**ABSTRACT**

The objective of this project is to create a client-side website to view and register for different services which are offered by an NGO or a donation center. The portal will consist of a home page which will direct to other pages.

One of the pages will contain information about the organization, agenda and motive. It will be a simple static page. The second page is a registration form for applying to open a sub NGO under the organization with or without a partnership. The third page leads to a donation portal which asks the user to enter his/her details and the specification of the object donated. The second and third page is connected to a backend.

The backend will consist of multiple tables. The first one will record the details entered to register for a new NGO. The second table will consist of information regarding the donor and the project donated.

The portal will also consist of a button on one of the pages that will enable the user to check the status of the registration of the NGO and also the status of the offered donation.

**CONTENT**

|  |  |  |
| --- | --- | --- |
|  | Abstract | 1 |
|  | Chapter 1: INTRODUCTION | 4 |
|  | Chapter 3: ABOUT THE PROJECT | 8 |
|  | Chapter 4:REQUIREMENT ANALYSIS | 9 |
|  | Chapter 5: FEASIBILITY STUDY | 10 |
|  | Chapter 6:ANALYSIS AND DESIGN | 11 |
|  | Chapter 7: DEPLOYMENT AND DEVELOPMENT | 15 |
|  | References | 19 |
|  | Plagiarized Report | 20 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**TABLE OF FIGURES**

|  |  |  |
| --- | --- | --- |
| S No. | About the Figure | Page No. |
| 1 | Fig 1: Software Development Life Cycle | 4 |
| 2 | Fig 2: Iterative Development | 5 |
| 3 | Fig 3: Waterfall Model | 7 |

4 Fig 4: Iterative Model 7

5 Fig 5: Class Diagram 11

6 Fig 6: Use Case Diagram 12

7 Fig 7: Sequence Diagram 13

8 Fig 8: Collaboration Diagram 14

9 Fig 9: Activity Diagram 14

10 Fig 10: Our Project 16-18

**CHAPTER-1**

**INTRODUCTION**

Software engineering is the study of and practice of engineering to build, design, develop, maintain, and retire software. SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.

The following figure is a graphical representation of the various stages of a typical SDLC:



Fig. 1 Source: www.tutorialpoints.com

1. Stage 1: Planning and Requirement Analysis

Requirement analysis is the most important and fundamental stage in SDLC. It is performed by the senior members of the team with inputs from the customer, the sales department, market surveys and domain experts in the industry. This information is then used to plan the basic project approach and to conduct product feasibility study in the economical, operational and technical areas.

1. Stage 2: Defining Requirements

Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved from the customer or the market analysts. This is done through an SRS (Software Requirement Specification) document which consists of all the product requirements to be designed and developed during the project life cycle.

1. Stage 3: Designing the Product Architecture

SRS is the reference for product architects to come out with the best architecture for the product to be developed. Based on the requirements specified in SRS, usually more than one design approach for the product architecture is proposed and documented in a DDS - Design Document Specification.

1. Stage 4: Building or Developing the Product

In this stage of SDLC the actual development starts and the product is built. The programming code is generated as per DDS during this stage. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle.

1. Stage 5: Testing the Product

This stage is usually a subset of all the stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC. However, this stage refers to the testing only stage of the product where product defects are reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS.

1. Stage 6: Deployment in the Market and Maintenance

Once the product is tested and ready to be deployed it is released formally in the appropriate market. Sometimes product deployment happens in stages as per the business strategy of that organization. The product may first be released in a limited segment and tested in the real business environment (UAT- User acceptance testing).

The Rational Unified Process (RUP) is an iterative software development process framework created by the Rational Software Corporation, a division of IBM since 2003. RUP is not a single concrete prescriptive process, but rather an adaptable process framework, intended to be tailored by the development organizations and software project teams that will select the elements of the process that are appropriate for their needs. RUP is a specific implementation of the Unified Process.

The 6 Best Practices of RUP are:

1. Develop iteratively, with risk as the primary iteration driver
2. Manage requirements
3. Employ a component-based architecture
4. Model software visually
5. Continuously verify quality
6. Control changes

Four project life-cycle phases:

* Inception
* Elaboration
* Construction
* Transition

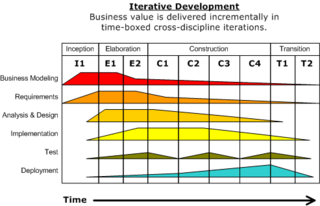


Fig. 2 Source: www.tutorialpoints.com

**CHAPTER-2**

**DEVELOPMENT METHODOLOGY**

* Waterfall Model

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.



Fig. 3 Source: www.tutorialpoints.com

* Iterative Model

Iterative process starts with a simple implementation of a subset of the software requirements and iteratively enhances the evolving versions until the full system is implemented. At each iteration, design modifications are made and new functional capabilities are added. The basic idea behind this method is to develop a system through repeated cycles (iterative) and in smaller portions at a time (incremental). This project uses this methodology.



Fig. 4 Source: www.tutorialpoints.com

**CHAPTER-3**

**ABOUT THE PROJECT**

A non-governmental organization (NGO) is any non-profit, voluntary citizens' group which is organized on a local, national or international level. Task-oriented and driven by people with a common interest, NGOs perform a variety of service and humanitarian functions, bring citizen concerns to Governments, advocate and monitor policies and encourage political particpation through provision of information. Some are organized around specific issues, such as human rights, environment or health. They provide analysis and expertise, serve as early warning mechanisms and help monitor and implement international agreements.

We have created a client-side NGO portal that creates a one stop destination to register for a NGO, donate and to view the workings and other aspects associated with a NGO. We have followed the RUP best practices and also taken in account the incremental model to achieve these tasks. During the making of the project, with the help of various diagrams we were able to structure our project in a clear and concise manner.

The following are the functional requirements on basis of which we defined our diagrams:

1. One of the pages will contain information about the organization, agenda and motive. It will be a simple static page. The second page is a registration form for applying to open a sub NGO under the organization. The third page leads to a donation portal which asks the user to enter his/her details and the specification of the object donated. The second and third page is connected to a backend.
2. The backend will consist of multiple tables. The first one will record the details entered to register for a new NGO. The second table will consist of information regarding the donor and the product donated.
3. The portal will also consist of a button on one of the pages that will enable the user to check the status of the registration of the NGO and also the status of the offered donation.

**CHAPTER-4**

**REQUIREMENT ANALYSIS**

* FUNCTIONAL REQUIREMENTS:

1) One of the pages will contain information about the organization, agenda and motive. It will be a simple static page. The second page is a registration form for applying to open a sub NGO under the organization with or without a partnership. The third page leads to a donation portal which asks the user to enter his/her details and the specification of the object donated. The second and third page is connected to a backend.

2) The backend will consist of multiple tables. The first one will record the details entered to register for a new NGO. The second table will only be used if the user opts for a partnership as well. The third table will consist of information regarding the donor and the product donated.

3)The portal will also consist of a button on one of the pages that will enable the user to check the status of the registration of the NGO and also the status of the offered donation.

* OTHER REQUIREMENTS:

1. Softwares: Xamppserver (front-end)

: MySQL (back-end)

1. Images and text to support the portal.
2. Languages used: HTML 5, CSS, PHP, SQL

**CHAPTER-5**

**FEASIBILITY STUDY**

* **Technical Feasibility:**According to the requirement analysis, as far as the technical requirements go i.e. MySQL and Xamppserver the project is technically feasible.
* **Social Feasibility:**There is a need for this product in the market and the business model is realistic as well.
* **Economic Feasibility:**According to the requirement the project is economically feasible as the cost of the project is less. Also, the deployment and testing phase will not incur any cost.
* The interface prototype was agreed upon and implemented.
* The project development and structure is adequate enough to support future developments as well.

**CHAPTER-6**

**ANALYSIS AND DESIGN**

UML (Unified Modeling Language) is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by the Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997. It was initially started to capture the behavior of complex software and non-software system and now it has become an OMG standard. This tutorial gives a complete understanding on UML.

Behavioral diagrams basically capture the dynamic aspect of a system. Dynamic aspect can be further described as the changing/moving parts of a system.

UML has the following five types of behavioral diagrams −

* Use case diagram
* Sequence diagram
* Collaboration diagram
* State chart diagram
* Activity diagram

1. Class Diagram

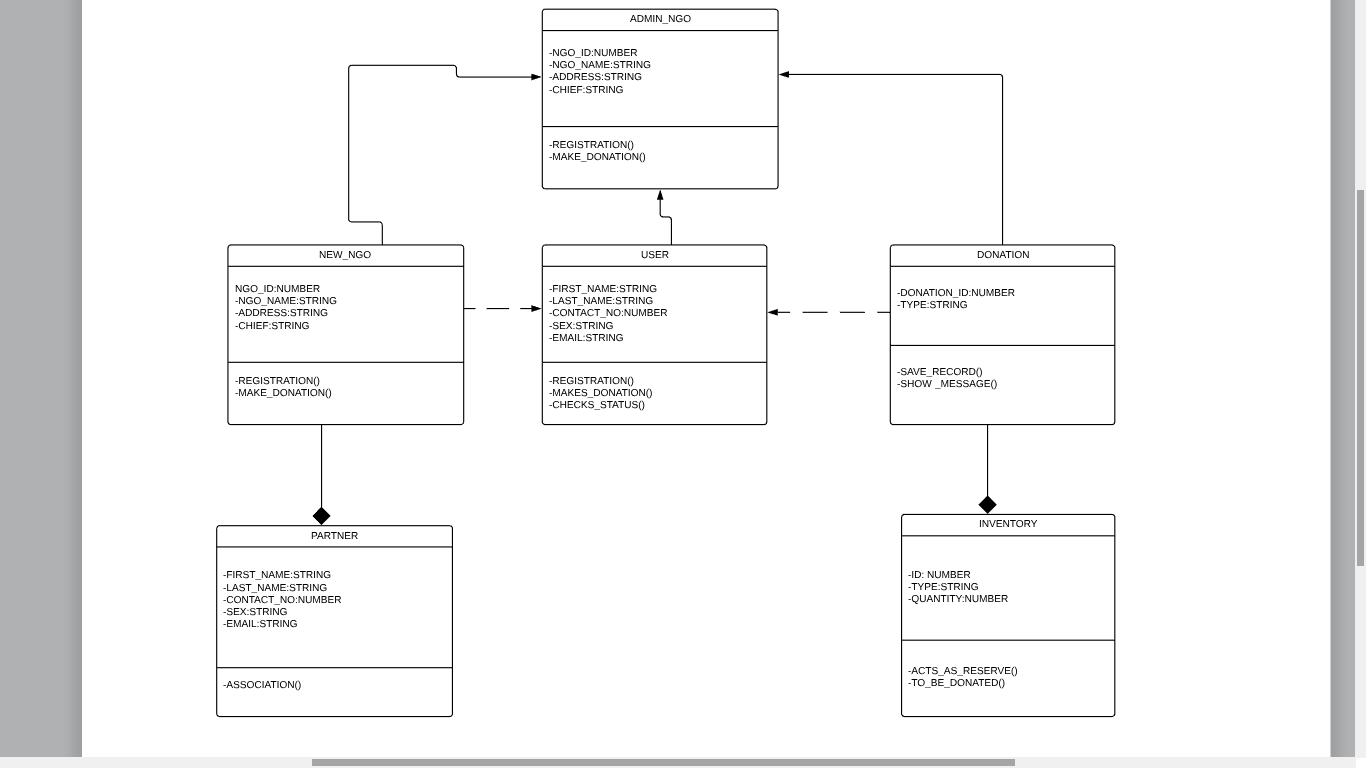


Fig. 5 Class Diagram

1. Use Case Diagram

Use case diagrams are a set of use cases, actors, and their relationships. They represent the use case view of a system. A use case represents a particular functionality of a system. Hence, use case diagram is used to describe the relationships among the functionalities and their internal/external controllers. These controllers are known as actors.

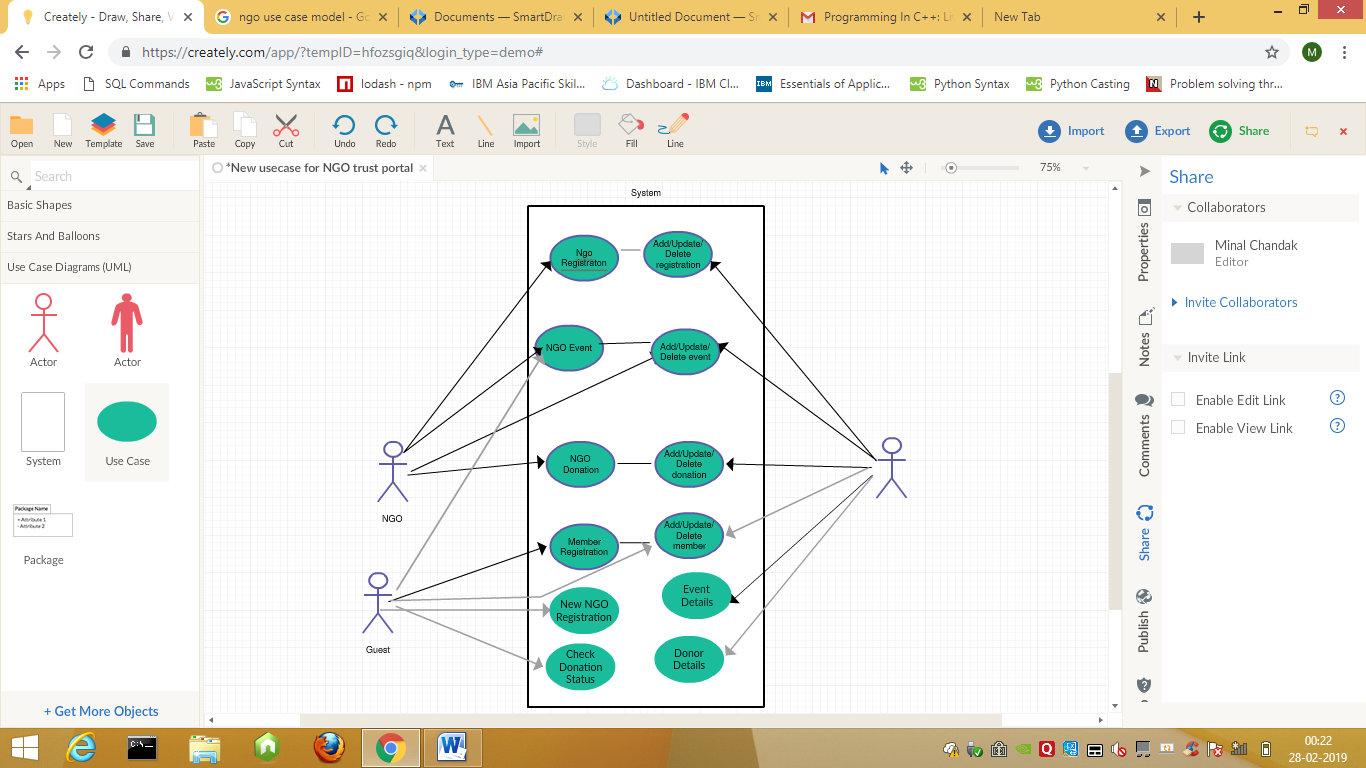


Fig. 6 Use Case Diagram

1. Sequence Diagram

A sequence diagram is an interaction diagram. From the name, it is clear that the diagram deals with some sequences, which are the sequence of messages flowing from one object to another. Interaction among the components of a system is very important from implementation and execution perspective. Sequence diagram is used to visualize the sequence of calls in a system to perform a specific functionality.



Fig. 7 Sequence Diagram

1. Collaboration Diagram

Collaboration diagram is another form of interaction diagram. It represents the structural organization of a system and the messages sent/received. Structural organization consists of objects and links. The purpose of collaboration diagram is similar to sequence diagram. However, the specific purpose of collaboration diagram is to visualize the organization of objects and their interaction.



Fig. 8 Collaboration Diagram

1. Activity Diagram

Activity diagram describes the flow of control in a system. It consists of activities and links. The flow can be sequential, concurrent, or branched. Activities are nothing but the functions of a system. Numbers of activity diagrams are prepared to capture the entire flow in a system. Activity diagrams are used to visualize the flow of controls in a system. This is prepared to have an idea of how the system will work when executed.



Fig. 9 Activity Diagram

**CHAPTER-7**

**DEPLOYMENT AND DEVELOPMENT**

<?php

$servername = "localhost";

$username = "root";

$password = "";

$dbname="ngo";

$firstname=filter\_input(INPUT\_POST,'firstname');

$lastname=filter\_input(INPUT\_POST,'lastname');

$usrtel=filter\_input(INPUT\_POST,'usrtel');

$usrmob=filter\_input(INPUT\_POST,'usrmob');

$email=filter\_input(INPUT\_POST,'email');

$city=filter\_input(INPUT\_POST,'city');

$code=filter\_input(INPUT\_POST,'code');

// Create connection

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

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

else

{

$var="insert into register (firstname,lastname,usrtel,usrmob,email,city,code)

values ( '$firstname','$lastname','$usrtel','$usrmob','$email','$city','$code')";

if($conn->query($var))

{

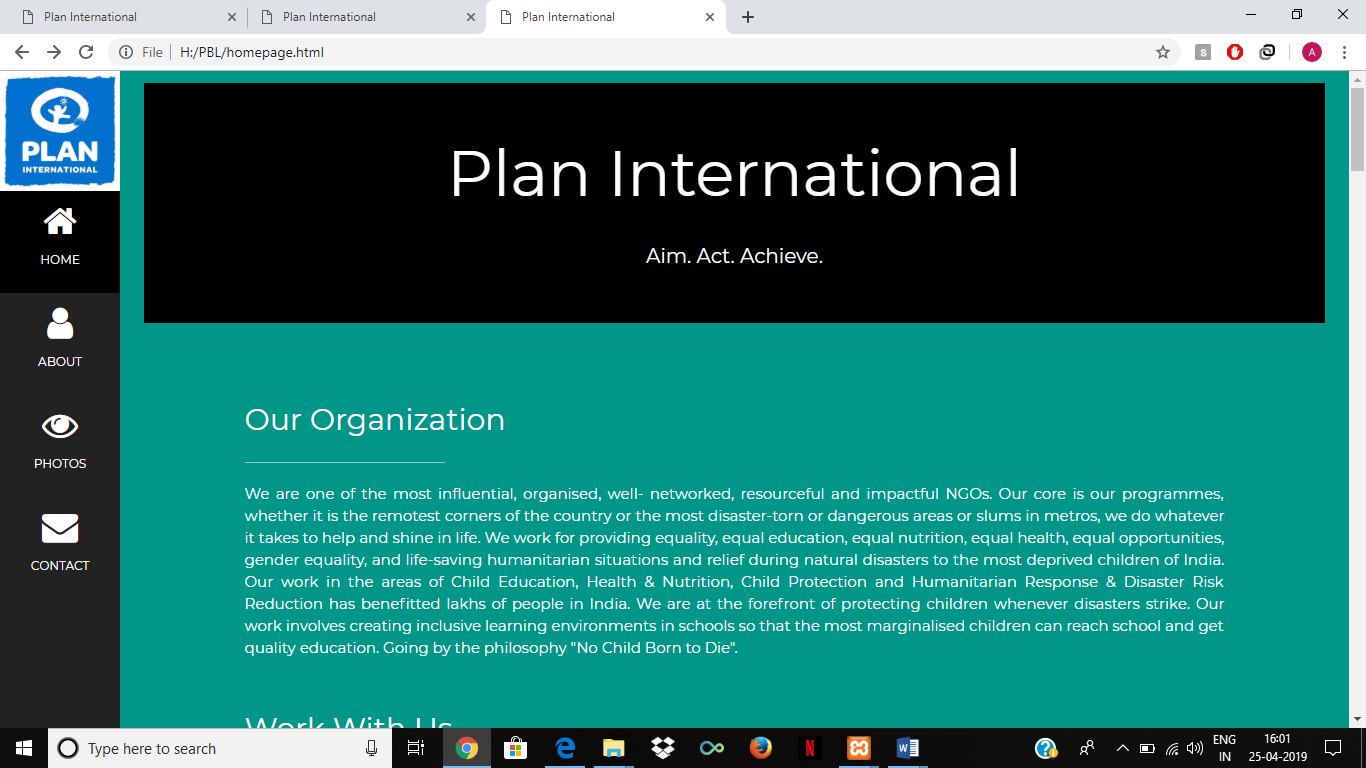
echo"<html><body><b> Record Entered</b></body></html>";

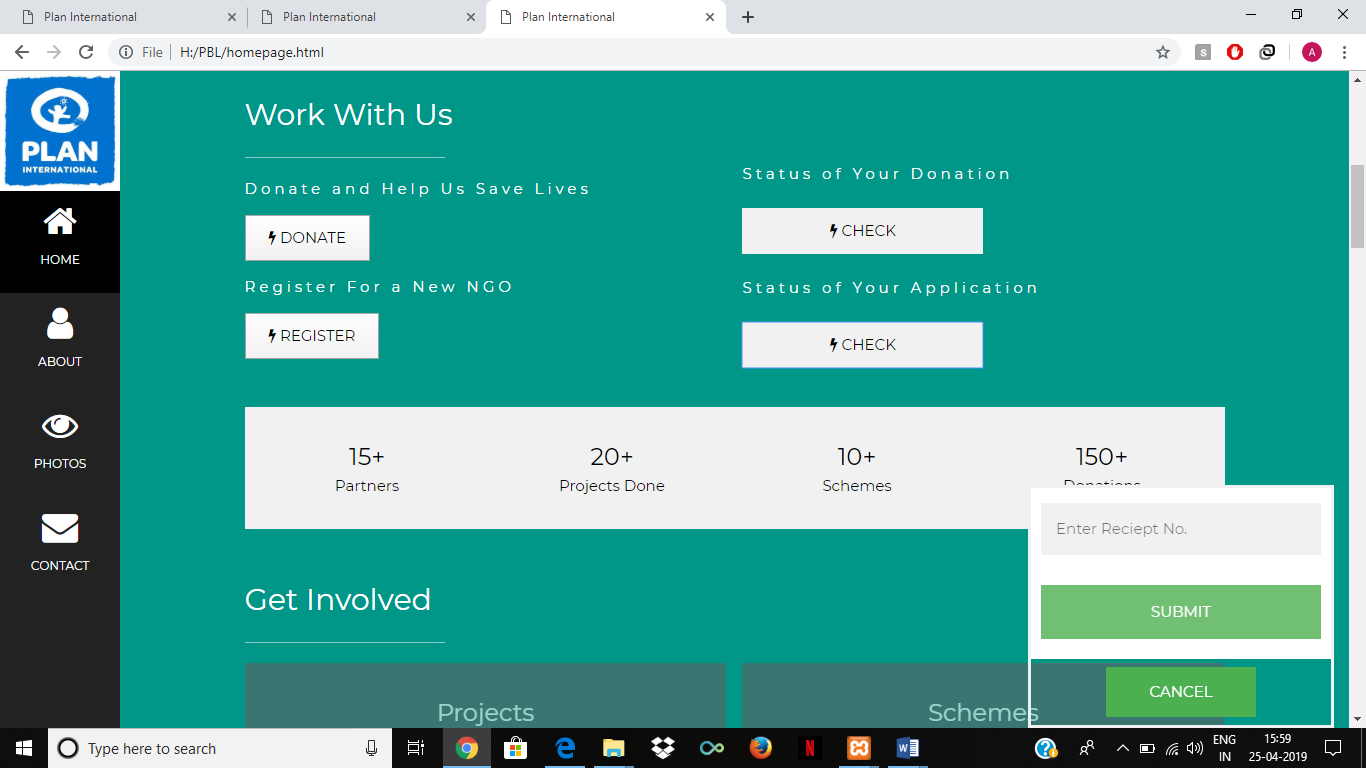
}

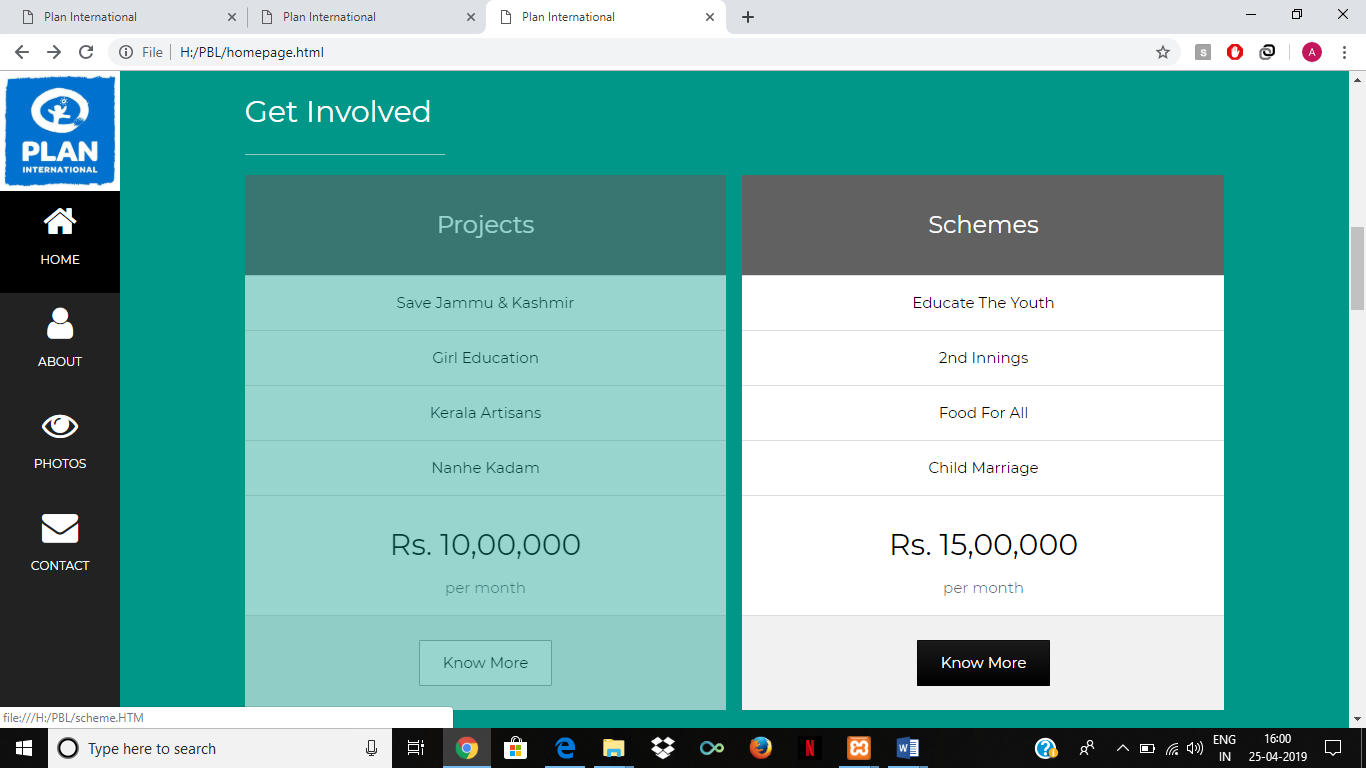
$conn->close();

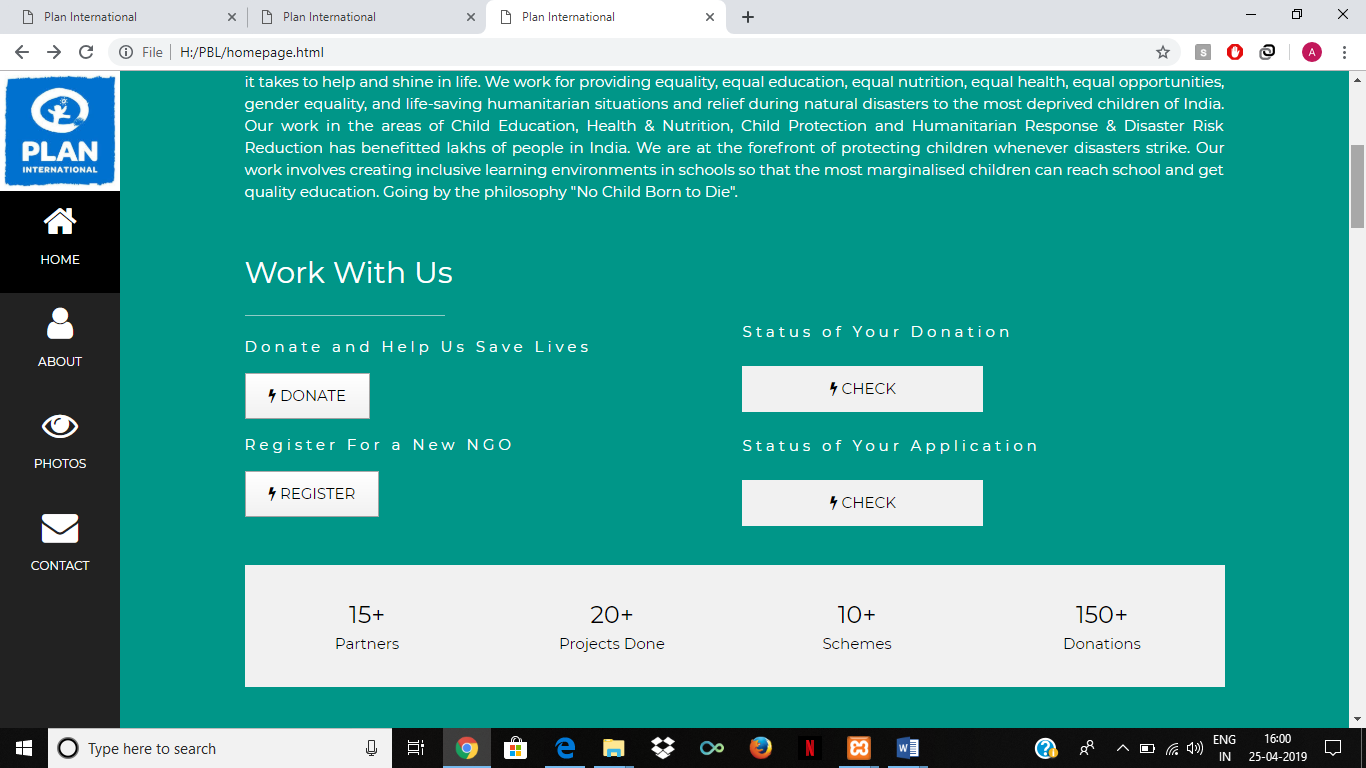
}

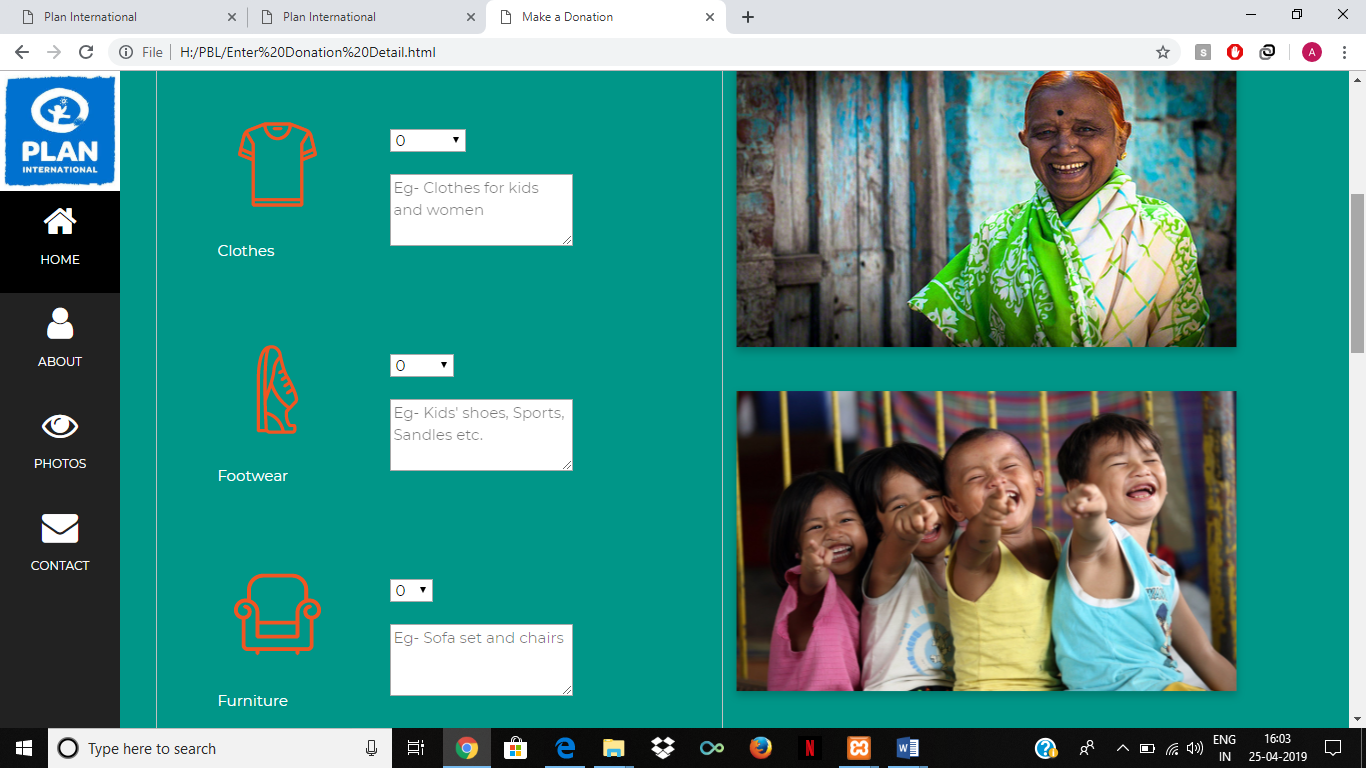
?>

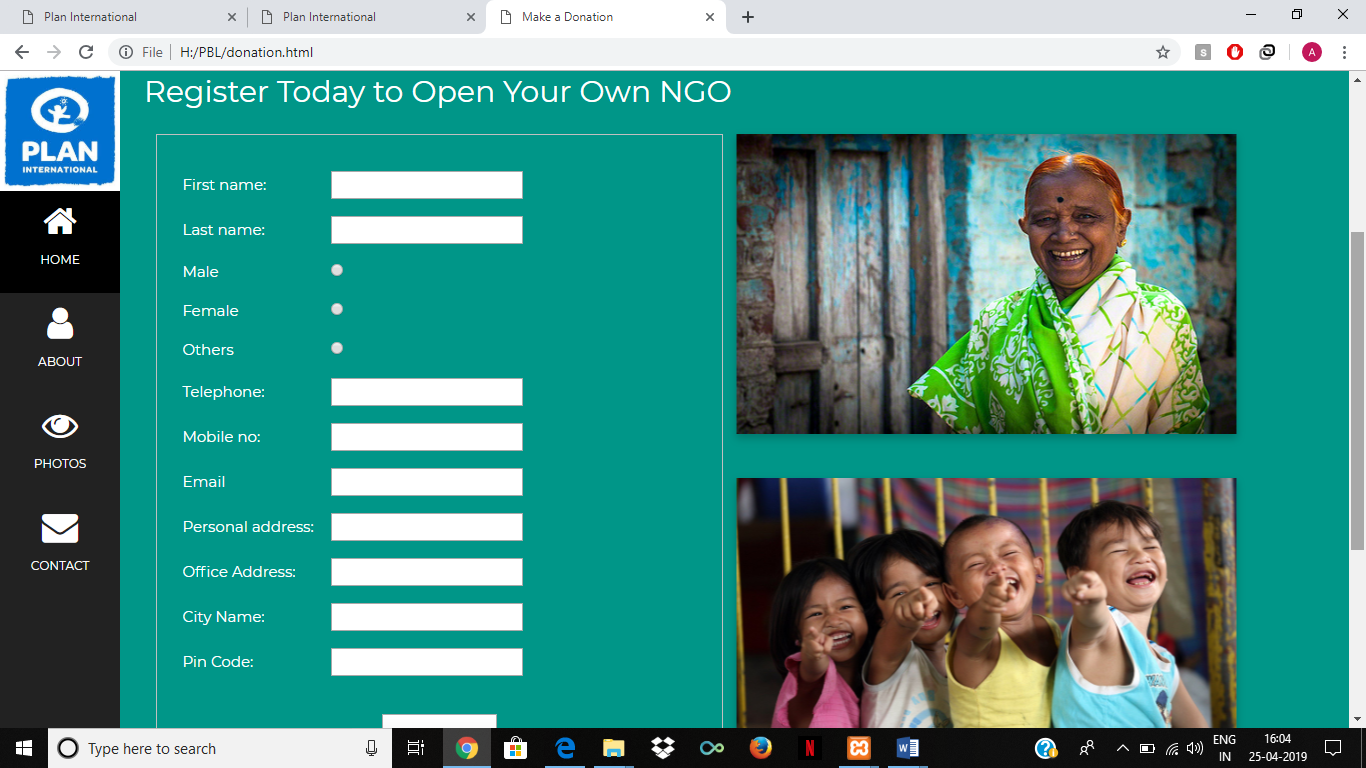












**REFERENCES**

<https://en.wikibooks.org/wiki/Introduction_to_Software_Engineering/Process/Life_Cycle>

<https://www.tutorialspoint.com/software_engineering/software_development_life_cycle.htm>

<https://www.esds.co.in/blog/introduction-to-software-development-life-cycle-sdlc-phases-models/#sthash.YO1HGMFx.dpbs>

<https://medium.com/omarelgabrys-blog/object-oriented-analysis-and-design-introduction-part-1-a93b0ca69d36>

**PLAGIARISM REPORT**