Product Requirements Document: Student Feedback System

1. Introduction

This document outlines the requirements for the **Student Feedback System**, an online platform designed to facilitate feedback collection from students on their assigned courses. The system aims to improve learning experiences by providing a structured way for students to share their opinions and for administrators to monitor and act upon this feedback.

---

2. Goals

The primary goals of the Student Feedback System are:

- **Empower Students:** Provide an intuitive and accessible platform for students to submit constructive feedback on courses they are assigned to.
- **Enhance Course Quality:** Offer administrators and potentially instructors insights into course performance and teaching effectiveness through collected feedback.
- **Streamline Administration:** Provide tools for administrators to manage courses, student assignments, and review all submitted feedback efficiently.
- **Secure Data:** Ensure the privacy and security of user data and feedback.

---

3. Target Audience

The system caters to three primary user roles:

- **Students:** Users registered in the system who are assigned to courses and will submit feedback.
- **Administrators:** Users with elevated privileges responsible for managing courses, assigning students, and viewing all system data (users, courses, feedback, assignments).
- **Instructors (Future/Implicit):** While not directly interacting in the current version, the system's output (feedback) is implicitly for instructors to improve.

---

4. Key Features

4.1. User Management

- **User Registration:**
  - **Description:** Allows new users to create an account with a unique username, email, and password.
  - **Requirements:**
    - System must validate unique usernames and emails.
    - Passwords must meet minimum security criteria (e.g., length).
    - Successful registration should lead to a clear confirmation and redirection to login.
- **User Login:**
  - **Description:** Enables registered users to authenticate using their username and password.
  - **Requirements:**
    - Successful login provides an authentication token for subsequent API requests.
    - System must distinguish between regular student users and administrators for redirection.
    - Failed login attempts should provide clear error messages.
- **User Logout:**
  - **Description:** Allows authenticated users to end their session and clear local authentication tokens.
  - **Requirements:**
    - Clears user's local session data (accessToken, refreshToken, isAdmin status).
    - Redirects the user to the login page.
- **Admin User Creation:**
  - **Description:** The system will support dedicated administrator accounts with higher privileges.

   - **Requirements:**
      - Admin users are initially created via Django's createsuperuser command on the backend.
      - Admin status is verified by the backend upon login (is_staff flag).

4.2. Course Management (Admin)

- **Add New Course:**
   - **Description:** Administrators can add new course entries to the system, including a name and description.
   - **Requirements:**
      - Only authenticated administrators can perform this action.
      - Course names must be unique.
      - Confirmation message on success, error message on failure.
- **Manage Courses (View All Courses):**
   - **Description:** Administrators can view a list of all courses in the system.
   - **Requirements:**
      - Only authenticated administrators can access this list.
      - Courses should display their name and description.
      - *(Future Enhancement)* Ability to edit or delete existing courses.

4.3. Student-Course Assignments (Admin)

- **Assign Course to Student:**
   - **Description:** Administrators can assign specific courses to individual student users.
   - **Requirements:**
      - Only authenticated administrators can perform this action.
      - The assignment process prevents duplicate student-course pairs.

        - Confirmation message on success, clear error message for duplicates or failures.

- **View All Assignments:**
    - **Description:** Administrators can view a comprehensive list of all student-course assignments in the system.
    - **Requirements:**
        - Only authenticated administrators can access this list.
        - Displays student username, course name, and assignment date.

4.4. Feedback Management

- **Submit Feedback (Student):**
    - **Description:** Authenticated students can submit feedback for courses they are assigned to.
    - **Requirements:**
        - Students can only submit feedback for courses they are *assigned* to.
        - Feedback includes a rating (1-5 stars) and optional comments.
        - Course name is pre-filled on the feedback form based on selected course.
        - Confirmation on success, clear error messages on failure (e.g., not assigned, invalid rating).
- **View Own Feedback (Student Dashboard):**
    - **Description:** Students can view a list of all feedback they have personally submitted.
    - **Requirements:**
        - Only authenticated students can view their own feedback.
        - Displays course name, rating, and comments.
- **View All Feedback (Admin):**

  - **Description:** Administrators can view all feedback entries submitted by all students.
  - **Requirements:**
    - Only authenticated administrators can access this list.
    - Feedback entries display student, course, rating, comments, and submission timestamp.

4.5. Dashboards

- **Student Dashboard:**
  - **Description:** Provides an overview for students, listing courses they are assigned to with an option to give feedback.
  - **Requirements:**
    - Dynamically loads assigned courses from the backend.
    - Offers clear navigation to submit feedback for each course.
- **Admin Dashboard:**
  - **Description:** Provides an administrative overview with quick statistics and navigation to management tools.
  - **Requirements:**
    - Displays total counts for courses, students, and feedback.
    - Provides direct links to add courses, manage courses, view all feedback, and assign courses.

---

5. Non-Functional Requirements

- **Performance:** API response times should be minimal for common operations (login, registration, submitting feedback, loading dashboards).
- **Security:**
  - User authentication using JWT (JSON Web Tokens).

    - API endpoints protected by appropriate permission classes (IsAuthenticated, IsAdminUser).
    - Passwords are securely hashed.
    - Client-side data (tokens, admin status) stored in localStorage.
- **Usability:**
    - Clear and intuitive user interface.
    - Responsive design for various screen sizes (mobile-friendly).
    - Clear success and error messages for user actions.
- **Maintainability:** Codebase should be well-structured, documented, and follow best practices for Django/DRF and JavaScript.
- **Scalability:** The architecture should allow for future expansion in terms of users, courses, and features.

---

6. Future Enhancements (Potential Features)

- **Instructor Accounts:** Dedicated role for instructors to view feedback specific to their courses.
- **Feedback Trends & Reporting:** Advanced analytics on feedback data (e.g., average rating per course, common keywords in comments).
- **Anonymous Feedback:** Option for students to submit feedback anonymously.
- **Feedback Editing/Deletion:** Allow students to edit or delete their own feedback within a certain timeframe.
- **Course Categories:** Group courses into categories for easier organization.
- **User Profiles:** Allow users to manage their profile information.
- **Notifications:** Notify students about new assignments or admins about new feedback.
- **API Documentation:** Implement Swagger/OpenAPI for automatic API documentation.