Design Document for Google Classroom Integration

**1. User Scenarios:**

**- Scenario 1**: Class Creation

**User:** Teacher

**Action:** Creates a new class, e.g., "Grade 9th Math - Semester 1."

**Outcome:** Class is created and visible in Google Classroom.

**- Scenario 2:** Topic and Sub-Topic Management

**User:** Teacher

**Action:** Adds topics (e.g., Algebra, Geometry) and sub-topics (e.g., Introduction to Rational Numbers) with descriptions and optional media links.

**Outcome**: Topics and materials are organized under the class.

**- Scenario 3:** Assignment Generation

**User:** Teacher

**Action:** Creates an assignment based on sub-topic, downloads it as a PDF, or assigns it directly in Google Classroom.

**Outcome:** Assignment is available for students.

**- Scenario 4:** Assignment Submission

**User:** Student

**Action:** Receives and submits the assignment in Google Classroom.

**Outcome:** Submission is tracked.

**2. Error Handling:**

**- Class Creation Errors:**

- Missing required fields (e.g., class name).

- API connection issues.

**- Topic/Sub-Topic Errors:**

- Duplicate topic names within the same class.

- Invalid URLs for media links.

**- Assignment Errors:**

- Failed PDF generation.

- Assignment **not** linked correctly to a class.

**- General Errors:**

- Authentication failures.

- Rate limiting by Google API.

**3. Workflow of Classes:**

1. **Teacher Login** (authentication with Google API)

2**. Class Management:** Create, update, or delete classes.

3. **Topic Management:** Add, reorder, or delete topics.

4. **Sub-Topic Management:** Add descriptions, media links, and assign to students.

5. **Assignment Handling:** Generate PDFs or assign directly in Google Classroom.

6. **Student Interaction:** View assignments, submit work, and receive feedback.

**4. Non-Functional Requirements:**

**- Performance:** Fast API responses and efficient PDF generation.

**- Scalability:** Support multiple classes, subjects, and users simultaneously.

**- Security:** Secure authentication using OAuth 2.0.

**- Usability:** Intuitive interface for teachers and students.

**- Reliability:** High availability with proper error handling and recovery mechanisms.

**5. Expected Input and Output:**

**- Input:**

- Class Name (e.g., "Grade 10 Science")

- Topics (e.g., Physics, Chemistry)

- Sub-Topics (e.g., "Newton's Laws" with descriptions)

- Assignment Details (questions, media links)

**- Output:**

- Confirmation of class, topic, and sub-topic creation.

- PDF files for assignments.

- Assignments visible in Google Classroom.

**6. Test Cases on Use Cases:**

**- Test Case 1:** Verify class creation with valid data.

Input: "Grade 8 English - Semester 2"

Expected Output: Class created successfully.

**- Test Case 2:** Handle missing topic names.

Input: Empty topic field.

Expected Output: Error message indicating the required field is missing.

**- Test Case 3:** Assign sub-topic to students.

Input: Sub-topic "Introduction to Algebra" assigned to student list.

Expected Output: Assignment appears in students' Google Classroom.

**- Test Case 4:** Verify PDF generation.

Input: Assignment details with questions and media links.

Expected Output: Downloadable PDF with correct content.

**- Test Case 5:** API error handling.

Scenario: Simulate API rate limiting.

Expected Output: Graceful error message with retry option.

**7. Customer Scenarios:**

**- Scenario 1:** Customer uploads JSON data for processing.

**Action:** The script validates the JSON and triggers specific API calls.

**Outcome:** Data is processed successfully, and the result is returned.

**- Scenario 2:** Customer automates repetitive API tasks.

**Action:** The script processes a batch of JSON inputs.

**Outcome:** All API tasks are completed with status reports.

**8. Sub-Systems and Components:**

**- Input Handler:** Validates and parses JSON inputs.

**- API Manager:** Manages calls to third-party APIs.

**- Error Logger:** Captures and logs errors.

**- Response Formatter:** Formats output for easy interpretation.

**9. Inputs and Outputs (Including Error Handling and Exceptions):**

**- Inputs:** JSON files with predefined schema.

**- Outputs:**

- API responses in JSON format.

- Error logs for failed processes.

**- Error Handling:**

- Invalid JSON structure.

- API timeout and failures.

- Exception handling for unexpected errors.

**10. Non-Functional Requirements:**

**- Scale:** Should handle large batches of JSON inputs efficiently.

**- Latency:** API calls should be optimized for low response times.

**- Error Logging:** Comprehensive logging for troubleshooting.

**- Languages Supported**: Initially supports Python with scope for multi-language support.

End of Design Document