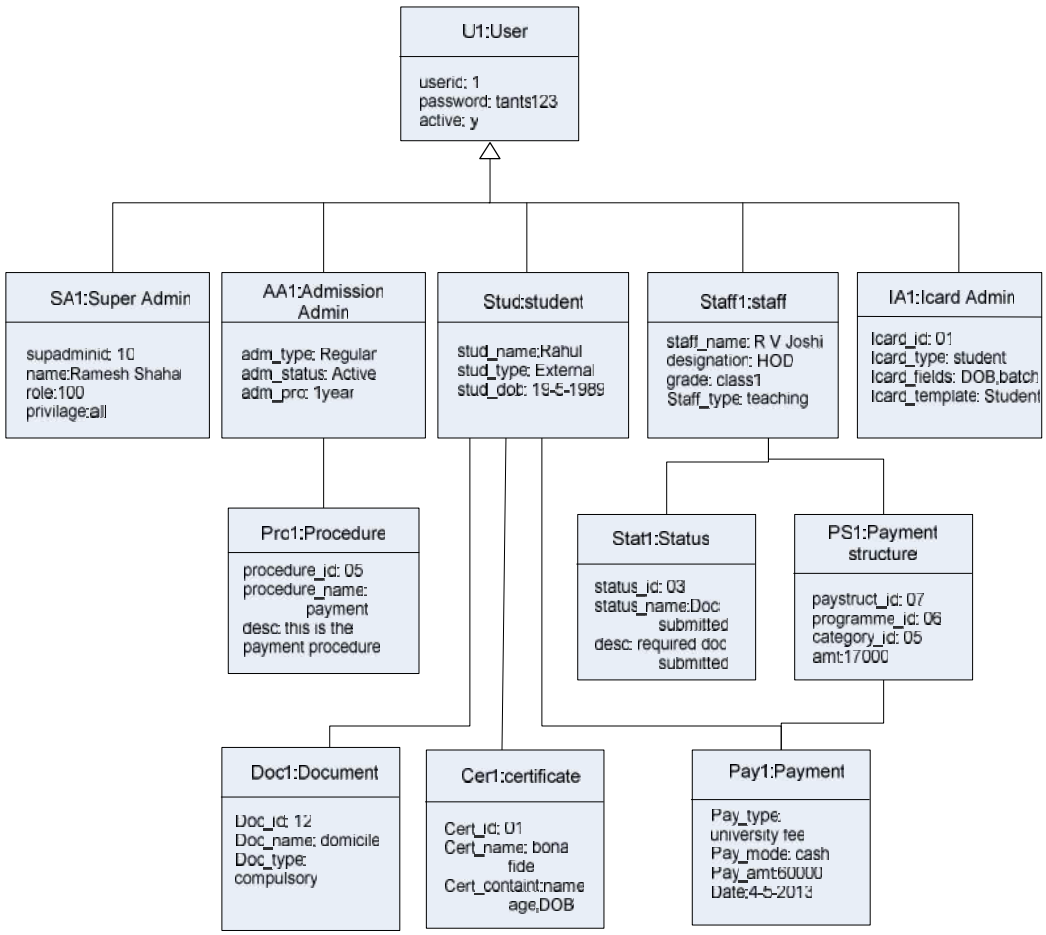
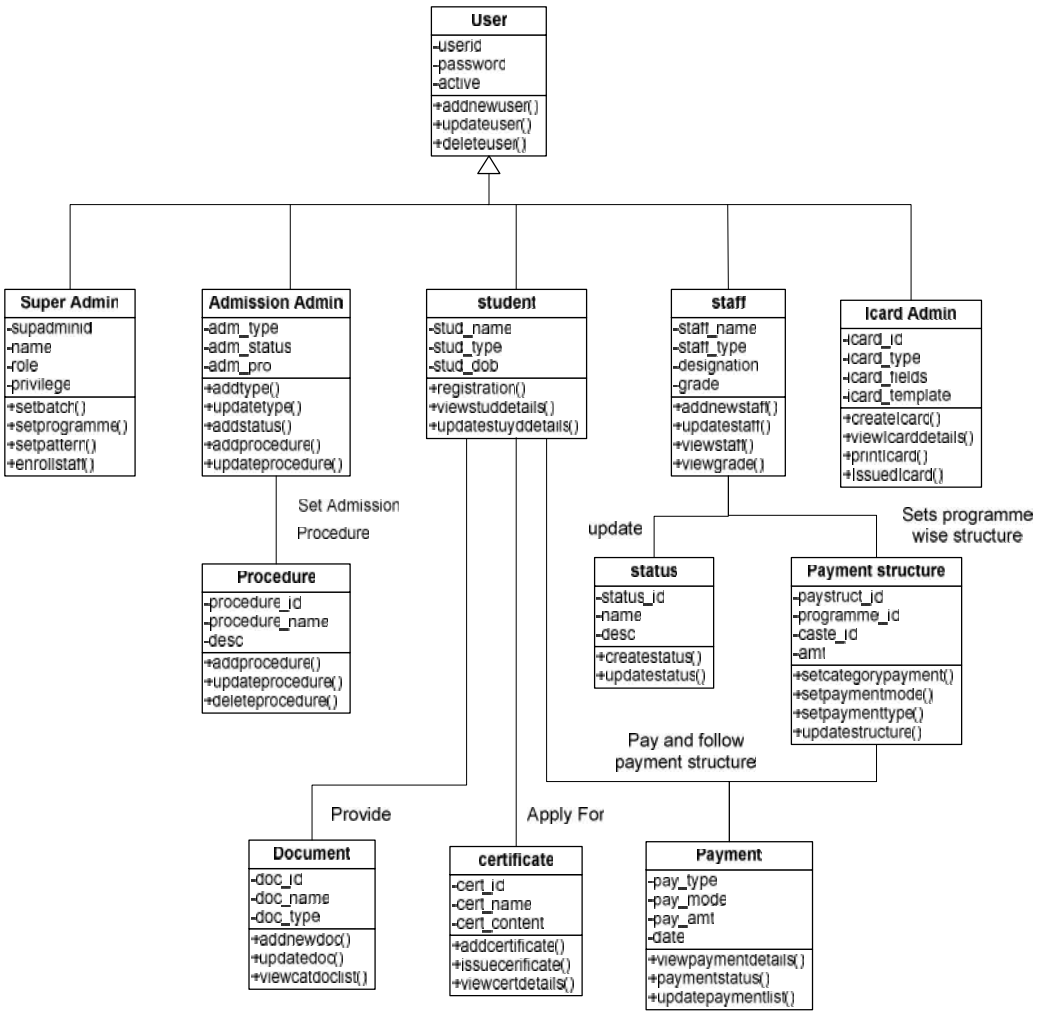
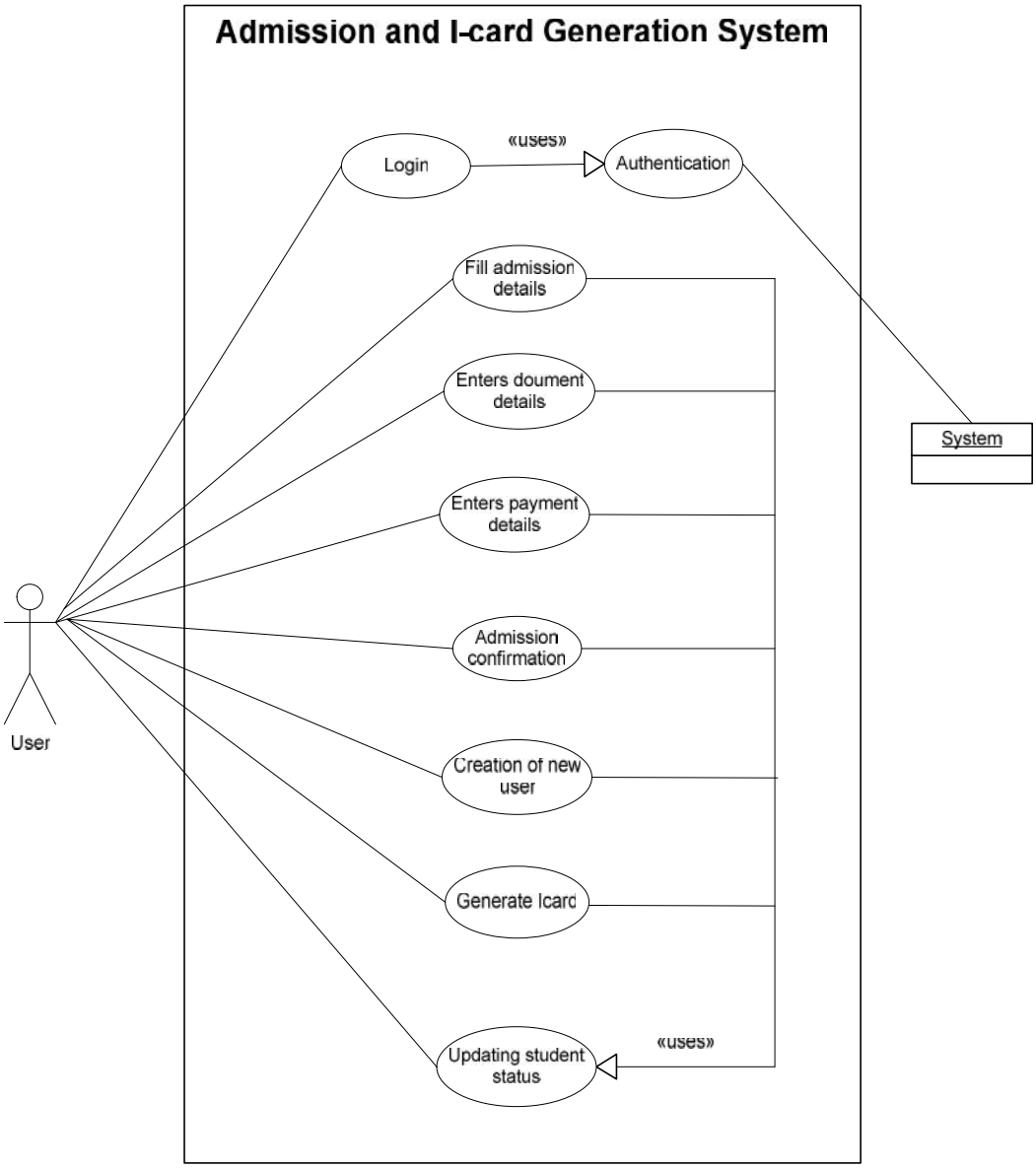
3.1 Object Diagram



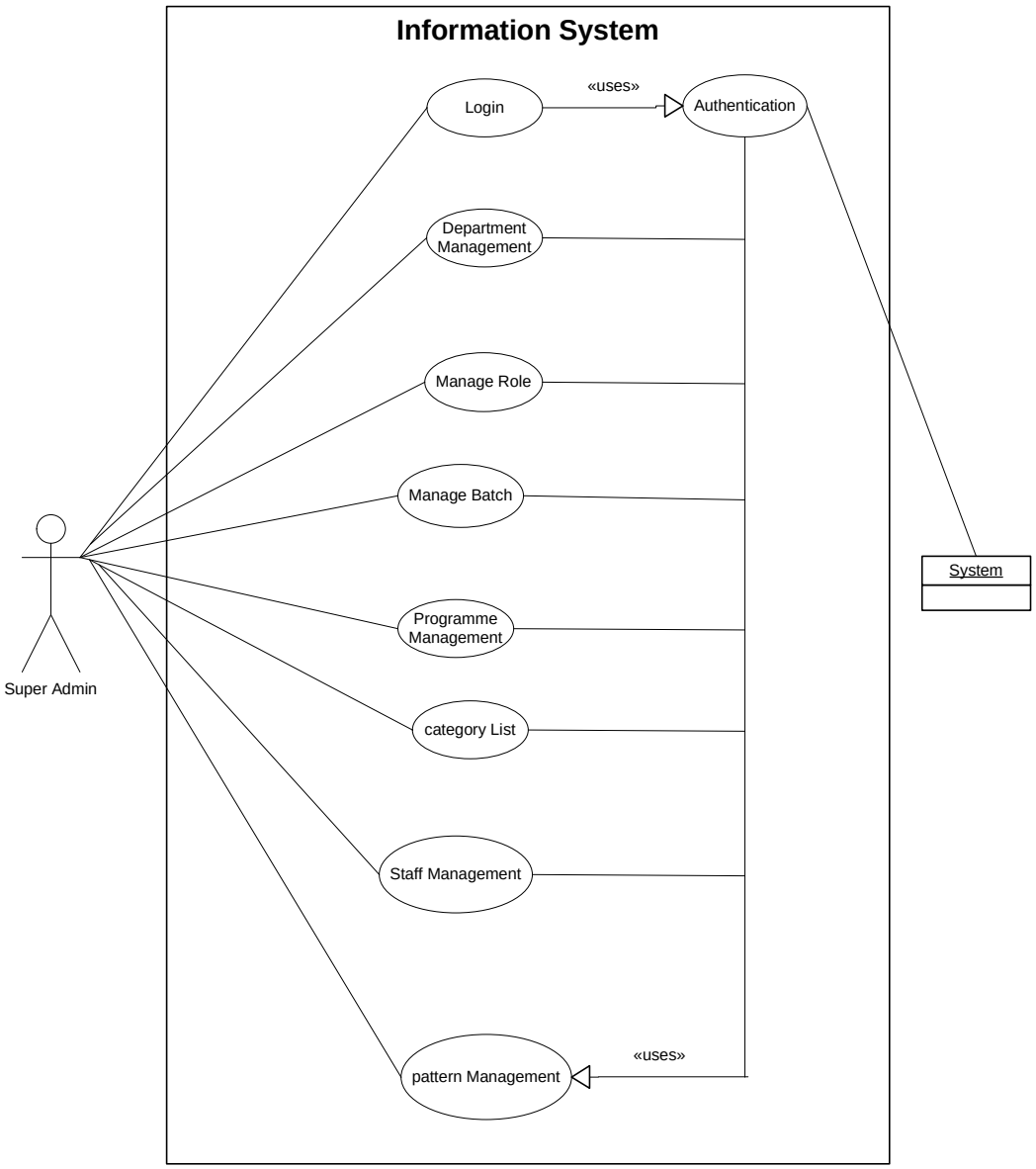
3.2 Class Diagram



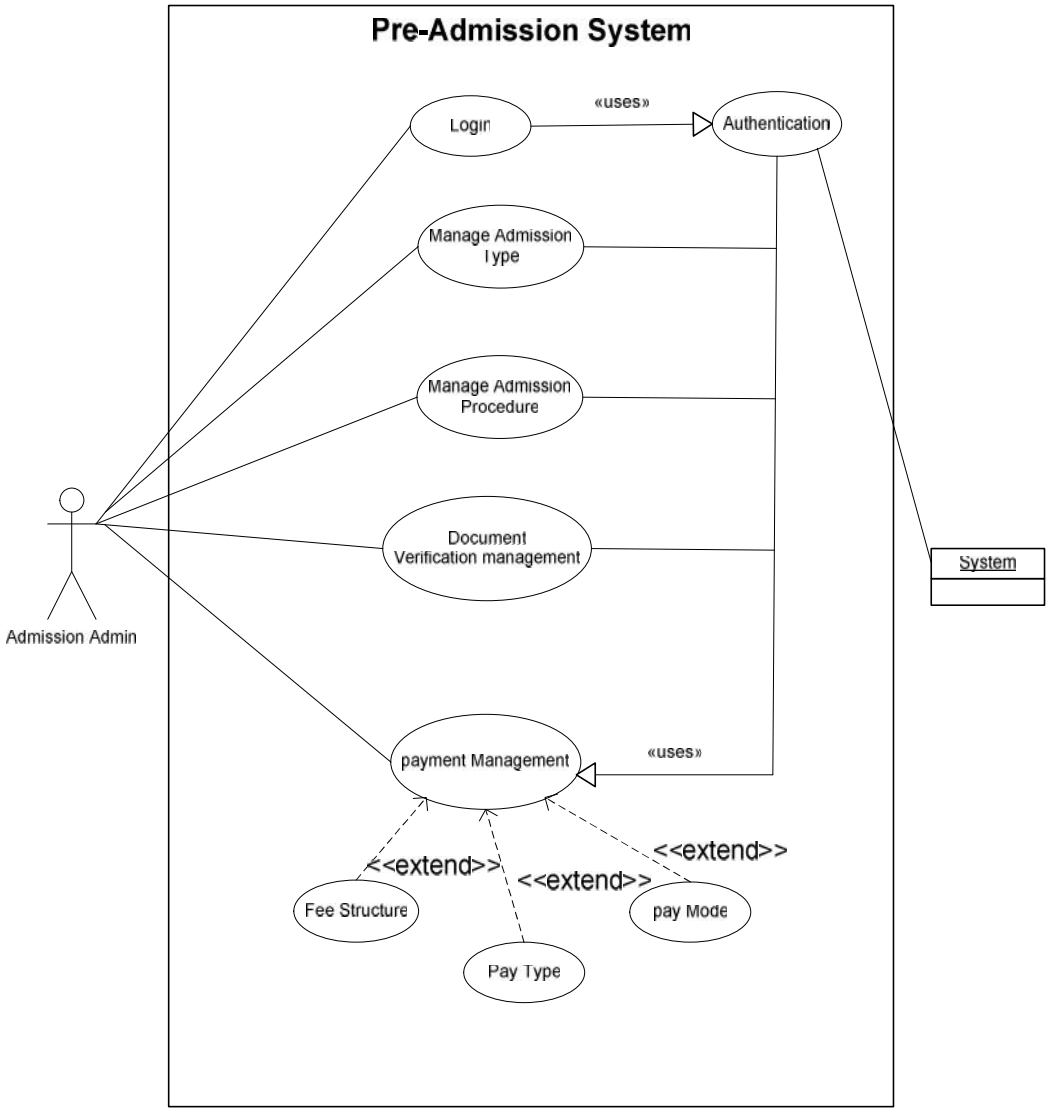
3.3 Use Case Diagrams (Business use case)



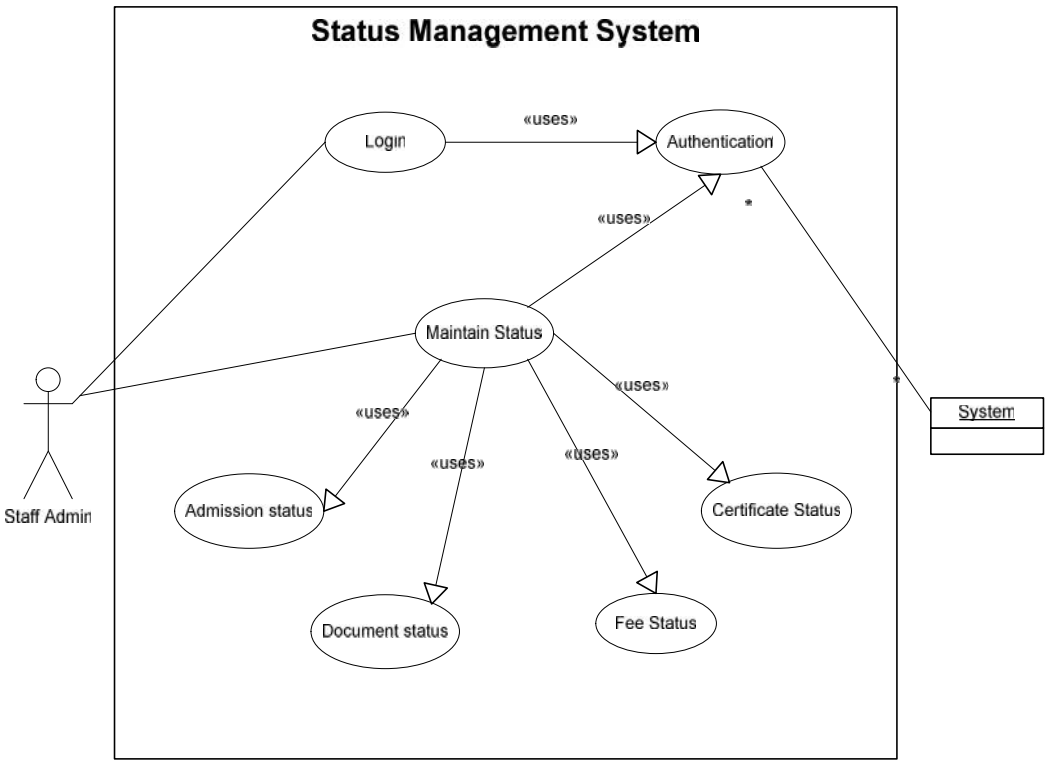
Use case for super admin



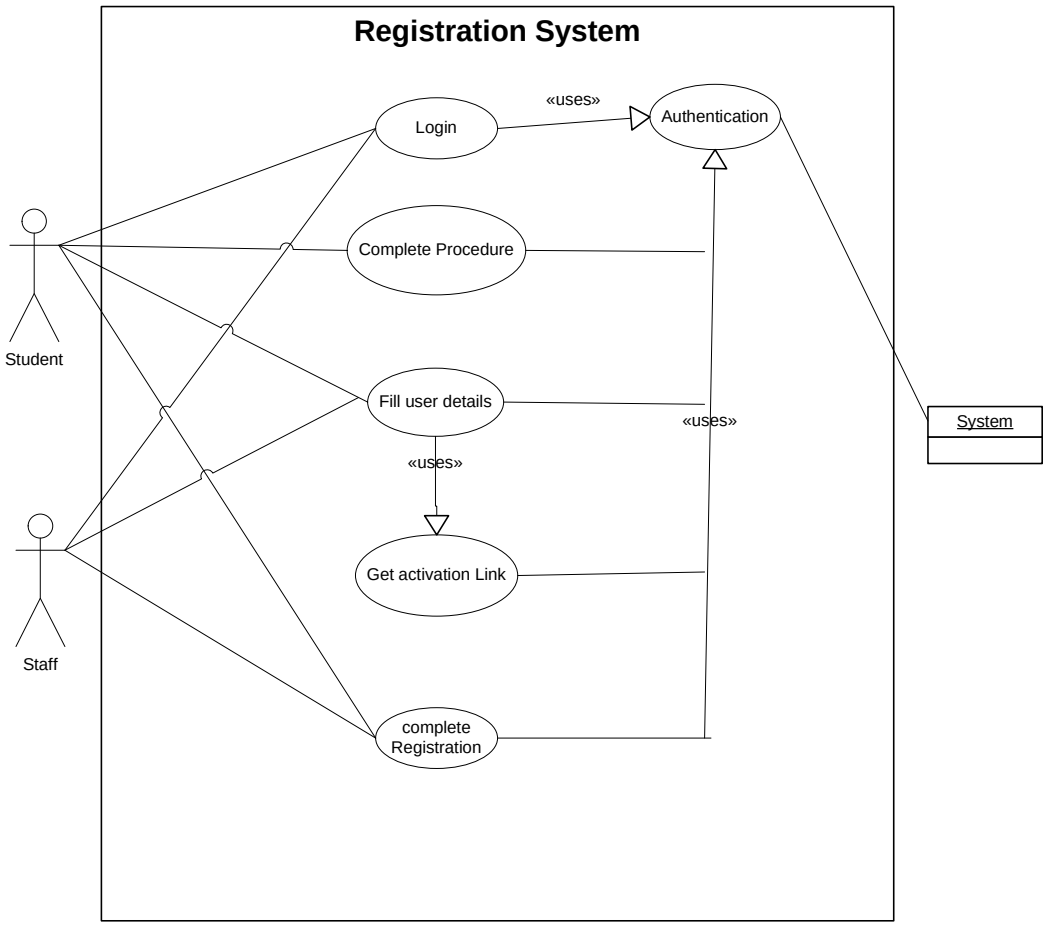
Use case for admission admin



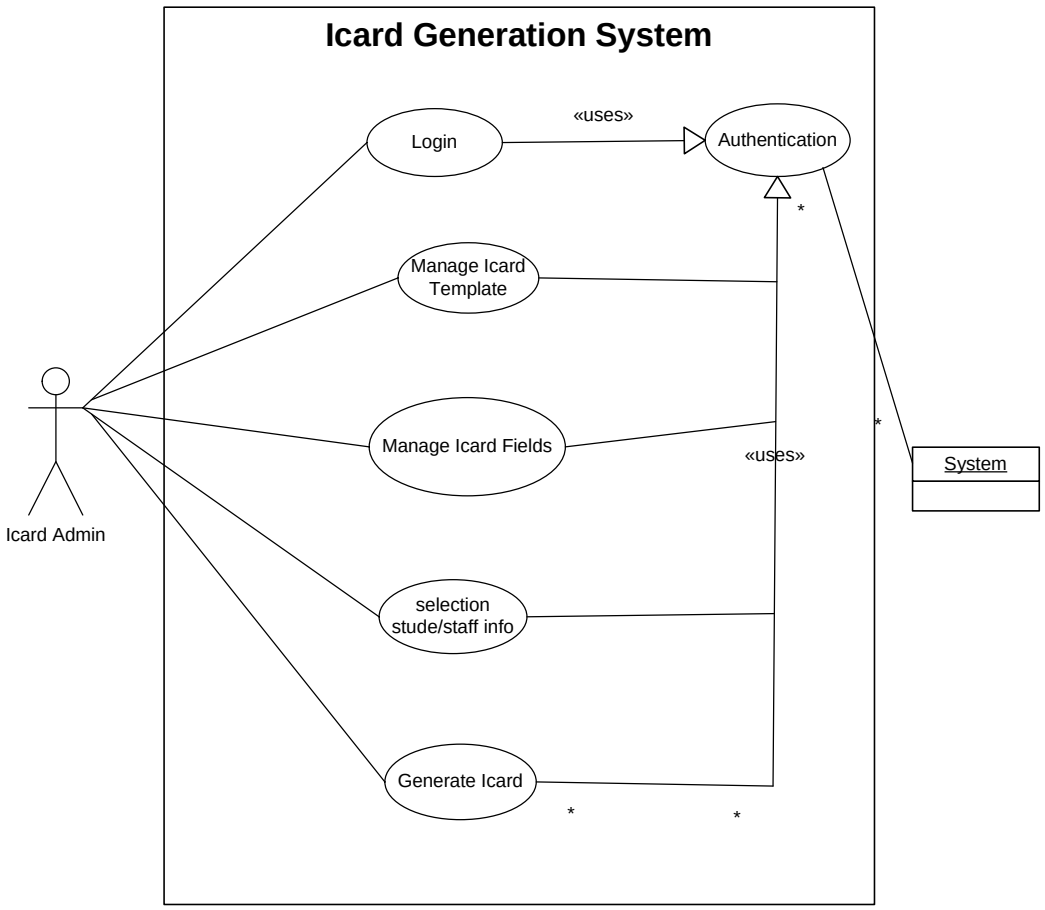
Use case for staff admin



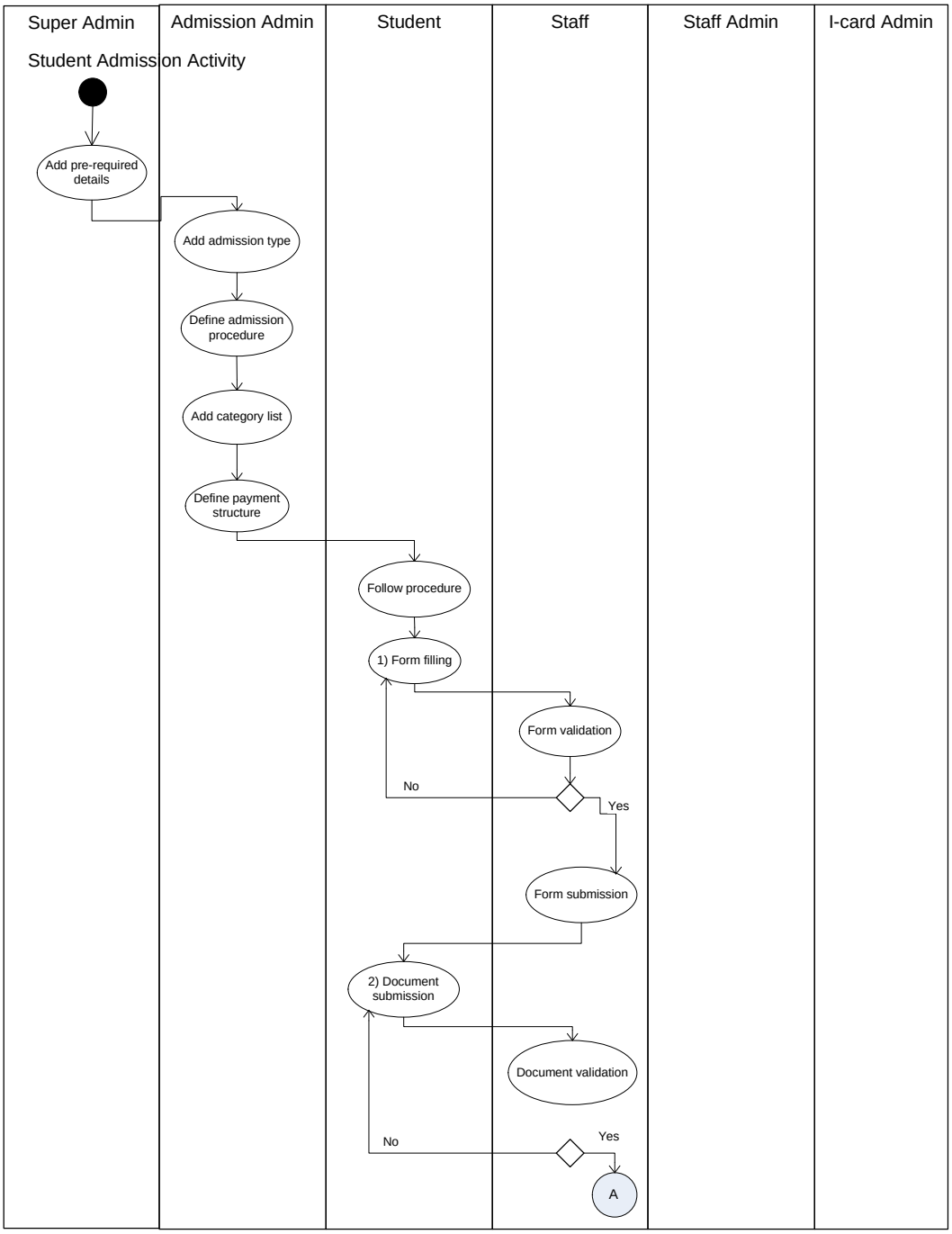
Use case for general user

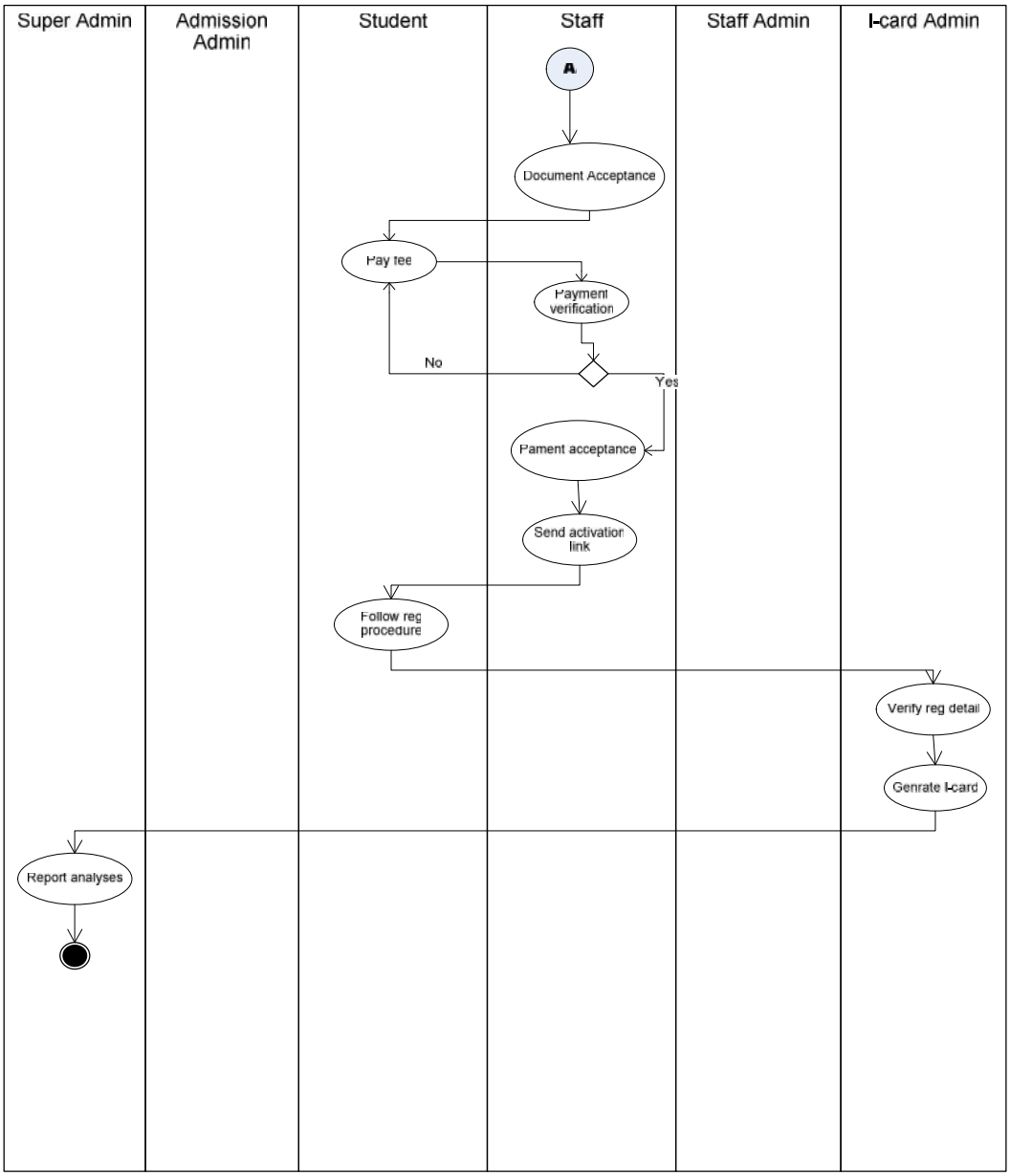


Use case for I-card admin

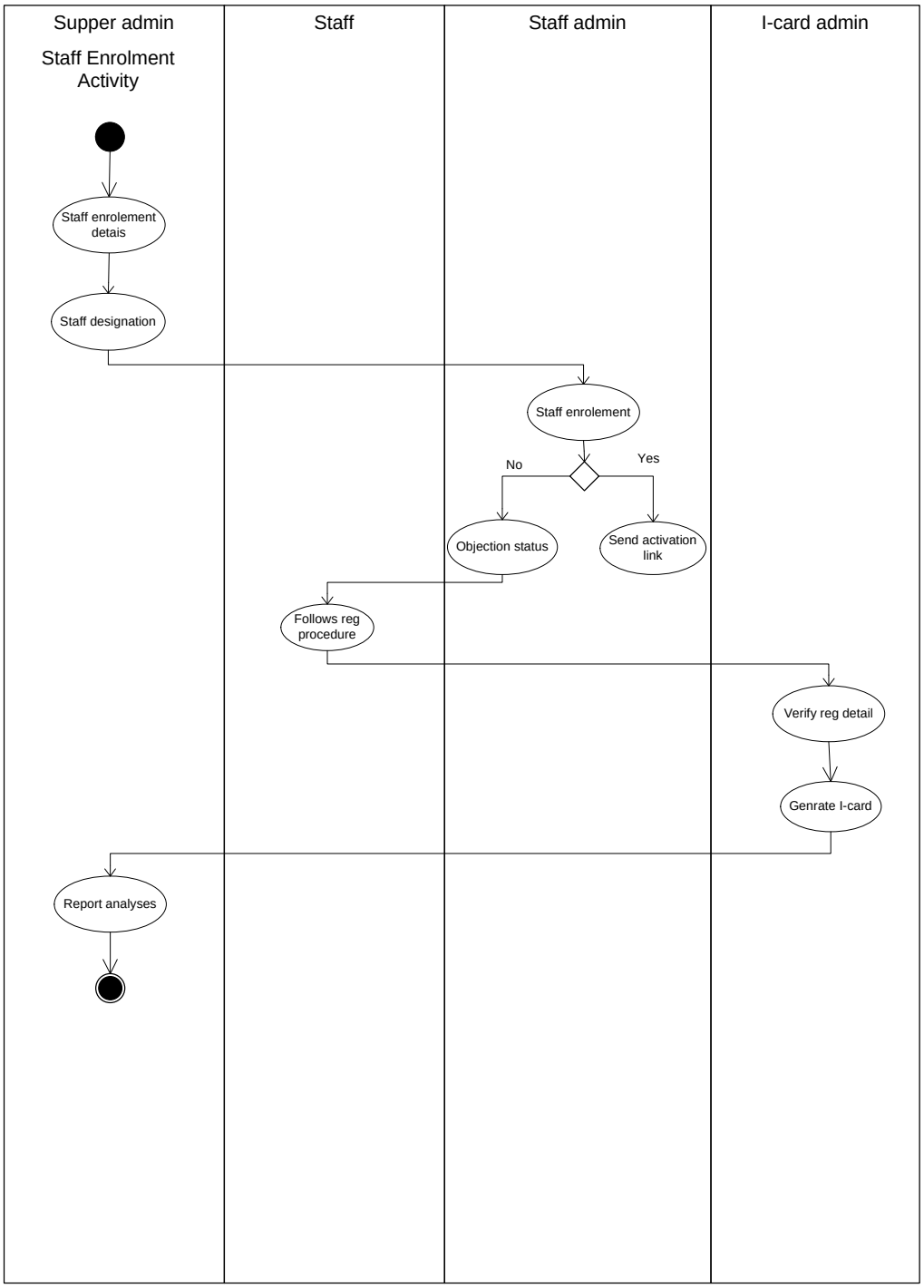


3.4 Activity Diagram

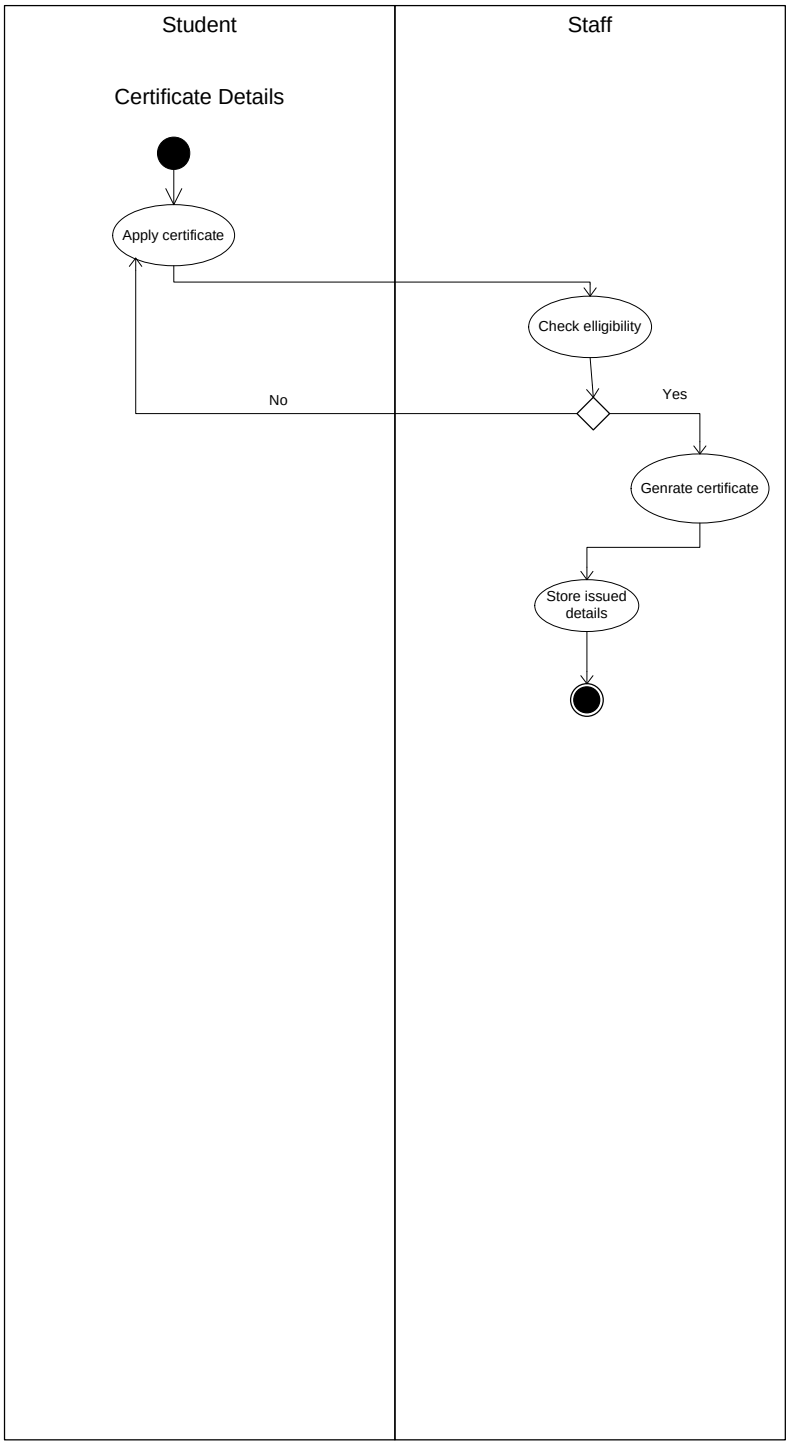



Super Admin
Admission Admin
Student
Staff
Staff Admin
I-card Admin
A
Document Acceptance
Pay fee
Payment verification
No
Yes
Pament acceptance
Send activation link
Follow reg procedure
Verify reg detail
Genrate I-card
Report analyses

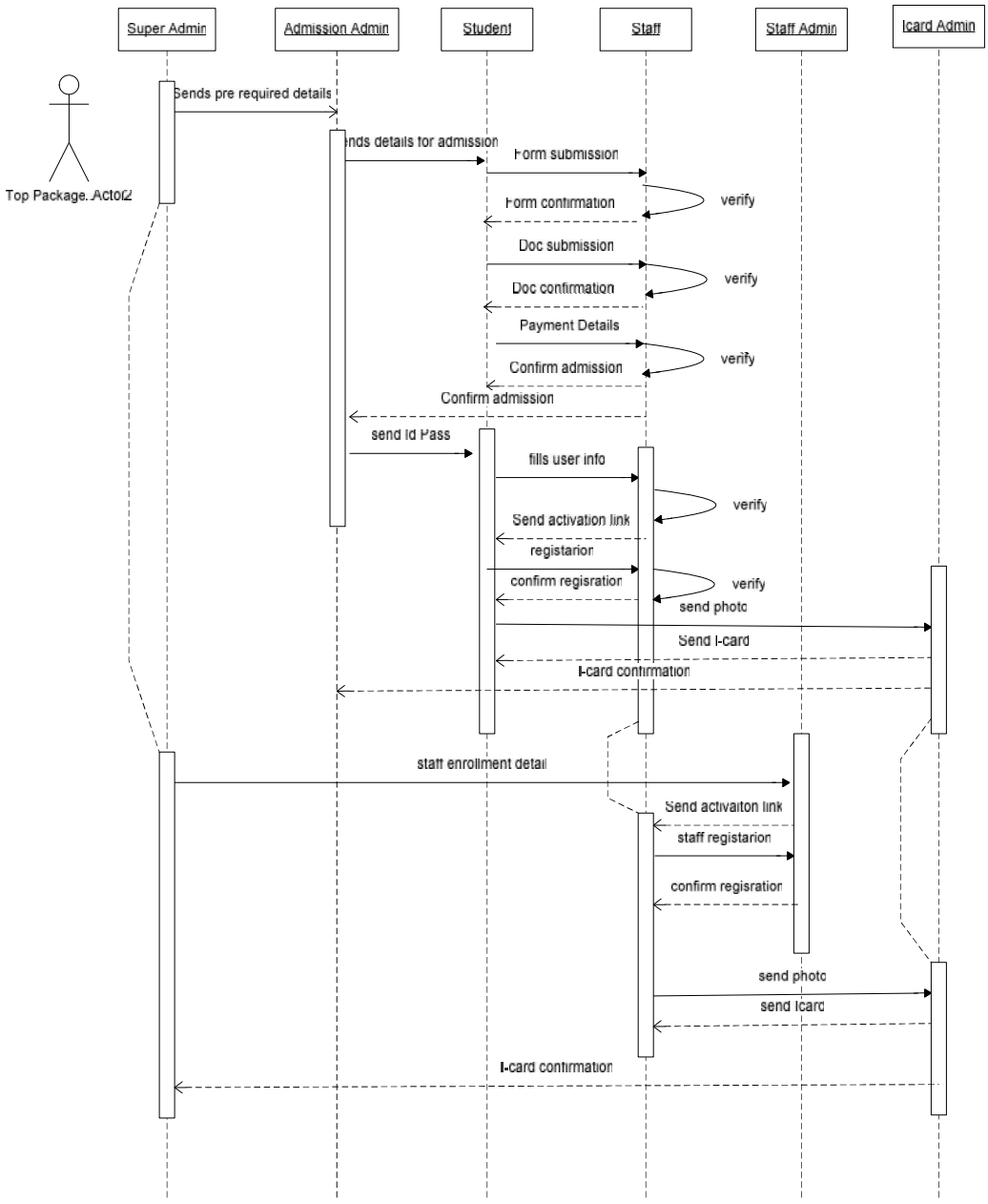
2) Staff Enrolment



3) Certificate Management

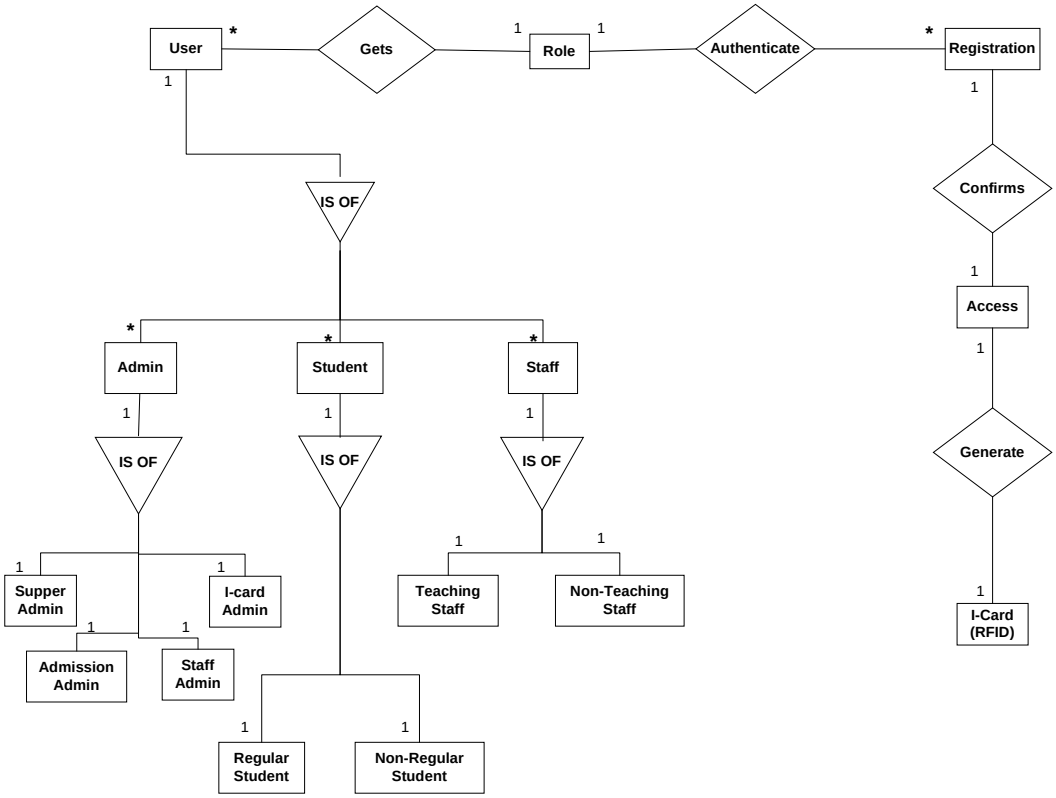


3.5 Sequence Diagrams

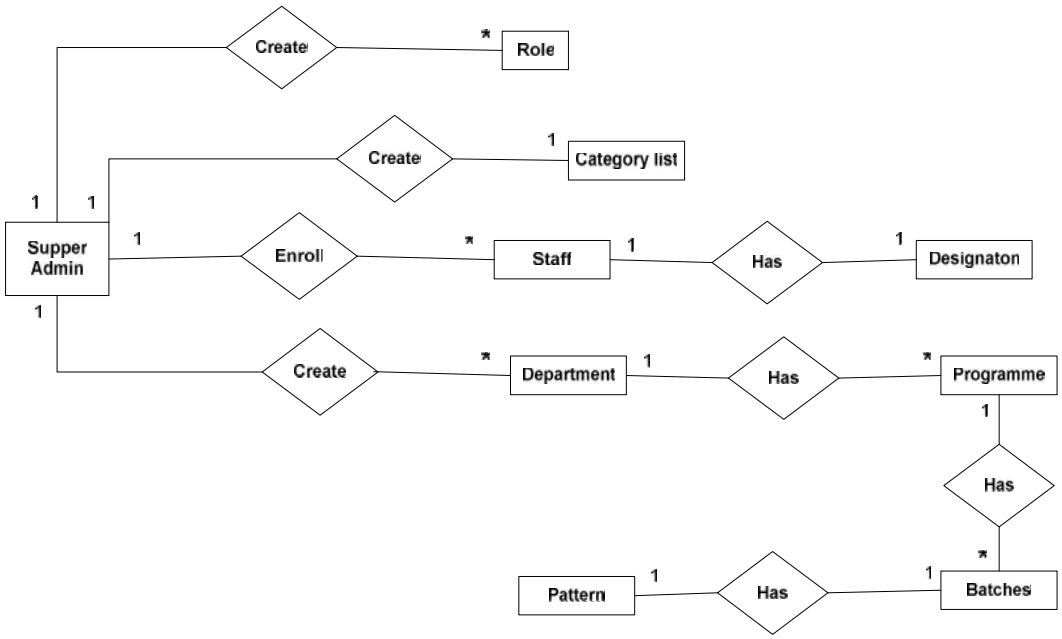


3.6 Entity Relationship Diagram

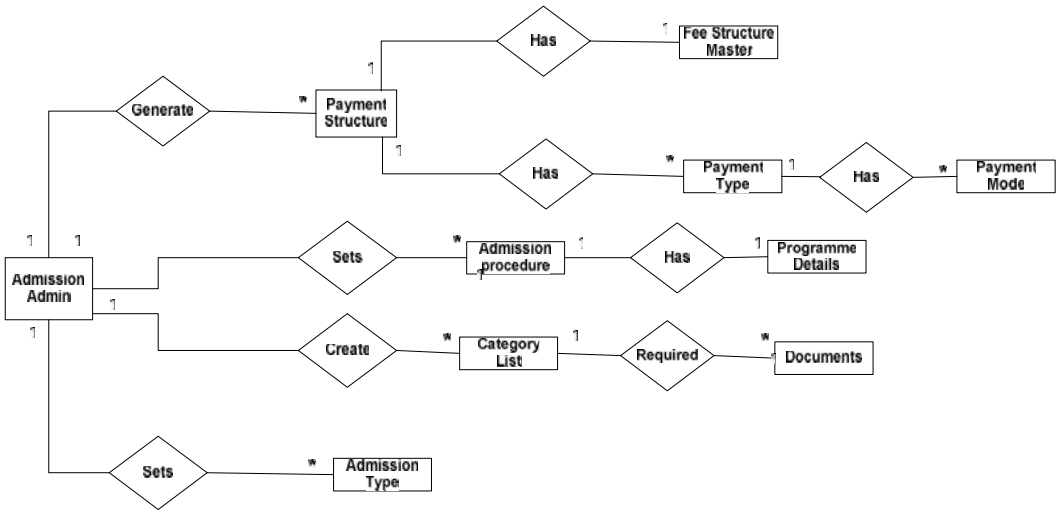
(Overview Of System)



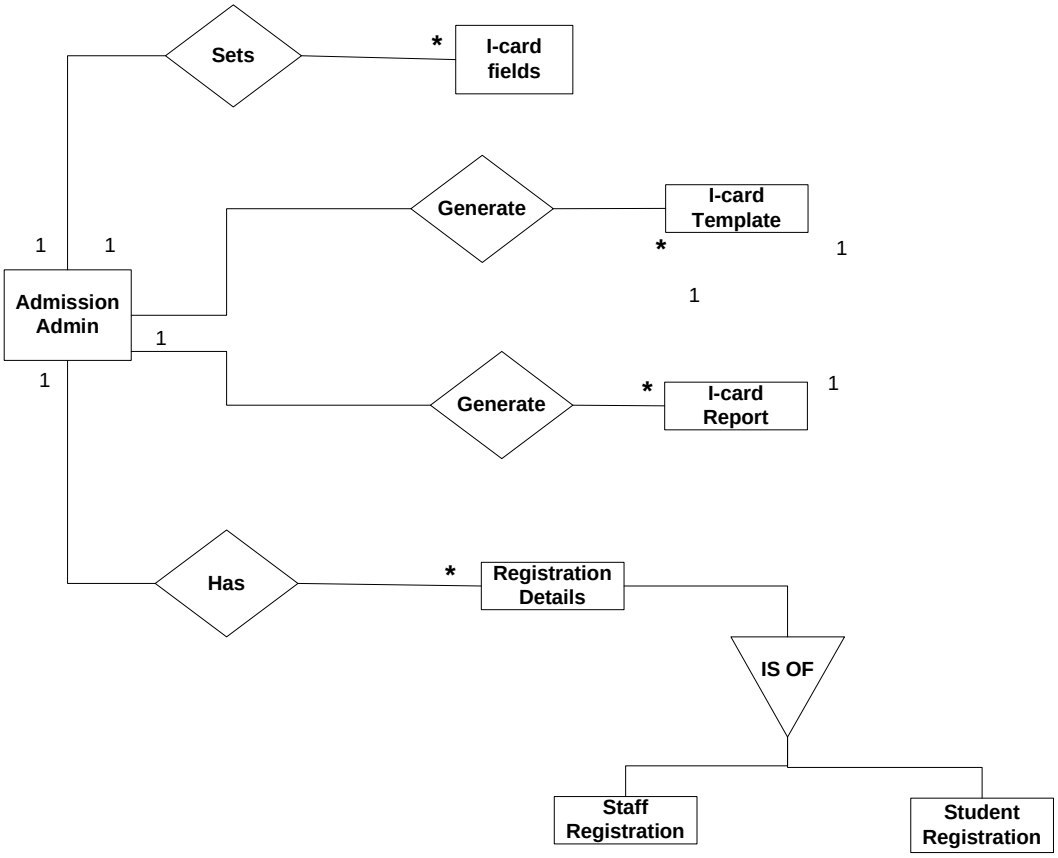
Super admin



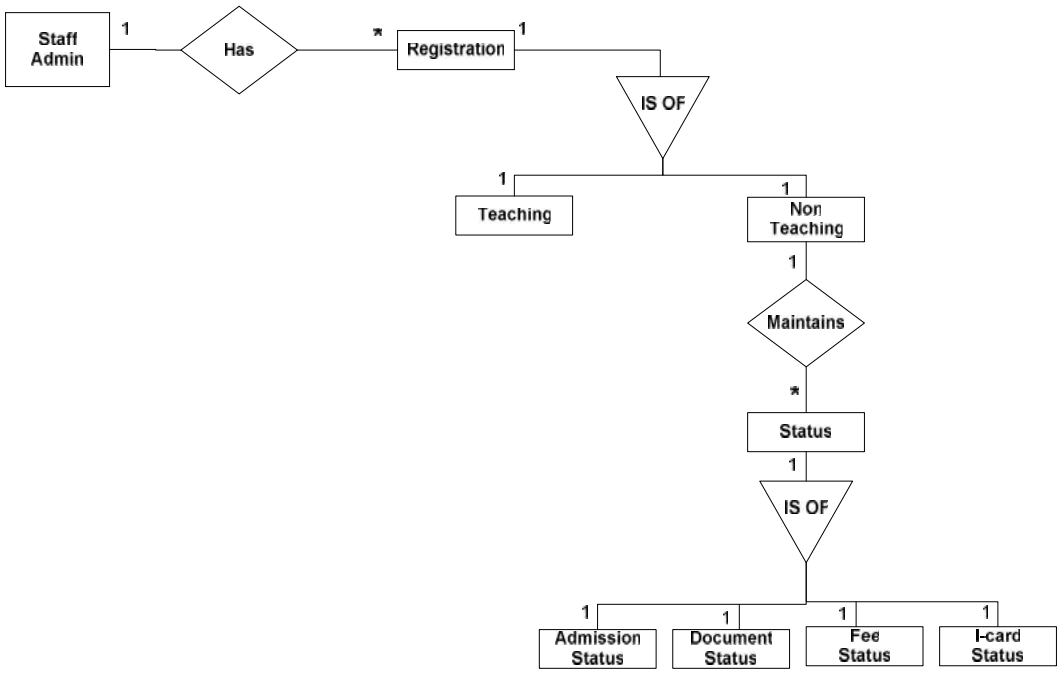
Admission admin



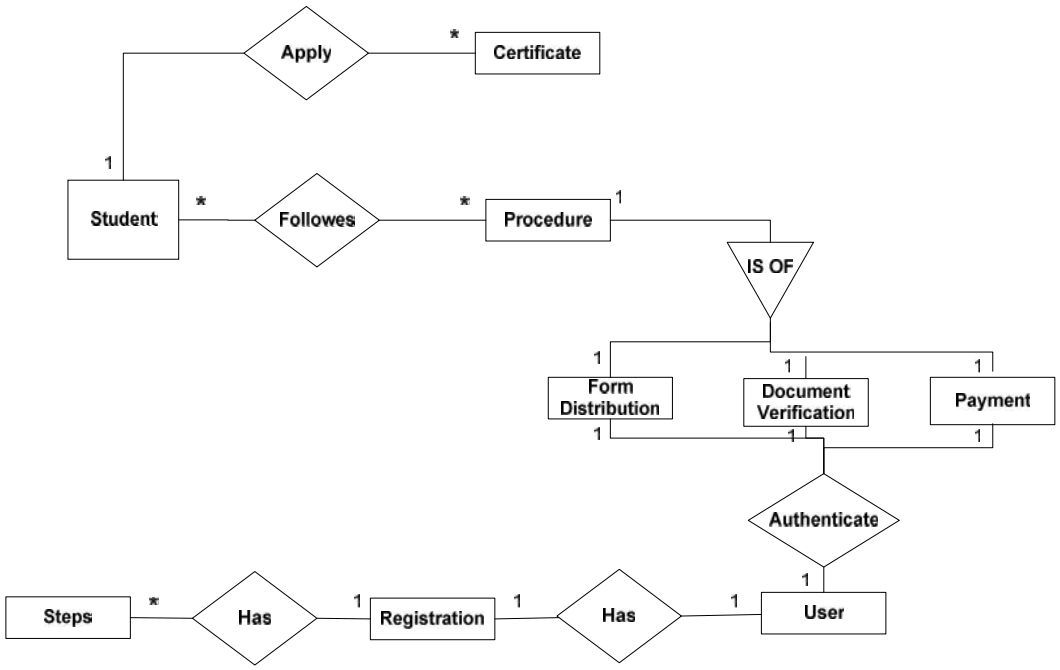
I-card admin


Sets
*
I-card fields
Generate
I-card Template
1
*
1
1
1
1
Admission Admin
1
1
Generate
*
I-card Report
1
Has
*
Registration Details
IS OF
Staff Registration
Student Registration

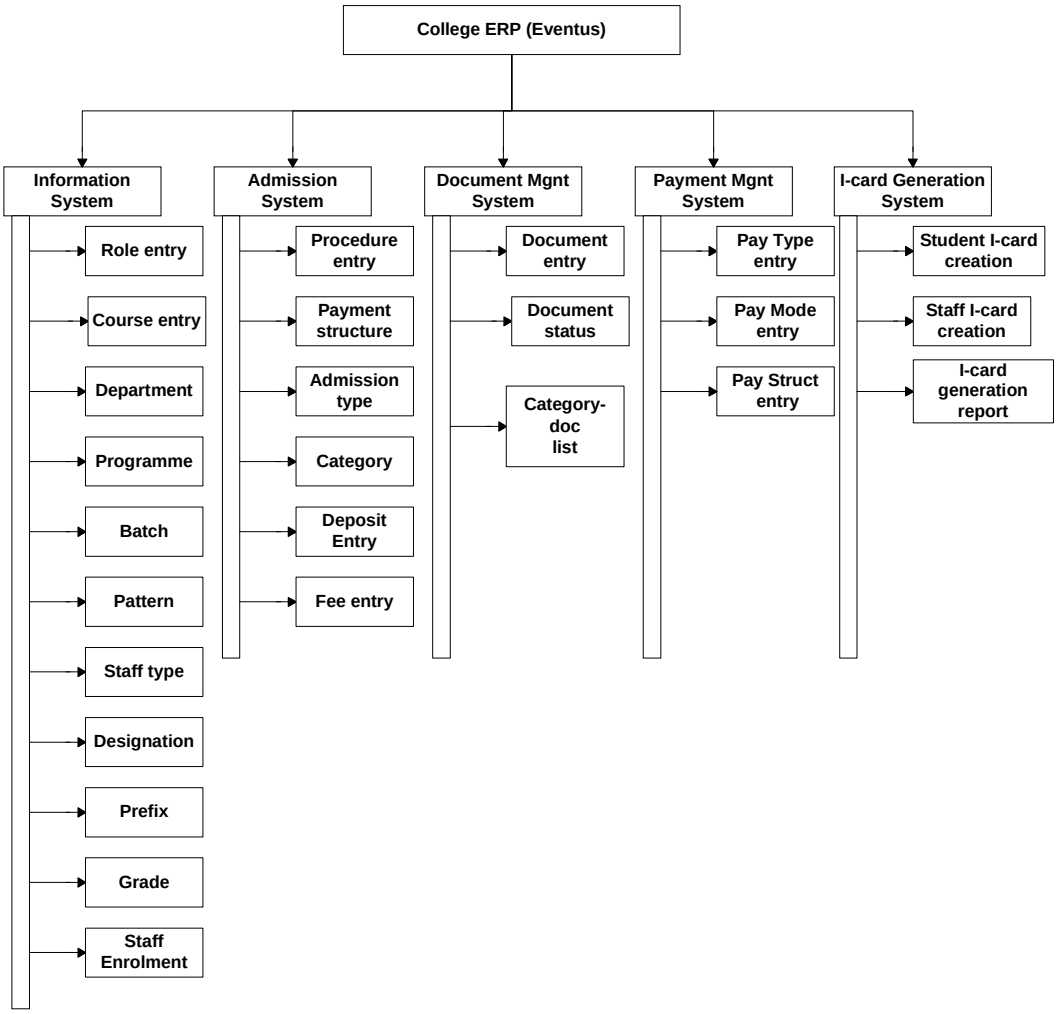
Staff admin



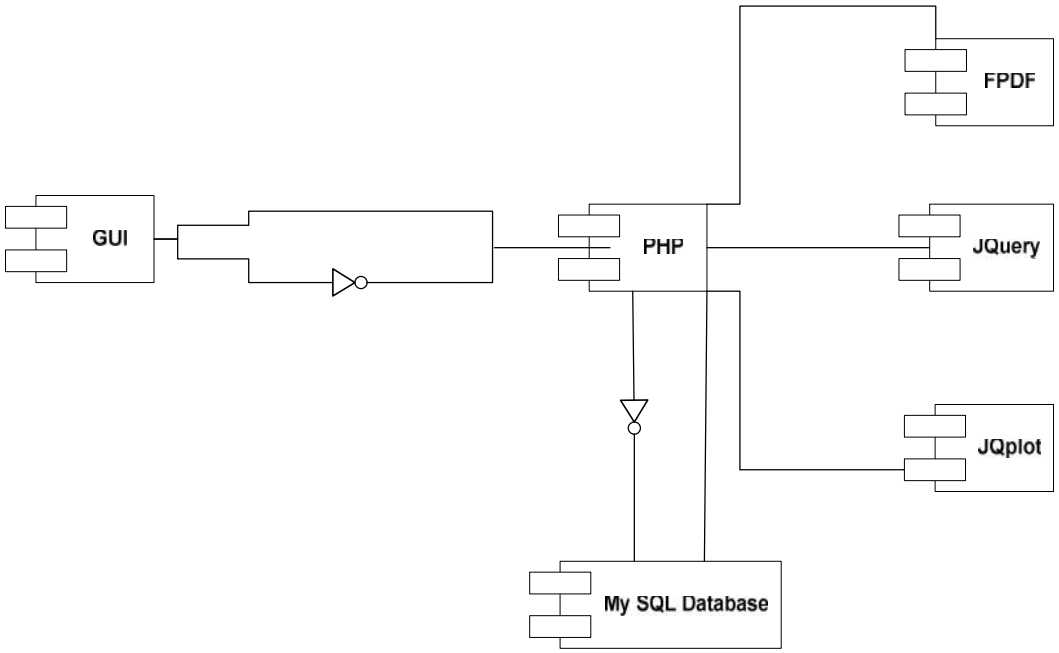
Student



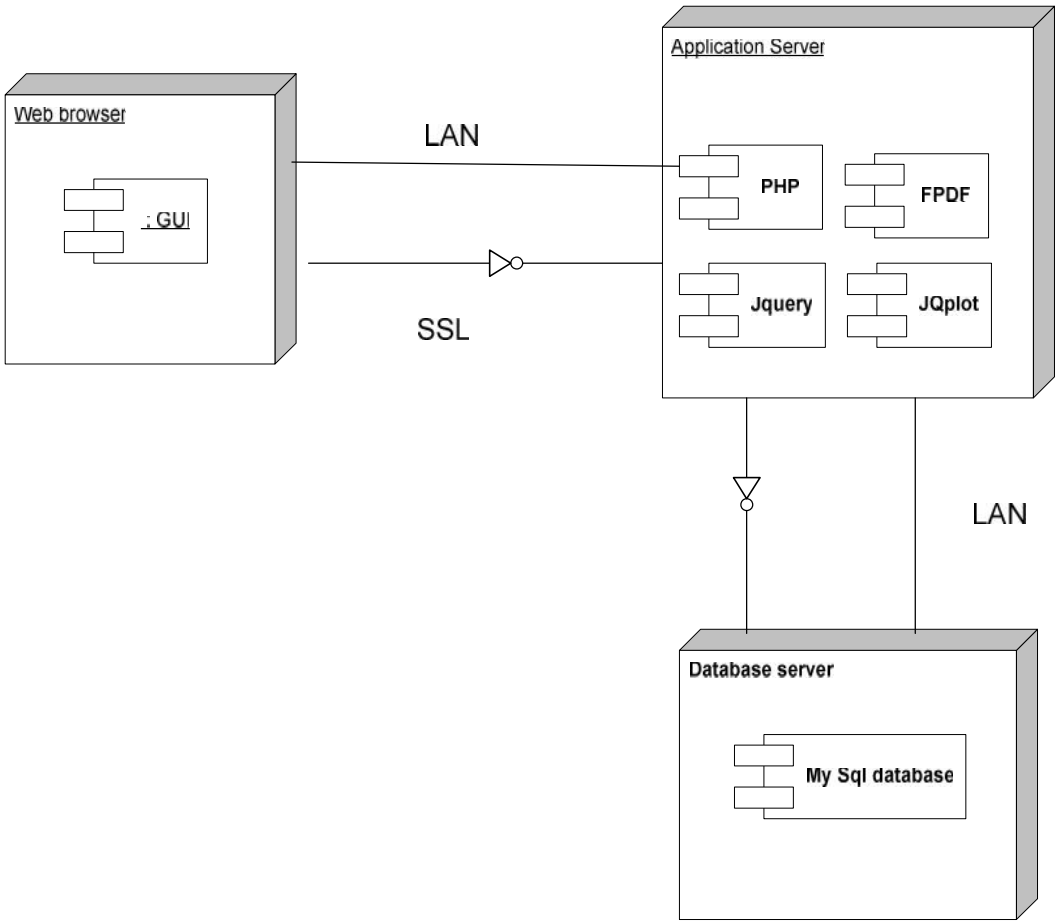
3.7 Module Hierarchy Diagram



3.8 Component Diagram



3.9 Deployment Diagram



3.10 Module Specifications

The System is based on following module which contains

- Information System.
- Admission
- Document
- Payment
- I-Card Generation

Brief information about above modules is as follows

- **Information System**

  This module contains the basic setting required for the use of the ERP modules. In this module the admin has to make setting for different information system such as student, staff, college etc.

  The various setting are provided to the respective administrator To start various sub systems.

- Admission

  This module contains the setting the various procedure for the different programme and for different year. The procedure is follow and appropriate entry is store. Various phases are to be followed in order to confirmation of admission

  The system is able to track the individual record and helps to guide the user.

- Document

  In the document management module all records related to document are store. This help to find out current status of applicant document and we can easily retrieve the information from it.

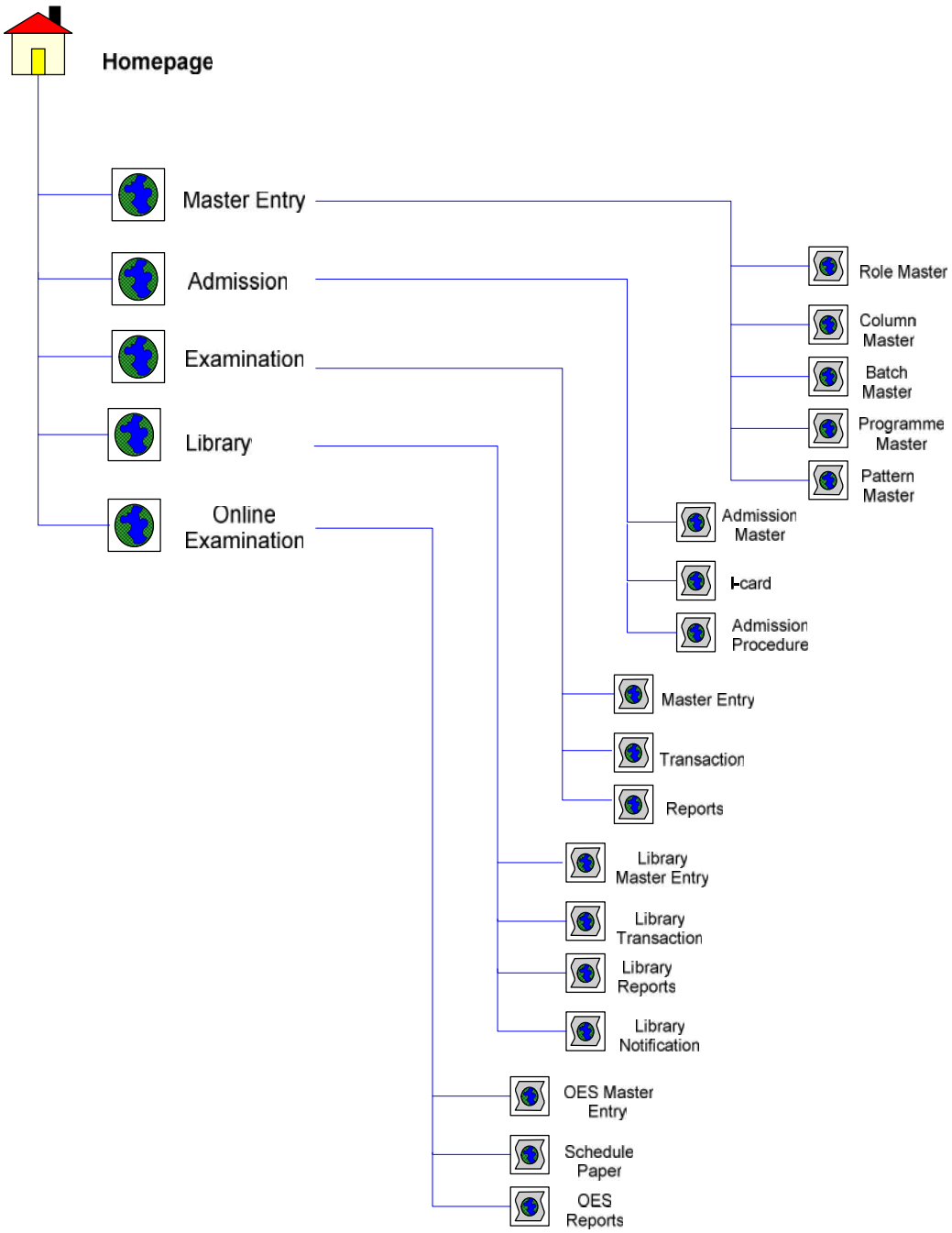
- Payment

  This stores the all payment related records in to sorted manner such that even installment details can easily find out. It has a facility to add new types of payment type, payment modes which can be modified as per the requirement. Finalizing the payment structure helps to faster the process.

- I-Card Generation

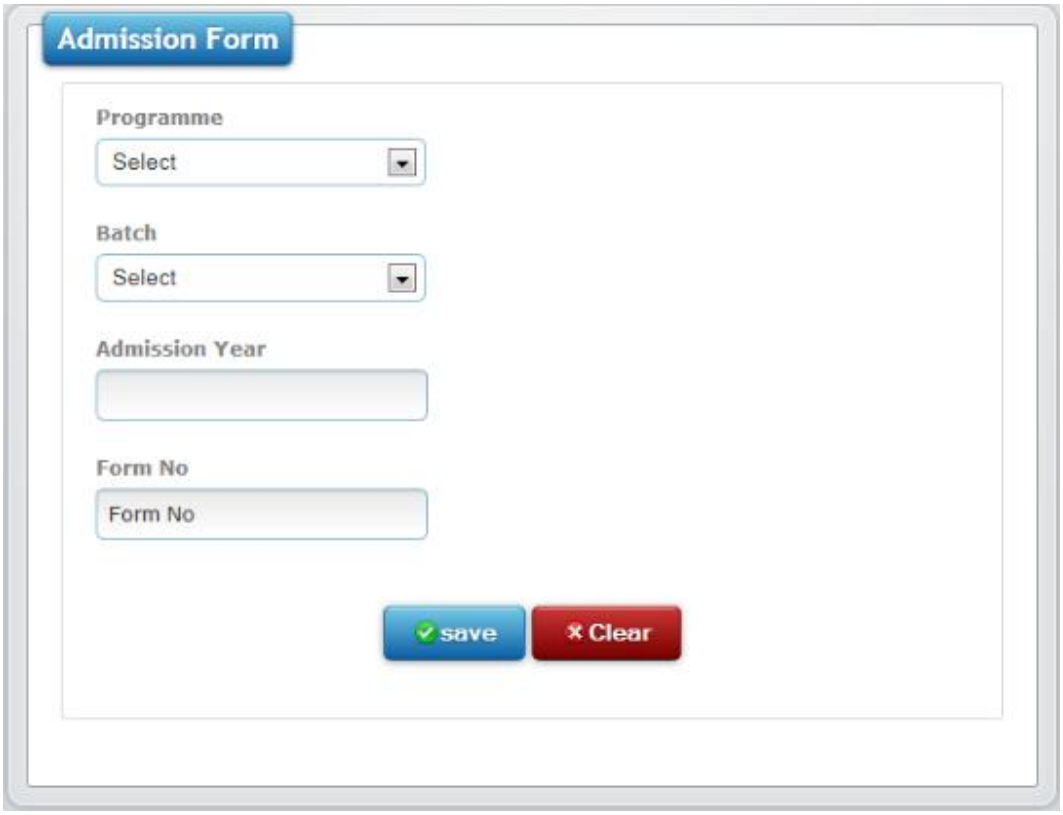
  This process generate I card with required fields and temples as per the admin specified. This helps to print faster and changes can be possible instantly on one click.

3.11 Web Site Map Diagram

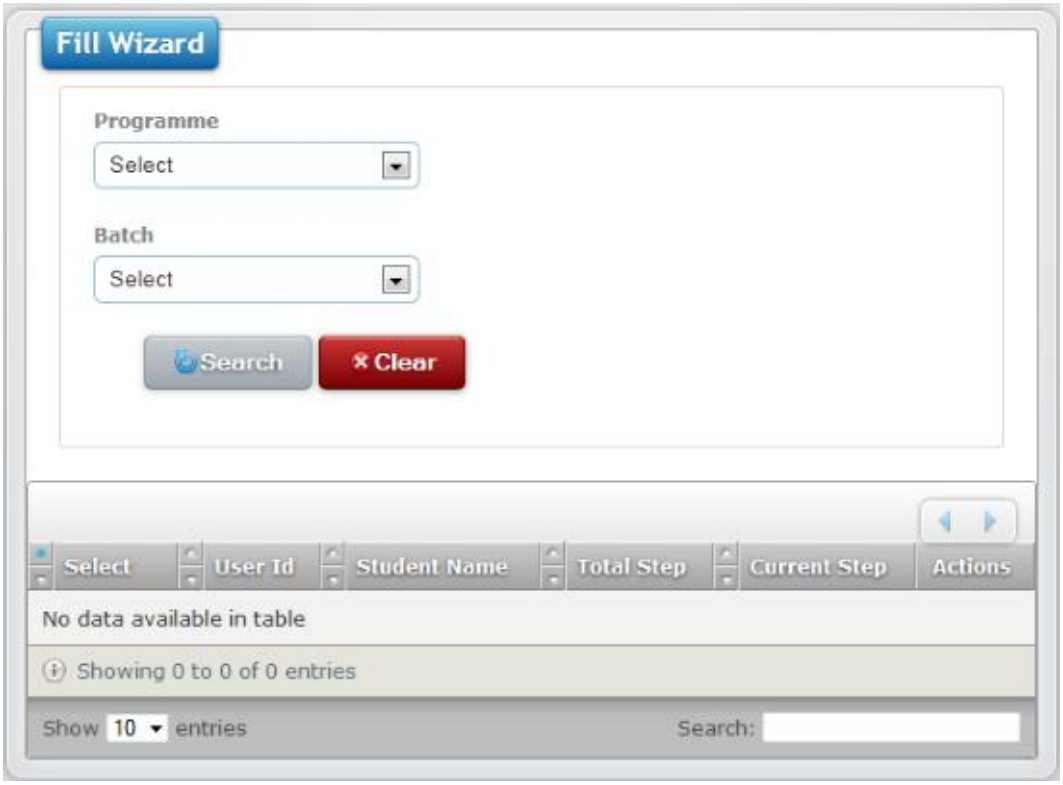


3.12 User Interface Design

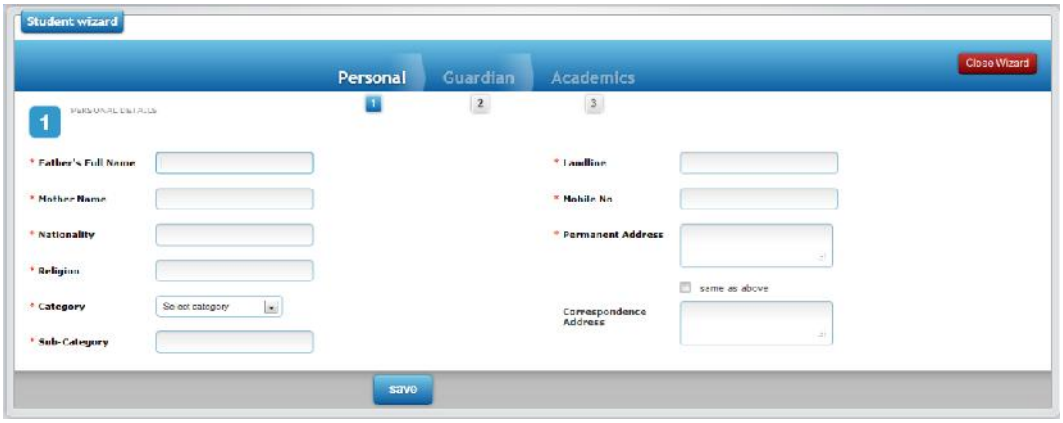
1) Admission Form Entry



2) Fill Wizard Form

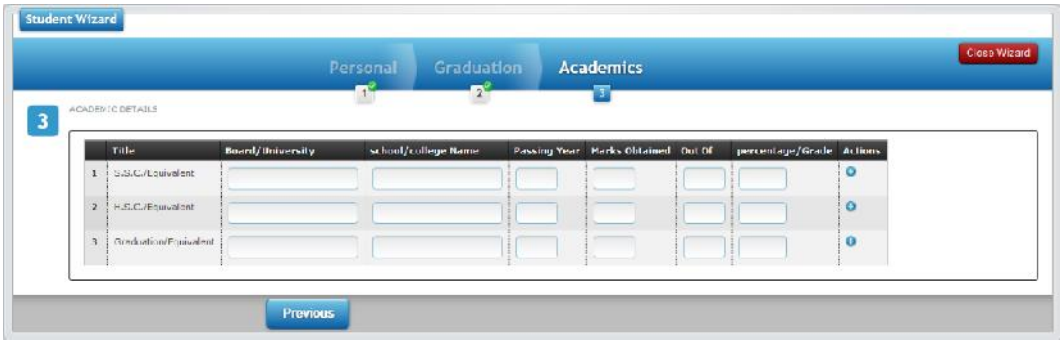


3) Personal Details Form



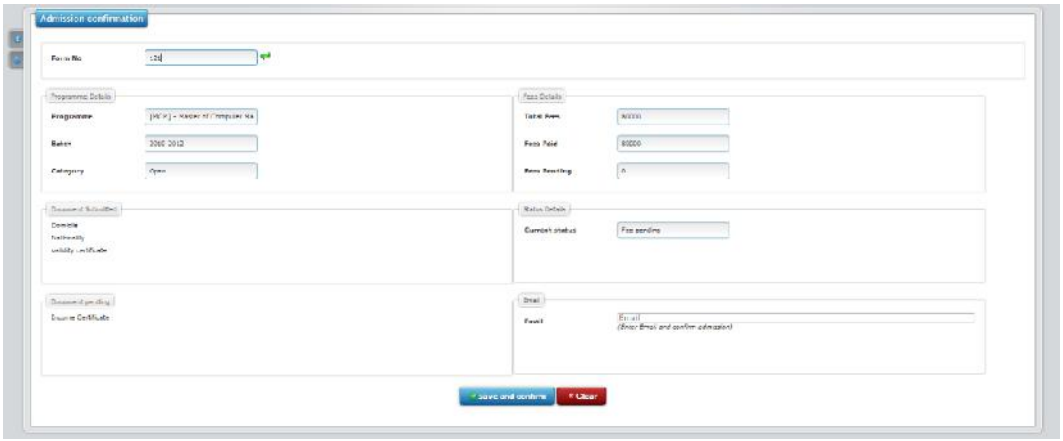
4) Guardian Detail Form

5) Academics Detail Form

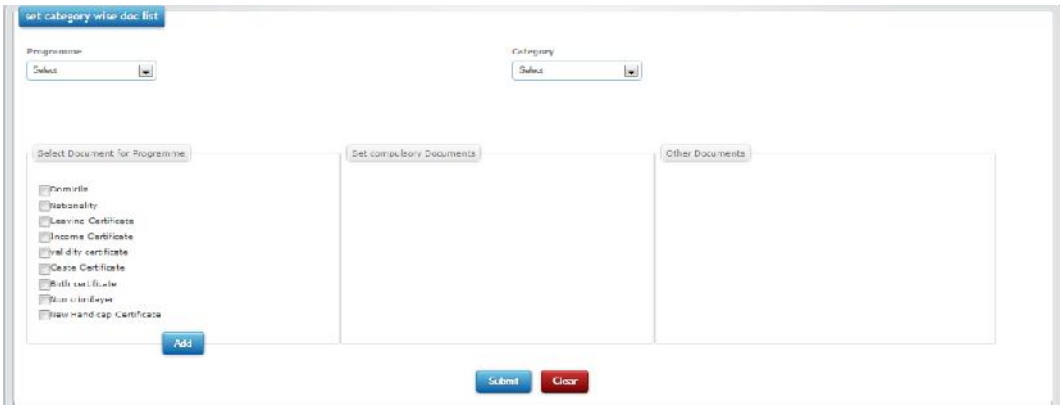


6)Document Submission Form

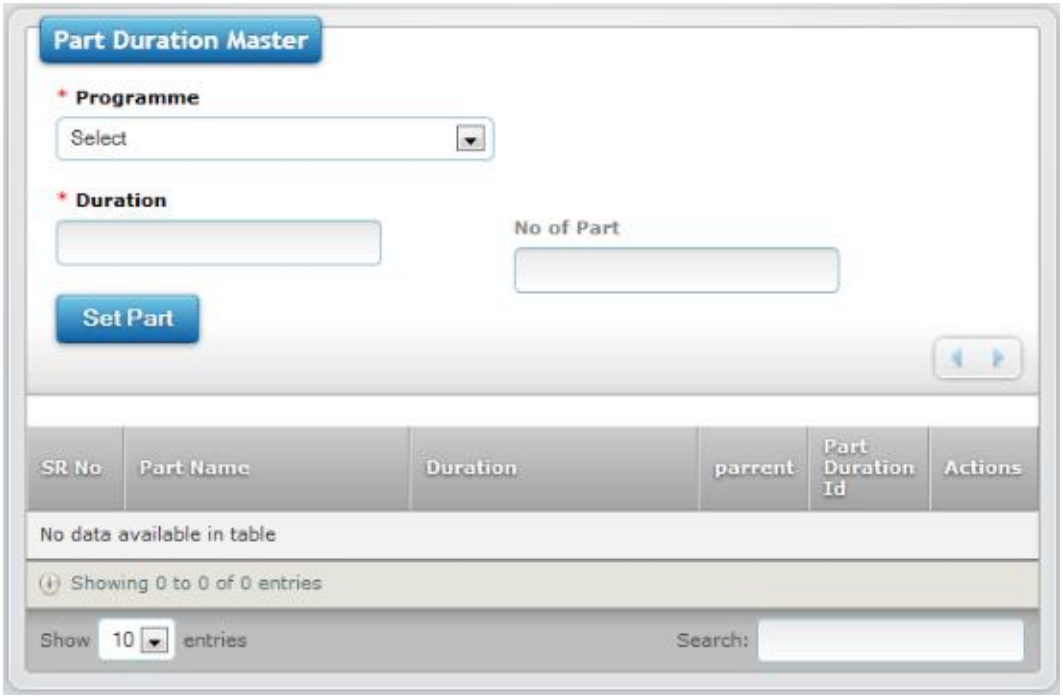
7) Admission Confirmation Form



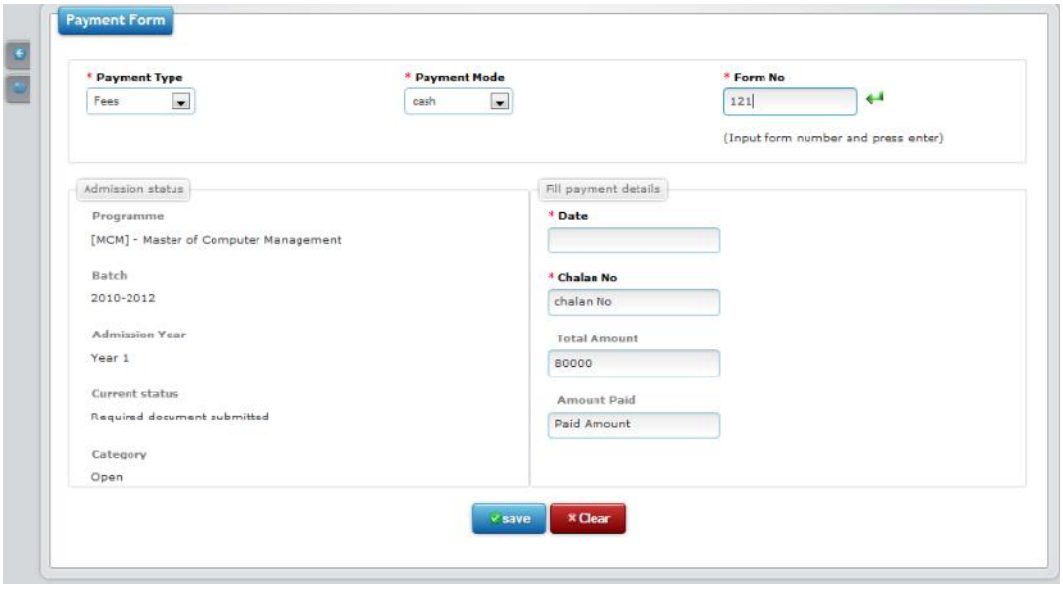
8) Category wise Document List Form



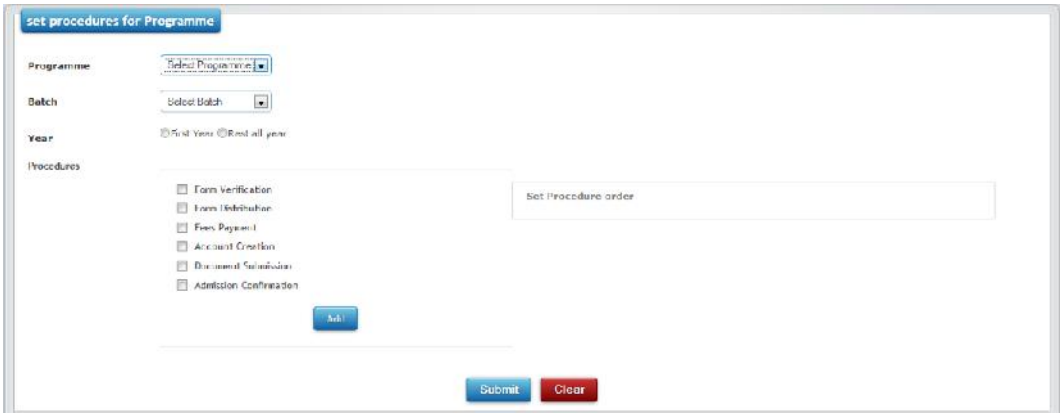
9) Part Duration Form



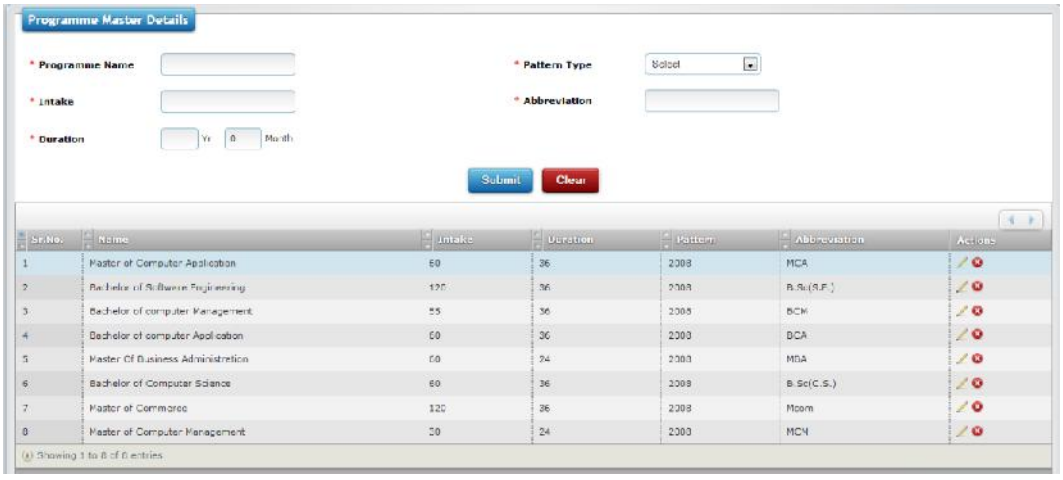
10) Payment Form



11) Programme Procedure Form



12) Programme Master Form



3.13 Table specifications

User Info Table:

Field Name	Data Type	Size	Constrains
user_id	Int	11	Primary Key
adm_sum_id	Int	11	Foreign key
prefix_id	Int	11	Foreign key
first_name	Varchar	30	
middle_name	Varchar	30	
last_name	Varchar	30	
gender	enum('M', 'F')		
dob	Date		
email	Varchar	100	
password	Varchar	32	
secondary_email	Varchar	100	
activation	Varchar	40	
deactivation	Varchar	40	
status_id	Int	11	Foreign key
current_step	Int	2	
total_step	Int	2	
wizard_status	Int	2	
role_master_id	Int	11	Foreign key
current_batch_id	Int	11	Foreign key
joining_batch_id	Int	11	Foreign key
image	Text		
display	enum('Y', 'N')		
date_created	datetime		
date_modified	timestamp		
created_by	Int	12	
modified_by	Int	12	

Role_master

Field Name	Data Type	Size	Constrains
role_master_id	Int	11	Primary Key
role_id	Int	11	
role_name	Varchar	50	
role_description	Text		
display	enum('Y', 'N')		
date_created	datetime		
date_modified	timestamp		
created_by	Int	12	
modified_by	Int	12	

Status_master

Field Name	Data Type	Size	Constrains
status_id	Int	11	Primary Key
status_name	Varchar	25	
description	text		
display	enum('Y', 'N')		
date_created	datetime		
date_modified	timestamp		
created_by	Int	12	
modified_by	Int	12	

Programme

Field Name	Data Type	Size	Constrains
programme_id	Int	11	Primary Key
programme_master_id	Int	11	Foreign key
pattern_start_year	Int	5	
pattern_end_year	Int	5	
prog_status_id	Int	11	Foreign key
no_of_part	Int	2	
degree_type_id	Int	11	Foreign key
display	enum('Y', 'N')		
date_created	datetime		
date_modified	timestamp		
created_by	Int	12	
modified_by	Int	12	

Procedure_master

Field Name	Data Type	Size	Constrains
pro_mas_id	Int	11	Primary Key
procedure_name	Varchar	50	
description	Text		
display	enum('Y', 'N')		
date_created	datetime		
date_modified	timestamp		
created_by	Int	12	
modified_by	Int	12	

Programme_procedure

Field Name	Data Type	Size	Constrains
prog_proc_id	Int	11	Primary Key
programme_id	Int	11	Foreign key
batch_id	Int	11	Foreign key
adm_year	Varchar	20	
pro_mas_id	Int	11	Foreign key
pro_ser_no	Int	11	
status_id	Int	11	Foreign key
display	enum('Y', 'N')		
date_created	Datetime		
date_modified	timestamp		
created_by	Int	12	
modified_by	Int	12	

College_details_master

Field Name	Data Type	Size	Constrains
college_id	Int	11	Primary Key
college_name	Varchar	100	
address	Varchar	100	
establishment	Date		
phone_no	Varchar	20	
email	Varchar	100	
website	Varchar	100	
fax	Varchar	20	
display	enum('Y', 'N')		
date_created	Datetime		
date_modified	timestamp		
created_by	Int	12	
modified_by	Int	12	

Personal_master

Field Name	Data Type	Size	Constrains
personal_id	int	11	Primary Key
user_id	Int	11	Foreign key
role_master_id	Int	11	Foreign key
mother_name	Varchar	50	
father_name	Varchar	50	
religion	Varchar	30	
caste	Varchar	30	
sub_caste	Varchar	30	
nationality	Varchar	30	
permanent_address	Varchar	100	
correspondence_address	Varchar	100	
landline	Varchar	15	
mobile	Varchar	15	
roll_no	Int	11	Foreign key
display	enum('Y', 'N')		
date_created	datetime		
date_modified	timestamp		
created_by	Int	12	
modified_by	Int	12	

Academic_details

Field Name	Data Type	Size	Constrains
academic_details_id	Int	11	Primary Key
academic_id	Int	11	Foreign key
year	Int	2	
marks	Int	5	
out_of	Int	5	
grade	Varchar	1	
display	enum('Y', 'N')		
date_created	datetime		
date_modified	timestamp		
created_by	Int	12	
modified_by	Int	12	

Payment_master

Field Name	Data Type	Size	Constrains
pay_id	Int	11	Primary Key
pay_type_id	Int	11	Foreign key
ref_table_id	Int	11	Foreign key
ref_record_id	Int	11	Foreign key
amt_paid	Int		
pay_date	datetime		
pay_mode_id	Int	11	Foreign key
desc	text		
display	enum('Y', 'N')		
date_created	datetime		
date_modified	timestamp		
created_by	Int	12	
modified_by	Int	12	

Fee_structure_master

Field Name	Data Type	Size	Constrains
fee_struct_id	Int	11	Primary Key
programme_id	Int	11	Foreign key
batch_master_id	Int	11	Foreign key
category_id	Int	11	Foreign key
fee_amt	Bigint	20	
display	enum('Y', 'N')		
date_created	datetime		
date_modified	timestamp		
created_by	Int	12	
modified_by	Int	12	

Pay_type

Field Name	Data Type	Size	Constrains
pay_type_id	Int	11	Primary Key
pay_type_name	Varchar	50	
description	Text		
display	enum('Y', 'N')		
date_created	datetime		
date_modified	timestamp		
created_by	Int	12	
modified_by	Int	12	

category_master

Field Name	Data Type	Size	Constrains
category_id	Int	11	Primary Key
cat_name	Varchar	50	
cat_desc	text		
display	enum('Y', 'N')		
date_created	datetime		
date_modified	timestamp		
created_by	Int	12	
modified_by	Int	12	

Cat_doc_master

Field Name	Data Type	Size	Constrains
cat_doc_id	int	11	Primary Key
doc_mas_ids	Varchar	100	
compulsory_doc_mas_ids	Varchar	100	
other_doc_mas_ids	Varchar	100	
programme_id	Int	11	Foreign key
category_id	Int	11	Foreign key
display	enum('Y', 'N')		
date_created	datetime		
date_modify	timestamp		
created_by	Int	12	
modified_by	Int	12	

Doc_stud_submission

Field Name	Data Type	Size	Constrains
doc_stud_submission_id	Int	11	Primary Key
doc_stud_id	Int	11	Foreign key
doc_mas_id	Int	11	Foreign key
submitted_date	date		
returned_date	date		
doc_status_id	Int	11	Foreign key
display	enum('Y', 'N')		
date_created	datetime		
date_modified	timestamp		
created_by	Int	12	
modified_by	Int	12	

Icard_fields

Field Name	Data Type	Size	Constrains
icard_field_id	int(11)	11	Primary Key
icard_field_name	varchar(50)		
description	text		
prog_proc_id	int(12)	11	Foreign key
display	enum('Y', 'N')		
date_created	datetime		
date_modified	timestamp		
created_by	int(11)	12	
modified_by	int(11)	12	

Part_duration_master

Field Name	Data Type	Size	Constrains
part_duration_id	Int	11	Primary Key
programme_id	Int	11	Foreign key
part_name	Varchar	50	
part_duration	Int	11	
part_postfix	Varchar	30	
parents	Varchar	100	
childs	Varchar	100	
display	enum('Y', 'N')		
date_created	datetime		
date_modified	timestamp		
created_by	Int	12	
modified_by	Int	12	

Admission_status

Field Name	Data Type	Size	Constrains
adm_status_id	Int	11	Primary Key
adm_status_name	Varchar	50	
description	text		
display	enum('Y', 'N')		
date_created	datetime		
date_modified	timestamp		
created_by	Int	12	
modified_by	Int	12	

Adm_summary

Field Name	Data Type	Size	Constrains
adm_sum_id	Int	11	Primary Key
form_no	Int	11	Foreign key
user_id	Int	11	Foreign key
prospectus	enum('Y', 'N')		
batch_master_id	Int	11	Foreign key
programme_id	Int	11	Foreign key
category_id	Int	11	Foreign key
part_duration_id	Int	11	Foreign key
chalan_no	Varchar	10	
total_amt	Int	8	
paid_amt	Int	8	
pending_amt	Int	8	
email	Varchar	100	
activation	Varchar	32	
deactivation	Varchar	32	
icard_master_id	Int	11	Foreign key
adm_status_id	Int	11	Foreign key
total_deposit	Int	8	
prog_proc_id	Int	11	Foreign key
display	enum('Y', 'N')		
date_created	Datetime		
date_modified	Timestamp		
created_by	Int	12	
modified_by	Int	12	

Batch_master

Field Name	Data Type	Size	Constrains
batch_master_id	int(11)	11	Primary Key
programme_id	int(11)	11	Foreign key
year	varchar(9)	9	
result_available_till	int(11)	2	
part_duration_id	int(11)	11	Foreign key
current_part_id	int(11)	11	Foreign key
status_id	int(12)	11	Foreign key
display	enum('Y', 'N')		
date_created	datetime		
date_modified	timestamp		
created_by	int(12)	12	
modified_by	int(12)	12	

3.14 Test Procedures and Implementation

Once the source code has been generated software must be tested to uncover (and correct) as many errors as possible before delivery of the product. Our goal is to build a series of test cases that have likelihood of finding errors. Testing begins “in the small” and progresses “to the large”.

Software testing is a process of executing programs with objective of finding error. S/w testing is simply how easily a computer program can be tested.

As per Glen Myers there are three objectives for testing which follows are as:

- A good test case is one that has a high probability of finding as yet undiscovered error.
- A successful test is one that uncovers as yet undiscovered error.
- Testing is a process of executing a program with the intent of finding an error.

Testing can't show the absence of defects, it can only show that s/w errors are present. Therefore it is necessary to keep this in mind while conducting the testing part of s/w. There are many testing principles as recommended by Davis. According to the principles, the testing phase of this project done. These are follows:

- All tests should be traceable to customer requirements. The testing should be done in the front of some of users to show the functionality of the system.
- Test should be planned long before testing begins.
- Testing should begin "In the small" and progress towards testing "In the large".
- To the most effective an independent third party should conduct testing.
- A good has a high probability of finding out the errors.
- A good test is not redundant.
- A good test should be either too simple not too complex.

In addition, data collected as testing is conducted provide a good induction of s/w quality as a whole.

Testing Methodologies

We tested our system according to methodologies and testing methodologies adopted are:

- Unit testing
- Sub-system testing
- System testing
- Acceptance testing

The tested methodologies were adopted in the below given sequence.

Unit testing ➔ Subsystem testing ➔ System testing ➔ Acceptance testing

Unit Testing (White Box Testing)

Each module of the system was tested separately to ensure its proper functioning.

Activities follows in the unit testing are:

- Ensure all loops are terminated properly in this system.
- Identification and removal of abnormal termination of all loops.

- Ensuring the all errors was trapped. We ensured that the system trapped all the errors by working on all the option we have in this system. For example, we tested the working of committees, tenders, meeting notices etc. properly at every terminal.
- Check the values returned from called program. We checked the values returned by filling every option. We went to the database and checked the values entered into the database as well as the confirmation message that was to be displayed on the screen after filling the form of the every option.

Sub System Testing

A sub system testing is a collection of modules. All subsystems were tested to ensure dataflow correctness, functional validity and interface integrity.

System Testing

This testing was done once all the subsystems had been completely developed and tested independently. The system was tested to ensure proper interaction between all the subsystems. It ensures the actual working

of the system. For example, if the employee ID does not exist how his committee could status could be displayed. The performance of the system was continuously monitored during this phase of testing.

Acceptance Testing (Black Box Testing)

This was the latest phase in the system testing process. The system was tested to check if confirmed to the customer's requirements. All the employee of Technical and financial department tested the respective system according to his or her requirements.

GUI Testing

Because many modern GUIs have the same look and feel, a series of standard test can be derived. Finite state modeling graph may be used to derive a series of tests that address specific data and program objects that are relevant to the GUI.

GUI testing is also known as **Functional Testing.** Functional Testing is a testing process to derive a set of input conditions that will fully exercise all

functional requirements for a program. It emphasizes to find error in following categories:

- Incorrect or missing functions.
- Interface errors.
- Error in data structure or external database access.
- Performance errors.

Testing the expected Application flow or logic

The application flow is actually a series of screens through which the user can interact with the system. These screens can be depicted as state diagram where each state transition occurs via requests and responses. The behaviors of the system can be easily verified against this state diagram.

Database Testing

This will include testing the correctness of the data that will ensure the following features:

- Database schema reflects exactly as per the design.
- Security and Privacy features are protected properly.
- All the data in the database is correct.
- Check for duplication, naming conventions is done.
- DBMS performs all the functions of insertion, deletion and updating correctly.
- Check for table, record, attributes, primary key, foreign key, constraints, and not null, referential integrity.
- Observing the state of the database before and after testing.
- Using equivalence partitioning and boundary value analysis where the data needs to be specified in a specific range.
- Checking response time for retrieval of data as per the request of the user.

Test Case Implementation

Test case 1

1.1	Test procedure for Login Form	
1	**Test code id**	ECESO1
2	**Purpose**	To log into the software by an authenticated user
3	**Procedure**	verifying above login id into the database; Appling encryption logic to the password using salted MD5 # algorithm and allowing the role base
4	**Test data**	Login Id, password, role
5	**Output**	Unable to role base functionality to the logged user of the system

1.2	Data Matrix				
	Input attr	Equivalence class partitioning		Boundary value analysis	
		valid	Invalid	Min	Max
1	Login Id	a-z,A-z,0 to 9	blank, other than	6	15
			a-z,A-z,0 to 9		
2	password	a-z,A-z,0 to 9	blank, other than	6	15
			a-z,A-z,0 to 9		

1.3	Test Case			
	Action	input	Expected output	
1	Enter Login Id	UI1306	valid	pass
	Enter Login Id	tant@ 1306#	Invalid	pass
2	Enter Password	xxxxxx	valid	pass
	Enter Password	$%zxcvb	Invalid	pass

Test case 2

2.1	Test procedure for Admission Form	
1	**Test code id**	ECESO2
2	**Purpose**	To enter required details about applicant
3	**Procedure**	select Programme from select box select Batch from select box Enter form No
4	**Test data**	Form No
5	**Output**	Unable to enter repeat entry of form

2.2	Data Matrix				
	Input attr	Equivalence class partitioning		Boundary value analysis	
		valid	Invalid	Min	Max
1	form no	a-z,A-z,0 to 9	blank, other than	6	15
			a-z,A-z,0 to 9		

2.3	Test Case			
	Action	input	Expected output	
1	Enter Form No	FN001	valid	pass
		fa9$0*	Invalid	pass

Test case 3

3.1	Test procedure for Personal Info Form	
1	**Test code id**	ECESO3
2	**Purpose**	To enter required details about user personal info
3	**Procedure**	The above mentioned inputs are stored to the database as per user id
4	**Test data**	user_id, role_id, father's full name, religion, mother's name
5	**Output**	caste, sub-caste, nationality

3.2	Data Matrix				
	Input attr	Equivalence class partitioning		Boundary value analysis	
		valid	Invalid	Min	Max
1	father full name	alphabetic and	all special symbol	2	30
		space	number		
2	mothers name	alphabetic and	all special symbol	2	30
		space	number		
3	religion	alphabetic and	all special symbol	2	30
		space	number		
4	nationality	alphabetic and	all special symbol	2	30
		space	number		
5	sub-cast	alphabetic and	all special symbol	2	30
		space	number		
6	correspondence	alphabetic and	all special symbol	2	50
	address	space, number			
7	permanent	alphabetic and	all special symbol	2	50
	address	space, number			
8	landline	number	all alphabetic	2	15
9	mobile	number	all alphabetic	2	15

3.3	Test Case			
	Action	input	Expected output	
1	Enter father name	Prasanna	valid	pass
		Prasanna65	Invalid	pass
2	Enter mothers	sema	valid	pass
	name	sema21	Invalid	pass
3	religion	Hindu	valid	pass
		7hindu	Invalid	pass
4	nationality	Indian	valid	pass
		Indo77	Invalid	pass
5	correspondence	apposit to fc college		
	address	pune	valid	pass
		Fc college #&&		
		pune	Invalid	pass
6	landline	123456	valid	pass
		as1234567	Invalid	pass
7	mobile	123456	valid	pass
		as1234567	Invalid	pass

Test case 4

4.1	Test procedure for Payment Form	
1	**Test code id**	ECESO4
2	**Purpose**	To enter required details about user Payment info
3	**Procedure**	Select the Payment Type, Payment Mode from select box Enter the form no & press enter Enter Chalan no and amount paid.
4	**Test data**	Form no, Chalan no, Amount paid
5	**output**	Amount paid is less than or equal to total amount.

4.2	Data Matrix				
	Input attr	Equivalence class partitioning		Boundary value analysis	
		valid	Invalid	Min	Max
1	Form no	a-z,A-z,0 to 9	blank, other than	6	15
			a-z,A-z,0 to 9		
2	Chalan no	a-z,A-z,0 to 9	blank, other than	2	10
			a-z,A-z,0 to 9		
3	Amount paid	number	all alphabetic	2	10

4.3	Test Case			
	Action	input	Expected output	
1	Enter Form No	FN001	valid	pass
		fa9$0*	Invalid	Pass
2	Enter chalan no	CH8785	valid	Pass
		CQ@#/	Invalid	Pass
3	Enter amount	43000	valid	Pass
		Asn32	Invalid	Pass
		-12000	Invalid	Pass
		@3200	Invalid	Pass

Test case 5

5.1	Test procedure for User Info Form	
1	**Test code id**	ECESO5
2	**PURPOSE**	To enter required details about newly created user
3	**Procedure**	Enter Form no, first, middle, last name of user. Enter email id. The mentioned inputs are stored to the database as per user id
4	**Test data**	user_id, First name , middle name, Last name
5	**Output**	User info stored successfully

5.2	Data Matrix				
	Input attr	Equivalence class partitioning		Boundary value analysis	
		valid	Invalid	Min	Max
1	Form no	a-z,A-z,0 to 9	blank, other than a-z,A-z,0 to 9	6	15
2	First name	alphabetic and space	all special symbol number	2	30
3	Middle name	alphabetic and space	all special symbol number	2	30
4	Last name	alphabetic and space	all special symbol number	2	30
5	Email	Alphanumeric And special Characters	Repeated use of @,.com	2	50

5.3	Test Case			
	Action	input	Expected output	
1	Enter first	Prasanna	valid	pass
	name	Prasanna65	Invalid	pass
2	Enter middle	Sham	valid	pass
	name	Shamm421	Invalid	pass
3	Enter last name	Shaha	valid	pass
		Sh*&12	Invalid	Pass
4	Enter Email	Sham12@gmail		
		.com	valid	Pass
		Sham@12@g	Invalid	Pass
		Sham.com	Invalid	Pass
		Sher.co@gmail	Invalid	Pass