**✅ MODULE: Login Page**

**🎯 MODULE PURPOSE:**

The **Login Page** is the secure entry point for all FinScoreIQPro users. It ensures authenticated access based on assigned credentials and determines user session identity, role access, and dashboard redirection.

**🔐 CURRENT FUNCTIONALITY (AS IS):**

* Simple username/email and password input fields
* "Login" button
* Displays role-based platform (but not yet personalized)
* Does **not** include:
  + “Forgot Password” link
  + “Reset Password” (post-login)
  + Session token management
  + CAPTCHA or brute-force protection
  + Sign out function (broken)

**🛠️ BUGS TO FIX:**

**1. ❌ Sign Out Button Not Working**

* **Issue**: Clicking "Sign Out" has no visible effect.
* **Expected Behavior**: Immediately logs the user out, clears session/token, and redirects to login page.
* **Fix**: Implement logout endpoint trigger → clear localStorage/sessionStorage → navigate to login page.

**2. ❌ Missing ‘Forgot Password’**

* **Issue**: No recovery flow exists for users who forgot credentials.
* **Fix**:
  + Add “Forgot Password?” below login form
  + Clicking it opens modal to:
    - Enter registered email
    - Send OTP or reset link to email
    - Verify OTP
    - Set new password
  + Use secure token with timeout (e.g., expires in 15 min)

**✨ ENHANCEMENTS:**

**A. 🔑 Add ‘Reset Password’ Post-Login**

* Inside profile menu (top right), add:
  + “Reset Password” option
  + On click, prompt old password, new password, confirm password
  + Validate match and strength (min 8 characters, 1 digit, 1 special character)

**B. 🧩 UI/UX Improvements**

* Use clear spacing between fields
* Disable login button until both fields are filled
* Show loader/spinner during login attempt
* Show error alert for invalid credentials (“Invalid username or password”)

**C. 🛡️ Security Features**

* Basic CAPTCHA (optional toggle in config)
* Brute-force lockout (after 5 invalid attempts, delay or block temporarily)
* Mask password input
* Auto-timeout of session after 30 min inactivity

**🔄 BACKEND/API NOTES:**

* **POST** /api/login  
  Request: { email, password }  
  Response: { token, userRole, userId, permissions }
* **GET** /api/logout  
  → invalidate session/token
* **POST** /api/forgot-password  
  { email } → send OTP/reset link
* **POST** /api/reset-password  
  { token, newPassword }
* Token should be JWT-based or securely stored in session.

**🧪 ACCEPTANCE CRITERIA:**

| **Feature** | **Required** | **Test Steps** |
| --- | --- | --- |
| Login with valid credentials | ✅ | Should redirect to dashboard |
| Login with invalid credentials | ✅ | Should show error message |
| Forgot Password flow works | ✅ | OTP email sent, password reset, login possible with new password |
| Reset Password post-login | ✅ | Allows changing password with validation |
| Signout button | ✅ | Logs out and returns to login page |
| Responsive layout | ✅ | Works across laptop, tablet, mobile |
| Security checks | Recommended | Brute-force lock, session timeout, CAPTCHA |

**🧠 ENGINEERING NOTES:**

| **Aspect** | **Notes** |
| --- | --- |
| Frontend Framework | React or Vue recommended (with form validation) |
| Authentication | JWT preferred with secure refresh |
| Session Handling | Expiry management, local/sessionStorage |
| Responsiveness | Mobile-friendly design using CSS grid/flex |
| UI Framework | Tailwind, Bootstrap, or custom CSS |

**✅ MODULE 2: Dashboard**

**🎯 Purpose**

To provide each user (based on role) with a **personalized, actionable view** into the system: scorecard usage, performance KPIs, and pending actions.

**🧾 Existing Problems / Gaps**

* Current dashboard is static with **placeholder tiles**
* **No role-based personalization**
* **No data charts or summaries** are pulling from backend APIs
* **Message icon** and **Signout** are non-functional (already addressed in Login)

**🔧 Bugs to Fix**

| **Bug** | **Description** | **Fix** |
| --- | --- | --- |
| 🔁 Signout button not working | User can’t log out | Use global logout() and redirect |
| 📭 Message icon unresponsive | Click event doesn’t trigger notification panel | Link to future Notification module (placeholder for now) |

**✨ Enhancements (Functionally Mandatory)**

**🔹 A. Role-Based Landing View**

System should **detect user role after login** and load a **customized dashboard** accordingly:

| **Role** | **Modules/Widgets visible** |
| --- | --- |
| Admin | All stats, config status, latest uploads, scorecard test metrics |
| Power User | Scorecard creation, test outcomes, pending changes |
| Approver | Pending approvals, efficiency matrix of scorecards |
| DSA/User | Recent bulk uploads, scoring results, personal stats |

**🔹 B. Tile-Level KPIs**

Tiles should be **dynamic, API-linked, and update with filters**:

| **Tile Title** | **Description** |
| --- | --- |
| 🧠 Total Scorecards | Active / Inactive |
| 🧪 A/B Tests Running | Active experiments in testing |
| 🥇 Best Performing Scorecard | Based on recent approval efficiency |
| 📈 Recent File Processings | Last 3 uploads (filename, scorecard, avg score) |
| ⚖️ Risk Bucket Summary | Breakdown by A/B/C/D across latest session |

➡️ Use icon + stat + brief label  
➡️ Backend endpoint: /dashboard/summary?role={userRole}

**🔹 C. Filters on Top Bar**

Enable user to refine dashboard data:

* Scorecard filter (dropdown list)
* Date range selector (default: last 7 days)
* Product type filter (if multi-product scorecards are in use)

**🔹 D. Charts / Visualizations**

Minimum required charts (use chart.js or d3.js):

| **Chart** | **Description** |
| --- | --- |
| 📊 Risk Bucket Distribution | Pie / bar chart of A/B/C/D scores (latest session or date range) |
| 📉 Approval Rate Over Time | Line chart of daily approval % vs avg score |
| 📈 Disbursal Intent Efficiency | % of high score buckets that completed journey |

**🔹 E. Quick Action Buttons (Contextual)**

If user is Power User or Admin, show:

* ➕ **Create New Scorecard**
* 📁 **View Past Tests**
* 🔄 **Sync AI Inputs** (reload AI-generated templates)

**📊 Technical Requirements**

| **Feature** | **Status** | **Detail** |
| --- | --- | --- |
| API-backed tiles | ✅ | Call /dashboard/summary + /dashboard/metrics |
| User Role-based UI | ✅ | Load widget sets conditionally |
| Chart rendering | ✅ | Use Chart.js with backend data |
| Configurable layout | Optional | Use tabs or toggle view modes |
| Responsive | ✅ | Mobile-compatible layout mandatory |

**✅ Acceptance Criteria**

* Dashboard loads different views by user type
* KPI tiles show accurate real-time data
* Charts are populated and dynamically filtered
* Signout and message icon fixed
* No frontend crashes or stale placeholders
* Admin → sees **all data**
* Power User → sees **test scores, usage**
* Approver → sees **efficiency, pending approvals**
* DSA → sees **bulk processing outcome summaries**

**✅ MODULE 3: Organisation Setup**

**🎯 Purpose**

To allow Admin or Power Users to create and manage organizations (institutions) on the platform, which will use scorecards for decisioning.

**🧩 CURRENT FUNCTIONALITY (To Be Preserved)**

| **Feature** | **Description** |
| --- | --- |
| Organisation Name | Free text input |
| Organisation Type | Dropdown (e.g., NBFC, Bank, Fintech) |
| Created By | Auto-filled |
| Created Date | Auto-filled |
| Status | Active/Inactive toggle |
| Save / Cancel | Submit or discard form |

**🪲 BUG TO FIX**

**1. Manual Org Code Input**

* **Current:** User is forced to type a code manually.
* **Expected:** System should auto-generate a unique code using naming convention.

🔧 **Fix & Logic:**

* Generate code using:
* First 4 characters of organisation name (uppercase, no special chars)
* + "-"
* Org Type (short-form e.g., NBFC, FI, BNK)
* + "-"
* Created date in YYYYMMDD
* Example:
* Organisation Name: "Kredit Finance"
* Org Type: NBFC
* Date: 2025-06-18
* ⟶ Code: "KRED-NBFC-20250618"
* **Implementation Notes:**
  + Display the code as a non-editable field once generated.
  + Regenerate if name/type is changed before save.
  + Add tool-tip: *“Auto-generated by system. Format: NAME-TYPE-DATE”*

**✨ ENHANCEMENTS REQUIRED**

**2. Optional Fields to Add**

| **Field** | **Type** | **Notes** |
| --- | --- | --- |
| Contact Person | Text | Admin-level SPOC from client |
| Contact Email | Email | Email ID for client access |
| Industry Sector | Dropdown | e.g., Lending, Insurance, Investments, Agri, Retail |
| Remarks | Textarea | Optional notes during onboarding |

All new fields should be **optional** and appear in **collapsed section: “Additional Details”**

**3. Validation & Alerts**

| **Condition** | **Behavior** |
| --- | --- |
| Org name already exists | ❌ Show error: “Organisation already exists” |
| Missing mandatory fields | ❌ Show client-side validation |
| Org Type not selected | ❌ Cannot proceed |

**4. Search & Filter (Organisation List Page)**

* Ability to view existing orgs
* Filter by:
  + Name
  + Type
  + Status
  + Created Date (From–To)

✅ Realtime table refresh with backend filtering

**🧪 ACCEPTANCE CRITERIA**

| **Area** | **Criteria** |
| --- | --- |
| Org Code Generation | Code is generated correctly, locked from editing, and updates dynamically |
| Save Form | Form saves successfully with full and partial inputs |
| Validation | Duplicate orgs prevented, mandatory fields checked |
| Optional Fields | Additional fields shown only on expand, saved correctly |
| Org List View | Able to view and filter organizations |

**🛠️ TECH NOTES FOR REPLIT**

| **Item** | **Value** |
| --- | --- |
| Code Format | ABCD-TYPE-YYYYMMDD |
| Org Types List | Fetch from constants/config (NBFC, Bank, FI, Fintech) |
| Auto-complete & Lookup | Use server-side name validation before save |
| Default Status | “Active” unless toggled off |

**✅ MODULE 4: User Management**

**🎯 Purpose:**

To enable creation, modification, role-based access control, and management of users belonging to various institutions within the FinScoreIQPro ecosystem.

**🧩 Current Functionality:**

* Add new users by institution
* Assign roles (Admin, Power User, Approver, DSA/User)
* View list of existing users

**🪲 Bugs to Fix:**

1. **Edit Button Not Working**
   * 📍 *Current Issue:* Clicking "Edit" on user list does not trigger edit form/modal.
   * ✅ *Fix:* Ensure editUser() function is properly invoked with correct user ID and that form is populated with existing data.
2. **No Role-to-Access Mapping Saved**
   * 📍 *Current Issue:* Role assignment does not activate or restrict module access dynamically.
   * ✅ *Fix:* Create access matrix by role and enforce in front-end and back-end.

**✨ Enhancements Required:**

**🔹 A. Role-Based Access Control (RBAC)**

Define role mappings and reflect them in system access behavior:

| **Role** | **Access Rights** |
| --- | --- |
| **Admin** | Full access to all modules and features |
| **Power User** | Scorecard Creation (Manual/AI), Testing, Dynamic Config |
| **Approver** | Scorecard Approval, View Efficiency Metrics, Activate/Deactivate |
| **DSA/User** | Access to Bulk Upload, Single App Evaluation, Output Download |

➡️ *Backend:* Include access flags in user table  
➡️ *Frontend:* Show/hide modules on dashboard accordingly

**🔹 B. Add Default Landing Page Option (per user)**

* Allow admin to set default module that loads on login (e.g., Dashboard, Bulk Upload)
* Store in user table: defaultModule

**🔹 C. Activate/Deactivate User**

* Add toggle button in user listing to deactivate/reactivate a user
* On deactivate:
  + Login is blocked
  + Audit trail records this action

**🔹 D. UI/UX Improvements**

* Show success/failure messages after user creation/update
* Add tooltips for role definitions
* Enable column-based sorting and filtering by:
  + Username
  + Role
  + Created Date
  + Status

**✅ Validations Required:**

| **Field** | **Rule** |
| --- | --- |
| Email | Unique, valid format |
| Username | Mandatory, 4–20 characters |
| Password | Minimum 6 characters (optional field for auto-email-based creation) |
| Role | Mandatory |
| Institution | Mandatory |
| Status (Active?) | Default: Yes |

**✅ Acceptance Criteria:**

* ✅ Edit button triggers edit form with user data populated
* ✅ Saving changes reflects updated data in user list
* ✅ Role-based access behaves correctly across modules
* ✅ Deactivated users cannot log in
* ✅ UI supports filtering, sorting, and success messages
* ✅ Audit logs track all user actions (create/edit/deactivate)

**🛠️ Replit Technical Guidelines:**

* **DB Changes:** Add accessMatrix and defaultModule fields to user schema
* **APIs to Add:**
  + PUT /users/:id/activate
  + GET /access-matrix/:role
* **UI Integration:**
  + Use dynamic menu rendering from access matrix JSON
* **Security:** Ensure backend enforces RBAC on APIs, not just frontend

**✅ Module 5: AI Scorecard Generator**

**🎯 Module Purpose**

This module enables Power Users to generate credit scorecards using AI by selecting:

* Products
* Risk Appetite
* Data Sources
* Processing Preferences
* Outcome Objectives

The AI will create a scientifically weighted, band-based scorecard with explainability, score breakdown, and risk categorization logic.

**🔧 Bugs / Issues Identified**

**1. Incorrect Weight Total ( >100%)**

* When users select multiple data sources, weights exceed 100%.
* **Fix:** Dynamically cap cumulative score to 100%. Introduce rebalancing if needed.

**2. Incomplete Variable Band View**

* Current UI shows variable scores but omits **score distribution within bands** (e.g., Age: <18 = 0, 18–30 = 5, etc.).
* **Fix:** Add expandable table for each variable to show bands and respective scores.

**3. Live Approval Rate Simulation Not Aligned**

* Approval rate simulation doesn’t match user-selected risk appetite (e.g., 65% output when 20% was expected).
* **Fix:** Align simulation engine to user input on desired approval rate. Adjust scoring thresholds or variable bands accordingly.

**4. Scorecard Output File Not Opening**

* PDF/Excel downloads are corrupted.
* **Fix:** Implement proper file writing logic. Ensure MIME type and file closure is correctly handled for both formats.

**✨ Enhancement Requirements**

**✅ A. UI & Menu Enhancements**

| **Feature** | **Instruction** |
| --- | --- |
| **Menu Renaming** | Change from “AI Generator” → “AI Scorecard Generator” |
| **Product Selection Format** | Convert dropdown to **checkbox format** to allow multi-select |

**✅ B. Structured 5-Step Flow**

**1. Data Sources + Quality Tab**

* Checkboxes for:
  + Internal App Data
  + Bureau Data (with list: CIBIL, CRIF)
  + Bank Statement Data
  + Alt Data (Telco, Device, Utility, Social)
* User must:
  + Choose sources
  + Provide data availability/quality %
* ✅ *Working Fine*

**2. Data Processing & Capability Tab**

* Allows user to:
  + Opt-in/out of bureau parsing
  + Choose derived fields (DPD, FOIR, etc.)
  + Enable/disable missing value imputation
* ✅ *Working Fine*

**3. Outcome Preferences Tab**

* Inputs:
  + Target Approval Rate (slider/input)
  + Desired Sensitivity
  + Risk Appetite (Conservative / Moderate / Aggressive)
* ✅ *Working Fine except for bug in simulation mentioned above*

**4. Scorecard Generation Page**

* Button: “Generate AI Scorecard”
* Show loading indicator + success toast
* **Before generation**, show:
  + Summary of selected inputs
  + "Confirm Inputs" dialog
* ✅ *Working fine*

**5. Post-Generation Scorecard Output UI**

**a. Scorecard Summary Tab**

* Show:
  + Total Categories
  + Weight % per category
  + Final score range (min/max)
  + Approval target, actual achieved

**b. Variables Tab**

* Category-wise layout
* Each variable with:
  + Assigned weight
  + Type (continuous / categorical)
  + Score range
  + Button → Expand to show **band-level scoring**

**c. Band-Level Scoring Tab**

* For each variable:
* Example: Variable = AGE (Score = 10)
* ┌────────────┬───────────┐
* │ Band │ Score │
* ├────────────┼───────────┤
* │ < 18 │ 0 │
* │ 18–30 │ 5 │
* │ 31–60 │ 10 │
* │ > 60 │ 2 │
* └────────────┴───────────┘

**d. Explainability + Efficiency Matrix Tab**

* Auto-generate explanation with:
  + What the scorecard prioritizes (top variables)
  + How it aligns with risk appetite
  + Accuracy, Sensitivity, and Lift metrics
  + Use business language (e.g., "Applicants with high income and low DPD are prioritized")

**e. Download Options**

* PDF & Excel with:
  + Scorecard structure
  + Variable bands
  + Explainability summary
  + Optional: JSON format (for system-to-system handoff)

**✅ C. Efficiency Matrix (New Requirement)**

| **Metric** | **Description** |
| --- | --- |
| Approval Rate Achieved | % of applicants in A + B buckets |
| Disbursal Rate | Based on test data (if available) |
| Decline Accuracy | % of D bucket who would have been non-starters |
| Lift | How well scorecard separates good/bad applicants |

**✅ D. Technical Implementation Notes**

| **Feature** | **Replit Instruction** |
| --- | --- |
| Score weight enforcement | Prevent >100%, auto-scale if needed |
| Variable Band Storage | Use nested object structure with weights |
| Auto Explainability Engine | Template-based NLP summarizer |
| Export Format | PDF and Excel via xlsx and pdfkit libraries |
| Version Control | Save each AI scorecard with metadata: version ID, timestamp, user |

**✅ Acceptance Criteria**

* AI-generated scorecard has:
  + Logical weights (summing to 100%)
  + Bands visible under each variable
  + Approval simulation matches risk appetite
  + Explainability + efficiency matrix
* Downloads open correctly
* Scorecard is versioned, savable, editable

**✅ Module 6: Scorecard Configuration**

**🎯 Purpose:**

Enable users to configure, modify, and activate/deactivate AI or manually created scorecards — including:

* Category and variable-level weights
* Band-level scoring logic for each variable
* Risk bucket mapping based on total scores
* Rules and version control

**🔧 BUGS & CORE ISSUES (Current System Status):**

| **Issue** | **Description** |
| --- | --- |
| 🛑 Module is **non-functional** or **poorly implemented** | Users can't meaningfully configure or edit scorecards |
| 🔒 No way to select & edit an existing scorecard | Configuration does not begin with selecting an existing scorecard |
| ❌ No tabs for category, variable, band or bucket setup | Entire UI lacks the granular editing layers shown in screenshots/word doc |
| ⚖ Total weight validation missing | Users can assign category or variable weights exceeding 100% |
| 🧾 No ability to activate/deactivate a scorecard | Business cannot control which scorecard is currently live |
| 📄 Word document-based structure not implemented | The required config interface from the Dynamic Scorecard Configuration Document was **completely ignored** |

**✅ FUNCTIONAL ENHANCEMENTS (TO BE IMPLEMENTED)**

**🔹 A. Scorecard Selection**

* UI dropdown to select from all **existing scorecards**
* Once selected, populate below sections dynamically with saved config values (or blank if new scorecard)

**🔹 B. Category Weight Setup Tab**

* List of **categories** (Intent, Credit Behavior, Income, Demographics, etc.)
* **Weight (%) field** for each category
* Validation:
  + Total = **exactly 100%**
  + Highlight or block save if sum ≠ 100%

**🔹 C. Variable Setup Tab (Under Each Category)**

* For each category, list assigned **variables**
* Each variable has:
  + Weight within the category (%)
  + Type: Numeric / Categorical
  + Value banding logic (if applicable)

Example:

| **Variable** | **Weight** | **Type** | **Bands (Editable)** |
| --- | --- | --- | --- |
| Age | 30% | Numeric | < 21 → 0, 21–30 → 5, 31–60 → 10, > 60 → 0 |
| Credit Score | 70% | Numeric | <600 → 0, 600–700 → 5, >700 → 10 |

* UI allows:
  + Add / remove variables under category
  + Define scoring bands with score mapping
  + Tooltips to guide user

**🔹 D. Risk Bucket Mapping Tab**

* Admin defines **score bands to buckets A, B, C, D**
* Example:

| **Score Range (%)** | **Bucket** | **Label** |
| --- | --- | --- |
| ≥ 90 | A | Auto-Approve |
| 78 – 89.99 | B | Recommend |
| 60 – 77.99 | C | Refer |
| < 60 | D | Decline |

* System must validate:
  + No overlap
  + Full range (0–100%) is covered

**🔹 E. Business Rules Engine Tab (Optional but Critical)**

* Allow user to define **conditional overrides**
* Example rules:
  + “If Credit Score < 550 and FOIR > 0.5 → Bucket D”
  + “If Age < 21 → Bucket D regardless of score”
* UI:
  + Rule creation with variable, condition, value, action (e.g., override bucket)

**🔹 F. Activation / Deactivation Logic**

* Show toggle/button: “Set this Scorecard as Active”
* Only **one active scorecard per product** allowed
* System should:
  + Automatically deactivate previous scorecard for the product
  + Maintain historical versioning and activation logs

**🔹 G. Download / Upload Configuration (Admin Only)**

* Allow download of full scorecard configuration as:
  + Excel
  + JSON
* Allow upload of config (for advanced users only)

**🔹 H. Validation & Warnings**

* Category weights must total 100%
* Variable weights must total 100% within each category
* Band scores should be within 0–100
* Risk buckets must be non-overlapping
* Show warnings for:
  + Inactive scorecard being edited
  + Weight overflows

**📦 DATA STRUCTURE (Backend Reference)**

**Example JSON Schema**

{

"scorecard\_id": "SC001",

"name": "AI\_Personal\_Loan\_Conservative",

"categories": [

{

"name": "Credit Behavior",

"weight": 40,

"variables": [

{

"name": "Credit Score",

"weight": 70,

"bands": [

{"range": "<600", "score": 0},

{"range": "600-700", "score": 5},

{"range": ">700", "score": 10}

]

},

...

]

},

...

],

"risk\_buckets": [

{"range": ">=90", "bucket": "A", "label": "Auto-Approve"},

{"range": "78–89.99", "bucket": "B", "label": "Recommend"},

...

],

"status": "active"

}

**✅ ACCEPTANCE CRITERIA**

| **Area** | **Success Condition** |
| --- | --- |
| 🎛 Tab-based UI | Tabs: Scorecard, Categories, Variables, Buckets, Rules |
| 🧮 Weight validation | Total = 100% for category & variable levels |
| 🪄 Band-level logic | Editable bands and scores per variable |
| 🧾 Bucket mapping | Score bands non-overlapping, editable |
| 📌 Rule engine | Working logic overrides |
| 🔄 Activation | Only one active per product, toggled smoothly |
| ⬇⬆ Upload/download config | Excel/JSON round-tripping |
| 🧠 Explainability | User-friendly UI to understand config logic |

**✅ MODULE 7: Testing Engine**

**🎯 Purpose:**

Allows users (typically Power Users or Admins) to test a scorecard against sample data to assess scoring logic, bucket accuracy, and business impact. Enables pre-deployment validation of any newly created or modified scorecard.

**🐞 BUGS TO FIX:**

1. **Corrupt Output File (JSON-like Excel)**
   * **Issue:** Excel file downloads in an unreadable or JSON-styled format.
   * **Fix:** Ensure result export is in .xlsx or .csv format with proper headers, row values, and encoding.
2. **Testing Never Completes (Infinite Loop)**
   * **Issue:** A/B tests show status as "Running" indefinitely, even after hours.
   * **Fix:**
     + Implement backend process monitoring: mark jobs as Failed if not complete in 10 mins.
     + Show correct end status: "Completed", "Failed", "Error".
     + Provide real-time refresh or manual refresh button.

**🛠️ FUNCTIONAL ENHANCEMENTS**

**🔹 A. Test Initiation**

* User selects **Scorecard ID** to test.
* Upload a **sample dataset** (same template as used in bulk processing).
* Add “Download Template” button once scorecard is selected.
* Validation:
  + ✅ All required fields match scorecard config
  + ❌ Prompt if mismatch or missing columns

**🔹 B. Real-time Progress Tracker**

* Show a **loading bar** or **progress tile** while file is processing.
* Display:
  + Total records
  + Processed so far
  + Estimated time remaining (if >1 min)

**🔹 C. Test Output Summary Panel**

Upon completion, display:

📊 Total Records Tested: 5,000

✅ Successful Evaluations: 4,998

❌ Errors: 2

🏷️ Score Range: 34 – 95

📈 Avg Score: 77.6

📉 Approval Rate (A+B): 42.1%

🚩 Risk Bucket Breakdown:

A: 902

B: 1,204

C: 2,113

D: 781

**🔹 D. Results Download**

* Provide clean Excel/CSV export:
  + Input fields
  + Score
  + Risk Bucket
  + Action Label (Auto-Approve, Recommend, Refer, Decline)
  + Optional toggle: Include variable-wise scores
  + Show “Reason Codes” if scoring engine supports
* Include Error Log (separate tab or file) with:
  + Row #
  + Error Type (e.g. “Missing income field”)

**🔹 E. Version & Run Management**

* Store historical test runs:
  + Scorecard tested
  + File name
  + Date/time
  + Outcome summary
  + Link to result file
* Allow users to **view or re-download past run results**

**🔹 F. Result Interpretation Help**

* Add small popup or sidebar to interpret:
  + What does Bucket A/B/C/D mean?
  + How to use these results for refinement?

**🔐 ACCESS RULES:**

* **Power User** and **Admin**: Full access
* **Approver**: Read-only access for test results
* **DSA/User**: No access

**✅ ACCEPTANCE CRITERIA**

| **Feature** | **Status** |
| --- | --- |
| Clean and readable Excel download | ✅ |
| Real-time test status (Running → Completed/Failed) | ✅ |
| Template download based on scorecard selected | ✅ |
| Outcome summary with risk split & approval rate | ✅ |
| Error logs captured and downloadable | ✅ |
| Historical run tracking enabled | ✅ |
| Interpretation helper UI exists | ✅ |
| Bug-free test execution under 10K records | ✅ |

**✅ Module 6 Addendum: Scorecard Configuration (Updated with Bucket Logic)**

&

**✅ Module 8 Addendum: Bulk File Processing (Updated with Org-Level Action Mapping)**

**✅ Addendum to Module 6: Scorecard Configuration – With Bucket Logic**

🔧 **New Functionalities to Include:**

1. **Define Default Bucket Cutoffs in Scorecard**
   * Configure via UI under each scorecard:
   * Bucket A: ≥ 90 → Action: Auto-Approve
   * Bucket B: 78–89 → Action: Recommend
   * Bucket C: 60–77 → Action: Refer
   * Bucket D: < 60 → Action: Decline
2. **Custom Action Labels at Org Level**
   * UI section: **"Bucket Action Mapping"** under Organisation Setup
   * For each scorecard used by the organisation:
     + Option to **map default buckets to custom actions**
     + Store these mappings in a separate DB table:
     + org\_id | scorecard\_id | bucket | custom\_action
     + -----------------------------------------------
     + 001 | V5.1 | B | Auto-Approve
     + 001 | V5.1 | C | Decline
3. **Optional: Custom Score Cutoffs (Admin Only)**
   * Controlled toggle: “Allow custom cutoffs?”
   * Admin can override default cutoffs if policy allows.

✅ **Acceptance Criteria:**

* Scorecard creation must enforce total weight = 100%
* Each variable must allow score-by-band assignment
* UI displays bucket logic as editable grid
* Org-level mapping stored and reflected in processing
* Exports show both bucket and mapped action

**✅ Addendum to Module 8: Bulk File Processing – Org-Specific Mapping**

🔧 **New Functionalities to Implement:**

1. **Apply Org-Specific Bucket Mapping During Processing**
   * Lookup scorecard → get default cutoffs
   * Lookup organisation → check for custom bucket-action mapping
   * Apply correct label per record during scoring
2. **Output File Columns (Expanded):**
   * Score
   * Bucket (A, B, C, D)
   * Mapped Action (Auto-Approve, Recommend, etc.)
   * Reason Code(s) (optional if explainability enabled)
3. **UI Summary to Include Effective Bucket Logic**
   * Add side card that displays:
   * Bucket → Mapped Action
   * A → Auto-Approve
   * B → Recommend
   * C → Refer
   * D → Decline

✅ **Acceptance Criteria:**

* Score result uses correct bucket action
* Bulk output reflects org-level mappings
* UI explains applied logic
* Legacy users (without mapping) default to system logic

**✅ Module 8: Bulk File Processing**

**🎯 Module Purpose**

Allow Power Users and DSA Users to upload a batch of loan applications, select the appropriate AI scorecard, and receive structured scoring outputs that classify applicants into risk buckets (A/B/C/D) along with mapped actions and optional scoring details.

**🔧 BUGS TO FIX**

1. **Incorrect Bucket Mapping in Output File**
   * ❌ Currently, both Bucket A and Bucket C are labeled as “Refer.”
   * ✅ **Fix**: Implement a standardized mapping (with option for org override):
   * A → Auto-Approve
   * B → Recommend
   * C → Refer
   * D → Decline
   * ✅ **Allow Organisation-level override** using mapping table:
   * org\_id | scorecard\_id | bucket | action\_label
2. **Output Excel/PDF File Corrupted**
   * ❌ Downloaded files do not open or contain incomplete headers.
   * ✅ **Fix**: Ensure file is properly written and closed server-side.
   * ✅ Use .xlsx or .csv (not .json disguised as Excel).
3. **“Processing Time” Shows Unrealistic Duration**
   * ❌ Example: 12471h or stuck status.
   * ✅ **Fix**:
     + Implement timeout threshold (e.g. 10 minutes max)
     + Store start\_time, end\_time, and status (Success / Failed / Timeout)
     + Retry/cancel function for long-running jobs

**🛠️ FUNCTIONAL ENHANCEMENTS**

**🔹 A. File Upload UI + Validation**

* UI must allow selection of:
  + Scorecard
  + Product(s)
* Show:
  + Template download **post-scorecard selection**
  + First 10 rows of uploaded data as preview
  + Validation before submission:
    - ✅ Required Columns Present
    - ⚠️ Extra Columns Ignored
    - ❌ Missing Columns Block Upload (with user error)

**🔹 B. Processing Logic**

* Use selected scorecard's bucket logic and score thresholds
* Lookup org-level custom bucket → action mappings (if any)
* Apply Reason Code logic if configured (e.g., why someone falls to Bucket C)
* Assign scoring at variable + band level (if “Include Variable Score” is toggled)

**🔹 C. Result Summary Dashboard (After Processing)**

Display:

🧾 Total Records Uploaded: 2,145

✅ Successfully Scored: 2,140

❌ Failed Records: 5

📊 Approval Rate: 48.1%

📉 Average Score: 78.4

🚩 High-Risk Applicants: 742 (Bucket D)

**🔹 D. Visual Charts**

Include:

* Risk Bucket Pie Chart:
* A – 234
* B – 410
* C – 954
* D – 542
* Action Label Distribution:
  + Org-specific mappings should be visible (e.g., Bucket B → Auto-Approve for OrgX)

**📁 RESULT FILE REQUIREMENTS**

**Fields to Include:**

* Application ID
* All input fields (as per template)
* **Score**
* **Bucket (A/B/C/D)**
* **Mapped Action Label (e.g., Refer, Auto-Approve)** — dynamic
* **Reason Codes** (optional, if enabled)
* **Variable-level Scores** (optional toggle)

**Output Formats:**

* Excel (.xlsx)
* Optional PDF (Summary View)
* CSV (raw dump)

**Additional Downloads:**

* **Error Log File** for skipped records
* **Template File** for each scorecard

**📂 Session History and Re-Download**

* All uploaded and processed files to be logged
* Allow:
  + View history by session ID / scorecard / date
  + Re-download results or logs

**⚙️ TECHNICAL REQUIREMENTS**

| **Feature** | **Required?** | **Remarks** |
| --- | --- | --- |
| Bucket-to-Action Mapping | ✅ | Org override available |
| Real-Time Validation of Upload | ✅ | With preview and alerts |
| Variable Score Output | Optional | Toggle UI |
| Error Log Output | ✅ | For failed/skipped rows |
| Reason Codes | Optional | Link to Scorecard Explainability |
| Session ID & Audit Trail | ✅ | UUID + Timestamps |
| Asynchronous Upload Handling | Recommended | Use job queue for files > 1000 rows |

**✅ ACCEPTANCE CRITERIA**

* Uploaded file processes without corruption
* Score is applied correctly from selected scorecard
* Bucket assignment is accurate and label matches org policy
* Summary UI is clear and accurate
* Result file is structured and openable
* Errors are logged and downloadable
* Charts match result file

**✅ Module 9: Single Application Scoring**

**🎯 Module Purpose**

This module enables **DSA Users** and **Power Users** to input and score **a single loan application in real time** using a selected AI Scorecard. The module should mimic how a live journey would be scored — returning the total score, risk bucket, action label, and optional explanation.

**🧩 CURRENT STATE & ISSUES IDENTIFIED**

* ✅ UI allows input of variables, but:
  + ❌ Input fields are not dynamically populated based on scorecard structure.
  + ❌ Final results do not show how the score was arrived at.
  + ❌ Output is limited to Score and Bucket only — **no action label, variable contribution, or reason code**.

**🔧 FIXES & FUNCTIONAL ENHANCEMENTS**

**🔹 A. Dynamic Field Auto-Generation**

✅ When a user selects a **Scorecard**, system must auto-render fields relevant to that scorecard:

* Input field type (Dropdown, Date, Numeric)
* Predefined choices (e.g., Age Group, Risk Grade)
* Mandatory vs Optional
* Tooltip with description (e.g., "Monthly Income should be in ₹")

🛠️ Source: Scorecard’s **Configuration** & **Variable Metadata**

**🔹 B. Input Validation**

* Basic checks:
  + Required fields not blank
  + Correct data format
* Immediate visual alerts (e.g., red border + message)
* Auto-convert formats (e.g., MM/YYYY → YYYY-MM)

**🔹 C. Score Execution Logic**

Upon clicking “Score Application”:

* Call scoring engine linked to selected scorecard
* Run through:
  + Preprocessing (if any)
  + Scoring function (AI or Rule-Based)
  + Band determination per variable
  + Total score calculation
  + Bucket assignment
  + Action label mapping (org-specific override)
  + Reason code (e.g., “Low Income”, “Too Many Enquiries”)

**🔹 D. Output Display Structure**

Display output clearly in sections:

| **Section** | **Data Points** |
| --- | --- |
| 📊 Score Summary | Total Score, Risk Bucket, Action Label |
| 🧩 Variable Scoring Breakdown | Variable → Value → Band → Assigned Score |
| 🧠 Reason Codes | Explanation behind assigned bucket (e.g., "Too many write-offs") |
| 📌 Scorecard Reference | Scorecard ID, Version, Config Date |
| 🔗 Action Decision | Action to be taken: "Auto Approve", "Refer", "Decline" |

**🔹 E. Optional Enhancements**

* **Button:** “Show Variable Weightage”
  + Pop-up view of how much weight each category and variable holds
* **Toggle:** “Explain Score” – Gives step-by-step logic flow
* **Save Entry:** Add to user history / audit trail
* **Export PDF:** One-click summary for download (nice-to-have)

**🔐 Access Rules by User Type**

| **User Role** | **Access to Single App Scoring** |
| --- | --- |
| Admin | ✅ Full |
| Power User | ✅ Full |
| Approver | ❌ None |
| DSA / User | ✅ Full (no config access) |

**🗂️ Audit Trail Logging**

Each scored application should log:

* Timestamp
* Scorecard used
* Inputs and final result
* Scored by (user ID)

**✅ ACCEPTANCE CRITERIA**

* Input form fields load based on scorecard structure
* Score is computed correctly as per latest config
* Bucket + Action label returned dynamically
* Reason codes are shown (if applicable)
* Visual explanation clearly rendered
* Input validations are working
* Output is downloadable (PDF optional)
* Result logged in audit trail

**✅ Module 10: Scorecard Approval Workflow**

**🎯 Module Purpose**

This module allows an authorized **Approver** to view, verify, and approve/reject changes related to scorecards created or modified by **Power Users**. It ensures no scorecard becomes "Active" without formal review, helping enforce governance, traceability, and compliance.

**🧩 CURRENT STATE & GAPS IDENTIFIED**

* 🔶 Scorecards can be created or edited but:
  + ❌ Approval workflow is either not working or not enforced.
  + ❌ Approver has no structured interface to compare **before/after** or **test results**.
  + ❌ Efficiency matrix is not visible during approval (needed for informed decision).
  + ❌ No tracking of who approved, rejected, or returned for changes.
  + ❌ No dashboard filter for “Pending Approvals”.

**🔧 FIXES & FUNCTIONAL ENHANCEMENTS**

**🔹 A. Workflow Trigger and Notification**

* Any newly created or modified scorecard goes to "Pending Approval" state.
* Notification should be generated:
  + For Approver(s) when a scorecard is submitted
  + For Power User when action is taken by Approver

**🔹 B. Approver Landing Screen**

List view of all scorecards **pending approval**:

| **Scorecard** | **Type** | **Created/Modified By** | **Date** | **Status** | **Action** |
| --- | --- | --- | --- | --- | --- |
| AI\_PersonalLoan\_V4.0 | AI | user123 | 2025-06-18 | Pending | View & Act |

**🔹 C. Scorecard Detail Comparison View**

Side-by-side comparison layout:

| **Section** | **Old Version (if any)** | **New Version** |
| --- | --- | --- |
| Weights | 20/30/50 | 25/25/50 |
| Variables | 12 | 14 |
| Score Bands | — | Defined |
| Active | Yes | No (pending) |

**🔹 D. Efficiency & Effectiveness Matrix View**

For the new scorecard, show **testing results summary**:

| **Metric** | **Value** |
| --- | --- |
| Total Records Tested | 10,000 |
| Approval Rate | 38.5% |
| Average Score | 76.3 |
| Bucket A Accuracy | 91.2% |
| D Rejection Precision | 97.5% |
| A/B Leakage | 3.2% |

📌 **NOTE:** Pull this from Testing Engine results linked to the scorecard ID.

**🔹 E. Approval Actions Available**

Approver can:

* ✅ **Approve** – scorecard becomes **Active**
* 🔁 **Send Back** – for improvements with remarks
* ❌ **Reject** – archived, cannot be used
* 🛑 **Hold** – keeps it in review without approval/rejection

**🔹 F. Mandatory Fields for Approver Action**

| **Field** | **Required** |
| --- | --- |
| Comments | ✅ |
| Action Taken (Approve, Reject, etc.) | ✅ |
| Viewed test results | ✅ checkbox before submitting |

**🔹 G. Post-Action Behavior**

* On **Approval**:
  + Old active scorecard becomes **Inactive**
  + New version becomes **Active**
  + Auto-log entry in Audit Trail
* On **Reject or Send Back**:
  + Notify original Power User with comments
  + Scorecard returns to “Draft” mode for editing

**🔐 Access Rules by User Type**

| **User Role** | **Access to Approval Workflow** |
| --- | --- |
| Admin | ✅ Full override |
| Power User | ❌ View only (own drafts) |
| Approver | ✅ Full |
| DSA / User | ❌ None |

**🗂️ Audit Trail Requirements**

For each approval:

* Scorecard ID and Version
* Action taken
* By whom
* Comments
* Timestamp

**✅ ACCEPTANCE CRITERIA**

* Approver sees pending scorecards
* Can compare old/new versions
* Efficiency matrix visible before approval
* System enforces approval before activation
* Audit trail entries generated correctly
* Notifications work for both Power User and Approver
* Approved scorecard becomes active, others are archived

**✅ Module 11: A/B Testing**

**🎯 Module Purpose**

This module enables **Power Users** and **Approvers** to compare two or more scorecards in a live or simulated environment to determine which performs better in terms of disbursal prediction, bucket precision, and overall effectiveness.

A/B testing supports scientific tuning and helps choose the optimal model to activate.

**🧩 CURRENT STATE & GAPS IDENTIFIED**

* A/B Testing UI exists but:
  + ❌ "Running" tests don’t return any output, no status update.
  + ❌ Results section lacks clarity – no comparative insights shown.
  + ❌ No linkage to test dataset or prior test results.
  + ❌ No configurable control for selecting sample size or matching variables.
  + ❌ No logs to trace who initiated the A/B test and what versions were tested.

**🔧 BUG FIXES**

**🔹 A. Stuck Status Bug**

* Problem: A/B Tests show as “Running” indefinitely.
* Fix:
  + Implement timeout logic – auto-fail after 15 mins if no backend response.
  + Add backend check every 60s to update test status (Success/Fail).

**🔹 B. Result Generation Failure**

* Problem: Test completes in backend but front-end never displays outcome.
* Fix:
  + Ensure result JSON returned is correctly parsed and rendered in UI.
  + Catch backend null/empty response and show user-friendly error.

**✨ FUNCTIONAL ENHANCEMENTS**

**🔹 A. A/B Test Setup Screen**

| **Field** | **Input Type** | **Description** |
| --- | --- | --- |
| Test Name | Text | Name the test |
| Select Scorecard A | Dropdown (Active) | Required |
| Select Scorecard B | Dropdown (Draft/Active) | Required |
| Test Dataset Upload | File Upload (.csv) | Mandatory |
| Sample Size Filter | Numeric (optional) | Optional row cap (e.g., 1,000) |
| Matching Criteria | Checkbox list | Ensure sample is balanced (age, income, etc.) |

✅ Must restrict duplicate combinations (e.g., A vs A).

**🔹 B. A/B Test Results Dashboard**

After test runs successfully, results should show side-by-side:

**📊 Performance Comparison**

| **Metric** | **Scorecard A** | **Scorecard B** |
| --- | --- | --- |
| Approval Rate (%) | 42.7 | 39.8 |
| Average Score | 77.1 | 74.5 |
| Bucket A Accuracy (%) | 91.2 | 86.4 |
| Bucket D Precision (%) | 97.5 | 94.0 |
| A/B Leakage (%) | 2.1 | 5.3 |
| High Risk % | 18.9 | 23.1 |

**📈 Charts**

* Risk Bucket Distribution (A/B/C/D)
* Score Distribution Histogram
* Disbursal Lift Chart

✅ Each test must be saved and timestamped with metadata (user, cards tested, test ID)

**🔹 C. Action Post-Test**

User can:

* ✅ Mark Better Scorecard for Activation (if authorized)
* ✅ Download Detailed Results CSV
* ✅ Archive Test
* ❌ Cannot re-run same test ID (must clone)

**🔹 D. CSV Result File Fields**

| **Field** | **Type** | **Description** |
| --- | --- | --- |
| Application ID | String | Unique ID from dataset |
| Scorecard A Score | Numeric | Score from model A |
| Scorecard B Score | Numeric | Score from model B |
| Bucket A | A/B/C/D | From Scorecard A |
| Bucket B | A/B/C/D | From Scorecard B |
| Recommendation | Text | Which model performed better |

Optional:

* Include band-level reason code mapping.

**🔒 User Role Controls**

| **Role** | **Access** |
| --- | --- |
| Admin | All A/B tests, result override |
| Power User | Can initiate and review A/B tests |
| Approver | View results, make scorecard decision |
| DSA/User | ❌ No access |

**✅ ACCEPTANCE CRITERIA**

* A/B Tests execute and complete with defined timeout
* UI displays results cleanly and accurately
* CSV download works and reflects test comparison
* Performance metrics help inform decision-making
* Each test stored with full traceability
* Permission logic enforced correctly

**🧩 Module 12: Documentation (Final Engineering Instructions for Replit)**

**🎯 Purpose**

To generate **dynamic documentation** for scorecards, A/B tests, testing outcomes, and configuration logic in an easily exportable and human-readable format. This acts as an **auto-generated credit decision audit trail** and guide for internal users (Power User / Approver).

**🧱 Core Functional Requirements**

**✅ A. Trigger Conditions**

Documentation should be **auto-generated** under the following conditions:

| **Event** | **Output** |
| --- | --- |
| New scorecard created via AI or manual | Scorecard Documentation (versioned) |
| Scorecard config updated | Configuration Log Document |
| A/B test completed | A/B Test Outcome Document |
| Testing engine results finalized | Test Summary Document |
| Scorecard approved or deactivated | Audit document for approval/deactivation |

**✅ B. Output Format**

* All documentation must be generated in **Markdown (.md)** format
* Export options: .pdf, .docx, .md (via Pandoc or equivalent)

**✅ C. Document Types and Structure**

| **Doc Type** | **Structure (Sections)** |
| --- | --- |
| Scorecard Document | 1. Header (Name, Version, Created By, Date)2. Category Weights3. Variables under each category4. Score Band Logic5. Risk Bucket Mapping6. Comments / Notes |
| Scorecard Configuration Log | 1. Scorecard Ref2. Variable Edits3. Weight Adjustments4. Timestamped Changes |
| A/B Test Result | 1. Test Name2. Date & Duration3. Scorecards Compared4. Outcome Summary (Approval Rate, Avg Score, Bucket Efficiency)5. Recommendation |
| Testing Engine Outcome | 1. Scorecard Used2. Records Tested3. Outcome Summary4. Risk Bucket Distribution5. Insights |
| Approval Workflow Log | 1. Scorecard Details2. Change Summary3. Approved By / Date4. Approval Comments |

**✅ D. File Naming Convention**

scorecard\_<NAME>\_v<VERSION>.md

abtest\_<NAME>\_<DATE>.md

testing\_<scorecard>\_<timestamp>.md

configlog\_<scorecard>\_<timestamp>.md

approval\_<scorecard>\_<status>.md

**✅ E. Download & Access**

* All documents should be downloadable from the **Documentation Module**
* Optional: Show filters (Scorecard Name, Doc Type, Date Range)
* Optional: Inline preview of Markdown in browser

**🔧 Technical Implementation Suggestions**

| **Area** | **Recommendation** |
| --- | --- |
| Markdown rendering | Use markdown-it (JS) or showdown.js to render inline |
| Export to PDF/Word | Use Pandoc (server-side) or JS libraries like html-docx-js or pdfmake |
| Storage | Save .md files and generated docs to a /docs folder mapped by scorecard or session |

**✅ Acceptance Criteria**

* Scorecard, A/B, Test, Config & Approval documents are generated automatically upon event triggers
* Each document follows defined structure
* Documents are versioned and downloadable
* All core user actions that influence model logic or decisions are auditable

**✅ Sample Markdown Template for “Scorecard Document”**

# 📄 Scorecard Document

\*\*Scorecard Name:\*\* AI\_Personal\_Loan

\*\*Version:\*\* v1.3

\*\*Created By:\*\* system\_AI\_v6

\*\*Created On:\*\* 2025-06-18

---

## 🔹 Category Weights

| Category | Weight (%) |

|-------------------|------------|

| Credit Profile | 40 |

| Income Stability | 25 |

| Intent Signals | 15 |

| Banking Behavior | 10 |

| Demographics | 10 |

| \*\*Total\*\* | \*\*100\*\* |

---

## 🔹 Variable Scoring (Sample)

### Category: Credit Profile

| Variable | Max Score | Description |

|----------------------|-----------|------------------------------------------|

| Credit Score | 15 | Based on CIBIL or Experian score |

| DPD\_0\_Last12M | 10 | No overdue accounts in past 12 months |

| Enquiry Count ≤ 2 | 5 | Low recent enquiry frequency |

| FOIR < 0.4 | 10 | Healthy repayment capacity |

---

## 🔹 Band-Level Logic: Credit Score

| Credit Score Range | Score |

|--------------------|-------|

| 300 – 649 | 0 |

| 650 – 699 | 5 |

| 700 – 749 | 10 |

| 750+ | 15 |

---

## 🔹 Risk Bucket Mapping

| Score Range | Risk Bucket | Label |

|-------------|-------------|---------------|

| ≥ 90 | A | Auto-Approve |

| 78 – 89.99 | B | Recommend |

| 60 – 77.99 | C | Refer |

| < 60 | D | Decline |

---

## 🔹 Bucket Assignment Policy

- Bucket Logic at Scorecard Level: Enabled

- Bucket Adjustment at Org Level: Not Enabled

- Smart Rescue Rule: Applied (2 of 5 override conditions)

---

## 📝 Notes

This scorecard is optimized for conservative disbursement and tuned to minimize false positives in Bucket A. Created via AI Scorecard Generator v6.1 and edited manually for band realignment.

**✅ Module 13: Audit Trail**

**🎯 Purpose**

To ensure traceability and accountability across every critical action taken by any user in the system. This includes changes to scorecards, configurations, user management, test outcomes, approvals, file uploads, and documentation exports.

**🔍 Current Status (as per user feedback):**

* **✅ Fully functional**
* **No bugs or gaps identified**
* **Exported audit log (CSV) is opening correctly**
* Format is readable and entries are timestamped

**✅ Required Functionality Summary**

| **Feature** | **Description** |
| --- | --- |
| **Comprehensive Event Capture** | Log all significant events including: login, logout, scorecard creation/edit, configuration updates, bulk uploads, A/B test creation, approvals, and documentation exports. |
| **User Identification** | Each log entry must include username, role, and organization ID (if applicable). |
| **Timestamp Accuracy** | All timestamps should be in ISO format with timezone support (e.g. 2025-06-18T12:34:56+05:30). |
| **Event Classification** | Allow grouping by event type: configuration change, user action, upload, approval, test, error. |
| **Audit Log Download** | Export entire filtered log (as CSV) by: date range, user, module, or event type. |
| **Pagination or Lazy Load** | If >1000 records, load on scroll or use pagination to avoid UI lag. |
| **Search & Filter** | By date, module name, action type, username. |

**🔐 Security and Governance Expectations**

| **Requirement** | **Detail** |
| --- | --- |
| **Tamper-proof** | Stored in append-only table or with hash chain for immutability |
| **Role-based Access** | Only Admins, Approvers should access full audit log |
| **Session Linking** | Tie actions to session ID for forensic analysis |
| **Data Retention** | Minimum 365 days; optional archive/export after that |

**✅ Acceptance Criteria**

* All critical actions across modules are being captured.
* Filters (user/date/module/type) are working as expected.
* Exported CSV matches UI filters and shows clean data.
* Search is performant (≤ 1s for 10,000 logs).
* Export filename should include date and filter scope.

**🧪 Sample Entry (Expected in CSV / Table)**

| **Timestamp** | **User** | **Role** | **Action** | **Module** | **Details** |
| --- | --- | --- | --- | --- | --- |
| 2025-06-18 12:14 | poweruser1 | PowerUser | Scorecard Created | AI Scorecard Generator | Scorecard: AI\_PL\_Conservative\_v3.1 |

**✅ MODULE 13: API Integration & Management**

**🎯 Purpose:**

Enable real-time and batch consumption of scoring services by lenders, partners, or internal systems via secure, well-documented APIs.

**🔑 Core Capabilities:**

| **Functionality** | **Description** |
| --- | --- |
| Real-time scoring API | Accepts a single loan application payload and returns a credit score, risk bucket, and reason codes. |
| Batch scoring API | Accepts a CSV file (base64 or S3 path), processes asynchronously, and provides download link once complete. |
| Scorecard metadata fetch | Endpoint to retrieve active scorecard details, version, fields, and weight config. |
| Auth & access control | Token-based authentication with role-based API access mapped to organization or user. |
| API dashboard (logs) | UI section showing last N API hits, latency, status (200/400/500), and errors. |
| Versioning | Each scorecard or scoring API is version-controlled to ensure backward compatibility. |
| Fallback / default scoring | If required fields are missing, allow system to score using default fallback rules. |

**📥 Real-Time Scoring API Specification**

**POST** /api/v1/score

**🔸 Headers:**

Authorization: Bearer <token>

Content-Type: application/json

X-Org-Code: ABC-FI-20250601

**🔸 Payload (Sample):**

{

"scorecardId": "SC-AI-PL-001",

"applicationData": {

"age": 32,

"monthlyIncome": 45000,

"creditScore": 760,

"dpd": 0,

"foir": 0.28,

"enquiryCount": 1,

"employmentType": "Salaried",

"loanAmount": 150000

}

}

**🔸 Response:**

{

"score": 87.5,

"bucket": "A",

"decision": "Auto-Approve",

"reasonCodes": ["Low FOIR", "High Credit Score", "Clean DPD History"]

}

**📂 Batch Scoring API Specification**

**POST** /api/v1/score/bulk

**🔸 Headers:**

Same as above

**🔸 Payload:**

{

"scorecardId": "SC-AI-PL-001",

"file": "<base64\_encoded\_csv>"

}

**🔸 Response:**

{

"sessionId": "UUID-202506181035",

"status": "Submitted",

"estimatedTime": "1–2 mins",

"downloadUrl": null

}

🔄 **Follow-up GET** /api/v1/score/bulk/status/{sessionId}  
Response contains:

* Success count
* Errors count
* Downloadable file URL

**🛡️ Security & Access Control**

* **Token Management:**
  + Each user/org gets unique API token (JWT or equivalent)
  + Tokens expire & are refreshable
* **Rate Limiting:**
  + 50 reqs/min per org for real-time
  + 5 concurrent bulk uploads per org
* **Access Roles:**
  + Admin: full
  + Power User: test API only
  + DSA: none (UI only)

**🚨 Error Handling**

| **HTTP Code** | **Scenario** | **Response Body Example** |
| --- | --- | --- |
| 400 | Invalid input or missing fields | { "error": "Field 'age' is missing." } |
| 401 | Unauthorized / token expired | { "error": "Token invalid or expired." } |
| 500 | Internal error or scoring failed | { "error": "Unable to generate score." } |

Retry logic for bulk: 2 attempts allowed. Logs available in API dashboard.

**📊 API Dashboard (UI Module)**

* API Usage Summary (Today / Weekly / Monthly)
* Per endpoint call count and latency
* Error type distribution
* Logs: Downloadable JSON/CSV of last 100 hits
* Searchable by:
  + Session ID
  + Org Code
  + Status

**🔃 Versioning Policy**

* Each API tied to specific scorecard version.
* /api/v1/score → uses active version unless version is specified.
* Deprecated versions still callable for 60 days.

**✅ Acceptance Criteria**

* ✅ API keys generated and managed via Admin interface
* ✅ Real-time scoring API gives accurate, reasoned output
* ✅ Batch scoring supports large files, returns correct download link
* ✅ All API traffic is logged and available for admin/audit
* ✅ Secure, token-authenticated access
* ✅ Documentation endpoint /api/docs auto-generated using OpenAPI/Swagger