ChatGPT

FCRA Compliance Test Suite for Credit Reporting Platform

This test suite outlines comprehensive test cases to ensure that a credit reporting platform complies with the Fair Credit Reporting Act (FCRA). The cases cover key FCRA areas such as consumer consent, adverse action notices, dispute handling, data security, permissible use of credit reports, audit logging, and data accuracy. Each section below focuses on one compliance area, with test cases including positive, negative, and edge scenarios. QA engineers can use these cases for manual testing or implement them as automated tests where applicable. All expected behaviors align with FCRA requirements (e.g., only accessing reports for permissible purposes ① , obtaining necessary consent ② , timely dispute resolution ③ , etc.).

Obtaining and Verifying Consumer Consent Before Pulling Credit Reports

FCRA emphasizes that consumer reports should only be obtained with a permissible purpose and often with the consumer's knowledge or consent. In particular, for certain uses like employment screening, **written consumer consent is required** before pulling a report ② . The platform must enforce consent collection and verification to prevent unauthorized credit pulls. The following test cases validate that the system properly requests and records consumer consent prior to retrieving credit data:

- **Test Case ID: CONS-01**
  Description: *Attempt to pull a credit report without obtaining the consumer's consent (negative test).* This verifies the system does not allow credit report access if consent is missing.
  Pre-conditions:
- The consumer's account exists on the platform.
- No record of consent for pulling a credit report is on file for this consumer.
  Test Steps:
- As an authorized user (e.g. a loan officer or system process), initiate a credit report request for the consumer **without** completing any consent workflow (e.g., do not check the consent box or do not have a signed consent form on record).
- Observe the system's response to the credit report request.
  Expected Results:
- The system **blocks the request** and does **not** retrieve the credit report.
- An error message or prompt is displayed, stating that consumer consent is required before accessing the credit report.
- The credit bureau is **not** contacted for the report (verified via logs or the absence of a credit pull record).

- The workflow should direct the user to obtain consent (e.g., show a consent form or checkbox for the consumer to agree).
  Type: Manual (can be automated by simulating a credit pull API call without consent and verifying the response).

- **Test Case ID: CONS-02**
  Description: *Pull a credit report after obtaining explicit consumer consent (positive test).* Ensures the system allows report access when consent is properly given and recorded.
  Pre-conditions:

- Consumer has provided consent (e.g., they checked the required consent box or signed the electronic form). This consent record is stored in the system (in the user's profile or consent log).
- The user initiating the pull has a permissible purpose (e.g., processing a loan application) to access the report.
  Test Steps:
- Have the consumer grant consent through the platform's UI (for example, by accepting terms or signing a consent form for credit report access).
- As an authorized user, initiate a credit report pull for the consumer after the consent is recorded.
- Monitor the system processing and retrieval of the credit report.
- Verify that the consent status is logged with a timestamp when the report is pulled.
  Expected Results:
- The credit report request proceeds **successfully**, since consent is on file.
- The consumer's credit report data is retrieved from the credit bureau and displayed/stored as expected.
- The system logs include an entry that consent was verified (e.g., "Consent on file – OK to pull report," with date/time).

- No errors or consent prompts appear, as consent was already obtained.
  Type: Automated (the consent-given scenario and successful pull can be scripted via UI or API automation, with log verification possibly done manually or via log parsing).

- **Test Case ID: CONS-03**
  Description: *Verify consent record retention and auditability (positive test).* Checks that once consent is given, it is properly recorded and can be audited, and it's associated with any credit report pulls.
  Pre-conditions:

- Consumer consent for credit report access has been obtained and stored (from a prior operation, e.g. CONS-02).
- There is at least one credit report pull associated with this consent.
  Test Steps:
- Navigate to the consumer's profile or the admin consent management area in the platform.
- Locate the consent record for credit report access (it should include details like date/time of consent, method of consent, what it covers).
- Verify that the consent record is linked to the credit report request in the audit logs or history (e.g., the report pull event references the consent record ID).
- Attempt to generate a report of all consents given by the consumer or all report pulls and their consent status (if such reporting is available to admins).
  Expected Results:
- The consumer's consent record is present, with correct details (name, date, purpose of consent).
- The consent record indicates it is **valid** (not expired or revoked) at the time of the report pull.
- Any audit log or history shows that for the date of the credit pull, there was a valid consent in place (for example, an entry "Credit report accessed on 2025-03-01 – Consent ID 12345 verified").

- There is **no** indication of missing consent for any accessed report; if the system provides a consent audit report, the consumer’s entries show compliance.

- Only authorized personnel can view the consent details (to protect privacy).
  Type: Manual (requires checking UI screens or admin reports; could be partially automated if the system provides an API for consent records).

- **Test Case ID: CONS-04**
  Description: *Consent revocation or expiration handling (edge case).* Tests how the system behaves if a previously given consent is no longer valid – for example, the consumer revoked consent or the consent was only valid for a one-time pull.
  Pre-conditions:

- Consumer had given consent, but the consent is now marked as revoked or expired in the system (perhaps an admin or the consumer themselves set a revocation, or a time-limited consent expired).
- The platform recognizes consent validity and can mark it as inactive.
  Test Steps:
- Attempt to pull the consumer’s credit report after the consent has been revoked or expired.
- Observe if the system detects that there is no active consent.
- If the platform has a consent renewal request process, attempt to trigger it (e.g., see if it prompts for new consent).
  Expected Results:
- The credit report request is **denied or halted** because the previously stored consent is no longer valid.
- An informative message is provided (e.g., “Consumer consent has expired or been withdrawn. New consent is required to access the credit report.”).
- No credit data is retrieved from the bureau without fresh consent.
- If the consumer subsequently re-consents (not part of this test case but as a follow-up), the system should then allow the pull, confirming the consent check is real-time.
  Type: Manual (setting up a revoked/expired consent scenario might require admin intervention; the validation of the blocked access can be observed in the UI or logs).

Proper Delivery of Pre-Adverse and Adverse Action Notices

When information from a consumer’s credit report leads to a negative decision (e.g., denial of credit or adverse employment decision), FCRA mandates that the person be notified. Typically this involves a **pre-adverse notice** (for employment) or notice of adverse action that informs the consumer of the decision and their rights, including a copy of the credit report and the FCRA Summary of Rights before final adverse action [4]. After a waiting period or if the decision is finalized, an **adverse action notice** must be sent with specific details (credit score used, contact info of the CRA, statement of consumer rights to dispute, etc. [5]). The following test cases ensure the platform properly generates and delivers these notices:

- **Test Case ID: AAN-01**
  Description: *Generation of pre-adverse action notice prior to adverse decision (positive test).* Ensures that when a decision is pending adverse (e.g., a loan denial or job rejection based on credit data), a pre-adverse notice is prepared and delivered to the consumer with required attachments.
  Pre-conditions:

- The platform has identified a scenario where adverse action may be taken based on a credit report (for example, a background check module flagged the consumer's credit report with issues that would lead to not hiring, or a loan application is about to be denied due to credit score).
- The consumer's contact information (email or mailing address) is on file for notice delivery.
  Test Steps:
- Trigger a situation that causes a pending adverse decision. For instance, run a credit check that results in a credit score below the lender's threshold, which normally would lead to denial.
- In the system's decision workflow, move to the step where adverse action would occur (but do not finalize the denial yet, to simulate the pre-adverse stage).
- Verify that the system generates a **pre-adverse action notice** addressed to the consumer. This notice should include a copy of the credit report and the "Summary of Your Rights Under the FCRA" as attachments (or as part of the content), as required before taking adverse action [4] .
- Check the method of delivery: if email, intercept the email; if via user portal, see the notice in the consumer's notifications; if mail, ensure it is queued for printing.
  Expected Results:
- A pre-adverse action notice is created **immediately** after the adverse decision trigger, but **before** any final adverse action is executed.
- The notice clearly states that an adverse decision is under consideration based on information in the consumer's credit report.
- The notice includes **copies of the credit report and FCRA rights summary** (either attached or linked for the consumer to access).
- The consumer is informed that they have the right to review the report and dispute any inaccurate information before a final decision is made.

- The system records that a pre-adverse notice was sent (timestamp, method, and content logged for compliance audit).
  Type: Manual – verify content and attachments of the notice (though generation trigger can be automated, the content review is manual).

- **Test Case ID: AAN-02**
  Description: *Delivery of adverse action notice after decision is finalized (positive test).* Checks that once an adverse action (denial of credit, etc.) is confirmed, the system sends a compliant adverse action notice to the consumer with all required details.
  Pre-conditions:

- The pre-adverse notice process (AAN-01) was completed and a sufficient time window has passed (e.g., 5 business days for employment scenarios) or the consumer has had the opportunity to respond.
- The decision has now been finalized as adverse (e.g., the loan is formally denied or the job offer is rescinded).
- Templates for adverse action notices are configured in the system with all legally required information.
  Test Steps:
- Finalize an adverse decision in the platform for the test consumer (e.g., mark the loan application as "Denied due to credit" or mark the background check as "Fail").
- Confirm that the system generates an **adverse action notice** to the consumer.

- Inspect the content of the adverse action notice. It should include:
    - The reason for the adverse action (e.g., "Your credit report from [CRA Name] was used in making this decision").
    - The consumer's credit score that was used, if a score was a factor [5].
    - The name, address, and contact phone of the credit bureau (CRA) that provided the report [5].
    - A statement that the CRA did not make the decision and cannot explain the reasons for the denial [6].
    - Notice of the consumer's right to obtain a free copy of their credit report from that CRA within 60 days and to dispute any inaccurate information with the CRA [7].
- Verify the notice was sent via the intended channel (email, mail, portal) and logged.
  Expected Results:
- The adverse action notice is **sent to the consumer promptly** after the decision (typically the same day of decision finalization).
- All required details are present in clear language. For example, *"We obtained your credit report from TransUnion, 555-1234. TransUnion did not make this decision and cannot give specific reasons for it. You have the right to a free copy of your report within 60 days and to dispute any information you believe is inaccurate."*
- The notice meets delivery format requirements (e.g., mailed letter or properly formatted email/PDF).
- The system's compliance log reflects that an adverse action notice was issued, with date/time and a reference to the content or template version used.

- **No PII leaks**: The notice should not reveal sensitive information beyond what is necessary (e.g., it might include partial reference numbers but not full account numbers aside from the credit report content).
  Type: Manual (to review content correctness), with potential Automation for checking the event of notice dispatch in system logs.

- **Test Case ID: AAN-03**
  Description: *Prevent final adverse action without pre-adverse notice (negative test).* Ensures the system enforces the sequence of notices by blocking an adverse action decision if a pre-adverse notice hasn't been sent when required.
  Pre-conditions:

- The platform is configured to require a pre-adverse notice for certain adverse actions (especially for employment decisions or internal policy for credit denials).
- A scenario exists where a user might attempt to directly mark an application as "Denied" without going through the pre-adverse notice flow.
  Test Steps:
- Attempt to take an adverse action on a consumer's application **without** having sent a pre-adverse notice. For example, as a manager user, directly try to mark a background check as failed and finalize the rejection in one step.
- Observe the system's behavior when skipping the pre-adverse step.
  Expected Results:
- The system should **not allow** the adverse action to be finalized if a pre-adverse notice is required and has not yet been sent.
- An alert or error is shown: e.g., "Cannot finalize denial – pre-adverse notice not yet sent to consumer."

- The workflow should guide the user to send the pre-adverse notice first (possibly by providing a button or automatic generation at this point).
- No adverse action notice is sent either, since the final decision didn't go through.

- The adverse decision remains in a pending state until the proper notice procedure is followed.
  Type: Automated or Manual. (This can be tested via the UI by attempting the action, or via an API call that tries to finalize an adverse action and expecting a failure response.)

- **Test Case ID: AAN-04**
  Description: *Validate adverse action notice content for compliance (edge/validation test).* Manually verify that the generated notices contain all elements mandated by FCRA and are accurate.
  Pre-conditions:

- An adverse action notice (from AAN-02) has been generated for a test consumer.
- The notice template used by the system is accessible or the delivered notice is captured.
  Test Steps:
- Retrieve the actual adverse action notice that was sent to the test consumer (e.g., open the email or document).
- Verify each component of the notice against legal requirements:
    - Does it clearly state the adverse decision and reference the consumer's application or account?
    - Does it provide the **CRA's name, address, phone**? (e.g., "Experian, [address], [phone]").
    - Does it include the consumer's credit score used in the decision (if applicable)?
    - Is the **non-involvement disclaimer** present (CRA didn't make the decision)?
    - Does it inform the consumer of their right to a free copy of the report within 60 days and the right to dispute inaccuracies?
    - If the notice is electronic, are links provided for the consumer to easily obtain their report or the Summary of Rights?
- Check that the notice is free of errors such as incorrect names, dates, or missing fields (e.g., no placeholder text like "{CRA_Name}").
  Expected Results:
- The adverse action notice content is **complete and correct** per FCRA guidelines. All required information is present and truthful.
- The language is clear and consumer-friendly (no confusing jargon).
- The notice references the correct CRA and includes how to contact them.
- If any required element is missing or wrong, that would be a defect (for example, omitting the credit score or failing to mention the 60-day right to a free report would violate FCRA).
- This test essentially serves as a content **validation** of the notice template.
  Type: Manual (content review).

Procedures for Handling Consumer Disputes and Correction of Inaccurate Information

Consumers have the right to dispute inaccurate or incomplete information on their credit reports, and the platform must facilitate this in compliance with FCRA. The law requires disputes to be investigated promptly (generally within 30 days) and any inaccuracies to be corrected or deleted [8] [9]. The system should allow consumers to submit disputes, track their status, and ensure corrections flow through. The following test cases cover the dispute process from submission to resolution:

- **Test Case ID: DIS-01**
  Description: *Submit a credit report dispute and verify acknowledgment (positive test).* Ensures the platform accepts a consumer's dispute request about an item on their credit report and provides confirmation.
  Pre-conditions:
- The consumer has an active account on the platform and has recently obtained their credit report through it.
- The credit report contains at least one item that the consumer deems inaccurate or incomplete (e.g., an account with an incorrect balance or a record that doesn't belong to them).
  Test Steps:
- Log in as the consumer (or use a consumer-facing portal) and navigate to the credit report details page.
- Locate the specific entry/information the consumer wants to dispute (for example, a tradeline or collection entry).
- Use the platform's dispute function (e.g., a "Dispute this item" button or form). Enter the dispute details, explaining why the information is wrong and providing any supporting documentation if the UI allows attachments.
- Submit the dispute through the platform.
- Observe the immediate response/acknowledgment from the system.
  Expected Results:
- The platform accepts the dispute submission **without errors**.
- The consumer sees a confirmation message or reference number indicating the dispute has been logged (e.g., "Your dispute has been submitted. Dispute ID: XYZ123").
- The disputed item is marked in the consumer's report view as "In Dispute" (optionally, the UI could highlight or flag the item to indicate the status).
- An entry is created in the system's dispute tracking log, including the date of submission and details of the disputed item.

- (If applicable, an automated communication is sent to the consumer's email confirming receipt of the dispute.)
  Type: Manual (submission via UI), with Automated verification possible for the presence of the dispute record in the database or via an API.

- **Test Case ID: DIS-02**
  Description: *Dispute investigation and resolution within 30 days (positive test).* Verifies that the platform completes the dispute resolution process in a timely manner and updates the information if needed, as per FCRA's 30-day requirement [3].
  Pre-conditions:

- A dispute has been logged for a consumer (from DIS-01) and is pending investigation.
- The current date is within a few days of the dispute submission (to simulate progression within the 30-day window).
- The system is integrated with the processes for investigating disputes (either via automated workflows or manual intervention by a credit reporting agency staff).
  Test Steps:

- (This step may be performed by an admin or via a back-end simulation) Simulate the passage of time or progress the system's workflow to near the 30-day mark after dispute submission.
- Ensure that the dispute is processed. For example, have a data furnisher or admin update the item's status: if the information was found inaccurate, correct it; if verified as accurate, mark the dispute as resolved with no change. (In a test environment, this could be done by calling an API or running a script to mimic the bureau's response.)
- Trigger the platform's logic that finalizes the dispute outcome and notifies the consumer.
- Check the consumer's credit report on the platform after resolution and any notifications sent.
  Expected Results:
- The dispute investigation is **completed within 30 days** of the initial dispute date (the test should ensure the date of resolution logged is ≤ 30 days from submission).
- If the information was indeed inaccurate or incomplete, the platform shows the corrected information or indicates the item was removed. For example, if a wrong late payment existed, it is now deleted or corrected in the report view [9] .
- If the information was verified as correct (no change), the item remains, but the dispute is marked as resolved. The consumer's report may include a note like "Consumer disputed this item; it was verified as accurate."
- The consumer receives a resolution notification. This could be an email or a secure message stating the outcome (e.g., "Your dispute has been resolved. Outcome: The disputed account has been updated/corrected." or "Outcome: The disputed information was verified by the data furnisher and remains unchanged. You may add a statement to your file if desired.").
- The system provides the consumer with an updated copy of their credit report if changes were made (this updated report should not count against any annual free report limit, as per FCRA when corrections occur [10]).

- All these actions are logged for compliance (date of resolution, outcome, communications sent).
  Type: Manual for orchestrating the end-to-end process (or using a staging environment with known data). Automated verification can check that the dispute status changes appropriately in the database and that notifications are sent.

- **Test Case ID: DIS-03**
  Description: *Dispute remains unresolved after 30 days (negative/edge case).* Tests that the system handles a breach of the 30-day resolution window by escalating or notifying appropriately (though in practice the goal is to avoid this).
  Pre-conditions:

- A dispute exists and 30 days have passed without a resolution (perhaps in a test scenario, no action was taken).
- The system's clock or test data is set such that the dispute creation date is >30 days in the past.
  Test Steps:
- Identify a disputed item that has exceeded the 30-day investigation period without a marked resolution.
- Observe any automated system behavior once the 30-day limit is crossed. For example, does the system automatically flag the dispute as overdue, or send an alert to administrators?
- If no automatic action, attempt to pull a new credit report for the consumer after the 30 days to see if the disputed item remains and how it's labeled.

- Also, check if any communication was sent to the consumer apologizing for delay or providing an update.

  Expected Results:
- The system should **not silently ignore** an overdue dispute. One of the following should happen:
  - The dispute is auto-escalated to a supervisor or compliance team for investigation.
  - The system automatically sends a notification to the consumer acknowledging the delay and possibly providing an update or new timeline.
  - If the platform has the capability, it might temporarily remove the disputed information from the view (to avoid continuing to report it as it's unverified beyond 30 days), though typically the CRA must delete or update unverified info by law [9].
- In logs, the dispute is flagged as "**Overdue**" or similar, indicating a compliance issue that needs attention.

- If the test simulates a real scenario, the expected result is that this situation should not occur under normal operation (it represents a failure to comply). The presence of an overdue dispute in a real audit would be a critical defect.

  Type: Manual (requires setting up a time-lapse scenario; can be partially automated by adjusting system date in a test environment if possible).

- **Test Case ID: DIS-04**

  Description: *Handling of frivolous or duplicate disputes (negative test).* Ensures the platform can identify if a consumer dispute is substantially the same as a previous one or lacks merit (frivolous), and responds according to FCRA guidelines (e.g., notifying the consumer of refusal to investigate because it's frivolous) [11].

  Pre-conditions:

- The consumer has submitted at least one dispute on an item already (which has been investigated or is in process).
- The consumer attempts to submit another dispute on the same exact item *before* any new information is available, or submits a clearly frivolous claim (for example, no details or evidence for the dispute, or an outlandish claim unrelated to the credit report).

  Test Steps:
- As the consumer, try to file a duplicate dispute for an item that is already under dispute or was resolved recently (e.g., dispute the same account again immediately, claiming the same issue).
- Alternatively, submit a dispute with obviously insufficient information (e.g., blank reason or gibberish text that doesn't pertain to the report).
- Submit the dispute request and observe the platform's response.

  Expected Results:
- The system should either **prevent the duplicate submission** or accept it but quickly flag it as frivolous/duplicate.
- If prevented: a message appears like "You already have an open dispute for this item. Please wait for the current dispute to be resolved before submitting another."
- If accepted then marked frivolous: the system may internally mark it and **notify the consumer** that it won't be investigated because it's duplicative or frivolous [11]. For instance, an email or message: "Your recent dispute has been deemed frivolous or duplicative and will not be processed. Reason: Duplicate dispute. If you have new information, please provide it."
- No new investigation timer is started for frivolous disputes (the original dispute timeline remains in effect if one was ongoing).

- All such actions (and reasons for marking as frivolous) are logged.
  Type: Manual (need to simulate user input of a frivolous dispute), with some Automated logic verification if accessible (e.g., checking a flag in the dispute record).

- **Test Case ID: DIS-05**
  Description: *Consumer adds a statement to file after dispute (edge case).* If a dispute does not resolve to the consumer's satisfaction (information remains, but consumer disagrees), FCRA allows the consumer to add a brief statement to their file [10]. This test checks that the platform supports this.
  Pre-conditions:

- A dispute was investigated but the outcome was that the information was verified (no change), or otherwise not fully resolved in the consumer's favor.
- The dispute case is now closed/resolved in the system.
  Test Steps:
- As the consumer, use the platform's interface to add a statement of dispute to the credit report. This might be a feature like "Add Personal Statement" after a dispute resolution. (If the platform doesn't allow direct input, perhaps via contacting support. For test, assume an input method exists.)
- Enter a brief statement (e.g., "Consumer states that the XYZ account is not theirs, despite verification."), respecting any character limit.
- Submit the statement.
- Pull the consumer's credit report through the platform or view the report again.
  Expected Results:
- The platform accepts the consumer's statement and confirms it's added to their file.
- The consumer's credit report view now includes the statement as a notation on the disputed item (or in a special "Consumer Statements" section of the report). For example: *"Consumer Statement: I dispute the accuracy of the XYZ account information."*
- If the platform provides the report to third parties (permissible users) thereafter, the statement should be included or summarized as per FCRA (usually CRA must include a notice of the consumer's statement in future reports) [12].
- This is logged in the system that the consumer provided a statement on date X, for audit purposes.
- The consumer can edit or remove their statement through the platform if needed (optional test not covered here).
  Type: Manual (requires UI interaction and checking report output).

Security and Privacy of Consumer Credit Data

Security and privacy are paramount since credit reports contain sensitive personal and financial information. The platform must enforce strict data access controls, encryption, and safe handling of this information, in line with FCRA and other data protection standards. FCRA expects CRAs and related systems to **protect consumer information from unauthorized access and data breaches** [13]. The following test cases cover authentication, authorization, data handling, and privacy measures:

- **Test Case ID: SEC-01**
  Description: *Authorized user access only (positive security test).* Verify that only authenticated, authorized users can access a consumer's credit report data on the platform.
  Pre-conditions:
- A valid user account with proper credentials (username/password, 2FA if enabled) exists.

- The user account has the role or permission to view credit reports (e.g., a loan officer role, or the consumer viewing their own report).
- A target consumer's credit report is available in the system (already pulled or ready to be pulled by authorized users).

  Test Steps:
- Log in to the platform with the authorized user's credentials.
- Navigate to the section where credit report information is displayed (for example, open a customer's application and click "View Credit Report").
- Ensure the page loads the credit report data.
- Log out, then try to directly access the same credit report URL (copy the URL from step 2) without being logged in.

  Expected Results:
- While logged in with valid credentials, the user can **successfully view** the consumer's credit report details (assuming they have a legitimate need).
- After logging out (or if the session expires), any attempt to access the credit report page or endpoint is **denied** (the user should be redirected to a login page or see an access denied message).
- The system should not allow unauthenticated sessions to retrieve any credit data (protecting privacy).

- All access requires going through proper login; session management should ensure no cached credentials allow re-entry without login.

  Type: Automated (login and access can be done via script, and a subsequent unauthenticated request should return a 401/redirect which can be checked).

- **Test Case ID: SEC-02**

  Description: *Role-based access control and least privilege (positive/negative test).* Ensure that users only see credit data if their role permits it, and sensitive fields are masked when appropriate.

  Pre-conditions:

- Different user roles exist (e.g., *LoanOfficer* with access to credit reports for their applicants, *CustomerServiceRep* with partial access, *Consumer* who can only see their own report, *UnauthorizedUser* with no access).
- The consumer's credit report data includes sensitive fields like Social Security Number (SSN), account numbers, etc.

  Test Steps:
- Log in as a user with full credit report access (e.g., LoanOfficer role). View a consumer's credit report.
- Verify that the user can see all relevant details (full tradeline info, etc.), except possibly certain highly sensitive data might be masked if not needed (for instance, maybe only last 4 digits of SSN are shown even to staff).
- Log in as a user with partial access (e.g., CustomerServiceRep). Attempt to view the same consumer's credit report.
- Log in as a user with no credit report access permission. Try to view the credit report (or via direct link).
- Also, test the consumer's own view: log in as the consumer themselves and view their report – ensure they see their information fully (except maybe their SSN is partially masked for security even to themselves?).

  Expected Results:

- The **authorized role** (LoanOfficer) can view the credit report section and sees data needed for their job. Sensitive identifiers (like SSN) are displayed according to policy (e.g., masked except last 4).
- The **limited role** (CustomerServiceRep) might see a redacted version or only certain sections of the report. If policy dictates, some fields or entire access may be read-only or restricted. For example, they might see credit summary info but not full account details.
- The **unauthorized user** either does not see the menu/tab for credit reports at all, or if they attempt direct access, they receive an “Access Denied” message. They cannot retrieve the data.
- The **consumer’s view** shows their own report. Typically, the consumer should see all their information, though sensitive personal info might still be partially masked to prevent shoulder-surfing risks. They should certainly see account and inquiry information pertaining to them.

- No role can see another consumer’s report unless it’s their job function and they have permissible purpose – this is ensured by backend checks, not just UI hiding.
  Type: Manual (to switch roles and visually confirm data access), possibly automated with different test accounts for each role and verifying API/UI responses contain or don’t contain certain fields.

- **Test Case ID: SEC-03**
  Description: *Data transmission and storage security (positive test).* Verify that sensitive credit data is encrypted in transit and at rest, and not exposed in logs or through the UI.
  Pre-conditions:

- The platform is deployed using HTTPS for all web traffic.
- There is access to logs or a way to inspect network calls for testing (e.g., browser developer tools or intercepting proxy, and application logs for sensitive events).
  Test Steps:
- While viewing a consumer credit report on the platform, check the browser’s address bar to confirm the connection is HTTPS (secure).
- (If possible, use a network analysis tool) Intercept the network traffic for the credit report request. Verify that the data is transferred over TLS and not in plaintext.
- Inspect application logs (or enable debug logging for the test) for any entries when the credit report was retrieved.
- Look for sensitive data in the logs (e.g., the system should not be logging the full content of the credit report or personal data in plaintext).
- If the platform stores the credit report data in a database, one might verify through a database admin (in a non-production test environment) that the data is encrypted or at least hashed where appropriate (this step might be more of a design review than a dynamic test).
  Expected Results:
- **Encryption in transit:** All communication channels used to fetch or display credit data are secure (HTTPS). No credit data is ever sent over an insecure protocol. If an intercept proxy is used, the data appears encrypted (or gibberish) unless the proxy has the appropriate certificate to decrypt TLS.
- **Secure storage:** There is evidence (from documentation or testing) that the credit data is encrypted at rest. For example, if we query the database directly, stored SSN or account numbers might be encrypted (unreadable without the application). *Note: This might be verified via a documentation review if direct DB checking is not feasible.*
- **No sensitive logging:** Application logs do not contain raw personal data from credit reports. At most, they might have an event “Credit report retrieved for user X” with an ID, but not the report content or PII.

- **Session security:** The user's session token or cookies are secure (HttpOnly, Secure flags, etc.) – although not explicitly asked, it's part of security best practices. (This could be a separate security test.)
  Type: Manual/Automated. (Manual to check network via developer tools; automated security scanning tools might also verify HTTPS usage. Checking logs may require manual inspection or scripts.)

- **Test Case ID: SEC-04**
  Description: *Session timeout and data cache handling (edge case).* Ensure that if a user is inactive or logs out, the credit report data is no longer accessible, and it's not left in browser cache in a way that another person could view it.
  Pre-conditions:

- A user is logged in and has viewed a credit report page.
- Session timeout is configured (say 15 minutes of inactivity) in the platform.
  Test Steps:
- While logged in and viewing a credit report, leave the session idle for longer than the timeout period (or manually expire the session via admin).
- Attempt to interact with the page or navigate to a different section after the timeout.
- Also, after logout or timeout, click the browser's "Back" button to see if the credit report page is still viewable from cache.
- Close the browser, reopen it, and try to directly navigate to the credit report URL from history.
  Expected Results:
- After the session times out, any action triggers a redirect to the login page. The credit report data should **not** be visible without re-authenticating.
- Using the browser's back function **should not** reveal the data. Ideally, the app should have cache-control headers to prevent caching of sensitive pages, or it should immediately redirect to login on any attempt to view after logout. The page might briefly flash but immediately hide or redirect.
- If the user closes the browser and reopens, the previous page should not be accessible at all (session is gone, and cached page either not available or requires login on refresh).

- This ensures privacy that another person can't walk up to an unlocked computer or use backdoors to see someone's report after the fact.
  Type: Manual (simulate user inactivity and use browser controls).

- **Test Case ID: SEC-05**
  Description: *Penetration of access controls (negative security test).* Attempt to bypass security by using direct resource access or parameter tampering to fetch a credit report that the user should not see.
  Pre-conditions:

- Two test accounts: User A (authorized to see only their own or certain reports) and User B (whose report User A should not access).
- Knowledge of resource identifiers (e.g., credit report IDs or user IDs).
  Test Steps:
- Log in as User A.

- Attempt to access User B’s credit report by manipulating a request. For example, use an API endpoint or URL intended to fetch a report and substitute User B’s customer ID or report ID. (This assumes one knows or guesses another ID – simulate that the ID might be obtainable or guessable).
- Try various methods: direct URL entry, AJAX call in dev console, or a custom API client with User A’s token calling a report retrieval for User B.
  Expected Results:
- The platform’s backend should detect that User A is not permitted to access User B’s data and **reject the request**. This could be a 403 Forbidden error or some generic “not allowed” message.
- The response should contain no credit data of User B.
- An audit log entry might be created for the attempted unauthorized access (for security monitoring), possibly flagging User A’s account or IP if it looks malicious.
- Under no circumstances should sensitive data leak to User A. Even if IDs are predictable, the authorization check must be enforced server-side.
  Type: Automated (security test scripts can attempt such access and verify HTTP 403/401 responses), with Manual verification of logs if needed.

Timely Response to Consumer Inquiries and Disputes

Consumers may contact the platform or CRA for various inquiries – requesting their own report, asking who has accessed their report, or general questions – and FCRA expects timely responses. For disputes, as noted, resolution should be within 30 days [8]. For general inquiries, while FCRA doesn’t specify an exact timeframe, good practice and related regulations require prompt attention (for example, providing the consumer’s file upon request usually within 15 days, and listing recent inquiries upon request [14]). The following test cases ensure the platform responds to consumer requests and inquiries in a timely and compliant manner:

- **Test Case ID: TIM-01**
  Description: *Consumer requests a copy of their credit report (positive test).* Verify that a consumer can obtain their own credit report through the platform quickly (at least one free copy annually, per FCRA).
  Pre-conditions:
- The consumer has an active account on the platform and has passed identity verification.
- The platform offers a feature for the consumer to request or download their credit report file (for personal review).
  Test Steps:
- Log in as the consumer. Navigate to the section where they can request their credit report (e.g., “Get my report” or “Download my report” feature).
- Initiate a request for the credit report. This might be an on-demand generation or a stored file if one was pulled recently.
- Measure the time taken for the report to be delivered or displayed.
- If the platform emails the report or provides a PDF, check the delivery medium (email inbox or download link).
  Expected Results:
- The consumer is able to request **at least one free copy of their credit report** through the platform, in line with their FCRA rights (typically one free report every 12 months, aside from any post-adverse-action free report) [14].

- The report is delivered **promptly**. Ideally, it should be available immediately for download or view. If there is a delay (for example, the system retrieves fresh data), it should be completed within minutes, not days.
- The delivered report contains all information in the consumer's file and is the same as what a third-party would see (full disclosure).
- If the consumer has recently received a report and is not eligible for another free one, the system either warns about possible charges or requires confirmation (not directly an FCRA requirement to charge, but just in practice). For the first free request, no charge should be applied.

- The platform logs that the consumer accessed their report on this date (for audit and so that this event itself isn't counted as a "third-party inquiry").
  Type: Manual (triggering via UI) or Automated (simulate consumer login via API if available and request report).

- **Test Case ID: TIM-02**
  Description: *Consumer requests the list of recent inquiries (positive test).* Ensures the platform can provide the consumer with a list of everyone who accessed their credit report, as required by FCRA (typically the past 1 year for credit inquiries, 2 years for employment inquiries) [14] .
  Pre-conditions:

- The consumer has had their credit report accessed by various parties (inquiries). For testing, ensure there are some inquiry records (could be from previous credit pulls by lenders, etc., logged in the system).
- A feature exists for the consumer to view or request the list of recent credit report inquiries.
  Test Steps:
- Log in as the consumer and navigate to a "Recent Inquiries" or "Who has viewed my credit report" section. (If not directly visible, perhaps by requesting a full file disclosure which includes the inquiry list.)
- Request the display or download of all recent inquiries.
- Verify the information presented to the consumer.
  Expected Results:
- The platform provides a **list of all entities who obtained the consumer's report**, including the name of the entity and date of inquiry, covering at least the past 12 months (and highlighting any employment-related inquiries in the past 24 months, if applicable) [14] .
- The list should match the audit trail: for example, if in earlier tests various users pulled the report, those users or companies appear on this list with correct dates.
- No unauthorized inquiries are present (i.e., the consumer should recognize the entries or it's part of legitimate activity; this is more a data check, but if something unknown appears it could indicate a problem).
- The consumer is not shown data beyond what's allowed (e.g., they won't see inquiries older than FCRA retention period for disclosure, typically 1–2 years). The platform should have filtered out any older records if they exist.

- The format is user-friendly (perhaps sorted by date, showing company names).
  Type: Manual (check via UI, or via a report file), could be Automated if the platform has an API for consumer disclosure.

- **Test Case ID: TIM-03**
  Description: *Timely initiation of dispute investigation (positive test).* Verify that when a consumer files a dispute, the platform (or CRA) initiates the investigation quickly and notifies data furnishers promptly, rather than waiting.
  Pre-conditions:

- A dispute was submitted by the consumer (as in DIS-01).
- The dispute is currently logged as open.
  Test Steps:
- Check the timestamp of when the dispute was submitted.
- Check internal logs or status to see if the platform has forwarded the dispute information to the credit bureau or data furnisher. (For example, the system might have an integration that sends the dispute details to the credit bureau within a day or in real-time.)
- If possible, query the status of the dispute through an admin interface or API soon after submission.
  Expected Results:
- The system forwards the dispute details to the relevant parties (credit bureau and/or creditor) **without delay**, typically the same day or within 1-2 business days of submission. This ensures the 30-day clock is utilized fully for investigation.
- The dispute status might update to "In Progress" shortly after submission, indicating it's been acknowledged by the system and likely sent out for investigation.
- The consumer may receive a confirmation that the investigation has started (some systems send a second notice once the bureau/furnisher has been notified).
- There should be no large gap (e.g., it shouldn't wait 10 days before even starting). If our test checks a few hours or next day and sees it in progress, that's good.

- All actions are logged with timestamps (e.g., "Dispute received on 2025-04-01 10:00, forwarded to Experian on 2025-04-01 10:05").
  Type: Manual/Automated (checking timestamps could be automated via database queries if accessible; otherwise, rely on logs or UI indicators manually).

- **Test Case ID: TIM-04**
  Description: *Response to a general consumer inquiry via support (edge case).* Test that if a consumer uses a support channel (contact form or email) to ask a question related to their credit report, the platform responds within a reasonable timeframe and addresses the inquiry.
  Pre-conditions:

- A support mechanism exists (e.g., "Contact us" form or support email/ticket system integrated with the platform).
- The consumer has a question, for example: "I see an account on my report that I don't recognize. What should I do?" (Not a formal dispute via the dispute flow, but a general inquiry possibly preceding a dispute.)
  Test Steps:
- Submit an inquiry through the support channel as a consumer. Provide identifying info and the question. Mark the time of submission.
- Check the support queue or status (if available to testers) to ensure the inquiry is logged.
- Wait for the response. Record the time and content of response when received. (You might simulate the support agent answering if this is a test environment, or have an SLA in place.)
  Expected Results:

- The inquiry is logged and a ticket/reference number is generated. The consumer might get an automated reply immediately: “We have received your question. Our team will respond within 2 business days.”
- The platform’s support team responds **within the advertised SLA** (for instance, if policy is 1-2 business days, the response comes in that time frame). Even though not explicitly an FCRA requirement, timely customer service aligns with good compliance practice.
- The content of the response should be helpful and address the question. In this example, it might guide the user on how to dispute the account or verify if it’s an error, thus steering them into the formal dispute process if needed.
- If the inquiry was in fact a request for their credit file or who accessed it, the support should handle it as a formal request under FCRA and provide the information or direct the consumer to the correct self-service feature (as tested in TIM-01 and TIM-02).
- All correspondence is documented in the system, in case of later audit or follow-up.
  Type: Manual (since it involves human response, unless a template auto-response, it can’t be fully automated).

Ensuring Only Permissible Purposes for Accessing Credit Data

FCRA strictly limits access to credit reports to **permissible purposes** – such as evaluating a credit application, account review, employment screening, tenant screening, insurance underwriting, etc. [1]. The platform must enforce that every credit report pull is tagged with an allowed purpose and that no user can pull a report without certifying a permissible reason. The following test cases verify these controls:

- **Test Case ID: PERM-01**
  Description: *Require permissible purpose selection for credit pull (positive test).* Ensures the UI/API forces the user to choose a reason for accessing a credit report and that reason is valid under FCRA.
  Pre-conditions:
- The platform’s credit report request interface is configured with a dropdown or field for “Permissible Purpose” (e.g., options like *Credit Application*, *Account Review*, *Employment*, *Tenant Screening*, etc.).
- The user has a need to pull a report (e.g., processing a loan application).
  Test Steps:
- Initiate the credit report pull process for a consumer.
- Observe the form or API parameters required. Attempt to submit the request **without** selecting a permissible purpose (or with a blank purpose field).
- Attempt to submit the request **with** a valid permissible purpose selected (e.g., “Credit transaction – extension of credit”).
  Expected Results:
- If the permissible purpose is not provided, the system **rejects** the request, prompting “Please select a permissible purpose for accessing the credit report.” The request should not go through without this selection.
- When a valid purpose is selected and the request is submitted, the system accepts it and proceeds to pull the report (assuming all other conditions like consent are met).
- The permissible purpose is recorded along with the credit pull event (for audit). For example, the log might show “Permissible Purpose: Credit Application” attached to that access.

- The list of purposes available does not include any inappropriate options. All options should correspond to those allowed by FCRA (no option for something like “Personal curiosity” or anything outside the scope).

Type: Manual (to try submitting the form with and without the field) or Automated (via form automation or API testing to ensure the parameter is required).

- **Test Case ID: PERM-02**
  Description: *Block access for impermissible purpose (negative test).* Try to use a purpose that is not allowed and ensure the system prevents the credit report retrieval.
  Pre-conditions:

- The platform might internally define some purposes as disallowed, or we simulate an attempt via API by injecting a non-allowed purpose code. (E.g., using an API call with purpose “Test/Other” that isn’t on the official list.)
  Test Steps:
- Use an API tool or manipulate the web request to provide a purpose value that is not in the permissible list (for example, an invalid code or a dummy string).
- Submit the credit report pull request with this invalid purpose.
  Expected Results:
- The request is **rejected**. The system returns an error response or message like “Invalid purpose specified” or “Unauthorized purpose – credit report not obtained.”
- No credit inquiry is sent to the bureau (so the consumer’s credit file should not show any record of an inquiry for this failed attempt).
- The event may be logged for security review (if someone is trying to bypass controls), but marked as blocked.

- The user interface, if used, might sanitize and not even allow such input; this test mainly ensures back-end validation is in place beyond just UI dropdown.
  Type: Automated (using API testing or dev tools to alter requests).

- **Test Case ID: PERM-03**
  Description: *Employment screening workflow with certification (edge case).* If the permissible purpose is employment, verify that the system requires the user (employer) to certify compliance (including that they have written consent) before allowing the report, per FCRA’s special requirements for employment reports [2] .
  Pre-conditions:

- The platform supports employment background credit checks as a use case.
- An employer user account is set up to request a report for a job applicant (consumer).
  Test Steps:
- As the employer user, select “Employment” as the permissible purpose for pulling a credit report on the applicant.
- Try to proceed without any further action.
- Observe if the system asks for additional certification or steps (like uploading the applicant’s signed consent, or checking a box “I certify I have obtained written consent from the applicant”).
- Complete the required certification and then submit the request.
  Expected Results:
- The system should **require** the employer to confirm they've complied with FCRA’s prerequisites for employment screening. This typically means the employer certifies: they disclosed to the applicant

that a report will be obtained, got the applicant’s written consent, and will use the information lawfully. The platform might implement this as a mandatory checkbox or form.

- If the employer tries to bypass without certifying (e.g., if the checkbox isn’t checked), the request is blocked (similar to consent tests).
- Only after certification (and possibly attaching the consent document if required) will the system allow the credit report to be pulled for employment purpose.
- The audit log for the credit pull should note that it was for employment and that certification was completed.

- If the platform doesn’t support employment checks at all, this test might be not applicable; but if it does, this is critical to test.
  Type: Manual (to simulate the employer UI flow), potential automation in verifying that without the checkbox the action cannot complete.

- **Test Case ID: PERM-04**
  Description: *Re-use of credit report data outside original purpose (negative/edge case).* Ensure that once a credit report is pulled for a specific permissible purpose, the data isn’t misused for other purposes without a new permissible purpose check.
  Pre-conditions:

- A consumer’s credit report was pulled for one purpose (e.g., Account Review for an existing customer). The data is stored in the system.
- A user attempts to use that stored data for a different purpose (e.g., marketing or another unrelated analysis) that is not originally intended.
  Test Steps:
- After having a credit report on file (from a prior permissible pull), attempt to access or export that data in a context that suggests a different use. For example, try to generate a marketing list using credit scores, or an analyst user tries to view the data outside the credit decision context.
- Alternatively, attempt to pull the report again under a different purpose without going through the consent/purpose selection (if someone tries to trick the system into using cached data for convenience).
  Expected Results:
- The system should treat each usage separately in terms of compliance. It should not allow a report pulled for one purpose to be silently reused for another **impermissible** purpose.
- If an employee tries to use the data for an unauthorized purpose, it should be considered a violation. The system might not technically prevent reading already pulled data, but this is where audit trails and policy come in (see Logging/Audit section). Ideally, the platform would have controls like “cannot export or use data unless you have a permissible reason”.
- If the test is via UI, perhaps there’s no feature to use the data elsewhere without going through proper channels. If via API, the data should be access-controlled in context.
- Expected outcome: either no direct way to misuse data exists, or if attempted, it’s flagged. In an audit, any access to stored reports is also tied to a permissible purpose (maybe the system forces the user to specify why they are viewing an old report).
  Type: Manual (this might be more of a process/policy test, but a creative manual attempt can be made to see if data can be pulled out improperly).

Logging and Audit Trails for Credit Report Access

FCRA compliance requires that every access to a consumer's credit report is tracked. This is important for audits and for consumers exercising their rights to know who accessed their data (14). The platform should maintain robust audit logs recording who, when, and why each credit report was accessed, and ensure these logs cannot be tampered with. The following test cases cover logging and audit trail features:

- **Test Case ID: LOG-01**
  Description: *Audit log entry for each credit report pull (positive test).* Verify that whenever a credit report is accessed, an audit trail entry is created with all relevant details.
  Pre-conditions:
- Audit logging is enabled in the system.
- At least one credit report access event will be performed (by any user or system process).
  Test Steps:
- Perform a credit report access – for example, have a user pull a credit report for a consumer (this can piggyback on tests like CONS-02 or PERM-01).
- After the action, access the audit log as an admin or through a database query. Filter the log for the specific event.
- Inspect the log entry for completeness.
  Expected Results:
- There is a **new log entry** corresponding to the credit report pull event.
- The entry includes: the identity of the user or system account that accessed the report, the date and time, the consumer whose report was accessed, and the permissible purpose/reason for access.
- It may also include the source (e.g., via UI or API) and an event ID.
- Example: *"2025-03-26 10:15:32 – UserID 45 (LoanOfficer JohnDoe) accessed credit report of ConsumerID 123 (Jane Smith) for purpose 'Loan Application'. Consent verified."*

- No credit report should ever be pulled without a corresponding log. If the log is missing an event, that's a defect.
  Type: Manual (checking logs via admin interface or DB). Could be automated if logs are exposed via an API or can be retrieved programmatically.

- **Test Case ID: LOG-02**
  Description: *Audit log access and reporting (positive test).* Ensure that authorized personnel can retrieve audit logs and that consumers can get a report of accesses to their data (which overlaps with TIM-02).
  Pre-conditions:

- There is a collection of audit log entries from various test events (multiple consumers, multiple users).
- An admin or compliance officer role exists to view audit logs.
  Test Steps:
- Log in as an admin/compliance officer.
- Navigate to the audit trail or reporting section of the platform.
- Search or filter for a specific consumer's ID or name to get all access events for that consumer's report.

- Also, try a general report of all credit report accesses in a date range.

  Expected Results:
- The admin user can retrieve **comprehensive logs** without any data omission. For a given consumer, every instance of their report being accessed is listed with details (who accessed, when, purpose).
- The results match known events from testing. (For example, if we pulled Jane Smith's report 3 times in tests, all 3 should show up with correct timestamps and user info.)
- The consumer-facing "who accessed my report" info (from TIM-02) should correlate with these logs, although the consumer might see a friendlier format (company names instead of user IDs).
- The system allows exporting these logs for audit (e.g., CSV download) while maintaining security (only admins can export).

- There are no privacy leaks in the admin log interface: an admin can see details necessary for compliance, but not e.g. the content of the credit report in the log (just metadata).

  Type: Manual (using UI to retrieve logs; could be partially automated with API if available).

- **Test Case ID: LOG-03**

  Description: *Log integrity and tamper-proofing (negative test).* Check that once an audit log entry is recorded, it cannot be altered or deleted by unauthorized users (and preferably not even by authorized users without trace).

  Pre-conditions:

- Audit logs from previous actions are present.
- You have at least two roles: an admin who can view logs, and a regular user. Possibly an admin function to delete logs (if it exists at all, likely it shouldn't or if it does, it's heavily restricted).

  Test Steps:
- As a regular user (who can pull credit reports), attempt to modify or delete an audit log entry. (Regular users typically have no direct access to logs, so this might involve trying to call an API endpoint if one exists or SQL injection attempts if relevant).
- As an admin, see if there is any provided functionality to edit or remove log entries. If so, attempt to use it (or if policy says logs are immutable, verify that no such function exists).
- If you have database access in the test environment, attempt a direct DB update on an audit log entry (to simulate malicious tampering) and see if checksums or other protections exist (optional advanced step).

  Expected Results:
- Regular users **cannot** alter or delete log entries. They shouldn't even be able to see them directly. Any attempt (like a crafted request) fails with access denied.
- Admin users typically should not be able to delete logs either through the UI. If the UI allows deleting for maintenance, it should require a high privilege and leave a supervisory trail. Ideally, the platform's design does not allow log deletion or editing at all, ensuring integrity.
- If an admin tries to delete a log entry via any provided interface, there should be a big warning and perhaps an audit of that action too. For compliance, normally logs are append-only.
- If we simulate DB tampering, in a real system there might be features like append-only databases or cryptographic signatures. In a test environment, it might not be evident, but you could check if any hash of logs exists. However, at minimum, after tampering, the system might break or the tampered data might be detectable.

- In summary, the logs are **write-once** for normal operations. The test passes if no one but perhaps DB admins can change them, and even then it would be outside application scope.
  Type: Manual (attempting UI operations; security testing for API), possibly penetration testing style.

- **Test Case ID: LOG-04**
  Description: *Consumer audit request vs. logs (consistency check).* Ensure that the information given to consumers about accesses (from TIM-02) exactly matches the internal audit logs, proving consistency.
  Pre-conditions:

- A consumer (say Consumer X) has had multiple credit report pulls by different parties, recorded in audit logs.
- The consumer has access to a "who viewed my report" feature.
  Test Steps:
- As an admin, retrieve all log entries for Consumer X's report accesses (similar to LOG-02). Note down the details (dates and who accessed).
- As Consumer X, request the list of recent inquiries (TIM-02) or a full file disclosure including inquiries.
- Compare the consumer-facing list with the admin log list.
  Expected Results:
- Every entry that appears in the admin audit log for Consumer X (that falls within disclosure time frames) is present in the consumer's list of inquiries. The names might be slightly different (for example, admin log might show internal user "JohnDoe" but consumer sees "Lender ABC Bank"), but they refer to the same event.
- There are **no extra entries** on one side that aren't on the other, except system-internal checks that might be logged but not reported to consumer (though typically only real pulls are logged).
- The timing and count match up. If a discrepancy is found, that indicates a logging error.
- This cross-verification confirms that the log system is reliable and also that consumer rights to know about inquiries are fulfilled from the same data source.
  Type: Manual – comparison of two outputs (admin vs consumer).

Data Accuracy and Completeness Checks

The platform must handle credit data carefully to maintain accuracy and completeness. FCRA obligates CRAs to follow **reasonable procedures to assure maximum possible accuracy** of the information [15]. While the platform might not generate the credit data (it comes from bureaus), it should present it correctly and allow corrections via disputes. These tests focus on ensuring data isn't lost or misrepresented and that corrections are updated:

- **Test Case ID: ACC-01**
  Description: *Verify all credit report fields are captured and displayed (positive test).* Ensures that when a credit report is retrieved, all relevant sections (personal info, accounts, inquiries, public records, etc.) are present and shown to the user.
  Pre-conditions:
- A sample credit report with known data is available (could be a test profile that has a variety of data: name, multiple addresses, several trade lines, inquiries, public records, etc.).
- The platform has mappings for all these data fields from the bureau format to its own display.
  Test Steps:

- Pull the credit report for the test profile (ensure the test profile's credit file contains diverse data).
- Once retrieved, inspect the UI or output where the credit report is shown.
- Cross-reference the displayed data with the source (if we have a copy of the bureau's raw data or a known reference).

  Expected Results:
- **Personal Identifying Information (PII):** The report shows the consumer's full name, Social Security Number (or at least a masked SSN), date of birth, and current & former addresses as provided by the bureau. All names/addresses on file should be listed, not just one (unless filtered by recency with disclosure to consumer).
- **Trade lines (Accounts):** Every account in the bureau data appears in the platform's report view. Each account's details (account name, number - masked, balance, status, payment history, etc.) are accurately presented and correspond exactly to the source data.
- **Inquiries:** The recent inquiries section shows all inquiries from the bureau data (soft inquiries may be separate if the platform differentiates, but for a full consumer report, both should be visible to the consumer).
- **Public Records:** If the bureau data had bankruptcy or liens/judgments (depending on if those still appear under modern rules), they are displayed correctly with dates and details.
- **No truncation or format errors:** For example, if an account has a 20-character description, it's fully shown or properly wrapped, not cut off. Numeric fields like balances or limits are accurately represented (e.g., $5,000 not mis-recorded as $500).

- If any field from the raw data is not used by the platform, that's acceptable only if it's truly not required to show; otherwise omission is a bug. (e.g., if the bureau provided an employer field but the platform doesn't display employment info, check if that's intended or a miss).

  Type: Manual (requires careful comparison with known correct data).

- **Test Case ID: ACC-02**

  Description: *Accuracy of data representation (positive test).* Ensure calculated or interpreted fields (like credit score, summaries) are correct given the raw data, and no data corruption occurs during import.

  Pre-conditions:

- The credit report data includes a credit score or the platform computes one from data (if applicable).
- The platform might also generate summary statistics (e.g., number of open accounts, total balance, etc.).

  Test Steps:
- Retrieve a credit report for a test consumer.
- Note the credit score shown (if provided by bureau or via partner).
- Verify the score against the source (if the bureau provided it, ensure it matches exactly). If the platform fetches scores separately, ensure it's the correct one for that consumer and credit pull.
- Check any summary info: e.g., "You have 5 open accounts, 2 closed accounts, oldest account 10 years old." Cross-check with the detailed data to confirm these summaries are accurate.

  Expected Results:
- The **credit score** displayed is accurate and matches the source for that report pull. No mismatch should occur (e.g., the system didn't accidentally display another person's score or an outdated score).
- Summary counts (if present) are consistent with the detailed listings. For instance, if detailed list shows 5 open accounts, the summary should not say 6. If it does, that's a bug in data aggregation.

- If the platform categorizes accounts (revolving vs installment), check a couple of accounts to ensure categorization is correct.
- No arithmetic errors in any totals (like total balance summing).

- Essentially, the platform's interpretation layer hasn't introduced any inaccuracies.
  Type: Manual (verify with visual inspection and maybe simple re-calculation).

- **Test Case ID: ACC-03**
  Description: *Handling of incomplete or unusual data (edge case).* Test how the system handles edge cases in credit data, such as missing fields or unexpected values (e.g., empty address, unusually long names, or a null account status).
  Pre-conditions:

- A credit report dataset is crafted or identified that has some anomalies, such as: no current address on file, or an account entry missing a field (some bureaus might leave blank if not reported), or a very long creditor name.
  Test Steps:
- Inject or use the anomalous credit report data for a test consumer (this might involve using a lower environment where we can edit the data feed).
- Pull that credit report through the platform.
- Observe how the data is displayed or handled.
  Expected Results:
- If certain fields are missing (e.g., no employer info or no current address), the platform should handle it gracefully – e.g., show "Not reported" or simply omit that section without error.
- If text fields are very long (like an extremely long name or address), the UI should wrap or truncate appropriately, but no crash or layout break should occur. Data should still be readable to the extent possible.
- If numeric fields are zero or null (e.g., an account with no credit limit reported), the platform might show "N/A" or 0, but should not mis-label it.
- No part of the system should fail or throw an exception due to unusual data. The presence of edge-case data should be accounted for in the design (for instance, not assuming every account has a phone number, etc.).

- If any data is out-of-bounds (say, a date in the future or an impossible value), the platform could flag it or display it as-is. It's not expected to fix it (that's up to disputes), but it should highlight if something is clearly off. However, highlighting might not be required by FCRA – the key is not introducing error.
  Type: Manual (requires custom test data and observation).

- **Test Case ID: ACC-04**
  Description: *Update of corrected information (positive test).* After a dispute leads to a correction or deletion (from DIS-02), verify that the platform now shows the updated accurate information and no remnants of the old, incorrect data remain.
  Pre-conditions:

- A dispute was resolved with a correction: for example, a wrong late payment was removed, or an account that didn't belong to the consumer was deleted.

- The credit bureau's updated data has been received by the platform (either via a new pull or an update feed).

Test Steps:

- Trigger the platform to refresh the consumer's credit report data after the dispute resolution. This could be an automatic refresh upon dispute resolution or manually pulling a fresh report.
- View the specific items that were disputed and corrected.
- Check for any trace of the old error.

Expected Results:

- The previously incorrect item is now **corrected** or removed in the consumer's credit report view. For example, if a collection record was removed, it no longer appears. If an address was corrected, only the correct address is shown.
- The data in all relevant sections is consistent with the correction. (If a tradeline was deleted, it should also disappear from any summary counts or totals.)
- There is no duplicate or stale entry lingering. Sometimes systems might accidentally keep an old copy – that should not happen.
- The consumer's report now reflects "maximum possible accuracy" post-correction, as expected after the dispute process (9) .
- The audit trail may note that an update occurred due to dispute resolution, which is fine (not directly visible to consumer, but for completeness one might check logs).

Type: Manual (need to visually verify the new report content), possibly automated by comparing data snapshots before and after the dispute.

- **Test Case ID: ACC-05**

Description: *Prevention of manual data alteration (negative test).* Ensure that platform users cannot manually edit the credit report data outside of formal channels (to preserve data integrity).

Pre-conditions:

- Credit report data for a consumer is loaded in the platform.
- There are user interfaces or admin tools for viewing this data.

Test Steps:

- As an admin or user, see if any edit functionality exists on the credit report fields (it generally should not, since data comes from the bureau).
- Try common tricks: double-click on fields to see if they become editable, attempt to find an "Edit" button, or use browser dev tools to see if any hidden form exists for these fields.
- If the platform has an admin override function (perhaps to add a note or fix a display issue), attempt to use it improperly.

Expected Results:

- The credit report data is **read-only** in the application for all users (except perhaps system processes). No direct edit or delete actions are provided on the UI for credit report lines, to ensure the data remains as reported by the CRAs.
- Any attempt to fudge it via the client side (like editing HTML) will not persist or affect actual data, as the server will ignore such changes.
- If an admin needs to correct something, it should go through the dispute mechanism or an official data reload, not manual editing. The platform should enforce that.
- Thus, the test passes if the tester finds no way to alter the data via the front end or standard APIs. If some editing feature does exist (maybe to annotate), it should be clearly separated from actual data

(like a note field, not changing the data itself).
Type: Manual (try to interact with UI elements).

Note: All the above test cases should be documented and executed with care to maintain compliance. Any failures in these tests could indicate a risk of FCRA non-compliance, which should be addressed immediately. Both manual and automated testing play important roles: automated tests ensure routine checks (like consent enforcement, role permissions, and required field validations) are continuously verified, while manual tests are crucial for content verification (notices) and complex scenario simulations. This suite provides a structured approach for QA teams to validate FCRA compliance thoroughly across the platform's features. Each test's results and evidence (screenshots, logs) should be recorded during execution for audit purposes and to demonstrate compliance.

---

① ② ③ ⑦ ⑧ ⑨ ⑩ ⑫ ⑭ Summary_of_ConsumerRights_001
https://policy.wright.edu/sites/policy.wright.edu/files/uploads/2017/Summary_of_ConsumerRights_001.pdf

④ ⑤ ⑥ ⑬ Fair Credit Reporting Act (FCRA) Compliance Checklist - Securiti
https://securiti.ai/fcra-compliance-checklist/

⑪ Disputing Errors on Your Credit Reports | Consumer Advice
https://consumer.ftc.gov/articles/disputing-errors-your-credit-reports

⑮ Compliance procedures, 15 U.S.C. § 1681e | Casetext
https://casetext.com/statute/united-states-code/title-15-commerce-and-trade/chapter-41-consumer-credit-protection/subchapter-iii-credit-reporting-agencies/section-1681e-compliance-procedures/analysis?citingPage=1&sort=relevance&sortCiting=date-ascending