PAYWELL

Be well Pay Well

By: Aspen Kim, Chanmin Kwon, Christopher Seo

Development Overview

01

User Interface

An app design mockup

02

Payment Plan

Suggested payment plan according to previously imputed data

03

Charity Matching

Program using random forest classifier

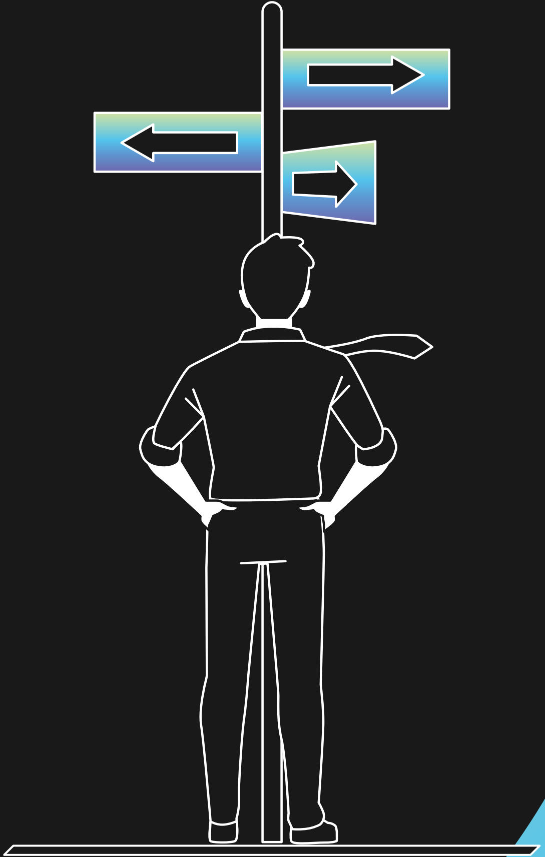
04

Coding

Program in detail

THE PROBLEM

- **41 million** Americans struggle with medical debt
- Many are unaware of hospital charity programs or other financial assistance options
- Studies show that **less than 20%** of eligible individuals take advantage of hospital charity programs.



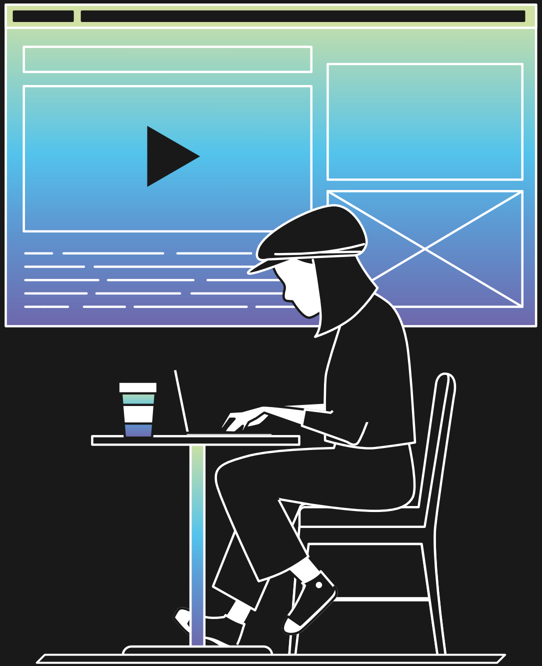
OUR APPLICATION

- Connects individuals burdened by medical debt to hospital charities and financial assistance resources.
- Empower users to regain financial stability through an intuitive and accessible platform

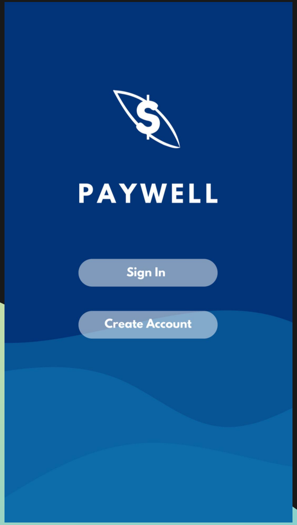
01

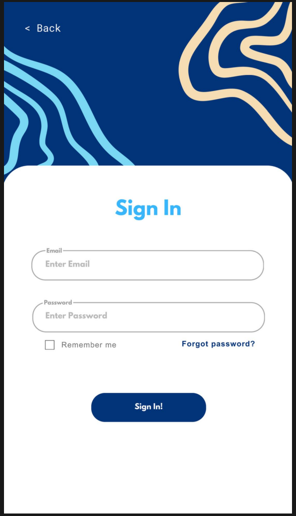
USER INTERFACE

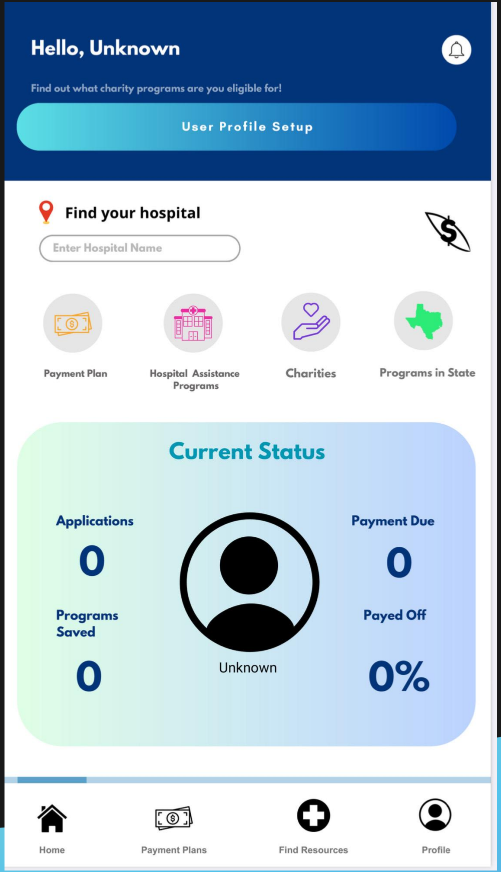
Application Mockup

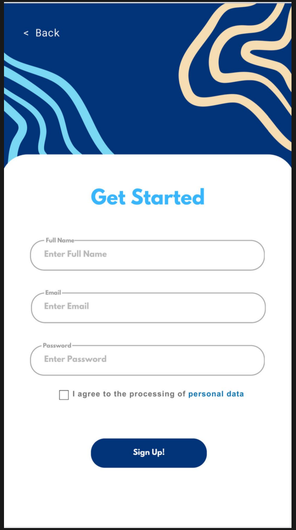


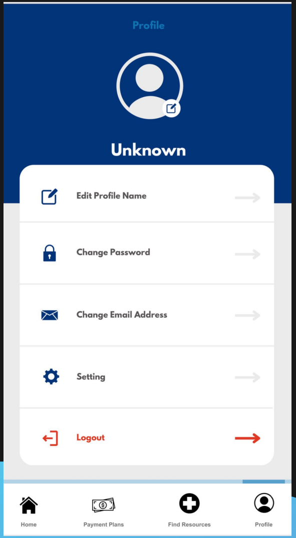
HOME SCREEN


PAYWELL
Sign In
Create Account


< Back
Sign In
Email
Enter Email
Password
Enter Password
Remember me
Forgot password?
Sign In!


Hello, Unknown
Find out what charity programs are you eligible for!
User Profile Setup
Find your hospital
Enter Hospital Name
Payment Plan
Hospital Assistance Programs
Charities
Programs in State
Current Status
Applications
0
Payment Due
0
Programs Saved
0
Payed Off
0%
Unknown
Home
Payment Plans
Find Resources
Profile


< Back
Get Started
Full Name
Enter Full Name
Email
Enter Email
Password
Enter Password
I agree to the processing of personal data
Sign Up!


Profile
Unknown
Edit Profile Name
Change Password
Change Email Address
Setting
Logout
Home
Payment Plans
Find Resources
Profile

02

PAYMENT PLAN

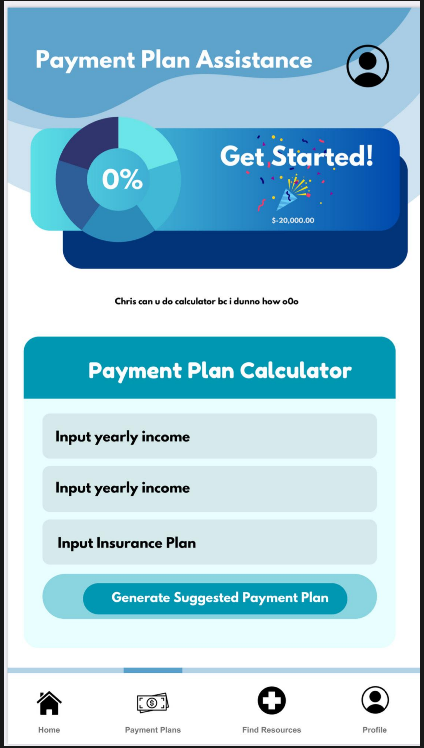
PAYMENT PLAN

Calculator

Input users data to output a calculated payment plan to best manage money and pay off debt

Suggested Plan

Overview of compatibility with suggested plans in order to optimize efficiency and recommends the best ones for user circumstance


Payment Plan Assistance

Get Started!

0%

$-20,000.00

Chris can u do calculator bc i dunno how o0o

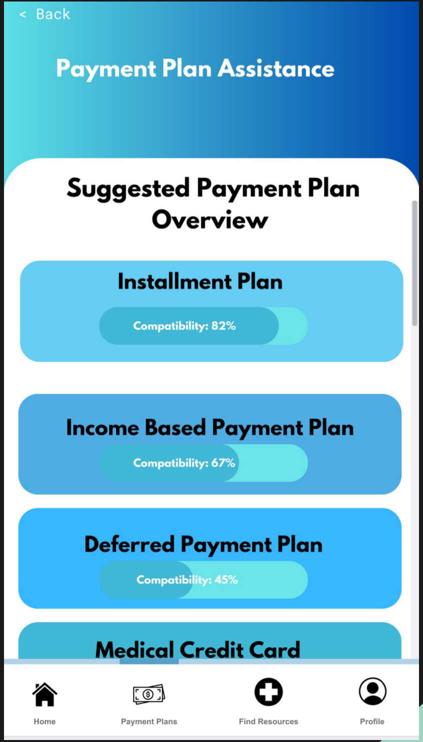
Payment Plan Calculator

Input yearly income

Input yearly income

Input Insurance Plan

Generate Suggested Payment Plan


< Back

Payment Plan Assistance

Suggested Payment Plan Overview

Installment Plan

Compatibility: 82%

Income Based Payment Plan

Compatibility: 67%

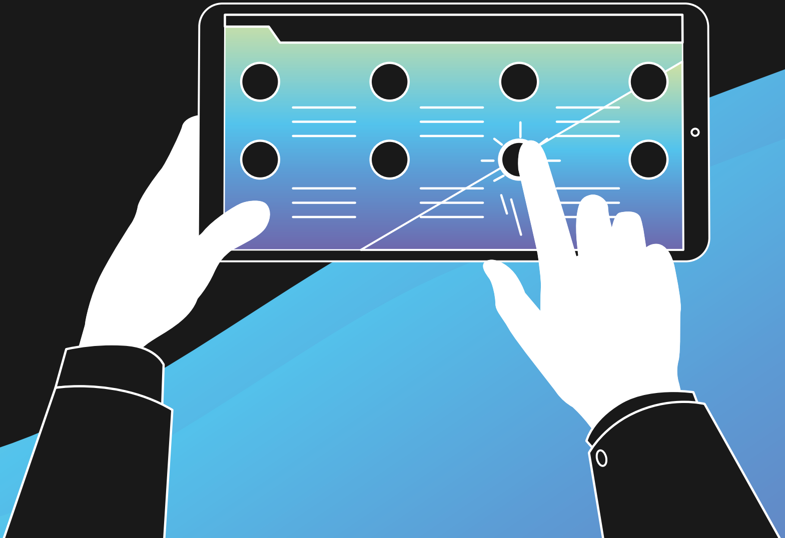
Deferred Payment Plan

Compatibility: 45%

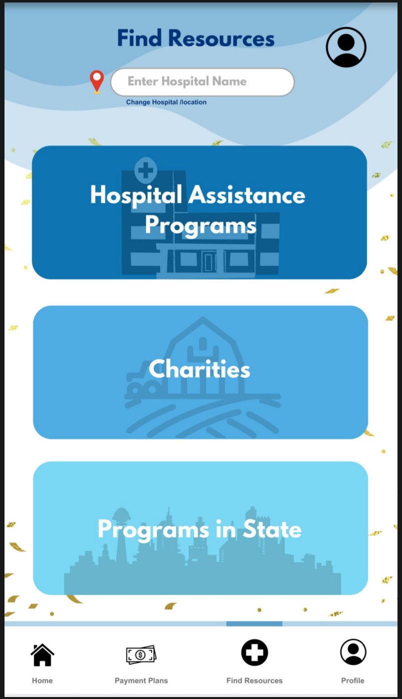
Medical Credit Card

03

CHARITY MATCHING



FIND RESOURCES



Find Resources

Enter Hospital Name

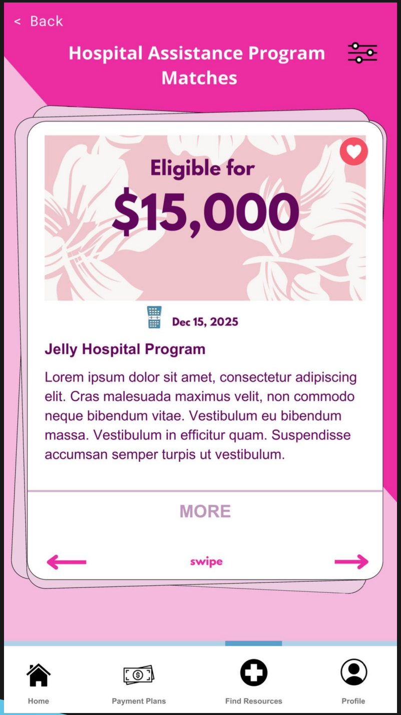
Change Hospital /location

Hospital Assistance Programs

Charities

Programs in State

Home Payment Plans Find Resources Profile



< Back

Hospital Assistance Program Matches

Eligible for

$15,000

Dec 15, 2025

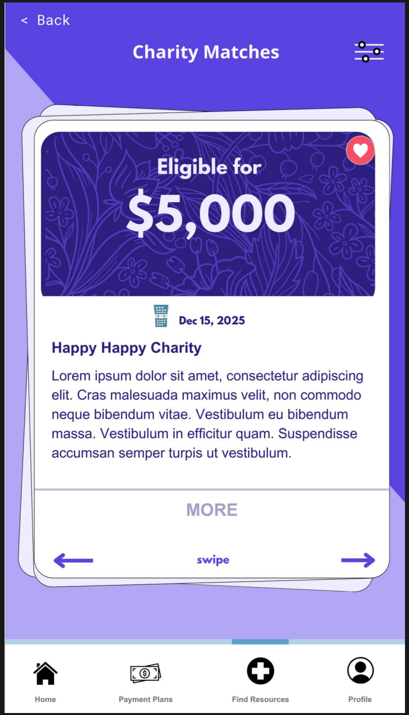
Jelly Hospital Program

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras malesuada maximus velit, non commodo neque bibendum vitae. Vestibulum eu bibendum massa. Vestibulum in efficitur quam. Suspendisse accumsan semper turpis ut vestibulum.

MORE

swipe

Home Payment Plans Find Resources Profile



< Back

Charity Matches

Eligible for

$5,000

Dec 15, 2025

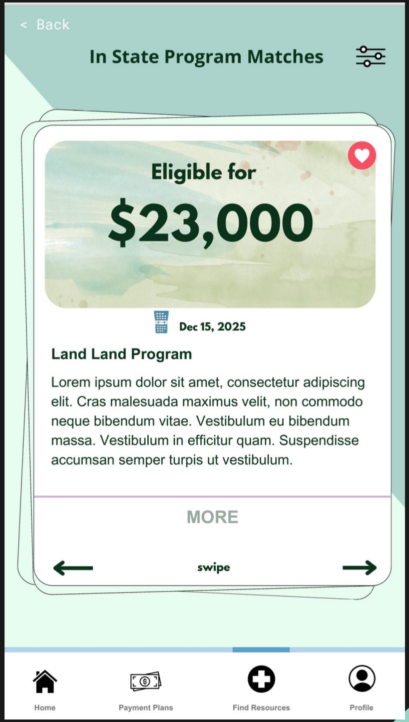
Happy Happy Charity

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras malesuada maximus velit, non commodo neque bibendum vitae. Vestibulum eu bibendum massa. Vestibulum in efficitur quam. Suspendisse accumsan semper turpis ut vestibulum.

MORE

swipe

Home Payment Plans Find Resources Profile



< Back

In State Program Matches

Eligible for

$23,000

Dec 15, 2025

Land Land Program

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras malesuada maximus velit, non commodo neque bibendum vitae. Vestibulum eu bibendum massa. Vestibulum in efficitur quam. Suspendisse accumsan semper turpis ut vestibulum.

MORE

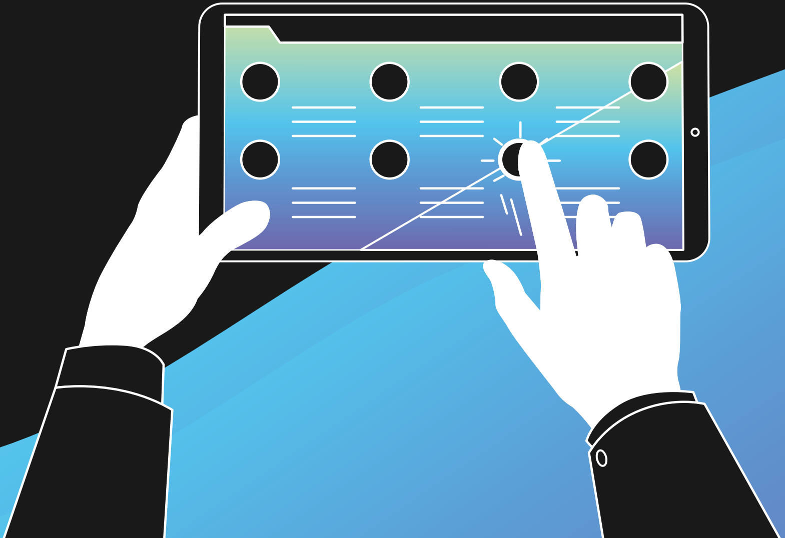
swipe

Home Payment Plans Find Resources Profile

From the find resources screen select which types of aid preferred and swipe right for more options and "like" to save the ones user is compatible with (the heart button in top right corner) Select more to see program in more detail

04

CODING



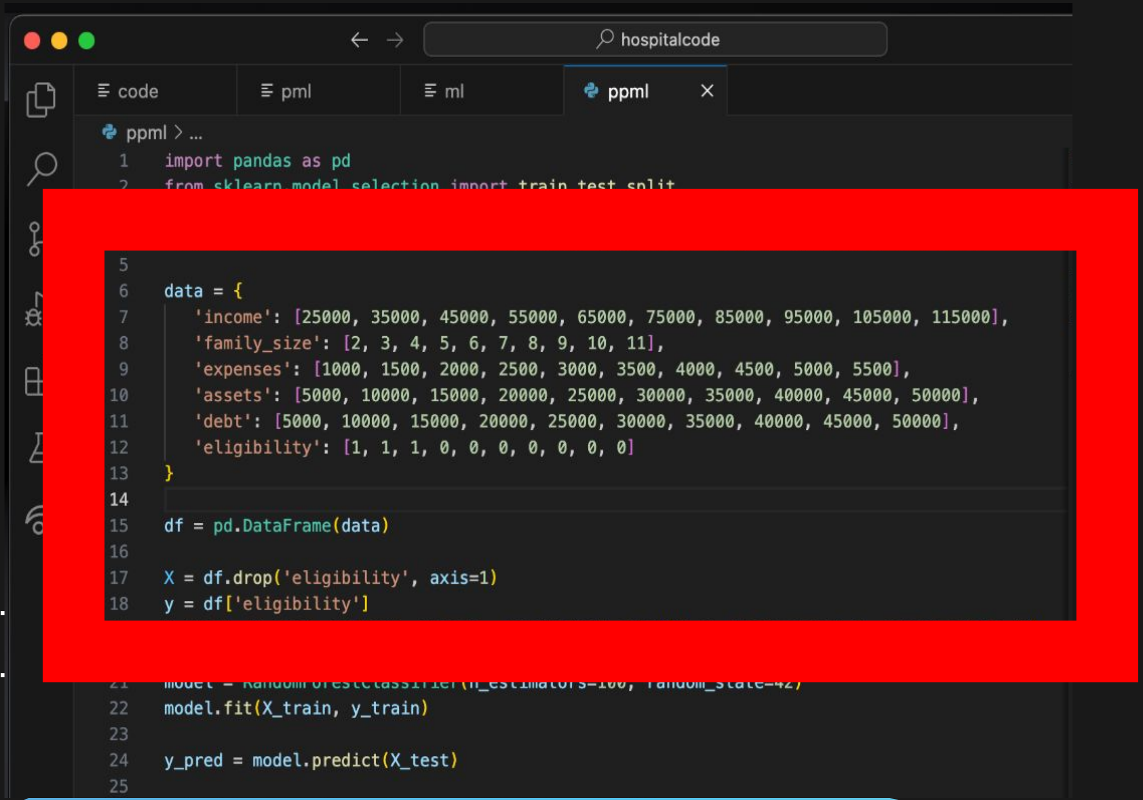
CODE BASE

Technologies: Flutter (frontend, cross-platform UI), Node.js/Django (backend, user data, API integrations), Firebase/MongoDB (database, scalable, secure storage).

Implementation Plan: Integrate APIs for charity programs and implement form validation and eligibility checks, develop secure authentication for user data protection.

Security Measures: Ensure HIPAA compliance for sensitive data, use encryption for data storage and transfer.

DATA PREPARATIONS

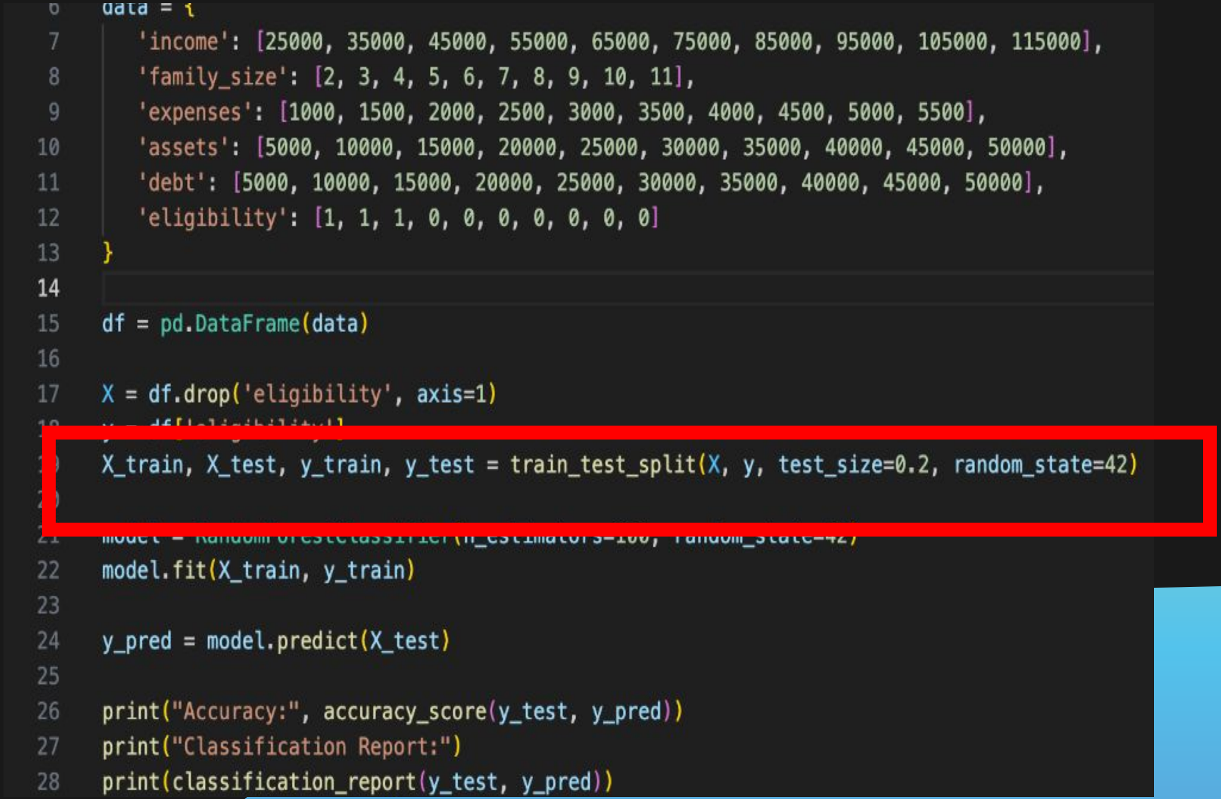


```
import pandas as pd
from sklearn.model_selection import train_test_split

data = {
    'income': [25000, 35000, 45000, 55000, 65000, 75000, 85000, 95000, 105000, 115000],
    'family_size': [2, 3, 4, 5, 6, 7, 8, 9, 10, 11],
    'expenses': [1000, 1500, 2000, 2500, 3000, 3500, 4000, 4500, 5000, 5500],
    'assets': [5000, 10000, 15000, 20000, 25000, 30000, 35000, 40000, 45000, 50000],
    'debt': [5000, 10000, 15000, 20000, 25000, 30000, 35000, 40000, 45000, 50000],
    'eligibility': [1, 1, 1, 0, 0, 0, 0, 0, 0, 0]
}

df = pd.DataFrame(data)

X = df.drop('eligibility', axis=1)
y = df['eligibility']
```

```
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

```

- Creation of the dataset
- Splitting the features (X)
- Splitting target variable (y)

TRAIN-TEST SPLIT

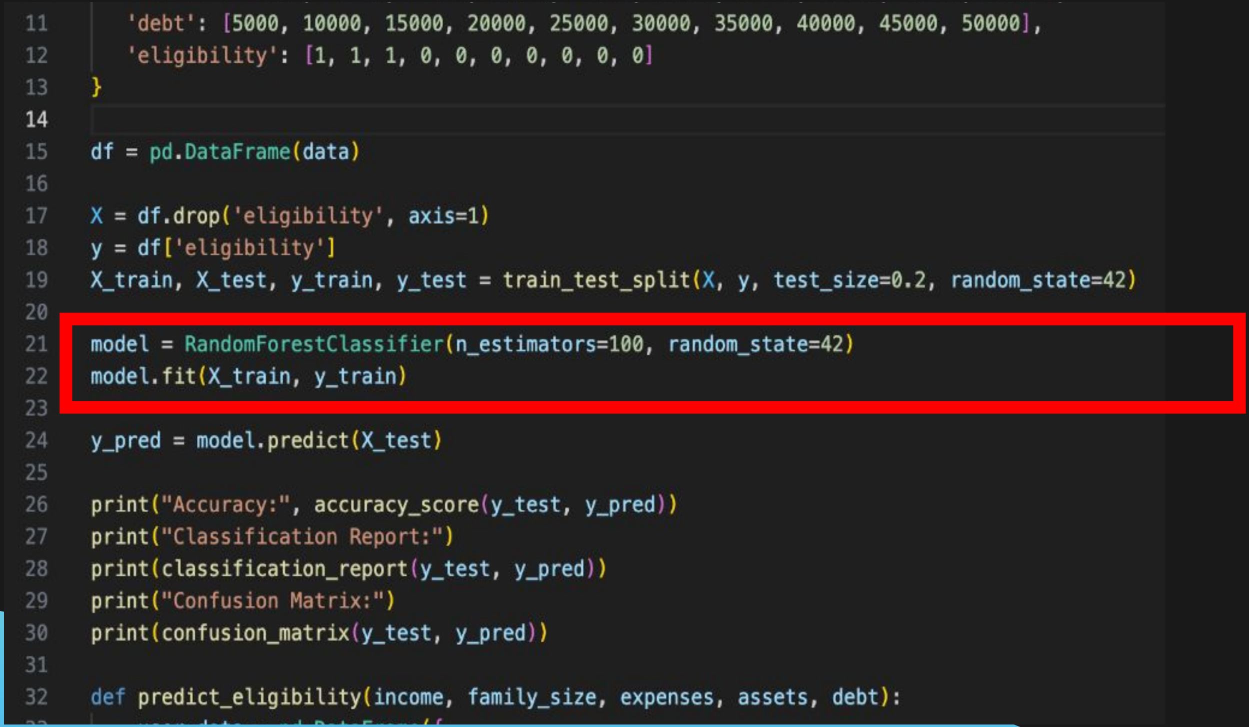


```
data = {
    'income': [25000, 35000, 45000, 55000, 65000, 75000, 85000, 95000, 105000, 115000],
    'family_size': [2, 3, 4, 5, 6, 7, 8, 9, 10, 11],
    'expenses': [1000, 1500, 2000, 2500, 3000, 3500, 4000, 4500, 5000, 5500],
    'assets': [5000, 10000, 15000, 20000, 25000, 30000, 35000, 40000, 45000, 50000],
    'debt': [5000, 10000, 15000, 20000, 25000, 30000, 35000, 40000, 45000, 50000],
    'eligibility': [1, 1, 1, 0, 0, 0, 0, 0, 0, 0]
}

df = pd.DataFrame(data)

X = df.drop('eligibility', axis=1)
y = df['eligibility']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:")
print(classification_report(y_test, y_pred))
```

Data is split into training and testing sets: 80% training data and 20% testing data

Ensures that the model isn't overfitted to the training data

RANDOM FOREST CLASSIFIER



```python
    'debt': [5000, 10000, 15000, 20000, 25000, 30000, 35000, 40000, 45000, 50000],
    'eligibility': [1, 1, 1, 0, 0, 0, 0, 0, 0, 0]
}

df = pd.DataFrame(data)

X = df.drop('eligibility', axis=1)
y = df['eligibility']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:")
print(classification_report(y_test, y_pred))
print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))

def predict_eligibility(income, family_size, expenses, assets, debt):
```

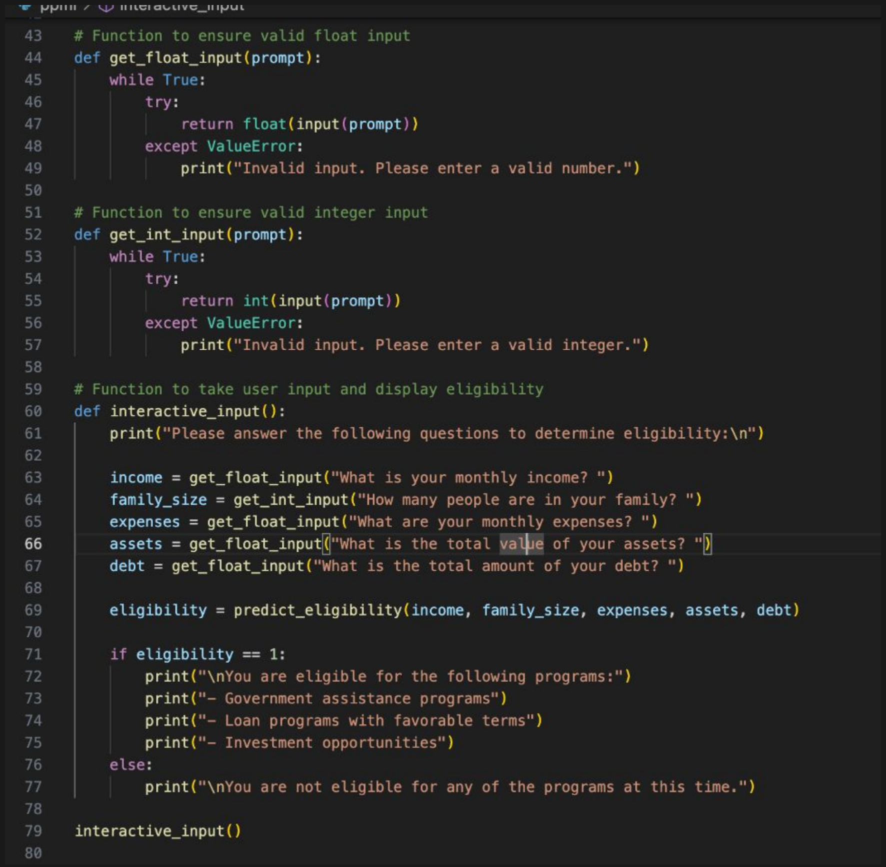
Builds multiple decision trees and combines

their predictions to improve accuracy

The model is trained on the training data

(X_train, y_train)

INTERACTIVE USER INPUT



```python
# Function to ensure valid float input
def get_float_input(prompt):
    while True:
        try:
            return float(input(prompt))
        except ValueError:
            print("Invalid input. Please enter a valid number.")

# Function to ensure valid integer input
def get_int_input(prompt):
    while True:
        try:
            return int(input(prompt))
        except ValueError:
            print("Invalid input. Please enter a valid integer.")

# Function to take user input and display eligibility
def interactive_input():
    print("Please answer the following questions to determine eligibility:\n")

    income = get_float_input("What is your monthly income? ")
    family_size = get_int_input("How many people are in your family? ")
    expenses = get_float_input("What are your monthly expenses? ")
    assets = get_float_input("What is the total value of your assets? ")
    debt = get_float_input("What is the total amount of your debt? ")

    eligibility = predict_eligibility(income, family_size, expenses, assets, debt)

    if eligibility == 1:
        print("\nYou are eligible for the following programs:")
        print("- Government assistance programs")
        print("- Loan programs with favorable terms")
        print("- Investment opportunities")
    else:
        print("\nYou are not eligible for any of the programs at this time.")

interactive_input()
```

User Input: income, family size, expenses, assets, and debt.

get_float_input(): Validates numerical inputs

get_int_input(): Validates integer input

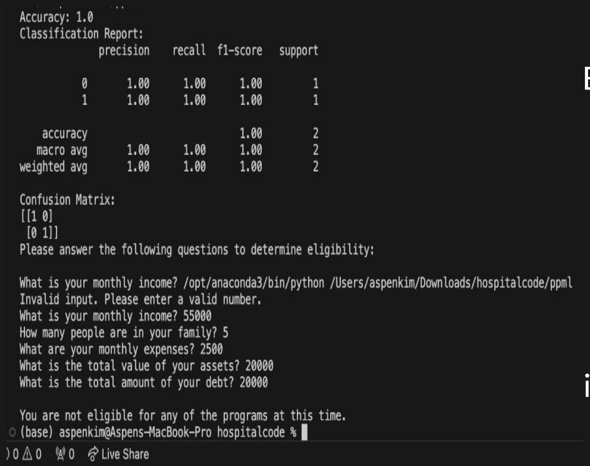
Uses trained RandomForestClassifier to predict eligibility

Output: Displays eligibility result:

Eligible: Lists potential programs (e.g., government assistance, loan programs).

Not Eligible: Informs user they are ineligible

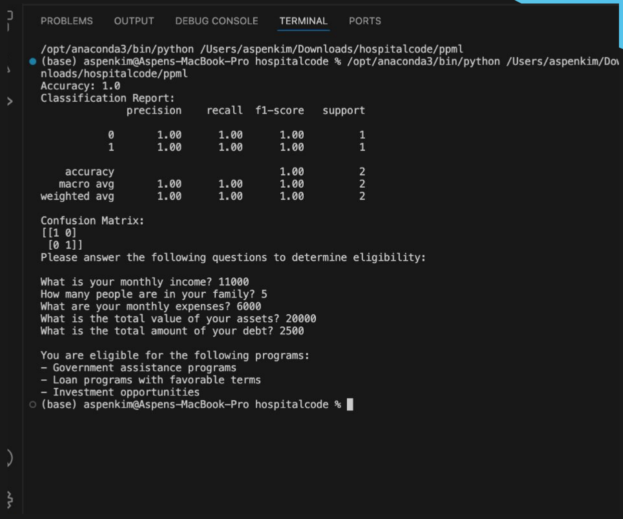
OUTPUT



```
Accuracy: 1.0
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         1
           1       1.00      1.00      1.00         1

    accuracy                           1.00         2
   macro avg       1.00      1.00      1.00         2
weighted avg       1.00      1.00      1.00         2

Confusion Matrix:
[[1 0]
 [0 1]]
Please answer the following questions to determine eligibility:

What is your monthly income? /opt/anaconda3/bin/python /Users/aspenkim/Downloads/hospitalcode/ppml
Invalid input. Please enter a valid number.
What is your monthly income? 55000
How many people are in your family? 5
What are your monthly expenses? 2500
What is the total value of your assets? 20000
What is the total amount of your debt? 20000

You are not eligible for any of the programs at this time.
(base) aspenkim@Aspens-MacBook-Pro hospitalcode %
```

Eligible (1): If the model predicts the user is eligible, it displays a list of programs the user can access.

Not Eligible (0): If the model predicts ineligibility, it informs the user they are not eligible for any programs.



```
/opt/anaconda3/bin/python /Users/aspenkim/Downloads/hospitalcode/ppml
(base) aspenkim@Aspens-MacBook-Pro hospitalcode % /opt/anaconda3/bin/python /Users/aspenkim/Dow
nloads/hospitalcode/ppml
Accuracy: 1.0
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         1
           1       1.00      1.00      1.00         1

    accuracy                           1.00         2
   macro avg       1.00      1.00      1.00         2
weighted avg       1.00      1.00      1.00         2

Confusion Matrix:
[[1 0]
 [0 1]]
Please answer the following questions to determine eligibility:

What is your monthly income? 11000
How many people are in your family? 5
What are your monthly expenses? 6000
What is the total value of your assets? 20000
What is the total amount of your debt? 2500

You are eligible for the following programs:
- Government assistance programs
- Loan programs with favorable terms
- Investment opportunities
(base) aspenkim@Aspens-MacBook-Pro hospitalcode %
```

THANK YOU FOR LISTENING

"The best way to predict the future is to create it." – Abraham Lincoln