**Software Design Plan**

|  |  |
| --- | --- |
| **WGU Student ID** | 011205284 |

# A. Business Case

## 1. Problem Statement

The issue pertains to the Endothon Finance web application, which generates loan profiles with inaccurate financial data. Rather than retrieving data from the five most recent fiscal years, the application incorrectly requests financial data from the business’s initial years of operation. This discrepancy in functionality leads to erroneous loan profiles, resulting in inaccurate information sent to lending partners. The associated ticket (#D480-AEN1) highlights this bug and calls for an update to the application’s logic to ensure it correctly retrieves the latest financial data, as originally intended.

## 2. Business requirements

**Accurate Financial Data Retrieval:**

* **Requirement:** The web application must retrieve financial data from the most recent five completed fiscal years for businesses that have been operational for over five years. This ensures that the loan profiles reflect the most up-to-date financial data, which is crucial for accurate financial assessments.
* **Failure:** The current web app retrieves data from the first five fiscal years of the business's operation instead of the most recent five. For example, if a company was established in 2000, the app incorrectly requests data from 2000, 2001, 2002, 2003, and 2004 instead of the correct years of 2018, 2019, 2020, 2021, and 2022. This results in loan profiles that do not reflect the business's current financial state, potentially leading to incorrect loan decisions.

**Adaptability for Newer Businesses:**

* **Requirement:** For businesses with less than five years of operation, the application must be adaptable enough to request available financial data for the completed years and generate forecasted data for the remaining future years to complete the five-year financial profile.
* **Failure:** The web application needs to be designed to handle this scenario appropriately, making it difficult for businesses that have been operational for less than five years. This lack of adaptability leads to complete or accurate loan profiles, complicating the loan application process for startups and newer businesses.

## 3. In-scope action items

Revise the logic for retrieving financial data to ensure the web application accurately pulls financial data from the most recent five fiscal years for businesses older than five years. This adjustment ensures compliance with the requirement for precise financial data retrieval for loan qualification.

Update the logic for businesses younger than five years to request forecasted financial data, ensuring the application adapts to the specific needs of newer companies. This change aligns with the business requirement for flexibility in economic data collection.

Validate data integrity across all business profiles to ensure the updated logic functions correctly for businesses of varying ages, ensuring loan profiles remain accurate, regardless of the company’s age.

To confirm proper behavior, incorporate testing for edge cases, particularly scenarios where a business is precisely five years old, to confirm adequate behavior. This ensures robust handling of all potential data collection scenarios.

## 4. Out-of-scope action items

**Redesign of the user interface:**

**Alignment:** While the user interface influences user experience, the issue pertains specifically to backend logic, not the user interface.

**Out of Scope:** The problem is with data retrieval, and user interface redesign is optional to resolve this bug.

**Expansion of loan types offered:**

**Alignment**: The business logic currently pertains only to financial data collection, and expanding loan types would be unrelated to this issue.

**Out of Scope:** The ticket focuses on resolving a logic issue and expanding loan offerings, which would fall under feature development rather than bug fixing.

# B. Requirements

## 1. Functional requirements

**Retrieve Financial Data for Businesses Older than Five Years:**

* The web app must retrieve financial data from the most recent five completed fiscal years for businesses that have been operational for over five years. This will ensure that the loan profiles reflect the most current financial status of the business, which aligns with the business requirement for accurate loan assessments.

**Request Forecasted Data for Businesses Less than Five Years Old:**

* The web app must handle businesses with less than five years of operation by retrieving all available historical data and generating forecasts for the remaining future years. This ensures that loan profiles for newer businesses are complete and provide a full five-year financial projection, as the business logic requires.

**Handle Edge Cases for Businesses Exactly Five Years Old:**

* The web app must ensure that businesses that are exactly five years old are handled appropriately by retrieving financial data for the last five fiscal years without requiring forecast data. This particular case must be accounted for to avoid errors in loan profile creation.

**Validate Business Establishment Year Input:**

* The web app must validate the input for the business establishment year before performing any data retrieval. This validation ensures that the correct logic path is followed based on the age of the business, preventing incorrect data retrieval and ensuring the accuracy of the loan profile.

## 2. Non-functional requirements

**Performance:** The system should handle data retrieval within a maximum of 2 seconds, even under peak load conditions.

**Security:** The financial data must be encrypted at rest and in transit to ensure compliance with data protection standards.

# C. Software Design

## 1. Software behavior

**Business established five years ago:**

* **Intended Response:** Retrieve the five most recent completed fiscal years.
* **Constraints:** The logic must disregard the current year and adjust based on the current date.

**Business established within the last five years:**

* **Intended Response**: Retrieve available financial data and request a forecast for the remaining years.
* **Constraints:** Logic must adjust dynamically based on business age and calculate future fiscal years accurately.

**Invalid establishment year input:**

* **Intended Response:** Provide an error message if the establishment year is invalid or missing.
* **Constraints:** The app should validate the input against the current year to prevent errors.

**Exactly five years old:**

* **Intended Response:** Retrieve data for the previous five years without forecast data.
* **Constraints:** Ensure the logic correctly requests forecasted years for exactly five-year-old businesses.

## 2. Software structure

**Segmentation of Development:** The web app’s logic will be refactored into modular components:

* **Data input validation module:** Handles the validation of the establishment year before querying financial data.
* **Data retrieval module:** Queries the financial database for the correct set of fiscal years based on the business age.
* **Forecasting module:** Handles the logic for businesses younger than five years, dynamically generating future financial year requests.
* **Error handling and logging module:** Captures and logs any errors during the input validation or data retrieval, ensuring robustness.

# D. Development Approach

## 1. Planned deliverables

**Financial data retrieval logic update:** This module will fix the current logic to retrieve the most recent fiscal years correctly.

* **Steps:** Refactor the existing logic, test with different business establishment years, and implement edge-case handling.

**Forecasting Functionality:** Module to handle projections for businesses younger than five years.

* **Steps:** Implement dynamic logic to request projected financial data and validate against future years.

**Test cases and validation scripts:** Scripts to validate that the logic behaves correctly across various scenarios.

**Steps:** Create comprehensive unit and integration tests, including edge cases.

**Documentation Update:** Document the new logic and its expected behavior.

**Steps:** Write updated technical documentation and instructions for maintaining the new logic.

## 2. Sequence of deliverables

**Step 1:** Update the financial data retrieval logic for older businesses.

**Justification:** This is the core functionality needed to address the issue in the ticket.

**Step 2:** Implement the forecasting module for newer businesses.

**Justification:** Handles the secondary functionality for businesses younger than five years.

**Step 3:** Develop test cases and validation scripts to ensure the new logic works.

**Justification:** Testing must follow after core functionality to verify correctness.

**Step 4:** Update documentation once the logic is finalized.

**Justification:** Documentation should be the final deliverable after successful testing.

## 3. Development environment

**Programming Language (Python):** Handles the backend logic for querying and processing data.

* **Purpose:** Ensure compatibility with the existing web app infrastructure.

**Database:** Stores and retrieves the financial data.

* **Purpose:** Ensures fast and reliable access to business financial data.

**Integrated Development Environment (IDE):** This is for the development and debugging of logic.

* **Purpose:** Streamlines the development process and debugging of the new logic

## 4. Development Methodology

**Agile Methodology:** The project will follow an agile methodology focusing on iterative development, with frequent testing and reviews.

* **Informed the planning process**: Agile allows for rapid iteration, addressing core issues first, then adding complexity.
* **Why agile over waterfall:** Agile allows quicker adjustments if unforeseen issues arise, especially in bug fixes where requirements may evolve based on early testing feedback.