AN INTERNSHIP REPORT ON

“CURRENCY DETECTION USING TENSORFLOW LITE”



Submitted to

SAVITRIBAI PHULE PUNE UNIVERSITY

In Partial Fulfilment of the Requirement for the Award of

BACHELOR OF ENGINEERING IN

INFORMATION TECHNOLOGY

BY

“PALLAVI AKOLKAR [IT3002]”



DEPARTMENT OF Information Technology

TRINITY COLLEGE OF ENGINEERING AND RESEARCH

Haveli Pisoli Annex, Pune - 411048

2024-2025

TRINITY COLLEGE OF ENGINEERING AND RESEARCH

DEPARTMENT OF INFORMATION TECHNOLOGY



CERTIFICATE

This is certify that the Internship Report entitled

“CURRENCY DETECTION USING TENSORFLOW LITE”

submitted by

PALLAVI AKOLKAR [IT3002]

Is a record of bonafide work carried out by PALLAVI AKOLKAR [IT3002] has successfully submitted a Internship Report entitled “ Currency Detection Using TensorFlow Lite” under the guidance of Prof. Richa Agarwal in the Academic Year 2024-25 at the Information Technology Department of Trinity College of Engineering and Research, under the Savitribai Phule Pune University.

Date: / /

Place: Pune

Prof. Richa Agarwal Dr. Vilas Gaikwad

Internship Coordinator HOD, IT Department

*Currency Detection using Tensorflow lite*

CERTIFICATE OF COMPLETION



3

DECLARATION

I do hereby declare that I have carried out the work presented in this Internship report and has not been previously submitted to any other university, college, or organisation for any academic qualification, degree, or certificate.

The work I have presented does not breach any existing copyright, and no portion of this report is copied from any work done earlier for a degree or otherwise.

Pallavi Akolkar IT3002

Contents

[1 Introduction 1](#_Toc17283)

[1.1 Company Overview 1](#_Toc17284)

[1.1.1 Industry Relevance 1](#_Toc17285)

[1.1.2 Internship Objective 1](#_Toc17286)

[1.2 Week-wise Training Details 2](#_Toc17287)

[1.2.1 Week 1: Technical Foundations in ML 2](#_Toc17288)

[1.2.2 Week 2: Soft Skills & Design Thinking 3](#_Toc17289)

[1.2.3 Week 3: Advanced Technology – Unsupervised Learning & Deep Learning 4](#_Toc17290)

[1.2.4 Week 4: Introduction to IoT & Internship Project Planning 5](#_Toc17291)

[1.2.5 Week 5: Deep Learning & Introduction to SAP ABAP 6](#_Toc17292)

[1.2.6 Week 6: Computer Vision & SAP ABAP 7](#_Toc17293)

[2 Project Overview 8](#_Toc17294)

[2.1 Problem Statement 8](#_Toc17295)

[2.2 Objectives 8](#_Toc17296)

[2.3 Scope 8](#_Toc17297)

[2.4 Expected Outcomes 8](#_Toc17298)

[3 Challenges Encountered During the Internship 9](#_Toc17299)

[3.1 Limited Dataset Availability 9](#_Toc17300)

[3.2 Variation in Lighting and Backgrounds 9](#_Toc17301)

[3.3 Torn and Crumbled Notes 9](#_Toc17302)

[3.4 Accuracy vs. Real-World Performance 9](#_Toc17303)

[4 Methodology Used 10](#_Toc17304)

[4.1 Data Collection & Preprocessing 10](#_Toc17305)

[4.2 Model Architecture 10](#_Toc17306)

[4.3 Training and Evaluation 10](#_Toc17307)

[5 Technology Stack & Implementation 11](#_Toc17308)

[5.1 Technology Stack 11](#_Toc17309)

[5.2 Features Implemented 11](#_Toc17310)

[5.3 Code 12](#_Toc17311)

[6 Analysis of Results 17](#_Toc17312)

[6.1 Accuracy of Model 17](#_Toc17313)

[6.2 Real-World Testing 17](#_Toc17314)

[6.3 Voice Feedback Effectiveness 17](#_Toc17315)

[6.4 Performance 17](#_Toc17316)

[6.5 Result 18](#_Toc17317)

[7 Conclusion 23](#_Toc17318)

[8 Future Scope 24](#_Toc17319)

# Introduction

## Company Overview

The SAP Edunet Foundation, in collaboration with Edunet Foundation, is a non-profit initiative under SAP’s Corporate Social Responsibility (CSR) umbrella that aims to empower youth through technologydriven education. It works towards equipping students with essential digital and professional skills required to thrive in the fast-evolving global workforce. One of their flagship initiatives, the Code Unnati Program, is designed to offer structured, hands-on training in emerging technologies, thereby bridging the digital skills gap.

The foundation partners with educational institutions, industry experts, and technology leaders to offer world-class curriculum and mentorship. The program focuses on building competencies in fields such as Artificial Intelligence, Machine Learning, Deep Learning, the Internet of Things (IoT), Computer Vision, and SAP’s proprietary development environment – Advanced Business Application Programming (ABAP) on SAP Business Technology Platform (BTP).

### Industry Relevance

As industries rapidly adopt automation, AI, and cloud technologies, there is an increasing demand for professionals who are not only tech-savvy but also capable of adapting to evolving tools and platforms. Skills in machine learning, IoT, deep learning, and enterprise resource planning (ERP) are now considered essential for new-generation IT professionals. This internship bridges the academic-industry divide by offering applied exposure to these transformative domains.

The program aligns with the demands of the Fourth Industrial Revolution (Industry 4.0) and contributes to creating a workforce that is ready to engage with smart, connected systems and data-centric business solutions. By training students in both technical competencies and 21st-century soft skills, it prepares them to take on industry challenges effectively.

### Internship Objective

The objective of this internship was to provide students with a comprehensive learning experience that integrates both technical and professional development. The training aimed to:

Equip participants with hands-on experience in machine learning, deep learning, and computer vision.

Introduce foundational and advanced concepts of IoT and SAP ABAP development.

Foster critical soft skills such as communication, design thinking, and problem-solving.

Enable students to work on a real-world project using the acquired skills.

The final outcome of the internship was a capstone project titled “Currency Detection Using TensorFlow Lite”, which involved building a lightweight AI model for recognizing Indian currency denominations, designed for mobile deployment. This provided an opportunity to integrate learning across multiple technologies into a tangible, impactful application.

## Week-wise Training Details

### Week 1: Technical Foundations in ML

The first week is designed to lay the foundation for the rest of the internship, ensuring every participant is well-versed in the essentials of programming and core machine learning concepts. The week begins with crash courses in Python and data analytics, focusing on equipping students with the skills necessary to manipulate data and build initial models.

1. Crash Courses:
   * Python Crash Course: This includes an introduction to Python syntax, variables, control structures (if-else, loops), functions, exception handling, and object-oriented programming concepts. Students also learn to work with Python libraries such as NumPy for numerical computations and Pandas for handling tabular data. The emphasis is on writing clean, efficient code with practical exercises such as data cleaning, parsing, and basic visualization.
   * Data Analytics Introduction: Students explore the basics of data analytics, including types of data (categorical, continuous), and techniques such as data preprocessing, normalization, and feature selection. Visualization libraries like Matplotlib and Seaborn are introduced to create plots, histograms, and correlation heatmaps that help in understanding data distributions and relationships.
2. Machine Learning Modules:
   * Linear Regression: Students learn to build and interpret linear regression models, understand concepts such as mean squared error, R-squared, and model fitting. A mini-project on predicting house prices or stock market trends reinforces learning.
   * Logistic Regression: This classification algorithm is introduced with examples like email spam detection and predicting admission chances. Students understand the sigmoid function, odds ratio, and confusion matrix.
   * K-Nearest Neighbors (KNN): The concept of instance-based learning is explained through realworld examples such as handwritten digit recognition and movie recommendation systems.

Students practice selecting the optimal value of K and using distance metrics.

* + Decision Trees: Using Gini impurity and information gain, students learn to split datasets into homogenous sets. The algorithm is implemented both manually and via Scikit-learn to understand tree depth, pruning, and overfitting.
  + Support Vector Machines (SVM): The concept of hyperplanes, support vectors, and kernel tricks are explained with 2D classification problems. Emphasis is placed on visualizing decision boundaries.
  + Ensemble Methods: Random Forest, Bagging, and Boosting techniques are introduced to demonstrate the power of model aggregation. Students implement them using Scikit-learn and compare their accuracy.

### Week 2: Soft Skills & Design Thinking

This week is strategically designed to shift focus from technical rigor to personal and professional development. It bridges the gap between academic learning and the expectations of the modern workplace. Students are introduced to the human side of technology—emphasizing empathy, communication, collaboration, and strategic thinking.

1. Communication Workshops

The week kicks off with immersive communication sessions. Students participate in structured workshops that cover a range of skills:

* + Verbal Communication: Activities include impromptu speaking rounds, mock client presentations, persuasive speech practice, and structured group discussions. Students are trained to handle QA sessions and are taught how to articulate their thoughts with clarity, relevance, and confidence.
  + Written Communication: Exercises include formal email writing, LinkedIn post crafting, and professional report formatting. Trainers highlight grammar, tone, and impact. Feedback is personalized, with peer reviews enhancing collaborative learning.

1. Design Thinking:

A full-day innovation sprint introduces students to the 5-stage design thinking process:

* + Empathize: Teams engage in role-playing and persona development to understand end-user needs.
  + Define: Problem statements are framed using tools like “How Might We...” questions.
  + Ideate: Brainstorming sessions are held, encouraging quantity over quality at first. Techniques like SCAMPER and mind-mapping are used.
  + Prototype: Low-fidelity prototypes (paper models, wireframes) are created collaboratively.
  + Test: Peer groups test solutions and offer constructive feedback. Iteration is encouraged.

1. Critical Thinking Exercises

Case studies and ethical dilemma discussions are conducted. Students are asked to evaluate problems logically, assess the validity of data sources, and propose decisions supported by evidence.

Brain teasers and logic puzzles promote analytical reasoning.

1. Placement Preparation:
   * Resume Building: Students learn modern resume formats, tailoring resumes to job roles, and the importance of action verbs and quantifiable achievements. Trainers review resumes oneon-one.
   * Interview Etiquette: From dressing professionally to answering behavioral questions using the STAR method, students practice common HR and technical interview questions. Emphasis is on body language, confidence, and honesty.
   * Mock Interviews: Simulated interviews conducted by mentors and industry guests help students experience real-world interview pressure. Feedback sessions are detailed, focusing on improvement areas.

### Week 3: Advanced Technology – Unsupervised Learning & Deep Learning

This week marks a transition into more advanced domains of machine learning, challenging students to think beyond labeled data and explore algorithms that uncover hidden patterns. It also introduces the foundational elements of deep learning, setting the stage for neural network-based models in upcoming weeks.

1. Unsupervised Learning: Unlike supervised methods, unsupervised learning does not rely on labeled data. Instead, it finds structure in the input. Students dive deep into three key techniques:
   * K-Means Clustering: Students begin by learning the core concept of clustering—grouping similar data points without prior labels. The process starts with understanding how centroids are initialized and how the algorithm iteratively refines them to minimize intra-cluster distance. Emphasis is placed on the importance of the elbow method for selecting the optimal number of clusters (K). Use-cases explored include customer segmentation based on shopping behavior and document classification using TF-IDF vectors. Students implement K-Means using both Scikit-learn and from-scratch logic to understand internal computations.
   * Principal Component Analysis (PCA): Students explore eigenvectors, eigenvalues, and how PCA reduces dimensions while retaining variance. They visualize high-dimensional datasets in 2D using PCA plots.
   * Hierarchical Clustering: Students understand agglomerative clustering, distance matrices, and dendrograms. Clustering genes based on expression patterns is one application explored.
2. Deep Learning Introduction:
   * Neural Networks Basics: The session begins with drawing parallels between biological and artificial neurons. Students learn the structure of a neural network—layers, neurons, weights, biases, and the concept of forward propagation. The idea of linear combinations followed by activation functions is illustrated. They create a basic neural network manually in Python (without libraries) to classify logical operations (AND, OR, XOR) and later transition to frameworks like TensorFlow or Keras.
   * Activation Functions: In-depth analysis of Sigmoid, ReLU, Tanh, and Softmax functions and their importance in backpropagation.
   * Backpropagation and Optimization: This section explains how neural networks learn. Students trace the backward flow of error through the network and update weights using gradients. The loss functions covered include Mean Squared Error and Cross Entropy. Optimizers like Stochastic Gradient Descent (SGD), RMSprop, and Adam are introduced. Students run experiments comparing the performance of these optimizers on simple datasets. Learning rates, batch sizes, and epochs are discussed in relation to convergence speed and accuracy.

### Week 4: Introduction to IoT & Internship Project Planning

This module introduces students to the ecosystem of IoT, its components, communication methods, and practical applications across various industries. The focus is on hands-on experimentation and system-level thinking.

1. IoT Fundamentals:
   * Architecture Overview: Students begin by understanding the layered architecture of IoT systems. This includes:
     + Sensors and Actuators: Devices that collect real-world data (e.g., temperature, humidity, motion) or trigger actions.
     + Microcontrollers & Boards: Hands-on sessions using Arduino Uno, ESP32, and Raspberry Pi. Students program microcontrollers to read sensor data and send output commands.
     + Gateways and Connectivity: Discussions on Wi-Fi, Bluetooth, LoRa, and Zigbee for short/longrange communication.
     + Cloud Platforms: Brief exposure to how data is pushed to the cloud (e.g., using ThingSpeak or Firebase) for storage and visualization.
   * Hands-on Sensor Projects:
     + Soil Moisture Sensor: Students assembled and programmed circuits to monitor soil moisture levels using analog sensors. Real-world use cases in smart irrigation were discussed.
     + Rainfall Detection Sensor: Using rain sensors and buzzers, students built basic weather monitoring systems that could send alerts when rain is detected.
   * Introduction to Basic Hardware Components:
     + LCD Displays (16x2): Used to display real-time sensor readings.
     + DC Motors & Servo Motors: Controlled using PWM for automation and robotics.
     + Buzzers: For alarm or notification systems.
     + Cameras: Raspberry Pi Camera Module for capturing image data in smart surveillance systems.
     + LEDs, Resistors, Breadboards, Jumpers: Used extensively for circuit building.
2. Internship Project Guidance – Phase 1:
   * Problem Selection: Students are guided to select real-world problems in domains like agriculture, health, or public safety. Mentors help evaluate feasibility, data availability, and scope.
   * Project Proposal: Teams draft proposals including problem statement, proposed solution, technologies involved, dataset (source or self-collected), and expected outcome.
   * Milestone Planning: Projects are broken into 3–4 milestones with deadlines. This planning includes phases like data collection, modeling, validation, and deployment.

### Week 5: Deep Learning & Introduction to SAP ABAP

Week 5 builds on neural networks and introduces students to image-based machine learning and enterprise software development.

1. Deep Learning (CNNs):
   * Convolutional Neural Networks: Students explore convolution layers, kernel sizes, stride, padding, and pooling layers.
   * Backpropagation in CNNs: Gradient flow across convolutional layers is demonstrated.
   * Hands-On: Image classification using datasets like CIFAR-10 or medical imaging datasets.

Projects like fruit recognition or sign language detection are explored.

1. Computer Vision Introduction:
   * Students use OpenCV to perform operations like image rotation, blurring, grayscale conversion, and edge detection.
   * Techniques such as contour detection, face detection using Haar Cascades, and object tracking are implemented.
2. SAP ABAP Introduction:
   * SAP Business Technology Platform (BTP): Introduction to enterprise architecture, cloud integrations, and business workflows.
   * ABAP Syntax: Writing simple programs for data entry, conditional logic, loops, and reports.
   * Hands-On Labs: Creating tables, writing basic SELECT queries, using ALV reports.
3. Internship Project Guidance – Phase 2:
   * Data Preparation: Students clean and preprocess data using Pandas, NumPy, or OpenCV.
   * Model Building: Suitable ML/DL models are implemented. Teams use real-time dashboards or mock-ups to demonstrate interfaces.
   * Prototype: Teams build a working version of their solution, integrate early features, and prepare for testing.

### Week 6: Computer Vision & SAP ABAP

The final week is focused on polishing skills, refining internship projects, and preparing for final presentations.

1. Advanced Computer Vision:
   * Real-time Applications: Face recognition, motion detection using webcams, and gesture recognition projects.
   * Image Segmentation: Basic introduction to semantic segmentation using U-Net or thresholding.
   * Deployment Readiness: Students explore saving models, exporting with TensorFlow Lite, and using them on edge devices like Raspberry Pi.
2. SAP ABAP Final Module:
   * Complex Data Handling: Fetching data from multiple tables, internal tables manipulation, using function modules.
   * Creating Interactive Reports: Students design ALV reports that allow filters and sorting.
3. Final Internship Project Phase:
   * Testing & Iteration: Teams test their prototypes with real or simulated input data.
   * Documentation: Teams prepare technical documentation, user guides, and a summary report.
   * Final Presentations: Each team presents their project to a panel of faculty, peers, and industry mentors. Presentations are evaluated on innovation, technical depth, usability, and clarity.
   * Feedback & Reflection: Individual reflections, feedback from mentors, and guidance on how to continue learning and improving are shared. Certificates and rewards are distributed.

# Project Overview

PROJECT TITLE: CURRENCY DETECTION USING TENSORFLOW LITE

## Problem Statement

Manual identification of currency by the visually impaired or in financial automation systems can lead to errors. There is a need for a fast, lightweight AI solution to detect Indian currency denominations.

## Objectives

The primary objectives of this project are:

* Build a model to recognize Rs. 10 to Rs. 500 currency notes
* Deploy it using TensorFlow Lite for mobile usability
* Provide accurate real-time predictions

## Scope

The scope and anticipated results of the project include:

* Works with Indian currency only
* Detects and labels denominations using the camera
* Basic Android interface for demonstration

## Expected Outcomes

The expected outcomes are as follows :

* Lightweight model (.tflite format)
* On-device prediction without cloud support
* Functional prototype for further enhancement

# Challenges Encountered During the Internship

During the internship, several technical and functional challenges were encountered while developing the food delivery application. These challenges served as important learning opportunities and strengthened practical development skills.

## Limited Dataset Availability

* No pre-existing public dataset available for Indian currency notes.
* Manually collected a diverse set of images for denominations from Rs. 10 to Rs. 500.
* Required careful planning to cover different note conditions, backgrounds, and orientations.

## Variation in Lighting and Backgrounds

* Model accuracy decreased significantly in low-light or overexposed conditions.
* Shadows, glare, and inconsistent backgrounds made real-time detection less reliable.

## Torn and Crumbled Notes

* Notes that were torn, crumpled, folded, or worn out often caused incorrect predictions
* Deformities affected how the model extracted features, leading to lower confidence in the output.

## Accuracy vs. Real-World Performance

* Despite achieving 94% accuracy during training and validation, real-world performance varied.
* Slight changes in positioning, light, or condition of the note affected recognition accuracy.
* Reinforced the importance of real-world testing and the limitations of controlled datasets.

# Methodology Used

The development of the food delivery application (Zomato Clone) followed the Software Development Life Cycle (SDLC) with a focus on the Agile methodology. Agile enabled incremental progress, constant feedback, and iterative refinement of features.

## Data Collection & Preprocessing

1. Collected images of Indian currency notes ranging from Rs. 10 to Rs. 500.
2. Captured images under various conditions — different lighting, angles, and backgrounds — to enhance model robustness.
3. Organized the data into labeled folders for each denomination.
4. Applied basic data augmentation techniques such as:
   * Rotation
   * Brightness and contrast adjustments
   * Zoom and flip (as supported by Teachable Machine’s internal preprocessing)
5. Images were automatically resized and normalized by Teachable Machine during the model training process.

## Model Architecture

1. Used Google’s Teachable Machine platform, which builds upon pre-trained Convolutional Neural Networks (CNNs).
2. Teachable Machine handles the internal architecture, leveraging transfer learning with an efficient backbone model (typically MobileNet or similar).
3. Model layers and parameters were managed by the platform to ensure quick training and deployment readiness.
4. Classification was set for multi-class image classification with six output classes (Rs. 10, Rs. 20, Rs. 50, Rs. 100, Rs. 200, Rs. 500).
5. Optimization used categorical cross-entropy loss with built-in performance metrics.

## Training and Evaluation

1. The dataset was split automatically by Teachable Machine into training and validation sets (approximately 80%-20%).
2. Achieved a validation accuracy of approximately 94%, based on real-time training statistics
3. Evaluated model performance by downloading and testing with a custom test set, focusing on:
   * Precision & recall
   * Real-world test images not seen during training
   * Observations of misclassification under low-light or partially occluded conditions

# Technology Stack & Implementation

## Technology Stack

The currency detection project made use of a variety of technologies from machine learning model creation to Android-based deployment. The model was built using a user-friendly platform (Teachable Machine) and later integrated into a native Android application. The core image classification was handled using TensorFlow Lite, a lightweight version of TensorFlow designed for mobile and embedded devices.

The entire application was developed in Java using Android Studio, leveraging the TensorFlow Lite interpreter to detect Indian currency denominations in real-time from live camera input.

|  |  |
| --- | --- |
| Category | Tools/Technologies Used |
| Model Training | Teachable Machine |
| Machine Learning | TensorFlow Lite |
| Programming Language | Java |
| Mobile Development | Android Studio |
| Image Handling | TensorFlow Lite APIs for Android |
| Deployment Platform | Android (tested on emulator and real device) |
| Utilities | TFLite Model Interpreter, Android CameraX API |

Table 1: Technology Stack Used in the Project

## Features Implemented

The Currency Detection app was developed to provide a simple, fast, and accessible solution for identifying Indian currency denominations using machine learning. Several essential and user-friendly features were integrated into the application to improve its usability and functionality.

1. Currency Denomination Detection: The app accurately detects Indian currency notes of denominations Rs. 10, Rs. 20, Rs. 50, Rs. 100, Rs. 200, and Rs. 500 using a trained TensorFlow Lite model.
2. Real-Time Image Classification: The mobile camera is used to scan notes live, with classification results displayed instantly on-screen without any delay.
3. Offline Functionality: The app functions entirely offline, with all processing and predictions happening locally using the embedded .tflite model.
4. Voice Feedback Feature:An integrated Text-to-Speech (TTS) system speaks out loud the predicted denomination, enhancing accessibility—especially useful for visually impaired users.
5. Lightweight and Fast: Built for mobile environments, the app is responsive and optimized to run efficiently even on low-end Android devices.
6. User-Friendly Interface: The app’s interface is clean, minimalistic, and intuitive, making it easy to use for all age groups.
7. Live Dynamic Output Display: As the user points the camera at a note, the predicted denomination updates in real-time and is displayed prominently on the screen.

## Code



Figure 1: MainActivity.java (1)

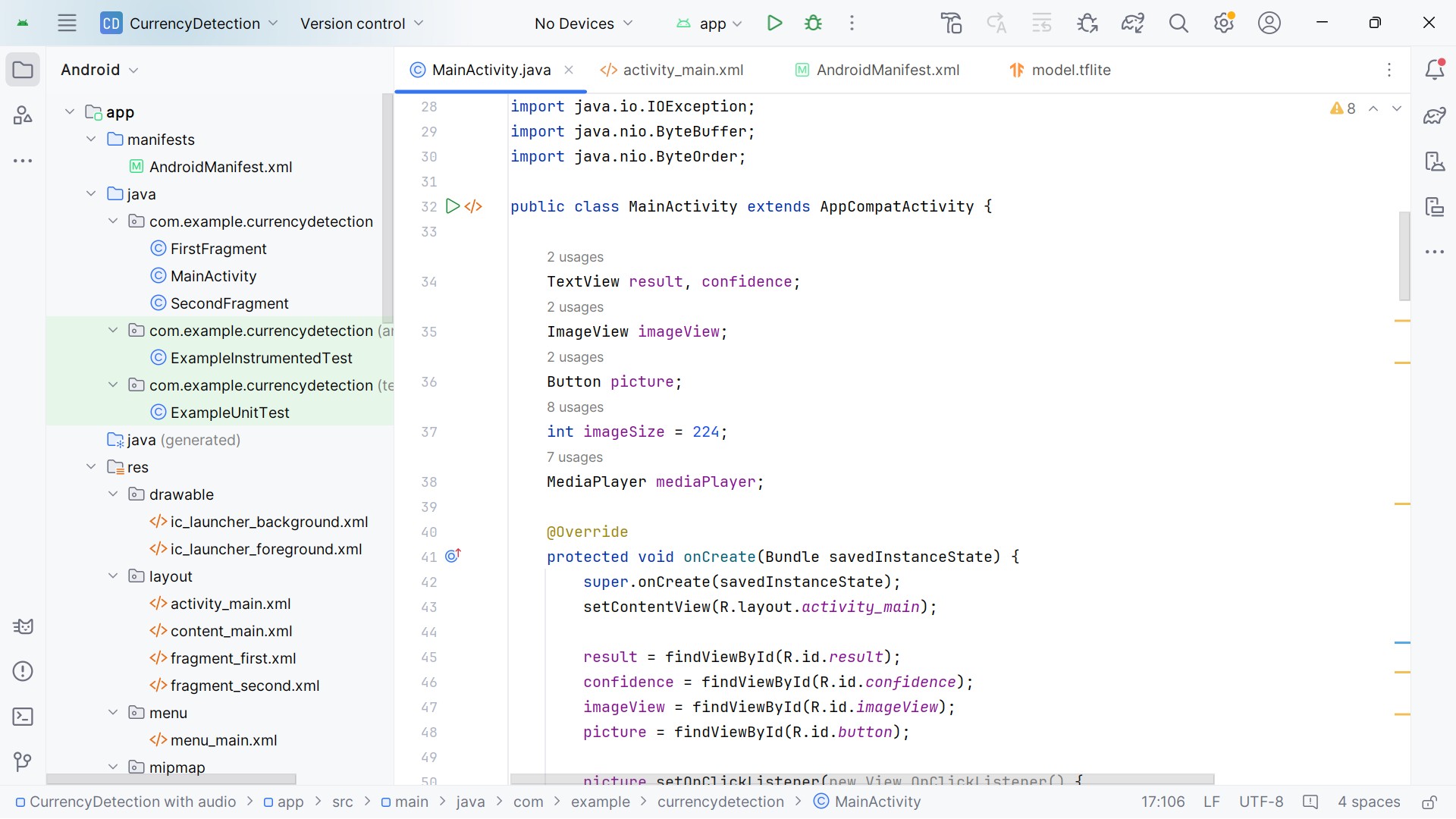


Figure 2: MainActivity.java (2)

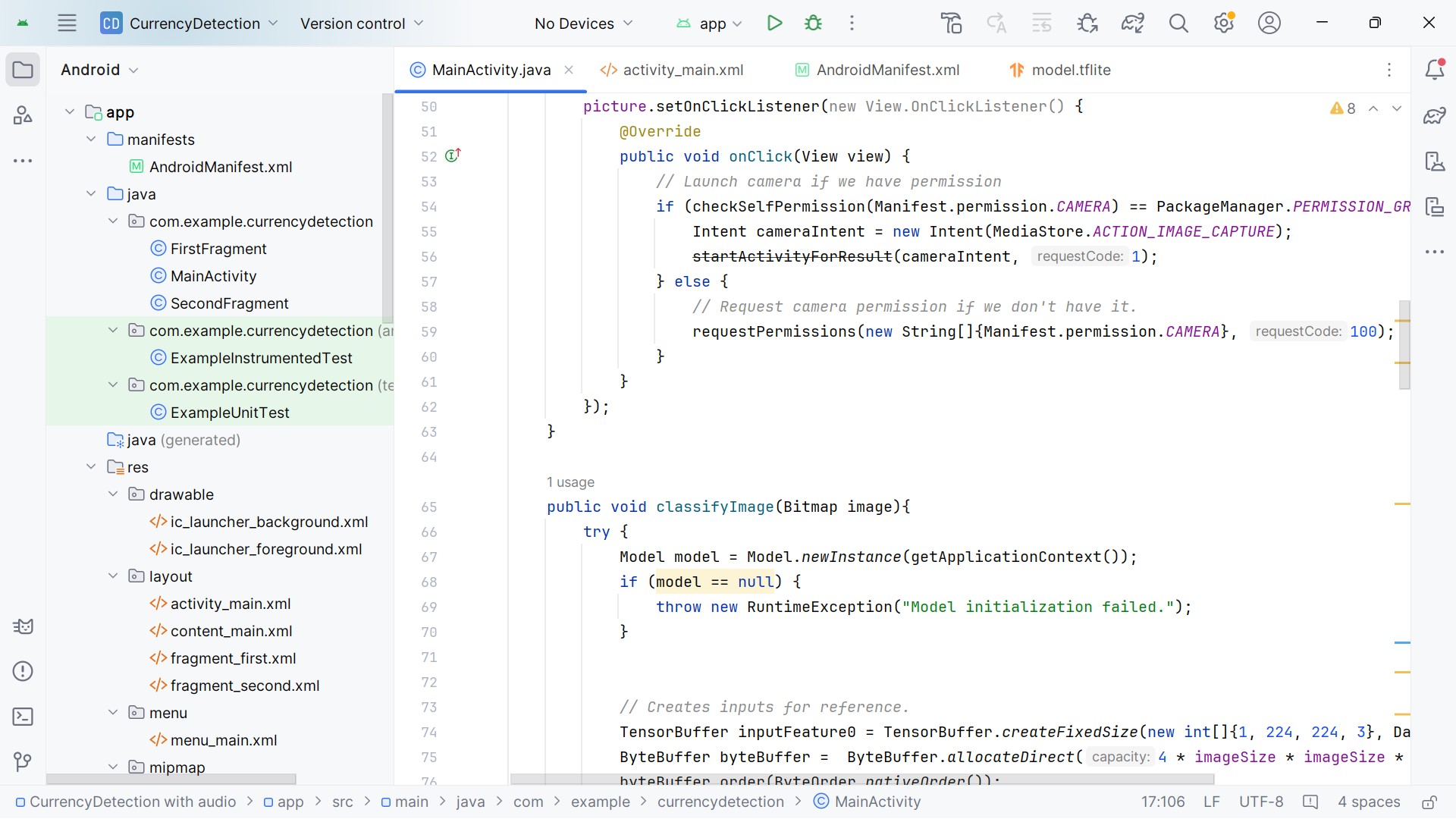


Figure 3: MainActivity.java (3)

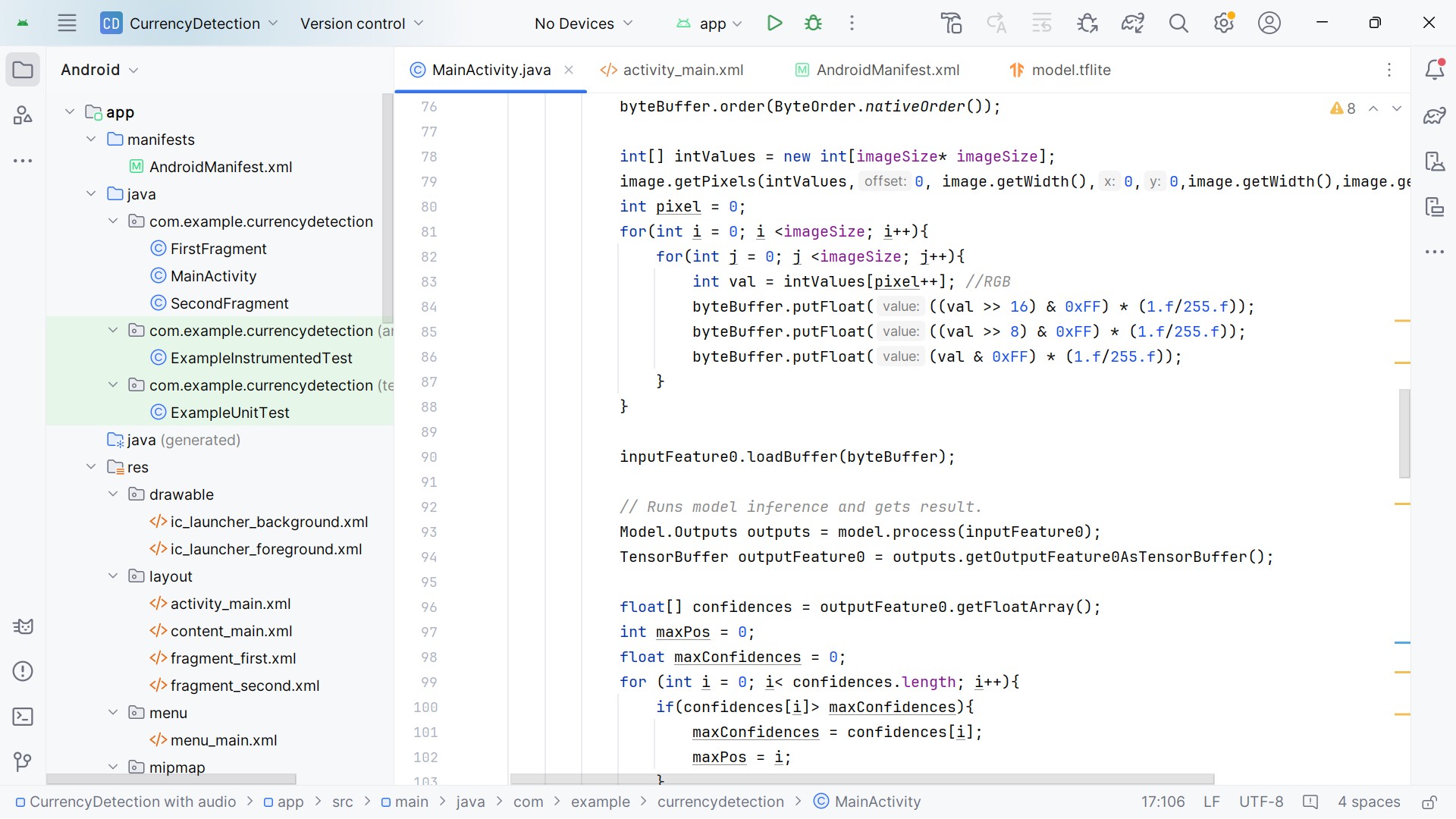


Figure 4: MainActivity.java (4)

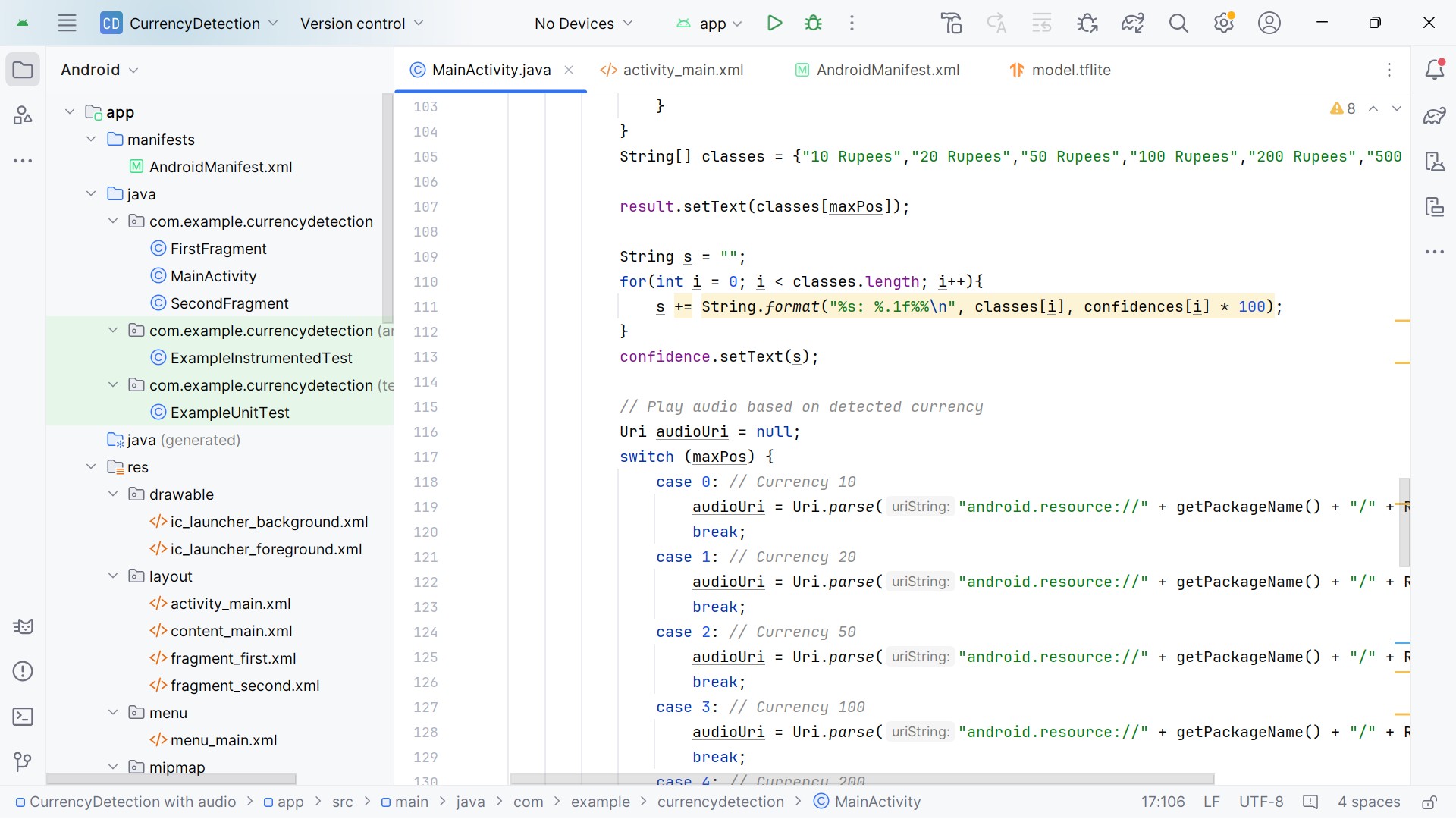


Figure 5: MainActivity.java (5)

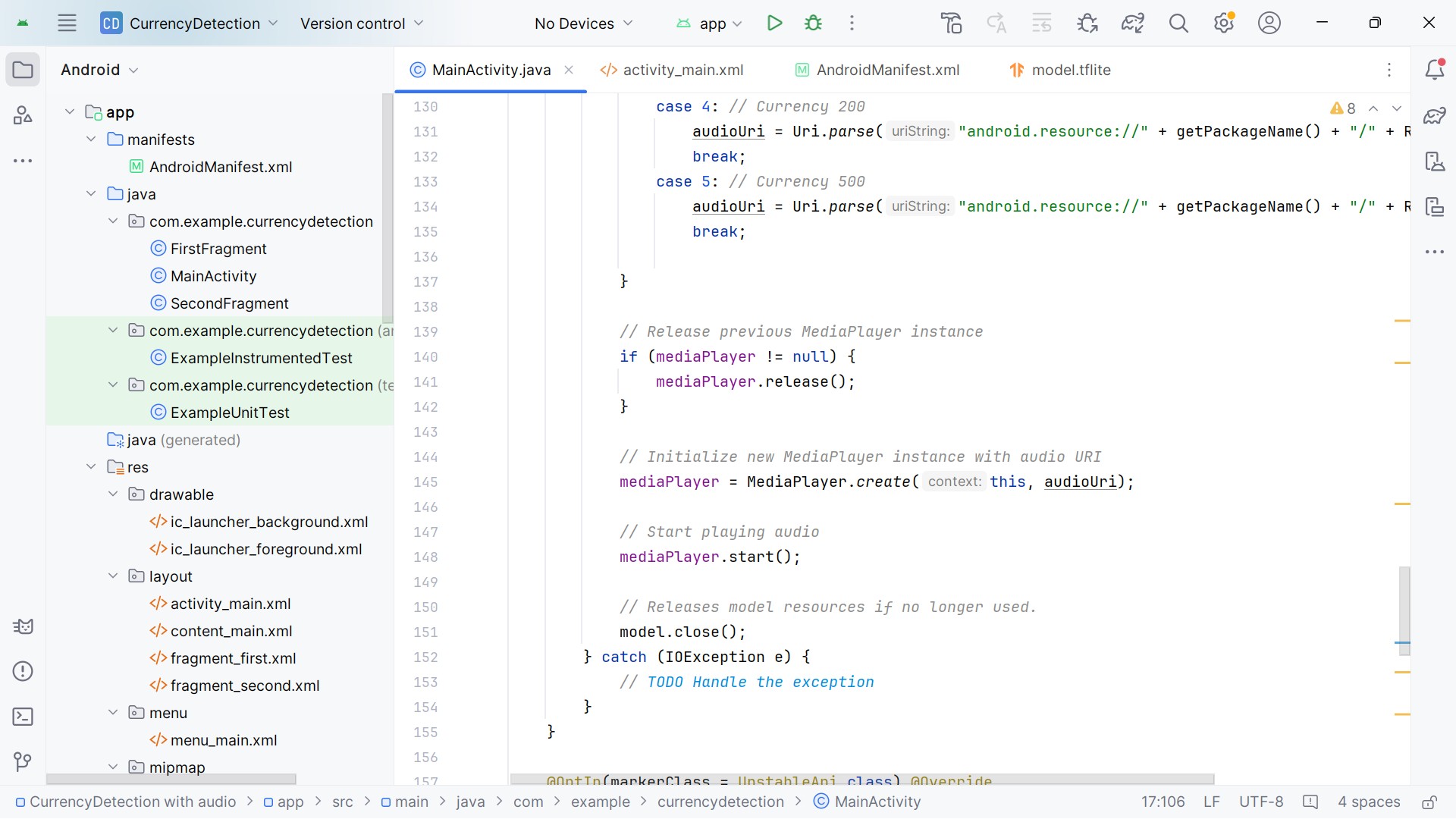


Figure 6: MainActivity.java (6)

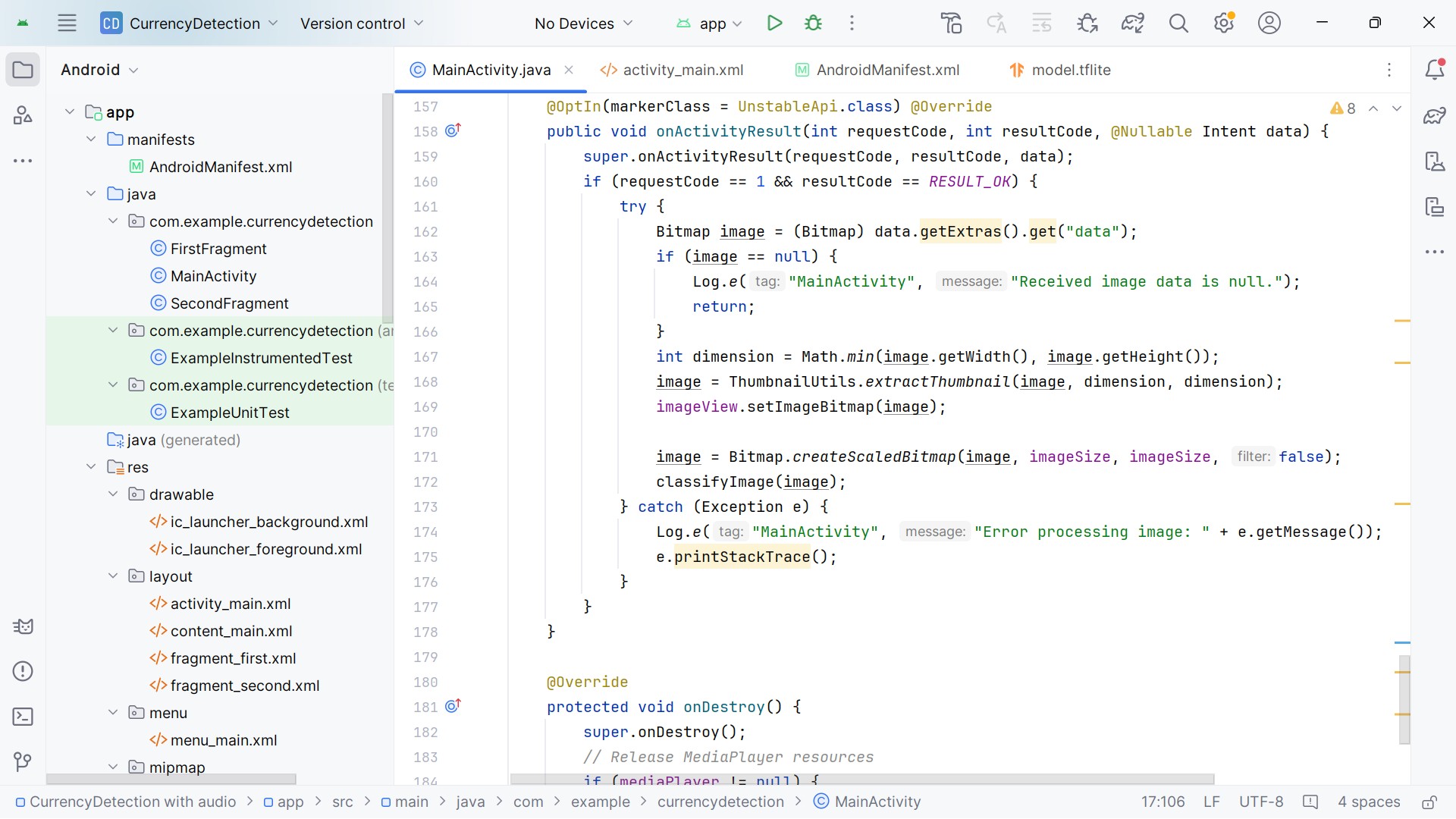


Figure 7: MainActivity.java (7)

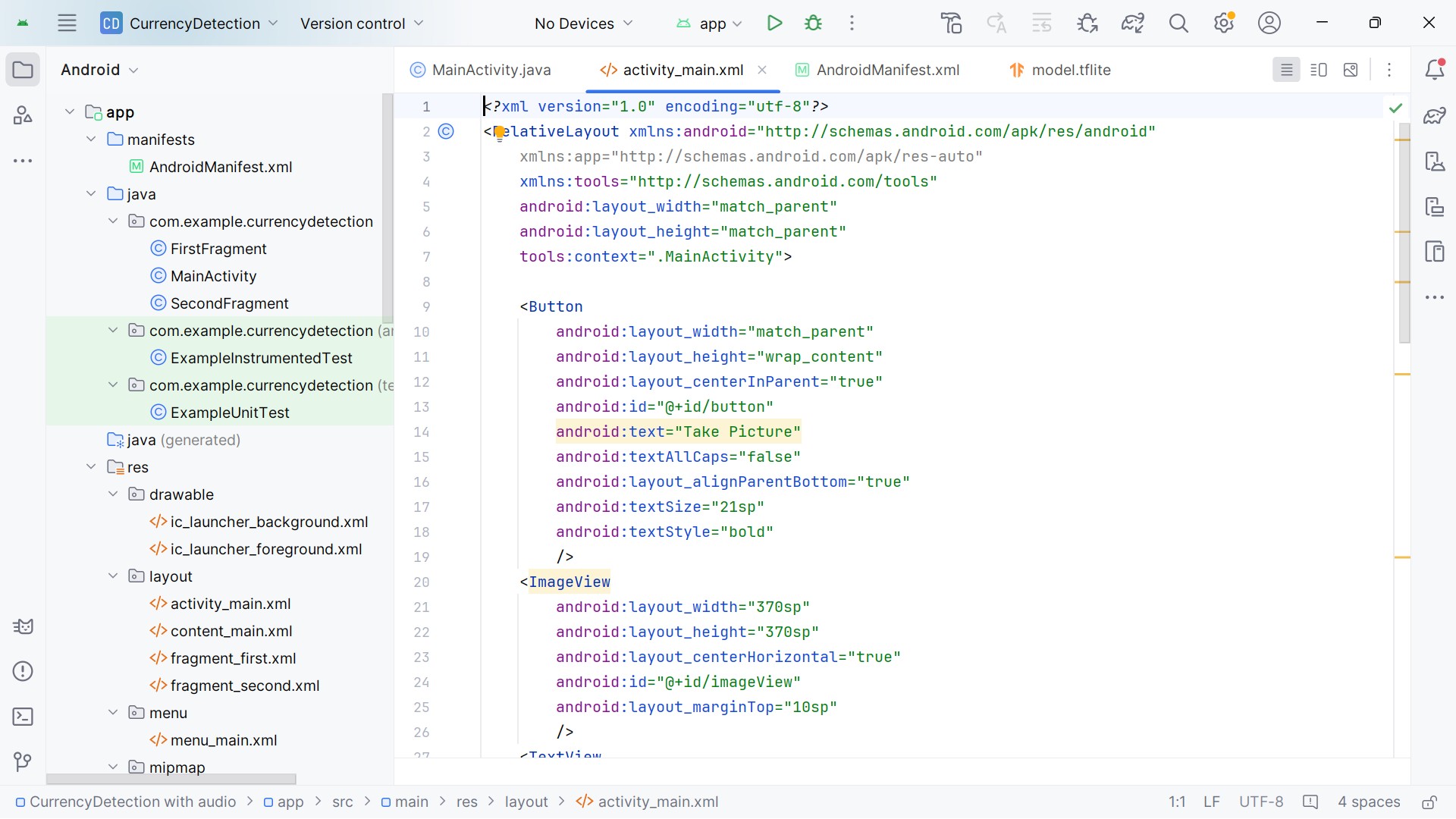


Figure 8: activity-main.xml (1)

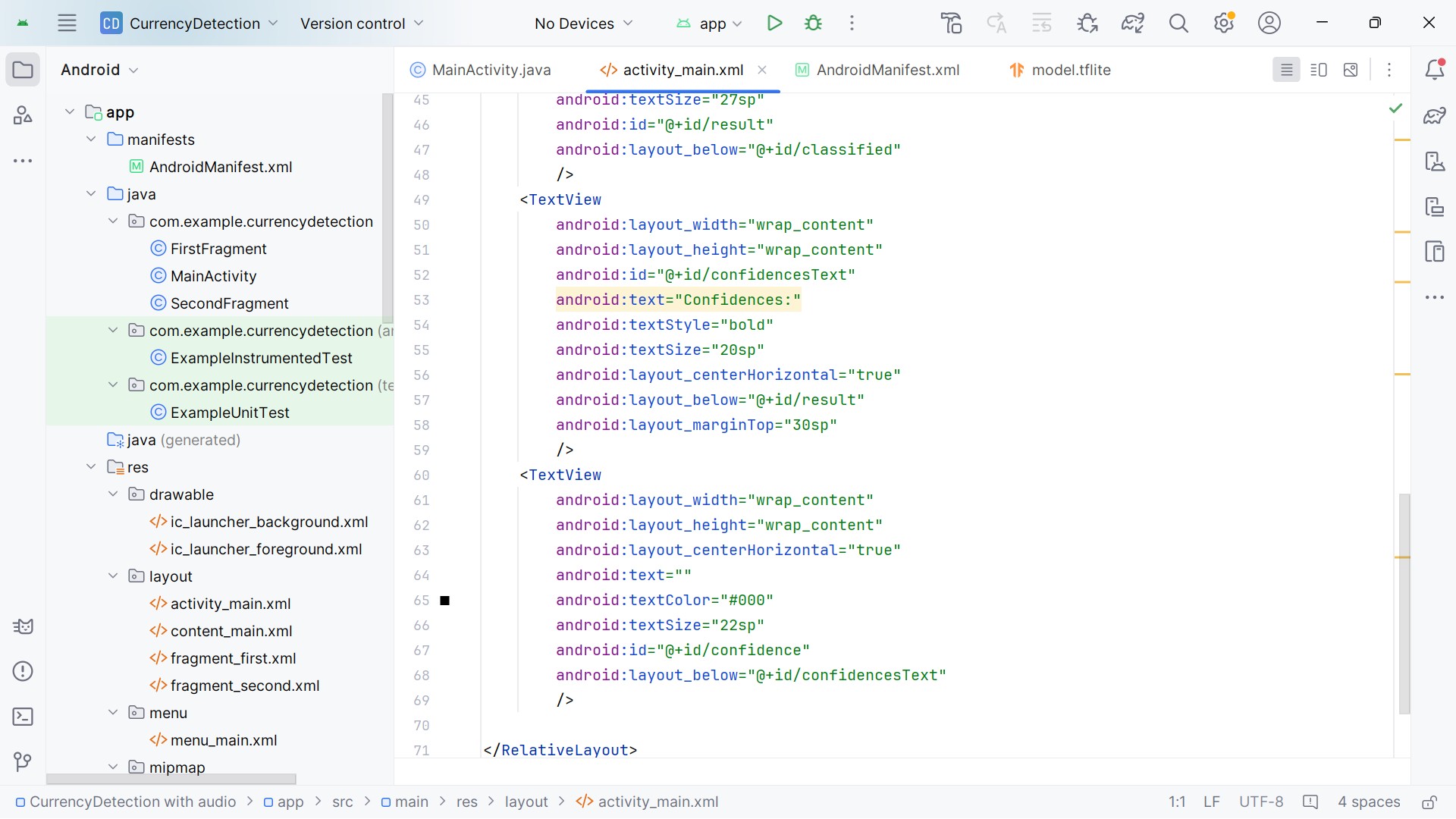


Figure 9: activity-main.xml (1)

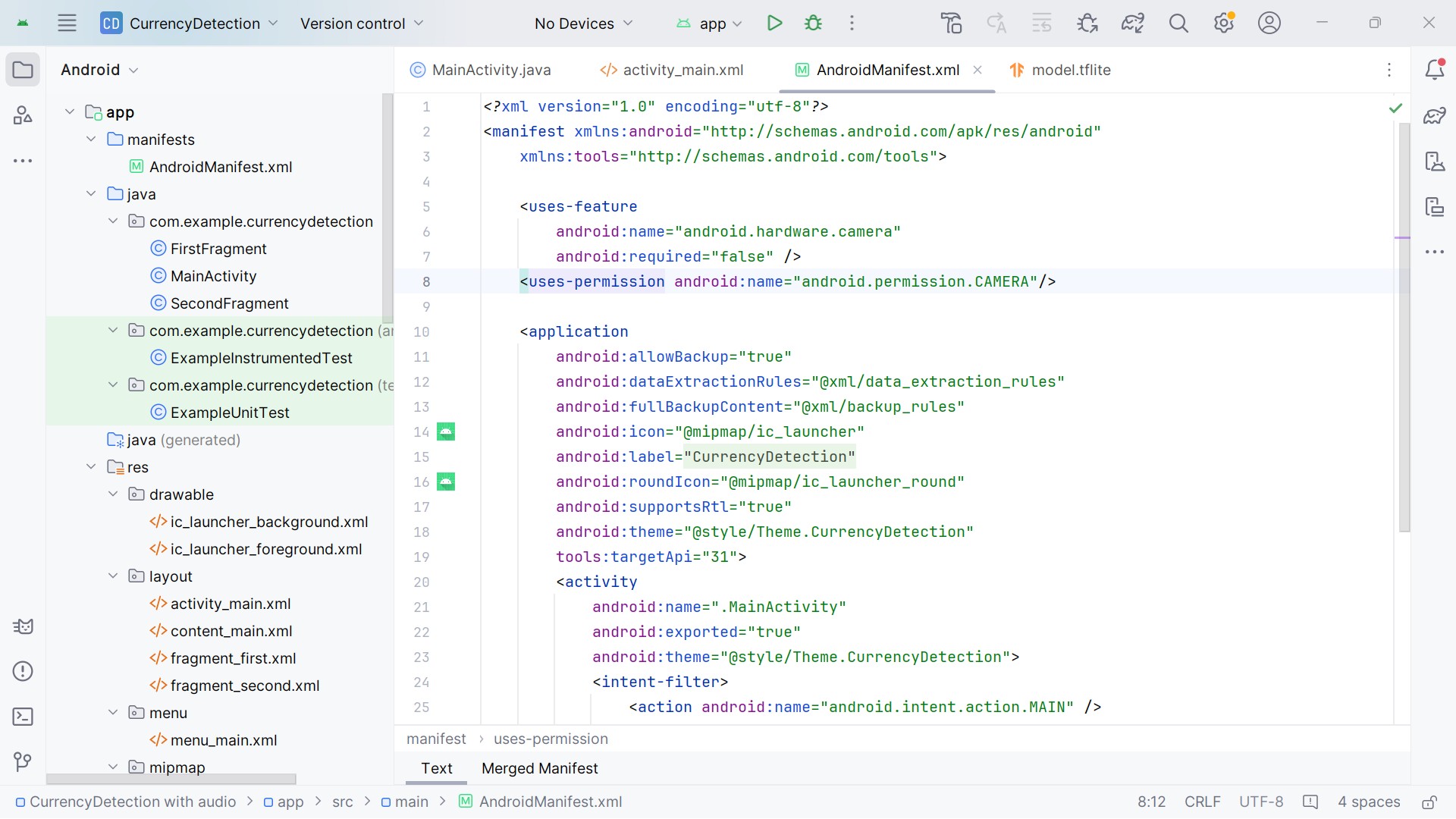


Figure 10: AndroidManifest.xml (1)

# Analysis of Results

The Indian Currency Detection app was tested across multiple scenarios to evaluate the accuracy, responsiveness, and robustness of the trained model. The primary goal was to ensure consistent performance in real-world conditions.

## Accuracy of Model

* The TensorFlow Lite model achieved a validation accuracy of approximately 94% during training on the Teachable Machine platform.
* The model performed well in classifying Rs. 10 to Rs. 500 denomination notes when tested in a controlled environment with good lighting.
* Confusion Matrix and manual testing revealed that most predictions were accurate, with very few misclassifications.

## Real-World Testing

* The app successfully identified currency notes using the live mobile camera feed in various conditions.
* Correct predictions were consistently observed for clean and well-lit currency images.
* Minor inaccuracies occurred in scenarios where:
  + Notes were torn, folded, or crumpled
  + Backgrounds were cluttered or lighting was dim or uneven
  + Notes were partially visible or captured at odd angles

## Voice Feedback Effectiveness

* The integrated Text-to-Speech feature accurately pronounced the detected currency denomination.
* Voice feedback added significant value for accessibility, especially for visually challenged users.

## Performance

* The model was lightweight and fast, with real-time prediction speed even on mid-range Android phones.
* The application ran smoothly without noticeable lag or delay in classification.

## Result



Figure 1: Homescreen

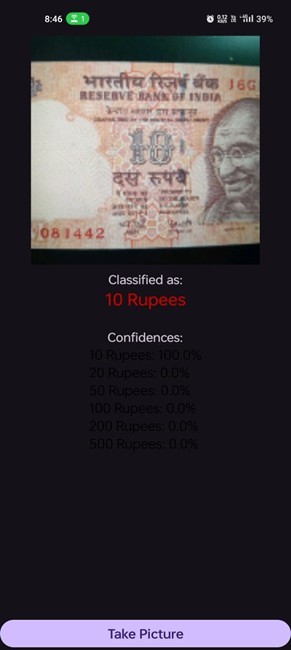
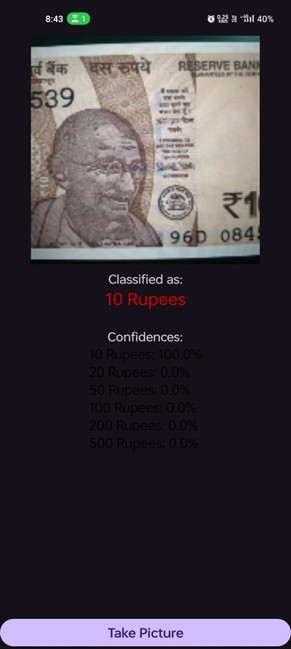


Figure 2: Detection of 10 rupees note both older and newer version of notes

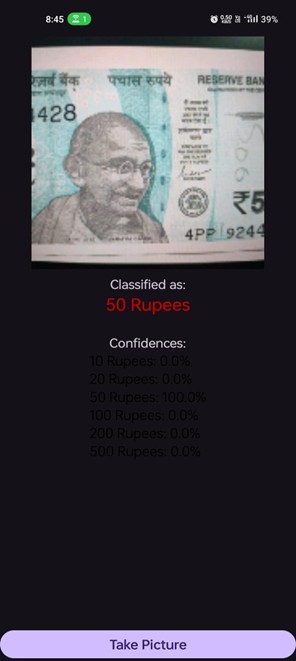
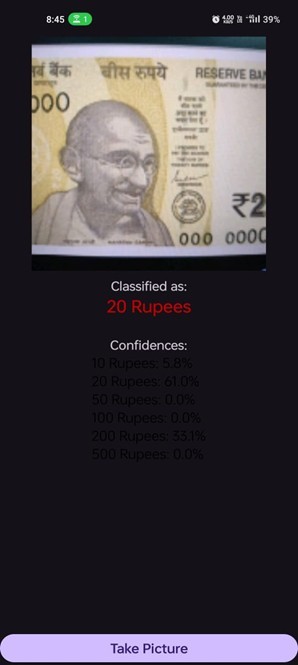


Figure 3: Detection of 20 & 50 rupees note

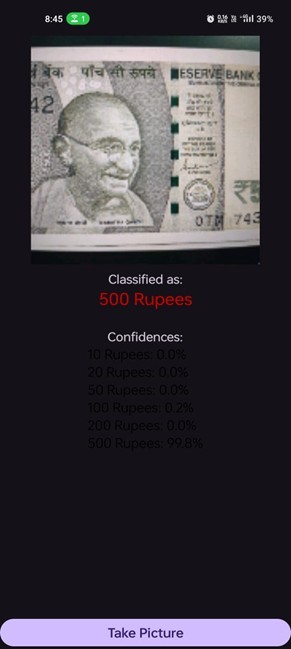
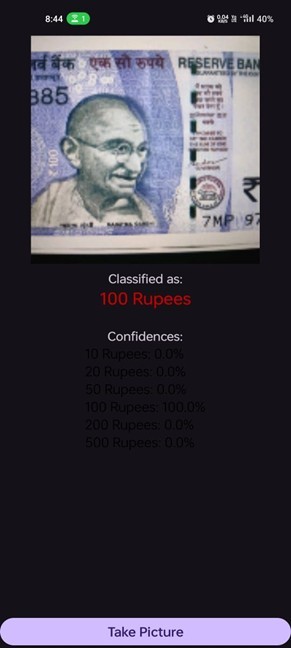


Figure 4: Detection of 100 & 500 rupees note both older and newer version of notes

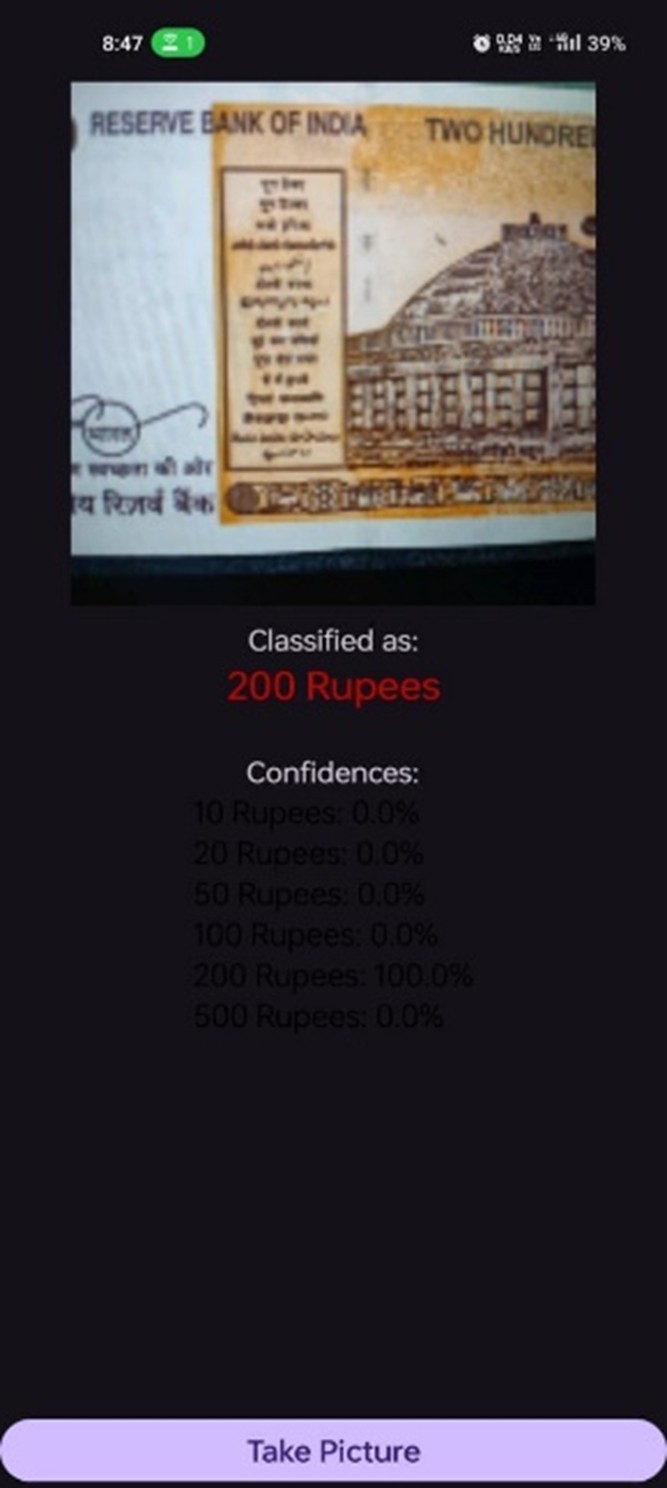
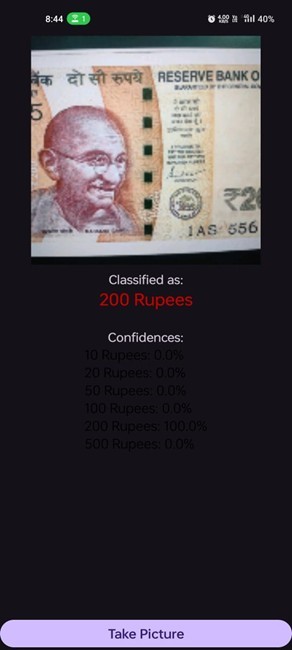


Figure 5: Detection of 200 rupees note both front and back part of the note

# Conclusion

The Code Unnati Program offered an enriching experience by providing in-depth exposure to cuttingedge technologies such as Machine Learning, Computer Vision, and Mobile App Development. Through the development of the project titled “Currency Detection Using TensorFlow Lite,” I was able to successfully integrate image classification along with a voice feedback feature, making the application both user-friendly and accessible. This internship gave me valuable hands-on experience with tools like TensorFlow Lite, Android Studio, and Java, significantly enhancing my technical proficiency in mobile application development and machine learning model deployment. The project enabled real-time, offline currency recognition, reflecting its practicality and potential for real-world use, especially in regions with limited internet access.

While working on the app, I encountered several challenges such as dataset limitations, inconsistent lighting conditions, and reduced accuracy with damaged or folded notes. Overcoming these issues not only improved the robustness of the model but also helped me strengthen my debugging and problemsolving abilities.

Overall, the internship bridged the gap between theoretical concepts and their practical applications, equipping me with the confidence and skills required to contribute meaningfully to future roles in AI and mobile development.

# Future Scope

1. Improved Model Accuracy: Enhance the model’s ability to accurately classify crumpled, torn, or partially visible notes by expanding the dataset and incorporating data augmentation techniques.
2. Expanded Currency Support: Include additional currency denominations or even support for international currency recognition to make the app globally applicable.
3. Integration with Payment Systems: Integrate the app with digital payment platforms for seamless currency verification during online transactions or ATMs.
4. Voice and Multi-language Support: Add support for multiple languages and voice feedback, catering to a wider audience, especially in multilingual regions.
5. Online Learning Feature: Enable the app to continuously learn and improve its predictions by collecting new data from user inputs and updating the model periodically.
6. Enhanced User Interface (UI): Improve the UI to support dark mode, larger font sizes, and better accessibility features for users with special needs.
7. Real-time Performance Optimization: Further optimize the app’s performance to work seamlessly on a broader range of devices, including entry-level smartphones, by reducing memory usage and improving inference speed.
8. Integration with IoT Devices: Explore the potential to use IoT devices like smart wallets or ATMs for real-time currency detection and validation, expanding its use case beyond smartphones.

References

1. SAP Edunet Foundation [https://codeunnati.edunetfoundation.com/.](https://codeunnati.edunetfoundation.com/)
2. Teachable Machine [https://teachablemachine.withgoogle.com/.](https://teachablemachine.withgoogle.com/)
3. Android Developers (TensorFlow Lite on Android) [https://www.tensorflow.org/lite/guide/android.](https://www.tensorflow.org/lite/guide/android)
4. Android Studio [https://developer.android.com/studio.](https://developer.android.com/studio)
5. Jupyter Notebooks [https://jupyter.org/.](https://jupyter.org/)
6. Google [https://google.com/.](https://google.com/)
7. GitHub [https://github.com/.](https://github.com/)
8. W3Schools [https://Youtube.com/.](https://Youtube.com/)
9. GeeksforGeeks [https://www.geeksforgeeks.org/.](https://www.geeksforgeeks.org/)
10. OpenCV [https://opencv.org/.](https://opencv.org/)
11. App Permissions in Android [https://developer.android.com/guide/topics/permissions/overview.](https://developer.android.com/guide/topics/permissions/overview)
12. Image Classification with TensorFlow Lite [https://www.tensorflow.org/lite/models/image\_classification/overview.](https://www.tensorflow.org/lite/models/image_classification/overview)