##### RURAL DEVELOPMENT

##### 15IT375L- MINOR PROJECT REPORT

###### ***Submitted by***

##### RISHABH GOEL (RA1511008010330)

**ACHINTYA PANDEY (RA1511008010334)**

**KARTIK BHOJAK (RA1511008010352)**

**ADITYA PATNAIK (RA1511008010354)**

**KUSHAGRA JAISWAL (RA1511008010400)**

***in partial fulfillment for the award of the degree***

***of***

##### BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY



SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR

APRIL 2018

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR

**BONAFIDE CERTIFICATE**

Certified that this project report **“RURAL DEVELOPMENT”** is the bonafide work of “**RISHABH GOEL, ACHINTYA PANDEY, KARTIK BHOJAK, ADITYA PATNAIK AND KUSHAGRA JAISWAL”** who carried out the project work under my supervision.

**SIGNATURE SIGNATURE**

**Mr. J. Prabakaran Dr. G. Vadivu**

**GUIDE**

**Assistant Professor (SG) HEAD OF THE DEPARTMENT**

**Information Technology**

INTERNAL EXAMINER

**DECLARATION**

We, Rishabh Goel(RA1511008010330), Achintya Pandey(RA1511008010334), Kartik Bhojak(RA1511008010352), Aditya Patnaik(RA1511008010354), Kushagra Jaiswal(RA1511008010400) studying in III year B.Tech Information Technology program at, SRM Institute of Science and Technology, Kattankulathur, Chennai, hereby declare that this project is an original work of us and we have not verbatim copied / duplicated any material from sources like internet or from print media, excepting some vital company information / statistics and data that is provided by the company itself.

Signature of the Student

Date: 13 April 2018

Place: Chennai

**ACKNOWLEDGEMENT**

The success and the final outcome of this project required guidance and assistance from different sources and we feel extremely fortunate to have got this all along the completion of our project. Whatever we have done is largely due to such guidance and assistance and we would not forget to thank them.

We express our sincere thanks to the Head of the Department, Department of Information Technology, Dr.G.Vadivu, for all the help and infrastructure provided to us to complete this project successfully and her valuable guidance.

We owe our profound gratitude to our project guide **Mr. J. Prabakaran**, who took keen interest in our project work and guided us all along, till the completion of our project work by providing all the necessary information for developing a good system.

We are thankful to and fortunate enough to get constant encouragement, support and guidance from all the Teaching staff of the Department of Information Technology which helped us in successfully completing our minor project work. Also, we would like to extend our sincere regards to all the non-teaching staff of the department of Information Technology for their timely support.

Rishabh Goel (RA1511008010330)

Achintya Pandey (RA1511008010334)

Kartik Bhojak (RA1511008010352)

Aditya Patnaik (RA1511008010354)

Kushagra Jaiswal (RA1511008010400)

**ABSTRACT**

The objective of this project is to provide a common platform for NGOs which would give them the power of being heard. This project is intended to create a greater awareness about different NGOs and their requests. To bring attention to environmental, financial, natural losses as a cross-cutting issue; to highlight the contribution of a specific NGO. Increase access to information about the requirements for other NGOs who are starting to venture the field. Promote the causes that are intended for public good.

**TABLE OF CONTENTS**

**CHAPTER NO. TITLE PAGENO.**

**ABSTRACT………………………………………………...v**

**LIST OF FIGURES……………………………………...viii**

**LIST OF ABBREVIATIONS…………………………….ix**

**1. OBJECTIVE……………………………………………………..x**

**2. PROBLEM STATEMENT……………………………………..xi**

**3. LITERATURE REVIEW………………………………………xii**

3.1 Apache Server…………………………………………..xii

3.2 Backend Frameworks…………………………………...xii

3.2.1 PHP …………………………………………xiii

3.2.2 MySQL……………………………………..xiv

3.3 Artificial Intelligence Services………………..xv

3.3.1 Engati-ChatBot………………………xv

3.3.2 Pubble-Community………………….xv

**4. REQUIREMENT ANALYSIS……………………………... xvii**

4.1 Software/Technology used at client end……...xvii

4.2 Software/Technology used at server end……..xvii

4.3 Miscellaneous Integrations…………………..xvii

4.4 Hardware requirements…………………………xviii

4.5 Functional requirements………………………...xviii

4.6 Non Functional requirements………………….xviii

**5. IMPLEMENTATION DETAILS……………………..xix**

5.1 Login ………………………………….xix

5.2 NGO Panel…………………………….xix

5.3 Donate page……………………………xx

5.4 NGO Profile……………………………xx

**6. FUTURE ADVANCEMENTS…………………………xxi**

6.1 Firebase………………………………………..xxi

6.2 AWS………………………………………..xxviii

6.3 Two-factor Authentication………………….....xv

**7. CONCLUSION………………………………………xxvii**

**8. APPENDICES………………………………………...xviii**

**9. REFERENCES……………………………………….. xliv**

**LIST OF FIGURES**

Figure 1 Use Case Diagram ……………………………………….18

Figure 2 Sequence Diagram ……………………………………….18

Figure 3 Dataflow Diagram………………………………………..19

Figure 4 Home Page………………………………………………..20

Figure 5 Gallery……………………………………………….........21

Figure 6 Registration Form…………………………………………21

Figure 7 Admin Login Database……………………………………22

Figure 8 NGO Registration Database………………………………25

Figure 9 NGO Requirements………………………………………..25

Figure 10 Admin Panel………………………………………………26

Figure 11 Donate Page……………………………………………….30

Figure 12 Forum……………………………………………………...30

Figure 13 NGO Requirement Analysis………………………………31

Figure 14 About us…………………………………………………...32

Figure 15 Chatbot…………………………………………………….43

Figure 16 SMTP Request……………………………………………..44

**LIST OF ABBREVIATIONS**

Html Hypertext Markup Language

Css Cascading Style Sheet

Php Hypertext Preprocessor (Personal Home Page)

Sql Structured Query Language

NGO Non Government Organisation

**OBJECTIVE**

The objective of this project is to provide a common platform for NGOs which would give them the power of being heard. This project is intended to create a greater awareness about different NGOs and their requests. To bring attention to environmental, financial, natural losses as a cross-cutting issue; to highlight the contribution of a specific NGO. Increase access to information about the requirements for other NGOs who are starting to venture the field. Promote the causes that are intended for public good. We are here with an interface which would serve as a membrane between those who are willing to get the enlightening feeling of gratitude by helping an individual, or a group, or a community, or people of an entire area and the people who are in need. We are stressing more on the idea of richer class contributing their helping hand into this project rather than talking about the issues that the poor people face and how they can be benefited from this project is because there are uncountable problems to be seeked into and solved but there are very few people who come out of their luxury shells to help.

**PROBLEM STATEMENT**

To develop a common platform for NGOs .To create greater awareness about NGO requests and requirements. Providing a single go to for a user to turn into a donator. Providing a differentiated NGO list keeping in mind the focus that a particular NGO has in terms of area being Rural or Urban. To support Global issues by taking insights from people visiting the platform. To answer requirement related queries of the user using A.I. To provide an abstract messenger to the user to improve Web User Satisfaction.

**LITERATURE REVIEW**

APACHE SERVICE:

Modern web servers such as apache are capable of operating in both multi-threaded and multi-process modes. In multi-threaded mode, Apache is structured as a pool of worker threads to handle HTTP requests. A worker thread processes the request until it completes and then accepts a new request. In the thread pool model, the threads are pre-created at the start up. Apache is the most widely used web server software. Developed and maintained by Apache Software Foundation, Apache is an open source software available for free. It runs on 67% of all webservers in the world. It is fast, reliable, and secure. It can be highly customized to meet the needs of many different environments by using extensions and modules. Most WordPress hosting providers use Apache as their web server software. However, WordPress can run on other web server software as well. So basically a web server is the software that receives your request to access a web page. It runs a few security checks on your HTTP request and takes you to the web page. Depending on the page you have requested, the page may ask the server to run a few extra modules while generating the document to serve you. It then serves you the document you requested. Pretty awesome isn’t it.

BACKEND FRAMEWORKS:

**Php**

The PHP Hypertext Preprocessor (PHP) is a programming language that allows web developers to create dynamic content that interacts with databases. PHP is basically used for developing web based software applications. This tutorial helps you to build your base with PHP. PHP started out as a small open source project that evolved as more and more people found out how useful it was. Rasmus Lerdorf unleashed the first version of PHP way back in 1994.

PHP is a server side, cross platform, html embedded scripting language. The engine that runs php is the middleware that generates dynamic contents.

PHP’s modular design provides modules for, Database connectivity for popular database such as Oracle, MS-SQL server. Here the focus is more on PHP’s interaction with MySQL database and Apache web server.

PHP acts as a module in Apache server itself and shares the same process address space of the webserver. Hence doesn’t have any inter-process communication overhead with the server. Disadvantage of this architecture is in the situations when the web server is the bottleneck there is no chance to offload PHP middle tier into a separate machine. PHP performs system functions, i.e. from files on a system it can create, open, read, write, and close them.

PHP can handle forms, i.e. gather data from files, save data to a file, through email you can send data, return data to the user.

You add, delete, modify elements within your database through PHP.

Access cookies variables and set cookies.

Using PHP, you can restrict users to access some pages of your website.

It can encrypt data.

In this step first we download the phpmailer library and put all the files in phpmail directory and then insert the main PHPMailerAutoload.php in our document to send mails and that's all now we can send mail and from localhost also but with using internet connection.

**MySQL**

MySQL is also a multi-threaded server where a thread cache is used instead of thread pool model. Thread in a thread cache are managed in dynamic fashion and they are not pre-created at start up. When the number of threads needed to serve the requests exceeds the thread cache size new threads will be created. The most comprehensive set of advanced features, management tools and technical support to achieve the highest levels of MySQL scalability, security, reliability, and uptime. MySQL Cluster enables users to meet the database challenges of next generation web, cloud, and communications services with uncompromising scalability, uptime and agility. MySQL is the most popular Open Source Relational SQL Database Management System. MySQL is one of the best RDBMS being used for developing various web-based software applications. MySQL is developed, marketed and supported by MySQL AB, which is a Swedish company. WordPress uses the PHP programming language to store and retrieve data from the MySQL database. To retrieve data from the database, WordPress runs SQL queries to dynamically generate content. SQL stands for Structured Query Language and is the programming language typically used to query databases.

For users that are not comfortable writing their own PHP and SQL scripts, most web hosting providers offer easy to use web applications to manage databases. One such web application is phpMyAdmin which allows users to manage their database using a web based graphical interface. You can manipulate your tables visually while phpMyAdmin runs the SQL queries for you.

ARTIFICIAL INTELLIGENCE SERVICES:

**Engati-ChatBot**

A **chatbot** is a computer program which conducts a conversation via auditory or textual methods. Such programs are often designed to convincingly simulate how a human would behave as a conversational partner, thereby passing the Turing test. Engati is one of the best free chatbot platform to build bots in minutes without programming. Build once and publish across 8 major platforms - FB messenger, kik, telegram, line, viber, skype, slack and website. Supports intelligent paths, train, copy, analytics and private labeling.

Using Engati you can leverage the power of machine learning, NLP/NLU to design your very own chatbot.

We have integrated the chatbot to answer user queries related to the NGOs and their requirements.

**Pubble-Community**

We have integrated a user community to support discussion on various topics in situ with global needs. For a while now, we've been fascinated with how people communicate on websites and how the experience could be improved. Ever wondered why people are still using the same tools (email, contact forms) that they used 20 years ago? Or why existing tools (live chat, helpdesks) make it hard to bring the right people into the conversation at the right time? Or ever thought wouldn't it be cool if messaging a business via their website was as easy as it is to send a WhatsApp or iMessage? We did and the more we dug into the issue we realised that awesome customer support is not just about what happens on the site. Just as important is what happens in the background to deliver the perfect response to the customer. This is why we are working on Pubble. Pubble is evolving the way that people communicate through websites.

**REQUIREMENT ANALYSIS**

Software/Technology used at client end:

* Html
* Css
* Animate.css
* Bootstrap
* LightBox Model
* JavaScript
* Jquery
* Wow.js

Software/Technology used at server end:

* Server used Xampp
* Php
* Php mailer implemented on Gmail Server
* Database- Sql

Miscellaneous Integrations

* Google Maps
* Google Charts
* Engati Chat Bot
* Pubble-Forum

Hardware requirements :

* Operating system : Windows XP or above
* Ram : 2GB or above .
* Processor : Intel Pentium p4 or above.
* Web server : Xampp or any other web server.
* Web Browser : Google Chrome /Mozilla Firefox / Microsoft edge
* Text editor : Visual Studio Code / Adobe Dreamweaver / Sublime Text

Functional requirements :

* NGOs with requirements, needs
* Login and registration.
* Languages required : HTML/CSS/JavaScript/PHP/Bootstrap/Jquery.
* HTML, CSS, Bootstrap and Jquery were required to make the front end of the website.
* PHP and MySqli, used to manage the database of the website.

Non Functional requirements :

* Requirement is fed by the NGO.
* Requirements are flashed on the Donate Page.
* User may provide a monetary help or can contact the needy NGO to meet other demands related to food, clothing etc.

**IMPLEMENTATION DETAILS**

Signup

NGO is made to register using the sign-up page that is form based.

The NGO-signup data is fed to the registration table**.** On the Admin Panel, the user is approved using the ngo-id column in registration table. If the user is disapproved, the entry from the registration table is deleted.

An auto generated reply mail is sent to the user, once approved, otherwise the user is notified with a disapproval mail. An e-mail is sent to the team.

Login

Function-check (email, pass, status) from registration table

When the NGO logs in, the email-id is being looked up in the registration table and the status column is checked for approval. If the status is set to approved, only then the password is being further matched with the password mentioned in the table.

NGO Panel

Function-set requirements (radio button-food, monetary, clothes, others)

Once the NGO has successfully logged in, they can see the NGO Panel on the screen. On clicking the set requirements tab, the requirements page is displayed with different categories to request on.

Once the NGO is done with the requirement data entry, they can click on submit to log the entries on the NGO-requirements table.

Donate page

Function–Join (registration table data, ngo-requirement table data)

On successful submission of NGO requirement in the set requirement tab on the NGO panel, a Join operation is used to fetch the data from two tables (registration table and ngo-requirement table). The requirement is now displayed on the donate page for the viewers/users.

On clicking the donate button for a specific requirement, a form is displayed with a QR-code on it (could be scanned if the help to be made is monetary) or a link to NGO-information is displayed on the form to make contact with the NGO to help in other ways.

NGO Profile

Function-Statistics for NGO

On the NGO Panel, there are profile and requirement analysis tabs other than the set requirements tab which provides a deep insight about the NGO information and its requests made in different categories.

The Requirement Analysis Tab displays a pictorial information of NGO-requests in different categories in the form of a Pie Chart (Google Chart).

**FUTURE ADVANCEMENTS**

We are planning to replace the database with **Firebase** and **AWS** and bringing **two-factor authentication.**

**Firebase**

Firebase is a mobile and web app development platform that provides developers with a plethora of tools and services to help them develop high-quality apps, grow their user base, and earn more profit.

Back in 2011, before Firebase was Firebase, it was a startup called Envolve. As Envolve, it provided developers with an API that enabled the integration of online chat functionality into their website.

What’s interesting is that people used Envolve to pass application data that was more than just chat messages. Developers were using Envolve to sync application data such as a game state in real time across their users.

This led the founders of Envolve, James Tamplin and Andrew Lee, to separate the chat system and the real-time architecture. In April 2012, Firebase was created as a separate company that provided Backend-as-a-Service with real-time functionality.

After it was acquired by Google in 2014, Firebase rapidly evolved into the multifunctional behemoth of a mobile and web platform that it is today.

Firebase Services

* Develop & test your app
* Realtime Database
* Auth
* Test Lab
* Crashlytics
* Cloud Functions
* Firestore
* Cloud Storage
* Performance Monitoring
* Crash Reporting
* Hosting
* Grow & Engage your audience
* Firebase Analytics
* Invites
* Cloud Messaging
* Predictions
* AdMob
* Dynamic Links
* Adwords
* Remote Config
* App Indexing

Realtime Database

The Firebase Realtime Database is a cloud-hosted NoSQL database that lets you store and sync between your users in realtime.

The Realtime Database is really just one big JSON object that the developers can manage in realtime.

Realtime Database => A Tree of Values

With just a single API, the Firebase database provides your app with both the current value of the data and any updates to that data.

Realtime syncing makes it easy for your users to access their data from any device, be it web or mobile. Realtime Database also helps your users collaborate with one another.

Another amazing benefit of Realtime Database is that it ships with mobile and web SDKs, allowing you to build your apps without the need for servers.

When your users go offline, the Realtime Database SDKs use local cache on the device to serve and store changes. When the device comes online, the local data is automatically synchronized.

The Realtime Database can also integrate with Firebase Authentication to provide a simple and intuitive authentication process.

**AWS**

Amazon Web Services (AWS) is a comprehensive, evolving cloud computing platform provided by Amazon. It provides a mix of infrastructure as a service (IaaS), platform as a service (PaaS) and packaged software as a service (SaaS) offerings. AWS launched in 2006 from the internal infrastructure that Amazon.com built to handle its online retail operations. AWS was one of the first companies to introduce a pay-as-you-go cloud computing model that scales to provide users with compute, storage or throughput as needed. Amazon Web Services is the leading cloud platform provider, offering over 250 applications and services -- including some from third parties -- to deploy, maintain, monitor and run applications in the cloud. AWS offers a variety of compute instances that allow companies to quickly scale up or scale down based on need. Two of the most commonly used AWS computing services are Amazon Elastic Compute Cloud (EC2) and Amazon Elastic Load Balancing. Amazon EC2 targets system admins and developers. These IT pros use EC2 to quickly acquire and "boot" computing instances in the cloud. Like many cloud services, EC2 pricing depends on usage. EC2 has a free service that includes up to 750 hours of Linux or Windows EC2 service hours per month for one year. Amazon offers three other pricing tiers: On-Demand, Reserve Instance and Spot Instance. On-Demand prices range from $.013 to $4.60 an hour depending on size, complexity and storage requirements, as well as location. AWS has relational and NoSQL databases, as well as in-memory caching and petabyte-scale data warehousing. Users can operate their own databases on AWS using EC2 and EBS. Two popular AWS database services are the Relational Database Service (RDS) and Redshift. Amazon RDS creates, operates and scales MySQL, Oracle, SQLServer or PostgreSQL servers on AWS. Additionally, RDS handles software installation, backups and patches, as well as routine administrative tasks. RDS pricing is based on instance hours and the amount of storage used. Amazon Redshift is a data warehouse service that works with many common business intelligence tools. It also offers columnar storage technology for databases that store data in columns rather than rows. Pricing is based on the number of instance hours starting at $ 0.25 per hour.

**Two-factor authentication**

Two-factor authentication (2FA), often referred to as two-step verification, is a security process in which the user provides two authentication factors to verify they are who they say they are. 2FA can be contrasted with single-factor authentication (SFA), a security process in which the user provides only one factor -- typically a password. Two-factor authentication provides an additional layer of security and makes it harder for attackers to gain access to a person's devices and online accounts, because knowing the victim's password alone is not enough to pass the authentication check. Two-factor authentication has long been used to control access to sensitive systems and data, and online services are increasingly introducing 2FA to prevent their users' data from being accessed by hackers who have stolen a password database or used phishing campaigns to obtain users' passwords.

The ways in which someone can be authenticated usually fall into three categories known as the factors of authentication, which include:

1. Knowledge factors -- something the user knows, such as a password, PIN or shared secret.

2. Possession factors -- something the user has, such as an ID card, security token or a smartphone.

3. Inherence factors, more commonly called biometrics -- something the user is. These may be personal attributes mapped from physical characteristics, such as fingerprints, face and voice. It also includes behavioral biometrics, such as keystroke dynamics, gait or speech patterns.

Systems with more demanding requirements for security may use location and time as fourth and fifth factors. For example, users may be required to authenticate from specific locations, or during specific time windows.

Multifactor authentication involves two or more independent credentials for more secure transactions.

Smartphones offer a variety of possibilities for 2FA, allowing companies to use what works best for them. Some devices have screens capable of recognizing fingerprints; a built-in camera can be used for facial recognition or iris scanning and the microphone can be used for voice recognition. Smartphones equipped with GPS can verify location as an additional factor. Voice or Short Message Service (SMS) may also be used as a channel for out-of-band authentication.

Apple iOS, Google Android, Windows 10 and BlackBerry OS 10 all have apps which support 2FA, allowing the phone itself to serve as the physical device to satisfy the possession factor.

Authenticator apps replace the need to obtain a verification code via text, voice call or email. For example, to access a website or web-based service that supports Google Authenticator, the user types in their username and password -- a knowledge factor. The user is then prompted to enter a six-digit number. Instead of having to wait a few seconds to receive a text message, Authenticator generates the number for them. These numbers change every 30 seconds and are different for every login. By entering the correct number, the user completes the user-verification process and proves possession of the correct device -- an ownership factor.

**CONCLUSION**

The target of the project is to provide a common platform to independent NGOs and to Organisations who are working for public good. The Webx System is Powered by a Fully transparent Model and Gives a clean sweep to Corruption after all “Transparency is Trust”. Here all NGOs work together under one roof. There is no requirement of building a need, since all NGOs are Self-sufficient and have their own incoming demands. From a donor’s perspective, all resources will be found at one place. The Product is Fast and Agile.

“A kind gesture can reach a wound that only compassion can heal”

**APPENDICES**

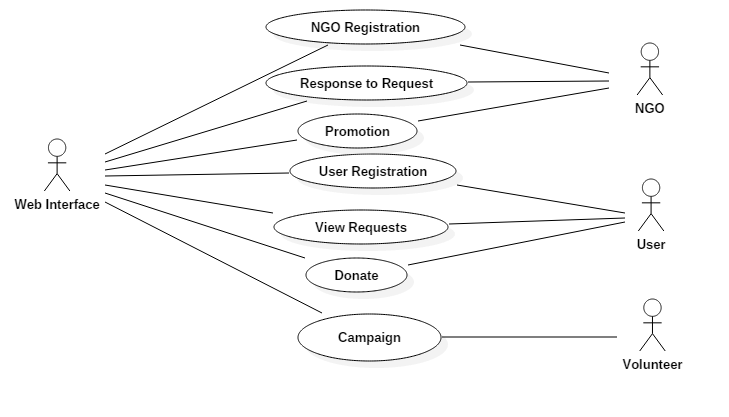
****

Figure 1

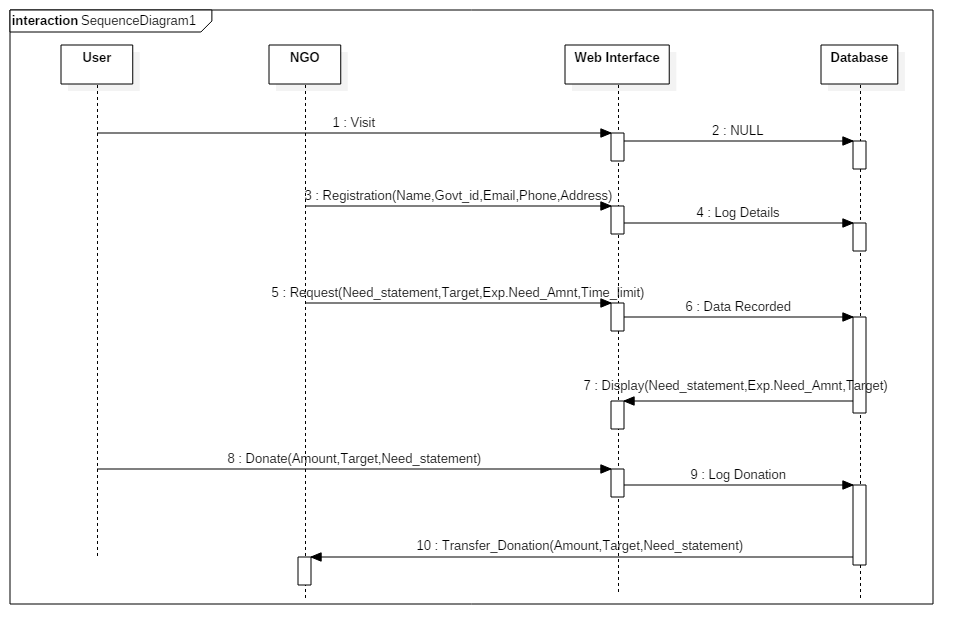
****

Figure 2

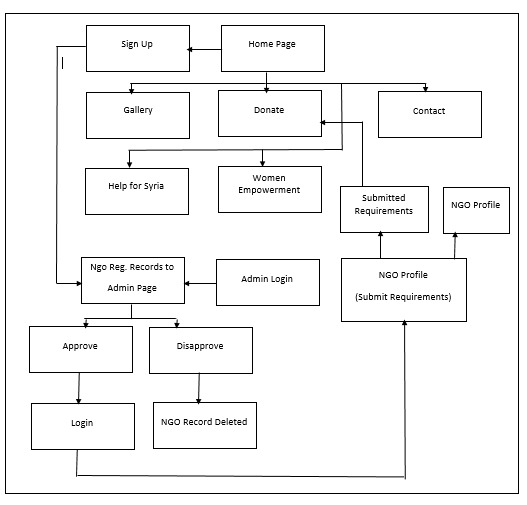


Figure 3

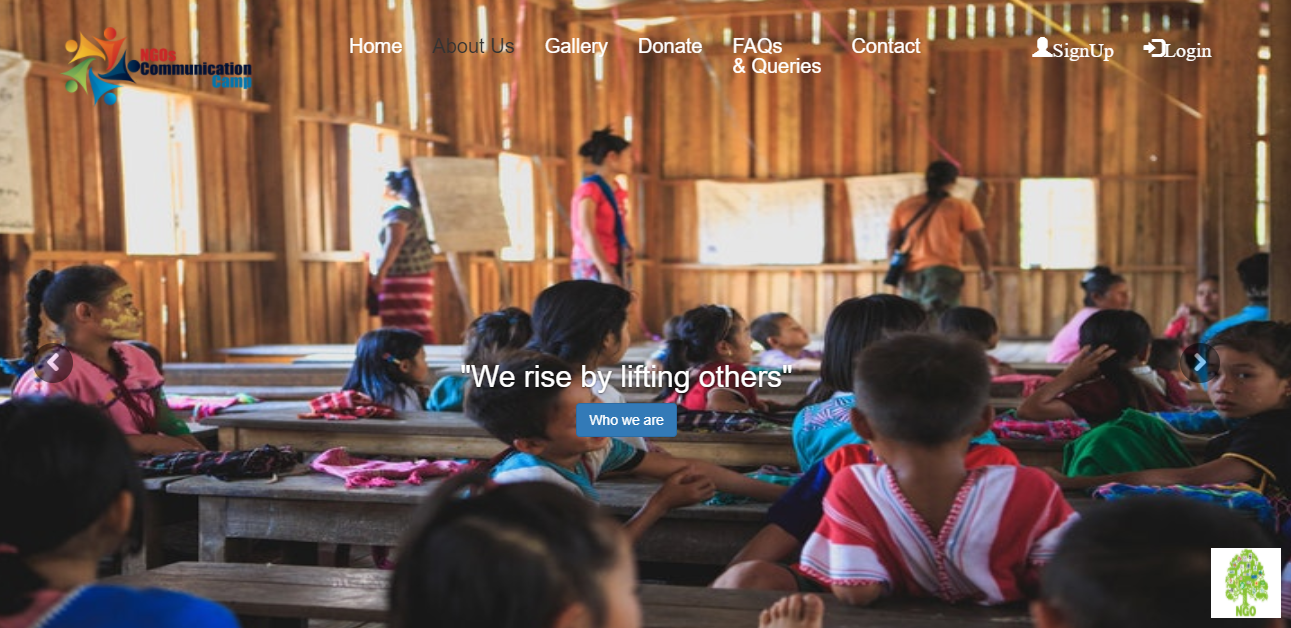
****

Figure 4

Figure 5

**Registration Module**

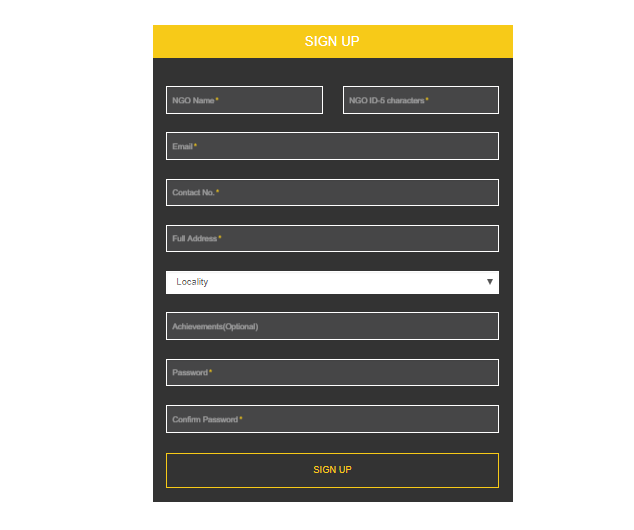


Figure 6

<?php

session\_start();

$\_SESSION['message']='';

$mysqli=mysqli\_connect('localhost','root','','miniproject');

if(!$mysqli)

{

die('couldnot connect'.mysqli\_connect\_error());

}

if($\_SERVER['REQUEST\_METHOD']=='POST')

{

if($\_POST['password']==$\_POST['confirmpassword'])

{

if(isset($\_POST['submit']))

{

$ngo\_name=$mysqli->real\_escape\_string($\_POST['ngo\_name']);

$ngoid=$mysqli->real\_escape\_string($\_POST['ngoid']);

$email=$mysqli->real\_escape\_string($\_POST['email']);

$phone=$mysqli->real\_escape\_string($\_POST['phone']);

$address=$mysqli->real\_escape\_string($\_POST['address']);

$locality=$mysqli->real\_escape\_string($\_POST['locality']);

$achievements=$mysqli->real\_escape\_string($\_POST['achievements']);

$password=md5($\_POST['password']);

$confirmpassword=md5($\_POST['confirmpassword']);

echo $ngoid;

$sql=mysqli\_query($mysqli,"INSERT INTO registration(NGO\_name,ngo\_id,Email,Phone,address,Area,achievements,Password,Confirm\_Password,status)VALUES('$ngo\_name','$ngoid','$email','$phone','$address','$locality','$achievements','$password','$confirmpassword','pending')");

if ($sql)

{

$\_SESSION['message']="Registration Successful";

header("refresh:3; url=/mini-project/login.php");

}

else

{

$\_SESSION['message']="Registration Unsuccessful";

echo "Error: " . $sql . "<br>" . mysqli\_error($mysqli);

}

}

}

else

{

$\_SESSION['message']="Passwords do not match!";

}

}

session\_destroy();

mysqli\_close($mysqli);

?>

<?php

if(isset($\_POST['submit']))

{

ini\_set('display\_errors', 1);

require 'autoload.php';

require 'class.phpmailer.php';

//require 'class.smtp.php';

$mail = new PHPMailer;

$mail->SMTPDebug = 4;

$mail->IsSMTP();

$mail->Host="smtp.gmail.com";

$mail->SMTPAuth=true;

$mail->Username = "rishabhgoel9797@gmail.com";

$mail->Password = "Computer@123";

//$mail->SetLanguage("en", 'language');

$mail->Mailer = "smtp";

$mail->SMTPSecure = 'tls';

$mail->Port = 587;

//$mail->setFrom("rishabhgoel9797@gmail.com");

$mail->setFrom('rishabhgoel9797@gmail.com','WebX NGO');

$mail->addAddress($\_POST['email'],$\_POST['ngo\_name']);

$mail->AddCC('kushagra.jaiswal18@gmail.com');

$mail->isHTML(true);

$mail->Subject = 'WebX NGO signup verification mail';

$mail->Body = 'Thank You for registering with WebX NGO. You wil be notified once you will be verified.';

$mail->AltBody = 'This is the plain text version of the email content.';

/\*try{

$mail->Send();

echo "Thanks. Bug report successfully sent.

We will get in touch if we have any more questions!";

} catch(Exception $e){

//Something went bad

echo "Mailer Error: - " . $mail->ErrorInfo;

}\*/

if(!$mail->Send()) {

echo "Mailer Error: " . $mail->ErrorInfo;

}

else {

echo "Message has been sent successfully";

}

}

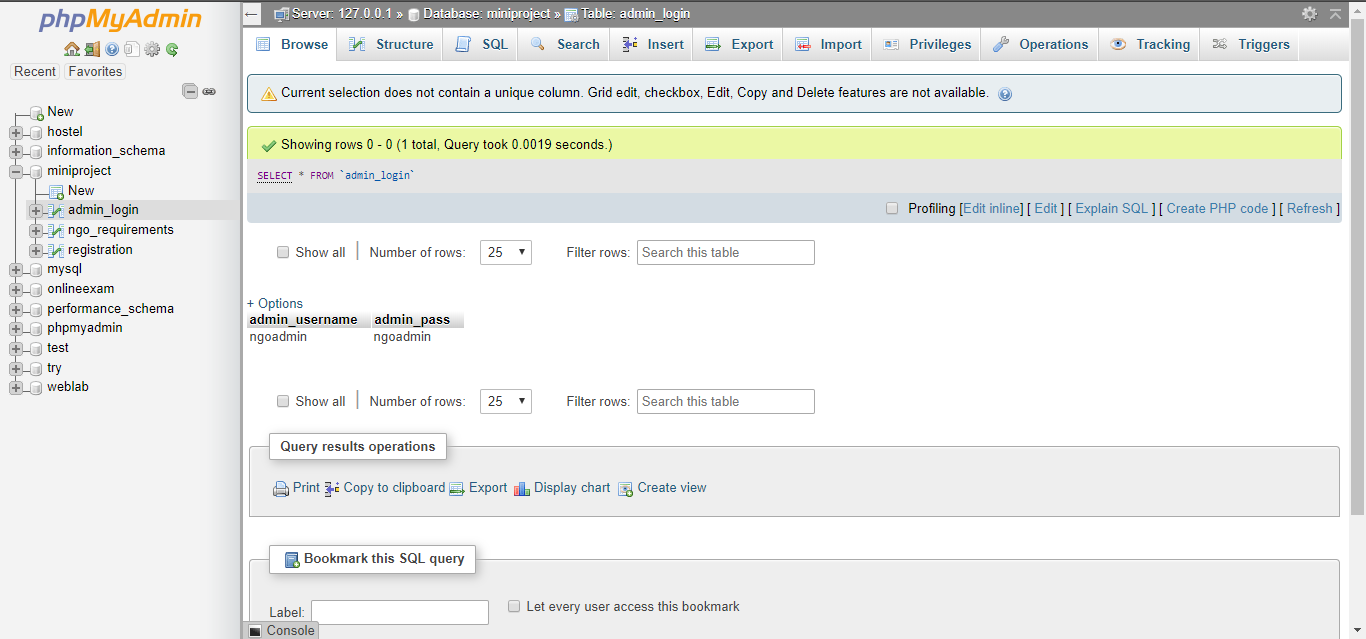
else

{

echo "not send";

}

?>

****Figure 7

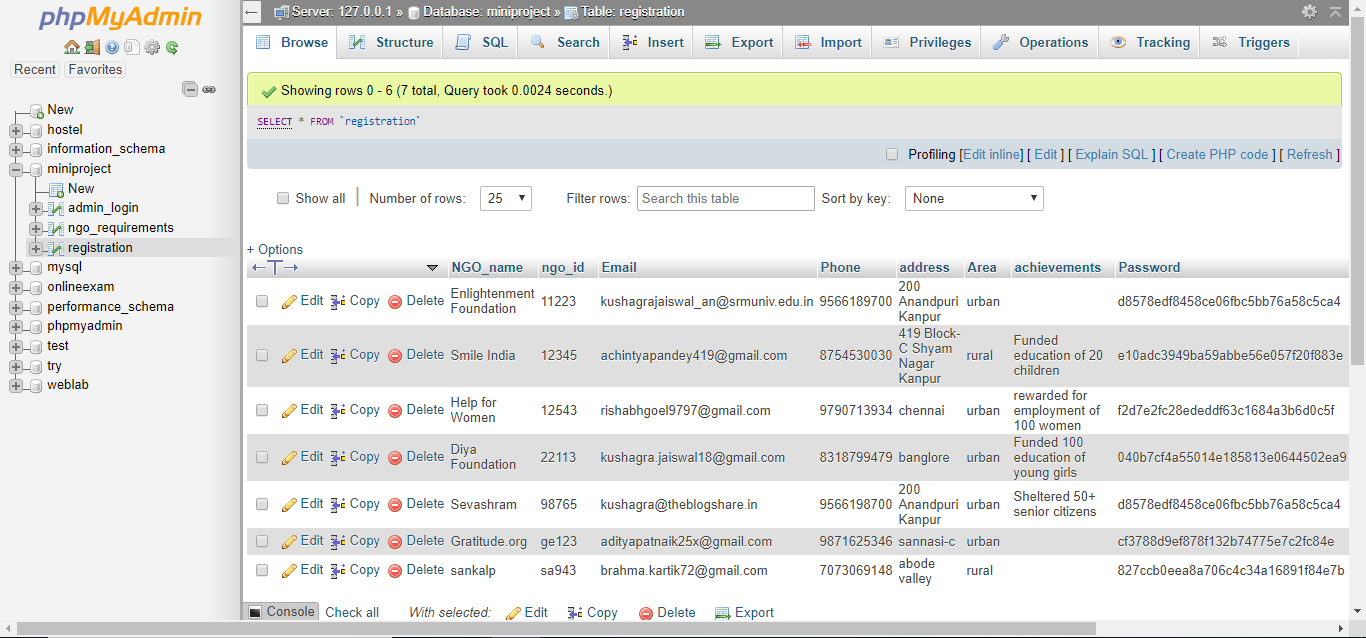
****

Figure 8

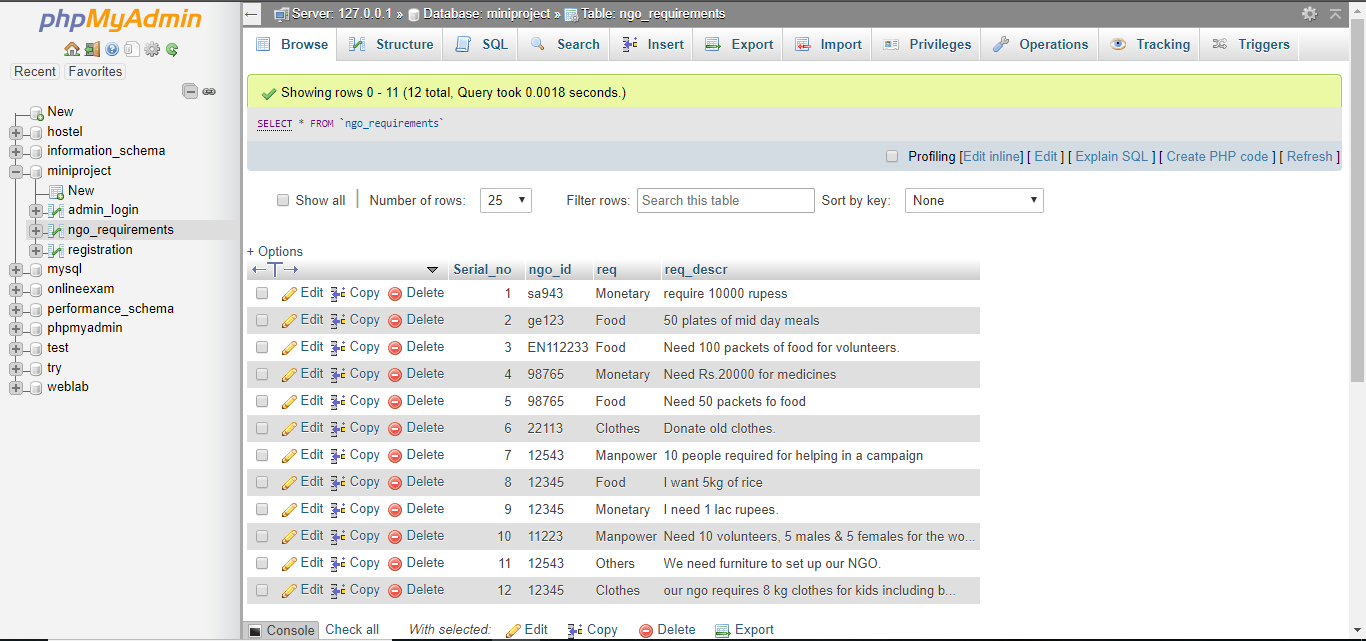
****

Figure 9

**Database Module**

<?php

session\_start();

session\_destroy();

?>

<!DOCTYPE html>

<html>

<head>

  <title>donate|NGO Panel</title>

  <link rel="stylesheet" href="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css">

<link href="http://maxcdn.bootstrapcdn.com/font-awesome/4.1.0/css/font-awesome.min.css" rel="stylesheet">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.0/jquery.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js"></script>

</head>

<body>

<?php

include('navwithlogin.php');

?>

<div class="container" >

<table class="table table-bordered">

<div class="table responsive">

<thead>

<tr>

<th>Serial No.</th>

<th>NGO ID</th>

<th>NGO name</th>

<th>Email</th>

<th>Phone</th>

<th>Address</th>

<th>Area</th>

<th>Requirements</th>

<th>Requirements Description</th>

<th>Be a helping hand</th>

</tr>

</thead>

<tbody>

<?php

include('connection.php');

$sql=mysqli\_query($conn,"select Serial\_no,ngo\_requirements.ngo\_id,req,req\_descr,NGO\_name,Email,Phone,address,Area from ngo\_requirements inner join registration on ngo\_requirements.ngo\_id=registration.ngo\_id");

$count=mysqli\_num\_rows($sql);

if(mysqli\_num\_rows($sql)>0)

{

while($row=mysqli\_fetch\_assoc($sql))

{

$ngid=$row["ngo\_id"];

echo "<tr>".

"<td>".$row["Serial\_no"]."</td>".

"<td>".$ngid."</td>".

"<td>".$row["NGO\_name"]."</td>".

"<td>".$row["Email"]."</td>".

"<td>".$row["Phone"]."</td>".

"<td>".$row["address"]."</td>".

"<td>".$row["Area"]."</td>".

"<td>"."<h4>"."<u>".$row["req"]."</u>"."</h4>"."</td>".

"<td>"."<h4>"."<u>".$row["req\_descr"]."</u>"."</h4>"."</td>".

'<td><button class="btn btn-primary btn-modal" name="donate" onclick="modalname()" value="21id">Donate</button></td>'.

"</tr>";

}

}

else

{

echo "no results";

}

mysqli\_close($conn);

?>

</tbody>

</div>

</table>

</div>

<!-- Modal -->

<div class="modal fade" id="myModal" tabindex="-1" role="dialog" aria-labelledby="myModalLabel">

<div class="modal-dialog" role="document">

<div class="modal-content">

<div class="modal-header">

<button type="button" class="close" data-dismiss="modal" aria-label="Close"><span aria-hidden="true">&times;</span></button>

<h4 class="modal-title" id="myModalLabel">NGO Details</h4>

</div>

<div class="modal-body">

<h2>

Contact the given NGO details to provide Non-Monetory help.

</h2>

<h3>If you want to help on a general basis scan this QR code</h3>

<img src="images/QR.jpg" class="img-responsive center-block">

</div>

<div class="modal-footer">

<button type="button" class="btn btn-default" data-dismiss="modal">Close</button>

</div>

</div>

</div>

</div>

<script>

  function modalname()

  {

    $(".btn-modal").click(function(){

$("#myModal").modal();

});

  }

</script>

<?php

include('footer.php');

?>

</body>

</html>

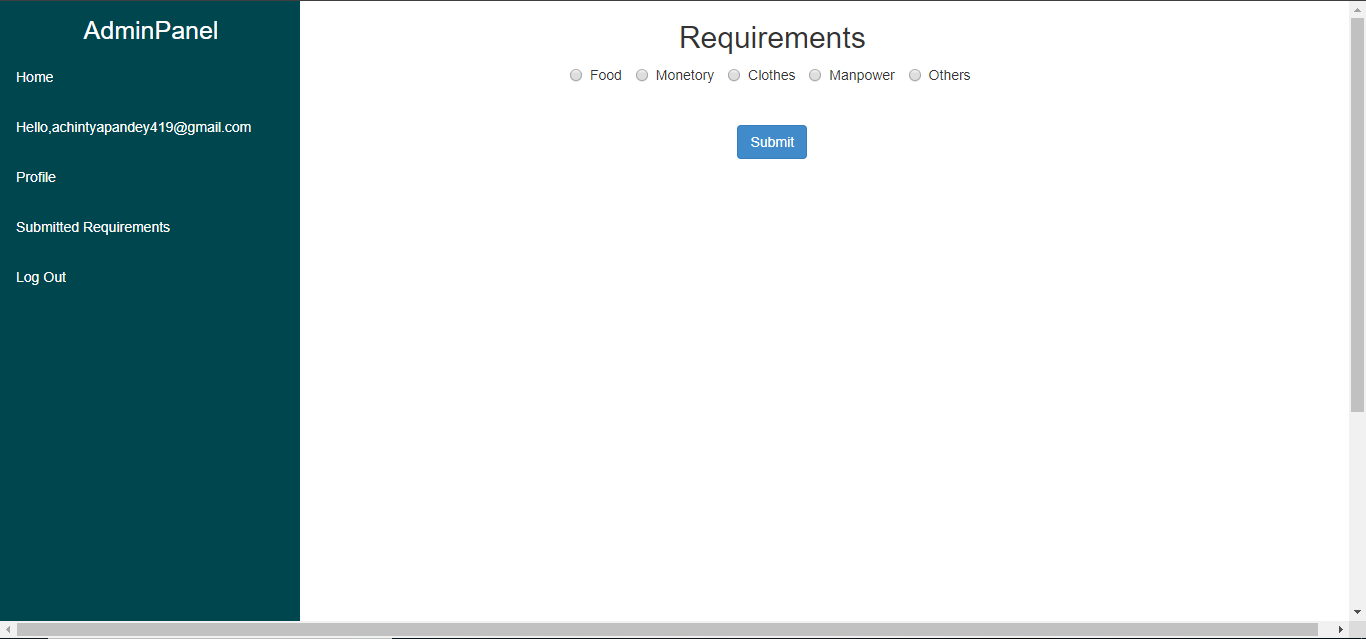


Figure 10

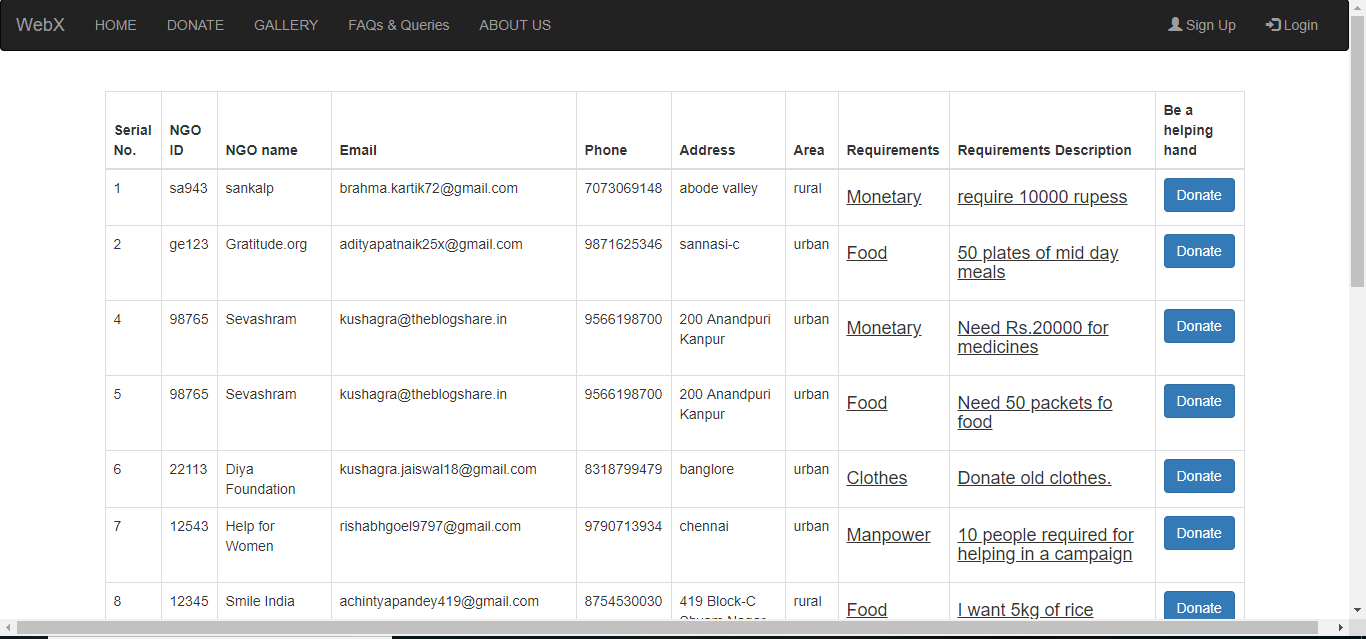


Figure 11

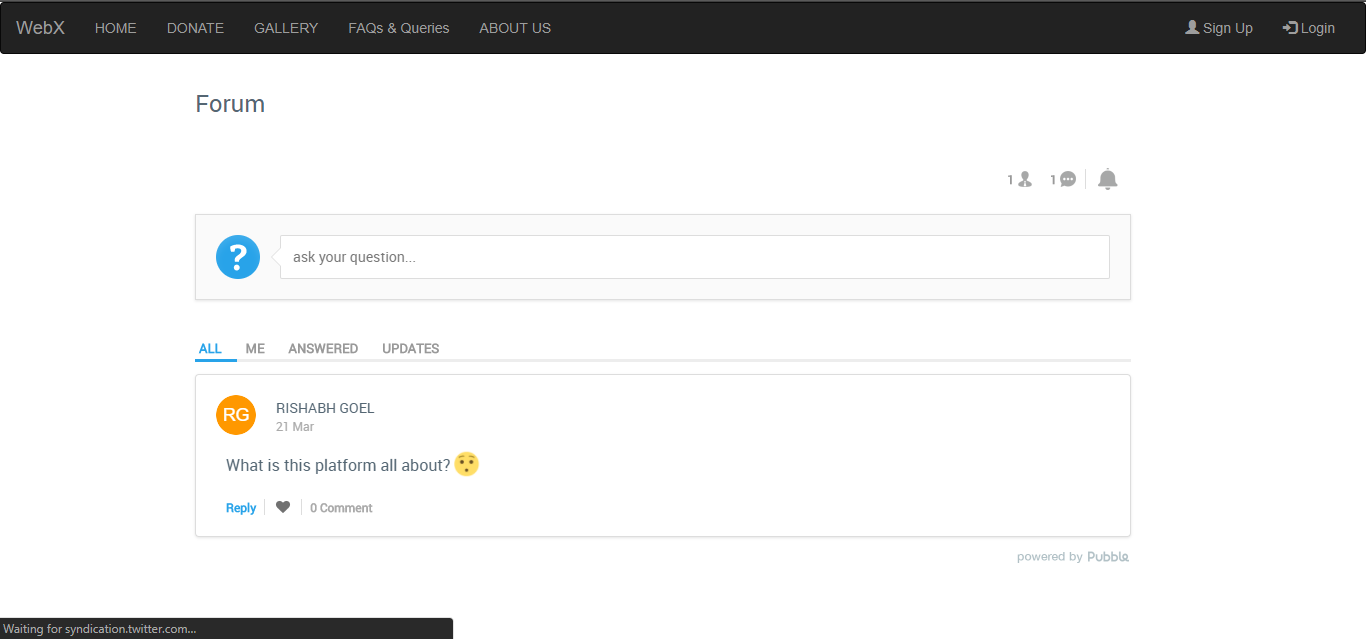


Figure 12

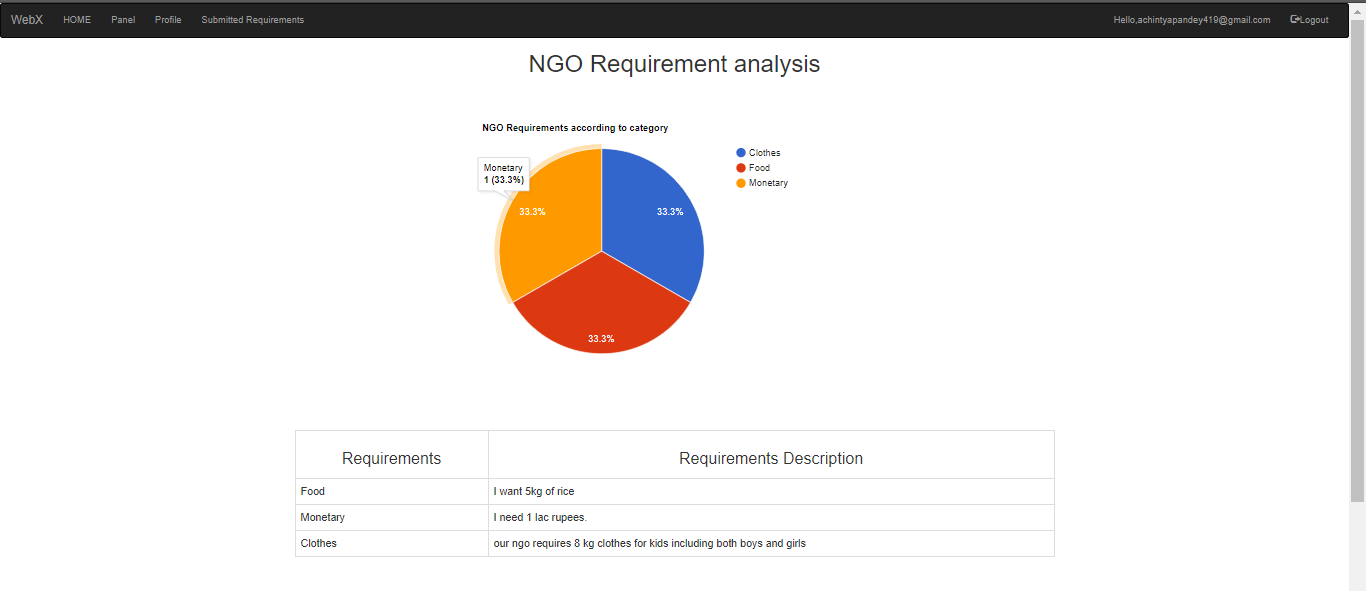


Figure 13

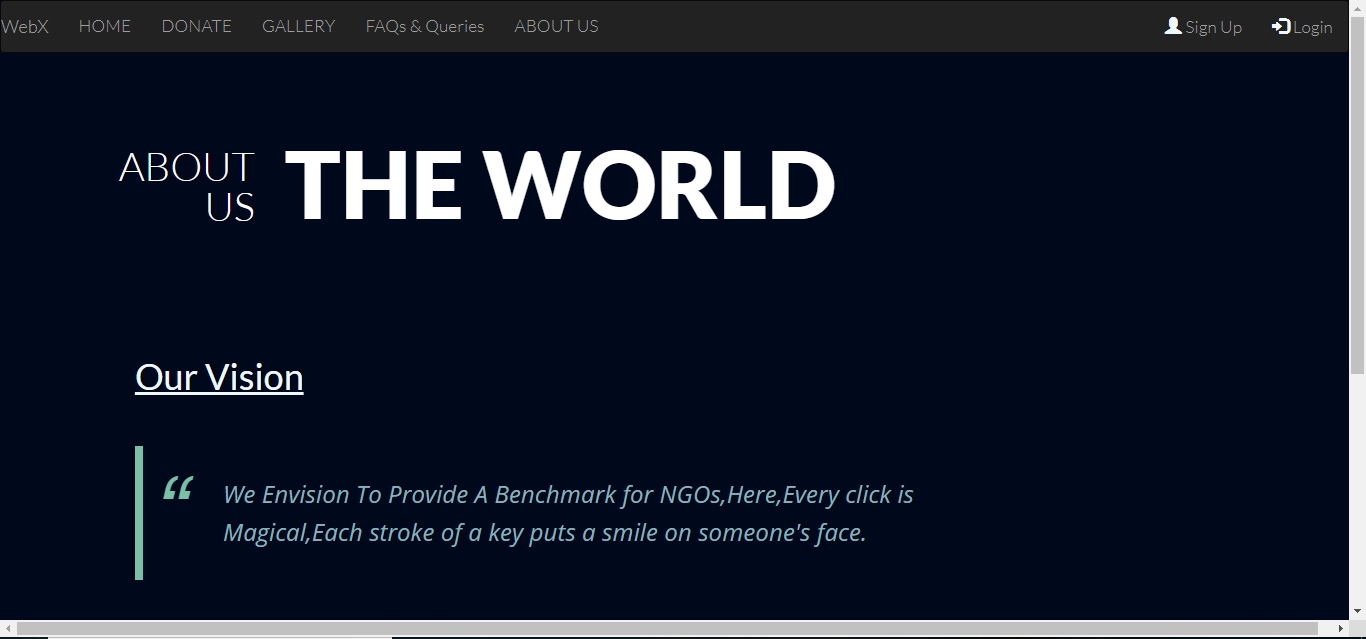


Figure 14

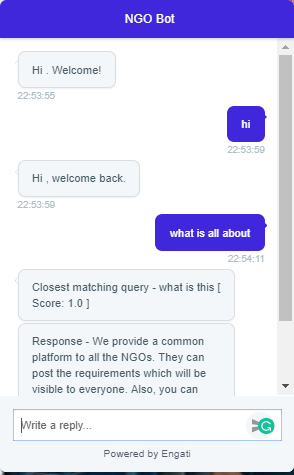


Figure 15

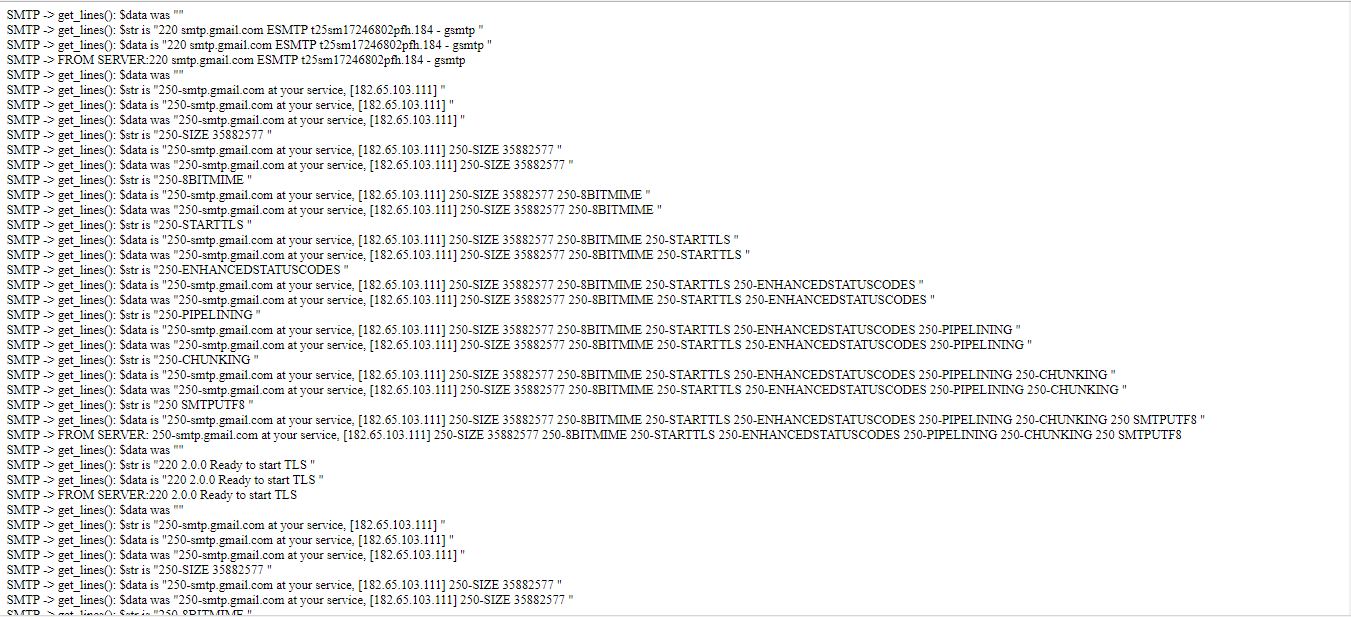


Figure 16

**REFERENCES**

1. <https://www.w3schools.com/>
2. <https://www.w3schools.com/css/css_examples.asp>
3. <https://www.w3schools.com/bootstrap/default.asp>
4. <https://github.com/PHPMailer/PHPMailer>
5. <https://developers.google.com/chart/>
6. <https://stackoverflow.com/questions/17364323/phpmailer-not-sending-and-not-giving-error>