**Test Cases for Login\_API (Login Functionality)**

**Test Case 1: Successful Login with Correct Credentials**

* **Test Objective**: Verify that the user can log in successfully with valid email and password.
* **Test Steps**:
  1. Send a POST request to /api/login with the following JSON body:
  2. {
  3. "usermail": "user@example.com",
  4. "password": "userpassword"
  5. }
  6. Verify that the response status is 200 OK.
  7. Verify the response message is {"message": "Login successful.", "isPremium": false}.
* **Expected Result**: User logs in successfully, and the response message confirms the login is successful.

**Test Case 2: Login with Invalid Email**

* **Test Objective**: Verify that login fails with an invalid email.
* **Test Steps**:
  1. Send a POST request to /api/login with the following JSON body:
  2. {
  3. "usermail": "wronguser@example.com",
  4. "password": "userpassword"
  5. }
  6. Verify that the response status is 401 Unauthorized.
  7. Verify the response message is {"message": "Invalid credentials."}.
* **Expected Result**: Login fails due to incorrect email, and an appropriate error message is returned.

**Test Case 3: Login with Incorrect Password**

* **Test Objective**: Verify that login fails with an incorrect password.
* **Test Steps**:
  1. Send a POST request to /api/login with the following JSON body:
  2. {
  3. "usermail": "user@example.com",
  4. "password": "wrongpassword"
  5. }
  6. Verify that the response status is 401 Unauthorized.
  7. Verify the response message is {"message": "Invalid credentials."}.
* **Expected Result**: Login fails due to incorrect password, and an appropriate error message is returned.

**Test Case 4: Login with Missing Email**

* **Test Objective**: Verify that the system returns an error when the email is missing.
* **Test Steps**:
  1. Send a POST request to /api/login with the following JSON body:
  2. {
  3. "password": "userpassword"
  4. }
  5. Verify that the response status is 400 BadRequest.
  6. Verify the response message is {"message": "UserMail and Password are required."}.
* **Expected Result**: The login attempt should fail due to the missing email field.

**Test Case 5: Login with Missing Password**

* **Test Objective**: Verify that the system returns an error when the password is missing.
* **Test Steps**:
  1. Send a POST request to /api/login with the following JSON body:
  2. {
  3. "usermail": "user@example.com"
  4. }
  5. Verify that the response status is 400 BadRequest.
  6. Verify the response message is {"message": "UserMail and Password are required."}.
* **Expected Result**: The login attempt should fail due to the missing password field.

**Test Cases for LogWise\_API (Log Analysis Functionality)**

**Test Case 1: Analyze Log Data with Valid Content**

* **Test Objective**: Verify that the system analyzes log data and generates descriptions for each entry.
* **Test Steps**:
  1. Send a POST request to /api/analyze with the following JSON body:
  2. {
  3. "file\_content": "Log file content here",
  4. "parsed\_log\_data": [
  5. {
  6. "message": "NullPointerException occurred",
  7. "details": "Exception in thread main"
  8. }
  9. ]
  10. }
  11. Verify that the response status is 200 OK.
  12. Verify that the response contains both the log entries and corresponding descriptions.
  13. Verify that the description for the error message NullPointerException occurred is generated.
* **Expected Result**: The log entries are analyzed, and appropriate descriptions are generated.

**Test Case 2: Analyze Log Data with Missing File Content**

* **Test Objective**: Verify that the system returns an error if no file content is provided.
* **Test Steps**:
  1. Send a POST request to /api/analyze with the following JSON body:
  2. {
  3. "parsed\_log\_data": [
  4. {
  5. "message": "NullPointerException occurred",
  6. "details": "Exception in thread main"
  7. }
  8. ]
  9. }
  10. Verify that the response status is 400 BadRequest.
  11. Verify the response message is {"error": "No file content provided."}.
* **Expected Result**: The system should return an error indicating that file content is required.

**Test Case 3: Analyze Log Data with Empty Parsed Data**

* **Test Objective**: Verify that the system returns an error if parsed log data is empty.
* **Test Steps**:
  1. Send a POST request to /api/analyze with the following JSON body:
  2. {
  3. "file\_content": "Log file content here",
  4. "parsed\_log\_data": []
  5. }
  6. Verify that the response status is 400 BadRequest.
  7. Verify the response message is {"error": "Parsed log data cannot be empty."}.
* **Expected Result**: The system should return an error indicating that parsed log data is required.

**Test Case 4: Analyze Log Data with Invalid Data Format**

* **Test Objective**: Verify that the system returns an error for invalid log data format.
* **Test Steps**:
  1. Send a POST request to /api/analyze with the following malformed JSON body:
  2. {
  3. "file\_content": "Log file content here",
  4. "parsed\_log\_data": "Invalid log data format"
  5. }
  6. Verify that the response status is 400 BadRequest.
  7. Verify the response message is {"error": "Invalid log data format."}.
* **Expected Result**: The system should return an error indicating that the log data format is invalid.

**Test Case 5: Analyze Log Data with Internal Server Error**

* **Test Objective**: Verify that the system handles unexpected errors and returns a 500 status code.
* **Test Steps**:
  1. Send a POST request to /api/analyze with the following JSON body (simulating a server error):
  2. {
  3. "file\_content": "Valid log content",
  4. "parsed\_log\_data": [
  5. {
  6. "message": "Error",
  7. "details": "Internal server error"
  8. }
  9. ]
  10. }
  11. Verify that the response status is 500 Internal Server Error.
  12. Verify the response message indicates that an unexpected error occurred.
* **Expected Result**: The system should return a 500 Internal Server Error with the appropriate message.

**Conclusion**

These test cases cover the major functionalities of the Login\_API (Login and Sign-up) and LogWise\_API (Log Analysis). They validate both normal operations and error handling scenarios.