**Chapter 1: Introduction**

* 1. **Aim of project**

Aim of this project is to build a web application for college department to manage all the information of activities done by Students, Faculties and Head of Department using MVC Laravel Framework.

* 1. **Project Scope**

Department Activity Portal will be used for all departments of the college.

* 1. **Project Objective**

The main objective is to automate the existing system of manually maintaining the records of students and faculty activities, to increase data accuracy, and to make activities information management more secure, effective, convenient and accessible.

**1.4 Project Modules:**

1. **Registration and Login**

1.1 Student Registration/Login

1.2 Faculty Registration/Login

1.3 Admin Registration/Login

1. **User Profile Management**

2.1 Add and Manage Personal Details

2.2 Verification of Accounts (by Admin only)

2.3 Insert Students Results (by Counselor only)

1. **Activity Management**

3.1 Add and Manage Activities Details (by Student and Faculty only)

* Attended and Organized Activities Details
* Seminar
* Workshop
* Expert Talks
* STTP
* Conference
* Industrial Visit
* Published Papers and Books Details
* Training and Internship Details
* R&D Projects Details
* Services inside and outside Institute Details

1. **Report Generation**
   1. Generate Personal Reports (by Student and Faculty only)

* Academic Year wise report
* Activity wise report
  1. Generate Faculty and Students Reports (by Counselor and Admin only)
* Academic Year wise report
* Activity wise report
* Faculty wise report
* Student wise report

**1.5 Basic Requirements:**

* Operating system: Windows 7 or newer
* Minimum 1.6 GHz processor
* Minimum 1 GB of RAM
* Minimum 20 GB hard disk space
* SQL Server
* Network connectivity

**Chapter 2: Analysis, Design Methodology and Implementation Strategy**

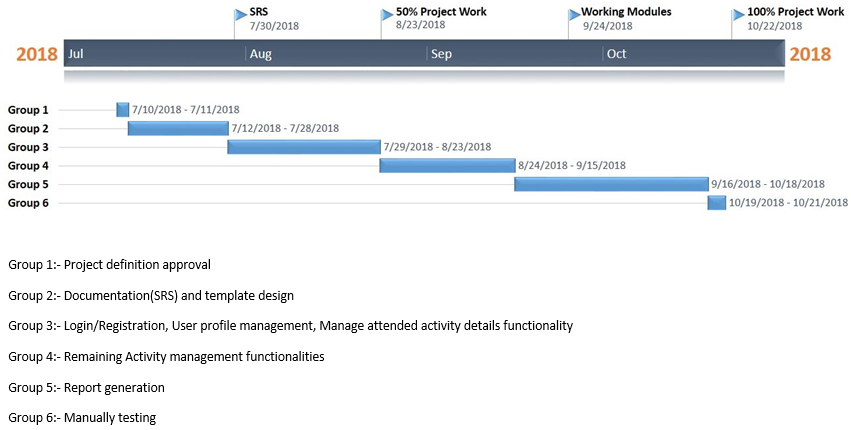
**2.1 Comparison of existing system with our project**

* In existing system information is collected and entered manually from students and faculties. While in our application students and faculties enters and manages their details by themselves.
* Our application gives more accurate report generation time in too much less time as compared to existing system.
* In existing system data redundancy is too much. While in our application, no data redundancy is there because database is normalized.

**2.2 Project Feasibility Study**

1. **Technical Feasibility:** The system is to be built in Laravel Framework which is open-source, can be easily installed in existing servers. The support of the classes and objects and rich data types in Laravel Framework ensures the safeguard for high performance and security of the system; therefore, this system is economically feasible.
2. **Economic Feasibility:** The work for the project can be undertaken with the existing computers and servers, software technologies. Also Laravel framework is free and open-source. So we don’t need to buy any license. It is also cost effective in the sense that has eliminated the paper work completely; therefore, this system is economically feasible.
3. **Operational Feasibility:** This system is only information management system, which needs small amount of resources. Existing computer can meet the conditions both in hardware and software; therefore, this system is feasible in operation.

**2.3 Project Timeline Chart**

****

**Figure 1. Project Timeline Chart**

**2.4 Detailed Module Description**

**Students:**

1. **Login/Registration**
2. **Manage personal details**

Student can add/modify their basic personal details like ID number, Enrollment number, Name, Email, Contact number, Branch, Profile image etc.

1. **Manage conducted and organized activities details**

Student can add/modify details of attended and conducted activities details like workshops, seminars, expert talks, Training and Internship, achievements, etc.

1. **Generate personal report**

Student generate and download report of personal activities.

**Faculties:**

1. **Login/Registration**
2. **Manage personal details**

Faculty can add/modify their basic personal details like Name, Email, Contact number, Branch, etc.

1. **Manage conducted and organized activities details**

Faculty can add/modify details of attended and conducted activities details like workshops, seminars, expert talks, STTP, Training and Internship, achievements, etc.

Also they can add/modify their published research papers details.

1. **Generate personal report**

Faculty can generate and download report of personal activities.

1. **Generate students report (for Counselor only)**

Faculty can generate and download report of students under their counselling.

**Admin:**

1. **Login/Registration**
2. **Manage personal details**

Admin can add/modify their basic personal details Name, Email, Contact number, Branch, Profile image etc.

1. **Verify students and Faculties Accounts**

When new student or faculty register into the system, admin can verify their accounts. Only admin verified accounts are allowed to add/manage their activities details.

Also, admin can assign faculty as a counselor for students’ batches.

1. **Generate students report**

Admin can generate and download report of students.

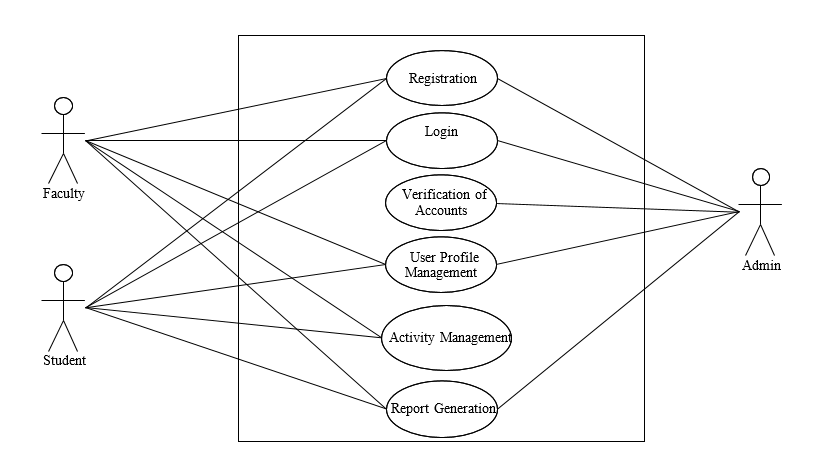
1. **Generate faculties report**

Admin can generate and download report of faculties.

.

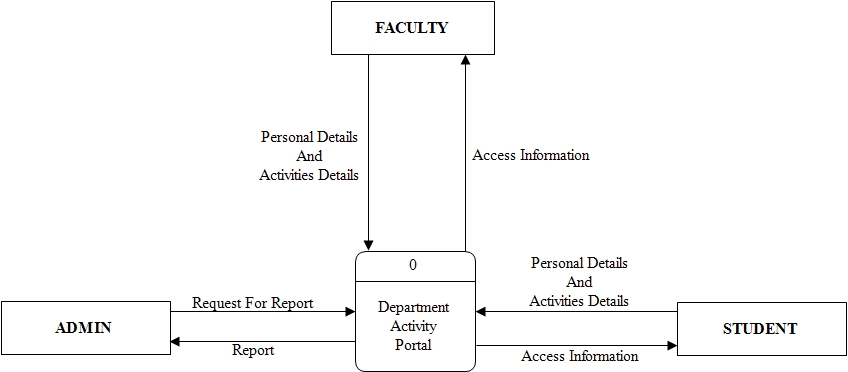
**2.5 Project SRS**

**2.5.1 Use Case Diagram**

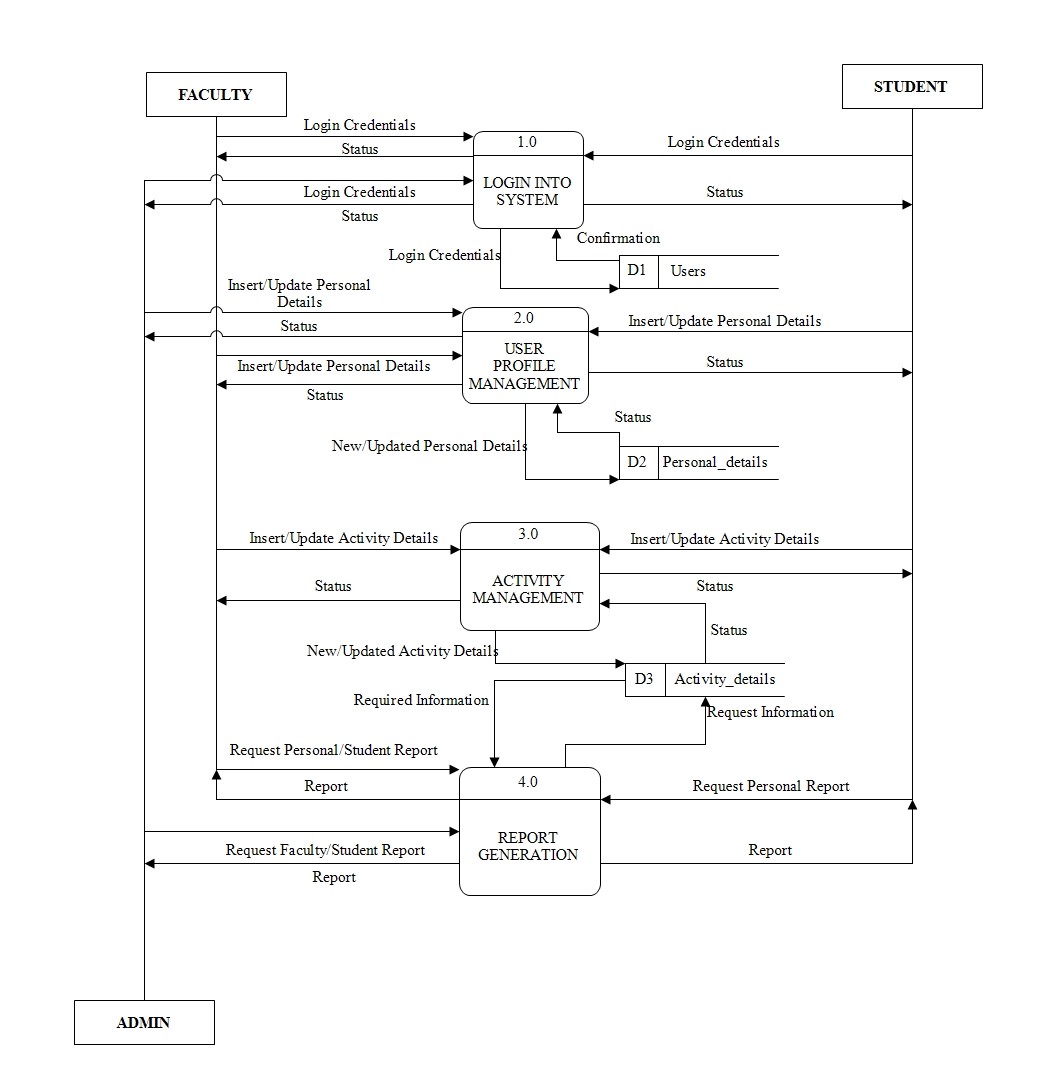
****

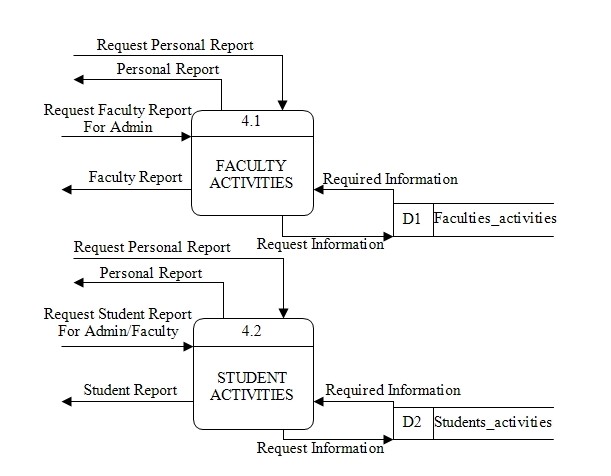
**Figure 2. Use Case Diagram**

**2.5.2 Data Flow Diagrams**



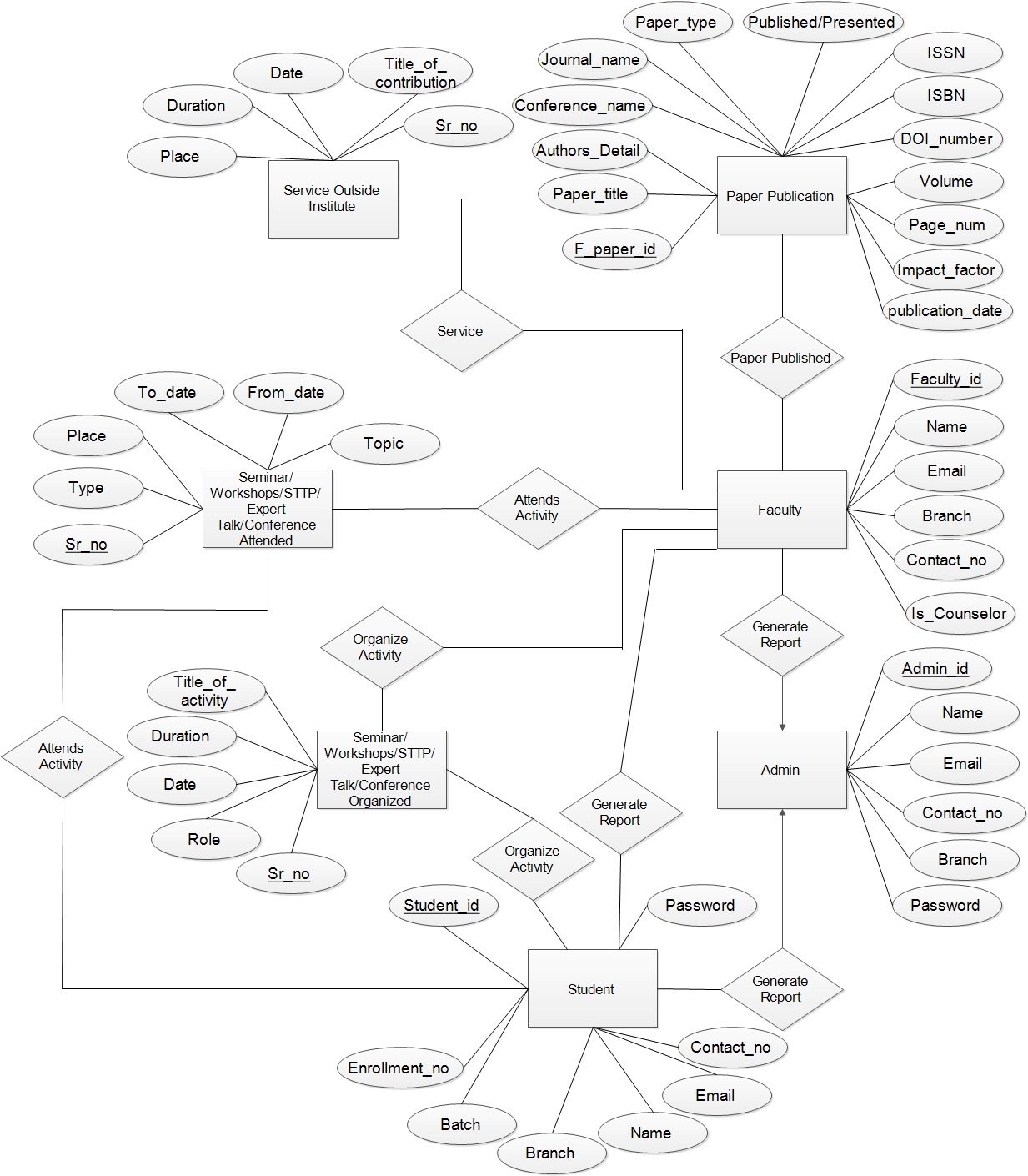
**Figure 3.Level 0 Data Flow Diagram**

**Figure 4.Level 1 Data Flow Diagram**

****

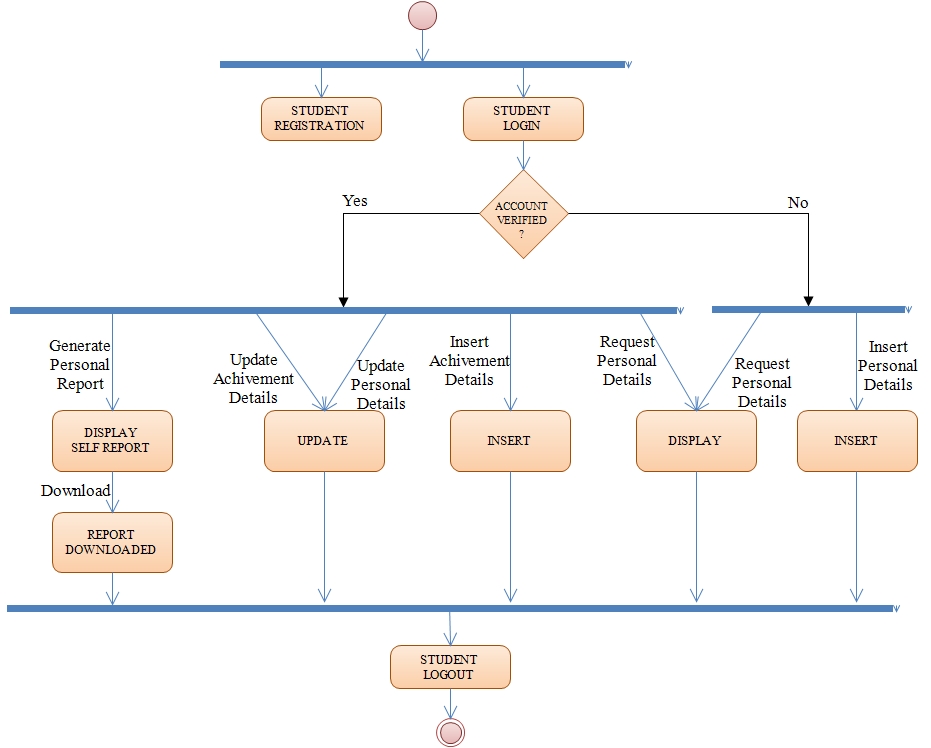
**Figure 5.Level 2 Data Flow Diagram**

**2.5.3 Entity Relationship Diagram**

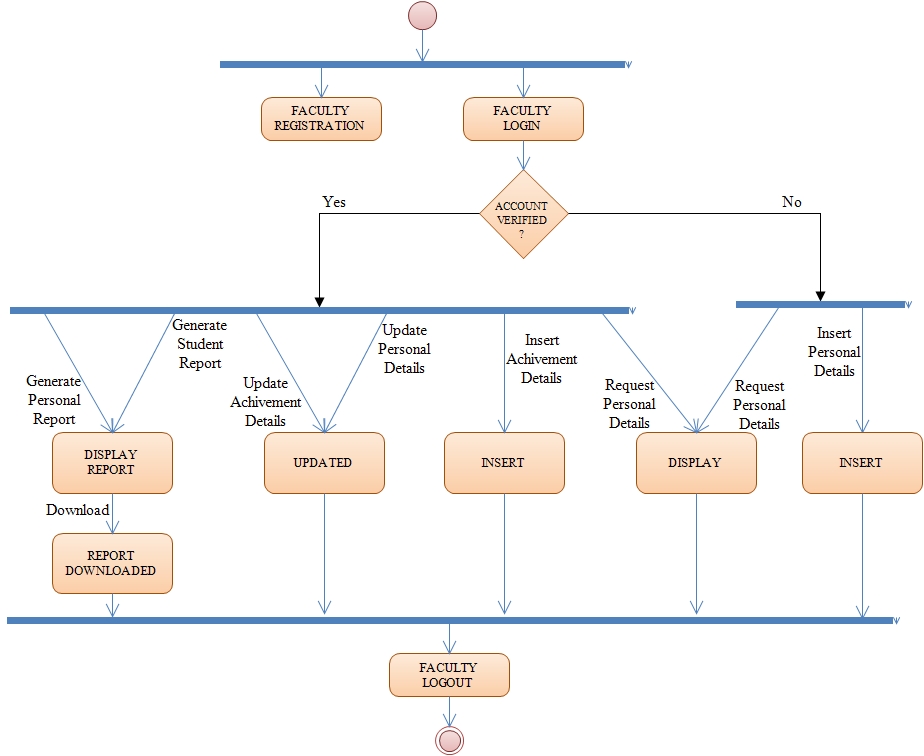
****

**Figure 6. Entity Relationship Diagram**

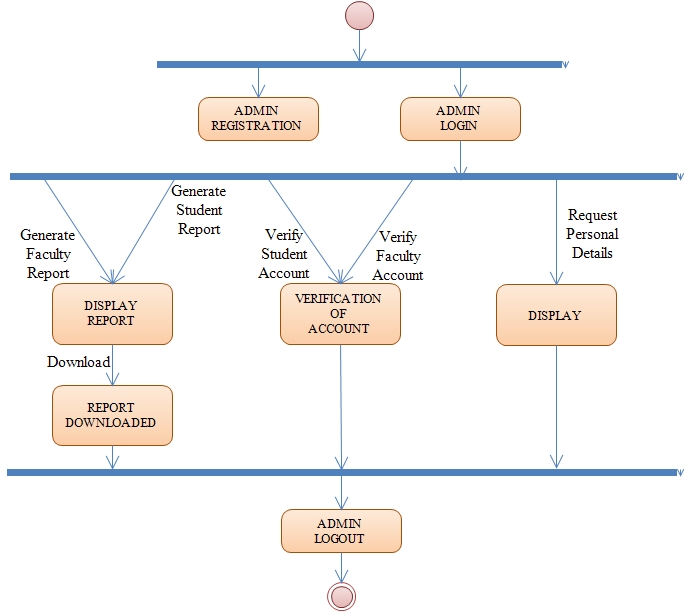
**2.5.4 State Diagram**

****

**Figure 7. State Diagram (student)**

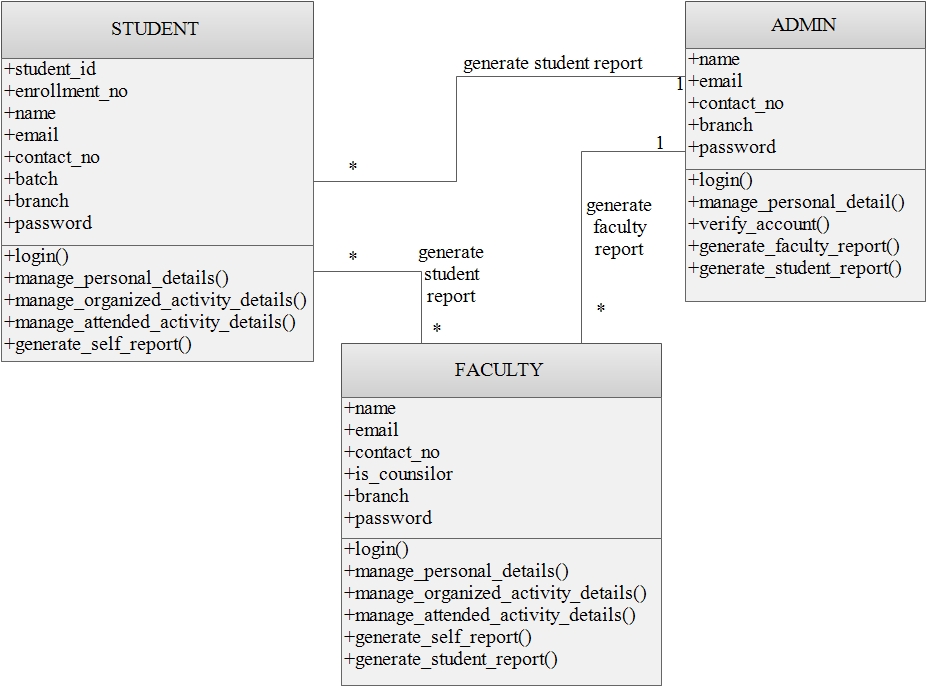
****

**Figure 8. State Diagram (faculty)**

****

**Figure 9. State Diagram (admin)**

**2.5.5 Class Diagram**

****

**Figure 10. Class Diagram**

**2.6 Database design and Normalization**

**What is Database Design?**

Database design is the process of producing a detailed data model of a database. This data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a data definition language, which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity.

**Database Design for Department Activity Portal:-**

Database name: activity\_portal

No. of tables: 14

**Database dictionary:-**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Name** | **Datatype** | **Constraints** | **Description** |
| 1 | student\_id | varchar(7) | PRIMARY\_KEY | To store the unique student id |
| 2 | batch | int(4) | NOT NULL | To store the student batch |
| 3 | branch | varchar(30) | NOT NULL | To store the student branch |
| 4 | name | varchar(40) | NOT NULL | To store the student name |
| 5 | profile\_image | varchar(100) | NOT NULL | To store the path of profile picture of student |
| 6 | password | varchar(191) | NOT NULL | To store the student password |
| 7 | is\_verifed | int(1) | DEFAULT "0" | To store whether student's account is verified or not |
| 8 | created\_at | timestamp | NOT NULL | To store the student profile creation time |
| 9 | updated\_at | timestamp | NOT NULL | To store the student profile updation time |

**Table 1. students**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Name** | **Datatype** | **Constraints** | **Description** |
| 1 | unique\_s\_id | int(10) | PRIMARY\_KEY, AUTO\_INCREMENT | To store the unique system generated student id |
| 2 | student\_id | varchar(7) | FOREIGN\_KEY | To store the unique student id |
| 3 | enrollment\_no | varchar(15) | UNIQUE | To store the student enrollment number |
| 4 | email | varchar(40) | UNIQUE | To store the student email id |
| 5 | contact\_no | int(10) | UNIQUE | To store the student contact number |
| 6 | created\_at | timestamp | NOT NULL | To store the record entry time |
| 7 | updated\_at | timestamp | NOT NULL | To store the record updation time |

**Table 2. student\_\_unique**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Name** | **Datatype** | **Constraints** | **Description** |
| 1 | faculty\_id | int(7) | PRIMARY KEY | To store the system generated faculty id |
| 2 | name | varchar(40) | NOT NULL | To store the faculty name |
| 3 | branch | varchar(30) | NOT NULL | To store the faculty branch |
| 5 | profile\_image | varchar(100) | NOT NULL | To store the path of profile picture of faculty |
| 6 | password | varchar(191) | NOT NULL | To store the faculty password |
| 7 | is\_verified | int(1) | DEFAULT "0" | To store whether student's account is verified or not |
| 8 | created\_at | timestamp | NOT NULL | To store the faculty profile created time |
| 9 | updated\_at | timestamp | NOT NULL | To store the faculty profile updated time |

**Table 3. faculties**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Name** | **Datatype** | **Constraints** | **Description** |
| 1 | f\_uniq\_details\_id | int(10) | PRIMARY\_KEY, AUTO\_INCREMENT | To store the system generated faculty unique details record id |
| 1 | faculty\_id | int(7) | FOREIGN\_KEY | To store the system generated faculty id |
| 2 | email | varchar(40) | UNIQUE | To store the faculty email |
| 3 | contact\_num | int(10) | UNIQUE | To store the faculty contact number |
| 4 | created\_at | timestamp | NOT NULL | To store the record entry time |
| 5 | updated\_at | timestamp | NOT NULL | To store the record updation time |

**Table 4. faculty\_unique**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Name** | **Datatype** | **Constraints** | **Description** |
| 1 | role\_id | int(10) | PRIMARY\_KEY, AUTO\_INCREMENT | To store the system generated faculty role id |
| 1 | faculty\_id | int(7) | FOREIGN\_KEY | To store the faculty id |
| 2 | is\_counselor | int(1) | DEFAULT "0" | To store whether the faculty is counselor or not |
| 3 | batch1 | int(4) | NULL | To store the current student batch 1 for which this faculty is counselor |
| 3 | batch2 | int(4) | NULL | To store the current student batch 2 for which this faculty is counselor |
| 3 | batch3 | int(4) | NULL | To store the current student batch 3 for which this faculty is counselor |
| 3 | batch4 | int(4) | NULL | To store the current student batch 4 for which this faculty is counselor |
| 6 | created\_at | timestamp | NOT NULL | To store the record entry time |
| 7 | updated\_at | timestamp | NOT NULL | To store the record updation time |

**Table 5. faculty\_role**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Name** | **Datatype** | **Constraints** | **Description** |
| 1 | admin\_id | int(10) | PRIMARY\_KEY | To store the system generated admin id |
| 2 | name | varchar(40) | NOT NULL | To store the admin name |
| 3 | branch | varchar(30) | NOT NULL | To store the admin branch |
| 4 | profile\_image | varchar(100) | NOT NULL | To store the path of profile picture of admin |
| 5 | password | varchar(191) | NOT NULL | To store the admin password |
| 6 | created\_at | timestamp | NOT NULL | To store the admin profile created time |
| 7 | updated\_at | int(1) | NOT NULL | To store the admin profile updated time |

**Table 6. users**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Name** | **Datatype** | **Constraints** | **Description** |
| 1 | ad\_uniq\_details\_id | int(10) | PRIMARY\_KEY, AUTO\_INCREMENT | To store the system generated admin unique details record id |
| 1 | admin\_id | int(10) | FOREIGN\_KEY | To store the system generated admin id |
| 2 | email | varchar(40) | UNIQUE | To store the admin email |
| 3 | contact\_num | int(10) | UNIQUE | To store the admin contact number |
| 4 | created\_at | timestamp | NOT NULL | To store the record entry time |
| 5 | updated\_at | timestamp | NOT NULL | To store the record updation time |

**Table 7. users\_unique**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Name** | **Datatype** | **Constraints** | **Description** |
| 1 | sr\_no | int(10) | AUTO\_INCREMENT, PRIMARY\_KEY | To store the auto incremented serial id |
| 2 | student\_id | varchar(7) | FOREIGN\_KEY | To store the student id |
| 3 | type | varchar(40) | NOT NULL | To store the attended activity type |
| 4 | topic | varchar(100) | NOT NULL | To store the topic of activity |
| 5 | place | varchar(20) | NOT NULL | To store the place of activity |
| 6 | from\_date | date | NOT NULL | To store the starting date of activity |
| 7 | to\_date | date | NOT NULL | To store the starting date of activity |
| 8 | file | varchar(100) | NOT NULL | To store the path of the certificate file |
| 9 | created\_at | timestamp | NOT NULL | To store the record entry time |
| 10 | updated\_at | timestamp | NOT NULL | To store the record updation time |

**Table 8. student\_attended**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Name** | **Datatype** | **Constraints** | **Description** |
| 1 | sr\_no | int(10) | AUTO\_INCREMENT, PRIMARY\_KEY | To store the auto incremented serial id |
| 2 | student\_id | varchar(7) | FOREIGN\_KEY | To store the student id |
| 3 | type | varchar(40) | NOT NULL | To store the type of activity organized |
| 4 | title\_of\_activity | varchar(100) | NOT NULL | To store the title of activity organized |
| 5 | place | varchar(20) | NOT NULL | To store the place of activity |
| 6 | from\_date | date | NOT NULL | To store the starting date of activity |
| 7 | to\_date | date | NOT NULL | To store the ending date of activity |
| 8 | role | varchar(50) | NOT NULL | To store the role of student for this ativity |
| 9 | file | varchar(100) | NOT NULL | To store the path of the certificate file |
| 10 | created\_at | timestamp | NOT NULL | To store the record entry time |
| 11 | updated\_at | timestamp | NOT NULL | To store the record updation time |

**Table 9. student\_organized**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Name** | **Datatype** | **Constraints** | **Description** |
| 1 | sr\_no | int(10) | AUTO\_INCREMENT, PRIMARY\_KEY | To store the auto incremented serial id |
| 2 | faculty\_id | int(7) | FOREIGN\_KEY | To store the faculty id |
| 3 | type | varchar(40) | NOT NULL | To store the attended activity type |
| 4 | topic | varchar(100) | NOT NULL | To store the topic of activity |
| 5 | place | varchar(20) | NOT NULL | To store the place of activity |
| 6 | from\_date | date | NOT NULL | To store the starting date of activity |
| 7 | to\_date | date | NOT NULL | To store the ending date of activity |
| 8 | file | varchar(100) | NOT NULL | To store the path of the certificate file |
| 9 | created\_at | timestamp | NOT NULL | To store the record entry time |
| 10 | updated\_at | timestamp | NOT NULL | To store the record updation time |

**Table 10. faculty\_attended**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Name** | **Datatype** | **Constraints** | **Description** |
| 1 | sr\_no | int(10) | AUTO\_INCREMENT, PRIMARY\_KEY | To store the auto incremented serial id |
| 2 | faculty\_id | int(7) | FOREIGN\_KEY | To store the faculty id |
| 3 | type | varchar(40) | NOT NULL | To store the type of activity organized |
| 4 | title\_of\_activity | varchar(100) | NOT NULL | To store the title of activity organized |
| 5 | place | varchar(20) | NOT NULL | To store the place of activity |
| 6 | from\_date | date | NOT NULL | To store the starting date of activity |
| 7 | to\_date | date | NOT NULL | To store the ending date of activity |
| 8 | role | varchar(50) | NOT NULL | To store the role of faculty for this ativity |
| 9 | file | varchar(100) | NOT NULL | To store the path of the certificate file |
| 10 | created\_at | timestamp | NOT NULL | To store the record entry time |
| 11 | updated\_at | timestamp | NOT NULL | To store the record updation time |

**Table 11. faculty\_organized**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Name** | **Datatype** | **Constraints** | **Description** |
| 1 | f\_paper\_id | int(10) | AUTO\_INCREMENT, PRIMARY\_KEY | To store the auto incremented record id |
| 2 | faculty\_id | int(7) | FOREIGN\_KEY | To store the faculty id |
| 3 | pub\_unique\_id | int(7) | FOREIGN\_KEY | To store the publication unique id |
| 4 | paper\_title | varchaar(200) | NOT NULL | To store paper title |
| 5 | title\_of\_contribution | varchar(40) | NOT NULL | To store the title of contribution |
| 6 | conference\_name | varchar(20) | NOT NULL | To store the conference name |
| 7 | journal\_name | varchar(20) | NOT NULL | To store the journal name |
| 8 | paper\_type | varchar(35) | NOT NULL | To store the paper type (international or national) |
| 9 | published/presented | varchar(10) | NOT NULL | To store whether the paper is published or presented |
| 10 | volume\_and\_issue | int(10) | NOT NULL | To store the number of papers published |
| 11 | page\_num | int(10) | NOT NULL | To store the number of page |
| 12 | impact\_factor | int(10) | NOT NULL | To store the impact effect |
| 13 | publication\_date | date | NOT NULL | To store the date of paper published |
| 14 | academic\_year | date | NOT NULL | To store the academic year |
| 15 | file | varchar(100) | NOT NULL | To store the path of the published paper file |
| 16 | created\_at | timestamp | NOT NULL | To store the record entry time |
| 17 | updated\_at | timestamp | NOT NULL | To store the record updation time |

**Table 12. paper\_publication**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Name** | **Datatype** | **Constraints** | **Description** |
| 1 | pub\_unique\_id | int(7) | AUTO\_INCREMENT, PRIMARY\_KEY | To store the auto incremented record id |
| 2 | ISSN | int(10) | NOT NULL | To store the ISSN number |
| 3 | ISBN | int(10) | NOT NULL | To store the ISBN number |
| 4 | DOI\_number | int(10) | NOT NULL | To store the DOI number |
| 5 | created\_at | timestamp | NOT NULL | To store the record entry time |
| 6 | updated\_at | timestamp | NOT NULL | To store the record updation time |

**Table 13. paper\_pub\_unique**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Name** | **Datatype** | **Constraints** | **Description** |
| 1 | sr\_no | int(10) | AUTO\_INCREMENT, PRIMARY\_KEY | To store the auto incremented serial id |
| 2 | faculty\_id | int(7) | FOREIGN\_KEY | To store the faculty id |
| 3 | title\_of\_contribution | varchar(100) | NOT NULL | To store the title of contribution |
| 4 | place | varchar(20) | NOT NULL | To store the place of activity |
| 5 | from\_date | date | NOT NULL | To store the starting date of activity |
| 6 | to\_date | date | NOT NULL | To store the ending date of activity |
| 7 | file | varchar(100) | NOT NULL | To store the path of the certificate file |
| 8 | created\_at | timestamp | NOT NULL | To store the record entry time |
| 9 | updated\_at | timestamp | NOT NULL | To store the record updation time |

**Table 14. service\_outside\_institute**

**Normalization:-**

What problems can be happens with the database?

-The table can be contain data redundancies.

-The table entries can be invite data inconsistencies.

The data redundancies yield the following anomalies:

-Update anomalies

-Deletion anomalies

Normalization is a formal process for assigning attributes to entities. It reduces and controls data redundancies. Helps eliminate data anomalies where insertions, deletions, updates destroy the integrity of the database. At the conclusion of this process the database will be logically correct.

Normalization works through a series of stages called normal forms.

1) First normal form (1NF)

2) Second normal form (2NF)

3) Third normal form (3NF)

4) Fourth normal form (4NF)

**1) First Normal Form (1NF):-**

A relational table must not contain repeating entries.

Repeating entries can be eliminated by adding at least the primary key column(s).

This leads to a much wider, longer table

Until 1NF is achieved we cannot create a relational database at all.

Since,

All key attributes are defined.

There are no repeating entries in the table.

All attributes are dependent on the primary key.

So, database is in 1NF.

**2) Second Normal Form (2NF):-**

A table is in 2NF if:

1. It is in 1NF,
2. It includes no partial dependencies; that is, no attribute is dependent on only a portion of the primary key.

Any table in database has no prime attribute that is dependent on any proper subset of any candidate key of the relation.

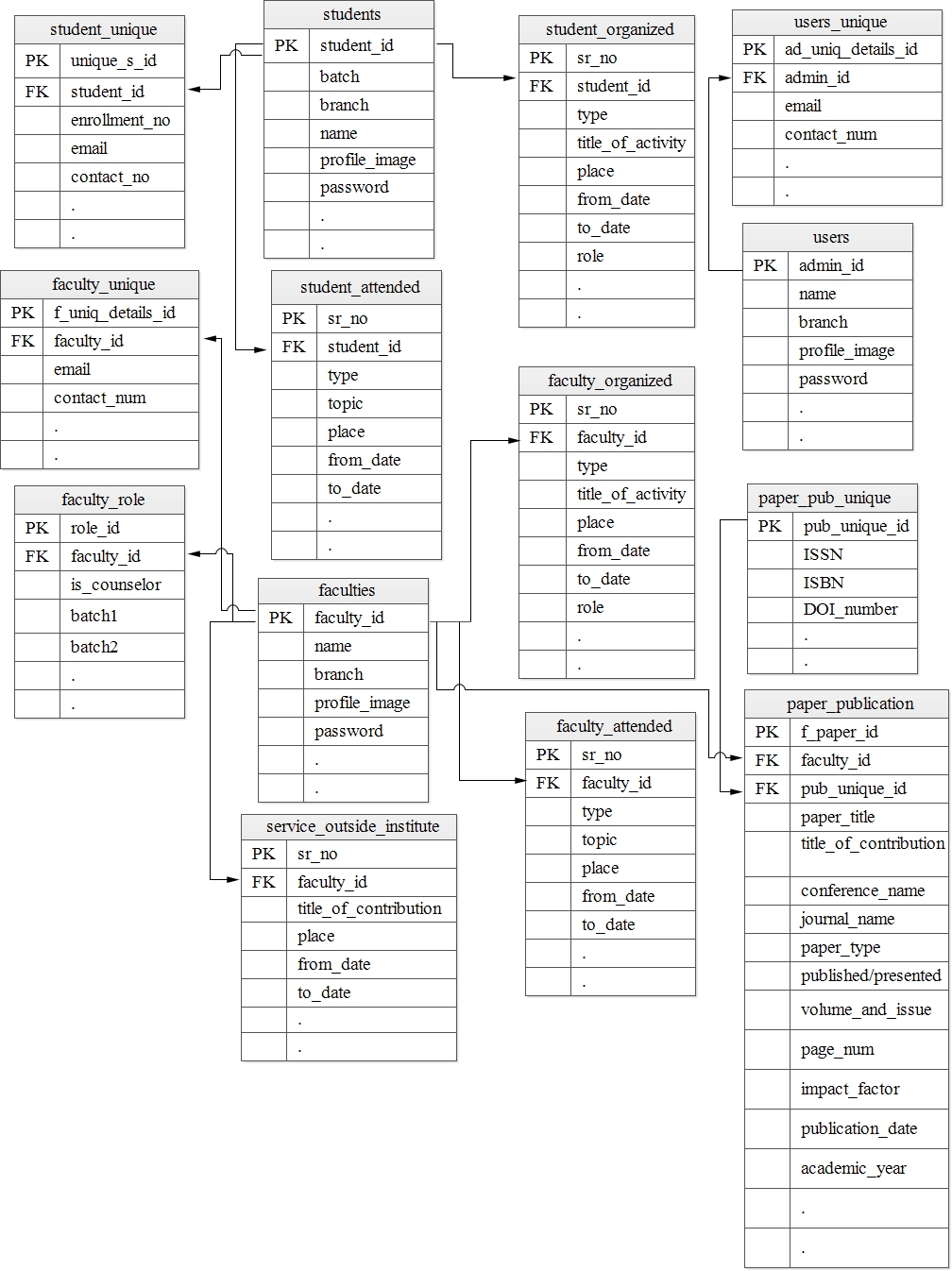
So, database is in 2NF.

**3) Third Normal Form (3NF):**

A table is in Third Normal Form (3NF) if:

1. It is in 2NF,
2. It contains no transitive dependencies.

Since any table in database contains no transitive dependencies our database is in 3NF.

1. **Database relation diagram**

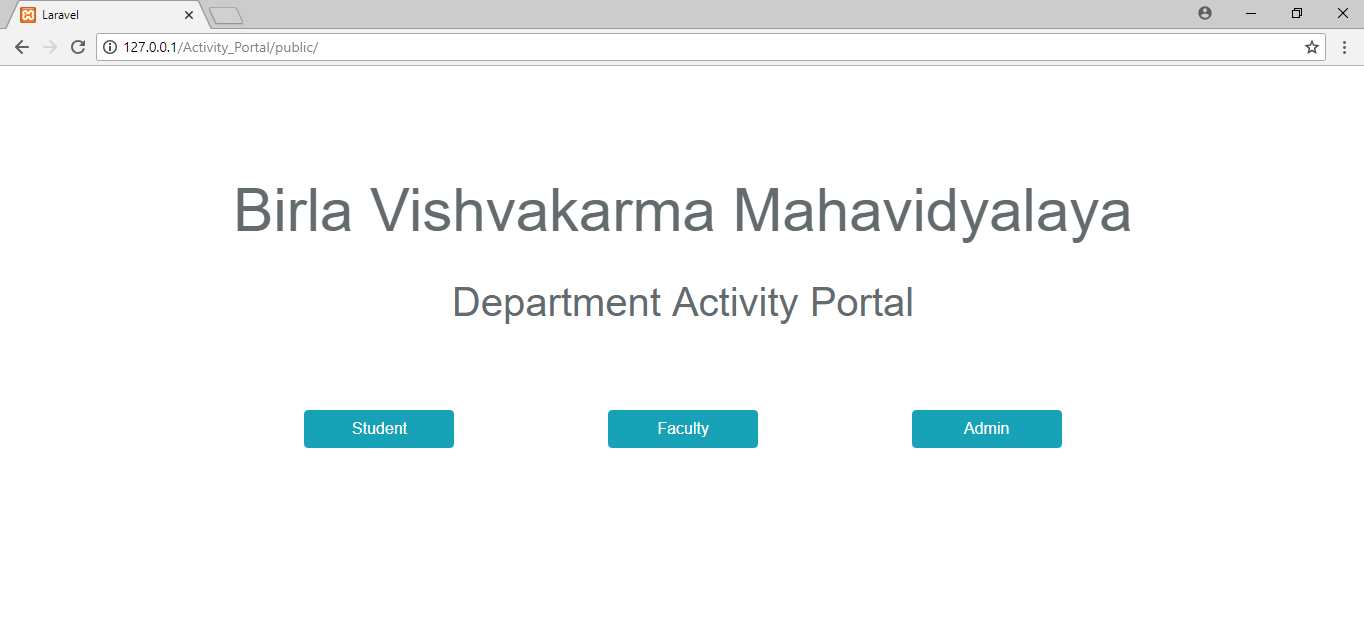
**Figure 11. Database Relation Diagram**

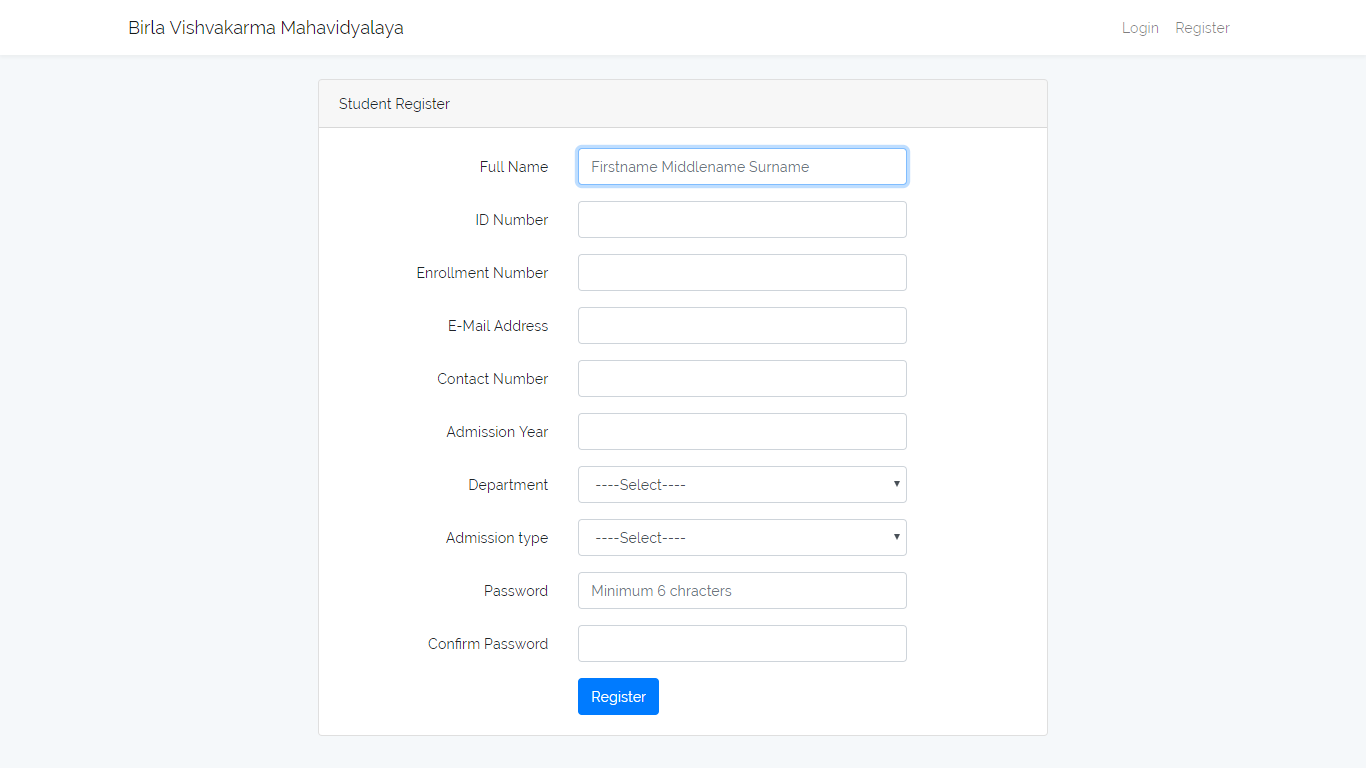
**Chapter 3: Implementation and Testing**

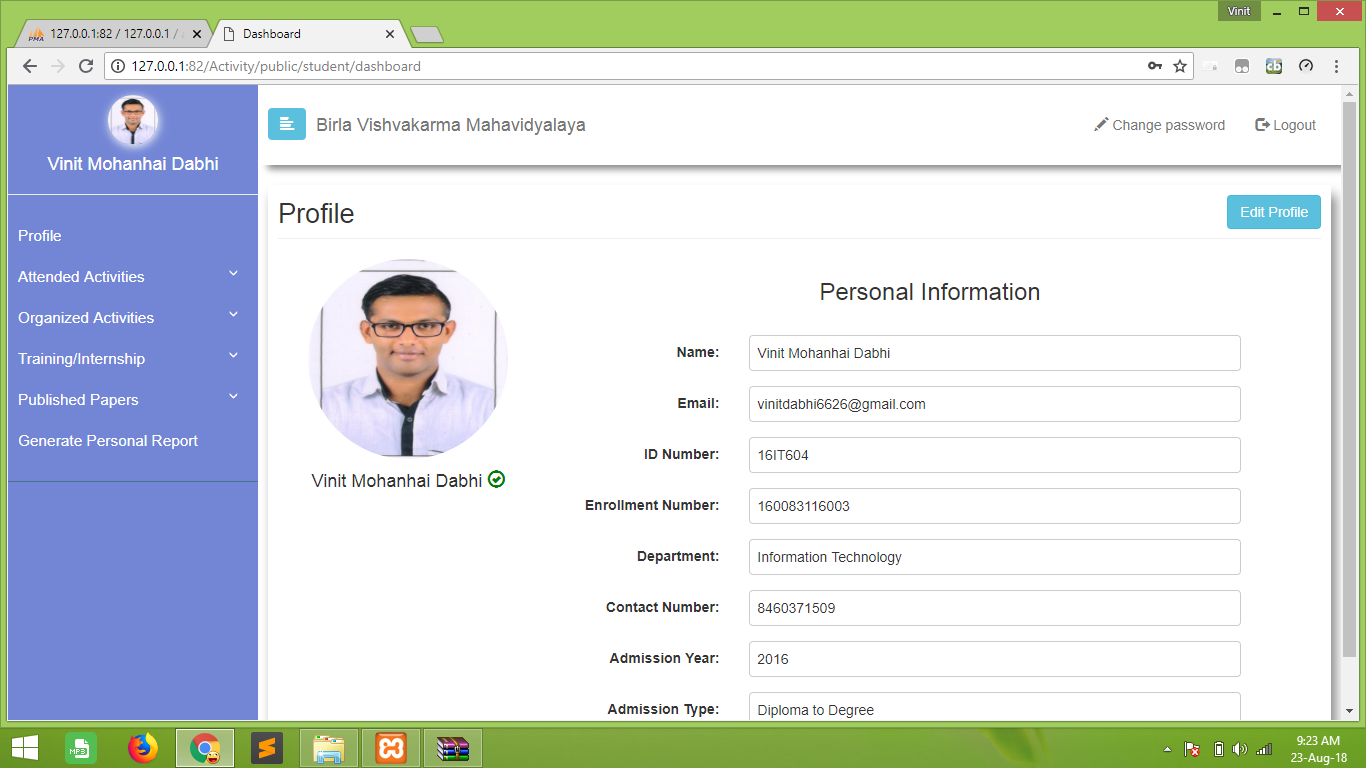
**3.1 Software and Tools**

1. SQL Server
2. Laravel Framework (v5.5)
3. PHP (v7.1.3)
4. Bootstrap (v3.3.7)
5. AngularJS (v1.6.4)
6. jQuery (v1.10.2)

**3.2 User Interface and Snapshots**

****

****

****