PROJECT REPORT

**DEVELOPING A BACKEND ADMIN FOR LEARNER’S ACADEMY**

Submitted in partial fulfillment of the requirement for the award of degree

OF

**FULL STACK JAVA DEVELOPER**

**PHASE-4 SESSION (16.01.2021-31.01.2021**)

**UNDER THE GUIDENCE OF**

**MRS. BHAWNA GUNWANI**

**COURSE INSTRUCTOR (SIMPLILEARN)**

**SUBMITTED BY**

**MANISH KUMAR ARORA**



**AT STUDY CENTRE**

**Learning Management System**

**SIMPLILEARN**

**CONTENTS**

1. **Introduction**
   1. **Project Details**
   2. **Developer Details**
   3. **Problem Description**
2. Sprint Plan
   1. **Number of Sprints**
   2. **Duration of Sprint**
   3. **User Stories**
   4. Task achieved in each Sprint
3. Algorithm of application
4. Flow Charts of application
5. **Implementation**
   1. **Software and Tools used**
   2. **Methodology**
   3. **Coding**
6. **Output Test - Output Screens**
7. **Core Concept used in the Project**
8. **Links to the GitHub repository to track and verify the project**
9. **Conclusion**
   1. **Summary**
   2. **Future Enhancements**
   3. **Unique Selling Points(USPs)**

**INTRODUCTION**

1. **Project Detail:**

The project “**Online Test Application**” aims to increase the scope of user’s interactions by developing dynamic web pages and menu driven options in order to create an application that enables users to provide online tests, review them, and display the results. The quiz application accepts the questions in JSON format. So, you can easily send the JSON from the server in the pre-defined format and the Angular quiz application will render the quiz at the client side. Once the questions are loaded, user can answer the questions and the events are being captured by the same component. When user finally submits the quiz, you may submit the answers to the server in JSON format. Or load the questions with answers and evaluate the users’ answers to show the quiz result immediately.

1. **Developer Detail:**

Manish Kumar Arora

Technical Support Associate, Engineering

MBA-IT, B.TECH-CSE

1. **Problem Description:**

This system contains three main modules: Quiz, Review, and Result. The quiz section of the online test application accepts the questions in JSON format. The JSON file can be easily shared from the server in the pre-defined format. The application renders the test at the client-side.

The “Review and display result” section allows users to declare the results immediately. We can simply call another JSON with the answers in it and evaluate and display the results immediately.

**SPRINT PLAN**

1. **Number of Sprints: 1**
2. **Duration of each sprint: One Week**
3. **User Stories: 3 user stories.**

Sprint1: 9.11.2020 – 15.11.2020

User Story1:

**Title: Quiz View**

**Priority: 1**

**Estimate: 1 D**

**As an** End User

**I want to** have initial Login Page

**So that I can** login into dashboard securely.

User Story2:

**Title: Review**

**Priority: 2**

**Estimate: 1 D**

**As an** End User

**I want to** have all different tasks options, set up class, student, subject and teacher list into database

**So that I can** choose between the multiple tasks to perform operations of assigning classes for subject, assigning teacher to class for a subject, generate and view class report.

User Story3:

**Title: Result**

**Priority: 1**

**Estimate: 2 D**

**As an** End User

**I want to** set up a master list for classes, subjects and teachers separately.

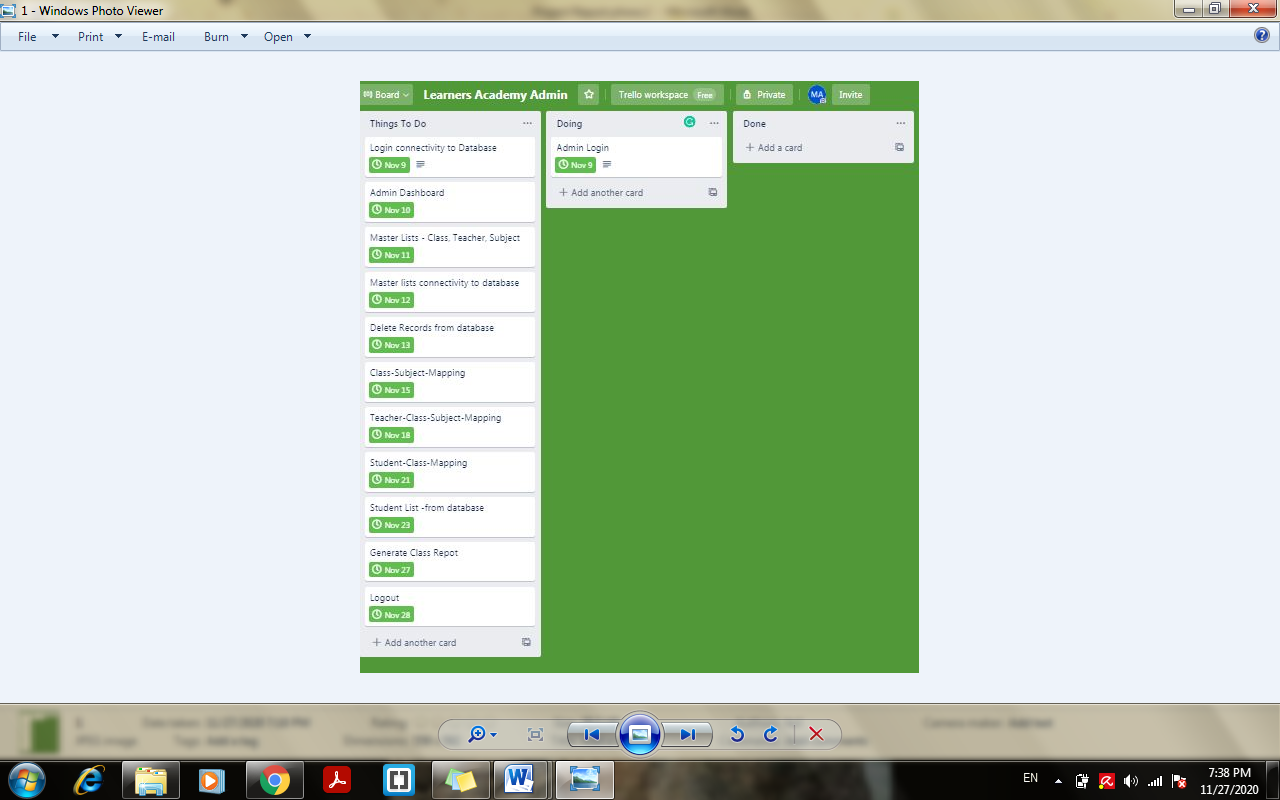
**So that I can** add new class, subjects and teacher to the list.

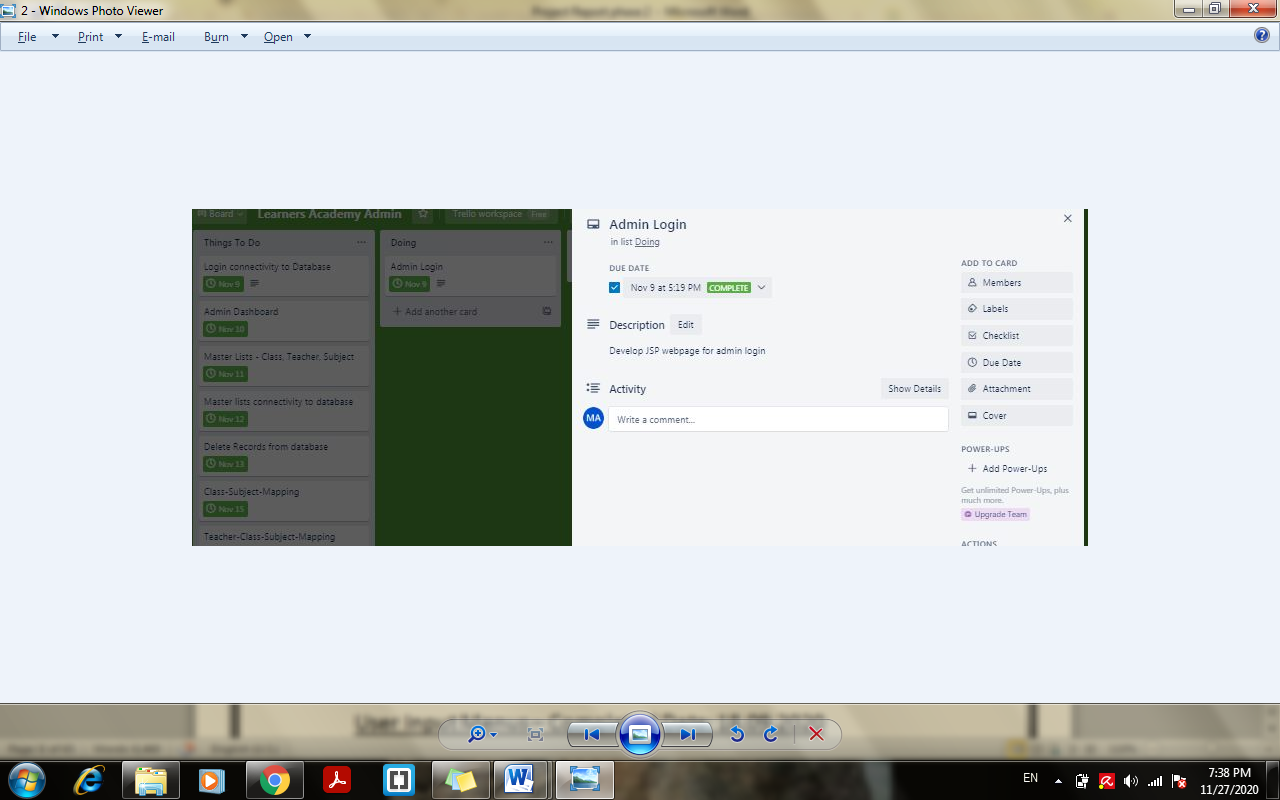
1. **Task achieved in each sprint:**
2. **Sprint One – Developed interactive admin login page, admin dashboard, web pages for first three task i.e. set up master list of class, subjects and teachers and created database table in with xampp for admin login, class list, subject and teacher along with that connected webpages to database using servlets. After that, developed JSP code to fetch records from database and display with deletion operation for class, subject and teacher on same page where admin wants to add new records.**
3. **Sprint Two – Developed pages for class-subject mapping through database to assign class for a subject and teacher-class-subject mapping page to assign teacher to different class for a subject using servlet, SQL and JSP pages. Developed web pages for student-class mapping and creating servlet for that to connect it to database in order to add new student to a particular class.**
4. **Sprint Three – Developed web pages to dynamically get master list of all students’ assigned to particular class, and fetch complete class information from database using servlets and webpages and SQL and finally developed logic to logout from dashboard activity board.**

**Detailed Screenshots for each user stories used in scrum planning in each sprint:**

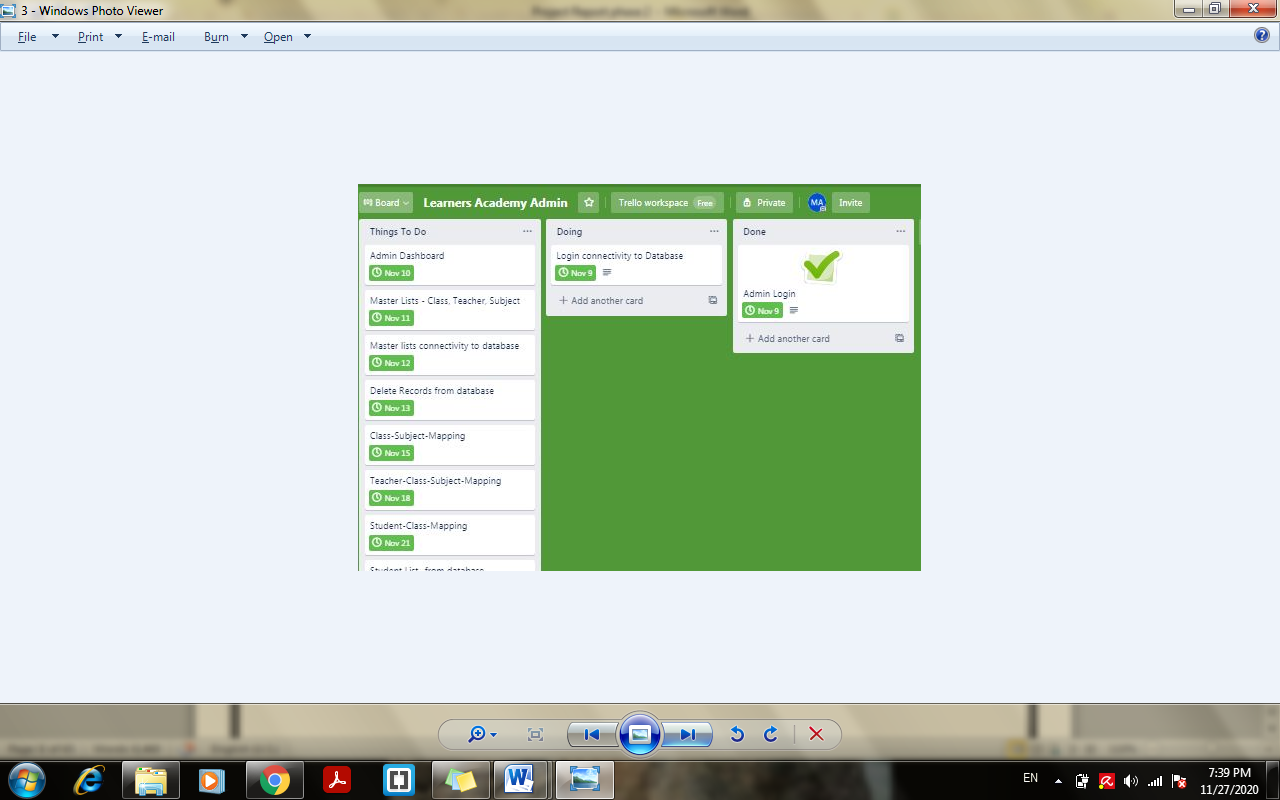
**Sprint 1: Date: 09.11.2020 – 15.11.2020**

1. **9.11.2020 – Admin Page creation starts:**

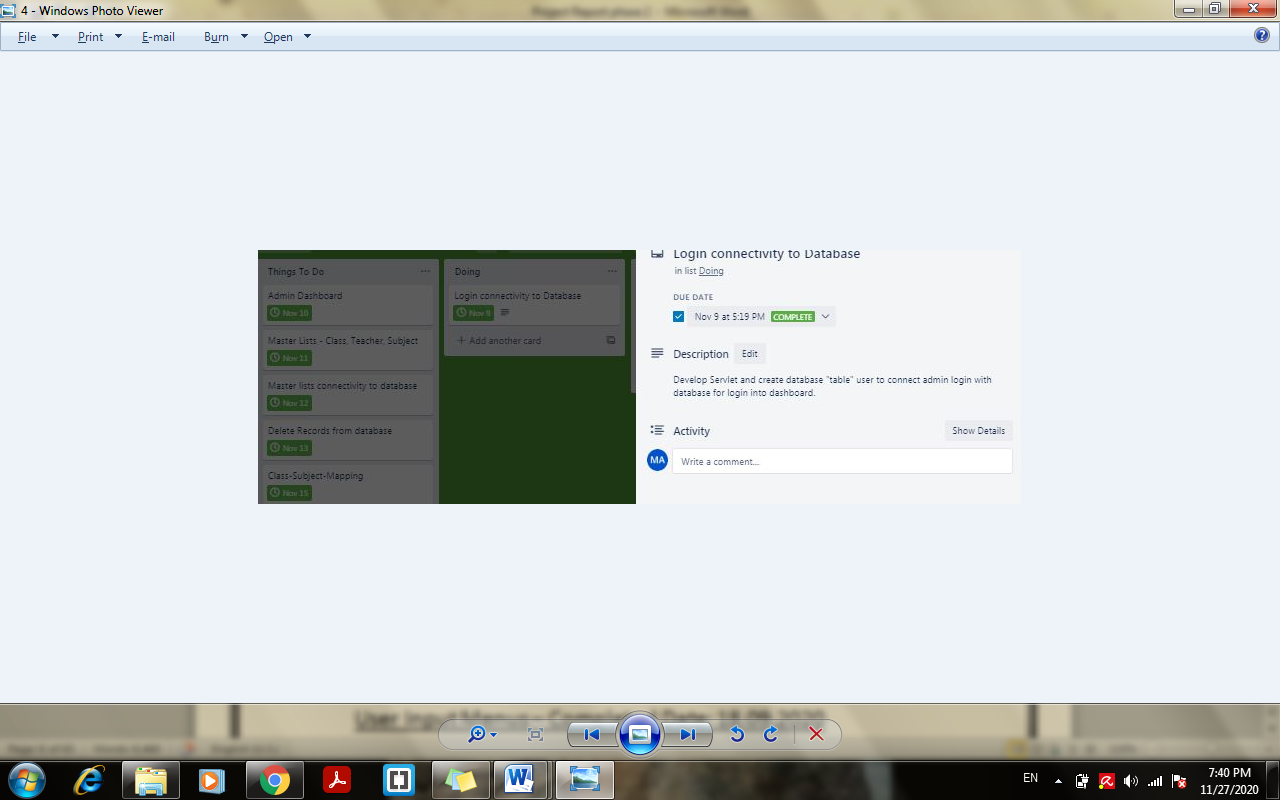


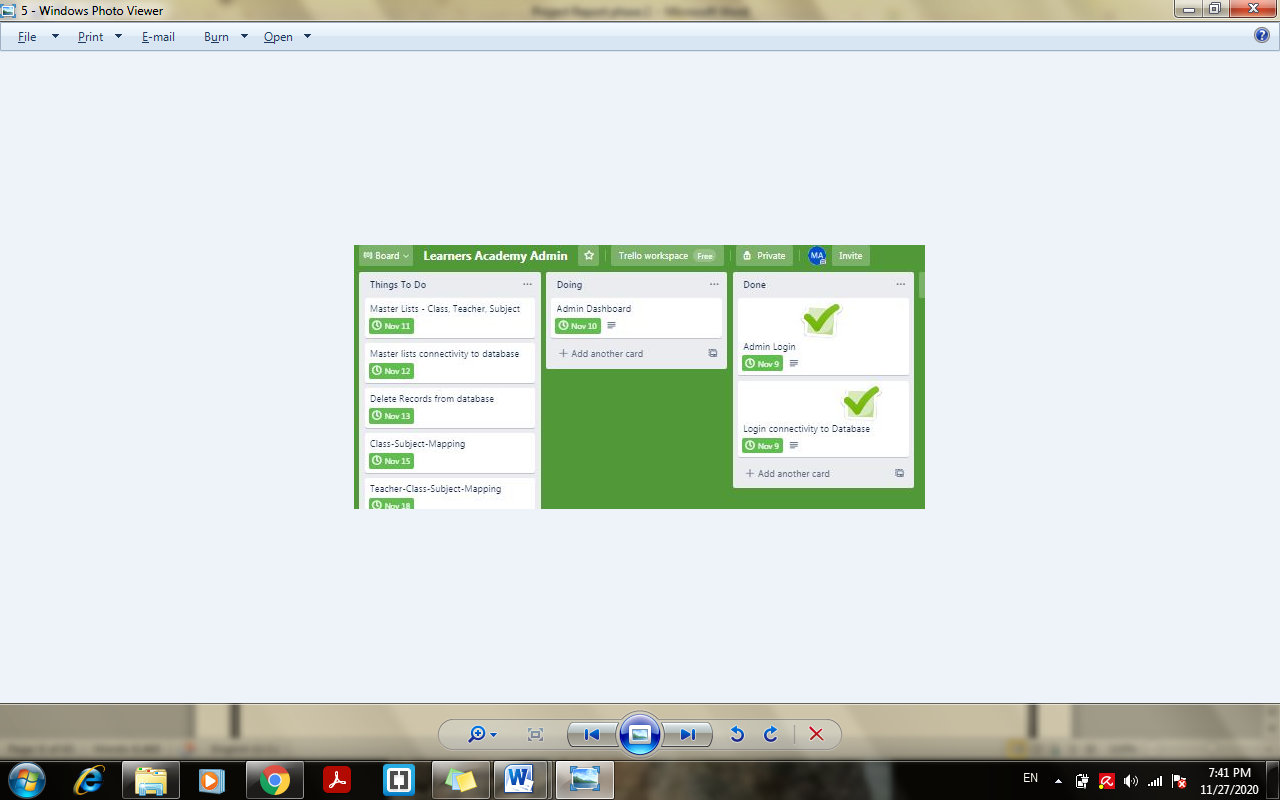


1. **9.11.2020 – Admin Page creation completed**

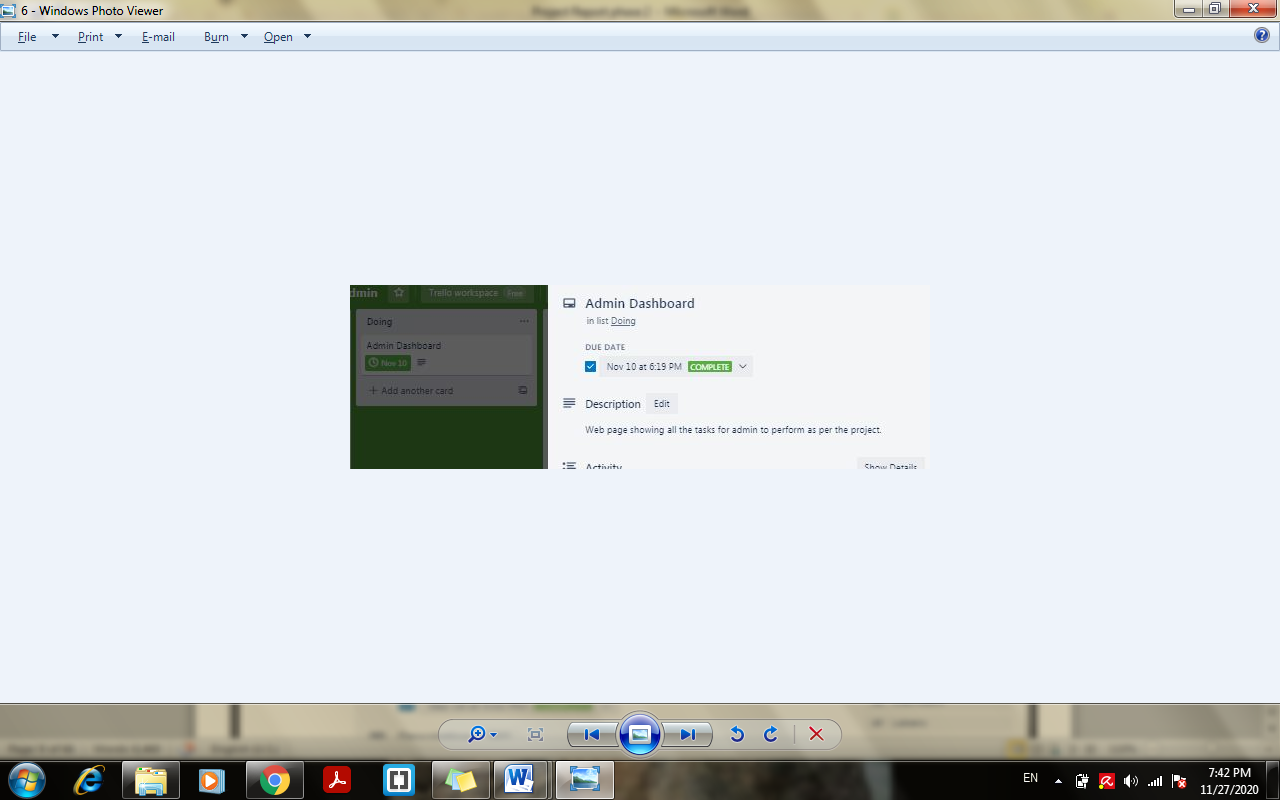


1. **9.11.2020 – Connectivity to database using servlet starts and completed**

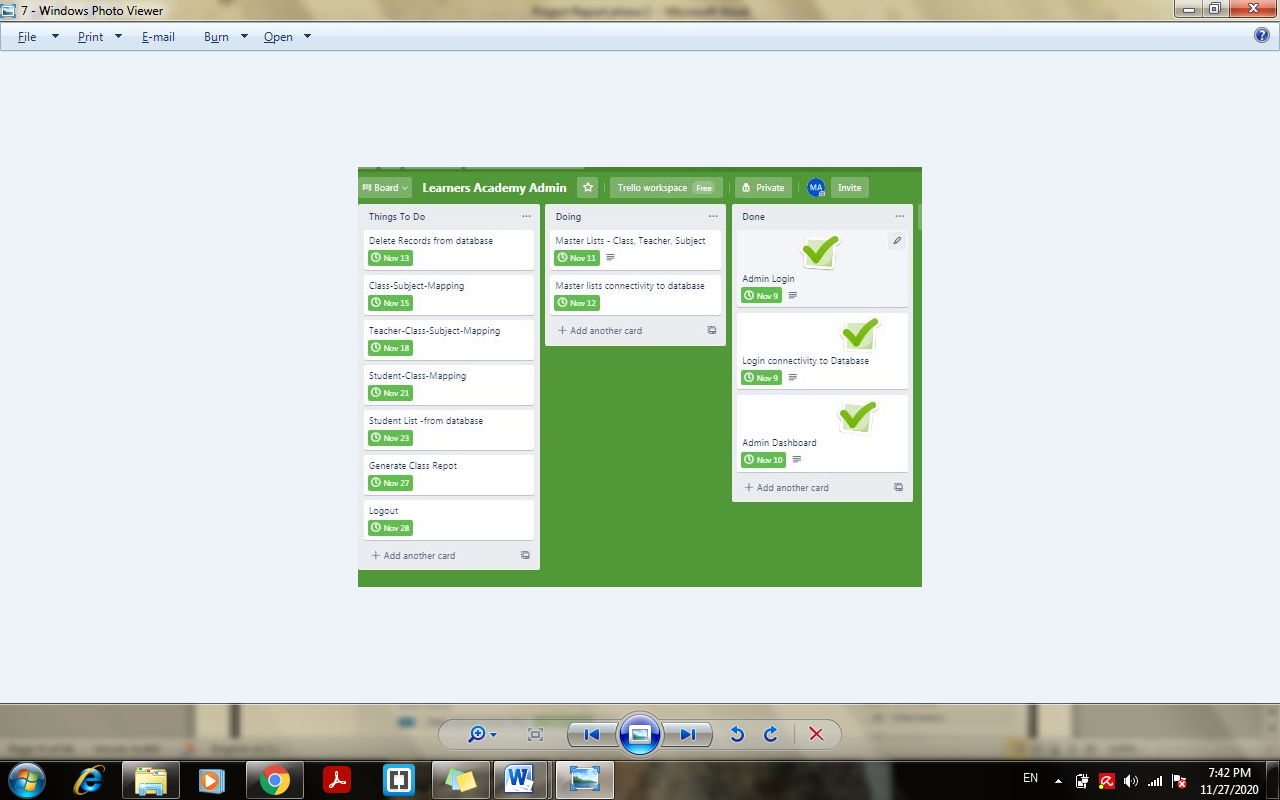




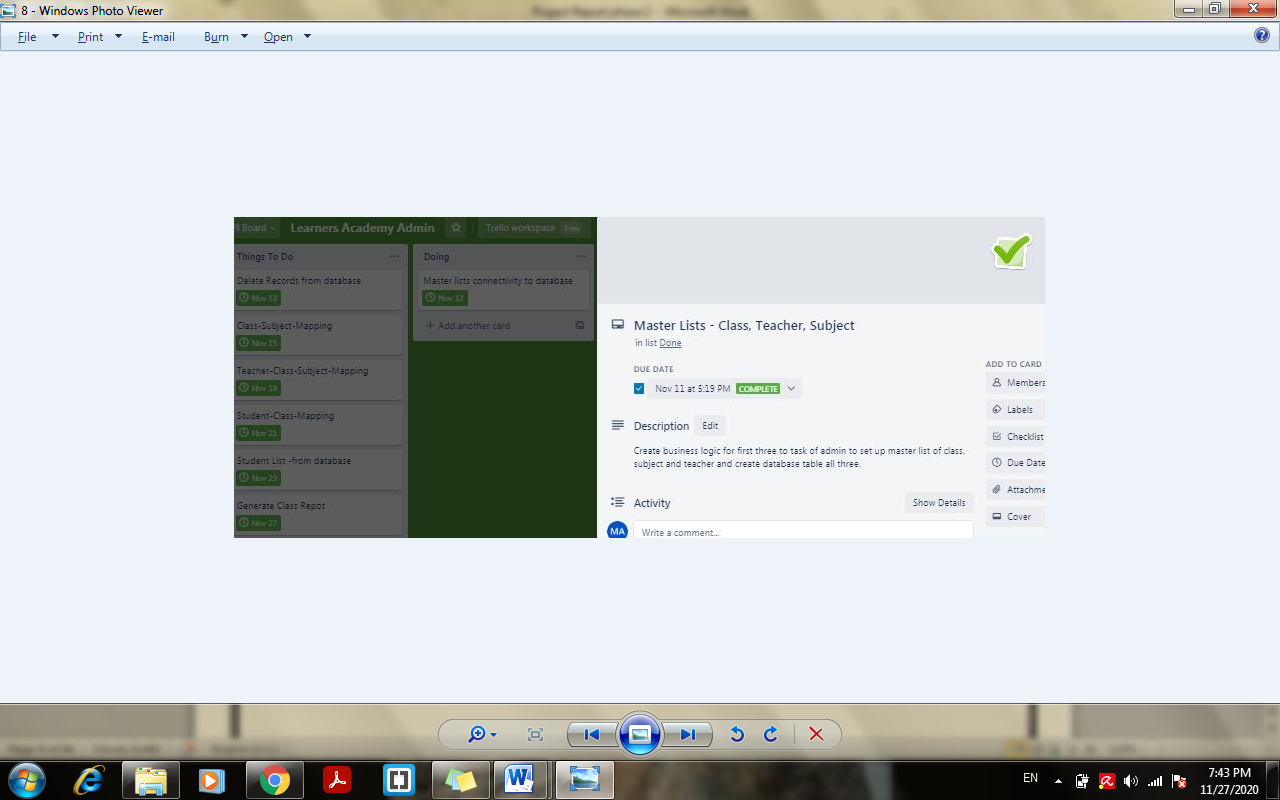
1. **10.11.2020 – Admin dashboard starts and completed**



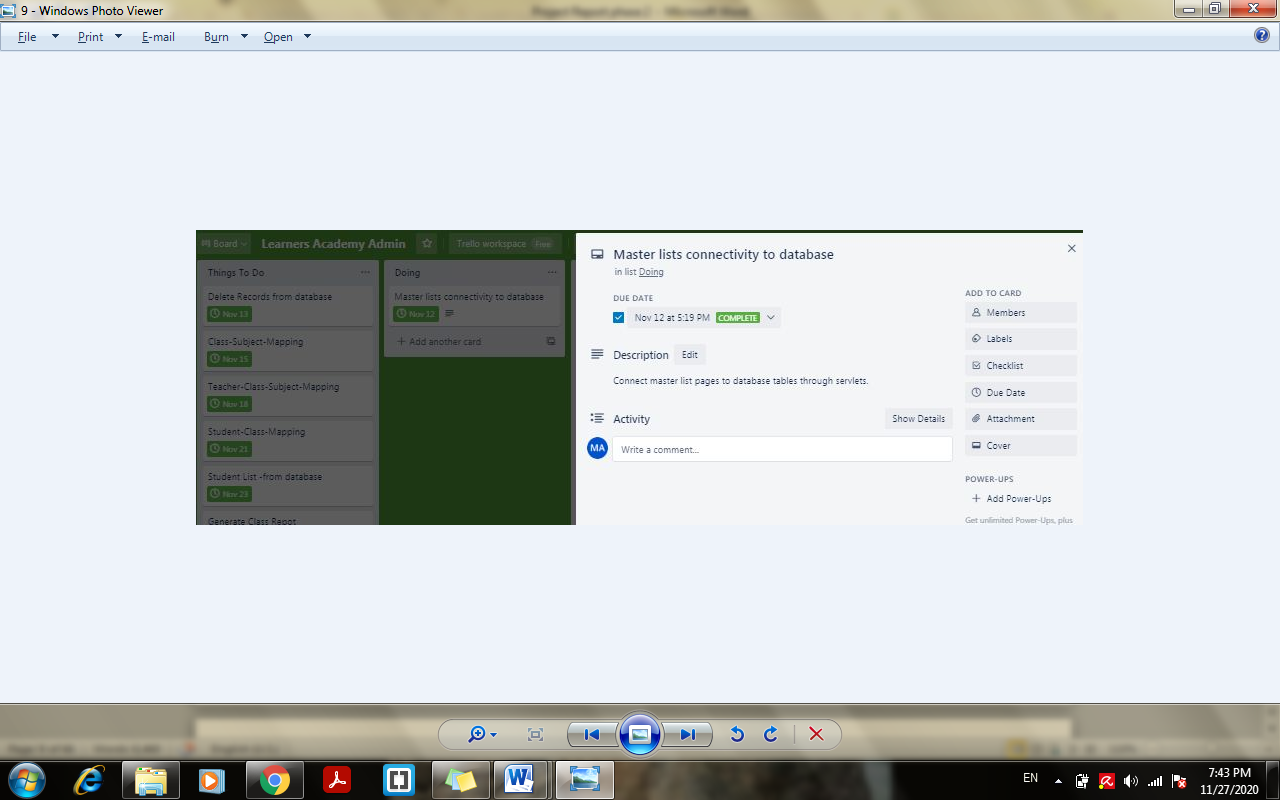
1. **11.11.2020 – Class teacher and student master list web page creation starts**



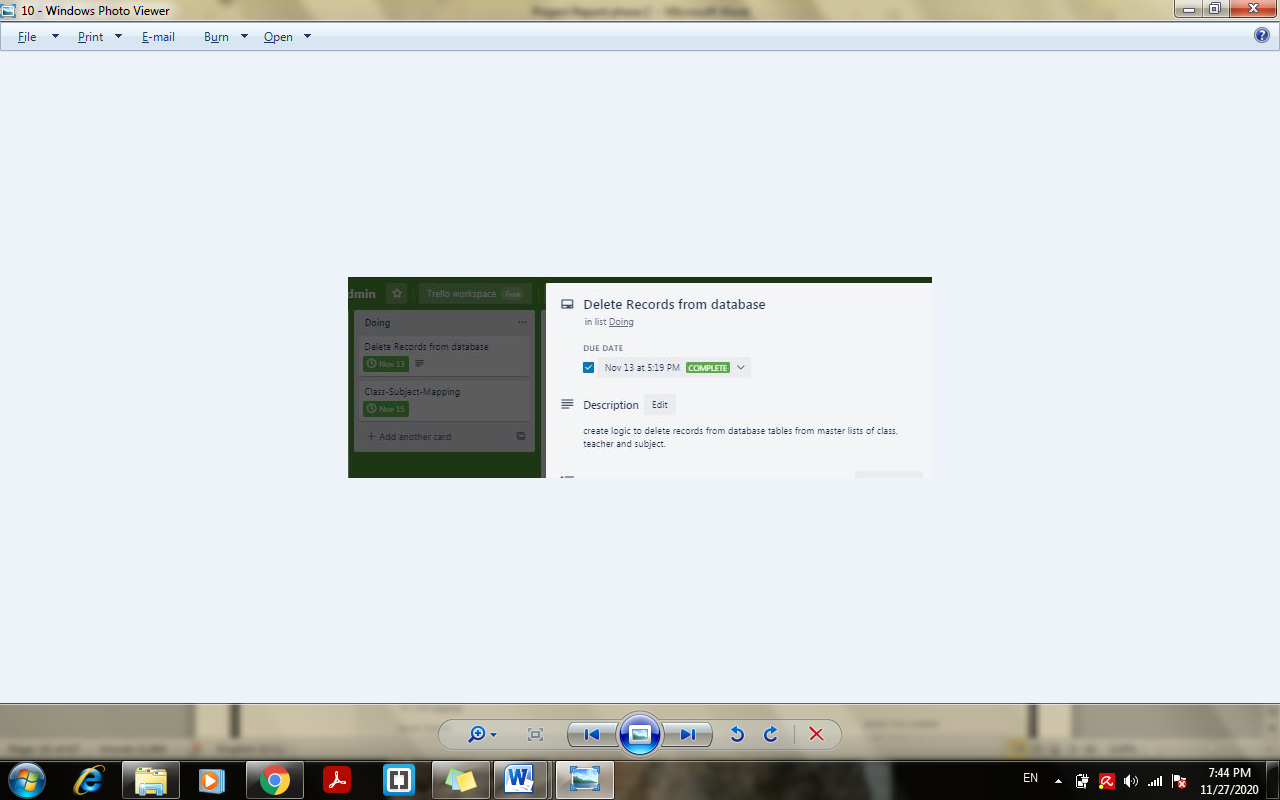
1. **11.11.2020 – Class teacher and student master list web page creation completed**



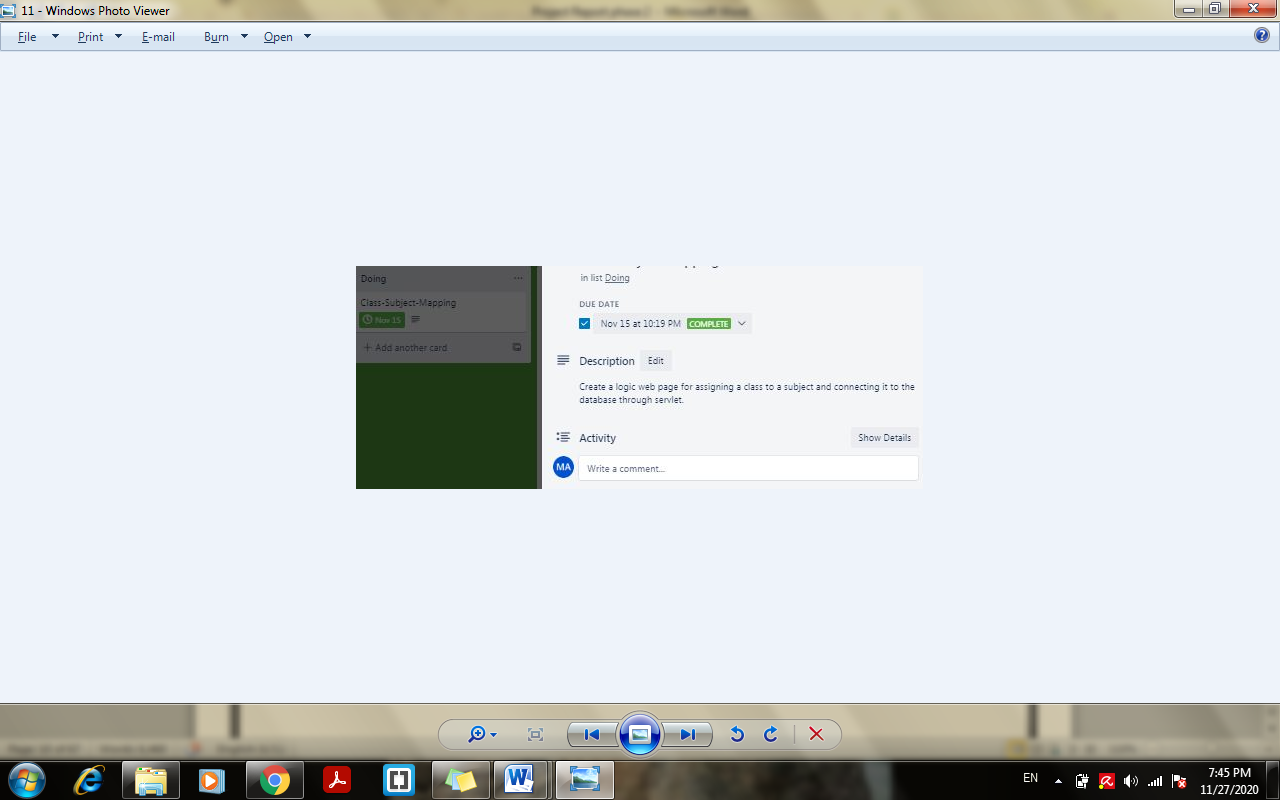
1. **12.11.2020 – Class, teacher ,and student master list connectivity to database completed**



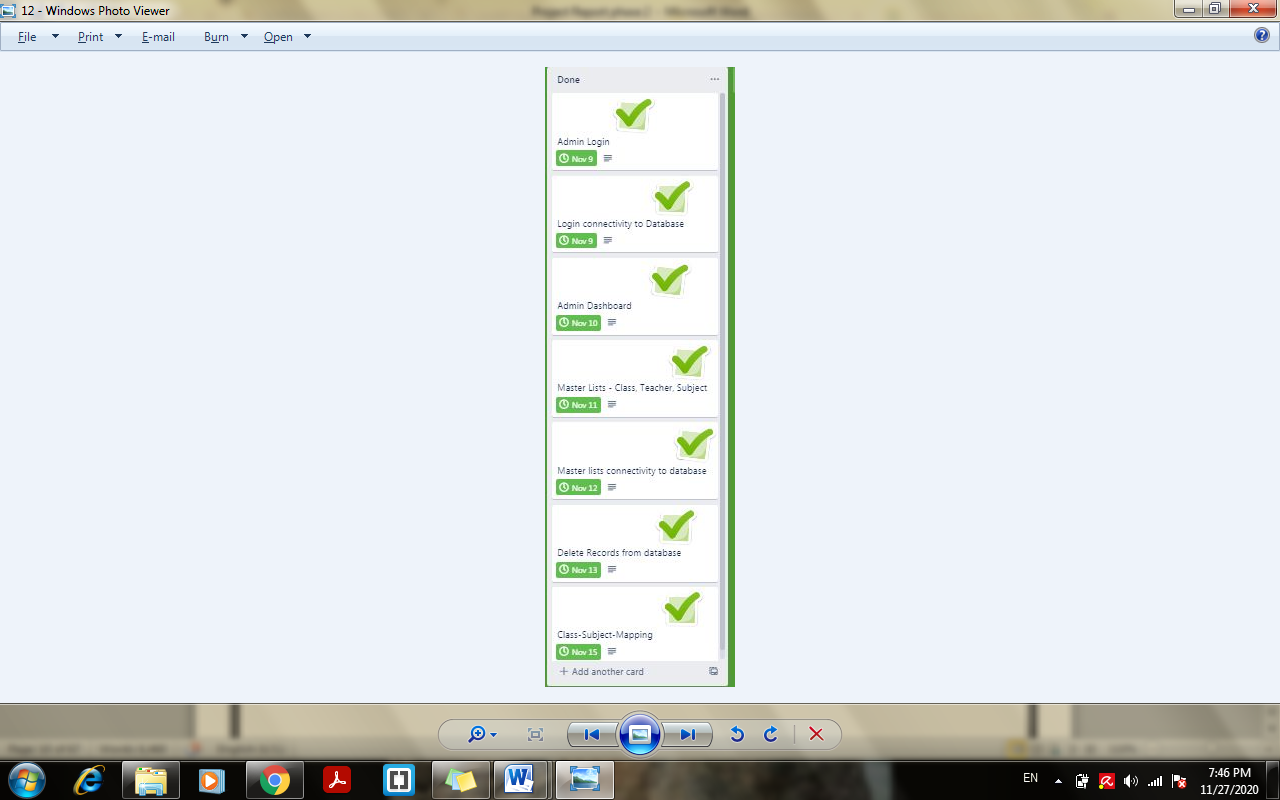
1. **13.11.2020 – Class teacher and student record deletion option from page completed**



1. **15.11.2020 – Class Subject mapping completed**



Sprint 1– Completed Date: 15.09.2020



**ALGORITHM OF APPLICATION**

1. **Login Page “Index.jsp” Algorithm:**
2. Page run on server - starts
3. Login page display: Enter username “admin”and password “admin”
4. Enter Submit -> go to :: B
5. **Servlet “LoginAdmin” LoginAdmin.java – Algorithm:**
6. Username , password-> takes the parameter from login page
7. Variable con- > connects with database
8. Variable rs - > execute query to check username and password from table “users”
9. IF username and password exist in table:: goto:: C
10. IF username and password not matched :: Print Invalid username and password
11. Click – try again to go back to login and re-try
12. **Admin Dashboard Page “success.jsp” Algorithm:**
13. Choose from multiple tasks to perform admin tasks:
    * + 1. IF “Set up master list of all subjects”->goto::D
        2. IF “Set up master list of all teachers”->goto::E
        3. IF “Set up master list of all classes”->goto::F
        4. IF “Assign classes for subject”->goto::G
        5. IF “Assign teachers to a class for a subject”-> goto::H
        6. IF “Add student and assign class”-> goto::I
        7. IF “Get a master list of all students”-> goto::J
        8. IF “View class report “->goto::K
14. IF Logout ->goto::L
15. **Subject addition page “subject.jsp” Algorithm:**
16. Display Pre-Existing Subjects if any below form with Delete Option using jdbc connectivity
17. IF -> Delete -> goto::D.4
18. Enter New subject
19. Click Submit-> goto:: D.1

**D.1 Servlet “addSubject” AddSubject.java Algorithm:**

1. Subname - > takes data from D
2. Con -> connects to database and load sql drivers
3. Var s -> execute query to insert data into the table “subject”
4. IF subject entered doesn’t exist -> Subject entered to list - > goto:: D.2
5. IF record entered exists -> goto:: D.3

**D.2. JSP page “subsuccess.jsp” Algorithm:**

1. Print “New Subject added to master list successfully”
2. Click Add More-> goto::D
3. Click on Admin Dashboard -> goto::C

**D.3. JSP page “subalreadyexist.jsp” Algorithm:**

1. Print “Wrong input! Subject Already Exist in the list”
2. Click “Try Again” -> goto::D
3. Click on Admin Dashboard -> goto::C

**D.4. JSP page “deleteSubject.jsp” Algorithm:**

1. Var subname->takes parameter from D.1
2. Var con -> connects to database “mysql”
3. Var st->execute delete operation and remains on same page
4. **Teacher addition page “teacher.jsp” Algorithm:**
5. Display Pre-Existing Teacher’s name if any below form with Delete Option using jdbc connectivity
6. IF -> Delete -> goto::E.1
7. Enter New subject
8. Click Submit-> goto:: E.2

**E.1. JSP page “deleteteacher.jsp” Algorithm:**

1. Var tname->takes parameter from I.1
2. Var con -> connects to database “mysql”
3. Var st->execute delete operation and remains on same page

**E.2. Servlet “addTeacher” AddTeacher.java Algorithm:**

1. tname - > takes data from I
2. Con -> connects to database and load sql drivers
3. Var s -> execute query to insert data into the table “teacher”
4. IF teacher name entered doesn’t exist -> Subject entered to list - > goto:: E.3
5. IF record entered exists -> goto:: E.4

**E.3. JSP Page “tsuccess.jsp”:**

1. Print “New Teacher added to master list successfully”
2. Click Add More-> goto::E
3. Click on Admin Dashboard -> goto::C

**E.4. JSP Page “teacheralreadyexist.jsp”:**

1. Print “Teacher is already in the list”
2. Click “Try Again” -> goto::E
3. Click on Admin Dashboard -> goto::C
4. **Class addition page “class.jsp” Algorithm:**

1. Display Pre-Existing Class name if any below form with Delete Option using jdbc connectivity
2. IF -> Delete -> goto::F.1
3. Enter New subject
4. Click Submit-> goto:: F.2

**F.1. JSP page “deleteclass.jsp” Algorithm:**

1. Var cname->takes parameter from F
2. Var con -> connects to database “mysql”
3. Var st->execute delete operation and remains on same page

**F.2. Servlet “addClass” AddClass.java Algorithm:**

1. cname - > takes data from F
2. Con -> connects to database and load sql drivers
3. Var s -> execute query to fetch data into the table “class”
4. IF class name entered doesn’t exist -> Class entered to list - > goto:: F.3
5. IF record entered exists -> goto:: N.4

**F.3. JSP Page “csuccess.jsp”:**

1. Print “New Class added to master list successfully”
2. Click Add More-> goto::N
3. Click on Admin Dashboard -> goto::C

**F.4. JSP Page “classalreadyexist.jsp” Algorithm:**

1. Print “Class already exist”
2. Click Add More-> goto::F
3. Click on Admin Dashboard -> goto::C
4. **ClassSubject.jsp Algorithm**

1. Initialize variable cname, subname
2. Cname ->fetch records from table “class”
3. While cname not empty -> fetch Class Name into scroll down list
4. Subname -> fetch records from table “subject”
5. While suname not empty -> fetch Subject name into scroll list
6. Select name from list > enter submit -> goto::G.1

**G.1. Servlet “assignclassSubject” AssignClassSubject.java Algorithm:**

1. Var cname, subname takes parameter from G
2. Variable s -> fetch records from table “classsubject”
3. IF s is not in the list -> goto::G.2
4. IF s is in the list -> goto:: G.3

**G.2. JSP page “assignclasssuccess.jsp” Algorithm:**

1. Print “New Subject assigned to class”
2. Click Assign More - > goto::G
3. Click Admin Dashboard -> goto::C

**G.3. JSP page “alreadyexist.jsp” Algorithm:**

1. Print “Class with same subject already in the list”
2. Click Try Again- > goto::G
3. Click Admin Dashboard -> goto::C
4. **JSP Page “teacherclassSubject.jsp” Algorithm:**
5. Initialize var cname, subname, tname
6. Var connection -> connect to database “mysql”
7. Cname ->fetch data from “class”
8. While cname is not empty -> get class name into scroll down list
9. Subname - >fetch data from table “subject”
10. While subname is not empty -> get subject name into scroll down list
11. Tname->fetch data from teacher
12. While tname is not empty -> get teacher name into scroll down list
13. Enter Assign->goto::H.1

**H.1. Servlet “assignteacherclassSubject” AssignTeacherClassSubject.java Algorithm:**

1. Var cname, subname, tname ->take parameter from P
2. Var con = connect to database “mysql”
3. Var st = creates connection
4. Var s = execute query and fetch data from tables class, subject and teacher
5. IF records not exist -> Insert new record to table tcs -> goto::H.2
6. IF records exist -> goto::H.3

**H.2. JSP Page “tcssuccess.jsp” Algorithm:**

1. Print “New Teacher assign to the class with new subject”
2. Click on Assign more->goto::H
3. Click on Admin Dashboard -> goto::C

**H.3. JSP page “tcsexist.jsp” Algorithm:**

1. Print “Teacher with same subject and class already exist ”
2. Click Try Again -> goto::H
3. Click on Admin Dashboard -> goto::C
4. **Student addition page “student.jsp” Algorithm:**
5. Enter Student name -> stored in “studname”
6. Fetch class dynamically from database table “class” into scroll down list
7. Initialize Var cname of type ResultSet
8. Var connection -> connect to database
9. Cname -> execute query and fetch data dynamically from database table “class” into scroll down list
10. Enter Submit -> goto::I.1

**I.1. Servlet “addStudent” AddStudent.java Algorithm:**

1. Var studname, cname - > take parameter from I
2. Var con -> connects to database “mysql”
3. Var s->execute query to fetch records from table student
4. IF s->records not available -> insert records->goto::I.2
5. IF s->records exist->goto::I.3

**I.2. JSP page “studsuccess.jsp”Algorithm:**

1. Print “New Student added successfully”
2. Click Add More ->goto::I
3. Click Admin Dashboard ->goto::C

**I.3. JSP page “studentexist.jsp” Algorithm:**

1. Print “Wrong input! Student name already exist in the class”
2. Click Try Again -> goto:: I
3. Click Admin Dashboard -> goto::C
4. **JSP page “chooseclass.jsp” Algorithm:**
5. Initialize var cname type ResultSet
6. Var connection -> connects database “mysql”
7. Cname->execute query and fetch data from table class
8. While cname->get value from table
9. className->cname
10. Enter Get Student List->goto::J.1
11. Click Admin Dashboard ->goto:: C

**J.1. Servlet “masterstudentlist” MasterStudentList.java Algorithm:**

1. Var -> takes parameter “className” from J
2. Var con -> connects to database “mysql”
3. Var rs -> fetch records needed from table student
4. While exist :: Var n ->studid, Var nm ->Student (from table)
5. Print Records
6. Click Go Back -> goto::J
7. **JSP Page “classreport.jsp” Algorithm:**
8. Initialize var cname of type ResultSet
9. Var connection -> connect to database “mysql”
10. Cname ->fetch data from class dynamically
11. className->cname
12. Click generate report->goto::K.1
13. Click Admin Dashboard ->goto:: C

**K.1. Servlet “masterreport” MasterReport.java Algorithm:**

1. Var c->className from K
2. Var con -> connects to database “mysql”
3. Var “stmt, st,ts”Type Statement -> to create connection
4. Var “rs, sub,t” type ResultSet->fetch data from table – student, classsubject, tcs for the class selected in K
5. While rs, sub,ts -> exist -> print report
6. Click Go Back -> goto::K
7. **JSP Page “logout.jsp” Algorithm:**
8. Goto::A

**FLOWCHART OF APPLICATION**

**USE CASE DIAGRAM**

**Flow Chart 1:**

**IMPLEMENTATION**

1. **Software and Tools used: This is Angular Web Project.**
2. Operating System - Windows 7 or upper version
3. Visual Studio Code- To write JavaScript or Typescript code JSON code for application
4. Trello – Online web app for sprint planning and make user cards.
5. Bootstrap and CSS – for designing the Quiz page
6. GitHub – Online repository to track and submit the project
7. MS-Word – To create user stories and flow chart
8. **Methodology: Agile-Scrum**
9. **Coding: Just provide the JSON to the quiz application and it will render the quiz**
10. **Code for angular quiz- “angular.json”**

{

    "id": 1,

    "name": "Angular Online Quiz",

    "description": "Angular Quiz (based on basic concepts)",

    "questions": [

        {

            "id": 1010,

            "name": "Which of the following is not a core AngularJS directive.",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "ng-app",

                    "isAnswer": false

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": "ng-model",

                    "isAnswer": false

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "ng-bind",

                    "isAnswer": false

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "ng-state",

                    "isAnswer": true

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        },

        {

            "id": 1011,

            "name": "Which of the following is true about lowercase filter?",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "Lowercase filter converts a text to lower case text.",

                    "isAnswer": true

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "Lowercase filter is a function which takes text as input.",

                    "isAnswer": false

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": "Both of the above.",

                    "isAnswer": false

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "None of the above.",

                    "isAnswer": false

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        },

        {

            "id": 1012,

            "name": " Who is sometimes called as father of Angularjs?",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "Brad Green",

                    "isAnswer": false

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "Misko Hevery",

                    "isAnswer": true

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": "Brad Richardson",

                    "isAnswer": false

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "Chuck Jazdzewski",

                    "isAnswer": false

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        },

        {

            "id": 1013,

            "name": "Which of the following is correct about TypeScript?",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "Angular is based on TypeScript.",

                    "isAnswer": false

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "This is a superset of JavaScript.",

                    "isAnswer": false

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": "TypeScript is maintained by Microsoft.",

                    "isAnswer": false

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "All of the above.",

                    "isAnswer": true

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        },

        {

            "id": 1014,

            "name": "What is the decorator used for configuring your module class?",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "@NgModule",

                    "isAnswer": true

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "@NgApp",

                    "isAnswer": false

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": "Both",

                    "isAnswer": false

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "None of above",

                    "isAnswer": false

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        },

        {

            "id": 1015,

            "name": "Which angular decorator allows us to define the pipe name that is globally available for use in any template in the across application?",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "pipeName",

                    "isAnswer": false

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "pipeDeco",

                    "isAnswer": false

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": "Pipe",

                    "isAnswer": true

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "None of the above",

                    "isAnswer": false

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        },

        {

            "id": 1016,

            "name": "Router is part of which of the following module?",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "@angular/core",

                    "isAnswer": false

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": "@angular/router",

                    "isAnswer": true

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "Both",

                    "isAnswer": false

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "None of the above",

                    "isAnswer": false

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        },

        {

            "id": 1017,

            "name": "Which character is used for chaining multiple pipes?",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "|",

                    "isAnswer": true

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": ":",

                    "isAnswer": false

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "Both",

                    "isAnswer": false

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "None of the above",

                    "isAnswer": false

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        },

        {

            "id": 1018,

            "name": "What does AOT stand for?",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "ahead-of-time compilation",

                    "isAnswer": true

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "Angular Object Templates",

                    "isAnswer": false

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": "Angular Object Templates",

                    "isAnswer": false

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "None of the above",

                    "isAnswer": false

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        },

        {

            "id": 1019,

            "name": "Which of the following is correct about package.json?",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "This file is used to give the options about TypeScript used for the Angular JS project.",

                    "isAnswer": false

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": "This file contains information about Angular project.",

                    "isAnswer": true

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "This file contains the system files required for Angular JS application.",

                    "isAnswer": false

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "All of the above.",

                    "isAnswer": false

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        }

    ]

}

1. **Code for javascript quiz- “javascript.json”**

{

    "id": 1,

    "name": "JSON Online Quiz",

    "description": "JSON Quiz (Basic Multiple Choice Questions for Developers)",

    "questions": [

        {

            "id": 1010,

            "name": "JSON Stands for \_\_\_\_\_\_\_\_?",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "Java Standard Output Network",

                    "isAnswer": false

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": "JavaScript Object Notation",

                    "isAnswer": true

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "JavaScript Output Name",

                    "isAnswer": false

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "Java Source Open Network",

                    "isAnswer": false

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        },

        {

            "id": 1011,

            "name": "Which of the following is not a type in JSON ?",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "date",

                    "isAnswer": true

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "object",

                    "isAnswer": false

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": "Array",

                    "isAnswer": false

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "string",

                    "isAnswer": false

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        },

        {

            "id": 1012,

            "name": "Who is the Father of JSON?",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "Douglas Crockford",

                    "isAnswer": true

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "Rasmus Lerdorf",

                    "isAnswer": false

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": "Douglas Michel",

                    "isAnswer": false

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "Dennis Ritchie",

                    "isAnswer": false

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        },

        {

            "id": 1013,

            "name": "What extension is used to save a JSON file?",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": ".json",

                    "isAnswer": true

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": ".js",

                    "isAnswer": false

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": ".javaN",

                    "isAnswer": false

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": ".on",

                    "isAnswer": false

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        },

        {

            "id": 1014,

            "name": "JSON string have used in \_\_\_\_\_\_\_\_\_\_\_\_",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "Duoble Quote",

                    "isAnswer": true

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "Single Quote",

                    "isAnswer": false

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": "Both 1 & 2",

                    "isAnswer": false

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "None of the above",

                    "isAnswer": false

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        },

        {

            "id": 1015,

            "name": "JSON elements are separated by the \_\_\_\_\_\_",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "line break",

                    "isAnswer": false

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "Semi-Colon",

                    "isAnswer": false

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": "comma",

                    "isAnswer": true

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "white-space",

                    "isAnswer": false

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        },

        {

            "id": 1016,

            "name": "The MIME type of JSON is \_\_\_\_\_",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "application/json",

                    "isAnswer": true

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": "application/x-json",

                    "isAnswer": false

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "application/javascript",

                    "isAnswer": false

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "text/json",

                    "isAnswer": false

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        },

        {

            "id": 1017,

            "name": "What is the common usage of JSON on modern websites?",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "To store information remotely",

                    "isAnswer": false

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": "To send and receive bits of data",

                    "isAnswer": true

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "To store information locally",

                    "isAnswer": false

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "All of the Above",

                    "isAnswer": false

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        },

        {

            "id": 1018,

            "name": "Which of the following code will throw an error?",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "JSON.parse(‘{}’);",

                    "isAnswer": false

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "JSON.parse(null);",

                    "isAnswer": false

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": "JSON.parse(undefined);",

                    "isAnswer": true

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "JSON.parse(‘[]’);",

                    "isAnswer": false

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        },

        {

            "id": 1019,

            "name": "\_\_\_\_\_\_\_ is not a valid way to parse JSON string",

            "questionTypeId": 1,

            "options": [

                {

                    "id": 1055,

                    "questionId": 1010,

                    "name": "JSON.parse()",

                    "isAnswer": false

                },

                {

                    "id": 1056,

                    "questionId": 1010,

                    "name": "JSON.eval()",

                    "isAnswer": true

                },

                {

                    "id": 1057,

                    "questionId": 1010,

                    "name": "jQuery.parseJSON()",

                    "isAnswer": false

                },

                {

                    "id": 1058,

                    "questionId": 1010,

                    "name": "None of the above",

                    "isAnswer": false

                }

            ],

            "questionType": {

                "id": 1,

                "name": "Multiple Choice",

                "isActive": true

            }

        }

    ]

}

1. **“quiz.component.ts”**

import { Component, OnInit } from '@angular/core';

import { QuizService } from '../services/quiz.service';

import { HelperService } from '../services/helper.service';

import { Option, Question, Quiz, QuizConfig } from '../models/index';

@Component({

  selector: 'app-quiz',

  templateUrl: './quiz.component.html',

  styleUrls: ['./quiz.component.css'],

  providers: [QuizService]

})

export class QuizComponent implements OnInit {

  quizes: any[];

  quiz: Quiz = new Quiz(null);

  mode = 'quiz';

  quizName: string;

  config: QuizConfig = {

    'allowBack': true,

    'allowReview': true,

    'autoMove': false,  // if true, it will move to next question automatically when answered.

    'duration': 600,  // indicates the time (in secs) in which quiz needs to be completed. 0 means unlimited.

    'pageSize': 1,

    'requiredAll': false,  // indicates if you must answer all the questions before submitting.

    'richText': false,

    'shuffleQuestions': false,

    'shuffleOptions': false,

    'showClock': false,

    'showPager': true,

    'theme': 'none'

  };

  pager = {

    index: 0,

    size: 1,

    count: 1

  };

  timer: any = null;

  startTime: Date;

  endTime: Date;

  ellapsedTime = '00:00';

  duration = '';

  constructor(private quizService: QuizService) { }

  ngOnInit() {

    this.quizes = this.quizService.getAll();

    this.quizName = this.quizes[0].id;

    this.loadQuiz(this.quizName);

  }

  loadQuiz(quizName: string) {

    this.quizService.get(quizName).subscribe(res => {

      this.quiz = new Quiz(res);

      this.pager.count = this.quiz.questions.length;

      this.startTime = new Date();

      this.ellapsedTime = '00:00';

      this.timer = setInterval(() => { this.tick(); }, 1000);

      this.duration = this.parseTime(this.config.duration);

    });

    this.mode = 'quiz';

  }

  tick() {

    const now = new Date();

    const diff = (now.getTime() - this.startTime.getTime()) / 1000;

    if (diff >= this.config.duration) {

      this.onSubmit();

    }

    this.ellapsedTime = this.parseTime(diff);

  }

  parseTime(totalSeconds: number) {

    let mins: string | number = Math.floor(totalSeconds / 60);

    let secs: string | number = Math.round(totalSeconds % 60);

    mins = (mins < 10 ? '0' : '') + mins;

    secs = (secs < 10 ? '0' : '') + secs;

    return `${mins}:${secs}`;

  }

  get filteredQuestions() {

    return (this.quiz.questions) ?

      this.quiz.questions.slice(this.pager.index, this.pager.index + this.pager.size) : [];

  }

  onSelect(question: Question, option: Option) {

    if (question.questionTypeId === 1) {

      question.options.forEach((x) => { if (x.id !== option.id) x.selected = false; });

    }

    if (this.config.autoMove) {

      this.goTo(this.pager.index + 1);

    }

  }

  goTo(index: number) {

    if (index >= 0 && index < this.pager.count) {

      this.pager.index = index;

      this.mode = 'quiz';

    }

  }

  isAnswered(question: Question) {

    return question.options.find(x => x.selected) ? 'Answered' : 'Not Answered';

  };

  isCorrect(question: Question) {

    return question.options.every(x => x.selected === x.isAnswer) ? 'correct' : 'wrong';

  };

  onSubmit() {

    let answers = [];

    this.quiz.questions.forEach(x => answers.push({ 'quizId': this.quiz.id, 'questionId': x.id, 'answered': x.answered }));

    // Post your data to the server here. answers contains the questionId and the users' answer.

    console.log(this.quiz.questions);

    this.mode = 'result';

  }

}

1. **To retrieve quiz json file - “quiz.service.ts”**

import { HttpClient } from '@angular/common/http';

import { Injectable } from '@angular/core';

@Injectable()

export class QuizService {

  constructor(private http: HttpClient) { }

  get(url: string) {

    return this.http.get(url);

  }

  getAll() {

    return [

      { id: 'data/angular.json', name: 'Angular' },

      { id: 'data/javascript.json', name: 'JavaScript' },

    ];

  }

}

1. **Quiz View (HTML Template) - “quiz.component.html”**

  <div class="row">

    <div class="col-6">

      <h2>Online Test Application</h2>

    </div>

    <div class="col-6 text-right">

      Select Quiz Type:

      <select [(ngModel)]="quizName" (change)="loadQuiz(quizName)">

        <option \*ngFor="let quiz of quizes" [value]="quiz.id">{{quiz.name}}</option>

      </select>

    </div>

  </div>

  <div id="quiz">

    <h2 class="text-center font-weight-normal">{{quiz.name}}</h2>

    <hr />

    <div \*ngIf="mode=='quiz' && quiz">

      <div \*ngFor="let question of filteredQuestions;">

        <div class="badge badge-info">Question {{pager.index + 1}} / {{pager.count}}.</div>

        <div \*ngIf="config.duration" class="badge badge-info float-right">Time: {{ellapsedTime}} / {{duration}}</div>

        <h3 class="font-weight-normal">{{pager.index + 1}}.

          <span [innerHTML]="question.name"></span>

        </h3>

        <div class="row text-left options">

          <div class="col-6" \*ngFor="let option of question.options">

            <div class="option">

              <label class="font-weight-normal" [attr.for]="option.id">

                <input id="{{option.id}}" type="checkbox" [(ngModel)]="option.selected" (change)="onSelect(question, option);" /> {{option.name}}

              </label>

            </div>

          </div>

        </div>

      </div>

      <hr />

      <div class="text-center">

        <button class="btn btn-default" \*ngIf="config.allowBack" (click)="goTo(0);">First</button>

        <button class="btn btn-default" \*ngIf="config.allowBack" (click)="goTo(pager.index - 1);">Prev</button>

        <button class="btn btn-primary" (click)="goTo(pager.index + 1);">Next</button>

        <button class="btn btn-default" \*ngIf="config.allowBack" (click)="goTo(pager.count - 1);">Last</button>

        <!-- <pagination \*ngIf="config.showPager" direction-links="false" total-items="totalItems" items-per-page="itemsPerPage" ng-model="currentPage" ng-change="pageChanged()"></pagination> -->

      </div>

    </div>

    <div class="row text-center" \*ngIf="mode=='review'">

      <div class="col-4 cursor-pointer" \*ngFor="let question of quiz.questions; let index = index;">

        <div (click)="goTo(index)" class="p-3 mb-2 {{ isAnswered(question) == 'Answered'? 'bg-info': 'bg-warning' }}">{{index + 1}}. {{ isAnswered(question) }}</div>

      </div>

    </div>

    <div class="result" \*ngIf="mode=='result'">

      <h2>Quiz Result</h2>

      <div \*ngFor="let question of quiz.questions; let index = index">

        <div class="result-question">

          <h5>{{index + 1}}. {{question.name}}</h5>

          <div class="row">

            <div class="col-6" \*ngFor="let Option of question.options">

              <input id="{{Option.id}}" type="checkbox" disabled="disabled" [(ngModel)]="Option.selected" /> {{Option.name}}

            </div>

          </div>

          <div class="p-1 m-2 alert {{ isCorrect(question) == 'correct'? 'alert-success': 'alert-danger'}}">Your answer is {{isCorrect(question)}}.</div>

        </div>

      </div>

      <h4 class="alert alert-info text-center">You may close this window now.</h4>

    </div>

    <hr />

    <div class = "text-center" \*ngIf="mode!='result'">

      <button class="btn btn-warning" (click)="mode = 'quiz'">Quiz</button>

      <button class="btn btn-info" (click)="mode = 'review'">Review</button>

      <button class="btn btn-primary" (click)="onSubmit();">Submit Quiz</button>

    </div>

  </div>

1. **For styling - “style.css”**

.btn{

    border-radius: 0;

}

.option{

    background-color:#cadaee;

    font-size:20px;

    padding:10px;

    margin:3px;

}

    #quiz .options input[type=checkbox] {

        height:22px;

        width:22px;

    }

.cursor-pointer{

    cursor: pointer;

}

.result-question{

    background-color:#eee;

    margin:4px;

    padding:6px;

}

@media only screen and (max-width: 480px) {

    h1,.h1{

        font-size:22px;

    }

    h2,.h2{

        font-size:20px;

    }

    h3,.h3{

        font-size:18px;

    }

    .option {

        font-size: 16px;

        padding: 6px;

    }

}

1. **“package.json”**

{

  "name": "quiz-app",

  "version": "0.0.3",

  "author": "Manish Kumar Arora",

  "scripts": {

    "ng": "ng",

    "start": "ng serve",

    "build": "ng build",

    "test": "ng test",

    "lint": "ng lint",

    "e2e": "ng e2e"

  },

  "private": true,

  "dependencies": {

    "@angular/animations": "^8.0.2",

    "@angular/common": "^8.0.2",

    "@angular/compiler": "^8.0.2",

    "@angular/core": "^8.0.2",

    "@angular/forms": "^8.0.2",

    "@angular/platform-browser": "^8.0.2",

    "@angular/platform-browser-dynamic": "^8.0.2",

    "@angular/router": "^8.0.2",

    "bootstrap": "^4.1.1",

    "core-js": "^3.1.4",

    "jquery": "^3.3.1",

    "rxjs": "^6.0.0",

    "zone.js": "^0.9.1"

  },

  "devDependencies": {

    "@angular-devkit/build-angular": "~0.800.3",

    "@angular/cli": "~8.0.3",

    "@angular/compiler-cli": "^8.0.2",

    "@angular/language-service": "^8.0.2",

    "@types/jasmine": "~3.3.13",

    "@types/jasminewd2": "~2.0.3",

    "@types/node": "^12.0.10",

    "codelyzer": "~5.1.0",

    "jasmine-core": "~3.4.0",

    "jasmine-spec-reporter": "~4.2.1",

    "karma": "~4.1.0",

    "karma-chrome-launcher": "~2.2.0",

    "karma-coverage-istanbul-reporter": "~2.0.5",

    "karma-jasmine": "~2.0.1",

    "karma-jasmine-html-reporter": "^1.4.2",

    "protractor": "~5.4.2",

    "ts-node": "~8.3.0",

    "tslint": "~5.17.0",

    "typescript": "3.4.5"

  }

}

**Models files:**

1. **“index.ts”**

export \* from './option';

export \* from './question';

export \* from './quiz';

export \* from './quiz-config';

1. **“option.ts”**

export class Option {

    id: number;

    questionId: number;

    name: string;

    isAnswer: boolean;

    selected: boolean;

    constructor(data: any) {

        data = data || {};

        this.id = data.id;

        this.questionId = data.questionId;

        this.name = data.name;

        this.isAnswer = data.isAnswer;

    }

}

1. **“question.ts”**

import { Option } from './option';

export class Question {

    id: number;

    name: string;

    questionTypeId: number;

    options: Option[];

    answered: boolean;

    constructor(data: any) {

        data = data || {};

        this.id = data.id;

        this.name = data.name;

        this.questionTypeId = data.questionTypeId;

        this.options = [];

        data.options.forEach(o => {

            this.options.push(new Option(o));

        });

    }

}

1. **“quiz-config.ts”**

export class QuizConfig {

    allowBack: boolean;

    allowReview: boolean;

    autoMove: boolean;  // if boolean; it will move to next question automatically when answered.

    duration: number;  // indicates the time in which quiz needs to be completed. 0 means unlimited.

    pageSize: number;

    requiredAll: boolean;  // indicates if you must answer all the questions before submitting.

    richText: boolean;

    shuffleQuestions: boolean;

    shuffleOptions: boolean;

    showClock: boolean;

    showPager: boolean;

    theme: string;

    constructor(data: any) {

        data = data || {};

        this.allowBack = data.allowBack;

        this.allowReview = data.allowReview;

        this.autoMove = data.autoMove;

        this.duration = data.duration;

        this.pageSize = data.pageSize;

        this.requiredAll = data.requiredAll;

        this.richText = data.richText;

        this.shuffleQuestions = data.shuffleQuestions;

        this.shuffleOptions = data.shuffleOptions;

        this.showClock = data.showClock;

        this.showPager = data.showPager;

    }

}

1. **“quiz.ts”**

import { QuizConfig } from './quiz-config';

import { Question } from './question';

export class Quiz {

    id: number;

    name: string;

    description: string;

    config: QuizConfig;

    questions: Question[];

    constructor(data: any) {

        if (data) {

            this.id = data.id;

            this.name = data.name;

            this.description = data.description;

            this.config = new QuizConfig(data.config);

            this.questions = [];

            data.questions.forEach(q => {

                this.questions.push(new Question(q));

            });

        }

    }

}

**OUTPUT TEST – OUTPUT SCREENSHOTS**

Please refer “screenshot file” added in git hub.

**CORE CONCEPT USED IN THE PROJECT**

This project Angular in just few lines of codes. Brief of all concepts used in the project are as follow:

1. Angular, bootstrap 4 and a bit of CSS.
2. The quiz application accepts the questions in JSON format. So, you can easily send the JSON from the server in the pre-defined format and the Angular quiz application will render the quiz at the client side.
3. Quiz application consists of mainly 3 components/Views: Quiz View, Review, and Result.
4. Apart from this, “quiz.component.ts” has been used as a component class for all the views and styles.css has been used to apply CSS style to the application.
5. The application has a small set of scripting part that has been handled by the controller: “quiz.component.ts”.
6. The component loads the questions by using loadQuiz('data/angular.json') method called at ngOnInit event.
7. The quiz questions should be provided in a pre-defined JSON format as mentioned in “angular.json” or other related JSON files present in Data folder.
8. Use of html CSS coding to make it more user interacting web interface.
9. Used only one service in Angular component: quiz.service. It is used to retrieve quiz json file, populate the list of quizzes, and check the result.
10. The Quiz View contains the UI for quiz questions, previous-next button, paging and related stuffs.
11. For buttons and pagination UI, bootstrap has been used.
12. The UI for questions and options has been set in styles.css.
13. To display the questions and options, \*ngFor has been used.
14. For styling and theme, bootstrap 4 has been used.
15. For further styling based on the applications need, CSS classes has been added in styles.css.

**LINK TO THE GITHUB REPOSITORY TO TRACK AND VERIFY THE PROJECT COMPLETION**

[**https://github.com/Manish-K-Arora/Phase4ProjectOnlinetestapplication**](https://github.com/Manish-K-Arora/Phase4ProjectOnlinetestapplication)

**CONCLUSION**

1. **Summary**

* The project has been developed mainly for users (client side) of “online test application”.
* The quiz configuration is an optional object that can be present in your <quiz>.json file. The config section allows you to customize your quiz the way you wish to do. The quiz application reads these configuration settings and applies these settings at the time of loading the quiz. The settings mainly consists of: shuffling the questions, showing/hiding pager, allowing back navigation, allowing auto move to next question.
* This quiz configuration looks like the following:
* config: QuizConfig = {
* 'allowBack': true,
* 'allowReview': true,
* 'autoMove': false,  // if true, it will move to next question automatically when answered.
* 'duration': 600,  // indicates the time (in secs) in which quiz needs to be completed. 0 means unlimited.
* 'pageSize': 1,
* 'requiredAll': false,  // indicates if you must answer all the questions before submitting.
* 'richText': false,
* 'shuffleQuestions': false,
* 'shuffleOptions': false,
* 'showClock': false,
* 'showPager': true,
* 'theme': 'none'
* };
* **Quiz Features include:**

1. **shuffleQuestions** property in config section is used to mention whether the questions should be randomized before showing or not. If true, the questions will be randomized. The default value is false.
2. **showPager** property in config section indicates whether to show the pager element in quiz or not. If false, the numeric pager will not be shown. In this case, the user can still navigate via. first, prev, next and last button. The default value is true.
3. **autoMove** property in config section indicates whether to move to next question automatically when the question is answered. If true, you don't need to press Next button to move to next question. The default value is false.
4. In mock test/practice exams, user might want to change the quiz dynamically. To do so, you just need to call **loadQuiz** method and pass the URL of the new quiz. The attached sample or the demo link illustrates this. (In the sample app, you can load the quiz dynamically via changing the drop-down at top right corner.)

* User friendly dynamic web based options are provided to user with variable options for selecting different Quiz from selector.
* The usage of software increases the efficiency, decreases the effort.
* It has been thoroughly tested and implemented.

1. **Future Enhancements:**

Some of the future enhancements that can be done to this system are:

* **Initial Login:** We can also provide initial registration page for users so that they can register their name and start the quiz and then the quiz will record the performance of that particular user and show the result.
* **Correct Answers Counter:** We can count the number of correct answers and display at the end after clicking the submit button and the result will be save and displayed to that user with name.
* **Question Type:** Currently, the application supports only multiple choice questions. In many scenarios, it is needed to have multiple answer type questions, true-false, etc. We can think of adding that feature.
* **Pick From Pool:** In some scenarios, you might want to provide a pool of many questions as a JSON and want the Quiz application to randomly select n questions for the quiz.
* **Optional Animation:** This is not an important feature but some people may want a bit of animation in his quiz application. I would, therefore, like to provide this feature as well.
* **Tailored Questions:** There are many cases when we require tailored questions in a survey. This means that the next set of questions should come based on the option selected in current question of the survey.

1. **Unique Selling Points(USPs)**

This is Angular technology based project with dynamic web pages with user-interactive options which are easy to use, where user can select the particular quiz and do the quiz with multiple choice answers which enhances the interaction of user with application and given timer to complete the quiz within the time limits.