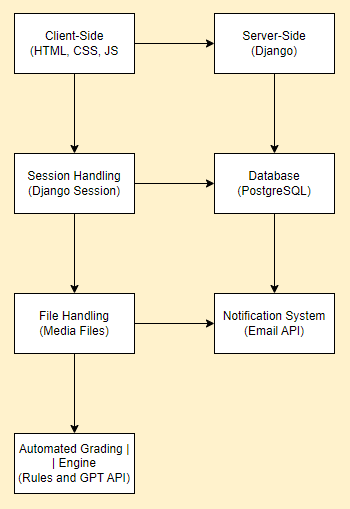
**Technical Documentation for AFSPADE Project**

**1. Project Overview**

1. **Project Name**: AFSPADE (Automated Feedback System for Programming Assignment and Educational Development)
2. **Purpose**: AFSPADE is an automated system designed to streamline the process of assigning, submitting, grading, and providing feedback on programming assignments. The system reduces the workload of lecturers by automating the feedback and grading process and allows students to receive instant, meaningful feedback on their code submissions.
3. **Target Audience**: Educational institutions, specifically Babcock University.
4. **Key Features**:
   1. Lecturer uploads assignments.
   2. Students receive notifications and submit assignments.
   3. Automated feedback on code syntax, logic, and readability.
   4. Grading system integrated for both students and lecturers to review.
   5. Support for multiple lecturers and courses.
   6. User authentication and management.

**2. System Architecture**

1. **Backend**: Django framework, using Python for logic, data processing, and server-side functionality.
2. **Frontend**: Django templates for rendering HTML views, with integrated CSS and JavaScript for interactivity and user experience.
3. **Database**: PostgreSQL, used to store user data, assignments, submissions, and grading records.
4. **Deployment**: Hosted on a cloud platform (e.g., AWS) with the Django application running on an Apache or Nginx server.
5. **Security**: Implements session-based authentication using Django’s built-in session management.



**3. User Roles and Permissions**

1. **Lecturer**:
   1. Upload and manage assignments.
   2. Review student submissions.
   3. View grading and feedback provided by the system.
   4. Download assignments and feedback for personal assessment.
2. **Student**:
   1. Receive notifications about new assignments.
   2. Submit assignments through the platform.
   3. View feedback and grading provided by the system.
3. **Admin**:
   1. Manage users (lecturers and students).
   2. Oversee system settings and maintenance.
   3. Monitor system usage and logs.

**4. Course Management**

1. **Course Registration**: Lecturers create courses by entering the course name, code, and description. Students enroll in courses using a course code provided by their lecturer.
2. **Course Structure**: Supports multiple levels and semesters, e.g., 100 Level – 1st Semester (CSC101), 2nd Semester (CSC102).

**5. Assignment Workflow**

1. **Assignment Upload**:
   1. Lecturers can either upload assignment files or directly input assignment questions into the system.
   2. Assignments include a deadline and any specific instructions.
2. **Student Submission**:
   1. Students submit their assignments by uploading files or entering code directly into a provided coding interface.
3. **Automated Grading and Feedback**:
   1. The system automatically checks submitted code for syntax, logical errors, and adherence to best practices.
   2. Feedback is generated with comments and hints, and a grade is assigned based on predefined rules.

**6. Notification System**

1. **Email Notifications**:
   1. Students receive email notifications for new assignments, deadlines, and received feedback.
   2. Lecturers are notified of submission deadlines and when feedback is available for review.

**7. User Authentication**

1. **Session-Based Authentication**:
   1. Uses Django’s session management for secure login and access control.
   2. Ensures that only authenticated users can access their respective dashboards and course materials.

**8. Database Structure**

1. **Users**: Stores information about lecturers, students, and admins.
2. **Courses**: Contains details of courses, including course code, name, and associated lecturer.
3. **Assignments**: Stores assignment details, submission deadlines, and uploaded files.
4. **Submissions**: Tracks student submissions, feedback, and grades.
5. **Notifications**: Logs notifications sent to users.

**9. Grading and Feedback**

1. **Grading Rules**: Configurable rules for grading assignments based on syntax, logic, and other criteria.
2. **Feedback**: Automatically generated based on the student's submission, highlighting mistakes and providing suggestions.

**10. Security Considerations**

1. **Data Protection**: All user data is stored securely with encrypted passwords and access controls.
2. **Session Management**: Sessions are securely managed to prevent unauthorized access..

**11. Deployment and Hosting**

**?**

**12. User Interface**

1. **Design**: A clean, intuitive interface with a consistent color scheme, ensuring ease of use for both students and lecturers.
2. **Navigation**: Easy navigation with tiles for key functions like uploading assignments, viewing submissions, and checking feedback.