

About the Project

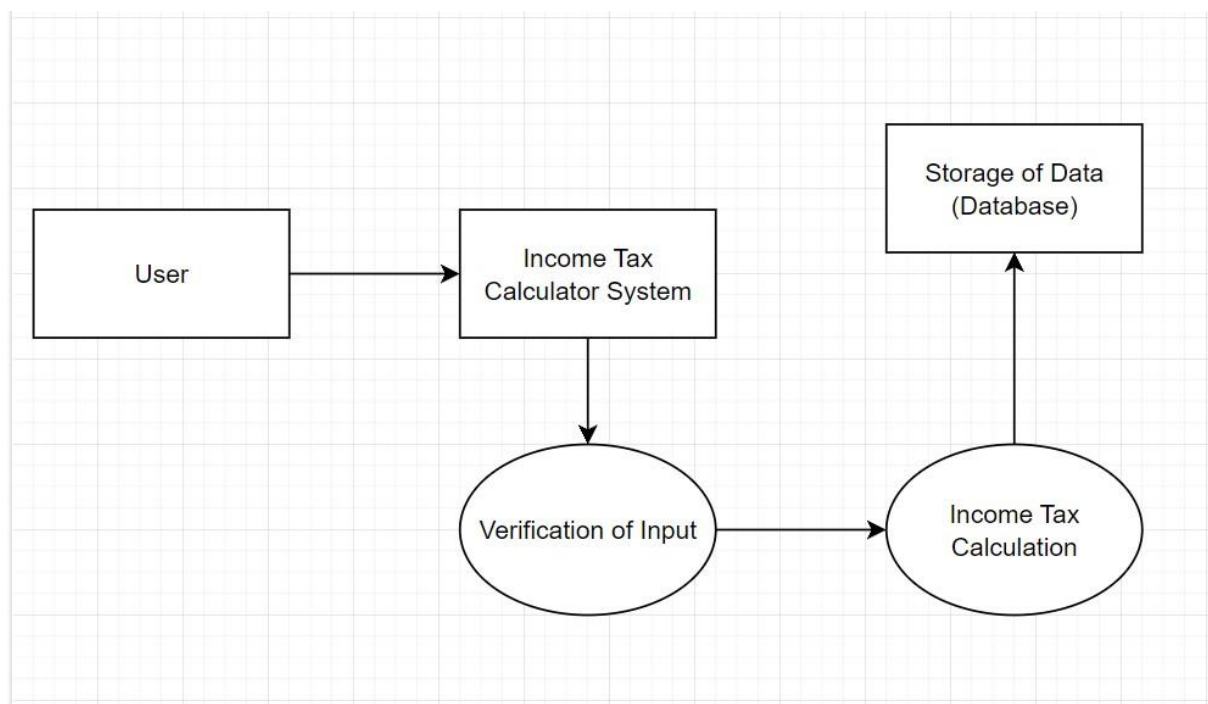
Income Tax Calculator Project is an income tax calculation software for individuals and corporations. Users can add & edit the records according to his/her needs as well as it can enable user to search for a certain record or deleting the required data. It was written using the C programming language.

This income tax calculator project is a simple yet helpful example to show how programming can be used to solve real-world problems. It allows users to add new records, calculate income tax based on their income and save the details for future reference. It reduces the efforts to manage and update your database with the additional features of the program such as the ability to search, modify, and delete taxpayer records.

Based on the taxpayer's income and tax rate, the project determines the amount of income tax to collect. It

provides taxpayers with an effective and practical way to calculate income tax without the need for manual calculations.

Individuals, businesses, and organizations can use this Income Tax Calculation Project to calculate their income tax. Financial planners, tax professionals, and anyone who wants to determine their income tax burden can benefit. Concluding we can say that this project is a great example of how programming can be used to automate procedures, calculations, and offer useful tools for both people and organizations.



Zero Level DFD (Data Flow Diagram) for
Income Tax Calculator

Modules Created

The Income Tax Calculator project consists of the following modules:

- **Add New Record**: This module allows users to add a new tax record to the system by entering the taxpayer's details such as name, income, and tax rate. The system determines the taxpayer's income tax liability when a new record is added using their income and tax rate.
- **List all Taxpayer along with income tax to be paid**: This module lists all the tax records in the system, along with the calculated income tax to be paid. The user can view all tax records and their respective tax amounts, which provides an easy way to track tax liabilities.
- **Search**: This module allows users to search for a specific tax record by entering the taxpayer's name. The system displays the tax record for the specified taxpayer, including their name, income, tax rate, and calculated tax amount.

- **Edit**: This module allows users to modify an existing tax record by updating the taxpayer's details such as name, income, and tax rate. The system recalculates the income tax liability based on the updated data whenever a tax record is updated.
- **Delete Record**: This module allows users to delete an existing tax record from the system. The system deletes the tax record from the database and makes the taxpayer's information unavailable.
- **Calculate Tax**: This module will find the tax on the user entered income using tax brackets and will store it in the database.

Each module in the Income Tax Calculator project serves a specific purpose, enabling users to perform different functions in the system. Users can navigate and use the various modules with ease thanks to the project's user-friendly interface.

Extra Creativity

The Income Tax Calculator project consists of the following extra features :

- **Color Header File** : It is user defined header file consisting of various declared functions to provide colors in the output screen which in result give an attractive , easy to understand and user-friendly interface.
- **Login Module**: This module will verify the user and will allow to run the code or the program after entering the correct password (Login Successful). If the incorrect password is entered then the user will be displayed with the message "Wrong Password, Login Failed. Try Again!" and the program will be closed.
- **Error Handling**: Each module of the program are carefully designed to handle all the possible errors with an ease and the user will be displayed with a short and proper message and rollback to its previous state.

For Eg:-

1.) If User enter Mobile Number of other than 10 digits then the user will be displayed with the message "Sorry, Mobile Number must be of 10 Digits! Please Try Again.".

2.) If in registration user enters a mobile no. which is already owned by other Tax Payer then the user will be displayed with the message "Sorry, Entered Mobile No. is already Owned by other Tax Payer! Please Try Again."

3.) If User enters $\text{income} < 0$ which is not possible then the user will be displayed with the message "Sorry, Income can't be negative! Please Try Again."

And many more...

COLOR Header File

```
#include<stdio.h>
#include<string.h>
void black() {
    printf("\033[0;30m");
}
void red() {
    printf("\033[0;31m");
}
void green() {
    printf("\033[0;32m");
}
void yellow() {
    printf("\033[0;33m");
}
void blue() {
    printf("\033[0;34m");
}
```

```
void purple() {  
    printf("\033[0;35m");  
}  
void cyan() {  
    printf("\033[0;36m");  
}  
void white() {  
    printf("\033[0;37m");  
}  
void reset() {  
    printf("\033[0m");  
}
```

Coding

```
#include<stdio.h>
```

```
#include<string.h>
```

```
typedef struct income_tax_calculator{  
    char first_name[50];
```



```
char last_name[50];  
char mobile_num[10];  
float income;  
float tax;  
}income;
```

```
void login(int *Verify);  
void add(income *TaxPayer, int *numPayerS);  
void list(income *TaxPayer, int numPayerS);  
void search(income *TaxPayer, int numPayerS);  
void edit(income *TaxPayer, int numPayerS);  
void delete(income *TaxPayer, int *numPayerS);  
void caltax(float income, float *tax);  
void main(){
```

```
    int numPayerS=0; //Variable to count the entries of  
    the TaxPayer in the database
```

```
    int choice; //To perform the operation
```

```
    income TaxPayer[100]; //Limited for 100 entries only
```

```
    int Verify=0; // variable to check if the password  
    entered is correct or not
```

```
    login(&Verify); // Login/Authentication
```

```
if(Verify==1){
do {
    cyan();
    printf("\n\n");
    printf("-----\n");
    printf("\tIncome Tax Calculator\n");
    printf("-----\n");
    printf("\n1. Add New Record\n");
    printf("2. List All Records\n");
    printf("3. Search Record\n");
    printf("4. Edit Record\n");
    printf("5. Delete Record\n");
    printf("6. Exit\n");
    purple();
    printf("\nEnter your choice: ");
    scanf("%d", &choice);
    system("cls");
    switch(choice) {
        case 1:
            add(TaxPayer, &numPayerS);
```

```
        break;
case 2:
    list(TaxPayer, numPayerS);
    break;
case 3:
    search(TaxPayer, numPayerS);
    break;
case 4:
    edit(TaxPayer, numPayerS);
    break;
case 5:
    delete(TaxPayer, &numPayerS);
    break;
case 6:
    system("cls");
    green();
    printf("Exiting Program. Thank You!\n");
    reset();
    break;
default:
```

```
        system("cls");
        red();
        printf("Invalid choice.\n");
        reset();
        break;
    }
}while(choice!=6);
}

red();

printf("You have successfully exited the
program...\n\n");

reset();

return 0;
}
```

```
void login(int *Verify) {
    char password[15];
    blue();
    printf("\nEnter Password: ");
    scanf("%s",&password);
```

```
system("cls");
FILE *pass;
pass=fopen("Password.txt","r");
char myString[15];
fgets(myString,15,pass);
fclose(pass);
if (strcmp(password, myString) == 0) {
    green();
    printf("Login Successful!\n\n");
    *Verify=1; //Value changed as pass is true
    reset();
}
else {
    red();
    printf("Wrong Password, Login Failed. Try
Again!.\n");
    reset(); //else program is closed
}
}

void add(income *TaxPayer, int *numPayerS) {
```

```
yellow();
```

income newTaxpayer; // We are creating a new Struct to take all the details and will store the details at the end in the array created

```
printf("Enter First name of Tax Payer: ");
```

```
scanf("%s",newTaxpayer.first_name);
```

```
printf("Enter Last name of Tax Payer: ");
```

```
scanf("%s",newTaxpayer.last_name);
```

```
printf("Enter Mobile Number of Tax Payer: ");
```

```
scanf("%s",newTaxpayer.mobile_num);
```

```
reset();
```

```
if(strlen(newTaxpayer.mobile_num)!=10){ //  
Checking enter mobile no. is of 10 digits or not
```

```
system("cls");
```

```
red();
```

```
printf("Sorry, Mobile Number must be of 10  
Digits!\nPlease Try Again.\n\n");
```

```
reset();
```

```
return;
```

```
}
```

```
int z;
```

```
for(z=0;z<*numPayerS;z++){ //Loop to verify if  
entered mobile_number is not duplicate
```

```
    if(strcmp(TaxPayer[z].mobile_num,newTaxpayer.  
mobile_num)==0){
```

```
        system("cls");
```

```
        red();
```

```
        printf("Sorry, Entered Mobile No. is already  
Owned by other Tax Payer!\nPlease Try Again.\n\n");
```

```
        reset();
```

```
        return;
```

```
    }
```

```
}
```

```
yellow();
```

```
printf("Enter income of taxpayer for the year: ");
```

```
scanf("%f", &newTaxpayer.income);
```

```
reset();
```

```
if(newTaxpayer.income<0){ // Income must be +ve
```

```
    system("cls");
```

```
    red();
```

```
        printf("Sorry, Income can't be negative!\nPlease  
Try Again.\n\n");
```

```
        reset();
```

```
        return;
```

```
    }
```

```
    caltax(newTaxpayer.income,&newTaxpayer.tax);
```

```
    TaxPayer[*numPayerS] = newTaxpayer;
```

```
    *numPayerS=*numPayerS+1;
```

```
    system("cls");
```

```
    green();
```

```
    printf("\nRecord added successfully!\n");
```

```
    reset();
```

```
}
```

```
void list(income *TaxPayer, int numPayerS) {
```

```
    if (numPayerS==0){ //Checking if there is a value or  
    not
```

```
        red();
```

```
        printf("Record List is Empty!\n");
```

```
        reset();
```

```
}
```



```

else{
    green();
    printf("\nTaxpayer\t\tMobile Number\t\tIncome\t\tTax to be Paid\n");
    reset();
    // Loop for printing all the details of the TaxPayers
    int i;
    for(i=0; i<numPayerS; i++) {
        white();
        printf("%s %s\t\t%s\t\t%.2f\t\t%.2f\n",
TaxPayer[i].first_name,TaxPayer[i].last_name,TaxPayer
[i].mobile_num, TaxPayer[i].income, TaxPayer[i].tax);
        reset();
    }
}

```

```

void search(income *TaxPayer, int numPayerS) {
    if(numPayerS==0) {
        system("cls");
        red();
    }
}

```

```
    printf("Record List is empty!\n");
    reset();
    return;
}

char num[10];

int found = 0; // Variable to check if the name exists
or not

purple();

printf("\nPlease Enter the Mobile Number of the
Taxpayer to search: ");

scanf("%s", num);

reset();

if(strlen(num)!=10){
    system("cls");
    red();

    printf("Sorry, Mobile Number must be of 10
Digits!\nPlease Try Again.\n\n");

    reset();
    return;
}

int i;
```

```

system("cls");
for(i=0; i<numPayerS; i++) {
    if(strcmp(TaxPayer[i].mobile_num,num)==0) {
//Prinitng details of the record found
        green();
        printf("\nRecord Found!");
        reset();
        white();
        printf("\nTaxpayer: %s %s\nMobile Number:
%s\nIncome: %.2f\nTax to be Paid: %.2f\n",
TaxPayer[i].first_name,TaxPayer[i].last_name,TaxPayer
[i].mobile_num, TaxPayer[i].income, TaxPayer[i].tax);
        reset();
        found = 1;
        break;
    }
}
if(found==0) {
system("cls");
red();
    printf("\nRecord not found!\n");
}

```

```
    reset();  
}  
}
```

```
void edit(income *TaxPayer, int numPayerS) {  
    char num[10];  
    int found=0;  
    if(numPayerS==0) {  
        system("cls");  
        red();  
        printf("Record List is empty!\n");  
        reset();  
        return;  
    }  
    purple();  
    printf("\nPlease Enter the Mobile Number of the  
    Taxpayer to edit: ");  
    scanf("%s", num);  
    reset();  
    if(strlen(num)!=10){
```

```
    system("cls");
    red();
    printf("Sorry, Mobile Number must be of 10
Digits!\nPlease Try Again.\n\n");
    reset();
    return;
}

int i;
float new_income;
system("cls");
for(i=0; i<numPayerS; i++) {
    if(strcmp(TaxPayer[i].mobile_num,num)==0) {
        green();
        printf("\nRecord Found!");
        reset();
        white();

        printf("\nTaxpayer: %s %s\nMobile Number:
%s\nIncome: %.2f\nTax to be Paid: %.2f\n",
TaxPayer[i].first_name,TaxPayer[i].last_name,TaxPayer
[i].mobile_num, TaxPayer[i].income, TaxPayer[i].tax);
        reset();
    }
}
```

```
yellow();  
printf("\nEnter new income for the taxpayer: ");  
scanf("%f", &new_income);  
reset();  
if(new_income<0){ // Income must be +ve  
system("cls");  
red();  
printf("Sorry, Income can't be negative!\nPlease  
Try Again.\n\n");  
reset();  
return;  
}  
  
TaxPayer[i].income=new_income; //Assigning  
value to the indice of income  
  
caltax(TaxPayer[i].income,&TaxPayer[i].tax);  
system("cls");  
green();  
printf("\nRecord updated successfully!\n");  
reset();  
found = 1;  
break;
```

```
    }  
}  
if(found==0) {  
    system("cls");  
    red();  
    printf("\nRecord not found!\n");  
    reset();  
}  
}
```

```
void delete(income *TaxPayer, int *numPayerS) {  
    char num[10];  
    int found=0;  
    if(*numPayerS==0) {  
        system("cls");  
        red();  
        printf("Record List is empty!\n");  
        reset();  
        return;  
    }  
}
```

```
purple();

printf("\nPlease Enter the Mobile Number of the
Taxpayer to delete: ");

scanf("%s", num);

reset();

if(strlen(num)!=10){

    system("cls");

    red();

    printf("Sorry, Mobile Number must be of 10
Digits!\nPlease Try Again.\n\n");

    reset();

    return;

}

int i;

system("cls");

for(i=0; i<*numPayerS; i++) {

if(strcmp(TaxPayer[i].mobile_num,num)==0) {

    green();

    printf("\nRecord Found!");

    reset();

    white();
```



```
printf("\nTaxpayer: %s %s\nMobile Number:
%s\nIncome: %.2f\nTax to be Paid: %.2f\n",
TaxPayer[i].first_name,TaxPayer[i].last_name,TaxPayer
[i].mobile_num, TaxPayer[i].income, TaxPayer[i].tax);
// Loop to shift the value at the previous indice as the
element is Found
```

```
reset();
```

```
int j;
```

```
for(j=i; j<(*numPayerS)-1; j++) { //Removing the
indice of the data that need to be deleted
```

```
TaxPayer[j] = TaxPayer[j+1];
```

```
}
```

```
*numPayerS=*numPayerS-1;
```

```
green();
```

```
printf("\nRecord deleted successfully!\n");
```

```
reset();
```

```
found = 1;
```

```
break;
```

```
}
```

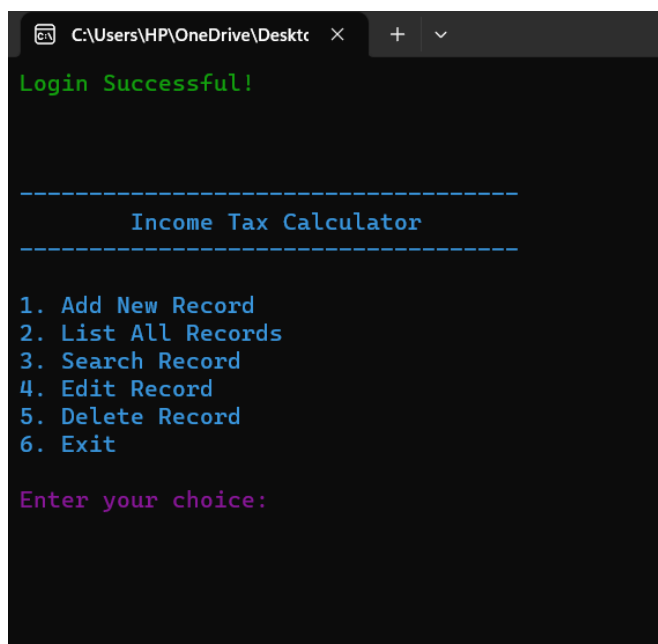
```
}
```

```
if(found==0) {
```

```
system("cls");  
red();  
    printf("\nRecord not found!\n");  
    reset();  
}  
  
}  
  
void caltax(float income,float *tax){  
    // We will find the tax of the user using tax brackets  
    if (income <= 250000) {  
        *tax = 0;  
    } else if (income > 250000 && income <= 500000) {  
        *tax = (income - 250000) * 0.05;  
    } else if (income > 500000 && income <= 1000000) {  
        *tax = (income - 500000) * 0.2 + 12500;  
    } else {  
        *tax = (income - 1000000) * 0.3 + 112500;  
    }  
}
```

Output

Welcome Screen



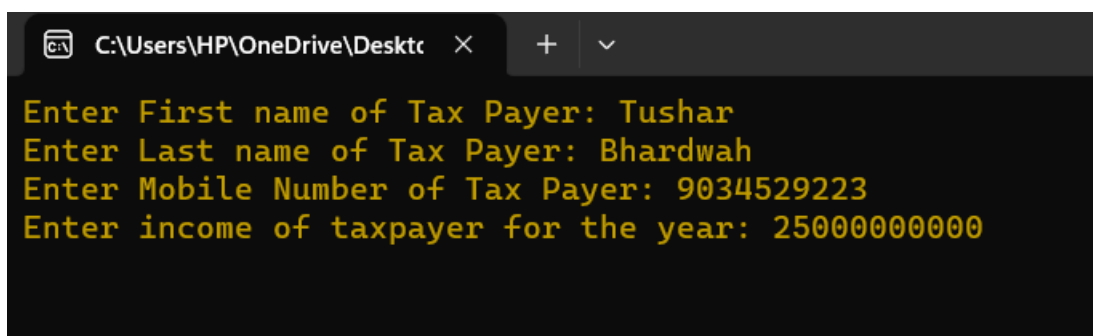
```
C:\Users\HP\OneDrive\Desktop x + v
Login Successful!

-----
Income Tax Calculator
-----

1. Add New Record
2. List All Records
3. Search Record
4. Edit Record
5. Delete Record
6. Exit

Enter your choice:
```

Adding a New Record



```
C:\Users\HP\OneDrive\Desktop x + v
Enter First name of Tax Payer: Tushar
Enter Last name of Tax Payer: Bhardwah
Enter Mobile Number of Tax Payer: 9034529223
Enter income of taxpayer for the year: 25000000000
```

List all the Tax Payers

```
C:\Users\HP\OneDrive\Desktc  X  +  v

Taxpayer      Mobile Number      Income      Tax to be Paid
Tushar Bhardwaj  7009343545      350000.00      5000.00
Tushar Bhardwah  9034529223      24999999488.00  7499812352.00
Tushar Shukla    5138735467      250001.00      0.05

-----
Income Tax Calculator
-----

1. Add New Record
2. List All Records
3. Search Record
4. Edit Record
5. Delete Record
6. Exit

Enter your choice:
```

Search for Record

```
C:\Users\HP\OneDrive\Desktc  X  +  v

Record Found!
Taxpayer: Tushar Bhardwah
Mobile Number: 9034529223
Income: 24999999488.00
Tax to be Paid: 7499812352.00

-----
Income Tax Calculator
-----

1. Add New Record
2. List All Records
3. Search Record
4. Edit Record
5. Delete Record
6. Exit

Enter your choice:
```

Edit a Record

```
C:\Users\HP\OneDrive\Desktc  X  +  v

Record Found!
Taxpayer: Tushar Bhardwaj
Mobile Number: 7009343545
Income: 250001.00
Tax to be Paid: 0.05

Enter new income for the taxpayer: 250001
```

Delete a Record

```
C:\Users\HP\OneDrive\Desktc  X  +  v

Record Found!
Taxpayer: Tushar Bhardwaj
Mobile Number: 7009343545
Income: 250001.00
Tax to be Paid: 0.05

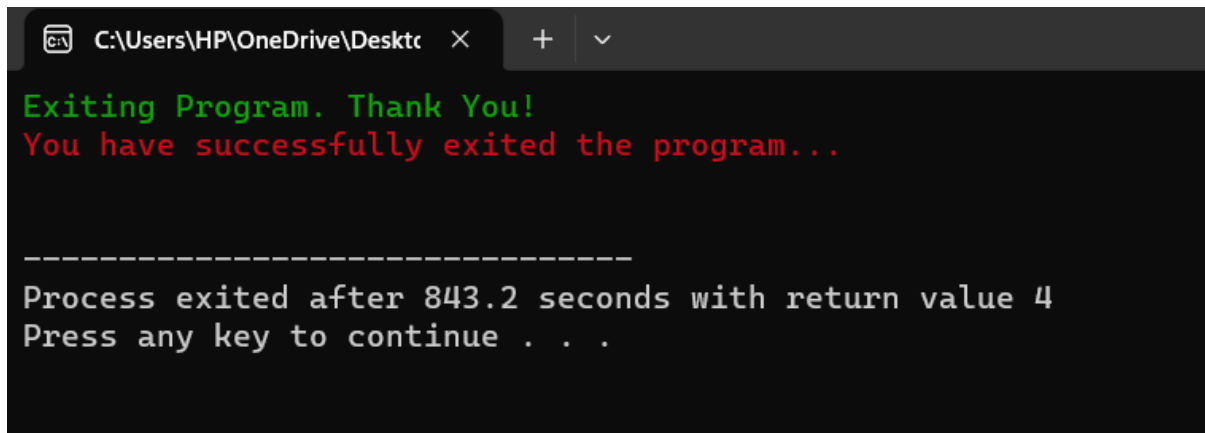
Record deleted successfully!

-----
Income Tax Calculator
-----

1. Add New Record
2. List All Records
3. Search Record
4. Edit Record
5. Delete Record
6. Exit

Enter your choice:
```

Exit from the Program



A screenshot of a Windows command prompt window. The title bar shows the file path 'C:\Users\HP\OneDrive\Desktop' and standard window controls. The command prompt displays the following text: 'Exiting Program. Thank You!' in green, 'You have successfully exited the program...' in red, a separator line of dashes, and 'Process exited after 843.2 seconds with return value 4' and 'Press any key to continue . . .' in white.

```
C:\Users\HP\OneDrive\Desktop >
Exiting Program. Thank You!
You have successfully exited the program...

-----
Process exited after 843.2 seconds with return value 4
Press any key to continue . . .
```

Conclusion

The Income Tax Calculator project is a useful software that gives taxpayers a quick and simple way to determine how much income tax they have to pay. The project is developed using the C programming language and consists of different modules such as Add New Record, List all Tax Payer along with income tax to be paid, Search, Edit, and Delete Record, which serve specific purposes.

The user-friendly interface of this income tax calculation project makes it simple for users to add, edit, search for, and remove tax records. The system calculates the income tax liability based on the taxpayer's income and tax rate, providing accurate and reliable results.

To determine their income tax liability, individuals, businesses, and organizations can make use of the Income Tax Calculator project. Tax experts, financial planners, and anyone else who needs to figure their income tax can all use it. The project can be further

improved by adding additional features such as generating tax reports, automated reminders, and integration with external tax databases.

Overall, the Income Tax Calculator project is an excellent example of how technology can simplify complex processes and make life easier for people. With its user-friendly interface and efficient calculations, it provides a valuable service to taxpayers.

Github:

https://github.com/TuShArBhArDwA/Income_Tax_Calculator.git
