ABSTRACT

The main objective of the project titled "**Assessment System**" was to enable organization to conduct quizzes and organize assessments to gauge the understandings and learnings of employees through workshops, training sessions and seminars held at the organization in just few simple steps and thereby making assessment and evaluation a hassle free affair.

Prior to this application, if the organization wanting to conduct assessments / quizzes, administer constant evaluation or gauge the understanding and learning of their employees/interns, has to either use a separate third-party application i.e. Google forms, Planet Ganges, etc. or a manual pen-paper examination and then get the responses / answers manually checked and verified. Also, the calculation of report card and performance analysis of the Employee in the assessment conducted was generated manually. The generated reports are sent to each Employee, their Project Mentor, Project Manager and Scrum Master through Email in order to do the needful which will improve their performance. If an employee wanting to gauge their understandings from the courses, trainings and workshops, has to use a separate third-party application.

This was a long and tedious process, which was simplified by the custom application titled Assessment System. This application provides a unified Intra-organization platform, which enables the organization in assessing and gauging the learnings and understanding of their employees. This internship report contains chapters which explains my five months of experience in the Company. It contains detailed description of the project and about the technology stack used for the development. Features such as create, view, update and delete Questions, Quizzes, Tags, scheduling Assessment, taking Assessment, viewing Detailed Reports and many other functionalities are triggered by this application.

ASP.NET was used for building the APIs at the Backend, which is a framework for building Web Applications, Services and Web APIs with .NET and C# and then subsequently, integrating it with UI at the Frontend, which was built using Angular 7, a framework for mobile and web application development with TS, HTML, CSS and Bootstrap. Moreover, SQL Server was used as the database for storing all the data.

TABLE OF CONTENTS

Title **Page No.**

LIST OF FIGURES

LIST OF PHOTOGRAPHS

LIST OF TABLES

CHAPTER 1 : INTRODUCTION

1.1 INTRODUCTION TO THE PROJECT :

This **Assessment System** provides a unified Intra-organization platform, which enables the organization to conduct quizzes, administer the continuous evaluation and organize assessments to gauge the understandings and learnings of employees / interns through workshops, training sessions and seminars. Also, the detailed reports help in monitoring and keep a track of knowledge of the employees in the organization.

The Assessment System has four personas which consists of :-

- **Content-Creator** who can create, view, update and delete questions, tags/subjects, assessments and quizzes.
- **Test-Administrator** who can schedule quizzes, assign employees to a schedule to take a quiz/assessment and remove as well. Can invite employees via email to take the quiz.
- **Employee** is able to take scheduled quizzes and gauge the understandings and learnings of oneself by opting for mock quizzes. Can view their detailed report and performance analysis.
- **Reporting-User** keeps track of the Results and Performance Analysis by Employee, Quiz and Subject/Tags.

The application is built considering all the security parameters, the session management is achieved with Google Access Token which is generated by Google OAuth API that provides Google SSO.

Visual Studio 2017 was used for building the APIs at the Backend with .NET framework and C# and then subsequently, integrating it with UI at the Frontend, which was built on Visual Studio Code using Angular 7, with TS, HTML and CSS alongside Bootstrap. Moreover, Microsoft SQL Server Management Studio 2017 which provides User Interface for SQL Server, which was used as the database for storing all the data.

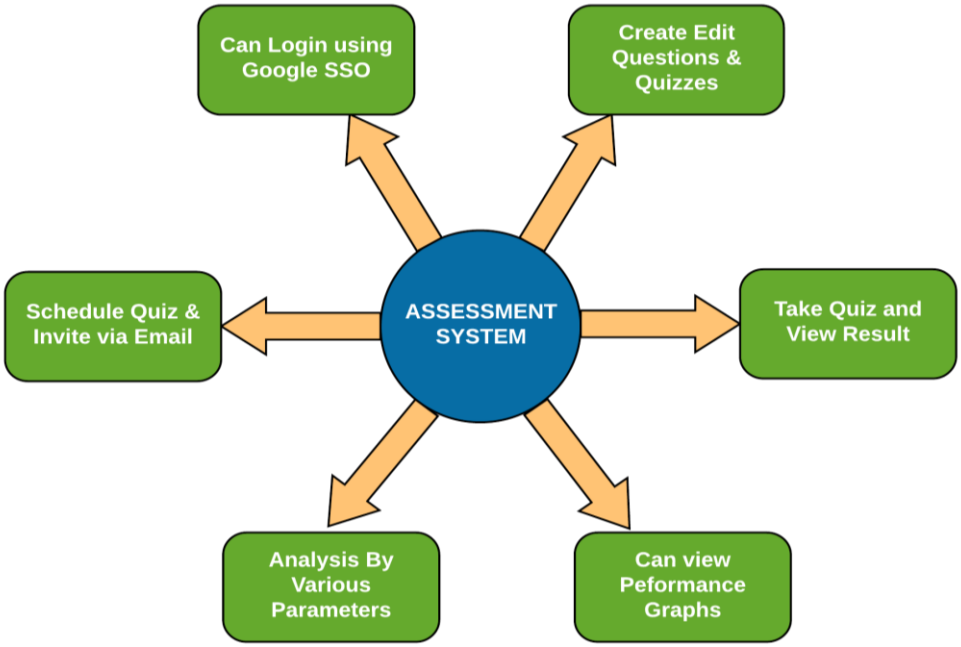

Fig 1.1: Overview of the Project

The project is made to in such a way so that it fulfills the basic requirements of Organization. It provides the compatibility with any platform or device as it runs on Web Browser (preferably Google Chrome).

Some APIs used are :-

- Google API : To enable the user to Login/Register using Google SSO OAuth.
- Gmail SMTP Client API : To send an email notifying the employee about the Scheduled Quiz.
- Image Resize API : To handle Image Size on Server.

1.2 OBJECTIVE :

The main objective of Assessment System is that to enable organization to conduct quizzes and organize assessments to gauge the understandings and learnings of employees through workshops, training sessions and seminars held at the organization.

In just few simple steps one can trigger a lot of functionalities present in the application which are as follows :-

- One Click Login/Sign-Up using Google SSO.

- A user can have multiple Roles else Employee(by default)
- Dashboard will appear as per the assigned Role(s) with different functionalities.
  - ❖ Content Creator
  - ❖ Test-Administrator
  - ❖ Employee
  - ❖ Reporting-User
- Role dependent dashboard have functionalities on the Side Navigation Menu as :
  - o Create, view, update, delete or archive Questions, Quizzes and Tags
  - o Schedule Assessment/Quiz
  - o Assign employees to Scheduled Assessment, Invite via Email
  - o Taking up the Assessment
  - o View the Detailed Report with different types of Performance Graphs

1.3 MOTIVATION :

The motivation to make this project came from the fact that a comprehensive tool was required to provide the organization a better solution for monitoring the employees' performance and knowledge. Prior to this Application, organization followed a series of complex procedural steps in order to conduct an Intra-organization Assessment of Employees, which were done using separate third-party application like Google Forms where the response are verified manually or some other organization services such as Planet Ganges, and so forth. Likewise, these application do not furnish or provide performance analysis of their employees with which the organization can gauge the learnings and understandings gained from the workshops and training session held.

1.4 DEFINITION AND OVERVIEW :

This document presents the structure of the system, such as the database architecture, application architecture (layers), application flow (navigation), innovation and technology architecture. The TDD uses non-technical to mildly-technical terms which should be understandable to the administrators of the system. The intended audience is the Content-creator, Test-Administrator, Employee, Reporting-User which can be from distinctive department such as Technology, Development, Human Resource, Finance, Marketing, etc.

1.5 OVERVIEW OF THE ORGANIZATION :

NINELEAPS TECHNOLOGY SOLUTIONS is a start-up established as an innovation company located in the Bangalore on 06 January 2014. Since its establishment it has been providing high quality services for software development. As an emerging company aiming high, provides rapid, effective and efficient solutions to client business needs while having a perfect insight of stability and consistency which reduces the maintenance activities.

Nineleaps Technology Solutions enabling the 2014 start-up to feature at **10th position** in the **LinkedIn's Top 10 Startups (India) for 2018**, alongside OYO Rooms and cure.fit.

Nineleaps work in the areas of mobile, web, analytics and Big Data promoting experimentation with new technology and techniques. They love open source platforms as they breed innovation and ingenuity. Open source tools and platforms are always the priority for services and development. The expert teams of design, development, project management, quality assurance, sales and marketing work in a close-knit network to help clients. They apply the best industry-proven practices, leading standards and methodologies in the entire life-cycle of each client desired custom software and web development project.

Fig 1.2: Work Domains of Nineleaps

Nineleaps boasts of having clients among the Fortune 500 companies, helping them to re-invent business to stay ahead in the changing industry dynamics. They have renowned clients in multitude of domains, e.g. Uber, Tesco, Ford, MemberSuite etc.

Notable Technological Information :-

- **Engineering Practices**
  - ➢ The iterative and incremental nature of **AGILE development** used by Nineleaps accelerates the speed to market.
- **Experience Design**
  - ➢ In-house design team simplifies the complex applications bringing best user experience to life.
- **System Architecture**
  - ➢ The solid foundation ensures robust systems, high performance, scalability and channel agnostic products.

Nineleaps Major Achievements as a Startup

The professional network, which counts our country India among its top markets, revealed the list "is fuelled by proprietary LinkedIn data including employment growth, engagement, job interest, and attraction of top talent combined with an editorial lens".

Major Principles on which Nineleaps works :-

IMPACT

- I – INCLUSION :- The action or state of including or of being included within a group or structure.
- M – METTLE :- A person's ability to cope well with difficulties, spirit and resilience.
- P – PIONEER :- A person who is among the first to research and develop a new area of knowledge or activity.
- A – ACCOUNTABLE :- Able to be explained or understood.
- C – COLLABORATION :- A state in which two bodies can work together, can co-exist together without any problems and conflicts.
- T – TRUST :- Firm faith and belief in our teammates.

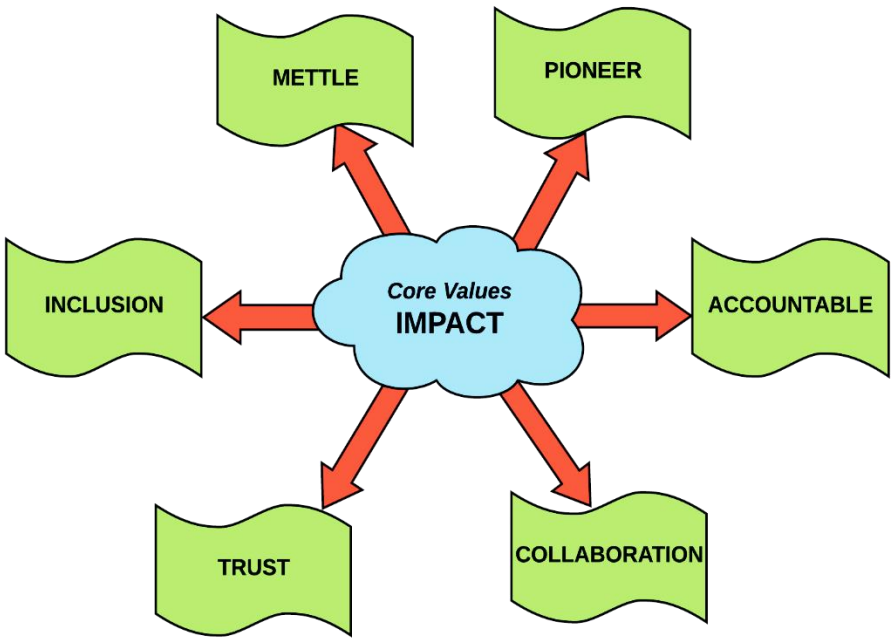

Fig 1.3: Core values of Nineleaps

CHAPTER 2 : OVERALL DESCRIPTION

2.1 EXISTING SYSTEM :

As talking about the existing system there was no application or tool for monitoring and generating the analysis report of the employees' performance which in turn will help in gauging the understandings and learnings thereby reflecting their knowledge in the allocated domain. For administering the assessment, the process followed by the company involves a series of long and tedious procedure, where the responses of each individual are evaluated manually or even if it did, the report could not be used for measuring individuals' performance.

The procedure involved was time consuming. There was an urgent need for a more efficient and automated process to make things faster and easier.

2.2 DRAWBACKS OF THE EXISTING SYSTEM :

Concerning the existing system, Google form or pen-paper strategy was used to administer the assessments or any third-party paid application was used. All the above methods had a lot of disadvantages.

Some of which can be as follows :-

- Time consuming process
- Incorporates a lot of manual efforts
- Vulnerable to discrepancy while evaluating the answers
- Recording and handling the evaluated data was tedious
- Performing analysis on the results of every individual was a huge challenge
- Wastage of assets and resources happens mostly

2.3 PROPOSED SYSTEM :

- The proposed system is platform-independent and can used on any device. Since the aim was to make the Assessment a hassle free affair, it was developed as a Web Application, it can be used from any sort of device, be it a Desktop, Mobile, Laptop

or a Tablet. All it needs, is any device with a Web Browser (preferably Google Chrome) with an Internet Connection. Thereby, making it simple, easy to use and portable.

- The proposed system allows domain login through Google SSO which is configured for authorized employees with a valid Nineleaps Email Id and access their information.
- This system is developed with role-based authentication which redirects to the dashboard accordingly.
- The application was divided and developed into 3 independent layers : -
  Frontend, Backend and the Database Layer.
- The Backend is built using ASP.NET WebAPI with C# and .NET framework 4.6 which is REST compliant which provides a lot of benefits as follows :-
  - RESTful APIs are independent of the frontend and is only concerned with the data.
  - RESTful APIs can be executed on tools like POSTMAN and SWAGGER.
  - It allows variety of data formats, like XML and JSON.
  - Provides better support with Browser Clients (Frontend - UI).
  - Provides superior performance as it uses lesser bandwidth which make it faster.
- The Backend is associated with the Database layer which uses SQL Server 2017.
- The Frontend is built using Angular 7, with TS, HTML and CSS alongside Bootstrap which make the User Interface responsive on any device.
- Moreover, the proposed system consist of four personas, provided with different functionalities according to the Role assigned which are as follows :-
  - **Content-Creator** who can create, view, update and delete questions, tags/subjects, assessments and quizzes.
  - **Test-Administrator** who can schedule quizzes, assign employees to a schedule to take a quiz/assessment and remove as well. Can invite employees via email to take the quiz.
  - **Employee** is able to take scheduled quizzes and gauge the understandings and learnings of oneself by opting for mock quizzes. Can view their detailed report and performance analysis.

- ❖ **Reporting-User** keeps track of the Results and Performance Analysis by Employee, Quiz and Subject/Tags.

➢ The proposed system incorporates validation at every single point, allowing only valid entries to be passed.

➢ The proposed system targets the solution for all the problems in the existing system, thereby making it easier to conduct assessments and keep continuous track of the understandings and learnings. It also helps in analyzing the performance and assists in doing the needful to improve.

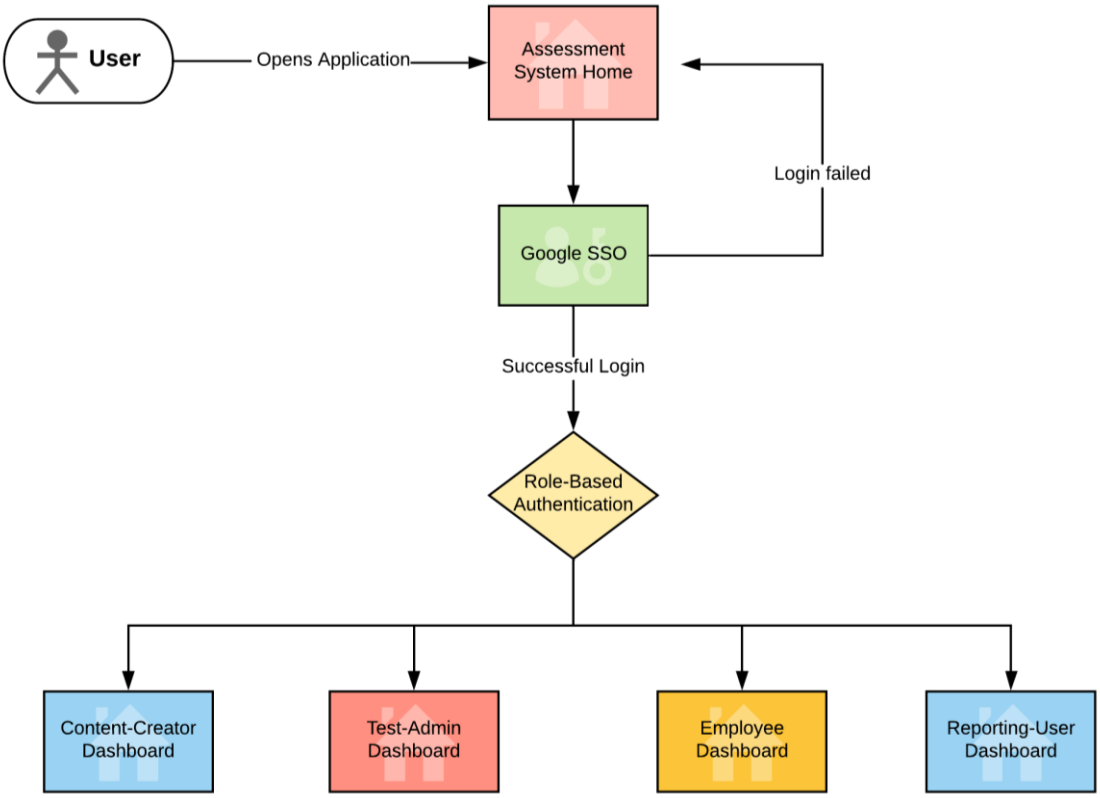

Fig 2.1: Flow of the Project

2.4 PRODUCT FEATURES AND ADVANTAGES :

- ❖ Provides a unified Intra-organization platform
- ❖ Fast and accurate.
- ❖ Portable and platform independent.
- ❖ No major hardware specifications required.

- Multiple roles allowed.
- Google SSO with Nineleaps Email Id only allowed, which eliminated the need to remember the credentials.
- Separate Dashboard for different roles.
- Can create, edit, update and delete / archive Questions, Quizzes and Tags in a few easy steps, which make creating assessment easier.
- Created questions is stored in the Question Bank.
- A question can have Image also
- Image is resized preserving the quality and aspect ratio, in order to handle the lazy loading of images at the client side and also to minimize the usage of the server space unnecessarily.
- Assessment can be scheduled in just few clicks.
- Employees are notified via Email about the Scheduled Quiz.
- Quizzes can be taken from the email itself without even hitting the Web-Application URL
- Difficulty Level like Beginner, Intermediate and Advanced are associated with the Entities to conduct assessment accordingly.
- Employees can opt for self-assessment i.e. Mock Quiz for testing their own skills.
- Search and filter provides desired data easily.
- Detailed Reports are generated for every Scheduled Quiz which can be viewed at any time in future.
- Employees' performance can be analysed using different Graphs for different performance parameters.

2.5 PROPOSED SYSTEM ARCHITECTURE :

Client - Server Architecture – In this approach, the server-side application code is kept completely separate from the client code. The server-side application has a clear responsibility to provide API endpoints while the client-side of the application strictly handles the user interface. Even the code resides in separate solutions. Each solution is deployed to its own server instance, one server handling the ASP.NET Web API, while the frontend side as the Node, serving the Angular web-client application & static files.

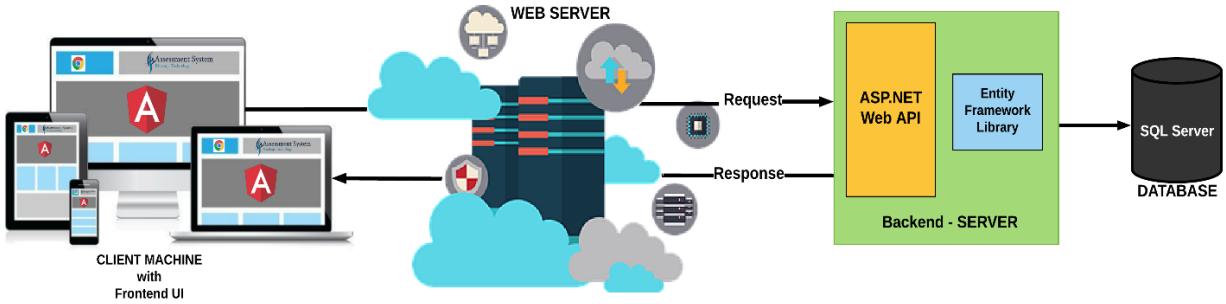

Fig 2.2: Client-Server Interaction

REST APIs are stateless, meaning that calls can be made independently of one another, and each call contains all of the data necessary to complete itself successfully. System that follow REST paradigm should not rely on data being stored on the server or sessions to determine what to do with a call, but rather solely rely on the data that is provided in that call itself. Identifying information is not being stored on the server when making calls. Instead, each call has the necessary data in itself, such as the API key, access token, user ID, providers, etc.

All the API endpoints are based on REST which is based on HTTP request methods (such as GET, POST, DELETE, PUT, and so on), responses and headers to communicate.

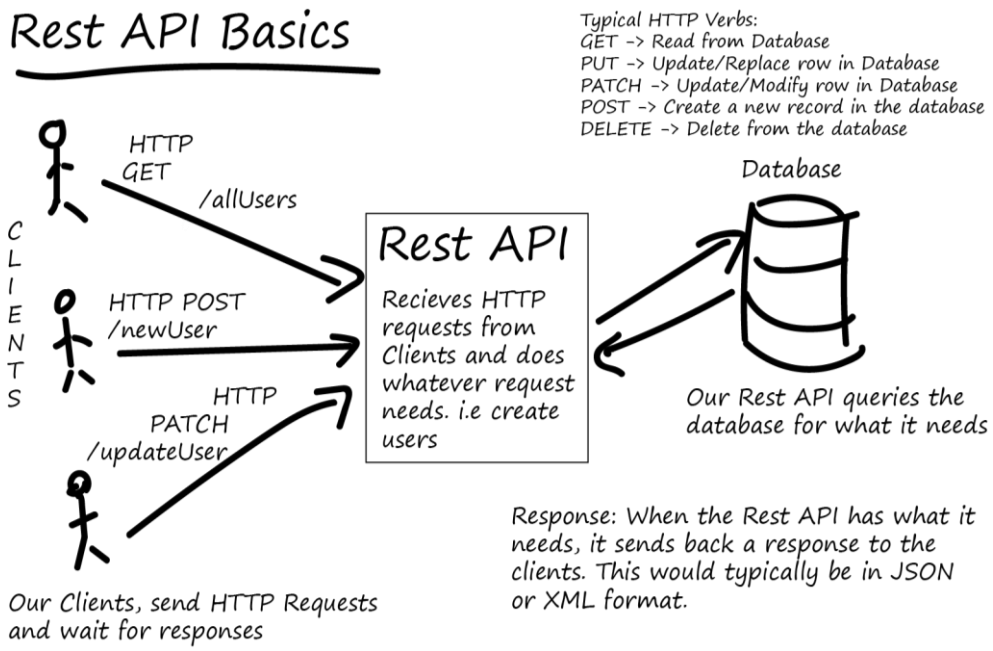

Fig 2.3: Rest API workflow

CHAPTER 3 : PROJECT IMPLEMENTATION

3.1 PROJECT DESIGN METHODOLOGY :

We know that technology is advancing everyday, in order to make User Experience better and simple. But for making any work a simple and a hassle free affair with the help of an application, it incorporates a lot complex methods in the development of the application. Thus, in order to make the development unambiguous, we follow some Software Development Life Cycle (SDLC) method such as waterfall, iterative or many other traditional methods. But these methods were not much helpful as complex requirement came into play.

The solution to these difficulties is the AGILE methodology, which has proved to be the most efficient one because :-

- It is an Adaptive Iterative Approach
- It delivers value to customer faster.
- Minimize Bureaucracy.
- It is followed in team, as they assume collective ownership to deliver success.
- It is a step by step proceedings, which maximizes learning.

In Agile, team has frequent collaboration with stakeholders (customer, product owner, developers, etc.). Continuous customer feedback, ensures that it allows us to make changes as and when needed rather than wait for the entire product to be developed before we get to know about any glitches. User requirements are divided into EPIC. EPICs are divided into User Stories. Each story is added to the Product Backlog, which is created by the Product Owner. Then Sprint Planning is done where chunks of stories are pulled to Sprint Backlog. Sprints can be 2-3 week long. End of the sprint demands the work should be potentially shippable. The Sprint Review on the product is conducted and retrospective is done on the process. Trello is one of the tools for Sprint design and making User Stories.

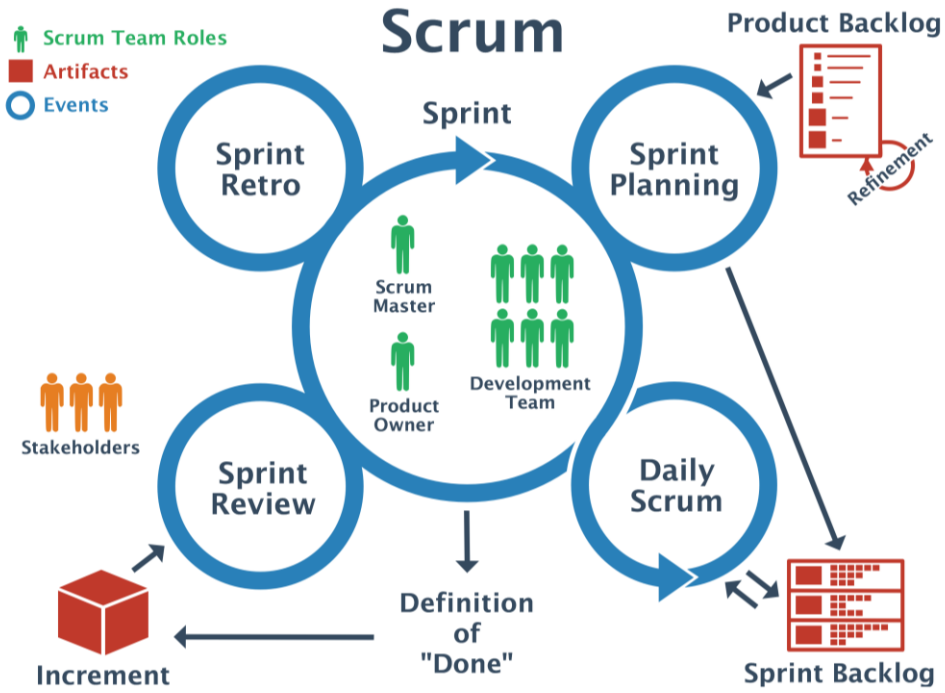

Fig 3.1: Scrum Methodology – An AGILE framework

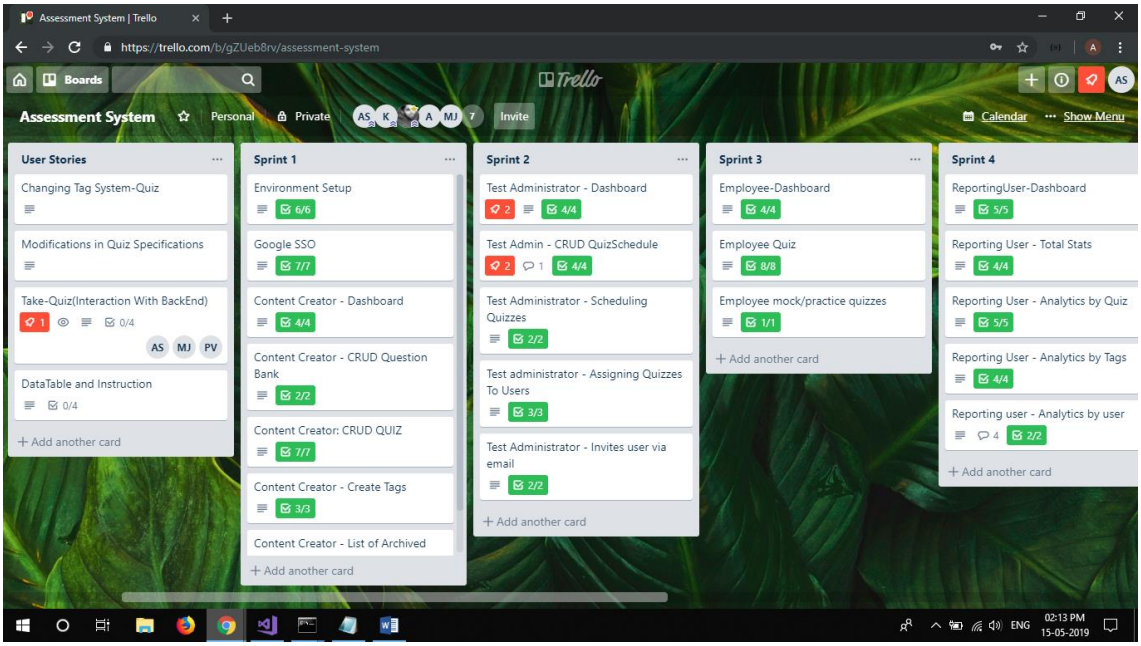

Fig 3.2: Trello Board

3.2 PROJECT DIRECTORY STRUCTURE :

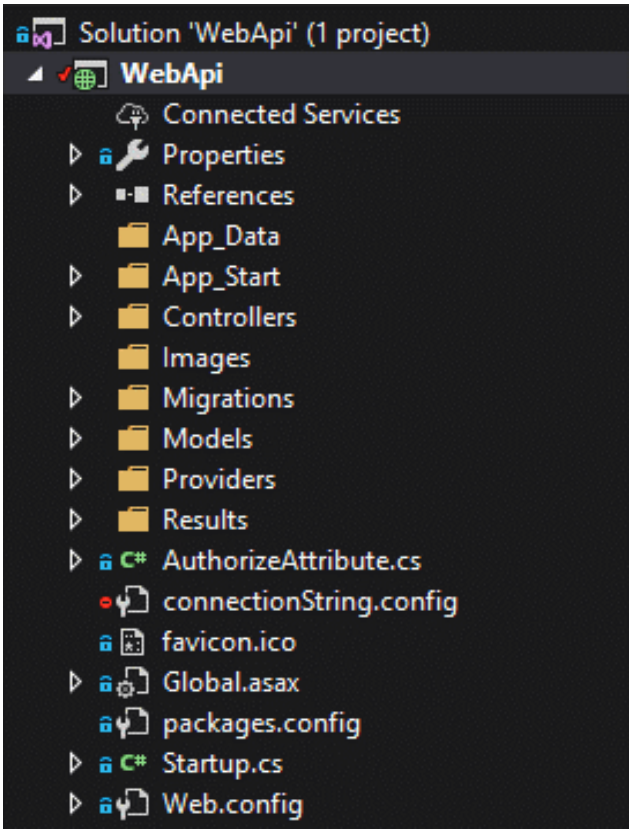

Fig 3.3: Directory Structure of ASP.NET Web API (Backend)

Models - Model represents domain specific data and business logic in MVC architecture. It maintains the data of the application. Model objects retrieve and store model state in the persistance store like a database.

Controller - The Controller is responsible for controlling the application logic and acts as the coordinator between the View and the Model.

3.3 PROJECT DESCRIPTION :

- This is the Main Home Screen with a **Google SSO** button and a button that redirects to **About Us** page.

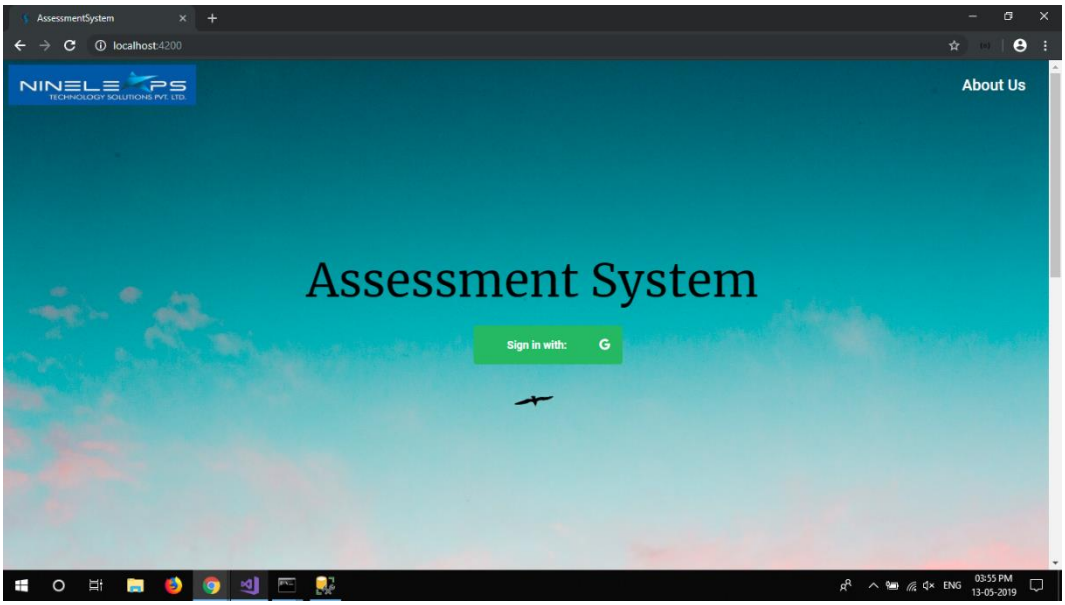

Fig 3.4

- This is the **Google Login Page** where the Nineleaps' user can enter their credentials to login to Assessment System.

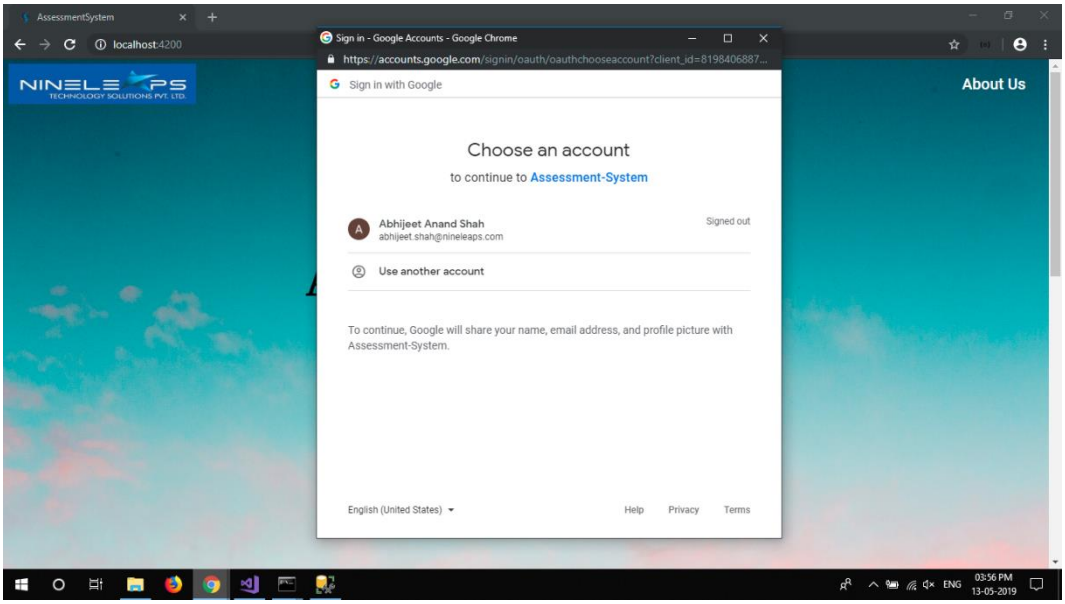

Fig 3.5

3.3.1 Content-Creator Features :

- This is the **Content-Creator Dashboard** for the User who have been assigned the **Role of Content-Creator**. The Dashboard contains the **Count of Quizzes, Tags and Questions** which the current User has created.

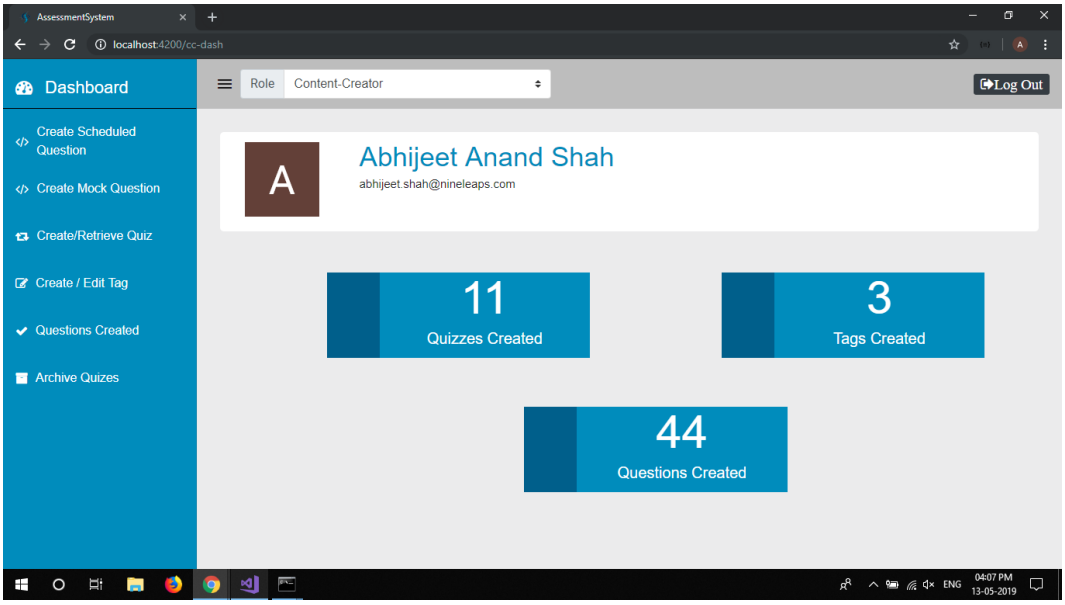

Fig 3.6

- This Dropdown is populated with only those Role which that user has been assigned. User can have **Multiple Roles** in an Organization.

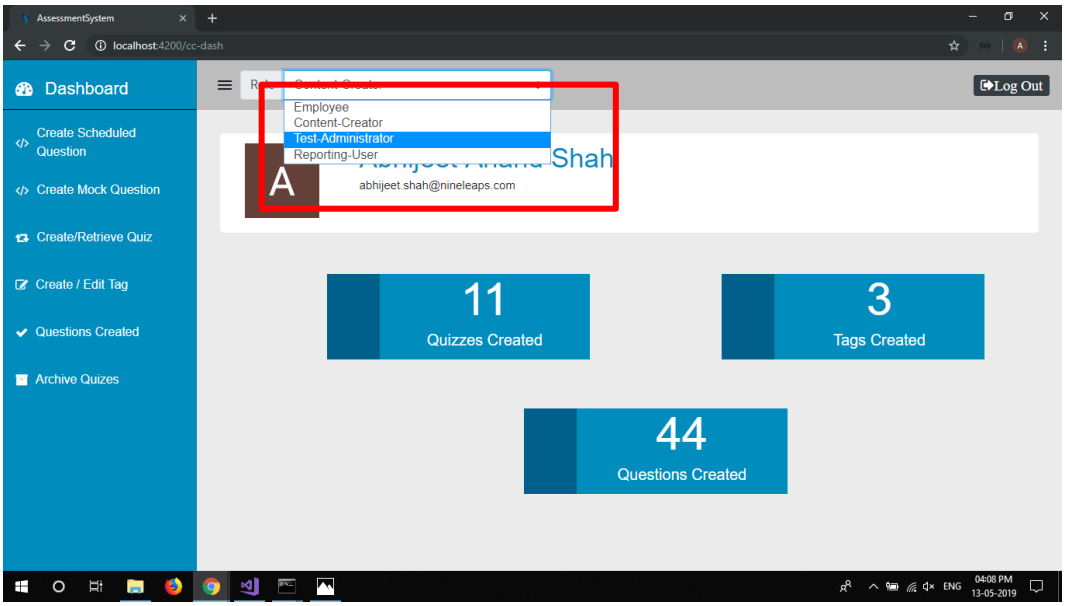

Fig 3.7

- The picture below depicts the UI to **Create a Question** of type Scheduled (similar for Mock Question) with all the necessary fields entered. As observed Option 3 and Option 4 are not mandatory. A Question may have an **Image** (optional).

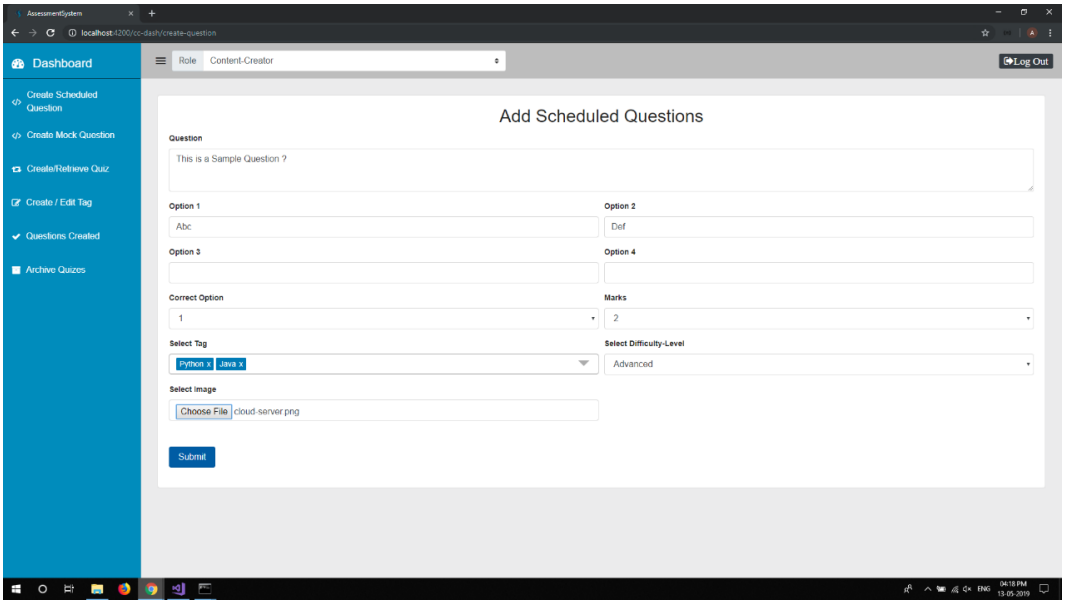

Fig 3.8

- The Client Machine or the Frontend makes a **POST** request with all the necessary details associated with the Question. On **successful creation of Question** the **ASP.NET Backend Server** responds with a **HTTP OK Status code - 200** which in turn is handled and a **Toaster Message** is displayed.

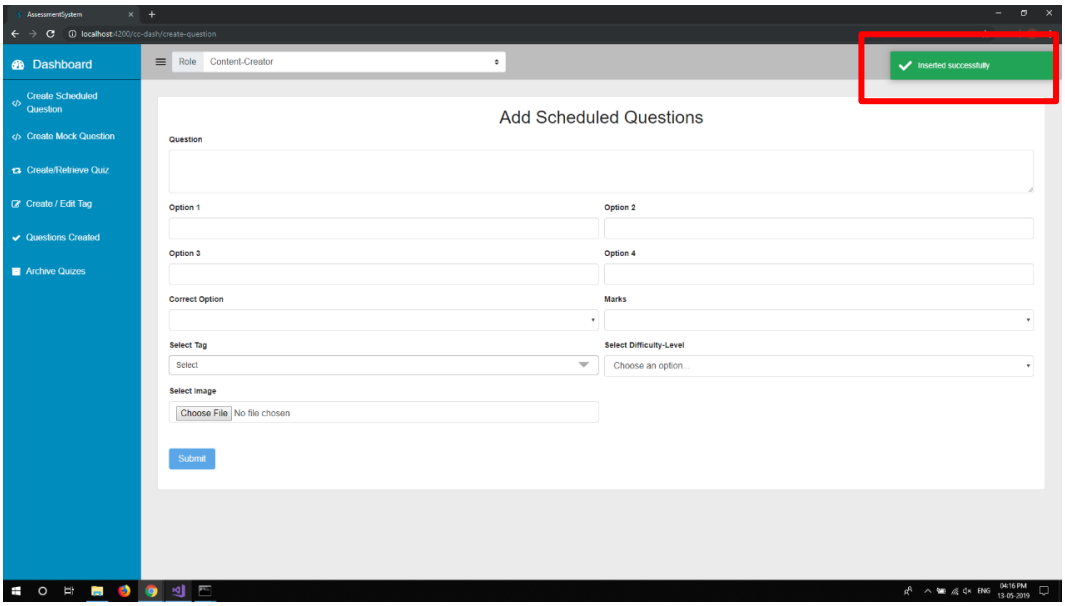

Fig 3.9

- The Content-Creator can **View the List of Question** he/she has created along with the **Action Button** which allows the user to **Edit (Blue)** or **Delete (Red)** the Question.

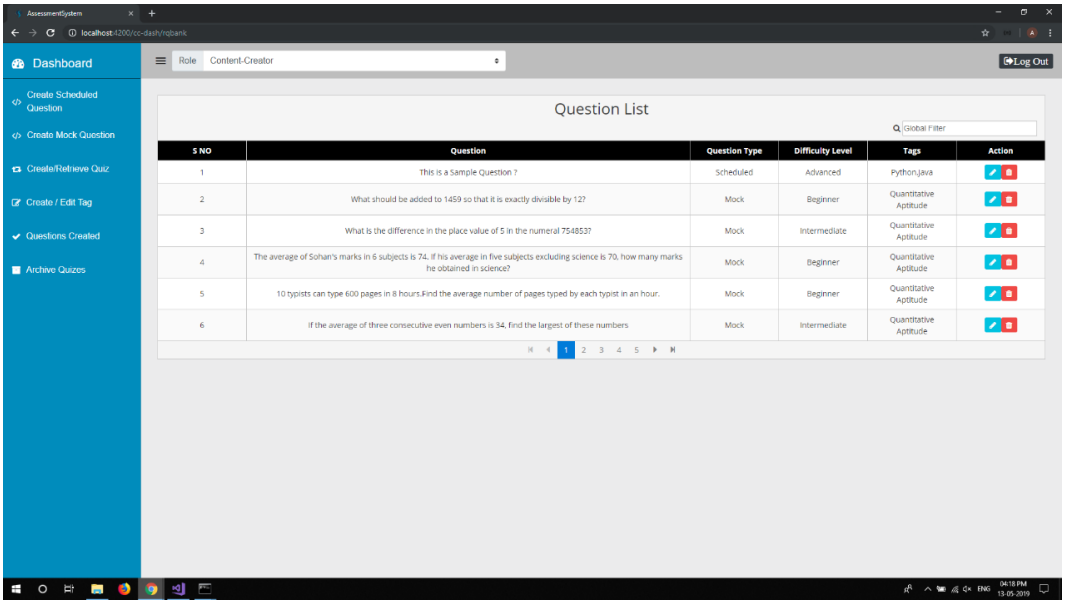

Fig 3.10

- The user can **View** a particular question which open in a Mat-Dialog Box along with the Labels.
- The user can **Edit** the Question which also opens in Mat-Dialog with a **Submit** Button to save the new changes.

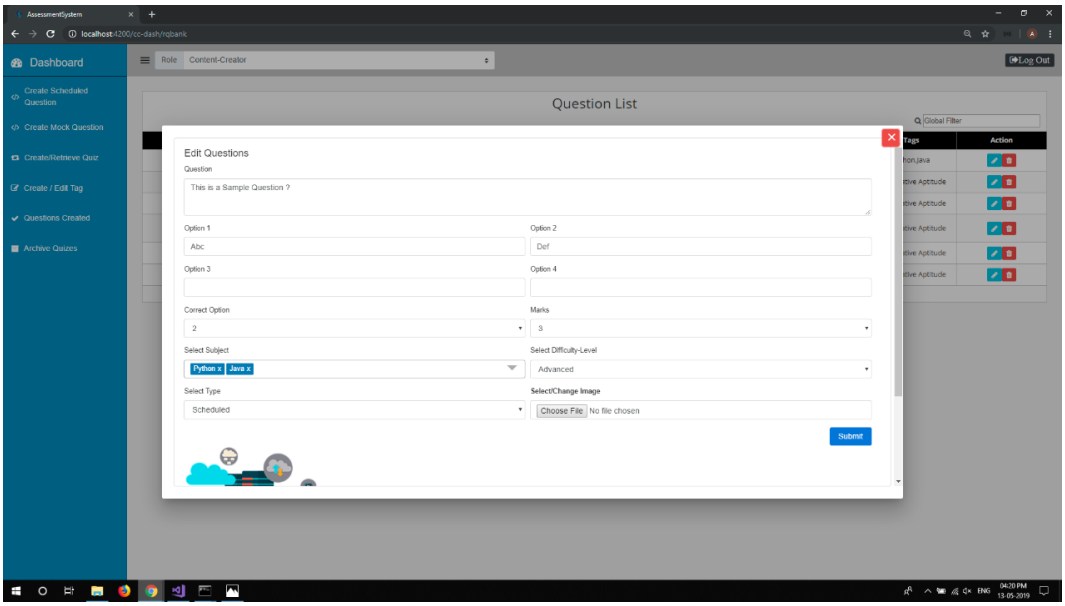

Fig 3.11

- On clicking the Delete (Red) button a prompt asks for confirmation.

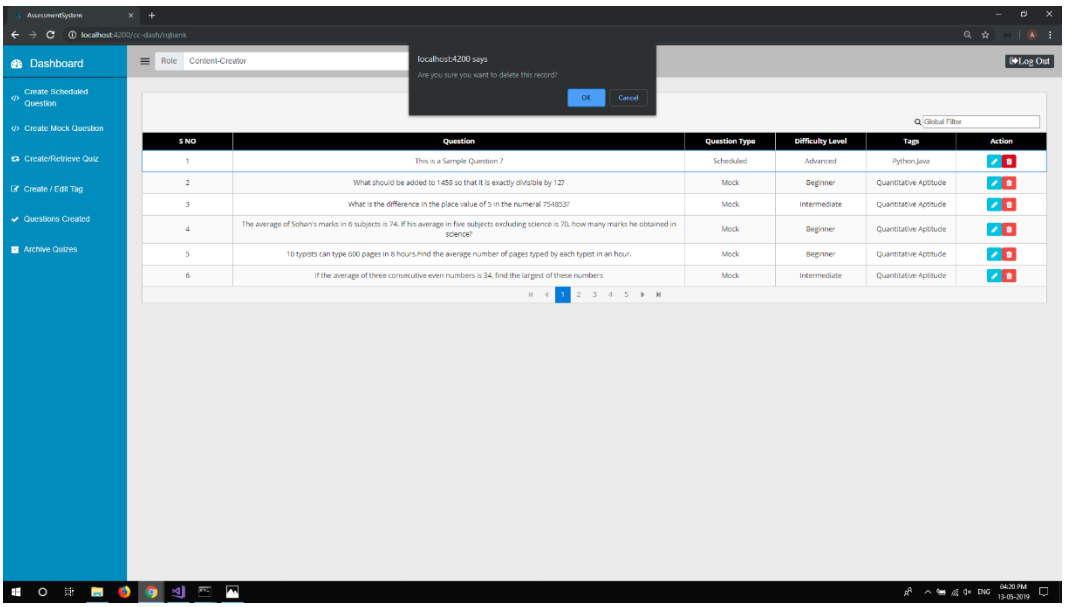

Fig 3.12

- The user can **create** a new quiz with **Create Quiz** button. The user can **View/Retrieve** the created **Quiz** along with the **Action Button** to **Edit (Blue)** or **Archive (Red)** a Quiz.

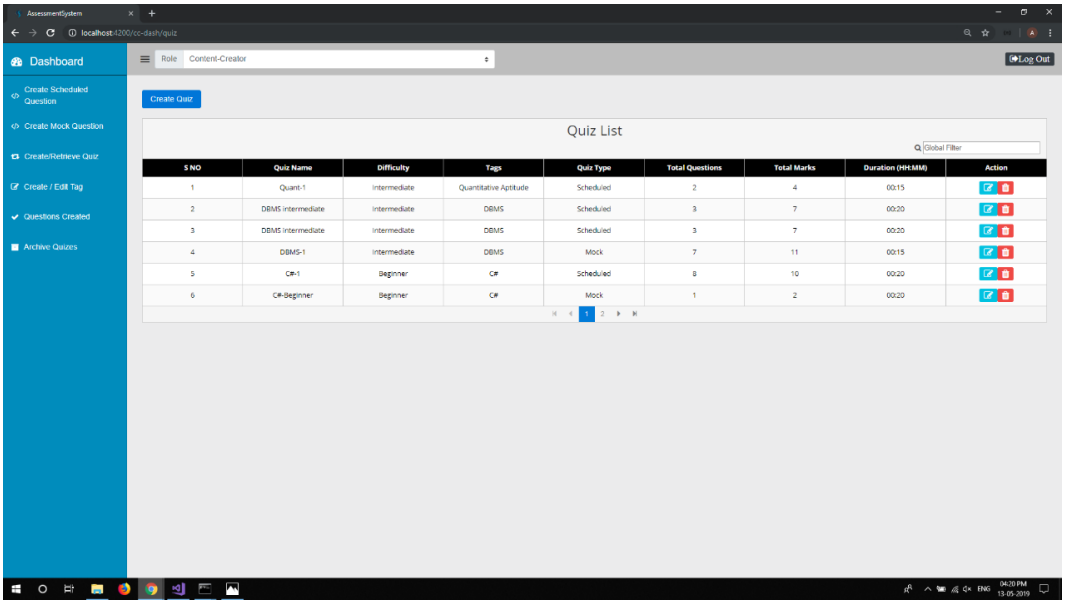

Fig 3.13

- The user can **Create** a New Quiz by entering the details and the **Submit** Button make the **POST** request to the Backend Server in order to create a **Quiz**.

  Quiz Difficulty can be Beginner, Intermediate or Advanced

  Quiz Type can be Mock (Random or Personalized) or Scheduled

  Quiz can be set Active/Inactive

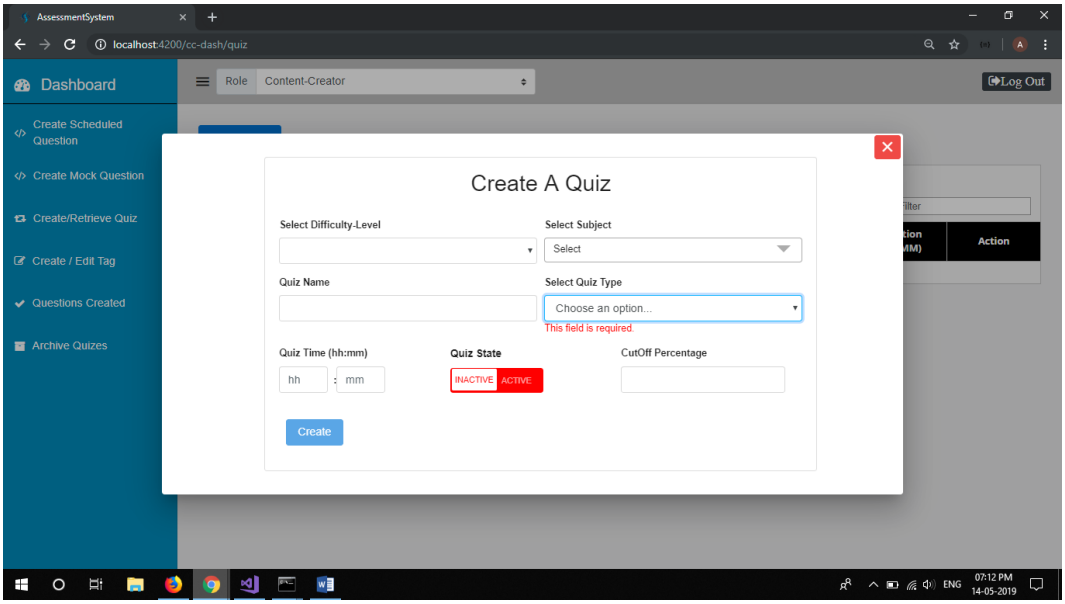

Fig 3.14

- On **Editing**, the user can **add/remove** existing question in the quiz or he may add a new question to the quiz.

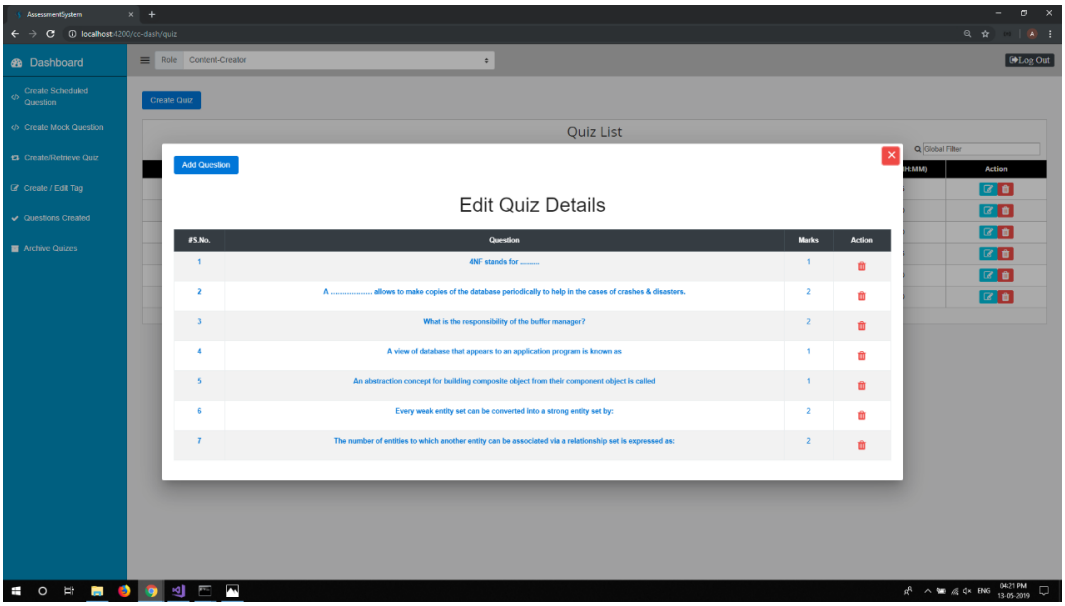

Fig 3.15

- The **List of Archived Quizzes** are available with an Unarchive button, which enables the user to **Restore** the Quiz at any point of time in future.

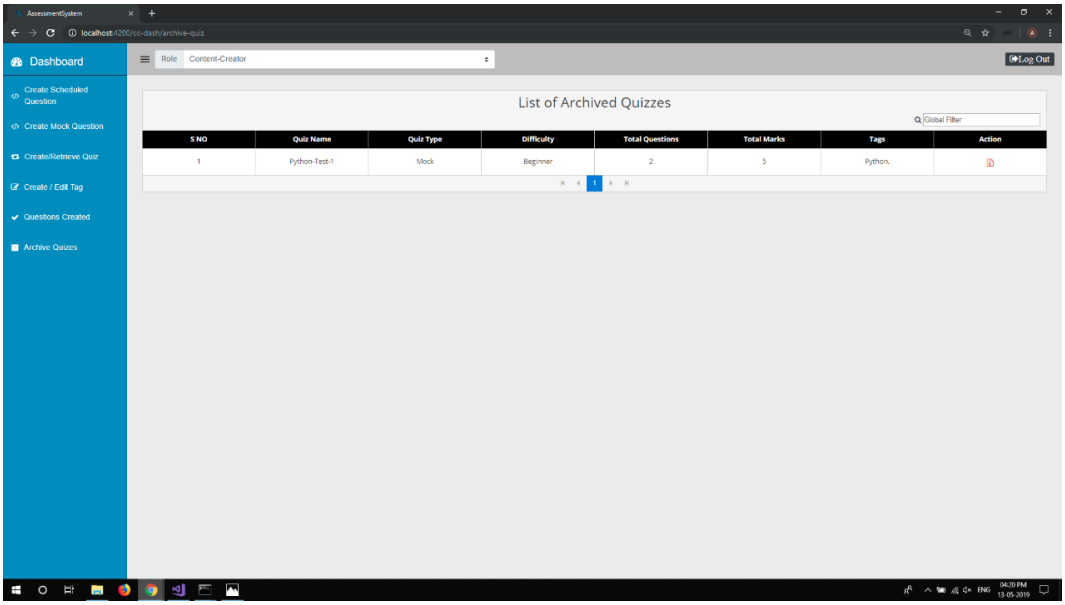

Fig 3.16

- The user can **View** and **Edit** the Tags from the Tags List and can create a New Tag from **Create Tag** button.

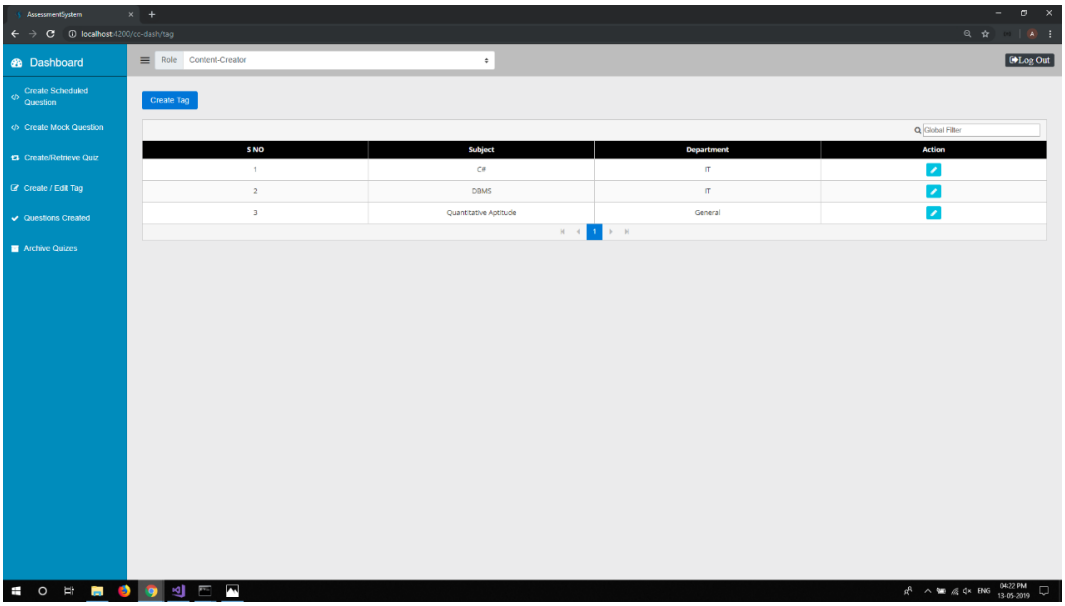

Fig 3.17

➢ The user can **Create** a New **Tag** by submit the required details and making a **POST** request to the Backend-Server.

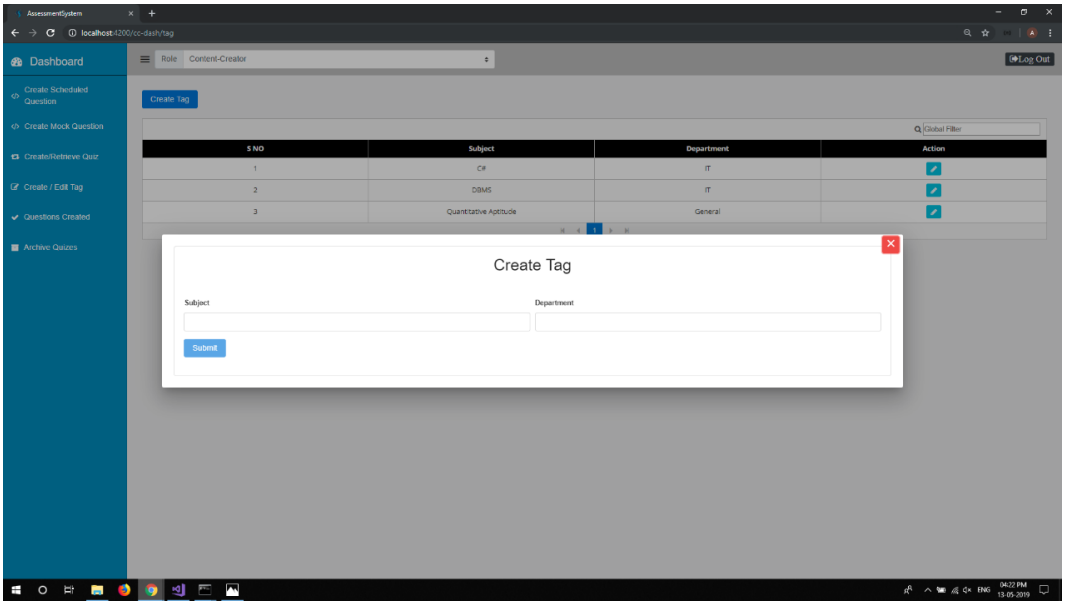

Fig 3.18

3.3.2 Test-Administrator Features :

➢ This is the **Test-Administrator Dashboard** for the User who have been assigned the **Role of Test-Administrator**. The Dashboard contains the **Count of Quiz-Schedule** which the current User has created.

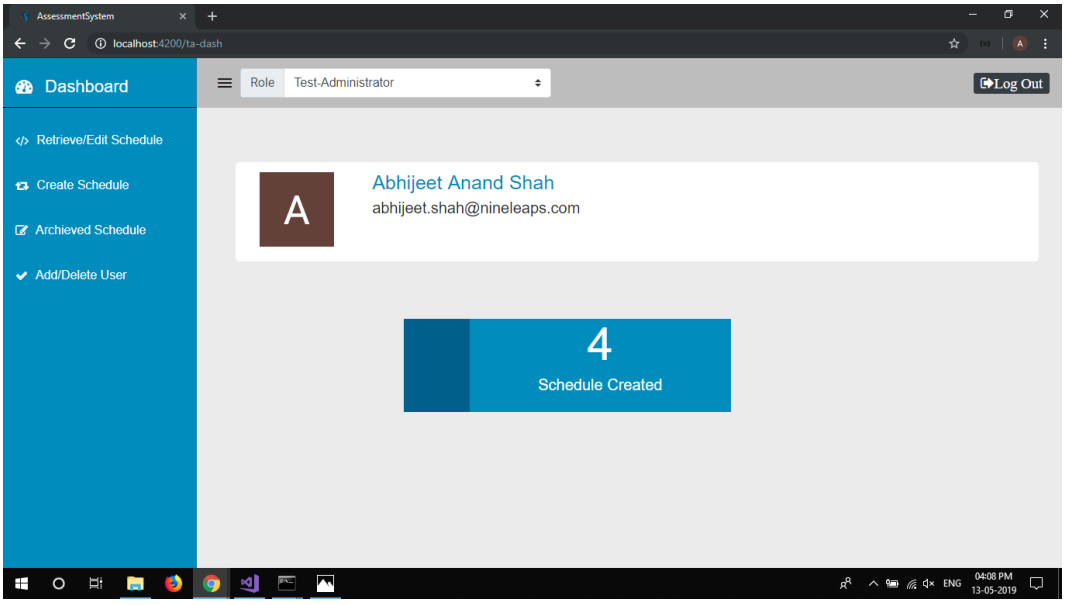

Fig 3.19

- The user can **Create** a **Quiz-Schedule** by selecting the **Start Date-Time** and the **End Date-Time** and the **Quiz** to be Schedule from the List of Active Quizzes.

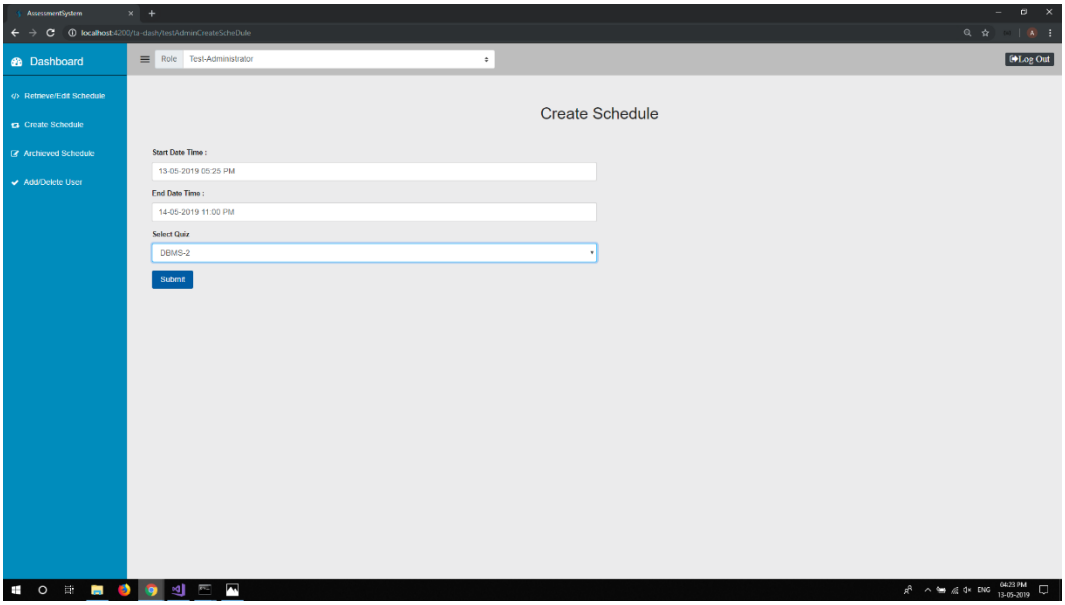

Fig 3.20

- The user can **Edit (Blue) / Archive (Red)** a Quiz Schedule. Start Date-Time and End Date-Time can be edited in a **Quiz-Schedule**.
- The **List of Archived Quiz-Schedules** are available with an Unarchive Action button, which enables the user to **Restore** the Quiz at any point of time in future.

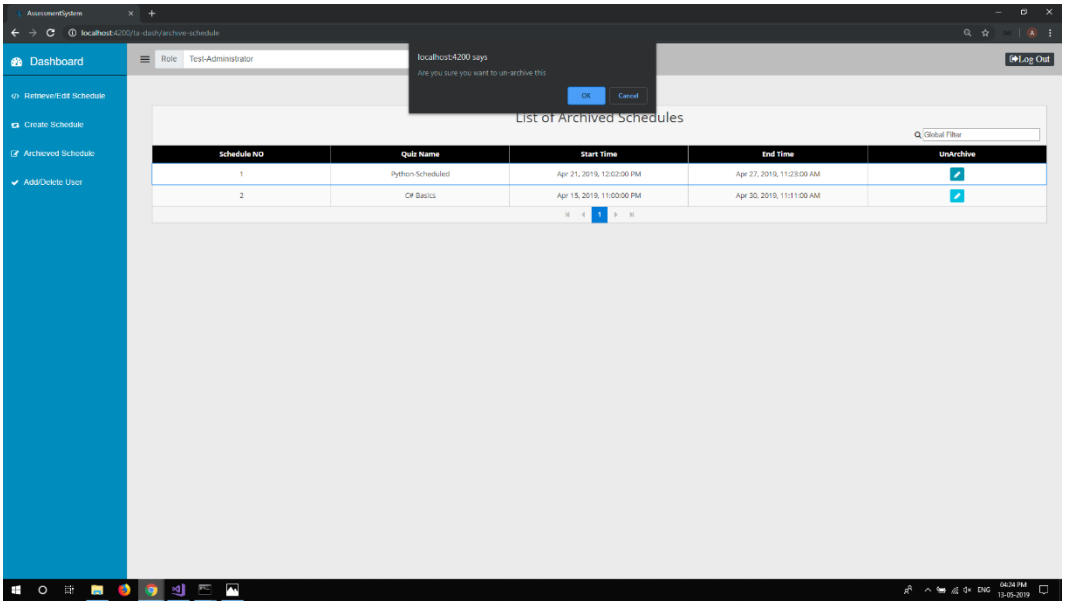

Fig 3.21

- The user can **Add/Remove** Employee in a Quiz-Schedule. An employee once assigned in a Quiz **cannot** be **assigned** the **same quiz** in another Schedule.

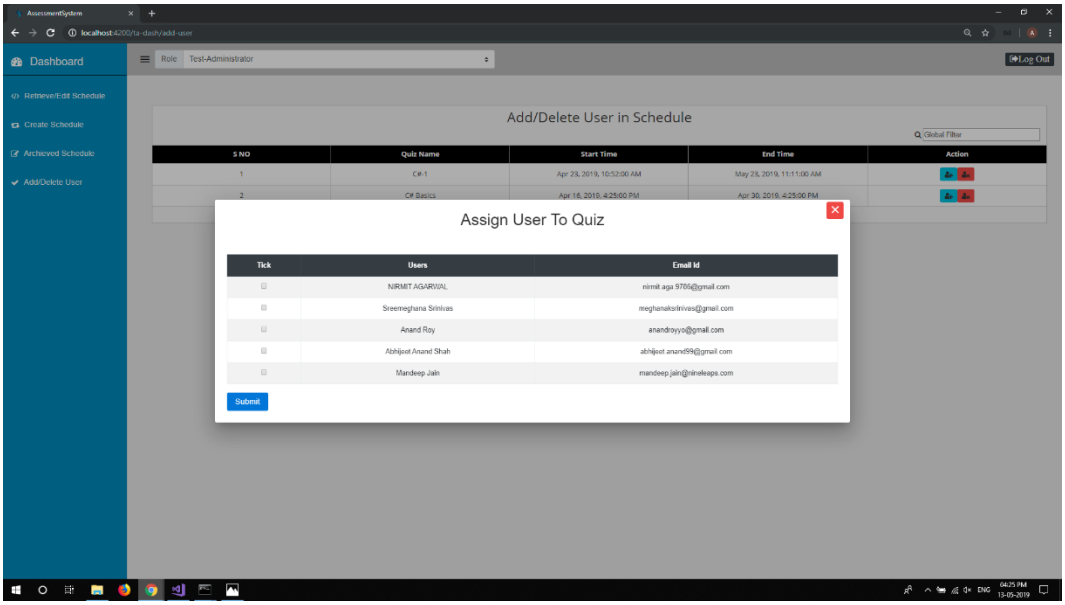

Fig 3.22

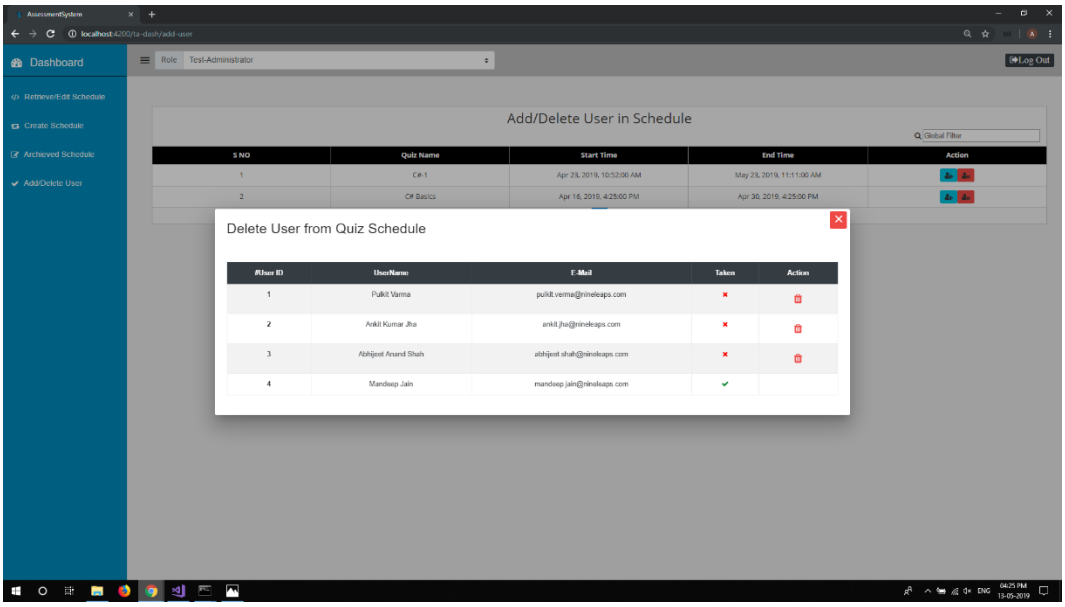

Fig 3.23

3.3.3 Employee Features :

- This is the **Employee Dashboard** for the User who have been assigned the **Role of Employee**. The Dashboard contains the **Count of Mocks Taken and Schedule Quiz Taken and also Accuracy and Recent Quiz Taken** by the current User.

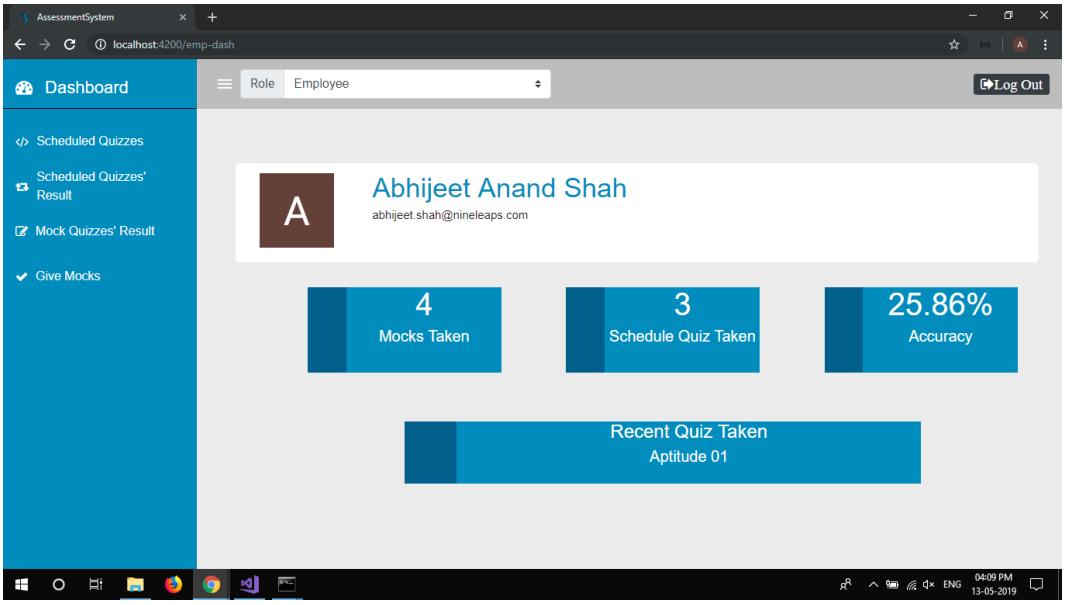

Fig 3.24

- The **Scheduled Quiz List** enlists the Quizzes scheduled for the current User i.e. Employee. Only those Quiz-Schedule will appear whose Start Date-Time and End Date-Time is valid. Quiz can be taken from the **Arrow (Blue)** button.

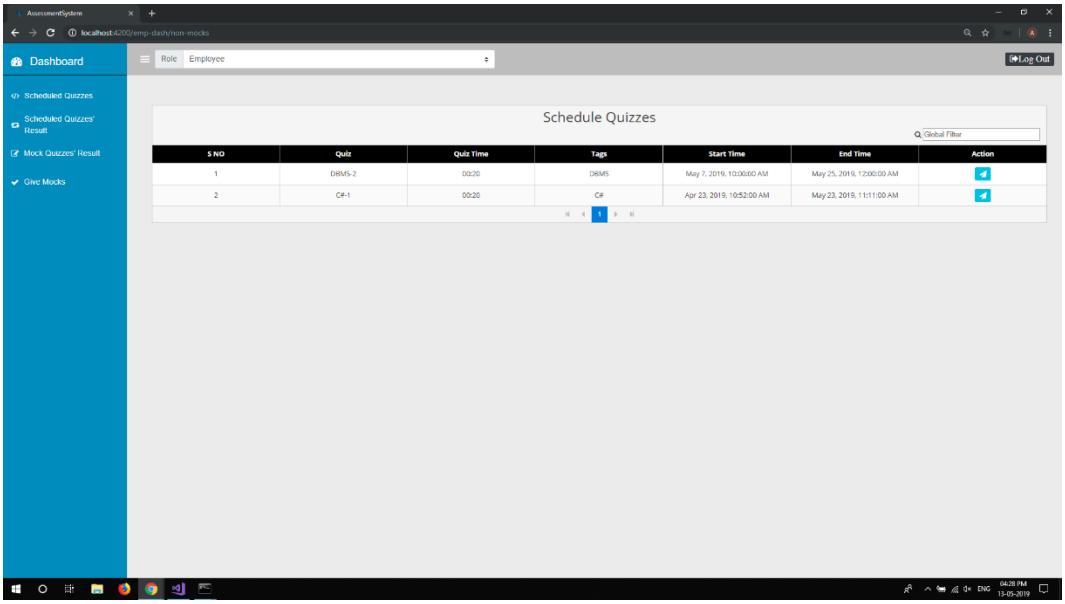

Fig 3.25

- Similarly the **Mock Quizzes** are **enlisted** in a separate Tab.

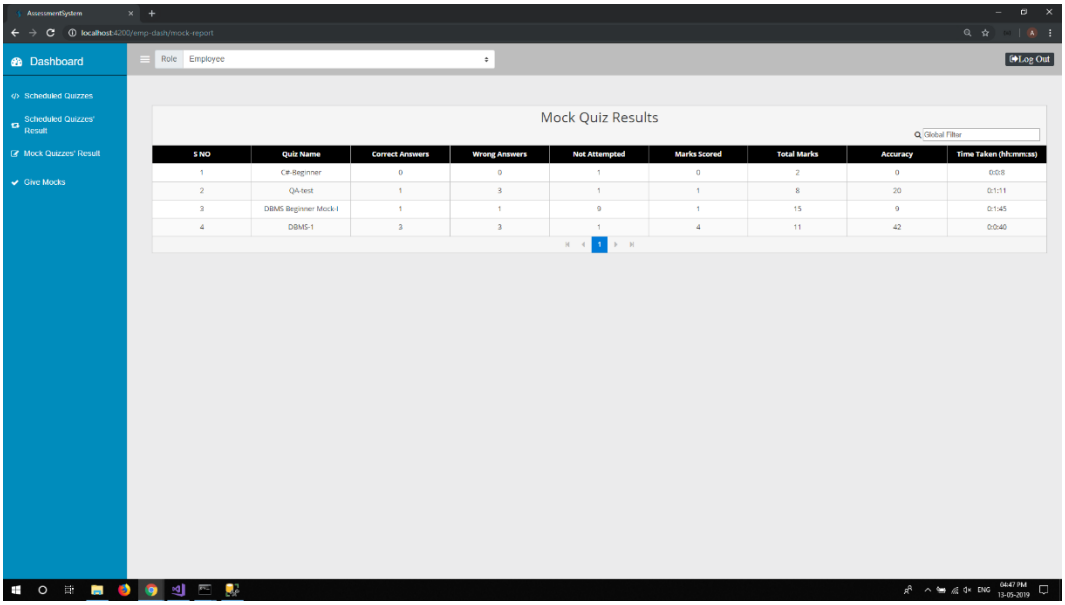

Fig 3.26

- The user can **Take** the **Quizzes** be it **Mock / Scheduled** to complete the **assessment task**.

  Active Question is marked **Yellow**

  Marked and Saved Question is colored **Green**

  Marked for Review is colored **Blue**

  Progress Bar & **Countdown timer** enables to keep **track** of Progress & Timeleft

  Question can be skipped with **Skip** & Quiz can be **Submit** at any point of Time

Fig 3.27

- On **submitting** the Quiz, the **Report** is generated for **Scheduled Quiz** which can be **viewed** at any point of time in future. Similarly, **Mock Quiz** reports are generated in separated Tab.

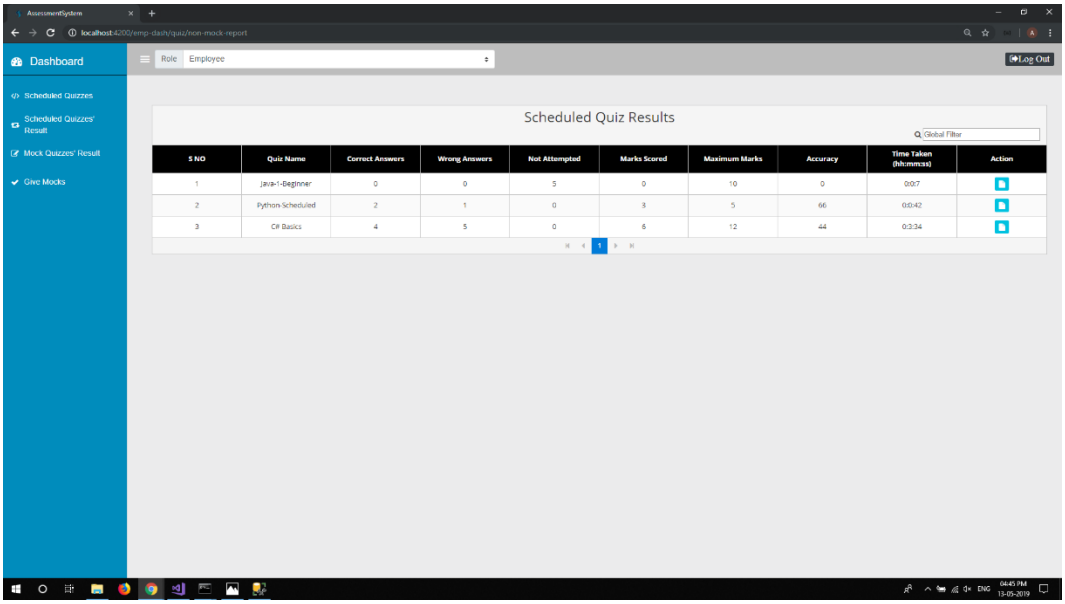

Fig 3.28

- **Detailed Report** can be viewed for Scheduled Quizzes from the **View (Blue)** Action button. In this, each **questions' response** and **correct answer** is shown along with the **Graphs** which depicts **Attempted / Unattempted** and **Correct / Incorrect**.

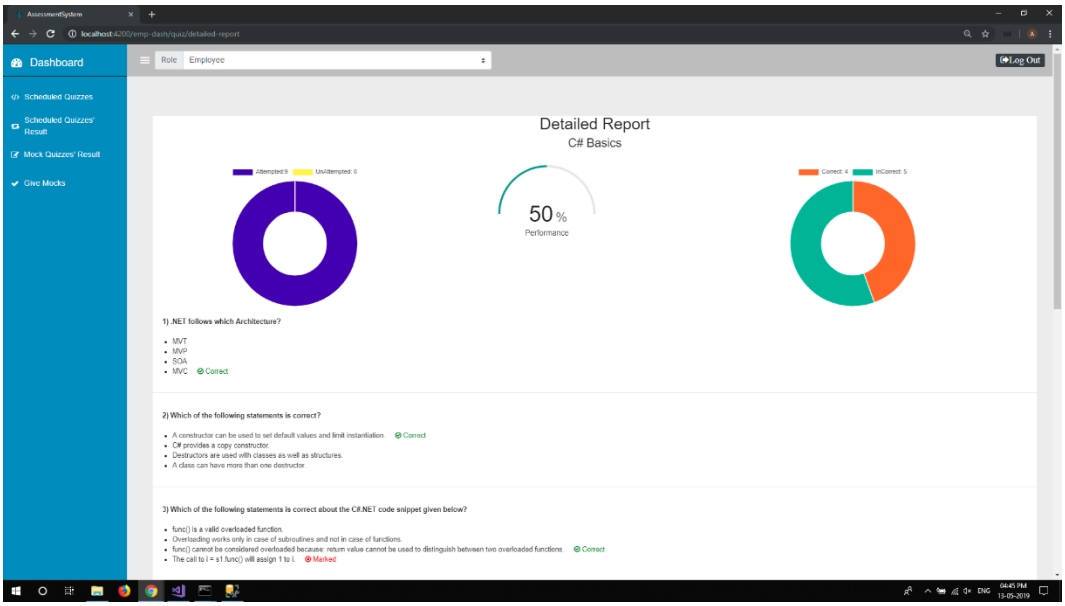

Fig 3.29

3.3.4 Reporting-User Features :

- This is the **Reporting-User Dashboard** for the user who have been assigned the **Role of Reporting-User**. The Dashboard contains the **Count of Mocks Taken and Schedule Quiz Taken and also Accuracy and Recent Quiz Taken** by the current User.

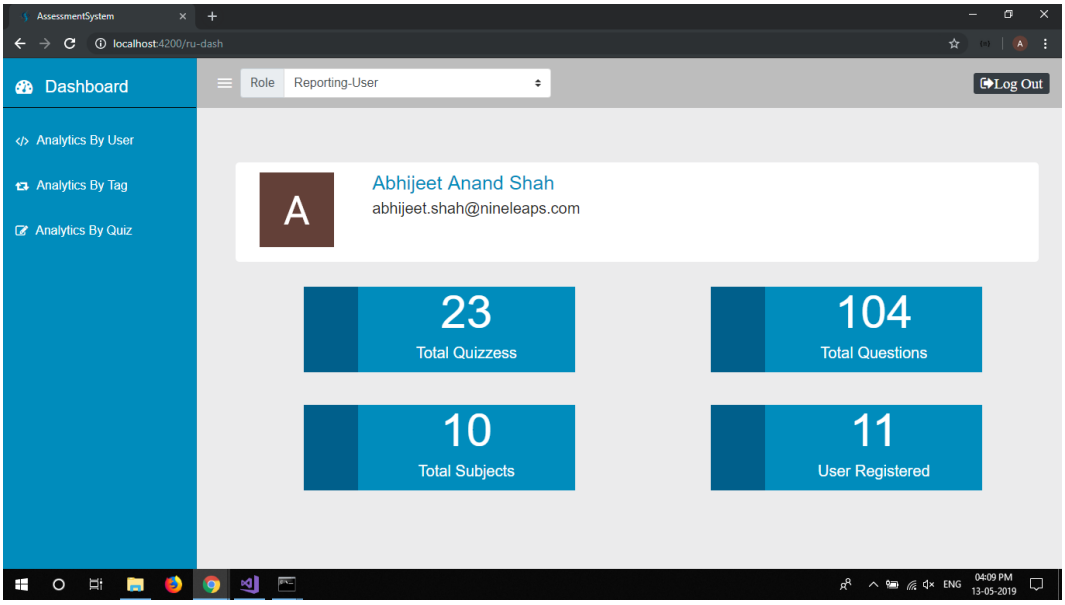

Fig 3.30

- The **Analytics By User** section enlist every user registered to the Web-Application. **View (Blue)** button opens up the Detailed Report of a user.

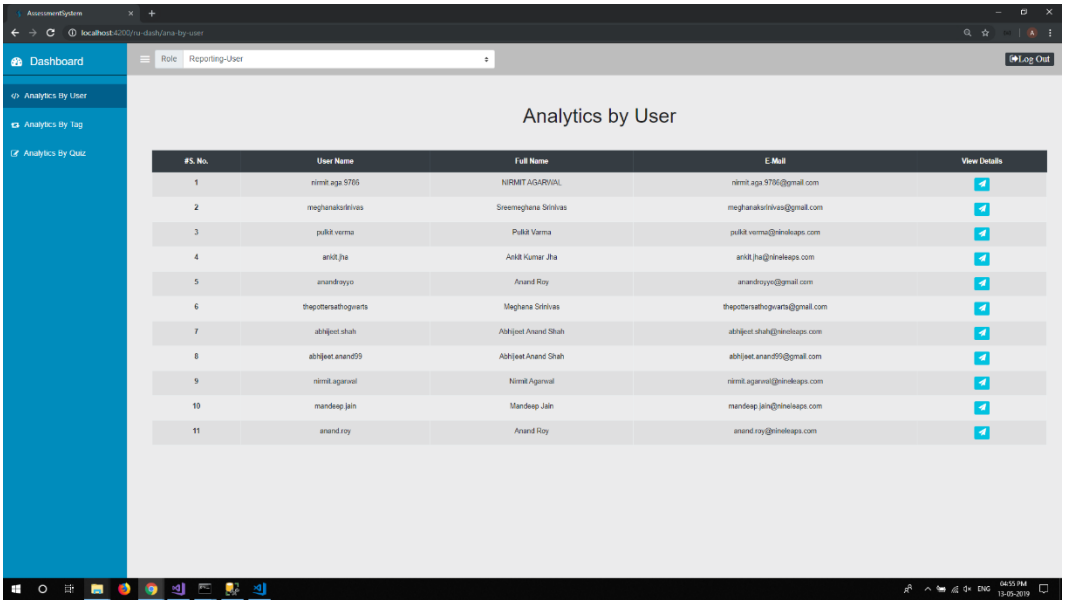

Fig 3.31

- The Overall Detailed Analysis has Graph with different Performance Parameters Stats and also the reports for each quiz the user has taken.

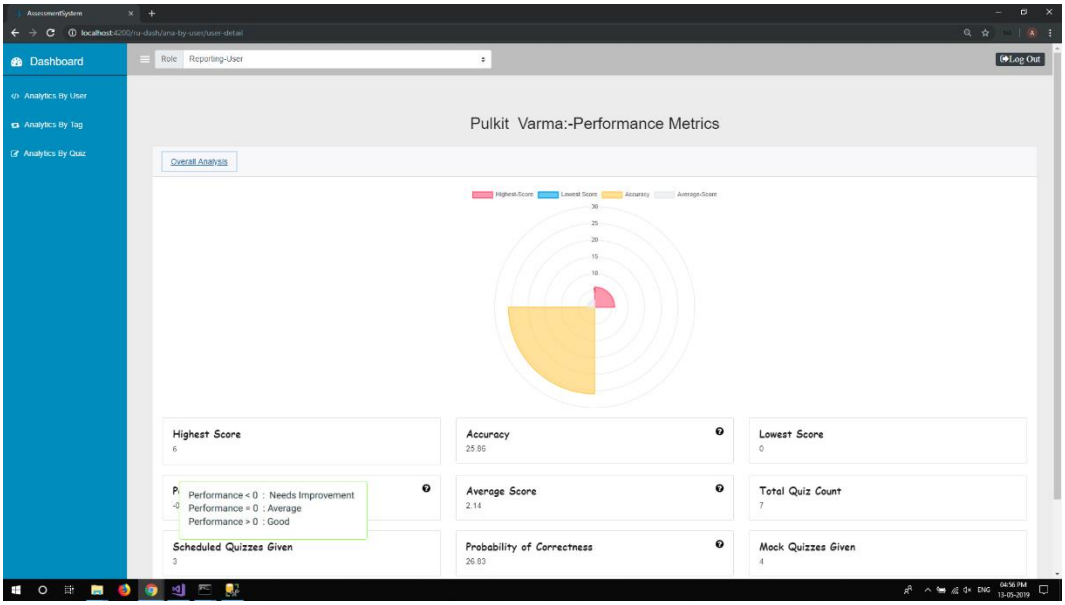

Fig 3.32

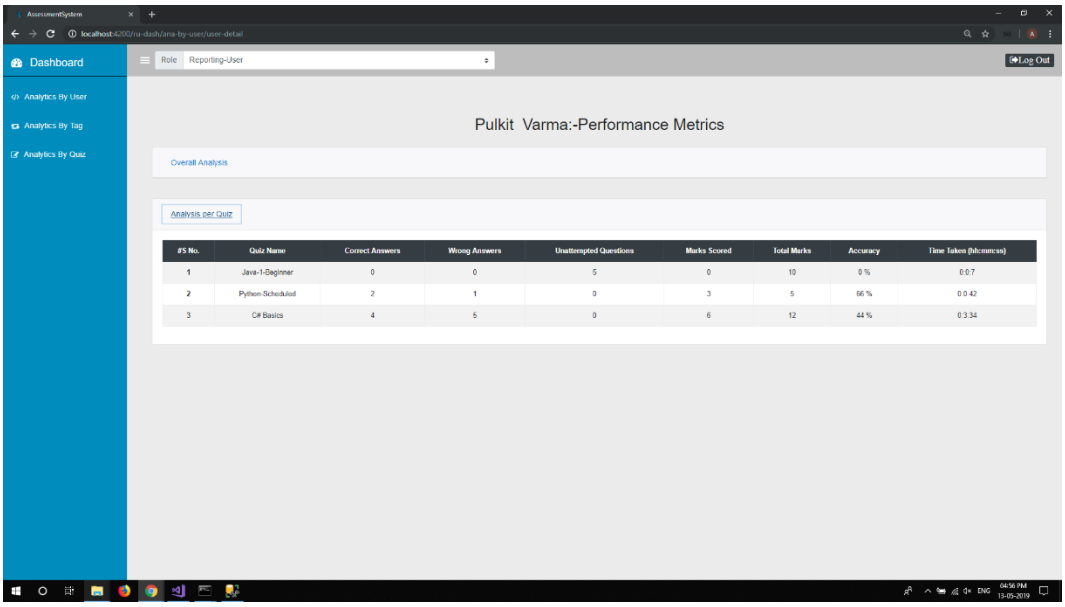

Fig 3.33

- The **Analytics By Quiz** section shows how the employee taking up the Quiz has perform overall. **View (Blue)** button opens up the Detailed Report of a Quiz.

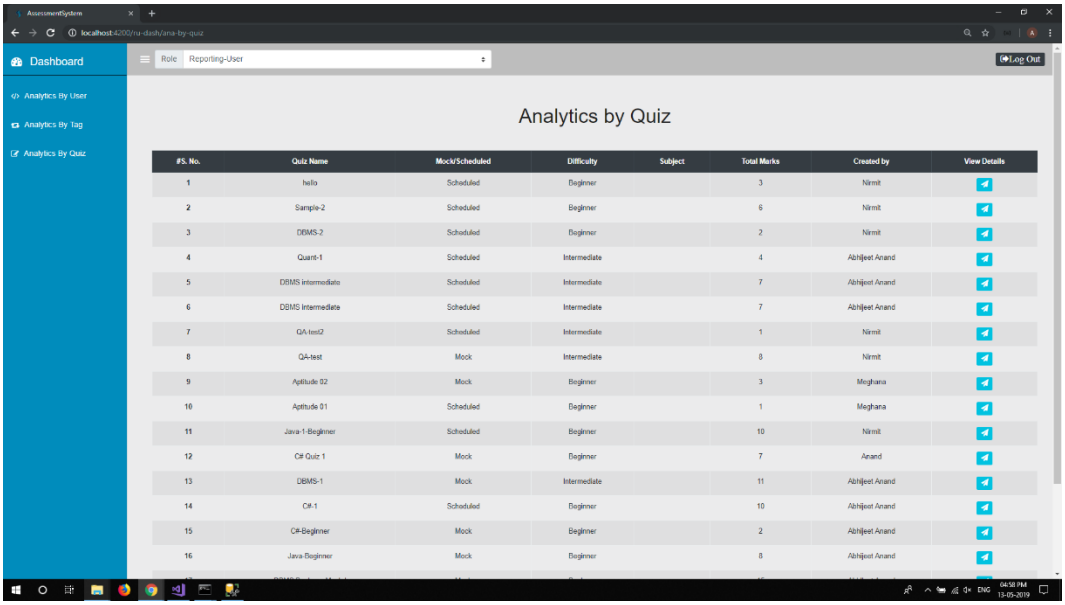

Fig 3.34

- The Overall Detailed Analysis of Quiz has Graph with different Performance Parameters Stats.

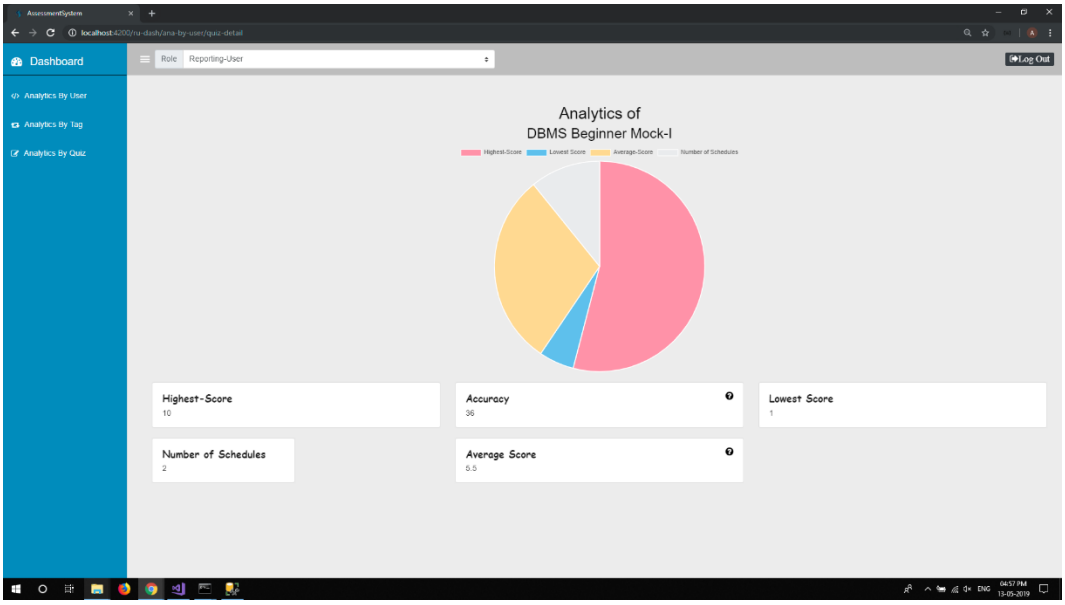

Fig 3.35

3.4 FUNCTIONAL REQUIREMENTS :

The Web APIs must :-

- Be able to utilize the application by logging in through Google SSO.
- Be able to distinguish among different roles.
- Be able to provide all the features with no or minimal delay.
- Be able to create, read, update and delete data from a databases easily.
- Be able to present all the feature in a such a way that it is a hassle free affair.
- Be able to handle all possible Validations.
- Be able to do Exception Handling.
- Be able to prevent data tampering.
- Be able to handle continuous interactions with the frontend.
- Be able to make it available for user to take the assessments.
- Be able to schedule a quiz and assign employee to the schedule.
- Be able to send email notifications.
- Be able to Archive Quizzes and to set Quiz State to Active or Inactive.
- Be able to invalidate the Quiz Schedule if the Current Date & Time is not between Start Date and End Date of the Scheduled quiz.
- Be able to generate the detailed report after quiz is taken and submitted.
- Be able to generate Graphs on different performance parameters.
- Be able to switch roles, if user has multiple role.
- Be able to perform analysis based on Quizzes, Subjects and each Employee/User.

3.5 NON FUNCTIONAL REQUIREMENTS :

The application must adhere to the following non-functional requirements, in order to develop a system that is clear, concise, and simple to use.

3.5.1 Functionality :

Measures shall be taken to prevent intruder access to the server and ensure the proper identification and verification of system's users. The user's authentication information will not be transported in plain text.

3.5.2 Environment :

Since it is a Web Application, it can run on any device on any platform with a Web Browser along with an Internet Connection and an authorized Nineleaps Email Id to utilize the Application.

3.5.3 Usability :

All the features of the application are accessed through Webpages with self-descriptive User Interface. Hence application presents a fairly intuitive and friendly graphic interface. Also there is no need to remember credentials or commands as application has interactive buttons and dropdown to trigger different features.

3.5.4 Reliability :

The server must be able to handle sufficient load without crashing. The server should respond with appropriate Http Status Codes which is handle at the UI, thereby triggering the interceptor with the message asking necessary action to be taken to complete the action.

3.5.5 Interoperability :

Side Navigation Bar, Button and Dropdowns can lead you to any functionality directly in just One-Click.

3.5.6 Security :

Application requires user authentication which generate Google Access Token / ID Token which is encrypted and it is valid for some duration only. The Access Token is required to access the API endpoints as request header in the Http call. Application prevents User from unauthorized access, even if the user hits the URL in the browser directly.

3.5.7 Standards :

Application has been developed, considering the coding standards such as - DRY (Don’t Repeat Yourself), SOLID, etc. Meaningful names to classes, methods and variables.

3.5.8 Maintainability :

Identification of deficiencies, failure causes is easy. Effort needed for modification and migration is not a tedious task.

3.6 REQUIREMENT SPECIFICATIONS :

3.6.1 SOFTWARE REQUIREMENTS :

3.6.1.1 Software Requirements to develop the Proposed System :

Operating System	Windows 7 or above
Language (Backend)	ASP.NET, C# and .NET framework
Language (Frontend)	NodeJS, Angular 7, HTML, CSS and Bootstrap
IDE (Backend)	Microsoft Visual Studio 2017
IDE (Frontend)	Visual Studio Code
Database	SQL Server 2017
Database GUI Tool	Microsoft SQL Server Management Studio 2017
Tools Used	▪ Postman (to run the APIs) ▪ Swagger (aides in development across the entire API lifecycle, from design and documentation, to test and deployment) ▪ GitHub (used for version control and collaborative coding in teams) ▪ Trello (the visual way for teams to collaborate on any project by using the AGILE method)

Table 1.1: Software Requirements to develop the System

3.6.1.2 Software Requirements to run the Proposed System :

Operating System	Can run on multiple platforms regardless of OS or device as long as the browser is compatible. E.g. : Windows, Linux, Mac OS, Android, etc.
Browser	Any Browser (preferably Google Chrome for high-performance, cause of Google's open source V8 JavaScript engine support)
Other	Internet Connectivity

Table 1.2: Software Requirements to run the System

3.6.1.3 Swagger UI (Other Requirements) :

Swagger UI allows anyone, be it your development team or your end consumers. It helps to visualize and interact with the API's resources without having any of the implementation logic in place. It's automatically generated from your OpenAPI (formerly known as Swagger) Specification, with the visual documentation making it easy for back end implementation and client side consumption.

- Dependency Free - The UI works in any development environment, be it locally or in the web.
- Human Friendly - Allow end developers to effortlessly interact and try out every single operation your API exposes for easy consumption.
- Easy to Navigate - Quickly find and work with resources and endpoints with neatly categorized documentation.
- All Browser Support - Cater to every possible scenario with Swagger UI working in all major browsers.
- Fully Customizable - Style and tweak your Swagger UI the way you want with full source code access.

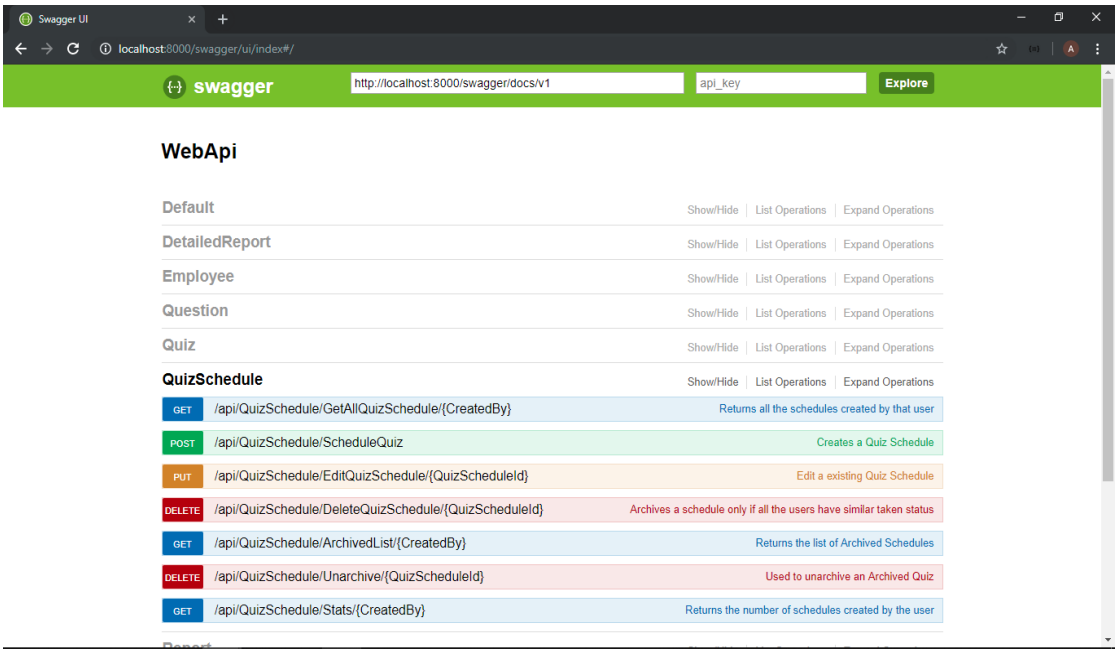

Fig 3.36: Swagger UI for Backend Developers

3.6.2 HARDWARE REQUIREMENTS :

3.6.2.1 Hardware Requirements to develop the Proposed System :

Processor	Intel CORE i5 8th Gen, CPU @ 1.60GHz
Disk	256 GB SSD
Memory	16 GB
System Type	64-bit

Table 1.3: Hardware Requirements to develop the System

3.6.2.2 Hardware Requirements to run the Proposed System :

To run the web application, we need a hardware compatible to run a Web Browser with V8 JavaScript Engine support.

Processor	Atleast Intel Core or above
Disk	1 GB or higher
Memory	256 MB or higher
System Type	32-bit

Table 1.4: Hardware Requirements to run the System

CHAPTER 4: PROJECT DESIGN

4.1 DOCUMENT PURPOSE :

The purpose for this document report is to specify the prerequisites of the project under development called **Assessment System**. This document portrays that the application can be used to conduct assessments and helps to administer the continuous evaluation of the employees. The employee can take up the scheduled quiz from the application or from the email notification. The performance of the employee can be seen through different Graphs for different performance parameters. The document will help the user to be aware of the features and its usage methods.

4.2 PROJECT SCOPE :

The scope of the product defines that the product can allow the employees to take quiz by signing in from their Nineleaps Gmail Account and also for other role to trigger their tasks.

- Maintaining the track of User
- Maintaining the Question bank when Content-Creator creates questions
- Maintaining the Quizzes and Tags generated
- Maintaining the record of Quiz scheduled to employees
- Maintaining the Detailed Report of the Quiz taken by the employee
- Maintaining the Analysis based on different parameter

4.3 PROJECT PERSPECTIVE :

The primary perspective of the product is to provide a unified Intra-organization platform to the organization, which enables them in assessing and gauging the learnings and understanding of their employees by conducting assessments in a few integral steps.

4.4 PROJECT DESIGN, FLOWCHART & OTHER RELATED DIAGRAMS :

4.4.1 Application Flowchart :

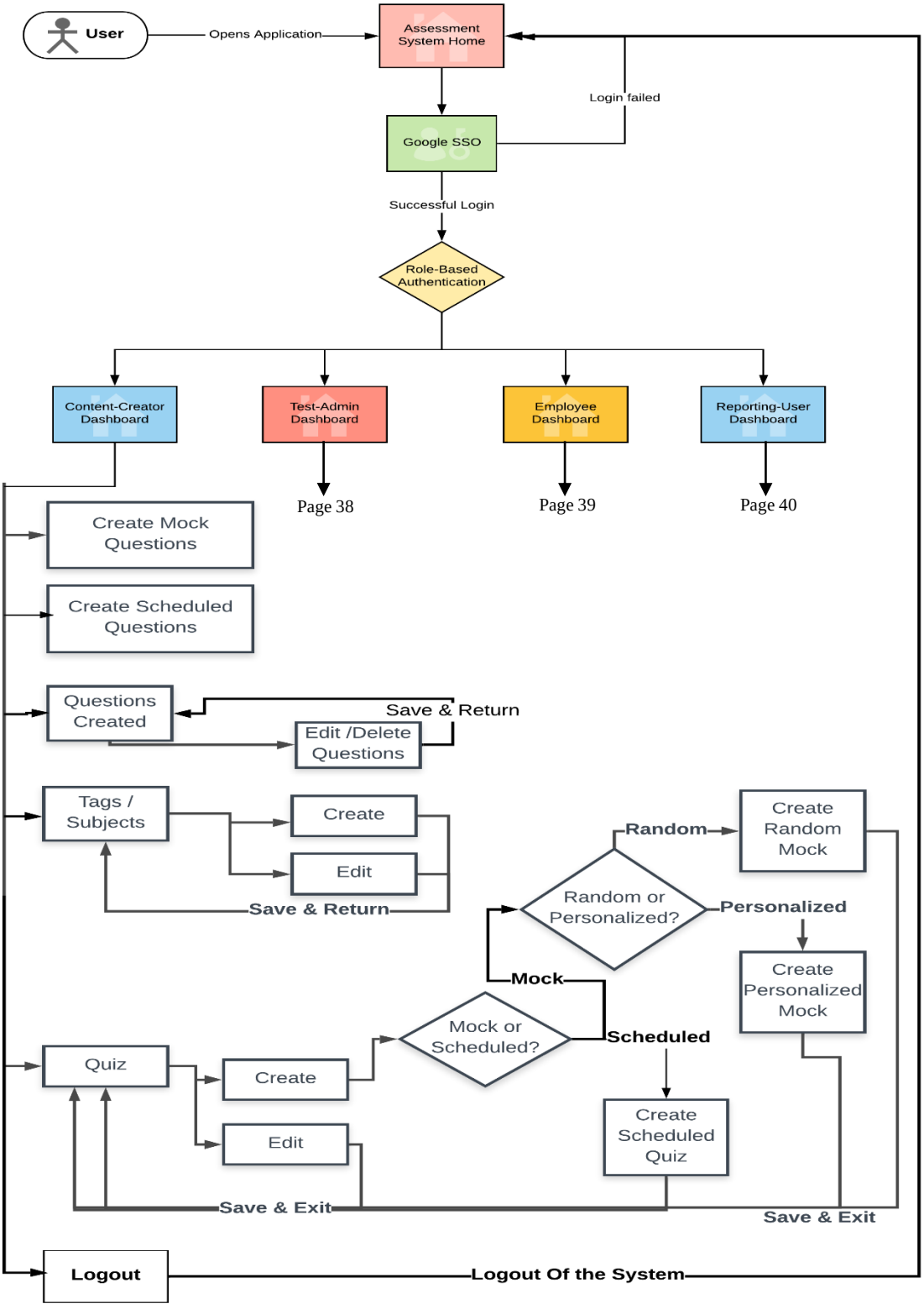

Fig 4.1: Application Flow for Content-Creator

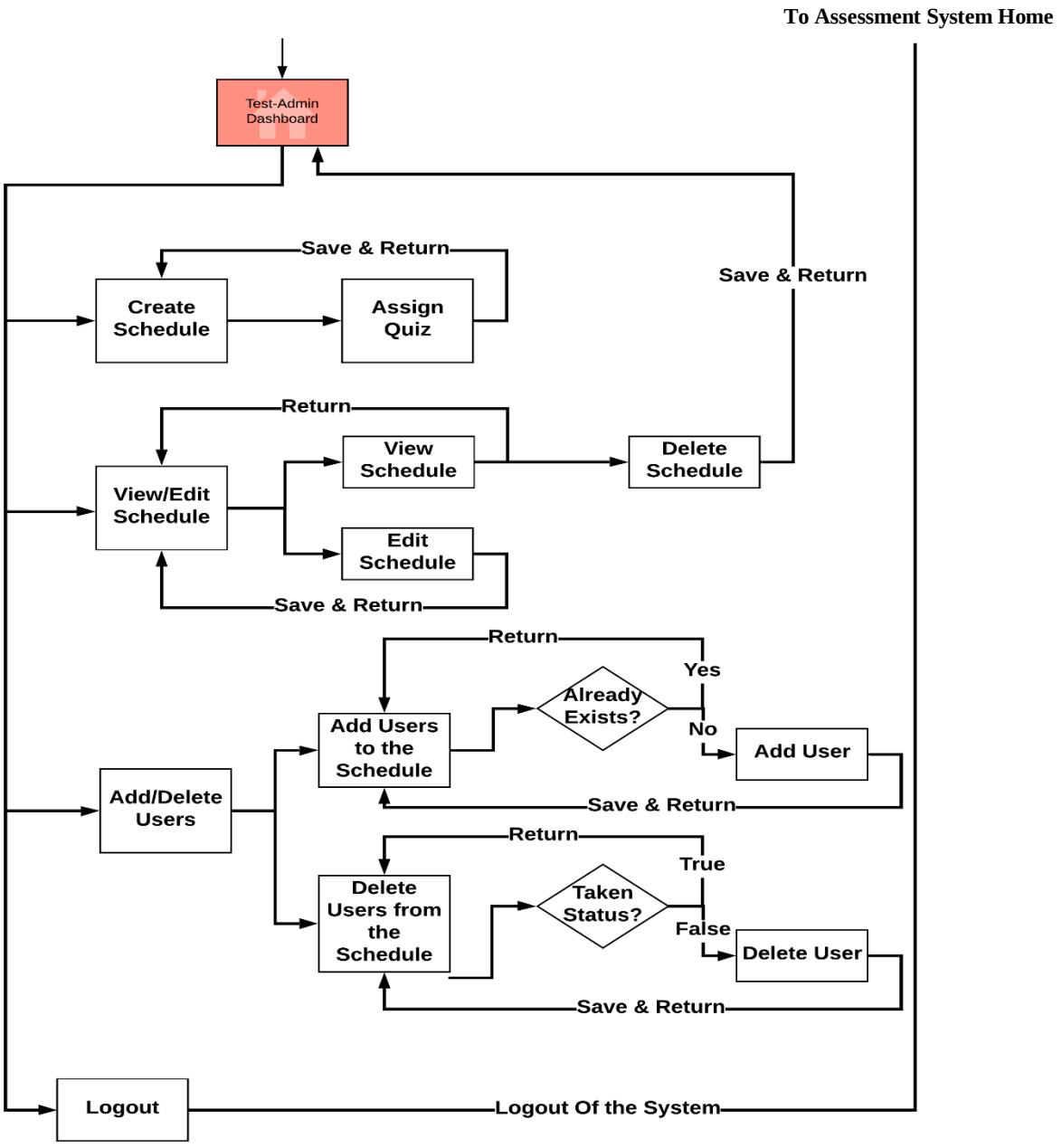

Fig 4.2: Application Flow for Test-Administrator

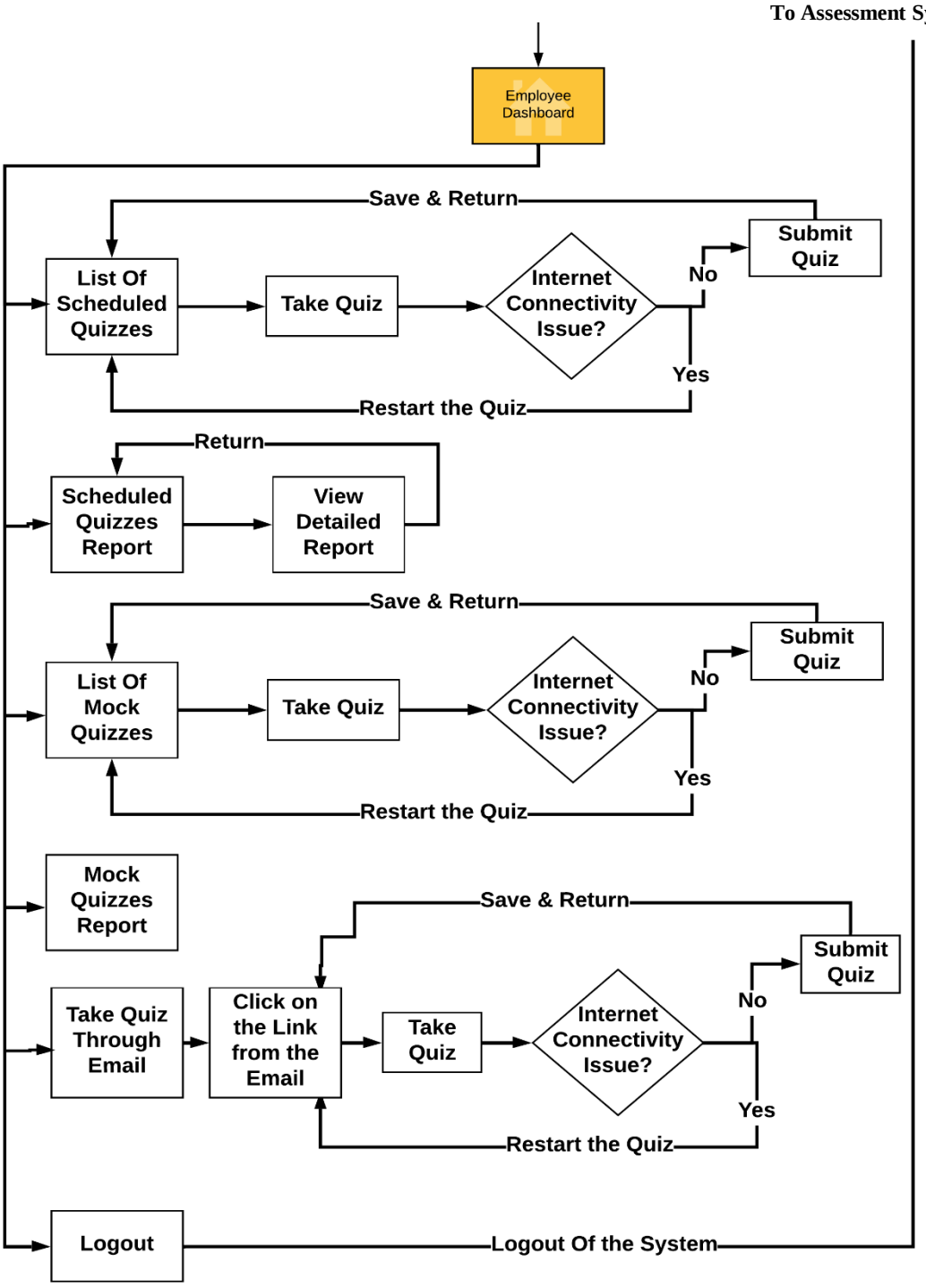

Fig 4.3: Application Flow for Employee

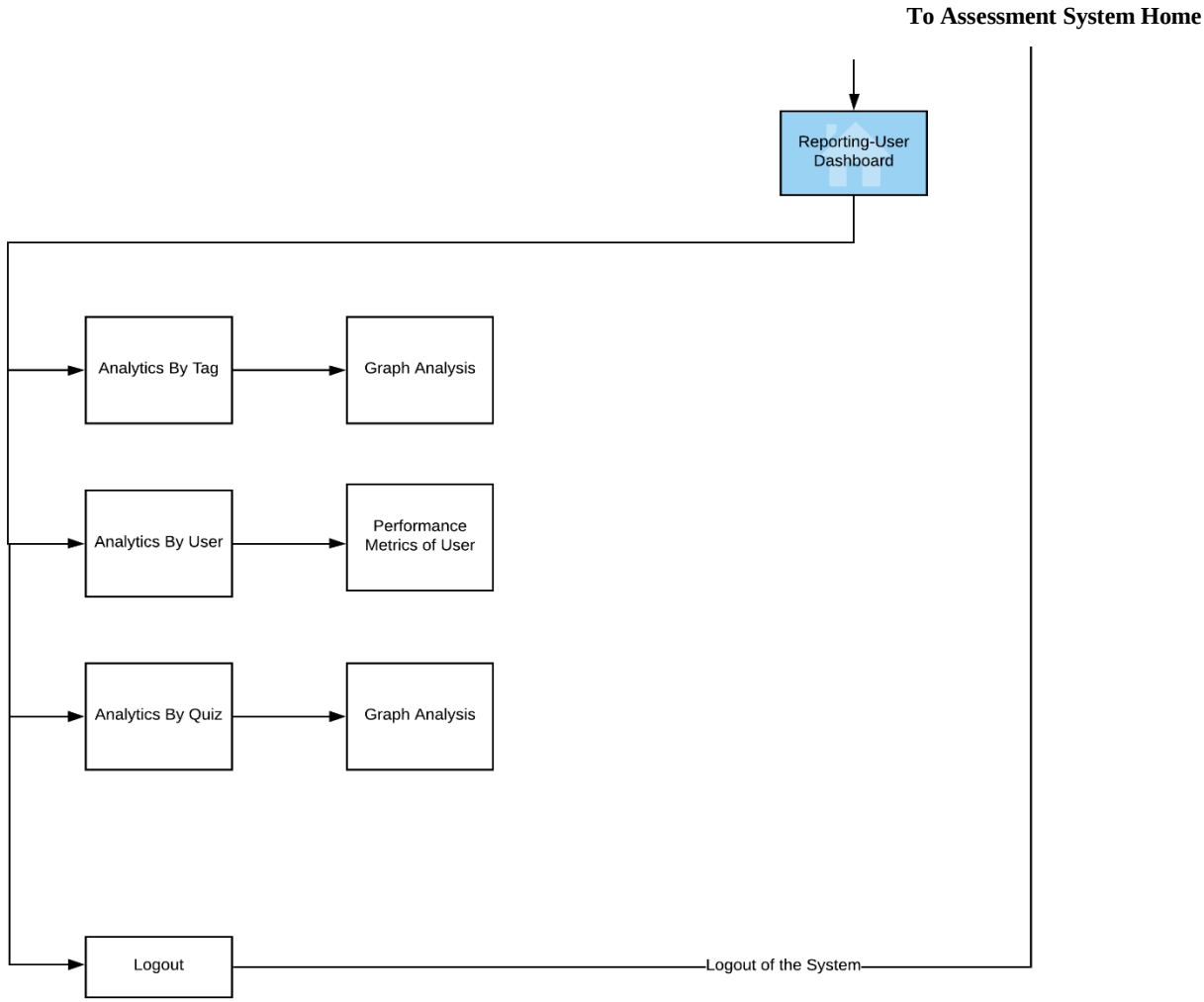

Fig 4.4: Application Flow for Reporting-User

4.4.2 Data Model Diagram :

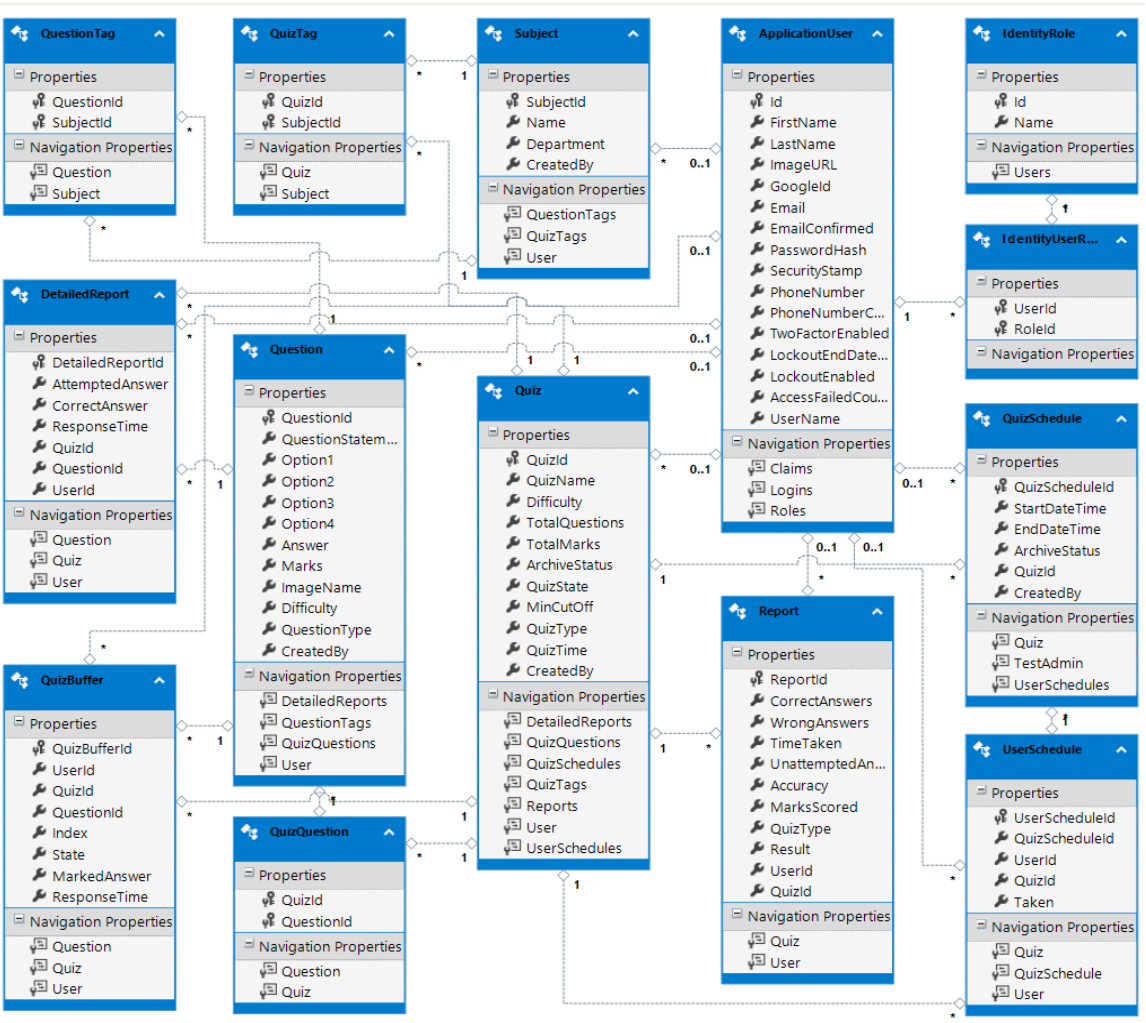

Fig 4.5: Data Model Diagram

4.4.3 Use-Case Diagram :

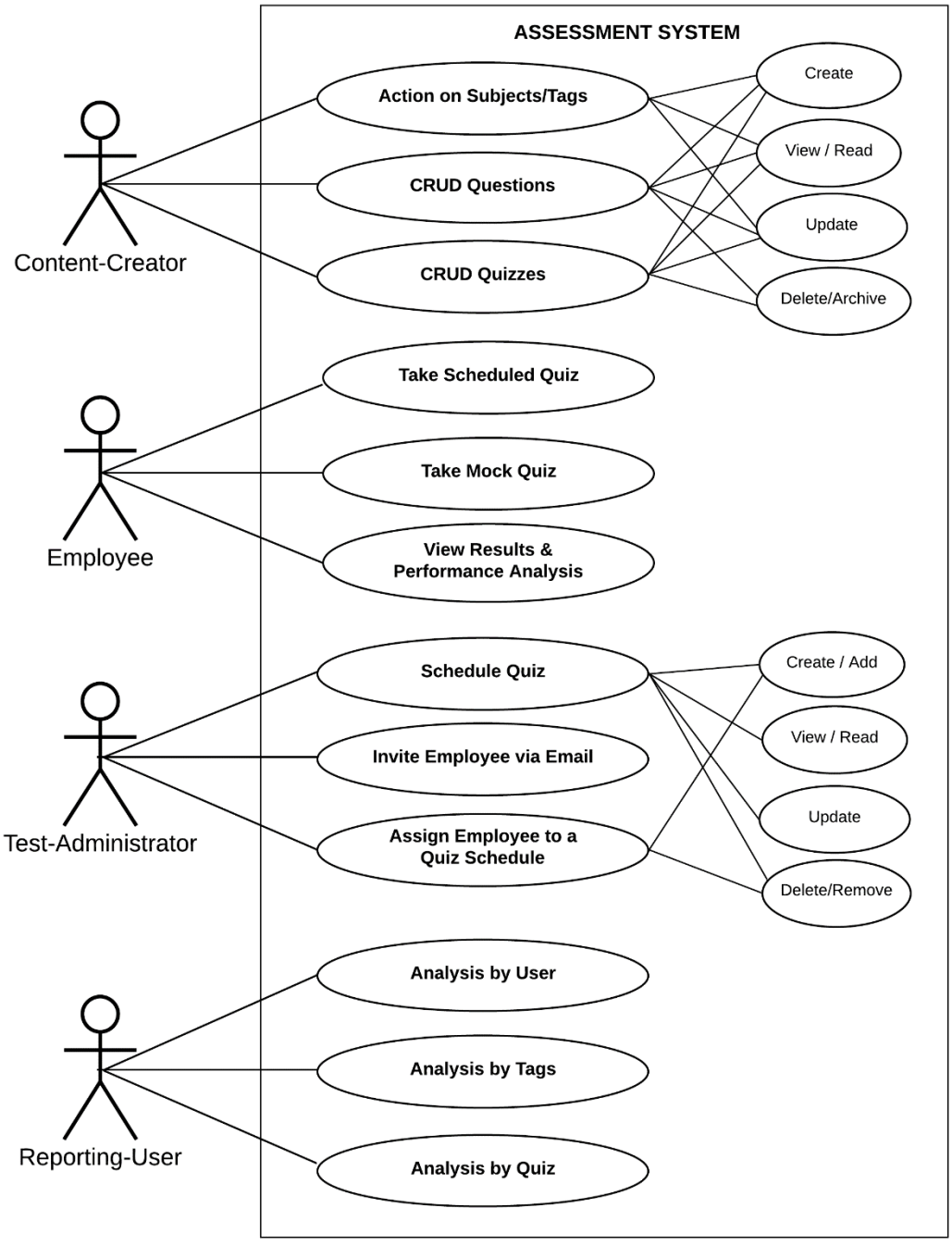

Fig 4.6: Use Case Diagram

4.4.4 Data Flow Diagram :

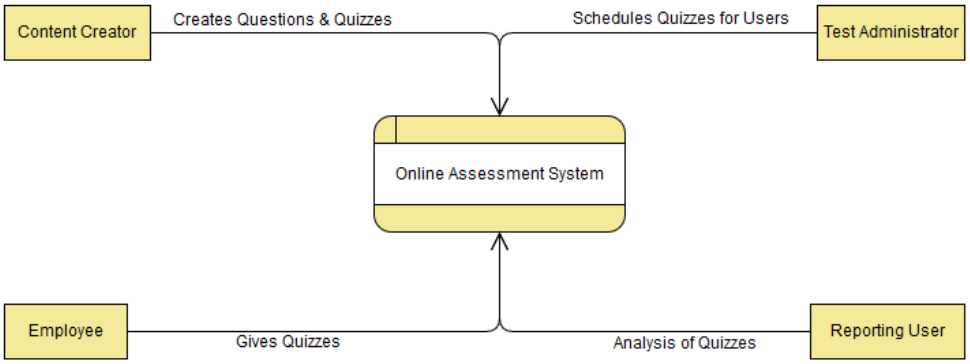

Fig 4.7: Data Flow Diagram - Level 0

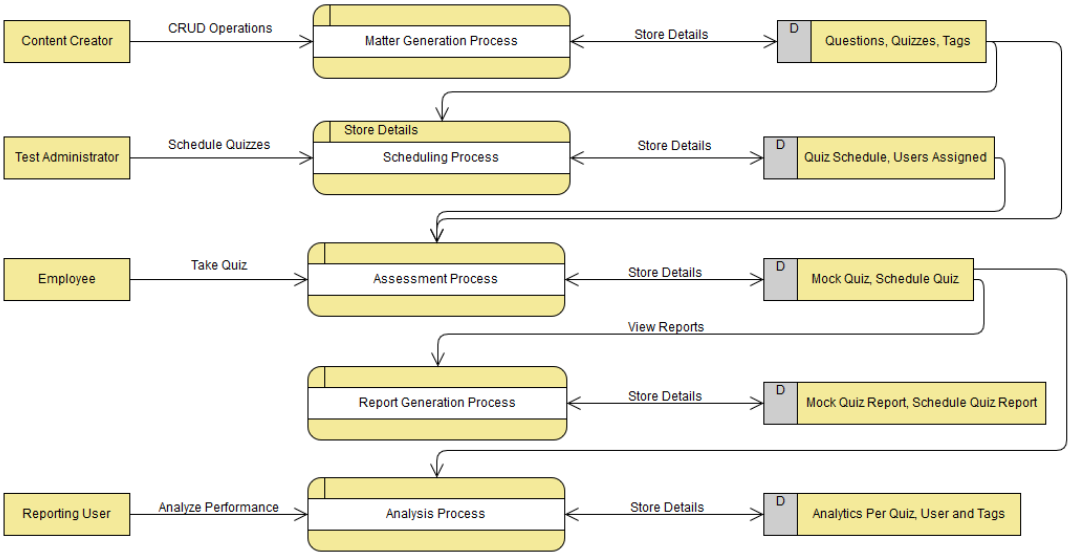

Fig 4.8: Data Flow Diagram – Level 1

4.4.5 Entity-Relationship Diagram.:

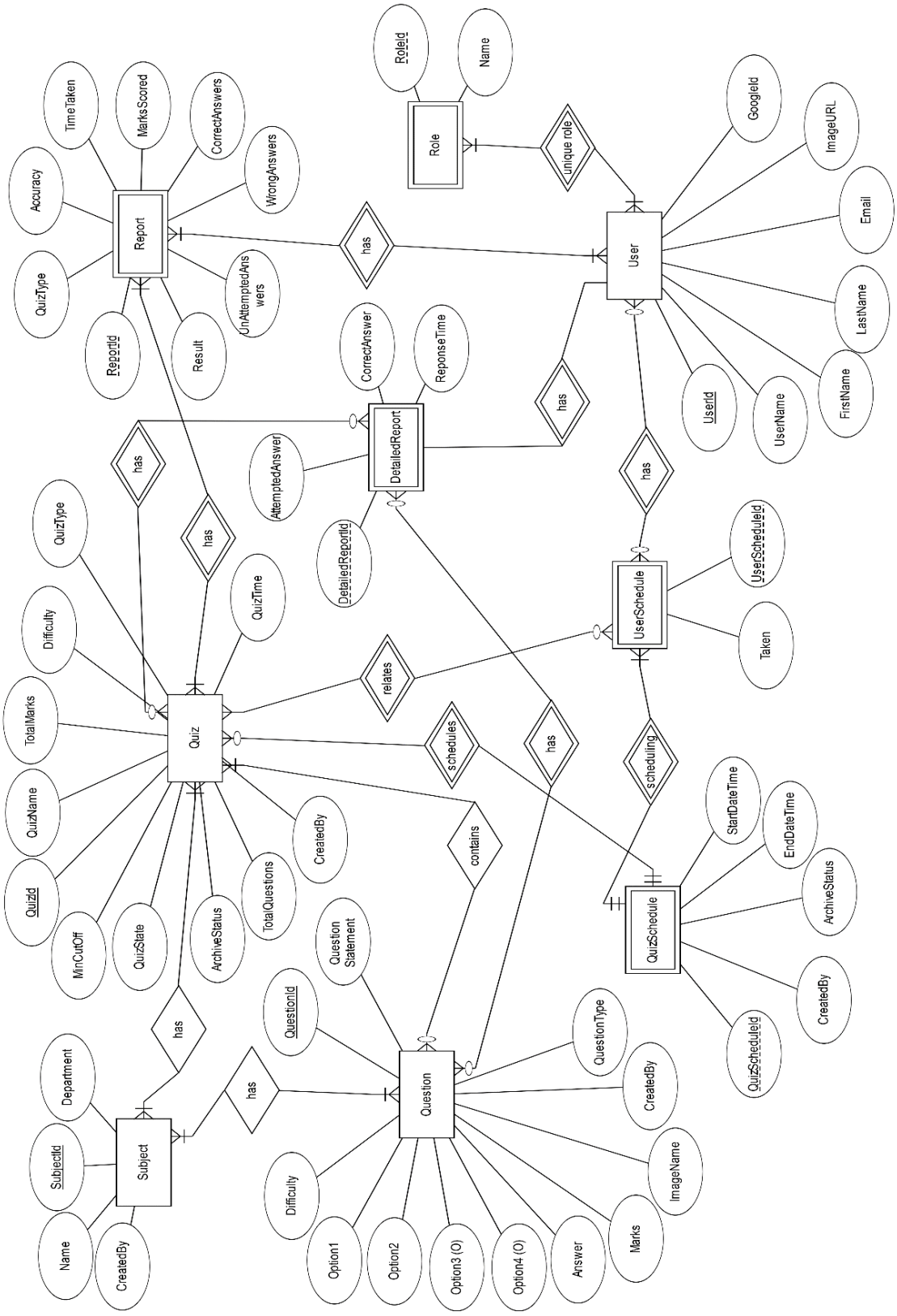

Fig 4.9: Entity-Relationship Diagram

CHAPTER 6: CONCLUSION

6.1 CONCLUSIONS :

In a nutshell, the internship that I had at Nineleaps Technology Solutions Pvt. Ltd. was very interesting, instructive and somehow challenging at times. It gave me lots of benefit and positive changes that enable me to enter the working environment. Through this training I was able to gain new insights and more comprehensive understanding about the real industry working condition and practice. The internship has provided me the opportunities to develop and improve my soft-skills and technical skills. All of this valuable experience knowledge that I have gained were not only acquired through the direct involvement in task given but also through other aspect of the training such as work observation, interaction with the senior employees. As a result of the internship, I am now more confident to enter the corporate world and build my future career.

At the end of our project we will be able to develop an application which will :

- Provide a unified Intra-organization platform
- Make the user aware of the learnings & understandings.
- Allow assessments to be conducted easily on courses, workshops & trainings held.
- Enable organization to track employees' knowledge
- Enables organization in assessing employees' performance
- Provides a platform to create assessment/quiz.
- Store all the questions being created in question bank
- Store detailed reports of every quiz taken by any employee

6.2 SCOPE FOR FUTURE WORK :

1. Can be taken to next level by implementing features that will allow the application to be used for hiring purposes in colleges.
2. Can include adaptive quiz.
3. Can include different Language-Compiler for Program executions.
4. Can implement Machine Learning in order to improve the analysis of performance parameters.
5. Can include a forum to discuss queries and issues related to development with employees within the organization.

REFERENCES

[1] *Stack Overflow.*
https://stackoverflow.com/

[2] *Review: Postman Client Makes Restful API Exploration a breeze*.
https://www.programmableweb.com/news/review-postman-client-makes-restful-api-exploration-breeze/brief/2014/01/27

[3] *ASP.NET - MVC Tutorial for beginners by Kudvenkat.*
https://www.youtube.com/playlist?list=PL6n9fhu94yhVm6S8I2xd6nYz2ZORd7X2v

[4] *ASP.NET WebAPI tutorial for beginners by Kudvenkat.*
https://www.youtube.com/watch?v=0pcM6teVdKk&list=PL6n9fhu94yhW7yoUOGNOfHurUE6bpOO2b

[5] *C# Fundamentals for Absolute Beginners.*
https://mva.microsoft.com/en-us/training-courses/c-fundamentals-for-absolute-beginners-16169?l=Lvld4EQIC_2706218949

[6] *Programming in C# Jump Start.*
https://mva.microsoft.com/en-US/training-courses/programming-in-c-jump-start-14254?l=j0iuozSfB_6900115888

[7] *Introduction to ASP.NET MVC-Intermediate.*
https://mva.microsoft.com/en-us/training-courses/introduction-to-asp-net-mvc-8322?l=nKZwZ8Zy_3504984382

[8] *ASP.NET Web API Overview and Getting Started Videos.*
https://docs.microsoft.com/en-us/aspnet/web-api/videos/getting-started/

[9] *CodAffection – Angular 7 CRUD with Web API.*
https://www.youtube.com/watch?v=jYvkMv7LzCw&list=PLjC4UKOOcfDQjfU3FHqKzTwKwoNYp0ki_

[10] *Implementation of Swagger UI in ASP.NET Web API Restful Service.*
https://www.youtube.com/watch?v=TbL1UdfwWWI

APPENDICES

API : API is the acronym for Application Programming Interface, which is a software intermediary that allows two applications to talk to each other. Each time you use an app like Facebook, send an instant message, or check the weather on your phone, you're using an API. When you use an application on your mobile phone, the application connects to the Internet and sends data to a server. The server then retrieves that data, interprets it, performs the necessary actions and sends it back to your phone. The application then interprets that data and presents you with the information you wanted in a readable way. This is what an API is - all of this happens via API.

To explain this better, let us take a familiar example. Imagine you're sitting at a table in a restaurant with a menu of choices to order from. The kitchen is the part of the "system" that will prepare your order. What is missing is the critical link to communicate your order to the kitchen and deliver your food back to your table. That's where the waiter or API comes in. The waiter is the messenger – or API – that takes your request or order and tells the kitchen – the system – what to do. Then the waiter delivers the response back to you; in this case, it is the food.

JSON : JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language.

JSON is built on two structures:

- A collection of name/value pairs. In various languages, this is realized as an object, record, struct, dictionary, hash table, keyed list, or associative array.
- An ordered list of values. In most languages, this is realized as an array, vector, list, or sequence.

These are universal data structures. Virtually all modern programming languages support them in one form or another. It makes sense that a data format that is interchangeable with programming languages also be based on these structures.

REST : Representational State Transfer, is the architectural style that defines a set of constraints to be used for creating web services. Web services that conform to the REST architectural style, or RESTful web services, provide interoperability between computer systems on the Internet. REST-compliant web services allow the requesting systems to access and manipulate textual representations of web resources by using a uniform and predefined set of stateless operations. Other kinds of web services, such as SOAP web services, expose their own arbitrary sets of operations.

"Web resources" were first defined on the World Wide Web as documents or files identified by their URLs. However, today they have a much more generic and abstract definition that encompasses everything or entity that can be identified, named, addressed, or handled, in any way whatsoever, on the web. In a RESTful web service, requests made to a resource's URI will elicit a response with a payload formatted in either HTML, XML, JSON, or some other format. The response can confirm that some alteration has been made to the stored resource, and the response can provide hypertext links to other related resources or collections of resources. When HTTP is used, as is most common, the operations available are GET, POST, PUT, DELETE, and other predefined CRUD HTTP methods.

By using a stateless protocol and standard operations, REST systems aim for fast performance, reliability, and the ability to grow, by re-using components that can be managed and updated without affecting the system as a whole, even while it is running.

MVC : In object-oriented programming development, model-view-controller (MVC) is the name of a methodology or design pattern for effectively and productively relating the UI to basic data models. The MVC design has been proclaimed by numerous engineers as a helpful pattern for the reuse of object code and a pattern that enables them to significantly reduce the time it takes to develop applications with User Interfaces.

The model-view-controller pattern proposes three main components or objects to be used in software development:

- Model, which represents the underlying, logical structure of data in a software application and the high-level class associated with it. This object model does not contain any information about the UI.
- View, which is a collection of classes representing the elements in the user interface (everything the user can see and respond to on the screen, such as buttons, display boxes, and so forth)
- Controller, which represents the classes connecting the model and the view, and is utilized to communicate between classes in the model and view.

HTTP : Hypertext Transfer Protocol (HTTP) is an application-layer protocol for transmitting hypermedia documents, such as HTML. It was designed for communication between web browsers and web servers, but it can also be used for other purposes. HTTP follows a classical client-server model, with a client opening a connection to make a request, then waiting until it receives a response. HTTP is a stateless protocol, meaning that the server does not keep any data (state) between two requests. There are two versions of HTTP, version HTTP/1.0 and the latest version HTTP/1.1.

A basic HTTP request involves the following steps:

- A connection to the HTTP server is opened.
- A request is sent to the server.
- Some processing is done by the server.
- A response from the server is sent back.
- The connection is closed.

URL : A uniform resource locator (URL) is the address of a resource on the Internet. A URL indicates the location of a resource as well as the protocol used to access it.

A URL contains the following information:

- The protocol used to a access the resource
- The location of the server (whether by IP address or domain name)
- The port number on the server (optional)
- The location of the resource in the directory structure of the server
- A fragment identifier (optional)
- Also known as a Universal Resource Locator (URL) or Web address. A URL is a type of uniform resource identifier (URI). In common practice, the term URI isn't used, or is used synonymously with URL, even though this is technically incorrect.

SMTP : Simple Mail Transfer Protocol is a set of communication guidelines that allow software to transmit an electronic mail over the internet is called Simple Mail Transfer Protocol. It is a program used for sending messages to other computer users based on e-mail addresses.

It provides a mail exchange between users on the same or different computers, and it also supports:

- It can send a single message to one or more recipients.
- Sending message can include text, voice, video or graphics.
- It can also send the messages on networks outside the internet.

The main purpose of SMTP is used to set up communication rules between servers. The servers have a way of identifying themselves and announcing what kind of communication they are trying to perform. They also have a way of handling the errors such as incorrect email address.