**Project Documentation — SA ID Number Checker**

**Salesforce Login  
Username:** [**wk-cloudsmiths@hotmail.com**](mailto:wk-cloudsmiths@hotmail.com) **Password: WKmaurice30  
Type: Developer  
  
Github URL :** [**https://github.com/wasimkhadaroo3093/cloudsmithsWK.git**](https://github.com/wasimkhadaroo3093/cloudsmithsWK.git)

**Digital Experience SF:** [**https://cloudsmiths2-dev-ed.develop.my.site.com/southafricaidchecker/**](https://cloudsmiths2-dev-ed.develop.my.site.com/southafricaidchecker/)

**1. Overview**

This project implements an end-to-end Salesforce solution for validating South African ID numbers and retrieving public/banking holidays via the Calendarific API.

User Stories implemented:

1. A web page (Experience Cloud Site) with an input field to enter a South African ID number.
2. Validate the ID both client-side (LWC) and server-side (Apex).
3. Save or update a record (SAIdRecord\_\_c) for each unique ID, tracking how many times it was searched.
4. Fetch banking/public holidays for the year of birth in the ID number.
5. Display results back to the visitor with a styled UI using South African flag colors.

**App Launcher**

**App Name: SA ID Checker -- (Blue)**

Contains: Lightning App “SA ID Checker” and object “SA ID”  
  
**App Name: SA ID Checker -- (Yellow)**

Open directly to Digital Experience.

[**https://cloudsmiths2-dev-ed.develop.my.site.com/southafricaidchecker/**](https://cloudsmiths2-dev-ed.develop.my.site.com/southafricaidchecker/)

**2. Architecture**

The solution follows a layered Apex architecture pattern. Each layer has a distinct responsibility:

**UT - Utility Layer**

Purpose: Encapsulates common algorithms and helper functions.

* In this project: UT001\_IdNumberUtil validates and decodes SA ID numbers.
* Rules: 13 digits, valid date, gender, citizenship digit, Luhn checksum.
* Returns a strongly typed DecodedId object.

**DM - Data Manager Layer**

Purpose: Handles all direct interaction with Salesforce data (SOQL/DML).

In this project: DM001\_SAIdRecord queries and upserts SAIdRecord\_\_c.

* Applies FLS & CRUD enforcement.

**EM - Entity Manager Layer**

Purpose: Shapes decoded or external data into SObject records.

In this project: EM001\_SAIdRecord maps decoded ID fields into SAIdRecord\_\_c.

* Keeps transformation logic separate from persistence.

**SM - Service Manager Layer**

Purpose: Orchestrates end-to-end business processes.

In this project: SM001\_IdCheckerService coordinates decode, save, and holiday retrieval.

* decodeAndSave(idNumber) -> Validate & upsert record.
* fetchZAHolidaysByYear(year) -> Call WS layer for holidays.

Keeps DML and callouts separated (avoids uncommitted work errors).

**WS - Web Service Layer**

Purpose: Encapsulates integrations with external APIs.

In this project: WS001\_CalendarificService makes REST GET calls to Calendarific.

* Uses API key.
* Parses JSON and filters public/banking holidays.
* Returns a list of WRP\_Types.Holiday.

**LTN - Lightning Controller Layer**

Purpose: Exposes @AuraEnabled methods to LWC.

In this project: LTN001\_IdNumberChecker bridges LWC <-> SM.

* Methods: decodeAndSave(String id), fetchHolidays(Integer year).

**3. LWC - Front End**

Component: csmIdNumberChecker

* Input + Button: center-aligned, fixed width, disables until 13 digits entered.
* Error Messages: friendly red banners if ID invalid.
* Info Panel: Green background, yellow text, red border (SA flag colors).
* Holidays Sidebar: right side, blue background, holidays displayed in yellow rows with scroll.
* Experience Cloud ready: exposed via lightningCommunity\_\_Page and lightningCommunity\_\_Default.

**4. Data Model**

Custom Object: SAIdRecord\_\_c

Label: SA ID

* IdNumber\_\_c (Text 13, Unique, External ID).
* DateOfBirth\_\_c (Date).
* Gender\_\_c (Picklist).
* IsCitizen\_\_c (Checkbox).
* SearchCount\_\_c (Number).

This ensures persistence of search history and user tracking.

**5. Unit Tests**

Comprehensive unit tests implemented for each layer:

**UT001\_IdNumberUtil\_TEST – Coverage 90%**

* Valid + invalid ID formats.
* Luhn checksum validation.

**DM001\_SAIdRecord\_TEST – Coverage 100%**

* Upsert + query correctness.

**WS001\_CalendarificService\_TEST – Coverage 94%**

* Mocked API response parsing.

**SM001\_IdCheckerService\_TEST – Coverage 84%**

* Decode & save flow.
* Holiday fetch flow.

**LTN001\_IdNumberChecker\_TEST – Coverage 100%**

* LWC entrypoints (decode+save + fetch holidays).

**6. Test Data (IDs Used)**

**Valid IDs**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID Number** | **Birthdate** | **Gender** | **Citizen** |
| 8001015009087 | 1980-01-01 | Male | Yes |
| 9205050008081 | 1992-05-05 | Female | Yes |
| 7503125801197 | 1975-03-12 | Male | No |
| 8607290054083 | 1986-07-29 | Female | Yes |
| 9902155801192 | 1999-02-15 | Male | No |
| 7009180023081 | 1970-09-18 | Female | Yes |
| 8412036789088 | 1984-12-03 | Male | Yes |
| 0306250123198 | 2003-06-25 | Female | No |

**Invalid Date (Month/Day)**

|  |  |  |
| --- | --- | --- |
| **ID Number** | **Reason it fails** | **Expected Error** |
| 9913325009087 | Month = 13, Day = 32 | Invalid birth date |
| 8002315009087 | February 31, 1980 (no such day) | Invalid birth date |
| 7504315009087 | April 31, 1975 (no such day) | Invalid birth date |
| 0102295009087 | Feb 29, 2001 (2001 not leap year) | Invalid birth date |

**Invalid Citizenship Digit**

|  |  |  |
| --- | --- | --- |
| **ID Number** | **Reason it fails** | **Expected Error** |
| 8001015009787 | 11th digit = 7 | Invalid citizenship digit |
| 9205051234987 | 11th digit = 3 | Invalid citizenship digit |

**Invalid Luhn (Checksum)**

|  |  |  |
| --- | --- | --- |
| **ID Number** | **Reason it fails** | **Expected Error** |
| 8001015009086 | Same as a valid one but checksum off | The ID number is not valid. Please check and try again. |
| 9205050008082 | Tweaked last digit | The ID number is not valid. Please check and try again. |

**7. Key Benefits**

* Clean Architecture: clear UT/DM/EM/SM/WS separation.
* Reusability: logic centralized in Service Manager; LWC is thin.
* Security: CRUD/FLS enforced in DM layer; Guest User configured properly.
* Scalable: easily extended for more external services or additional ID checks.
* Testable: robust UT suite ensures correctness.

**8. Conclusion**

This project shows:

* Solid Salesforce best practices (layered code, testing, separation of concerns).
* UI & UX aligned to User Stories with aesthetic SA flag theme.
* Strong test coverage and error handling.