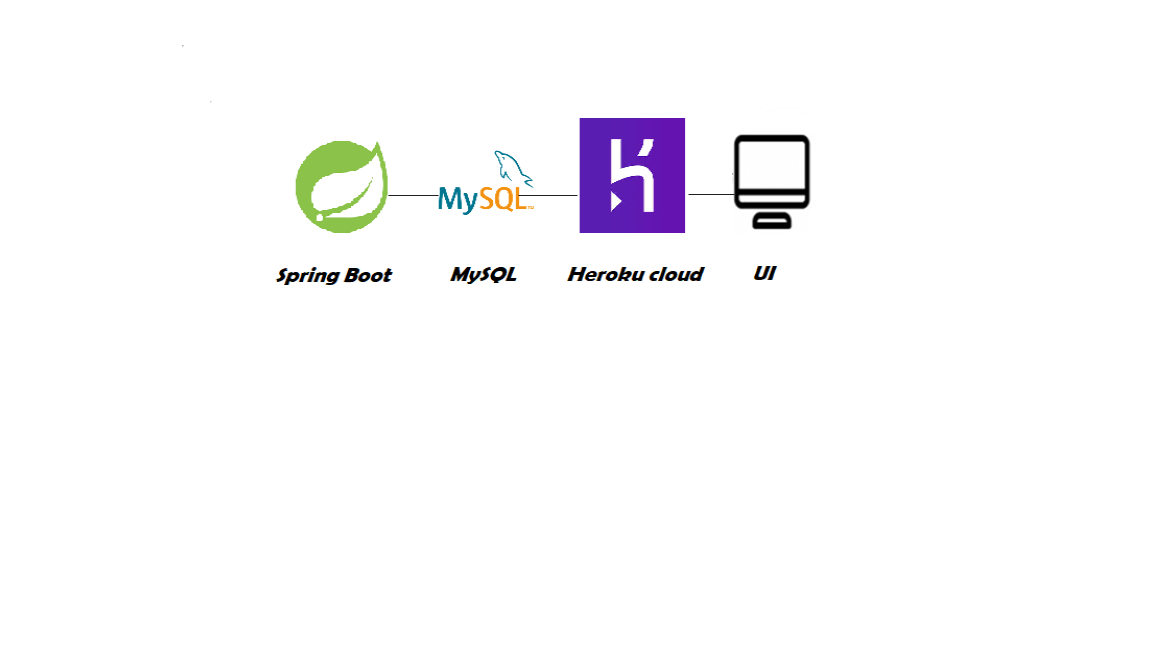
**Expense Tracker App**

**-SANJEEVAN K S**

In today’s busy and expensive life we are in a great rush to make money. But at the end of the month, we broke off. As we are unknowingly spending money on little and unwanted things. So, we have come over with the idea to track our earnings. The daily expense tracker aims to help everyone who is planning to know their expenses and save from it. An expense tracker is a spring boot app that users can access from URL and update their daily expenses so that they are well known for their expenses. Here users can choose categories for expense types like food, clothing, rent, and bills where they have to enter the money that has been spent and also can add some information in additional information to specify the expense. Users will be able to see graphs of expenses. Users can save those graphs for their own reference.

**Architecture:**



**Learning Outcomes:**

By the end of this project:

* You’ll be able to work with Spring boot, remote SQL, and Heroku Cloud deployment.
* Build a spring boot application that will take the user inputs, update the remote SQL, and show graphs for an expense.

**Project Workflow:**

* The user interacts with the application.
* Register by giving the details as a user.
* The database will have all the details whenever the user will log in to the system they will be able to see their expenses and its graphical representation.

**Install Spring IDE**

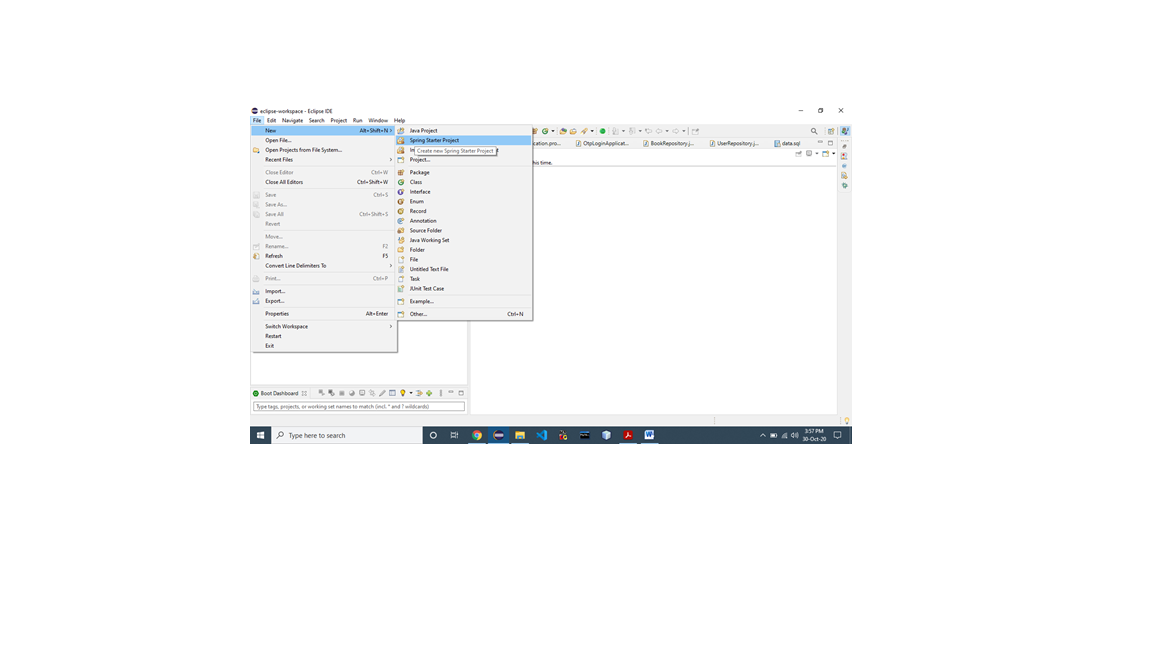
Install Spring IDE (STS/eclipse IDE is Ideal to complete this project)

* To install **STS IDE**, please refer to [STS for spring boot IDE Installation Steps](https://www.youtube.com/watch?v=cDqlT7O8H0Q&feature=youtu.be)
* To install **Eclipse IDE**, please refer to [eclipse IDE Installation steps](https://www.youtube.com/watch?v=xqS31gwunp0&feature=youtu.be)

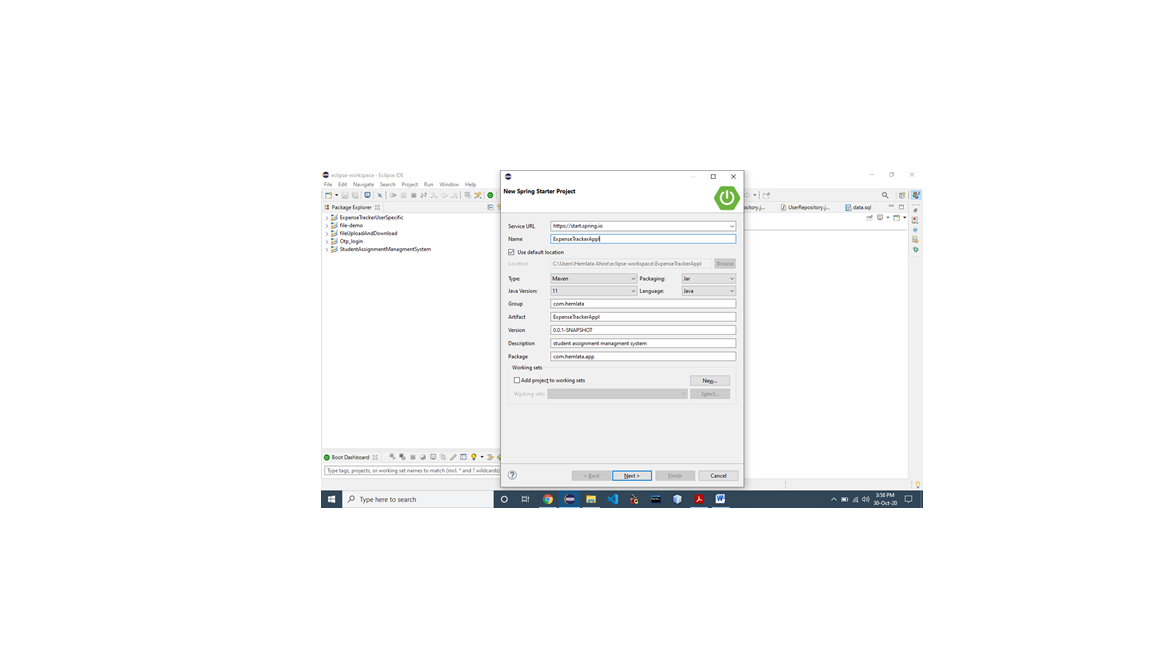
**Method 1**

**Method 1:** Inside the IDLE

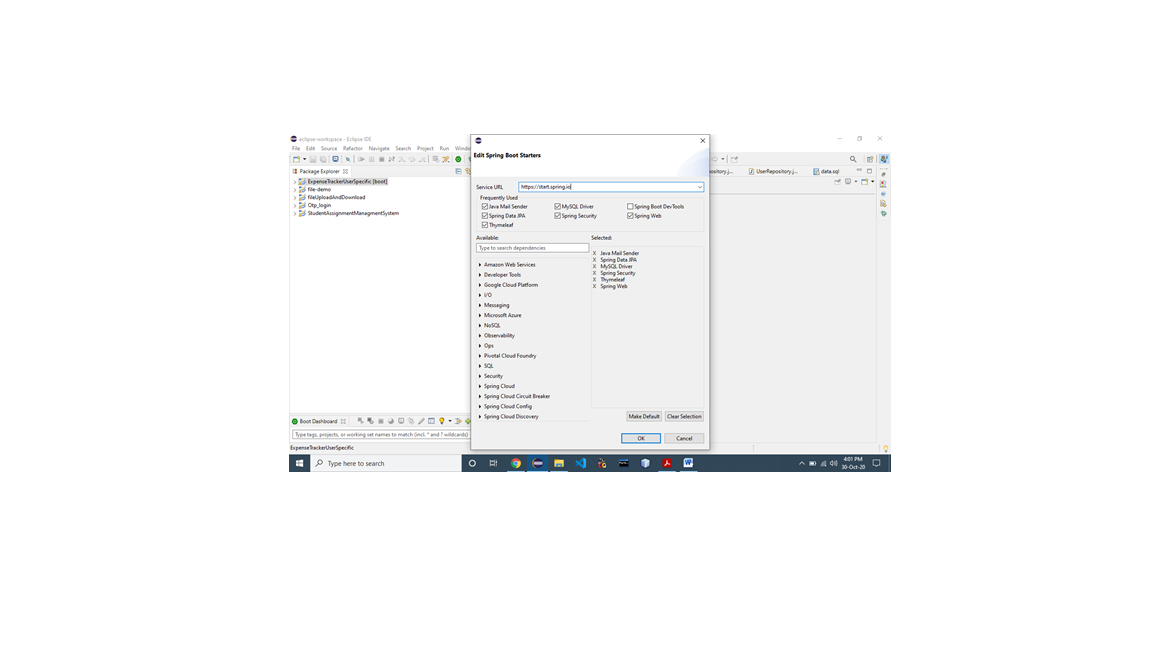
* Open eclipse IDLE
* Click on file menu => New => Project (as shown in below image.)



* From the newly popped up window select “spring starter project” and click on “Next”.
* Enter details of a project to be created, then click on “Next”



* Choose required dependencies (shown below) and click on “finish”.

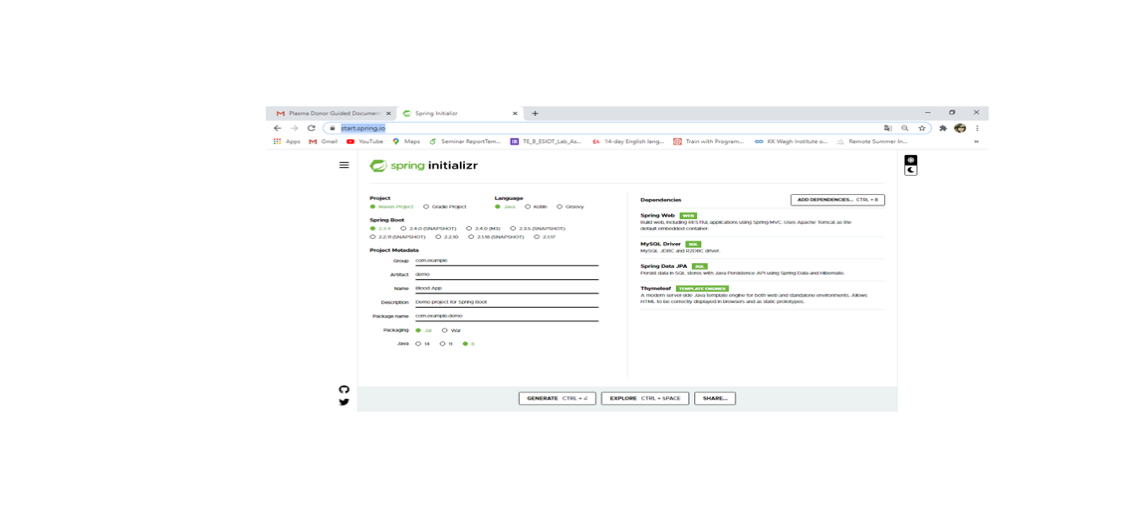


**Method 2**

**Method 2:** Using spring initializer

To create a project using spring initialize follow the below steps:

* Go to  [Spring Initialize (https://start.spring.io/)](https://start.spring.io/)
* Add **Name** for application And choose **required dependencies**



The above steps will generate a zip file you need to download and import in

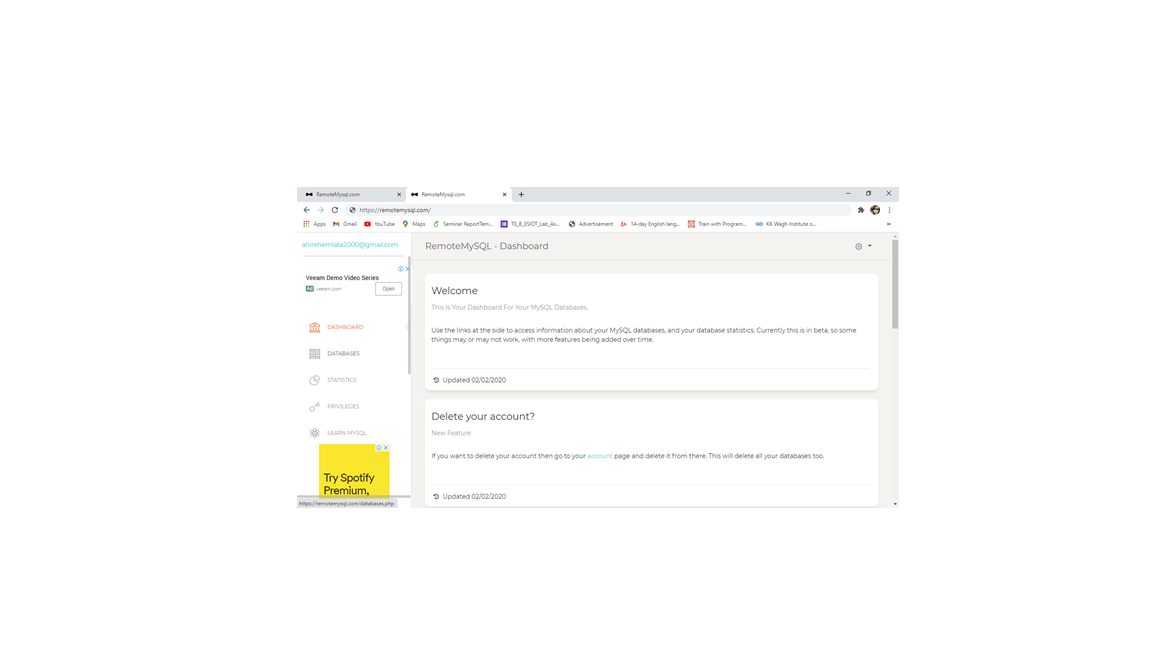
**Create A Database**

Goto<https://remotemysql.com/>

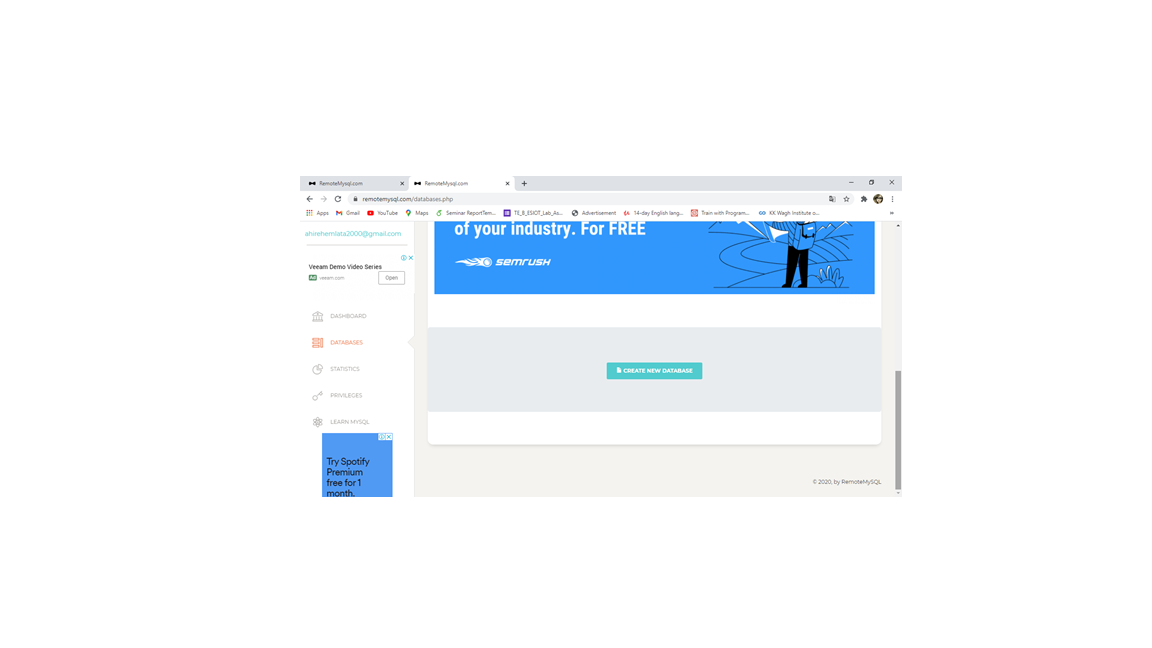
Signup and login for free

After login follow the below steps

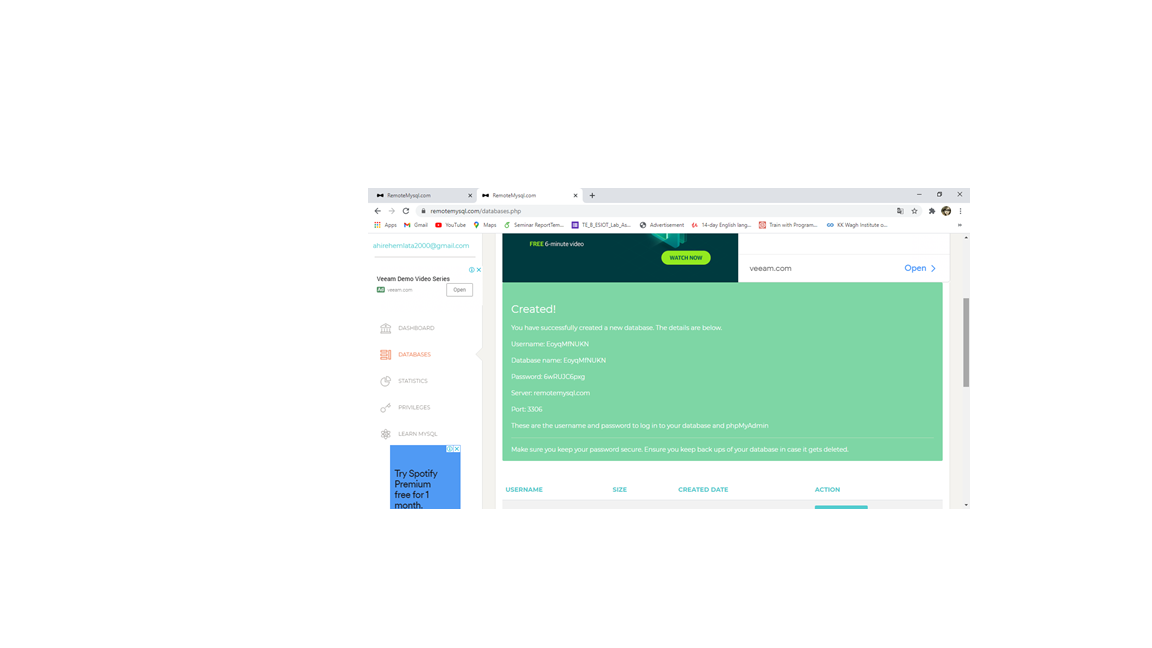
**Step 1: Goto Databases tab**



**Step 2:** Scroll down to the end of the page. And click on create a new database



**Step 3: Copy these credentials and save it for further connection.**



**Create A Model Class**

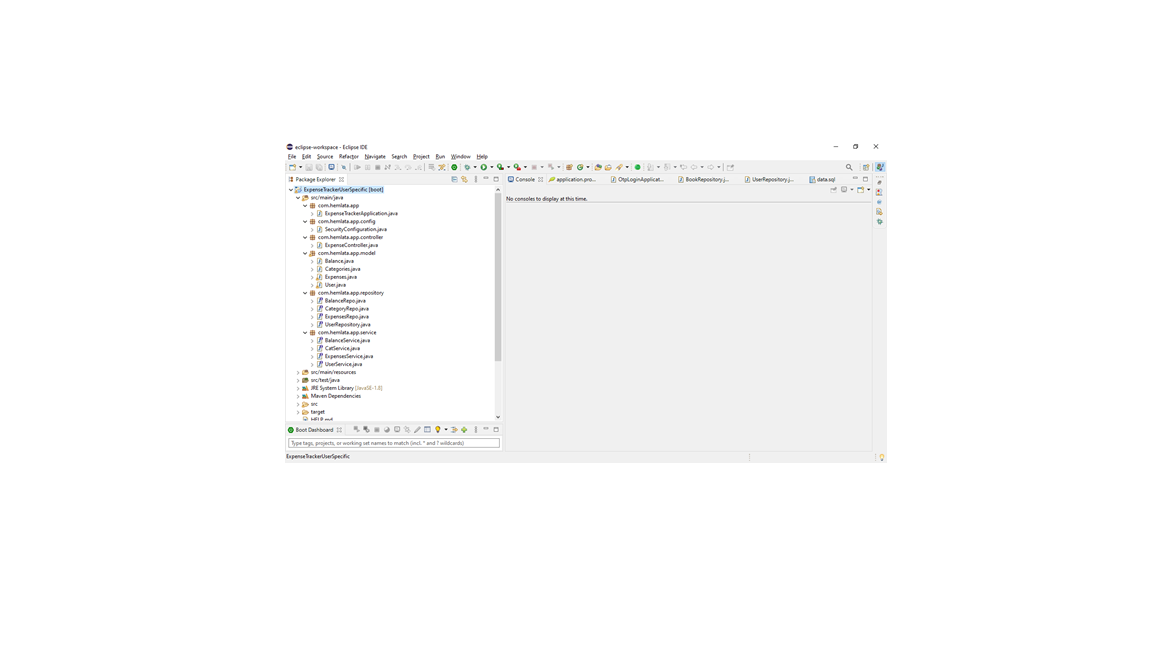
Create a Model class to store. The model class is responsible to create or define the schema of the table to be created.

Here we need to models

1. User – user table will store details about users who had registered in the system

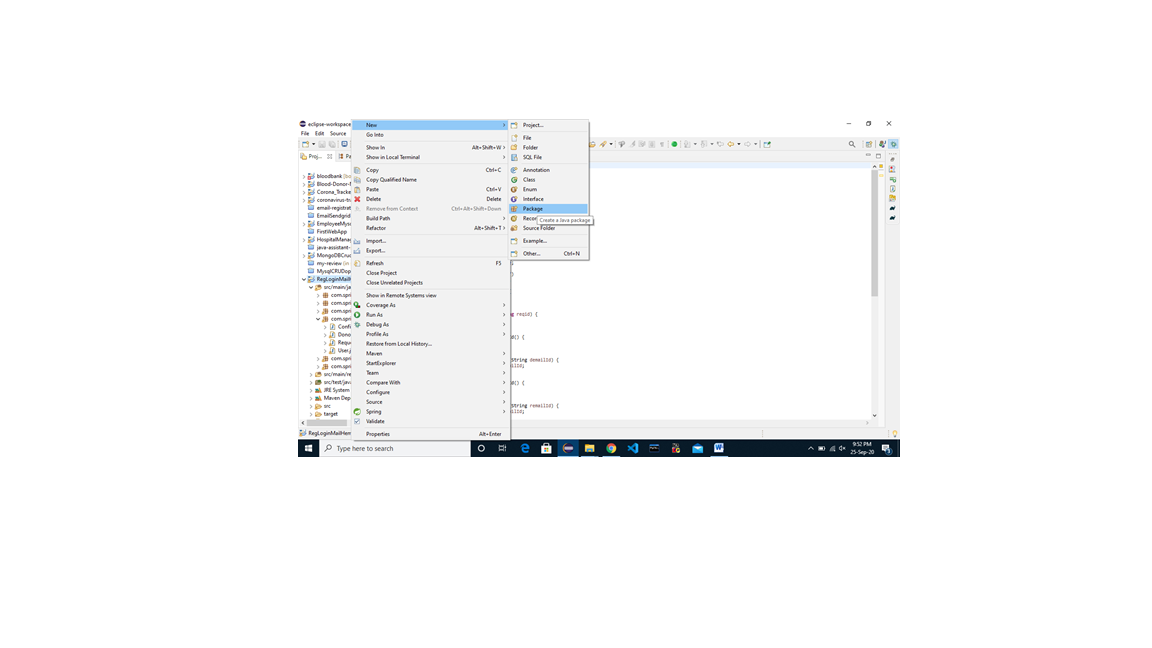
2. Expenses – expenses table will store details about all the expenses made by a particular user.

3. Balance – balance tables consist of the balance that you can spend.



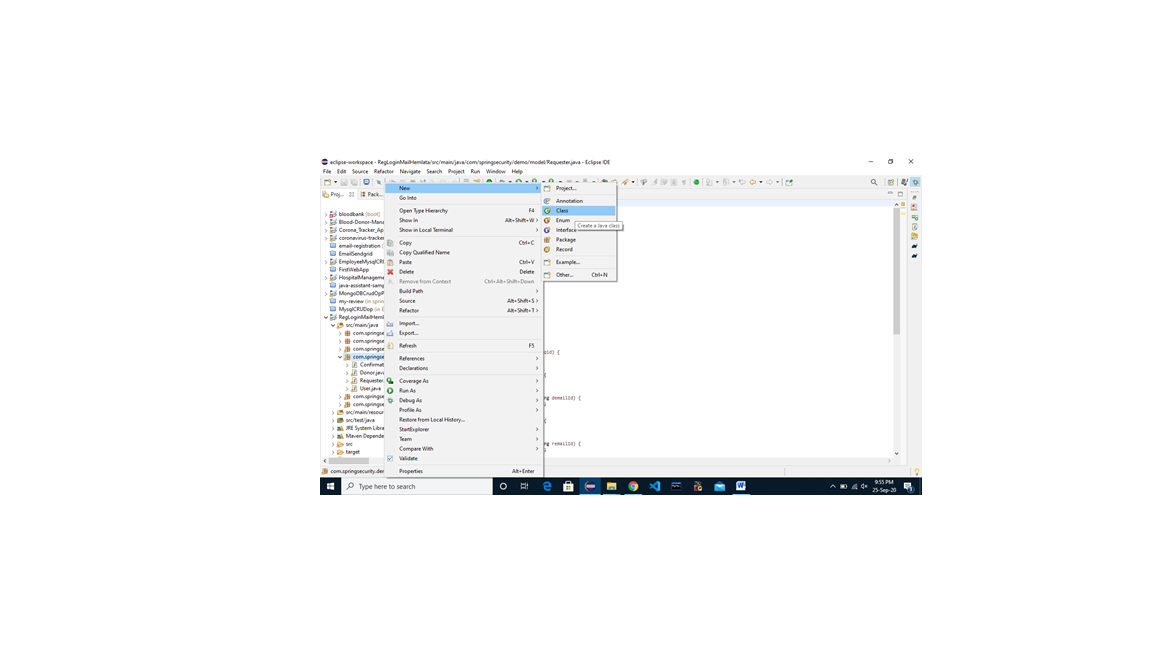
**Step 1:** Create a model package

To create model package Right click on the project => select “New” => choose package => Enter name of the package (e.g com.hemlata.app.model)



**Step 2:** Create User, Expenses, Balance class under the model package.

To create class under model package: Right click on the model package => select “New” => Choose option “class” => enter name of the class as User.



Repeat above step to create Expenses, Balance class

Refer Code [From Here](https://github.com/smartbridgesip/SpringBoot/tree/main/exp-tracker-hem/src/main/java/com/hemlata/app/model)

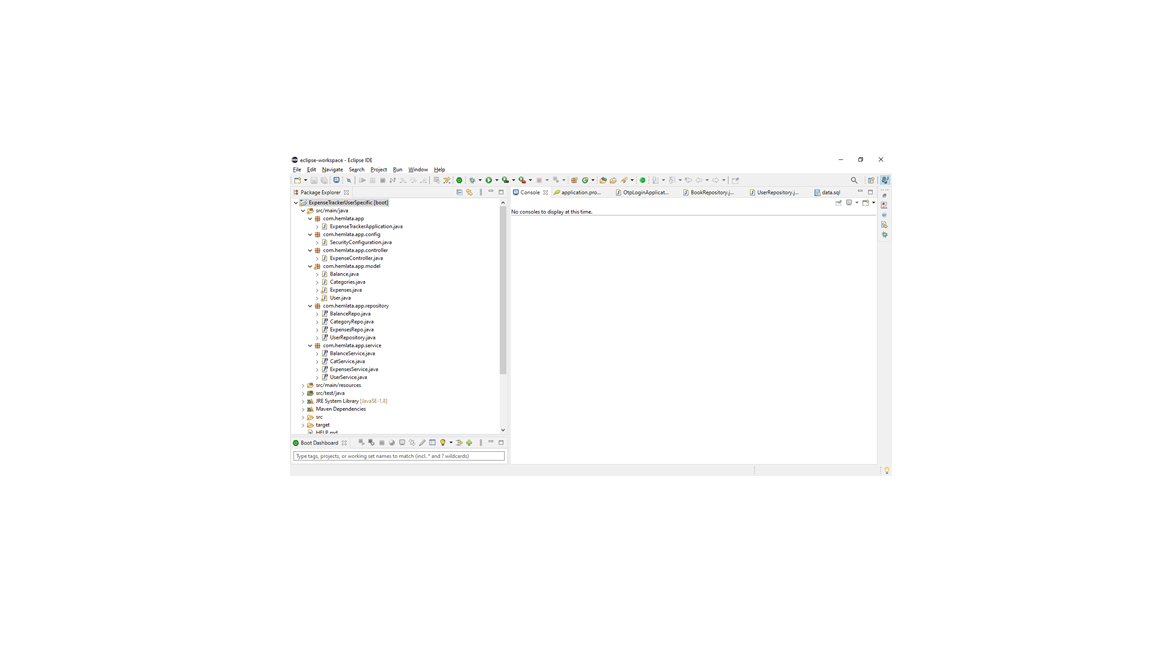
**Create Repository Package**

Repositories are responsible to define a set of operations to be carried out on the tables/models created in earlier steps.

**Step 1:** Right click on the project => select “New” => choose “package” => Enter package name (e.g com.hemlata.app.repositories)

**Step 2:** Right click on the package => choose “New” => select “class” option => Enter name of the class (e.g **UserRepo**)

**Step 3:** Repeat the above step to create ExpenseRepo class and BalanceRepo.



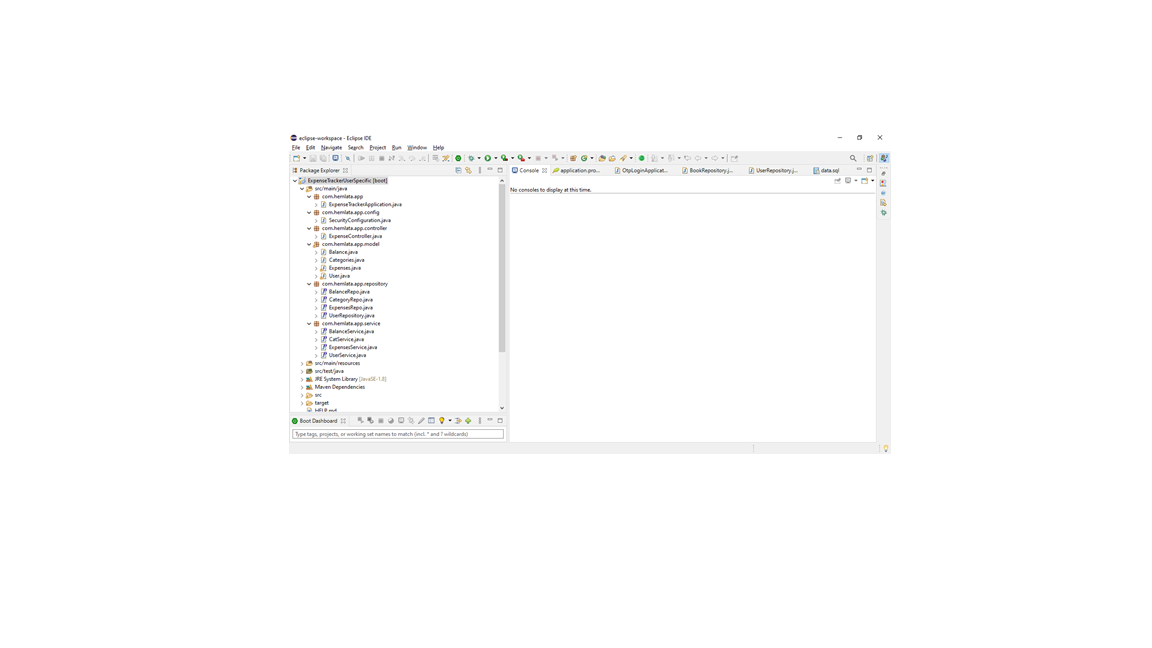
Refer Code [From Here](https://github.com/smartbridgesip/SpringBoot/tree/main/exp-tracker-hem/src/main/java/com/hemlata/app/repository)

**Create Service Package**

This package will contain the classes which define how the services will be delivered to the user.

**Step 1:** Right click on the project => select “New” => choose “package” => Enter package name (e.g com.hemlata.app.services)

**Step 2:** Right click on the package => choose “New” => select “class” option => Enter name of the class (e.g **UserService**)



Repeat the above step to create ExpenseService class and BalanceService class.

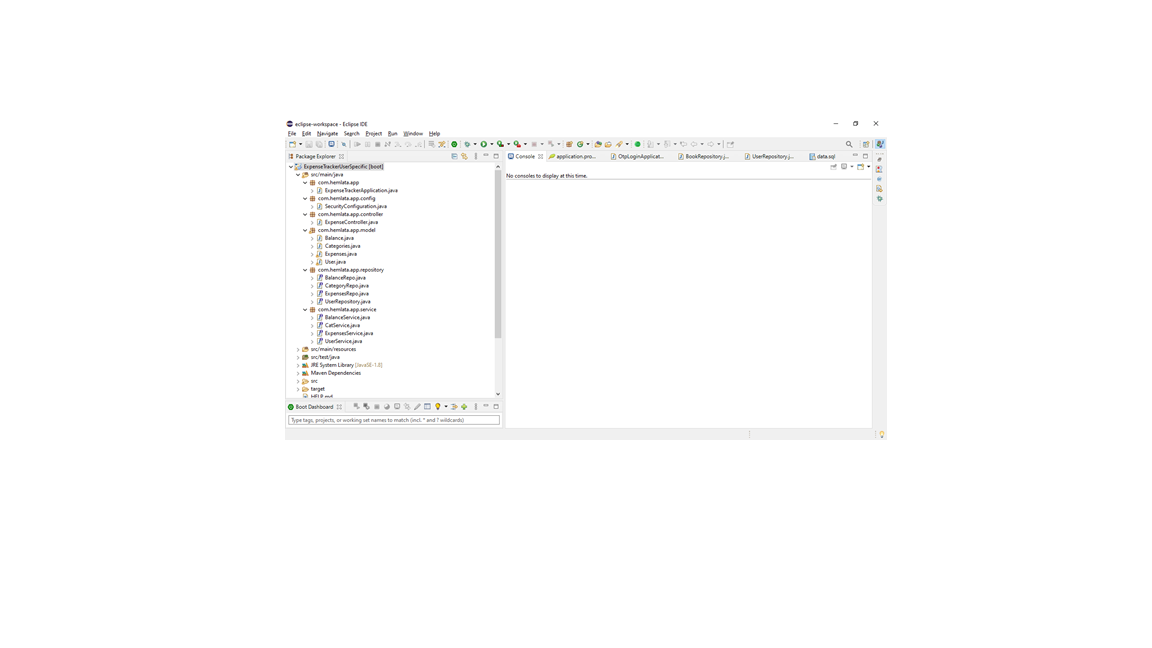
Refer Code [From Here](https://github.com/smartbridgesip/SpringBoot/tree/main/exp-tracker-hem/src/main/java/com/hemlata/app/service)

**Create Controller Package**

The controller class is responsible to handle all API requests like GET, POST, etc.

**Step 1:** Right click on the project => select “New” => choose “package” => Enter package name (e.g com.hemlata.app.controller)

**Step 2:** Right click on the package => choose “New” => select “class” option => Enter name of the class (e.g ExpenseController)

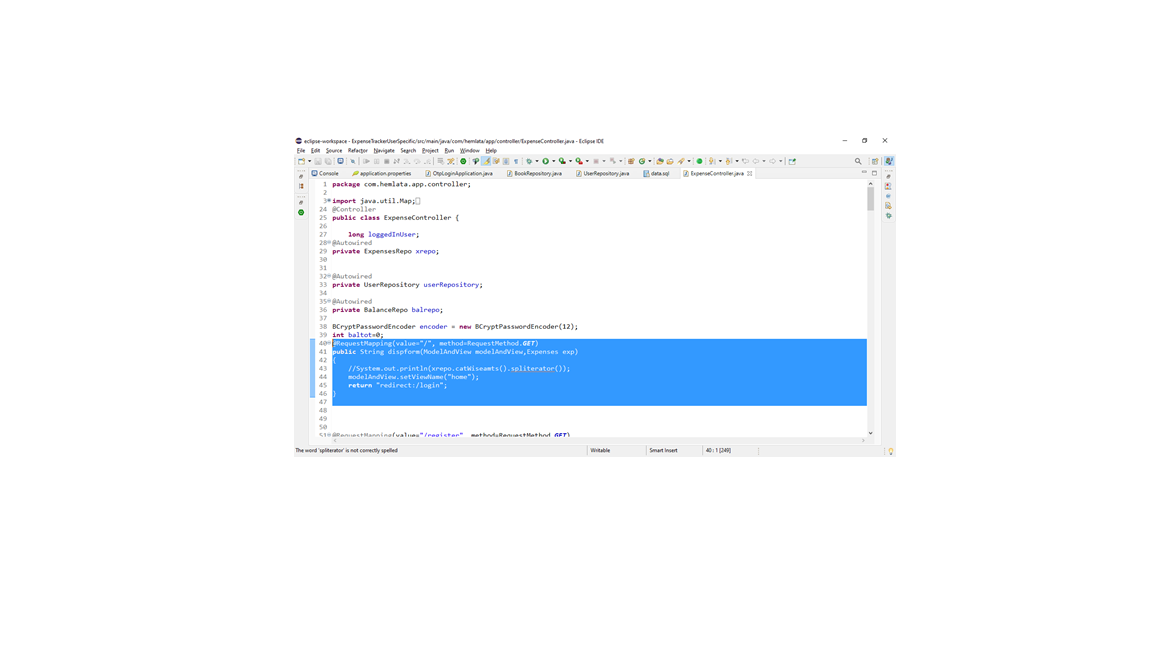


**Step 3:** write code to handle **‘/’ and ‘/home’** API request

**@RequestMapping** annotation accepts two parameters URL and Method type. Whenever a request for an URL is received controller will execute the above block of code.

The above code simply returns the HTML page or a UI whenever **the “/”** URL is requested

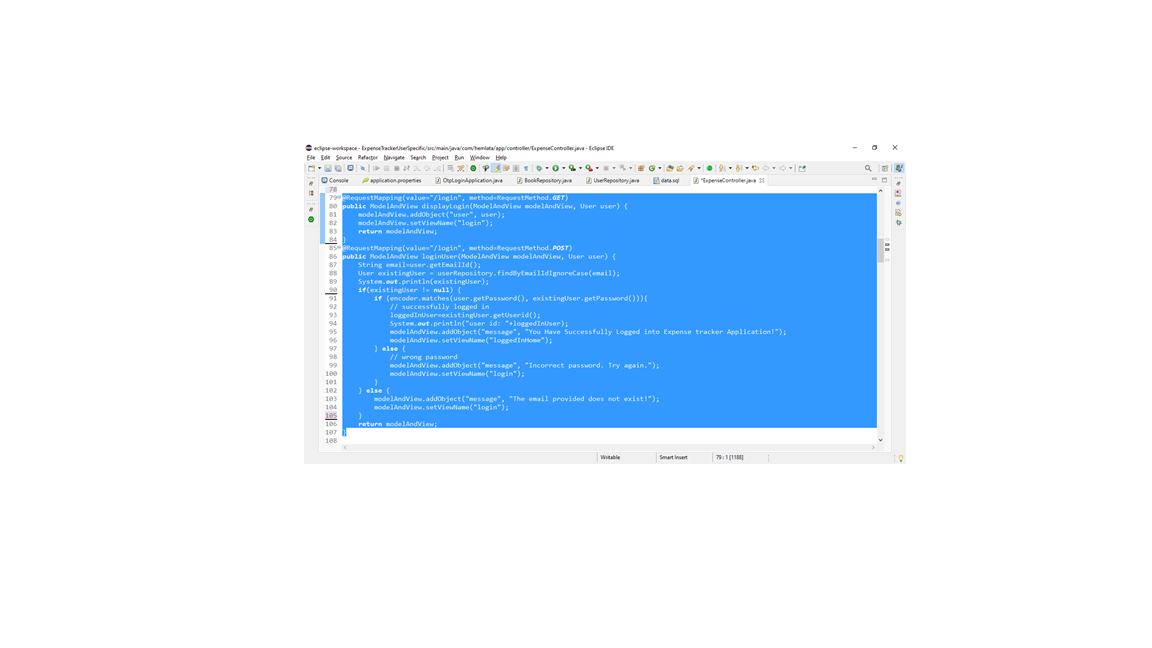
**Step 3:** write code to handle **‘/register’** API request



Whenever users will hit the register button this API will be invoked. And registration.html file will be displayed as a UI.

If the request method is posted then check if a user does not exist then create a user object and add to the database. Then display successRegistration.

**Step 4:** write code to handle **‘/login’** API request



Here if the request method is GET then display the login.html page.

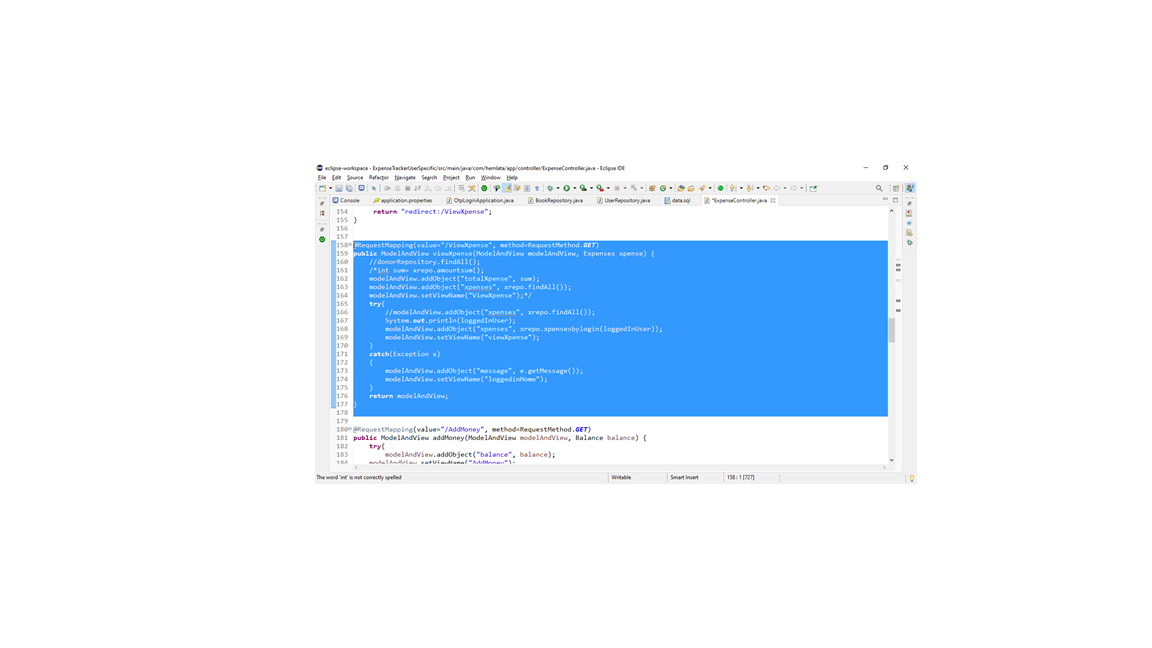
If the request method is Post the check the entered credentials if matches then show login success page otherwise ask to re-enter credentials.

**Step 5:** write code to handle **‘addXpense** API request



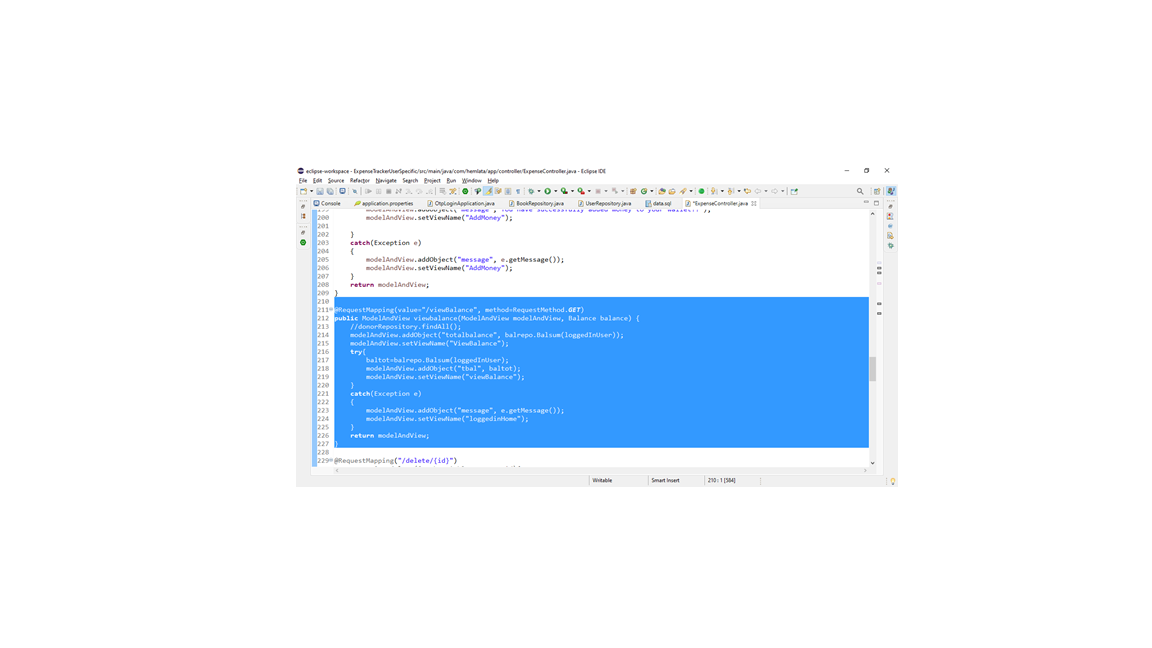
Whenever addXpense new Expense object is created and saved into database.

**Step 6:** write code to handle **‘viewXpense** API request

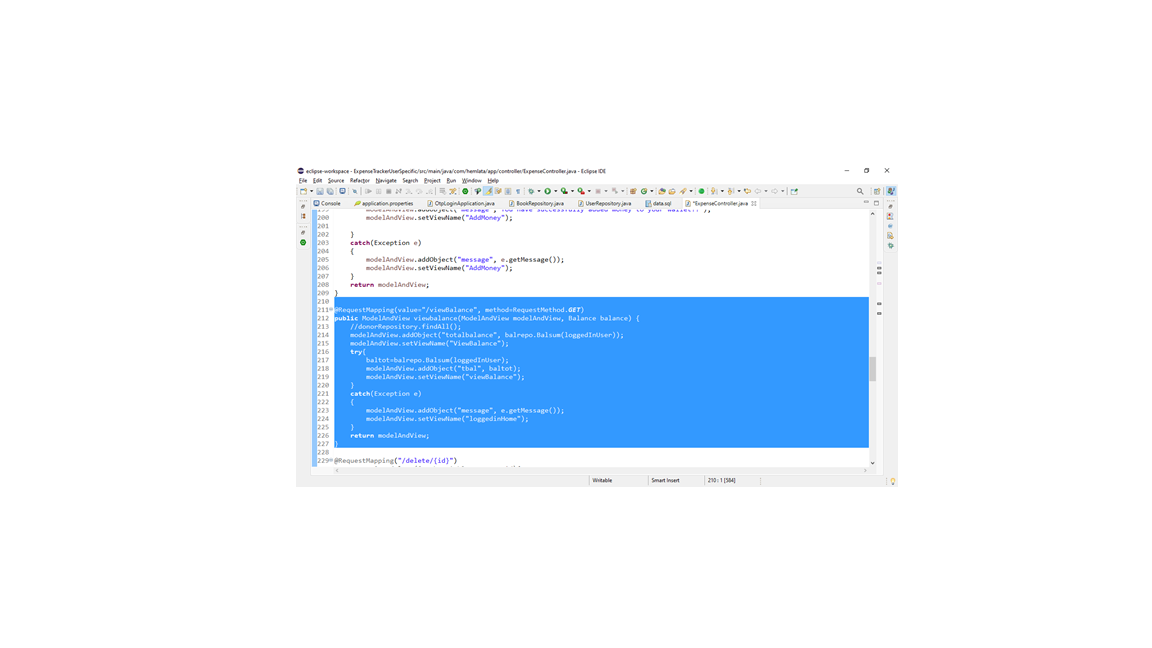


If such URL is requested a list of expenses of a particular user will be fetched and displayed.

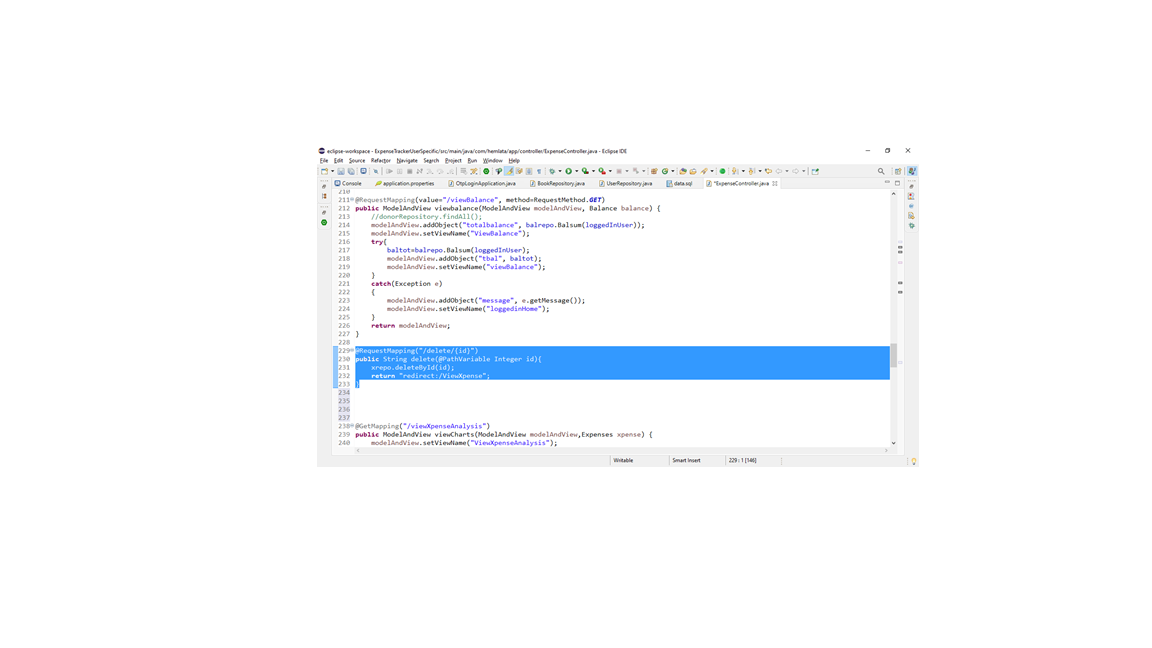
**Step 7:** write code to handle **‘addMoney’** API request



**Step 7:** write code to handle **‘viewMoney’** API request



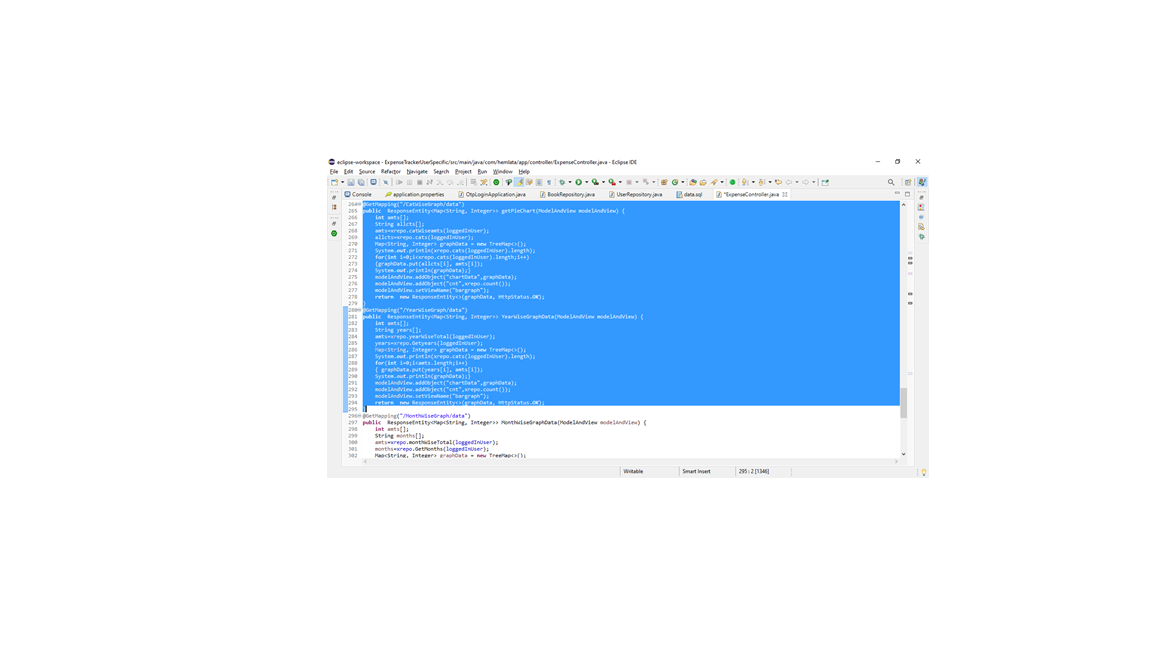
**Step 8:** write code to handle **‘DeleteXpense’** API request

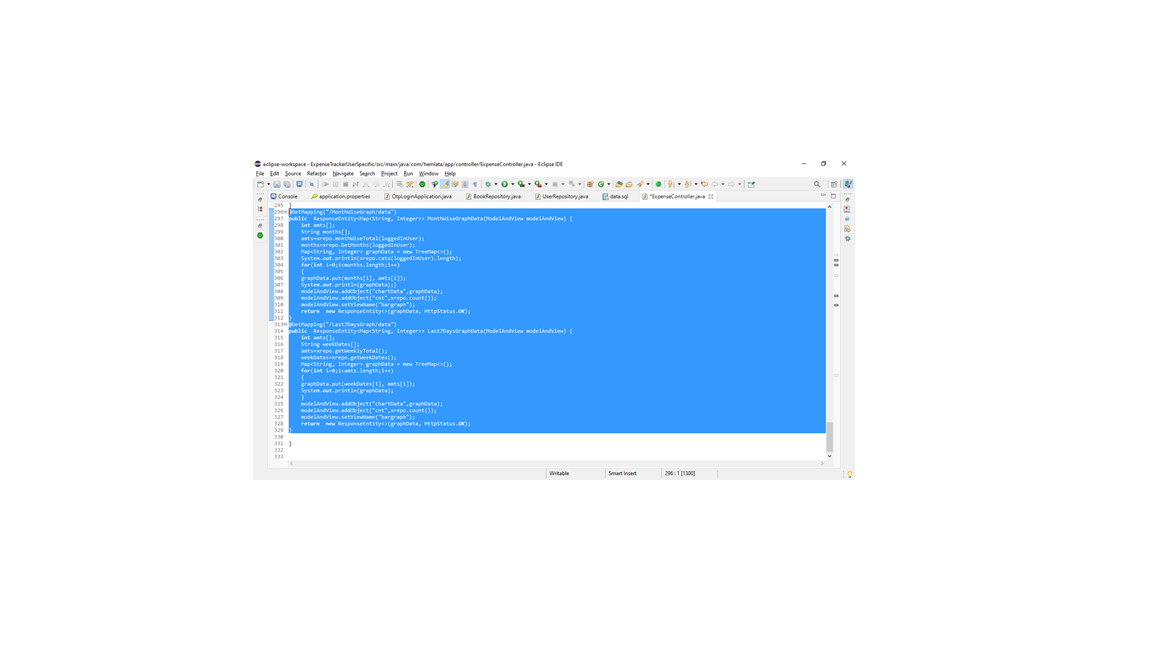


**Step 9:** write code to handle **“ViewAnalysis”** API request

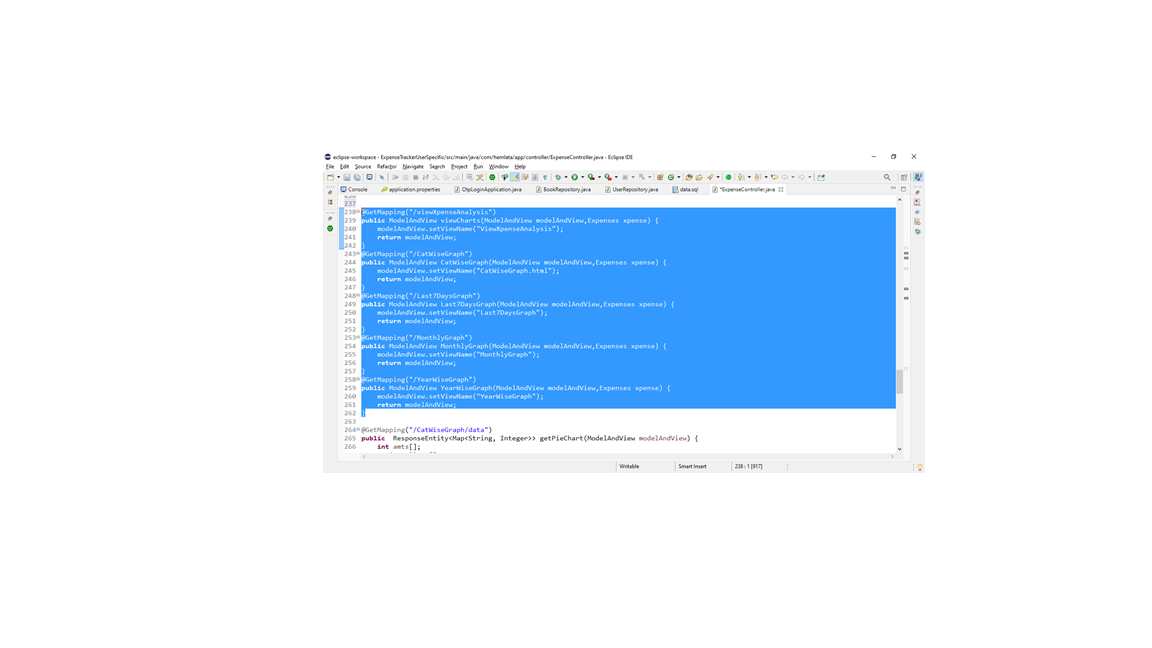
View analysis api is able to show different graphs based on data available in the database. To generate graphs we need to create an api to fetch data for graphs and then generate the graphs from fetched data.

**SubStep 1**: To create an api to fetch data.





**SubStep 2**: create an API to Display generated graph from fetched data



To see full source code for controller [Click Here](https://github.com/smartbridgesip/SpringBoot/tree/main/exp-tracker-hem/src/main/java/com/hemlata/app/controller)

