

java CLOUD ASSESSMENT TOOL



March 13, 2019

capgemini

Capgemini Pvt. Limited Mumbai

Table Index

[Scope: 2](#_Toc3384509)

[Requirements: 2](#_Toc3384510)

[Java requirement: 2](#_Toc3384511)

[Angular requirement: 2](#_Toc3384512)

[Database requirement: 2](#_Toc3384513)

[Procedure: 3](#_Toc3384514)

[1) Database Creation 3](#_Toc3384515)

[2) Spring boot application creation and implementation 3](#_Toc3384516)

[3) Angular Project Creation and implementation 3](#_Toc3384517)

[Database Creation Steps 4](#_Toc3384518)

[Database Installation 4](#_Toc3384519)

[Database Creation 4](#_Toc3384520)

[Spring Boot Application Setup 7](#_Toc3384521)

[Spring boot application creation and implementation 7](#_Toc3384522)

[Configuration of Project 9](#_Toc3384523)

[Folder Structure: 10](#_Toc3384524)

[Package 1: com.cg.jcat.api 11](#_Toc3384525)

[Package 2: com.cg.jcat.api.controller 11](#_Toc3384526)

[Package 3: com.cg.jcat.api.service 12](#_Toc3384527)

[Package 4: com.cg.jcat.api.dao 13](#_Toc3384528)

[Package 5: com.cg.jcat.api.repository 14](#_Toc3384529)

[Package 6: com.cg.jcat.api.entity 15](#_Toc3384530)

[Package 7: com.cg.jcat.api.exception 15](#_Toc3384531)

[JUNIT test cases: 16](#_Toc3384532)

[Package 1: com.cg.jcat.api.service 16](#_Toc3384533)

[Package 1: com.cg.jcat.api.dao 16](#_Toc3384534)

[Angular Setup 17](#_Toc3384535)

[Angular Project Creation and implementation 17](#_Toc3384536)

[Folder Structure: 19](#_Toc3384537)

[Project Structure 19](#_Toc3384538)

[Package 1: com.cg.jcat.api.module 20](#_Toc3384539)

[Package 2: com.cg.jcat.api.component 21](#_Toc3384540)

[Package 3: com.cg.jcat.api.service 21](#_Toc3384541)

[Package 4: com.cg.jcat.api.entity 22](#_Toc3384542)

[Package 5: com.cg.jcat.api.router 23](#_Toc3384543)

[Package 5: com.cg.jcat.api.view 24](#_Toc3384544)

[Package 5: com.cg.jcat.api.utility 25](#_Toc3384545)

[Conclusion 25](#_Toc3384546)

Scope: This document contains details about development of cloud assessment tool using JAVA in back end, ANGULAR in front end and MYSQL for database.

Cloud Assessment Tool is developed to assist an application for cloudability. It also provides information for Migration Patterns (REHOST, PUBLIC-PASS and RE-PLATEFORM) and cloud providers (Public, Private).It is a Configurable Application.

It will decide application is cloudable or not on the basis of some questions which will be client specific. It provides some features like:

* Client can add, update and delete questions as well as applications.
* It can have many users where admin can add, update and delete user.
* This tool have a great feature where client can change language as per use means provide “MULTI LENGUAL feature”.
* It provides a feature to generate report of the application if assessment is done.
* It provides “Audit trail”.
* Import export feature makes client feasible to add application.
* Client can add rules for the questions they have created.
* Providers, Platform and cloudability will decide, based on the rule which has been set.

# Requirements:

## Java requirement:

1. Eclipse
2. JRE, JDK should be installed
3. Set JAVA\_HOME in environment variables
4. Spring Boot with ORM Tool(Hibernate framework with JPA)
5. Maven – latest version

## Angular requirement:

1. Visual Studio Code
2. Angular CLI should be installed in Node.js cmd
3. Node.js (set path in environment variable)

## Database requirement:

1. My-SQL Workbench (latest)

# Procedure:

## Database Creation

## Spring boot application creation and implementation

## Angular Project Creation and implementation

# Database Creation Steps

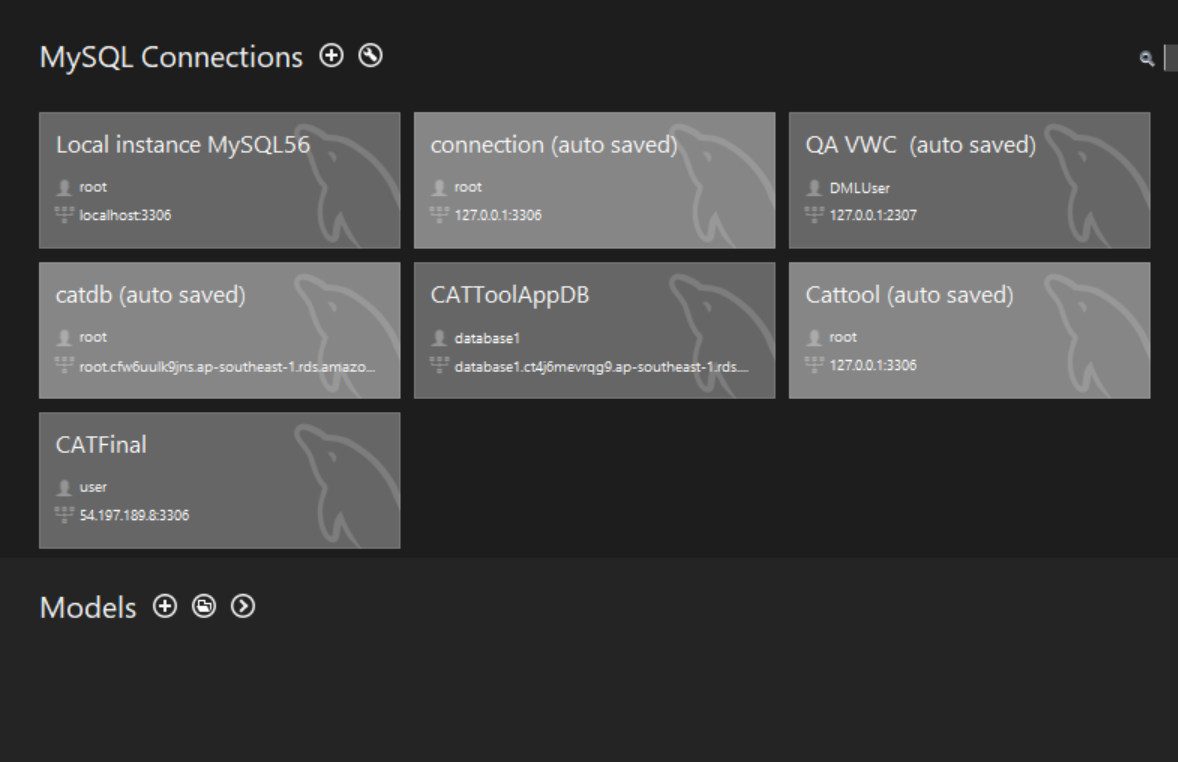
## Database Installation

* Install MySQL workbench from <https://www.mysql.com/products/workbench/>

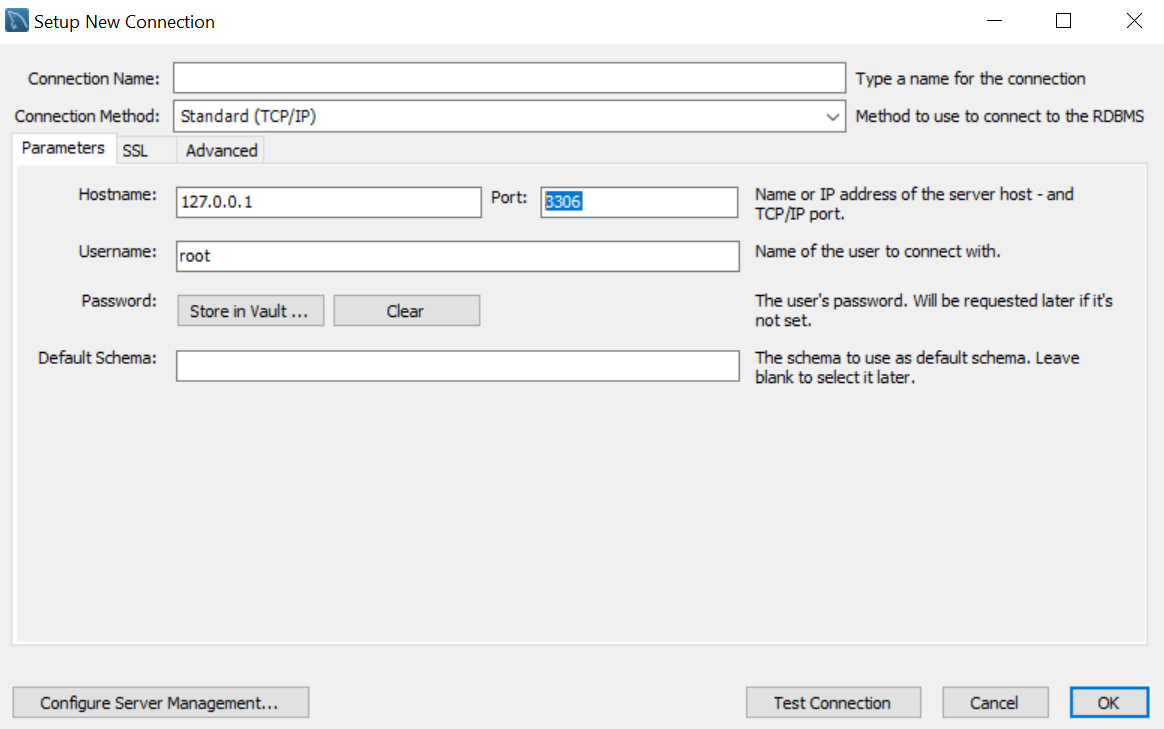
Note: - Need admin rights for installation

## Database Creation

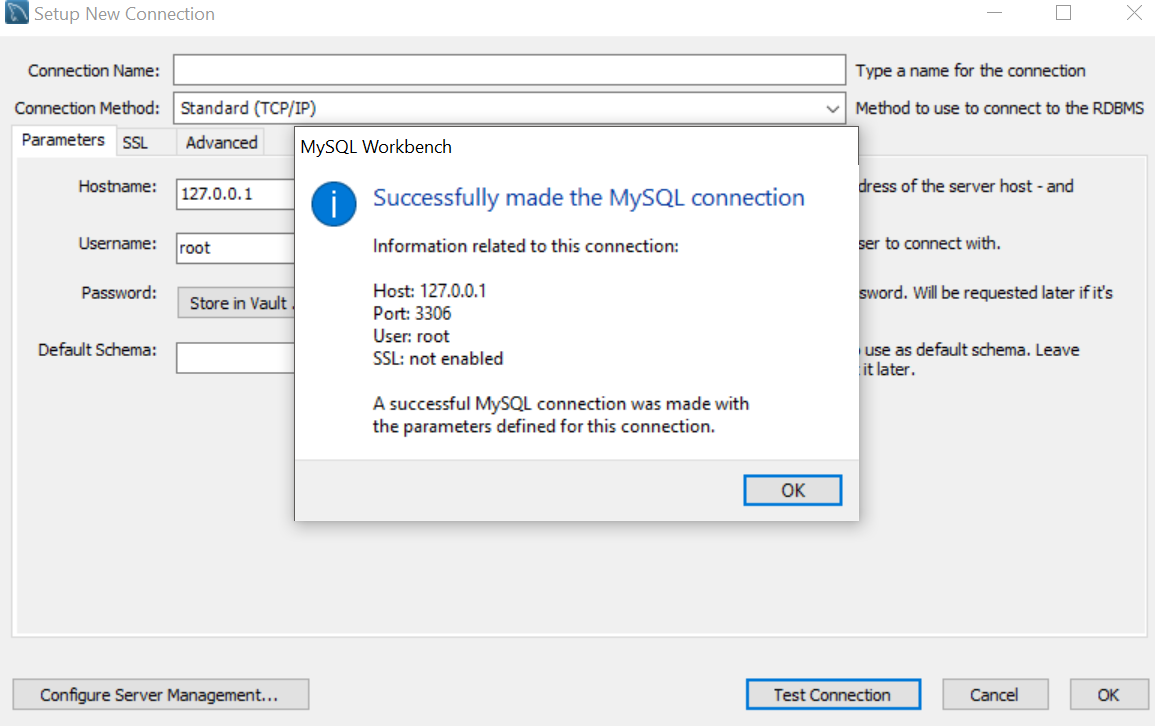
* Open MySQL workbench.
* Click on MySQL connection, click “+”.



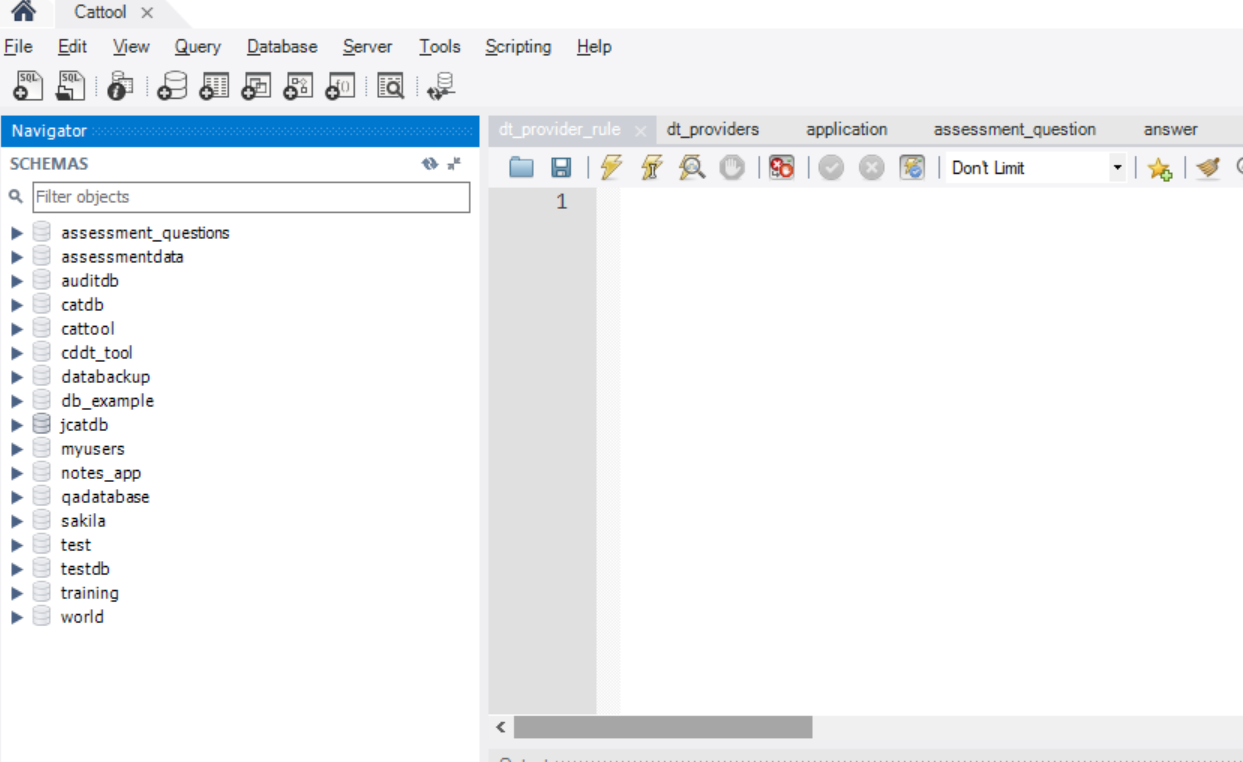
* Then enter hostname and port and the test connection.



* After clicking on “Test Connection” following window prompt. Then click on “ok “.

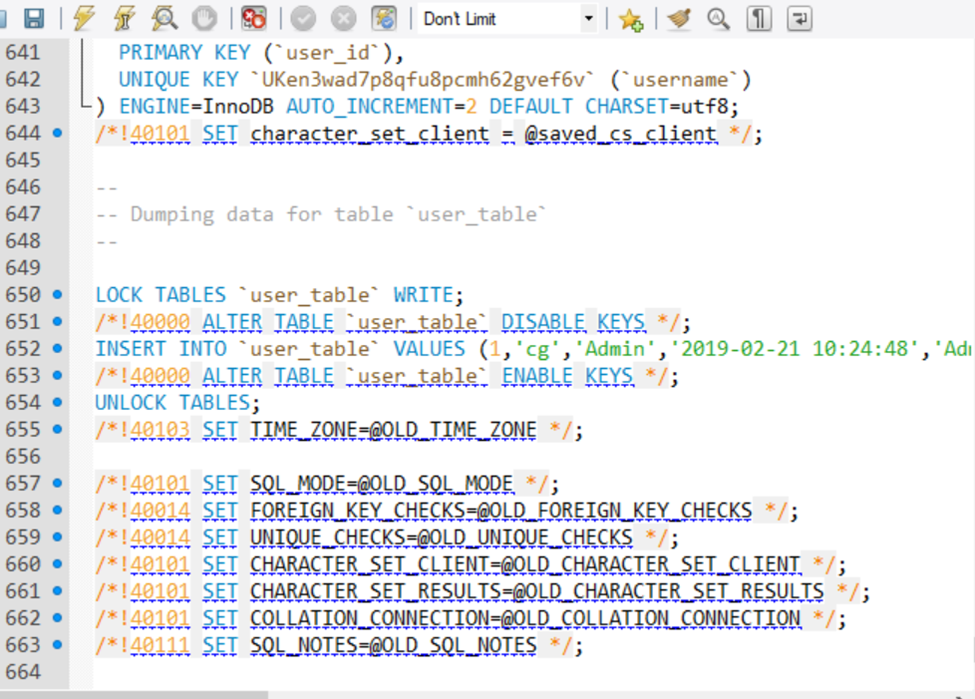


* Now you can see you connection has been established and then double click on your connection and it will open your connection. Now create database using “create database <database name>”. And then you will be able to see database name in Navigator.



Note: User can to import the existing MySQL file into the workbench

* Open MySQL Workbench.
* Open .sql file and copy all the data to MySQL.
* Then click on execute command.



# Spring Boot Application Setup

## Spring boot application creation and implementation

* Go to the site

<https://start.spring.io/>

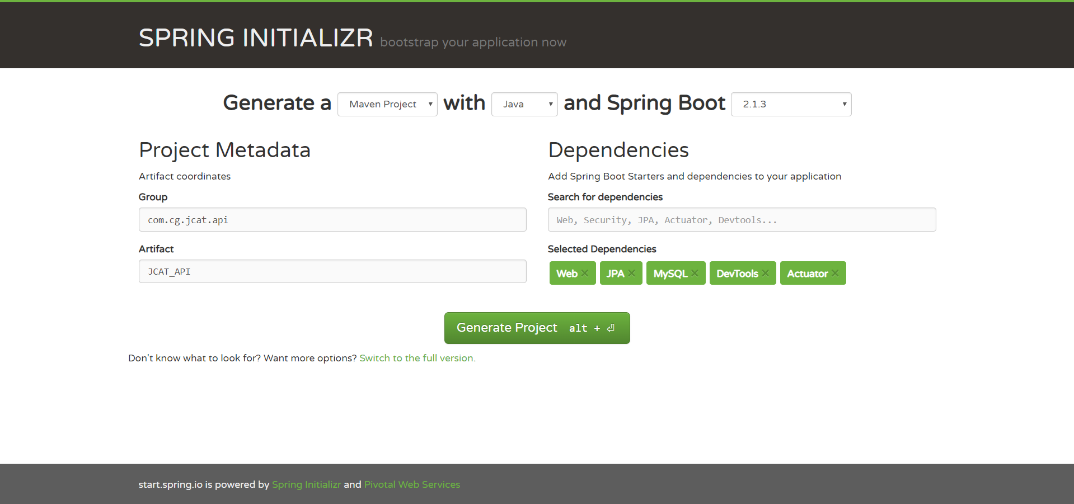
* Fill the following details for project creation.

Group-name: Package name

Artifact: Project name

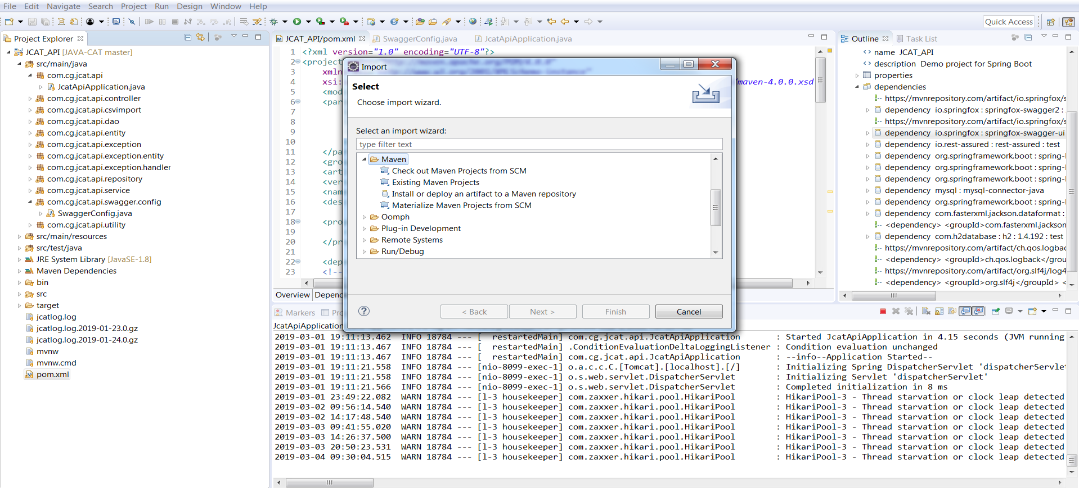
Search for dependency: Web, JPA, MySQL, DevTools, Actuator

Then click on generate project it will download a zip file.

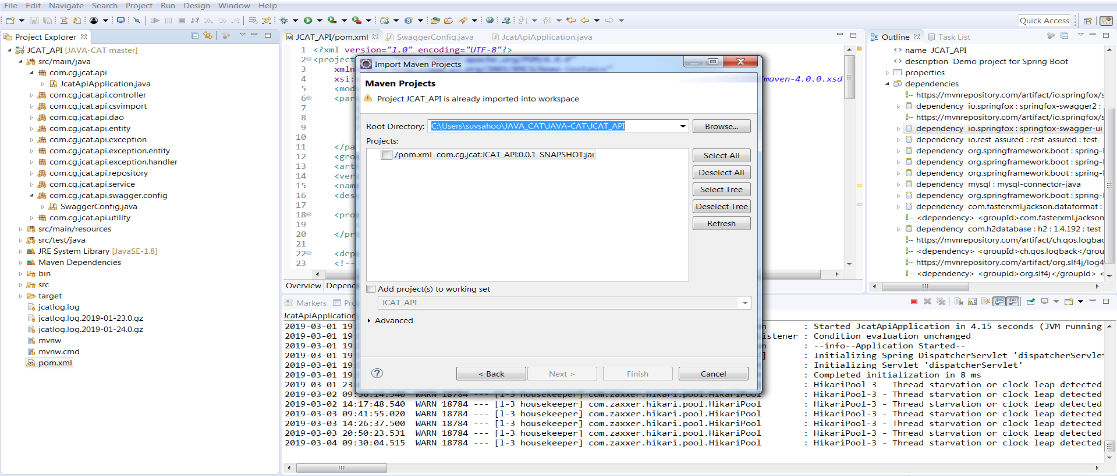


* Then open eclipse then click on “File” and then choose “Import project ”

And where you have to select “Existing Maven Projects” as shown below



* Click on “Next” and specify path from where you want to import the project in the root directory section and click on finish. Then you will able to see your project in “Project Explorer”



Note:- User can use already create project from SVN and import to eclipse

## Configuration of Project

server.port: Assign port where you want to run projects

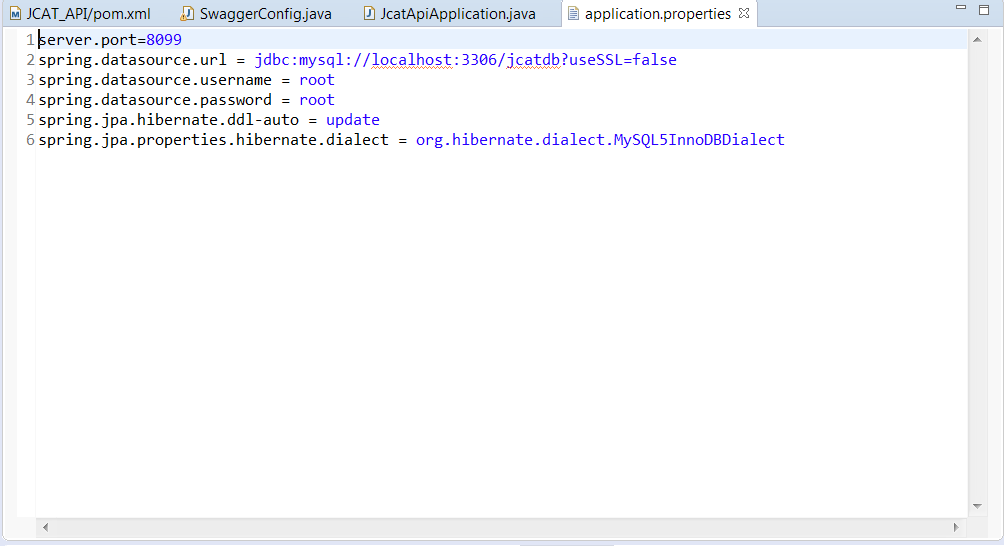
datasource url: Database server url

datasource username and password: mysql username and password

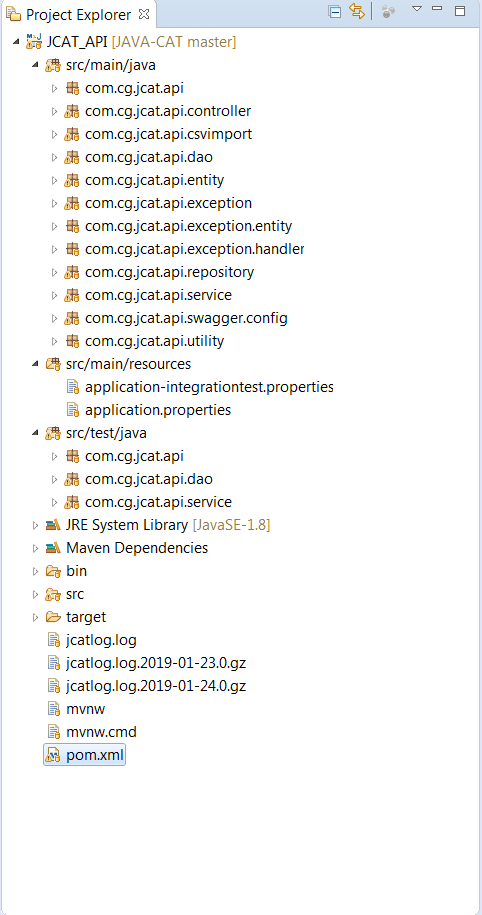
datasource ddl-auto: update query first time will create table once and then update table.

Hibernate.dialect- use mysql dialect.

Other than it logger and other details can specify here.



## Folder Structure:

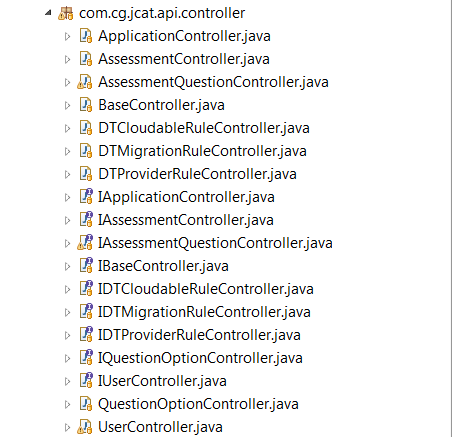


### Package 1: com.cg.jcat.api

Under this package spring boot creates a main class which will responsible to run project.

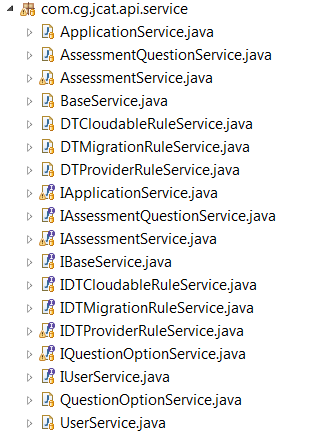
### Package 2: com.cg.jcat.api.controller

This package contains all the controller classes and interfaces. Controller handle http requests. When API will call by UI, directly request come in controller. Controller interface talks with http request which internally calls controller classes. Then controller class call service interface.



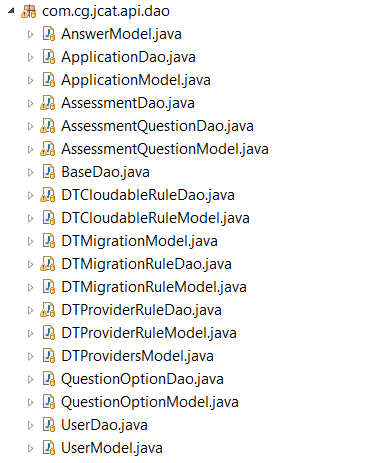
### Package 3: com.cg.jcat.api.service

The package contains validation and business logic and service interface is called from controller classes and its method implements by service class. Further service class calls Data access object layer.



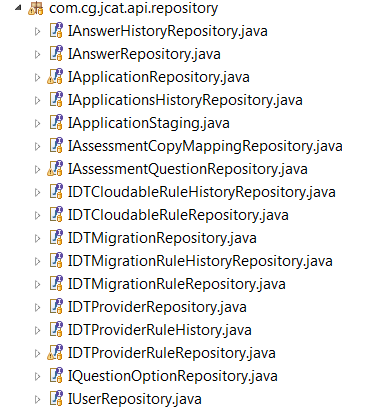
### Package 4: com.cg.jcat.api.dao

The package contains dao class which communicate with repository and it also contains model class which is a partial implementation of entity class for sending data to the user. Dao class converts entity object to model and vice versa. Means this layer will retrieve data from database. DAO class called from service class and dao class will call repository interface to retrieve data from database.



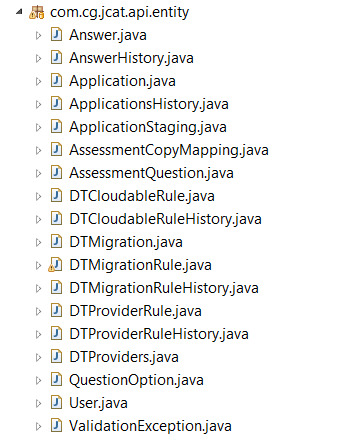
### Package 5: com.cg.jcat.api.repository

This package contains all repository interface which map java object with database object. Where jpa repository uses hibernate framework for mapping the objects. This repository interface called from dao(data access object) class and retrieve data from database.



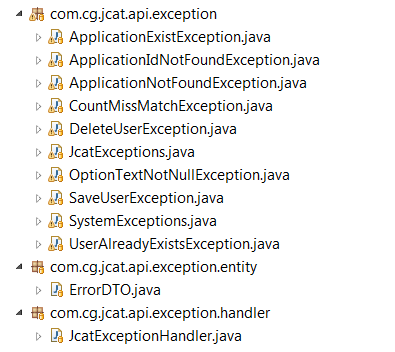
### Package 6: com.cg.jcat.api.entity

This package contains all entity which is POJO (Plain Old Java Object) class which has some attributes which describe behavior.



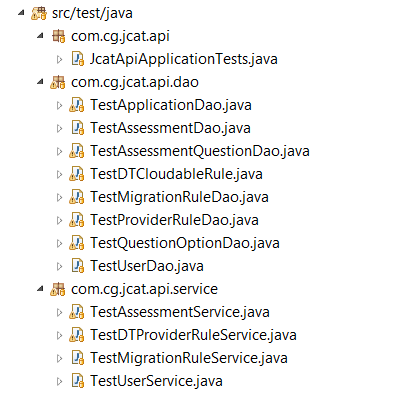
### Package 7: com.cg.jcat.api.exception

This package contains all the exceptional classes which will be thrown from the different classes. Exception is the base class of JcatException. ErrorDTO class contains all the fields which will be shown to the user when exception occurs. Exception handler class is the global class for handling all the exceptions thrown by the different controller.



## JUNIT test cases:

JUNIT test cases used to test method which implemented in class. For services mockito object is used means provide a dummy data for testing and for dao it uses h2 database for testing where h2 database’s configuration is given in application-integrationtest.properties.



### Package 1: com.cg.jcat.api.service

It contains test cases for all service classes.

### Package 1: com.cg.jcat.api.dao

It contains test cases for all dao classes.

# Angular Setup

## Angular Project Creation and implementation

* Open Node.js
* Install Angular CLI. For that run below command,

*npm install -g @angular/cli*

* Open VS code and run the following command to create a new project in angular.

*ng new jcat\_ui*

* Now project has created and to run project get project path where project present and go to Node.js and paste path

*cd <*jcat\_ui project path*>*

* Now run command

*npm install*

* Now run project by entering command

*ng serve*

* If you have existing project you can use it by –

Open node.js

Provide the project path, enter command

*cd <*jcat\_ui project path*>*

Then run command

*npm install*

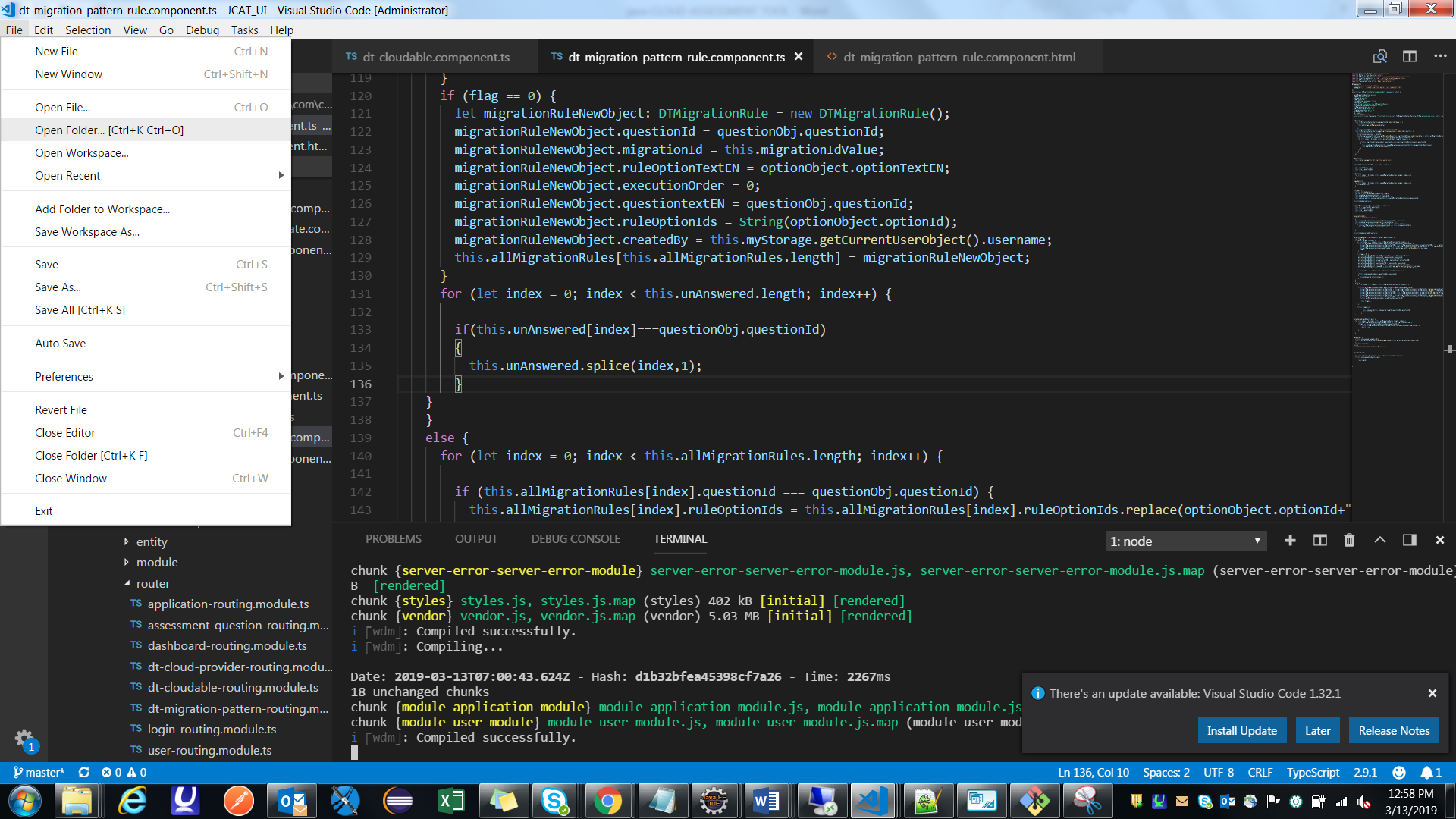
Now you can run the project, by using

*ng serve*

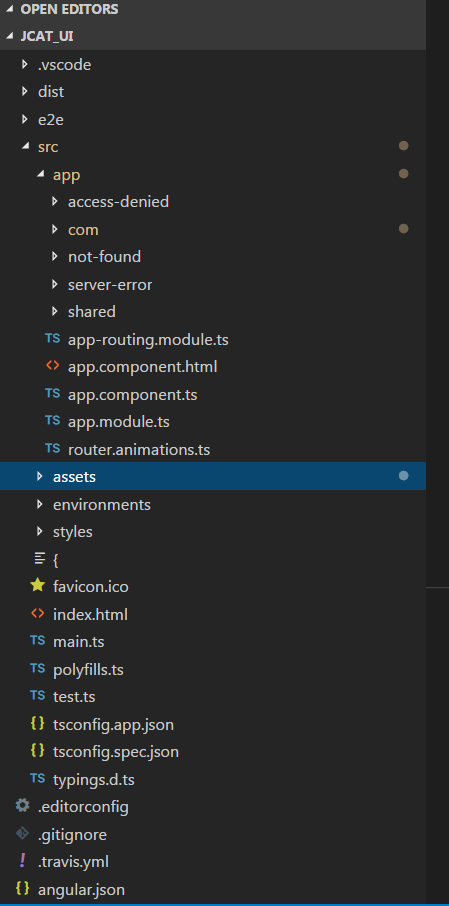
To see the project code open VS code

Open your project there by clicking on

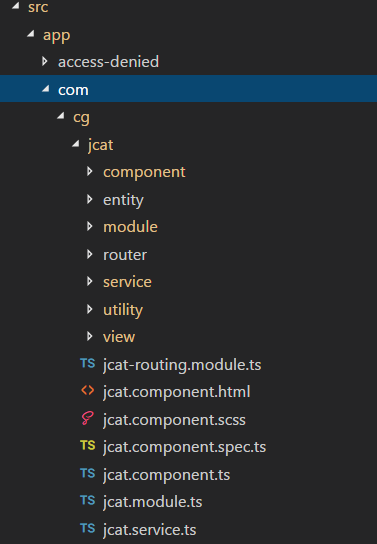
file->open Folder / drag and drop the folder.



## Folder Structure:

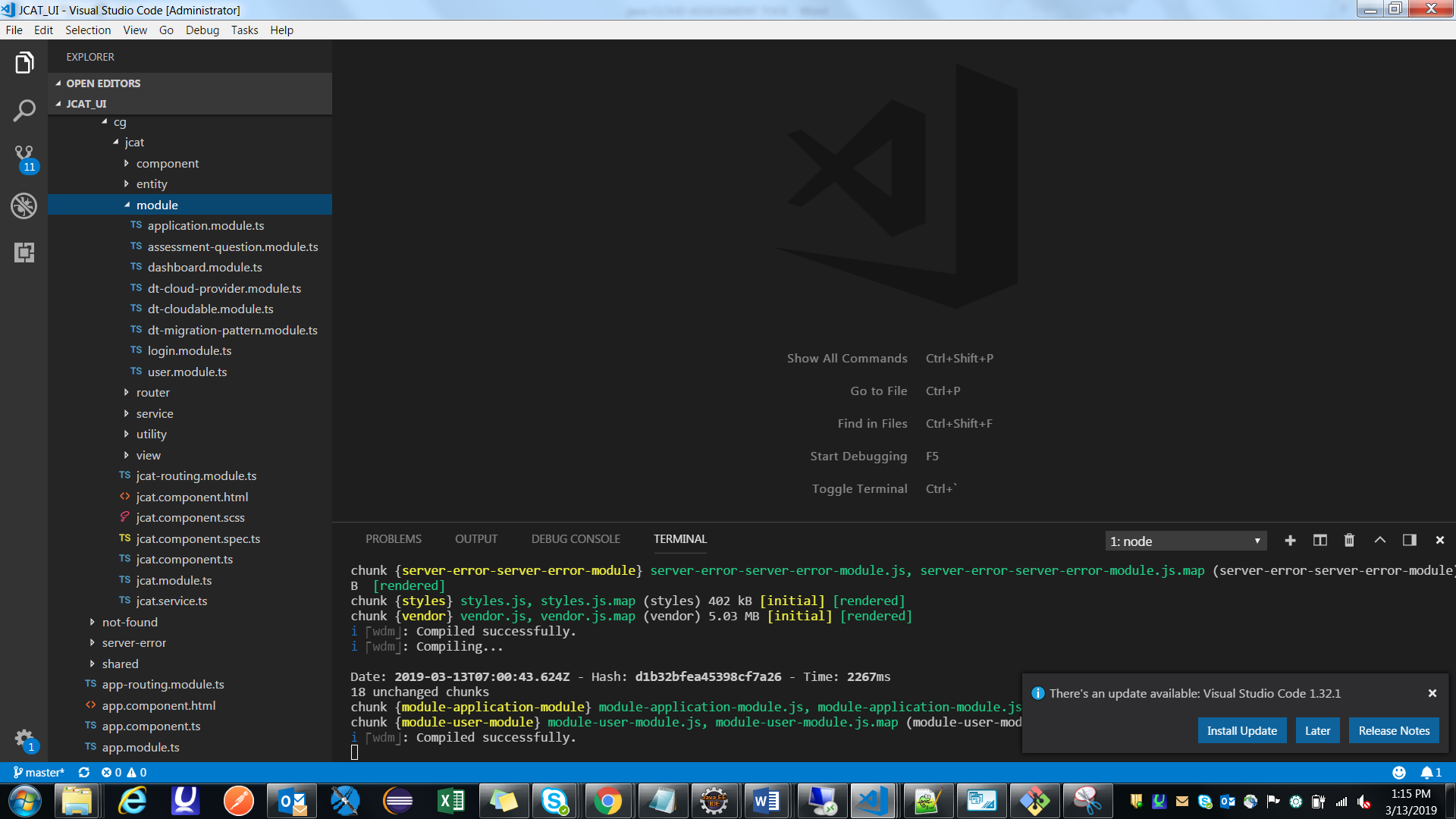
**

### Project Structure

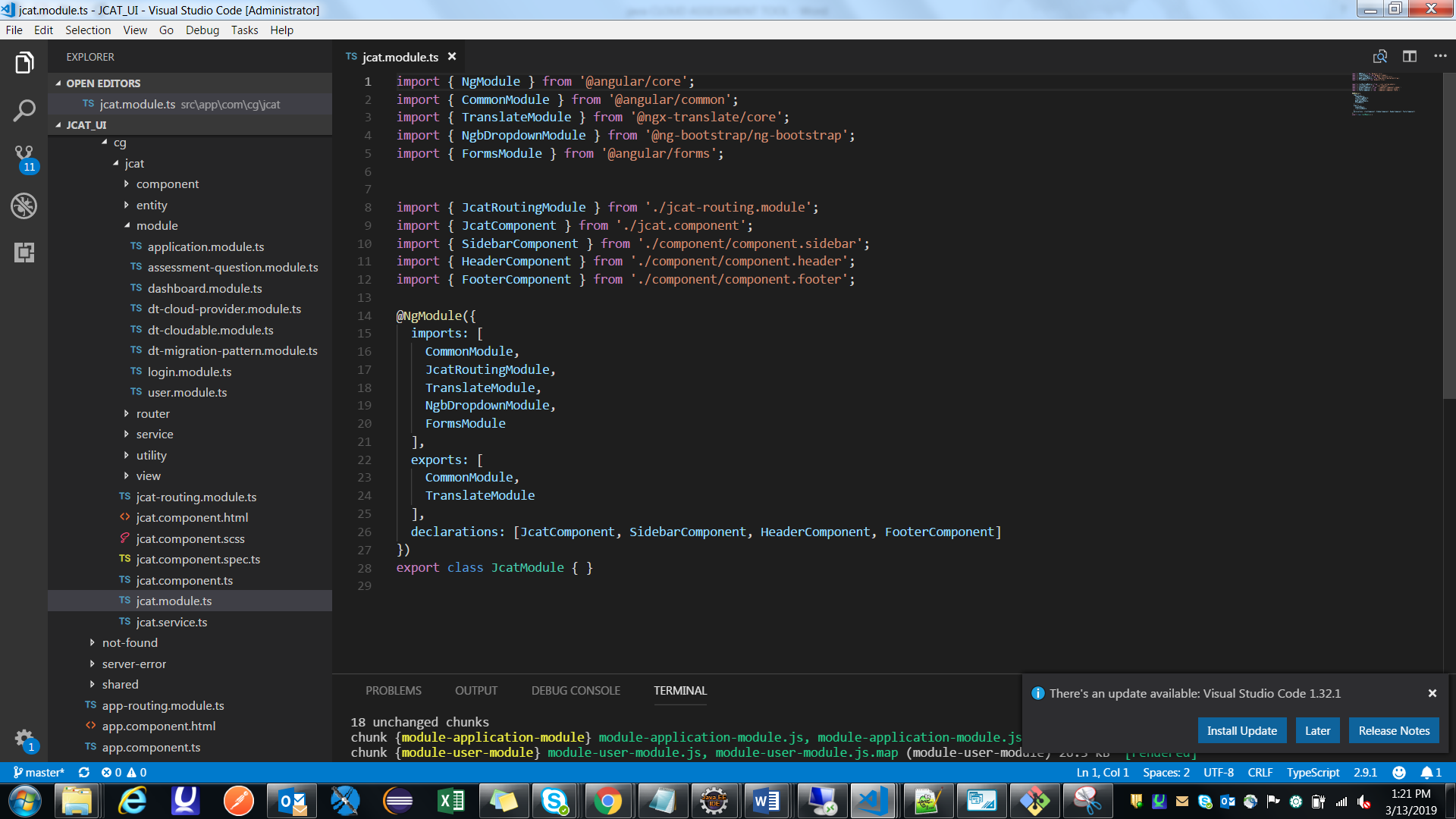
**

### Package 1: com.cg.jcat.api.module

This package contains all the module classes. Where you can group the components, directives, pipes, and services, which are related to the application.

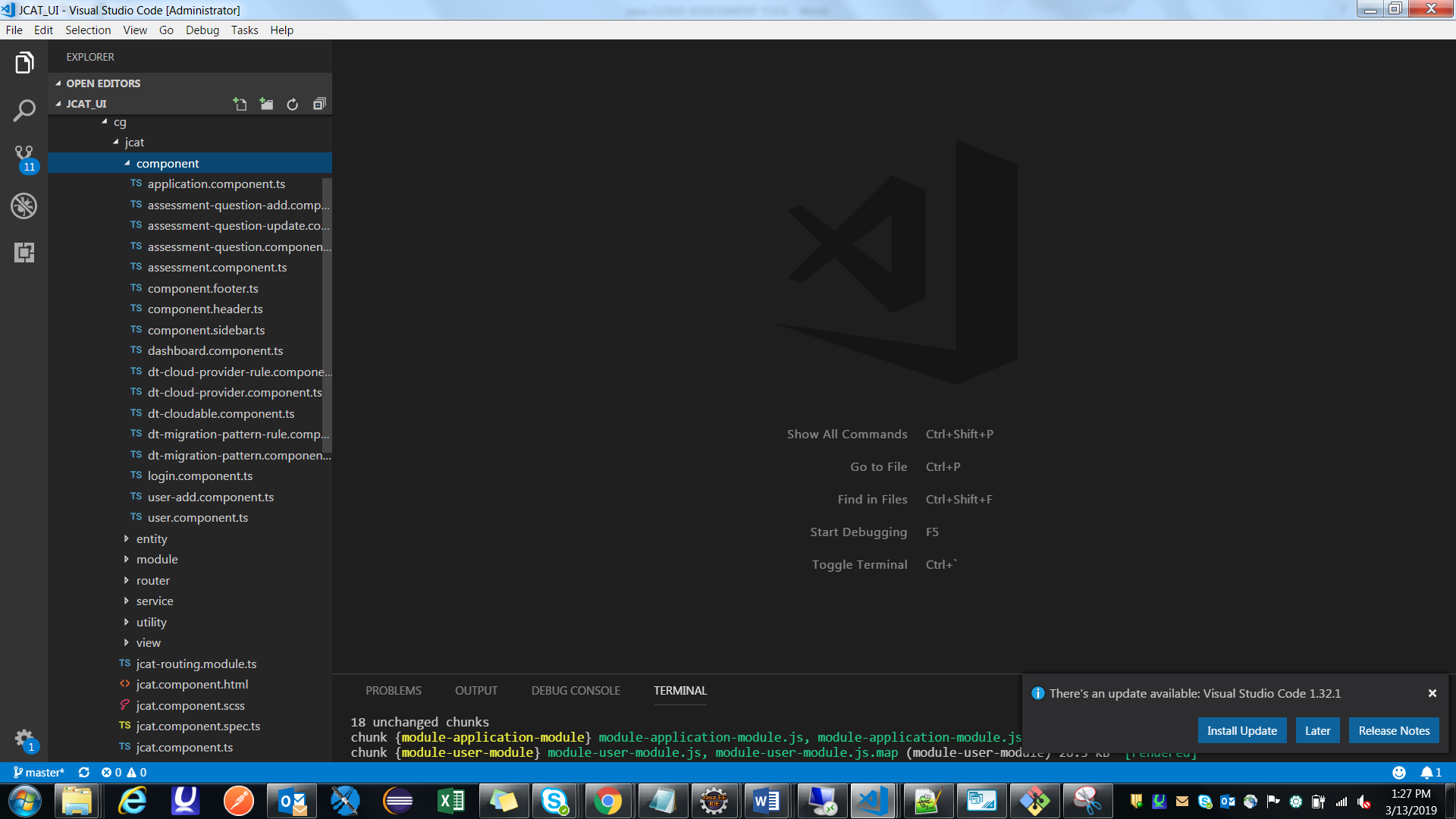


All the current component and child components should be declare in the declaration section, all the services related to the component should be declare in the providers section and all the libraries and routing modules should be declare in the import section of the module.



### Package 2: com.cg.jcat.api.component

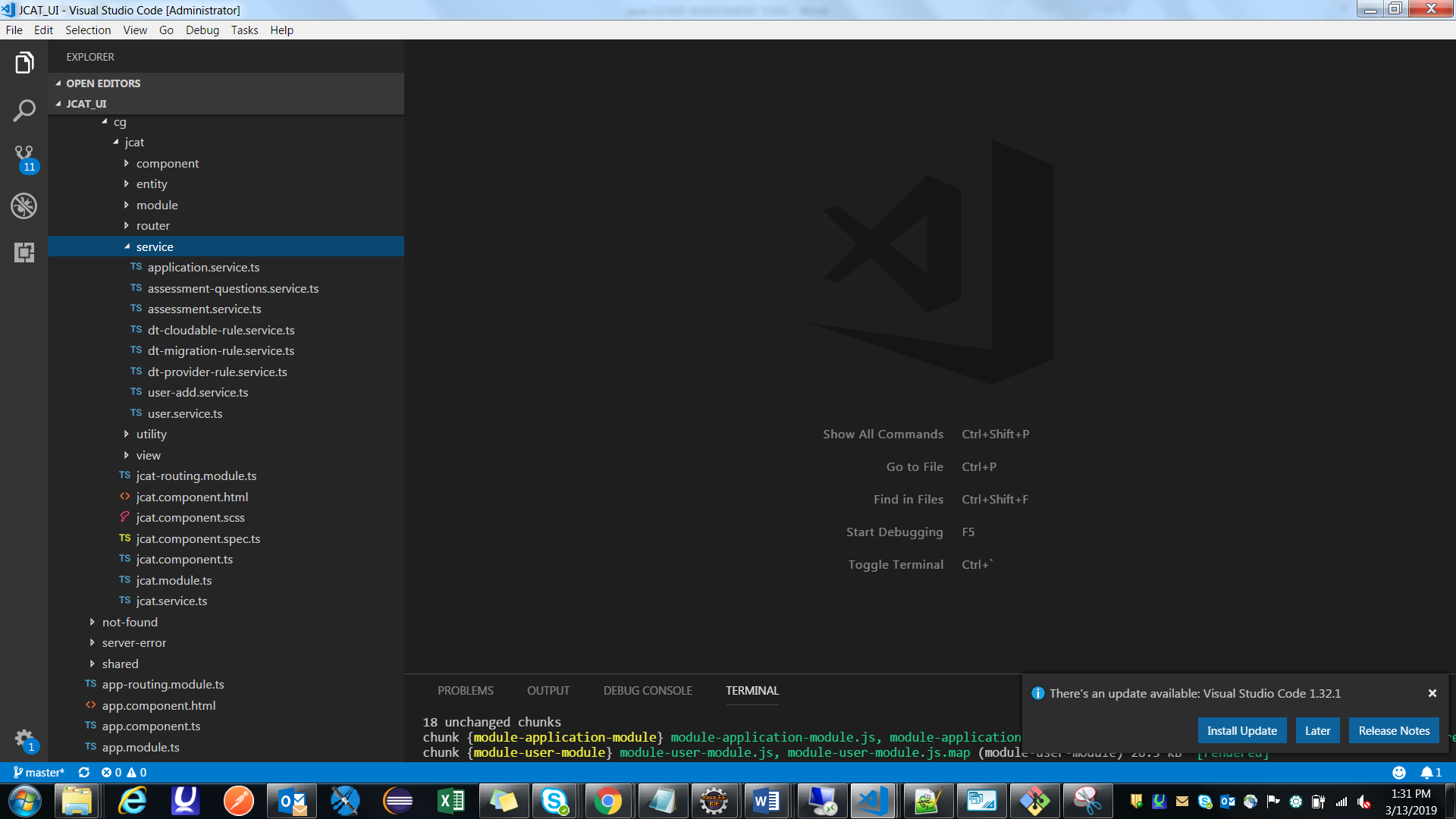
Components are basically classes that interact with the .html file of the component, which gets displayed on the browser.



All the business logic should be present In the component which directly communicate with the view to show the result and service to get the result from API.

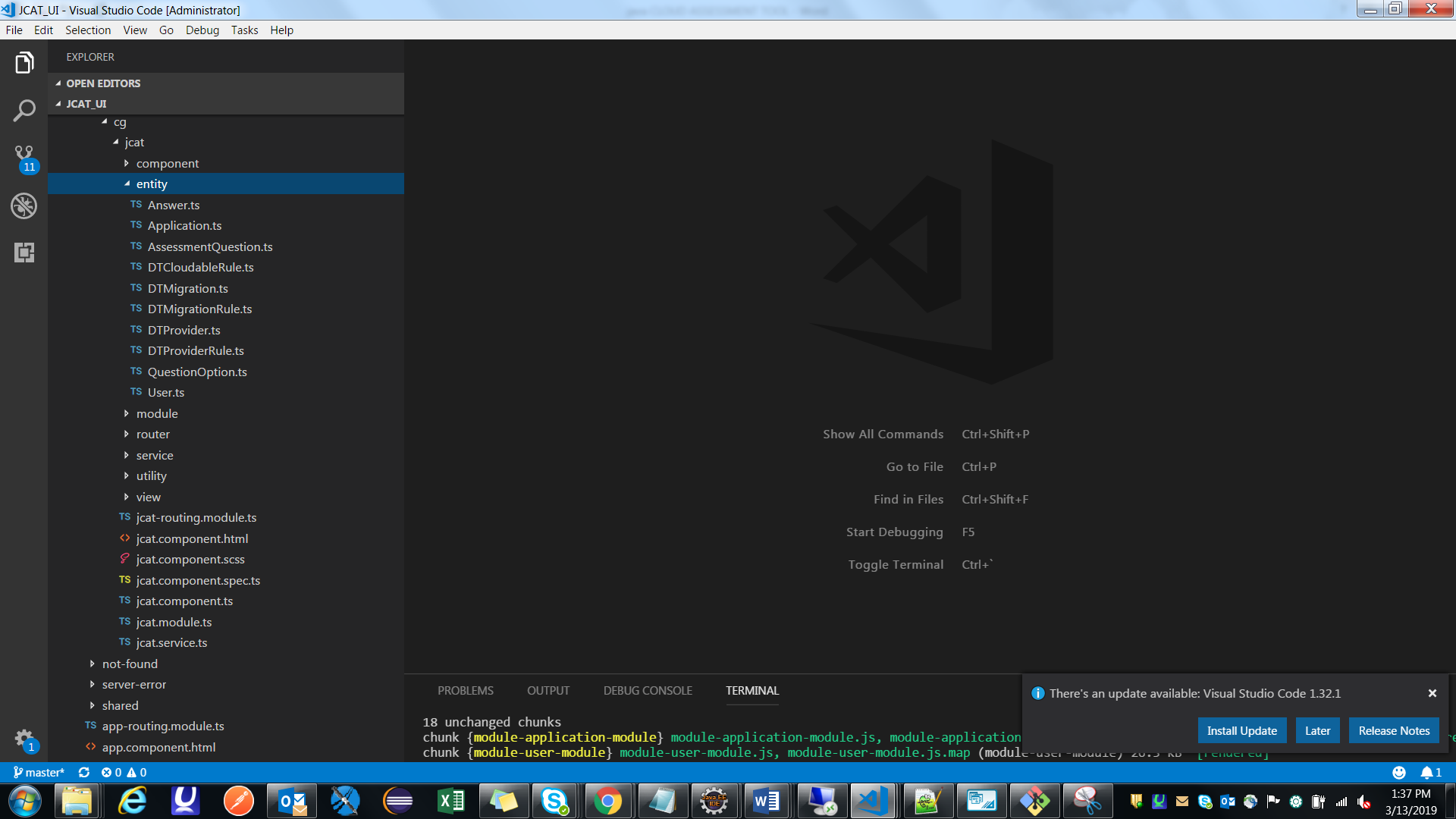
### Package 3: com.cg.jcat.api.service

Service uses methods for all component as well as sub components to call the API for collecting data from JAVA. Every component is connect to service to get that data.

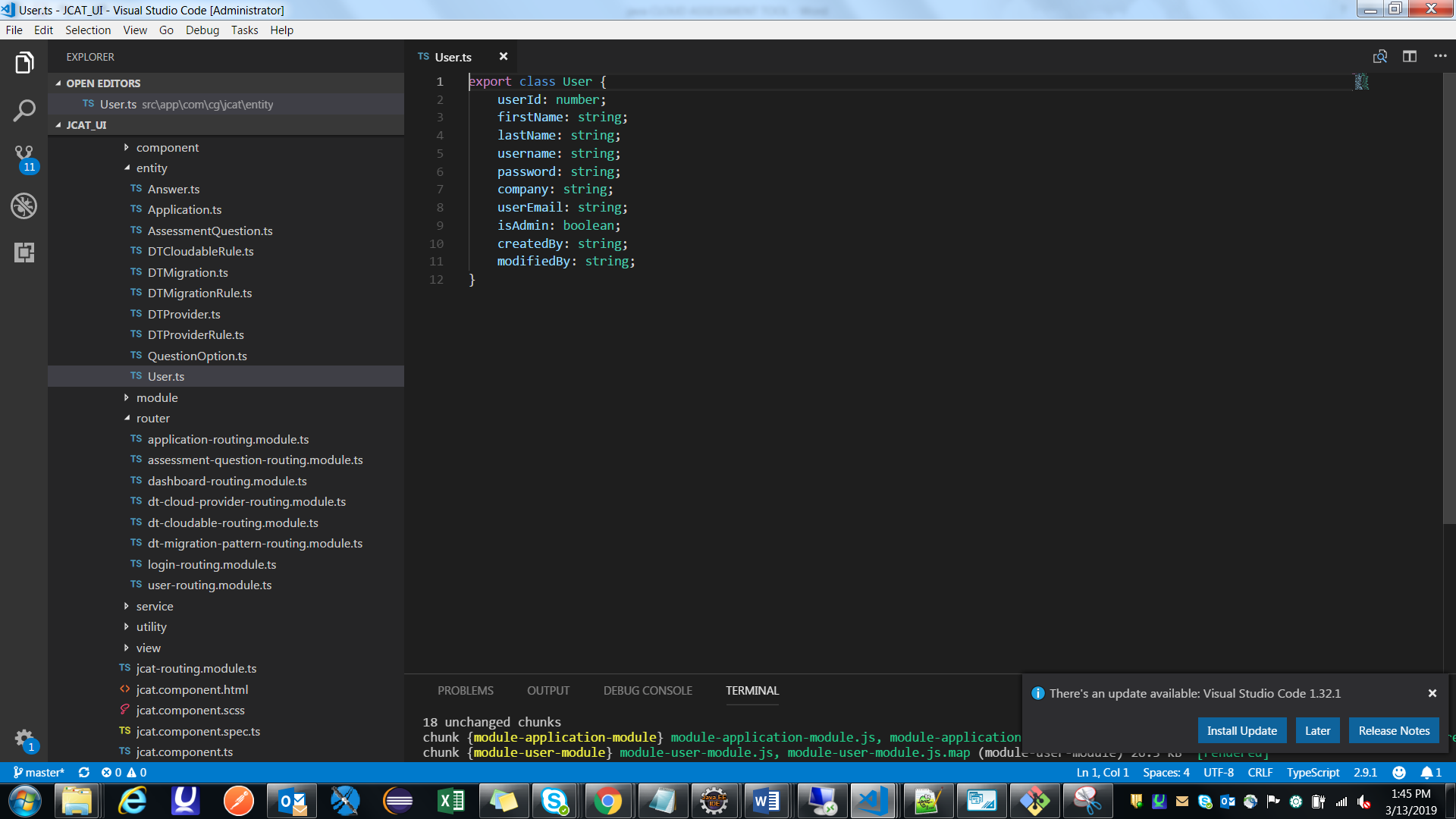


### Package 4: com.cg.jcat.api.entity

It contains all the model classes, which can be used throughout the project.

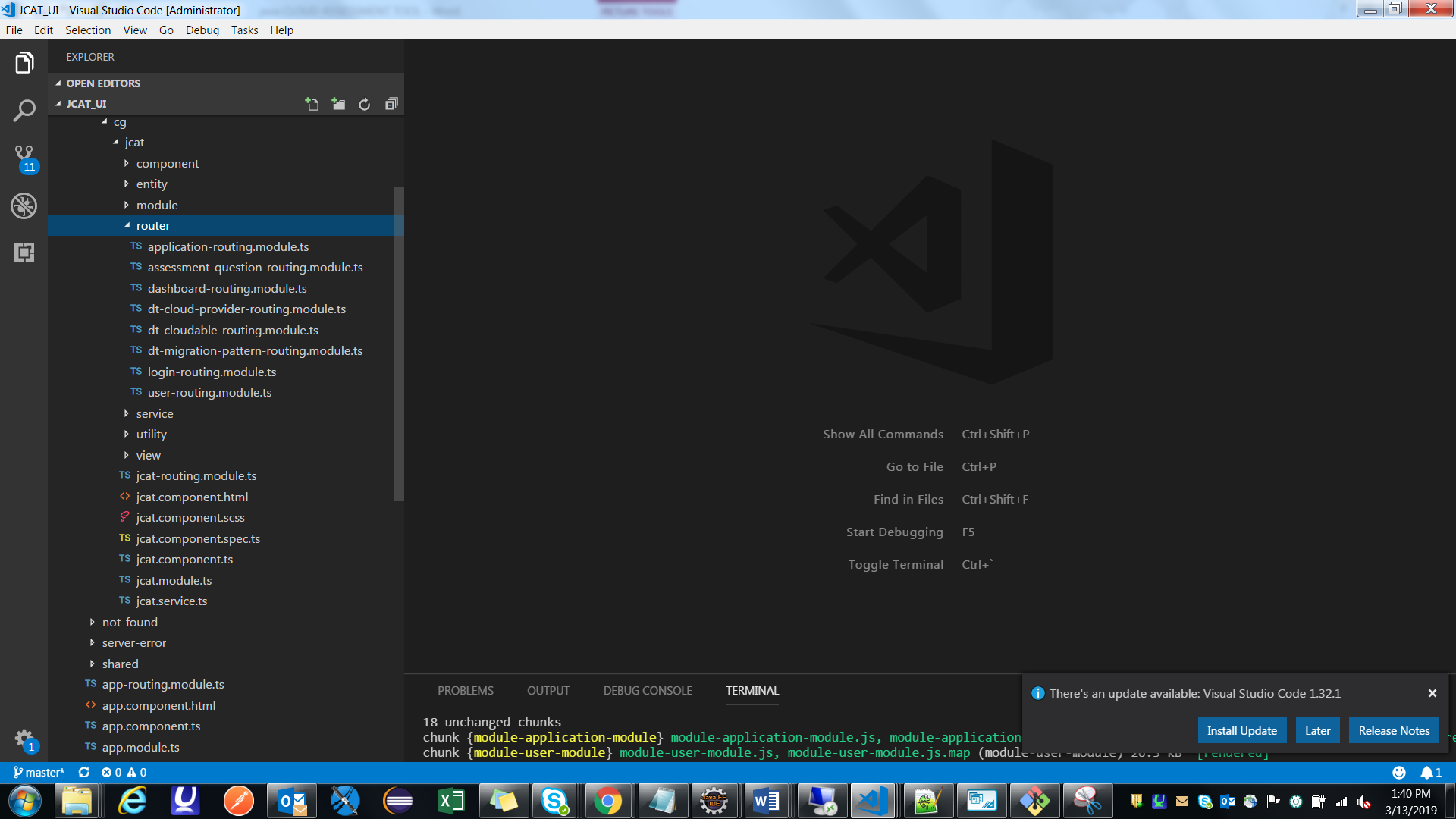


Entity class contains all the attributes coming from a json.

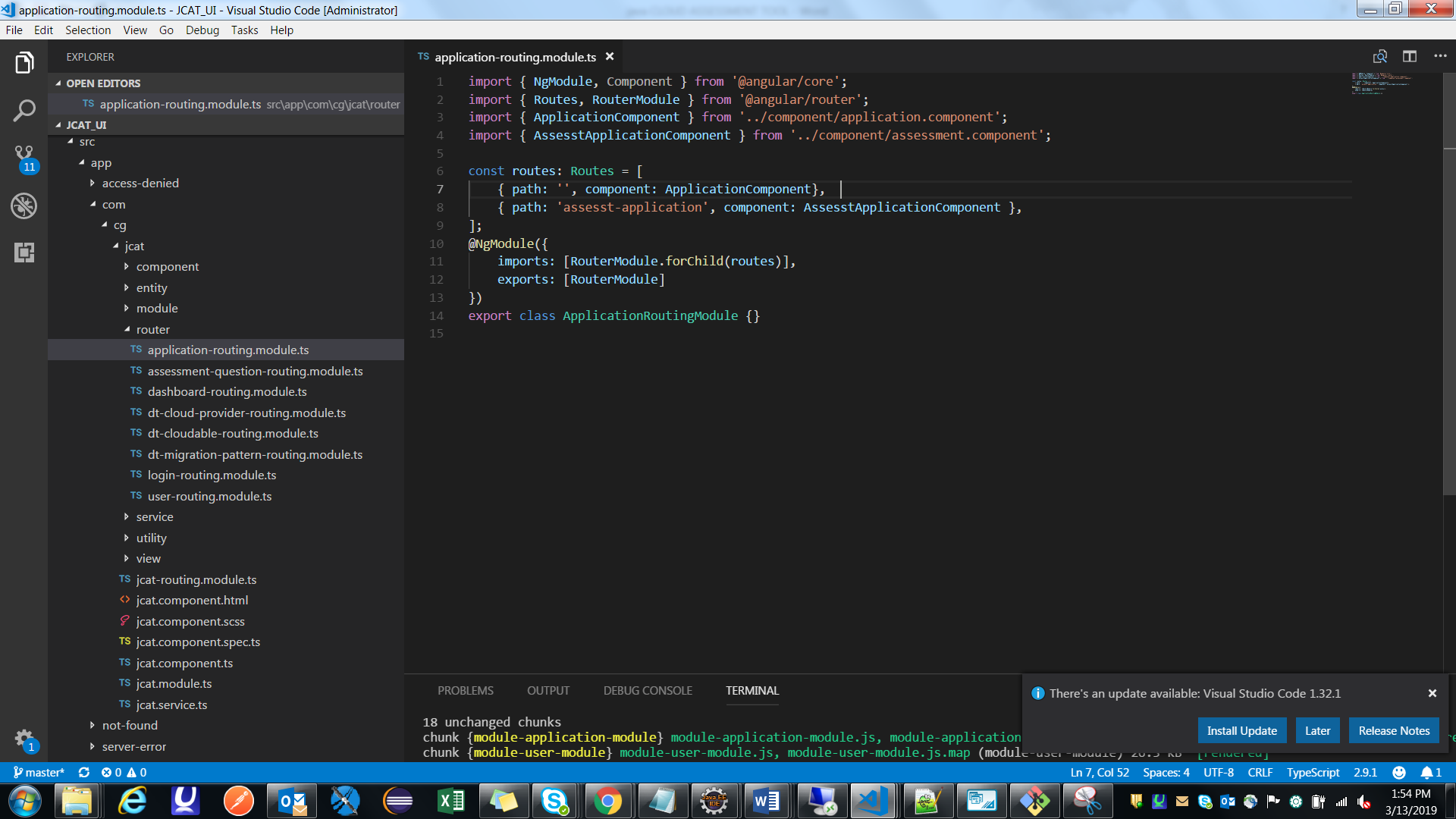


### Package 5: com.cg.jcat.api.router

Routing basically means navigating between pages. This can be achieved using routing. You can route from one component to other component.

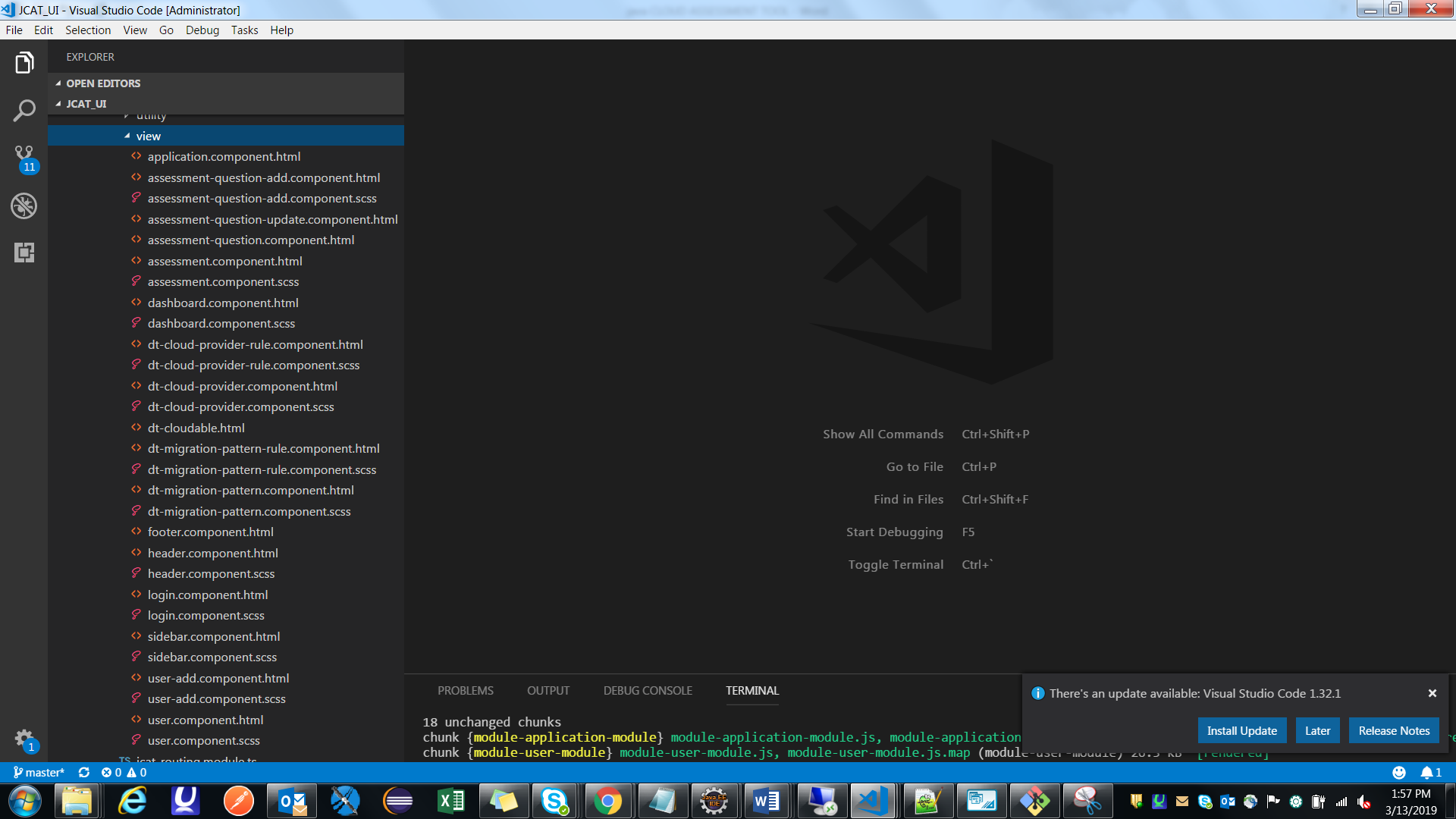


This routing module contains constant field which is of type array and contain paths of different component to route.



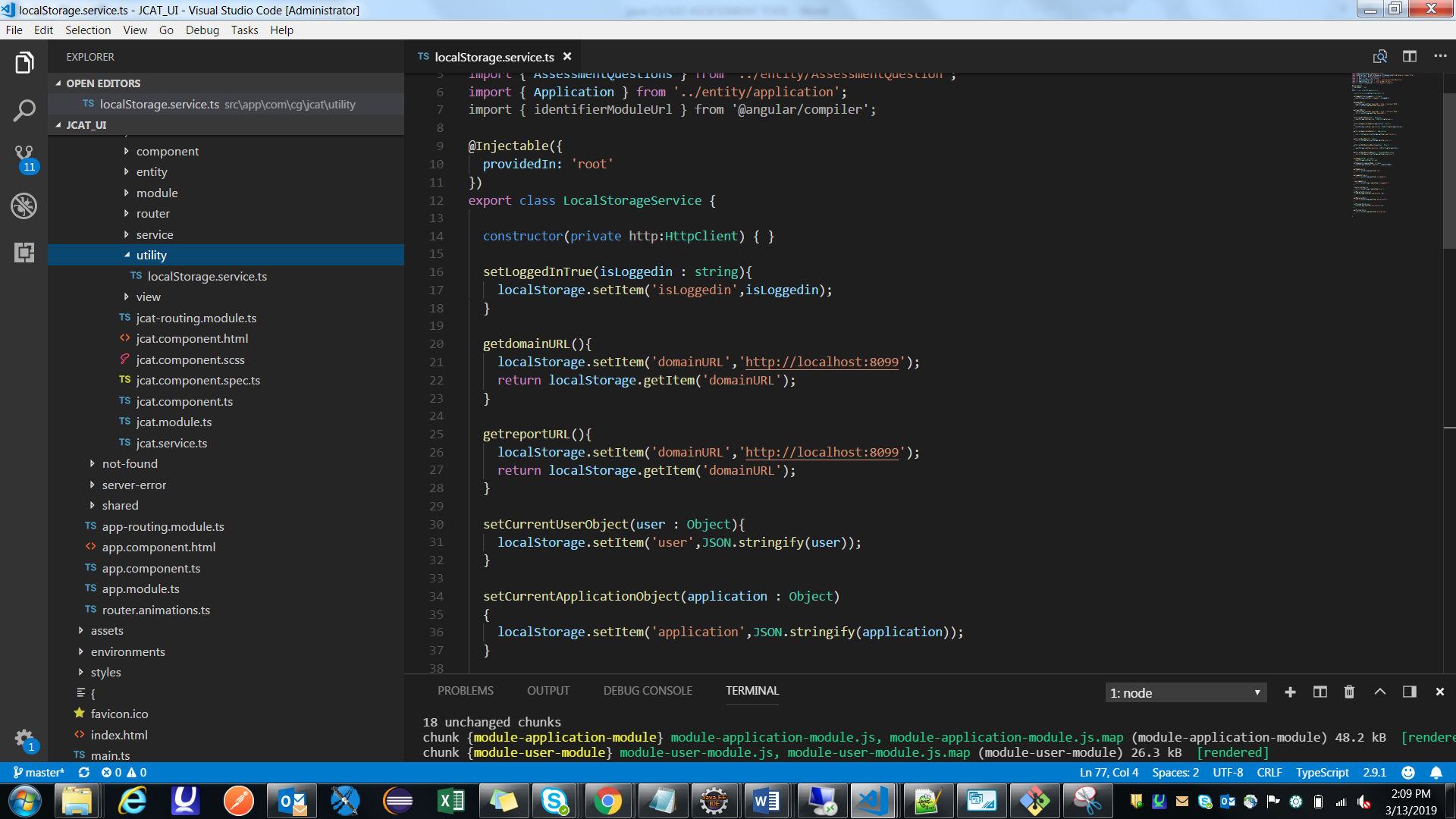
### Package 5: com.cg.jcat.api.view

It contain all html pages which is directly connected with component. Component load html page and show data on UI.



### Package 5: com.cg.jcat.api.utility

This package contains a class which contains common methods for all the component. You can store data by set method and retrieve data by using get method. It will provide reusability of code and reduce redundant data. Here data is store in localstorage so all the components can easily access is. Localstorage is a temporary storage for the project.



## Conclusion

This application will help u to assist applications to check cloudability and also provide platform and providers. It has new features like multilingual, client can switch language, Audit Trail to maintain record, fast transaction because of localstorage.