**Automated Test Plan for Live Streaming Product**

**1. Introduction**

* Objective: The objective of this automated test plan is to ensure that Company B’s API and its integration into Company A’s platform function correctly across all platforms. The plan focuses on scalability, functionality, and backward compatibility, ensuring minimal manual intervention in future testing.
* Scope:
  + API Testing: Validate the functionality, scalability, and integrity of Company B's API.
  + Client Applications Testing: Ensure that CompanyA's client apps (Desktop Web, Mobile Web, iOS, Android, FireTV, AndroidTV, Roku, Chromecast) interact correctly with the API.
  + Regression Testing: Automatically re-run test cases to ensure existing features are not broken by new updates.

**2. Test Objectives**

* Validate the core functionalities of the APIs, Web Apps, and Native Apps.
* Ensure all user interfaces are functional and user-friendly.
* Verify data integrity and accuracy throughout the application.
* Test the application on different browsers, operating systems, devices, and platforms.
* Identify and log any defects or issues for resolution.

**3. Tools**

* **API Testing**
  + [Postman](https://learning.postman.com/docs/introduction/overview/) for API functional testing and automation.
  + [Newman](https://learning.postman.com/docs/collections/using-newman-cli/command-line-integration-with-newman/) for running Postman tests in CI/CD pipelines.
* **Client Application Testing**
  + [Selenium](https://www.selenium.dev/documentation/) (Python) for automating web application testing across different browsers.
  + [Appium](https://appium.io/docs/en/latest/guides/migrating-1-to-2/) (Python) for mobile application testing on Android and iOS.
  + [BrowserStack](https://www.browserstack.com/docs/?utm_source=google&utm_medium=cpc&utm_campaign=Search-Brand-US-AMER-Navigational) for cross-browser and cross-device testing.
* **Continuous Integration/Continuous Deployment (CI/CD):**
  + [Jenkins](https://www.jenkins.io/doc/tutorials/) for integrating automated tests into the deployment pipeline.

**4.1 Features to be Tested**

* Authentication: Verify that only authenticated users can post comments, photos, and reactions, while non-authenticated users can only view them.
* Post and Reaction Functionality: Test the ability to post comments, photos, and reactions.
* View Functionality: Ensure that non-authenticated users can view comments, photos, and reactions but cannot interact.
* Cross-Platform Compatibility: Test the functionality across all platforms (Desktop Web, Mobile Web, iOS Native, Android Native, FireTV, AndroidTV, Roku, Chromecast).

**4.2 Features Not to be Tested**

* Third-party integrations (beyond the scope of this test plan)
* Non-functional requirements like load testing (handled separately)

**5.1 API Testing**

* Verify Endpoint Responses:
  + Check if the API returns the expected data structure for various endpoints.
  + Ensure that all required fields (e.g., id, title, body, userId for posts) are present.
* Data Consistency Tests:
  + Validate that data returned from the API is consistent across multiple requests.
  + Compare data across different endpoints for related information (e.g., verifying userId consistency).
* Edge Case Testing:
  + Test with missing or invalid parameters to ensure proper error handling.
* Backward Compatibility:
  + Ensure that new API versions do not break existing functionality.

**5.2 Client Applications Testing**

* Cross-Platform Functional Testing:
  + Automate user interactions such as login, viewing streams, posting comments, and uploading photos across all supported platforms.
* UI Consistency:
  + Ensure that the user interface is consistent and functional across different browsers and devices.
* API Integration:
  + Validate that the client applications properly integrate with Company B's API, particularly during live streams.

**5.3 Regression Testing**

* Automated Regression Suite:
  + Regularly run a suite of automated tests to catch regressions after new code deployments.
* Version Control:
  + Maintain test scripts to ensure they are updated with new features and changes.

**5.4 Test Execution**

* Environment:
  + Test in a staging environment that closely mimics the production setup.
* Test Data:
  + Use mock data for testing APIs and user interactions to avoid impacting production data.
* Execution Schedule:
  + API tests to be run on every code commit via CI/CD.
  + Web App UI tests to be run on every code commit via CI/CD.
  + Native App UI tests to be run on every code commit via CI/CD.
  + Full regression tests to be run before every major deployment.

**6. Reporting**

* Test Reports:
  + Automated test results will be captured and reported via Jenkins.
* Dashboard:
  + Use a test management tool or custom dashboard to track test execution, failures, and performance metrics.
* Defect Tracking:
  + Automatically log any defects found during testing into a defect tracking system.

**7. Maintenance**

* Test Script Maintenance:
  + Regularly review and update test scripts to align with evolving requirements.
* Continuous Improvement:
  + Analyze test results to identify areas of improvement in the automated testing process.

**8. Risks and Mitigation**

* Risk: API changes might break existing automated tests.
  + Mitigation: Implement backward compatibility tests and version control.
* Risk: Flakiness in UI tests due to browser updates.
  + Mitigation: Use cloud-based testing tools like BrowserStack that automatically update environments.

**9. Approval**

* Prepared by: [Your Name]
* Approved by: [Project Manager/Lead QA]