CHAPTER 5

IMPLEMENTATION AND TESTING

5.1 Implementation Approaches

- **Waterfall model**

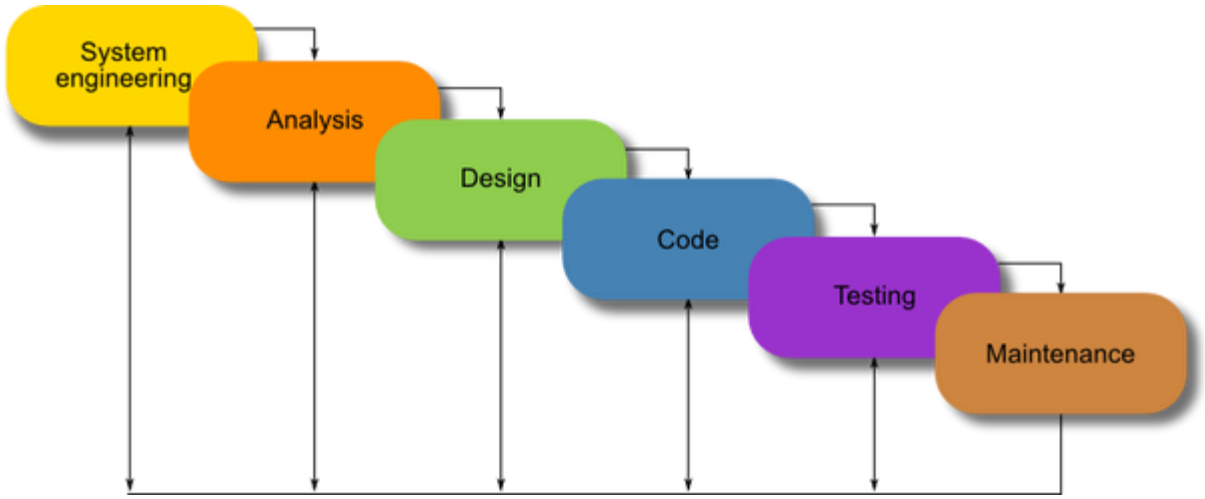


Figure:5.1 Waterfall Model Diagram

The Waterfall Model was the first Process Model to be introduced. It is very simple to understand and use. In a Waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases. The waterfall model is the earliest SDLC approach that was used for software development.

In "The Waterfall" approach, the whole process of software development is divided into separate phases. The outcome of one phase acts as the input for the next phase sequentially. This means that any phase in the development process begins only if the previous phase is complete. The waterfall model is a sequential design process in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Analysis, Design, Construction, Testing, Production/Implementation, and Maintenance.

- **Advantages**

  - The advantage of waterfall development is that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one.
  - The waterfall model progresses through easily understandable and explainable phases and thus it is easy to use.
  - It is easy to manage due to the rigidity of the model – each phase has specific deliverables and a review process.
  - In this model, phases are processed and completed one at a time and they do not overlap. The waterfall model works well for smaller projects where requirements are very well understood.

- **Disadvantages**

  - It is difficult to estimate time and cost for each phase of the development process.
  - Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought-out in the concept stage.
  - Not a good model for complex and object-oriented projects.
  - Not suitable for the projects where requirements are at a moderate to high risk of changing.

- **Iterative model**

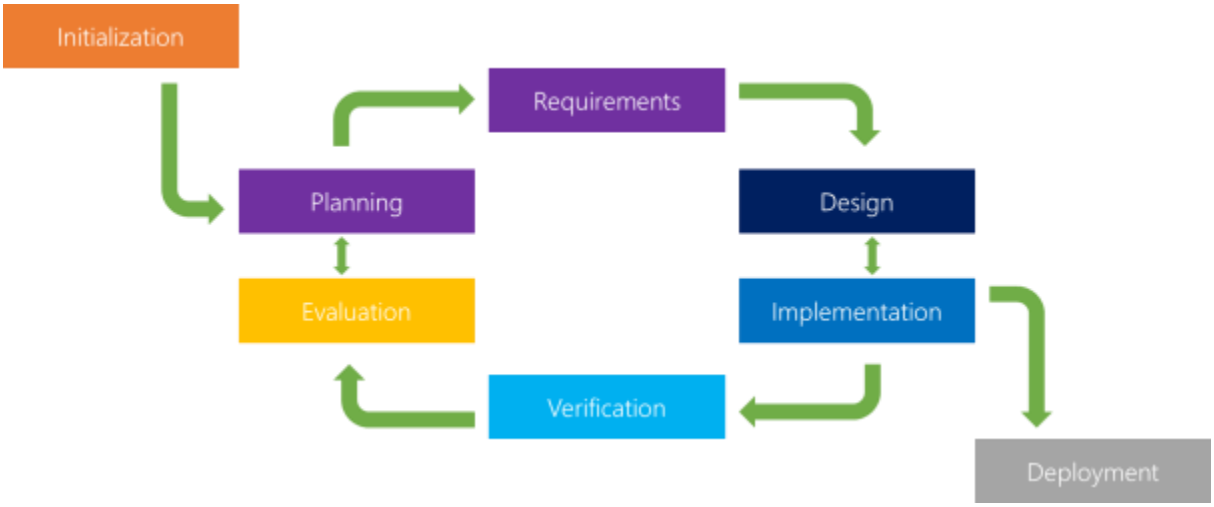


Figure:5.2 Iterative Model Diagram

The iterative model is a particular implementation of a software development life cycle (SDLC) that focuses on an initial, simplified implementation, which then progressively gains more complexity and a broader feature set until the final system is complete. When discussing the iterative method, the concept of incremental development will also often be used liberally and interchangeably, which describes the incremental alterations made during the design and implementation of each new iteration.

- Consider an iterative life cycle model which consists of repeating the following four phases in sequence:

A Requirements phase : in which the requirements for the software are gathered and analysed. Iteration should eventually result in a requirements phase that produces a complete and final specification of requirements.

A Design phase : in which a software solution to meet the requirements is designed. This may be a new design, or an extension of an earlier design.

An Implementation and Test phase : when the software is coded, integrated and tested.

A Review phase : in which the software is evaluated, the current requirements are reviewed, and changes and additions to requirements proposed.

- Advantages

  - Inherent Versioning
  - Rapid Turnaround
  - Suited for Agile Organizations
  - Easy Adaptability
  - Generates working software quickly and early during the software life cycle.
  - More flexible – less costly to change scope and requirements.
  - Easier to test and debug during a smaller iteration.
  - Easier to manage risk because risky pieces are identified and handled during its iteration.
  - Each iteration is an easily managed milestone.

- Dis Advantages

  - Costly Late-Stage Issues
  - Increased Pressure on User Engagement
  - Feature Creep
  - Each phase of an iteration is rigid and do not overlap each other.
  - Problems may arise pertaining to system architecture because not all requirements are gathered up front for the entire software life cycle.

- **V model**

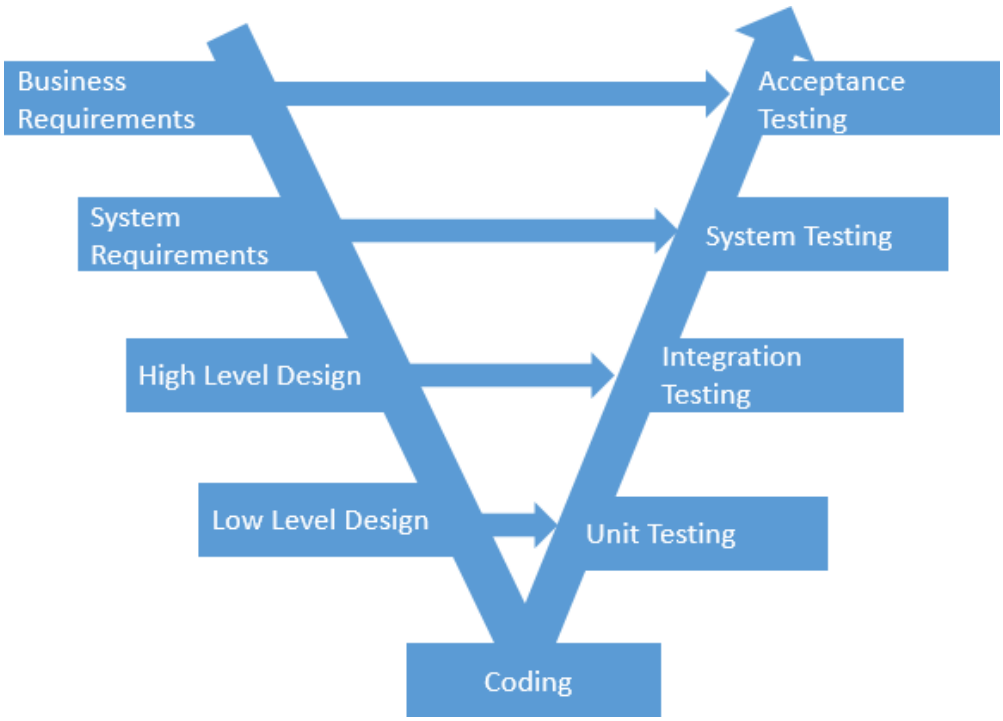


Figure 5.3 V Model Diagram

The V-model is an SDLC model where execution of processes happens in a sequential manner in a V-shape. It is also known as Verification and Validation model.

The V-Model is an extension of the waterfall model and is based on the association of a testing phase for each corresponding development stage. This means that for every single phase in the development cycle, there is a directly associated testing phase. This is a highly-disciplined model and the next phase starts only after completion of the previous phase.

- **Why preferred?**

It is easy to manage due to the rigidity of the model. Each phase of V-Model has specific deliverables and a review process.

Proactive defect tracking – that is defects are found at early stage.

- **When to use?**

  Where requirements are clearly defined and fixed.

  The V-Model is used when ample technical resources are available with technical expertise.

- **Advantages**

  - This is a highly-disciplined model and Phases are completed one at a time.
  - Works well for smaller projects where requirements are very well understood.
  - Simple and easy to understand and use.
  - Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.

- **Disadvantages**

  - High risk and uncertainty.
  - Not a good model for complex and object-oriented projects.
  - Poor model for long and ongoing projects.
  - Not suitable for the projects where requirements are at a moderate to high risk of changing.

- **Agile model**

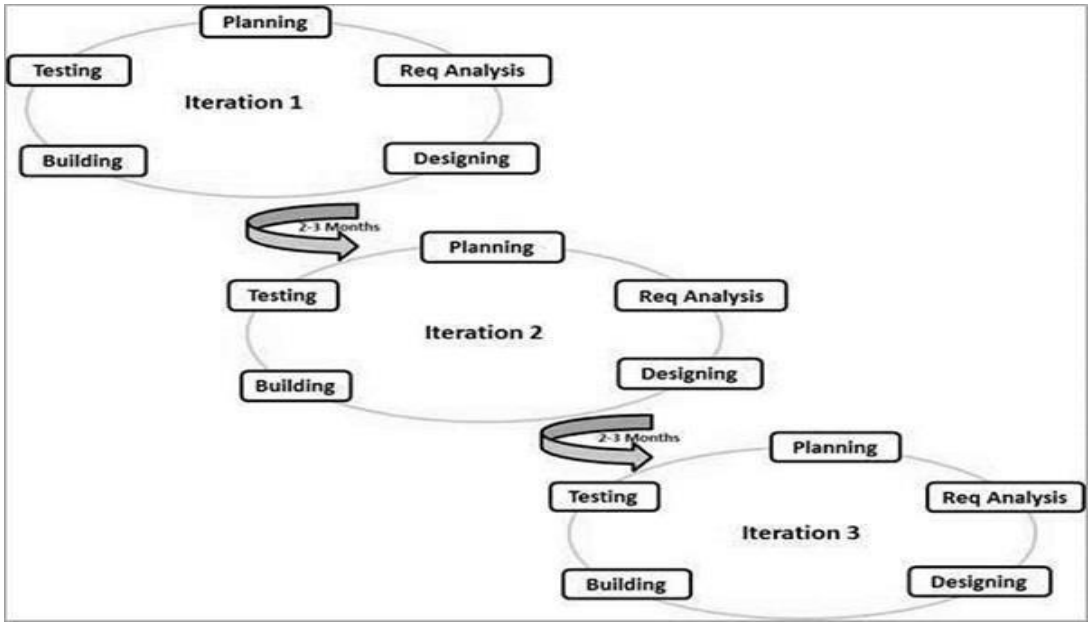


Figure 5.4 Agile Model Diagram

Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product. Agile Methods break the product into small incremental builds. These builds are provided in iterations. Each iteration typically lasts from about one to three weeks.

- **Agile Testing Methods:**
  - Scrum
  - Crystal
  - Dynamic Software Development Method(DSDM)
  - Feature Driven Development(FDD)
  - Lean Software Development
  - eXtreme Programming(XP)

- **Advantages**
  - It is a very realistic approach to software development.
  - Promotes teamwork and cross training.
  - Functionality can be developed rapidly and demonstrated.
  - Resource requirements are minimum.
  - Suitable for fixed or changing requirements
  - Delivers early partial working solutions.
  - Good model for environments that change steadily.
  - Minimal rules, documentation easily employed.

- **Dis advantages**
  - Not suitable for handling complex dependencies.
  - More risk of sustainability, maintainability and extensibility.
  - An overall plan, an agile leader and agile PM practice is a must without which it will not work.
  - Strict delivery management dictates the scope, functionality to be delivered, and adjustments to meet the deadlines.

- **RAD model**

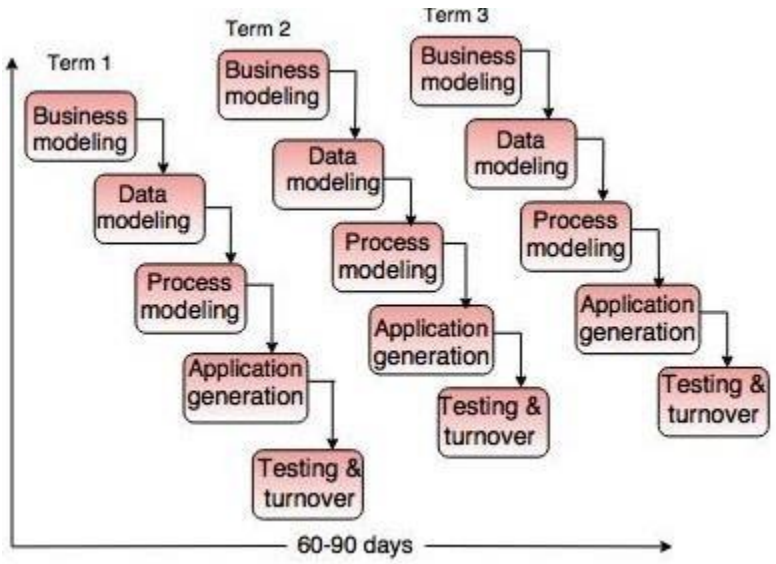


Figure 5.5 RAD Model Diagram

The RAD (Rapid Application Development) model is based on prototyping and iterative development with no specific planning involved. The process of writing the software itself involves the planning required for developing the product.

Rapid Application Development focuses on gathering customer requirements through workshops or focus groups, early testing of the prototypes by the customer using iterative concept, reuse of the existing prototypes (components), continuous integration and rapid delivery.

- The following pointers describe the typical scenarios where RAD can be used :

  - RAD should be used only when a system can be modularized to be delivered in an incremental manner.
  - It should be used if there is a high availability of designers for modeling.

- It should be used only if the budget permits use of automated code generating tools.
- RAD SDLC model should be chosen only if domain experts are available with relevant business knowledge.
- Should be used where the requirements change during the project and working prototypes are to be presented to customer in small iterations of 2-3 months.

- **Advantages**
  - Changing requirements can be accommodated.
  - Progress can be measured.
  - Iteration time can be short with use of powerful RAD tools.
  - Productivity with fewer people in a short time.
  - Reduced development time.
  - Increases reusability of components.

- **Dis advantages**
  - Only system that can be modularized can be built using RAD.
  - Requires highly skilled developers/designers.
  - High dependency on modelling skills.
  - Management complexity is more.
  - Suitable for systems that are component based and scalable.

- **Adopted approach:**

Iterative model

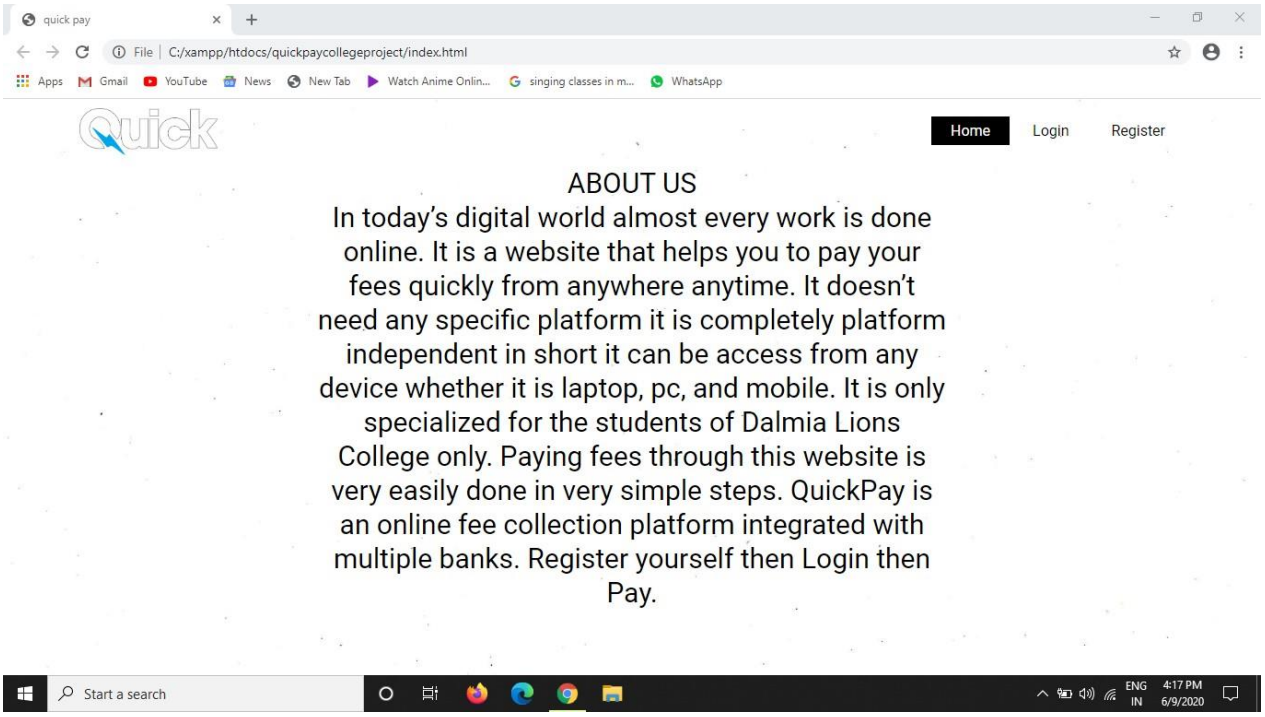
The iterative model is a particular implementation of a software development life cycle (SDLC) that focuses on an initial, simplified implementation, which then progressively gains more complexity and a broader feature set until the final system is complete. When discussing the iterative method, the concept of incremental development will also often be used liberally and interchangeably, which describes the incremental alterations made during the design and implementation of each new iteration.

Consider an iterative life cycle model which consists of repeating the following four phases in sequence:

- **A Requirements phase**: in which the requirements for the software are gathered and analyzed. Iteration should eventually result in a requirements phase that produces a complete and final specification of requirements.
- **A Design phase**: in which a software solution to meet the requirements is designed. This may be a new design, or an extension of an earlier design.
- **An Implementation and Test phase**: when the software is coded, integrated and tested.
- **A Review phase**: in which the software is evaluated, the current requirements are reviewed, and changes and additions to requirements proposed.

5.2 Coding details And Code efficiency

- **Home Page :**



- **Code:**

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>quick pay</title>
        <link href="style.css" rel="stylesheet" type="text/css">
        <link
href="https://fonts.googleapis.com/css?family=Roboto&display=swap"
rel="stylesheet">
        <link
href="https://fonts.googleapis.com/css?family=Open+Sans&display=swap"
rel="stylesheet"
</head>
```

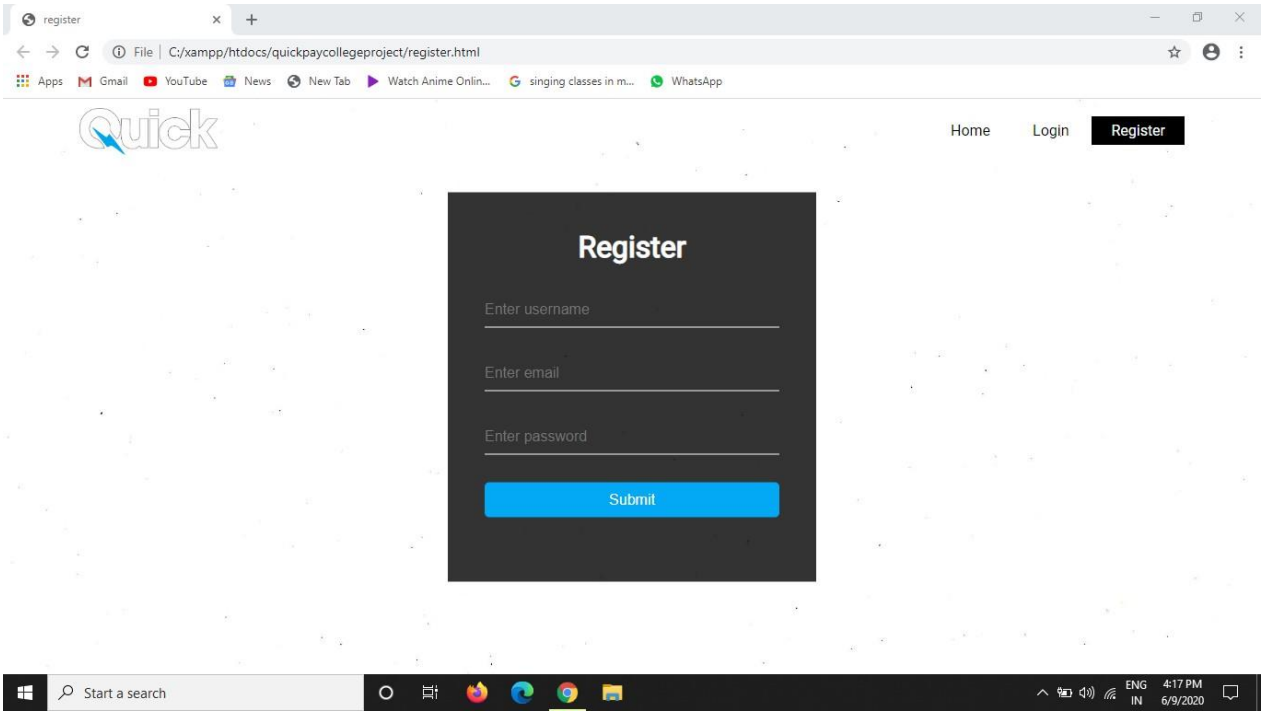
```
<body>
        <header>
        <div class="main">
                <div class="logo"><img src="newLogoPNG.png"
alt="logo"></div>
                <ul>
                <li class="active"><a href="index.html">Home</a></li>
                        <li><a
href="http://localhost/quickpaycollegeproject/login.html">Login</a></li>
                        <li><a
href="http://localhost/quickpaycollegeproject/register.html">Register</a></li>

                </ul>

                </div>
                <div class="intro">

                <p>ABOUT US<br>
In today’s digital world almost every work is done online. It is a website that
helps you to pay your fees quickly from anywhere anytime. It doesn’t need any
specific platform it is completely platform independent in short it can be access
from any device whether it is laptop, pc, and mobile.
It is only specialized for the students of Dalmia Lions College only. Paying fees
through this website is very easily done in very simple steps.
QuickPay is an online fee collection platform integrated with multiple banks.
Register yourself then Login then Pay.
</p>
                </div>

        </header>
```

```
</body>
</html>
```

- **Functionality:**

The above GUI is for user. There is two options for user login and registration. There is details to know user how to use this application. how can it helps users to done payment in few minutes. If user already register and they want to see their details they can easily login by clicking on the login button and enter valid username and password .

➢ Registration Page :



➢ Code :

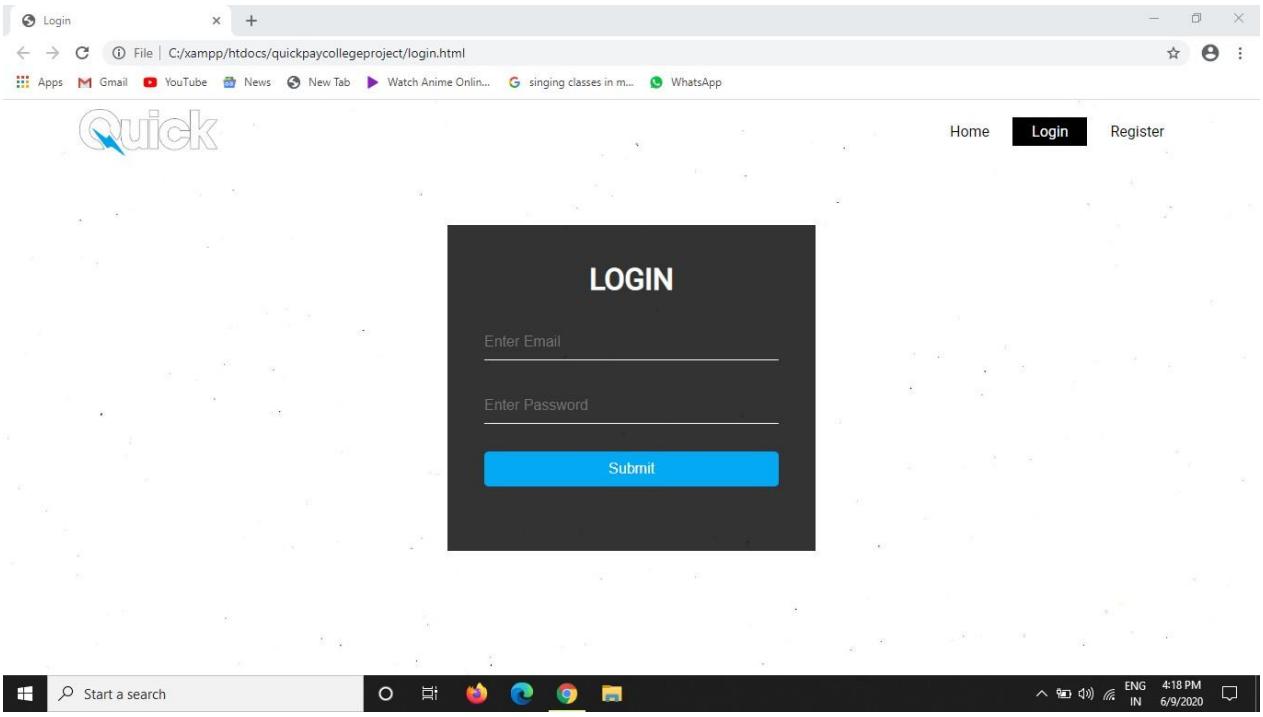
```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>register</title>
<link rel="stylesheet" type="text/css" href="style.css">
        <link href="https://fonts.googleapis.com/css?family=Roboto&display=swap"
rel="stylesheet">
</head>
<body>
<header>
<div class="main">
                <div class="logo">
                <img src="newLogoPNG.png" alt="logo">

                </div>
```

```
                <ul>
                <li ><a href="index.html">Home</a></li>
                        <li><a
href="http://localhost/quickpaycollegeproject/login.html">Login</a></li>
                        <li class="active"><a
href="http://localhost/quickpaycollegeproject/register.html">Register</a></li>


                </ul>
</div>
<div class="form-container">
        <h1>Register</h1>
        <form method="post" action="functions.php"
        <form>
        <input type="text" name="username" placeholder="Enter username">
                <input type="email" name="email" placeholder="Enter email">
                <input type="password" name="password" placeholder="Enter
password">
                <input type="submit" name="submit-signup" class="submit-btn">
        </form>
                </div>
</header>
</body>
</html>
```

- **Functionality:**

  when user click on the register button on the home page this gui will open . where user have to enter valid details.

- **Login Page :**



- **Code:**

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Login</title>
                    <link rel="stylesheet" type="text/css" href="style.css">
                    <link
href="https://fonts.googleapis.com/css?family=Roboto&display=swap"  rel="stylesheet">
</head>
<body>
<header>
<div class="main">
                     <div class="logo">
                     <img src="newLogoPNG.png" alt="logo">

                     </div>
```

```
                    <ul>
                    <li ><a href="index.html">Home</a></li>
                        <li class="active"><a
href="http://localhost/quickpaycollegeproject/login.html">Login</a></li>
                        <li><a
href="http://localhost/quickpaycollegeproject/register.html">Register</a></li>

                    </ul>



                </div>
<div class="form-container">
                    <h1>LOGIN</h1>
                    <form method="post" action="functions.php"
                    <form>

                     <input type="email" name="email" placeholder="Enter Email">
                     <input type="password" name="password" placeholder="Enter
Password">
                     <input type="submit" name="submit-login" class="submit-btn">
                    </form>



                    </div>
                    </header>

</body>
</html>
```

- **Functionality:**

  Once user registered they get user id and password to login. They have to enter valid login id and password to login successfully.

- **Coding of database connectivity:**

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "quickpay";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
die("Connection failed: " . $conn->connect_error);
}

?>
```

5.3 Testing Approach

- **SOFTWARE TESTING**

SOFTWARE TESTING is defined as an activity to check whether the actual results match the expected results and to ensure that the software system is defect free. It involves execution of a software component or system component to evaluate one or more properties of interest. Software testing also helps to identify errors, gaps or missing requirements in contrary to the actual requirements.

- **Types of Software testing**

- **STATIC TESTING:**

STATIC TESTING is a software testing technique by which we can check the defects in software without actually executing it. Its counter-part is Dynamic Testing which checks an application when the code is run.Static testing is done to avoid errors at an early stage of development as it is easier to find sources of failures then failures themselves.Static testing helps to find errors that may not be found by Dynamic Testing.

- The two main types of static testing techniques are
  - **Manual examinations**:

Manual examinations include analysis of code done manually, also known as REVIEWS**.**

  - **Automated analysis using tools:**

Automated analysis are basically static analysis which is done using tools.

- **DYNAMIC TESTING**

DYNAMIC TESTIN Gis defined as a software testing type, which checks the dynamic behaviour of the code is analysed.The main aim of the Dynamic tests is to ensure that software works properly during and after the installation of the software ensuring a stable application without any major flaws(this statement is made because no software is error free, testing only can show presence of defects and not absence) Dynamic Testing is classified into two categories

- White Box Testing
- Black Box Testing

- **FUNCTIONAL TESTING**

Functional testing is a type of software testing whereby the system is tested against the functional requirements/specifications.

Functions (or features) are tested by feeding them input and examining the output. Functional testing ensures that the requirements are properly satisfied by the application. This type of testing is not concerned with how processing occurs, but rather, with the results of processing. It simulates actual system usage but does not make any system structure assumptions.

During functional testing, Black Box Testing technique is used in which the internal logic of the system being tested is not known to the tester.

Functional testing is more effective when the test conditions are created directly from user/business requirements.

- **NON-FUNCTIONAL TESTING**

Non-functional testing is defined as a type of Software testing to check non-functional aspects (performance, usability, reliability, etc.) of a software application. It is designed to test the readiness of a system as per non-functional parameters which are never addressed by functional testing.

- Non-functional testing should increase usability, efficiency, maintainability, and portability of the product.
- Helps to reduce production risk and cost associated with non-functional aspects of the product.
- Optimize the way product is installed, setup, executes, managed and monitored.
- Collect and produce measurements, and metrics for internal research and development.
- Improve and enhance knowledge of the product behaviour and technologies in use.

- **BLACK BOX TESTING**

Black box testing is defined as a testing technique in which functionality of the Application Under Test (AUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on software requirements and specifications.

Figure 5.6 Black-Box Diagram

In Black Box Testing we just focus on inputs and output of the software system without bothering about internal knowledge of the software program.

The above Black-Box can be any software system you want to test. For Example, an operating system like Windows, a website like Google, a database like Oracle or even your own custom application. Under Black Box Testing, you can test these applications by just focusing on the inputs and outputs without knowing their internal code implementation.

- **WHITE BOX TESTING**

White box testing is testing of a software solution's internal structure, design, and coding. In this type of testing, the code is visible to the tester. It focuses primarily on verifying the flow of inputs and outputs through the application, improving design and usability, strengthening security. White box testing is also known as Clear Box testing, Open Box testing, Structural testing, Transparent Box testing, Code-Based testing, and Glass Box testing. It is usually performed by developers.

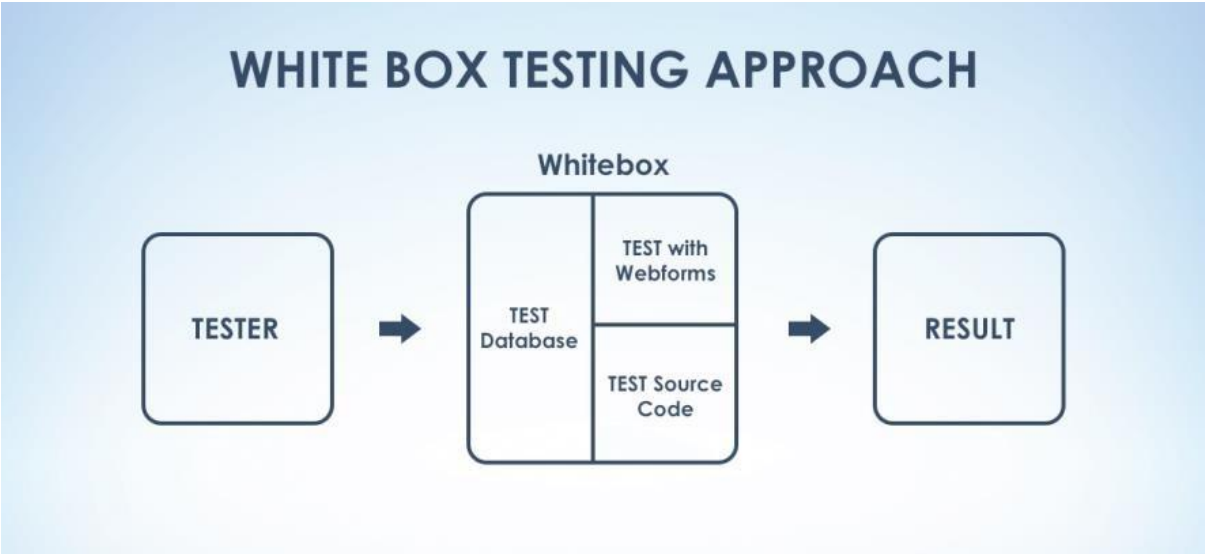


Figure 5.7 White-Box Diagram

It is one of two parts of the Box Testing approach to software testing. Its counterpart, Black box testing, involves testing from an external or end-user type perspective. On the other hand, White box testing is based on the inner workings of an application and revolves around internal testing.

The term "WhiteBox" was used because of the see-through box concept. The clear box or WhiteBox name symbolizes the ability to see through the software's outer shell (or "box") into its inner workings.

- **Levels of testing**
  - Unit/component
  - Integration
  - System
  - Acceptance alpha beta
- **Unit/component testing**

The most basic type of testing is unit, or component, testing.

Unit testing aims to verify each part of the software by isolating it and then perform tests to demonstrate that each individual component is correct in terms of fulfilling requirements and the desired functionality.

This type of testing is performed at the earliest stages of the development process, and in many cases it is executed by the developers themselves before handing the software over to the testing team.

The advantage of detecting any errors in the software early in the day is that by doing so the team minimises software development risks, as well as time and money

- **Integration testing**

Integration testing aims to test different parts of the system in combination in order to assess if they work correctly together. By testing the units in groups, any faults in the way they interact together can be identified.

There are many ways to test how different components of the system function at their interface; testers can adopt either a bottom-up or a top-down integration method.

In bottom-up Integration Testinng, testing builds on the results of unit testing by testing higher-level combination of units (called modules) in successively more complex scenarios.

It is recommended that testers start with this approach first, before applying the top-down approach which tests higher-level modules first and studies simpler ones later.

- **System testing**

The next level of testing is system testing. As the name implies, all the components of the software are tested as a whole in order to ensure that the overall product meets the requirements specified.

System testing is a very important step as the software is almost ready to ship and it can be tested in an environment which is very close to that which the user will experience once it is deployed.

System testing enables testers to ensure that the product meets business requirements, as well as determine that it runs smoothly within its operating environment. This type of testing is typically performed by a specialized testing team.

- **Acceptance testing**

Finally, acceptance testing is the level in the software testing process where a product is given the green light or not. The aim of this type of testing is to evaluate whether the system complies with the end-user requirements and if it is ready for deployment.

The testing team will utilise a variety of methods, such as pre-written scenarios and test cases to test the software and use the results obtained from these tools to find ways in which the system can be improved.

The scope of acceptance testing ranges from simply finding spelling mistakes and cosmetic errors, to uncovering bugs that could cause a major error in the application.

- **Alpha testing**

Alpha Testing is a type of software testing performed to identify bugs before releasing the product to real users or to the public. Alpha Testing is done at developer site.

- **Beta testing**

Beta Testing is performed by real users of the software application in a real environment. Beta testing is done at customer site.

- **Testing approach used in our project:**
  Unit Testing, Integrated Testing and Acceptance Testing
- **Unit Testing**: Here, in my project I have done unit testing. Different types of units were divided into their specific similar categories and were tested one after the other. Various bugs and errors where found but later they were debugged.

  For e.g.: Below units were tested

  Module interface
  Loops
  Validation test
  Database etc.
- **Integrated Testing**: After testing various units in my project I integrated different components in the project.

  Admin and User Login
- **Acceptance Testing:** In Acceptance testing I have performed Beta testing in my project. I have tested the project with my guide and people who had similar projects like me. I got various feedback and suggestion from the people who tested my project. I have collected the response and analyzed on it. This helped to make my project way better and made help me to eliminate huge coding's and hidden waste material and add some important features such as Design etc.

5.4 Test Cases

- **Registration:**

Test Case Id	First name	Last name	Email	Registration (mobile) no	Expected Result	Actual Result	Status
T1	Swara	chavan	swarachava@gmail.com	9876543210	Login Successful	Login Successful	Pass
T2	_____	chavan	swarachava@gmail.com	9876543210	Enter valid user fname	Login failed	fail
T3	Swara	_____	swarachava@gmail.com	9876543210	Enter valid lname	Login failed	fail
T4	Swara	chavan	swarachava@gmail.com	987654321	Enter valid mobile number	Login failed	fail
T5	_____	_____	_____	_____	Enter valid details	Login failed	fail
T6	Swara	chavan	Swarachava1726@gmail.com	9876543210	Enter valid email id	Login failed	fail
T7	Swara1	chavan	swarachava@gmail.com	9876543210	Enter valid fname this field cannot contain any number	Login failed	fail
T8	Swara	Chavan7	swarachava@gmail.com	9876543210	Enter valid lname this field cannot contain any number	Login failed	fail
T9	$swara	chavan	swarachava@gmail.com	9876543210	Do not contain	Login failed	fail

					special character		
T10	Swara	$chavan	swarachava@gmail.com	9876543210	Do not contain any special character	Login failed	fail
T11	Swara	chavan	swarachavagmail@.com	9876543210	Please enter email id in proper format	Login failed	fail

Table 5.1 Registration Page

- **User login:**

Test Case Id	Username/Email	Password/Register no	Expected Result	Actual Result	Status
T1	swarachava@gmail.com	9876543210	Login Successful	Login Successfull	Pass
T2	swar@gmail.com	9876543210	Enter valid usernaeme/Email	Login failed	fail
T3	swarachava@gmail.com	9876543201	Invalid password	Login failed	fail
T4	_____	_____	Enter valid username & password	Login fail	fail

Table 5.2 User Login

- **Admin Login:**

Test Case Id	Username	Password	Expected Result	Actual Result	Status
T1	Admin	Admin	Login successful	Login successfull	pass
T2	Admin	Root	Invalid password	Login failed	fail
T3	Admin	Admin	Invalid usename	Login fail	fail
T4	_____	_____	Enter valid username & password	Login fail	fail

Table 5.3 Admin Login

a

Chapter 6

Results and discussions

6.1 Test reports

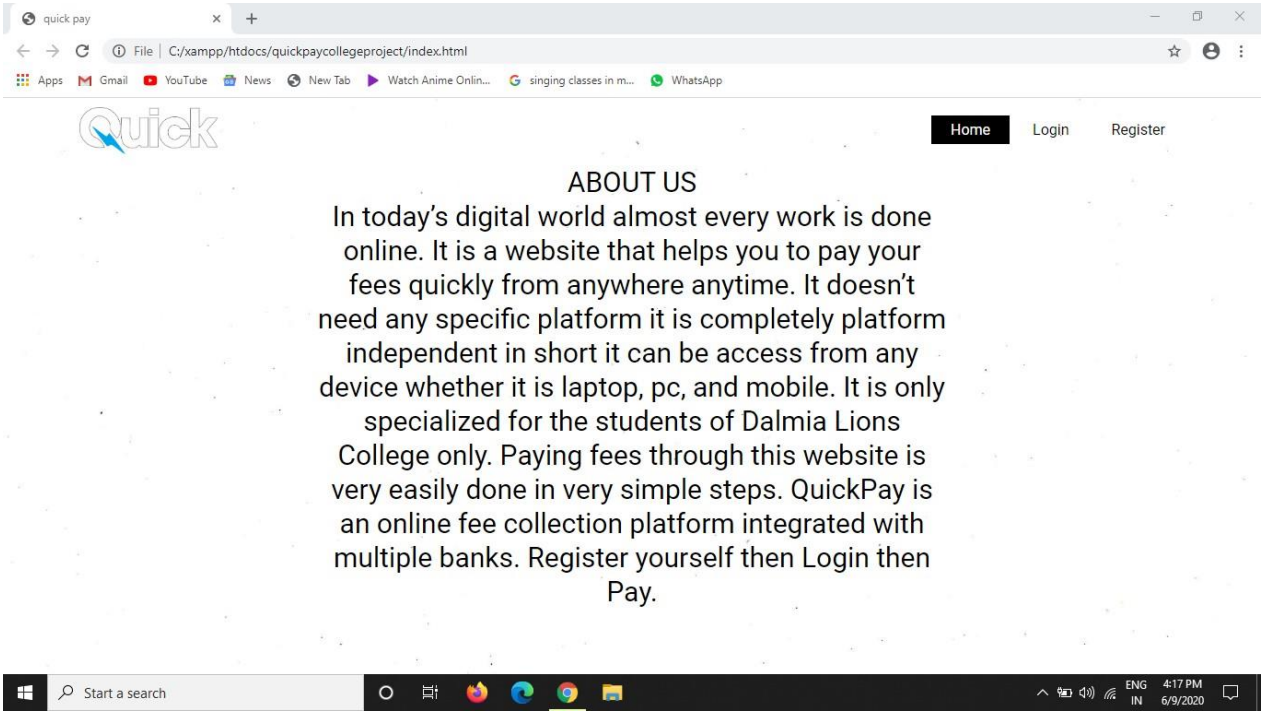
Test Report

Test cycle system test

EXECUTED	PASSED			50	
	FAILED			5	
	(Total) TEST EXECUTED (PASSED 128 + FAILED)				55
PENDING					2
INPROGRESS					3
BLOCKED					0
(Sub-Total) TEST PLANNED					60
(PENDING + IN PROGRESS + BLOCKED + TEST EXECUTED)					

6.2 User Documentation

The QUICKPAY is consist of two panels which are Admin panel, student or user panel. The student panel are based on any applications. The Admin panel is the main panel that generates the excel sheet of the students who pay the fees.



- **Working of system:**

First click on the given link to start.

If you are not registered then first click on register button on the top of the home page.

Fill proper and true details.

Once you register you will get user id and password to login.

You can login any time.

When you do payment must check your internet connection is properly working or not.

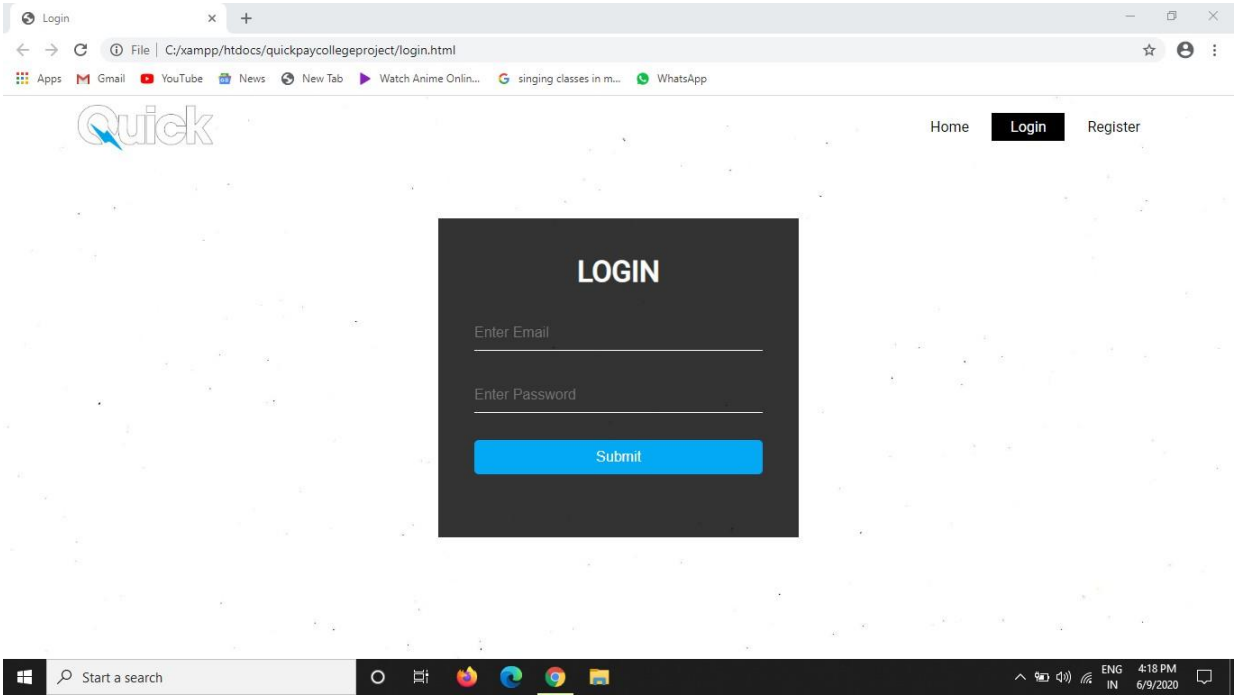
User can login via the user id which is their email id and password .

User can do payment after login.

User can check details of their payment.

User get receipt when they done payment they can download easily.

If user want to change their details they can change easily.



CHAPTER 7

CONCLUSION

7.1 Conclusion

It has been an amazing experience for me while working on this project. There are many things that I learnt while developing this project. Design patterns, security, testing, debugging and a user's perspective, etc are some of the things that I learnt while working on this project. Class diagram, Use case diagram, Activity diagram, etc were of great help during the development stage of the project. These diagrams have shown me the importance of planning, resource gathering as well as the importance of these various diagrams in the software development cycle. There were a few bugs that I found through rigorous testing which have been patched. These all experiences will surely help in my future projects.

The main objective of the website is to pay the fees online instantly in faster and smarter way it is a platform independent student can pay fees from anywhere at anytime. the website will be simplest and beneficial to students for paying their college fees the requirement to use this website is just basic knowledge of internet, security.website will give you decent amount of security. Scope of this website is to pay the fees in fastest way.this website will have an effect which will make an step towards the paperless work.

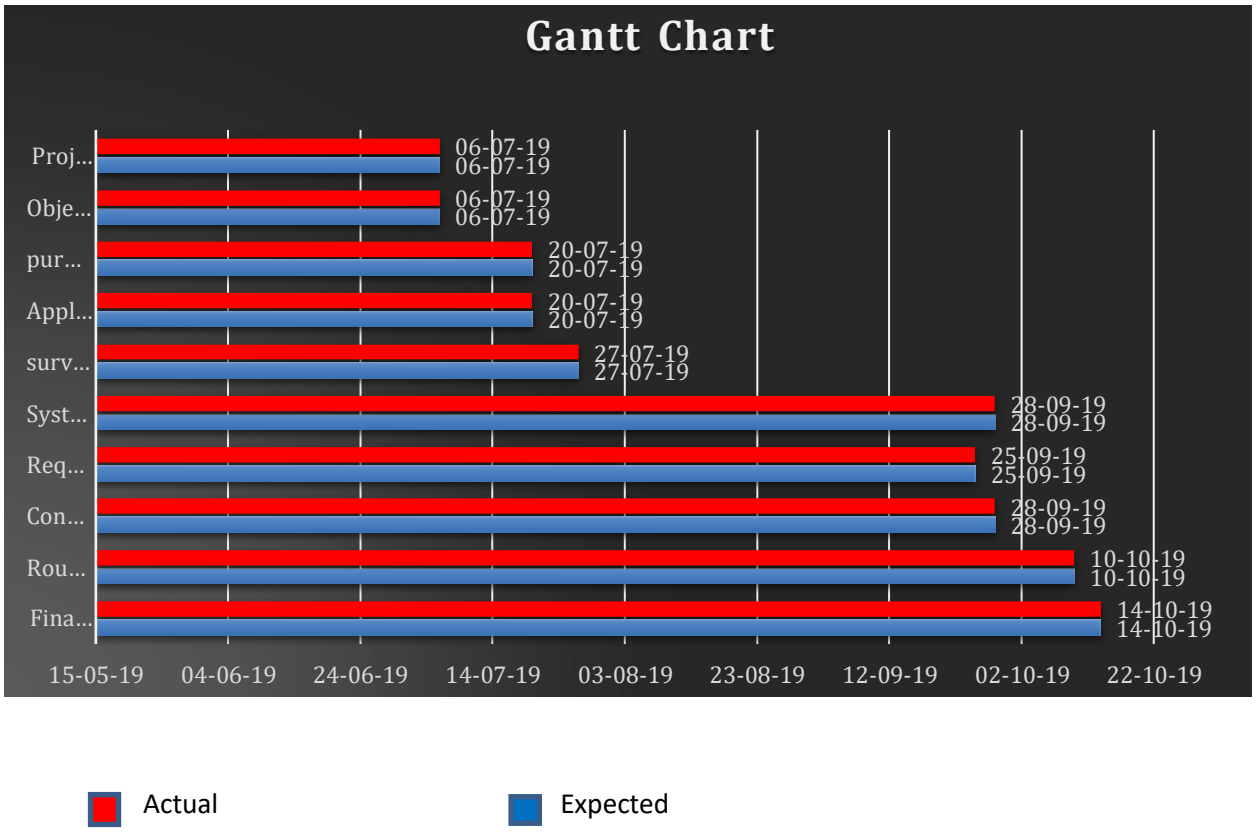
7.2 Limitations of the system

- Apart from the several advantages of the application there are certain limitations in the software which are also focused.
- Website is not completely secure.
- The system sometimes not be able to respond because of the load on the system.
- If server gets crash then entire data maybe lost.
- A website will be probably be slower than an application hosted on your company's server.you need to decide if a slight reduction in speed is worth the worldwide access.

7.3 Future Scope of the Project

- Performance can be increased in terms of speed and memory.
- Since the project is totally based on payment so there should be a good security from getting hacked.
- The current server can not handle the large number of client at the same time, so a better server is needed to get this problem solved.
- The Project can be expanded by including other cities and states.
- There is a huge scope of enhancement in the project by adding functionalities to the system.
- the project can be enhanced in the near future by adding more modules and can be implemented for better use.
- Any colleges can use this website if we can change some of the code.
- We are not added the payment gateway now if any college want this website they can add payment gateway and use this website.

Gantt Chart


Gantt Chart
Proj...
Obje...
pur...
Appl...
surv...
Syst...
Req...
Con...
Rou...
Fina...
06-07-19
06-07-19
06-07-19
06-07-19
20-07-19
20-07-19
20-07-19
20-07-19
27-07-19
27-07-19
28-09-19
28-09-19
25-09-19
25-09-19
28-09-19
28-09-19
10-10-19
10-10-19
14-10-19
14-10-19
15-05-19
04-06-19
24-06-19
14-07-19
03-08-19
23-08-19
12-09-19
02-10-19
22-10-19
Actual
Expected

References

- www.tutorialspoint.com
- www.studytonight.com
- www.youtube.com
- www.w3school.com
- https://en.wikipedia.org/wiki/php
- https://en.wikipedia.org/wiki/java
- www.codeproject.com