### BPP Business School Coursework Cover Sheet

Please use this document as the cover sheet for the 1st page of your assessment.

Please complete the below table – the grey columns

|  |  |
| --- | --- |
| Module Name | **Programming for Data Analysts** |
| Student Reference Number  (SRN) | **BP0274581** |
| Assessment Title | **A Programming Solution Report for ZFS** |

Please complete the yellow sections in the below declaration:

**BP0274581**

Declaration of Original Work:

I hereby declare that I have read and understood BPP’s regulations on plagiarism and that this is my original work, researched, undertaken, completed and submitted in accordance with the requirements of BPP School of Business and Technology.

The word count, excluding contents table, bibliography and appendices, is **2500** words.

Student Reference Number:

Date: **31-10-2023**

By submitting this coursework you agree to all rules and regulations of BPP regarding assessments and awards for programs.

Please note that by submitting this assessment you are declaring that you are fit to sit this assessment.

BPP University reserves the right to use all submitted work for educational purposes and may request that work be published for a wider audience.

**A Programming Solution Report for ZFS**

**Table of Contents**

1. Introduction 5
   1. Purpose Statement 5
   2. Main Objectives 5
   3. Business Case 5
      1. Challenges Faced by ZFS 5
      2. Benefits of Proposed Solution 6
         1. Improves Efficiency 6
         2. Risk Mitigation 6
         3. Scalability 6
         4. Cost Reduction 6
         5. Better Customer Experience 6
         6. Audit Support 6
         7. Data-Driven Decisions 6
2. Approach 7
   1. Software Development Life Cycle 7
   2. Importance of SDLC 7
      1. Structured Approach 7
      2. Quality Assurance 7
      3. Requirements Management 7
      4. Risk Management 7
      5. Project Management 7
      6. Predictability 8
      7. Scalability 8
      8. Documentation 8
      9. Change Management 8
      10. Compliance 8
      11. Long-term Maintenance 8
   3. SDLC for this Project 8
      1. Requirements 8
         1. Non-functional Requirements 8
            1. Why Google Collab? 9
            2. Why Python? 9
         2. Functional Requirements 9
      2. Design 9
         1. Flowchart 10
         2. Pseudocode 11
      3. Implementation 11
         1. Libraries 11
            1. Pandas 12
            2. Tabula 12
            3. Matplotlib.pyplot 15
            4. Pygwalker 16
         2. Task 1: Loan Data Automation 18
            1. Data Files Uploading 18
            2. Replacing Numeric Values with Labels 19
            3. Combining data sets 20
         3. Task2 – Descriptive Analysis 22
            1. Data Indexing 22
            2. Data Cleaning 22
            3. Exploratory Data Analysis 29
            4. Visualization 29
      4. Verification 33
         1. Unit testing 33
         2. Black box testing 34
         3. Functional testing 34
      5. Maintenance 34
3. Recommendations 34
4. Conclusion 35
5. Appendix 1 – Code 36
6. References 37
7. **Introduction**

Zappy Financial Services (ZFS) operates to provide loans to clients who run small businesses. A high volume of loan applications has been observed by the company in the last few months.

* 1. **Purpose Statement**

The purpose of this report is to discuss the functioning/benefits of an automated programming solution for “Zappy Financial Services” that allows expediting as well as streamlining the loan application process. Hence, replacing the manual effort with an automated programming solution is the main purpose.

* 1. **Main Objectives**

The objectives include discussing the following aspects of programming solutions to expedite the loan application processing time:

* Python code that can help upload PDF and Excel data files
* Extraction of data
* Combining datasets into a single Data frame
* Cleaning the data
* Exploratory Data Analysis
* Visualization for analyzing trends in customer data
* Benefits of the current designed approach for ZFS
* Future suggestions
  1. **Business Case**
     1. **Challenges Faced by ZFS**

The financial conditions of many small companies are not stable nowadays (Building the AI Bank of the Future, 2021). Therefore, a large number of customers are applying at ZFS for loans that are causing the following challenges:

* The volume of applications at ZFS has increased causing a backlog in application processing time
* Increases frustration among customers
* High demand for experienced data analysts at ZFS
* The delay in loan applications decreases the market value of ZFS
* Market competition
  + 1. **Benefits of Proposed Solution**
       1. **Improves Efficiency**

The automated solution will expedite the loan application process at ZFS as a large number of data files containing huge records can be uploaded/processed quickly.

* + - 1. **Risk Mitigation**

By using data-driven decision-making and an Artificial Intelligence model, risk management can be handled by ZFS.

* + - 1. **Scalability**

This report presents a generic solution to handle a large number of loan data files in different formats.

* + - 1. **Cost Reduction**

Manual data cleansing demands experienced resources at ZFS. However, this programming solution will save expenses of resources.

* + - 1. **Better Customer Experience**

Due to the fast approvals of loan applications, the loan team will have an opportunity to deal with customers in person or one-on-one. This will provide customers with a better experience.

* + - 1. **Audit Support**

This solution will render easy monitoring and reporting for the compliance team, stakeholders, and audits.

* + - 1. **Data-Driven Decisions**

With the help of this solution, loan teams will become able to make enhanced data-driven decisions for the approvals of loans.

1. **Approach**
   1. **Software Development Life Cycle**

It is a distinct and systematic approach that allows the software product to pass through different phases as below (JiravaÚstav, 2004):

* Requirements
* Design
* Implementation
* Verification
* Maintenance
  1. **Importance of SDLC**
     1. **Structured Approach**

SDLC allows the development of a complex software product by breaking the complex process into manageable stages that make it easier to track the progress of the project (Opstal., 2010).

* + 1. **Quality Assurance**

A high-quality software product can be delivered by using different software testing and quality control methodologies (Hossain, 2023).

* + 1. **Requirements Management**

The analysis of the project becomes easier and the needs of the customer can be understood at the start to mitigate the risk of costly changes in later stages (Shankar et al., 2023).

* + 1. **Risk Management**

The delays in the project can be avoided if the risk evaluation of the project is done in the early stages (Memon et al., 2018).

* + 1. **Project Management**

SDLC allows following a framework for planning, keeping track and resource allocation (Obeidat, 2013):

* + 1. **Predictability**

The development process of the software becomes easier with SDLC and it is paramount to meet the deadlines (Hossain, 2023).

* + 1. **Scalability**

SDLC helps to adjust the framework according to the requirements of the project. For instance, the size of the project decides which framework to be adopted (Wenzi et al., 2022).

* + 1. **Documentation**

SDLC helps maintain the following (Shankar et al., 2023):

* Software
* Knowledge transfer
* Compliance with regulatory requirements
  + 1. **Change Management**

SDLC allows teams to evaluate the impact of modifications in the software product and make informed decisions accordingly (Taherdoost, 2023).

* + 1. **Compliance**

This approach allows us to analyze whether the software product meets the standards of compliance (Memon et al., 2018).

* + 1. **Long-term Maintenance**

SDLC involves the maintenance stage where software product is analyzed to be up to date throughout the life cycle (Obeidat, 2013).

* 1. **SDLC for this Project**
     1. **Requirements**

The first stage allows the collection of the requirements by reviewing the problem statement. There are two types of requirements as below (White, 2010):

* + - 1. **Non-functional Requirements**

The platform suggested to be used for this project is Google Collab. Besides, the suggested language is Python.

* + - * 1. **Why Google Collab?**

Google Collab is recommended as it (Pessoa et al., 2018):

* Allows a fully configured environment
* Free of charge
* Works in the same way on different hardware systems
* Easy to detect errors while running the code
* No dependency on the hardware resources
  + - * 1. **Why Python?**

The Python language is being used in this project as it has many benefits (Nitnaware, 2019):

* Easily readable
* User-friendly and simple language
* Open source/free
* Compatible with many platforms
* Supports a large number of libraries
  + - 1. **Functional Requirements**
* Design a solution to upload Excel/PDF files
* Extract data from the Excel/PDF files
* Combine the Data frames acquired from PDF and Excel datasets into a data frame
* Cleaning the Data frame
* Exploratory data analysis
* Visualization through graphs
  + 1. **Design**

The programming or coding is being done in this phase (Memon et al., 2018).

* + - 1. **Flowchart**

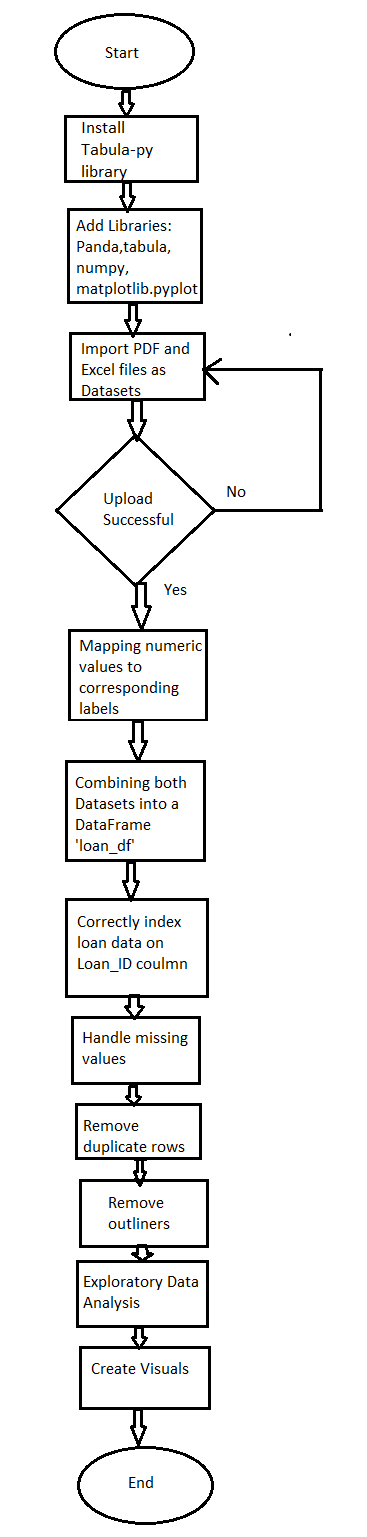


Figure 1.

* + - 1. **Pseudocode**

1. Start
2. Install libraries
3. Import libraries
4. Import PDF Data File as a data set
5. If the upload is successful, extract the Data set from tables on all pages of a PDF file.

If not, import the PDF file again

1. Upload Excel Data File
2. If the Upload is successful, extract the Data set from an Excel file.
3. If not, import the Excel file again.
4. Combine both datasets into a data frame
5. Ensure that loan data is correctly indexed on the Loan\_ID column
6. Handle missing values
7. Drop the duplicate values
8. Remove outliers
9. Exploratory data analysis to calculate average and percentage values
10. Create Visuals of Exploratory data analysis results
11. End
    * 1. **Implementation**

In the implementation phase, coding is being started by the developer. The tasks in this phase are divided into units/modules (Memon et al., 2018).

* + - 1. **Libraries**

Firstly, the installation of tabula and pygwalker is done on Google Collab notebook by using the pip command as in Figure 2:

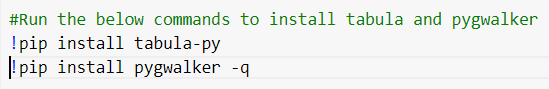


Figure 2.



Figure 3.

Afterwards, the libraries are imported as in Figure 3. The following are the purposes of the libraries used for the ZFS programming solution:

* + - * 1. **Pandas**

This library helps perform analysis of data in Python by manipulating the data. Pandas render data structures/functions for data analysis. The main purposes for using Pandas in this project are (Nitnaware, 2019):

* Data Cleaning
* Data Transformation
* Data Analysis by using various mathematical functions
* Data input/output
* Data visualization
  + - * 1. **Tabula**

This open-source Java library is used to extract the data from PDF files. Especially, the extraction of tables from PDF files is performed by this library. The key purposes are as follows:

* Table/data extraction from PDF files
* Data processing of extracted data (Cleaning/manipulation)
* Data analysis of extracted tables
* Integrating with other data analysis libraries (i.e. Matplotlib)
* Report Generation

For instance, tabula helps to extract all pages of a PDF file in the program as shown in Figure 4.

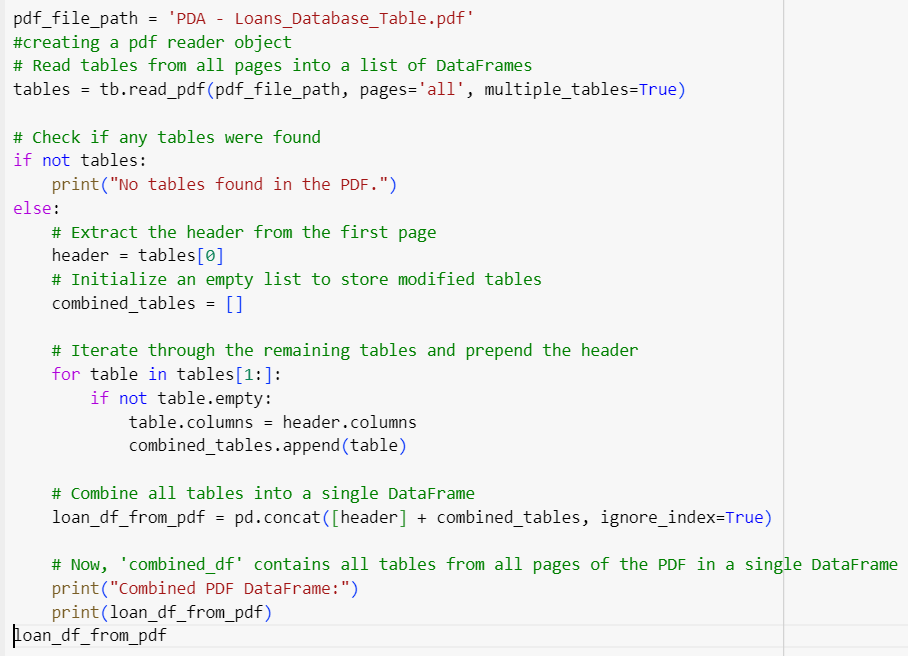


Figure 4.

The output of this segment of the program would be the extracted loan table from the PDF file as shown in Figure 5 and Figure 6.

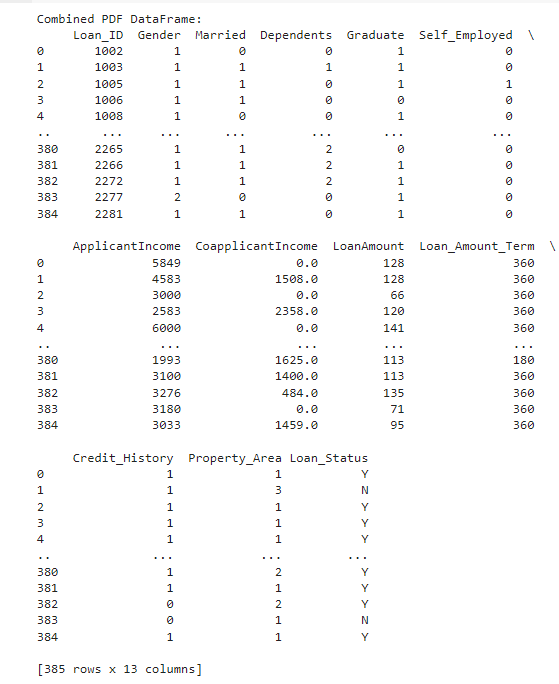


Figure 5.

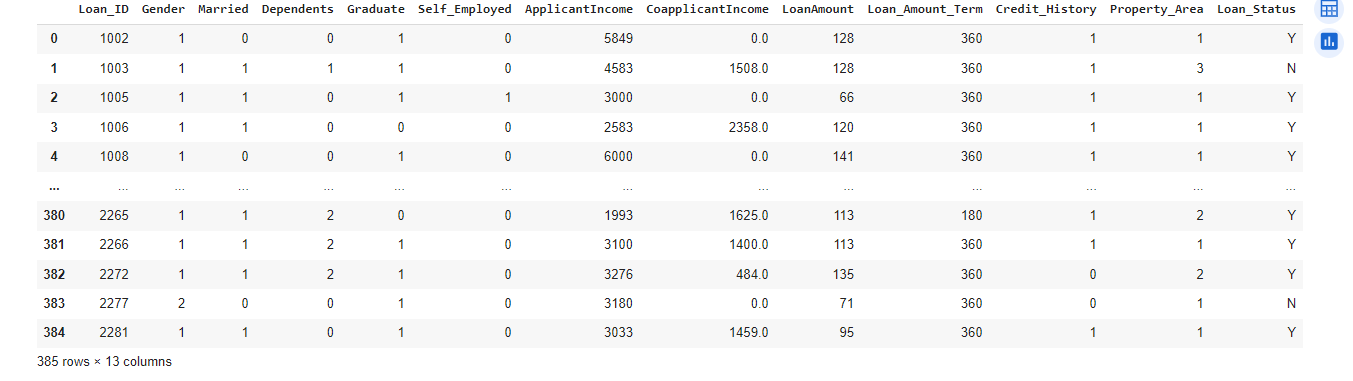


Figure 6.

* + - * 1. **Matplotlib.pyplot**

This Python library is used to make visualizations such as below:

* Graphs
* Plots
* Charts

In the current project, ‘matplotlib.pyplot’ has been used for the following purposes:

* **Data Visualization:** Various types of exploratory data analysis pie charts have been created in this report to visualize the results of analysis by using this library. For instance, Figure 7 shows the pie chart:

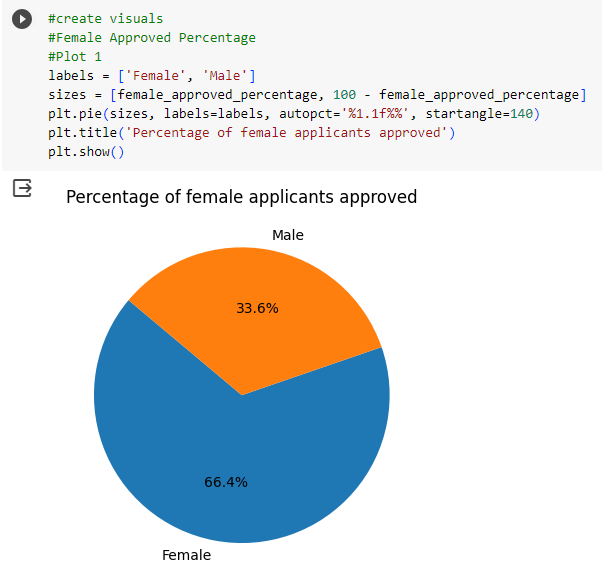


Figure 7.

* **Customization:** By using the ‘matplotlib.pyplot’ library, customized plots/charts have been drawn in different colors by using the desired labels as shown in Figure 8 and Figure 10.

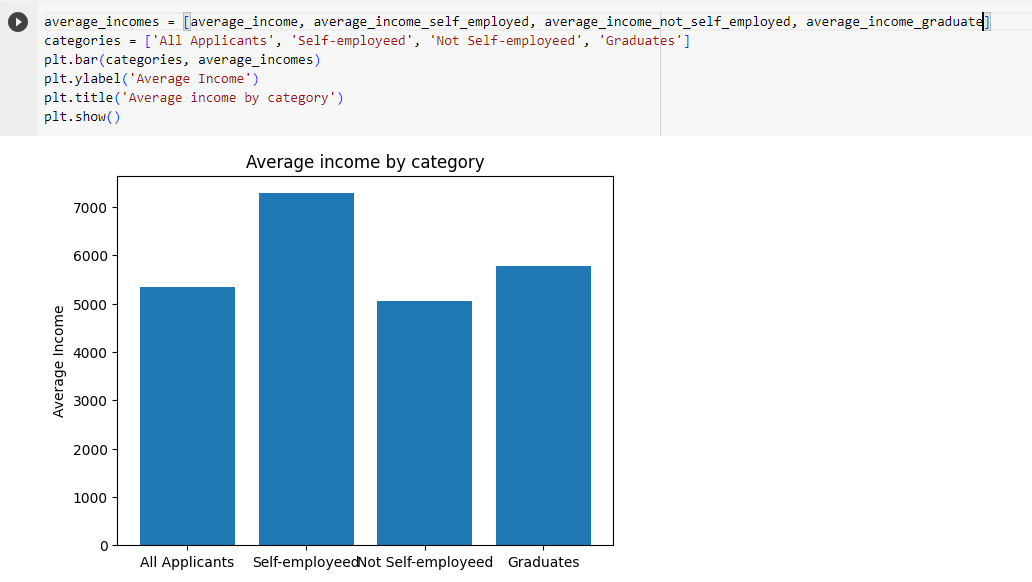


Figure 8.

* + - * 1. **Pygwalker**

This library allows one to view the raw data in a Data frame. In the current report, the data frame extracted from Excel/PDF files can be visualized by using pygwalker functions.

**Input:**

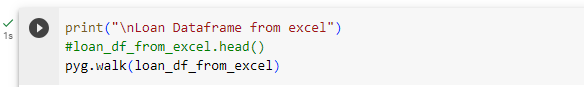


Figure 9.

**Output:**



Figure 10.

For instance, Figure 10 shows the bar graph of the Excel data set. This plot between Men and women (married/unmarried) has been drawn in different colors to review their loan status, credit history and loan amount term.

Similarly, the scatter chart of the PDF data set in Figure 11 shows the trends between loan status and credit history of married and unmarried customers according to their loan terms.

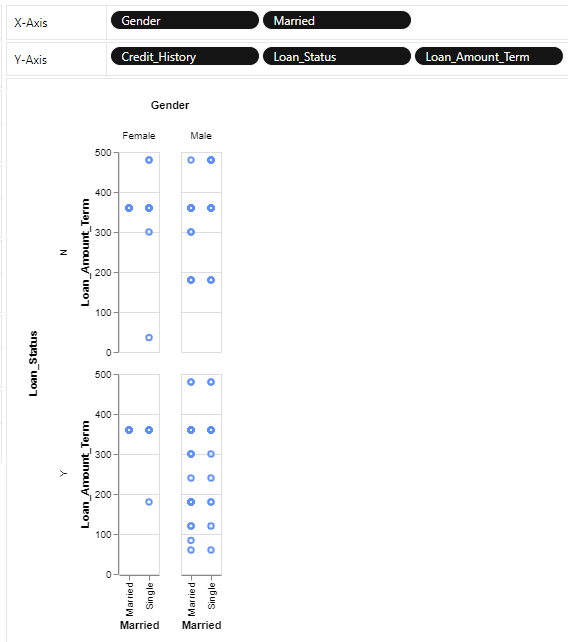


Figure 11.

* + - 1. **Task 1: Loan Data Automation**
         1. **Data File Uploading**

Firstly, the read\_pdf function has been used in the project to extract data from all the pages of the PDF file as shown in Figure 4. Secondly, to extract data sets from an Excel file, the read\_excel function has been used as shown in Figure 12.

**Input:**

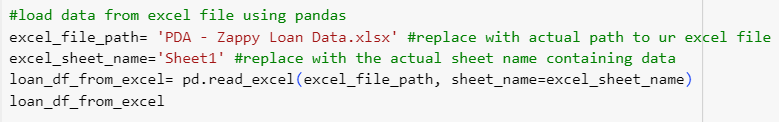


Figure 12.

**Output:**

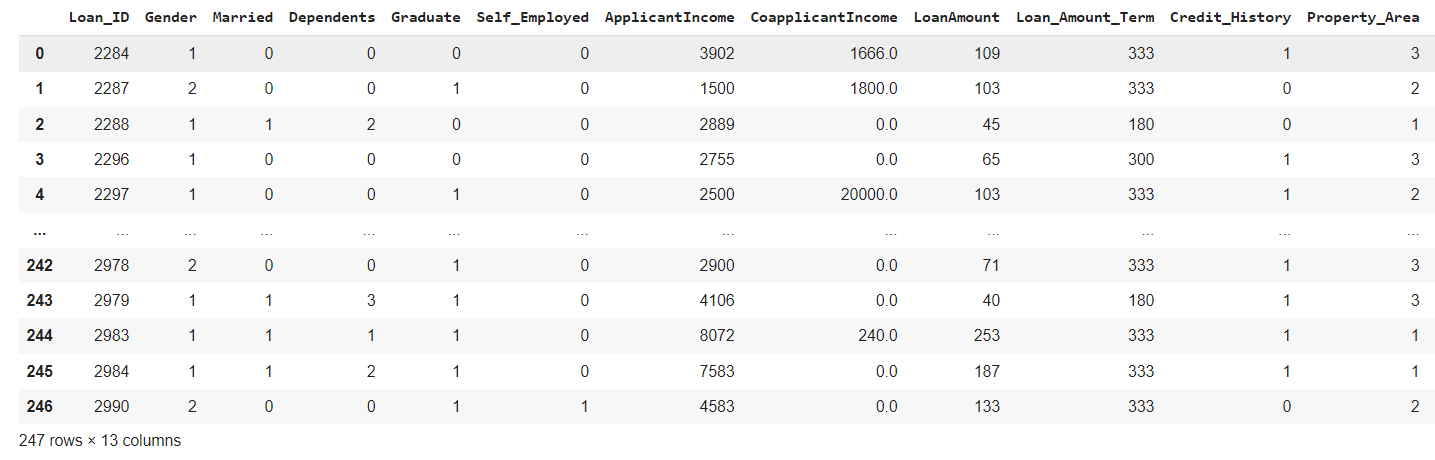


Figure 13.

* + - * 1. **Replacing Numeric Values with Labels**

Currently, there are several values in Excel and PDF files. Such as, the gender of customers is denoted by 0 or 1. The program intends to replace the numeric values with the corresponding labels as in Figure 14.

**Input:**

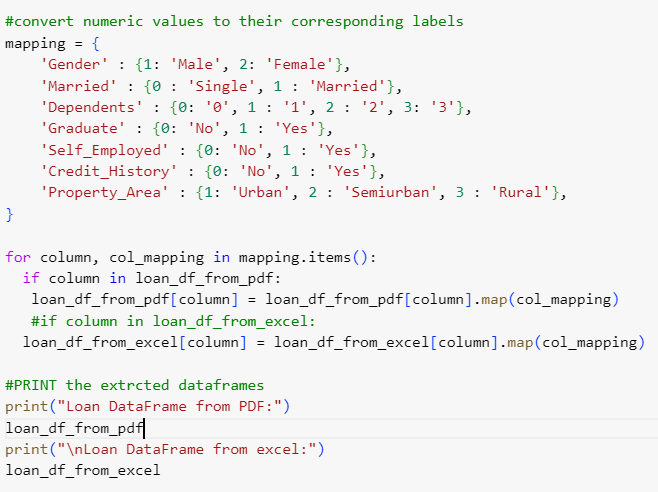


Figure 14.

The extracted pdf data has been shown in Figure 15 after replacing with the labels.

**Output:**

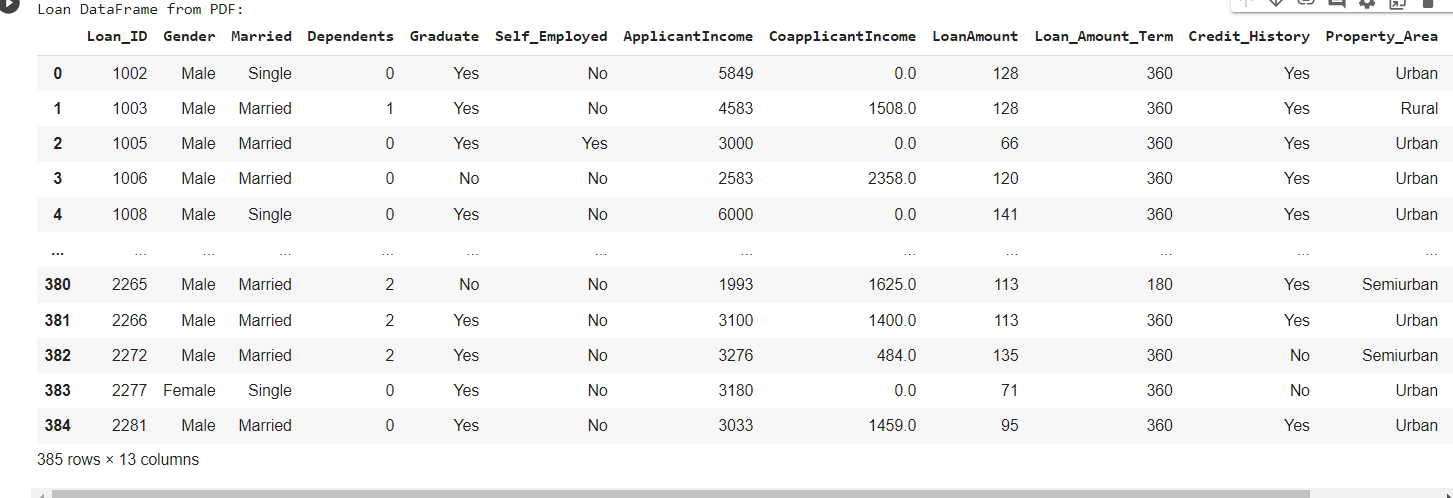


Figure 15.

Similarly, the Excel data frame after replacing numbers with labels is shown in Figure 16.

**Output:**

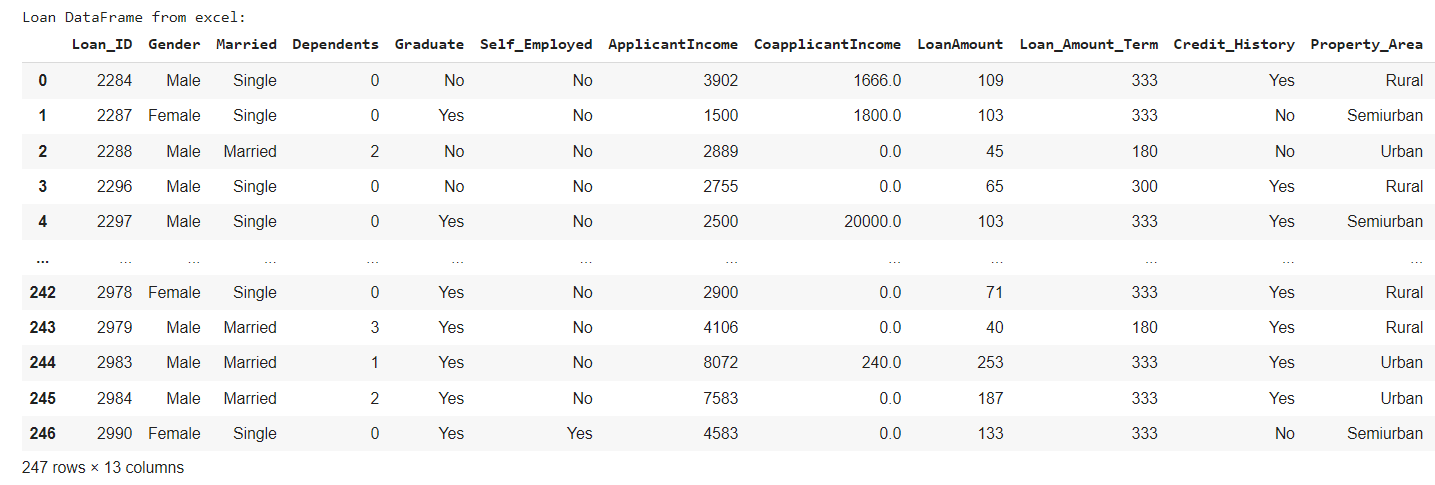


Figure 16.

* + - * 1. **Combining data sets**

Both pdf and Excel datasets have been combined by using the ‘concat’ function in the programming project as shown in Figure 17.

**Input:**

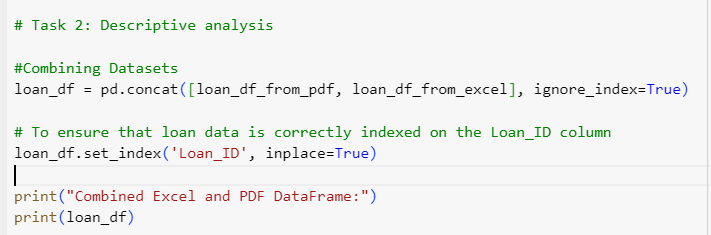


Figure 17.

**Output:**

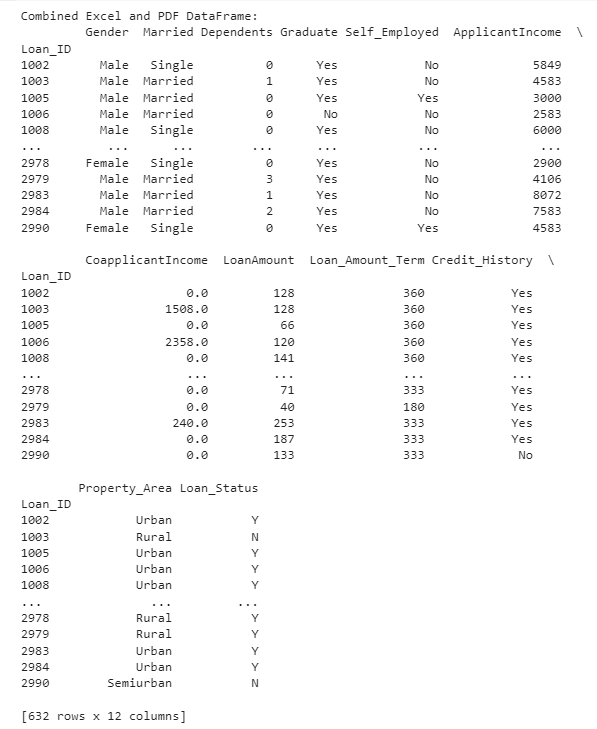


Figure 18.

* + - 1. **Task 2 – Descriptive Analysis**

Once the extracted data from two different data files (excel and PDF) has been combined, the descriptive analysis can be performed on the data frame.

* + - * 1. **Data Indexing**

The set\_index function has been used in the program to verify if loan data is properly indexed on the Loan\_ID field.

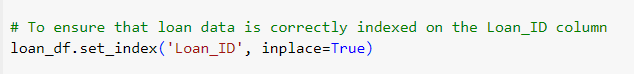


Figure 19.

* + - * 1. **Data Cleaning**
* **Duplicate Values:** Duplicate rows in the data frame have been identified by using the duplicated() function as shown in Figure 20.

**Input:**

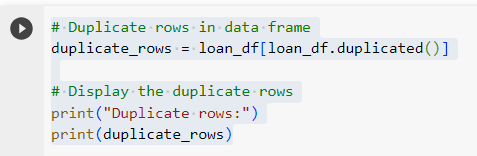


Figure 20.

**Output:**

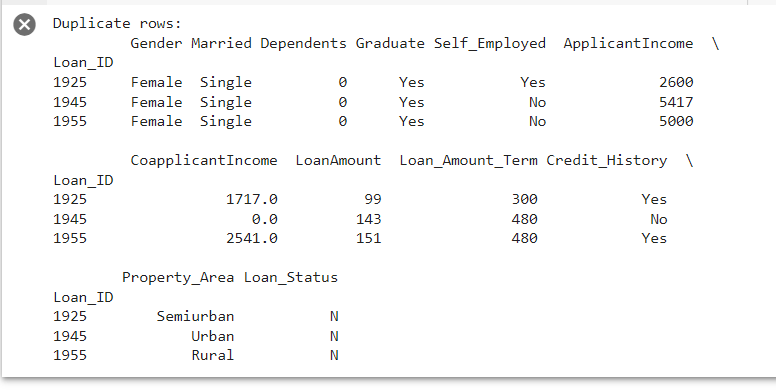


Figure 21.

To remove the duplicate rows, different approaches have been used in the program to analyze the results as in Figure 22.

**Input:**



Figure 22.

**Output:**

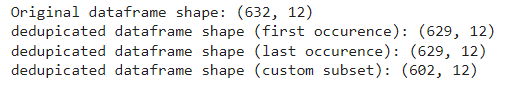


Figure 23.

* **Missing Values:** These can be handled by dropping as below:

**Input:**



Figure 24.

**Output:**

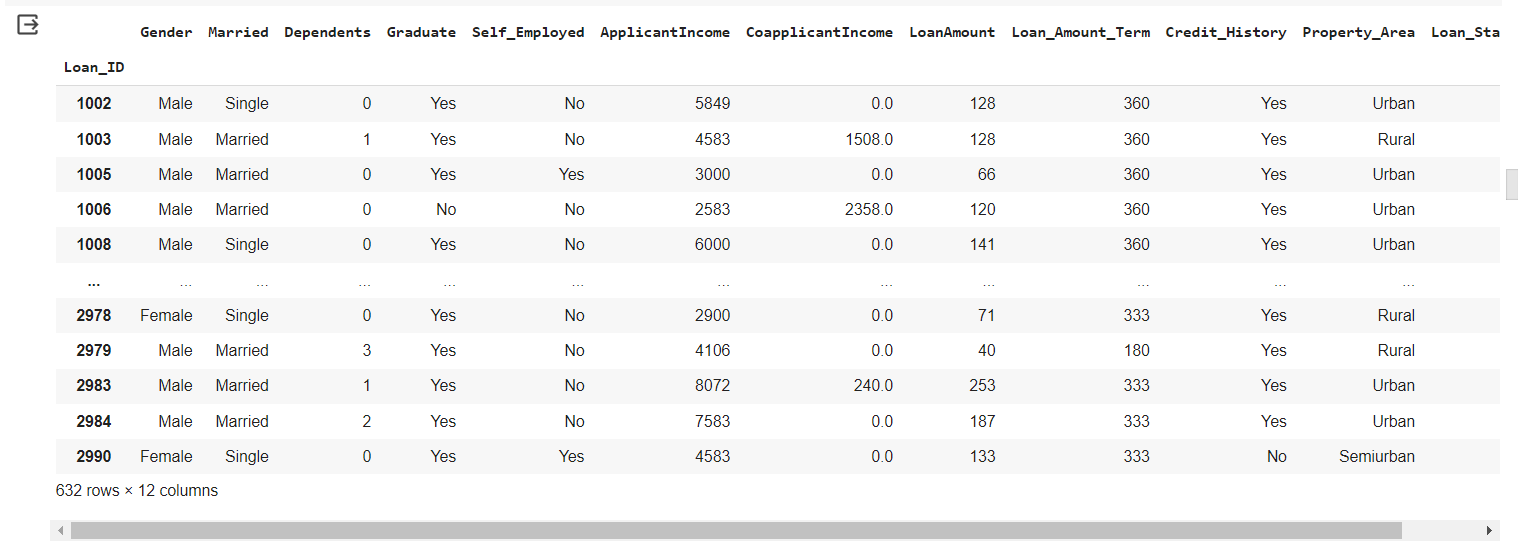


Figure 25.

* **Outliers:** Thedata points that are too far from the usual data points of the loan data frame have been identified by analyzing loan\_df\_deduplicated. Firstly, the stats of this data frame describe the minimum and maximum values of different fields.

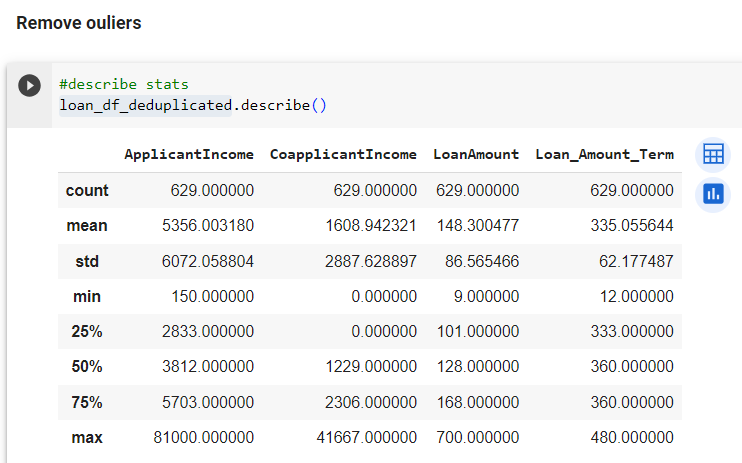


Figure 26.

Figure 27 and 28 shows the outliers for the ‘ApplicantIncome’ and ‘CoApplicantIncome’ fields.

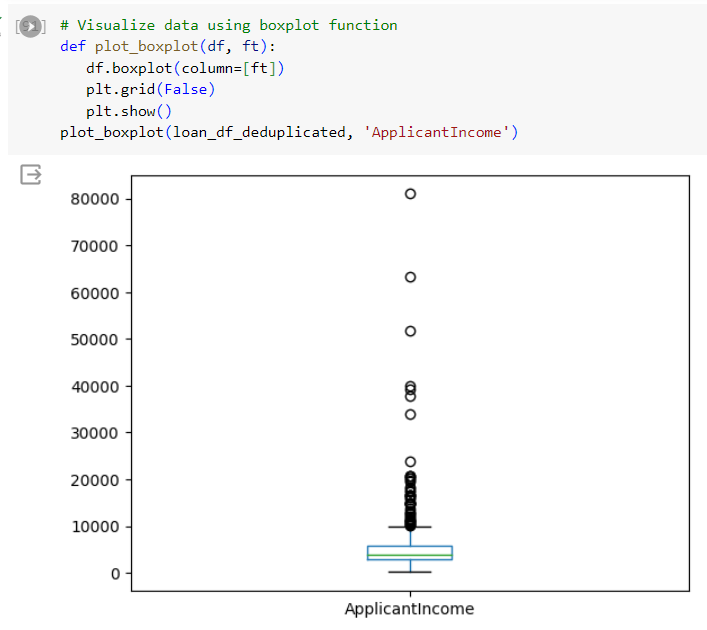


Figure 27.

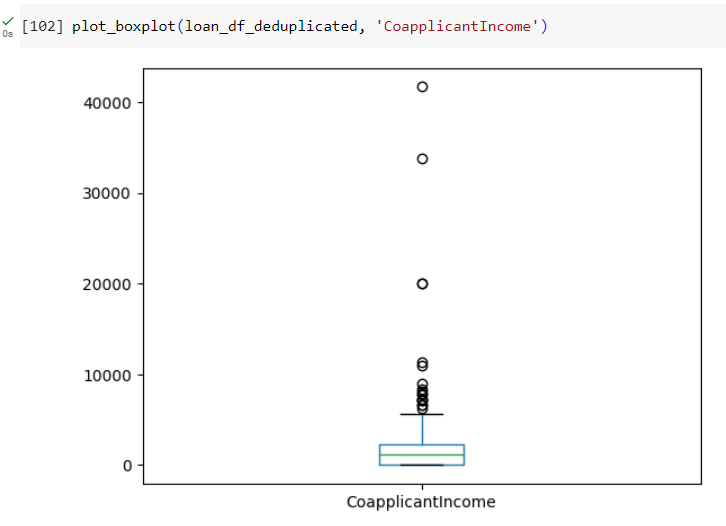


Figure 28.

A remove outliers function has been made as shown in Figure 29:

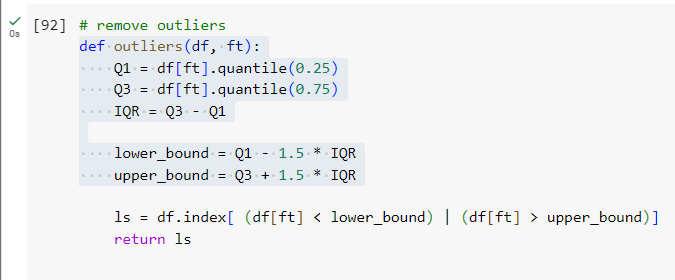


Figure 29.

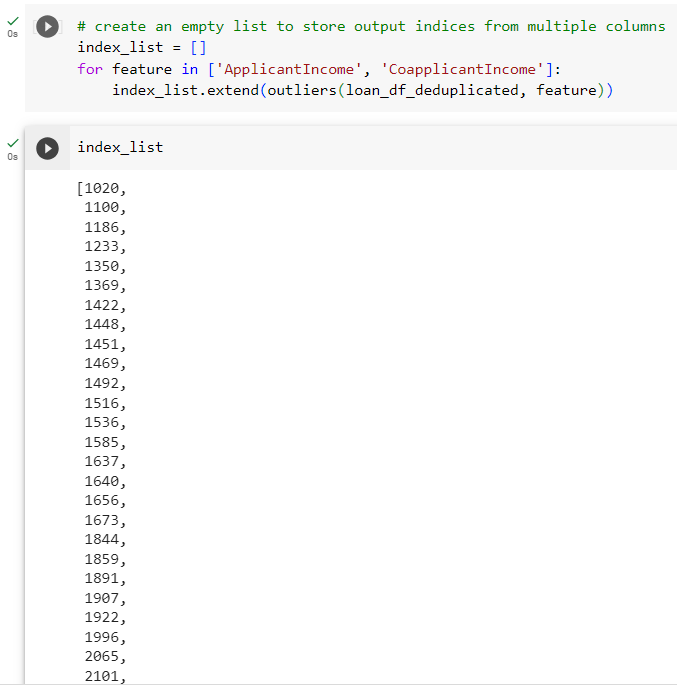


Figure 30.

The outliers for ‘ApplicantIncome’ have been removed as shown in Figure 31.

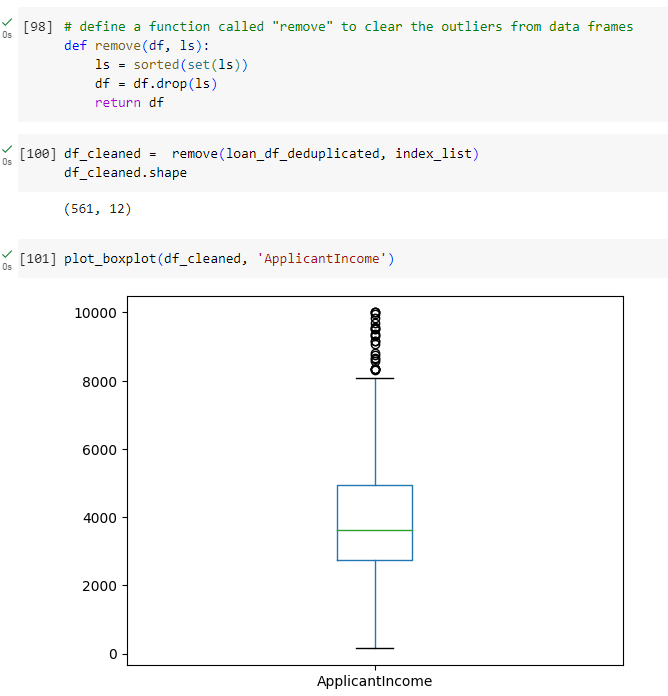


Figure 31.

Likewise, Figure 32 represents the removed outliers for the ‘CoapplicantIncome’ field:

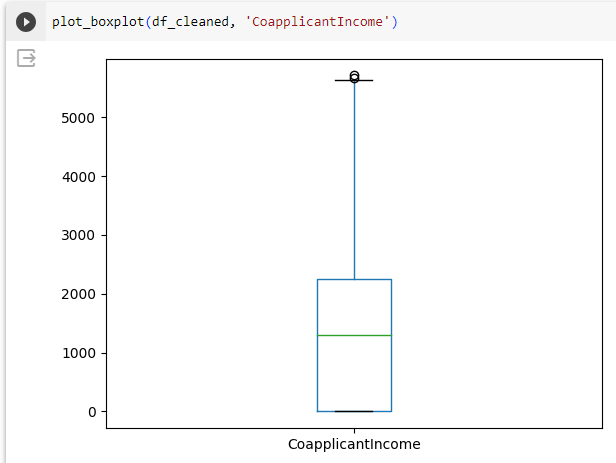


Figure 32.

* + - * 1. **Exploratory Data Analysis**

The exploratory data analysis of ZFS applicants’ data has been performed by using different mathematical operations as below:

**Input:**

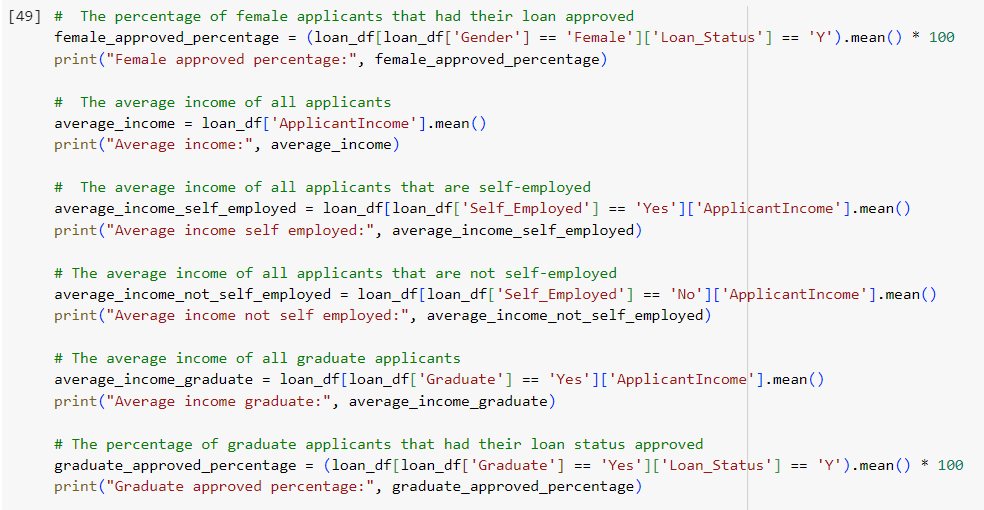


Figure 33.

**Output:**

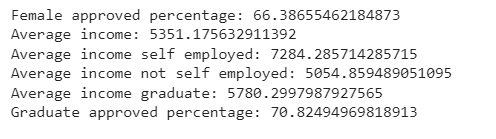


Figure 34.

This analysis calculates the values/percentages as shown in Figure 34.

* + - * 1. **Visualization**

Firstly, Figure 7 in a previous section shows the percentage of men and women applicants whose loans have been approved. This value is calculated from the loan data frame.

Secondly, Figure 35 represents the average income of all applicants according to a loan data frame. Hence, the average income of applicants is 5351 and its percentage is 53.5%.

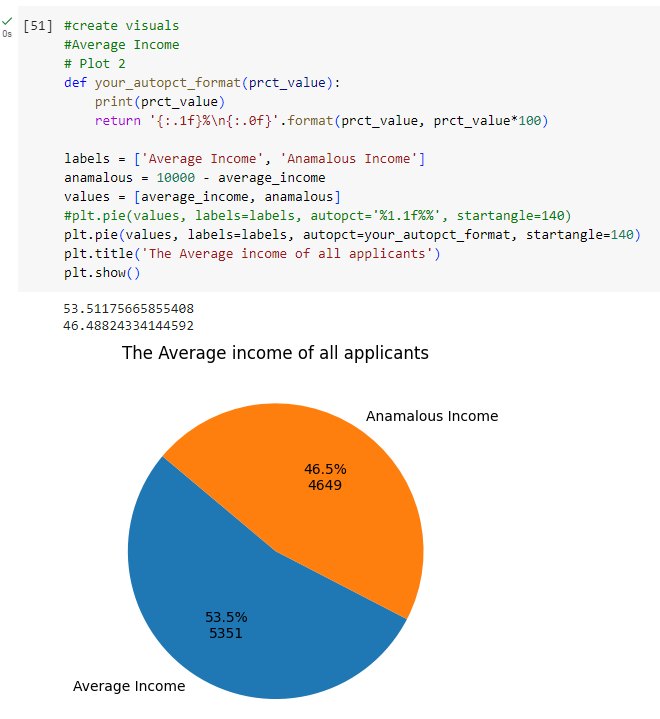


Figure 35.

Thirdly, Figure 36 shows that the average income of self-employed applicants is 5903. Conversely, the average income of those who are not self-employed is 4097.



Figure 36.

Fourthly, the average income of applicants who graduated is 5903 as shown in Figure 37. Whereas, applicants who have not done graduation, their an average income is 4097 as shown in Figure 37.

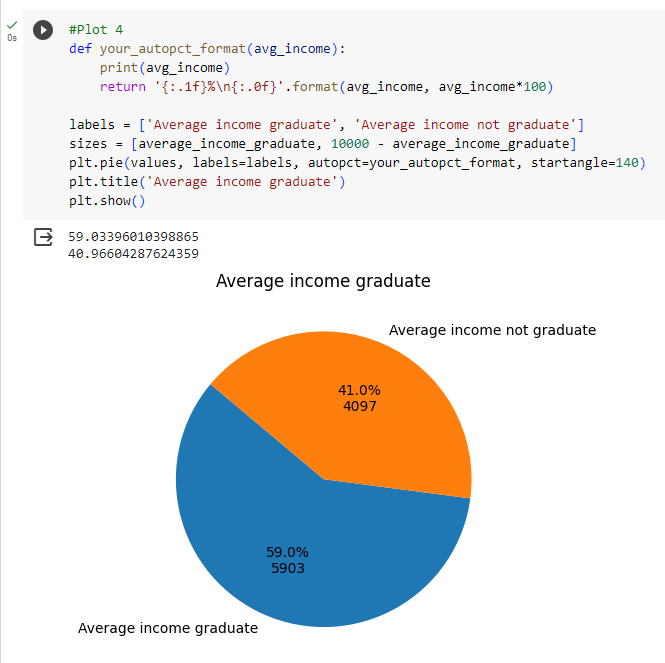


Figure 37.

Lastly, Figure 38 shows that the percentage of applicants with graduation degrees whose loans are approved is 70.8%. On the other hand, 29.2% of applicants have not graduated but their loans are approved.

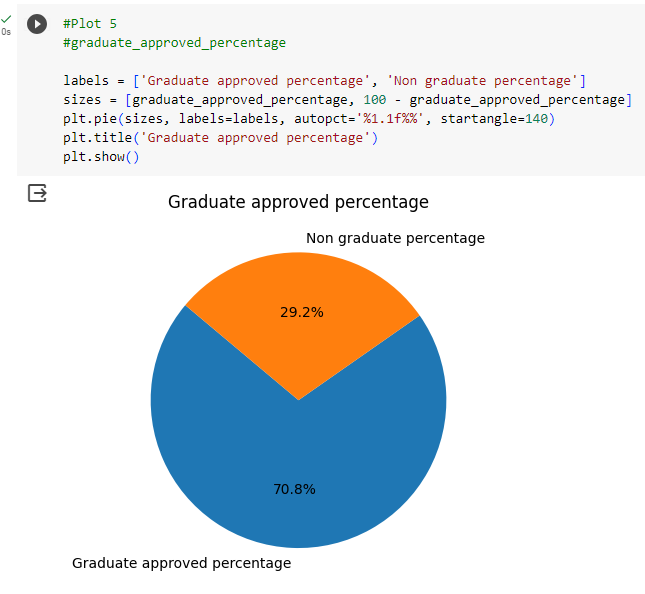


Figure 38.

* + 1. **Verification**

The quality assurance engineer verifies the code in this stage. Besides, the developer performs unit testing in the environment (JiravaÚstav, 2004).

* + - 1. **Unit testing**

The developer performs this testing at his end after finishing the coding (Olan, 2003).

**Test case 1:**

Verify there is no error when the code is executed.

* **Steps to perform:** Run the programming file with the .ipynb extension.
* **Expected Result:** There shouldn’t be any error when the program is executed.
* **Actual Result:** There is no error in executing the program.
  + - 1. **Black Box Testing**

This involves verifying whether the external behaviour of the software is working as expected or not ([Vukašinović](https://www.researchgate.net/profile/Mladen-Vukasinovic?_sg%5B0%5D=o-Rj1pNWpkvP7EVR2uCWlOadpQONclI2hN_Jprz95fl6Oc8iXYbX0Ml8ZAgqA5Mk21T3fkw.6xvi4oZR7tgXc8T9gLJs667q0DeTSrEcH8tcJ-_GD3AheuzZh116zpKSUh3HfMhqdVmyt30KC1AF-Turbqn1Hw&_sg%5B1%5D=tAfnmvt7xyPDfS7NWdm81rZXPAjdOLEmLZl3XZ0Jsq7pJPaPaft98dcWvURNjYAitNJLK2w._n2mOcodFpYqpx0IPuGfNAxSQRtg206Qwe6S0w7vFv9QrK-lYThpzmtIi5fdKXchko7TgCQUX2SON6UifVUU1Q&_tp=eyJjb250ZXh0Ijp7ImZpcnN0UGFnZSI6InNlYXJjaCIsInBhZ2UiOiJwdWJsaWNhdGlvbiIsInByZXZpb3VzUGFnZSI6InNlYXJjaCIsInBvc2l0aW9uIjoicGFnZUhlYWRlciJ9fQ), 2023).

**Test case 2:**

Verify that the read\_pdf function can read tables from all the pages of a PDF file.

* **Steps to perform:** Run the programming command in sequence as shown in Figure 4.
* **Expected Result:** The program should be able to read a table from all the pages of the PDF file.
* **Actual Result:** Figure 5 represents that all the pages have been read from the PDF file.
  + - 1. **Functional Testing**

This type of testing ensures that the functional requirements of the code are met successfully (Lawanna, 2012).

* + 1. **Maintenance**

This involves focusing on the following aspects in the future (Memon et al., 2018):

* Bug fixing
* Upgrade
* Enhancement with time

1. **Recommendations**

The following are the recommendations to automate the loan approval system (Lamba, 2022):

* Time Data synchronization and validation techniques should be used to mitigate the risks of duplicate data and missing values extracted from the database of ZFS
* Data columns should have a standard format. For instance, by mapping the labels to the values in columns so that data analysis can be done easily.
* Establishing a decision engine that would make rejection or approval decisions based on machine learning models (Bondavalli, 2010).
* Use of predictive models and logistic regression
* Integration with customer relationship management to inform applicants about the decisions of their applications
* Data protection through encryption

1. **Conclusion**

ZFS has a manual loan application system that causes delays in processing the requests. The automated Python programming solution will add value to the business of ZFS by bringing scalability, managing risk, improving efficiency, providing better customer experience, and making quick data-driven decisions. The proposed programming solution can allow uploading loan data files without manual effort, merging files, cleaning, exploratory data analysis, visualization, and reporting. However, to improve the loan approval system, ZFS is recommended to implement time data synchronization and verification techniques to handle real-time data-related issues by using machine learning and predictive model technologies (Gowrishankar, 2018).

1. **Appendix 1 – Code**

The proposed programming solution in Python that will add business value for ZFS is attached as follows:

****

1. **References**

* Bondavalli, A., Brancati, F., Ceccarelli, A., Vadursi, M. (2010) Experimental Validation of a Synchronization Uncertainty-Aware Software Clock. doi: 245-254. 10.1109/SRDS.2010.35.
* Building the AI Bank of the Future (2021) *Mckinsey & Company.* Available at: <https://www.mckinsey.com/~/media/mckinsey/industries/financial%20services/our%20insights/building%20the%20ai%20bank%20of%20the%20future/building-the-ai-bank-of-the-future.pdf> (Accessed: 21 October 2023).
* Gowrishankar, S. and Veena, A. (2018) ‘Parts of Python programming language’, *Introduction to Python Programming*, pp. 35–66. doi:10.1201/9781351013239-2.
* Hossain, M. (2023) Software Development Life Cycle (SDLC) Methodologies for Information Systems Project Management. doi: 10.36948/ijfmr.2023.v05i05.6223.
* JiravaÚstav, P. (2004) System Development Life Cycle. Available at: <https://www.researchgate.net/publication/44982277_System_development_life_cycle>
* Lamba, M., Margam, M. (2022) Predictive Modeling. doi: 10.1007/978-3-030-85085-2\_8.
* Lawanna, A. (2012) ‘The Theory of Software Testing’, *ITS Journal - Intelligent Transportation Systems Journal.*
* Memon, S., Maham, U. (2018) Importance of Software Life Cycle in Development of Quality Software.
* Nitnaware, R. (2019) ‘Basic Fundamental of Python Programming Language and The Bright Future’, *A Peer-Reviewed Journal About.*
* Obeidat, J. (2013) A new vision for information technology project management through selecting SDLC model.
* Ostan, M. (2003) Unit testing: Test early, test often.
* Opstal, T. (2010) A Structured Approach to Software Security. doi: 10.1007/978-3-8348-9363-5\_28.
* Pessoa, T., Medeiros, R., Nepomuceno, T., Filho, P. (2018) Performance Analysis of Google Colaboratory as a Tool for Accelerating Deep Learning Applications.
* Shankar, S.P. and Chaudhari, S.S. (2023) ‘Framework for the automation of SDLC phases using artificial intelligence and Machine Learning Techniques’, *International Journal on Recent and Innovation Trends in Computing and Communication*, 11(6s), pp. 379–390. doi: 10.17762/ijritcc.v11i6s.6944.
* Taherdoost, H. (2023) Change Management. doi: 10.1007/978-3-031-39626-7\_13.
* [Vukašinović](https://www.researchgate.net/profile/Mladen-Vukasinovic?_sg%5B0%5D=o-Rj1pNWpkvP7EVR2uCWlOadpQONclI2hN_Jprz95fl6Oc8iXYbX0Ml8ZAgqA5Mk21T3fkw.6xvi4oZR7tgXc8T9gLJs667q0DeTSrEcH8tcJ-_GD3AheuzZh116zpKSUh3HfMhqdVmyt30KC1AF-Turbqn1Hw&_sg%5B1%5D=tAfnmvt7xyPDfS7NWdm81rZXPAjdOLEmLZl3XZ0Jsq7pJPaPaft98dcWvURNjYAitNJLK2w._n2mOcodFpYqpx0IPuGfNAxSQRtg206Qwe6S0w7vFv9QrK-lYThpzmtIi5fdKXchko7TgCQUX2SON6UifVUU1Q&_tp=eyJjb250ZXh0Ijp7ImZpcnN0UGFnZSI6InNlYXJjaCIsInBhZ2UiOiJwdWJsaWNhdGlvbiIsInByZXZpb3VzUGFnZSI6InNlYXJjaCIsInBvc2l0aW9uIjoicGFnZUhlYWRlciJ9fQ), M. (2023) SOFTWARE QUALITY ASSURANCE.
* Wenzi, S., Miladinovic, I. (2022) Improving Scalability of Software Engineering Courses. doi: 10.1007/978-3-031-21569-8\_36.
* White, A. (2010) ‘A review of the impact of requirements on software project development using a control theoretic model’, *Journal of Software Engineering and Applications*, 03(09), pp. 852–857. doi:10.4236/jsea.2010.39099.