# Jdoodle Application Test Strategy/ Test Procedure

### **1. Functionality Testing:**

* Code Execution:
  + Verify that the compiler correctly compiles and executes Java code.
  + Test with various Programming Language versions and configurations.
  + Check for proper handling of syntax errors and warnings.
* Input/Output Handling:
  + Test standard input/output functionality.
  + Verify the correctness of console outputs for different code scenarios.
* Libraries and APIs:
  + Test the usage of programming language libraries and APIs.
  + Ensure compatibility with commonly used libraries.
* Error Handling:
  + Verify that the platform provides meaningful error messages.
  + Test error scenarios, such as compilation failures or runtime errors.
* Security:
  + Ensure the platform prevents malicious code execution.
  + Test for security vulnerabilities, especially in code execution.

### **2. Performance Testing:**

* Execution Time:
  + Test the compiler's performance for various code complexities.
  + Verify that execution time is within acceptable limits.
* Concurrency:
  + Test the platform's behavior under concurrent user loads.
  + Check for potential performance bottlenecks.
* Scalability:
  + Evaluate the compiler's scalability by testing with a large number of concurrent users and large code files.

### **3. Compatibility Testing:**

* Browser Compatibility:
  + Test the platform across different web browsers (Chrome, Firefox, Safari, Edge) to ensure compatibility.

### **4. Integration Testing:**

* IDE Integration:
  + Test compatibility with popular programming languages Integrated Development Environments (IDEs).
  + Ensure seamless code transfer between the online compiler and local IDEs.

### **5. Accessibility Testing:**

* Accessibility Standards:
  + Ensure that the platform complies with accessibility standards.

### **6. Data Security:**

* Data Encryption:
  + Ensure that user data, including code snippets, is encrypted during transmission.
  + Verify secure storage practices.

### **7. Load Testing:**

* Stress Testing:
  + Subject the platform to high loads to identify performance thresholds.
  + Test how the system recovers from sudden traffic spikes.

### **8. Documentation Testing:**

* User Guides:
  + Verify the accuracy and completeness of user documentation.
  + Ensure that users can easily find information on platform features and usage.

### **9. Cross-Browser Testing:**

* Browser Compatibility:
  + Test the platform across different web browsers to ensure a consistent experience.

### **10. Continuous Testing:**

* Automated Testing:
  + Implement automated testing for critical functionalities.
  + Utilize continuous integration tools to run tests on code changes.

### **11.Bug Reporting:**

* User Feedback:
  + Encourage users to provide feedback on their experience.
  + Monitor and respond to user-reported issues promptly.

### **12. Regression Testing:**

* Code Changes:
  + Perform regression testing after each code deployment or significant changes.
  + Ensure that existing functionalities remain unaffected.

### 

### **13. API Testing:**

* API Endpoints:
  + If applicable, test the API endpoints for correctness and reliability.