**1. INTRODUCTION**

**1.1) Purpose**

• The Software is designed to ease the task of result analysis of a department for NBA.

• It maintains two level of users:

- Admin Level

- Staff Level

• The Software includes:

- Maintaining Student & their marks, Staff& their related subject details.

- Students are maintained according to their PRN number.

- Staff are maintained according to their ID.

- Depending on stored records of student, performance throughout four years is calculated.

**1.2) Overview**

- Project is related to Result Analysis for NBA.

- The project maintains two level of users:

• Admin Level and Staff Level

-Main facilities available in this project are:

• Maintaining records of students.

• Maintaining details of student along with their marks and staff detailsaccording to academic year.

- In this project collection of data is from different student, staff and marks of each student.

- Changes in student, staff details will be updated in the database.

- Admin can add users as per requirement (can add staff as user).

**2. DATABASE PLANNING**

**Database Planning:**

Database Planning is nothing but the sequential arrangement of the operations to be performed on the given database and arranging the operated data, forms or newly created codes in sequence so that the project can function satisfactorily.

The management activities allow the stages of database application to be realized as efficiently and effectively as possible.

Database planning is the preliminary stage to achieve the objective of project. The various stages included in the Database Planning are as follows:

1. Identification of enterprise plans and goals with subsequent determination of information system needs.
2. Evaluation of current information system to determine existing strengths and weaknesses.
3. Appraisal of IT opportunities that might yield competitive advantage.

**2.1) Problem Statement and Objectives:**

To overcome the drawback of manual work to calculate performance of students for NBA.

**2.1.1) Problems with conventional system**

**1. Lack of immediate retrievals:** The information is very difficult to retrieve and to find particular information like- E.g. - To find out about the mark’s history, the user has to go through various registers and excel files. This results in inconvenience and wastage of time.

**2. Lack of immediate information storage:** The information stored in excel sheets and registers takes time and efforts to be stored at right place.

**3. Lack of prompt updating:** Various changes to information like student details or their marks records are difficult to make as paper work involved.

**4. Error prone manual calculation:** Manual calculations are error prone and take a lot of time this may result in incorrect information. For example success rate of students with or without backlog.

**5. Preparation of accurate and prompt reports:** This becomes a difficult task as information is difficult to collect from various registers

**2.1.2) Goals of proposed system:**

**1. Planned approach towards working:** The working in the organization will be well planned and organized. The data will be stored properly in data stores, which will help in retrieval of information as well as its storage.

**2. Accuracy:** The level of accuracy in the proposed system will be higher. All operations would be done correctly and it ensures that whatever information is coming from the centre is accurate.

**3. Reliability:**  The reliability of the proposed system will be high due to the above stated reasons. The reason for the increased reliability of the system is that now there would be proper storage of information.

**4. No Redundancy:** In the proposed system utmost care would be that no information is repeated anywhere, in storage or otherwise. This would assure economic use of storage space and consistency in the data stored.

**5. Immediate retrieval of information:** The main objective of proposed system is to provide for a quick and efficient retrieval of information. Any type of information would be available whenever the user requires.

**6. Immediate storage of information:** In manual system there are many problems to store the largest amount of information.

**7. Easy to Operate:** The system should be easy to operate and should be such that it can be developed within a short period of time and fit in the limited budget of the user.

**Alternative Solutions:**

**1. Improved Manual System:** One of the alternative solutions is the improvement of the manual system. Anything, which can be done by using automated methods, can be done manually. But the question arises how to perform thing manually in a sound manner. Following are some suggestions, which can be useful in the manual system.

A more sophisticated register maintenance for various students’ information and a good system for storing marks records can be maintained at central place.

Adequate staff may be maintained so that updates are made at the very moment at the same time. Proper person for proper work should be made responsible so that a better efficiency could be achieved. This needs a lot of work force.

**2. Batch System:** Another alternative solution can be used of computer based batch system for maintaining the information regarding student marks, staff of particular subject etc. A batch system refers to a system in which data is processed in a periodical basis.

The batch system is able to achieve most of the goals and sub goals. But a batch system data is processed in sequential basis. Therefore batch system is not suggested.

**3. Online System:** This system provides online storage/ updating and retrieval facility. This system promises very less or no paper work and also provides help to mechanics and operational staff.

In this system everything is stored electronically so very less amount of paper work is required and information can be retrieved very easily without searching here and there into registers. This system is been discussed here.

**3. SYSTEM DEFINITION**

**3.1) Background**

Software manages the facilities like:

- Make a record of intake per year.

- Maintains record of marks of each student throughout four years.

- Find grades according to marks stored.

- Maintains record of teaching staff from college.

- Updates of subjects if change during academic years.

-Updates student information like migrated to, marks.

These are the various jobs that need to be done in an each branch by the respective staff. All this work is done on computers.

The work is done as follows:

- Initially records are entered in database as per admissions are taken place in college.

- Each department maintains their respective records.

-According to respective academic year record of staff for particular subject is maintained.

-If in case staff changes we can update or delete it.

-In case of student records student get migrated or leave college we can update or delete entry from database.

- We can store marks in database of each student and can maintain record throughout four years.

- By using this software we need not to carry all the paper work or store information in various excel sheets.

- All the detailed information of each student, staff will stored in database permanently.

**3.2) Scope:**

It can be used for any department of any college in making result analysis for NBA.

**3.3) User Characteristics**

Every user (admin or staff) should be:

• Comfortable of working with computer.

• He must have knowledge of NBA system.

• He must also have basic knowledge of English.

**3.4) System analysis:**

Analysis of the system can be divided into following 3 phases:

* Structured Analysis
* Package Selection
* System specifications

**4. Requirement Collection and Analysis**

**4.1) Requirement gathering and analysis:**

This activity typically involves interviewing the end-users and studying the existing documents to collect all possible information regarding the system. If the project involves automating some existing procedures.

The main purpose of this activity is to clearly understand the exact requirements of the NBA system. After the exact requirements are understood, the next step is to identify and resolve the various requirements problems. The most important requirement problems are the problems of anomalies, inconsistencies and incompleteness.

**4.2) Software Requirements Specification (SRS)**

Once the analyst has performed all the above mentioned activities, all the requirements are to be systematically organized in the form of an **SRS** document. The SRS document usually contains all the user requirements in an informal form.

An SRS document clearly documents the following aspects of a system:

* Environmental characteristics
  + Hardware
  + Software
* Functional Requirements
* Non-Functional Requirements

**4.2.1) Environmental Characteristics:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Hardware Requirements | | | | |
| Processor | | RAM | | Disk Space |
| i7 Core Processer, 4th Generation | | 8 GB | | 1TB |
| Software Requirements | | | | |
| Operating System | Application Software | | Database | |
| Win-8,Win-8.1 | Microsoft Visual Studio 2015 | | MongoDB | |

**4.2.2) Functional Requirements:**

Each high-level functional requirement corresponds to an instance of use of the system by the user in some way. Each high-level requirement typically involves accepting some data from the user, transforming it to required response, and outputting the response to the user.

Some of the functional requirements which we have tried to gather for our project on Result Analysis for NBA are as:

Students take admission in college and choose branch on their own interest.

Branch has to make record of each student admitted. During academic year some students may be migrated or leave the college. So branch has to make update of that too.

**A.** The Things of Interest include

A.1 Staff able to do data entry

A.2 Students

A.3 Teaching Staff

A.4 Marks of students.

**B.** These things of interest are related as follows

B.1 Staff doing data entry enters records of student and their marks.

B.2 Teaching staff role is to add or edit record of subjects.

B.3 Students are related as marks scored by them are stored in the system.

**4.2.3) Non-Functional Requirements:**

Non-Functional requirements are the ones that specify criteria that can be used to judge the operation of a system, rather than specific behaviours.

The ones concerned for the project are as:

* **Security**

Security is the degree of protection against danger, damage, loss and crime. In order to ensure security, a login is maintained which is user name and password secured and hence forth is accessible by only certain trustworthy people from admin.

* **Maintainability**

Maintainability refers to the ease with which a product can be maintained in order to:

* + Isolate requirements of their cause
  + Meet new requirements
  + Make future management easier or,
  + Cope with a changed environment
* **Performance**

Good performance is maintained by providing a reliable and high quality service to the staff as staff satisfaction is the top most priority.

* **Testability**

Testability refers the capability of a system to be tested. As this system depicts a real life scenario it can be easily tested, as to how it is able to store student details & their marks and generate the success rate of students.

* **Usability**

Usability is the ease of use and learns ability. System is made very user friendly using interactive GUI so that it is easy to use, understand and maintain as well.

* **Robustness**

Robustness is the ability of a system to cope with errors during execution. As the system has been tested for all sorts of invalid and unexpected inputs hence it is Robust in nature.

* **Accessibility**

Accessibility can viewed as the “ability to access” and benefit from some system or entity. System is accessible only to admin for special functions and hence is easily accessible and easy to maintain.

* **Portability**

It describes that how easy it is to reuse the system. It requires generalized abstraction between the application logic and system interfaces. This system is quite portable as a few minor changes and it can cater to various departments.

**4.2.4) Sources of information/requirement gathering:**

Information can be gathered from two major sources:

1. Internal Sources

* Professional staff
* System Documentation or manuals.
* Reports and transaction documents.

2. External Sources

* University Documents.
* Internet
* Network guides/books/articles

**5.Database Design**

**5.1 Conceptual Design:**

**Administrator handling the system:**

After performing specified tasks, system shut down.

Maintains the system by performing certain tasks

Admin Logs in the system

Maintains Student Record

Manages User details & Subjects.

Maintains Marks of Student according to Semester.

Maintains Staff Record

Analysis of Marks according to stored data.

**5.2 Logical Model:**

**Students**

**Scored**

**Marks**

**In**

**Subjects**

**Staffs**

**Teaching**

**FE**

**SE**

**TE**

**BE**

**ISA**

\*

1

\*

1

\*

1

|  |
| --- |
| **Subject\_Table** |
| **Sr. No.** |
| **Academic Year** |
| **Class** |
| **Semester** |
| **Subject Name** |
| **Max Marks** |
| **Staff Name** |

**5.3 Relational Schema:**

|  |
| --- |
| **Student\_Table** |
| **PRN No** |
| **Student Name** |
| **Academic Year** |
| **Admission Type** |

|  |
| --- |
| **Staff\_Table** |
| **Staff Id** |
| **Staff Name** |

|  |
| --- |
| **FE\_Marks\_Table** |
| **PRN No** |
| **Seat No** |
| **Subject 1** |
|  |
| **Subject 22** |
| **Sem I Total** |
| **Sem II Total** |
| **Sem I Grade** |
| **Sem II Grade** |

|  |
| --- |
| **SE\_Marks\_Table** |
| **PRN No** |
| **Seat No** |
| **Subject 1** |
|  |
| **Subject 22** |
| **Sem I Total** |
| **Sem II Total** |
| **Sem I Grade** |
| **Sem II Grade** |

|  |
| --- |
| **BE\_Marks\_Table** |
| **PRN No** |
| **Seat No** |
| **Subject 1** |
|  |
| **Subject 22** |
| **Sem I Total** |
| **Sem II Total** |
| **Sem I Grade** |
| **Sem II Grade** |

|  |
| --- |
| **TE\_Marks\_Table** |
| **PRN No** |
| **Seat No** |
| **Subject 1** |
|  |
| **Subject 22** |
| **Sem I Total** |
| **Sem II Total** |
| **Sem I Grade** |
| **Sem II Grade** |

**5.4 Schema Diagram:**

**Tables:**

1. **Student:**

|  |  |  |  |
| --- | --- | --- | --- |
| **PRN\_No** | **Student\_Name** | **Academic\_Year** | **Admission\_Type** |

1. **Staff:**

|  |  |
| --- | --- |
| **Staff\_Id** | **Staff\_Name** |

1. **Subjects:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sr\_No** | **Academic\_Year** | **Class** | **Semester** | **Subject\_Name** | **Marks** | **Staff\_Name** |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PRN\_No** | **Seat\_No** | **Subject\_1** | **……** | **Subject\_22** | **Sem1**  **\_Total** | **Sem1**  **\_Grade** | **Sem2**  **\_Total** | **Sem2**  **\_Grade** |

1. **FE\_Marks:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PRN\_No** | **Seat\_No** | **Subject\_1** | **……** | **Subject\_22** | **Sem1**  **\_Total** | **Sem1**  **\_Grade** | **Sem2**  **\_Total** | **Sem2**  **\_Grade** |

1. **SE\_Marks:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PRN\_No** | **Seat\_No** | **Subject\_1** | **……** | **Subject\_22** | **Sem1**  **\_Total** | **Sem1**  **\_Grade** | **Sem2**  **\_Total** | **Sem2**  **\_Grade** |

1. **TE\_Marks:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PRN\_No** | **Seat\_No** | **Subject\_1** | **……** | **Subject\_22** | **Sem1**  **\_Total** | **Sem1**  **\_Grade** | **Sem2**  **\_Total** | **Sem2**  **\_Grade** |

1. **BE\_Marks:**

### 6. Normalized Tables

1. **Student:**

|  |  |  |  |
| --- | --- | --- | --- |
| **PRN\_No** | **Student\_Name** | **Academic\_Year** | **Admission\_Type** |

**PRN\_No->Student\_Name,Academic\_Year,Admission\_Type**

**The table Student is in 1 NF as:**

* All the underlying domains of the relation contain atomic (indivisible) values.
* No repeating groups in the table.
* All attributes dependent on primary key.

**The table Student is in 2 NF as:**

* It is in 1 NF
* It includes no partial dependencies as every non key attribute is fully dependent on the primary key.

**The table Student is in 3 NF as:**

* It is in 2 NF
* Every non key attribute is non-transitively dependent upon the primary key.

1. **Staff:**

|  |  |
| --- | --- |
| **Staff\_Id** | **Staff\_Name** |

**Staff\_Id->Staff\_Name**

**The Staffs in 1 NF as:**

* The underlying domain of the relation contains atomic (indivisible) values.
* No repeating groups in the table.

**The table Staffs in 2 NF as:**

* It is in 1 NF
* It includes no partial dependencies as there are no non key attributes.

**The table Staffs in 3 NF as:**

* It is in 2 NF
* It includes no transitive dependencies as there are no non key attributes.

1. **Subjects:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sr\_No** | **Academic\_Year** | **Class** | **Semester** | **Subject\_Name** | **Marks** | **Staff\_Name** |

**Sr\_No+Academic\_Year+Class+Semester->Subject\_Name**

**Subject\_Name->Marks, Staff\_Name**

**The table Subjects is in 1 NF as:**

* All the underlying domains of the relation contain atomic (indivisible) values.
* No repeating groups in the table.
* All attributes dependent on primary key.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr\_No** | **Academic\_Year** | **Class** | **Semester** | **Subject\_Name** |

**Sr\_No+Academic\_Year+Class+Semester->Subject\_Name**

|  |  |  |
| --- | --- | --- |
| **Subject\_Name** | **Marks** | **Staff\_Name** |

**Subject\_Name->Marks, Staff\_Name**

**The table Subjects is in 2 NF as:**

* It is in 1 NF
* It includes no partial dependencies as every non key attribute is fully dependent on the primary key.

**The table Subjects is in 3 NF as:**

* It is in 2 NF
* Every non key attribute is non-transitively dependent upon the primary key.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **PRN\_No** | **Seat\_No** | **Subject\_1** | **……** | **Subject\_22** | **Sem1**  **\_Total** | **Sem1**  **\_Grade** | **Sem2**  **\_Total** | **Sem2**  **\_Grade** |

1. **FE\_Marks, SE\_Marks,TE\_Marks&BE\_Marks:**

**PRN\_No->Seat\_No, Subject\_1,…,Subject\_22, Sem1\_Total, Sem2\_Total**

**Sem1\_Total->Sem1\_Grade**

**Sem2\_Total->Sem2\_Grade**

**The above tables are in 1 NF as:**

* All the underlying domains of the relation contain atomic (indivisible) values.
* No repeating groups in the table.
* All attributes dependent on primary key.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **PRN\_No** | **Seat\_No** | **Subject\_1** | **……** | **Subject\_22** |

**PRN\_No->Seat\_No, Subject\_1,…,Subject\_22, Sem1\_Total, Sem2\_Total**

|  |  |
| --- | --- |
| **Sem1\_Total** | **Sem1\_Grade** |

**Sem1\_Total->Sem1\_Grade**

|  |  |
| --- | --- |
| **Sem2\_Total** | **Sem2\_Grade** |

**Sem2\_Total->Sem2\_Grade**

**The above tables are in 2 NF as:**

* It is in 1 NF
* It includes no partial dependencies as every non key attribute is fully dependent on the primary key.

**The above tables are in 3 NF as:**

* It is in 2 NF
* Every non key attribute is non-transitively dependent upon the primary key.

### 7. DBMS Selection

**Back-end desired features:**

1. Multiple user support.

2. Efficient data handling.

3. Provide inherent features for security.

4. Efficient data retrieval and maintenance.

5. Stored procedures.

6. Popularity.

7. Operating System compatible.

8. Easy to install.

9. Various drivers must be available.

10. Easy to implant with the Front-end.

According to above stated features we selected MongoDB as the backend.

**8. Application Design**

**Front-end selection:**

An important issue for the development of a project is the selection of suitable front-end. When we decided to develop the project we went through an extensive study to determine the most suitable platform that suits the needs of the organization as well as helps in development of the project.

The aspects of our study included the following factors.

1. It must have a graphical user interface that assists employees that are not from IT background.

2. Scalability and extensibility.

3. Flexibility.

4. Robustness.

5. According to the organization requirement and the culture.

6. Must provide excellent reporting features with good printing support.

7. Platform independent.

8. Easy to debug and maintain.

9. Event driven programming facility.

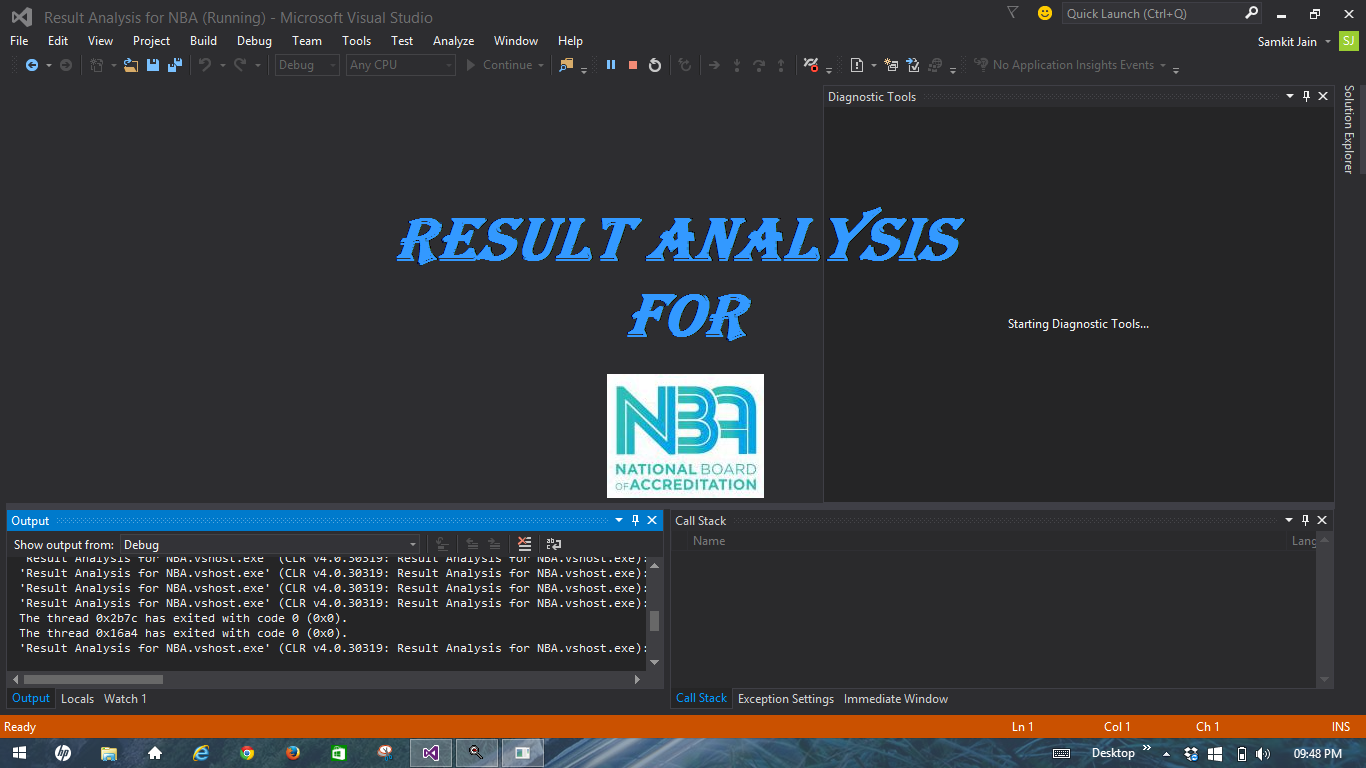
10. Front end must support some popular back end like MongoDB.

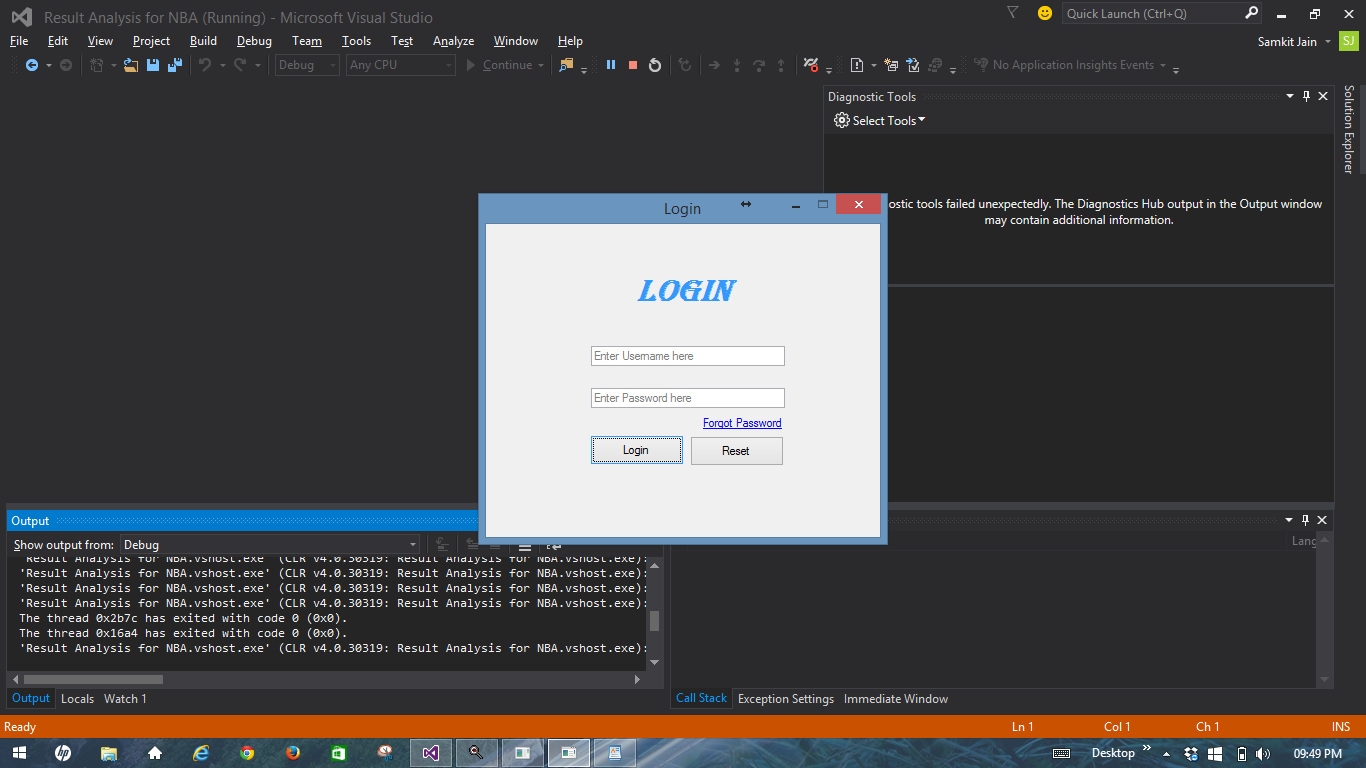
According to the above stated features we selected C# as the front-end for developing our project. As for C#

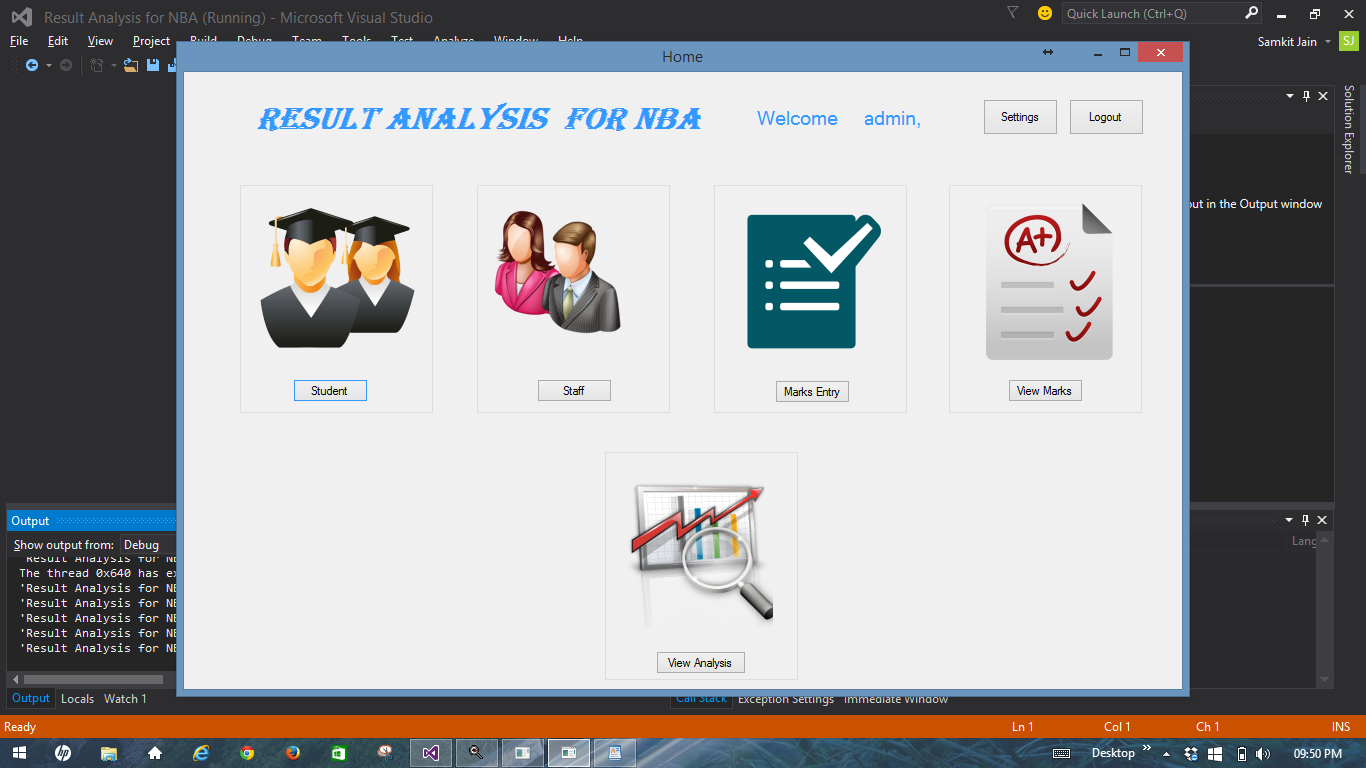
1. Rapid Application Development (RAD) of GUI application.
2. Access to database using: MONGODB CSHARP DRIVER
3. Direct connections to the Windows API
4. Database editing applications

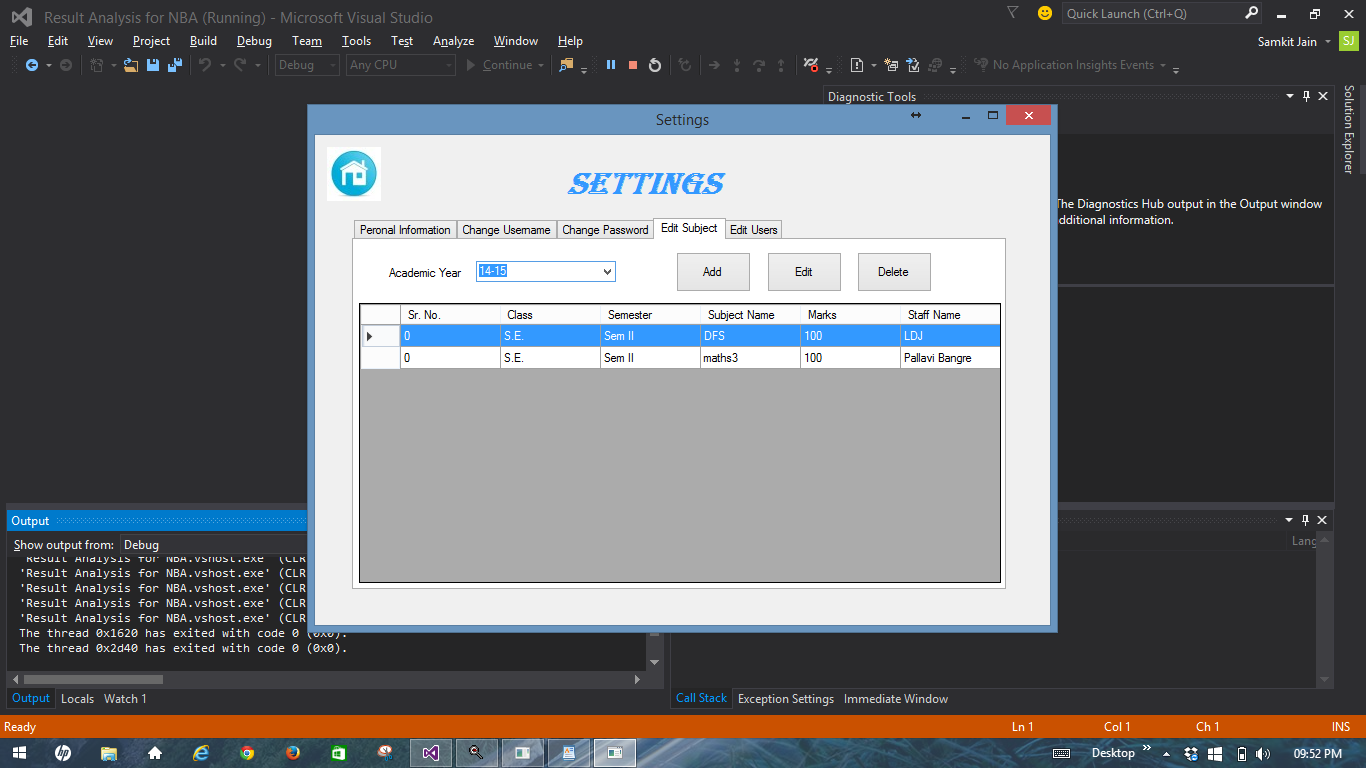
**9. Project Implementation**

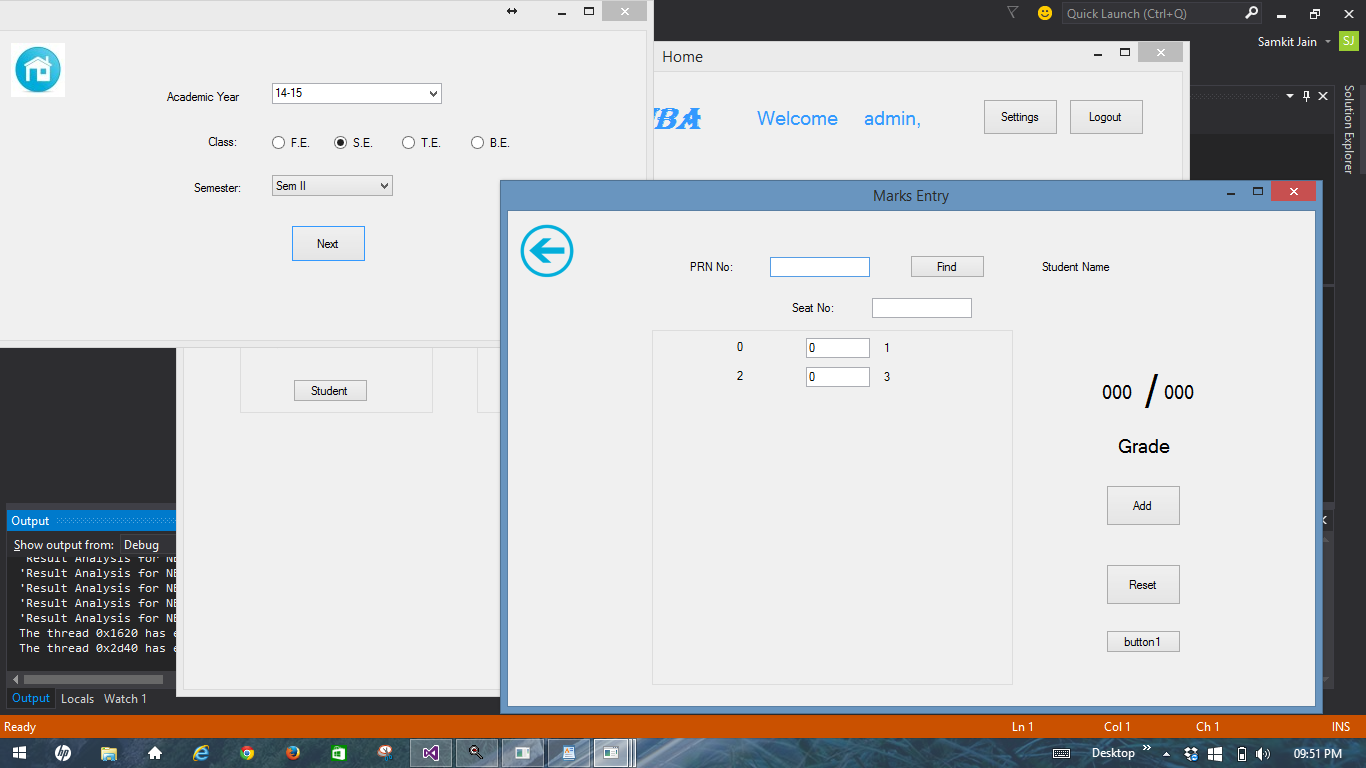
**9.1 Implementation of Front-End:**

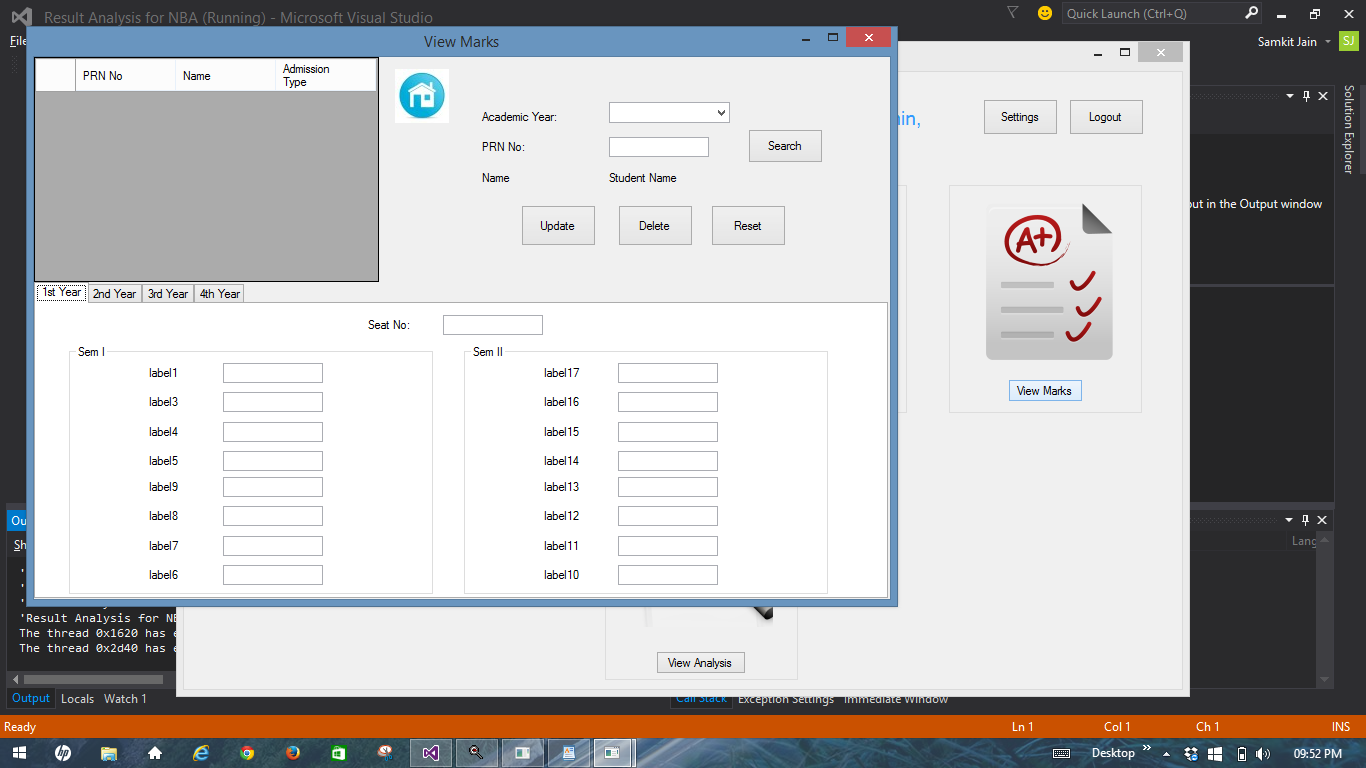












**Code: (for validation and insertion in customer document)**

using System;

usingSystem.Collections.Generic;

usingSystem.ComponentModel;

usingSystem.Data;

usingSystem.Drawing;

usingSystem.Linq;

usingSystem.Text;

usingSystem.Threading.Tasks;

usingSystem.Windows.Forms;

usingMongoDB.Bson;

usingMongoDB.Driver;

usingMongoDB.Driver.Builders;

namespaceResult\_Analysis\_for\_NBA.Settings

{

publicpartialclassfrmView\_Subject : Form

{

publicfrmView\_Subject()

{

InitializeComponent();

}

publicstringSubject\_Name;

publicbool update;

publicstringu\_name, pusername, password, ptype, security, answer, email, mobileno;

privatevoidpicBack\_Click(object sender, EventArgs e)

{

frmSettingsobj = newfrmSettings();

obj.u\_name = u\_name;

obj.username = pusername;

obj.password = password;

obj.type = ptype;

obj.security = security;

obj.answer = answer;

obj.email = email;

obj.mobileno = mobileno;

obj.o = 1;

this.Hide();

obj.Show();

}

privatevoidfrmView\_Subject\_Load(object sender, EventArgs e)

{

MongoClient client = newMongoClient("mongodb://localhost");

MongoServer server = client.GetServer();

MongoDatabasedb = server.GetDatabase("NBA");

MongoCursor<Staffs> staff = db.GetCollection<Staffs>("Staffs").FindAll();

foreach (Staffs i in staff)

{

if (!cmbStaff\_Name.Items.Contains(i.staff\_name))

cmbStaff\_Name.Items.Add(i.staff\_name);

}

MongoCursor<Subjects> sub = db.GetCollection<Subjects>("Subjects").FindAll();

foreach (Subjects i in sub)

{

if (!cmbacademicyear.Items.Contains(i.academic\_year))

cmbacademicyear.Items.Add(i.academic\_year);

}

if (update)

{

MongoCursor<Subjects> put = db.GetCollection<Subjects>("Subjects").FindAll();

foreach (Subjects i in put)

{

if (Subject\_Name == i.Subject\_Name)

{

cmbacademicyear.Text = i.academic\_year;

if (i.clas == "F.E.")

rdfe.Checked = true;

elseif (i.clas == "S.E.")

rdse.Checked = true;

elseif (i.clas == "T.E.")

rdte.Checked = true;

else

rdbe.Checked = true;

cmbsem.Text = i.semester;

cmbsrno.Text = i.srno.ToString();

txtSubject\_Name.Text = i.Subject\_Name;

txtMarks.Text = i.marks;

cmbStaff\_Name.Text = i.Staff\_Name;

break;

}

}

btnAdd.Text = "Update";

txtSubject\_Name.Enabled = false;

}

}

privatevoidbtnAdd\_Click(object sender, EventArgs e)

{

if (validateForm())

{

bool f = true;

stringclas;

if (rdfe.Checked)

clas = "F.E.";

elseif (rdse.Checked)

clas = "S.E.";

elseif (rdte.Checked)

clas = "T.E.";

elseif (rdbe.Checked)

clas = "B.E.";

else

{

MessageBox.Show("Please select the class");

return;

}

MongoClient client = newMongoClient("mongodb://localhost");

MongoServer server = client.GetServer();

MongoDatabasedb = server.GetDatabase("NBA");

MongoCursor<Subjects> put = db.GetCollection<Subjects>("Subjects").Find(Query.EQ("academic\_year", cmbacademicyear.Text.Trim()));

foreach (Subjects i in put)

{

if (clas == i.clas&&cmbsem.Text.Trim() == i.semester&&Convert.ToInt32(cmbsrno.Text) == i.srno)

{

f = false;

if (!update)

{

MessageBox.Show("Subject Sr. No. already exists in this Academic Year!");

txtSubject\_Name.Focus();

}

break;

}

}

**10. Testing**

**10.1 Testing**

* Testing a piece of code consists of subjecting the code to a set of test inputs (or test ‘cases) and observing if the piece of code behaves as expected or not.
* If the code fails to behave as expected, then the conditions under which failure occurs are noted for later debugging and correction.
* Thus testing provides a practical way of reducing defects in a system and increasing the user’s confidence in a system.
* The main objectives of testing are as follows:
* To make sure that the program during operation will perform as per the specifications lay down at the time of system analysis.
* To ensure that the control incorporated in the program and the system functionality are significant.
* To make sure that during the operation, incorrect inputs and processing will be detected and handled.
  1. **Testing the Project:**

1. **Data integrity through constraints:**

A constraint refers to a condition or a check that is applied to a column (field) or set of columns in a table. The constraints applied to maintain data integrity are also known as **Integrity Constraints**.

These constraints ensure database integrity, thus are sometimes called database integrity constraints. A few of them used in the project are as:

1. **Primary Key Constraint:**

This constraint declares a column as the primary key of the table. This constraint is similar to unique constraint except that only one column

(Or one group of columns) can be applied in this constraint. The primary key does not allow NULL values.

PRN\_No, Staff\_Id,Sr\_No+Academic\_Year+Class+Semester, PRN\_No+Seat\_No are the primary keys which avoids duplication of rows and does not allow null values.

**b) Foreign Key Constraint:**

Whenever two tables are related by a common column (or set of columns) then the related columns(s) in the parent table (or primary table) should be either declared as a primary key or unique key and the related column(s) in the child table (or related table) should have Foreign Key Constraint.

For instance,

PRN\_No is acting as a foreign key in FE\_Marks,SE\_Marks,TE\_Marks&BE\_Markswhich references the PRN\_No of the student table which is acting as the primary key.

1. **Unique Constraint:**

This constraint ensures that no two rows have the same value in the specified column(s).

For instance,

Unique constraint has been applied on the Staff\_Id number of the customer as no two Staff \_Idnumbers can ever be identical.

1. **NOT NULL Constraint:**

By default tables can contain NULL values. By applying the NOT NULL constraint on a particular column(s), it becomes mandatory to enter value for that column(s).

For instance,

NOT NULL has been enforced on thePRN\_No,Student\_Name, Admission\_Type&Academic\_Year of the student table.

1. **Check Constraint:**

This constraint limits values that can be inserted into a column of a table.

For instance,

The check constraint has been applied in order to ensure that the students’admission type can be only ‘F.E.’, ‘D.S.E.’.

1. **Validating the tables:**

Data entered through tables is added in accordance with the constraints applied.

For example,

Entering only numeric values for usermobile number.

1. **Checking of Errors:**

Different command buttons and toolbar buttons are tested so as to check whether it performs the correct intended function or not. Combo boxes and list boxes are checked so as to know whether all the options are present or not. Common issues like submission of forms without filling in the required entries are checked.

1. **Unique Constraint Violated:**

The value being depicted as primary key cannot be NULL and redundant.

Hence while inserting data, it is made sure that each and every customer is assigned a unique customer\_idandauniqueemployee\_id for an employee working in the service centre. It should also be made sure that the appropriate data is entered for fields not allowing NOT NULL values.

1. **Parent Key Not found:**

The parent key should be present in the parent table and then only the foreign key value(s) should be inserted in the child table.

For instance,

It is made sure that, details of a student have been saved prior to entering his/her marks in system.

**Conclusion**

The project on a **Result Analysis for NBA** is for computerizing the working in a college.

The software takes care of all the requirements of an average college and is capable to provide easy and effective storage of information related to students’ marks that come up to the college.This software makes easy to analyse students’ performance throughout engineering.This software can be used for any engineering college for Result Analysis of NBA as it is made very flexible.

**Bibliography**

1. Silberschatz A., Korth H., Sudarshan S., "Database System Concepts", 6thEdition, McGraw Hill Publishers, ISBN 0-07-120413-X
2. S.K.Singh, “Database Systems : Concepts, Design and Application”, 2nd Edition, Pearson, 2013, ISBN 978-81-317-6092-5
3. Connally T., Begg C., "Database Systems", 3rd Edition, Pearson Education, 2002, ISBN 81-7808861-4
4. Kristina Chodorow, Michael Dirolf, “MongoDB: The Definitive Guide” ,O’Reilly Publications
5. [www.w3schools.com](http://www.w3schools.com)
6. [www.mongodb.org](http://www.mongodb.org)
7. <http://www.c-sharpcorner.com/>
8. [https://**msdn**.microsoft.com/](https://msdn.microsoft.com/)
9. <http://www.codeproject.com/>
10. <https://msdn.microsoft.com/en-us/library/vstudio/cc148994(v=vs.100).aspx>