PERSONAL EXPENSE MANAGER

Submitted in the partial fulfillment of the Degree in Bachelor of Technology in Computer Science and Engineering

Submitted by

SECTION : K21GX

Name	Registration Number	Roll Number
Girish Nandan Shukla	12108200	RK21GXA44
Aman Singh Vatsa	12108407	RK21GXA41
Aryan rana	12108149	RK21GXA16

TABLE OF CONTENT

ACKNOWLEDGEMENT

I would like to thank my teacher Dr. Om Prakash Yadav for giving such opportunity to do this wonderful project on Personal Expense Manager. I would also like to thanks all my team member who participated in this project.

I also like to thanks my seniors who has given feedbacks and supported us to complete this project before deadline.

INTRODUCTION

A personal expense manager project is a software application designed to help individuals manage their personal finances. The goal of the project is to create a user-friendly platform that allows individuals to easily track their income and expenses, set budgets, and analyse their spending habits.
The personal expense manager project will include features such as expense categorisation, reporting, and budget tracking. Users will be able to input their income and expenses, which will be automatically categorised and displayed in an easy-to-read format. The platform will also allow users to set monthly budgets for different categories, such as food, entertainment, and transportation.
The project will be designed with user experience in mind, ensuring that the platform is intuitive and easy to use.
It will also include helpful features such as bill reminders and investment tracking, as well as tools for debt management.
Overall, the personal expense manager project aims to provide individuals with a comprehensive tool to help them manage their personal finances and make informed financial decisions.
By using this platform, users will have greater control over their finances and be better equipped to achieve their financial goals.

ROLES OF EACH TEAM MEMBER.

ARYAN RANA
12108149

- Gathering information.
- Forming documentation.
- Making report.
- Implementing codes.
- presentation.

AMAN SINGH VASTA
12108407

- Checking documentation.
- Organizing structure of presentation.
- Implementing codes.

GIRISH NANDAN SHUKLA
12108200

- Testing end application.
- Organizing codes.
- Implementing codes.

PROPOSED SYSTEM

- DESCRIPTION

We have made Expense Manager System in java language. It is software where you can track a record of your expenses and manage them.

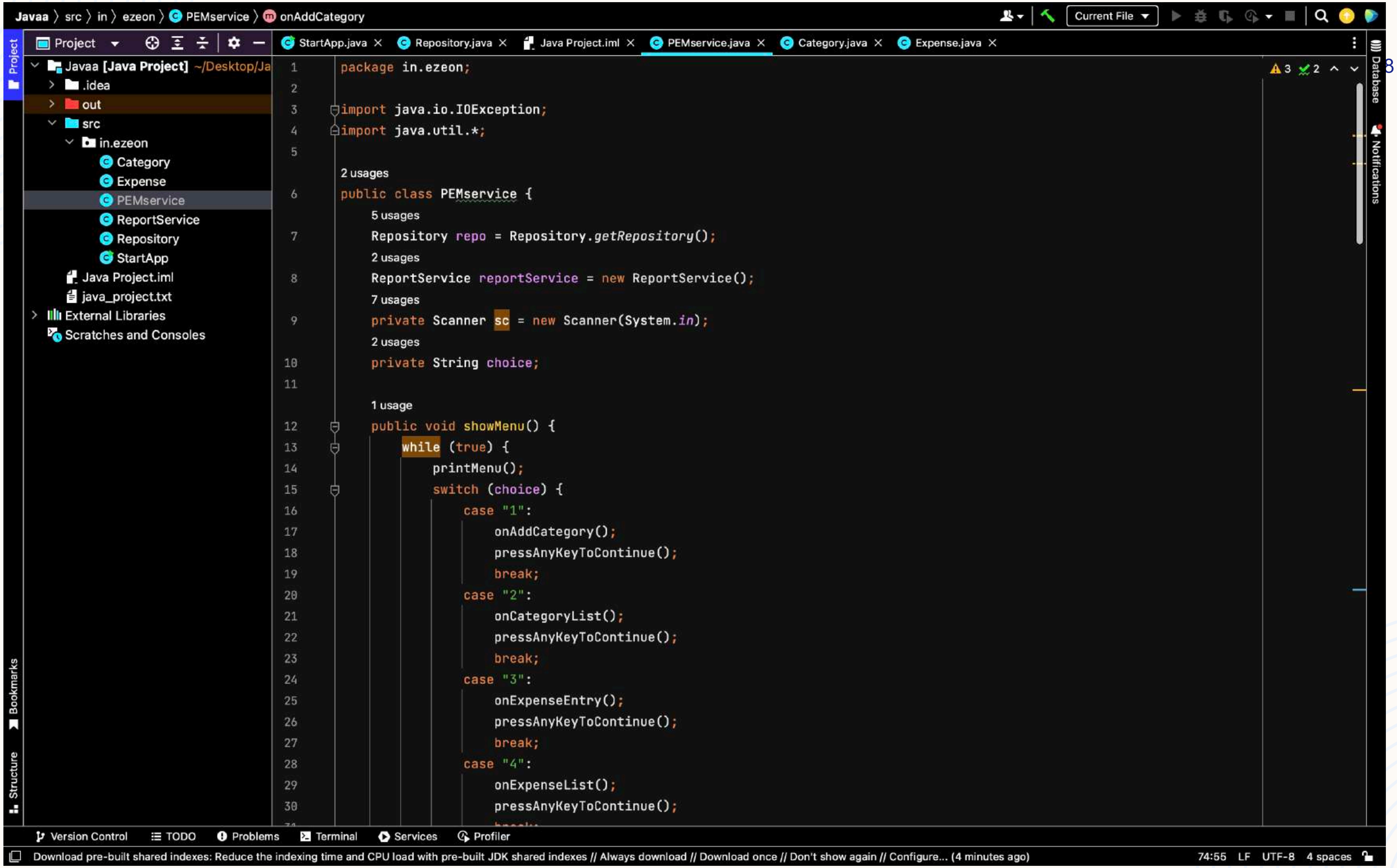
- RULES STATEMENT

```
Personal Expense Manager

Requirement:
1. Category Master : User can add or manage category
2. Expense Entry category wise
3. Expense List
4. Reports
          A. Category Expense Report


User Interface
1. Desktop Application/Window App : AWT/Swing/JavaFx
2. CUI: Character User Interface (switch-case)
3. Web GUI: HTML/CSS
4. Android App

We are going to use CUI



Database/Repository
1. JDBC API with MySQL/Oracle Database
2. File Based Repository
3. Collection API as Repository (We will manage the whole data temporarily
4. Cloud Repository

We will use Collection API
```

MAIN CODE

GITHUB LINK

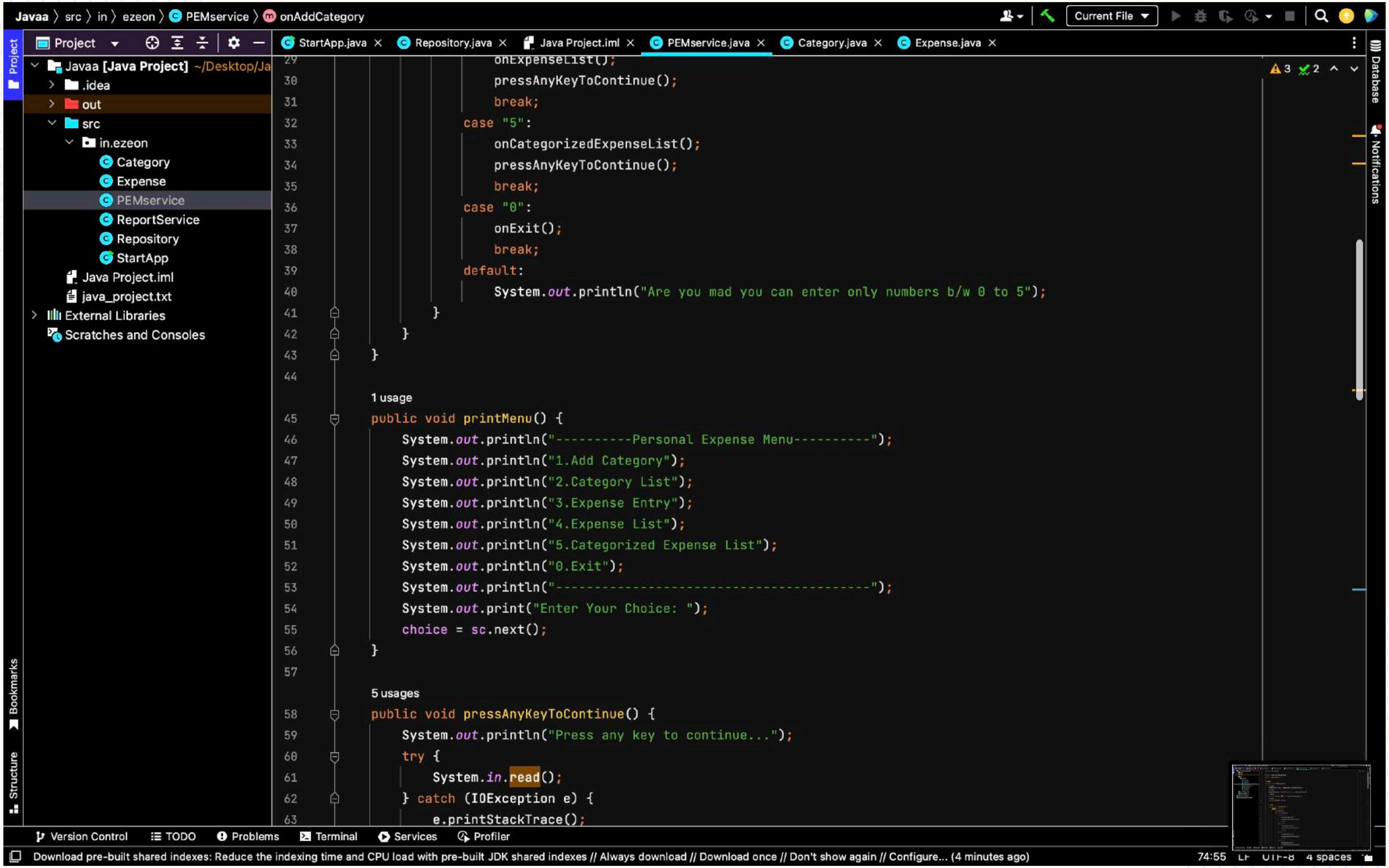
https://github.com/GIRISHSHUKLAA/Personal-Expense-Manager



```java
package in.ezeon;

import java.io.IOException;
import java.util.*;

2 usages
public class PEMservice {
    5 usages
    Repository repo = Repository.getRepository();
    2 usages
    ReportService reportService = new ReportService();
    7 usages
    private Scanner sc = new Scanner(System.in);
    2 usages
    private String choice;

    1 usage
    public void showMenu() {
        while (true) {
            printMenu();
            switch (choice) {
                case "1":
                    onAddCategory();
                    pressAnyKeyToContinue();
                    break;
                case "2":
                    onCategoryList();
                    pressAnyKeyToContinue();
                    break;
                case "3":
                    onExpenseEntry();
                    pressAnyKeyToContinue();
                    break;
                case "4":
                    onExpenseList();
                    pressAnyKeyToContinue();
```

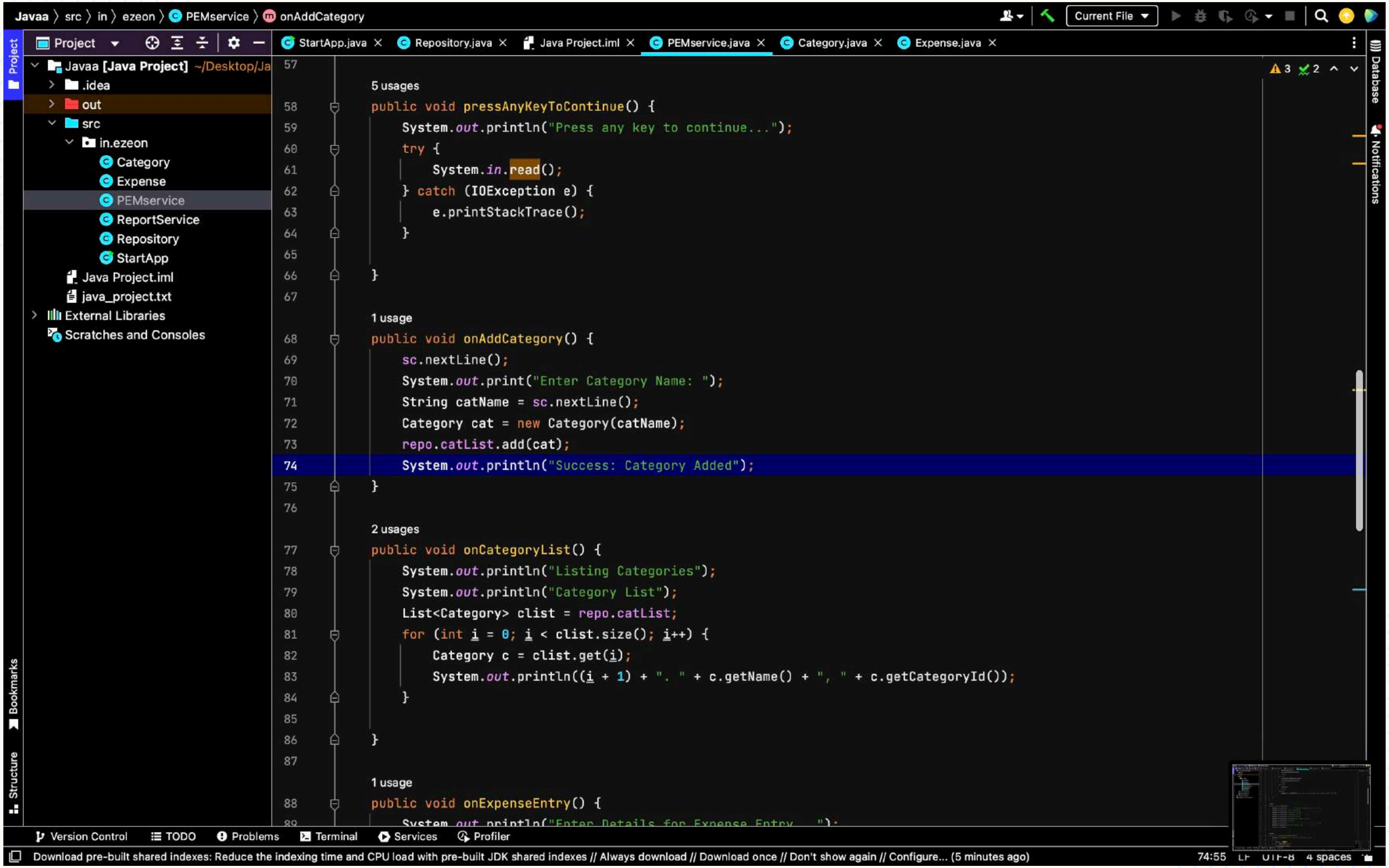
Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK shared indexes // Always download // Download once // Don't show again // Configure... (4 minutes ago)

74:55 LF UTF-8 4 spaces



```java
                    onExpenseList();
                    pressAnyKeyToContinue();
                    break;
                case "5":
                    onCategorizedExpenseList();
                    pressAnyKeyToContinue();
                    break;
                case "0":
                    onExit();
                    break;
                default:
                    System.out.println("Are you mad you can enter only numbers b/w 0 to 5");
            }
        }
    }

    1 usage
    public void printMenu() {
        System.out.println("----------Personal Expense Menu----------");
        System.out.println("1.Add Category");
        System.out.println("2.Category List");
        System.out.println("3.Expense Entry");
        System.out.println("4.Expense List");
        System.out.println("5.Categorized Expense List");
        System.out.println("0.Exit");
        System.out.println("----------------------------------------");
        System.out.print("Enter Your Choice: ");
        choice = sc.next();
    }

    5 usages
    public void pressAnyKeyToContinue() {
        System.out.println("Press any key to continue...");
        try {
            System.in.read();
        } catch (IOException e) {
            e.printStackTrace();
```

Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK shared indexes // Always download // Download once // Don't show again // Configure... (4 minutes ago)   74:55   LF   UTF-8   4 spaces



Javaa 〉 src 〉 in 〉 ezeon 〉 PEMservice 〉 onAddCategory

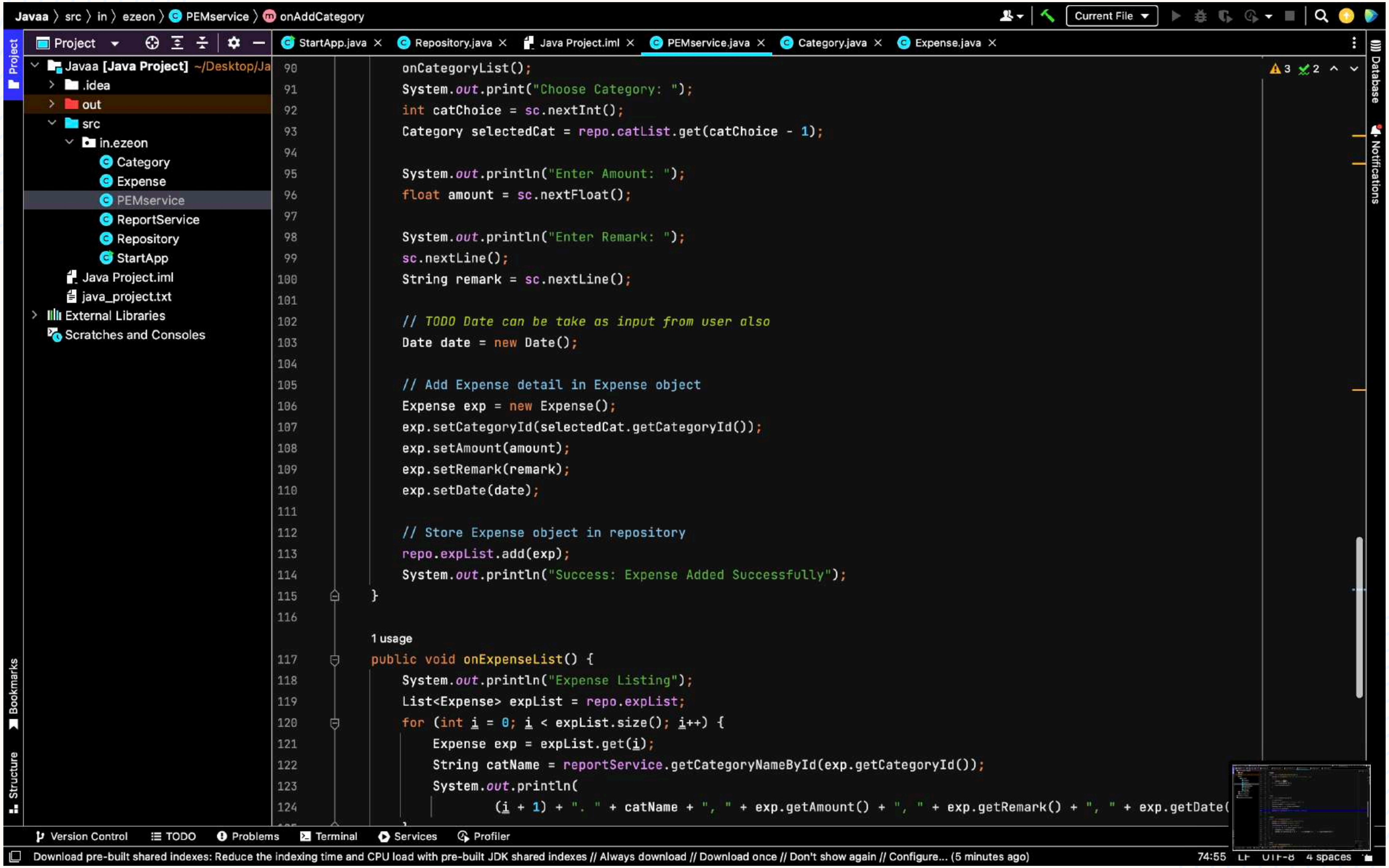
StartApp.java Repository.java Java Project.iml PEMservice.java Category.java Expense.java

```
Javaa [Java Project] ~/Desktop/Ja
  .idea
  out
  src
    in.ezeon
      Category
      Expense
      PEMservice
      ReportService
      Repository
      StartApp
    Java Project.iml
    java_project.txt
  External Libraries
  Scratches and Consoles
```

```java

    5 usages
    public void pressAnyKeyToContinue() {
        System.out.println("Press any key to continue...");
        try {
            System.in.read();
        } catch (IOException e) {
            e.printStackTrace();
        }

    }

    1 usage
    public void onAddCategory() {
        sc.nextLine();
        System.out.print("Enter Category Name: ");
        String catName = sc.nextLine();
        Category cat = new Category(catName);
        repo.catList.add(cat);
        System.out.println("Success: Category Added");
    }

    2 usages
    public void onCategoryList() {
        System.out.println("Listing Categories");
        System.out.println("Category List");
        List<Category> clist = repo.catList;
        for (int i = 0; i < clist.size(); i++) {
            Category c = clist.get(i);
            System.out.println((i + 1) + ". " + c.getName() + ", " + c.getCategoryId());
        }

    }

    1 usage
    public void onExpenseEntry() {
        System.out.println("Enter Details for Expense Entry...");
```

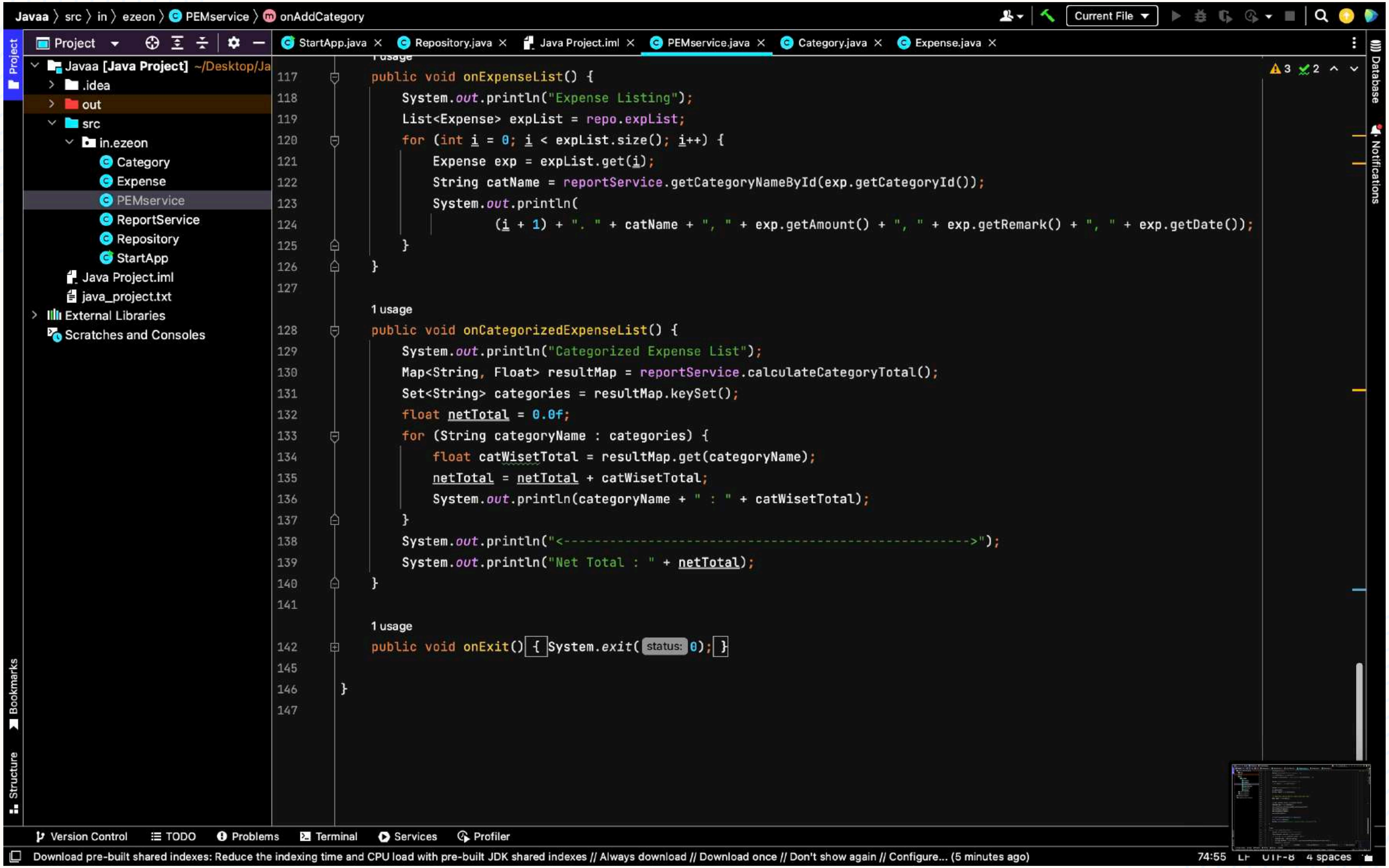
Version Control TODO Problems Terminal Services Profiler

Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK shared indexes // Always download // Download once // Don't show again // Configure... (5 minutes ago)   74:55 LF UTF-8 4 spaces



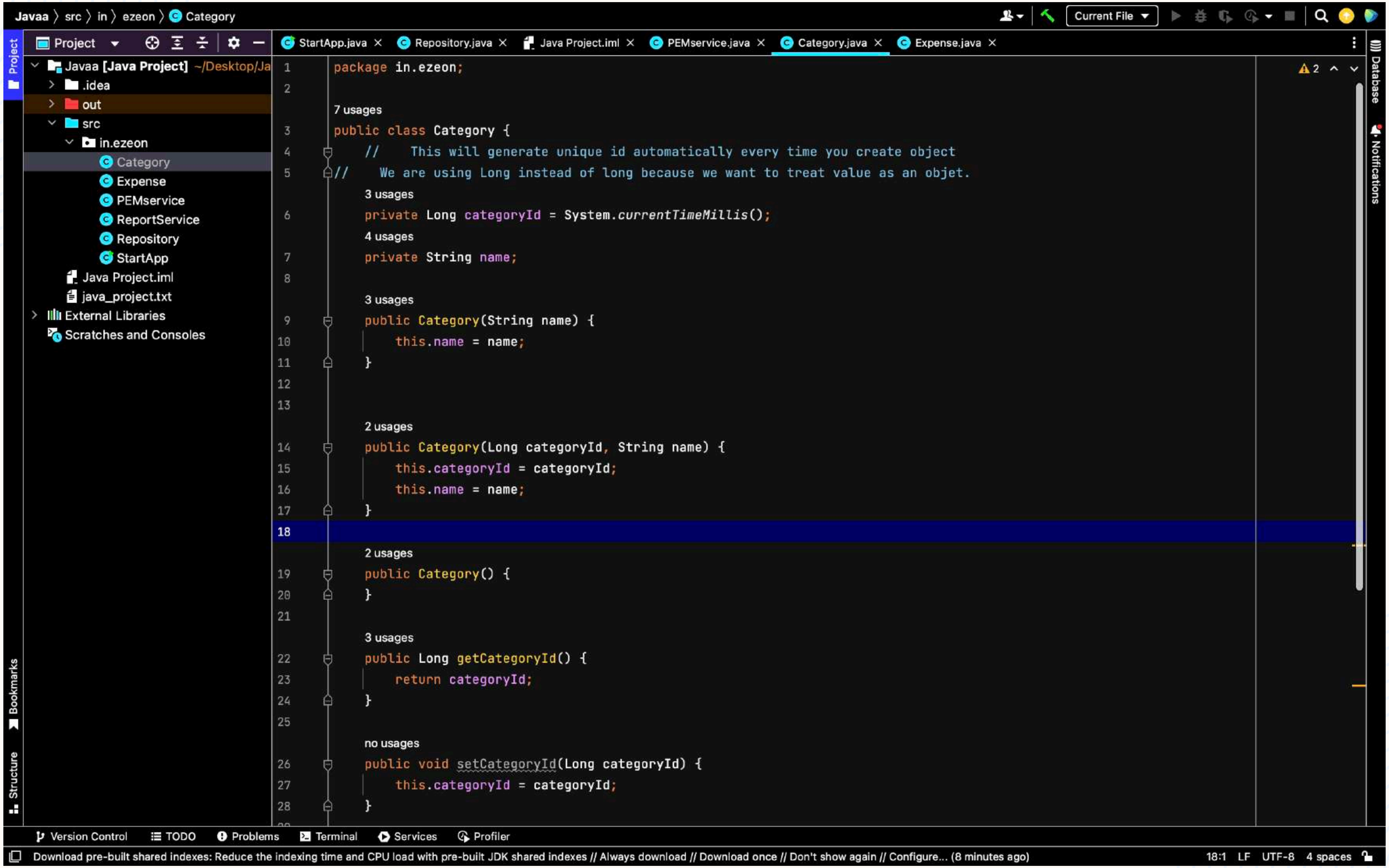
```java
        onCategoryList();
        System.out.print("Choose Category: ");
        int catChoice = sc.nextInt();
        Category selectedCat = repo.catList.get(catChoice - 1);

        System.out.println("Enter Amount: ");
        float amount = sc.nextFloat();

        System.out.println("Enter Remark: ");
        sc.nextLine();
        String remark = sc.nextLine();

        // TODO Date can be take as input from user also
        Date date = new Date();

        // Add Expense detail in Expense object
        Expense exp = new Expense();
        exp.setCategoryId(selectedCat.getCategoryId());
        exp.setAmount(amount);
        exp.setRemark(remark);
        exp.setDate(date);

        // Store Expense object in repository
        repo.expList.add(exp);
        System.out.println("Success: Expense Added Successfully");
    }

    1 usage
    public void onExpenseList() {
        System.out.println("Expense Listing");
        List<Expense> expList = repo.expList;
        for (int i = 0; i < expList.size(); i++) {
            Expense exp = expList.get(i);
            String catName = reportService.getCategoryNameById(exp.getCategoryId());
            System.out.println(
                    (i + 1) + ". " + catName + ", " + exp.getAmount() + ", " + exp.getRemark() + ", " + exp.getDate(
```

Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK shared indexes // Always download // Download once // Don't show again // Configure... (5 minutes ago)   74:55   LF   UTF-8   4 spaces



```
public void onExpenseList() {
    System.out.println("Expense Listing");
    List<Expense> expList = repo.expList;
    for (int i = 0; i < expList.size(); i++) {
        Expense exp = expList.get(i);
        String catName = reportService.getCategoryNameById(exp.getCategoryId());
        System.out.println(
                (i + 1) + ". " + catName + ", " + exp.getAmount() + ", " + exp.getRemark() + ", " + exp.getDate());
    }
}

1 usage
public void onCategorizedExpenseList() {
    System.out.println("Categorized Expense List");
    Map<String, Float> resultMap = reportService.calculateCategoryTotal();
    Set<String> categories = resultMap.keySet();
    float netTotal = 0.0f;
    for (String categoryName : categories) {
        float catWisetTotal = resultMap.get(categoryName);
        netTotal = netTotal + catWisetTotal;
        System.out.println(categoryName + " : " + catWisetTotal);
    }
    System.out.println("<------------------------------------------------------->");
    System.out.println("Net Total : " + netTotal);
}

1 usage
public void onExit() { System.exit( status: 0); }

}
```

Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK shared indexes // Always download // Download once // Don't show again // Configure... (5 minutes ago)



Javaa 〉 src 〉 in 〉 ezeon 〉 Category

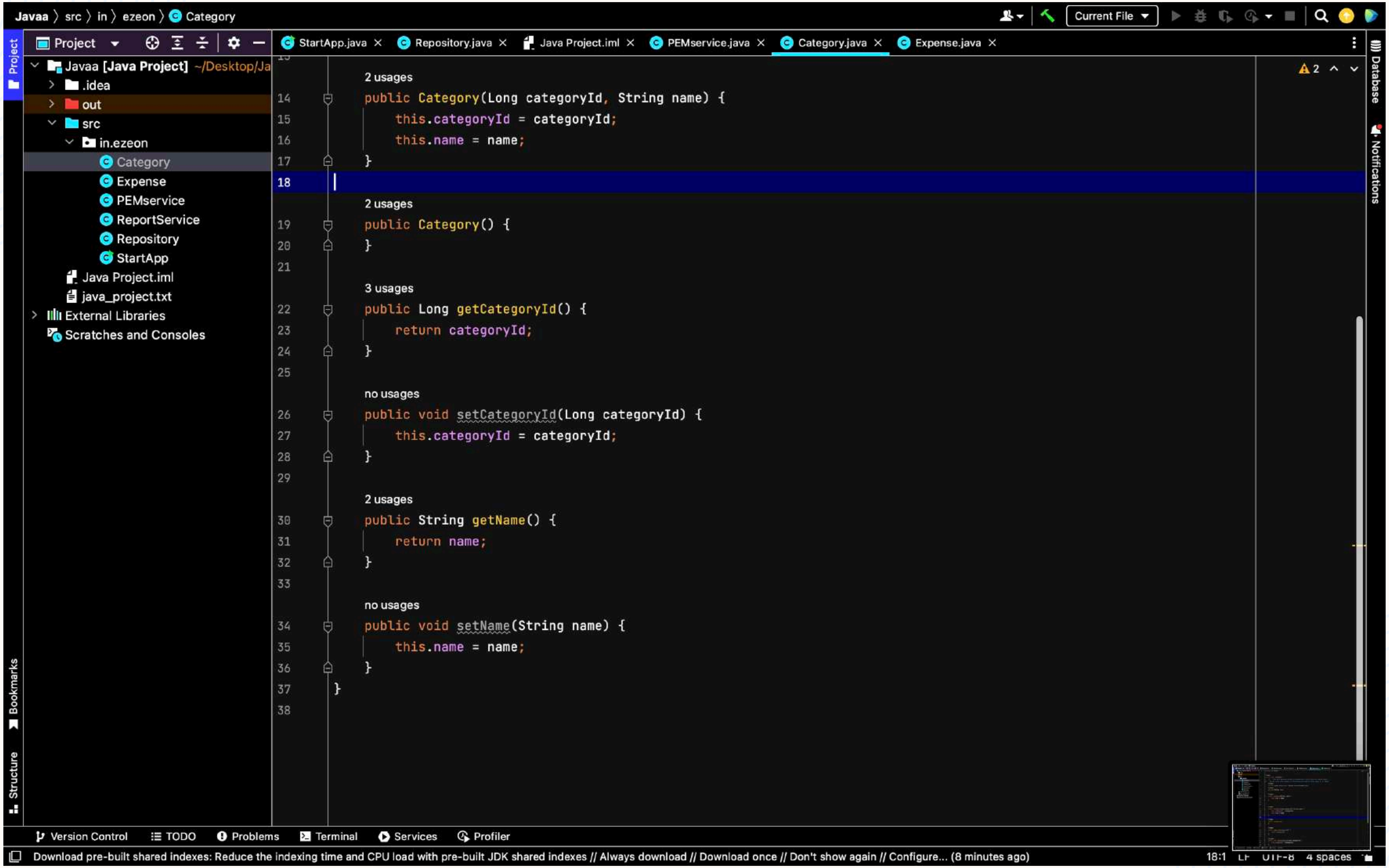
StartApp.java | Repository.java | Java Project.iml | PEMservice.java | Category.java | Expense.java

Project: Javaa [Java Project] ~/Desktop/Ja
- .idea
- out
- src
  - in.ezeon
    - Category
    - Expense
    - PEMservice
    - ReportService
    - Repository
    - StartApp
  - Java Project.iml
  - java_project.txt
- External Libraries
- Scratches and Consoles

```java
package in.ezeon;

7 usages
public class Category {
    //    This will generate unique id automatically every time you create object
//    We are using Long instead of long because we want to treat value as an objet.
    3 usages
    private Long categoryId = System.currentTimeMillis();
    4 usages
    private String name;

    3 usages
    public Category(String name) {
        this.name = name;
    }


    2 usages
    public Category(Long categoryId, String name) {
        this.categoryId = categoryId;
        this.name = name;
    }

    2 usages
    public Category() {
    }

    3 usages
    public Long getCategoryId() {
        return categoryId;
    }

    no usages
    public void setCategoryId(Long categoryId) {
        this.categoryId = categoryId;
    }
```

Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK shared indexes // Always download // Download once // Don't show again // Configure... (8 minutes ago)

18:1 LF UTF-8 4 spaces



Javaa 〉 src 〉 in 〉 ezeon 〉 Category

StartApp.java × Repository.java × Java Project.iml × PEMservice.java × Category.java × Expense.java ×

```
Javaa [Java Project] ~/Desktop/Ja
  .idea
  out
  src
    in.ezeon
      Category
      Expense
      PEMservice
      ReportService
      Repository
      StartApp
  Java Project.iml
  java_project.txt
External Libraries
Scratches and Consoles
```

```java
    2 usages
    public Category(Long categoryId, String name) {
        this.categoryId = categoryId;
        this.name = name;
    }

    2 usages
    public Category() {
    }

    3 usages
    public Long getCategoryId() {
        return categoryId;
    }

    no usages
    public void setCategoryId(Long categoryId) {
        this.categoryId = categoryId;
    }

    2 usages
    public String getName() {
        return name;
    }

    no usages
    public void setName(String name) {
        this.name = name;
    }
}
```

Version Control   TODO   Problems   Terminal   Services   Profiler

Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built JDK shared indexes // Always download // Download once // Don't show again // Configure... (8 minutes ago)   18:1   LF   UTF-8   4 spaces

EXPLANATION OF THE CODE.

This code defines a class called `ReportService` which has two methods - `calculateCategoryTotal()` and `getCategoryNameById()`.

The `calculateCategoryTotal()` method takes no arguments and returns a `Map<String, Float>`. This method is responsible for calculating the total expenses for each category. It does this by iterating over the list of expenses stored in the `Repository` instance and grouping the expenses by category name. It uses a `TreeMap` to store the results, where the key is the category name and the value is the total expense for that category. If an expense belongs to a category that is already present in the `TreeMap`, the expense amount is added to the existing total. Otherwise, a new entry is created for that category with the expense amount as the total.

The `getCategoryNameById()` method takes a `Long` argument representing a category ID and returns a `String` representing the name of the category. It searches for the category with the given ID in the list of categories stored in the `Repository` instance and returns the name of the category if found. If no category with the given ID is found, it returns `null`.

The `Repository` class and the `Expense` and `Category` classes that are used in this code are not provided in the given code snippet, so it is not possible to know what those classes contain and how they are implemented. However, based on the usage in this code, it can be inferred that the `Repository` class is a central data storage mechanism that contains lists of expenses and categories. .

It provides a menu-driven interface for users to manage their personal expenses. The program defines a class called "PEMservice", which contains various methods to handle the different menu options.
The program uses a Repository class to store and manage data related to categories and expenses. It also uses a ReportService class to generate reports related to expenses.
The main method of the program is "showMenu", which displays the main menu and handles user input. It uses a while loop to repeatedly display the menu and wait for user input. Based on the user's choice, the program calls the appropriate method to perform the requested action.

The program provides the following menu options:

1. Add Category - Allows the user to add a new category to the system.
2. Category List - Displays a list of all categories in the system.
3. Expense Entry - Allows the user to add a new expense to the system.
4. Expense List - Displays a list of all expenses in the system.
5. Categorized Expense List - Displays a list of expenses grouped by category.
6. Exit - Exits the program.

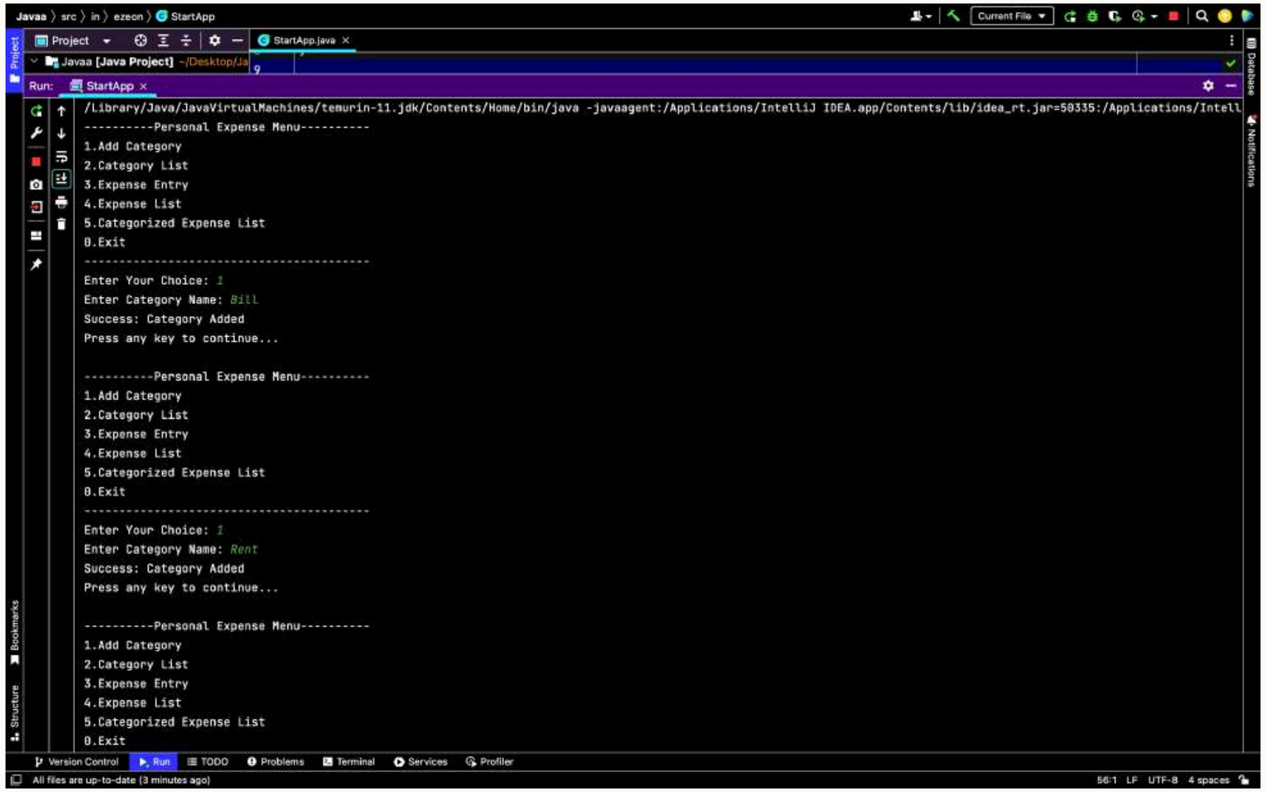
Each menu option has a corresponding method in the PEMservice class that handles the action associated with that option. For example, the "onAddCategory" method handles the "Add Category" option by prompting the user for a category name and adding it to the repository.

The program also includes helper methods, such as "printMenu" and "pressAnyKeyToContinue", which display messages and wait for user input.

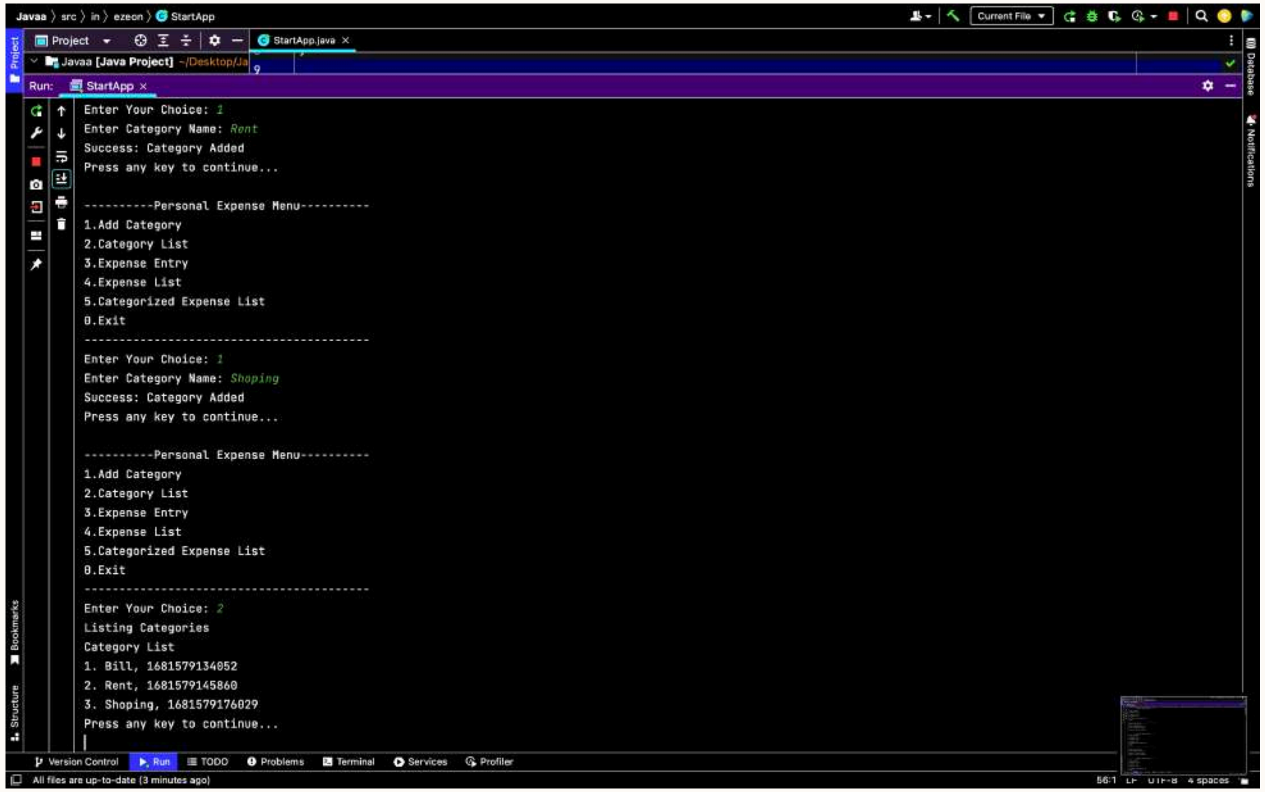
Overall, this program provides a simple but effective interface for managing personal expenses.

SCREENSHOT OF OUTPUTS OF ROCK PAPER AND SCISSOR GAME

Adding Categories and Listing Categories



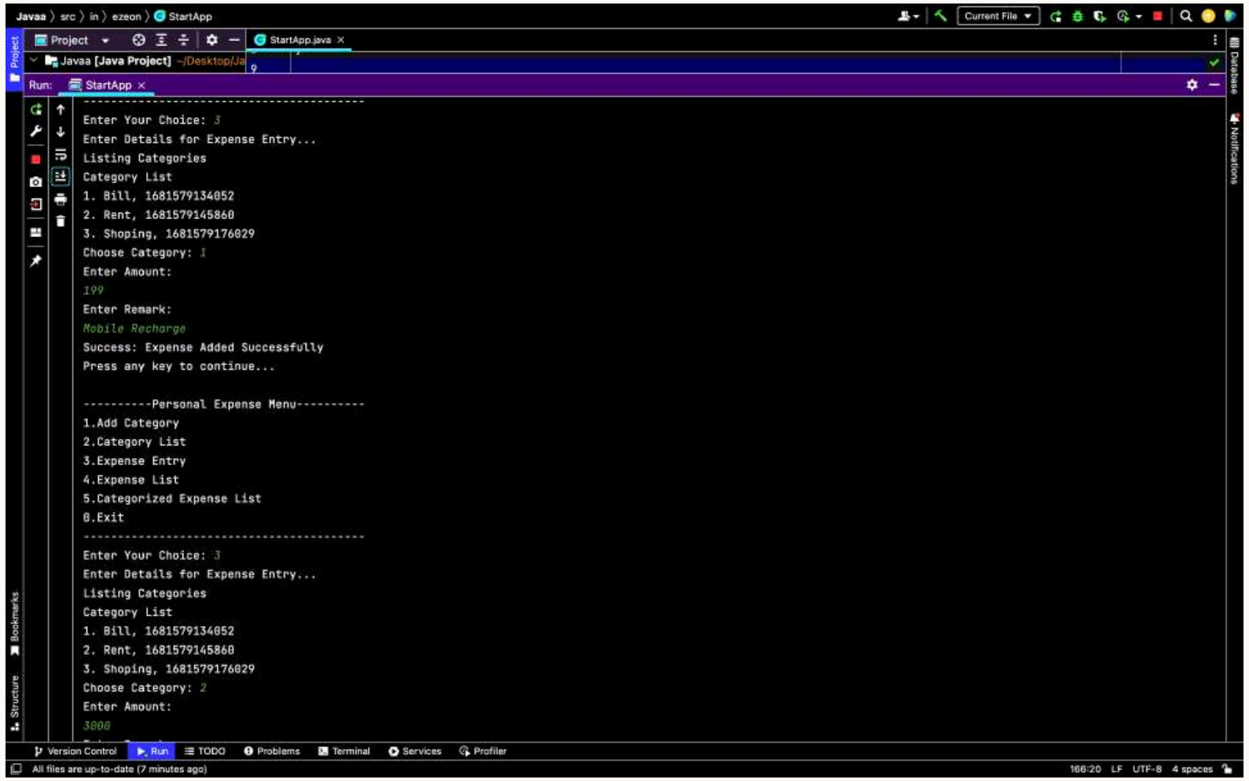
```
/Library/Java/JavaVirtualMachines/temurin-11.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=50335:/Applications/Intell
----------Personal Expense Menu----------
1.Add Category
2.Category List
3.Expense Entry
4.Expense List
5.Categorized Expense List
0.Exit
-----------------------------------------
Enter Your Choice: 1
Enter Category Name: Bill
Success: Category Added
Press any key to continue...

----------Personal Expense Menu----------
1.Add Category
2.Category List
3.Expense Entry
4.Expense List
5.Categorized Expense List
0.Exit
-----------------------------------------
Enter Your Choice: 1
Enter Category Name: Rent
Success: Category Added
Press any key to continue...

----------Personal Expense Menu----------
1.Add Category
2.Category List
3.Expense Entry
4.Expense List
5.Categorized Expense List
0.Exit
```



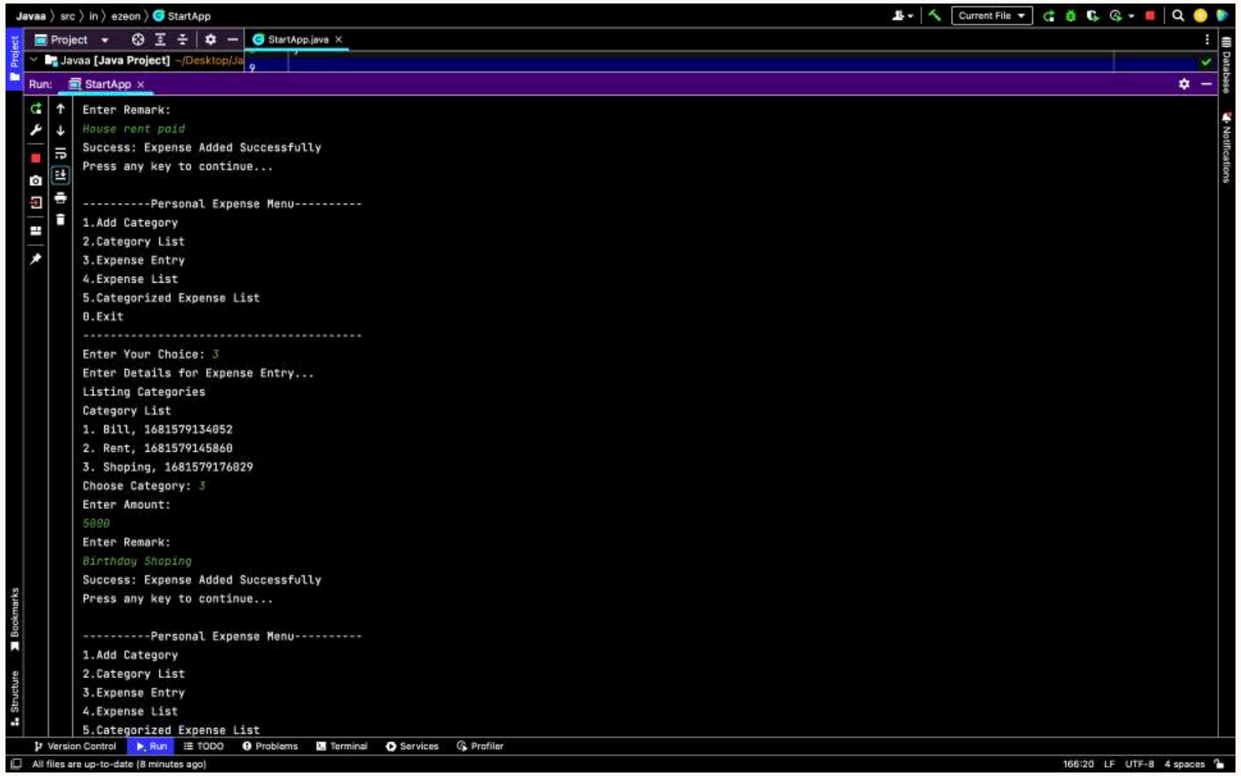
```
Enter Your Choice: 1
Enter Category Name: Rent
Success: Category Added
Press any key to continue...

----------Personal Expense Menu----------
1.Add Category
2.Category List
3.Expense Entry
4.Expense List
5.Categorized Expense List
0.Exit
-----------------------------------------
Enter Your Choice: 1
Enter Category Name: Shoping
Success: Category Added
Press any key to continue...

----------Personal Expense Menu----------
1.Add Category
2.Category List
3.Expense Entry
4.Expense List
5.Categorized Expense List
0.Exit
-----------------------------------------
Enter Your Choice: 2
Listing Categories
Category List
1. Bill, 1681579134052
2. Rent, 1681579145860
3. Shoping, 1681579176029
Press any key to continue...
```

SCREENSHOT OF OUTPUTS OF ROCK PAPER AND SCISSORS GAME

Adding Expenses Category Wise



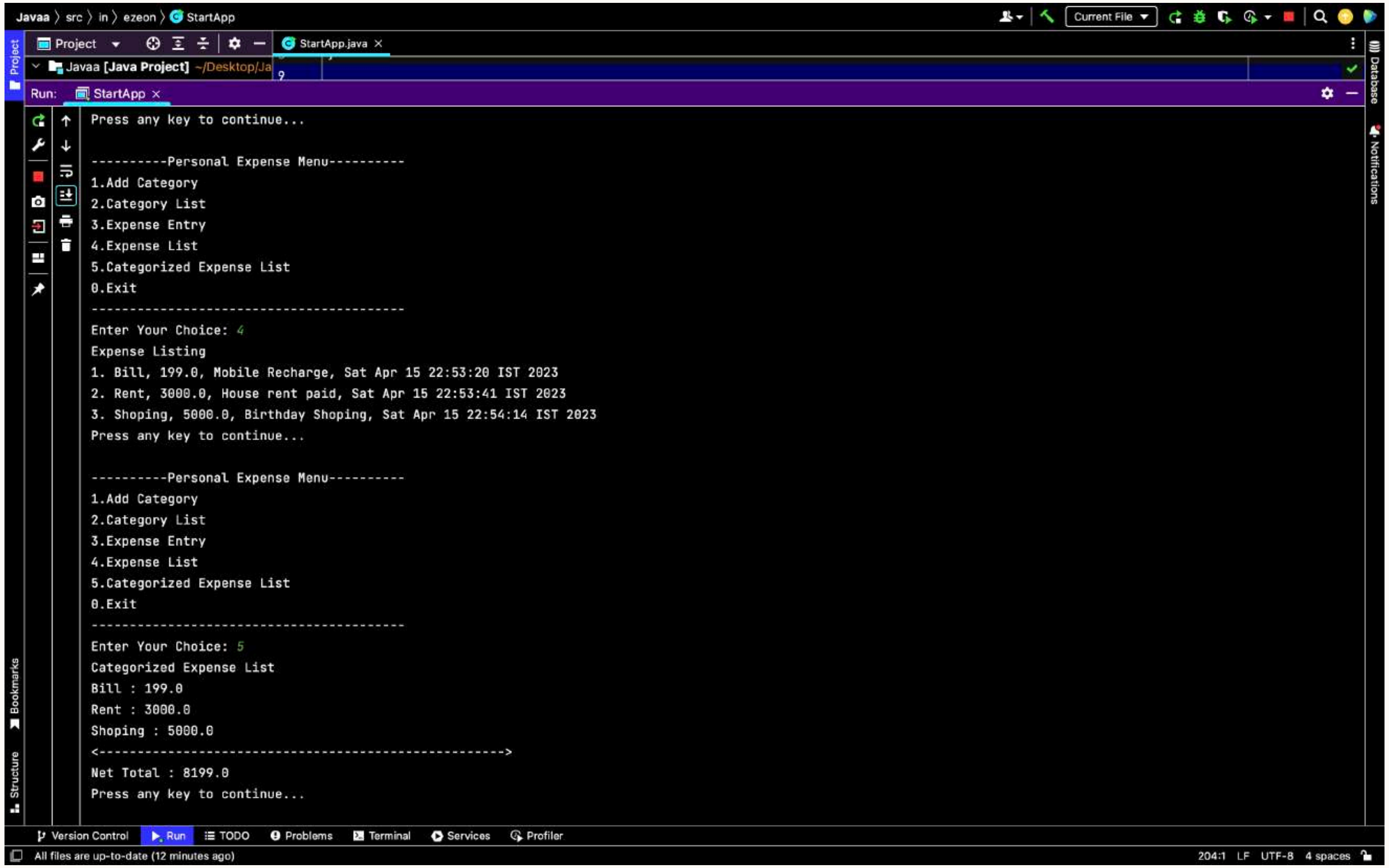
```
----------------------------------------
Enter Your Choice: 3
Enter Details for Expense Entry...
Listing Categories
Category List
1. Bill, 1681579134052
2. Rent, 1681579145860
3. Shoping, 1681579176029
Choose Category: 1
Enter Amount:
199
Enter Remark:
Mobile Recharge
Success: Expense Added Successfully
Press any key to continue...

----------Personal Expense Menu----------
1.Add Category
2.Category List
3.Expense Entry
4.Expense List
5.Categorized Expense List
0.Exit
----------------------------------------
Enter Your Choice: 3
Enter Details for Expense Entry...
Listing Categories
Category List
1. Bill, 1681579134052
2. Rent, 1681579145860
3. Shoping, 1681579176029
Choose Category: 2
Enter Amount:
3000
```



```
Enter Remark:
House rent paid
Success: Expense Added Successfully
Press any key to continue...

----------Personal Expense Menu----------
1.Add Category
2.Category List
3.Expense Entry
4.Expense List
5.Categorized Expense List
0.Exit
----------------------------------------
Enter Your Choice: 3
Enter Details for Expense Entry...
Listing Categories
Category List
1. Bill, 1681579134052
2. Rent, 1681579145860
3. Shoping, 1681579176029
Choose Category: 3
Enter Amount:
5000
Enter Remark:
Birthday Shoping
Success: Expense Added Successfully
Press any key to continue...

----------Personal Expense Menu----------
1.Add Category
2.Category List
3.Expense Entry
4.Expense List
5.Categorized Expense List
```

SCREENSHOT OF OUTPUTS OF ROCK PAPER AND SCISSORS GAME

Expense List And Categorised Expense List



```
Press any key to continue...

----------Personal Expense Menu----------
1.Add Category
2.Category List
3.Expense Entry
4.Expense List
5.Categorized Expense List
0.Exit
-----------------------------------------
Enter Your Choice: 4
Expense Listing
1. Bill, 199.0, Mobile Recharge, Sat Apr 15 22:53:20 IST 2023
2. Rent, 3000.0, House rent paid, Sat Apr 15 22:53:41 IST 2023
3. Shoping, 5000.0, Birthday Shoping, Sat Apr 15 22:54:14 IST 2023
Press any key to continue...

----------Personal Expense Menu----------
1.Add Category
2.Category List
3.Expense Entry
4.Expense List
5.Categorized Expense List
0.Exit
-----------------------------------------
Enter Your Choice: 5
Categorized Expense List
Bill : 199.0
Rent : 3000.0
Shoping : 5000.0
<---------------------------------------------------->
Net Total : 8199.0
Press any key to continue...
```

CONCLUSION

In conclusion, the personal expense manager project is a valuable software application that helps individuals manage their personal finances. With its features such as budget tracking, expense categorisation, reporting, bill reminders, investment tracking, and debt management tools, the platform provides a comprehensive solution to help users gain a better understanding of their spending habits and make informed financial decisions.
By using this project, individuals can easily input their income and expenses, set budgets, and track their progress toward their financial goals. The intuitive design of the platform ensures that users can navigate it easily and quickly find the information they need. Moreover, the project also helps individuals to identify areas where they may be overspending and adjust their budget accordingly. Overall, the personal expense manager project is an excellent tool for anyone looking to take control of their finances and make better financial decisions. It provides a complete solution for managing personal finances and helps users achieve their long-term financial goals.

REFERENCES

- Geek for Geeks
- Google.com
- Wikipedia
- Stechies.com
- Stack Overflow

THANK YOU