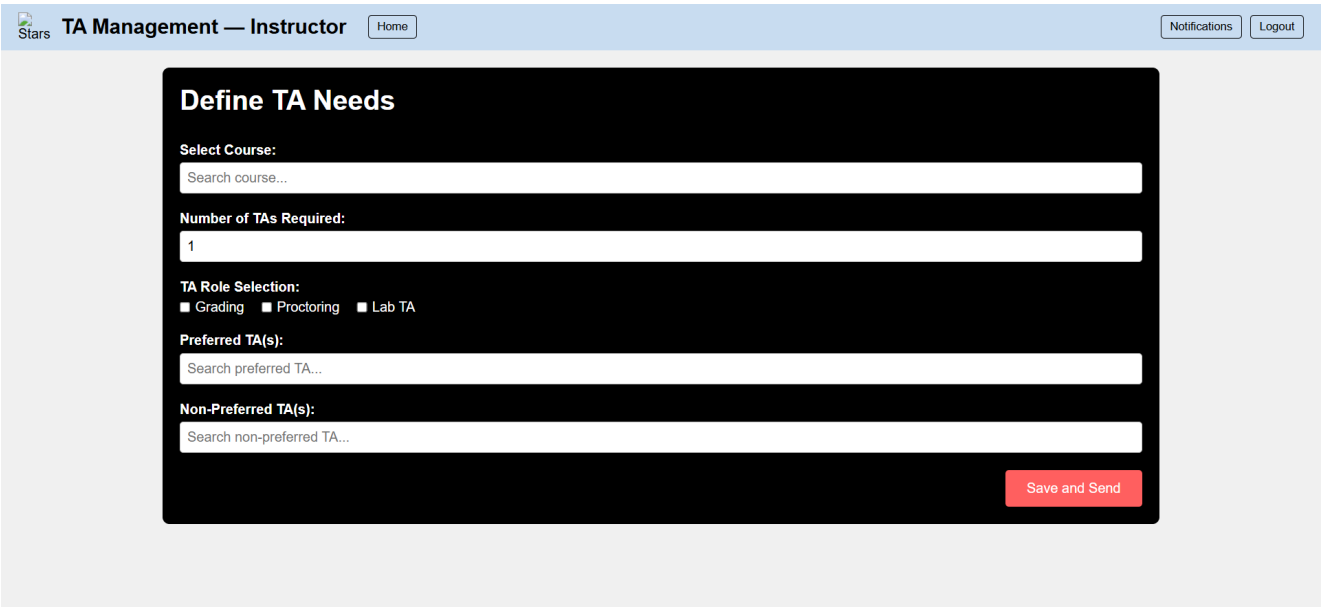


Figure 11: Prefer TA Form for Course Coordinator

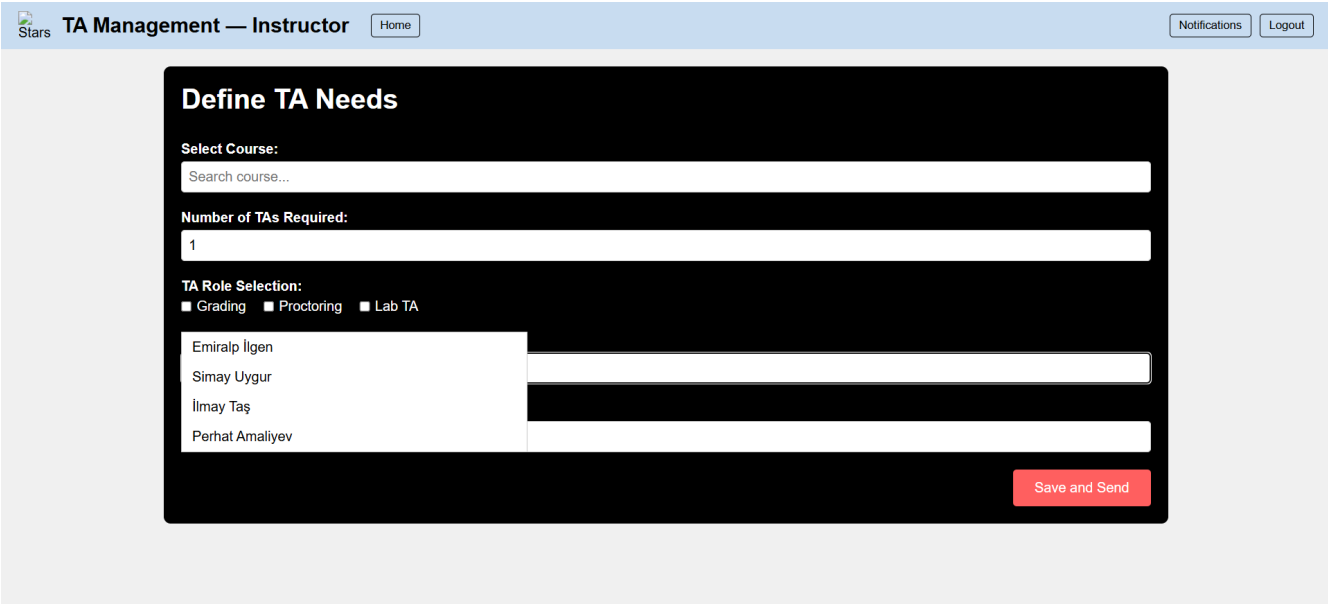


Figure 12: Prefer TA Form for Course Coordinator Search Bar of Preferred TA(s) When Clicked

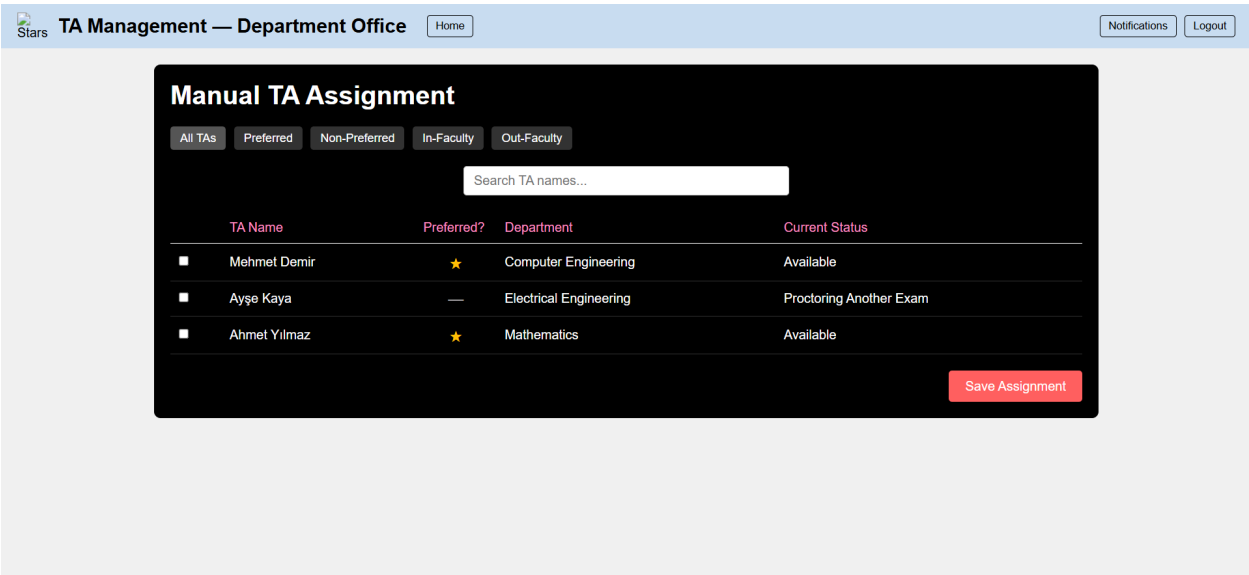


Figure 13: Manual TA Assignment Page for the Department's Office

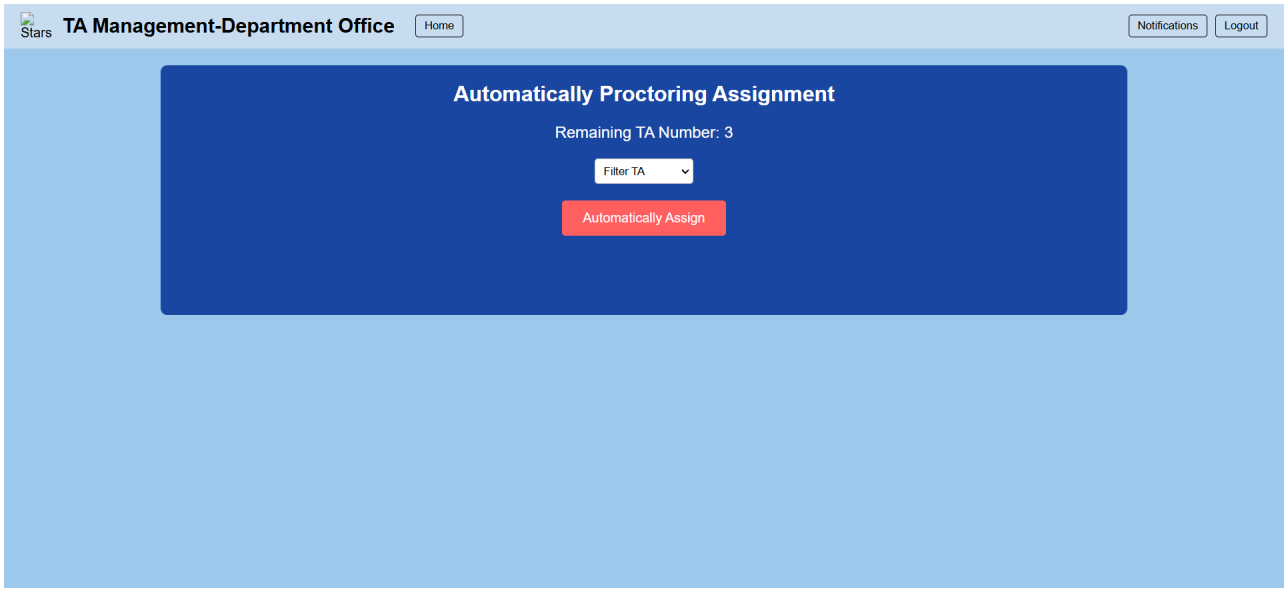


Figure 14: TA Proctoring Automatic Assignment Page for the Department's Office

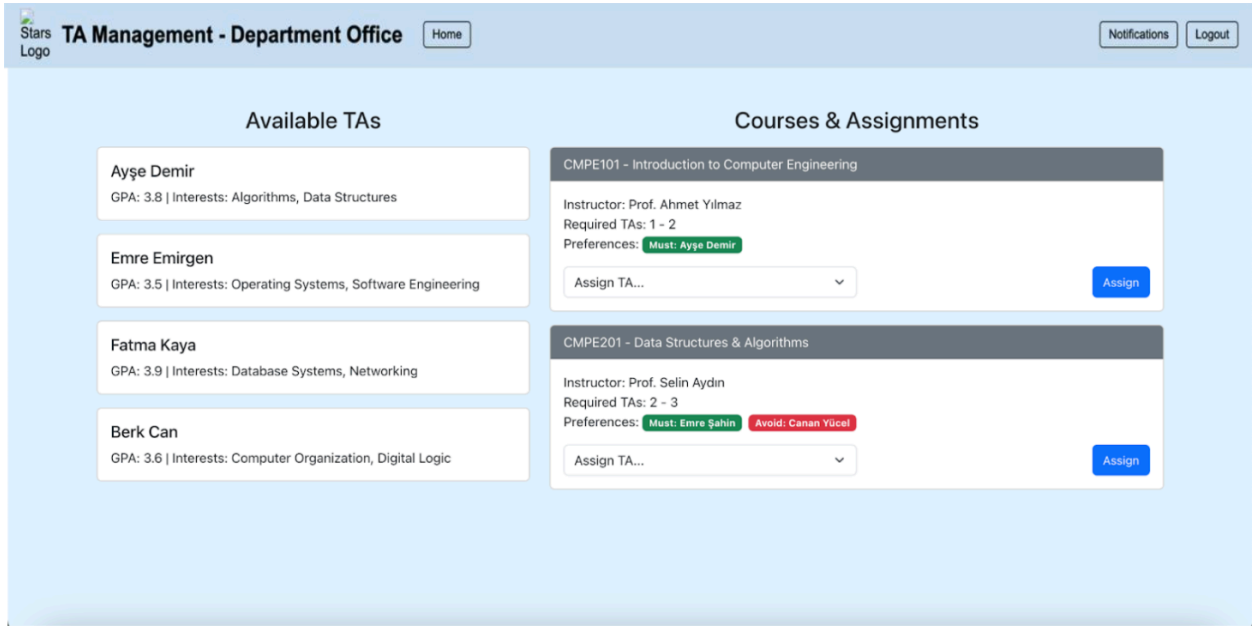


Figure 15: TA assignments to the courses by Department Office/Chair member

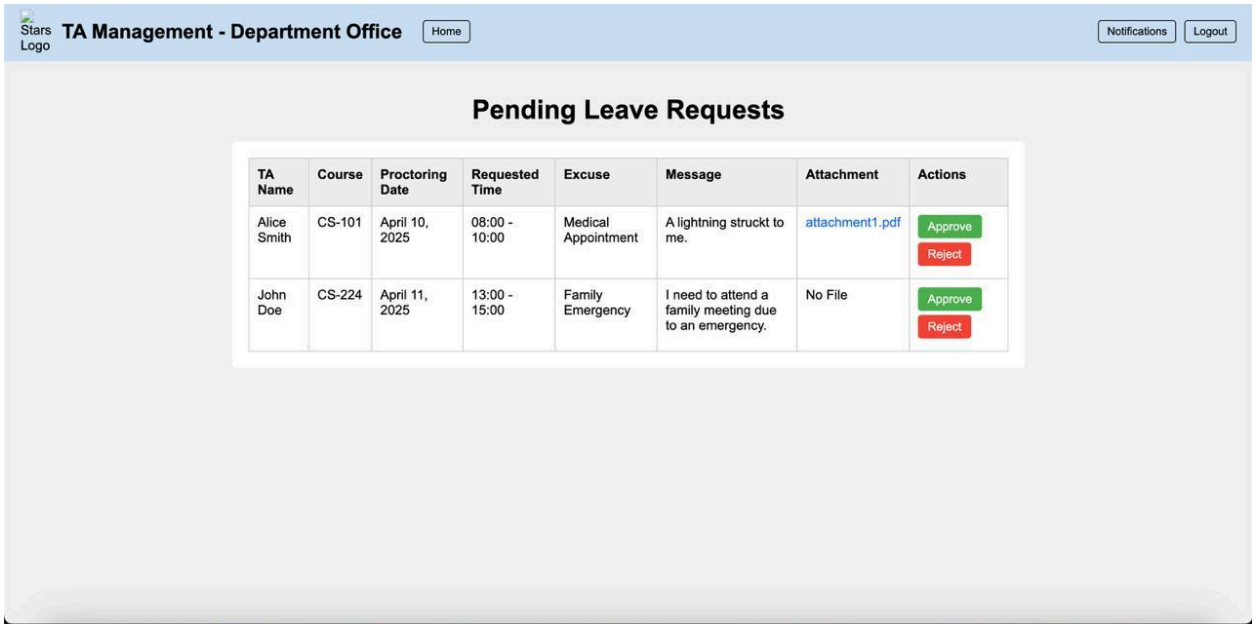


Figure 16: Awaiting Leave Requests Interface for Department Office Member

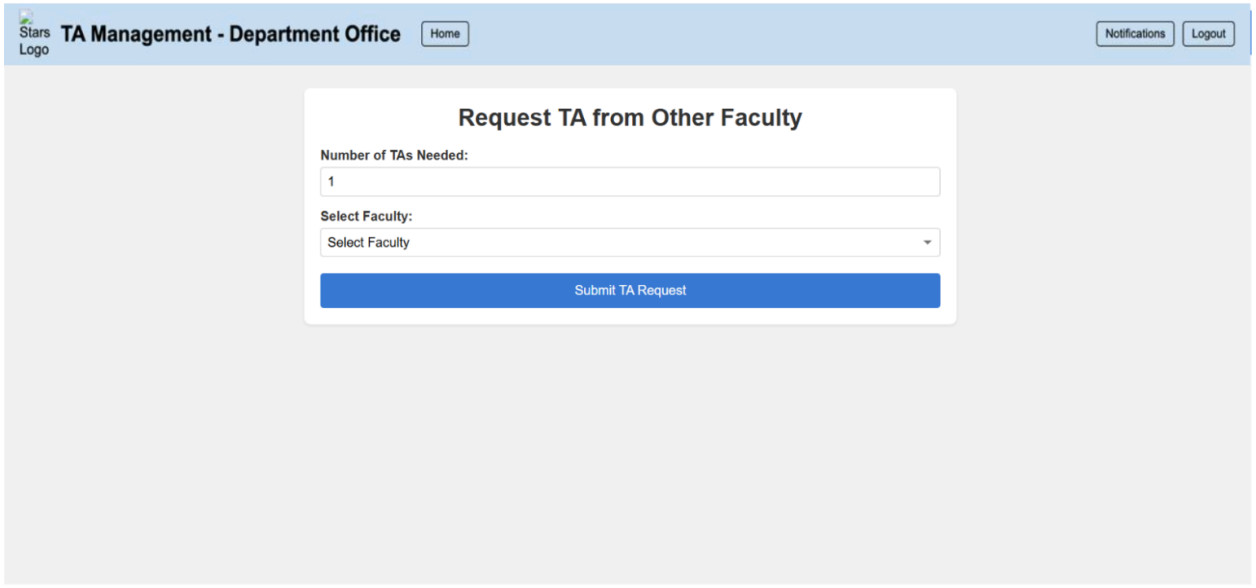


Figure 17: Request TA from Other Faculties for Department Office Member

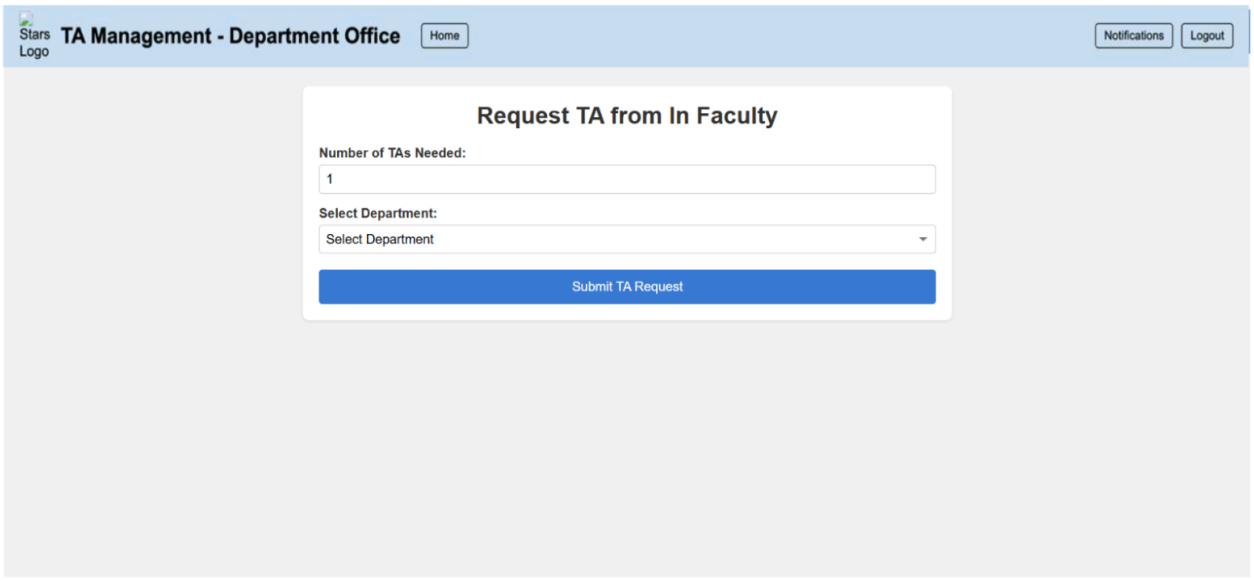


Figure 18: Request TA from Own Faculties for Department Office Member