National College of Ireland

Configuration Manual

MSc Research Project
Fintech

Fernando Garcia Garcia
Student ID: 23142448

School of Computing
National College of Ireland

Supervisor: Brian Byrne

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Fernando Garcia……

Student ID: 23142448……..……

Programme: Fintech…… **Programme:**Fintech……

Module: MSC Research Project……..……

Supervisor: Brian Byrne……..……

Submission Due Date: August 12, 2024……..……

Project Title: Modernizing Pensions: Blockchain's Potential to Revolutionize Retirement for Young Mexicans……..…

Word Count: 5773 …… **Page Count** 19……..……..

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ……

Date: ……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

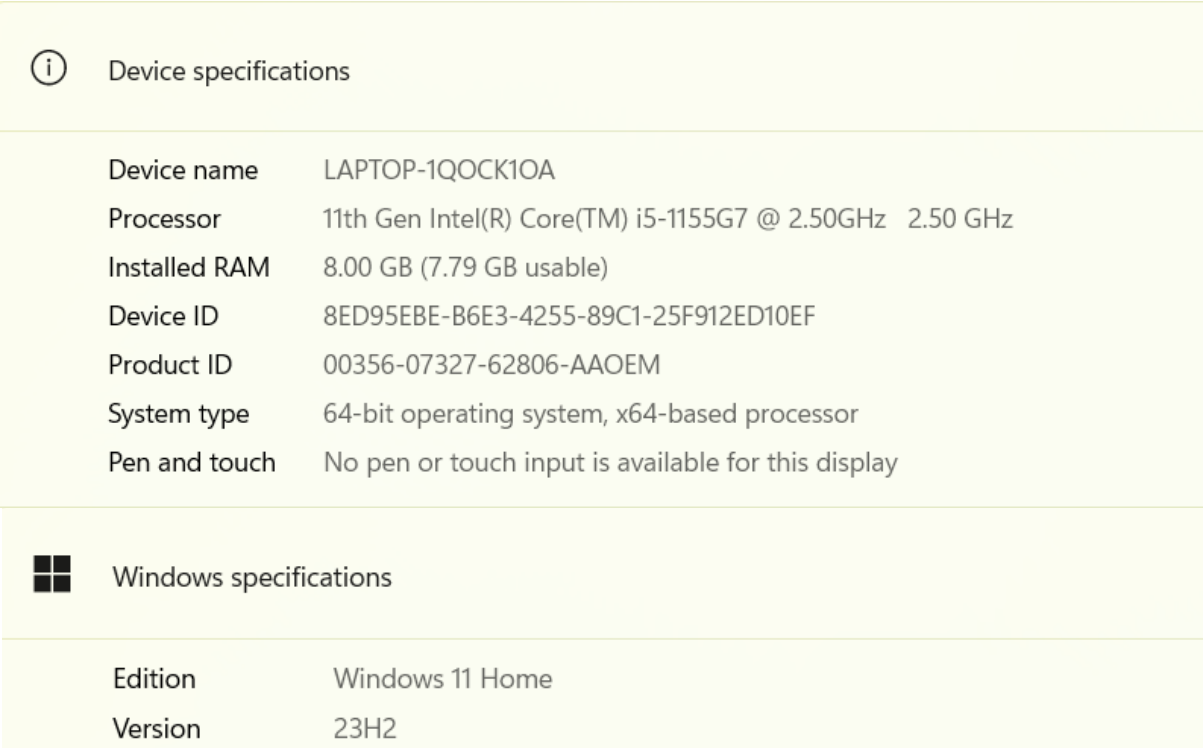
Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Fernando Garcia Garcia
Student ID: x23142448

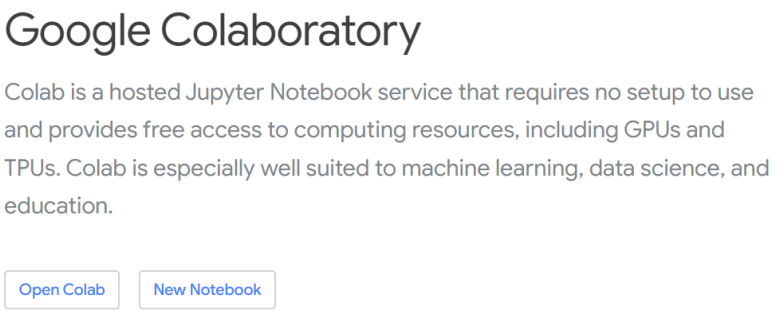
1 Hardware/Software

1.1 Hardware/Software



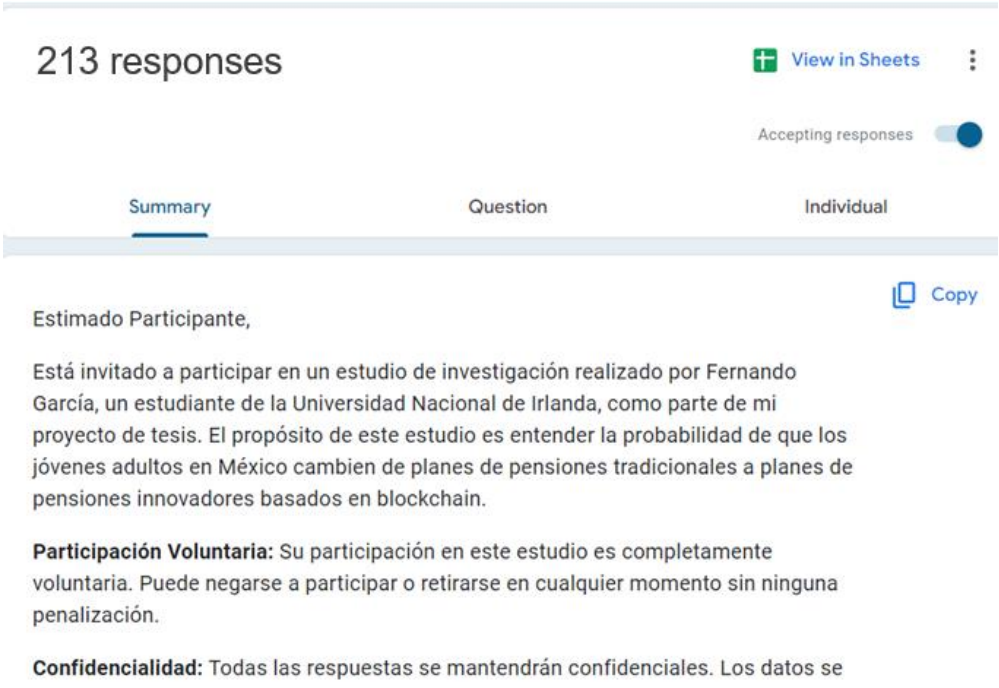
1.2 Data Analysis

For this project, the latest version of Google Colab (version 1.0.0) was utilized, running Python 3.12.5. This environment provides the necessary libraries and tools pre-installed, allowing for efficient and scalable execution of Python code without the need for extensive local configuration.



2 Data Collection

Google Forms was used to gather information for this research. It was chosen because it allows for an unlimited number of questions and responses for free. The setup was easy, and it provided a convenient way to distribute the survey and collect responses.



Estimado Participante, *

Está invitado a participar en un estudio de investigación realizado por Fernando García, un estudiante de la Universidad Nacional de Irlanda, como parte de mi proyecto de tesis. El propósito de este estudio es entender la probabilidad de que los jóvenes adultos en México cambien de planes de pensiones tradicionales a planes de pensiones innovadores basados en blockchain.

Participación Voluntaria: Su participación en este estudio es completamente voluntaria. Puede negarse a participar o retirarse en cualquier momento sin ninguna penalización.

Confidencialidad: Todas las respuestas se mantendrán confidenciales. Los datos se almacenarán de manera segura y solo el equipo de investigación tendrá acceso a ellos. Su nombre u otra información identificativa no se utilizará en ningún informe o publicación resultante de este estudio.

Impacto Potencial: Si bien participar en este estudio no presenta riesgos significativos, proporcionará información valiosa sobre las actitudes y percepciones de los jóvenes adultos hacia soluciones de pensiones innovadoras. Sus respuestas ayudarán a moldear las futuras políticas y estrategias de pensiones en México, potencialmente beneficiando a las generaciones futuras.

Compromiso de Tiempo: La encuesta tomará aproximadamente 10-15 minutos en completarse.

Información de Contacto: Si tiene alguna pregunta sobre esta investigación, no dude en contactarme en x23142448@student.ncirl.ie

Al proceder con esta encuesta, usted reconoce que ha leído y comprendido la información proporcionada y acepta participar en este estudio.

Gracias por su tiempo y valiosa contribución.

Atentamente, Fernando García
Universidad Nacional de Irlanda (NCI)

- ◯ Estoy de acuerdo
- ◯ No estoy de acuerdo

Fig 1 Consent Page

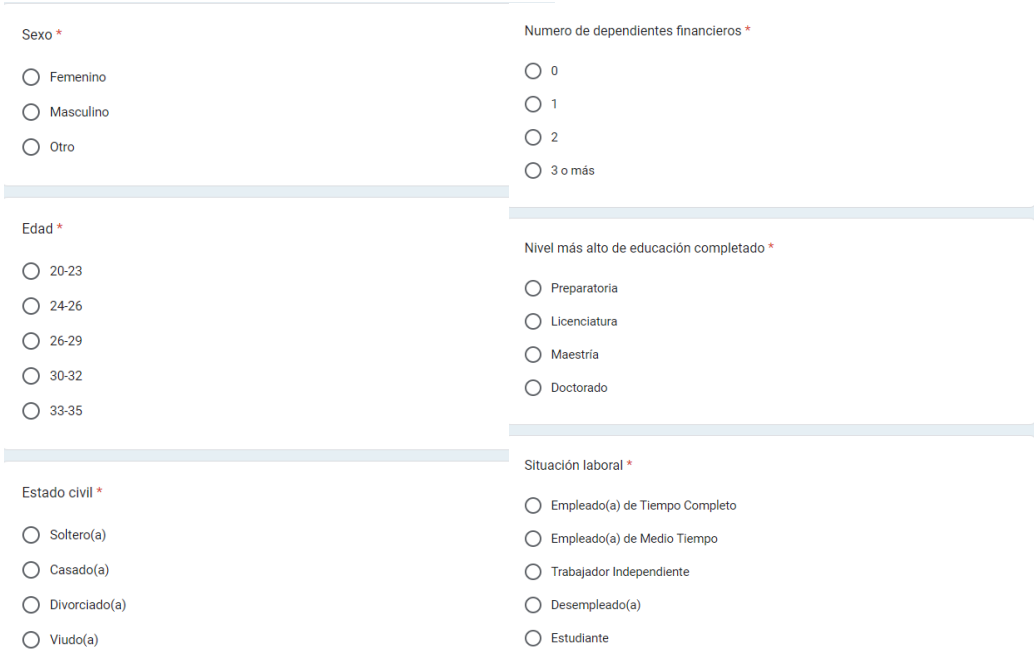


Fig 2 Questions 1-6

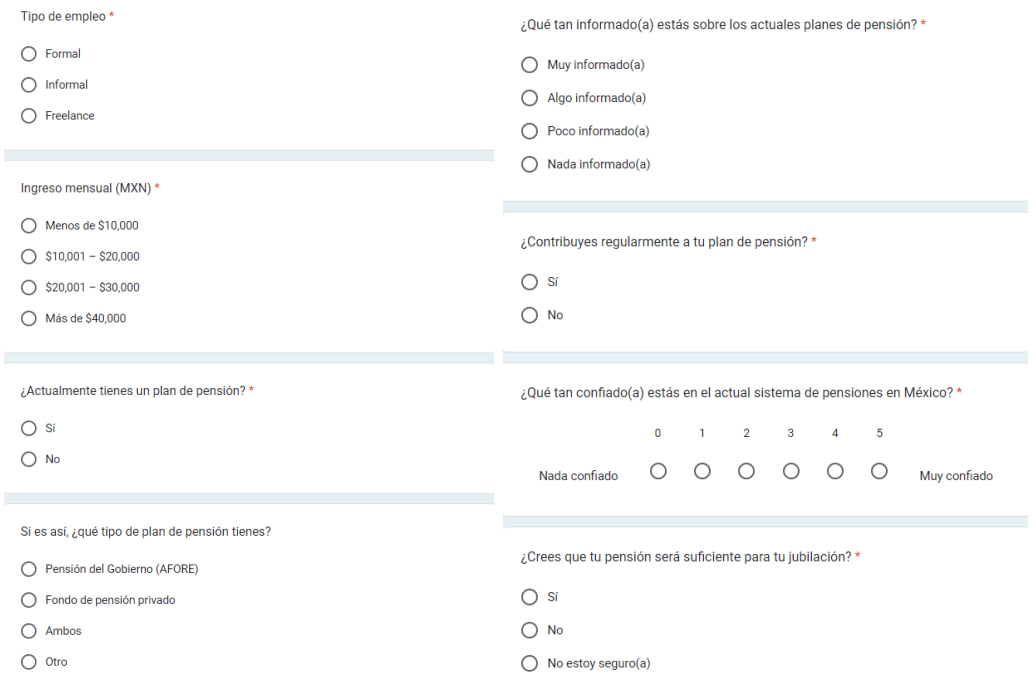


Fig 3 Questions 7-14

¿Tienes (o has tenido) algún ahorro adicional para la jubilación (por ejemplo, inversiones, cuentas de ahorro)? *

- Sí
- No

¿Cómo calificarías tu conocimiento financiero en general? *

	0	1	2	3	4	5	
Nada Alfabetizado	○	○	○	○	○	○	Muy Alfabetizado

¿Qué tan cómodo(a) te sientes usando servicios financieros digitales (por ejemplo, banca en línea, pagos móviles)? *

	0	1	2	3	4	5	
Nada cómodo	○	○	○	○	○	○	Muy cómodo

Antes de este estudio, ¿habías oído hablar de la tecnología blockchain? *

- Sí
- No

Si es así, ¿qué tan familiarizado(a) estás con la tecnología blockchain? *

- Muy familiarizado(a)
- Algo familiarizado(a)
- Poco familiarizado(a)
- Nada familiarizado(a)

¿Cuánto confías en la tecnología blockchain para transacciones financieras en una escala del 0 al 5? *

	0	1	2	3	4	5	
Nada de confianza	○	○	○	○	○	○	Confianza total

¿Considerarías cambiar tu plan de pensión a un sistema basado en blockchain? *

- Definitivamente sí
- Probablemente sí
- No estoy seguro(a)
- Probablemente no
- Definitivamente no

Fig 4 Questions 15-21

¿Cuáles son tus principales preocupaciones sobre el uso de la tecnología blockchain para las pensiones? (Selecciona todas las que correspondan) *

- ☐ Preocupaciones de seguridad
- ☐ Falta de comprensión
- ☐ Problemas de confianza
- ☐ Problemas regulatorios
- ☐ Volatilidad

¿Crees que la tecnología blockchain puede mejorar la transparencia de los fondos de pensiones? *

- Sí
- No
- No estoy seguro(a)

¿La implementación de la tecnología blockchain te haría más propenso(a) a invertir en un plan de pensión? *

- Sí
- No
- No estoy seguro(a)

¿Estarías dispuesto(a) a aprender más sobre la tecnología blockchain si se implementara en los sistemas de pensiones? *

- Sí
- No

¿Qué se necesitaría para hacer la transición de un sistema de pensiones tradicional a un sistema de pensiones basado en blockchain? *

Your answer

Back Submit Clear form

Fig 5 Questions 22-26

	A	B	C	D	E	F	G	H	I	J	K
1	Timestamp		Sex	Age	Status	Highest Level of Educ	Employment St	Type of Employr	Monthly Income (In MXN)	Do you currer	If yes, what type of pension plan c
2	12/06/2024 00:42	I Agree	Male	27-29	Single	Bachelor's Degree	Full-time	Freelance	$10,001 – $20,000	Yes	Private pension fund
3	12/06/2024 01:37	I Agree	Male	20-23	Single	Bachelor's Degree	Full-time	Formal	Less than $10,000	Yes	Government Pension (AFORE)
4	12/06/2024 01:39	I Agree	Male	20-23	Married	Bachelor's Degree	Full-time	Formal	Less than $10,000	Yes	Other
5	12/06/2024 03:15	I Agree	Female	20-23	Single	Bachelor's Degree	Full-time	Formal	Less than $10,000	Yes	Both
6	12/06/2024 03:37	I Agree	Male	20-23	Single	High school	Student	Formal	Less than $10,000	Yes	Both
7	12/06/2024 03:38	I Agree	Female	27-29	Single	Master's Degree	Full-time	Formal	$20,001 – $30,000	Yes	Both
8	12/06/2024 04:06	I Agree	Male	27-29	Married	Bachelor's Degree	Full-time	Formal	$30,001 – $40,000	Yes	Government Pension (AFORE)
9	12/06/2024 05:18	I Agree	Female	20-23	Single	High school	Full-time	Formal	Less than $10,000	Yes	Government Pension (AFORE)
10	12/06/2024 07:01	I Agree	Male	20-23	Single	High school	Full-time	Formal	$10,001 – $20,000	No	
11	12/06/2024 07:13	I Agree	Female	20-23	Single	High school	Student	Formal	Less than $10,000	No	
12	12/06/2024 07:49	I Agree	Female	20-23	Single	Bachelor's Degree	Full-time	Formal	$20,001 – $30,000	Yes	
13	12/06/2024 08:35	I Agree	Female	27-29	Married	Master's Degree	Unemployed	Informal	Less than $10,000	Yes	Both
14	12/06/2024 09:50	I Agree	Female	27-29	Single	Master's Degree	Full-time	Formal	$20,001 – $30,000	Yes	Government Pension (AFORE)
15	12/06/2024 11:54	I Agree	Female	27-29	Married	Master's Degree	Full-time	Formal	$20,001 – $30,000	Yes	Government Pension (AFORE)
16	12/06/2024 15:01	I Agree	Female	27-29	Divorced	Bachelor's Degree	Full-time	Formal	More than $40,000	Yes	Government Pension (AFORE)
17	12/06/2024 17:09	I Agree	Male	27-29	Married	Bachelor's Degree	Full-time	Formal	$20,001 – $30,000	No	
18	12/06/2024 17:49	I Agree	Male	27-29	Married	Bachelor's Degree	Full-time	Informal	More than $40,000	Yes	
19	12/06/2024 18:38	I Agree	Female	27-29	Widowed	Bachelor's Degree	Full-time	Formal	$30,001 – $40,000	Yes	Private pension fund
20	12/06/2024 19:13	I Agree	Female	20-23	Single	High school	Student	Formal	Less than $10,000	No	
21	12/06/2024 21:32	I Agree	Male	27-29	Married	Bachelor's Degree	Full-time	Formal	$30,001 – $40,000	No	
22	12/06/2024 23:49	I Agree	Female	27-29	Married	Bachelor's Degree	Full-time	Formal	More than $40,000	Yes	Government Pension (AFORE)
23	13/06/2024 00:19	I Agree	Male	27-29	Single	Bachelor's Degree	Self-employed	Formal	$20,001 – $30,000	Yes	
24	13/06/2024 04:43	I Agree	Female	27-29	Married	High school	Full-time	Formal	Less than $10,000	No	

Fig 6 CSV Survey Data (After being translated)

3 Data Analysis

3.1 Percentages

```
# Count occurrences of each category in the 'status' column
status_counts = df['Status'].value_counts(normalize=True) * 100

# Print the counts
print("Count of each Status:")
print(status_counts)
```

```
# Count occurrences of each category in the 'Income' column
status_counts = df['Monthly Income (In MXN)'].value_counts(normalize=True) * 100

# Print the counts
print(status_counts)
```

3.2 Concatenating for word count

```
# Concatenate the three columns into a single series, ignoring NaNs
concerns = pd.concat([
    df["What are your main concerns about using blockchain technology for pensions? (Select all that apply)"].dropna(),
    df["What are your main concerns about using blockchain technology for pensions? (2)"].dropna(),
    df["What are your main concerns about using blockchain technology for pensions? (3)"].dropna()
], axis=0)

# Count the occurrences of each value
concerns_counts = concerns.value_counts()

# Calculate the percentage of each concern
total_entries = concerns_counts.sum()
concerns_percentages = (concerns_counts / total_entries) * 100

# Print the counts and percentages
print("Concerns about using blockchain technology for pensions (counts):")
print(concerns_counts)
print("\nConcerns about using blockchain technology for pensions (percentages):")
print(concerns_percentages)
```

3.3 Word count for open-ended question

```
# Download necessary NLTK data
nltk.download('punkt')
nltk.download('stopwords')

# Extract the open-ended responses column
open_ended_responses = df["What would it take to transition from a traditional pension system to a blockchain pension system?"].dropna()

# Combine all responses into a single string
all_text = ' '.join(open_ended_responses)

# Convert to lowercase
all_text = all_text.lower()

# Remove punctuation
all_text = all_text.translate(str.maketrans('', '', string.punctuation))

# Tokenize the text
words = word_tokenize(all_text)

# Remove stopwords
stop_words = set(stopwords.words('english'))  # Assuming the text is in English
filtered_words = [word for word in words if word not in stop_words]

# Count word frequencies
word_counts = Counter(filtered_words)

# Get the most common words
most_common_words = word_counts.most_common(10)
```

3.4 Correlation matrix

```
def print_unique_values(df, columns):
    for col in columns:
        unique_values = df[col].unique()
        print(f"\nUnique values in '{col}' column:")
        print(unique_values)

# Load the CSV file with the appropriate encoding
def load_data(file_path):
    return pd.read_csv(file_path, encoding='ISO-8859-1')

# File path to the CSV file
file_path = 'Survey_Answers_Revised.csv'  # Replace with your actual file path

# Reload the data to ensure original values are used
df = load_data(file_path)

# Identify date columns and convert them to datetime format
date_columns = ['Timestamp']
for col in date_columns:
    df[col] = pd.to_datetime(df[col], errors='coerce')

# Exclude the first two columns
df = df.iloc[:, 2:]

# Convert 'Age' to numerical using the specified categories
age_mapping = {
    '20-23': 21.5,
    '24-26': 25,
    '27-29': 28
}
df['Age'] = df['Age'].map(age_mapping)
```

```
# Assuming 'Monthly Income (In MXN)' is a categorical variable with income ranges, map them to numerical values
income_mapping = {
    'Less than $10,000': 5000,
    '$10,001 - $20,000': 15000,
    '$20,001 - $30,000': 25000,
    '$30,001 - $40,000': 35000,
    'More than $40,000': 45000
}
df['Monthly Income (In MXN)'] = df['Monthly Income (In MXN)'].map(income_mapping)

# Convert 'yes'/'no' answers to binary values for multiple columns
binary_columns = [
    'Do you currently have a pension plan?',
    'Do you have (or have had) any additional retirement savings (e.g., investments, savings accounts)?',
    'Before this study, have you heard about blockchain technology?',
    'Would you be willing to learn more about blockchain technology if it were to be implemented in pension systems?'
]
for col in binary_columns:
    df[col] = df[col].map({'Yes': 1, 'No': 0})

# Convert other relevant columns to numerical if they are not already
df['How confident are you in the current pension system in Mexico?'] = pd.to_numeric(df['How confident are you in the current pension system in Mexico?'], errors='coerce')
df['How would you rate your overall financial literacy?'] = pd.to_numeric(df['How would you rate your overall financial literacy?'], errors='coerce')
df['How comfortable are you with using digital financial services (e.g., online banking, mobile payments)?'] = pd.to_numeric(df['How comfortable are you with using digital financial s
df['How much do you trust blockchain technology for financial transactions on a scale of 0 to 5?'] = pd.to_numeric(df['How much do you trust blockchain technology for financial transa
```

```
# Combine all numerical columns for correlation analysis
numerical_columns = [
    'Age',
    'Monthly Income (In MXN)',
    'How confident are you in the current pension system in Mexico?',
    'How would you rate your overall financial literacy?',
    'How comfortable are you with using digital financial services (e.g., online banking, mobile payments)?',
    'How much do you trust blockchain technology for financial transactions on a scale of 0 to 5?',
    'Do you currently have a pension plan?',
    'Do you have (or have had) any additional retirement savings (e.g., investments, savings accounts)?',
    'Before this study, have you heard about blockchain technology?',
    'Would you be willing to learn more about blockchain technology if it were to be implemented in pension systems?'
]

# Drop rows with any NaN values in the numerical columns to avoid issues in the correlation matrix
df_clean = df[numerical_columns].dropna()

# Calculate the correlation matrix
correlation_matrix = df_clean.corr()

# Plot the heatmap with r values
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, fmt=".2f", cmap='coolwarm', vmin=-1, vmax=1)
plt.title("Correlation Matrix Heatmap")
plt.show()
```

3.5 Ordinal logistic regression

```
# Load the dataset with the correct encoding
file_path = 'Survey_Answers_Revised.csv'
data = pd.read_csv(file_path, encoding='ISO-8859-1')

# Ensure the 'Would you consider switching your pension plan to a blockchain-based system?' column is categorical
dependent_var = 'Would you consider switching your pension plan to a blockchain-based system?'
data[dependent_var] = data[dependent_var].astype('category')

# Define the independent variables
independent_vars = ['Age', 'Sex', 'Status', 'Highest Level of Education Completed', 'Monthly Income (In MXN)']

# Drop rows with missing values in these columns
data = data.dropna(subset=[dependent_var] + independent_vars)

# Encode categorical variables (independent variables)
X = pd.get_dummies(data[independent_vars], drop_first=True)

# Define the dependent variable
y = data[dependent_var].cat.codes  # Convert categories to numeric codes for the model

# Fit the ordinal logistic regression model
model = OrderedModel(y, X, distr='logit')
result = model.fit(method='bfgs')

# Display the results
print(result.summary())
```

3.6 Decision Tree

```
# Handle missing values if necessary (e.g., drop rows with missing values)
df = df.dropna(subset=['Likelihood to Switch', 'Age', 'Income Category'])

# Prepare data for regression
X = df[['Age', 'Sex', 'Status', 'Highest Level of Education Completed', 'Income Category']]
X = pd.get_dummies(X, drop_first=True)  # One-hot encode categorical variables
y = df['Likelihood to Switch']

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Fit the decision tree model
tree = DecisionTreeClassifier(max_depth=3, random_state=42)
tree.fit(X_train, y_train)

# Plot the decision tree
plt.figure(figsize=(20, 10))
plot_tree(tree, feature_names=X.columns.tolist(), class_names=[str(i) for i in np.unique(y)], filled=True)
plt.title("Decision Tree for Likelihood to Switch")
plt.show()

# Predict and evaluate
y_pred_tree = tree.predict(X_test)
print(classification_report(y_test, y_pred_tree))
```

3.7 Random Forest

```
# Prepare data for regression
X = df[['Age', 'Sex', 'Status', 'Highest Level of Education Completed', 'Income Category']]
X = pd.get_dummies(X, drop_first=True)  # One-hot encode categorical variables
y = df['Likelihood to Switch']

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Handle class imbalance using SMOTE
smote = SMOTE(random_state=42)
X_train_smote, y_train_smote = smote.fit_resample(X_train, y_train)

# Fit the random forest model
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(X_train_smote, y_train_smote)

# Feature importance
feat_importances = pd.Series(rf.feature_importances_, index=X.columns)
feat_importances.nlargest(10).plot(kind='barh')
plt.title('Top 10 Feature Importances')
plt.show()

# Predict and evaluate
y_pred_rf = rf.predict(X_test)
print(classification_report(y_test, y_pred_rf))
```

3.8 Ridge regression

```
# Prepare data for regression
X = df[['Age', 'Sex', 'Status', 'Highest Level of Education Completed', 'Income Category']]
X = pd.get_dummies(X, drop_first=True)  # One-hot encode categorical variables
y = df['Likelihood to Switch']

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Define custom Ordinal Ridge Regression model
class OrdinalRidgeRegression:
    def __init__(self, alpha=1.0):
        self.alpha = alpha
        self.model = LogisticIT()
        self.ridge = Ridge(alpha=self.alpha)

    def fit(self, X, y):
        self.model.fit(X, y)
        self.ridge.fit(X, y)
        self.coef_ = self.model.coef_ - self.alpha * self.ridge.coef_

    def predict(self, X):
        return self.model.predict(X)

# Instantiate and fit the model
ordinal_ridge = OrdinalRidgeRegression(alpha=1.0)
ordinal_ridge.fit(X_train, y_train)

# Predict and evaluate
y_pred = ordinal_ridge.predict(X_test)
print(classification_report(y_test, y_pred))
```

3.9 Lasso regression

```
# Prepare data for regression
X = df[['Age', 'Sex', 'Status', 'Highest Level of Education Completed', 'Income Category']]
X = pd.get_dummies(X, drop_first=True)  # One-hot encode categorical variables
y = df['Likelihood to Switch']

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Fit the Lasso regression model
lasso = Lasso(alpha=0.1)  # You can adjust the alpha parameter for regularization strength
lasso.fit(X_train, y_train)

# Predict and evaluate
y_pred = lasso.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error: {mse}')
print(f'R^2 Score: {r2}')

# Print coefficients
print('Coefficients:', lasso.coef_)

# Feature importance plot
plt.figure(figsize=(10, 6))
coef_series = pd.Series(lasso.coef_, index=X.columns)
coef_series = coef_series[coef_series != 0]  # Only plot non-zero coefficients
coef_series.sort_values().plot(kind='barh')
plt.title('Lasso Regression Feature Importances')
plt.show()
```