**Test Plan for WhatsApp Mobile Application**

**1. Objective**

The objective of this test plan is to **evaluate both the positive and negative aspects** of the WhatsApp mobile application. It aims to ensure the **reliability, functionality, usability, and stability** of the app across supported mobile platforms, and to identify areas for **enhancement and risk mitigation**.

**2. Test Environment**

**Platforms & Devices**

| **Platform** | **Versions to be Tested** | **Device Coverage** |
| --- | --- | --- |
| Android | 9.0, 10.0, 11.0 | Low-end, mid-range, and high-end smartphones |
| iOS | 12.0, 13.0, 14.0 | iPhone SE, iPhone XR, iPhone 11 and above |

**Network Configurations**

* Wi-Fi (stable connection)
* 4G, 5G mobile networks
* Simulated low/no network conditions using network throttling tools

**3. Test Strategy**

**Approach**

| **Type of Testing** | **Description** |
| --- | --- |
| **Manual Testing** | Used for exploratory, UI, usability, and complex test scenarios |
| **Automation Testing** | Repetitive and regression test cases automated using **Selenium with Java** |
| **Positive Testing** | Validate standard workflows: messaging, calls, group chats, media sharing |
| **Negative Testing** | Test invalid inputs, unsupported files, boundary values, stress scenarios |

**Test Coverage Areas**

1. **Messaging Functionality**
   * Send/receive messages
   * Delivery/read receipts
   * Emoji, stickers, voice notes
2. **Calls**
   * Voice and video calls (1-on-1 and group)
   * Call interruption handling
3. **Group Chats**
   * Create, rename, and delete groups
   * Add/remove participants
   * Group media sharing
4. **File Sharing**
   * Send images, videos, documents
   * File limits and format handling
5. **Status Updates**
   * Create/view/delete status
   * Privacy controls
6. **Settings & Privacy**
   * Account setup
   * Two-step verification
   * Block/report contacts
7. **Negative Test Scenarios**
   * Invalid phone numbers during sign-up
   * Attempt to send corrupt or unsupported files
   * Large message flood (spam simulation)
   * Operating with minimal storage or network

**4. Entry Criteria**

* All required **test cases have been authored and reviewed**
* Required **test environments** (devices, OS versions, network setup) are available and configured
* Test data is complete and accessible
* Initial build is **successfully deployed**

**5. Exit Criteria**

* **100% of planned test cases executed**
* All **critical and high-severity defects logged, fixed, and verified**
* **No open showstopper bugs**
* **Test summary report** is created, reviewed, and approved
* Regression and sanity checks passed in final release candidate

**6. Roles & Responsibilities**

| **Role** | **Responsibilities** |
| --- | --- |
| **Test Manager** | Overall planning, progress tracking, coordination with stakeholders |
| **Testers** | Write and execute test cases, log defects, retest, and document results |
| **Developers** | Analyze and fix defects, assist with test automation |
| **Project Manager** | Monitor project timelines, resource allocation, and delivery health |

**7. Tools**

| **Purpose** | **Tool Used** |
| --- | --- |
| **Automation Testing** | Selenium WebDriver with Java |
| **Bug Tracking** | JIRA |
| **Test Case Management** | Excel Sheets / Google Sheets |
| **Reporting & Metrics** | Excel, JIRA Dashboards |

**Conclusion**

By executing this comprehensive test plan, the QA team will be able to **identify both strengths and weaknesses** in WhatsApp’s functionality. It ensures that the app performs reliably under a wide range of **user, device, and network conditions**, ultimately providing a **robust and user-friendly experience**.

**Test Plan: Coffee Machine**

**1. Objective**

The objective of this test plan is to **assess the functionality, usability, and performance** of the coffee machine under various scenarios, ensuring **reliability, safety, and quality**. This includes validating both **positive behaviours** (e.g., successful brewing) and **negative edge cases** (e.g., water shortage, overloading beans).

**2. Test Environment**

Testing will be conducted in a **controlled laboratory setting** designed to simulate real-world usage. The environment includes:

* **Power Supply**: Stable AC outlet (110V or 220V depending on region)
* **Water Supply**: Clean water tank or filtered input
* **Coffee Beans**: Variety (light, medium, dark roast; whole beans and ground coffee)
* **Environmental Conditions**: Room temperature between 20–25°C; normal humidity
* **Machine Variants**: Single-cup, espresso, drip coffee machines (if applicable)

**3. Test Strategy**

**Approach**

| **Test Type** | **Description** |
| --- | --- |
| **Manual Testing** | Functional testing of buttons, brewing cycles, display, safety mechanisms |
| **Automated Testing** | Limited scope (e.g., UI panel testing or embedded UI, if applicable), using Selenium with Java |
| **Positive Testing** | Normal operations like selecting brew type, setting timer, and starting brew |
| **Negative Testing** | Abnormal inputs such as no water, door open, overfill, broken sensors |
| **Regression Testing** | Re-validation after firmware updates or hardware fixes |
| **Performance Testing** | Speed of heating, brewing time, multi-cup support, temperature accuracy |
| **Usability Testing** | Ease of use for different users, label clarity, accessibility |
| **Safety Testing** | Overheating, electrical failure, steam venting safety |

**Sample Test Scenarios**

**Positive Cases**

* Brew a cup of coffee with correct water and beans
* Display shows status messages (e.g., “Brewing”, “Ready”)
* Machine stops after brewing is complete
* Alerts for “Add Water” or “Empty Grounds”

**Negative Cases**

* Try brewing with no water
* Power loss during brewing
* Door left open
* Beans overfilled
* Press multiple conflicting buttons simultaneously
* Incorrect timer setup (e.g., past time)

**4. Entry & Exit Criteria**

**Entry Criteria**

* Finalized requirements/specs from the product team
* Test cases documented and reviewed
* Test environment set up (machine, beans, water, power)
* Test data (usage logs, consumables) prepared
* Tools configured (Selenium, JIRA, Excel)

**Exit Criteria**

* All functional and non-functional test cases executed
* No **critical or high-priority defects** remain open
* Defects logged in **JIRA**, retested and verified
* Final **test summary report** created and reviewed
* Sign-off from **Product Owner and QA Lead**

**5. Roles & Responsibilities**

| **Role** | **Responsibility** |
| --- | --- |
| **Test Lead** | Create and maintain test plan, schedule, assign tasks, oversee test progress |
| **Testers** | Execute manual and automated test cases, report defects, verify fixes |
| **Developers** | Debug and resolve reported defects, support QA team during troubleshooting |
| **Product Owner** | Review results, provide clarifications, approve release readiness |

**6. Tools to Be Used**

| **Tool** | **Purpose** |
| --- | --- |
| **Selenium with Java** | Automate testing for embedded UI panels, if digital interface exists |
| **JIRA** | Track test cases, report defects, assign issues to developers |
| **Excel Sheets** | Manage test cases, track test data (bean types, cycles, results), and reporting |

**Conclusion**

By following this test plan, the QA team will comprehensively evaluate the **reliability, performance, usability, and safety** of the coffee machine. Through a balanced approach of **positive and negative testing**, we ensure that the product meets user expectations and regulatory standards for quality.

**Test Case: Coffee Machine**

**Test Scenarios**

**Scenario 1: Successful Coffee Brewing (Positive)**

| **Step** | **Action** | **Expected Result** |
| --- | --- | --- |
| 1 | Power on the machine | Machine display shows "Ready" |
| 2 | Select drink type: "Espresso" | Display updates to customization options |
| 3 | Choose strength: "Medium", Cup size: "Small" | Selected values are shown on screen |
| 4 | Press "Brew" | Brewing process starts; progress shown on screen |
| 5 | Wait for drink to dispense | Coffee is dispensed into the cup |
| 6 | Confirm machine shows "Ready" again | Machine resets to ready state; previous selection cleared |

**Defect Example (if occurs)**:

* *DEFECT\_ID: CM\_DEF\_003*
* *Issue: Machine hangs after brewing – does not return to Ready state*

**Scenario 2: Missing Ingredient (Negative)**

| **Step** | **Action** | **Expected Result** |
| --- | --- | --- |
| 1 | Remove water tank or leave it empty | Machine should detect missing water |
| 2 | Attempt to brew any coffee type | Error message: "Add Water to Continue" displayed; brewing blocked |
| 3 | Refill tank and try again | Error clears; brewing allowed |

**Potential Defect**:

* *DEFECT\_ID: CM\_DEF\_007*
* *Issue: Brewing proceeds despite water tank being empty (safety failure)*

**Scenario 3: Ingredient Overload (Negative)**

| **Step** | **Action** | **Expected Result** |
| --- | --- | --- |
| 1 | Overfill the coffee bean container (beyond max limit) | Warning or block message: "Too Many Beans – Remove Excess" |
| 2 | Try to brew with overload | Brewing prevented until corrected |
| 3 | Remove excess beans and retry | Normal brewing proceeds |

**Usability Check**: Ensure user guidance is clear for correcting issue.

**Scenario 4: Malfunction During Dispensing (Negative)**

| **Step** | **Action** | **Expected Result** |
| --- | --- | --- |
| 1 | Initiate brewing (Latte with milk frothing enabled) | Machine starts as expected |
| 2 | During frothing, block milk nozzle manually | Machine detects blockage, stops safely |
| 3 | Error message: "Milk Frother Blocked – Clean Nozzle" appears | User prompted to clear the blockage |

**Defect Possibility**:

* *DEFECT\_ID: CM\_DEF\_010*
* *Issue: Milk frother continues operating despite block, leading to spillage*

**Scenario 5: Incorrect User Input (Negative)**

| **Step** | **Action** | **Expected Result** |
| --- | --- | --- |
| 1 | Press conflicting buttons at the same time (e.g., Brew + Cancel) | System handles gracefully, either ignores or prompts clarification |
| 2 | Observe UI behavior | UI does not crash or freeze |

**Defect Risk**:

* *DEFECT\_ID: CM\_DEF\_013*
* *Issue: UI freezes when multiple inputs are registered simultaneously*

**Scenario 6: Usability and Consistency**

| **Step** | **Action** | **Expected Result** |
| --- | --- | --- |
| 1 | Navigate through drink customization menu | Menu flow is intuitive; back and cancel buttons work consistently |
| 2 | Check button labels and messages | All text is clear, non-technical, and grammatically correct |
| 3 | Try setting a custom drink and cancel midway | Cancelling restores default state without crashing |

**Defect Example**:

* *DEFECT\_ID: CM\_DEF\_015*
* *Issue: Menu doesn’t reset after canceling, retains previous selections*

**📋 Expected Results Summary**

* Coffee should only be brewed when all conditions (water, beans, settings) are valid
* Clear user messages for errors (e.g., missing ingredients, blocked components)
* No operation should result in a crash, freeze, or unsafe condition
* User interface should be **intuitive, consistent**, and **fail-safe**

**🛠️ Tools Used**

| **Tool** | **Usage** |
| --- | --- |
| **Selenium with Java** | Automate touchscreen UI testing (if supported) |
| **JIRA** | Defect tracking, test case linking, test execution logs |
| **Excel Sheets** | Manual test tracking, test data input logs |

**Conclusion**

This test case ensures a **realistic, user-centric validation** of the coffee machine’s functionality. It not only ensures basic brewing features work correctly, but also verifies the machine’s ability to handle **invalid inputs, malfunctions, and edge cases**, ensuring a **safe and reliable user experience**.

**Test Case: WhatsApp**

**Test Scenarios**

**Scenario 1: Positive – Send and Receive Message**

| **Step** | **Action** |
| --- | --- |
| 1 | Launch WhatsApp |
| 2 | Navigate to "Chats" tab |
| 3 | Select a contact |
| 4 | Type and send a message ("Hello!") |
| 5 | Observe if the message is delivered and read |
| 6 | Check double tick and blue tick indicators |

**Expected Result**: Message is sent, delivered (✓✓), and read (✓✓ in blue).  
**Potential Defect**: Message stuck on single tick despite strong internet — *DEFECT\_ID: WA\_DEF\_010*

**Scenario 2: Negative – Attempt to Send Message Without Internet**

| **Step** | **Action** |
| --- | --- |
| 1 | Disable Wi-Fi/data |
| 2 | Send a message to any contact |
| 3 | Re-enable internet after 30 seconds |

**Expected Result**: Message shows a clock icon; sends successfully once reconnected.  
**Potential Defect**: Message does not send post reconnection — *DEFECT\_ID: WA\_DEF\_013*

**Scenario 3: Positive – Make Voice and Video Calls**

| **Step** | **Action** |
| --- | --- |
| 1 | Open chat with a contact |
| 2 | Tap on the voice call icon |
| 3 | End call and initiate a video call |

**Expected Result**: Calls are connected without delay, video is clear, audio is synced.  
**Potential Defect**: Audio not transmitted during video call — *DEFECT\_ID: WA\_DEF\_021*

**Scenario 4: Negative – Video Call with Low Bandwidth**

| **Step** | **Action** |
| --- | --- |
| 1 | Use a network throttling tool to simulate 2G/Edge speeds |
| 2 | Initiate a video call |
| 3 | Observe call stability and behavior |

**Expected Result**: Video quality drops gracefully or call switches to audio-only mode.  
**Potential Defect**: App crashes due to bandwidth handling — *DEFECT\_ID: WA\_DEF\_025*

**Scenario 5: Positive – Send Media Files (Images, Video, Document)**

| **Step** | **Action** |
| --- | --- |
| 1 | Open chat > Tap attachment icon |
| 2 | Send image, video (up to 16 MB), and a document |
| 3 | Confirm successful upload and visibility on receiver’s side |

**Expected Result**: Media is sent and viewed without quality loss.  
**Potential Defect**: Document upload stuck at 99% — *DEFECT\_ID: WA\_DEF\_034*

**Scenario 6: Negative – Send Unsupported File Format**

| **Step** | **Action** |
| --- | --- |
| 1 | Attempt to attach and send .exe or .bat file |

**Expected Result**: App shows an error or blocks upload.  
**Potential Defect**: File bypasses filter and uploads — *DEFECT\_ID: WA\_DEF\_041*

**Scenario 7: Usability – Navigation and Menu Labels**

| **Step** | **Action** |
| --- | --- |
| 1 | Explore tabs: Chats, Status, Calls |
| 2 | Tap settings and navigate to Account, Privacy, Chat Backup |
| 3 | Review text labels, tooltips, and accessibility options |

**Expected Result**: UI is intuitive; all labels are clear; navigation is consistent.  
**Potential Defect**: Overlapping text in dark mode — *DEFECT\_ID: WA\_DEF\_052*

**Scenario 8: Stability – Switch Between Apps Mid-Call**

| **Step** | **Action** |
| --- | --- |
| 1 | Make a voice call |
| 2 | Minimize WhatsApp and open another app (e.g., YouTube) |
| 3 | Return to WhatsApp and verify call continuity |

**Expected Result**: Call remains active; audio is stable.  
**Potential Defect**: Call drops automatically — *DEFECT\_ID: WA\_DEF\_058*

**📎 Bug Reporting Guidelines (for JIRA)**

When logging a bug:

* **Summary**: Clear and concise (e.g., "Voice call drops when switching apps")
* **Steps to Reproduce**: List precise steps taken
* **Expected Result**: Describe what should happen
* **Actual Result**: Describe what actually happened
* **Environment**: Device model, OS version, WhatsApp version
* **Attachment**: Screenshots or screen recording (if applicable)
* **Severity**: Blocker / Major / Minor / Trivial

**🛠️ Tools for Execution**

| **Tool** | **Purpose** |
| --- | --- |
| **JIRA** | Test case management and defect tracking |
| **Android Studio/iOS Simulator** | Device testing and network simulation |
| **Postman (if API-based)** | Backend testing for media or message APIs |
| **Excel/Google Sheets** | Backup recordkeeping and manual test logs |

**Conclusion**

In this test case, we have covered the primary functionalities of WhatsApp, such as:

* **Message sending and delivery validation** (ensuring messages are sent and received in a timely manner)
* **Media sharing** (testing the ability to send and receive images, videos, and documents)
* **Voice and video calls** (ensuring smooth connectivity and error-free performance under normal and low-bandwidth conditions)
* **User interface** and **navigation usability** (ensuring the app is intuitive and consistent)

**Outcome:**

* The **positive tests** demonstrated that WhatsApp successfully handles standard operations, including message delivery, media sharing, and calls under normal conditions.
* The **negative tests** revealed areas for improvement, including how the app manages network interruptions, unsupported file uploads, and low-bandwidth video calling.
* By documenting potential defects with clear steps to reproduce and expected outcomes, the development team can prioritize fixes in upcoming releases.