QUIZ MANAGER

Final Advanced Java Project

ABSTRACT-USER GUIDE & TECHNICAL SPECIFICATION

Sravan Nallala

[M.sc Software Engineering

Table of Contents

1. Subject Description:

This project is made to provide an evaluation of the advanced java course the goal of this project is to develop a program (API oriented, Web-based) that helps in dealing with quiz assessments.

The usual problem while preparing an evaluation is to:

- Constitute an appropriate evaluation regarding of the required level
- Reuse former questions
- Organize sample evaluations
- Correct automatically the MCQ questions.

2. Subject Analysis:

2.1Major Features

- Authentication: A user should have a valid login in-order to go into the application.
- Admin: Can be able to create the quiz questions and review the scores.
- Student: Can be able to take the test and view scores.

2.2Application Feasibility

 The application is developed with Java, Springs and hibernate. The SQL calls are dynamically called by using H2 Data Base connection for embedded database.

 We have used Java, Spring and Hibernate in the backend and AngularJS pages in the front-end.

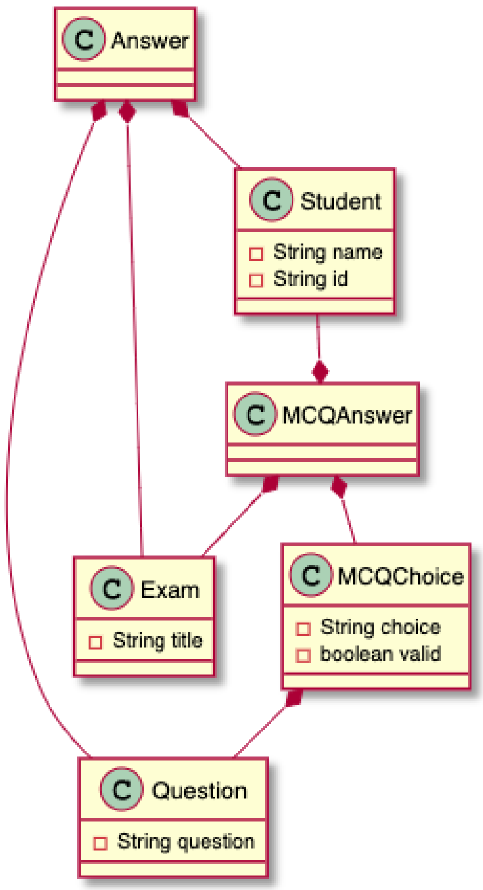
2.3Expected Results

- The expected results of the project are making a quiz Manager Application. There would be two flows of the application. Admin and Student, Both will be having different logins. Based on the login either Admin or Student.
- Admin: Admin will be able to create the questions
- Student: Will be able to take the test and get the score of his after the test.

2.4Scope of the Application

 The application management is restricted to the authenticated users. It manages both the students and the users. This is helpful for the dynamic management.

2.5UML Scope



2.6 Technology

The implementation of this project has been done by using Frontend (HTML, CSS, Angular JS, JavaScript), Backend (Java 8), H2 (Data Base), Business Logic Framework (Hibernate, Spring ORM, Spring MVC) and Server (Tomcat9).

2.7Objectives and concentrations:

This Quiz Management Application handles 2 Identities

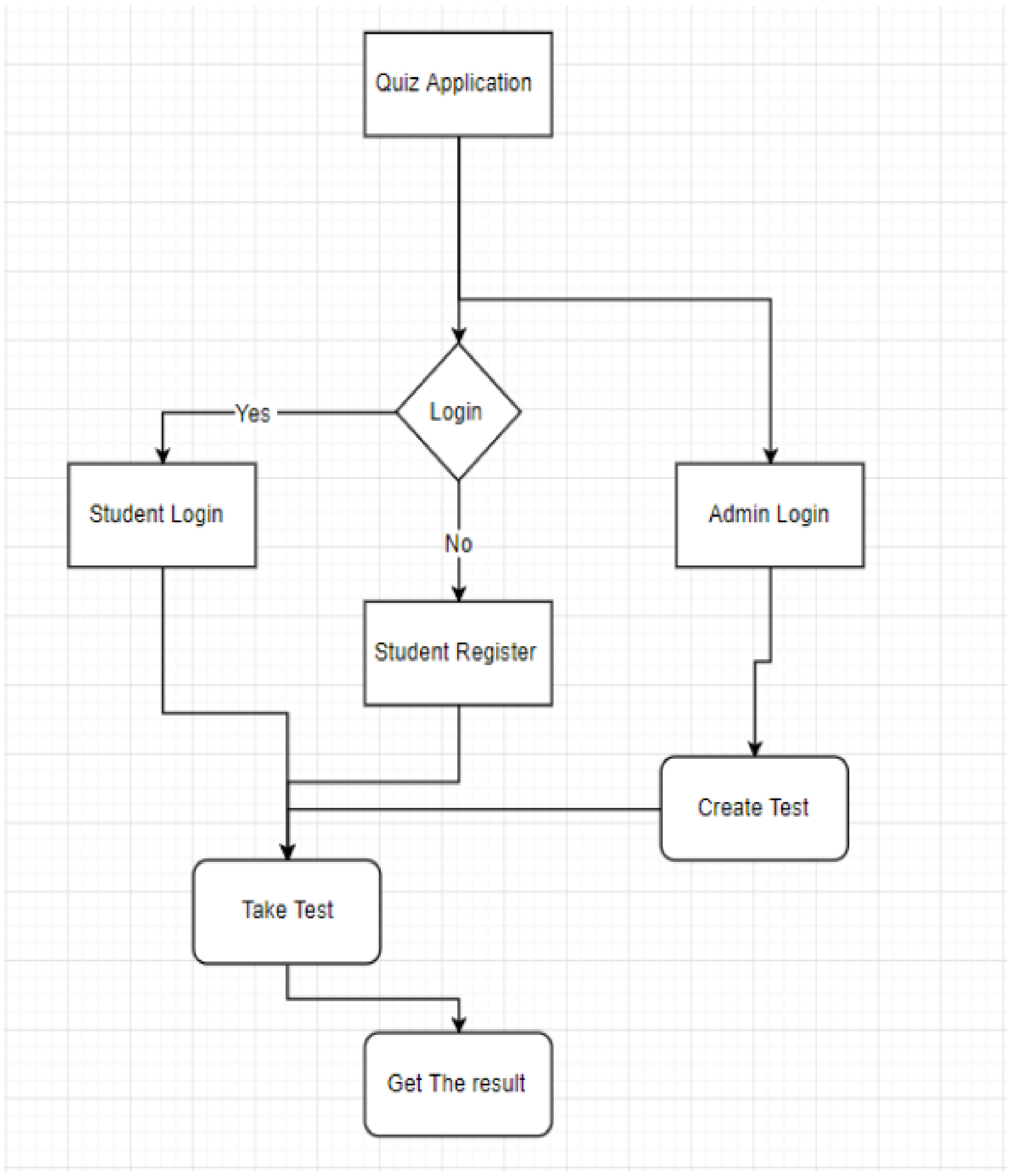
-*Student*
-*Admin*
Students are having privilege to create the account with valid User Name and Password, Then attend the quiz, view the result at the same time.

Admin on the other hand is able to create Quiz Questions.

2.8 User Interface

Application will be accessed through a Browser Interface. The interface would be viewed best using 1024 x 768 and 800 x 600 pixels resolution setting. The software would be fully compatible with Microsoft Internet Explorer for version 6 and above. No user would be able to access any part of the application without logging on to the system.

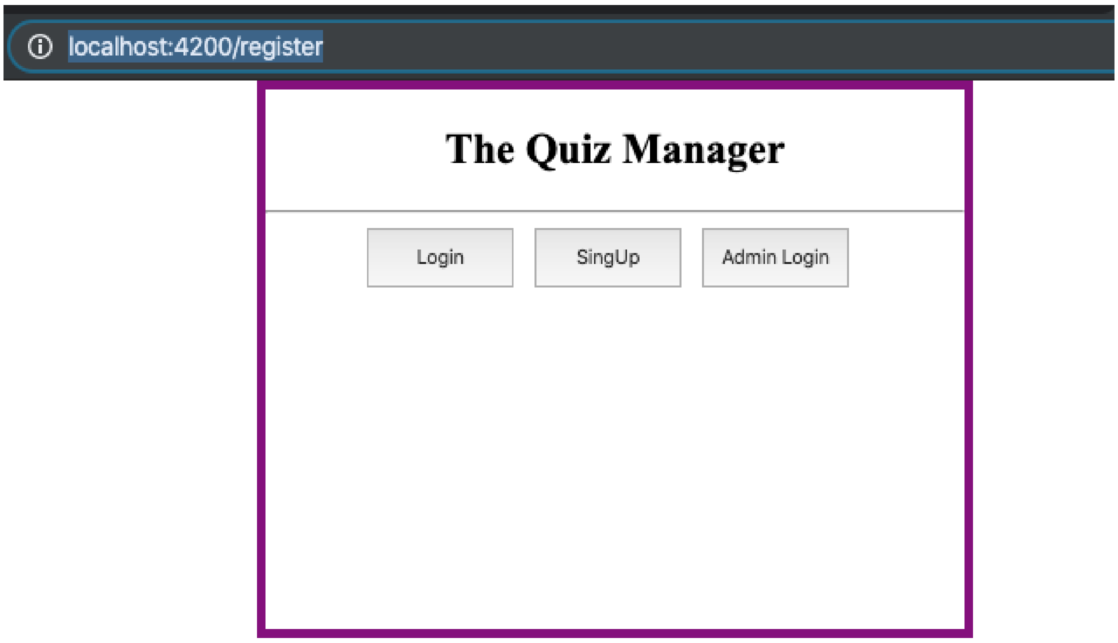
3. Conception



3.1 Home Page

Here we have implemented many service calls. The Spring MVC architecture makes the code very feasible to understand

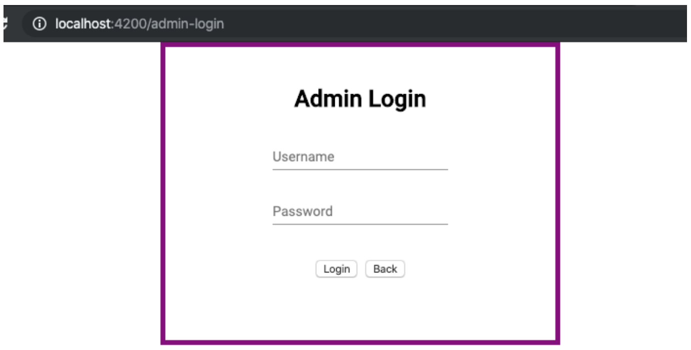
URL:http://localhost:4200/register



3.2 Admin Login

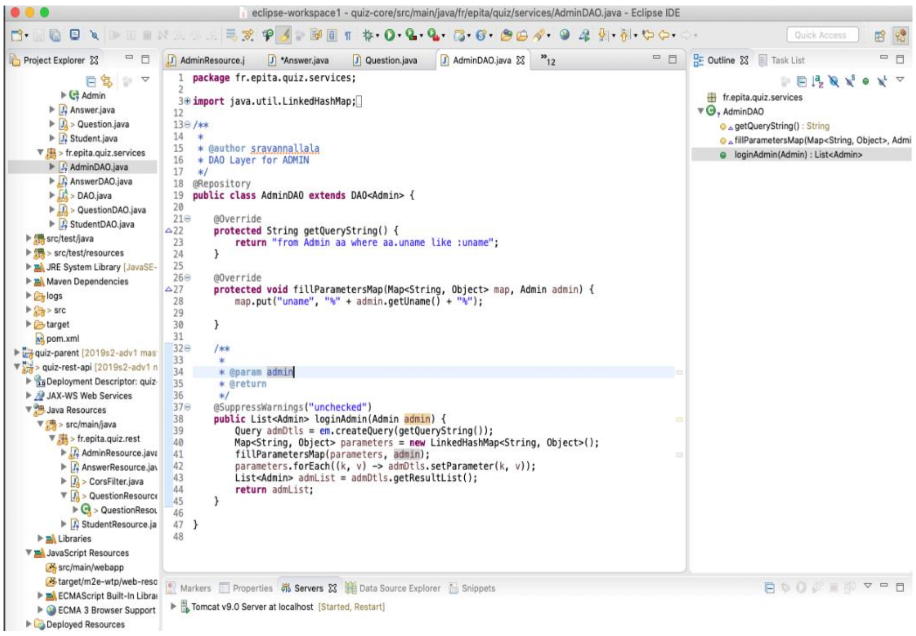
By providing this functionality, the admin can add questions with respective options and perform all actions required for the quiz application.

URL:http://localhost:4200/admin-login



Authentication:

The web application cannot move forward without the login. There are two logins for student and admin. And a register page for the student, in-order to login.

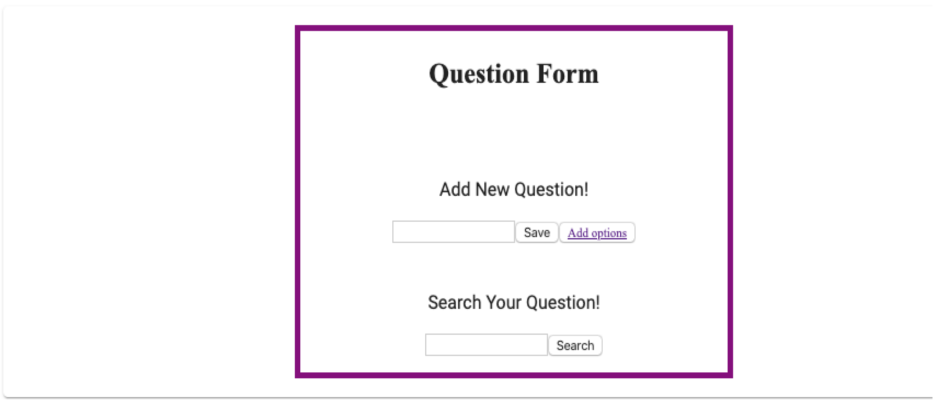

eclipse-workspace1 - quiz-core/src/main/java/fr/epita/quiz/services/AdminDAO.java - Eclipse IDE

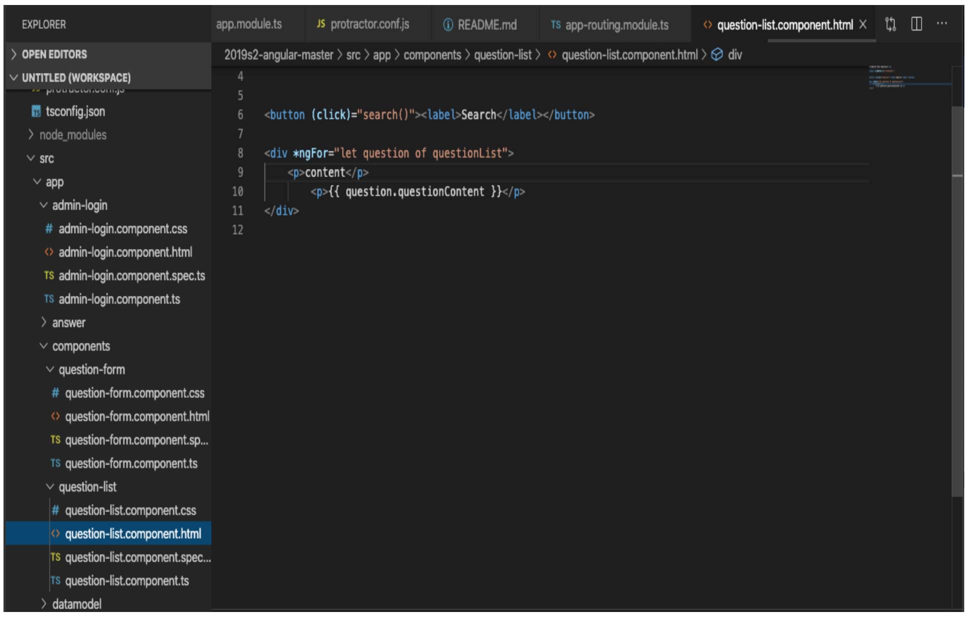
```java
package fr.epita.quiz.services;

import java.util.LinkedHashMap;

/**
 *
 * @author sravannallala
 * DAO Layer for ADMIN
 */
@Repository
public class AdminDAO extends DAO<Admin> {

    @Override
    protected String getQueryString() {
        return "from Admin aa where aa.uname like :uname";
    }

    @Override
    protected void fillParametersMap(Map<String, Object> map, Admin admin) {
        map.put("uname", "%" + admin.getUname() + "%");

    }

    /**
     *
     * @param admin
     * @return
     */
    @SuppressWarnings("unchecked")
    public List<Admin> loginAdmin(Admin admin) {
        Query admDtls = em.createQuery(getQueryString());
        Map<String, Object> parameters = new LinkedHashMap<String, Object>();
        fillParametersMap(parameters, admin);
        parameters.forEach((k, v) -> admDtls.setParameter(k, v));
        List<Admin> admList = admDtls.getResultList();
        return admList;
    }

}
```

Tomcat v9.0 Server at localhost [Started, Restart]

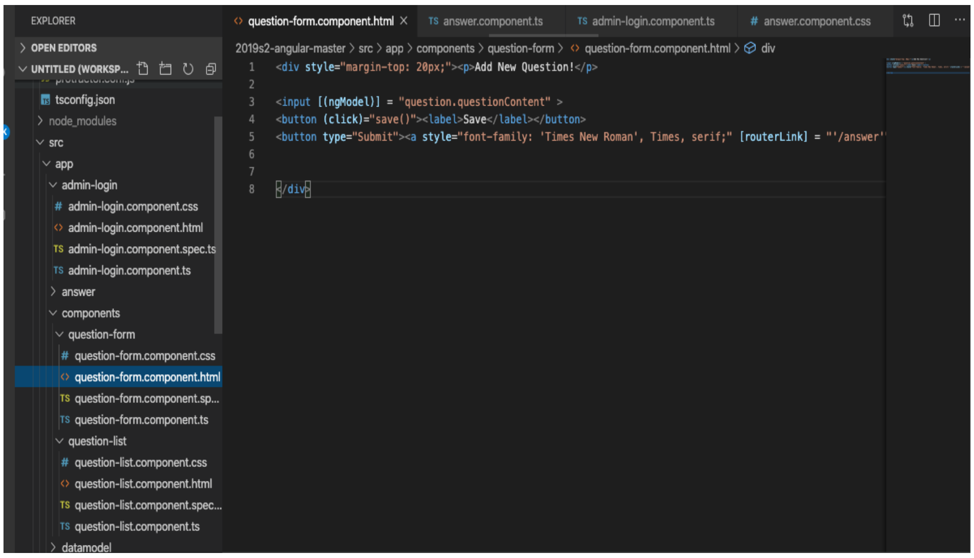
3.3 Add Questions

URL:http://localhost:4200/answer




EXPLORER
OPEN EDITORS
UNTITLED (WORKSPACE)
tsconfig.json
node_modules
src
app
admin-login
admin-login.component.css
admin-login.component.html
admin-login.component.spec.ts
admin-login.component.ts
answer
components
question-form
question-form.component.css
question-form.component.html
question-form.component.sp...
question-form.component.ts
question-list
question-list.component.css
question-list.component.html
question-list.component.spec...
question-list.component.ts
datamodel
app.module.ts
protractor.conf.js
README.md
app-routing.module.ts
question-list.component.html
2019s2-angular-master › src › app › components › question-list › question-list.component.html › div

```
<button (click)="search()"><label>Search</label></button>

<div *ngFor="let question of questionList">
    <p>content</p>
        <p>{{ question.questionContent }}</p>
</div>
```

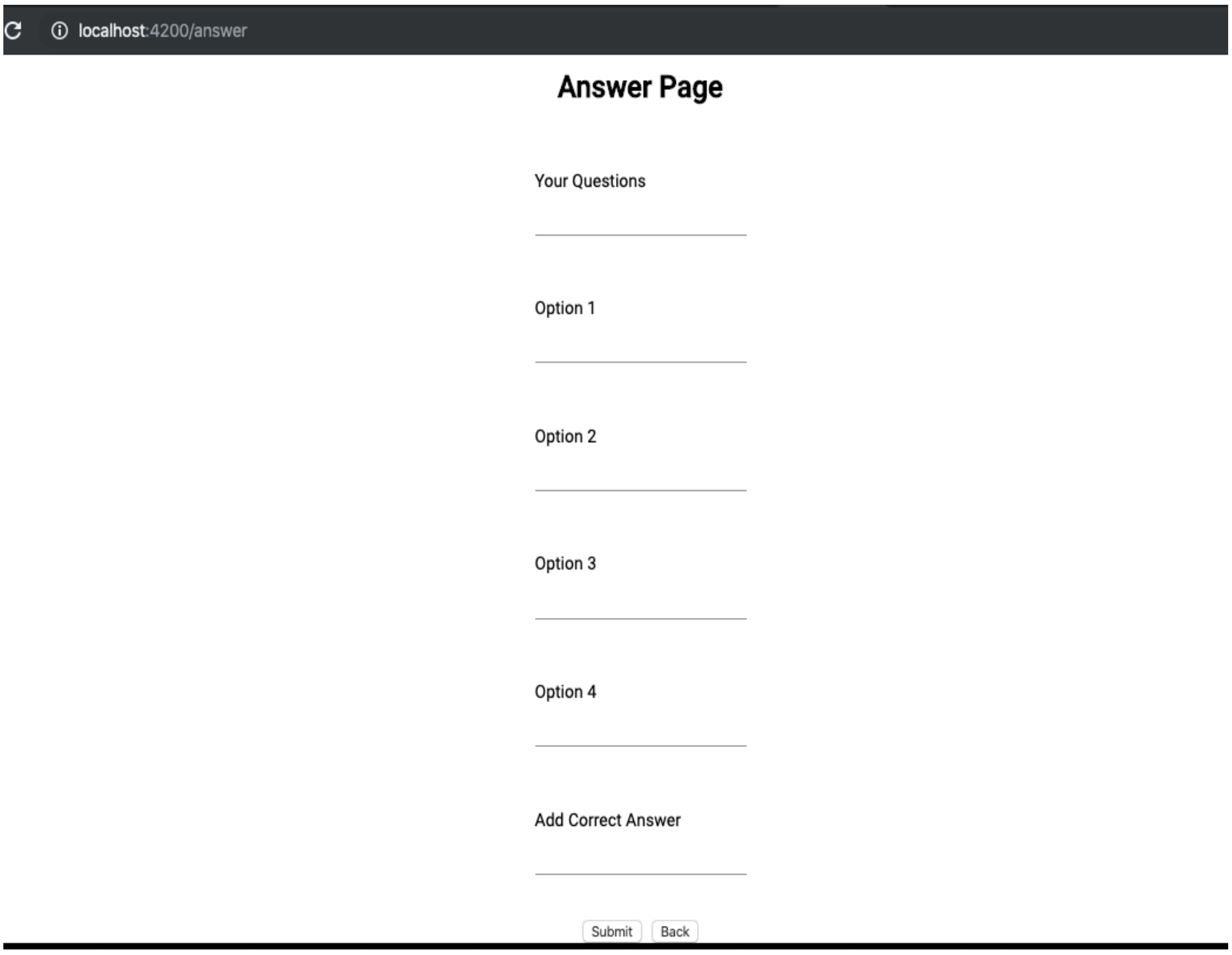


EXPLORER
OPEN EDITORS
UNTITLED (WORKSP...
tsconfig.json
node_modules
src
app
admin-login
admin-login.component.css
admin-login.component.html
admin-login.component.spec.ts
admin-login.component.ts
answer
components
question-form
question-form.component.css
question-form.component.html
question-form.component.sp...
question-form.component.ts
question-list
question-list.component.css
question-list.component.html
question-list.component.spec...
question-list.component.ts
datamodel
question-form.component.html
answer.component.ts
admin-login.component.ts
answer.component.css
2019s2-angular-master > src > app > components > question-form > question-form.component.html > div

```html
<div style="margin-top: 20px;"><p>Add New Question!</p>

<input [(ngModel)] = "question.questionContent" >
<button (click)="save()"><label>Save</label></button>
<button type="Submit"><a style="font-family: 'Times New Roman', Times, serif;" [routerLink] = "'/answer'


</div>
```

3.4 Register Answer

URL:http://localhost:4200/answer


localhost:4200/answer

Answer Page

Your Questions

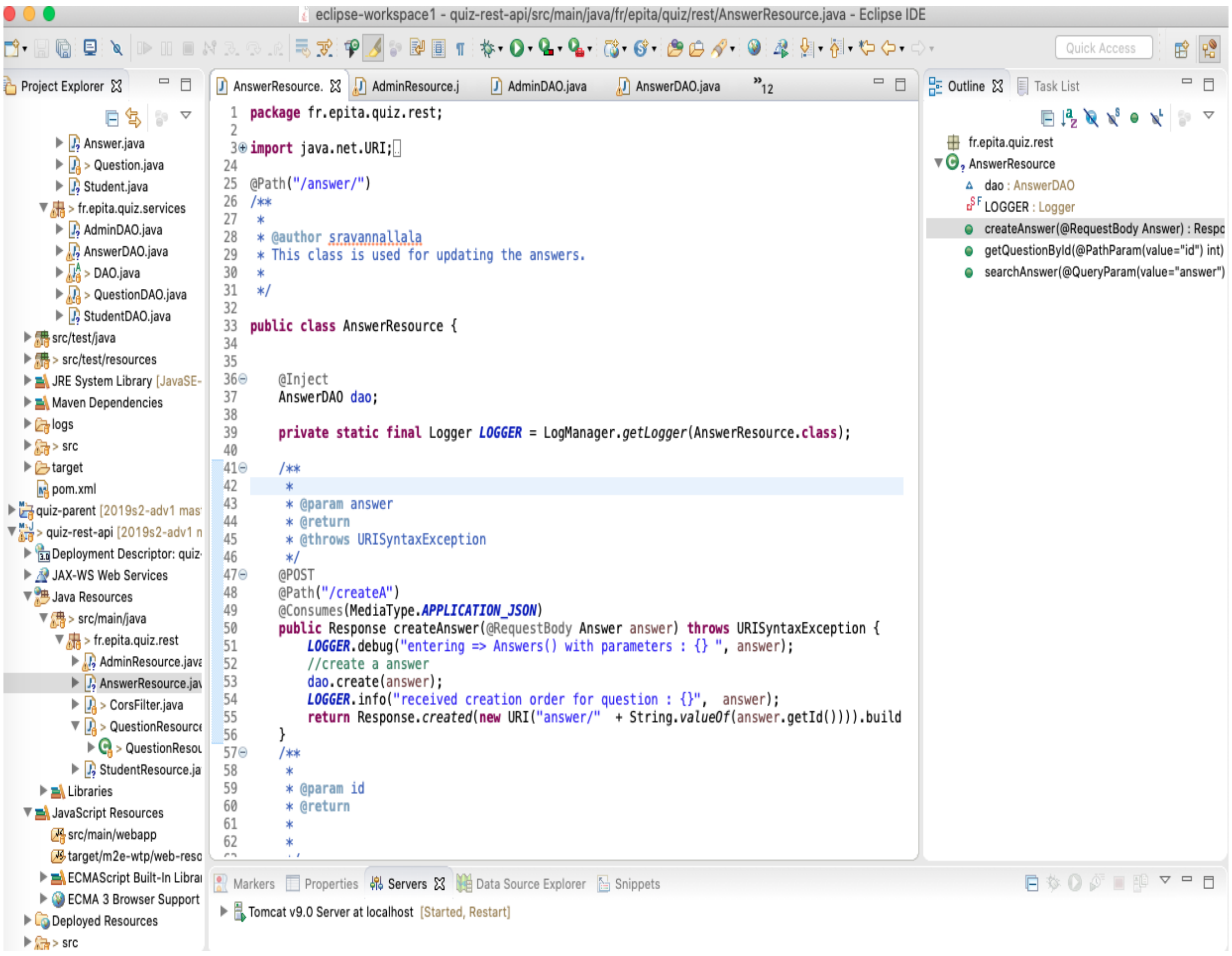
Option 1

Option 2

Option 3

Option 4

Add Correct Answer

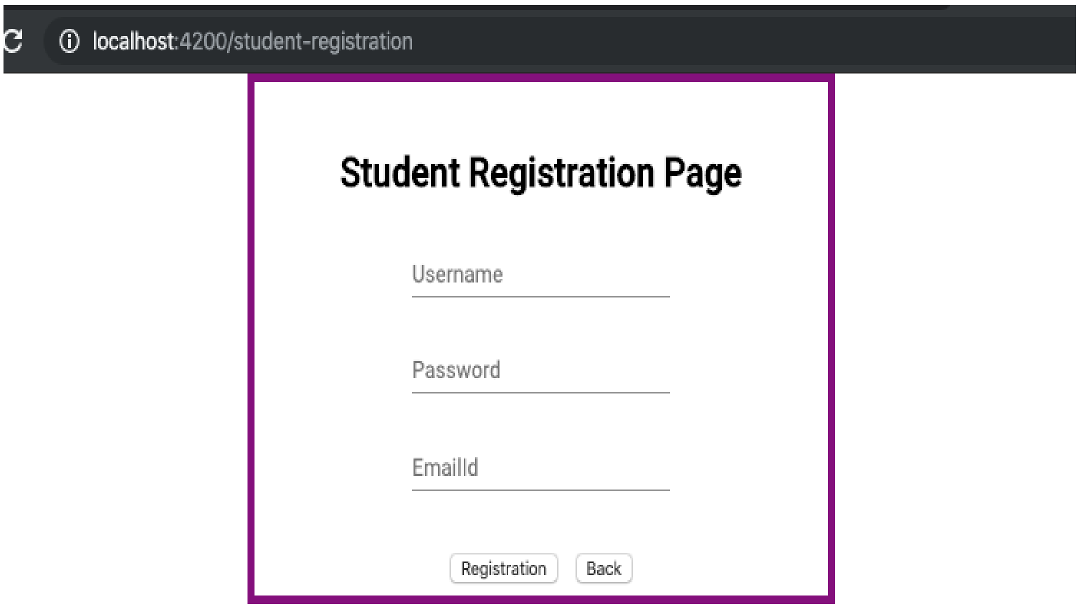


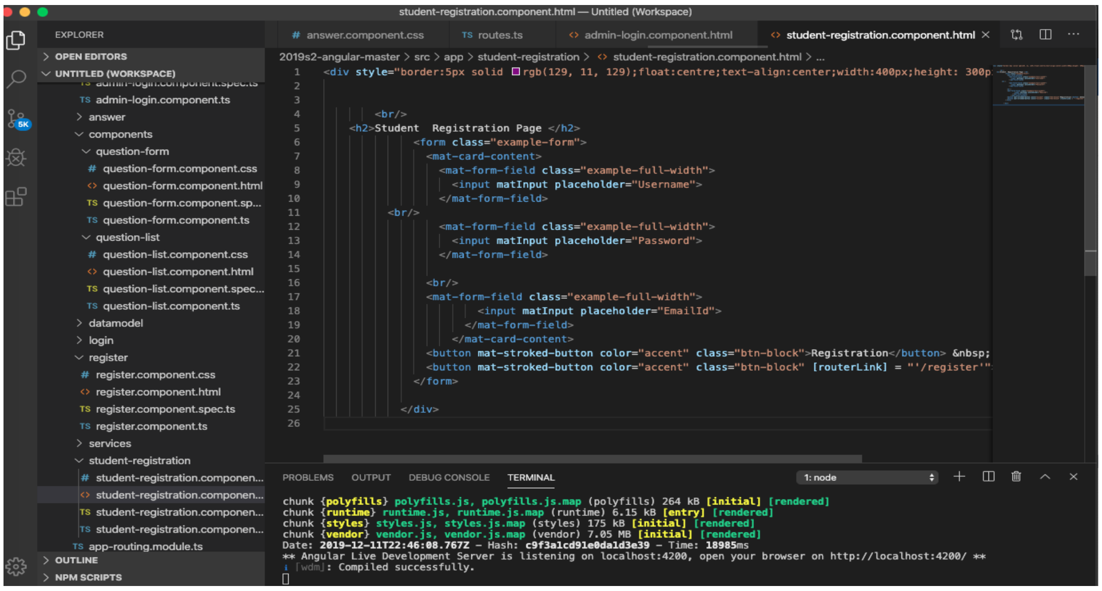
```
package fr.epita.quiz.rest;

import java.net.URI;

@Path("/answer/")
/**
 *
 * @author sravannallala
 * This class is used for updating the answers.
 *
 */

public class AnswerResource {


    @Inject
    AnswerDAO dao;

    private static final Logger LOGGER = LogManager.getLogger(AnswerResource.class);

    /**
     *
     * @param answer
     * @return
     * @throws URISyntaxException
     */
    @POST
    @Path("/createA")
    @Consumes(MediaType.APPLICATION_JSON)
    public Response createAnswer(@RequestBody Answer answer) throws URISyntaxException {
        LOGGER.debug("entering => Answers() with parameters : {} ", answer);
        //create a answer
        dao.create(answer);
        LOGGER.info("received creation order for question : {}",  answer);
        return Response.created(new URI("answer/"  + String.valueOf(answer.getId()))).build
    }
    /**
     *
     * @param id
     * @return
     *
     *
```

Tomcat v9.0 Server at localhost [Started, Restart]

3.5 Student Registration

URL:http://localhost:4200/student-registration


localhost:4200/student-registration
Student Registration Page
Username
Password
EmailId
Registration
Back

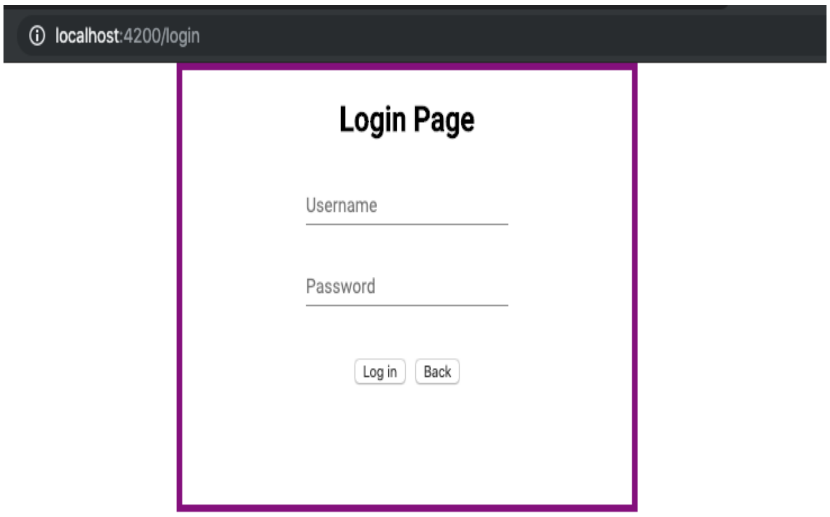

student-registration.component.html — Untitled (Workspace)
answer.component.css
routes.ts
admin-login.component.html
student-registration.component.html
2019s2-angular-master > src > app > student-registration > student-registration.component.html > ...
EXPLORER
OPEN EDITORS
UNTITLED (WORKSPACE)
admin-login.component.ts
answer
components
question-form
question-form.component.css
question-form.component.html
question-form.component.sp...
question-form.component.ts
question-list
question-list.component.css
question-list.component.html
question-list.component.spec...
question-list.component.ts
datamodel
login
register
register.component.css
register.component.html
register.component.spec.ts
register.component.ts
services
student-registration
app-routing.module.ts
OUTLINE
NPM SCRIPTS
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
1: node

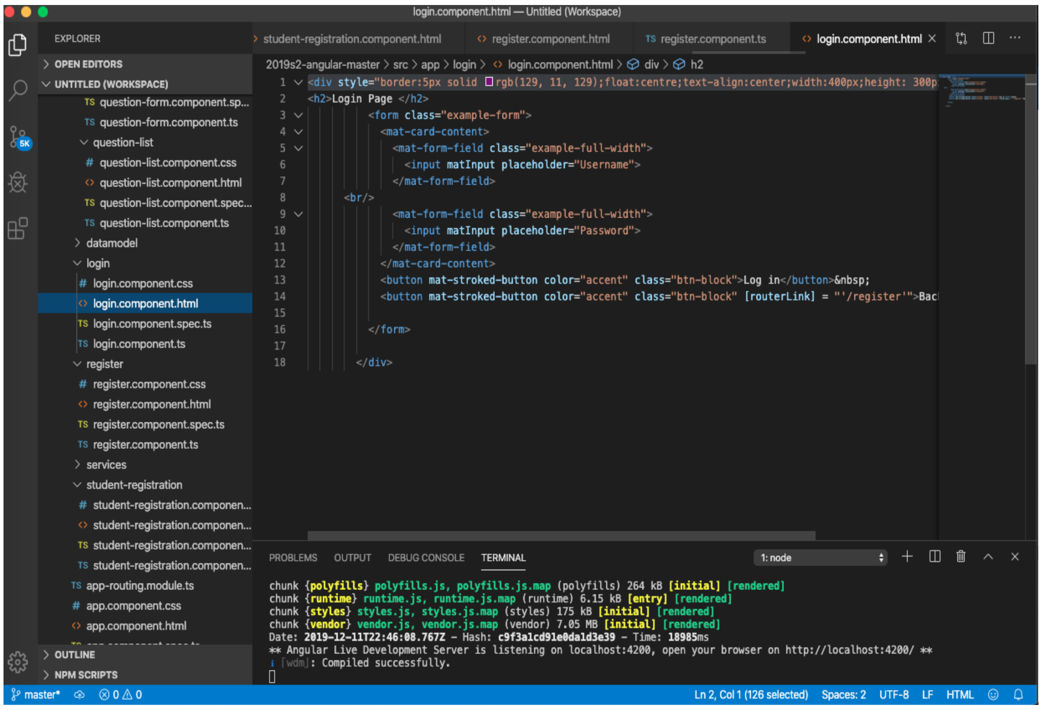
```html
<div style="border:5px solid rgb(129, 11, 129);float:centre;text-align:center;width:400px;height: 300p


        <br/>
    <h2>Student  Registration Page </h2>
              <form class="example-form">
                <mat-card-content>
                  <mat-form-field class="example-full-width">
                    <input matInput placeholder="Username">
                  </mat-form-field>
          <br/>
                  <mat-form-field class="example-full-width">
                    <input matInput placeholder="Password">
                  </mat-form-field>

                <br/>
                <mat-form-field class="example-full-width">
                        <input matInput placeholder="EmailId">
                      </mat-form-field>
                    </mat-card-content>
                <button mat-stroked-button color="accent" class="btn-block">Registration</button> &nbsp;
                <button mat-stroked-button color="accent" class="btn-block" [routerLink] = "'/register'"
              </form>

            </div>

```

```
chunk {polyfills} polyfills.js, polyfills.js.map (polyfills) 264 kB [initial] [rendered]
chunk {runtime} runtime.js, runtime.js.map (runtime) 6.15 kB [entry] [rendered]
chunk {styles} styles.js, styles.js.map (styles) 175 kB [initial] [rendered]
chunk {vendor} vendor.js, vendor.js.map (vendor) 7.05 MB [initial] [rendered]
Date: 2019-12-11T22:46:08.767Z - Hash: c9f3a1cd91e0da1d3e39 - Time: 18985ms
** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **
ℹ ｢wdm｣: Compiled successfully.
```

3.6 Student Login

URL:http://localhost:4200/login

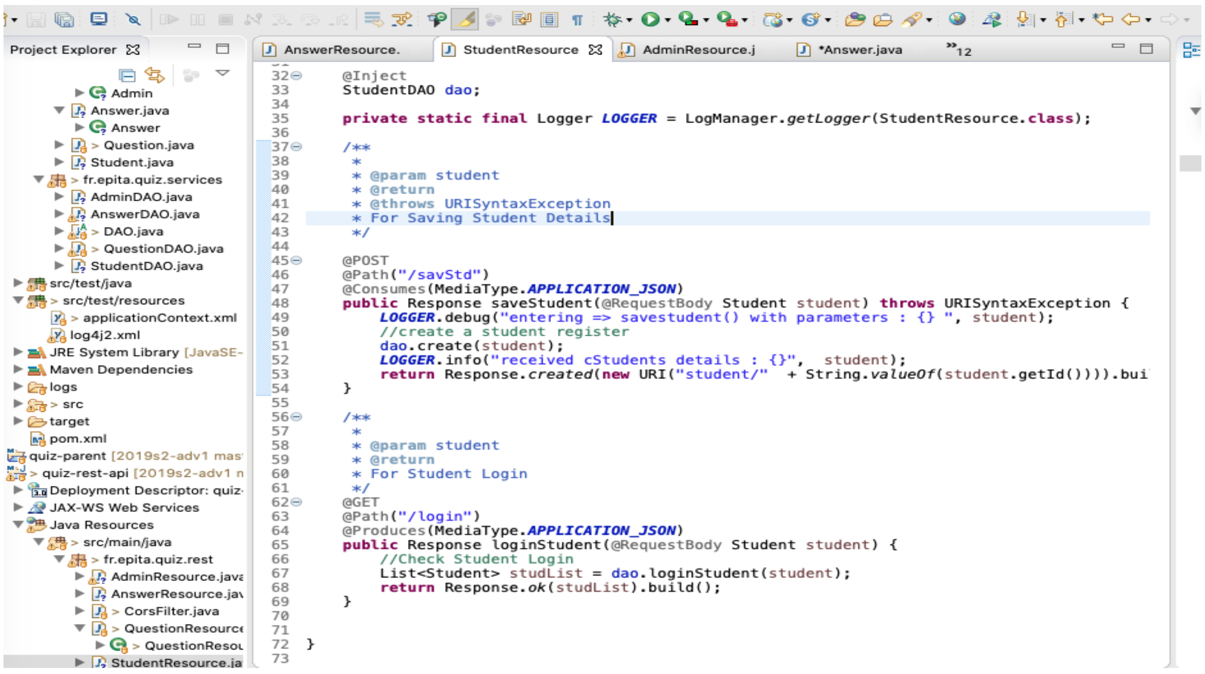



login.component.html — Untitled (Workspace)
student-registration.component.html
register.component.html
register.component.ts
login.component.html
2019s2-angular-master > src > app > login > login.component.html > div > h2
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
1: node
master*
Ln 2, Col 1 (126 selected)
Spaces: 2
UTF-8
LF
HTML

```html
<div style="border:5px solid rgb(129, 11, 129);float:centre;text-align:center;width:400px;height: 300p
<h2>Login Page </h2>
          <form class="example-form">
            <mat-card-content>
              <mat-form-field class="example-full-width">
                <input matInput placeholder="Username">
              </mat-form-field>
        <br/>
              <mat-form-field class="example-full-width">
                <input matInput placeholder="Password">
              </mat-form-field>
            </mat-card-content>
            <button mat-stroked-button color="accent" class="btn-block">Log in</button>&nbsp;
            <button mat-stroked-button color="accent" class="btn-block" [routerLink] = "'/register'">Bac

          </form>

        </div>
```

```
chunk {polyfills} polyfills.js, polyfills.js.map (polyfills) 264 kB [initial] [rendered]
chunk {runtime} runtime.js, runtime.js.map (runtime) 6.15 kB [entry] [rendered]
chunk {styles} styles.js, styles.js.map (styles) 175 kB [initial] [rendered]
chunk {vendor} vendor.js, vendor.js.map (vendor) 7.05 MB [initial] [rendered]
Date: 2019-12-11T22:46:08.767Z - Hash: c9f3a1cd91e0da1d3e39 - Time: 18985ms
** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **
i ｢wdm｣: Compiled successfully.
```

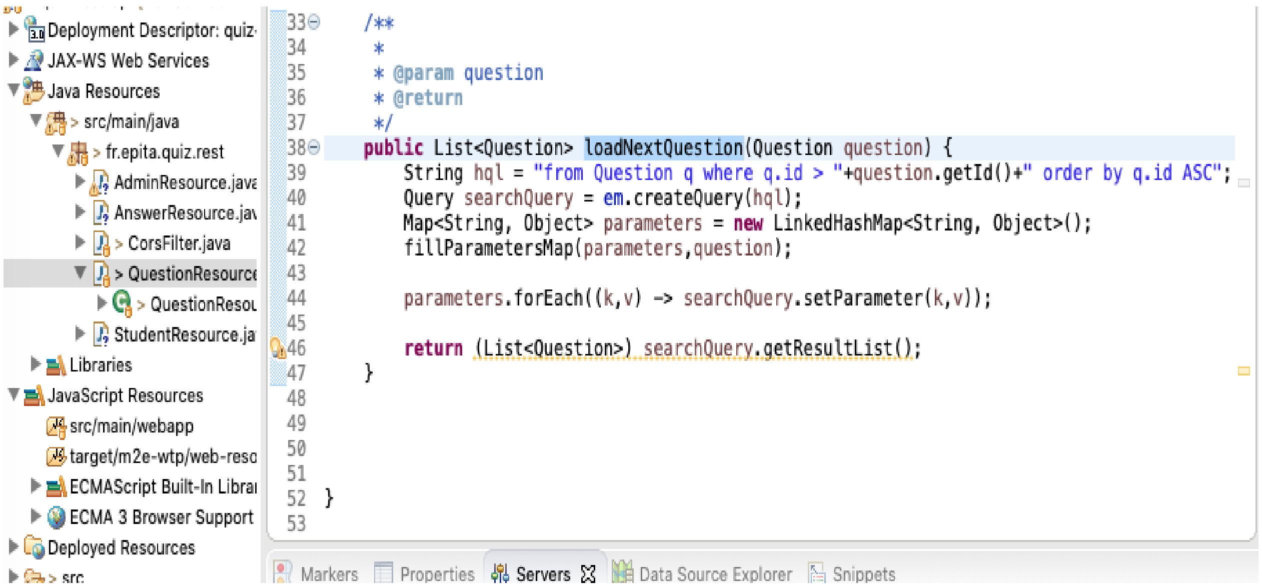
Student and Admin Controller:



```java
    @Inject
    StudentDAO dao;

    private static final Logger LOGGER = LogManager.getLogger(StudentResource.class);

    /**
     *
     * @param student
     * @return
     * @throws URISyntaxException
     * For Saving Student Details
     */

    @POST
    @Path("/savStd")
    @Consumes(MediaType.APPLICATION_JSON)
    public Response saveStudent(@RequestBody Student student) throws URISyntaxException {
        LOGGER.debug("entering => savestudent() with parameters : {} ", student);
        //create a student register
        dao.create(student);
        LOGGER.info("received cStudents details : {}",  student);
        return Response.created(new URI("student/"  + String.valueOf(student.getId()))).bui
    }

    /**
     *
     * @param student
     * @return
     * For Student Login
     */
    @GET
    @Path("/login")
    @Produces(MediaType.APPLICATION_JSON)
    public Response loginStudent(@RequestBody Student student) {
        //Check Student Login
        List<Student> studList = dao.loginStudent(student);
        return Response.ok(studList).build();
    }


}
```

3.7 Verifying the MCQ-Choice questions and providing the answers:

Here we have implemented the model class to reuse the functionality for all questions and making the code easily accessible.



```java
    /**
     *
     * @param question
     * @return
     */
    public List<Question> loadNextQuestion(Question question) {
        String hql = "from Question q where q.id > "+question.getId()+" order by q.id ASC";
        Query searchQuery = em.createQuery(hql);
        Map<String, Object> parameters = new LinkedHashMap<String, Object>();
        fillParametersMap(parameters,question);

        parameters.forEach((k,v) -> searchQuery.setParameter(k,v));

        return (List<Question>) searchQuery.getResultList();
    }





}
```

Providing the result to student after the test:

Here we will be evaluating the student marks based on his answers.

4 Future Scopes

- Edit functionality Like Delete and Update.
- Student Database with Scores.
- Advanced Security features.
- Advanced UI.
- Advanced Navigation features.
- Timer – Timeout Functionality.

5 Bibliography

http://thomas- broussard.fr/work/java/courses/project/advanced.xhtml

http://stackoverflow.com

https://www.youtube.com/watch?v=ZfOljMf7L-o&list=PLOUYE-
KsFYc_0ekC_3EEAIkiRuEIaymmJ