PROJECT REPORT

SUBMITTED TO

**DEPARTMENT OF COMPUTER SCIENCE**



**VES COLLEGE OF ARTS, SCIENCE AND COMMERCE**

**SINDHI SOCIETY, CHEMBUR MUMBAI - 400 071.**

**Online Test Management**

**(Website)**

For Partial Fulfilment for Degree of

**Bachelor of Science (Computer Science)**

**2018-2019**

HEAD OF DEPARTMENT PROJECT GUIDE

**Mr. KAMLAKAR BHOPATKAR Mr. KAMLAKAR BHOPATKAR**

**Mrs. MADHAVI VAIDYA**

SUBMITTED BY

**SHRIKANT VIJAYKUMAR NARVEKAR**

**VIVEKANAND EDUCATION SOCIETY’S**

**COLLEGE OF ARTS, SCIENCE AND COMMERCE**

**Sindhi Society, Chembur Mumbai 400071. Phones: 25227514/25227470.**

***NAAC Re-Accredited ‘A’ Grade***

**CERTIFICATE**

This is to certify that **Mr.SHRIKANT VIJAYKUMAR NARVEKAR of T.Y.B.Sc. (Computer Science)** affiliated to University of Mumbai has successfully completed a project work entitled.

**Online Test Management**

**(Website)**

As partial fulfilment of the requirement for the degree of **B.Sc. (Computer Science)** for the academic year 2018-2019.

CO-ORDINATOR OF DEPARTMENT PROJECT GUIDE

**Mr. Kamlakar Bhopatkar Mr. Kamlakar Bhopatkar**

**Ms. Madhavi Vaidya**

Date: \_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_\_\_

**EXAMINER**

Date: \_\_\_\_\_\_\_\_\_\_\_ College Seal:

**ACKNOWLEDGEMENT**

I have great pleasure in presenting this project entitled **“Online Test Management”** and I grab this opportunity to convey my immense regards towards all the people who with their invaluable contributions made this project successful.

It gives me great pleasure in presenting this project report. Its justification will never complete if I don’t express my vote of thanks to our **V.E.S. College** and **Principal Dr. Mrs. J.K. Phadnis**.

I sincerely thank and express my profound gratitude to our Project Guides **Mr. Kamlakar Bhopatkar** and **Ms. Madhavi Vaidya** for timely and prestigious guidance required for the project completion at each phase of the project development. I would also like to thank **Ms.Priyanka Sul**.

I also owe to my **friends** who have been a constant source of help to solve the problems that cropped up during the development of the project, positive criticism, suggestions, constant support, encouragement and the guidance force towards the successful completion of the project.

**INDEX**

|  |  |  |
| --- | --- | --- |
| **SR.NO** | **Topic** | **PAGE NO** |
| **1** | **Desription of system** | **6** |
| **2** | **Limitation of present System** | **7** |
| **3** | **Proposed system and its advantages** | **8** |
| **4** | **Technologies Used** | **11** |
| **5** | **Fact Finding Technique** | **12** |
| **6** | **Basic Scenario and Use case Diagram** | **13** |
| **7** | **ER diagram** | **14** |
| **8** | **List of tables with attributes** | **17** |
| **9** | **Activity Diagram** | **21** |
| **10** | **Class Diagram** | **23** |
| **11** | **Object Diagram** | **26** |
| **12** | **Sequence Diagram** | **29** |
| **13** | **State Chart** | **33** |
| **14** | **Component Diagram** | **37** |
| **15** | **Deployment Diagram** | **38** |
| **16** | **Test Cases** | **40** |
| **17** | **Screen layouts and report layouts** | **48** |
| **18** | **Future enhancements** | **58** |
| **19** | **References and Bibliography** | **59** |

**PRELIMINARY INVESTIGATION**

**DESCRIPTION OF THE CURRENT SYSTEM**

The teacher or course builder creates an account with an exam builder. In such an exam system you can create questions and add them to the exam. You can choose between multiple choice questions or free text questions. The students are provided with a link to the online exam, they sign up and can take the exam. They see the results immediately afterwards.

**LIMITATIONS OF THE CURRENT SYSTEM**

* **Proprietary:** The website will be paid to set a paper.
* **Un-Even System:** Student have to receive a link from the teacher to see the result.
* **Too much paper work:** The current system is manual.So ,most of the time exams are performed using papers and the teacher has to correct each of the papers respectively.

**PROPOSED SYSTEM**

Online test application will provide an Online platform wherein students will attend online objective questions. Grades will be provided as per the number of questions that were answered correctly. This concept will use an admin panel as the backend for loading the questions and also identify whether the student has selected a correct answer or not. Questions will be segregated on the basis of different subjects.

The option will be there to add subjects also. Admin panel will be saving the data and communicating with DB via web service or API.  It will have a flow wherein a subject will be selected and that will load respective questions. Once all questions are answered result will be published there and then.

A student can see grades as per the number of questions that were answered correctly or will calculate the marks for all correct answers and show percentage of marks.

The whole process of conducting the test and checking them and then grading them was a tedious manual process and time consuming too. This whole process can be avoided by adapting to a new online text conducting application that will allow  a easy to approach UI wherein students can save their answers and also being graded much faster and their by publish results soon.

This software will also be secured as it will provide code match system which admin will provide to the student which each of them has to enter to start the exam. Student will also register themselves to appear to this exam. After successful registration admin will approve the registration and then the student will login to the online application system. We can also implement time constraint check for the completion of the exam after which student cannot attempt any question.

User Types: Admin and Student

Modules:

**Admin:**

* Subject: Will have the option to add subjects for which questions will be uploaded.
* Questions: This module will select a subject and then add questions to it along with correct answers for the same. This actually helps in checking whether student gave the correct answer or not. This part is basically admin panel which manages the question and answer that are save in DB and then provide instant results too.
* Grades: Admin will define grades as per percentage of correct answers given by the student. All those calculation are taken care in this module.

**Student:**

* Exam: Once logged in a student will have set of questions which will be objective and he or she has to select the correct answer. It  will be a UI with question set being loaded on the basis of the subject selected.
* View Result: This module will be a UI where in result will be shown in a grid grouping them subject wise. Grade and percentage of marks will also be shown for the student who has appeared for the exam.

**Contact Us:** Will show info related to contact numbers and also search filters.

**ADVANTAGES OF THE PROPOSED SYSTEM**

* **User-friendly**

The UI of the website is easy to understand so that the admin user can easily manipulate his/her question paper.

* **Eco-friendly**

Instead of wasting papers , we can use this website so that students can attempt the test easily

* **Time Efficient**

Teachers don’t have to worry about checking the paper the system will automatically check the papers and give the marks accordingly.

**TECHNOLOGIES USED**

* **Microsoft Visual Studio 2015**

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. Visual Studio supports 36 different programming languages and allows the code editor and debugger to support nearly any programming language, provided a language-specific service exists. Built-in languages include C, C++, C++/CLI, Visual Basic .NET, C#, F#,JavaScript, TypeScript, XML, XSLT, HTML, and CSS.

* **SQL Server Management Studio 2017**

SSMS is an integrated environment for managing any SQL infrastructure, from SQL Server to SQL Database. SSMS provides tools to configure, monitor, and administer instances of SQL. Use SSMS to deploy, monitor, and upgrade the data-tier components used by your applications, as well as build queries and scripts.

Server Management Studio is used to query, design, and manage your databases and data warehouses, wherever they are - on your local computer, or in the cloud.

* **StarUML**

StarUML is a UML tool by MKLab. The software was licensed under a modified version of GNU GPL which is Compatible with UML 2.x standard metamodel and diagrams: Class, Object, Use Case, Component, Deployment, Composite Structure, Sequence, Communication, Statechart, Activity and Profile Diagram. Work with same UX in multiple platforms including macOS, Windows and Linux. Supports many shorthands in Quick Edit to create elements and relationships at once such as sub-classes, supporting interfaces, etc

**FACT FINDING TECHNIQUES**

Fact Finding is the formal process of viewing results, interviews, questionnaires, sampling and other techniques to collect information about the system requirement and preferences. This enables the analyst to be an expert so that:

* They can ensure that the system meets the needs of the organization.
* They can build credibility with the client/user as their domain knowledge increases the credibility of their suggestions.

Fact Finding techniques play an important role in understanding the project at the base level. It helps us to get an idea of how the current system actually works. It also helps us in identifying what the errors in the current system will look like.

The following are the techniques, which I have used to gather information from the client:

* **Review existing documents**:

One effective way of capturing information is documentation. In this, I have reviewed the previous test records they had available and it helped me in understanding the way in which they operate the system.

* **Observation**:

It is a fact finding technique wherein the system analyst either watches a person perform activities or participates in them, to learn more about the system.

My observation has aided me in understanding how an employee in the lab creates a test and how they have worked out the payment procedure.

* **Interview**:

A technique by which we can collect the requirement efficiency, we need conduction of a number of interviews to collect data so that we can understand the working of the system.

I had spoken to the client and discussed in detail about their system to acquire knowledge in regards to the design of the proposed system, so as to fulfil the client requirements.

**BASIC SCENARIO**

* Admin login
* Add Subject
* Add Questions
* Set/Add Grading System/Pattern
* Admin Logout
* Student Login
* Select Subject
* Attempt Test
* View Result
* Student Logout

**USE CASE DIAGRAM**

Use cases are the sequence of actions that form a single unit of work for an actor.A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by "Use case diagrams are the blueprints for your system".They provide the simplified and graphical representation of what the system must actually do.

**Elements of of Use Case Diagram**

**Actors**

An actor in the Unified Modeling Language (UML) "specifies a role played by a user or any other system that interacts with the subject. Actors may represent roles played by human users, external hardware, or other subjects. Actors do not necessarily represent specific physical entities but merely particular facets of some entities that are relevant to the specification of its associated use cases. Actors represent the role of the future users of the system. Actors model the user's perspective of the system. Actors are located outside the system; therefore, in order to depict actors, it is important to define the boundaries between actors and the system. There are three primary types of actors: Users of the system, other systems interacting with our system, and time

**Use Cases**

Use cases are a simple and yet powerful way to express the functional requirements of a system. Use cases describe how users can use the system and what the system can do for users. Therefore, use cases are an important tool to build a consensus between the system's stakeholders and the system's developers. If stakeholders cannot agree on what the system should provide, chances that the project can be successful are very slim. Use cases have improved the communication between stakeholders and the development team and have made the process of gathering system requirements easier and more formal. Use cases provide a visual representation of the conceptual model of the system.

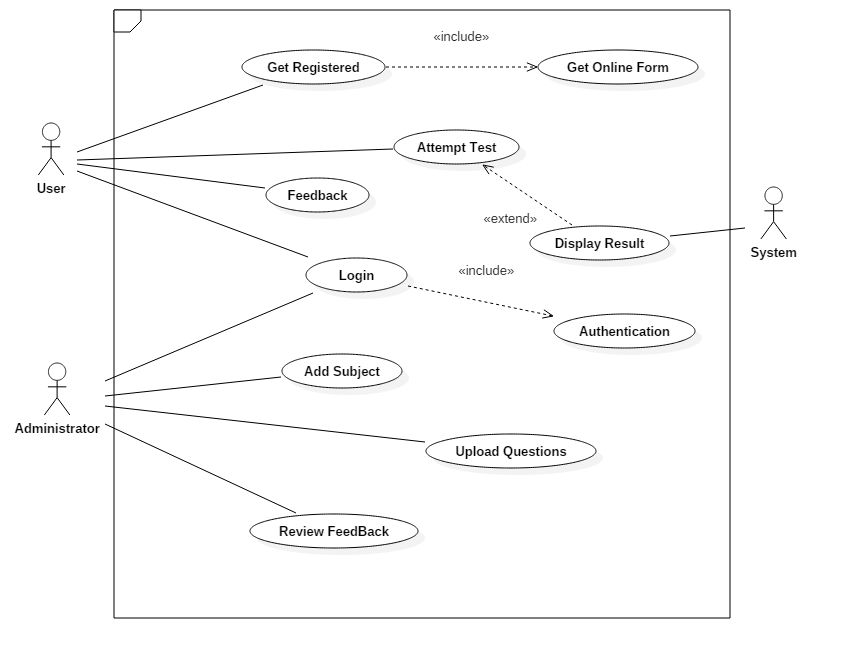
**System Boundary**

A System Boundary element is a non-UML element used to define conceptual boundaries. You can use System Boundaries to help group logically related elements from a visual perspective, not as part of the UML model.A system boundary defines the scope of what a sysytem will be. A system cannot have infinite functionality.So, it follows that use cases also need to have definitive limits defines.The system boundary is hown as a rectangle spanning all the use cases in the system.

**Relationships**

* **Extends:** Extend is a directed relationship that specifies how and when the behavior defined in usually supplementary extending use case can be inserted into the behavior defined in the extended use case. Extend relationship is shown as a dashed line with an open arrowhead directed from the extending use case to the extended (base) use case. The arrow is labeled with the keyword «extend».For Example: Registration use case is complete and meaningful on its own. It could be extended with optional Get Help On Registration use case.
* **Includes:** Use case include is a directed relationship between two use cases which is used to show that behavior of the included use case (the addition) is inserted into the behavior of the including (the base) use case. Include relationship between use cases is shown by a dashed arrow with an open arrowhead from the including (base) use case to the included (common part) use case. The arrow is labeled with the keyword «include». The include relationship could be used:
* to simplify large use case by splitting it into several use cases,
* to extract common parts of the behaviors of two or more use cases.

**Use Case Diagram**

****

**ENTITY RELATIONSHIP DIAGRAM**

In software engineering, an entity–relationship model (ER model) is a data model for describing a database in an abstract way. This article refers to the techniques proposed in Peter Chen's 1976 paper. However, variants of the idea existed previously and have been devised subsequently such as super type and subtype data entities and commonality relationships.

**Entity:**

An entity may be defined as a thing which is recognized as being capable of an independent existence and which can be uniquely identified. An entity is an abstraction from the complexities of a domain. When we speak of an entity, we normally speak of some aspect of the real world which can be distinguished from other aspects of the real world. An entity may be a physical object such as a house or a car, an event such as a house sale or a car service, or a concept such as a customer transaction or order. Although the term entity is the one most commonly used, following Chen we should really distinguish between an entity and an entity-type. An entity-type is a category. An entity, strictly speaking, is an instance of a given entity-type. There are usually many instances of an entity-type. Because the term entity-type is somewhat cumbersome, most people tend to use the term entity as a synonym for this term.

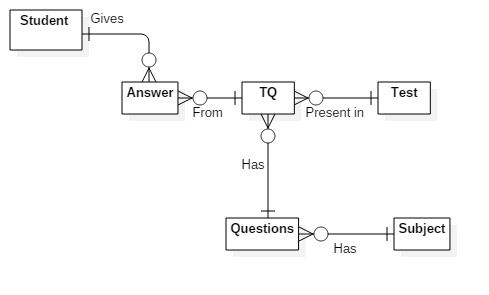
**Relationship:**

A relationship captures how entities are related to one another. Relationships can be thought of as verbs, linking two or more nouns. Examples: an owns relationship between a company and a computer, a supervises relationship between an employee and a department, a performs relationship between an artist and a song, a proved relationship between a mathematician and a theorem.

**Cardinality:**

The cardinality defines the relationship between the entities in terms of numbers. An entity may be optional: for example, a sales rep could have no customers or could have one or many customers; or mandatory: for example, there must be at least one product listed in an order. There are several different types of cardinality notation; crow's foot notation, used here, is a common one. In crow's foot notation, a single bar indicates one, a double bar indicates one and only one (for example, a single instance of a product can only be stored in one warehouse), a circle indicates zero, and a crow's foot indicates many. The three main cardinal relationships are: one-to-one, expressed as 1:1; one-to-many, expressed as 1: M; and many-to-many, expressed as M: M. every entity (unless it is a weak entity) must have a minimal set of uniquely identifying attributes, which is called the entity's primary key.

Entity–relationship diagrams don't show single entities or single instances of relations. Rather, they show entity sets and relationship sets. Example: a particular song is an entity. The collection of all songs in a database is an entity set.

****

**LIST OF TABLES**

**student\_details**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column\_Name** | **Data\_Type** | **Size** | **Constraints** |
| student\_id | smallint | Max | Primary key |
| student\_name | varchar | 50 | Not Null |
| username | varchar | 20 | Not Null |
| password | varchar | 25 | Not Null |
| email | varchar | 40 | Not Null |
| feedback | varchar | 150 | Null |

**subject\_details**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column\_Name** | **Data\_Type** | **Size** | **Constraints** |
| subject\_code | smallint | Max | Primary key |
| subject\_name | varchar | 20 | Not Null |
| description | varchar | 150 | Null |

**answer\_keys**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column\_Name** | **Data\_Type** | **Size** | **Constraints** |
| answer\_id | int | Max | Primary key |
| question\_id | smallint | Max | Foreign key |
| answer\_option | varchar | 50 | Not Null |
| student\_id | smallint | Max | Foreign key |
| marks\_obtained | int | Max | Not Null |

**questionnaire**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column\_Name** | **Data\_Type** | **Size** | **Constraints** |
| question\_id | smallint | Max | Primary key |
| questions | varchar | 25 | Not Null |
| subject\_code | smallint | Max | Foreign key |
| option1 | varchar | 100 | Not Null |
| option2 | varchar | 100 | Not Null |
| option3 | varchar | 100 | Not Null |
| option4 | varchar | 100 | Not Null |
| correct\_option | varchar | 7 | Not Null |

**test**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column\_Name** | **Data\_Type** | **Size** | **Constraints** |
| test\_id | smallint | Max | Primary key |
| test\_date | datetime | Max | Not Null |
| min\_questions | smallint | Max | Not Null |
| max\_marks | smallint | Max | Not Null |
| student\_id | smallint | Max | Foreign key |

**tq**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column\_Name** | **Data\_Type** | **Size** | **Constraints** |
| tq\_id | smallint | Max | Primary key |
| question\_id | datetime | Max | Foreign key |
| test\_id | smallint | Max | Foreign key |

**ACTIVITY DIAGRAM**

Activity diagrams are graphical representations of workflows of stepwise activities and action with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams are intended to model both computational and organizational processes (i.e. workflows). Activity diagrams show the overall flow of control. Activity diagrams are constructed from a limited number of shapes, connected with arrows.

The most important shape types:

• rounded rectangles represent actions;

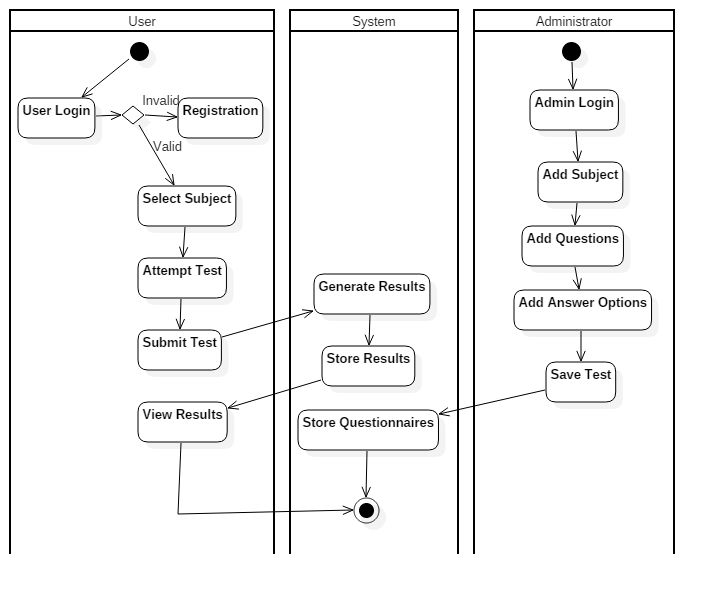
• diamonds represent decisions;

• bars represent the start (split) or end (join) of concurrent activities;

• a black circle represents the start (initial state) of the workflow;

• An encircled black circle represents the end (final state).

Arrows run from the start towards the end and represent the order in which activities happen. Hence they can be regarded as a form of flowchart. Activity diagram is another important diagram in UML to describe dynamic aspects of the system. Activity diagram is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent. Activity diagrams deals with all type of flow control by using different elements like fork, join etc. The basic purposes of activity diagrams are similar to other four diagrams. It captures the dynamic behaviour of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another. Activity is a particular operation of the system. Activity diagrams are not only used for visualizing dynamic nature of a system but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in activity diagram is the message part. It does not show any message flow from one activity to another. Activity diagram is some time considered as the flow chart. Although the diagrams looks like a flow chart but it is not. It shows different flow like parallel, branched, concurrent and single.

****

**CLASS DIAGRAM**

A class diagram consists of a group of classes and interfaces reflecting important entities of the system, and the relationships between these classes and interfaces. Classes in a class diagram are interconnected in a hierarchical fashion. Class Diagrams describe the static structure of a system, or how it is structured, rather than how it behaves. A class diagram is represented in 3 parts:

**ELEMENTS OF CLASS DIAGRAM:**

**Class**: A class represents an entity of a given system that provides certain functionality of a given entity. The properties of a class are called as attributes.

A class is represented by a rectangle, which is divided into compartments. These contain

* The upper part holds the name of the class.
* The middle part contains the attributes of the class.
* The bottom part gives the methods or operations the class can take or undertake.

**Association**: Association represents the static relationship shared among the objects of two classes. Associations are normally represented as a line, with each end connected to a class.

**Aggregation**: Aggregation is a variant of the “has a” or association relationship; aggregation is more specific than association. It is an association that represents a part whole or part of relationship.

**Composition**: Composition is a stronger variant of the “owns a” or association relationship; composition is more specific than aggregation. It is represented with a solid diamond shape.

**Generalization**: The Generalization relationship indicates that one of the two related classes (the subtype) is considered to be a specialized form of the other and super type is considered as Generalization of subtype.

**Multiplicity**: Multiplicity notations indicate the number of instances of one class linked to one instance of the other class. They are placed near the end of an association.

0..1 - zero or one instance

1 - exactly one instance

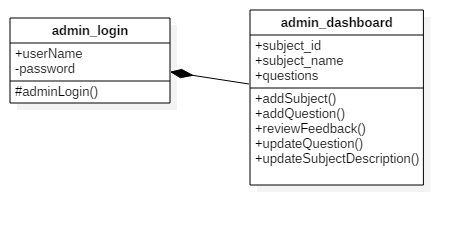
0.. \* or \* - zero or more instances

1.. \* - one or many instances (at least one)

**Student class diagram**

****

**Admin class diagram**

****

**OBJECT DIAGRAM**

**Object diagrams** are derived from class diagrams so object diagrams are dependent upon class diagrams.

Object diagrams represent an instance of a class diagram. The basic concepts are similar for class diagrams and object diagrams. Object diagrams also represent the static view of a system but this static view is a snapshot of the system at a particular moment.

Object diagrams are used to render a set of objects and their relationships as an instance.

The difference is that a class diagram represents an abstract model consists of classes and their relationships. But an object diagram represents an instance at a particular moment which is concrete in nature.

It means the object diagram is more close to the actual system behavior. The purpose is to capture the static view of a system at a particular moment.

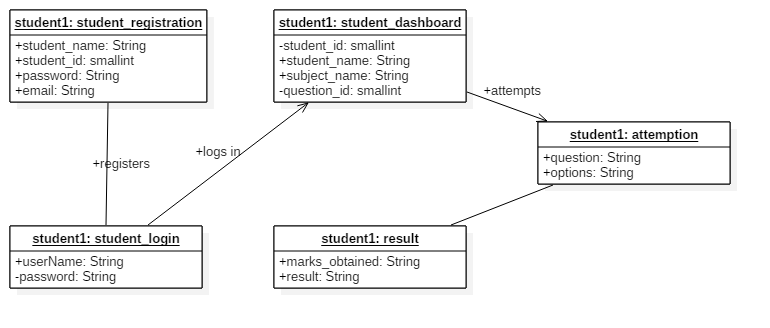
So the purpose of the object diagram can be summarized as:

* Forward and reverse engineering.
* Object relationships of a system.
* Static view of an interaction.
* Understand object behavior and their relationship from practical perspective.

**ELEMENTS OF OBJECT DIAGRAM:**

* **Objects**: Objects represent particular entities. These are instances of classes.
* **Links**: Links represent particular relationships between objects. These are instances of associations. A link is shown as a solid line.

**Student object diagram**

****

**Admin object diagram**

****

**SEQUENCE DIAGRAM**

Sequence Diagram represents an interaction among objects in the form of messages ordered in a sequence by time. In a sequence diagram, you arrange objects across the x-axis. We can place the object that starts an interaction to the extreme left. The objects that come later in the message sequence are placed to the right of the interaction-initiating object. The messages sent and received by the objects in an interaction are placed along the y-axis in an increasing order of time.

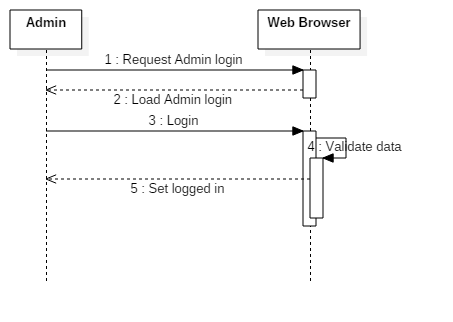
An object can be created or destroyed when an operation is performed. You can use sequence diagram to depict the creation and destruction of an object. In addition, sequence diagram depict the change in the focus of control of an object in the form of a rectangular bar. The length of the rectangular bar depicts the duration of time for which an object interacts with another object.

UML sequence diagrams are used to represent or model the flow of messages, events and actions between the objects or components of a system. Time is represented in the vertical direction showing the sequence of interactions of the header elements, which are displayed horizontally at the top of the diagram.

Sequence Diagrams are used primarily to design, document and validate the architecture, interfaces and logic of the system by describing the sequence of actions that need to be performed to complete a task or scenario. UML sequence diagrams are useful design tools because they provide a dynamic view of the system behaviour which can be difficult to extract from static diagrams or specifications.

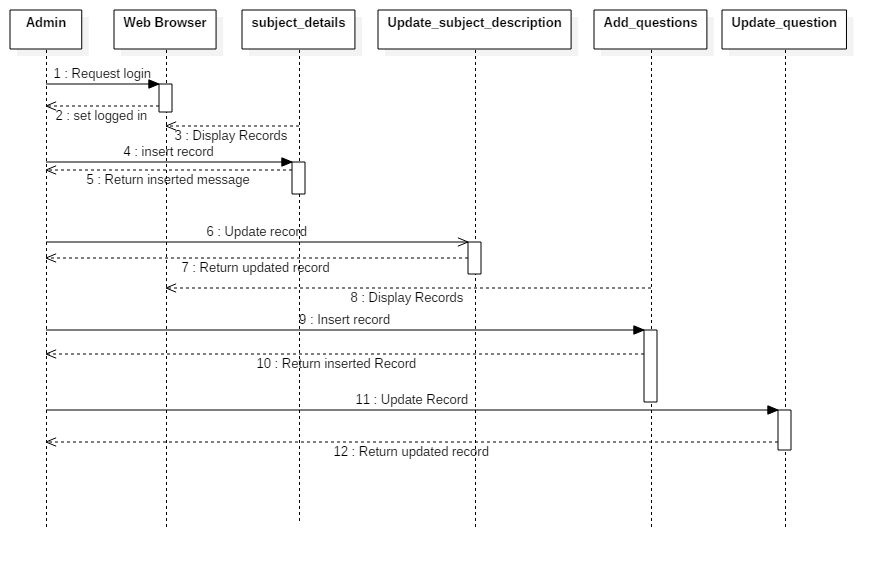
Although UML sequence diagrams are typically used to describe object-oriented software systems, they are also extremely useful as system engineering tools to design system architectures, in business process engineering as process flow diagrams, as message sequence charts and call flows for telecom/wireless system design, and for protocol stack design and analysis.

**Sequence diagram for Admin Login**

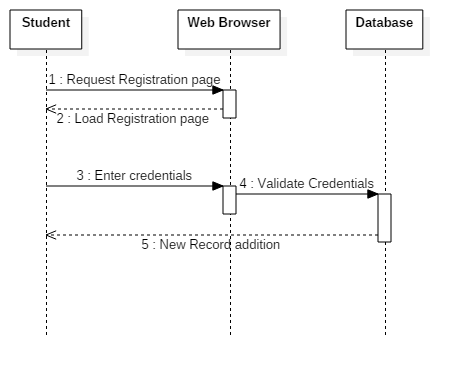


**Sequence diagram for Student Login**

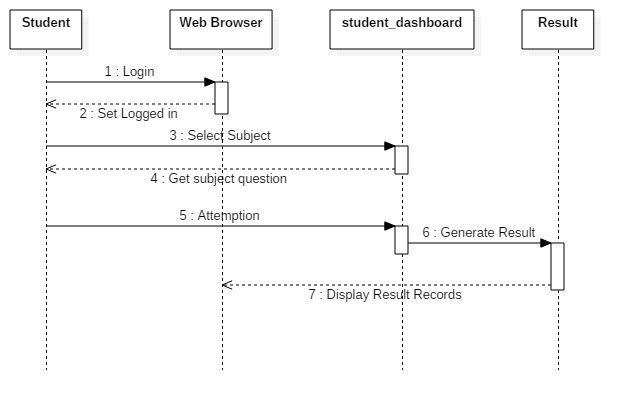
****

**Sequence Diagram for Admin Details**

**Sequence Diagram for Student Registration**



**Sequence Diagram for Student Details**

****

**STATE DIAGRAM**

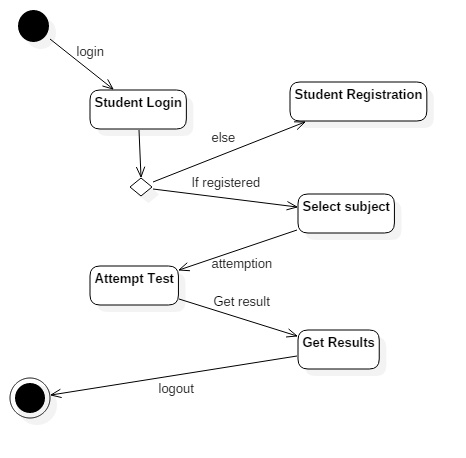
A state chart diagram is a behavior which specifies the sequence of states an object visits during its lifetime in response to events, together with its responses to those events.

**ELEMENTS OF A STATE DIAGRAM:**

* **Initial State**: This shows the starting point or a first activity of the flow, denoted by a solid circle. This is also called as a “pseudo state,” where the state has no variables describing it further and no activities.
* **State**: Represents the state of object at an instant of time. In a state diagram, there will be multiple of such symbols, one for each state of the Object we are discussing. Denoted by a rectangle with rounded corners and compartments (such as a class with rounded corners to denote and Object). We will describe this symbol in detail a little later.
* **Event and Action**: A trigger that causes a transition to occur is called as an event or action. Every transition need not occur due to the occurrence of an event or action directly related to the state that transitioned from one state to another. As described above, an event/action is written above a transition that it causes.
* **Final State**: The end of the state diagram is shown by a bull’s eye symbol, also called a final state. A final state is another example of a pseudo state because it does not have any variable or action described.

**Admin State chart diagram**

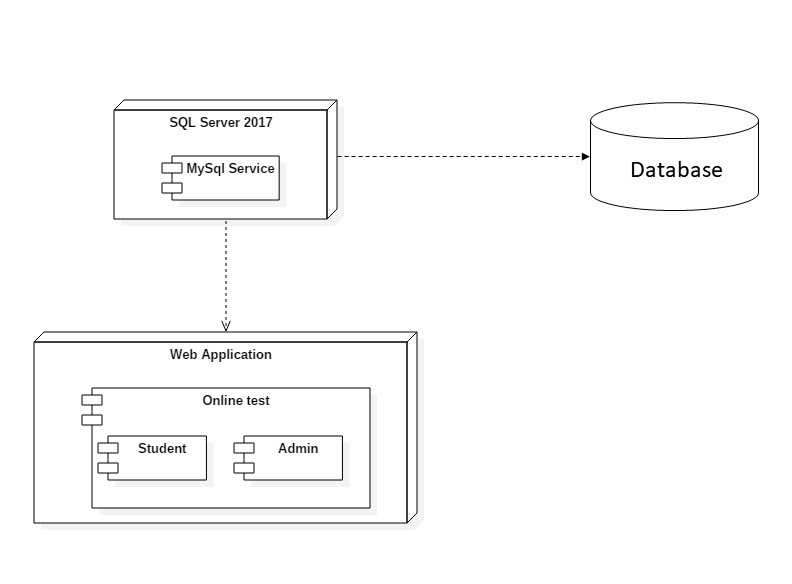
****

**Student state chart diagram**

**SYSTEM DESIGN**

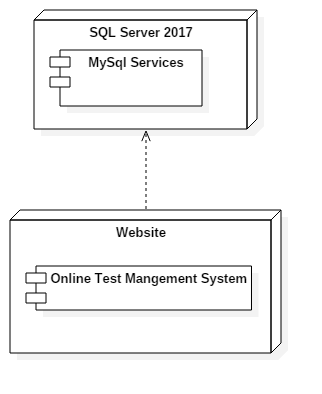
**COMPONENT DIAGRAM**

* Component Diagram provides the realization of a set of interfaces and form the executable parts of a software system. Examples of components can be an executable file, a COM+ component, or an Enterprise Java Bean (EJB) or a .NET component. In UML we represent a component as a rectangle with tabs. Each Component has a name that identifies it.
* A Component is a standalone piece of software because it performs a complete function.
* Relationship among Components, Classes and Interfaces. Component realizes a set of interfaces in which each interface specifies the function offered by a class. A class represents a real world entity and contains the code for implementing behavior that is specific to the entity that the class represents. Thus various classes contained in a component and their relationship using a component diagram.

****

**DEPLOYMENT DIAGRAM**

* The Deployment diagram is drawn to visualize the hardware on which the software components need to be deployed. A deployment diagram is drawn immediately after identifying classes, interfaces and their relationships.
* The various computer systems or processing devices on which components are deployed are called nodes. We may have all the components on one node or on different nodes. In UML a node is represented as a 3-D rectangular box containing the components that execute within the node. A node may be located anywhere in network. Distinction among nodes is achieved by assigning each node a name and classifying the various nods into types depending on the type of components they execute. Nodes are associated with each other by a connection, which represents a communication channel. It depicts how the nodes are connected with each other.
* The dependency on node on components is depicted using dashed lines. This means that a node uses the services of the components that are executing on another node.

****

**SYSTEM CODING**

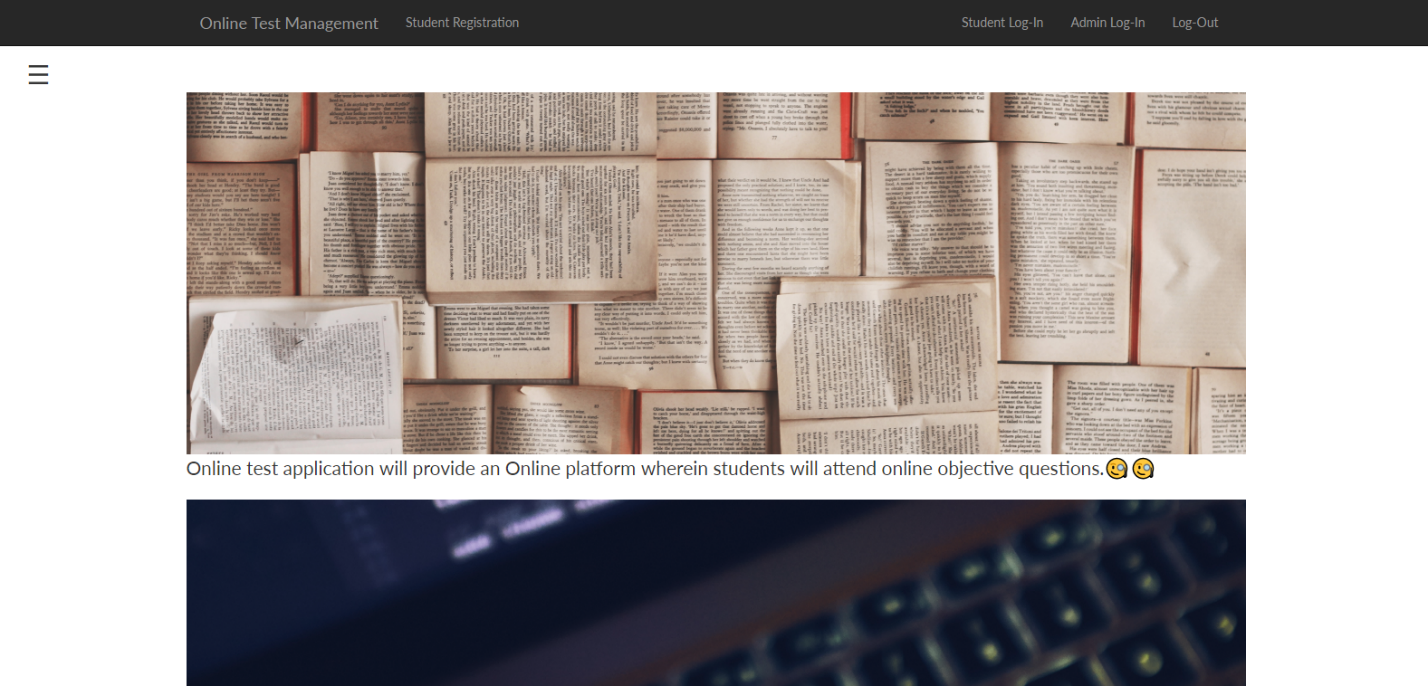
**TEST CASES, TEST DATA AND RESULTS**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Module/ Sub Module Name** | **Form Name** | **Test Condition** | **Pre-requisite** | **Steps/ Procedures** | **Input Test Data** | **Expected Output** | **Actual Output** | **Pass/ Fail** |
| T01 | Admin Login | admin\_login | To Check login page with valid input | Admin login page should be open | Click on login button with blank username and password | Username  Password | Username Field required  Password Field required | Username Field required  Password Field required | Pass |
| T02 | Admin Login | admin\_login |  | Admin login page should be open | Click on login button after entering only username | Username  Password | Password Field required | Password Field required | Pass |
| T03 | Admin Login | admin\_login |  | Admin login page should be open | Click on login button after entering only password | Username  Password | Username Field required | Username Field required | Pass |
| T04 | Admin Login | admin\_login |  | Admin login page should be open | Correct Username and wrong Password | Username  Password | Invalid Password | Invalid Password | Pass |
| T05 | Admin Login | admin\_login |  | Admin login page should be open | Correct Password and wrong Username | Username  Password | Invalid Username | Invalid Username | Pass |
| T06 | Admin Login | admin\_login |  | Admin login page should be open | Correct Password and Username | Username  Password | Opens Admin Dashboard | Opens Admin Dashboard | Pass |
| T07 | Admin Dashboard | admin\_dashboard1 | To check valdidations of each page | Admin dashboard page should be open | Click on Add Subjects button | Add Subject button | Opens subject\_details page | Opens subject\_details page | Pass |
| T08 | Subject Details | subject\_details | To check valdidations | Subject Details page should be open | Click on Save Details button with blank subject code, subject name and subject description | Subject Code  Subject Name  Subject Description | subject code field is required  Subject Name field is required. | subject code field is required  Subject Name field is required. | Pass |
| T09 | Subject Details | subject\_details |  | Subject Details page should be open | Click on Save Details button with blank subject name and subject description | Subject Code  Subject Name  Subject Description | Subject Name field is required. | Subject Name field is required. | Pass |
| T10 | Subject Details | subject\_details |  | Subject Details page should be open | Click on Save Details button with blank subject description | Subject Code  Subject Name  Subject Description | Saves details and Opens Admin dashboard page | Saves details and Opens Admin dashboard page | Pass |
| T11 | Update Subject Description | update\_sub\_description | To check valdidations | update subject description page should be open | Click on Update Details button with blank subject code and subject description | Subject Code  Subject Description | subject code field is required  Subject Description field is required | subject code field is required  Subject Description field is required | Pass |
| T12 | Update Subject Description | update\_sub\_description |  | update subject description page should be open | Click on Update Details button with subject code and subject description | Subject Code  Subject Description | Updates subject description and opens Admin Dashboard page | Updates subject description and opens Admin Dashboard page | Pass |
| T13 | Add Questions | questionnaire | To check valdidations | questionnaire page should be open | Click on Save details button with  blank subject code,question,option1,option2,option3,option4 and correct option | subject code  questionoption1  option2  option3  option4  correct option | subject code field is required  question field is required option1 field is required  option2 field is required  option3 field is required  option4 field is required  correct option field is required | subject code field is required  question field is required option1 field is required  option2 field is required  option3 field is required  option4 field is required  correct option field is required | Pass |
| T14 | Add Questions | questionnaire |  | questionnaire page should be open | Click on Save details button with  subject code,question,option1,option2,option3,option4 and correct option | subject code  questionoption1  option2  option3  option4  correct option | Saves data and opens questionnaire page and subject code read only enabled | Saves data and opens questionnaire page and subject code read only enabled | Pass |
| T15 | Add Questions | questionnaire |  | questionnaire page should be open | Click on Back to Dasboard Button | subject code  questionoption1  option2  option3  option4  correct option | Opens Admin Dashboard | Opens Admin Dashboard | Pass |
| T16 | Update Questions | update\_question | To check valdidations | update question page should be open | Click on Update details button with  blank question idquestion,option1,option2,option3,option4 and correct option | question id  questionoption1  option2  option3  option4  correct option | question id field is required  question field is required option1 field is required  option2 field is required  option3 field is required  option4 field is required  correct option field is required | question id field is required  question field is required option1 field is required  option2 field is required  option3 field is required  option4 field is required  correct option field is required | Pass |
| T17 | Update Questions | update\_question |  | update question page should be open | Click on Update details button with  question idquestion,option1,option2,option3,option4 and correct option | question id  questionoption1  option2  option3  option4  correct option | Update data and opens update question page | Update data and opens update question page | Pass |
| T18 | Update Questions | update\_question |  | update question page should be open | Click on back to dashboard button | question id  questionoption1  option2  option3  option4  correct option | Opens Admin Dashboard Page | Opens Admin Dashboard Page | Pass |
| T19 | Admin Dashboard | admin\_dashboard1 | To check logout validation | admin\_dashboard1 page should be open | Click on logut button in nav-bar | username  password | Opens Home page and log out | Opens Home page and log out | Pass |
| T20 | Admin Dashboard | admin\_dashboard1 | To check login validation | admin\_dashboard1 page should be open | Click on Admin Log-In button in nav-bar | username  password | Opens Admin Log-in page with alert box | Opens Admin Log-in page with alert box | Pass |
| T21 | Student Login | student\_login | To Check login page with valid input | Student login page should be open | Click on login button with blank username and password | Username  Password | Username Field required  Password Field required | Username Field required  Password Field required | Pass |
| T22 | Student Login | student\_login |  | Student login page should be open | Click on login button after entering only username | Username  Password | Password Field required | Password Field required | Pass |
| T23 | Student Login | student\_login |  | Student login page should be open | Click on login button after entering only password | Username  Password | Username Field required | Username Field required | Pass |
| T24 | Student Login | student\_login |  | Student login page should be open | Correct Username and wrong Password | Username  Password | Invalid Credentials | Invalid Credentials | Pass |
| T25 | Student Login | student\_login |  | Student login page should be open | Correct Password and wrong Username | Username  Password | Invalid Credentials | Invalid Credentials | Pass |
| T26 | Student Login | student\_login |  | Student login page should be open | Correct Password and Username | Username  Password | Opens student dashboard | Opens student dashboard | Pass |
| T27 | Student Dashboard | student-dashboard | To check valdidations | student-dashboard should be open | Select one value from the dropdown list | dropdown list | opens start\_examination page | opens start\_examination page | Pass |
| T28 | Start Examination | start\_examination | To check valdidations | Start Examination should be open | Click on image | Image button | Opens attemption page | Opens attemption page | Pass |
| T29 | Attemption | attemption | To check valdidations | Attemption should be open | Click on next question  button | next question button | Displays next question from the database | Displays next question from the database | Pass |
| T30 | Attemption | attemption |  | Attemption should be open | Click on finish attempt button | finish attempt button | Opens student result page | Opens student result page | Pass |
| T31 | Student Result | student\_result  View 1 | To check valdidations | Student Result should be open | Click on image | Image button | Opens student\_result view 2 | Opens student\_result view 2 | Pass |
| T31 | Student Result | student\_result  View 2 |  | Student Result should be open | Click on Exit button | Exit button | Opens Student Dashboard page | Opens Student Dashboard page | Pass |
| T32 | Student Registration | Register | To check valdidations | Register should be open | Click on Register button with blank Username  Student name  Email Address  Password  Confirm Password | Username  Student name  Email Address  Password  Confirm Password | Username  Student name,  Email Address,  Password,  Confirm Password  field is required. | Username  Student name,  Email Address,  Password,  Confirm Password  field is required. | Pass |
| T33 | Student Registration | Register |  | Register should be open | Click on Register button with space in Username  and correct  Student name  Email Address  Password  Confirm Password | Username  Student name  Email Address  Password  Confirm Password | Invalid Username | Invalid Username | Pass |
| T34 | Student Registration | Register |  | Register should be open | Click on Register button with  correct Username  Student name  Email Address  Password  Confirm Password | Username  Student name  Email Address  Password  Confirm Password | registration Successful and opens student login page | registration Successful and opens student login page | Pass |

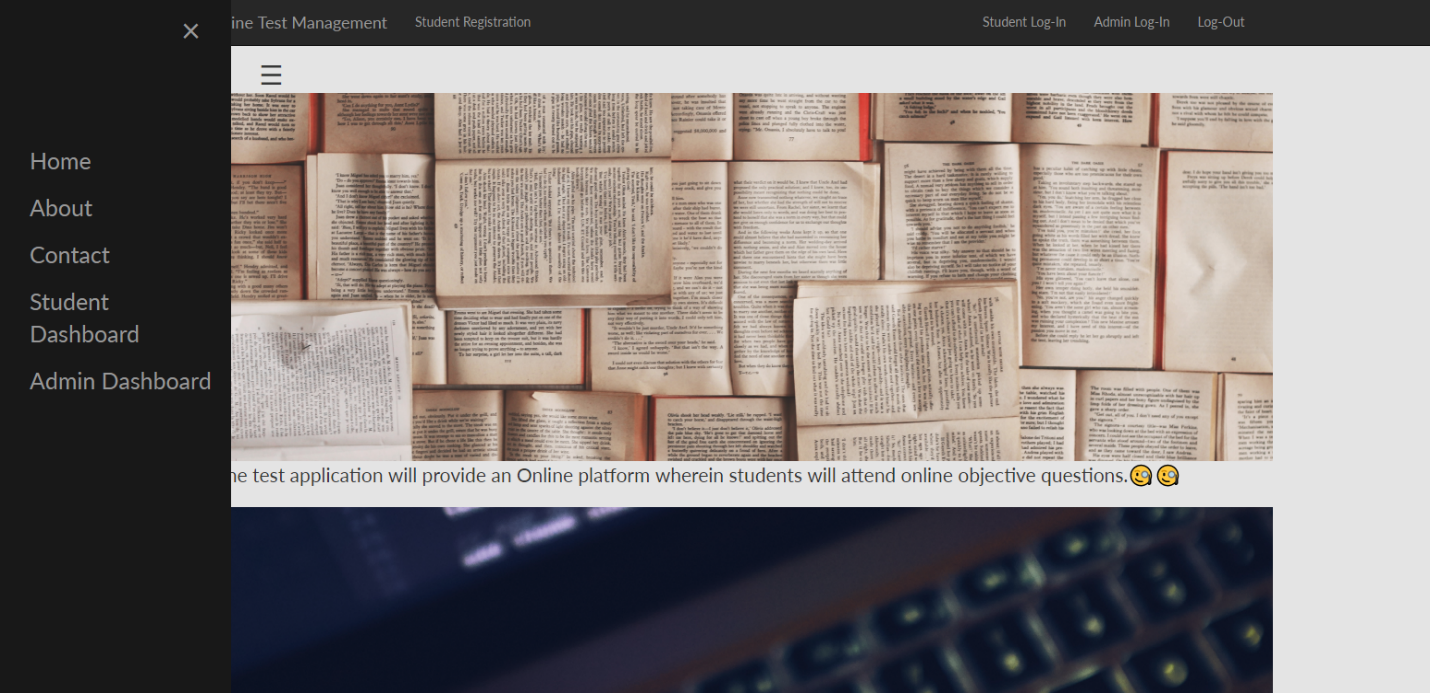
**SYSTEM IMPLEMENTATION**

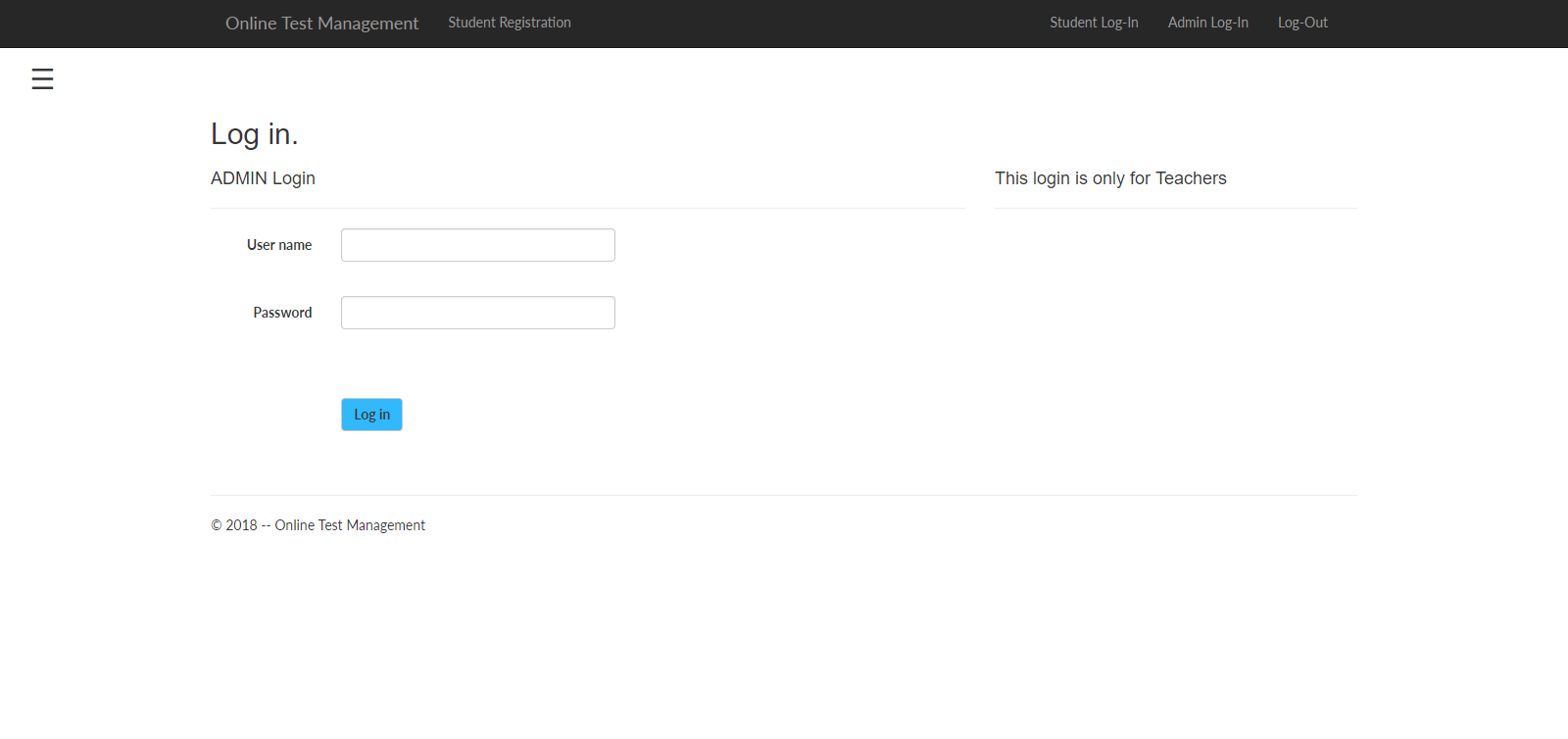
**SCREEN LAYOUTS**

**Home Page**

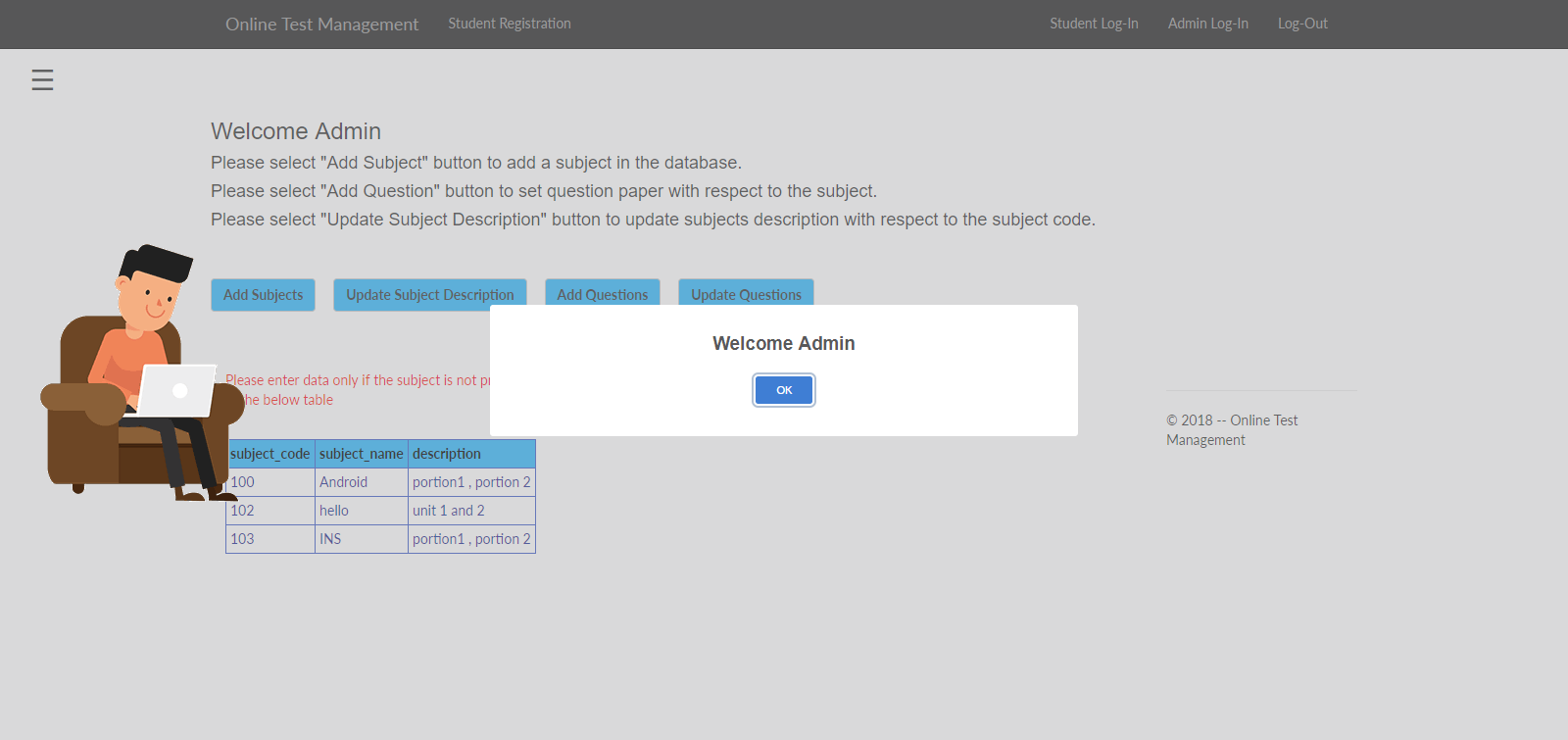
****

**Side Navigation menu**

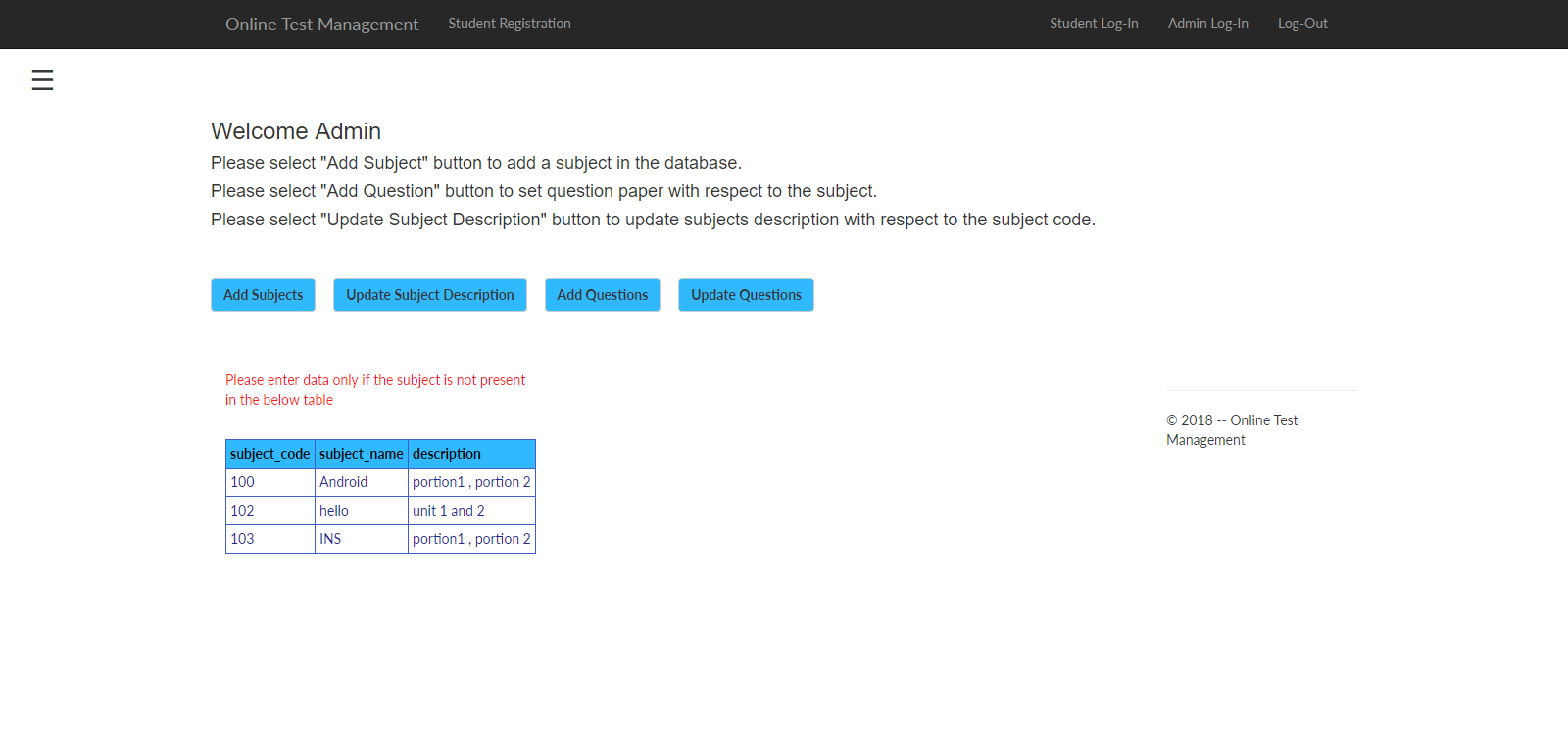
****

**Admin Login**

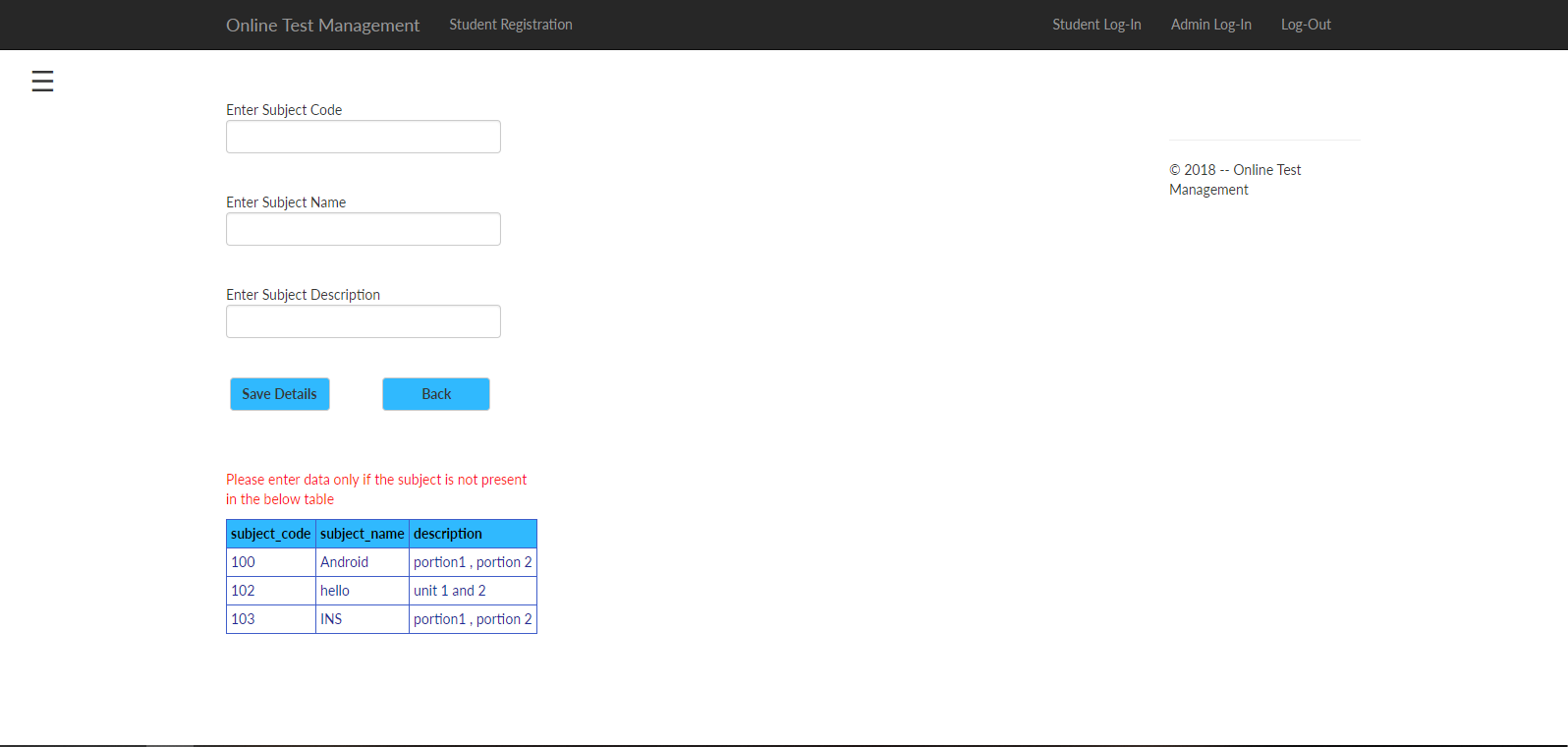
**Admin Dashboard welcome page**

****

**Admin Dashboard**

****

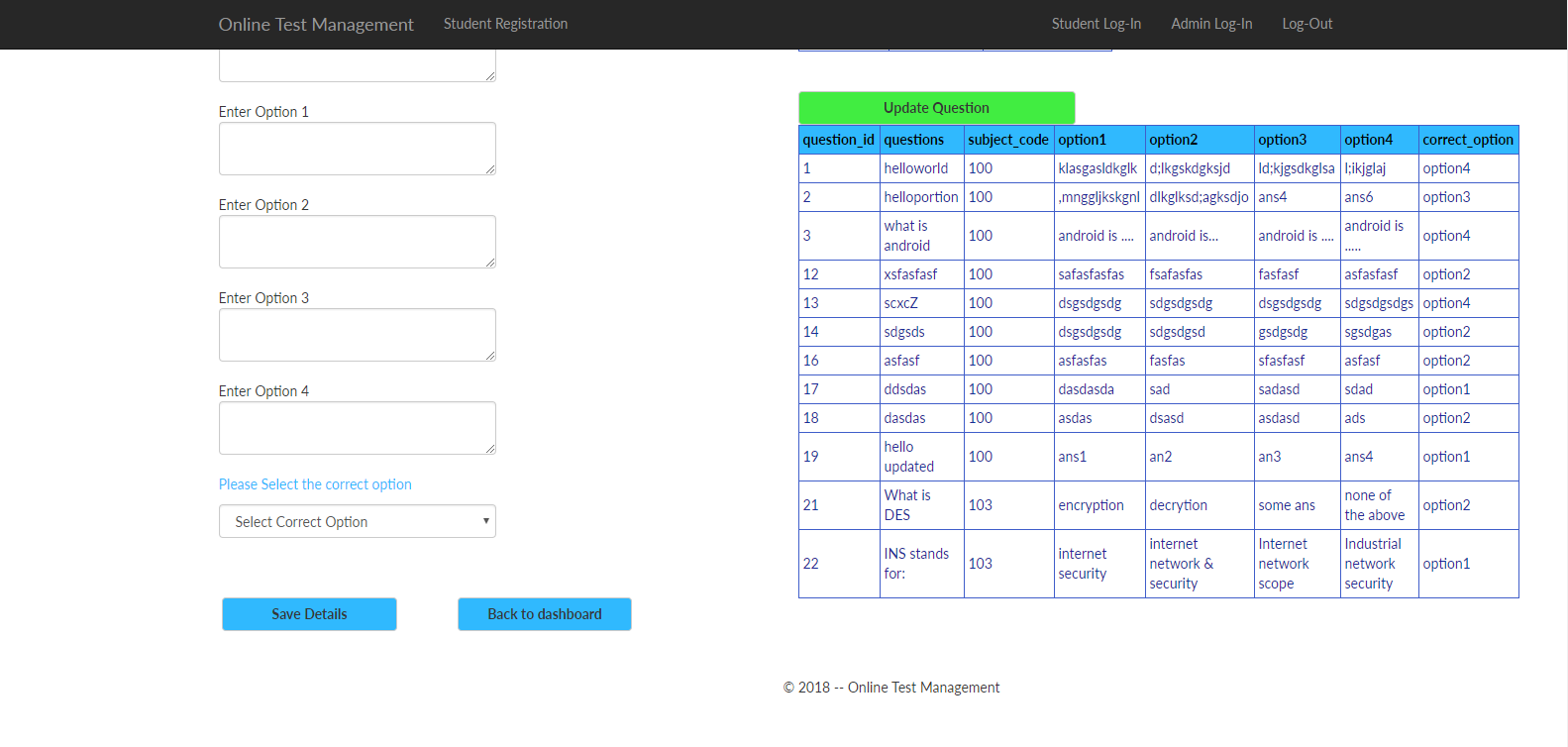
**Add Subject page**

****

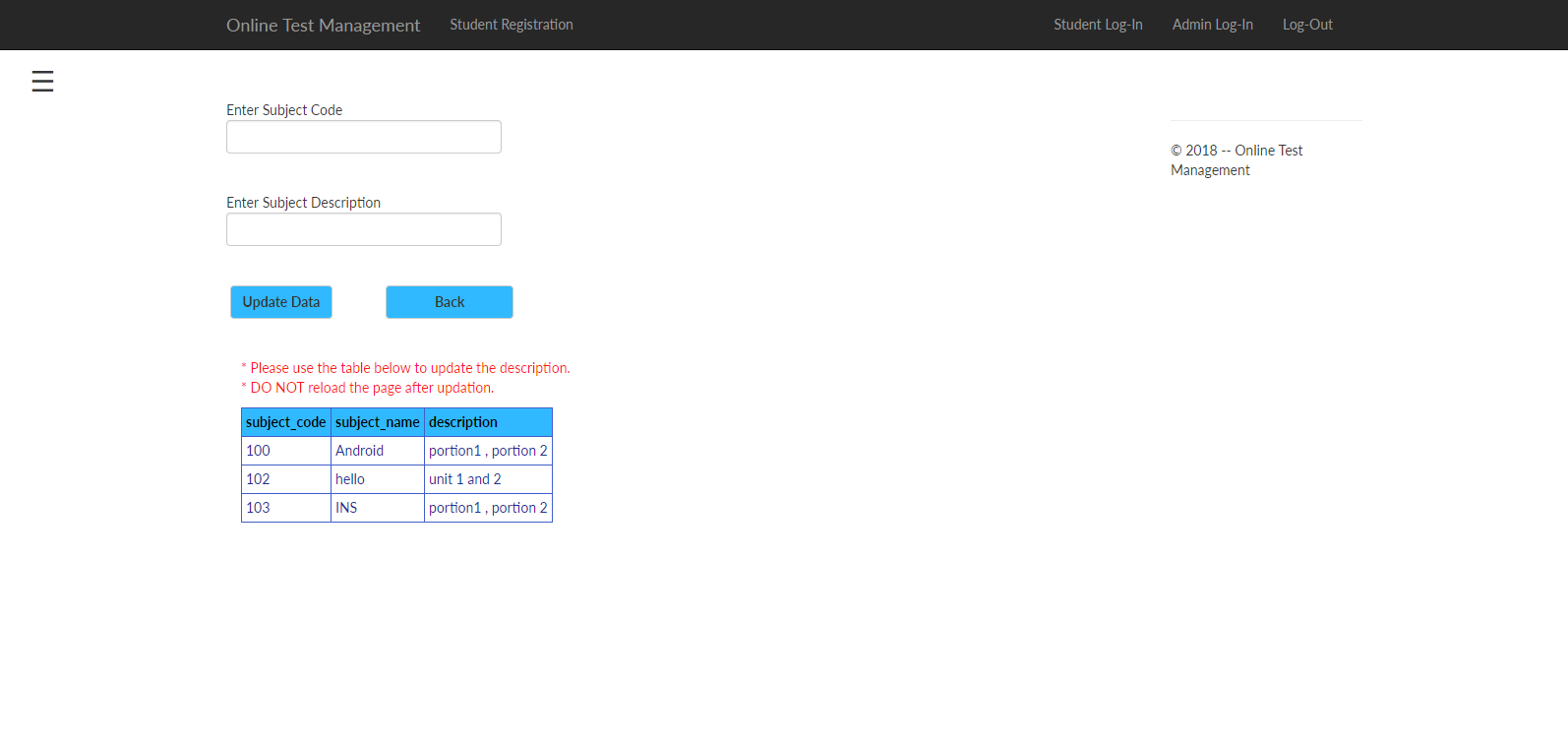
**Add Questions part-1**

****

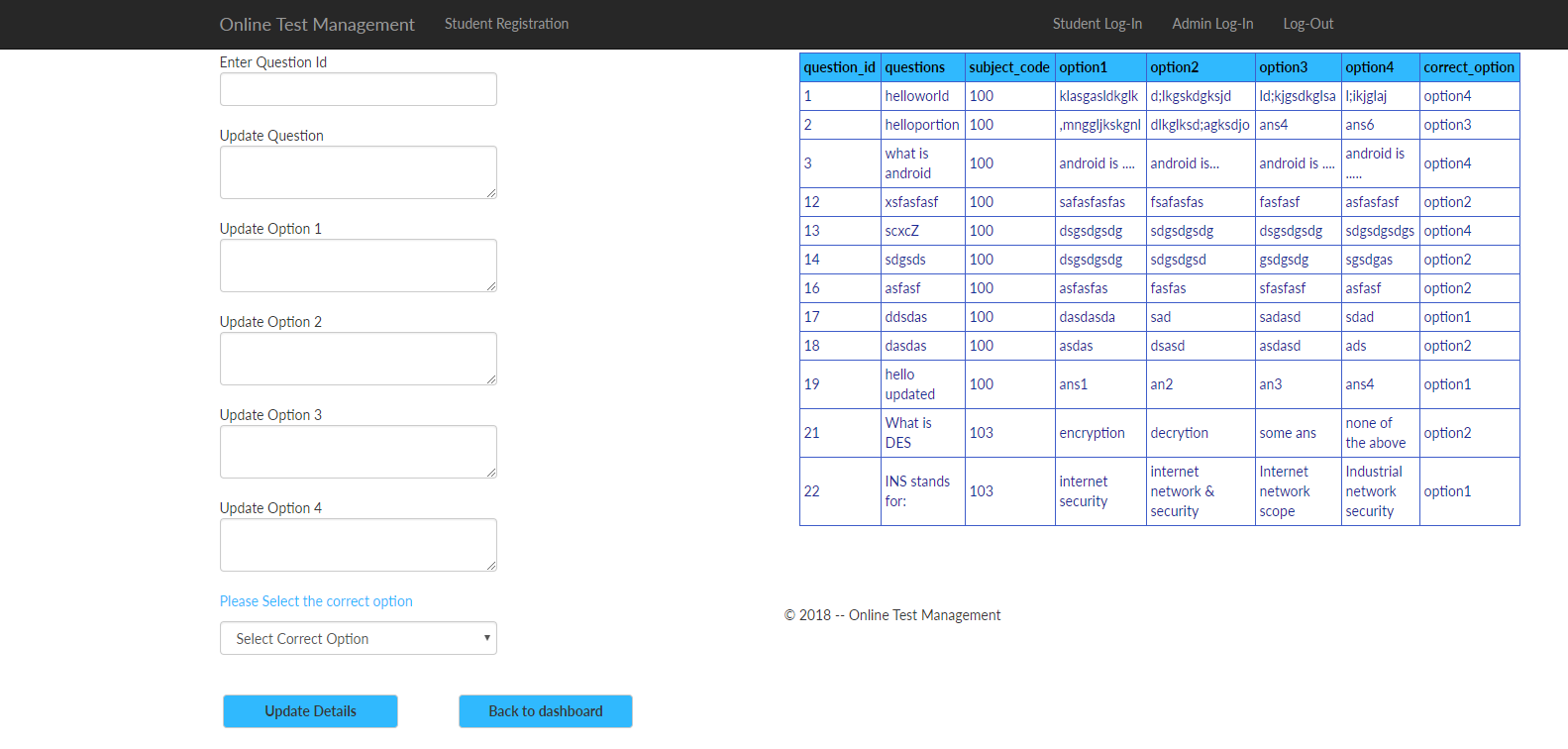
**Add Questions part-2**

****

**Update subject description page**

****

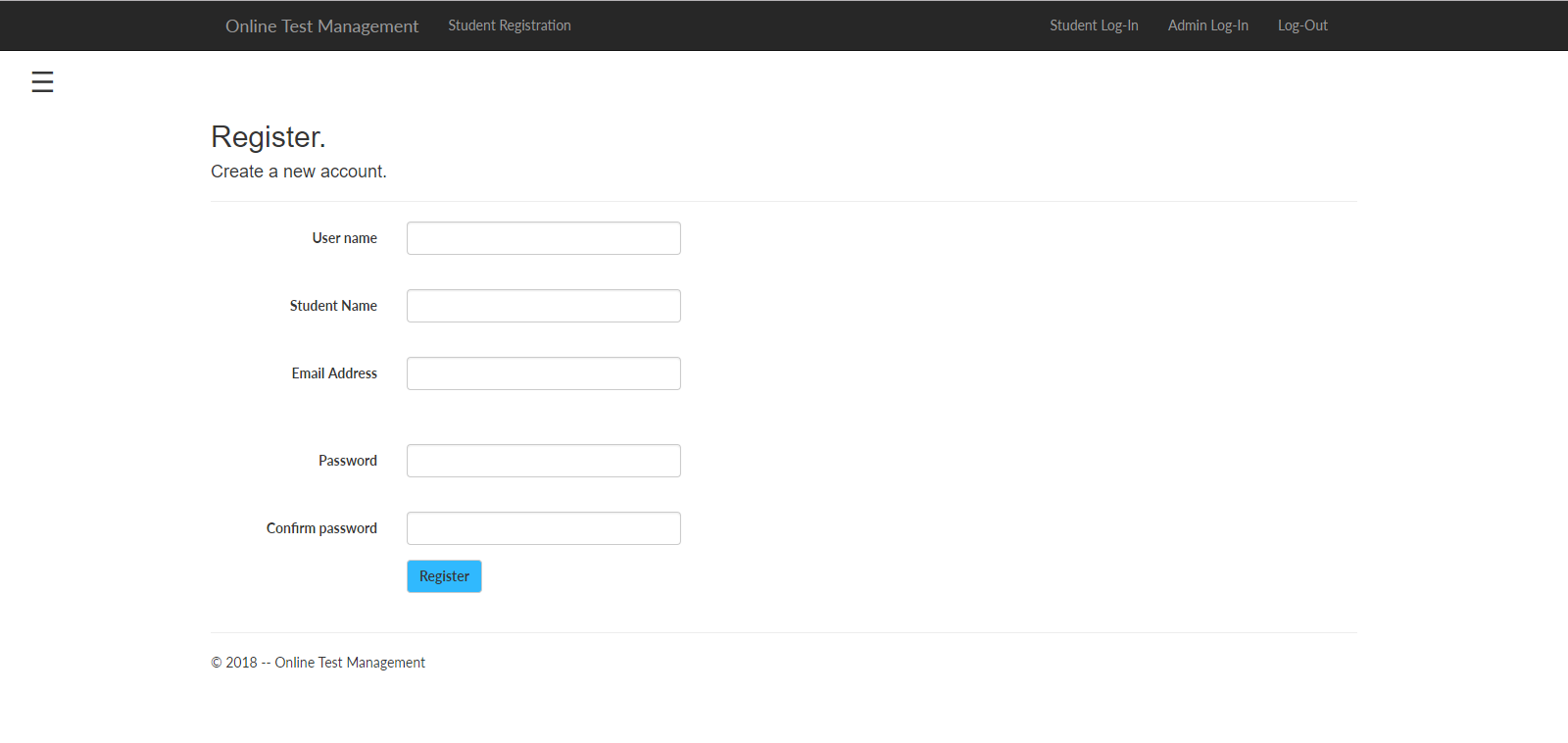
**Update question page**

****

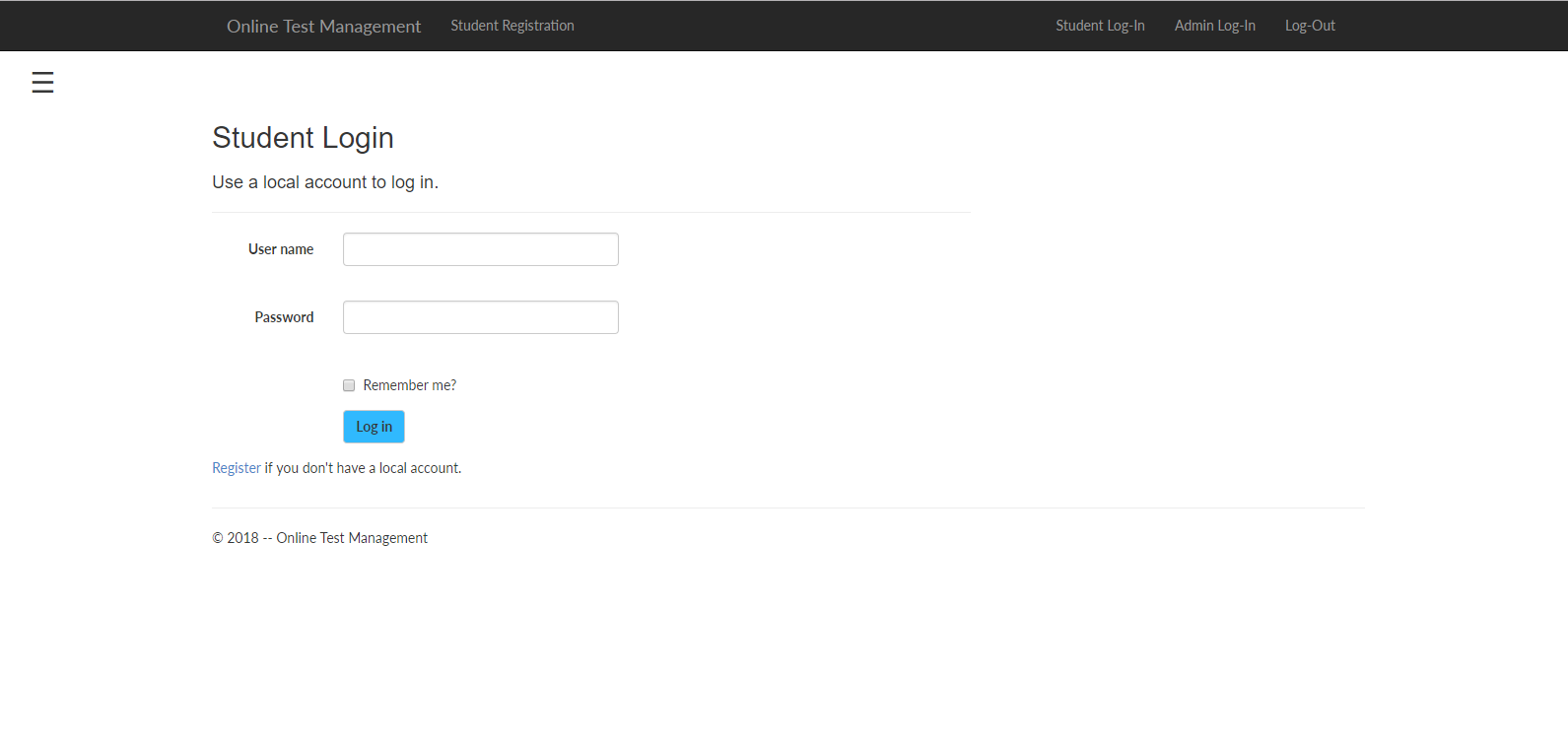
**Log Out**

****

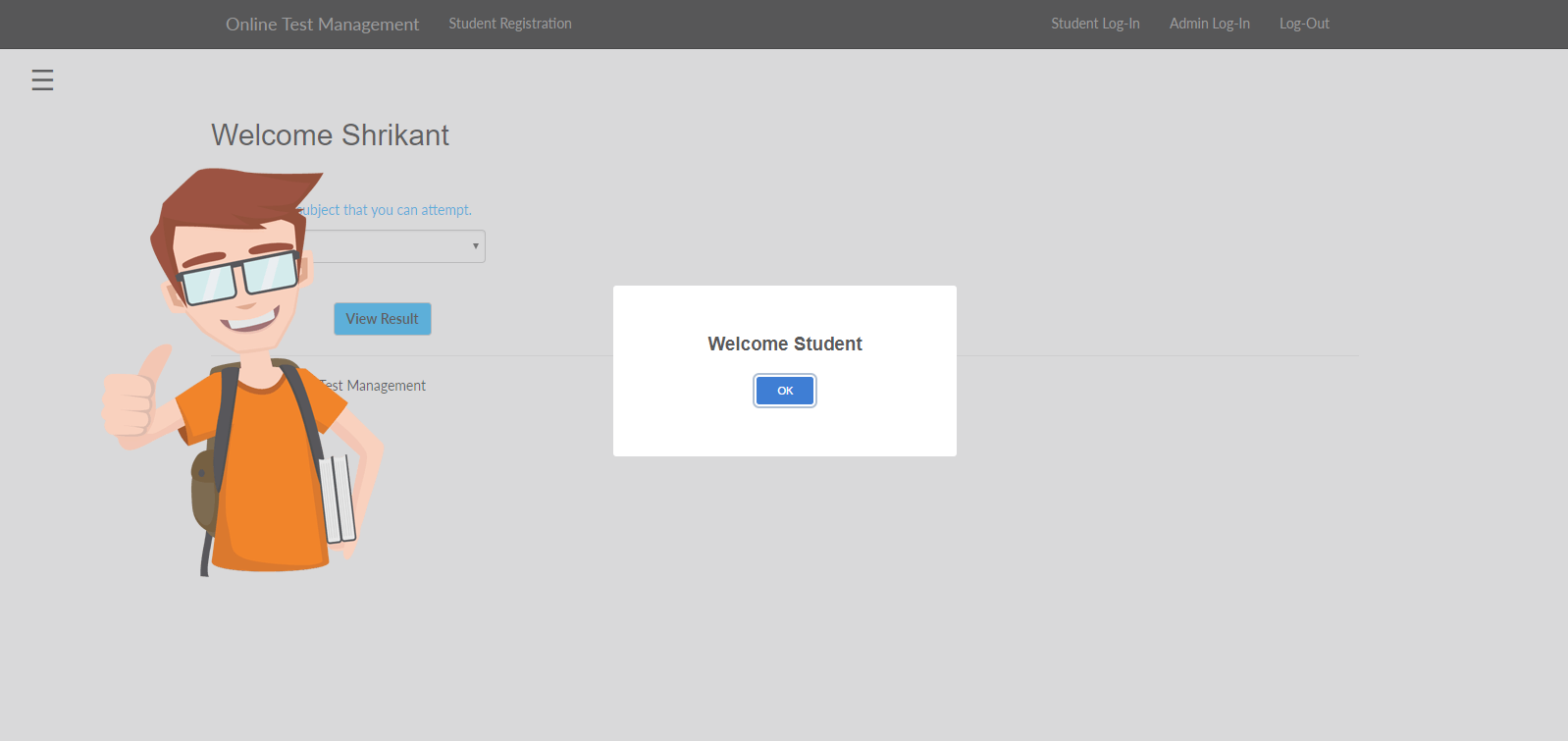
**Student Registration Page**

****

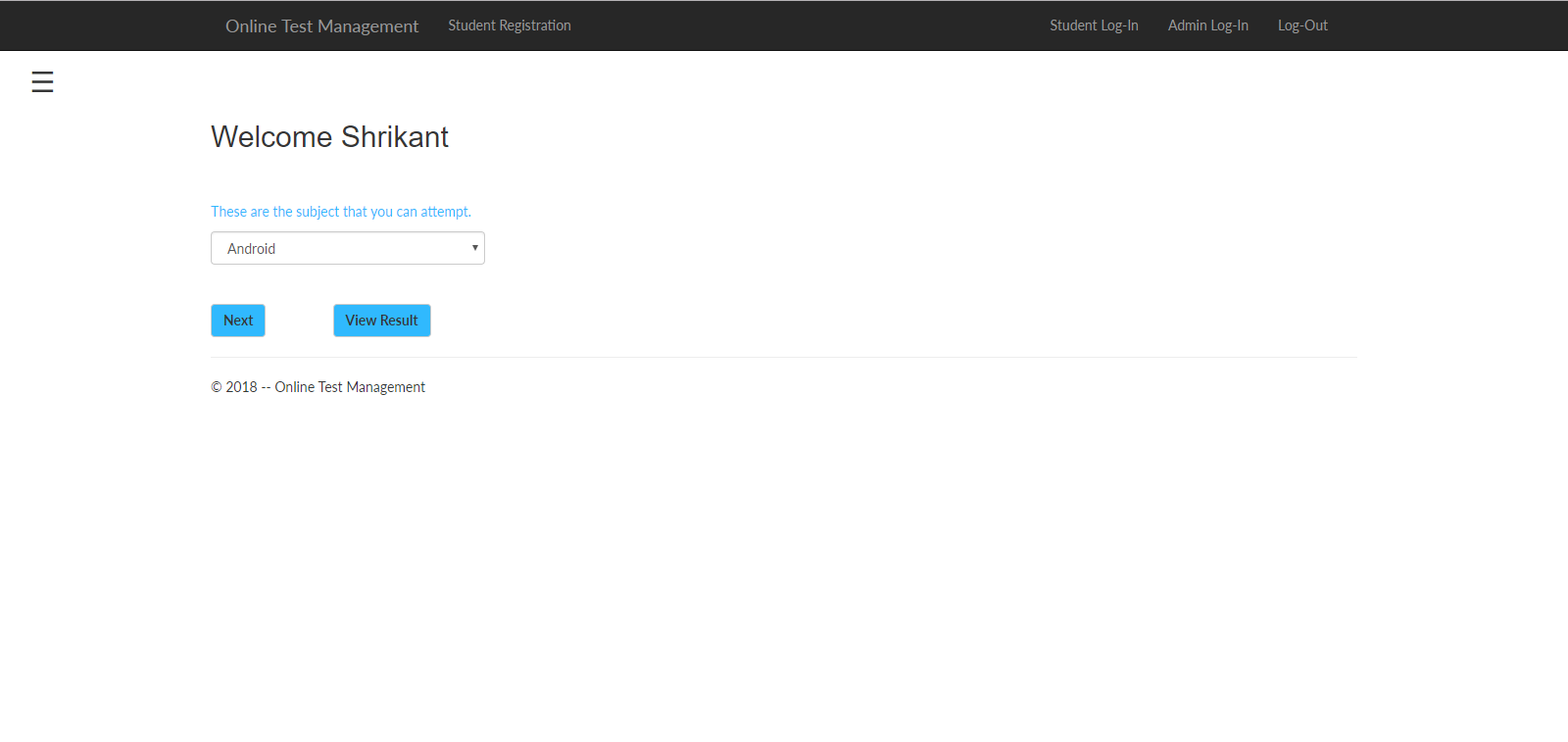
**Student login page**

****

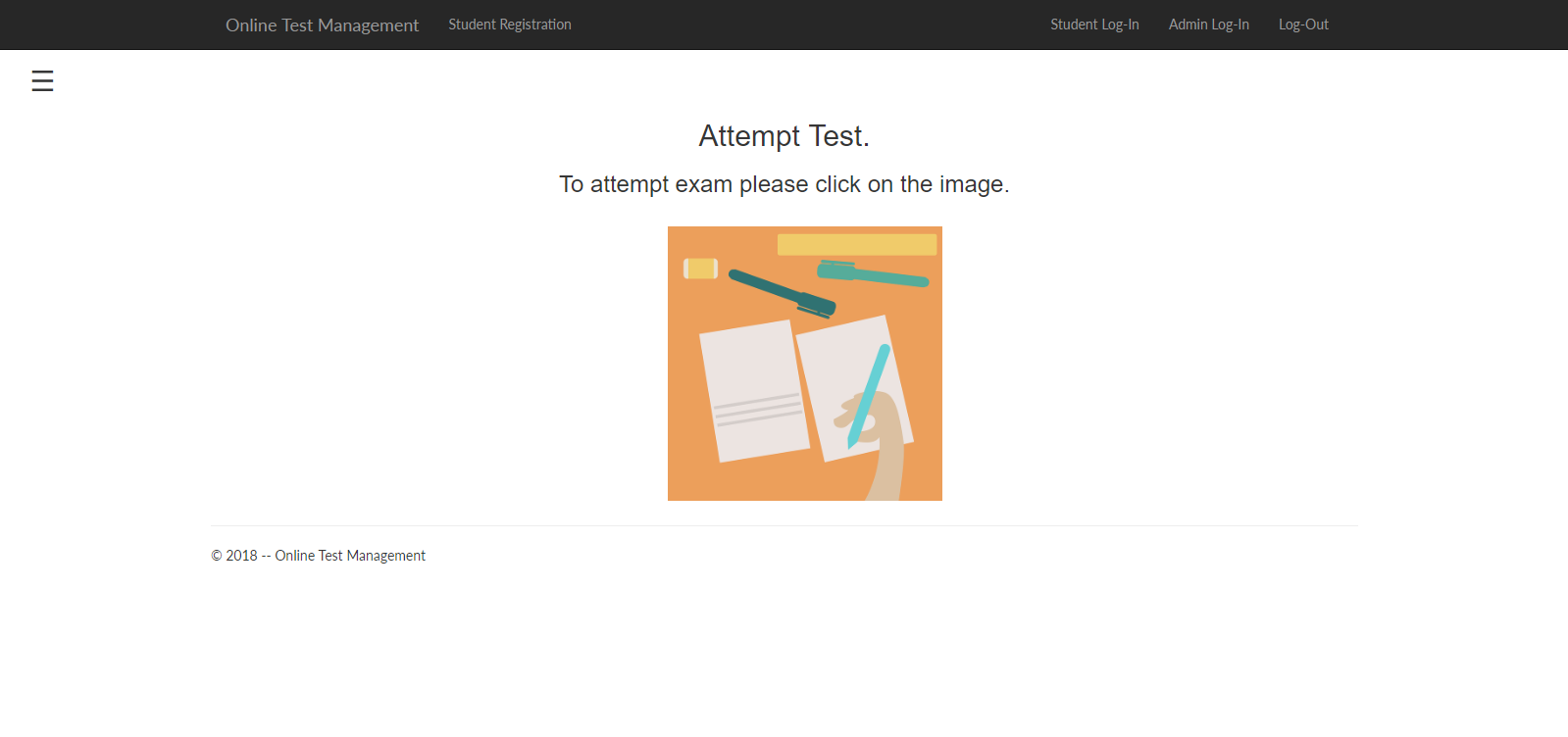
**Student welcome page**

****

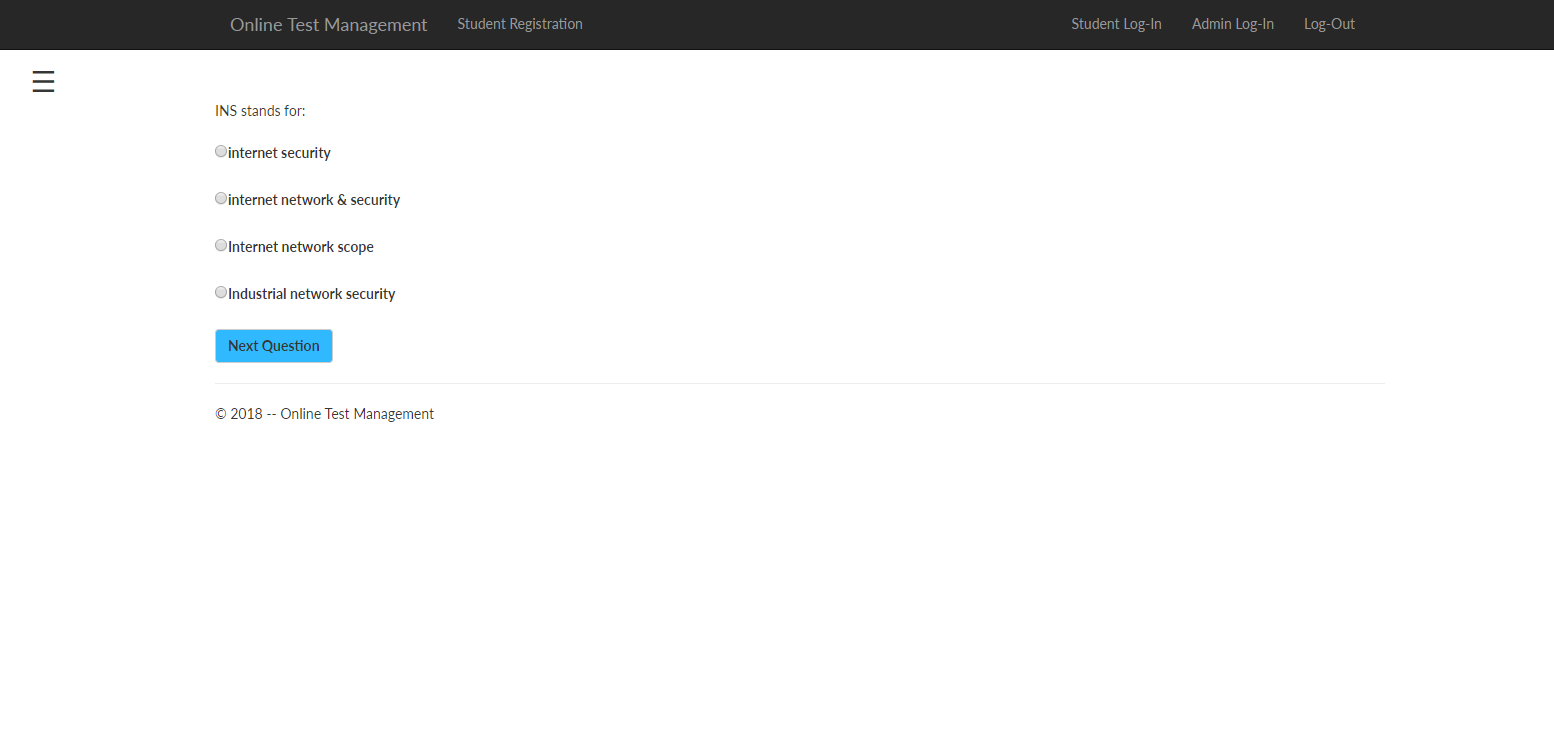
**Student Dashboard**

****

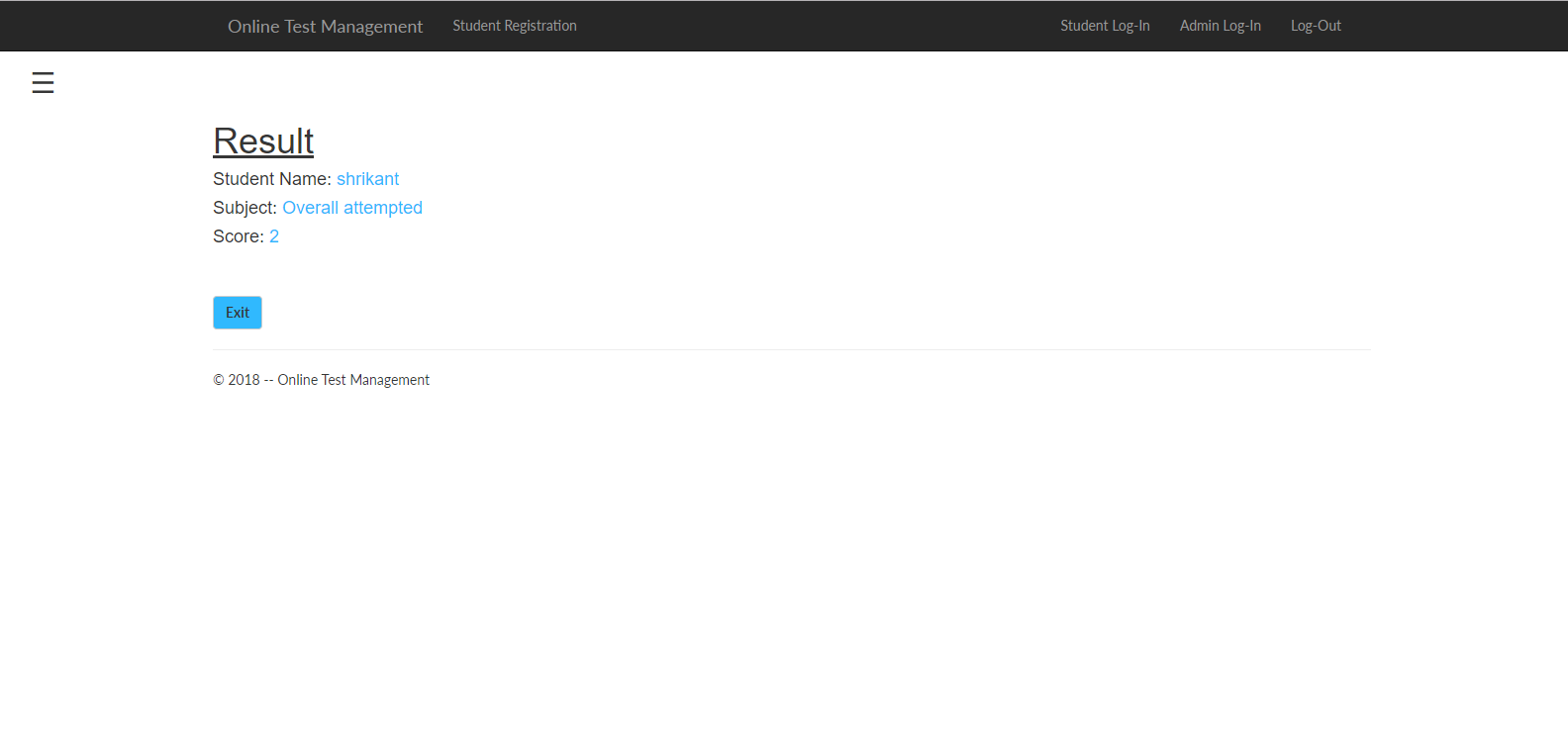
**Examination Page**

****

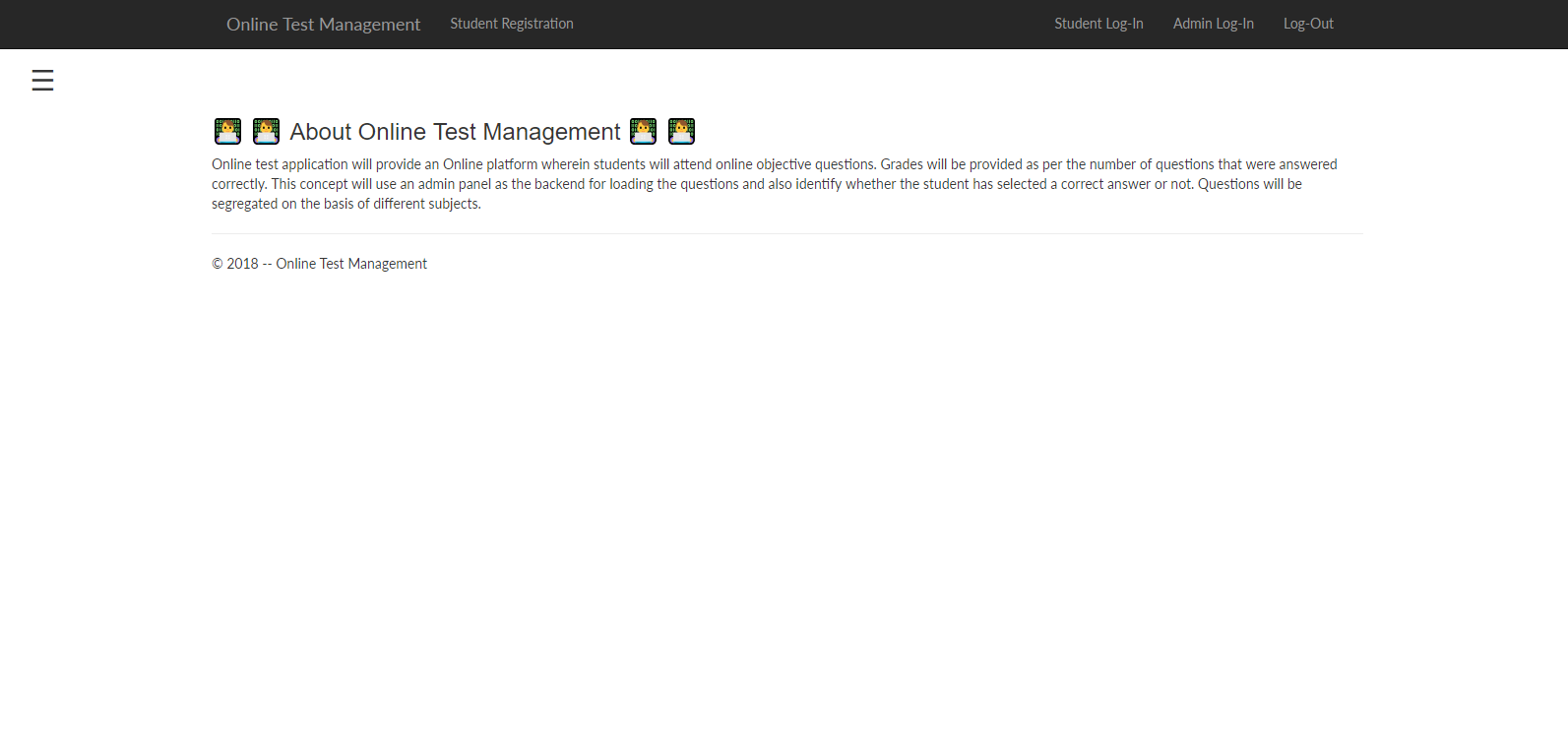
**Attemption page**

****

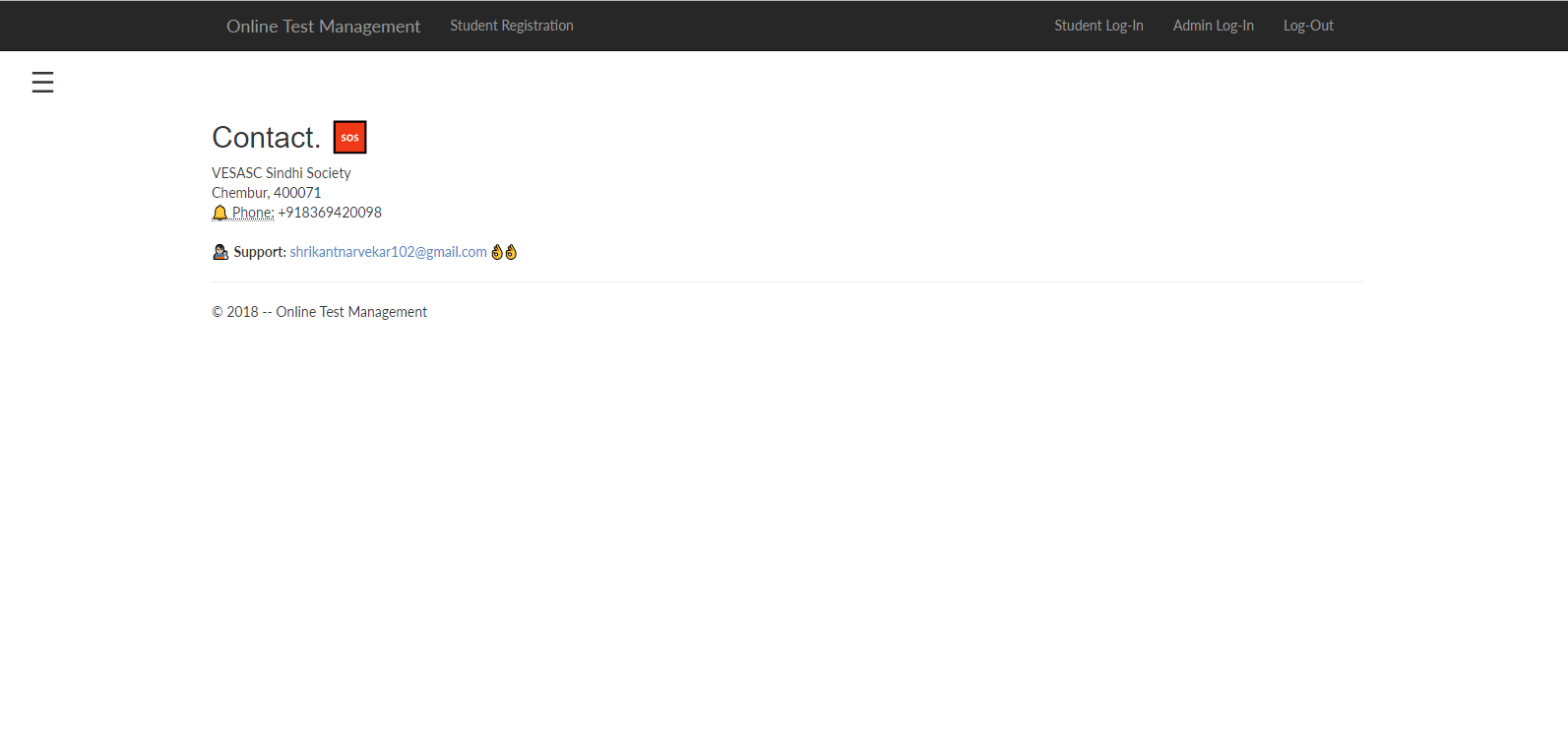
**Result Page**

****

**About Page**

****

**Contact Page**

****

**FUTURE ENHANCEMENTS**

Future Enhancements are those features that can be incorporated in the application in its future versions and provide new features and better working of the present application.

Following are some of the few enhancements that can be incorporated in this application in future:

* To display score individually with respect to the subject
* To Improve user interface for more ease
* To add percentage on every subject.
* Email system to view result in there respective email addresses

**REFERENCES AND BIBLIOGRAPHY**

**Books Referred:**

* Beginning ASP.NET 4 in C# 2010 by Matthew MacDonald
* Murach’s HTML5 and CSS3 by Zak Ruvalcaba,Anne Bohem, Shroff Publishers and Distributors
* Ramakrishnam, Gehrke, Database Management Systems, McGraw‐Hill, 2007

**Websites Referred:**

* https://sweetalert2.github.io
* [www.asp.net](http://www.asp.net)
* [www.codeproject.com](http://www.codeproject.com)