

CHAPTER 1

INTRODUCTION

The project titled as “ONLINE FEE PAYMENT SYSTEM “is a web based application. An institute have different branches at different locations want to control and maintain the accountant salary and student personal and payment details. Web application provides facility for reporting, new student details, payment details ,and modify details of student and salary of the accountant. It allows you to make payment of your fees in your own currency giving you transparency and a clear understanding of the cost of course fee. Make the payment using internet/telephone banking or at your local bank. Ensures that 100% of the funds transferred reach the college and that no expensive bank charges are deducted that may delay your registration. Ensures that the payment can be easily identified and allocated by the College giving you peace of mind. It is a secure and quick way to make the transfer ensuring that the College receives the payment promptly. You will be notified upon receipt of your payment.

1.1 Objective

We know that how difficult the process of paying the fees of colleges because there is many student which are also pay the fees in the bank so we have to stand for long time and it is a very time consuming process .Some parents which live in other villages or cities they have to cover long distance only for paying the fees of colleges .So to overcome this problem we developed the system in that system we can pay the online fees without going to the college bank .This process is easy as we can directly transfer the amount from our account to the college bank account. To review the existing system used in paying fees so that its strength and weaknesses are identified. To design a new system that enables students and their to pay fees online from wherever they are using credit and debit cards. To implement the prototype of the designed system. To test and validate the system prototype.

1.2 Existing System

We know that how difficult the process of paying the fees of colleges because there is many student which are also pay the fees in the bank so we have to stand for long time and it is a very time consuming process .Some parents which live in other villages or cities they have to cover long distance only for paying the fees of colleges .So to over this problem we developed the system in that system we can pay the online fees without going to the college bank .This process is easy as we can directly transfer the amount from our account to the college bank account.

1.3 Proposed System

In order to overcome the drawbacks in existing system we propose a new System. In that we can transfer the amount online .So it is very easy and time saving process, and the Each user will have its own enrollment number and password for login. And for checking the balance and transfer the amount the user needs to know its account number, user name and password. In this project the we give the access to the admin and user when the user want to perform certain operation like checking the balance, transfer the amount .To perform this operation the user needs to enter the enrollment number and password. then the user enters in to the system and transfer and check his balance. To check the balance there is need to enter the account number, user name and password then it will show the current balance of user if we enter wrong user name and password it will show the error message to the user .for transferring amount there is need to enter the account number, user name ,password of college and amount then the transaction is completed and it will show the message that transaction is successful.

CHAPTER 2

ANALYSIS

It gives the information regarding analysis done for the proposed system. System Analysis is done to capture the requirement of the user of the proposed system. It also provides the information regarding the existing system and also the need for the proposed system. The key features of the proposed system and the requirement specifications of the proposed system are discussed below.

2.1 Hardware Requirements

Processor type	:	Pentium IV-compatible processor or faster
Processor speed	:	Minimum 2.0 GHz or faster
RAM	:	512 MB or more
HARD DISK	:	50GB
Monitor	:	VGA or higher resolution 800x600 or higher resolution
Pointing device	:	Microsoft Mouse or compatible pointing device
CD-ROM	:	Actual requirements will vary based on system configuration
Keyboard	:	110 keys enhanced

2.2 Software Requirements

Operating system	:	windows 10
Database	:	MySQL
Front End	:	HTML,CSS
Back End	:	Python 3.8
Web Browser	:	chrome, internet explorer

2.3 Functional Requirements

Functional requirements are those requirements which deal with what the system should do or provide for users.

- Describes the behavior of the system as it relates to the system's functionality.
- Includes the description of the required functions, outlines of associated reports or online queries, and details of data to be held in the system. Specified by users themselves.

2.4 Non - Functional Requirements

Non-functional requirements are those requirements which elaborate the performance characteristic of the system and define the constraints on how the system will do so.

- Defines the constraints, targets or control mechanisms for the new system.
- Describes how, how well or to what standard a function should be provided.
- Specified by technical peoples e.g. Architect, Technical leaders and software developers.

CHAPTER 3

DESIGN

The process of design involves “conceiving and planning out in mind and making a drawing, pattern or a sketch”. The system design transforms a logical representation of what a given system is required to do into the physical reality during development. Important design factors such as reliability, response time, throughput of the system, maintainability, expandability etc., should be taken into account. Design constraints like cost, hardware limitations, standard compliance etc., should also be dealt with it. The task of system design is to take the description and associate with it a specific set of facilities-men, machines (computing and other), accommodation, etc., to provide complete specifications of a workable system.

3.1 SCHEMA DAIGRAM

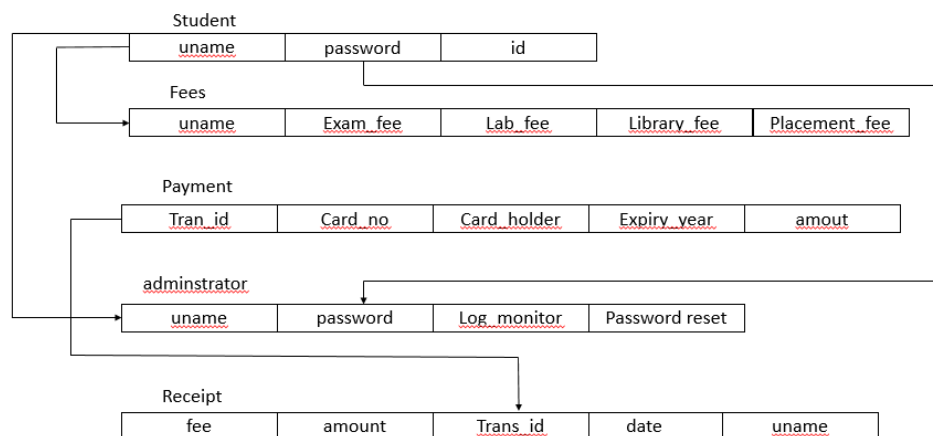


Fig 3.1: Schema diagram

ER DIAGRAM

An Entity Relationship(ER) Diagram is a type of flowchart that illustrates how “entities” such as people, object, or concepts relate to each other within a system. An Entity Relationship diagram describes inter_ related things of interest in a specific domain of knowledge. An ER model is composed of entity types and specific relationship that can exit between instances of those entity types

3.2 ER DIAGRAM

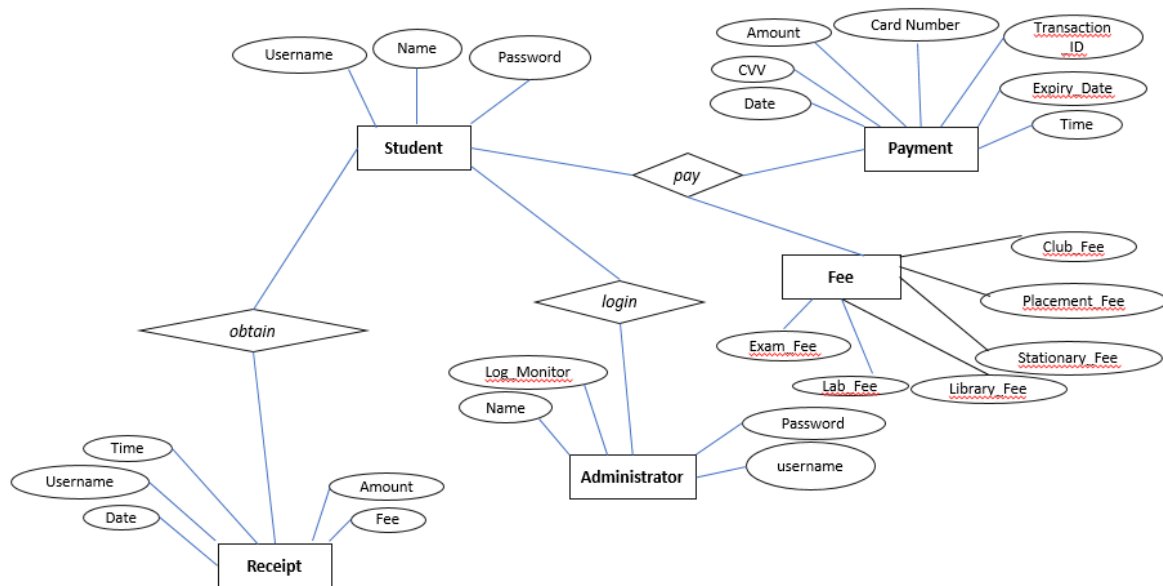









Fig 3.2: ER Diagram

3.3 TABLE










3.3.1 STUDENT_LOGIN

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
uname	VARCHAR(45)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
password	VARCHAR(100)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'0'
privilege	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'student'
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

3.3.2 FEES

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 uname	VARCHAR(45)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 exam_fee	INT(11)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'0'
 lab_fee	INT(11)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'0'
 club_fee	INT(11)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'0'
 placement_fee	INT(11)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'0'
 stationary_fee	INT(11)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'0'
 library_fee	INT(11)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'0'

3.3.3 PAYMENT

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 transaction_id	VARCHAR(100)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 uname	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'administrator'
 payment_type	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'exam_fee'
 amount_to_be_paid	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'0'
 amount_paid	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'0'
 amount_left	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'0'
 card_number	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'9999999999999999'
 cvv	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'999'
 expiry_month	INT(11)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

CHAPTER 4

IMPLEMENTATION

4.1 Tools used

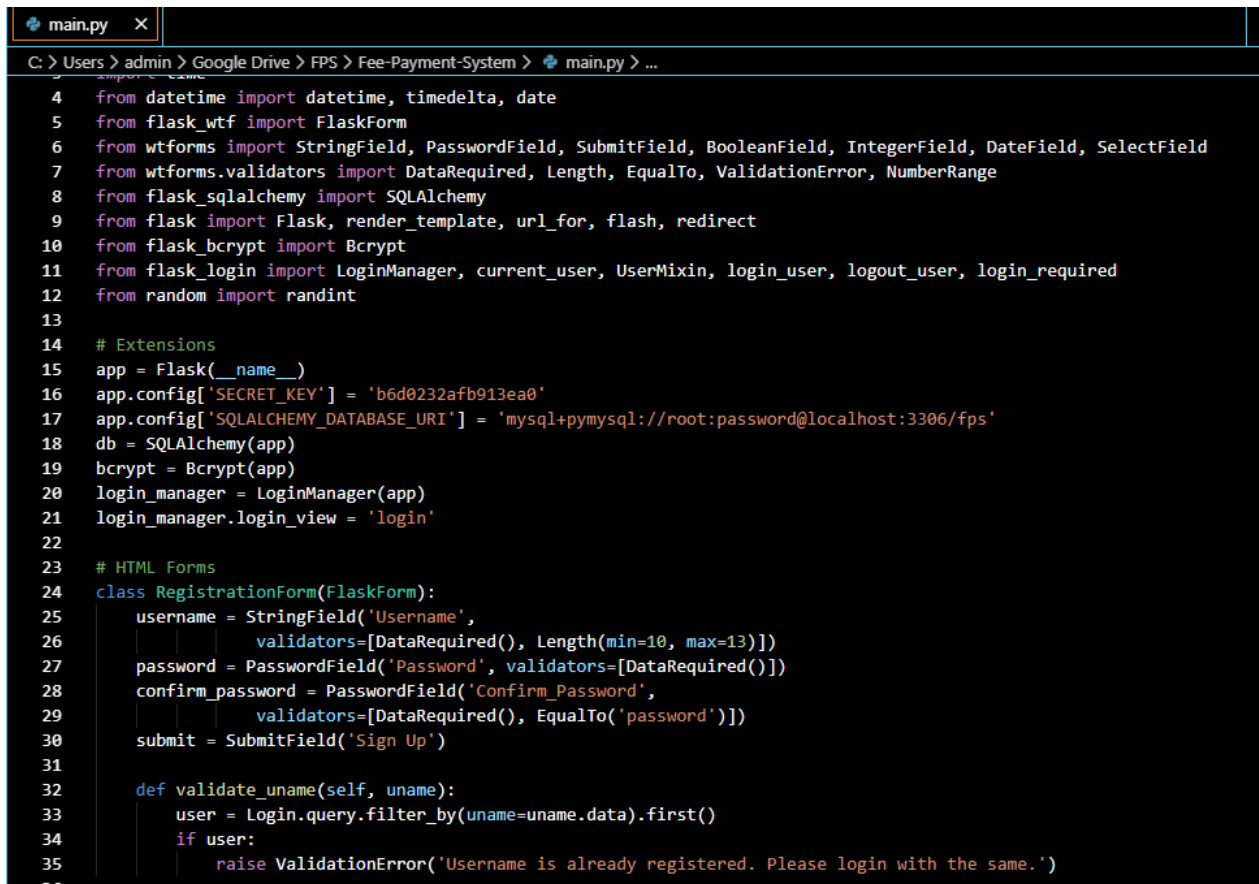
4.1.1 Flask

Flask is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. Extensions are updated far more frequently than the core Flask program.

Applications that use the Flask framework include Pinterest, LinkedIn, and the community web page for Flask itself.

- It Contains development server and debugger
- Integrated support for unit testing
- Uses Jinja2 templating
- Support for secure cookies (client side sessions)
- 100% WSGI 1.0 compliant
- Unicode-based
- Extensive documentation
- Google App Engine compatibility
- Extensions available to enhance features desired

4.2 Code Snippet



```

main.py x
C:\Users\admin> Google Drive > FPS > Fee-Payment-System > main.py > ...
4 from datetime import datetime, timedelta, date
5 from flask_wtf import FlaskForm
6 from wtforms import StringField, PasswordField, SubmitField, BooleanField, IntegerField, DateField, SelectField
7 from wtforms.validators import DataRequired, Length, EqualTo, ValidationError, NumberRange
8 from flask_sqlalchemy import SQLAlchemy
9 from flask import Flask, render_template, url_for, flash, redirect
10 from flask_bcrypt import Bcrypt
11 from flask_login import LoginManager, current_user, UserMixin, login_user, logout_user, login_required
12 from random import randint
13
14 # Extensions
15 app = Flask(__name__)
16 app.config['SECRET_KEY'] = 'b6d0232afb913ea0'
17 app.config['SQLALCHEMY_DATABASE_URI'] = 'mysql+pymysql://root:password@localhost:3306/fps'
18 db = SQLAlchemy(app)
19 bcrypt = Bcrypt(app)
20 login_manager = LoginManager(app)
21 login_manager.login_view = 'login'
22
23 # HTML Forms
24 class RegistrationForm(FlaskForm):
25     username = StringField('Username',
26                             validators=[DataRequired(), Length(min=10, max=13)])
27     password = PasswordField('Password', validators=[DataRequired()])
28     confirm_password = PasswordField('Confirm Password',
29                                     validators=[DataRequired(), EqualTo('password')])
30     submit = SubmitField('Sign Up')
31
32     def validate_username(self, username):
33         user = Login.query.filter_by(username=username.data).first()
34         if user:
35             raise ValidationError('Username is already registered. Please login with the same.')
36

```

Fig 4.2 Code for creation of front page

4.1.2 MySQL

Structured Query Language is a special-purpose programming language designed . MySQL is based on a client-server model. The core of MySQL is MySQL server, which handles all of the database instructions . MySQL server is available as a separate program for use in a client-server networked environment and as a library that can be embedded (or linked) into separate applications. MySQL operates along with several utility programs which support the administration of MySQL databases. Commands are sent to MySQL Server via the MySQL client, which is installed on a computer. MySQL was originally developed to handle large databases quickly.

Although MySQL is typically installed on only one machine, it is able to send the database to multiple locations, as users are able to access it via different MySQL client interfaces. These interfaces send SQL statements to the server and then display the results.

CHAPTER 5

TESTING

Software testing is the process used to help identify the correctness, completeness, security and quality of developed computer software. With that in mind, testing can never completely establish the correctness of arbitrary computer software. In computability theory, a field of computer science, an elegant mathematical proof concludes that it is impossible to solve the halting problem, the question of whether an arbitrary program will enter an infinite loop, or halt and produce output. In other words, testing is criticism or comparison that is comparing the actual value with an expected one.

There are many approaches to software testing, but effective testing of complex products is essentially a process of investigation, not merely a matter of creating and following rote procedure. One definition of testing is “the process of questioning a product in order to evaluate it”, where the “questions” are things the tester tries to do with the product, and the product answers with its behavior in reaction to the probing of the tester. Although most of the intellectual processes of testing are nearly identical to that of review or inspection, the word testing is connoted to mean the dynamic analysis of the product, putting the product through its paces. The quality of the application can, and normally does, vary widely from system to system but some of the common quality attributes includes reliability, stability, portability, maintainability and usability.

5.1 Testing Strategies

Designing effective test cases is important but so is the strategy to use them to execute them. If it is conducted in haphazard manner time is wasted and unnecessary effort is expended. Thus it seems reasonable to establish a systematic strategy for testing software.

5.1.1 Unit testing

Unit testing involves the design of test cases that validates the internal program logic functionality properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the

application .It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results

5.1.2 Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfied, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

5.1.3Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items

Valid Input : valid user name and password which is already registered in database

Invalid Input : the user is not registered

Functions : if valid input, the user goes to login else an error is encountered

Output : get logged in and use the portal as a student or as a company

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

5.1.4 System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

5.1.5 Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

5.1.6 Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

5.1.7 Test Results: All the test cases mentioned above passed successfully. No defects encountered.

5.2 Test Cases

5.2.1 Test Case Specifications

- During the payment process try to change the payment gateway language
- After successful payment, test all the necessary components, whether it is retrieved or not
- Check what happens if payment gateway stops responding during payment
- During the payment process check what happens if the session ends
- During the payment process check what happens in the backend
- Check what happens if payment process fails

- Check the Database entries whether they store credit card details or not
- During the payment process check error pages and security pages
- Check settings of pop-up blocker, and see what happens if a pop-up blocker is on and off
- Between payment gateway and application check buffer pages
- Check on successful payment, a success code is sent to the application and a confirmation page is shown to the user
- Verify whether the transaction processes immediately or processing is hand to your bank
- After successful transaction check if the payment gateway returns to your application
- Check all format and messages when successful payment process
- Unless you don't have an authorization receipt from the payment gateway, good should not be shipped
- Inform the owner for any transaction processed through e-mail. Encrypt the content of the mail
- Check the amount format with currency format
- Check if each of the payment options is selectable
- Check if each listed payment option opens the respective payment option according to specification
- Verify whether the payment gateway defaults to the desired debit/credit card option

CHAPTER 6

RESULTS

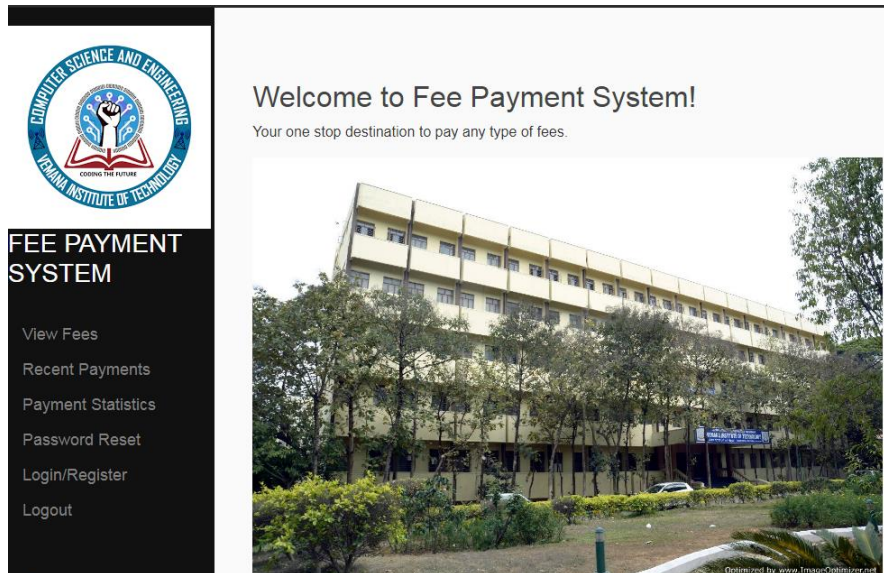


Fig 6.1-Home page

The screenshot displays the login interface of the Fee Payment System. At the top, a dark blue header bar contains the text 'Fee Payment System' on the left and a 'Login' link on the right. The main content area is light gray. A white-bordered box contains the login form. Inside this box, the heading 'Log In' is at the top. Below it are two input fields: 'Username' and 'Password'. Under the password field is a checkbox labeled 'Remember Me'. A blue 'Login' button is positioned below the checkbox. A link labeled 'Forgot Password?' is located at the bottom of the login box. Below the login box, the text 'Need An Account?' is followed by a blue 'Register Here' link.

Fig 6.2-Student login

Fee Payment System Login

Login Unsuccessful. Please verify username and password

Log In

Username
1vi17cs119

Password

☐ Remember Me

[Forgot Password?](#)

[Need An Account? Register Here](#)

Fig 6.3- Invalid Login and password

Fee Payment System Fees Statistics Logout

Fees Status:

#	Fees	Amount	Date	Payment
1	Exam Fees	34	05 January 2020	<input type="button" value="Pay"/>
2	Lab Fees	89	10 December 2019	<input type="button" value="Pay"/>
3	Library Fees	56	30 November 2019	<input type="button" value="Pay"/>
4	Stationary Fees	45	25 November 2019	<input type="button" value="Pay"/>
5	Placement Fees	34	02 February 2020	<input type="button" value="Pay"/>
6	Club Fees	23	22 March 2020	<input type="button" value="Pay"/>

Fig 6.4-Student Fee Status

#	USN	Date	Fees	Amount	Card	Transaction ID
1	1VI17CS114	27/11/2019	exam	1200	1234567890123456	1049716769
2	1VI17CS114	27/11/2019	exam	1200	123456789123456	1731734220
3	1VI17CS114	27/11/2019	exam	1200	1234567891234567	3757347201
4	1vi17cs116	19/8/2020	exam_fee	0	789	6789
5	1VI17CS114	27/11/2019	exam	1200	123456789123456	9854941303

Total Amount Received in this Date Range: 4800

Fig 6.5-calculating total amount received in this data range

Previous Payments:

USN	Date	Fees	Amount	Card	Transaction ID
1V117CS114	27/11/2019	exam	1200	1234567890123456	1049716769
1V117CS114	27/11/2019	exam	1200	123456789123456	1731734220
1vi17cs114	17/6/2019	exam_fee	0	234	23456
1V117CS114	27/11/2019	exam	1200	1234567891234567	3757347201
1V117CS114	27/11/2019	exam	1200	123456789123456	9854941303

Fig 6.6-Previous Payment

Password Reset

Reset Password of an account

Username

Password

Confirm_Password

Fig 6.7-Password Reset

CONCLUSION

In the end, conclude that there was a need for a system that eases the life of students in case of fees payment. This application eliminates all those unnecessary steps that were required before such as standing in long queues of the banks for DD, standing in long queues in college to submit the DD. This approach provides an efficient solution to all those problems. Of course, the application can later be extended for different purposes related to the college administration and various other activities but currently, it's limited to just paying the fees.

REFERENCES

BOOKS REFERRED:

- [1]. Gunderloy, Jorden BPB Publications (2000) - “Mastering SQL Server”
- [2]. Laura Thomson(5th Edition) - MySQL Web Development.
- [3]. Thereon Willis wrox publications(2000) - “Beginning SQL Server”

WEBSITES REFERRED:

<https://www.aukdc.edu.in/onlinefee>
www.guindytimes.com/articles/online-fees-payment
<https://www.netpnb.com/web/L001/webpages/annamalai.html>